Package 'mgss'

October 13, 2022

Type Package

Title A Matrix-Free Multigrid Preconditioner for Spline Smoothing

Version 1.2

Description Data smoothing with penalized splines is a popular method and is well established for one- or two-dimensional covariates. The extension to multiple covariates is straightforward but suffers from exponentially increasing memory requirements and computational complexity. This toolbox provides a matrix-free implementation of a conjugate gradient (CG) method for the regularized least squares problem resulting from tensor product B-spline smoothing with multivariate and scattered data. It further provides matrix-free preconditioned versions of the CG-algorithm where the user can choose between a simpler diagonal preconditioner and an advanced geometric multigrid preconditioner. The main advantage is that all algorithms are performed matrix-free and therefore require only a small amount of memory. For further detail see Siebenborn & Wagner (2021).

License MIT + file LICENSE

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.5), combinat (>= 0.0-8), statmod (>= 1.1), Matrix (>= 1.2)

LinkingTo Rcpp

RoxygenNote 7.1.1

Encoding UTF-8

Repository CRAN

Suggests testthat

BugReports https://github.com/SplineSmoothing/MGSS

NeedsCompilation yes

Author Martin Siebenborn [aut, cre, cph], Julian Wagner [aut, cph]

Maintainer Martin Siebenborn <martin.siebenborn@uni-hamburg.de>

Date/Publication 2021-05-10 07:50:06 UTC

R topics documented:

`CG_smooth` *High-dimensional spline smoothing using a matrix-free CG-method.*

Description

Fits a smooth spline to a set of given observations using penalized splines with curvature or difference penalty and multiple covariates. The underlying linear system is solved with a matrix-free conjugated gradient (CG) method.

Usage

```
CG_smooth(
  m,
  q,
  lambda,
  X,
  y,
  pen_type = "curve",
  l = NULL,
  alpha_start = NULL,
  K_max = NULL,
  tolerance = 1e-06,
  print_error = TRUE
)
```

Arguments

`m`	Vector of non-negative integers. Each entry gives the number of inner knots for the respective covariate.
`q`	Vector of positive integers. Each entry gives the spline degree for the respective covariate.
`lambda`	Positive number as weight for the penalty term.
`X`	Matrix containing the covariates as columns and the units as rows.
`y`	Vector of length `nrow(X)` as the variable of interest.
`pen_type`	Utilized penalization method. Either `"curve"` for the curvature penalty or `"diff"` for the difference penalty. Defaults to `"curve"`.

l	Positive integer vector of length P indicating for the penalty degree. Only required if pen_type = "diff".
alpha_start	Vector of length prod(m+q+1) as starting value for the CG-method. Defaults to zero.
K_max	Positive integer as upper bound for the number of CG-iterations. Defaults to prod(m+q+1).
tolerance	Positive number as error tolerance for the stopping criterion of the CG-method. Defaults to 1e-6.
print_error	Logical, indicating if the iteration error should be printed or not.

Value

Returns a list containing the input m, q, and Omega. Further gives the fitted spline coefficients alpha, the fitted values fitted_values, the residuals residuals, the root mean squared error rmse and the R-squared value R_squared.

Examples

```
data <- generate_test_data(100, 2)
X <- data$X_train
y <- data$y_train
CG_smooth(m = c(7,7), q = c(3,3), lambda = 0.1, X = X, y = y)
```

---

estimate_trace *Trace estimation of the hat matrix.*

Description

Estimates the trace of the (unknown) hat-matrix by stochastic estimation in a matrix-free manner.

Usage

```
estimate_trace(m, q, lambda, X, pen_type = "curve", l = NULL, n_random = 5)
```

Arguments

m	Vector of non-negative integers. Each entry gives the number of inner knots for the respective covariate.
q	Vector of positive integers. Each entry gives the spline degree for the respective covariate.
lambda	Positive number as weight for the penalty term.
X	Matrix containing the covariates as columns and the units as rows.
pen_type	Utilized penalization method. Either "curve" for the curvature penalty or "diff" for the difference penalty. Defaults to "curve".

l	Positive integer vector of length P indicating for the penalty degree. Only required if pen_type = "diff".
n_random	Positive integer for the number of random vectors in the trace estimate. Defaults to 5.

Value

An estimate of the trace of the hat-matrix.

Examples

```
data <- generate_test_data(100, 2)
X <- data$X_train
estimate_trace(m = c(7,7), q = c(2,2), lambda = 0.1, X = X)
```

---

generate_test_data *Generate multi-dimensional test data for spline smoothing.*

Description

Generate a P-dimensional test data set based on a sigmoid function.

Usage

```
generate_test_data(n, P, split = 0.8)
```

Arguments

n	Numer of samples
P	Spatial dimension
split	A value between 0 and 1 for the train / test split.

Value

A list of the covarite matrices for the train and test data X_train and X_test and of the variable of interest y_train and y_test.

