# HW5

John Rothen

10/3/2020

## Problem 3.5.4

### Part 1

For ease of typing, lambda is abbreviated L.

$B_j^{\text{`}} = argmin(.5(B_j^{LS} - B_j)^2 + L||B_j||_1$.

For the case of z> L, $argmin(z^2 + Lambda(b))$, which is minimized at Bj, causing the result to be z- Lambda. as for the case of z< -L, this is minimized to z + Lambda in a similar fashion. The case where lambda is greater than or equal to z causes the minimum to occur at zero.

### Part 2

Consider the condition of the soft-thresholding operator $S(z; L) = 0$ if $|z| <= L$. This means, that Lmax would be the greatest Lambda such that $(L < |z|)$.

### Part 3

```
soft.thr <- function(z,L){
  #Lambda is given as L here for ease of use
  if(z > L){out = z-L}
  if(abs(z) <= L){out = 0}
  if(z < (-1*L)){out = z+L}
  out
}
```

### Part 4

```
pen_ls <- function(y,xmat,lambda){
  n <- length(y)
  df <- as.data.frame(cbind(y,xmat)) #create dataframe
  lm.temp <- lm(y~., data=df) #generate LM
  betas <- lm.temp$coefficients[2:11] #obtain Beta LS
  bhats <- rep(0,10) #prepare empty vector
  for(i in 1:length(betas)){
    bhats[i]<- soft.thr(betas[i], lambda) #calculate new estimates
  }
  bhats
}
```

## Part 5

```r
#Data generation
gen_data <- function(n, b, sigma = 1) {
  p <- length(b)
  x <- matrix(rnorm(n * p), n, p)
  colnames(x) <- paste0("x", 1:p)
  y <- c(x %*% b) + rnorm(n, sd = sigma)
  return(data.frame(y, x))
}

n <- 200
b <- c(1, -1, 2, rep(0, 7))
set.seed(920)
dat <- gen_data(n, b)

#Test

#recombine the x matrix
xmat <- as.matrix(cbind(dat$x1,dat$x2,dat$x3,dat$x4,dat$x5,dat$x6,dat$x7,dat$x8,dat$x9,dat$x10))

# relabel y for ease of use
y <- dat$y

#pick random small lambda
lambda = .001

#Test
test<-pen_ls(y, xmat, lambda)
# extract the original betas for reference
n <- length(y)
df <- as.data.frame(cbind(y,xmat)) #create dataframe
lm.temp <- lm(y~., data=df) #generate LM
betas <- lm.temp$coefficients[2:11] #obtain Beta LS

#present results
data.frame(test,betas)
```

