Math 564: HW#5

Aleksei Sorokin | A20394300 | asorokin@hawk.iit.edu

Problem 1: Ex 9.1

```
lambdas <- c( 4.603,  1.175,  0.203,  0.015,  0.003,  0.001)
evec1 <-   c(-0.462, -0.462, -0.321, -0.202, -0.462, -0.465)
evec2 <-   c( 0.058,  0.053, -0.596,  0.798, -0.046,  0.001)
evec3 <-   c(-0.149, -0.278,  0.728,  0.562, -0.196, -0.128)
evec4 <-   c(-0.793,  0.122, -0.008,  0.077,  0.590,  0.052)
evec5 <-   c( 0.338, -0.150,  0.009,  0.024,  0.549, -0.750)
evec6 <-   c(-0.135,  0.818,  0.107,  0.018, -0.312, -0.450)
```

Part a

We count the number of j's for which

$$\kappa_j = \sqrt{\frac{\lambda_1}{\lambda_j}} \geq 15.$$

```
kappas <- sqrt(lambdas[1]/lambdas)
kappas # condition indices
```

```
## [1]  1.000000  1.979254  4.761814 17.517610 39.170567 67.845413
```

```
sum(kappas>15) # sets of collinearity
```

```
## [1] 3
```

Part b

λ_6, λ_5, and λ_4 are small so we look at their eigenvectors.

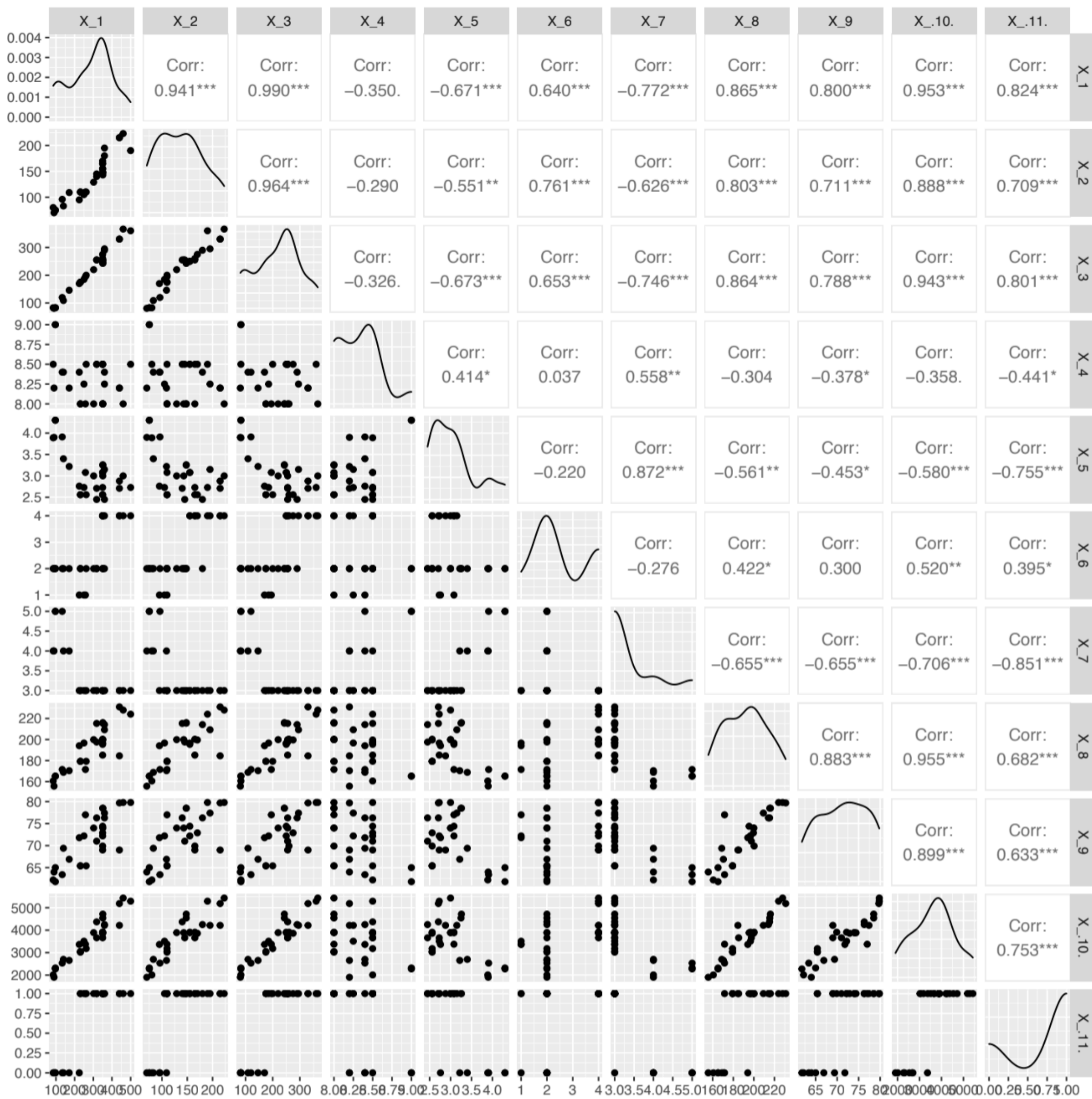
- V_6 indicates that collinearity exists between all variables except X_4
- V_5 indicates that collinearity exists between all variables except X_3
- V_4 indicates that collinearity exists between all variables except X_3

