Solution to Homework 8

1 Problem 1

1.1 Risk of the MLE

Given $\theta_i = \frac{1}{i^2}$ and $Z_i \sim N(\theta_i, 1)$. The risk of the MLE, $\hat{\theta}_i = Z_i$ is given by

$$R(Z^n, \theta^n) = \sum_{i=1}^{n} \mathbb{E}_\theta \left(Z_i - \theta_i\right)^2 = \sum_{i=1}^{n} 1 = n$$

1.2 Risk of the Linear Estimator

For simplicity, let $Z_i = \theta_i + \epsilon_i$, where $\theta_i = \frac{1}{i}$ and $\epsilon_i \sim N(0,1)$. Then by definition of risk function, we calculate the square risk as follows.

$$\begin{aligned}
R(\hat{\theta}, \theta) &= R(bZ, \theta) \\
&= \sum_{i=1}^{1000} \mathbb{E}_\theta \left[(bZ_i - \theta_i)^2\right] \\
&= \sum_{i=1}^{1000} \mathbb{E}_\theta \left[(b(\theta_i + \epsilon_i) - \theta_i)^2\right] \\
&= \sum_{i=1}^{1000} \mathbb{E}_\theta \left[(b-1)^2\theta_i^2 + 2b(b-1)\theta_i\epsilon_i + b^2\epsilon_i^2\right] \\
&= (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + 2b(b-1)\sum_{i=1}^{1000} \frac{1}{i^2} \underbrace{\mathbb{E}\left[\epsilon_i\right]}_{=0} + b^2 \sum_{i=1}^{1000} \left(\underbrace{\mathbb{E}\left[\epsilon_i\right]^2}_{=0} + \underbrace{Var\left(\epsilon_i\right)}_{=1} \right) \\
&= (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + 1000b^2
\end{aligned}$$

This result is consistent with what we get from Stein's theorem. By example 7.20, we know that for $\hat{\theta}_i = bZ_i$,

$$\hat{R}(Z) = (2b-1)n\sigma^2 + (1-b)^2 \sum_{i=1}^{n} Z_i^2$$

Here for $n = 1000$, $\theta_i = \frac{1}{i^2}$ and $\sigma^2 = 1$,

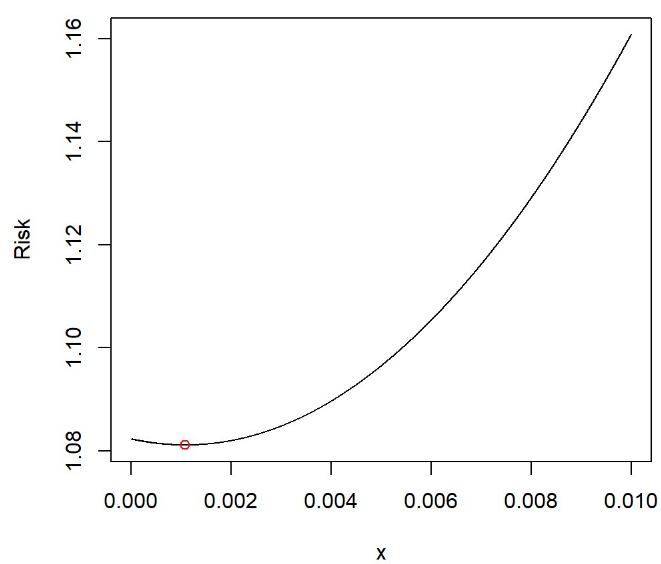
$$\hat{R}(Z) = (2b - 1) \cdot 1000 + (1 - b)^2 \sum_{i=1}^{1000} Z_i^2$$

Take expectation, we can show that it is still an unbiased estimator for $R(bZ, \theta)$:

$$\begin{aligned}
\mathbb{E}\left[\hat{R}(Z)\right] &= \mathbb{E}\left[(2b-1)n + (1-b)^2 \sum_{i=1}^{1000} Z_i^2\right] \\
&= (2b-1)n + (1-b)^2 \sum_{i=1}^{1000} \left(\mathbb{E}\left[Z_i\right]^2 + Var\left(Z_i\right)\right) \\
&= (2b-1)n + (1-b)^2 \sum_{i=1}^{1000} \left(\frac{1}{i^4} + 1\right) \\
&= (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + (2b-1)n + n(1-b)^2 \\
&= (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + nb^2 = (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + 1000b^2
\end{aligned}$$

1.3 Plot Risk w.r.t. b

The plot $R(bZ, \theta) = (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + 1000b^2$ can be found as follows.



1.4 Optimal b

Take derivative w.r.t. b and set it to zero:

$$\frac{\partial R(bZ,\theta)}{\partial b} = \frac{\partial (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + 1000b^2}{\partial b} = (2b-2)\sum_{i=1}^{1000}\frac{1}{i^4} + 2000b = 0$$

Then,

$$b_* = \frac{\sum_{i=1}^{1000} \frac{1}{i^4}}{1000 + \sum_{i=1}^{1000} \frac{1}{i^4}} \approx 0.001081153$$

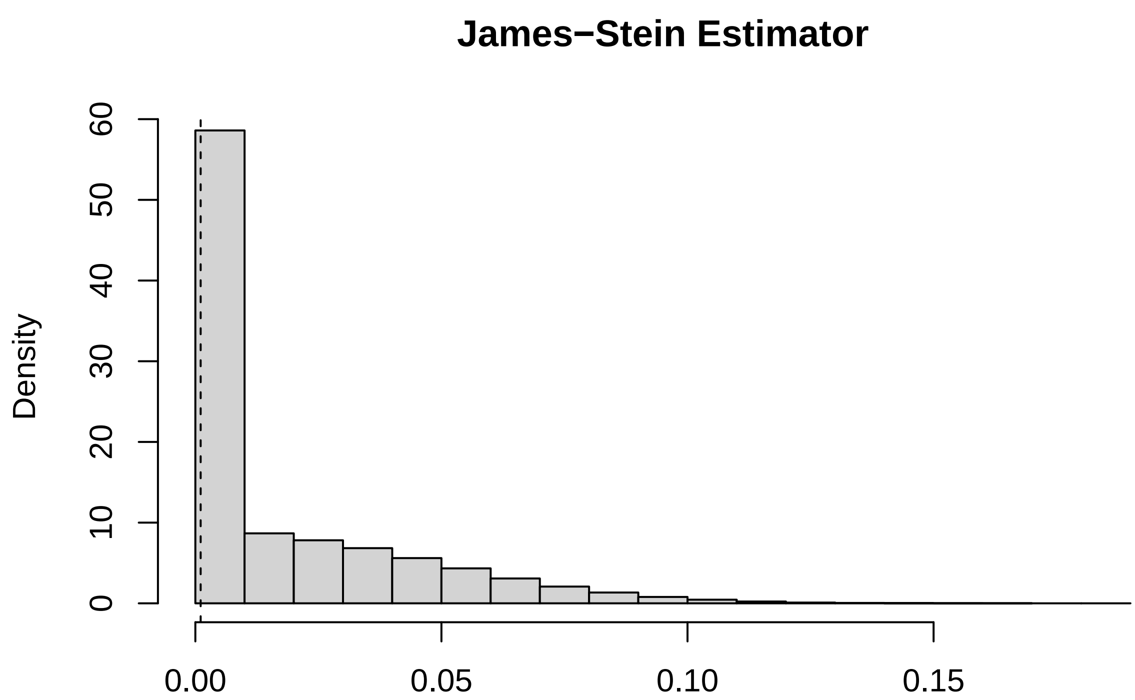
1.5 Simulation

θ^n is defined as follows.

```
n <- 1000
i <- 1:n
theta <- i^(-2)
```

We then do simulation for 100,000 times. However, we need to note that James-Stein estimator is not unbiased.

```
bhat_record <- c()
for (t in 1:100000){
  Z <- theta + rnorm(n)
  bhat <- max(0, 1 - n/sum(Z^2))
  bhat_record[t] <- bhat
}
hist(bhat_record, breaks = 20, probability = T, main = "James-Stein Estimator", xlab = "")
abline(v = 0.001081153, lty = 2)
```