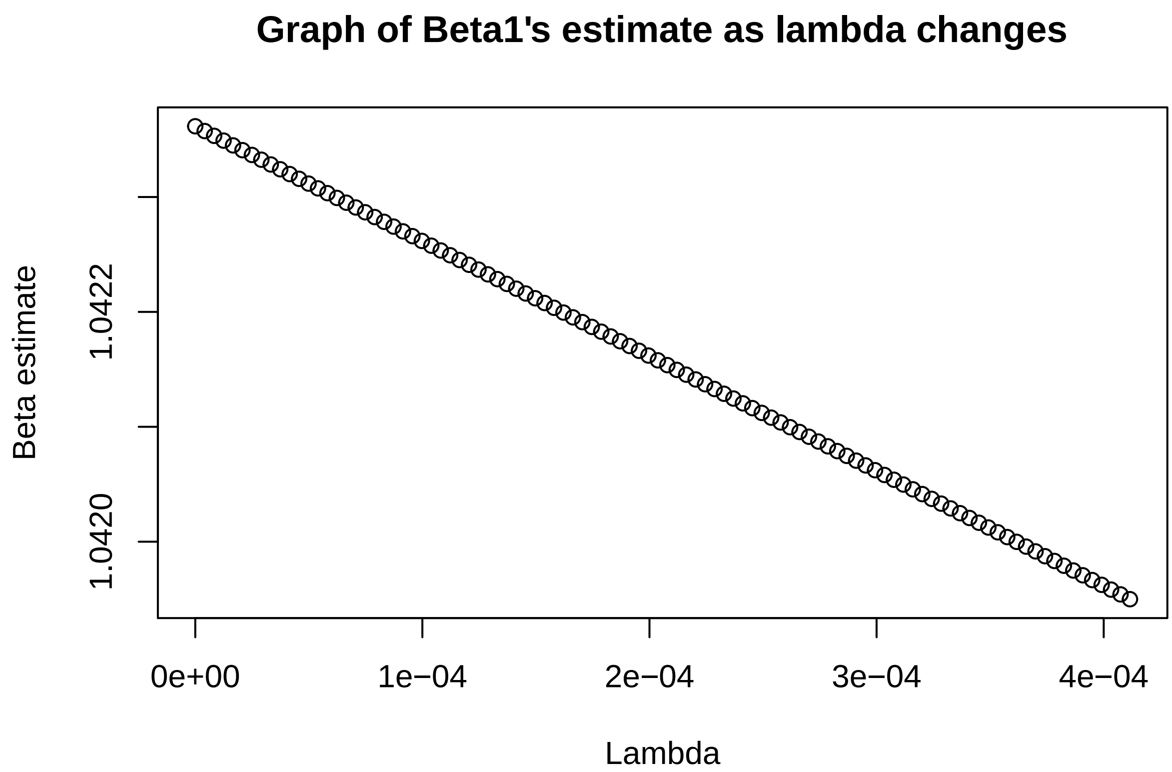
```
##             test         betas
## V2   1.041361554  1.0423615540
## V3  -0.941310825 -0.9423108248
## V4   2.054084058  2.0550840584
## V5  -0.005426430 -0.0064264302
## V6   0.000000000 -0.0004115676
## V7   0.004267272  0.0052672720
## V8  -0.042339507 -0.0433395071
## V9   0.089269447  0.0902694472
## V10  0.097624057  0.0986240566
## V11  0.173302597  0.1743025971
```