Problem 2: Ex 9.3

```
df = read.table('table9.17.txt',header=T)
```

Part a

```
library(ggplot2)
library(GGally)
dfx = df[,c(2:12)]
cmat = cor(dfx)
ggpairs(dfx)
```



The scatter plot shows evidence of strong collinearity between

- X_1 and X_2, X_3
- X_2 and X_3

Weak collinearity exists elsewhere as well.

Ignore the density plots on the diagonal

Part b

```
ev <- eigen(cmat)
evals <- ev$values
evals # eigenvalues
```

```
##  [1] 7.702574847 1.403077880 0.773435643 0.577055424 0.211498935 0.141941470
```

```
##  [7] 0.095142049 0.050092536 0.033266309 0.008417705 0.003497202
```

```r
evecs <- ev$vectors
evecs # eigenvectors
```

```
##              [,1]         [,2]        [,3]         [,4]         [,5]        [,6]
##  [1,] -0.3529639 -0.112431387  0.03114403 -0.006932422  0.026272973 -0.09512815
##  [2,] -0.3299718 -0.260762001  0.07836539 -0.194970349 -0.142783457 -0.23889898
##  [3,] -0.3510109 -0.139829772  0.04294522 -0.004153543 -0.084990459 -0.18488343
##  [4,]  0.1610427 -0.552726480  0.11863260  0.785849610  0.096920435  0.09122188
##  [5,]  0.2663779 -0.346997347 -0.43309789 -0.352178691  0.516283052  0.07200995
##  [6,] -0.2047881 -0.548146807  0.41844801 -0.380746710 -0.007176897  0.38287792
##  [7,]  0.3040550 -0.352222407 -0.22122179 -0.134117215 -0.050372348 -0.57691563
##  [8,] -0.3232988 -0.078466513 -0.36961713  0.180329365 -0.200485930 -0.20407455
##  [9,] -0.3026624  0.006019985 -0.54645511  0.094905101  0.106514020  0.51959464
## [10,] -0.3446125 -0.100475266 -0.26679114  0.040652506 -0.028959499 -0.14008874
## [11,] -0.3117090  0.181885175  0.24279993  0.119155548  0.800493659 -0.27479473
##              [,7]        [,8]        [,9]        [,10]        [,11]
##  [1,]  0.26787382 -0.25888638  0.49677393 -0.290946296  0.617904045
##  [2,]  0.34910433  0.05057424 -0.65243209  0.290811120  0.258528596
##  [3,]  0.35518667 -0.06800437  0.03290868 -0.466442937 -0.681570251
##  [4,]  0.09287761 -0.06188507 -0.06292276  0.051311641  0.012735988
##  [5,]  0.06450059 -0.43886854 -0.13804308 -0.086127357 -0.045372936
##  [6,] -0.37681067  0.16574908  0.13359309 -0.004651702 -0.059626414
##  [7,] -0.02079064  0.55944398  0.24949398 -0.055978181  0.049028663
##  [8,] -0.67496023 -0.15486222 -0.25287357 -0.294111256  0.091346835
##  [9,]  0.19659254  0.52415223 -0.01482782 -0.055178229  0.052597726
## [10,] -0.06284718 -0.20261712  0.39402290  0.714256660 -0.259679096
## [11,] -0.16382124  0.22167146 -0.06274209  0.017189710 -0.009773591
```

```r
kappas <- sqrt(evals[1]/evals)
kappas # condition indices
```

```
##  [1]  1.000000  2.343026  3.155774  3.653501  6.034814  7.366536  8.997704
##  [8] 12.400279 15.216531 30.249703 46.930759
```

```r
kappas[length(evals)] # condition number
```

```
## [1] 46.93076
```

```r
sum(kappas>15) # sets of collinearity
```

```
## [1] 3
```

