Package 'BASS'

March 17, 2017

Version 0.2.2

Date 2017-3-15

Title Bayesian Adaptive Spline Surfaces

Description Bayesian fitting and sensitivity analysis methods for adaptive spline surfaces. Built to handle continuous and categorical inputs as well as functional or scalar output. An extension of the methodology in Denison, Mallick and Smith (1998) <doi:10.1023/A:1008824606259>.

License GPL-3

Suggests knitr

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation yes

Author Devin Francom [aut, cre],
Bruno Sanso [ths],
Agner Fog [ctb, cph]

Maintainer Devin Francom <devinfrancom@gmail.com>

Repository CRAN

Date/Publication 2017-03-17 07:24:00 UTC

R topics documented:

bass 2
plot.bass 5
plot.bassSob 6
predict.bass 7
print.bass 8
sobol 8
summary.bass 9

Index **11**

bass *Bayesian Adaptive Spline Surfaces (BASS)*

Description

Fits a BASS model using RJMCMC. Optionally uses parallel tempering to improve mixing. Can be used with scalar or functional response. Also can use categorical inputs.

Usage

```
bass(xx, y, maxInt = 3, maxInt.func = 3, maxInt.cat = 3, xx.func = NULL,
  degree = 1, maxBasis = 1000, npart = NULL, npart.func = NULL,
  nmcmc = 10000, nburn = 9000, thin = 1, g1 = 0, g2 = 0, h1 = 10,
  h2 = 10, a.tau = 1, b.tau = NULL, w1 = 5, w2 = 5,
  temp.ladder = NULL, start.temper = NULL, curr.list = NULL,
  save.yhat = TRUE, small = FALSE, verbose = TRUE)
```

Arguments

xx	a data frame or matrix of predictors. Categorical predictors should be included as factors.
y	a response vector (scalar response) or matrix (functional response).
maxInt	integer for maximum degree of interaction in spline basis functions. Defaults to the number of predictors, which could result in overfitting.
maxInt.func	(functional response only) integer for maximum degree of interaction in spline basis functions describing the functional response.
maxInt.cat	(categorical input only) integer for maximum degree of interaction of categorical inputs.
xx.func	a vector, matrix or data frame of functional variables.
degree	degree of splines. Stability should be examined for anything other than 1.
maxBasis	maximum number of basis functions. This should probably only be altered if you run out of memory.
npart	minimum number of non-zero points in a basis function. If the response is functional, this refers only to the portion of the basis function coming from the non-functional predictors. Defaults to 20 or 0.1 times the number of observations, whichever is smaller.
npart.func	same as npart, but for functional portion of basis function.
nmcmc	number of RJMCMC iterations.
nburn	number of the nmcmc iterations to disregard.
thin	keep every thin samples
g1	shape for IG prior on σ^2.
g2	scale for IG prior on σ^2.

h1	shape for gamma prior on λ.
h2	rate for gamma prior on λ. This is the primary way to control overfitting. A large value of h2 favors fewer basis functions.
a.tau	shape for gamma prior on τ.
b.tau	rate for gamma prior on τ. Defaults to one over the number of observations, which centers the prior for the basis function weights on the unit information prior.
w1	nominal weight for degree of interaction, used in generating candidate basis functions. Should be greater than 0.
w2	nominal weight for variables, used in generating candidate basis functions. Should be greater than 0.
temp.ladder	temperature ladder used for parallel tempering. The first value should be 1 and the values should increase.
start.temper	when to start tempering (after how many MCMC iterations). Defaults to 1000 or half of burn-in, whichever is smaller.
curr.list	list of starting models (one element for each temperature), could be output from a previous run under the same model setup.
save.yhat	logical; should predictions of training data be saved?
small	logical; if true, returns a smaller object by leaving out curr.list and other unnecessary objects. Use in combination with save.yhat to get smaller memory footprint for very large models.
verbose	logical; should progress be displayed?

Details

Explores BASS model space by RJMCMC. The BASS model has

$$y = f(x) + \epsilon, \;\; \epsilon \sim N(0, \sigma^2)$$

$$f(x) = a_0 + \sum_{m=1}^{M} a_m B_m(x)$$

and $B_m(x)$ is a BASS basis function (tensor product of spline basis functions). We use priors

$$a \sim N(0, \sigma^2/\tau(B'B)^{-1})$$

$$M \sim Poisson(\lambda)$$

as well as the priors mentioned in the arguments above.

Value

An object of class 'bass'. The other output will only be useful to the advanced user. Rather, users may be interested in prediction and sensitivity analysis, which are obtained by passing the entire object to the predict.bass or sobol functions.

