Package 'picasso'

October 14, 2022

Type Package

Title Pathwise Calibrated Sparse Shooting Algorithm

Version 1.3.1

Date 2019-02-10

Author Jason Ge, Xingguo Li, Haoming Jiang, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao

Maintainer Jason Ge <jiange@princeton.edu>

Depends R (>= 2.15.0), MASS, Matrix

Imports methods

Description Computationally efficient tools for fitting generalized linear model with convex or non-convex penalty. Users can enjoy the superior statistical property of non-convex penalty such as SCAD and MCP which has significantly less estimation error and overfitting compared to convex penalty such as lasso and ridge. Computation is handled by multi-stage convex relaxation and the PathwIse CAlibrated Sparse Shooting algOrithm (PICASSO) which exploits warm start initialization, active set updating, and strong rule for coordinate preselection to boost computation, and attains a linear convergence to a unique sparse local optimum with optimal statistical properties. The computation is memory-optimized using the sparse matrix output.

License GPL-3

Repository CRAN

NeedsCompilation yes

Date/Publication 2019-02-21 21:20:18 UTC

R topics documented:

`picasso-package` *PICASSO: PathwIse CAlibrated Sparse Shooting algOrithm*

Description

This package provides computationally efficient tools for fitting generalized linear model with convex and non-convex penalty. Users can enjoy the superior statistical property of non-convex penalty such as SCAD and MCP which has significantly less estimation error and overfitting compared to convex penalty such as l1 and ridge. Computation is handled by multi-stage convex relaxation and the PathwIse CAlibrated Sparse Shooting algOrithm (PICASSO) which exploits warm start initialization, active set updating, and strong rule for coordinate preselection to boost computation, and attains a linear convergence to a unique sparse local optimum with optimal statistical properties. The computation is memory-optimized using the sparse matrix output.

Details

Package:	picasso
Type:	Package
Version:	0.5.4
Date:	2016-09-20
License:	GPL-2

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso`.

coef.gaussian — *Extract Model Coefficients for an object with S3 class* "gaussian"

Description

Extract estimated regression coefficient vectors from the solution path.

Usage

```
## S3 method for class 'gaussian'
coef(object, lambda.idx = c(1:3), beta.idx = c(1:3), ...)
```

Arguments

object	An object with S3 class "gaussian"
lambda.idx	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are c(1:3).
beta.idx	The indices of the estimate regression coefficient vectors in the solution path to be displayed. The default values are c(1:3).
...	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

coef.logit — *Extract Model Coefficients for an object with S3 class* "logit"

Description

Extract estimated regression coefficient vectors from the solution path.

Usage

```
## S3 method for class 'logit'
coef(object, lambda.idx = c(1:3), beta.idx = c(1:3), ...)
```

Arguments

`object`	An object with S3 class "`logit`"
`lambda.idx`	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are `c(1:3)`.
`beta.idx`	The indices of the estimate regression coefficient vectors in the solution path to be displayed. The default values are `c(1:3)`.
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

---

coef.poisson — *Extract Model Coefficients for an object with S3 class* "`poisson`"

Description

Extract estimated regression coefficient vectors from the solution path.

Usage

```
## S3 method for class 'poisson'
coef(object, lambda.idx = c(1:3), beta.idx = c(1:3), ...)
```

Arguments

`object`	An object with S3 class "`poisson`"
`lambda.idx`	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are `c(1:3)`.
`beta.idx`	The indices of the estimate regression coefficient vectors in the solution path to be displayed. The default values are `c(1:3)`.
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

`coef.sqrtlasso` *Extract Model Coefficients for an object with S3 class* `"sqrtlasso"`

Description

Extract estimated regression coefficient vectors from the solution path.

Usage

```
## S3 method for class 'sqrtlasso'
coef(object, lambda.idx = c(1:3), beta.idx = c(1:3), ...)
```

Arguments

`object`	An object with S3 class `"sqrtlasso"`
`lambda.idx`	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are `c(1:3)`.
`beta.idx`	The indices of the estimate regression coefficient vectors in the solution path to be displayed. The default values are `c(1:3)`.
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

`eyedata` *The Bardet-Biedl syndrome Gene expression data from Scheetz et al. (2006)*

Description

Gene expression data (20 genes for 120 samples) from the microarray experiments of mammalian-eye tissue samples of Scheetz et al. (2006).

