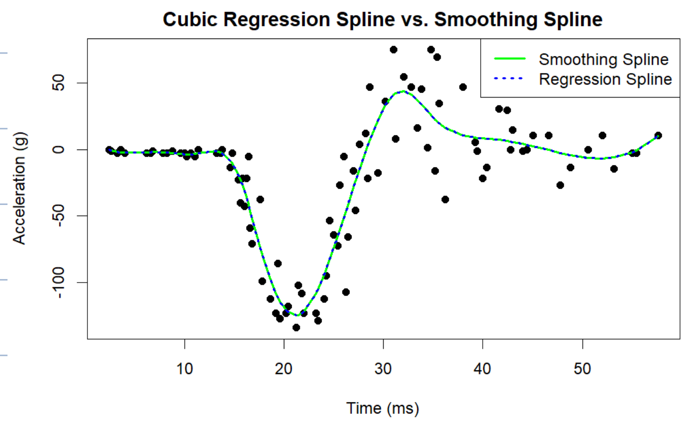
11. Get the motorcycle data from the book website. Fit a cubic regression spline with equally spaced knots. Use leave-one-out cross-validation to choose the number of knots. Now fit a smoothing spline and compare the fits.

Two method of fit almost overlap with each other



15. Let $Y_i \sim N(\mu_i, 1)$ for $i = 1, \ldots, n$ be independent observations. Find the estimators that minimizes each of the following penalized sums of squares:

$$(a) \qquad \sum_{i=1}^{n}(Y_i - \widehat{\mu}_i)^2 + \lambda \sum_{i=1}^{n} \widehat{\mu}_i^2$$

$$\frac{\partial f}{\partial \hat{\mu}_i} = 2(Y_i - \hat{\mu}_i)\cdot(-1) + 2\lambda\hat{\mu}_i$$

$$= 2\lambda\hat{\mu}_i - 2(Y_i - \hat{\mu}_i)$$

$$= 2\lambda\hat{\mu}_i - 2Y_i + 2\hat{\mu}_i$$

$$= 2(\lambda+1)\hat{\mu}_i - 2Y_i$$

Let $\frac{\partial f_i}{\partial \hat{\mu}_i} = 0 \Rightarrow \hat{\mu}_i^* = Y_i / \lambda+1$

$$(b) \qquad \sum_{i=1}^{n}(Y_i - \widehat{\mu}_i)^2 + \lambda \sum_{i=1}^{n} |\widehat{\mu}_i|$$

when $\hat{\mu}_i \neq 0$. $\quad \frac{\partial f_i}{\partial \hat{\mu}_i} = -2(Y_i - \hat{\mu}_i) + \lambda \, \text{sign}(\hat{\mu}_i)$

when $\hat{\mu}_i = 0$ $\quad -\lambda \le -2Y_i \le \lambda$

① If $Y_i > \frac{\lambda}{2} \Rightarrow \hat{\mu}_i^* = Y_i - \frac{\lambda}{2}$

② If $Y_i < -\frac{\lambda}{2} \Rightarrow \hat{\mu}_i^* = Y_i + \frac{\lambda}{2}$

③ If $-\frac{\lambda}{2} \le Y_i \le \frac{\lambda}{2} \Rightarrow \hat{\mu}_i^* = 0$

(c) $$\sum_{i=1}^{n}(Y_i - \widehat{\mu}_i)^2 + \lambda \sum_{i=1}^{n} I(\widehat{\mu}_i = 0).$$

① If $Y_i \neq 0$ $\quad \begin{cases} \text{i) } Y_i^2 < \lambda, & \hat{\mu}_i^* = 0 \\ \text{ii) } Y_i^2 \ge \lambda, & \hat{\mu}_i^* = Y_i \end{cases}$

② If $Y_i = 0$ $\quad \hat{\mu}_i^* = 0$

Suppose we assume homoscedasticity in the Boston housing example. The variance can be estimated using various nonparametric regression methods. Calculate the smoothing matrices (using the optimal bandwidth/tuning parameter chosen by cross-validation) for Nadaraya-Waston kernel regression, local linear regression, smoothing splines, and regression splines (10 quantile-based knots). With the smoothing matrices, obtain and compare the estimates for σ^2.

