Package ‘SALES’

October 12, 2022

Title The (Adaptive) Elastic Net and Lasso Penalized Sparse Asymmetric Least Squares (SALES) and Coupled Sparse Asymmetric Least Squares (COSALES) using Coordinate Descent and Proximal Gradient Algorithms

Version 1.0.2

Author Yuwen Gu <yuwen.gu@uconn.edu>, Hui Zou <zouxx019@umn.edu>

Maintainer Yuwen Gu <yuwen.gu@uconn.edu>

Imports grDevices, graphics, stats, methods, Matrix

Description A coordinate descent algorithm for computing the solution paths of the sparse and coupled sparse asymmetric least squares, including the (adaptive) elastic net and Lasso penalized SALES and COSALES regressions.

License GPL (>= 2)

Encoding UTF-8

URL https://github.com/knightgu/SALES

Repository CRAN

Date/Publication 2022-08-15 18:30:16 UTC

RoxygenNote 7.2.1

Suggests testthat

R topics documented:

- coef .................................................. 2
- coef.cpernet ....................................... 3
- coef.cv.cpernet .................................. 4
- coef.cv.ernet ..................................... 5
- coef.ernet ......................................... 7
- cpernet ............................................. 8
- cv.cpernet ......................................... 11
- cv.ernet ........................................... 13
- ernet ................................................ 15
**extract_model_coefficients**

**Name**

`coef`

**Description**

`coef` is a generic function which extracts model coefficients from objects returned by modeling functions. `coefficients` is an alias for it.

**Usage**

```
coef(object, ...)  
```

**Arguments**

- `object` an object for which the extraction of model coefficients is meaningful.
- `...` other arguments.

**Value**

Coefficients extracted from the model object.

**See Also**

`coef.ernet`, `coef.cpernet`
### Description

Computes the coefficients or returns a list of the indices of the nonzero coefficients at the requested values for \( \lambda \) from a fitted cpernet object.

### Usage

```r
## S3 method for class 'cpernet'
coef(object, s = NULL, type = c("coefficients", "nonzero"), ...)
```

### Arguments

- `object`: fitted cpernet model object.
- `s`: value(s) of the penalty parameter \( \lambda \) at which predictions are to be made. Default is the entire sequence used to create the model.
- `type`: type "coefficients" computes coefficients at the requested values for \( s \). Type "nonzero" returns a list of the indices of nonzero coefficients for each value of \( s \). Default is "coefficients".
- `...`: not used. Other arguments to predict.

### Details

\( s \) is the new vector at which predictions are requested. If \( s \) is not in the \( \lambda \) sequence used for fitting the model, the `coef` function will use linear interpolation to make predictions. The new values are interpolated using a fraction of coefficients from both left and right \( \lambda \) indices.

### Value

The object returned depends on type.

### Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu &lt;yuwen.gu@uconn.edu&gt;

### See Also

cpernet, predict.cpernet, print.cpernet, plot.cpernet
Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
 pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2 <- cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
               pf.mean = pf, pf.scale = pf2, intercept = TRUE,
               standardize = FALSE, lambda2 = lambda2)

mean.coef <- as.vector(coef(m2, s = m2$lambda[50])[1])
scale.coef <- as.vector(coef(m2, s = m2$lambda[50])[2])
```

---

**coef.cv.cpernet**  
*Get coefficients from a cv.cpernet object*

Description

This function gets coefficients from a cross-validated cpernet model, using the fitted cv.cpernet object, and the optimal value chosen for lambda.

Usage

```r
## S3 method for class 'cv.cpernet'
coef(object, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

- `object`: fitted cv.cpernet object.
- `s`: value(s) of the penalty parameter lambda at which predictions are required. Default is the value `s="lambda.1se"` stored on the CV object, it is the largest value of lambda such that error is within 1 standard error of the minimum. Alternatively `s="lambda.min"` can be used, it is the optimal value of lambda that gives minimum cross validation error cvm. If `s` is numeric, it is taken as the value(s) of lambda to be used.
- `...`: not used. Other arguments to predict.

Details

This function makes it easier to use the results of cross-validation to get coefficients or make coefficient predictions.
coef.cv.ernet

Value

The object returned depends the ... argument which is passed on to the predict method for cpernet objects.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

cv.cpernet, predict.cv.cpernet

Examples

gf.set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2.cv <- cv.cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
pf.mean = pf, pf.scale = pf2,
standardize = FALSE, lambda2 = lambda2)
as.vector(coef(m2.cv, s = "lambda.min")$beta)
as.vector(coef(m2.cv, s = "lambda.min")$theta)

---

cov.ernet  Get coefficients from a cv.ernet object

Description

This function gets coefficients from a cross-validated ernet model, using the fitted cv.ernet object, and the optimal value chosen for lambda.