Part c

λ_{11} and λ_{10} are small so we look at their eigenvectors.

- V_{11} indicates that collinearity exists between all variables except X_{11}
- V_{10} indicates that collinearity exists between all variables except X_6

Part d

```r
library(car)
model <- lm(Y~.+1,data=df)
vifvals = vif(model)
vifvals
```

```
##        X_1        X_2        X_3        X_4        X_5        X_6        X_7
## 128.834832  43.921063 160.436093   2.057834   7.780750   5.326714  11.735038
```

```
##        X_8        X_9     X_.10.     X_.11.
## 20.585810  9.419449  85.675755  5.142547
```

```
which(vifvals>10) # predictors effected by collinearity
```

```
##    X_1    X_2    X_3    X_7    X_8 X_.10.
##      1      2      3      7      8     10
```

Problem 3: Ex 10.2

```
evals = c(1.93,1.06,0.01) # eigenvalues
evec1 = c(0.5,0.484,0.718)
evec2 = c(-0.697,0.717,0.002)
evec3 = c(0.514,0.501,-0.696)
V = cbind(evec1,evec2,evec3) # eigenvector matrix
alpha = c(0.67,-0.02,-0.56)
```

Part a

Since $\bar{x}_j = 0$ for $j = 1, \dots, p$ we see that $\hat{\beta}_0 = \bar{y} - \bar{x}_1\hat{\beta}_1 - \dots \bar{x}_p\hat{\beta}_p = \bar{y}$.

Part b

```
kappas = sqrt(evals[1]/evals)
kappas
```

```
## [1]  1.000000  1.349353 13.892444
```

```
sum(kappas>15) # sets of collinearity
```

```
## [1] 0
```

Although none of the κ_j values are above 15, we suspect weak collinearity exists between all variables as indicated by the small λ_3 value and consistently large values across V_3.

Part c

$$R^2 = \frac{SSR}{SST} = \frac{SSR}{SSR + SSE}$$

```
SSR = 86.6542
SSE = 12.3458
R2 = SSR/(SSR+SSE)
R2
```

```
## [1] 0.8752949
```

Part d

If V_1 is the first eigenvector then $C_1 = ZV_1 = 0.5Z_1 + 0.484Z_2 + 0.718Z_3$.