See the codes

```
```{r}
# 1. Nadaraya-Watson Kernel Regression
bw_nw <- npregbw(formula = y ~ x, regtype = "lc", bwmethod = "cv.aic")
h_nw <- bw_nw$bw
Epanechnikov <- function(u) 0.75 * (1 - u^2) * (abs(u) <= 1)
S_nw <- matrix(0, n, n)
for (i in 1:n) {
  u <- (x - x[i]) / h_nw
  K <- Epanechnikov(u)
  S_nw[i, ] <- K / sum(K)
}
sigma2_nw <- estimate_sigma2(y, S_nw)

S_nw
sigma2_nw
```

```
```{r}
# 2. Local Linear Regression
bw_ll <- npregbw(formula = y ~ x, regtype = "ll", bwmethod = "cv.aic")
h_ll <- bw_ll$bw
S_ll <- matrix(0, n, n)
for (i in 1:n) {
  xi <- x[i]
  u <- (x - xi) / h_ll
  K <- Epanechnikov(u)
  valid <- K > 0
  X <- cbind(1, x[valid] - xi)
  W <- diag(K[valid])
  if (sum(valid) >= 2 && qr(X)$rank == 2) {
    XWX_inv <- solve(t(X) %*% W %*% X)
    M_i <- XWX_inv %*% t(X) %*% W
    S_ll[i, valid] <- M_i[1, ]
  } else {
    S_ll[i, valid] <- K[valid] / sum(K[valid])
  }
}
sigma2_ll <- estimate_sigma2(y, S_ll)

S_ll
sigma2_ll
```

```
```{r}
# 3. Smoothing Splines

#use cross-validation to choose best smoothing parameter
spar = seq(0.01, 1, by = 0.01)
cv = rep_len(NA, length(spar))
for(i in 1:length(spar)){
    tempfit = smooth.spline(x, y, spar = spar[i], cv=TRUE, all.knots = TRUE)
    cv[i] = tempfit$cv.crit
}

#use the optimal smoothing parameter to produce a final fit
fit = smooth.spline(x, y, spar = spar[which(cv == min(cv))], cv=TRUE, all.knots = TRUE)

#calculate the smoothing matrix
S_ss = matrix(nrow = length(x), ncol = length(x))
for(j in 1:length(x)){
    yi = rep_len(0, length(x))
    yi[j] = 1
    S_ss[,j] = predict(smooth.spline(x, yi, lambda = fit$lambda, cv=TRUE,
                                     all.knots = TRUE), x)$y
}

S_ss

estimate_sigma2(y, S_ss)
```
```

```
```{r}
# 4. Regression Splines (10 knots)
knots <- quantile(x, probs = seq(1/10, 9/10, length.out = 10))
X_bs <- bs(x, knots = knots, intercept = TRUE)
fit_rs <- lm(y ~ X_bs)

# Method 1: Direct computation using linear algebra
S_rs <- X_bs %*% solve( t(X_bs) %*% X_bs) %*% t(X_bs)

S_rs

estimate_sigma2(y, S_rs)
```
```

````
#Boston Housing
```{r}
# Function to estimate sigma^2 using smoothing matrix S
estimate_sigma2 <- function(y, S) {
  y_hat <- S %*% y
  RSS <- sum((y - y_hat)^2)
  tr_S <- sum(diag(S))
  n <- length(y)
  sigma2 <- RSS / (n - tr_S)
  return(sigma2)
}

# Extract predictor and response
x <- Boston$rm
y <- Boston$medv
n <- nrow(Boston)
```
````

$$\hat{\sigma}^2_{NW} = 32.25$$

$$\hat{\sigma}^2_{LL} = 32.43$$

$$\hat{\sigma}^2_{SS} = 32.57$$

$$\hat{\sigma}_{RS} = 35.06$$

The matrices are computed in codes but are too large to demonstrate in P.D.F.