For the risk of the MLE, it has been derived previously, which is $n = 1000$. For the risk of the linear estimator, it has also been calculated:

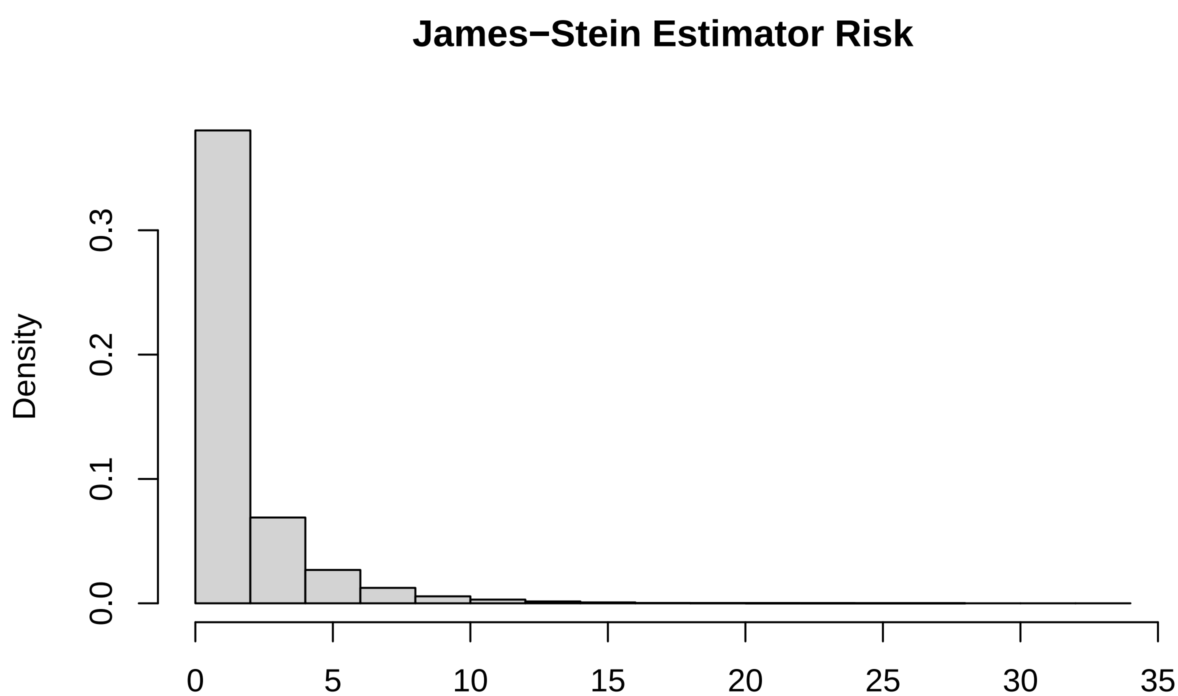
$$R(bZ, \theta) = (b-1)^2 \sum_{i=1}^{1000} \frac{1}{i^4} + 1000b^2$$

To begin with, we compare the risk of James-Stein estimator with that of MLE. Then we relationship between the risk of these two estimators with Pinsker bound.

```r
i <- 1:1000
sum_quartic <- sum(i^(-4))
risk_record <- sum_quartic * (bhat_record - 1)^2 + 1000 * bhat_record^2
summary(risk_record)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.081   1.082   1.082   1.966   1.911  33.143
```

```r
hist(risk_record, breaks = 20, probability = T, main = "James-Stein Estimator Risk", xlab = "")
```



Therefore, James-Stein estimator achieves much lower risk than MLE. For Pinsker bound, we have $\sigma^2 = n\sigma_n^2 = 1000$. If

$$\Theta_n(c) = \left\{(\theta_1, \cdots, \theta_n) : \sum_{i=1}^{n} \theta_i^2 \leqslant c^2\right\}$$

then the Pinsker bound is given by

$$\frac{\sigma^2 c^2}{\sigma^2 + c^2} = \frac{1000c^2}{1000 + c^2}$$

The next question is how to choose c^2. We have calculated that

$$\sum_{i=1}^{1000} \frac{1}{i^4} \approx 1.082323$$

We should also note that the limit of the infinite series can be derived by simple calculus (Riemann series theorem).

$$\lim_{n\to\infty} \sum_{i=1}^{n} \frac{1}{i^4} = \frac{\pi^4}{90} \approx 1.082323$$

Let $c^2 = \frac{\pi^4}{90}$, the Pinsker bound is then given by

```
c2 <- pi^4 / 90
Pinsker <- (1000 * c2) / (1000 + c2)
print(paste("The Pinsker bound is", Pinsker))
```

```
## [1] "The Pinsker bound is 1.08115307661712"
```

```
cat(paste("The quartiles of James-Stein estimator is", "\n", paste(quantile(risk_record), collapse = ",
```

```
## The quartiles of James-Stein estimator is
##  1.08115307628597,1.0823232333783,1.0823232333783,1.91100148461158,33.1432085672814
```

Therefore, the minimum simulated risk of James-Stein estimator almost achieves Pinsker bound.

Problem 2

(a)

According to the SURE formula,

$$\hat{R}(z) = n\sigma_n^2 + 2\sigma_n^2 \sum_{i=1}^{n} D_i + \sum_{i=1}^{n} g_i^2$$

where

$$g(Z_1, ..., Z_n) = \hat{\theta}^n - Z^n = (\hat{\theta}_1 - Z_1, \cdots, \hat{\theta}_n - Z_n)$$

and

$$\hat{\theta}_i = \begin{cases} -(Z_i + \lambda)^2, & Z_i < -\lambda \\ 0, & -\lambda \leqslant Z_i \leqslant \lambda \\ (Z_i - \lambda)^2, & Z_i > \lambda \end{cases}$$

Then because D_i is specified as the partial derivative with respect to z_i,

$$D_i = \frac{\partial g}{\partial z_i} = \begin{cases} -2(z_i + \lambda) - 1, & z_i < -\lambda \\ -1, & -\lambda \leqslant z_i \leqslant \lambda \\ 2(z_i - \lambda) - 1, & z_i > \lambda \end{cases}$$

Combine everything together, the SURE risk is given by

$$\hat{R}(z) = n\sigma_n^2 + 2\sigma_n^2 \sum_{i=1}^{n} D_i + \sum_{i=1}^{n} (\hat{\theta}_i - Z_i)^2$$

where D_i and g_i have been derived above. Finally because $Z_i \sim N(\theta_i, \sigma_n^2)$, the expectation of $\hat{R}(z) = R$ can be easily calculated through definition of condition expectation by calculus. However, the result of $R = \mathbb{E}[\hat{R}(z)]$ does not have explicit solution. Here I only give a sketch version of $\mathbb{E}[D_i]$ to show the logic.