Usage

```
data(eyedata)
```

Format

The format is a list containing conatins a matrix and a vector. 1. x - an 120 by 200 matrix, which represents the data of 120 rats with 200 gene probes. 2. y - a 120-dimensional vector of, which represents the expression level of TRIM32 gene.

Details

This data set contains 120 samples with 200 predictors

Author(s)

Xingguo Li, Tuo Zhao, Tong Zhang and Han Liu
Maintainer: Xingguo Li <xingguo.leo@gmail.com>

References

1. T. Scheetz, k. Kim, R. Swiderski, A. Philp, T. Braun, K. Knudtson, A. Dorrance, G. DiBona, J. Huang, T. Casavant, V. Sheffield, E. Stone .Regulation of gene expression in the mammalian eye and its relevance to eye disease. *Proceedings of the National Academy of Sciences of the United States of America*, 2006.

See Also

picasso-package.

Examples

```
data(eyedata)
image(x)
```

`picasso` *PathwIse CAlibrated Sparse Shooting algOrithm (PICASSO)*

Description

The function "picasso" implements the user interface.

Usage

```
picasso(X, Y, lambda = NULL, nlambda = 100, lambda.min.ratio =
                 0.05, family = "gaussian", method = "l1",
                 type.gaussian = "naive", gamma = 3, df = NULL,
                 standardize = TRUE, intercept = TRUE, prec = 1e-07,
                 max.ite = 1000, verbose = FALSE)
```

Arguments

- `X` — X is an n by d design matrix where n is the sample size and d is the data dimension.
- `Y` — Y is the n dimensional response vector. Y is numeric vector for `family=``gaussian''` and `family=``sqrtlasso''`, or a two-level factor for `family=``binomial''`, or a non-negative integer vector representing counts for `family = ``gaussian''`.
- `lambda` — A sequence of decresing positive values to control the regularization. Typical usage is to leave the input `lambda = NULL` and have the program compute its own `lambda` sequence based on `nlambda` and `lambda.min.ratio`. Users can also specify a sequence to override this. Default value is from $lambda.max$ to `lambda.min.ratio*lambda.max`. The default value of $lambda.max$ is the minimum regularization parameter which yields an all-zero estimates.
- `nlambda` — The number of values used in `lambda`. Default value is 100.
- `lambda.min.ratio` — The smallest value for `lambda`, as a fraction of the uppperbound (`MAX`) of the regularization parameter. The program can automatically generate `lambda` as a sequence of length = `nlambda` starting from `MAX` to `lambda.min.ratio*MAX` in log scale. The default value is `0.05`.Caution: logistic and poisson regression can be ill-conditioned if lambda is too small for nonconvex penalty. We suggest the user to avoid using any lambda.min.raito smaller than 0.05 for logistic/poisson regression under nonconvex penalty.
- `family` — Options for model. Sparse linear regression and sparse multivariate regression is applied if `family = "gaussian"`, sqrt lasso is applied if `family = "sqrtlasso"`, sparse logistic regression is applied if `family = "binomial"` and sparse poisson regression is applied if `family = "poisson"`. The default value is `"gaussian"`.
- `method` — Options for regularization. Lasso is applied if `method = "l1"`, MCP is applied if `method = "mcp"` and SCAD Lasso is applied if `method = "scad"`. The default value is `"l1"`.
- `type.gaussian` — Options for updating residuals in sparse linear regression. The naive update rule is applied if `opt = "naive"`, and the covariance update rule is applied if `opt = "covariance"`. The default value is `"naive"`.
- `gamma` — The concavity parameter for MCP and SCAD. The default value is `3`.
- `df` — Maximum degree of freedom for the covariance update. The default value is `2*n`.
- `standardize` — Design matrix X will be standardized to have mean zero and unit standard deviation if `standardize = TRUE`. The default value is `TRUE`.
- `intercept` — Does the model has intercept term or not. Default value is `TRUE`.
- `prec` — Stopping precision. The default value is 1e-7.
- `max.ite` — Max number of iterations for the algorithm. The default value is 1000.
- `verbose` — Tracing information is disabled if `verbose = FALSE`. The default value is `FALSE`.

Details

For sparse linear regression,

$$\min_{\beta} \frac{1}{2n}||Y - X\beta - \beta_0||_2^2 + \lambda R(\beta),$$

where $R(\beta)$ can be ℓ_1 norm, MCP, SCAD regularizers.

