Package ‘NVCSSL’

September 18, 2023

Type Package

Title Nonparametric Varying Coefficient Spike-and-Slab Lasso

Version 2.0

Date 2023-09-17

Author Ray Bai

Maintainer Ray Bai <raybaistat@gmail.com>


License GPL-3

LazyData true

Depends R (>= 3.6.0)

Imports stats, splines, dae, plyr, Matrix, GIGrvg, MASS, MCMCpack, grpreg, mvtnorm

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-09-17 22:50:19 UTC

R topics documented:

  NVC_frequentist .................................................. 2
  NVC_predict ..................................................... 5
  NVC_SSL .......................................................... 6
  SimulatedData .................................................... 11

Index 13
NVC_frequentist

Fits frequentist penalized nonparametric varying coefficient (NVC) models

Description

This function implements frequentist penalized nonparametric varying coefficient (NVC) models. It supports the following penalty functions: the group lasso penalty of Yuan and Lin (2006), the group minimax concave penalty (MCP) of Breheny and Huang (2015), and the group smoothly clipped absolute deviation (SCAD) penalty of Breheny and Huang (2015). This function solves a penalized regression problem of the form,

\[ \arg\max_{\gamma} \frac{1}{N} \ell(\gamma) + \text{pen}_\lambda(\gamma), \]

where \( N \) is the total number of observations, \( \ell(\gamma) \) is the loss function, and \( \text{pen}_\lambda(\cdot) \) is a penalty function with regularization parameter \( \lambda > 0 \). Since the objective function is rescaled by \( 1/N \), the penalty \( \lambda \) is typically smaller than the spike hyperparameter \( \lambda_0 \) used by the NVC_SSL function. The BIC criterion is used to select the optimal tuning parameter \( \lambda \).

Usage

NVC_frequentist(y, t, X, n_basis=8, penalty=c("gLASSO","gSCAD","gMCP"),
lambda=NULL, include_intercept=TRUE)

Arguments

y 
\( N \times 1 \) vector of response observations \( y_{11}, \ldots, y_{1m}, \ldots, y_{n1}, \ldots, y_{nm} \)
t 
\( N \times 1 \) vector of observation times \( t_{11}, \ldots, t_{1m}, \ldots, t_{n1}, \ldots, t_{nm} \)
x 
\( N \times p \) design matrix with columns \( [X_1, \ldots, X_p] \), where the \( k \)th column contains the entries \( x_{ik}(t_{ij}) \)’s
n_basis 
number of basis functions to use. Default is \( n\_basis=8 \).
penalty 
string specifying which penalty function to use. Specify "gLASSO" for group lasso, "gSCAD" for group SCAD, or "gMCP" for group MCP.
lambda 
grid of tuning parameters. If \( \lambda \) is not specified (i.e. \( \lambda=NULL \)), then the program automatically chooses a grid for \( \lambda \). Note that since the objective function is scaled by \( 1/N \), the automatically chosen grid for \( \lambda \) typically consists of smaller values than the default grid for \( \lambda_0 \) used by the function NVC_SSL.
include_intercept 
Boolean variable for whether or not to include an intercept function \( \beta_0(t) \) in the estimation. Default is \( \text{include\_intercept}=\text{TRUE} \).
Value

The function returns a list containing the following components:

- **t_ordered**
  All $N$ time points ordered from smallest to largest. Needed for plotting.

- **classifications**
  $p \times 1$ vector of indicator variables, where "1" indicates that the covariate is selected and "0" indicates that it is not selected. These classifications are determined by the optimal lambda chosen from BIC. Note that this vector does not include an intercept function.

- **beta_hat**
  $N \times p$ matrix of the estimates for varying coefficient functions $\beta_k(t), k = 1, \ldots, p$, using the optimal lambda chosen from BIC. The $k$th column in the matrix is the $k$th estimated function at the observation times in $t$._ordered.

- **beta0_hat**
  Estimate of the intercept function $\beta_0(t)$ at the observation times in $t$._ordered for the optimal lambda chosen from BIC. This is not returned if `include_intercept = FALSE`.

- **gamma_hat**
  Estimated basis coefficients (needed for prediction) for the optimal lambda.

- **lambda_min**
  The individual lambda which minimizes the BIC. If only one value was originally passed for lambda, then this just returns that lambda.