$$\begin{aligned} \mathbb{E}[D_i] = &\mathbb{E}[D_i | Z_i < -\lambda] P(Z_i < -\lambda) + \\ &\mathbb{E}[D_i | -\lambda \leqslant Z_i \leqslant \lambda] P(-\lambda \leqslant Z_i \leqslant \lambda) + \\ &\mathbb{E}[D_i | Z_i > \lambda] P(Z_i > \lambda) \end{aligned}$$

where,

$$\begin{aligned} \mathbb{E}[D_i | Z_i < -\lambda] P(Z_i < -\lambda) &= P(Z_i < -\lambda) \int_0^\infty x f_{D_i | Z_i < -\lambda}(x) dx \\ &= P(Z_i < -\lambda) \int_0^\infty x \frac{f_{D_i}(x)}{P(Z_i < -\lambda)} dx \\ &= \int_0^\infty x f_{D_i}(x) dx \end{aligned}$$

Plug in everything and by numeric methods, we can calculate the true risk of curved soft threshold estimator.

(b)

Problem (1)

The set-up of problem 1 is $Z_i \sim N(\theta_i, 1)$. For each observation/sample generated from this normal means model, we can find an optimal λ by minimizing SURE estimator as follows.

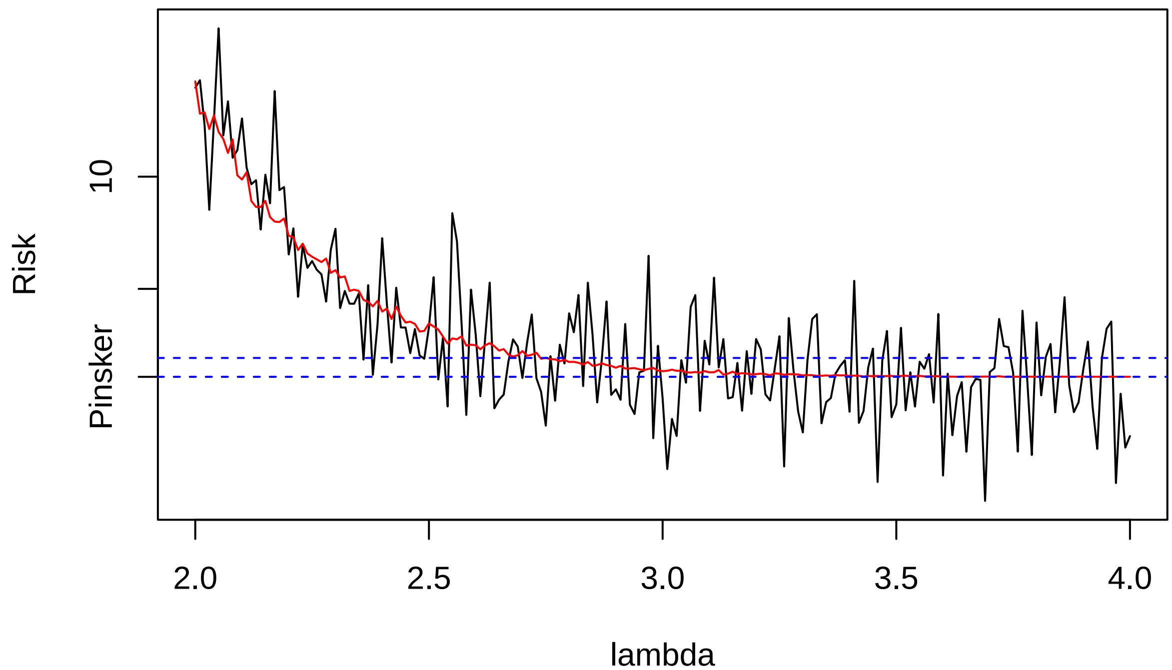
$$\hat{R}(z) = n\sigma_n^2 + 2\sigma_n^2 \sum_{i=1}^{n} D_i + \sum_{i=1}^{n} (\hat{\theta}_i - Z_i)^2 = 1000 + 2 \sum_{i=1}^{n} D_i + \sum_{i=1}^{n} (\hat{\theta}_i - Z_i)^2$$

To begin with, we first define 2 functions to calculate curved soft threshold estimator, $\hat{\theta}_i$ and D_i separately, where I use λ as a global parameter.

```
gettheta <- function(zi){
  if (zi > lambda){
    return ((zi - lambda)^2)
  }else if(zi < -lambda){
    return (- (zi + lambda)^2)
  }else{
    return (0)
  }
}
getD <- function(zi){
  if (zi > lambda){
    return (2 * (zi - lambda) - 1)
  }else if(zi < -lambda){
    return (-2 * (zi + lambda) - 1)
  }else{
    return (-1)
  }
}
gettheta <- Vectorize(gettheta)
getD <- Vectorize(getD)
```

Then, for each λ, we simulate Z^n for 500 times. Because SURE estimator is unbiased, we can approximate true risk by take average over the 500 $\hat{R}(z)$'s. Finally we can find the optimal λ related to this model (the model in problem 1).

```
lambda.grid <- seq(from = 2, to = 4, by = 0.01)
risk.record <- c()
risk.record.true <- c()
theta <- (1:1000)^(-2)
for (i in 1:length(lambda.grid)){
  lambda <- lambda.grid[i]
  sure_record <- c()
  risk_true <- c()
  for (j in 1:500){
    Z <- theta + rnorm(1000)
    thetahat <- gettheta(Z)
    risk.hat <- 1000 + 2 * sum(getD(Z)) + sum((thetahat - Z)^2)
    sure_record[j] <- risk.hat
    risk_true[j] <- sum((thetahat - theta)^2)
  }
  risk.record[i] <- mean(sure_record)
  risk.record.true[i] <- mean(risk_true)
}
plot(lambda.grid, risk.record, type = "l", xlab = "lambda", ylab = "Risk", yaxt = "n")
points(lambda.grid, risk.record.true, type = "l", col = "red")
axis(2, at = c(1.081153, 5, 10), label = c("Pinsker", "5", "10"))
abline(h = 1.081153, lty = 2, col = "grey")
abline(h = 1.082, lty = 2, col = "blue")
abline(h = 1.922, lty = 2, col = "blue")
```



We should also note that here because we know the exact value of θ, we can calculate the true risk, which I use red line in the above plot. That is to say, black line should be an unbiased estimator for red line. It can be told from the plot that the risk decreases monotonically with λ increasing and it converges to the Pinsker bound. When $\lambda \to \infty$, the curved soft threshold estimator can be approximately seen as zero estimator. Therefore, the final comment here is that the curved soft threshold estimator is less efficient than James-Stein estimator, which can also be concluded by observing the below plot where blue dotted lines denote quartiles (IQR) of James-stein estimator.

New Case

In this new case, we changed the true θ^n and do similar comparison. To begin with, we simulate the distribution of James-Stein estimator and the corresponding risk. Also here the true risk of linear estimator is given by the following formula.

