Package 'glmgen'

June 3, 2015

Type Package

Title Fast algorithms for generalized lasso problems

Version 0.0.3

Date 2014-09-03

Author Taylor Arnold, Veeranjaneyulu Sadhanala, Ryan Tibshirani

Maintainer Taylor Arnold <taylor.arnold@acm.org>

Description Efficient algorithms for generalized lasso problems. Specialized implementations are provided to deal with particular problem classes, such as trend filtering.

License GPL-2 | GPL-3

NeedsCompilation yes

R topics documented:

`coef.glmgen` *Get coefficients from a glmgen object*

Description

Get coefficients from a glmgen object

Usage

```
## S3 method for class 'glmgen'
coef(object, lambda = NULL, ...)
```

Arguments

object	output of summary.iolm
lambda	optional vector of lambda values to calculate coefficients at. If missing, will use break points in the fit.
...	optional, currently unused, arguments

---

predict.trendfilter *Get predictions from a trendfilter object*

Description

Get predictions from a trendfilter object

Usage

```
## S3 method for class 'trendfilter'
predict(object, lambda = NULL, ...)
```

Arguments

object	output of summary.iolm
lambda	optional vector of lambda values to calculate coefficients at. If missing, will use break points in the fit.
...	optional, currently unused, arguments

---

print.glmgen *Print the output of a glmgen object*

Description

Print the output of a glmgen object

Usage

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

Arguments

x	object to print
...	optional, currently unused, arguments

print.summary.glmgen *Print the output of a glmgen summary object*

Description

Print the output of a glmgen summary object

Usage

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

Arguments

x	object to print
...	optional, currently unused, arguments

summary.glmgen *Summarize a generic glmgen object*

Description

Summarize a generic glmgen object

Usage

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

Arguments

object	object to summarize
...	optional, currently unused, arguments

summary.trendfilter *Summarize a trendfilter glmgen object*

Description

Summarize a trendfilter glmgen object

Usage

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

Arguments

object	object to summarize
...	optional, currently unused, arguments

`trendfilter` *Fit a trend filtering model*

Description

Find the trend filtering solution of some degree `k` for an arbitrary set of penalty values `lambda`. Can handle link functions of Gaussian, binomial, and Poisson penalized loss functions.

Usage

```
trendfilter(y, x, weights, k = 2L, family = c("gaussian", "logistic",
  "poisson"), method = c("admm"), lambda, nlambda = 50L,
  lambda.min.ratio = 1e-05, thinning = NULL, verbose = FALSE,
  control = trendfilter.control.list())
```

Arguments

`y`	vector of observed data points.
`x`	optional vector of observed data locations. If missing, will be assumed to be the integers 1 through the length of `y`.
`weights`	optional vector of sample weights. If missing, the weights will be assumed to be constant (unity) across all samples.
`k`	the polynomial order of the trendfilter fit; a nonnegative integer (orders larger than 3 are not recommended). For instance, constant trend filtering (i.e., the fused lasso) uses `k` equal to 0, linear trend filtering uses `k` equal to 1, quadratic trend filtering uses `k` equal to 2, etc.
`family`	the family for the link function in the trend filtering estimator. Can be either "gaussian", "logistic", or "poisson".
`method`	the method used to calculate the fit. Currently only 'admm' is supported.
`lambda`	a sequence of lambda values at which to produce a fit. Can be left blank (highly recommended for general use), at which point the algorithm will determine appropriate lambda values.
`nlambda`	if `lambda` is missing, this determines the number of lambda values dynamically constructed by the algorithm.
`lambda.min.ratio`	if `lambda` is missing, this determines the ratio between the largest and smallest `lambda` values. The values are evenly spaced on a log scale, so this ratio should typically be set fairly small.
`thinning`	logical. If true, then the data are preprocessed so that a smaller, better conditioned data set is used for fitting. When set to `NULL`, the default, function will auto detect whether thinning should be applied (i.e., cases in which the numerical fitting algorithm will struggle to converge).
`verbose`	logical. Should the function print out intermediate results as it is running.
`control`	an optional named list of control parameters to pass to the underlying algorithm; see Details for more information. Names not matching any valid parameters will be silently ignored.

Details

Further algorithmic parameters can be passed by using `trendfilter.control.list`.

Value

an object of class 'trendfilter'.

Author(s)

Taylor Arnold, Aaditya Ramdas, Veeranjaneyulu Sadhanala, Ryan Tibshirani

References

Tibshirani, R. J. (2014), "Adaptive piecewise polynomial estimation via trend filtering", Annals of Statistics 42 (1): 285–323.

Ramdas, A. and Tibshirani R. J. (2014), "Fast and flexible ADMM algorithms for trend filtering", arXiv: 1406.2082.

See Also

`trendfilter.control.list`

Examples

```
set.seed(0)
 n = 100
 x = runif(n, min=-2*pi, max=2*pi)
 y = 1.5*sin(x) + sin(2*x) + rnorm(n, sd=0.2)
 out = trendfilter(y, x, k=2)

 xx = seq(min(x),max(x),length=100)
 lambda = out$lambda[25]
 yy = predict(out,x.new=xx,lambda=lambda)
 plot(x,y)
 lines(xx,yy,col=2)
```

`trendfilter.control.list` *Control list for tuning trend filtering algorithm*

Description

Constructs the control parameters for the trend filtering algorithm. Allows the user to customize as many or as little as desired.

Usage

```
trendfilter.control.list(rho = 1, obj_tol = 1e-06, max_iter = 200L,
  max_iter_newton = 50L, x_cond = 1e+11, alpha_ls = 0.5, gamma_ls = 0.8,
  max_iter_ls = 20L)
```

Arguments

rho	this is a scaling factor for the augmented Lagrangian parameter in the ADMM algorithm. To solve a given trend filtering problem with locations x at a tuning parameter value lambda, the augmented Lagrangian parameter is set to be rho * lambda * ((max(x)-min(x))/n)^k.
obj_tol	the tolerance used in the stopping criterion; when the relative change in objective values is less than this value, the algorithm terminates.
max_iter	number of ADMM iterations used; ignored for k=0.
max_iter_newton	for non-Gaussian GLM losses, the number of outer iterations used in Newton's method.
x_cond	condition number to control the degree of thinning, when applicable. Lower numbers enforce more thinning.
alpha_ls	tuning parameter for the line search used in the proximal Newton procedure for non-Gaussian GLM losses.
gamma_ls	tuning parameter for the line search used in the proximal Newton for non-Gaussian GLM losses.
max_iter_ls	tuning parameter for the number of line search iterations in the proximal Newton procedure for non-Gaussian GLM losses.

Value

a list of parameters.

Author(s)

Taylor Arnold, Veeranjaneyulu Sadhanala, Ryan Tibshirani

See Also

trendfilter

Examples

```
set.seed(0)
 n = 100
 x = runif(n, min=-2*pi, max=2*pi)
 y = 1.5*sin(x) + sin(2*x) + rnorm(n, sd=0.2)
 out = trendfilter(y, x, k=2, control=trendfilter.control.list(rho=3))
```

Index