- **lambda0_all**
  Grid of all $L$ regularization parameters in lambda. Note that since the objective function is scaled by $1/N$ for the penalized frequentist methods in the NVC_frequentist function, the lambda0_all grid that is chosen automatically by NVC_frequentist typically consists of smaller values than the default values in the lambda0_all grid for NVC_SSL.

- **BIC_all**
  $L \times 1$ vector of BIC values corresponding to all $L$ entries in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.

- **beta_est_all_lambda**
  List of length $L$ of the estimated varying coefficients $\beta_k(t), k = 1, \ldots, p$, corresponding to all $L$ lambdas in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.

- **beta0_est_all_lambda**
  $N \times L$ matrix of estimated intercept function $\beta_0(t)$ corresponding to all $L$ entries in lambda0_all. The $l$th column corresponds to the $l$th entry in lambda0_all. This is not returned if `include_intercept=FALSE`.

- **gamma_est_all_lambda**
  $dp \times L$ matrix of estimated basis coefficients corresponding to all entries in lambda0_all. The $l$th column corresponds to the $l$th entry in lambda0_all.

- **classifications_all_lambda**
  $p \times L$ matrix of classifications corresponding to all the entries in lambda0_all. The $l$th column corresponds to the $l$th entry in lambda0_all.

- **iters_to_converge**
  Number of iterations it took for the group ascent algorithm to converge for each entry in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.
References


Examples

```r
## Load data
data(SimulatedData)
attach(SimulatedData)
y = SimulatedData$y
t = SimulatedData$t
id = SimulatedData$id
X = SimulatedData[,4:103]

## Fit frequentist penalized NVC model with the SCAD penalty.
## Can set penalty as "gLASSO", "gSCAD", or "gMCP".
## No need to specify an 'id' argument when using NVC_frequentist() function
NVC_gSCAD_mod = NVC_frequentist(y, t, X, penalty="gSCAD")

## Classifications. First varying coefficients are selected as nonzero
NVC_gSCAD_mod$ classifications

## Optimal lambda chosen from BIC
NVC_gSCAD_mod$ lambda_min

## Plot first estimated varying coefficient function
plot(t_ordered, beta_hat[,1], lwd=3, type='l', col='blue',
     xlab="Time", ylim = c(-12,12), ylab=expression(beta[1]))

## Plot third estimated varying coefficient function
plot(t_ordered, beta_hat[,3], lwd=3, type='l', col='blue',
     xlab="Time", ylim = c(-4,2), ylab=expression(beta[3]))

## Plot fifth estimated varying coefficient function
plot(t_ordered, beta_hat[,5], lwd=3, type='l', col='blue',
     xlab="Time", ylim = c(0,15), ylab=expression(beta[5]))
```
NVC_predict

Prediction for nonparametric varying coefficient (NVC) models

Description

This is a function to predict the responses \(y(t_{\text{new}})\) for new subjects at new time points \(t_{\text{new}}\) with new covariates \(X_{\text{new}}\). The function accepts an estimated NVC model that was fit using either the NVC_SSL or NVC_frequentist functions and returns the predicted \(y(t)\)'s. This function can be used for either out-of-sample predictions or for in-sample predictions if the "new" subjects are the same as the ones used to obtain the fitted NVC model.

Usage

NVC_predict(NVC_mod, t_new, id_new, X_new)

Arguments

NVC_mod an object with a fitted NVC model returned by the NVC_SSL or NVC_frequentist function

t_new vector of new observation times

id_new vector of new labels, where a label corresponds to one of the new subjects

X_new new design matrix with columns \([X_1, \ldots, X_p]\) where the \(k\)th column corresponds to the \(k\)th covariate. \(X_{\text{new}}\) must have the \(p\) columns, i.e. the same number of varying coefficients estimated by NVC_mod.

