Example on Bayesian Spectral Analysis Regression

Statlab2

2017-08-04

Preliminaries

To use `bsamGP`, you need to first install the package. Use RGui menu "Package/Install package(s) from local zip files…" or R command "install.packages".

You then load the `bsamGP` package using the `library` or `require` function:

```
library(bsamGP)
```

This needs to be done every time you start R.

To get help on the functions in R (and in `bsamGP`), use `help()` or `?`. For example, to view the help file for the `bsar` function, type one of the following:

```
help(bsar) # ?bsar
```

Bayesian Spectral Analysis Regression (BSAR)

Let's now proceed to the monotone convex function estimating via the following BSAR model.

$$Y_i = \boldsymbol{w}_i^\top \boldsymbol{\beta} + f(x_i) + \epsilon_i, \;\; \epsilon_i \sim N(0, \sigma^2),$$

where $\boldsymbol{w}_i$ and $\boldsymbol{\beta}$ are $p+1$–dimensional vectors of covariates and coefficients; f is an unknown function of the scalar x_i estimated by

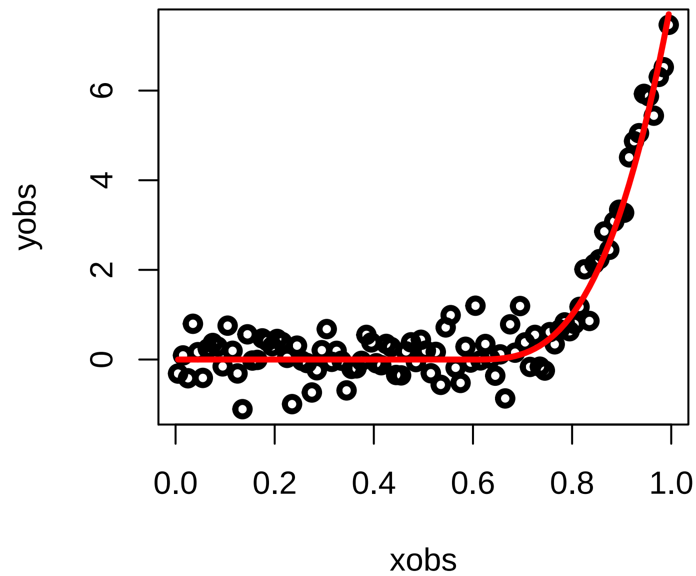
$$\begin{aligned} f(x) &= \delta \sum_{j=0}^{\infty} \sum_{k=0}^{\infty} \theta_j \theta_k \varphi_{j,k}^b(x) + \alpha(x - 0.5) \\ \varphi_{j,k}^b(x) &= \int_0^x \int_0^s \varphi_j(t)\varphi_k(t) dt\, ds - \int_0^1 \int_0^x \int_0^s \varphi_j(t)\varphi_k(t) dt\, ds\, dx. \end{aligned}$$

Data generation

We will consider estimating the following monotone convex function where $x \in [0,1]$.

$$f(x) = (5x-3)^3 I(5x-3>0)$$

```
set.seed(1)
ftn = function(x) (5 * x-3)^3 * (5 * x-3>0)
n = 100
xobs = (2 * (1:n)-1)/(2 * n)
yobs = ftn(xobs) + rnorm(n, sd=0.5)
plot(xobs,yobs,lwd=3)
lines(xobs,ftn(xobs),lwd=3,col=2)
```



Model Fitting

To fit BSAR model, we first set up the MCMC parameters, the number of basis and prior information for spectral coefficients.

```
# MCMC parameters
nblow0 = 1000; # Initialization period for adpative metropolis
nblow = 20000; # Number of MCMC in transition period
smcmc = 2000; # Number of MCMC for analysis
nskip = 10; # Number of MCMC to skip after nblow
ndisp = 1000; # number of saved draws to be displayed on screen
maxmodmet = 5; # Maximum number of times to modify metropolis
# Prior information
iflagprior = 0; # 1 = Lasso Smoother, 0 = T Smoother
nbasis = 30; # number of cosine basis functions @
```

To generate a posterior sample for the Bayesian semiparametric regression model, use the function `bsar`.

```
# Fit the model
fout = bsar(y=yobs, x=xobs, w=NULL, shape='IncreasingConvex',
           xmin=0, xmax=1,nbasis=nbasis,
           mcmc=list(maxmodmet=maxmodmet,nblow0=nblow0,nblow=nblow,smcmc=smcmc,nskip=nskip,ndisp=ndisp)
           prior=list(iflagprior=iflagprior))
```

```
## Initializing MCMC parameters ...
## Burnin ...
## function[1]: pmet = 0.4466
## Main iterations ...
## MCMC draws 1000 of 2000 (CPU time: 14.281 s)
```

```
## MCMC draws 2000 of 2000 (CPU time: 18.719 s)
## function[1]: pmet = 0.4466
```