See Also

predict.bass for prediction and sobol for sensitivity analysis.

Examples

```
## Not run:
#############################################################################################################
### univariate example
#############################################################################################################
## simulate data (Friedman function)
f<-function(x){
  10*sin(pi*x[,1]*x[,2])+20*(x[,3]-.5)^2+10*x[,4]+5*x[,5]
}
sigma<-1 # noise sd
n<-500 # number of observations
x<-matrix(runif(n*10),n,10) #10 variables, only first 5 matter
y<-rnorm(n,f(x),sigma)

## fit BASS, no tempering
mod<-bass(x,y)
plot(mod)
## fit BASS, tempering
mod<-bass(x,y,temp.ladder=1.3^(0:8),start.temper=1000)
plot(mod)

## prediction
npred<-1000
xpred<-matrix(runif(npred*10),npred,10)
pred<-predict(mod,xpred,verbose=TRUE) # posterior predictive samples
true.y<-f(xpred)
plot(true.y,colMeans(pred),xlab='true values',ylab='posterior predictive means')
abline(a=0,b=1,col=2)

## sensitivity
sens<-sobol(mod)
plot(sens,cex.axis=.5)

#############################################################################################################
### functional example
#############################################################################################################
## simulate data (Friedman function with first variable as functional)
sigma<-1 # noise sd
n<-500 # number of observations
nfunc<-50 # size of functional variable grid
xfunc<-seq(0,1,length.out=nfunc) # functional grid
x<-matrix(runif(n*9),n,9) # 9 non-functional variables, only first 4 matter
X<-cbind(rep(xfunc,each=n),kronecker(rep(1,nfunc),x)) # to get y
y<-matrix(f(X),nrow=n)+rnorm(n*nfunc,0,sigma)

## fit BASS
mod<-bass(x,y,xx.func=xfunc)
plot(mod)
```

```
## prediction
npred<-100
xpred<-matrix(runif(npred*9),npred,9)
Xpred<-cbind(rep(xfunc,each=npred),kronecker(rep(1,nfunc),xpred))
ypred<-matrix(f(Xpred),nrow=npred)
pred<-predict(mod,xpred) # posterior predictive samples (each is a curve)
matplot(ypred,apply(pred,2:3,mean),type='l',xlab='observed',ylab='mean prediction')
abline(a=0,b=1,col=2)
matplot(t(ypred),type='l') # actual
matplot(t(apply(pred,2:3,mean)),type='l') # mean prediction

## sensitivity
sens<-sobol(mod,mcmc.use=1:10) # for speed, only use a few samples
plot(sens) # functional variable labelled "a"

sens.func<-sobol(mod,mcmc.use=1:10,func.var=1)
plot(sens.func)

## End(Not run)

## minimal example for CRAN testing
mod<-bass(1:2,1:2,nmcmc=2,nburn=1)
```

`plot.bass` *BASS Plot Diagnostics*

Description

Generate diagnostic plots for BASS model fit.

Usage

```
## S3 method for class 'bass'
plot(x, quants = c(0.025, 0.975), ...)
```

Arguments

`x`	a `bass` object.
`quants`	quantiles for intervals, if desired. NULL if not desired.
`...`	graphical parameters.

Details

The first two plots are trace plots for diagnosing convergence. The third plot is posterior predicted vs observed, with intervals for predictions. The fourth plot is a histogram of the residuals (of the posterior mean model), with a red curve showing the assumed Normal density (using posterior mean variance). If `bass` was run with `save.yhat = FALSE`, the third and fourth plots are omitted.

See Also

bass, predict.bass, sobol

Examples

```
# See examples in bass documentation.
```

`plot.bassSob` *Plot BASS sensitivity indices*

Description

Generate plots for sensitivity analysis of BASS.

Usage

```
## S3 method for class 'bassSob'
plot(x, ...)
```

Arguments

`x`	a `bassSob` object, returned from `sobol`.
`...`	graphical parameters.

Details

If `func.var` in the call to `sobol` was `NULL`, this returns boxplots of sensitivity indices and total sensitivity indices. If there were functional variables, they are labeled with letters alphabetically. Thus, if I fit a model with 4 categorical/continuous inputs and 2 functional inputs, the functional inputs are labeled a and b. If `func.var` was not `NULL`, then posterior mean functional sensitivity indices are plotted, along with the functional partitioned variance. Variables and interactions that are excluded did not explain any variance.