Examples

```
generate_test_data(100, 2)
```

MGCG_smooth *High-dimensional spline smoothing using a matrix-free multigrid preconditioned CG-method.*

Description

Fits a smooth spline to a set of given observations using penalized splines with curvature penalty and multiple covariates. The underlying linear system is solved with a matrix-free preconditioned conjugated gradient method using a geometric multigrid method as preconditioner.

Usage

```
MGCG_smooth(
  G,
  q,
  lambda,
  X,
  y,
  w = 0.1,
  nu = c(3, 1),
  alpha_start = NULL,
  K_max = NULL,
  tolerance = 1e-06,
  print_error = TRUE,
  coarse_grid_solver = "Cholesky"
)
```

Arguments

G	Positive integer greater than one for the maximum number of grids.
q	Vector of positive integers. Each entry gives the spline degree for the respective covariate.
lambda	Positive number as weight for the penalty term.
X	Matrix containing the covariates as columns and the units as rows.
y	Vector of length nrow(X) as the variable of interest.
w	Damping factor of the Jacobi smoother. Defaults to 0.1.
nu	Two-dimensional vector of non-negative integers. Gives the number of pre- and post-smoothing steps in the multigrid algorithm.
alpha_start	Vector of length prod(m+q+1) as starting value for the MGCG-method. Defaults to zero.
K_max	Positive integer as upper bound for the number of MGCG-iterations. Defaults to prod(m+q+1).
tolerance	Positive number as error tolerance for the stopping criterion of the MGCG-method. Defaults to 1e-6.

`print_error` Logical, indicating if the iteration error should be printed or not.

`coarse_grid_solver`
Utilized coarse grid solver. Either `"PCG"` for diagonal preconditioned CG or `"Cholesky"` for Cholesky decomposition. Defaults to `"Cholesky"`.

Value

Returns a list containing the input `m = 2^G-1`, `q`, and `Omega`. Further gives the fitted spline coefficients `alpha`, the fitted values `fitted_values`, the residuals `residuals`, the root mean squared error `rmse` and the R-squared value `R_squared`.

References

Siebenborn, M. and Wagner, J. (2019) A Multigrid Preconditioner for Tensor Product Spline Smoothing. arXiv:1901.00654

Examples

```
data <- generate_test_data(100, 2)
X <- data$X_train
y <- data$y_train
MGCG_smooth(G = 3, q = c(3,3), lambda = 0.1, w = 0.8, X = X, y = y)
```

PCG_smooth *High-dimensional spline smoothing using a matrix-free PCG-method.*

Description

Fits a smooth spline to a set of given observations using penalized splines with curvature or difference penalty and multiple covariates. The underlying linear system is solved with a matrix-free preconditioned conjugated gradient (PCG) method using a diagonal preconditioner.

Usage

```
PCG_smooth(
  m,
  q,
  lambda,
  X,
  y,
  pen_type = "curve",
  l = NULL,
  alpha_start = NULL,
  K_max = NULL,
  tolerance = 1e-06,
  print_error = TRUE
)
```

Arguments

m	Vector of non-negative integers. Each entry gives the number of inner knots for the respective covariate.
q	Vector of positive integers. Each entry gives the spline degree for the respective covariate.
lambda	Positive number as weight for the penalty term.
X	Matrix containing the covariates as columns and the units as rows.
y	Vector of length nrow(X) as the variable of interest.
pen_type	Utilized penalization method. Either "curve" for the curvature penalty or "diff" for the difference penalty. Defaults to "curve".
l	Positive integer vector of length P indicating for the penalty degree. Only required if pen_type = "diff".
alpha_start	Vector of length prod(m+q+1) as starting value for the PCG-method. Defaults to zero.
K_max	Positive integer as upper bound for the number of PCG-iterations. Defaults to prod(m+q+1).
tolerance	Positive number as error tolerance for the stopping criterion of the PCG-method. Defaults to 1e-6.
print_error	Logical, indicating if the iteration error should be printed or not.

Value

Returns a list containing the input m, q, and Omega. Further gives the fitted spline coefficients alpha, the fitted values fitted_values, the residuals residuals, the root mean squared error rmse and the R-squared value R_squared.

Examples

```
data <- generate_test_data(100, 2)
X <- data$X_train
y <- data$y_train
PCG_smooth(m = c(7,7), q = c(3,3), lambda = 0.1, X = X, y = y)
```

---

predict_smooth *Predictions from model*

Description

Makes predictions of new observations from a fitted spline model.

Usage

```
predict_smooth(model_smooth, X)
```

Arguments

model_smooth	A spline model resulting from CG_smooth, PCG_smooth, or MGCG_smooth.
X	Matrix containing the new observations.

Value

Vector of length nrow(X) of predictions.

Examples

```
data <- generate_test_data(100, 2)
X <- data$X_train
y <- data$y_train
result <- PCG_smooth(m = c(7,7), q = c(3,3), lambda = 0.1, X = X, y = y, print_error = FALSE)
X_test <- data$X_test
predict_smooth(model_smooth = result, X = X_test)
```

Index