Usage

## S3 method for class 'cv.ernet'
coef(object, s = c("lambda.1se", "lambda.min"), ...)

---

set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2.cv <- cv.cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
pf.mean = pf, pf.scale = pf2,
standardize = FALSE, lambda2 = lambda2)
as.vector(coef(m2.cv, s = "lambda.min")$beta)
as.vector(coef(m2.cv, s = "lambda.min")$theta)
Arguments

- object: fitted \texttt{cv.ernet} object.
- \texttt{s}: value(s) of the penalty parameter \texttt{lambda} at which predictions are required. Default is the value \texttt{s=“lambda.1se”} stored on the \texttt{CV} object, it is the largest value of \texttt{lambda} such that error is within 1 standard error of the minimum. Alternatively \texttt{s=“lambda.min”} can be used, it is the optimal value of \texttt{lambda} that gives minimum cross validation error \texttt{cvm}. If \texttt{s} is numeric, it is taken as the value(s) of \texttt{lambda} to be used.
- ... not used. Other arguments to predict.

Details

This function makes it easier to use the results of cross-validation to get coefficients or make coefficient predictions.

Value

The object returned depends the ...argument which is passed on to the \texttt{predict} method for \texttt{ernet} objects.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

\texttt{cv.ernet, predict.cv.ernet}

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.90
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
lambda2 <- 1
m1.cv <- cv.ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf,

  pf2 = pf2, standardize = FALSE, intercept = FALSE,

  lambda2 = lambda2)

as.vector(coef(m1.cv, s = "lambda.min"))
```
**getcoef.ernet**

Get coefficients from an ernet object

### Description

Computes the coefficients or returns a list of the indices of the nonzero coefficients at the requested values for \( \lambda \) from a fitted ernet object.

### Usage

```r
## S3 method for class 'ernet'
coef(object, s = NULL, type = c("coefficients", "nonzero"), ...)
```

### Arguments

- **object**: fitted ernet model object.
- **s**: value(s) of the penalty parameter \( \lambda \) at which predictions are to be made. Default is the entire sequence used to create the model.
- **type**: type "coefficients" computes coefficients at the requested values for \( s \). Type "nonzero" returns a list of the indices of nonzero coefficients for each value of \( s \). Default is "coefficients".
- **...**: not used. Other arguments to predict.

### Details

\( s \) is the new vector at which predictions are requested. If \( s \) is not in the \( \lambda \) sequence used for fitting the model, the `coef` function will use linear interpolation to make predictions. The new values are interpolated using a fraction of coefficients from both left and right \( \lambda \) indices.

### Value

The object returned depends on type.

### Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

### See Also

ernet, predict.ernet, print.ernet, plot.ernet
```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.90
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
lambda2 <- 1
m1 <- ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf,
            pf2 = pf2, standardize = FALSE, intercept = FALSE,
            lambda2 = lambda2)
as.vector(coef(m1, s = m1$lambda[5]))
```

**Description**

Fits regularization paths for coupled sparse asymmetric least squares regression at a sequence of regularization parameters.

**Usage**