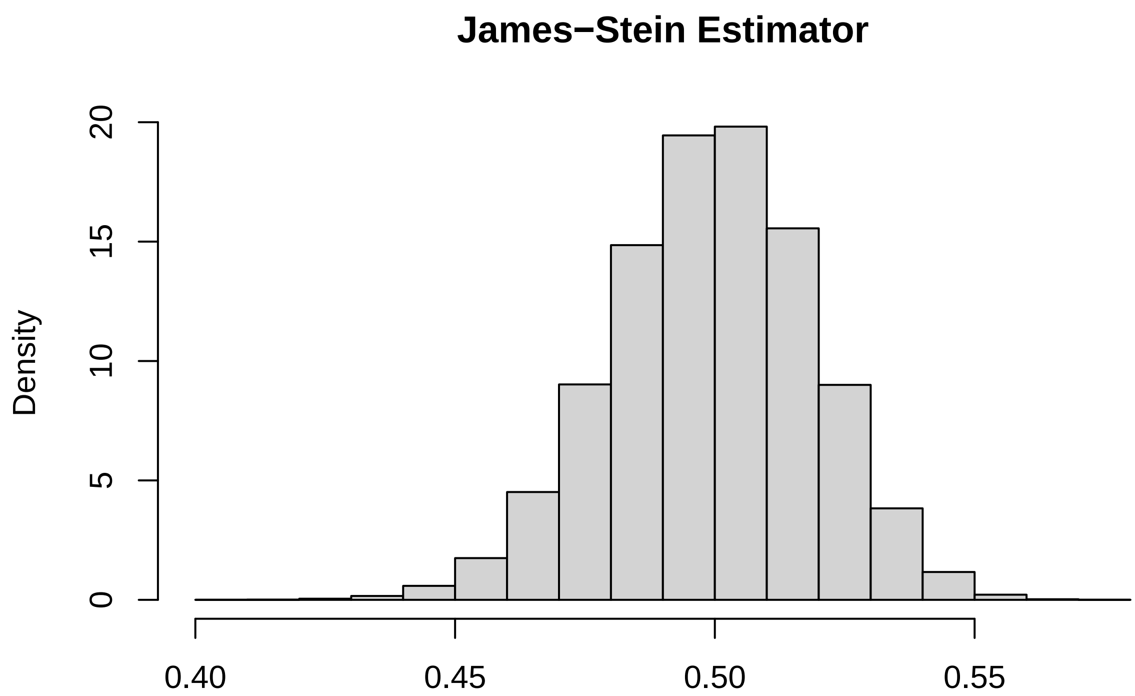
$$R(bZ, \theta) = \sum_{i=1}^{1000} \mathbb{E}_\theta[(bZ_i - \theta_i)^2] = 1000(2b^2 - 2b + 1)$$

For the Pinsker bound, because here $\sum_{i=1}^{1000} \theta_i^2 \leqslant 1000$,

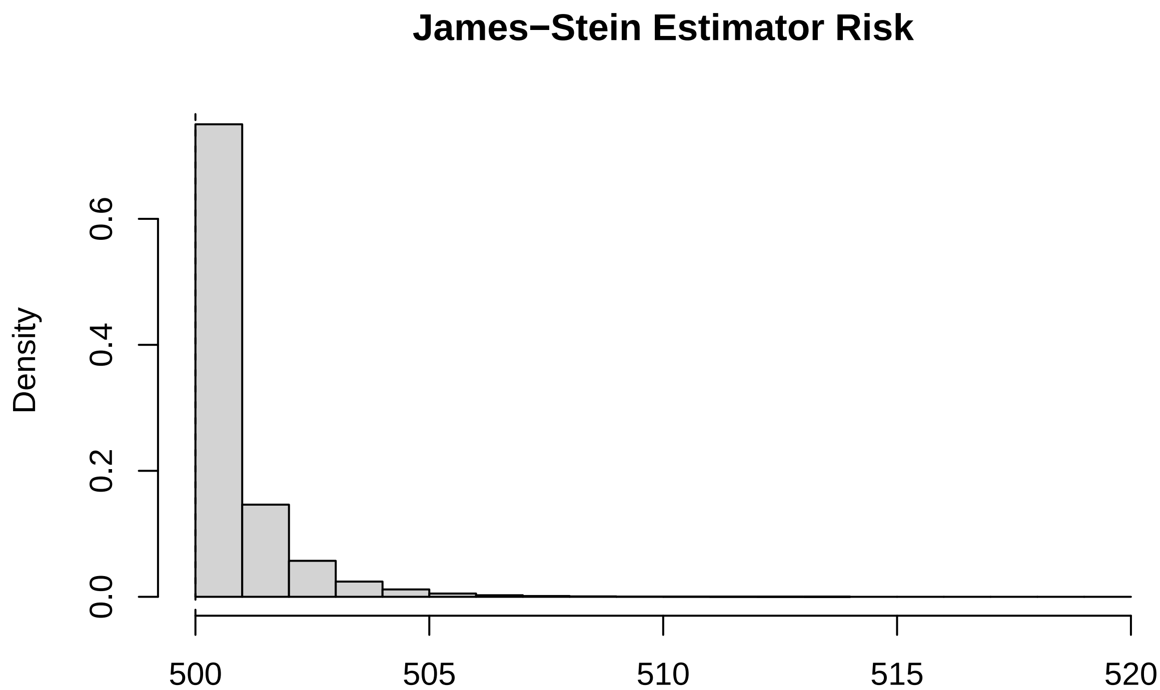
$$\frac{\sigma^2 c^2}{\sigma^2 + c^2} = \frac{1000 \times 1000}{1000 + 1000} = 500$$

```
theta <- c(rep(10, 10), rep(0, 990))
bhat_JS <- c()
for (t in 1:100000){
  Z <- theta + rnorm(1000)
  JS <- max(0, 1 - 1000/sum(Z^2))
```

```
  bhat_JS[t] <- JS
}
risk_JS <- 1000 * (2 * bhat_JS^2 - 2 * bhat_JS + 1)
hist(bhat_JS, xlab = "", main = "James-Stein Estimator", probability = T)
```




```
hist(risk_JS, xlab = "", main = "James-Stein Estimator Risk", probability = T, breaks = 20)
abline(v = 500, lty = 2)
```



The risk of James-Stein estimator can also be explored by its quartiles:

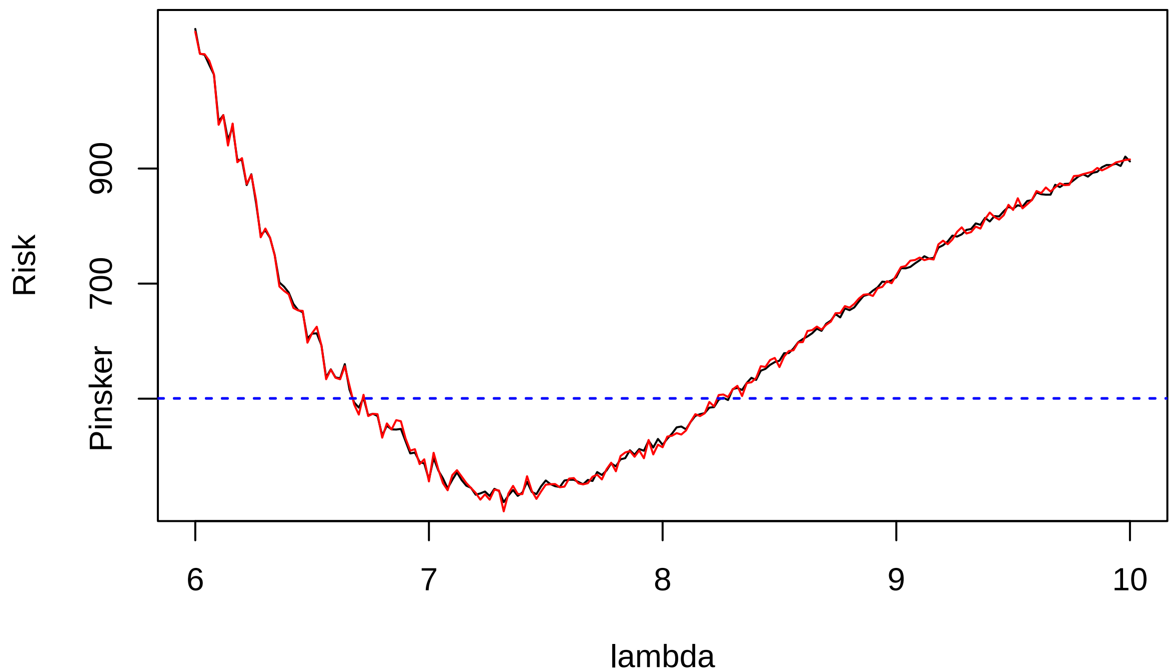
```
quantile(risk_JS)
```

```
##       0%      25%      50%      75%     100%
## 500.0000 500.0761 500.3444 500.9994 519.5383
```

The above result is consistent with Pinsker bound. Next we use similar simulation as before in order to compare the risk of James-Stein estimator with that of curved soft threshold.

```
lambda.grid <- seq(from = 6, to = 10, by = 0.02)
risk.record <- c()
risk.record.true <- c()
for (i in 1:length(lambda.grid)){
  lambda <- lambda.grid[i]
  sure_record <- c()
  risk_true <- c()
  for (j in 1:200){
    Z <- theta + rnorm(1000)
    thetahat <- gettheta(Z)
    risk.hat <- 1000 + 2 * sum(getD(Z)) + sum((thetahat - Z)^2)
    sure_record[j] <- risk.hat
    risk_true[j] <- sum((thetahat - theta)^2)
  }
  risk.record[i] <- mean(sure_record)
  risk.record.true[i] <- mean(risk_true)
}
plot(lambda.grid, risk.record, type = "l", xlab = "lambda", ylab = "Risk", yaxt = "n")
```

```
points(lambda.grid, risk.record.true, type = "l", col = "red")
axis(2, at = c(500, 700, 900), label = c("Pinsker", "700", "900"))
abline(h = 500, lty = 2, col = "grey")
abline(h = quantile(risk_JS)[2], lty = 2, col = "blue")
abline(h = quantile(risk_JS)[4], lty = 2, col = "blue")
```



Therefore, even though James-Stein can still achieve Pinsker bound, this curved soft threshold estimator can be more efficient if we use some clever ways to choose λ in this comparison. The underlying reason is that by using curved soft threshold, we can separately estimate those $\hat{\theta}_i$'s whose corresponding Z_i's is derived from $\theta_i = 10$ and those that have true $\theta_i = 0$.

Problem 3

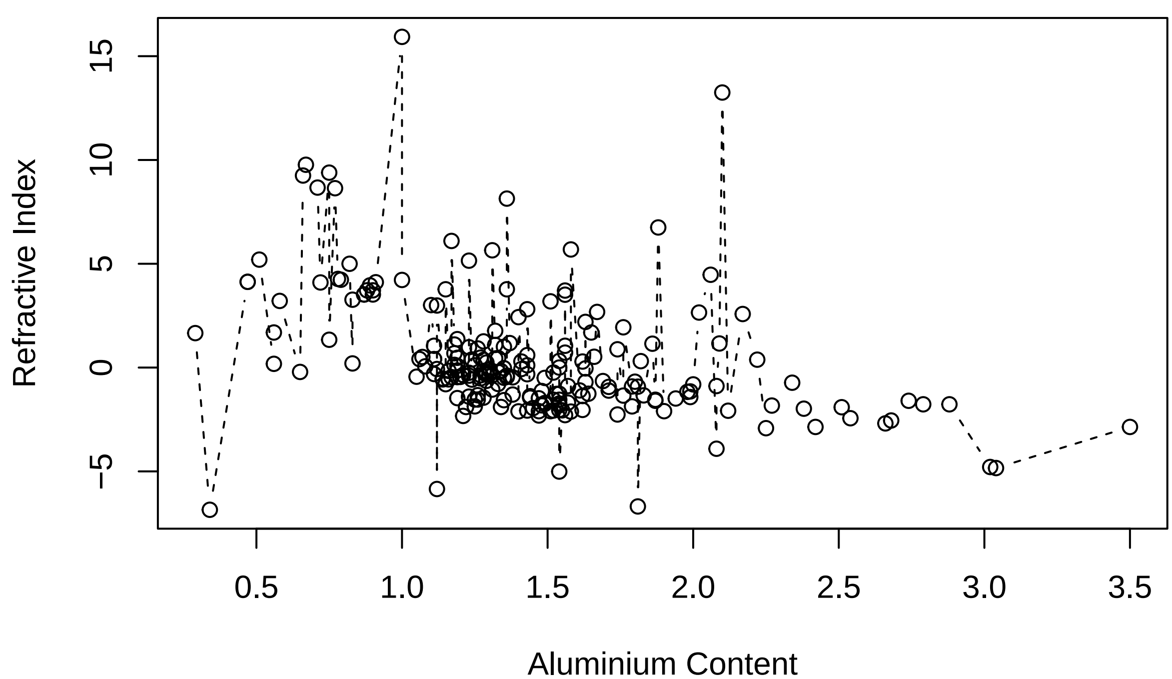
To begin with, we read the data as follows.

```
glass <- read.table("glass.dat")
head(glass[,c(1,4)])
```

```
##      RI   Al
## 1  3.01 1.10
## 2 -0.39 1.36
## 3 -1.82 1.54
## 4 -0.34 1.29
## 5 -0.58 1.24
## 6 -2.04 1.62
```

The difficulty of this problem is that we need to deal with irregular design. To see this, we plot the used data as follows and it can be observed that x are not equally spaced.

```
X <- glass$Al
Y <- glass$RI
Y <- Y[order(X)]
X <- X[order(X)]
plot(X, Y, xlab = "Aluminium Content", ylab = "Refractive Index", type = "b", lty = 2)
```



Next, we need to use Gram-Schmidt method to find a special basis for this series. In order to compute Z_i from Y_i, we need the values of basis functions at data points. Here I will construct two matrix, both of which have dimension 214×214. This is because we have 214 observations and we use 214 basis functions. The first matrix denotes the commonly-used cosine basis of regular design. We will transform it to the second matrix which corresponds to our dataset. Each row of the two matrix denotes each basis function. To be more specific, each row has values of basis functions at data points.

```
n <- length(X)

cosine_basis <- matrix(1, nrow = n, ncol = n)
for (i in 2:n){
  cosine_basis[i,] <- sqrt(2) * cos((i - 1) * pi * X)
}

new_basis <- matrix(1, nrow = n, ncol = n)
for (i in 2:n){
  new_basis[i,] <- cosine_basis[i,] # update this row step by step
  for (j in 1:(i-1)){
    coefficient_new_basis <- as.numeric(cosine_basis[i,] %*% new_basis[j,])
    coefficient_new_basis <- coefficient_new_basis / 214
    new_basis[i,] <- new_basis[i,] - coefficient_new_basis*new_basis[j,]
  }
  new_basis[i,] <- new_basis[i,] / sqrt((sum(new_basis[i,]^2)) / length(X))
}
```