Part e

Let $X_{n\times 3} = (X_1, X_2, X_3)$, $Z_{n\times 3} = (Z_1, Z_2, Z_3)$, $\theta_{3\times 1} = (\theta_1, \theta_2, \theta_3)^T$, $\alpha_{3\times 1} = (\alpha_1, \alpha_2, \alpha_3)^T$, and let V be a matrix whose j^{th} column is the j^{th} eigenvector. Since X is standardized, we can say $X = Z$. Also note that $\theta = V\alpha$. This implies that

$$\tilde{Y} = \frac{Y - \bar{y}}{s_y} = Z\theta = X\theta = XV\alpha$$

$$\therefore \hat{Y}_{PC} = \bar{y} + s_y XV\alpha.$$

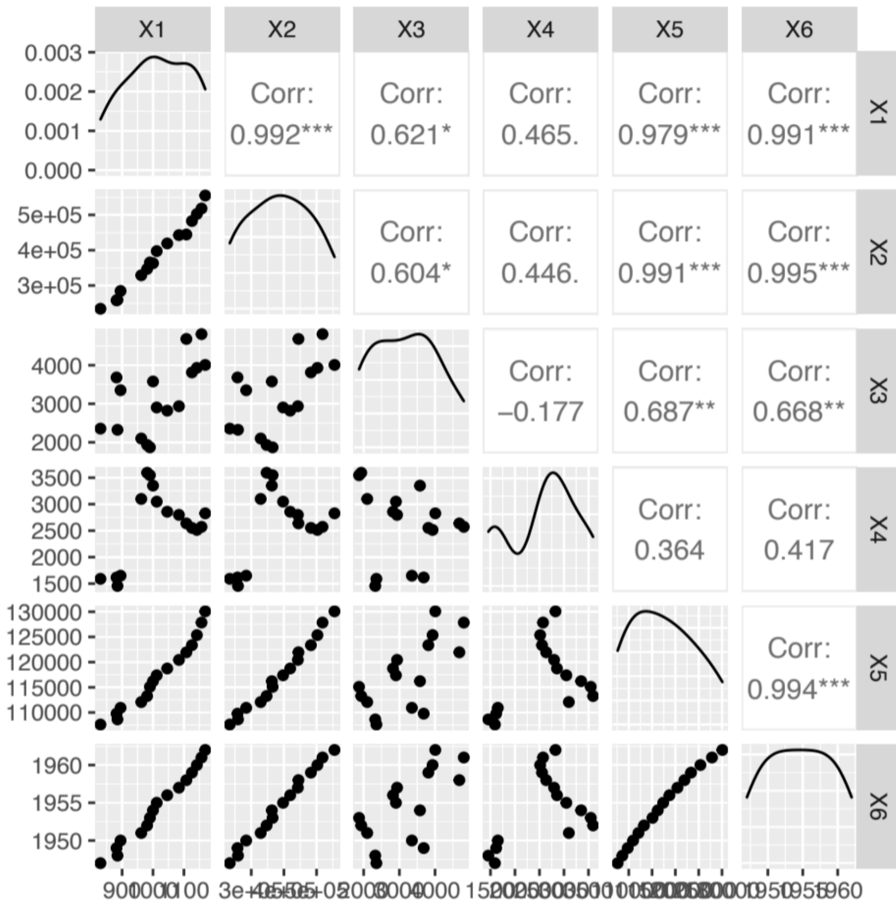
So $\beta_0 = \bar{y}$ and $(\beta_1, \beta_2, \beta_3)^T = s_y V\alpha$.

Problem 4

```r
df = read.table('table10.19.txt',header=T)
```

Part 1

```r
library(ggplot2)
library(GGally)
dfx = df[,c(2:7)]
cmat = cor(dfx)
ggpairs(dfx)
```



