Package: NNPreg (via r-universe)

November 9, 2026

Title Nuclear Norm Penalized Regression

Version 0.1.0

Description Fits nuclear norm penalized multivariate linear regression models using the fast iterative shrinkage-thresholding algorithm (FISTA) of Beck and Teboulle (2009) <doi:10.1137/080716542>. Provides fixed-penalty estimation, regularization paths, K-fold cross-validation, prediction, coefficient extraction, and diagnostic plots.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 8.0.0

Imports graphics, stats

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Repository https://mayukhc13.r-universe.dev

Date/Publication 2026-10-24 17:30:43 UTC

RemoteUrl https://github.com/mayukhc13/nnpreg

RemoteRef HEAD

RemoteSha 7ba57c8a883ed13b73c3361f1bdcca87dfd4e099

Contents

coef.nnp 2
nnp 2
nnp_cv 3
nnp_lambda_seq 4
nnp_solver 5
nnp_solver_path 6
plot.nnp 6
predict.nnp 7

print.nnp 8
print.nnp.path 8
summary.nnp 9

Index **10**

`coef.nnp` *Extract coefficients from an NNP model*

Description

Returns the estimated coefficient matrix from a fitted nuclear norm penalized regression model.

Usage

```
## S3 method for class 'nnp'
coef(object, ...)
```

Arguments

`object`	An object of class "nnp" returned by nnp.
`...`	Additional arguments. Currently ignored.

Value

A numeric matrix containing the estimated coefficient matrix.

`nnp` *Nuclear norm penalized regression (glmnet-style interface)*

Description

Fits a nuclear norm penalized regression model either at a fixed value of the penalty parameter or using K-fold cross-validation.

Usage

```
nnp(
  X,
  Y,
  lambda = NULL,
  nfold = 5,
  nlambda = 20,
  rule = c("min", "1se"),
  maxit = 150,
  tol = 1e-07,
  eps = 1e-08,
  seed = NULL
)
```

Arguments

X	Numeric design matrix of dimension n x p.
Y	Numeric response matrix of dimension n x q.
lambda	Penalty parameter. If NULL, K-fold cross-validation is used with a default logarithmic lambda sequence. If a single value is supplied, the model is fitted at that value. If a vector is supplied, K-fold cross-validation is performed over that sequence.
nfold	Number of cross-validation folds.
nlambda	Number of lambda values when lambda = NULL.
rule	Selection rule for cross-validation: "min" or "1se".
maxit	Maximum number of FISTA iterations.
tol	Convergence tolerance.
eps	Small positive constant used in the convergence criterion.
seed	Optional random seed used when generating CV folds.

Value

An object of class "nnp" containing the fitted model and relevant information.

---

nnp_cv *Cross-validated nuclear norm penalized regression*

Description

Performs K-fold cross-validation for nuclear norm penalized regression and selects the tuning parameter by either the minimum cross-validation error or the one-standard-error rule.

Usage

```
nnp_cv(
  X,
  Y,
  nfold = 5,
  nlambda = 30,
  lambda_seq = NULL,
  rule = c("min", "1se"),
  maxit = 150,
  tol = 1e-07,
  eps = 1e-08,
  foldid = NULL,
  seed = NULL
)
```

Arguments

X	Numeric design matrix of dimension n x p.
Y	Numeric response matrix of dimension n x q.
nfold	Number of cross-validation folds.
nlambda	Number of lambda values when lambda_seq = NULL.
lambda_seq	Optional user-supplied lambda sequence.
rule	Selection rule: "min" or "1se".
maxit	Maximum number of FISTA iterations.
tol	Convergence tolerance.
eps	Small positive constant used in the convergence criterion.
foldid	Optional vector specifying the fold assignment for each observation. If NULL, folds are generated randomly.
seed	Optional random seed used when generating folds.

Value

An object of class "nnp.cv" containing the selected model, cross-validation errors, lambda values, fold assignments, and related information.

---

nnp_lambda_seq *Generate a lambda sequence for nuclear norm penalized regression*

Description

Generates a logarithmically spaced, decreasing sequence of positive tuning parameters for nuclear norm penalized regression.

