Package 'BayesPPR'

May 18, 2026

Title Bayesian Projection Pursuit Regression

Version 0.1.0

Description Bayesian fitting of projection pursuit regression model.
Built to handle continuous and categorical inputs and
scalar output (Collins et al., 2023 <DOI:10.1007/s11222-023-10334-z>).

License MIT + file LICENSE

Encoding UTF-8

Imports methods

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

URL https://github.com/gqcollins/BayesPPR

BugReports https://github.com/gqcollins/BayesPPR/issues

NeedsCompilation no

Author Gavin Collins [aut, ctb],
J. Derek Tucker [ctb, cre] (ORCID:
<https://orcid.org/0000-0001-8844-2169>)

Maintainer J. Derek Tucker <jdtuck@sandia.gov>

Repository CRAN

Date/Publication 2026-05-18 18:10:21 UTC

Contents

---

bppr *Bayesian Projection Pursuit Regression*

Description

Fits a BayesPPR model using RJMCMC. Can handle categorical features.

Usage

```
bppr(
  X,
  y,
  n_ridge_mean = 10,
  n_ridge_max = NULL,
  n_act_max = NULL,
  df_spline = 4,
  prob_relu = 2/3,
  prior_coefs = "zs",
  shape_var_coefs = NULL,
  scale_var_coefs = NULL,
  n_dat_min = NULL,
  scale_proj_dir_prop = NULL,
  adapt_act_feat = TRUE,
  w_n_act = NULL,
  w_feat = NULL,
  n_post = 1000,
  n_burn = 9000,
  n_adapt = 0,
  n_thin = 1,
  print_every = 1000,
  bppr_init = NULL
)
```

Arguments

X	a data frame or matrix of predictors. Categorical features should be coded as numeric.
y	a numeric response vector.
n_ridge_mean	mean for Poisson prior on the number of ridge functions.
n_ridge_max	maximum number of ridge functions allowed in the model. Used to avoid memory overload. Defaults to 150 unless the number of observed responses is small.
n_act_max	maximum number of active variables in any given ridge function. Defaults to 3 unless categorical features are detected, in which case the default is larger.
df_spline	degrees of freedom for spline basis. Stability should be examined for anything other than 4.

prob_relu — prior probability that any given ridge function uses a relu transformation.

prior_coefs — form of the prior distribution for the basis coefficients. Default is "zs" for the Zellner-Siow prior. The other option is "flat", which is an improper prior.

shape_var_coefs — shape for IG prior on the variance of the basis function coefficients. Default is for the Zellner-Siow prior. For the flat, improper prior, shape_var_coefs is ignored.

scale_var_coefs — scale for IG prior on the variance of the basis function coefficients. Default is for the Zellner-Siow prior. For the flat, improper prior, scale_var_coefs is ignored.

n_dat_min — minimum number of observed non-zero datapoints in a ridge function. Defaults to 20 or 0.1 times the number of observations, whichever is smaller.

scale_proj_dir_prop — scale parameter for generating proposed projection directions. Should be in (0, 1); default is about 0.002.

adapt_act_feat — logical; if TRUE, use adaptive proposal for feature index sets and number of active features.

w_n_act — vector of weights for number of active variables in a ridge function, used in generating proposed basis functions. If adapt_act_feat == FALSE, it is also used for the prior distribution. Default is rep(1, n_act_max).

w_feat — vector of weights for feature indices used in a ridge function, used in generating proposed basis functions. If adapt_act_feat == FALSE, it is also used for the prior distribution. Default is rep(1, ncol(X)).

n_post — number of posterior draws to obtain from the Markov chain after burn-in.

n_burn — number of draws to burn before obtaining n_post draws for inference. If prior_coefs == "flat" then these are absorbed into the adapt phase.

n_adapt — number of adaptive MCMC iterations to perform before burn-in. Skips sampling basis coefficients and residual variance to save time.

n_thin — keep every n_thin posterior draws after burn-in.

print_every — print the iteration number every print_every iterations. Use print_every = 0 to silence.

bppr_init — list of initial values for the Markov chain. Used by bppr_resume.

Details

Explores BayesPPR model space using RJMCMC. The BayesPPR model has

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

$$f(x) = \beta_0 + \sum_{j=1}^{M} \beta_j B_j(x)$$

and $B_j(x)$ is a natural spline basis expansion. We use priors

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

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

as well as the hyper-prior on the variance τ of the coefficients β mentioned in the arguments above.

Value

An object of class "bppr". Predictions can be obtained by passing the entire object to the predict.bppr function.

See Also

predict.bppr for prediction.