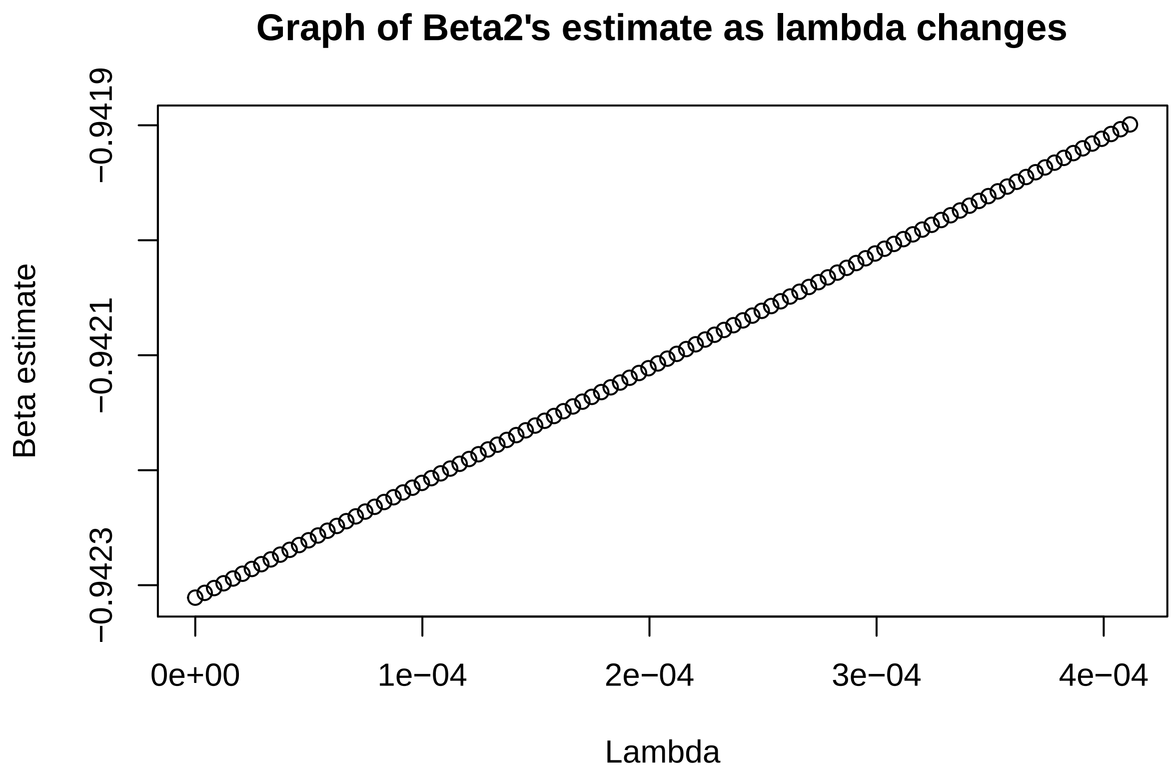
## Part 6

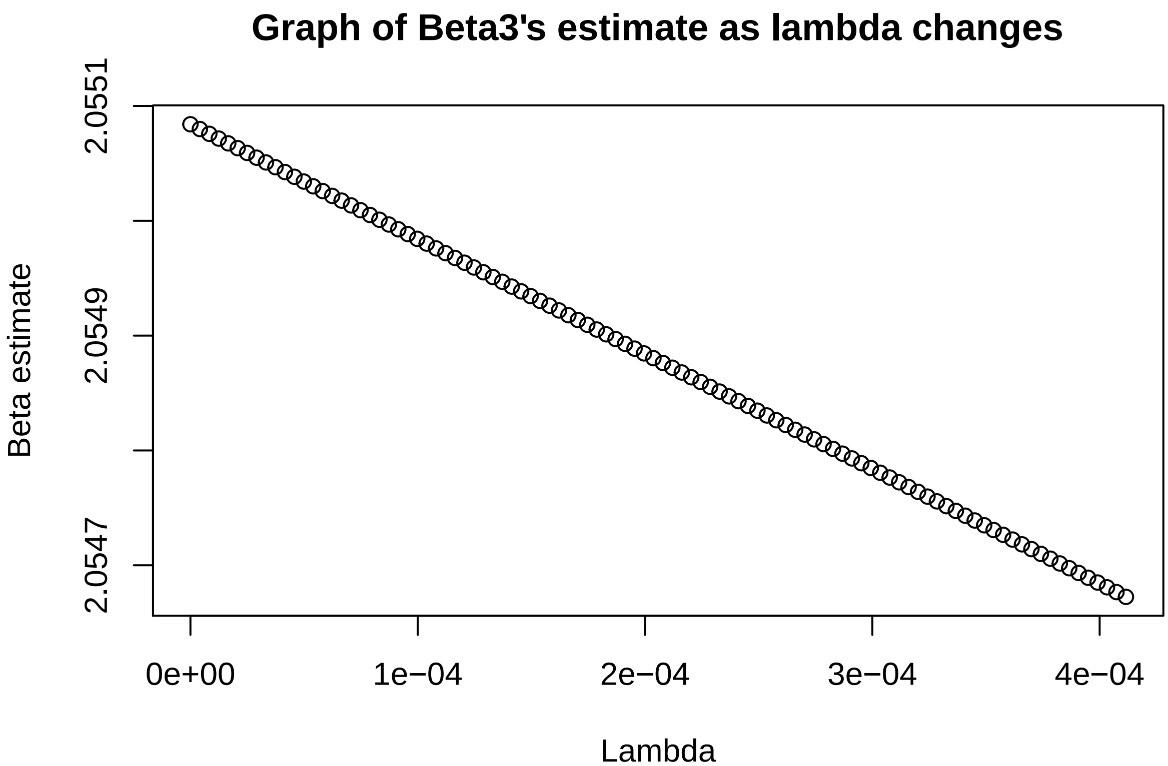
Using the conclusion from part 2, we know that Lmax will be the largest number smaller than $|B_i^{LS}|$. The smallest beta here is beta5, $-4.1156759 \times 10^{-4}$. This means that Lambda must be less than $4.1156759 \times 10^{-4}$.