Then by (8.27), normal means Z_j's can be defined by

$$Z_j = \frac{1}{n}\sum_{i=1}^{n} Y_i \phi_j(x_i)$$

```
Z <- c()
for (j in 1:214){
  Z[j] <- sum(Y * new_basis[j,]) / 214
}
```

In order to use theorem (8.13), we need to first estimate the variance. Here I adopt (8.12) as follows.

$$\hat{\sigma}^2 = \frac{1}{n - J_n}\sum_{i=J_n+1}^{n} Z_i^2$$

where, as default value,

$$J_n = 53 \approx \frac{n}{4}$$

Therefore,

```
sig_hat <- sqrt(sum(Z[54:214]^2) / 161)
print(paste("Esimated standard deviation is", sig_hat))
```

```
## [1] "Esimated standard deviation is 0.189370427790791"
```

For comparison, the standard deviation estimated from nonparametric regression is roughly 2.5. The problem here is that the variance that we estimated is variance of Z_i but previously we estimated the variance of Y_i. Then we can use (8.15) from theorem (8.13) to estimate the risk and choose optimal b.

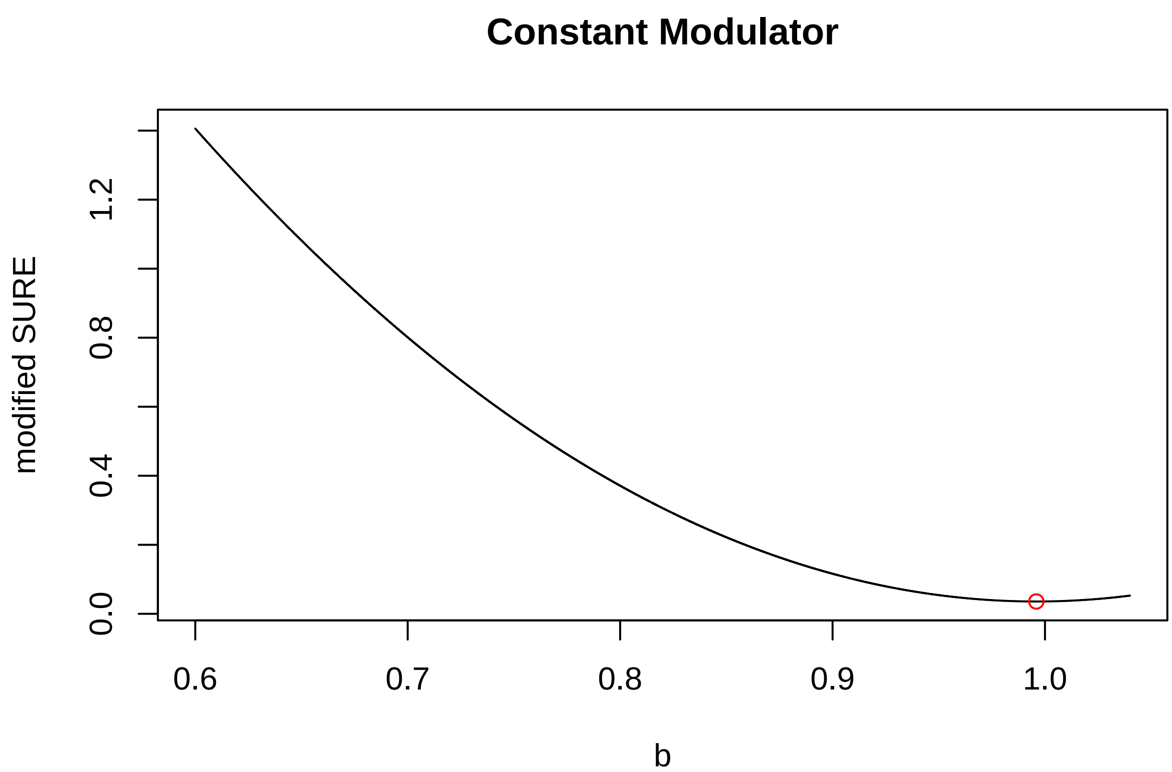
$$\hat{R}(b) = \sum_{j=1}^{n}\left(Z_j^2 - \frac{\hat{\sigma}^2}{n}\right)_+ (1 - b_j)^2 + \frac{\hat{\sigma}^2}{n}\sum_{j=1}^{n} b_j^2$$

Here it is somehow too difficult to use monotone modulator. I will try both constant modulator and nested subset modulator and compare the risk of them.

Constant Modulator

Here I try different $b^n = (b, b, ..., b)$ and choose the one with minimum estimated (modified) SURE risk given above.

```
b_record <- seq(from = 0.6, to = 1.04, by = 0.00001)
risk_record <- c()
for (k in 1:length(b_record)){
  b <- rep(b_record[k], 214)
  risk_first_term <- sum(sapply(Z^2 - (sig_hat^2) / 214, max, 0)) * ((1 - b_record[k])^2)
  risk_second_term <- (sig_hat^2 * sum(b^2)) / 214
  risk_record[k] <- risk_first_term + risk_second_term
}
plot(b_record, risk_record, type = "l", xlab = "b", ylab = "modified SURE", main = "Constant Modulator")
points(b_record[which.min(risk_record)], risk_record[which.min(risk_record)], col = "red")
```



The optimal b and related risk is:

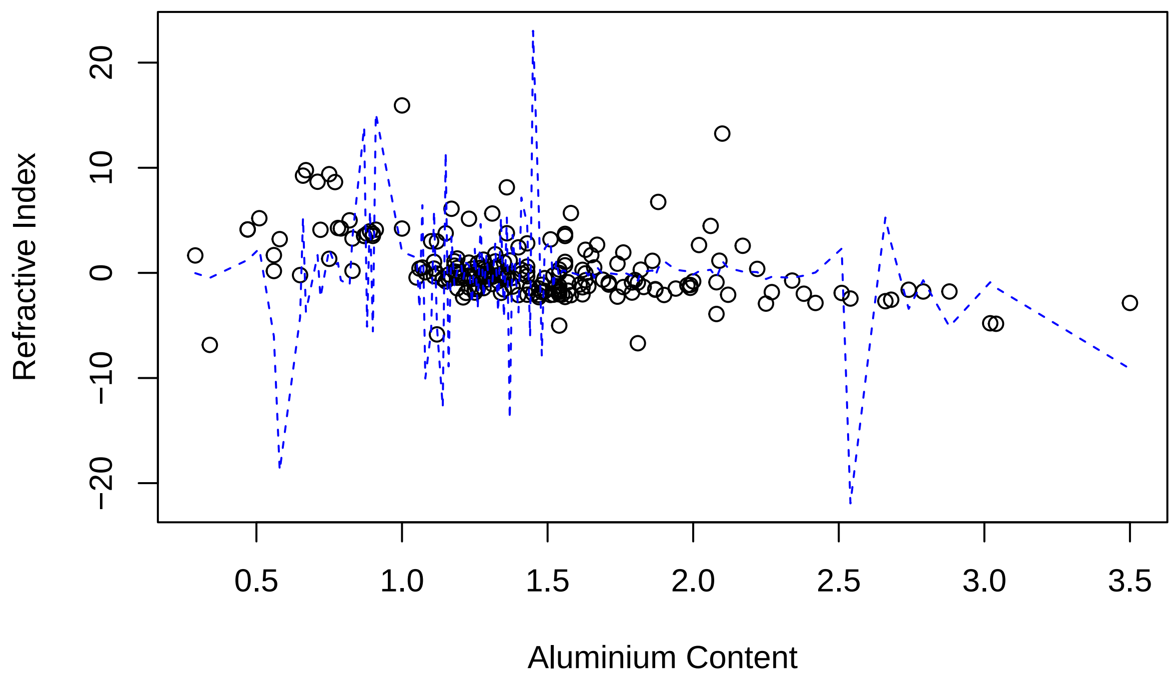
```
print(paste("The optimal b is", b_record[which.min(risk_record)], "with risk", risk_record[which.min(ri
```

```
## [1] "The optimal b is 0.9959 with risk 0.0357140561450447"
```

Here the estimated risk is quite lower than that of nonparametric regression which is roughly 7. Then the fitted model is as follows. Here I decreased b to a large extent in order to smooth it. I use $b = 0.2$. Note that $b = 0.2$ corresponds to risk equal to 5.573 which is similar to what we get in nonparametric regression.

```
coefficient_fitted <- diag(0.2 * Z, nrow = 214, ncol = 214)
Yhat <- apply(coefficient_fitted %*% new_basis, 2, sum)
plot(X, Yhat, xlab = "Aluminium Content", ylab = "Refractive Index", type = "n")
points(X, Y)
```

```
points(X, Yhat, type = "l", col = "blue", lty = 2)
```



In order to estimate the variance of ϵ_i in $Y_i = r(X_i) + \epsilon_i$, we need to construct the L matrix and use $\frac{\sum (Y_i - \hat{r}(X_i))^2}{n - 2v + \tilde{v}}$.

```
L <- matrix(NA, nrow = 214, ncol = 214)
for (k in 1:214){
  for (i in 1:214){
    L[k,i] <- sum(0.2 * (new_basis[,k] * new_basis[,i])) / 214
  }
}
v <- sum(diag(L))
v_tilde <- sum(diag(t(L) %*% L))
print((sum((Y - Yhat)^2)) / (214 - 2*v + v_tilde))
```

```
## [1] 9.608412
```

which is quite similar to what we get in nonparametric regression!

Nested Subset Modulator

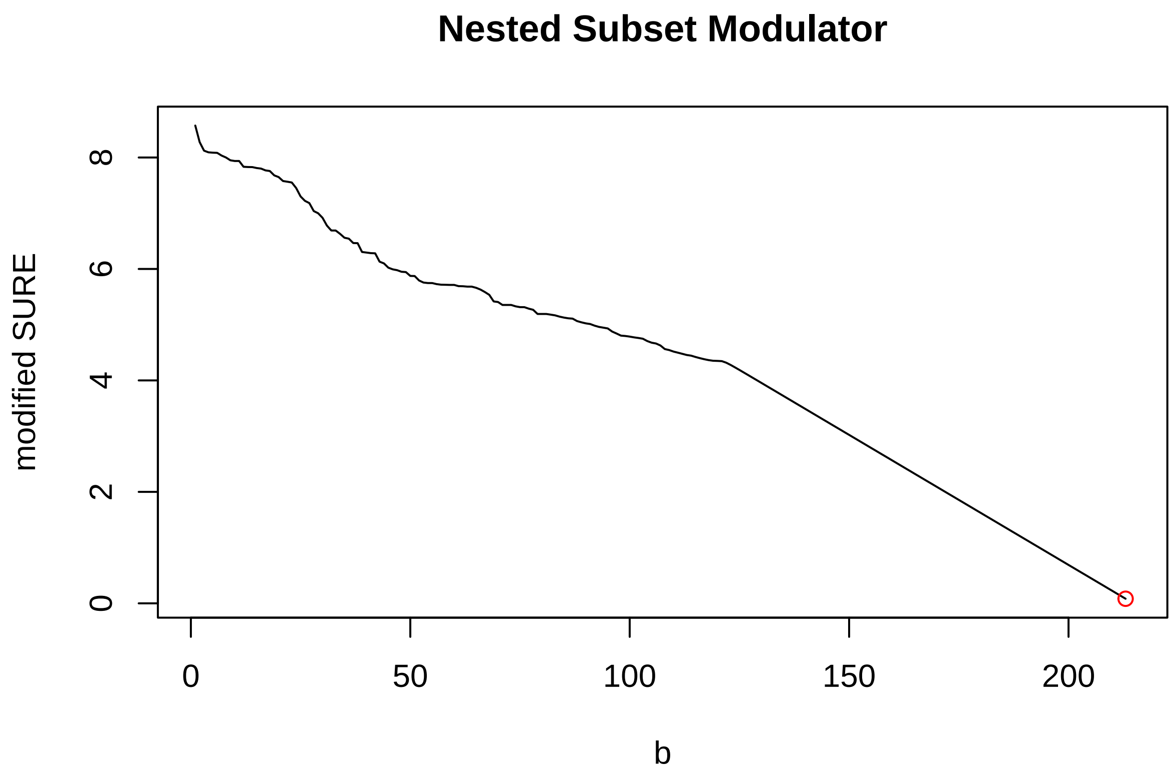
Here we need to choose how many 1 we would like to use:

$$(\underbrace{1, ..., 1}_{n_0}, 0, ..., 0)$$

Similarly the modified SURE risk estimator is used to choose optimal n_0.

$$\hat{R}(b) = \sum_{j=1}^{n} \left(Z_j^2 - \frac{\hat{\sigma}^2}{n} \right)_+ (1 - b_j)^2 + \frac{\hat{\sigma}^2}{n} \sum_{j=1}^{n} b_j^2$$

```
risk_record <- c()
for (n0 in 1:214){
  b <- c(rep(1, n0), rep(0, 214 - n0))
  risk_first_term <- sum(sapply(Z^2 - (sig_hat^2) / 214, max, 0)[(n0+1):214])
  risk_second_term <- (sig_hat^2 * sum(b^2)) / 214
  risk_record[n0] <- risk_first_term + risk_second_term
}
plot(1:214, risk_record, type = "l", xlab = "b", ylab = "modified SURE", main = "Nested Subset Modulato
points(which.min(risk_record), risk_record[which.min(risk_record)], col = "red")
```