The scatter plot shows evidence of strong collinearity between

- X_1 and X_2, X_5, X_6
- X_2 and X_5, X_6
- X_5 and X_6

```r
ev <- eigen(cmat) # eigenvalues
evals <- ev$values
evals
```

```
## [1] 4.6033770958 1.1753404993 0.2034253724 0.0149282587 0.0025520658
## [6] 0.0003767081
```

```r
V <- ev$vectors # eigenvector matrix
V
```

```
##              [,1]          [,2]        [,3]          [,4]          [,5]         [,6]
```

```
## [1,] -0.4618349  0.0578427677  0.1491199  0.792873559 -0.337937826  0.13518707
## [2,] -0.4615043  0.0532122862  0.2776823 -0.121621225  0.149573192 -0.81848082
## [3,] -0.3213167 -0.5955137627 -0.7283057  0.007645795 -0.009231961 -0.10745268
## [4,] -0.2015097  0.7981925480 -0.5616075 -0.077254979 -0.024252472 -0.01797096
## [5,] -0.4622794 -0.0455444698  0.1959846 -0.589744965 -0.548578173  0.31157087
## [6,] -0.4649403  0.0006187884  0.1281157 -0.052286554  0.749542836  0.45040888
```

```
kappas <- sqrt(evals[1]/evals)
kappas # condition indices
```

```
## [1]   1.000000   1.979048   4.757028  17.560372  42.470986 110.544153
```

```
kappas[length(evals)] # condition number
```

```
## [1] 110.5442
```

```
sum(kappas>15) # sets of collinearity
```

```
## [1] 3
```

λ_6 and λ_5 are small so we look at their eigenvectors.

- V_6 indicates that collinearity exists between all variables except X_4
- V_5 indicates that collinearity exists between X_1, X_2, X_5, X_6
- V_4 indicates that collinearity exists between X_1, X_2, X_5

Part 2

The original model is

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_6 X_6 + \epsilon.$$

```
m.original = lm(Y~.+1,data=df)
m.original$coefficients
```

```
##   (Intercept)            X1            X2            X3            X4
## -3.482259e+06  1.506187e+00 -3.581918e-02 -2.020230e+00 -1.033227e+00
##            X5            X6
## -5.110411e-02  1.829151e+03
```

The standardized model is

$$\tilde{Y} = \theta_0 + \theta_1 Z_1 + \cdots + \theta_6 Z_6 + \epsilon'.$$

```
m.transformed = lm(scale(Y)~scale(X1)+scale(X2)+scale(X3)+scale(X4)+scale(X5)+scale(X6)-1,data=df)
m.transformed$coefficients
```

```
##   scale(X1)   scale(X2)   scale(X3)   scale(X4)   scale(X5)   scale(X6)
##  0.04628202 -1.01374635 -0.53754258 -0.20474069 -0.10122111  2.47966438
```

The θ_j coefficients can also be computed directly using

$$\hat{\theta}_j = \frac{s_j}{s_y}\hat{\beta}_j.$$

```
thetas = m.original$coefficients[2:7]*apply(dfx,2,sd)/sd(df$Y)
thetas
```

```
##          X1          X2          X3          X4          X5          X6
##  0.04628202 -1.01374635 -0.53754258 -0.20474069 -0.10122111  2.47966438
```