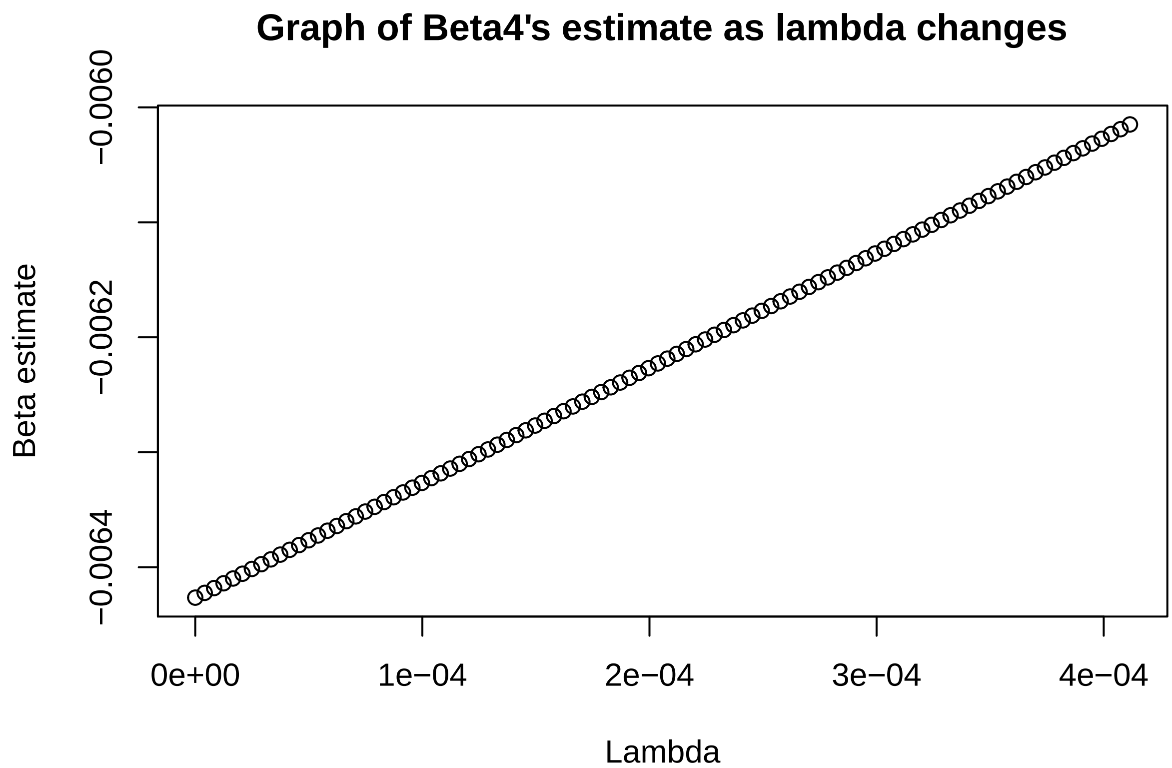
We will let L max be this value minus .0000000001, (although to be more precise, one would choose Lmax to be $4.1156759 \times 10^{-4}$ - smallest representable number in double).