```r
cpernet(
  x,
  y,
  w = 1,
  nlambdas = 100L,
  method = "cper",
  lambda.factor = ifelse(2 * nobs < nvars, 0.01, 1e-04),
  lambda = NULL,
  lambda2 = 0,
  pf.mean = rep(1, nvars),
  pf2.mean = rep(1, nvars),
  pf.scale = rep(1, nvars),
  pf2.scale = rep(1, nvars),
  exclude,
  dfmax = nvars + 1,
  pmax = min(dfmax * 1.2, nvars),
  standardize = TRUE,
  intercept = TRUE,
  eps = 1e-08,
  maxit = 1000000L,
```
tau = 0.8
)

Arguments

x
matrix of predictors, of dimension (nobs * nvars); each row is an observation.

y
response variable.

w
weight applied to the asymmetric squared error loss of the mean part. See details. Default is 1.0.

nlambda
the number of lambda values (default is 100).

method
a character string specifying the loss function to use. Only cper is available now.

lambda.factor
The factor for getting the minimal lambda in the lambda sequence, where we set \( \min(\lambda) = \lambda\text{.factor} \times \max(\lambda) \) with \( \max(\lambda) \) being the smallest value of lambda that penalizes all coefficients to zero. The default value depends on the relationship between \( N \) (the number of observations) and \( p \) (the number of predictors). If \( N < p \), the default is 0.01. If \( N > p \), the default is 0.0001, closer to zero. A very small value of lambda.factor will lead to a saturated fit. The argument takes no effect if there is a user-supplied lambda sequence.

lambda
a user-supplied lambda sequence. Typically, by leaving this option unspecified users can have the program compute its own lambda sequence based on nlambda and lambda.factor. It is better to supply, if necessary, a decreasing sequence of lambda values than a single (small) value. The program will ensure that the user-supplied lambda sequence is sorted in decreasing order.

lambda2
regularization parameter lambda2 for the quadratic penalty of the coefficients. Default is 0, meaning no L2 penalization.

pf.mean, pf.scale
L1 penalty factor of length \( p \) used for adaptive LASSO or adaptive elastic net. Separate L1 penalty weights can be applied to each mean or scale coefficient to allow different L1 shrinkage. Can be 0 for some variables, which imposes no shrinkage and results in that variable being always included in the model. Default is 1 for all variables (and implicitly infinity for variables listed in exclude).

pf2.mean, pf2.scale
L2 penalty factor of length \( p \) used for adaptive elastic net. Separate L2 penalty weights can be applied to each mean or scale coefficient to allow different L2 shrinkage. Can be 0 for some variables, which imposes no shrinkage. Default is 1 for all variables.

exclude
indices of variables to be excluded from the model. Default is none. Equivalent to an infinite penalty factor.

dfmax
limit the maximum number of variables in the model. Useful for very large \( p \), if a partial path is desired. Default is \( p + 1 \).

pmax
limit the maximum number of variables ever to be nonzero. For example once \( \beta \) enters the model, no matter how many times it exits or re-enters the model through the path, it will be counted only once. Default is \( \min(dfmax \times 1.2, p) \).
standardize logical flag for variable standardization, prior to fitting the model sequence. The coefficients are always returned to the original scale. Default is TRUE.

intercept Should intercept(s) be fitted (default=TRUE) or set to zero (FALSE).

eps convergence threshold for coordinate descent. Each inner coordinate descent loop continues until the maximum change in any coefficient is less than eps. Defaults value is 1e-8.

maxit maximum number of outer-loop iterations allowed at fixed lambda values. Default is 1e7. If the algorithm does not converge, consider increasing maxit.

tau the parameter tau in the coupled ALS regression model. The value must be in (0,1) and cannot be 0.5. Default is 0.8.

Details

Note that the objective function in cpernet is

$$w^* 1' \Psi(y - X \beta, 0.5)/N + 1' \Psi(y - X \beta - X \theta, \tau)/N + \lambda_1 * \|\beta\|_1 + 0.5 \lambda_2 \|\beta\|_2^2 + \mu_1 * \|\theta\|_1 + 0.5 \mu_2 \|\theta\|_2^2,$$

where $\Psi(u, \tau) = |\tau - I(u < 0)| * u^2$ denotes the asymmetric squared error loss and the penalty is a combination of L1 and L2 terms for both the mean and scale coefficients.

For faster computation, if the algorithm is not converging or running slow, consider increasing eps, decreasing nlambda, or increasing lambda.factor before increasing maxit.

Value

An object with S3 class cpernet.

call the call that produced this object.

b0, t0 intercept sequences both of length length(lambda) for the mean and scale respectively.

beta, theta p*length(lambda) matrices of coefficients for the mean and scale respectively, stored as sparse matrices (dgCMatrix class, the standard class for sparse numeric matrices in the Matrix package). To convert them into normal R matrices, use as.matrix().

lambda the actual sequence of lambda values used.

df.beta, df.theta the number of nonzero mean and scale coefficients respectively for each value of lambda.

dim dimensions of coefficient matrices.

npasses total number of iterations summed over all lambda values.

ejerr error flag, for warnings and errors, 0 if no error.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>
References


See Also

`plot.cpernet, coef.cpernet, predict.cpernet, print.cpernet`

Examples

```r
cv.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2 <- cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
        pf.mean = pf, pf.scale = pf2,
        standardize = FALSE, lambda2 = lambda2)
```

---

**cv.cpernet**

Cross-validation for cpernet

Description

Does k-fold cross-validation for cpernet, produces a plot, and returns a value for `lambda`. This function is based on the `cv` function from the `glmnet` package.

Usage