Examples

```
################################
### 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]
}
n <- 500 # number of observations
p <- 10 #10 variables, only first 5 matter
X <- matrix(runif(n*p), n, p)
y <- f(X) + rnorm(n)

## fit BPPR
fit <- bppr(X, y)

## plot diagnostics
plot(fit)

## get out-of-sample predictions
X_test <- matrix(runif(n*p), n, p)
preds <- predict(fit, X_test) # posterior predictive samples


## minimal example for CRAN testing
fit <- bppr(matrix(1:50), 1:50, n_post = 2, n_burn = 0, n_adapt = 0)
```

---

bppr_pca *BayesPPR for Multivariate Response*

Description

Fits a BayesPPR model to multivariate response using RJMCMC. Can handle categorical features.

Usage

```
bppr_pca(
  X,
  Y,
```

```
  n_pc = NULL,
  prop_var = 0.99,
  n_cores = 1,
  par_type = "fork",
  n_ridge_mean = 10,
  n_ridge_max = NULL,
  n_act_max = NULL,
  df_spline = 4,
  prob_relu = 2/3,
  prior_coefs = "zs",
  shape_var_coefs = NULL,
  scale_var_coefs = NULL,
  n_dat_min = NULL,
  scale_proj_dir_prop = NULL,
  adapt_act_feat = TRUE,
  w_n_act = NULL,
  w_feat = NULL,
  n_post = 1000,
  n_burn = 9000,
  n_adapt = 0,
  n_thin = 1,
  print_every = NULL,
  bppr_init_list = NULL
)
```

Arguments

X	a data frame or matrix of predictors. Categorical features should be coded as numeric.
Y	a data frame or matrix of responses, with columns representing different dimensions of the response.
n_pc	number of principle components to be used in pca approximation of Y.
prop_var	proportion of variance to be explained by the first n_pc principle components. Used for automatic selection of n_pc if `is.null(n_pc)`.
n_cores	number of cores the user desired to utilize for parallel computation.
par_type	a character variable dicatating the type of parallel computation to be performed. Supported values include `"fork"`, which uses `parallel::mclapply()`, and `"socket"`, which uses `parallel::parLapply()`.
n_ridge_mean	mean for Poisson prior on the number of ridge functions.
n_ridge_max	maximum number of ridge functions allowed in the model. Used to avoid memory overload. Defaults to 150 unless the number of observed responses is small.
n_act_max	maximum number of active variables in any given ridge function. Defaults to 3 unless categorical features are detected, in which case the default is larger.
df_spline	degrees of freedom for spline basis. Stability should be examined for anything other than 4.
prob_relu	prior probability that any given ridge function uses a relu transformation.

`prior_coefs` form of the prior distribution for the basis coefficients. Default is `"zs"` for the Zellner-Siow prior. The other option is `"flat"`, which is an improper prior.

`shape_var_coefs` shape for IG prior on the variance of the basis function coefficients. Default is for the Zellner-Siow prior. For the flat, improper prior, `shape_var_coefs` is ignored.

`scale_var_coefs` scale for IG prior on the variance of the basis function coefficients. Default is for the Zellner-Siow prior. For the flat, improper prior, `scale_var_coefs` is ignored.

`n_dat_min` minimum number of observed non-zero datapoints in a ridge function. Defaults to 20 or 0.1 times the number of observations, whichever is smaller.

`scale_proj_dir_prop` scale parameter for generating proposed projection directions. Should be in (0, 1); default is about 0.002.

`adapt_act_feat` logical; if `TRUE`, use adaptive proposal for feature index sets and number of active features.

`w_n_act` vector of weights for number of active variables in a ridge function, used in generating proposed basis functions. If `adapt_act_feat == FALSE`, it is also used for the prior distribution. Default is `rep(1, n_act_max)`.

`w_feat` vector of weights for feature indices used in a ridge function, used in generating proposed basis functions. If `adapt_act_feat == FALSE`, it is also used for the prior distribution. Default is `rep(1, ncol(X))`.

`n_post` number of posterior draws to obtain from the Markov chain after burn-in.

`n_burn` number of draws to burn before obtaining `n_post` draws for inference. If `prior_coefs == "flat"` then these are absorbed into the adapt phase.

`n_adapt` number of adaptive MCMC iterations to perform before burn-in. Skips sampling basis coefficients and residual variance to save time.

`n_thin` keep every n_thin posterior draws after burn-in.

`print_every` print the iteration number every print_every iterations. Use `print_every = 0` to silence.

`bppr_init_list` list of length `n_pc`, each element containing a list initial values for the Markov chain. Used by bppr_resume.