The output returns `bsam` class object. The `print` function on `bsam` object summarizes the fit.

```
print(fout)
```

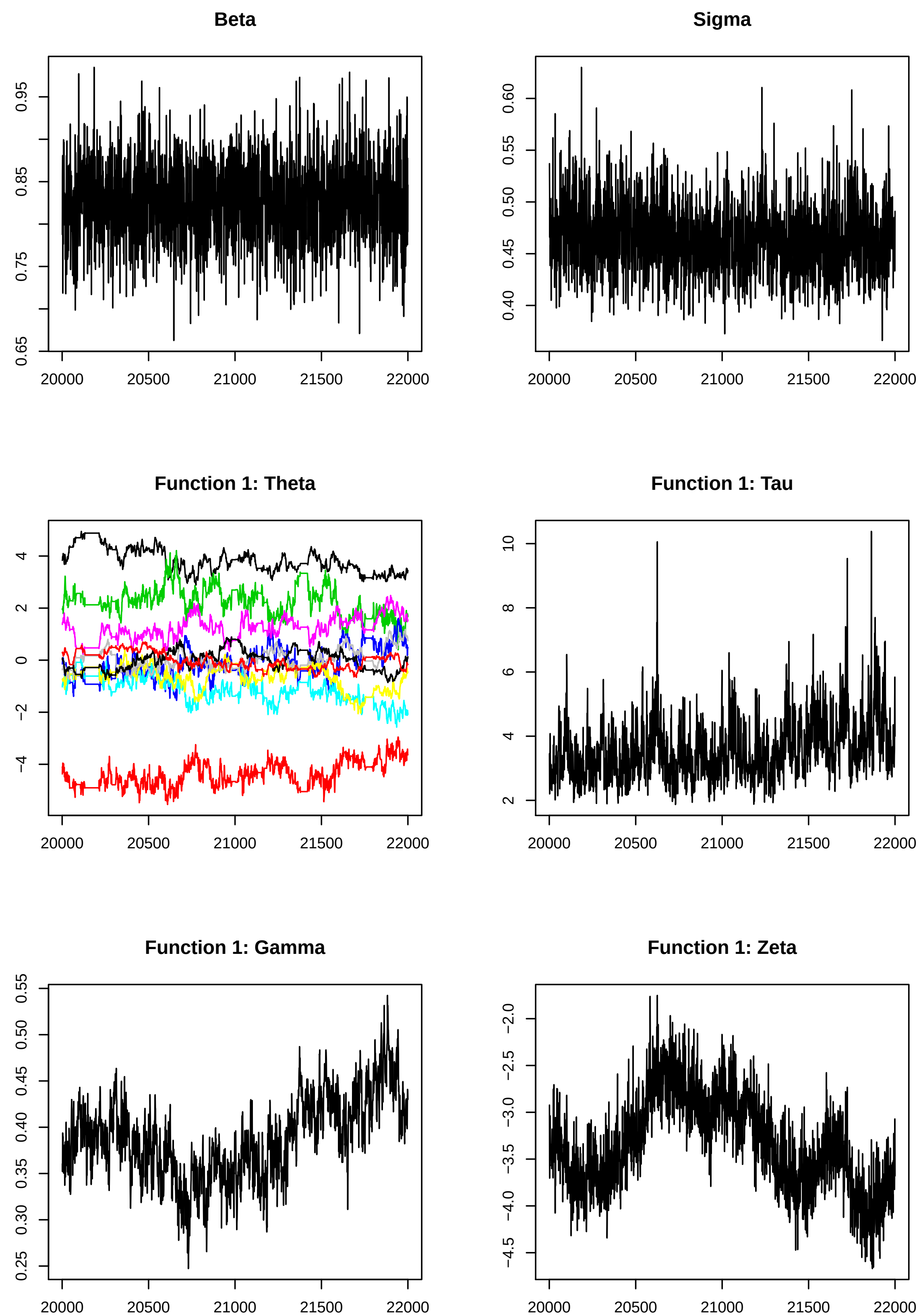
```
##
## Call:
## bsar.default(y = yobs, w = NULL, x = xobs, xmin = 0, xmax = 1,
##     nbasis = nbasis, mcmc = list(maxmodmet = maxmodmet, nblow0 = nblow0,
##         nblow = nblow, smcmc = smcmc, nskip = nskip, ndisp = ndisp),
##     prior = list(iflagprior = iflagprior), shape = "IncreasingConvex")
##
## Bayesian Spectral Analysis Regression (BSAR)
##
## Model:
## Y = w'beta + sum_k f_k(x_k) + N(0,sigma^2)
## xmin < x < xmax
## Normalize f: int_a^b f(x)dx = 0
## beta has the Intercept
##
## Define
## Z(x) = sum_j=0^J theta_j*phi_j(x)
## g^a(x) = int_0^x |Z(s)|^2 dx
##        = theta' Phi^a(x)*theta
## where Phi^a(x) is matrix with
## phi^a_{j,k}(x) = int_0^x phi_j(s) phi_k(s) ds in (j,k)
## g^b(x)   = int_0^x g^a(s) ds
##          = theta' Phi^b(x)*theta
## where Phi^b(x) is a matrix with
## phi^b_{j,k}(x) = int_0^x phi^a_{j,k}(s) ds
##
## *************************************************************************************
##
## Number of nonparametric components =  1
##
## Model for f
## Removed theta_0^2 from f but kept theta_0*theta_j
## Increasing convex f
## f(x) = g^b(x) +  alpha*(x-xmid)
## alpha > 0
## Model for f includes linear term alpha
##
## *************************************************************************************
##
## Cosine basis
## xrange   = xmax-xmin
## phi_0(x) = 1/sqrt(xmax-xmin) for xmin < x < xmax
## phi_k(x) = sqrt(2/(xmax-xmin))*cos(pi*k*(x-xmin)/(xmax-xmin))
##
## Scale invariant priors:
## beta|sigma       ~ N(b0,sigma2*B0)
## alpha|sigma      ~ N(m0,sigma2*v0)I(delta*alpha>0)
## theta_0|sigma    ~ N(0,sigma*v0)I(theta_0 > 0)
```

```
## theta_k|sigma    ~ N(0,sigma*tau^2*exp(-gamma*k)
##
## Smoothing parameters tau and gamma
## Choice of two priors for tau2
## 1.) T Smoother: tau2 ~ IG(r0/2,s0/2)
## gamma ~ Exp(w0)
##
## Note: posterior of tau and gamma have banana contours
## zeta = ln(tau2) - kbar*gamma is less dependent with gamma or log(gamma)
##
## *********************************************************************************
##
## Number of Cosine basis functions     =  30
## Number of observations               =  100
## Number of covariates (no intercept)  =  0
##
## MCMC