For sparse logistic regression,

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (\log(1 + e^{x_i^T \beta + \beta_0}) - y_i x_i^T \beta) + \lambda R(\beta),$$

where $R(\beta)$ can be ℓ_1 norm, MCP, and SCAD regularizers.

For sparse poisson regression,

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^{n} (e^{x_i^T \beta + \beta_0} - y_i (x_i^T \beta + \beta_0) + \lambda R(\beta),$$

where $R(\beta)$ can be ℓ_1 norm, MCP or SCAD regularizers.

Value

An object with S3 classes `"gaussian"`, `"binomial"`, and `"poisson"` corresponding to sparse linear regression, sparse logistic regression, and sparse poisson regression respectively is returned:

`beta`	A matrix of regression estimates whose columns correspond to regularization parameters for sparse linear regression and sparse logistic regression. A list of matrices of regression estimation corresponding to regularization parameters for sparse column inverse operator.
`intercept`	The value of intercepts corresponding to regularization parameters for sparse linear regression, and sparse logistic regression.
`Y`	The value of `Y` used in the program.
`X`	The value of `X` used in the program.
`lambda`	The sequence of regularization parameters `lambda` used in the program.
`nlambda`	The number of values used in `lambda`.
`family`	The `family` from the input.
`method`	The `method` from the input.
`path`	A list of `d` by `d` adjacency matrices of estimated graphs as a graph path corresponding to `lambda`.
`sparsity`	The sparsity levels of the graph path for sparse inverse column operator.
`standardize`	The `standardize` from the input.
`df`	The degree of freecom (number of nonzero coefficients) along the solution path for sparse linear regression, nd sparse logistic regression.
`ite`	A list of vectors where the i-th entries of ite[[1]] and ite[[2]] correspond to the outer iteration and inner iteration of i-th regularization parameter respectively.
`verbose`	The `verbose` from the input.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

References

1. J. Friedman, T. Hastie and H. Hofling and R. Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 2007.
2. C.H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Annals of Statistics*, 2010.
3. J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 2001.
4. R. Tibshirani, J. Bien, J. Friedman, T. Hastie, N. Simon, J. Taylor and R. Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society: Series B*, 2012.
5. T. Zhao, H. Liu, and T. Zhang. A General Theory of Pathwise Coordinate Optimization. Techinical Report, Princeton Univeristy.

See Also

picasso-package.