Details

Explores BayesPPR model space using RJMCMC. The BayesPPR model has

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

$$f(x) = \beta_0 + \sum_{j=1}^{M} \beta_j B_j(x)$$

and $B_j(x)$ is a natural spline basis expansion. We use priors

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

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

as well as the hyper-prior on the variance τ of the coefficients β mentioned in the arguments above.

Value

An object of class "bppr_pca". Predictions can be obtained by passing the entire object to the predict.bppr_pca function.

See Also

predict.bppr_pca for prediction.

Examples

```
# See examples in bppr documentation.
```

---

bppr_resume *Resume Sampling of BPPR Parameters*

Description

Resumes fitting a BayesPPR model where the last Markov chain left off, using the same model parameters.

Usage

```
bppr_resume(
  object,
  append = FALSE,
  n_post = 1000,
  n_burn = 9000,
  n_adapt = 0,
  n_thin = 1,
  print_every = 1000
)
```

Arguments

object	an object of class "bppr" containing the values of the current Markov chain, the latest of which constitute initial values for the new chain.
append	logical; if TRUE, new samples will be appended to samples in object. Otherwise, output will only include new samples.
n_post	number of posterior draws to obtain from the Markov chain after burn-in.
n_burn	number of draws to burn before obtaining n_post draws for inference. If prior_coefs == "flat" then these are absorbed into the adapt phase.
n_adapt	number of adaptive MCMC iterations to perform before burn-in. Skips sampling basis coefficients and residual variance to save time.
n_thin	keep every n_thin posterior draws after burn-in.
print_every	print the iteration number every print_every iterations. Use print_every = 0 to silence.

Details

Continues exploring BayesPPR model space using RJMCMC, beginning at the values contained in the last iteration of `object`.

Value

An object of class `"bppr"`. Predictions can be obtained by passing the entire object to the `predict.bppr` function.

See Also

predict.bppr for prediction.

`plot.bppr` *BPPR Plot Diagnostics*

Description

Generate diagnostic plots for BPPR model fit.

Usage

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

Arguments

`x`	a `bppr` object.
`quants`	quantiles for intervals, if desired. NULL if not desired.
`pred`	logical, whether posterior predictions should be plotted (defaults to TRUE).
`...`	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 `pred=FALSE` the third and fourth plots are omitted.

Value

no return value

See Also

bppr, predict.bppr

Examples

```
# See examples in bppr documentation.
```

`predict.bppr` *BayesPPR Predictions*

Description

Predict function for BayesPPR. Outputs posterior predictive samples for the desired MCMC iterations.

Usage

```
## S3 method for class 'bppr'
predict(object, newdata, idx_use = NULL, ...)
```

Arguments

`object`	a fitted model, output from the `bppr` function.
`newdata`	a matrix of new input values at which to predict. The columns should correspond to the same variables used in the `bppr` function.
`idx_use`	index of Markov samples to use when generating predictions.
`...`	further arguments passed to or from other methods.

Details

Bare-bones methods. Could be improved for efficiency.

Value

A matrix with the same number of rows as `newdata` and columns corresponding to all MCMC iterations indexed by idx_use. These are samples from the posterior predictive distribution.

See Also

bppr for model fitting.

Examples

```
# See examples in bppr documentation.
```

predict.bppr_pca *BayesPPR Predictions for Multivariate Response*

Description

Predict function for BayesPPR. Outputs the posterior predictive samples for the desired MCMC iterations.

Usage

```
## S3 method for class 'bppr_pca'
predict(object, newdata, idx_use = NULL, n_cores = 1, par_type = "fork", ...)
```

Arguments

object	a fitted model, output from the bppr function.
newdata	a matrix of new input values at which to predict. The columns should correspond to the same variables used in the bppr function.
idx_use	index of Markov samples to use when generating predictions.
n_cores	number of cores the user desired to utilize for parallel computation.
par_type	a character variable dicatating the type of parallel computation to be performed. Supported values include "fork", which uses parallel::mclapply(), and "socket", which uses parallel::parLapply().
...	further arguments passed to or from other methods.

Details

Bare-bones methods. Could be improved for efficiency.

Value

An matrix where the first dimension corresponds to MCMC iterations indexed by idx_use, the second dimensions is the same number of rows as newdata, and the third corresponds to the dimension of the response.

See Also

bppr for model fitting.

Examples

```
# See examples in bppr documentation.
```

print.bppr *Print BPPR Details*

Description

Print some of the details of a BPPR model.

Usage

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

Arguments

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

Value

no return value

summary.bppr *Summarize BPPR Details*

Description

Summarize some of the details of a BPPR model.

Usage

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

Arguments

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

Value

no return value

Index