Part 3

$$C = ZV, \qquad \theta = V\alpha$$

```
dfx_scaled = scale(dfx)
C = as.data.frame(as.matrix(dfx_scaled)%*%V)
C$Y = scale(df$Y)
names(C) = c('C1','C2','C3','C4','C5','C6','Y_tilde')
pcr.full = lm(Y_tilde~.-1,data=C)
alpha = pcr.full$coefficients
theta = V%*%alpha
theta # theta estimates
```

```
##             [,1]
## [1,]  0.04628202
## [2,] -1.01374635
## [3,] -0.53754258
## [4,] -0.20474069
## [5,] -0.10122111
## [6,]  2.47966438
```

```
s = 'Y_tilde~C1'
p = length(theta)
theta_mat = matrix(nrow=p,ncol=(2+p))
i = 1
while(i<=p){
  pcr = lm(as.formula(paste(s,'-1')),data=C)
  alpha = as.matrix(pcr$coefficients)
  theta = V[,1:i]%*%alpha
  r2 = summary(pcr)$r.squared
  theta_mat[i,1] = i
  theta_mat[i,2] = r2
  theta_mat[i,3:(p+2)] = theta
  s = paste(s,sprintf('+C%d',i+1))
  i = i+1}
theta_mat = as.data.frame(theta_mat)
names(theta_mat) = c('ncomp','R^2',paste('theta',1:p,sep=''))
theta_mat
```

```
##   ncomp       R^2      theta1     theta2     theta3      theta4     theta5
## 1     1 0.9142532  0.20581723  0.2056699  0.1431951  0.08980304  0.2060153
## 2     2 0.9288835  0.21227070  0.2116068  0.0767541  0.17885680  0.2009339
## 3     3 0.9859670  0.29126362  0.3587028 -0.3090495 -0.11864220  0.3047524
## 4     4 0.9861215  0.37192875  0.3463293 -0.3082717 -0.12650194  0.2447531
## 5     5 0.9939980 -0.22176065  0.6090996 -0.3244904 -0.16910870 -0.7189894
## 6     6 0.9954790  0.04628202 -1.0137463 -0.5375426 -0.20474069 -0.1012211
##      theta6
## 1 0.2072011
## 2 0.2072702
## 3 0.2751366
## 4 0.2698171
## 5 1.5866145
## 6 2.4796644
```

Use C_1, C_2, C_3, and C_4 since R^2 is large enough and including C_5 changes the θ_j's significantly. Also, κ_5 and κ_6 are large, so adding C_5 would reintroduce collinearity.

