Oct19

2022-10-19

PLS closed-form solution

1. Deriving the closed form solution for PLS with MSE loss and L2 penalty on consecutive smoothing R_t. Starting from the objectives in the matrix form.

$\|I - RW\|_2^2 + \lambda \|R_{t+1} - R_t\|_2^2$

$=> \|I - RW\|_2^2 + \lambda \|DR\|_2^2$

where $D \in R^{(d-1)*d}$ is the difference matrix, with all 1's on the diagonal, and -1 on the off-diagonal above.

where W is a square matrix with diagonol being the vector $w_t = \sum_{a=1}^{t} I_{t-a} s_a$

$=> W^T R^T RW - 2W^T R^T I + I^T I + \lambda R^T D^T DR$

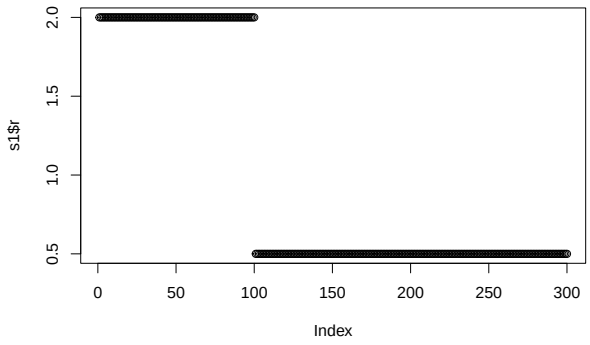
Taking derivative and equate to 0

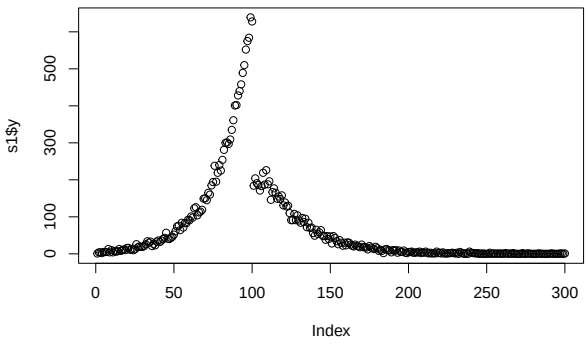
$=> 2W^T R^T W - 2W^T I + 2\lambda R^T D^T D = 0$

$=> R^T (W^T W + \lambda D^T D) = W^T I$

$=> R^T = (W^T W + \lambda D^T D)^{-1} W^T I$

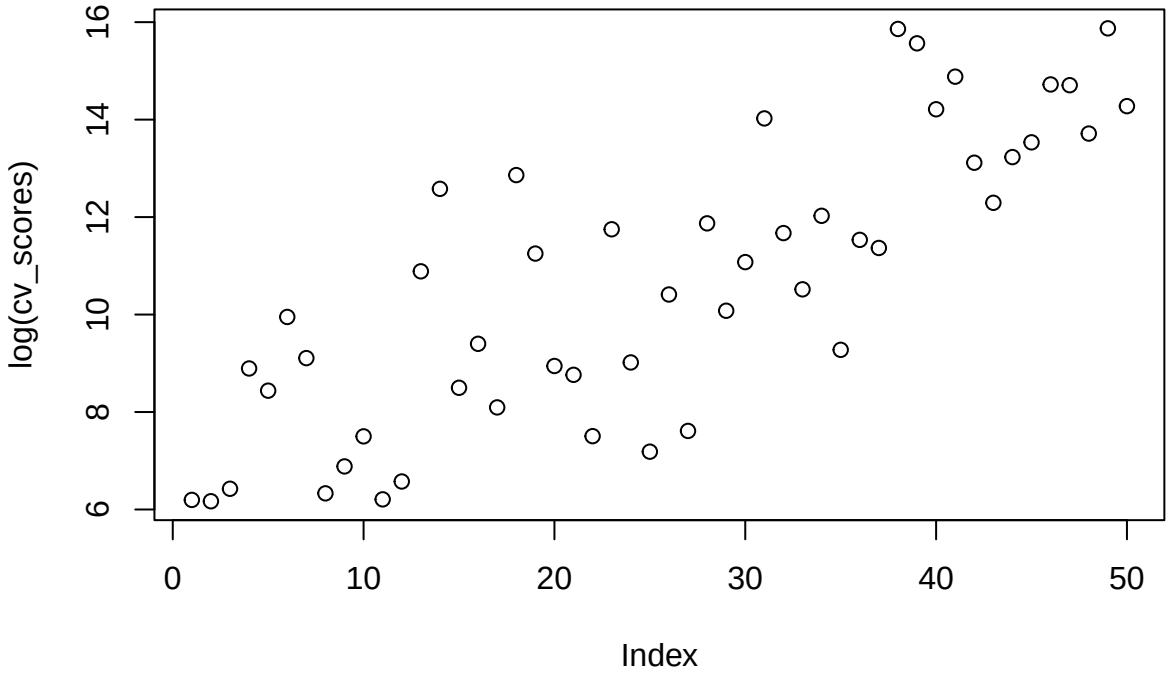
```
### Import s1
s1 <- read.csv("../data/processed/a.csv")

### Plotting
plot(s1$r)
plot(s1$y)
```