See Also

bass, predict.bass, sobol

Examples

```
# See examples in bass documentation.
```

predict.bass *BASS Prediction*

Description

Predict function for BASS. Outputs the posterior predictive samples based on the specified MCMC iterations.

Usage

```
## S3 method for class 'bass'
predict(object, newdata, newdata.func = NULL,
  mcmc.use = NULL, verbose = FALSE, ...)
```

Arguments

object	a fitted model, output from the bass function.
newdata	a matrix of new input values at which to predict. The columns should correspond to the same variables used in the bass function.
newdata.func	a matrix of new values of the functional variable. If none, the same values will be used as in the training data.
mcmc.use	a vector indexing which MCMC iterations to use for prediction.
verbose	logical; should progress be displayed?
...	further arguments passed to or from other methods.

Details

Efficiently predicts when two MCMC iterations have the same basis functions (but different weights).

Value

If model output is a scalar, this returns a matrix with the same number of rows as newdata and columns corresponding to the the MCMC iterations mcmc.use. These are samples from the posterior predictive distribution. If model output is functional, this returns an array with first dimension corresponding to MCMC iteration, second dimension corresponding to the rows of newdata, and third dimension corresponding to the rows of newdata.func.

See Also

bass for model fitting and sobol for sensitivity analysis.

Examples

```
# See examples in bass documentation.
```

print.bass *Print BASS Details*

Description

Print some of the details of a BASS model.

Usage

```
## S3 method for class 'bass'
print(x, ...)
```

Arguments

x	a bass object, returned from bass.
...	further arguments passed to or from other methods.

sobol *BASS Sensitivity Analysis*

Description

Decomposes the variance of the BASS model into variance due to main effects, two way interactions, and so on, similar to the ANOVA decomposition for linear models. Uses the Sobol' decomposition, which can be done analytically for MARS models.

Usage

```
sobol(bassMod, mcmc.use = NULL, func.var = NULL, xx.func.var = NULL,
  verbose = TRUE)
```

Arguments

bassMod	a fitted model output from the bass function.
mcmc.use	an integer vector indexing which MCMC iterations to use for sensitivity analysis.
func.var	an integer indicating which functional variable to make sensitivity indices a function of. Disregard if bassMod is non-functional or if scalar sensitivity indices are desired.
xx.func.var	grid for functional variable specified by func.var. Disregard if func.var is not specified. If func.var is specified and xx.func.var not specified, the grid used to fit bass will be used.
verbose	logical; should progress be displayed?

Details

Performs analytical Sobol' decomposition for each MCMC iteration in mcmc.use (each corresponds to a MARS model), yeilding a posterior distribution of sensitivity indices. Can obtain Sobol' indices as a function of one functional variable.

Value

If non-functional (`func.var = NULL`), a list with two elements:

`S`	a data frame of sensitivity indices with number of rows matching the length of `mcmc.use`. The columns are named with a particular main effect or interaction. The values are the proportion of variance in the model that is due to each main effect or interaction.
`T`	a data frame of total sensitivity indices with number of rows matching the length of `mcmc.use`. The columns are named with a particular variable.

Otherwise, a list with four elements:

`S`	an array with first dimension corresponding to MCMC samples (same length as `mcmc.use`), second dimension corresponding to different main effects and interactions (labeled in `names.ind`), and third dimension corresponding to the grid used for the functional variable. The elements of the array are sensitivity indices.
`S.var`	same as `S`, but scaled in terms of total variance rather than percent of variance.
`names.ind`	a vector of names of the main effects and interactions used.
`xx`	the grid used for the functional variable.

See Also

bass for model fitting and predict.bass for prediction.

Examples

```
# See examples in bass documentation.
```

`summary.bass` *Summarize BASS Details*

Description

Summarize some of the details of a BASS model.

Usage

```
## S3 method for class 'bass'
summary(object, ...)
```

Arguments

`object`	a `bass` object, returned from `bass`.
`...`	further arguments passed to or from other methods.

Index

∗Topic **Sobol**
    sobol, 8
∗Topic **analysis**
    bass, 2
∗Topic **data**
    bass, 2
∗Topic **decomposition**
    sobol, 8
∗Topic **functional**
    bass, 2
∗Topic **nonparametric**
    bass, 2
∗Topic **regression,**
    bass, 2
∗Topic **splines,**
    bass, 2

bass, 2, *6, 7*, *9*

plot.bass, 5
plot.bassSob, 6
predict.bass, *4*, *6*, 7, *9*
print.bass, 8

sobol, *4*, *6, 7*, 8
summary.bass, 9