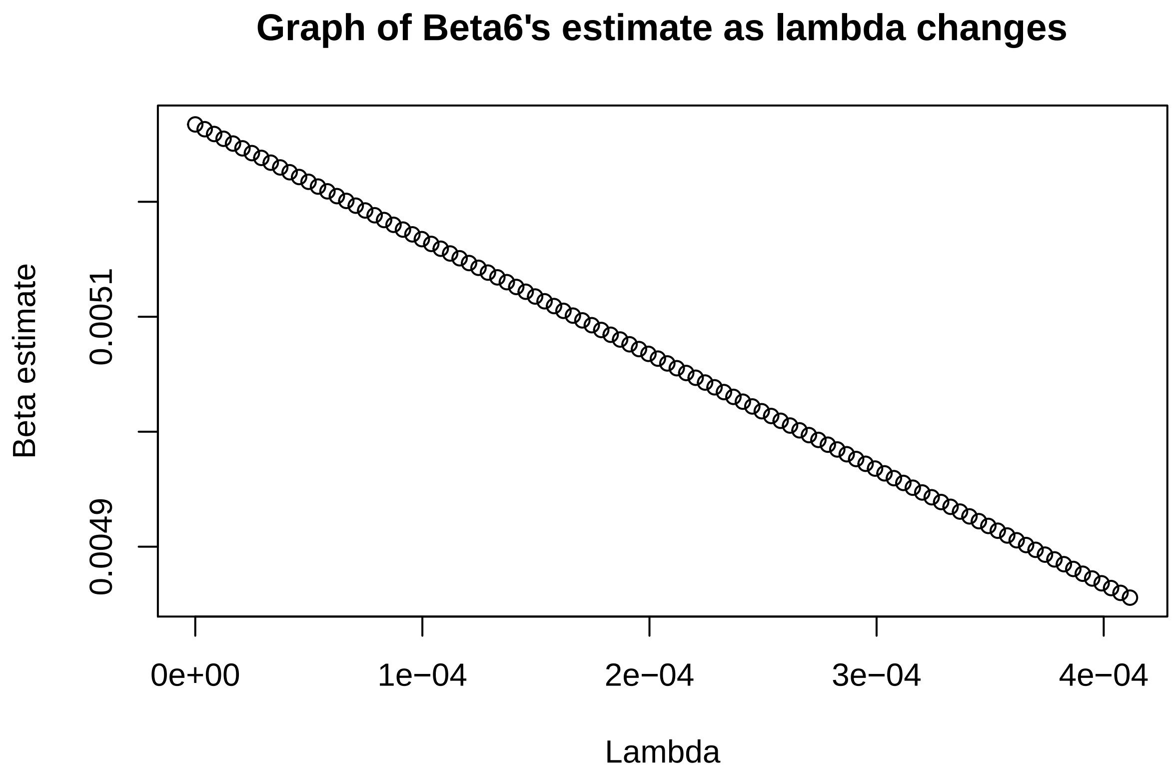
```
#Plot
lmax = abs(betas[5])-.0000000001
seq.l <- seq(0, lmax, length.out=100)
t <- rep(0,100)



for(i in 1:length(seq.l)){
  temp <- seq.l[i]
  t[i] <- pen_ls(y, xmat, temp)[1]
}
plot(seq.l, t, xlab = "Lambda", ylab= "Beta estimate", main= "Graph of Beta1's estimate as lambda change
```