Value

The function returns a list containing the following components:

id vector of each \(i\)th subject’s label

time vector of each \(j\)th observation time for each \(i\)th subject

y_pred vector of predicted responses corresponding to each \(j\)th observation time for each \(i\)th subject

References


Examples

```
# Load simulated data
data(SimulatedData)
attach(SimulatedData)
y = SimulatedData$y
t = SimulatedData$t
id = SimulatedData$id
```
X = SimulatedData[,4:103]

## Fit frequentist penalized NVC model with the group lasso penalty.
## No need to specify an 'id' argument when using NVC_frequentist() function.
NVC_gLASSO_mod = NVC_frequentist(y=y, t=t, X=X, penalty="gLASSO")

## Make in-sample predictions. Here, we DO need to specify 'id' argument
NVC_gLASSO_predictions = NVC_predict(NVC_gLASSO_mod, t_new=t, id_new=id, X_new=X)

## Subjects
NVC_gLASSO_predictions$id

## Observation times
NVC_gLASSO_predictions$time

## Predicted responses
NVC_gLASSO_predictions$y_pred

## Fit NVC-SSL model to the data instead. Here, we do need to specify id
NVC_SSL_mod = NVC_SSL(y=y, t=t, id=id, X=X)
NVC_SSL_predictions = NVC_predict(NVC_SSL_mod, t_new=t, id_new=id, X_new=X)

## Subjects
NVC_SSL_predictions$id

## Observation times
NVC_SSL_predictions$time

## Predicted responses
NVC_SSL_predictions$y_pred

---

**Description**

This function implements the Nonparametric Varying Coefficient Spike-and-Slab Lasso (NVC-SSL) model of Bai et al. (2023) for high-dimensional NVC models. The function returns the MAP estimator for the varying coefficients $\beta_k(t), k = 1, \ldots, p$, obtained from the ECM algorithm described in Bai et al. (2023). The BIC criterion is used to select the optimal spike hyperparameter $\lambda_0$.

If the user specifies return_CI=TRUE, then this function will also return the 95 percent pointwise posterior credible intervals for the varying coefficients $\beta_k(t), k = 1, \ldots, p$, obtained from Gibbs
sampling. If the number of covariates $p$ is large, then the user can additionally use the approximate MCMC algorithm introduced in Bai et al. (2023) ($\text{approx.MCMC=TRUE}$) which is much faster than the exact Gibbs sampler and gives higher simultaneous coverage.

Finally, this function returns the number of iterations and the runtime for the ECM algorithms and MCMC algorithms which can be used for benchmarking and timing comparisons.

Usage

```r
NVC_SSL(y, t, id, X, n_basis=8,
lambda0=seq(from=300,to=10,by=-10), lambda1=1,
a=1, b=ncol(X), c0=1, d0=1, nu=n_basis+2, Phi=diag(n_basis),
include_intercept=TRUE, tol=1e-6, max_iter=100,
return_CI=FALSE, approx_MCMC=FALSE,
n_samples=1500, burn=500, print_iter=TRUE)
```

Arguments

- **y**: $N \times 1$ vector of response observations $y_{11}, \ldots, y_{1m_1}, \ldots, y_{n1}, \ldots, y_{nm_n}$
- **t**: $N \times 1$ vector of observation times $t_{11}, \ldots, t_{1m_1}, \ldots, t_{n1}, \ldots, t_{nm_n}$
- **id**: $N \times 1$ vector of labels, where each unique label corresponds to one of the subjects
- **X**: $N \times p$ design matrix with columns $[X_1, \ldots, X_p]$, where the $k$th column contains the entries $x_{ik}(t_{ij})$'s
- **n_basis**: number of basis functions to use. Default is $n\_basis=8$.
- **lambda0**: grid of spike hyperparameters. Default is to tune $\lambda_0$ from the grid of decreasing values $(300, 290, \ldots, 20, 10)$.
- **lambda1**: slab hyperparameter. Default is $\lambda_1=1$.
- **a**: hyperparameter in $B(a, b)$ prior on mixing proportion $\theta$. Default is $a=1$.
- **b**: hyperparameter in $B(a, b)$ prior on mixing proportion $\theta$. Default is $b=p$.
- **c0**: hyperparameter in Inverse-Gamma$(c_0/2, d_0/2)$ prior on measurement error variance $\sigma^2$. Default is $c_0=1$.
- **d0**: hyperparameter in Inverse-Gamma$(c_0/2, d_0/2)$ prior on measurement error variance $\sigma^2$. Default is $d_0=1$.
- **nu**: degrees of freedom for Inverse-Wishart prior on $\Omega$. Default is $n\_basis+2$.
- **Phi**: scale matrix in the Inverse-Wishart prior on $\Omega$. Default is the identity matrix.
- **include_intercept**: Boolean variable for whether or not to include an intercept function $\beta_0(t)$ in the estimation. Default is include\_intercept=TRUE.
- **tol**: convergence criteria for the ECM algorithm. Default is tol=1e-6.
- **max_iter**: maximum number of iterations to run ECM algorithm. Default is max\_iter=100.
- **return_CI**: Boolean variable for whether or not to return the 95 percent pointwise credible bands. Set return\_CI=TRUE if posterior credible bands are desired.
approx_MCMC  Boolean variable for whether or not to run the approximate MCMC algorithm instead of the exact MCMC algorithm. If approx_MCMC=TRUE, then an approximate MCMC algorithm is used. Otherwise, if approx_MCMC=FALSE, the exact MCMC algorithm is used. This argument is ignored if return_CI=FALSE.