Examples

```
################################################################
## Sparse linear regression
## Generate the design matrix and regression coefficient vector
n = 100 # sample number
d = 80 # sample dimension
c = 0.5 # correlation parameter
s = 20  # support size of coefficient
set.seed(2016)
X = scale(matrix(rnorm(n*d),n,d)+c*rnorm(n))/sqrt(n-1)*sqrt(n)
beta = c(runif(s), rep(0, d-s))

## Generate response using Gaussian noise, and fit sparse linear models
noise = rnorm(n)
Y = X%*%beta + noise

## l1 regularization solved with naive update
fitted.l1.naive = picasso(X, Y, nlambda=100, type.gaussian="naive")

## l1 regularization solved with covariance update
fitted.l1.covariance  = picasso(X, Y, nlambda=100, type.gaussian="covariance")

## mcp regularization
fitted.mcp = picasso(X, Y, nlambda=100, method="mcp")

## scad regularization
fitted.scad = picasso(X, Y, nlambda=100, method="scad")
```

```
## lambdas used
print(fitted.l1.naive$lambda)

## number of nonzero coefficients for each lambda
print(fitted.l1.naive$df)

## coefficients and intercept for the i-th lambda
i = 30
print(fitted.l1.naive$lambda[i])
print(fitted.l1.naive$beta[,i])
print(fitted.l1.naive$intercept[i])

## Visualize the solution path
plot(fitted.l1.naive)
plot(fitted.l1.covariance)
plot(fitted.mcp)
plot(fitted.scad)


################################################################
## Sparse logistic regression
## Generate the design matrix and regression coefficient vector
n <- 100  # sample number
d <- 80   # sample dimension
c <- 0.5   # parameter controlling the correlation between columns of X
s <- 20    # support size of coefficient
set.seed(2016)
X <- scale(matrix(rnorm(n*d),n,d)+c*rnorm(n))/sqrt(n-1)*sqrt(n)
beta <- c(runif(s), rep(0, d-s))

## Generate response and fit sparse logistic models
p = 1/(1+exp(-X%*%beta))
Y = rbinom(n, rep(1,n), p)

## l1 regularization
fitted.l1 = picasso(X, Y, nlambda=100, family="binomial", method="l1")

## mcp regularization
fitted.mcp = picasso(X, Y, nlambda=100, family="binomial", method="mcp")

## scad regularization
fitted.scad = picasso(X, Y, nlambda=100, family="binomial", method="scad")

## lambdas used
print(fitted.l1$lambda)

## number of nonzero coefficients for each lambda
print(fitted.l1$df)

## coefficients and intercept for the i-th lambda
i = 30
print(fitted.l1$lambda[i])
print(fitted.l1$beta[,i])
```

```
print(fitted.l1$intercept[i])

## Visualize the solution path
plot(fitted.l1)

## Estimate of Bernoulli parameters
param.l1 = fitted.l1$p


################################################################
## Sparse poisson regression
## Generate the design matrix and regression coefficient vector
n <- 100  # sample number
d <- 80   # sample dimension
c <- 0.5   # parameter controlling the correlation between columns of X
s <- 20    # support size of coefficient
set.seed(2016)
X <- scale(matrix(rnorm(n*d),n,d)+c*rnorm(n))/sqrt(n-1)*sqrt(n)
beta <- c(runif(s), rep(0, d-s))/sqrt(s)

## Generate response and fit sparse poisson models
p = X%*%beta+rnorm(n)
Y = rpois(n, exp(p))

## l1 regularization
fitted.l1 = picasso(X, Y, nlambda=100, family="poisson", method="l1")

## mcp regularization
fitted.mcp = picasso(X, Y, nlambda=100, family="poisson", method="mcp")

## scad regularization
fitted.scad = picasso(X, Y, nlambda=100, family="poisson", method="scad")

## lambdas used
print(fitted.l1$lambda)

## number of nonzero coefficients for each lambda
print(fitted.l1$df)

## coefficients and intercept for the i-th lambda
i = 30
print(fitted.l1$lambda[i])
print(fitted.l1$beta[,i])
print(fitted.l1$intercept[i])

## Visualize the solution path
plot(fitted.l1)
```

plot.gaussian *Plot Function for "gaussian"*

Description

Visualize the solution path of regression estimate corresponding to regularization paramters.

Usage

```
## S3 method for class 'gaussian'
plot(x, ...)
```

Arguments

`x`	An object with S3 class `"gaussian"`.
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

`plot.logit` *Plot Function for "logit"*

Description

Visualize the solution path of regression estimate corresponding to regularization paramters.

Usage

```
## S3 method for class 'logit'
plot(x, ...)
```

Arguments

`x`	An object with S3 class `"logit"`.
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

plot.poisson — *Plot Function for "poisson"*

Description

Visualize the solution path of regression estimate corresponding to regularization paramters.

Usage

```
## S3 method for class 'poisson'
plot(x, ...)
```

Arguments

`x`	An object with S3 class "`poisson`".
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

plot.sqrtlasso — *Plot Function for "sqrtlasso"*

Description

Visualize the solution path of regression estimate corresponding to regularization paramters.

Usage

```
## S3 method for class 'sqrtlasso'
plot(x, ...)
```

Arguments

`x`	An object with S3 class "`sqrtlasso`".
`...`	Arguments to be passed to methods.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

---

predict.gaussian *Prediction for an object with S3 class* "gaussian"

Description

Predicting responses of the given design data.

Usage

```
## S3 method for class 'gaussian'
predict(object, newdata, lambda.idx = c(1:3), Y.pred.idx = c(1:5), ...)
```

Arguments

object	An object with S3 class "gaussian"
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the traning data of the are used.
lambda.idx	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are c(1:3).
Y.pred.idx	The indices of the predicted response vectors in the solution path to be displayed. The default values are c(1:5).
...	Arguments to be passed to methods.

Details

predict.gaussian produces predicted values of the responses of the newdata from the estimated beta values in the object, i.e.

$$\hat{Y} = \hat{\beta}_0 + X_{new}\hat{\beta}.$$

Value

Y.pred	The predicted response vectors based on the estimated models.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

---

predict.logit *Prediction for an object with S3 class* "logit"

Description

Predicting responses of the given design data.