```r
cv.cpernet(
  x,
  y,
  w = 1,
  lambda = NULL,
  pred.loss = "loss",
  nfolds = 5,
  foldid,
  tau = 0.8,
  ...
)
```
Arguments

x  
x matrix as in \texttt{cpernet}.

y  
response variable \( y \) as in \texttt{cpernet}.

w  
weight applied to the asymmetric squared error loss of the mean part. Default is 1.0.

\texttt{lambda}  
optional user-supplied lambda sequence; default is \texttt{NULL}, and \texttt{cpernet} chooses its own sequence.

\texttt{pred.loss}  
loss function used to calculate cross-validation error. The only option now is "loss", which is the asymmetric squared error loss (ASEL).

\texttt{nfolds}  
number of folds. Default value is 5. Although \texttt{nfolds} can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowed is 3.

\texttt{foldid}  
an optional vector of values between 1 and \texttt{nfolds}, identifying what fold each observation is in. If supplied, \texttt{nfolds} will be supressed.

\texttt{tau}  
the asymmetry coefficient \( \tau \) used in the asymmetric squared error loss.

\ldots  
other arguments that can be passed to \texttt{cpernet}.

Details

The function runs \texttt{cpernet} \texttt{nfolds}+1 times. The first gets the \texttt{lambda} sequence, and the remainder fits the model with each of the folds removed. The average error and standard deviation over the folds are computed.

Value

an object of class \texttt{cv.cpernet} is returned, which is a list with the ingredients of the cross-validation fit.

\texttt{lambda}  
the values of \texttt{lambda} used in the fits.

\texttt{cvm}  
the mean cross-validated error - a vector of length \texttt{length(lambda)}.

\texttt{cvsd}  
estimate of standard error of \texttt{cvm}.

\texttt{cv.upper}  
upper curve = \texttt{cvm+cvsd}.

\texttt{cv.lower}  
lower curve = \texttt{cvm-cvsd}.

\texttt{nzero}  
a list of two components, each representing the number of non-zero coefficients at each \texttt{lambda} in the mean and scale part.

\texttt{name}  
a text string indicating type of measure (for plotting purposes).

\texttt{cpernet.fit}  
a fitted \texttt{cpernet} object for the full data.

\texttt{lambda.min}  
The optimal value of \texttt{lambda} that gives minimum cross validation error \texttt{cvm}.

\texttt{lambda.1se}  
The largest value of \texttt{lambda} such that error is within 1 standard error of the minimum.
cv.ernet

Cross-validation for ernet

Description

Does k-fold cross-validation for ernet, produces a plot, and returns a value for lambda. This function is based on the cv function from the glmnet package.

Usage

```
cv.ernet(
  x,
  y,
  lambda = NULL,
  pred.loss = "loss",
  nfolds = 5,
  foldid, 
  tau = 0.5, 
  ...)
```
Arguments

x  x matrix as in \texttt{ernet}.

y  response variable \(y\) as in \texttt{ernet}.

\texttt{lambda}  optional user-supplied lambda sequence; default is \texttt{NULL}, and \texttt{ernet} chooses its own sequence.

\texttt{pred.loss}  loss function used to calculate cross-validation error. The only option now is "loss", which is the asymmetric squared error loss (ASEL).

\texttt{nfolds}  number of folds. Default value is 5. Although \texttt{nfolds} can be as large as the sample size (leave-one-out CV), it is not recommended for large datasets. Smallest value allowed is 3.

\texttt{foldid}  an optional vector of values between 1 and \texttt{nfolds}, identifying what fold each observation is in. If supplied, \texttt{nfolds} will be supressed.

\tau  the asymmetry coefficient \(\tau\) used in the asymmetric squared error loss.

...  other arguments that can be passed to \texttt{ernet}.

Details

The function runs \texttt{ernet} \texttt{nfolds}+1 times; the first to get the \texttt{lambda} sequence, and the remainder to compute the fit with each of the folds removed. The average error and standard deviation over the folds are computed.

Value

an object of class \texttt{cv.ernet} is returned, which is a list with the ingredients of the cross-validation fit.

\texttt{lambda}  the values of \texttt{lambda} used in the fits.

\texttt{cvm}  the mean cross-validated error - a vector of length \texttt{length(lambda)}.

\texttt{cvsd}  estimate of standard error of \texttt{cvm}.

\texttt{cvupper}  upper curve = \texttt{cvm}+\texttt{cvsd}.

\texttt{cvlower}  lower curve = \texttt{cvm}-\texttt{cvsd}.

\texttt{nzero}  number of non-zero coefficients at each \texttt{lambda}.

\texttt{name}  a text string indicating type of measure (for plotting purposes).

\texttt{ernet.fit}  a fitted \texttt{ernet} object for the full data.

\texttt{lambda.min}  The optimal value of \texttt{lambda} that gives minimum cross validation error \texttt{cvm}.

\texttt{lambda.1se}  The largest value of \texttt{lambda} such that error is within 1 standard error of the minimum.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>
See Also

`ernet`

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.90
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
lambda2 <- 1
m1.cv <- cv.ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf,
                   pf2 = pf2, standardize = FALSE, intercept = FALSE,
                   lambda2 = lambda2)
```

`ernet`

Regularization paths for the sparse asymmetric least squares (SALES) regression (or the sparse expectile regression)

Description

Fits regularization paths for the Lasso or elastic net penalized asymmetric least squares regression at a sequence of regularization parameters.