Part 4

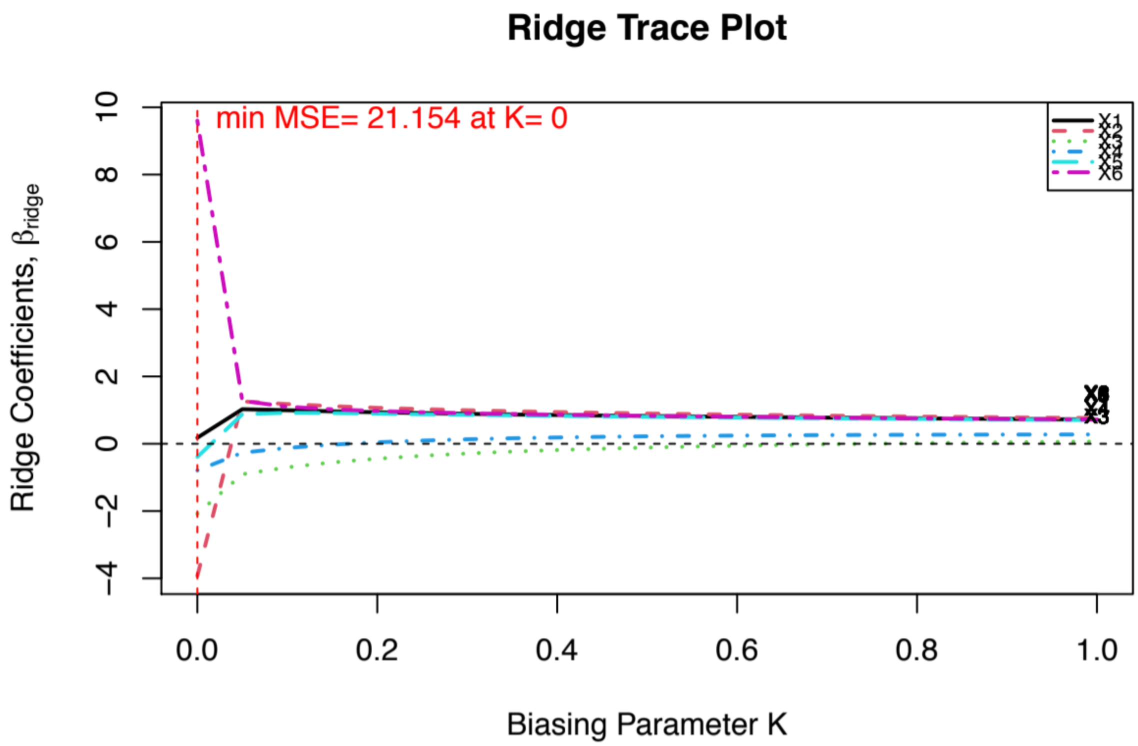
```
library(lmridge)
df_scaled = as.data.frame(apply(df,2,scale))
ridge <- lmridge(Y~.-1,data=df_scaled,K=seq(0,1,0.05),scaling='sc')
as.data.frame(coef(ridge)) # thetas
```

```
##              X1       X2       X3       X4       X5      X6
## K=0     0.04628 -1.01375 -0.53754 -0.20474 -0.10122 2.47966
## K=0.05 0.26492  0.32501 -0.23396 -0.06869  0.22629 0.32892
## K=0.1  0.25706  0.30423 -0.18084 -0.03170  0.23710 0.28379
## K=0.15 0.24855  0.28832 -0.14304 -0.00680  0.23476 0.26357
## K=0.2  0.24122  0.27575 -0.11464  0.01101  0.23030 0.25075
## K=0.25 0.23493  0.26548 -0.09250  0.02421  0.22560 0.24138
## K=0.3  0.22944  0.25685 -0.07476  0.03426  0.22112 0.23398
## K=0.35 0.22457  0.24943 -0.06024  0.04206  0.21694 0.22783
## K=0.4  0.22020  0.24294 -0.04816  0.04819  0.21307 0.22256
## K=0.45 0.21621  0.23717 -0.03796  0.05306  0.20947 0.21793
## K=0.5  0.21255  0.23198 -0.02924  0.05698  0.20612 0.21378
## K=0.55 0.20915  0.22726 -0.02172  0.06013  0.20298 0.21001
## K=0.6  0.20597  0.22292 -0.01517  0.06269  0.20003 0.20655
## K=0.65 0.20298  0.21892 -0.00943  0.06476  0.19724 0.20333
## K=0.7  0.20015  0.21519 -0.00438  0.06644  0.19459 0.20033
## K=0.75 0.19747  0.21170  0.00011  0.06779  0.19207 0.19750
## K=0.8  0.19491  0.20842  0.00411  0.06889  0.18966 0.19483
## K=0.85 0.19247  0.20532  0.00769  0.06976  0.18736 0.19230
## K=0.9  0.19013  0.20238  0.01090  0.07044  0.18514 0.18988
## K=0.95 0.18787  0.19958  0.01380  0.07097  0.18301 0.18756
## K=1    0.18570  0.19691  0.01642  0.07137  0.18095 0.18534
```