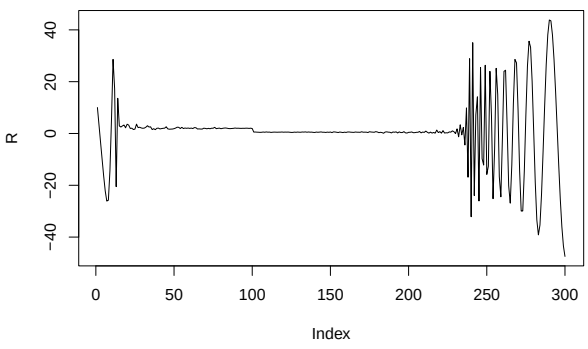


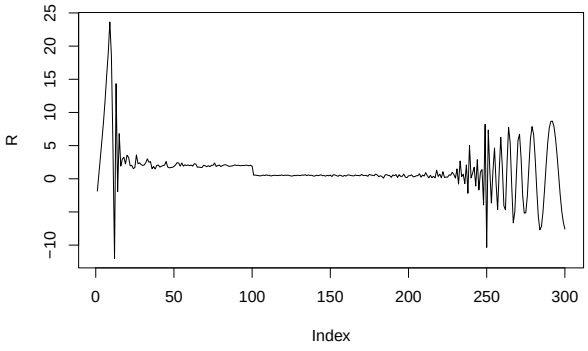
```r
r1_length <- nrow(s1)

### Construct difference matrix
D = diag(r1_length)
D[row(D) == col(D)-1] = -1
D = D[1:(nrow(D)-1),]


### Construct the Iwt vector
source("../function/get_iwt.R")
source("../function/disc_gamma.R")
source("../constant/constant.R")

s1_iwt <- get_iwt(s1$y, disc_gamma(1:nrow(s1), sid_ebola_shape, sid_ebola_scale))
W <- diag(s1_iwt)
Y <- s1$y

# R = solve((t(W)%*%W-lambda*t(D)%*%D))%*%t(W)%*%Y

get_r = function(Y,W,D, lambda=10) solve((t(W)%*%W-lambda*t(D)%*%D))%*%t(W)%*%Y

get_obj = function(Y,W,R,D) sum(Y-W%*%R)^2 + sum(lambda*(D%*%R)^2)
```

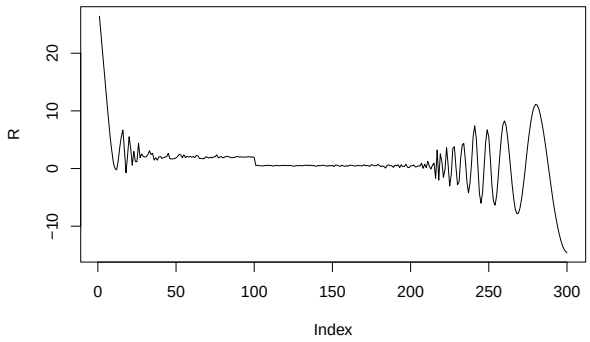
```r
### Function: Cross validation on ridge regression
CV <- function(W, Y, lambdas = exp(seq(0.1,10,0.2))){
  ### Length of data
  dat_length = length(Y)

  ### Get grid of lambda
  cv_scores = c()
  best = 0

  I = diag(rep(1, dat_length))

  for (lambda in lambdas){
    L = solve((t(W)%*%W-lambda*t(D)%*%D))%*%t(W)
    H = I-t(L)%*%W
    H_tilde = diag(diag(H))
    HHY = H%*%solve(H_tilde)%*%Y
    E_cv = 1/(dat_length)*sum(HHY^2)
    cv_scores = c(cv_scores, E_cv)
  }

  return(cv_scores)

}

lambdas = exp(seq(0.1,10,0.2))

cv_scores = CV(W, Y, lambdas = lambdas)
plot(log(cv_scores))
```

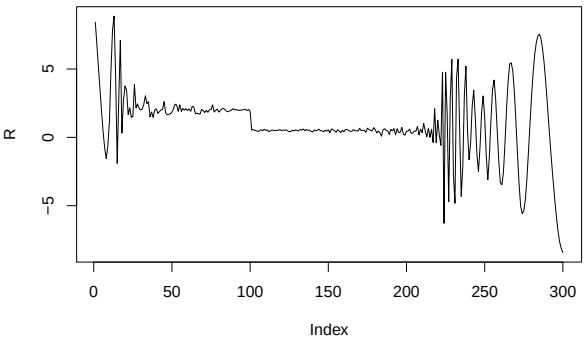


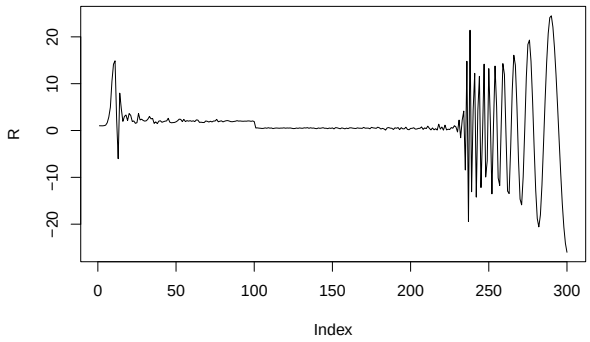

```
ordered = order(cv_scores)
lambdas = lambdas[ordered]

### Plot the best 5
for (optim_lambda in lambdas[1:5]){
  R = get_r(Y, W, D, optim_lambda)
  plot(R, type="l")
}
```











Smoothness at head and tail

Start and end of the synthetic datasets have low case counts, therefore prediction not accurate. Model should take this into consideration, but should not put too much effort in trying to smoothing them.

Should derive model that accommodate regions with low case counts.