Documentation for the R-code to implement the GDS methodology in “The Double Dantzig”

GARETH M. JAMES[*] and PETER RADCHENKO[*]

Description

This is a set of functions that fit the Generalized Dantzig Selector (GDS) and the Double Dantzig (DD) approaches. The main fitting functions for GDS and DD are “gds()” and “dd()”. In addition “dd.cv()” performs cross-validation to choose the tuning parameter for the double dantzig procedure. We provide documentation for these functions below.

Installation

To install the software download the file “gdscode”, open R and type

```
> source(‘‘gdscode.txt’’)
```

(or what ever name you gave the file) don’t forget to give the directory structure in addition if needed. Now if you type ls() you should see

```
> ls()
[1] "dd"          "dd.cv"       "gds"         "linearoptim" "X"
[6] "ybinom"
```

“X” and “ybinom” are a set of predictors and associated binomial responses. The other objects are various functions. Before gds() or dd() can be run we must attach the lpSolve library (you may need to download this from the web by clicking on Packages and Install Packages on the R gui). Try typing the following commands.

```
> library(lpSolve)
> testfit <- dd(ybinom,X,lambda=.3,int=F)
> testfit
$gds.beta
 [1] 0.75870507 0.32519125 0.09831298 0.31870754 0.08166578 0.00000000
 [7] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
[13] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
```

[*]Marshall School of Business, University of Southern California

```
[19] 0.00000000 0.00000000

$dd.beta
 [1] 1.5981210 1.3111098 0.7112378 1.1103524 1.0970770 0.0000000 0.0000000
 [8] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[15] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000

$gauss.beta
 [1] 1.6034914 1.3172906 0.7149947 1.1153788 1.1037017 0.0000000 0.0000000
 [8] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
[15] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000

$lambda
[1] 0.3

$lambda2
[1] 0.001
```

The vectors gds.beta and dd.beta represent the estimates using GDS and DD while gauss.beta corresponds to the Gauss-Dantzig estimates (DD with $\lambda_2 = 0$). This data was generated using $\beta_1 = \cdots = \beta_5 = 1.5$ and $\beta_6 = \cdots = \beta_{20} = 0$ with no intercept. Hence, all three methods pick out the correct set of βs. However, the GDS estimates are biased towards zero while the DD and Gauss-Dantzig estimates are closer to the truth. Next try

```
> cvfit <- dd.cv(ybinom,X,c(.2,.3,.4),int=F,trace=T,cv=30)
[1] "Finished lambda =  0.2"
[1] "Finished lambda =  0.3"
[1] "Finished lambda =  0.4"
[1] "Min lambda =  0.3"
> cvfit$error
[1] 108.24543  96.66586 104.48414
```

Here we have computed the cross-validated deviance for $\lambda = 0.2, 0.3, 0.4$. Your answer will vary somewhat because of randomization in the cross-validation.

Details of all the functions are provided below.

dd Function

This is the main function to compute the GDS, DD and Gauss-Dantzig estimates.

Arguments

dd() allows for up to ten inputs. They are:

y : A vector of n responses.

X : This is an n by p matrix. The jth column corresponds to the recorded values for the jth predictor $\mathbf{X}_j$.

lambda : This corresponds to the tuning parameter λ in the GDS algorithm. In practice the value used for the tuning parameter is actually $\lambda\sqrt{2\log p}$ so that the number of predictors is automatically taken into account. In general the larger that λ is the larger the feasible region for the linear program. If λ is chosen large enough β will simply be estimated as a zero vector (this is the sparsest solution). As λ is decreased we force a closer fit to the data. dd.cv() can be used to select λ. By default lambda=1.

tol : The minimum percentage absolute change in the β vector in the iterative procedure for convergence to be declared. By default tol=0.001.

maxit : The maximum number of iterations before the iterative procedure stops even if it has not reached convergence. By default maxit=10.

family : This specifies the family to be assumed for the response. Common examples include gaussian, binomial and poisson. If gaussian is chosen then the standard Dantzig Selector is implemented. By default family=binomial.

int : Should the model include an intercept term i.e. β_0? By default int=T.

lambda2 : This is the value of λ to use for the second fit to the reduced set of predictors when computing the Double Dantzig fit. By default lambda2=0.001.

tol.zero : This indicates how close to zero the GDS estimate for β needs to be before the corresponding coefficient is set to zero when calling the DD. By default tol.zero=10^{-6}.

gauss : This indicates whether the Gauss-Dantzig estimator should also be computed. By default gauss=T.