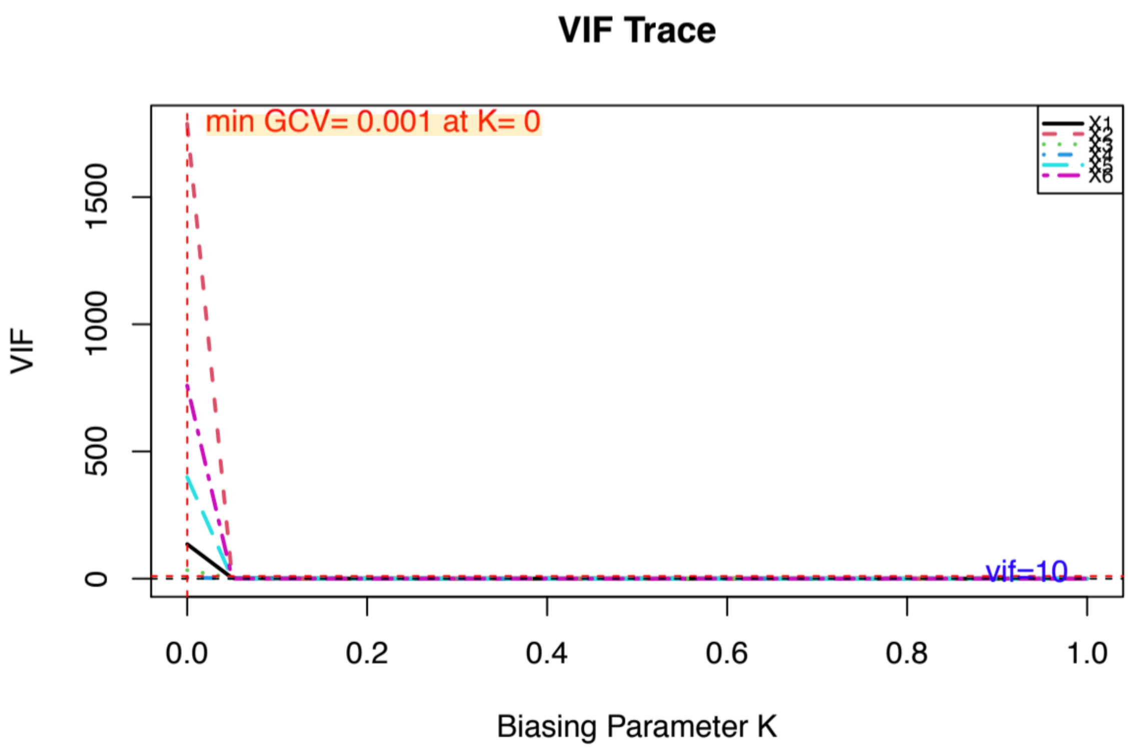
```
as.data.frame(vif(ridge)) # vifs
```

```
##                X1         X2       X3      X4        X5        X6
## k=0    135.53244 1788.51348 33.61889 3.58893 399.15102 758.98060
## k=0.05   2.45277    0.46422  1.98165 1.52810   1.69282   0.65690
## k=0.1    0.83483    0.26393  1.45027 1.17264   0.60058   0.22826
## k=0.15   0.43974    0.19259  1.12241 0.95159   0.33297   0.13726
## k=0.2    0.28286    0.15309  0.90408 0.80017   0.22396   0.10138
## k=0.25   0.20401    0.12762  0.75026 0.68989   0.16757   0.08277
## k=0.3    0.15838    0.10985  0.63708 0.60588   0.13401   0.07149
## k=0.35   0.12935    0.09678  0.55086 0.53963   0.11212   0.06392
## k=0.4    0.10958    0.08679  0.48331 0.48594   0.09686   0.05848
## k=0.45   0.09540    0.07894  0.42914 0.44147   0.08570   0.05435
## k=0.5    0.08482    0.07260  0.38485 0.40398   0.07720   0.05109
## k=0.55   0.07664    0.06738  0.34802 0.37191   0.07053   0.04843
## k=0.6    0.07016    0.06301  0.31698 0.34413   0.06516   0.04620
## k=0.65   0.06489    0.05929  0.29048 0.31982   0.06074   0.04428
## k=0.7    0.06054    0.05609  0.26763 0.29836   0.05703   0.04261
## k=0.75   0.05686    0.05329  0.24774 0.27927   0.05387   0.04113
## k=0.8    0.05372    0.05083  0.23028 0.26217   0.05115   0.03981
## k=0.85   0.05101    0.04864  0.21484 0.24678   0.04876   0.03860
## k=0.9    0.04862    0.04667  0.20110 0.23284   0.04665   0.03750
## k=0.95   0.04652    0.04490  0.18880 0.22016   0.04477   0.03649
## k=1      0.04463    0.04328  0.17773 0.20859   0.04308   0.03554
```

```
plot(ridge,type="ridge",ylab='theta') # the y axis label should be theta instead of beta
```




```
plot(ridge,type='vif')
```



The θ_j's appear to converges for k somewhere between 0.25 and 0.3. The VIF's also appear to stabely between 1 and 10 when k somewhere between 0.25 and 0.3.