title: " Coordinate Descent for Lasso"
author: "Harini"
date: "15 September 2020"
output: Doc_document

Coordinate Descent for Lasso

First, prepare the Boston Housing Data. Check [Stat542_ Variable Selection.html] on relevant background information.

```
library(MASS)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

```
myData = Boston
names(myData)[14] = "Y"
iLog = c(1, 3, 5, 6, 8, 9, 10, 14);
myData[, iLog] = log(myData[, iLog]);
myData[, 2] = myData[, 2] / 10;
myData[, 7] = myData[, 7]^2.5 / 10^4
myData[, 11] = exp(0.4 * myData[, 11]) / 1000;
myData[, 12] = myData[, 12] / 100;
myData[, 13] = sqrt(myData[, 13]);

X = as.matrix(myData[, -14]) y
= myData$Y

lam.seq = c(0.30, 0.2, 0.1, 0.05, 0.02, 0.005)
```

Next write my own function to implement CD, which should output estimated Lasso coefficients similar to the one output by R.

```
MyLasso = function(X, y, lam.seq, maxit , standardize){
  # X: n-by-p design matrix without the intercept
  # y: n-by-1 response vector
  # lam.seq: sequence of lambda values
  # maxit: number of updates for each lambda
  # standardize: if True, center and scale X and y.

  n = length(y)
  p = dim(X)[2]
  nlam = length(lam.seq)

  if(standardize==TRUE){
    Xmean = apply(X,2,mean)
    Xsd = apply(X,2,sd)*sqrt((n-1)/n)
    X = t((t(X)-Xmean)/Xsd)

    ymean = mean(y)
```

```
    ysd = sd(y)*sqrt((n-1)/n)
    y = (y-ymean)/ysd
    #  Center and scale X and y
    #  Record the corresponding means and scales
  }

  # Initilize coef vector b and residual vector r
  b = rep(0, p)
  r = y
  B = matrix(0,nlam,p+1)

  one_step_lasso = function(r, x, lam){
    xx  = sum(x^2)
    xr = sum(r*x)
    b = (abs(xr) - lam/2)/xx
    b = sign(xr)*ifelse(b>0, b, 0)
    return(b)
  }

  #  Triple nested loop
  for(m in 1:nlam){
    lam = (2*n)*lam.seq[m]/ysd # assign lambda value
    for(step in 1:maxit){
      for(j in 1:p){

        r = r + (X[,j]*b[j])
        b[j] = one_step_lasso(r, X[, j], lam)
        r = r - X[, j] * b[j]

      }
    }
    B[m, -1] = b
  }

  if(standardize==TRUE){
    for(m in 1:nlam){
      B[m,-1] = B[m,-1]/Xsd*ysd
      B[m,1] = ymean - sum(B[m,-1]%*%Xmean)
    }
    # scale back the coefficients and update the intercepts B[, 1]
  }

  return(t(B))
}
```

Check the accuracy of my algorithm against the output from glmnet.

```
lam.seq = c(0.30, 0.2, 0.1, 0.05, 0.02, 0.005)
lasso.fit = glmnet(X, y, alpha = 1, lambda = lam.seq, standardize = TRUE)
coef(lasso.fit)
```

```
## 14 x 6 sparse Matrix of class "dgCMatrix"
##                      s0          s1          s2          s3          s4
## (Intercept)  3.16239335  3.5089461  3.855935763  3.778455800  3.542129253
```

```
## crim                .                .                .                .                .
## zn                  .                .                .                .                .
## indus               .                .                .                .                .
## chas                .                .                .                .                0.066692255
## nox                 .                .                .                .                .
## rm                  .                .                .                0.240416063      0.417249062
## age                 .                .                .                .                .
## dis                 .                .                .                .                -0.004681106
## rad                 .                .                .                .                .
## tax                 .                .                .                -0.055308804     -0.080579126
## ptratio             .                .                -0.004310305     -0.023647910     -0.033626958
## black               .                .                .                0.008515144      0.031234249
## lstat               -0.03741741      -0.1388176       -0.237634045     -0.244558377     -0.243489936
##                     s5
## (Intercept)  3.925458057
## crim                .
## zn                  .
## indus               -0.005398865
## chas                0.099119798
## nox                 -0.125162756
## rm                  0.459272519
## age                 .
## dis                 -0.120437573
## rad                 0.014980251
## tax                 -0.149039129
## ptratio             -0.038211176
## black               0.043402206
## lstat               -0.256708941
```

```
myout = MyLasso(X, y, lam.seq, maxit = 50, standardize = TRUE)
rownames(myout) = c("Intercept", colnames(X))
myout
```

```
##                    [,1]          [,2]          [,3]          [,4]          [,5]
## Intercept     3.16239335    3.5089461   3.855935052   3.777919609   3.541573070
## crim          0.00000000    0.0000000   0.000000000   0.000000000   0.000000000
## zn            0.00000000    0.0000000   0.000000000   0.000000000   0.000000000
## indus         0.00000000    0.0000000   0.000000000   0.000000000   0.000000000
## chas          0.00000000    0.0000000   0.000000000   0.000000000   0.066670285
## nox           0.00000000    0.0000000   0.000000000   0.000000000   0.000000000
## rm            0.00000000    0.0000000   0.000000000   0.240724202   0.417613831
## age           0.00000000    0.0000000   0.000000000   0.000000000   0.000000000
## dis           0.00000000    0.0000000   0.000000000   0.000000000  -0.004686483
## rad           0.00000000    0.0000000   0.000000000   0.000000000   0.000000000
## tax           0.00000000    0.0000000   0.000000000  -0.055335623  -0.080634128
## ptratio       0.00000000    0.0000000  -0.004303294  -0.023646161  -0.033638505
## black         0.00000000    0.0000000   0.000000000   0.008522442   0.031255297
## lstat        -0.03741741   -0.1388176  -0.237638248  -0.244528823  -0.243439646
##                    [,6]
## Intercept     3.927045881
## crim          0.000000000
## zn            0.000000000
## indus        -0.005275075
## chas          0.099049491
## nox          -0.123495061
```

```
## rm           0.459679619
## age          0.000000000
## dis         -0.119808332
## rad          0.015118868
## tax         -0.149512107
## ptratio     -0.038234008
## black        0.043452192
## lstat       -0.256682319
```

The maximum difference between the two coefficient matrices is less than 0.005.

```
max(abs(coef(lasso.fit) - myout))
```

```
## [1] 0.001667695
```