Usage

`ernet(
  x,
  y,
  nlambda = 100L,
  method = "er",
  lambda.factor = ifelse(nobs < nvars, 0.01, 1e-04),
  lambda = NULL,
  lambda2 = 0,
  pf = rep(1, nvars),
  pf2 = rep(1, nvars),
  exclude,
  dfmax = nvars + 1,
  pmax = min(dfmax * 1.2, nvars),
  standardize = TRUE,
  intercept = TRUE,
  eps = 1e-08,
  maxit = 1000000L,
)"`
\text{ \texttt{tau} = 0.5 \ )}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{x} \quad \text{matrix of predictors, of dimension (nobs * nvars); each row is an observation.}
  \item \texttt{y} \quad \text{response variable.}
  \item \texttt{nlambda} \quad \text{the number of lambda values (default is 100).}
  \item \texttt{method} \quad \text{a character string specifying the loss function to use. only \texttt{er} is available now.}
  \item \texttt{lambda.factor} \quad \text{The factor for getting the minimal lambda in the lambda sequence, where we set } \min(\lambda) = \lambda.factor \times \max(\lambda) \text{ with } \max(\lambda) \text{ being the smallest value of lambda that penalizes all coefficients to zero. The default depends on the relationship between } N \text{ (the number of rows in the matrix of predictors) and } p \text{ (the number of predictors). If } N < p, \text{ the default is } 0.01. \text{ If } N > p, \text{ the default is } 0.0001, \text{ closer to zero. A very small value of lambda.factor will lead to a saturated fit. It takes no effect if there is a user-supplied lambda sequence.}
  \item \texttt{lambda} \quad \text{a user-supplied lambda sequence. Typically, by leaving this option unspecified users can have the program compute its own lambda sequence based on nlambda and lambda.factor. It is better to supply, if necessary, a decreasing sequence of lambda values than a single (small) value. The program will ensure that the user-supplied lambda sequence is sorted in decreasing order before fitting the model.}
  \item \texttt{lambda2} \quad \text{regularization parameter lambda2 for the quadratic penalty of the coefficients.}
  \item \texttt{pf} \quad \text{L1 penalty factor of length } p \text{ used for the adaptive LASSO or adaptive elastic net. Separate L1 penalty weights can be applied to each coefficient to allow different L1 shrinkage. Can be 0 for some variables, which imposes no shrinkage, and results in that variable always be included in the model. Default is 1 for all variables (and implicitly infinity for variables listed in exclude).}
  \item \texttt{pf2} \quad \text{L2 penalty factor of length } p \text{ used for adaptive elastic net. Separate L2 penalty weights can be applied to each coefficient to allow different L2 shrinkage. Can be 0 for some variables, which imposes no shrinkage. Default is 1 for all variables.}
  \item \texttt{exclude} \quad \text{indices of variables to be excluded from the model. Default is none. Equivalent to an infinite penalty factor.}
  \item \texttt{dfmax} \quad \text{the maximum number of variables allowed in the model. Useful for very large } p \text{ when a partial path is desired. Default is } p + 1.
  \item \texttt{pmax} \quad \text{the maximum number of coefficients allowed ever to be nonzero. For example once } \beta \text{ enters the model, no matter how many times it exits or re-enters the model through the path, it will be counted only once. Default is } \min(dfmax*1.2, p).
  \item \texttt{standardize} \quad \text{logical flag for variable standardization, prior to fitting the model sequence. The coefficients are always returned to the original scale. Default is TRUE.}
  \item \texttt{intercept} \quad \text{Should intercept(s) be fitted (default is TRUE) or set to zero (FALSE)?}
\end{itemize}
eps  convergence threshold for coordinate descent. Each inner coordinate descent loop continues until the maximum change in any coefficient is less than eps. Defaults value is 1e-8.

maxit  maximum number of outer-loop iterations allowed at fixed lambda values. Default is 1e7. If the algorithm does not converge, consider increasing maxit.

tau  the parameter $\tau$ in the ALS regression model. The value must be in (0,1). Default is 0.5.

Details

Note that the objective function in ernet is

$$1'\Psi_\tau(y - X\beta)/N + \lambda_1 \|\beta\|_1 + 0.5\lambda_2 \|\beta\|_2^2,$$

where $\Psi_\tau$ denotes the asymmetric squared error loss and the penalty is a combination of weighted L1 and L2 terms.

For faster computation, if the algorithm is not converging or running slow, consider increasing eps, decreasing nlambda, or increasing lambda.factor before increasing maxit.

Value

An object with S3 class ernet.

call  the call that produced this object
b0  intercept sequence of length length(lambda)
beta  a p*length(lambda) matrix of coefficients, stored as a sparse matrix (dgCMatrix class, the standard class for sparse numeric matrices in the Matrix package.). To convert it into normal type matrix use as.matrix().
lambda  the actual sequence of lambda values used
df  the number of nonzero coefficients for each value of lambda.
dim  dimension of coefficient matrix
npasses  total number of iterations summed over all lambda values
jerr  error flag, for warnings and errors, 0 if no error.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

References

plot.cpernet

### Description

Produces a coefficient profile plot of the coefficient paths for a fitted cpernet object. This function is modified based on the `plot` method in the `glmnet` package.