n_samples  number of MCMC samples to save for posterior inference. The default is to save n_samples=1500. This is ignored if return_CI=FALSE.

burn  number of initial MCMC samples to discard during the warm-up period. Default is burn=500. This is ignored if return_CI=FALSE.

print_iter  Boolean variable for whether or not to print the progress of the algorithms. Default is print_iter=TRUE.

**Value**

The function returns a list containing the following components:

- t_ordered  all N time points ordered from smallest to largest. Needed for plotting classifications
- beta_hat  N × p matrix of the MAP estimates for varying coefficient functions β_k(t), k = 1, ..., p, using the optimal lambda0 chosen from BIC. The kth column in the matrix is the kth estimated function at the observation times in t_ordered.
- beta0_hat  MAP estimate of the intercept function β_0(t) at the observation times in t_ordered for the optimal lambda0 chosen from BIC. This is not returned if include_intercept = FALSE.
- gamma_hat  MAP estimates of the basis coefficients (needed for prediction) for the optimal lambda0.
- beta_post_mean  N × p matrix of the posterior mean estimates of the varying coefficient functions. The kth column in the matrix is the kth posterior mean estimate for β_k(t) at the observation times in t_ordered. This is not returned if return_CI=FALSE.
- beta_CI_lower  N × p matrix of the lower endpoints of the 95 percent pointwise posterior credible interval (CI) for the varying coefficient functions. The kth column in the matrix is the lower endpoint for the CI of β_k(t) at the observation times in t_ordered. This is not returned if return_CI=FALSE.
- beta_CI_upper  N × p matrix of the upper endpoints of the 95 percent pointwise posterior credible interval (CI) for the varying coefficient functions. The kth column in the matrix is the upper endpoint for the CI of β_k(t) at the observation times in t_ordered. This is not returned if return_CI=FALSE.
- beta0_post_mean  Posterior mean estimate of the intercept function β_0(t) at the observation times in t_ordered. This is not returned if return_CI=FALSE.
- beta0_CI_lower  Lower endpoints of the 95 percent pointwise posterior credible intervals (CIs) for the intercept function β_0(t) at the observation times in t_ordered. This is not returned if return_CI=FALSE.
beta0_CI_upper  Upper endpoints of the 95 percent pointwise posterior credible intervals (CIs) for the intercept function $\beta_0(t)$ at the observation times in $t_{\text{ordered}}$. This is not returned if return_CI=FALSE.

gamma_post_mean  Posterior mean estimates of all the basis coefficients. This is not returned if return_CI=FALSE.

gamma_CI_lower  Lower endpoints of the 95 percent posterior credible intervals for the basis coefficients. This is not returned if return_CI=FALSE.

gamma_CI_upper  Upper endpoints of the 95 percent posterior credible intervals for the basis coefficients. This is not returned if return_CI=FALSE.

post_incl  $p \times 1$ vector of estimated posterior inclusion probabilities (PIPs) for each of the varying coefficients. The $k$th entry in post_incl is the PIP for $\beta_k$. This is not returned if return_CI=FALSE.

lambda0_min  the individual $\lambda_0$ which minimizes the BIC. If only one value was originally passed for $\lambda_0$, then this just returns that $\lambda_0$.

lambda0_all  grid of all $L$ regularization parameters in $\lambda_0$. Note that since the objective function is scaled by $1/N$ for the penalized frequentist methods in the NVC_frequentist function, the lambda_all grid that is chosen automatically by NVC_frequentist typically consists of smaller values than the default values in the lambda0_all grid for NVC_SSL.