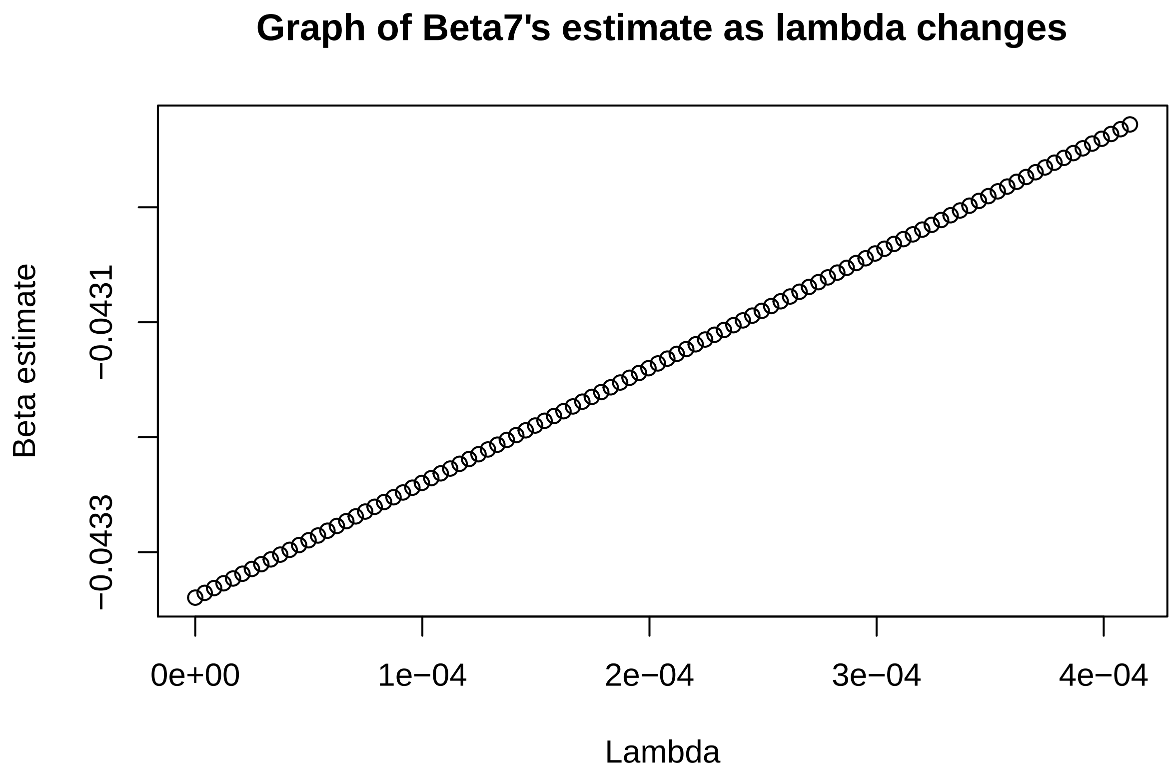


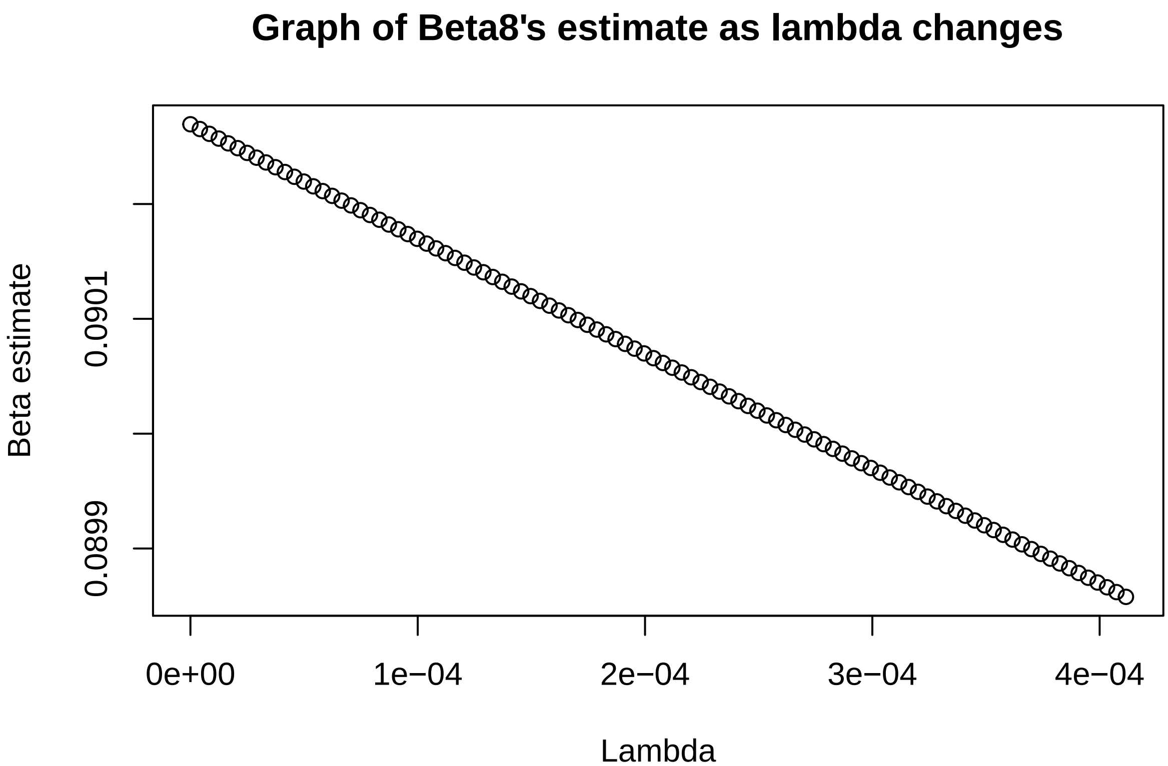

Graph of Beta2's estimate as lambda changes
Beta estimate
−0.9419
−0.9421
−0.9423
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda

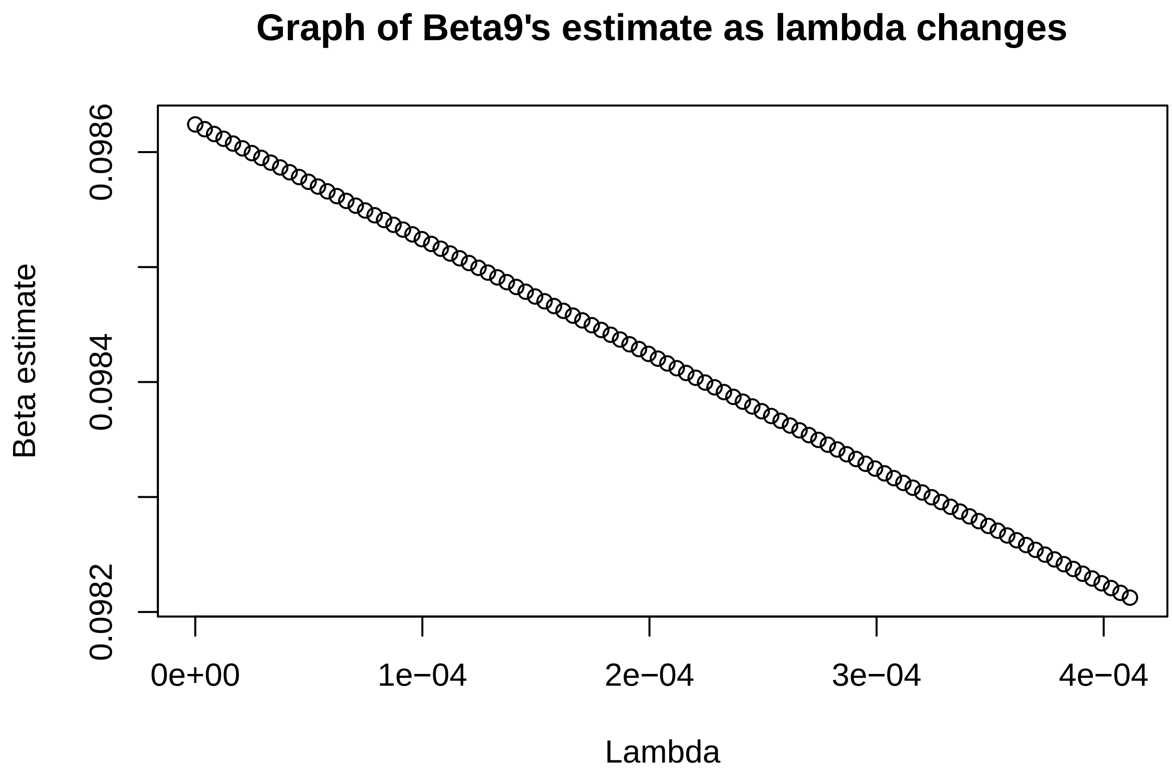

Graph of Beta3's estimate as lambda changes
Beta estimate
2.0551
2.0549
2.0547
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda


Graph of Beta4's estimate as lambda changes
Beta estimate
−0.0060
−0.0062
−0.0064
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda


Graph of Beta6's estimate as lambda changes
Beta estimate
0.0049
0.0051
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda


Graph of Beta7's estimate as lambda changes
Beta estimate
−0.0431
−0.0433
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda


Graph of Beta8's estimate as lambda changes
Beta estimate
0.0901
0.0899
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda


Graph of Beta9's estimate as lambda changes
Beta estimate
0.0986
0.0984
0.0982
0e+00
1e−04
2e−04
3e−04
4e−04
Lambda