### Usage

```r
## S3 method for class 'cpernet'
plot(x, xvar = c("norm", "lambda"), color = FALSE, label = FALSE, ...)
```

### Arguments

- `x`: fitted `cpernet` model
- `xvar`: what is on the x-axis. "norm" plots against the L1-norm of the coefficients, "lambda" against the log-lambda sequence.
- `color`: if TRUE, plot the curves with rainbow colors. Otherwise, plot the curves with gray colors. Default is FALSE.
- `label`: if TRUE, label the curves with variable sequence numbers. Otherwise, do not put labels. Default is FALSE.
- `...`: other graphical parameters to plot.

### Details

Two coefficient profile plots are produced, one for the mean coefficients and the other for the scale coefficients.
plot.cv.cpernet

Author(s)
Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

plot.cv.cpernet

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
 pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2 <- cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
               pf.mean = pf, pf.scale = pf2, intercept = TRUE,
               standardize = FALSE, lambda2 = lambda2)
plot(m2)
```

Description

Plots the cross-validated curve, and upper and lower standard deviation curves, as a function of the lambda values used. This function is modified based on the plot.cv.glmnet function from the glmnet package.

Usage

```r
## S3 method for class 'cv.cpernet'
plot(x, sign.lambda = 1, ...)
```

Arguments

- `x`: fitted cv.cpernet object
- `sign.lambda`: either plot against log(lambda) (default) or its negative if sign.lambda=-1.
- `...`: other graphical parameters to plot

plot.cv.cpernet  
Plot the cross-validated curve produced by cv.cpernet

---

plot.cv.cpernet(object, sign.lambda = 1, ...)
Details

A plot is produced.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

plot.cpernet

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
 pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2.cv <- cv.cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
                     pf.mean = pf, pf.scale = pf2,
                     standardize = FALSE, lambda2 = lambda2)
plot(m2.cv)
```

plot.cv.ernet  

Plot the cross-validated curve produced by cv.ernet

Description

Plots the cross-validated curve, and upper and lower standard deviation curves, as a function of the lambda values used. This function is modified based on the plot.cv.glmnet function from the glmnet package.

Usage

```r
## S3 method for class 'cv.ernet'
plot(x, sign.lambda = 1, ...)
```
plot.ernet

Arguments

- `x`: fitted `cv.ernet` object
- `sign.lambda`: either plot against `log(lambda)` (default) or its negative if `sign.lambda=-1`.
- `...`: other graphical parameters to plot

Details

A plot is produced.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

`plot.ernet`

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.90
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
lambda2 <- 1
m1.cv <- cv.ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf,
                  pf2 = pf2, standardize = FALSE, intercept = FALSE,
                  lambda2 = lambda2)
plot(m1.cv)
```

Description

Produces a coefficient profile plot of the coefficient paths for a fitted ernet object. This function is modified based on the `plot` method in the `glmnet` package.

Usage

```r
## S3 method for class 'ernet'
plot(x, xvar = c("norm", "lambda"), color = FALSE, label = FALSE, ...)
```
Arguments

- **x**: fitted `ernet` model
- **xvar**: what is on the x-axis. “norm” plots against the L1-norm of the coefficients, “lambda” against the log-lambda sequence.
- **color**: if TRUE, plot the curves with rainbow colors. Otherwise, plot the curves with gray colors. Default is FALSE.
- **label**: if TRUE, label the curves with variable sequence numbers. Otherwise, do not put labels. Default is FALSE.
- **...**: other graphical parameters to plot.

Details

A coefficient profile plot is produced.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

`plot.cv.ernet`

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.90
pf <- abs(rnorm(p))
 pf2 <- abs(rnorm(p))
lambda2 <- 1
m1 <- ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf, 
            pf2 = pf2, standardize = FALSE, intercept = FALSE, 
            lambda2 = lambda2)
plot(m1)
```
**predict**

**Model predictions**

**Description**

`predict` is a generic function for predictions from the results of various model fitting functions. The function invokes particular methods which depend on the class of the first argument.

**Usage**

```r
predict(object, ...)
```

**Arguments**

- `object` a model object for which prediction is desired.
- `...` additional arguments affecting the predictions produced.

**Value**

The form of the value returned by `predict` depends on the class of its argument. See the documentation of the particular methods for details of what is produced by that method.

**See Also**

`predict.ernet`, `predict.cpernet`.

---

**predict.cpernet**  
*Make predictions from a cpernet object*

**Description**

Similar to other predict methods, this function predicts fitted values from a cpernet object.