BIC_all  $L \times 1$ vector of BIC values corresponding to all $L$ entries in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.

beta_est_all_lambda0  list of length $L$ of the estimated varying coefficients $\beta_k(t), k = 1, ..., p$, corresponding to all $L$ lambdas in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.

beta0_est_all_lambda0  $N \times L$ matrix of estimated intercept function $\beta_0(t)$ corresponding to all $L$ entries in lambda0_all. The $l$th column corresponds to the $l$th entry in lambda0_all. This is not returned if include_intercept=FALSE.

gamma_est_all_lambda0  $dp \times L$ matrix of estimated basis coefficients corresponding to all entries in lambda0_all. The $l$th column corresponds to the $l$th entry in lambda0_all.

classifications_all_lambda0  $p \times L$ matrix of classifications corresponding to all the entries in lambda0_all. The $l$th column corresponds to the $l$th entry in lambda0_all.

ECM_iters_to_converge  number of iterations it took for the ECM algorithm to converge for each entry in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.

ECM_runtimes  $L \times 1$ vector of the number of seconds it took for the ECM algorithm to converge for each entry in lambda0_all. The $l$th entry corresponds to the $l$th entry in lambda0_all.

gibbs_runtime  number of minutes it took for the Gibbs sampling algorithm to run for the total number of MCMC iterations given in gibbs_iters.

gibbs_iters  total number of MCMC iterations run for posterior inference.
References


Examples

```r
## Load data
data(SimulatedData)
attach(SimulatedData)
y = SimulatedData$y
t = SimulatedData$t
id = SimulatedData$id
X = SimulatedData[,4:103]

## Fit NVC-SSL model. Default implementation uses a grid of 30 lambdas.
## Below illustration uses just two well-chosen lambdas
NVC_SSL_mod = NVC_SSL(y, t, id, X, lambda0=c(60,50))

## NOTE: Should use default, which will search for lambda0 from a bigger grid
# NVC_SSL_mod = NVC_SSL(y, t, id, X)

## Classifications. First 6 varying coefficients are selected as nonzero
NVC_SSL_mod$classifications

## Optimal lambda chosen from BIC
NVC_SSL_mod$lambda0_min

## Plot first estimated varying coefficient function
beta_hat= NVC_SSL_mod$beta_hat
plot(t, beta_hat[,1], lwd=3, type='l', col='blue',
     xlab="Time", ylim = c(-12,12), ylab=expression(beta[1]))

## Plot third estimated varying coefficient function
plot(t, beta_hat[,3], lwd=3, type='l', col='blue',
     xlab="Time", ylim = c(-4,2), ylab=expression(beta[3]))

## Plot fifth estimated varying coefficient function
plot(t, beta_hat[,5], lwd=3, type='l', col='blue',
     xlab="Time", ylim = c(0,15), ylab=expression(beta[5]))

## If you want credible intervals, then set return_CI=TRUE to also run Gibbs sampler.
## Below, we run a total of 1000 MCMC iterations, discarding the first 500 as burnin
## and keeping the final 500 samples for inference.
NVC_SSL_mod_2 = NVC_SSL(y, t, id, X, return_CI=TRUE, approx_MCMC=FALSE,
```
Simulated data for illustration

Description

This is a simulated dataset for illustration. It contains a total of \( N = 436 \) observations at irregularly spaced time points for \( n = 50 \) subjects. There are \( p = 100 \) covariates.

Usage

data(SimulatedData)

Details

This simulated dataset contains \( N = 436 \) observations for \( n = 50 \) subjects, with \( p = 100 \) covariates. The first column \( y \) gives the response variables, the second column \( t \) gives the observation times, the third column \( id \) gives the unique IDs for each of the 50 subjects, and columns 4-103 (\( x_1, \ldots, x_{100} \)) give the covariate values.

This synthetic dataset is a slight modification from Experiment 2 in Section 5.1 of Bai et al. (2023). We use \( p = 100 \) for illustration, instead of \( p = 500 \) as in the paper.
References

Index

NVC_frequentist, 2
NVC_predict, 5
NVC_SSL, 6
SimulatedData, 11