Package 'spfda'

March 18, 2022

Type Package

Title Function-on-Scalar Regression with Group-Bridge Penalty

Version 0.9.1

License MIT + file LICENSE

Description Implements a group-bridge penalized function-on-scalar regression model proposed by Wang et al. (2020) <arXiv:2006.10163>, to simultaneously estimate functional coefficient and recover the local sparsity.

URL https://github.com/dipterix/spfda, http://dipterix.org/spfda/

BugReports https://github.com/dipterix/spfda/issues

Imports stats,
splines,
graphics,
mathjaxr

Encoding UTF-8

RoxygenNote 7.1.2

RdMacros mathjaxr

Language en-US

Suggests grpreg,
refund

R topics documented:

fosr_vs *Ported function from 'refund' package*

Description

A modified version of fosr.vs, but with groups parameter to allow grouping time points rather than the whole coefficient when the underlying functions are locally supported.

Usage

```
fosr_vs(
  formula,
  data,
  nbasis = 10,
  method = c("ls", "grLasso", "grMCP", "grSCAD"),
  epsilon = 1e-05,
  max.iter_num = 100,
  groups = NULL
)
```

Arguments

formula, data, nbasis, method, epsilon, max.iter_num	see fosr.vs
groups	integer vector with length of number of time-points of how time-points should be grouped; default is NULL, indicating there is no local sparsity.

spfda *Sparse Function-on-scalar Regression with Group Bridge Penalty*

Description

Function-on-scalar regression model, denote n as total number of observations, p the number of coefficients, K as the number of B-splines, T as total time points.

Usage

```
spfda(
  Y,
  X,
  lambda,
  time = seq(0, 1, length.out = ncol(Y)),
  nsp = "auto",
  ord = 4,
  alpha = 0.5,
  W = NULL,
  init = NULL,
  max_iter = 50,
  inner_iter = 50,
```

```
  CI = FALSE,
  ...
)
```

Arguments

`Y`	Numeric $n \times T$ matrix, response function.
`X`	Numeric $n \times p$ matrix, design matrix
`lambda`	Regularization parameter γ
`time`	Time domain, numerical length of T
`nsp`	Integer or 'auto', number of B-splines K; default is 'auto'
`ord`	B-spline order, default is `4`; must be ≥ 3
`alpha`	Bridge parameter α, default is `0.5`
`W`	A $T \times T$ weight matrix or `NULL` (identity matrix); default is `NULL`
`init`	Initial γ; default is `NULL`
`max_iter`	Number of outer iterations
`inner_iter`	Number of $ADMM$ iterations (inner steps)
`CI`	Logical, whether to calculate theoretical confidence intervals
`...`	Ignored

Details

This function implements "Functional Group Bridge for Simultaneous Regression and Support Estimation" (https://arxiv.org/abs/2006.10163). The model estimates functional coefficients $\beta(t)$ under model

$$y(t) = X\beta(t) + \epsilon(t)$$

with B-spline basis expansion

$$\beta(t) = \gamma B(t) + R(t),$$

where $R(t)$ is B-spline approximation error. The objective function

$$\|(Y - X\gamma B)W\|_2^2 + \sum_{j,m} \left\|\gamma_j^T \mathbf{1}(B^t > 0)\right\|_1^\alpha .$$

The input response variable is a matrix. If $y_i(t)$ are observed at different time points, please interpolate (e.g. `kernel`) before feeding in.

Value

A `spfda.model` object (environment) with following elements:

B B-spline basis functions used

error Root Mean Square Error ('RMSE')

CI Whether confidence intervals are calculated

gamma B-spline coefficient $\gamma_{p\times K}$

generate_splines Function to generate B-splines given time points

K Number of B-spline basis functions

knots B-spline knots used to fit the model

predict Function to predict responses $\beta(t)$ given new `X` and/or time points

raw A list of raw variables

Examples

```
dat <- spfda_simulate()
x <- dat$X
y <- dat$Y

fit <- spfda(y, x, lambda = 5, CI = TRUE)

BIC(fit)

plot(fit, col = c("orange", "dodgerblue3", "darkgreen"),
     main = "Fitted with 95% CI", aty = c(0, 0.5, 1), atx = c(0,0.2,0.8,1))
matpoints(fit$time, t(dat$env$beta), type = 'l', col = 'black', lty = 2)
legend('topleft', c("Fitted", "Underlying"), lty = c(1,2))

print(fit)
coefficients(fit)
```

`spfda_simulate` *Generates toy example data*

Description

Synthesized functional signals with heterogeneous error. The underlying three coefficients correspond to 'dense', 'global sparse', and 'local sparse' functions. See https://arxiv.org/abs/2006.10163 for detailed configurations.

Usage

```
spfda_simulate(n = 1000, n_timepoints = 100, err = 1, scale = c(1, 1, 1))
```

Arguments

`n`	Total number of observations
`n_timepoints`	Total number of time points
`err`	Error magnitude
`scale`	the scale of coefficients length of 1 or 3.

Value

A list of data generated: `X` is scalar predictor, `Y` is functional response.

`spfda_weight` *Calculates weight matrices*

Description

Calculates weight matrices

Usage

```
spfda_weight(X, Y, bandwidth, part)
```

Arguments

`X`	design matrix
`Y`	response matrix
`bandwidth`	numeric band-width
`part`	list of time point boundaries

Value

the weight matrix

Index