**Usage**

```r
## S3 method for class 'cpernet'
predict(object, newx, s = NULL, type = "response", ...)
```
predict.cpernet

Arguments

- **object**: fitted cpernet model object.
- **newx**: matrix of new values for \( x \) at which predictions are to be made. \NOTE: \( \text{newx} \) must be a matrix, predict function does not accept a vector or other formats of \( \text{newx} \).
- **s**: value(s) of the penalty parameter \( \lambda \) at which predictions are to be made. Default is the entire sequence used to create the model.
- **type**: type of prediction required. Only \text{response} is available. Gives predicted response for regression problems.
- **...**: Not used. Other arguments to predict.

Details

\( s \) is the new vector at which predictions are to be made. If \( s \) is not in the lambda sequence used for fitting the model, the predict function will use linear interpolation to make predictions. The new values are interpolated using a fraction of predicted values from both left and right \lambda indices.

Value

The object returned depends on type.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

cpernet, coef.cpernet, plot.cpernet, print.cpernet

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2 <- cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
               pf.mean = pf, pf.scale = pf2,
               standardize = FALSE, lambda2 = lambda2)
predict(m2, newx = x, s = m2$lambda[50])
```
predict.cv.cpernet

Make predictions from a cv.cpernet object

Description

This function makes predictions from a cross-validated cpernet model, using the fitted cv.cpernet object, and the optimal value chosen for lambda.

Usage

## S3 method for class 'cv.cpernet'
predict(object, newx, s = c("lambda.1se", "lambda.min"), ...)

Arguments

object: fitted cv.cpernet object.
newx: matrix of new values for x at which predictions are to be made. Must be a matrix. See documentation for predict.cpernet.
s: value(s) of the penalty parameter lambda at which predictions are to be made. Default is the value s = "lambda.1se" stored on the CV object. Alternatively s = "lambda.min" can be used. If s is numeric, it is taken as the value(s) of lambda to be used.
... not used. Other arguments to predict.

Details

This function makes it easier to use the results of cross-validation to make a prediction.

Value

The object returned depends the ... argument which is passed on to the predict method for cpernet objects.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

cv.cpernet, coef.cv.cpernet, plot.cv.cpernet
Examples

```r
csv.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2.cv <- cv.cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
fmean = pf, pfscale = pf2,
standardize = FALSE, lambda2 = lambda2)
as.vector(predict(m2.cv, newx = x, s = "lambda.min"))
```

predict.cv.ernet  

Make predictions from a cv.ernet object

Description

This function makes predictions from a cross-validated ernet model, using the fitted `cv.ernet` object, and the optimal value chosen for `lambda`.

Usage

```r
## S3 method for class 'cv.ernet'
predict(object, newx, s = c("lambda.1se", "lambda.min"), ...)
```

Arguments

- `object`: fitted `cv.ernet` object.
- `newx`: matrix of new values for `x` at which predictions are to be made. Must be a matrix. See documentation for `predict.ernet`.
- `s`: value(s) of the penalty parameter `lambda` at which predictions are to be made. Default is the value `s = "lambda.1se"` stored on the CV object. Alternatively `s = "lambda.min"` can be used. If `s` is numeric, it is taken as the value(s) of `lambda` to be used.
- `...`: not used. Other arguments to predict.

Details

This function makes it easier to use the results of cross-validation to make a prediction.
predict.ernet

Value

The object returned depends on the argument which is passed on to the `predict` method for `ernet` objects.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

`cv.ernet`, `coef.cv.ernet`, `plot.cv.ernet`

Examples

```r
set.seed(1)  
n <- 100  
p <- 400  
x <- matrix(rnorm(n * p), n, p)  
y <- rnorm(n)  
tau <- 0.90  
pf <- abs(rnorm(p))  
pf2 <- abs(rnorm(p))  
lambda2 <- 1  
m1.cv <- cv.ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf,  
                   pf2 = pf2, standardize = FALSE, intercept = FALSE,  
                   lambda2 = lambda2)  
as.vector(predict(m1.cv, newx = x, s = "lambda.min"))
```

---

**predict.ernet**  
Make predictions from an ernet object

Description

Similar to other predict methods, this functions predicts fitted values from a fitted ernet object.

Usage

```r
## S3 method for class 'ernet'  
predict(object, newx, s = NULL, type = "response", ...)
```
Arguments

- **object**: fitted `ernet` model object.
- **newx**: matrix of new values for \( x \) at which predictions are to be made. **NOTE**: `newx` must be a matrix, predict function does not accept a vector or other formats of `newx`.
- **s**: value(s) of the penalty parameter \( \lambda \) at which predictions are to be made. Default is the entire sequence used to create the model.
- **type**: type of prediction required. Only `response` is available. Gives predicted response for regression problems.
- **...**: Not used. Other arguments to predict.

Details

\( s \) is the new vector at which predictions are to be made. If \( s \) is not in the \( \lambda \) sequence used for fitting the model, the predict function will use linear interpolation to make predictions. The new values are interpolated using a fraction of predicted values from both left and right \( \lambda \) indices.

Value

The object returned depends on `type`.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

See Also

- `ernet`
- `coef.ernet`
- `plot.ernet`
- `print.ernet`

---

**Description**

Print a summary of the `cpernet` path at each step along the path.

**Usage**

