Package 'MultivarTV'

April 8, 2018

Type Package

Title Mesh Based Solutions to Multivariate Total Variation Problems

Version 1.0

Date 2018-04-05

Author Brayan Ortiz

Maintainer Brayan Ortiz <brayan@uw.edu>

Description Efficient procedures written in C++ for fitting approximate solutions to multivariate total variation denoising problems. The algorithm uses the alternating direction method of multipliers (ADMM), as described by Boyd et al. (2011).

License GPL (>= 2)

Imports Rcpp (>= 0.12.16)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 6.0.1

NeedsCompilation yes

R topics documented:

`MultivarTV-package` *Mesh Based Solutions to Multivariate Total Variation Problems*

Description

Efficient procedures written in C++ for fitting approximate solutions to multivariate total variation denoising problems. The algorithm uses the alternating direction method of multipliers (ADMM), as described by Boyd et al. (2011).

Details

The DESCRIPTION file:

Package:	MultivarTV
Type:	Package
Title:	Mesh Based Solutions to Multivariate Total Variation Problems
Version:	1.0
Date:	2018-04-05
Author:	Brayan Ortiz
Maintainer:	Brayan Ortiz <brayan@uw.edu>
Description:	Efficient procedures written in C++ for fitting approximate solutions to multivariate total variation denoising
License:	GPL (>= 2)
Imports:	Rcpp (>= 0.12.16)
LinkingTo:	Rcpp, RcppArmadillo
RoxygenNote:	6.0.1

Index of help topics:

```
MultivarTV-package      Mesh Based Solutions to Multivariate Total
                        Variation Problems
gen_mesh                Generate a mesh
mvtv                    MVTV Generic Class
mvtv.default            Default Multivariate Total Variation Denoising
                        Solver
plotFits                Plotting Fitted Surface, p=1
plotResiduals           Plotting Residuals
predict.mvtv            MVTV Predict Function
```

This section should provide a more detailed overview of how to use the package, including the most important functions.

Author(s)

Brayan Ortiz

Maintainer: Brayan Ortiz <brayan@uw.edu>

References

This optional section can contain literature or other references for background information.

See Also

Optional links to other man pages

Examples

```
## Optional simple examples of the most important functions
## Use \dontrun{} around code to be shown but not executed
```

`gen_mesh` *Generate a mesh*

Description

Single function to handle creating a mesh regularly across domain of predictors. Mesh created is a convex hull of predictor space.

Usage

```
gen_mesh(data, m, mesh)
```

Arguments

`data`	n by p matrix of inputs
`m`	vector of length p with number of knots desired for each predictor
`mesh`	NULL; otherwise, takes user defined mesh.

`mvtv` *MVTV Generic Class*

Description

Defining MVTV Generic Class

Usage

```
mvtv(data, ...)
```

Arguments

`data`	n by p matrix of data
`...`	ignore

`mvtv.default` *Default Multivariate Total Variation Denoising Solver*

Description

Create a mesh and find cross-validated best approximation to total variation denoising problem.

Usage

```
## Default S3 method:
mvtv(data, y, m, mesh = NULL, n_lambda = 100L,
  ftrue = NULL, lambdas = NULL, folds = 5L, verbose = TRUE)
```

Arguments

`data`	n by p matrix of inputs
`y`	response column vector
`m`	vector of number of mesh points per predictor
`mesh`	user can supply or NULL for regularly spaced mesh, which will be returned
`n_lambda`	number of logarithmically spaced tuning parameters
`ftrue`	prediction target. If NULL, use observed data.
`lambdas`	user can supply vector of lambdas to be solved over. If NULL, function generates n_lambda logarithmically spaced lambdas from 0.00001*lambda_max and lambda_max, where lambda_max is our approximation of smallest lambda where regularization ends.
`folds`	number of folds for cross-validation
`verbose`	Default: true, prints out current working penalty and number of iters to solve.

Examples

```
# Approximating Bivariate Fused Lasso for Uniform Data
## Generate Data
set.seed(117)
x <- matrix(runif(100),ncol = 2)
y <- matrix(runif(50),ncol=1)
m <- matrix(c(3,3))

## Find Total Variation Solution over range of lambdas and whole data set
mvtv_fold1 <- mvtv(x,y,m,folds=1, verbose = FALSE)

## Find 5-fold validated MVTV Model over range of lambdas
mvtv_fold5 <- mvtv(x,y,m,folds=5, verbose = FALSE)
```

plotFits *Plotting Fitted Surface, p=1*

Description

Plotting fitted values for an 'mvtv' Object

Usage

```
plotFits(mvtvmodel, addmesh = FALSE)
```

Arguments

mvtvmodel	object of class 'mvtv'
addmesh	If TRUE, plot has vertical grey lines along x-axis value of mesh

plotResiduals *Plotting Residuals*

Description

Plotting residuals for an 'mvtv' Object

Usage

```
plotResiduals(mvtvmodel)
```

Arguments

mvtvmodel	object of class 'mvtv'

predict.mvtv *MVTV Predict Function*

Description

Use fitted 'mvtv' object to predict new data.

Usage

```
## S3 method for class 'mvtv'
predict(mvtvobject, data = NULL)
```

Arguments

mvtvobject	object produced by mbtv.default
data	n by p matrix of inputs

Examples

```
# Approximating Bivariate Fused Lasso for Uniform Data
## Generate Data
set.seed(117)
x <- matrix(runif(100),ncol = 2)
y <- matrix(runif(50),ncol=1)
m <- matrix(c(3,3))

## Find 5-fold validated MBS Model over range of lambdas
mbs_fold5 <- mvtv(x,y,m,folds=5,verbose=FALSE)

# Access fitted values of training data; equivalent to mbs_fold5$fitted
fitted.values <- predict(mbs_fold5)
newdata <- matrix( runif(50), ncol = 2) # Generate new data
newfits <- predict(mbs_fold5, newdata) # Fit new data
```

Index