Usage

```
nnp_lambda_seq(
  X,
  Y,
  nlambda = 30,
  lambda_min_ratio = 0.001,
  lambda_max_factor = 2
)
```

Arguments

X	Numeric design matrix of dimension n x p.
Y	Numeric response matrix of dimension n x q.
nlambda	Number of lambda values.
lambda_min_ratio	Smallest lambda as a fraction of lambda_max.
lambda_max_factor	Largest lambda as a multiple of lambda_max.

Details

By default, the sequence follows the convention used in the paper: from 2 * lambda_max down to 0.001 * lambda_max, where lambda_max = ||X'Y||_2.

Value

A decreasing numeric vector of lambda values.

`nnp_solver` *FISTA solver for nuclear norm penalized regression*

Description

Solves the nuclear norm penalized regression problem

Usage

```
nnp_solver(X, Y, lambda, A0 = NULL, maxit = 150, tol = 1e-07, eps = 1e-08)
```

Arguments

`X`	Numeric design matrix of dimension n x p.
`Y`	Numeric response matrix of dimension n x T.
`lambda`	Non-negative nuclear-norm penalty parameter.
`A0`	Optional initial value for the coefficient matrix. Defaults to the zero matrix.
`maxit`	Maximum number of FISTA iterations.
`tol`	Relative convergence tolerance.
`eps`	Small positive constant used in the denominator of the relative convergence criterion.

Details

min_B ||Y - X B||*F^2 + lambda ||B||*

using the FISTA algorithm.

Value

An object of class `"nnp.fit"` containing the estimated coefficient matrix, penalty parameter, number of iterations, and convergence information.

nnp_solver_path *Nuclear norm solution path*

Description

Computes the nuclear norm penalized estimator over a sequence of tuning parameters using warm starts.

Usage

```
nnp_solver_path(X, Y, lambdas, maxit = 150, tol = 1e-07, eps = 1e-08)
```

Arguments

X	Numeric design matrix of dimension n x p.
Y	Numeric response matrix of dimension n x q.
lambdas	Numeric vector of positive tuning parameters.
maxit	Maximum number of FISTA iterations for each lambda.
tol	Convergence tolerance.
eps	Small positive constant used in the convergence criterion.

Details

The lambda sequence is sorted in decreasing order so that the solution at the previous lambda can be used as the initial value for the next FISTA optimization.

Value

An object of class "nnp.path", containing the fitted coefficient matrices and convergence information for each lambda.

plot.nnp *Plot method for NNP objects*

Description

Produces cross-validation, rank, coefficient-norm, or combined diagnostic plots for a fitted NNP model.

Usage

```
## S3 method for class 'nnp'
plot(
  x,
  type = c("cv", "rank", "coef", "all"),
  log_lambda = TRUE,
  add_rug = FALSE,
  ...
)
```

Arguments

x	An object of class "nnp" returned by nnp.
type	Type of plot. One of "cv", "rank", "coef", or "all".
log_lambda	Logical; if TRUE, use log(lambda) on the horizontal axis.
add_rug	Logical; if TRUE, add a rug showing the lambda grid to the CV plot.
...	Additional graphical parameters.

---

predict.nnp *Predict method for NNP objects*

Description

Computes fitted responses from a nuclear norm penalized regression model using the supplied new design matrix.

Usage

```
## S3 method for class 'nnp'
predict(object, newx, ...)
```

Arguments

object	An object of class "nnp".
newx	New design matrix with the same number of columns as the design matrix used to fit the model.
...	Additional arguments.

Value

A numeric matrix of predicted responses.

print.nnp *Print method for NNP objects*

Description

Prints information about fixed-lambda and cross-validated nuclear norm penalized regression fits.

Usage

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

Arguments

x	An object of class "nnp" or "nnp.path".
...	Additional arguments.

print.nnp.path *Print method for NNP solution paths*

Description

Print method for NNP solution paths

Usage

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

Arguments

x	An object of class "nnp.path".
...	Additional arguments.

summary.nnp *Summary method for NNP objects*

Description

Provides a summary of fixed-lambda and cross-validated nuclear norm penalized regression fits.

Usage

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

Arguments

object	An object of class "nnp".
...	Additional arguments.

Value

The supplied object, invisibly.

Index

coef.nnp, 2

nnp, 2, 2, *7*
nnp_cv, 3
nnp_lambda_seq, 4
nnp_solver, 5
nnp_solver_path, 6

plot.nnp, 6
predict.nnp, 7
print.nnp, 8
print.nnp.path, 8

summary.nnp, 9