transition draws                =  20000
## MCMC draws saved for estimation      =  2000
## Save every nskip draws               =  10
## MCMC draws total                     =  40000
##
## Function =  1
## Proportion of Theta Accepted after burn-in    =  0.44655
##
## R-Square                       =  0.9378
##
## Log Integrated Likelihood
## LIL Gelfand & Dey              =  -102.6756
## LIL Newton & Raftery (biased)  =  -67.3496
##
## beta
## const is Y intercept
##           PostM    PostStd PostM/STD
## const 0.8268324 0.04701043  17.58828
##
## sigma
## PostM sigma  =  0.4650313
## PostS sigma  =  0.03396708
##
## *********************************************************************************
##
## Function = 1
## Increasing convex function
##
## Linear term alpha in x for constrainted f
## Posterior mean   of alpha =  0.1208287
## Posterior stddev of alpha =  0.1067718
##
## theta_k ~ N(0,sigma*tau2*exp(-gamma*k))
##
## Tau
## PostM   PostS
## 3.577263 0.938478
##
```

```
## Gamma
##      PostM      PostS
## 0.38849995 0.04447833
##
## Zeta = ln(tau2) - mean(k)*gamma
##      PostM      PostS
## -3.3407003  0.4893772
##
## Cosine Basis weights theta
##     PostMean PostSTD    Ratio
## T0   3.82455 0.47397  8.06921
## T1  -4.39168 0.47680 -9.21071
## T2   2.25348 0.57582  3.91355
## T3  -0.12302 0.59218 -0.20775
## T4  -1.14814 0.48609 -2.36198
## T5   1.23613 0.43352  2.85140
## T6  -0.66332 0.45148 -1.46921
## T7   0.05528 0.35379  0.15626
## T8   0.00000 0.37483  0.00000
## T9  -0.01742 0.30766 -0.05661
## T10  0.11611 0.24293  0.47795
## T11  0.00205 0.31170  0.00659
## T12  0.01143 0.15192  0.07525
## T13  0.03309 0.13812  0.23959
## T14  0.01322 0.13788  0.09585
## T15 -0.02363 0.08463 -0.27921
## T16 -0.01188 0.12984 -0.09151
## T17  0.01051 0.08044  0.13069
## T18  0.02333 0.06563  0.35555
## T19 -0.01476 0.05437 -0.27144
## T20 -0.00503 0.04944 -0.10176
## T21 -0.01513 0.03510 -0.43098
## T22 -0.00215 0.02419 -0.08867
## T23  0.00520 0.02629  0.19773
## T24 -0.01641 0.02699 -0.60823
## T25 -0.00503 0.01321 -0.38052
## T26  0.02211 0.01738  1.27239
## T27  0.00411 0.01190  0.34553
## T28  0.00790 0.00953  0.82925
## T29  0.00553 0.00968  0.57154
## T30  0.00242 0.00646  0.37408
## ***************************************************************************************
```