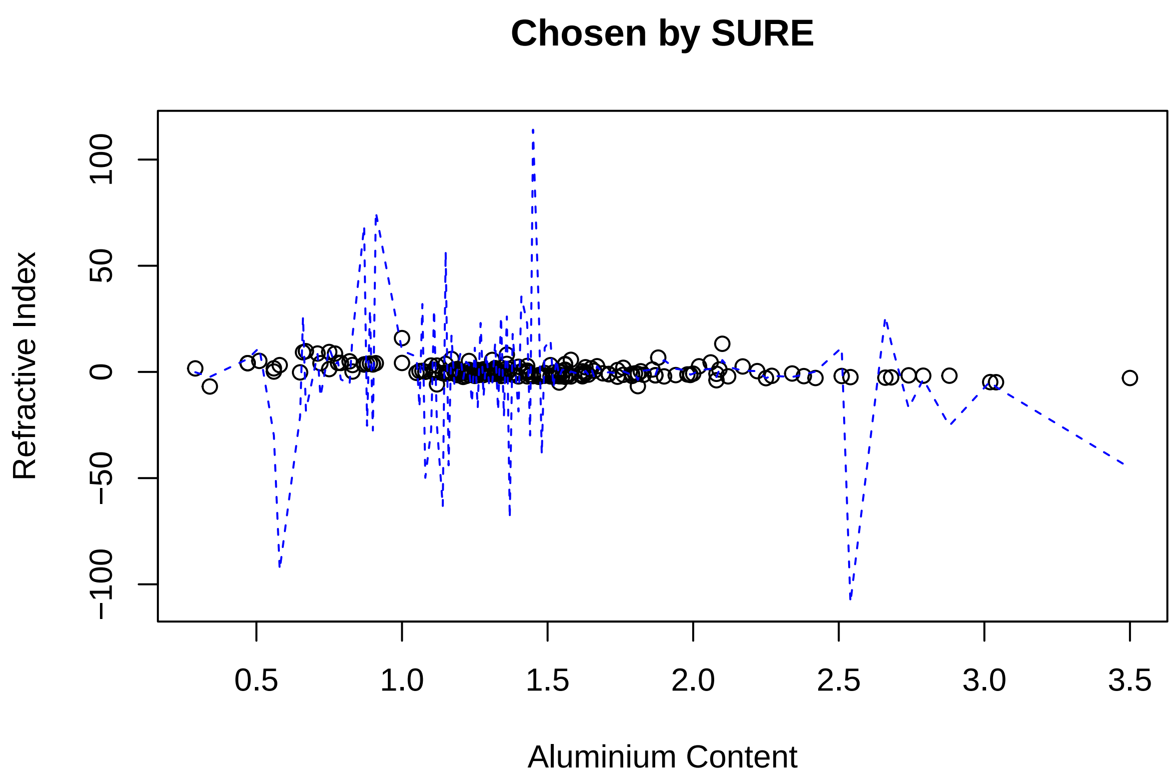

```
print(paste("Best n0 chosen by SURE is", which.min(risk_record), "with risk", risk_record[which.min(risk
```

```
## [1] "Best n0 chosen by SURE is 213 with risk 0.0825279902121986"
```

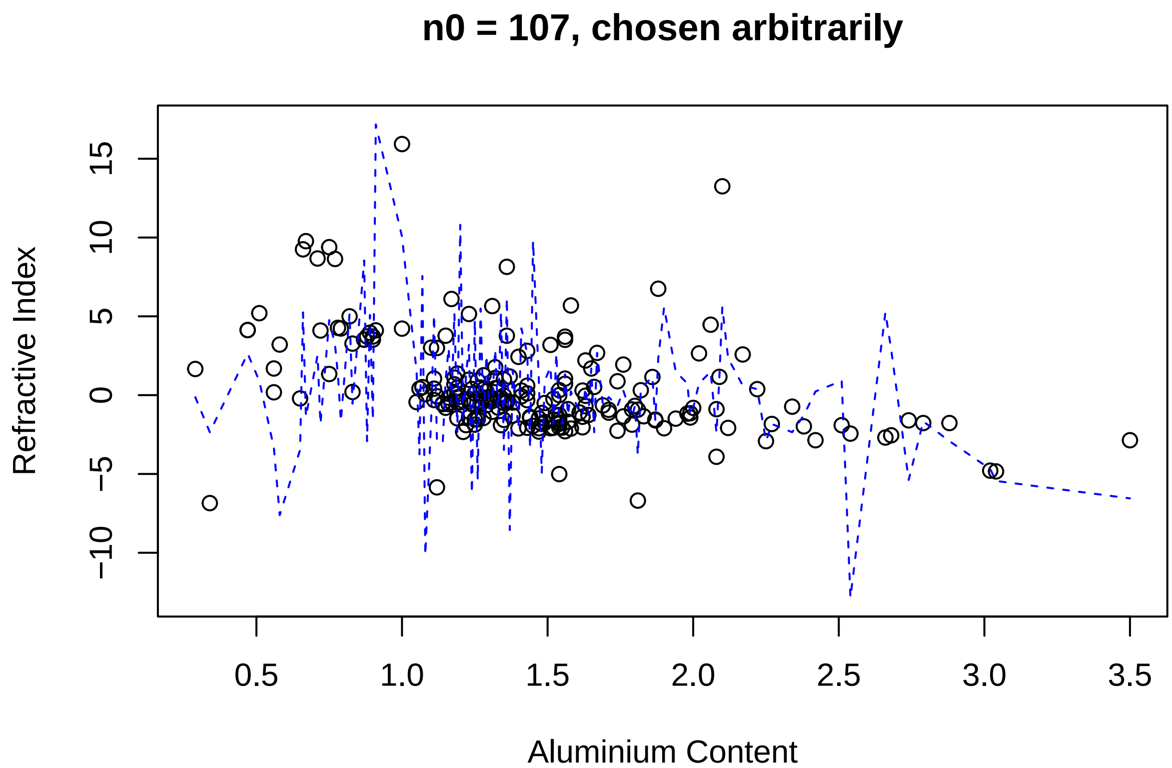
which is also much lower than nonparametric regression. Here we can conclude that in this dataset it is not appropriate to use cosine basis. We use the following plots to show that by using modified SURE estimator, it is under smoothed.

```
b <- c(rep(1, 213), rep(0, 214 - 213))
coefficient_fitted <- diag(b * Z, nrow = 214, ncol = 214)
Yhat <- apply(coefficient_fitted %*% new_basis, 2, sum)
plot(X, Yhat, xlab = "Aluminium Content", ylab = "Refractive Index", type = "n", main = "Chosen by SURE
```

```
points(X, Y)
points(X, Yhat, type = "l", col = "blue", lty = 2)
```




```
b <- c(rep(1, 107), rep(0, 214 - 107))
coefficient_fitted <- diag(b * Z, nrow = 214, ncol = 214)
Yhat <- apply(coefficient_fitted %*% new_basis, 2, sum)
plot(X, Yhat, xlab = "Aluminium Content", ylab = "Refractive Index", type = "n", main = "n0 = 107, chos
points(X, Y)
points(X, Yhat, type = "l", col = "blue", lty = 2)
```



In this case, the variance of ϵ_i is estimated as

```
L <- matrix(NA, nrow = 214, ncol = 214)
for (k in 1:214){
  for (i in 1:214){
    L[k,i] <- sum(b * (new_basis[,k] * new_basis[,i])) / 214
  }
}
v <- sum(diag(L))
v_tilde <- sum(diag(t(L) %*% L))
print((sum((Y - Yhat)^2)) / (214 - 2*v + v_tilde))
```

```
## [1] 5.54435
```

which is also quite similar to what we get in nonparametric regression.