Value

dd() returns a list containing six components. Some of these just correspond to the inputs i.e. lambda and lambda2. The remaining components in the list are:

gds.beta : A vector corresponding to the GDS estimates for the coefficients.

dd.beta : A vector corresponding to the DD estimates for the coefficients.

gauss.beta : A vector corresponding to the Gauss-Dantzig estimates for the coefficients.

iterations : The number of iterations required to fit the GDS. If this number is equal to maxit then GDS may not have converged.

Note, if an intercept is selected then the first element of each β vector corresponds to β_0.

dd.cv Function

This function uses cross validation to estimate the optimal value for λ. It takes as input a vector of λ values and estimates the cross validated deviance for each input.

Arguments

dd.cv allows for up to nine inputs. They are:

y : A vector of n responses.

X : This is an n by p matrix. The jth column corresponds to the recorded values for the jth predictor $\mathbf{X}_j$.

lambda : This corresponds to the tuning parameter λ in the GDS algorithm except it should be a vector of λ's for which you wish to compute the cross-validated deviance. In practice the value used for the tuning parameter is actually $\lambda\sqrt{2\log p}$ so that the number of predictors is automatically taken into account.

cv : The number of parts to divide the data set into for performing cross-validation. By default cv=10.

s : An optional vector indicating the order to divide the observations into for the cross-validation step. Specifying s removes the random component. If s is not specified the observations are randomly divided up. By default s=NULL.

family : This specifies the family to be assumed for the response. Common examples include gaussian, binomial and poisson. If gaussian is chosen then the standard Dantzig Selector is implemented. By default family=binomial.

int : Should the model include an intercept term i.e. β_0? By default int=T.

lambda2 : This is the value of λ to use for the second fit to the reduced set of predictors when computing the Double Dantzig fit. By default lambda2=0.001.

trace : If trace=T then a message is printed for each value of λ to indicate the cross-validation has been completed. In addition the optimal λ is printed on the screen. By default trace=F which indicates no messages.

Value

dd.cv() will print out the optimal CV value for λ (if trace=T). In addition it returns a list with four components. lambda is just the vector of values that were supplied. In addition

error : A vector corresponding to cross-validated deviances for the different values of lambda.

s : The permutation of the data that was used.

optimal : The optimal cross-validated value of λ.

gds Function

This function computes the GDS estimates. dd() calls gds() to compute the DD fit. Note that dd() returns the GDS estimates so the only reason to call gds() directly is to save the small amount of extra computation involved in computing the DD solution.

Arguments

gds() allows for up to seven inputs. They are:

y : A vector of n responses.

X : This is an n by p matrix. The jth column corresponds to the recorded values for the jth predictor $\mathbf{X}_j$.

lambda : This corresponds to the tuning parameter λ in the GDS algorithm. In practice the value used for the tuning parameter is actually $\lambda\sqrt{2\log p}$ so that the number of predictors is automatically taken into account. In general the larger that λ is the larger the feasible region for the linear program. If λ is chosen large enough β will simply be estimated as a zero vector (this is the sparsest solution). As λ is decreased we force a closer fit to the data. By default lambda=1.

tol : The minimum percentage absolute change in the β vector in the iterative procedure for convergence to be declared. By default tol=0.001.

maxit : The maximum number of iterations before the iterative procedure stops even if it has not reached convergence. By default maxit=10.

family : This specifies the family to be assumed for the response. Common examples include gaussian, binomial and poisson. If gaussian is chosen then the standard Dantzig Selector is implemented. By default family=binomial.

int : Should the model include an intercept term i.e. β_0? By default int=T.

Value

gds() returns a list containing four components. One of these, lambda, just corresponds to the input. The remaining components in the list are:

beta : A vector corresponding to the GDS estimates for the coefficients.

iterations : The number of iterations required to fit the GDS. If this number is equal to maxit then GDS may not have converged.

L1 : The L1 norm of the standardized β's.

Note, if an intercept is selected then the first element of the β vector corresponds to β_0.