Usage

```
## S3 method for class 'logit'
predict(object, newdata, lambda.idx = c(1:3), p.pred.idx = c(1:5), ...)
```

Arguments

object	An object with S3 class "logit"
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the traning data of the are used.
lambda.idx	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are c(1:3).
p.pred.idx	The indices of the predicted response vectors in the solution path to be displayed. The default values are c(1:5).
...	Arguments to be passed to methods.

Details

predict.logit produces predicted values of the responses of the newdata from the estimated beta values in the object, i.e.

$$\hat{p} = \frac{e^{\hat{\beta}_0 + X_{new}\hat{\beta}}}{1 + e^{\hat{\beta}_0 + X_{new}\hat{\beta}}}.$$

Value

p.pred	The predicted response vectors based on the estimated models.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

`predict.poisson` *Prediction for an object with S3 class* "poisson"

Description

Predicting responses of the given design data.

Usage

```
## S3 method for class 'poisson'
predict(object, newdata, lambda.idx = c(1:3), p.pred.idx = c(1:5), ...)
```

Arguments

object	An object with S3 class "poisson"
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the traning data of the are used.
lambda.idx	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are c(1:3).
p.pred.idx	The indices of the predicted response vectors in the solution path to be displayed. The default values are c(1:5).
...	Arguments to be passed to methods.

Details

`predict.poisson` produces predicted response mean (which is also the parameter for poisson distribution) for the `newdata` from the estimated `beta` values in the `object`, i.e.

$$\hat{p} = e^{\hat{\beta}_0 + X_{new}\hat{\beta}}.$$

Value

p.pred	The predicted response mean vectors based on the estimated models.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

predict.sqrtlasso *Prediction for an object with S3 class* "sqrtlasso"

Description

Predicting responses of the given design data.

Usage

```
## S3 method for class 'sqrtlasso'
predict(object, newdata, lambda.idx = c(1:3), Y.pred.idx = c(1:5), ...)
```

Arguments

object	An object with S3 class "sqrtlasso"
newdata	An optional data frame in which to look for variables with which to predict. If omitted, the traning data of the are used.
lambda.idx	The indices of the regularizaiton parameters in the solution path to be displayed. The default values are c(1:3).
Y.pred.idx	The indices of the predicted response vectors in the solution path to be displayed. The default values are c(1:5).
...	Arguments to be passed to methods.

Details

predict.sqrtlasso produces predicted values of the responses of the newdata from the estimated beta values in the object, i.e.

$$\hat{Y} = \hat{\beta}_0 + X_{new}\hat{\beta}.$$

Value

Y.pred	The predicted response vectors based on the estimated models.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

print.gaussian — *Print Function for an object with S3 class* "gaussian"

Description

Print a summary of the information about an object with S3 class "gaussian".

Usage

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

Arguments

x	An object with S3 class "gaussian".
...	Arguments to be passed to methods.

Details

This call simply outlines the options used for computing a lasso object.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

picasso and picasso-package.

print.logit — *Print Function for an object with S3 class* "logit"

Description

Print a summary of the information about an object with S3 class "logit".

Usage

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

Arguments

x	An object with S3 class "logit".
...	Arguments to be passed to methods.

Details

This call simply outlines the options used for computing a logit object.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

`print.poisson` *Print Function for an object with S3 class* `poisson`

Description

Print a summary of the information about an object with S3 class `"poisson"`.

Usage

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

Arguments

`x`	An object with S3 class `"poisson"`.
`...`	Arguments to be passed to methods.

Details

This call simply outlines the options used for computing a sparse poisson regression object.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

`print.sqrtlasso` *Print Function for an object with S3 class* `"sqrtlasso"`

Description

Print a summary of the information about an object with S3 class `"sqrtlasso"`.

Usage

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

Arguments

`x`	An object with S3 class `"sqrtlasso"`.
`...`	Arguments to be passed to methods.

Details

This call simply outlines the options used for computing a lasso object.

Author(s)

Jason Ge, Xingguo Li, Mengdi Wang, Tong Zhang, Han Liu and Tuo Zhao
Maintainer: Jason Ge <jiange@princeton.edu>

See Also

`picasso` and `picasso-package`.

Index