```
## S3 method for class 'cpernet'
print(x, digits = max(3, getOption("digits") - 3), ...)
```
Arguments

- `x`: fitted `cpernet` object.
- `digits`: significant digits in the output.
- `...`: additional print arguments.

Details

The call that produced the `cpernet` object is printed, followed by a three-column matrix with columns `Df1`, `Df2` and `Lambda`. The `Df1` and `Df2` columns are the number of nonzero mean and scale coefficients respectively.

Value

A three-column matrix, the first two columns are the number of nonzero mean and scale coefficients respectively and the third column is `Lambda`.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.30
pf <- abs(rnorm(p))
pf2 <- abs(rnorm(p))
w <- 2.0
lambda2 <- 1
m2 <- cpernet(y = y, x = x, w = w, tau = tau, eps = 1e-8,
               pf.mean = pf, pf.scale = pf2,
               standardize = FALSE, lambda2 = lambda2)
print(m2)
```

---

**print.ernet**  
*Print an ernet object*

Description

Print a summary of the ernet path at each step along the path.
Usage

## S3 method for class 'ernet'
print(x, digits = max(3, getOption("digits") - 3), ...)

Arguments

x       fitted ernet object.
digits  significant digits in the output.
...     additional print arguments.

Details

The call that produced the ernet object is printed, followed by a two-column matrix with columns Df and Lambda. The Df column is the number of nonzero coefficients.

Value

a two-column matrix, the first columns is the number of nonzero coefficients and the second column is Lambda.

Author(s)

Yuwen Gu and Hui Zou

Maintainer: Yuwen Gu <yuwen.gu@uconn.edu>

Examples

```r
set.seed(1)
n <- 100
p <- 400
x <- matrix(rnorm(n * p), n, p)
y <- rnorm(n)
tau <- 0.90
pf <- abs(rnorm(p))
pr2 <- abs(rnorm(p))
lambda2 <- 1
m1 <- ernet(y = y, x = x, tau = tau, eps = 1e-8, pf = pf,
             pr2 = pr2, standardize = FALSE, intercept = FALSE,
             lambda2 = lambda2)
print(m1)
```
Index

* models
  coef.cpernet, 3
  coef.cv.cpernet, 4
  coef.cv.ernet, 5
  coef.ernet, 7
  cpernet, 8
  cv.cpernet, 11
  cv.ernet, 13
  ernt, 15
  plot.cpernet, 18
  plot.cv.cpernet, 19
  plot.cv.ernet, 20
  plot.ernet, 21
  predict.cpernet, 23
  predict.cv.cpernet, 25
  predict.cv.ernet, 26
  predict.ernet, 27
  print.cpernet, 28
  print.ernet, 29

* regression
  coef.cpernet, 3
  coef.cv.cpernet, 4
  coef.cv.ernet, 5
  coef.ernet, 7
  cpernet, 8
  cv.cpernet, 11
  cv.ernet, 13
  ernt, 15
  plot.cpernet, 18
  plot.cv.cpernet, 19
  plot.cv.ernet, 20
  plot.ernet, 21
  predict.cpernet, 23
  predict.cv.cpernet, 25
  predict.cv.ernet, 26
  predict.ernet, 27
  print.cpernet, 28
  print.ernet, 29

coef, 2
coef.alspath (coef.ernet), 7
coef.cpalspath (coef.cpernet), 3
coef.cpernet, 2, 3, 11, 24
coef.cv.cpernet, 4, 25
coef.cv.ernet, 5, 27
coef.ernet, 2, 7, 18, 28
cpernet, 3, 5, 8, 10, 12, 13, 18, 24, 25, 29
cv.cpernet, 4, 5, 11, 12, 19, 25
cv.ernet, 6, 13, 14, 21, 26, 27
ernet, 6, 7, 14, 15, 17, 22, 27, 28, 30
plot.cpernet, 3, 11, 18, 20, 24
plot.cv.ernet, 19, 19, 25
plot.cv.ernet, 20, 22, 27
plot.ernet, 7, 18, 21, 21, 28
predict, 5, 6, 23, 25, 27
predict.alspath (predict.ernet), 27
predict.cpalspath (predict.cpernet), 23
predict.cpernet, 3, 11, 23, 23
predict.cv.cpernet, 5, 25
predict.cv.ernet, 6, 26
predict.ernet, 7, 18, 23, 27
print.cpernet, 3, 11, 24, 28
print.ernet, 7, 18, 28, 29

class, 23