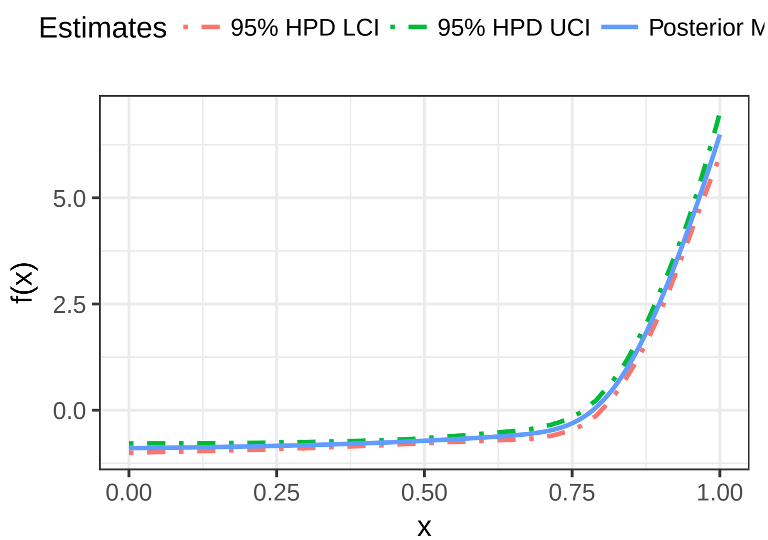
The `plot` function on `bsam` object returns traceplots of each parameters for diagnostics.

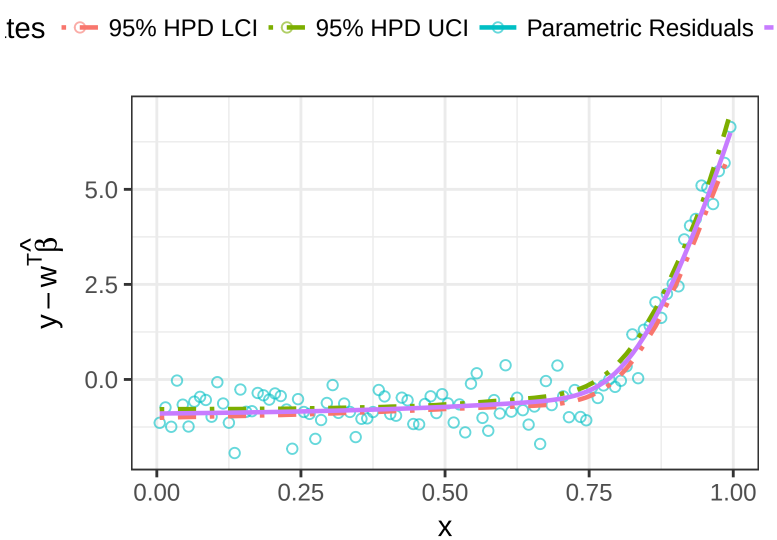
```
plot(fout)
```


Beta
0.65
0.75
0.85
0.95
20000
20500
21000
21500
22000
Sigma
0.40
0.45
0.50
0.55
0.60
20000
20500
21000
21500
22000
Function 1: Theta
−4
−2
0
2
4
20000
20500
21000
21500
22000
Function 1: Tau
2
4
6
8
10
20000
20500
21000
21500
22000
Function 1: Gamma
0.25
0.30
0.35
0.40
0.45
0.50
0.55
20000
20500
21000
21500
22000
Function 1: Zeta
−4.5
−4.0
−3.5
−3.0
−2.5
−2.0
20000
20500
21000
21500
22000

We may visualize fitted mean curve and 95% highest posterior density (HPD) interval with `plot` function on fitted object from `fitted` method.

```r
fit = fitted(fout)
plot(fit)
```





For more detailed examples and real data applications, see Jo, S., Choi, T., Park, B., & Lenk, P. (2017) "bsamGP: An R package for Bayesian Spectral Analysis Models using Gaussian Process Priors", *Preprint* .