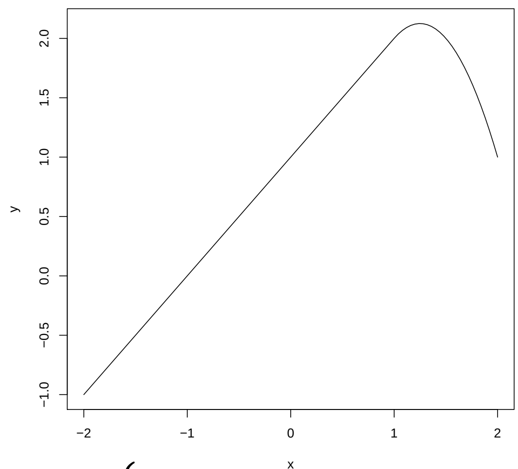
HW #9: Polynomials and Splines (Chap.7, 7.1-7.5)

1. **(Chap. 7, # 3, p. 298)** Suppose we fit a curve with basis functions $b_1(X) = X$ and $b_2(X) = (X-1)^2 I\{X \geq 1\}$. We fit the regression model

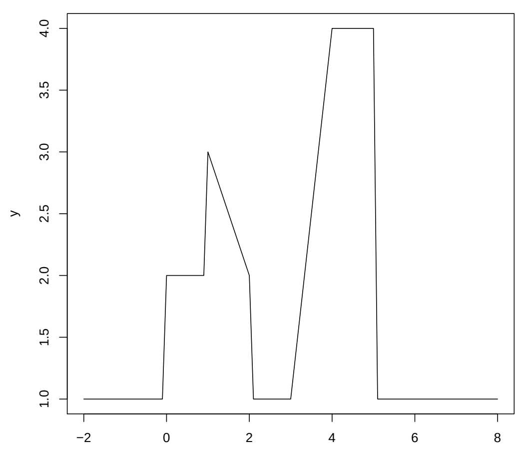
$$Y = \beta_0 + \beta_1 b_1(X) + \beta_2 b_2(X) + \varepsilon$$

and obtain the estimated slopes $\hat{\beta}_0 = 1$, $\hat{\beta}_1 = 1$, $\hat{\beta}_2 = -2$. Sketch the estimated curve between $X = -2$ and $X = 2$. Report the intercepts, slopes, and other relevant information.



$$Y = \begin{cases} X+1 & \text{when } X < 1 \\ X^2 - X + 2 & \text{when } X \geq 1 \end{cases}$$

2. **(Chap. 7, # 4, p. 298)** Repeat the previous exercise with basis functions $b_1(X) = I\{0 \leq X \leq 2\} - (X-1)I\{1 \leq X \leq 2\}$ and $b_2(X) = (X-3)I\{3 \leq X \leq 4\} + I\{4 < X \leq 5\}$ and estimated slopes $\hat{\beta}_0 = 1$, $\hat{\beta}_1 = 1$, $\hat{\beta}_2 = 3$.

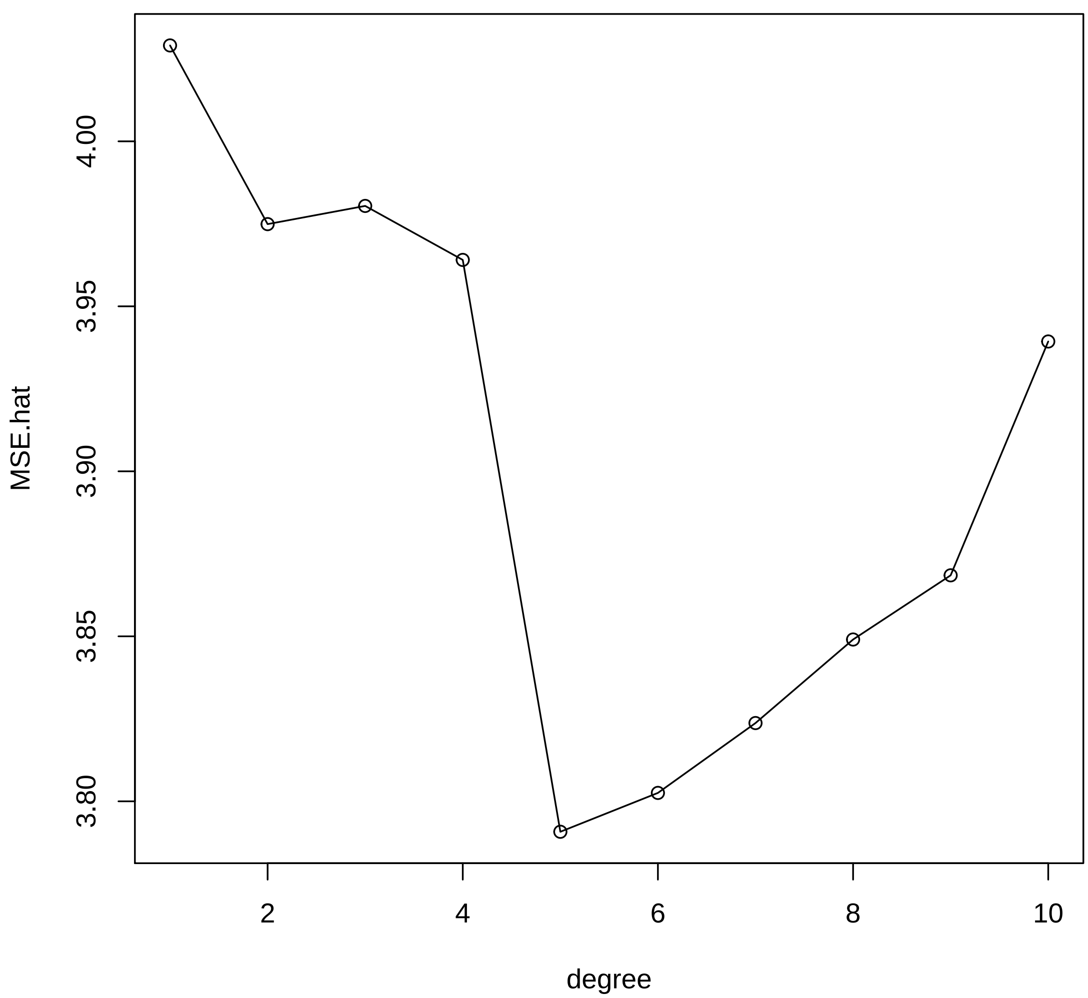


$$Y = \begin{cases} 1 & \text{when } \{X < 0\} \bigcup \{2 < X < 3\} \bigcup \{X > 5\} \\ 2 & \text{when } 0 \leq X < 1 \\ 3 - X & \text{when } 1 \leq X \leq 2 \\ 3X - 8 & \text{when } 3 \leq X \leq 4 \\ 4 & \text{when } 4 < X \leq 5 \end{cases}$$

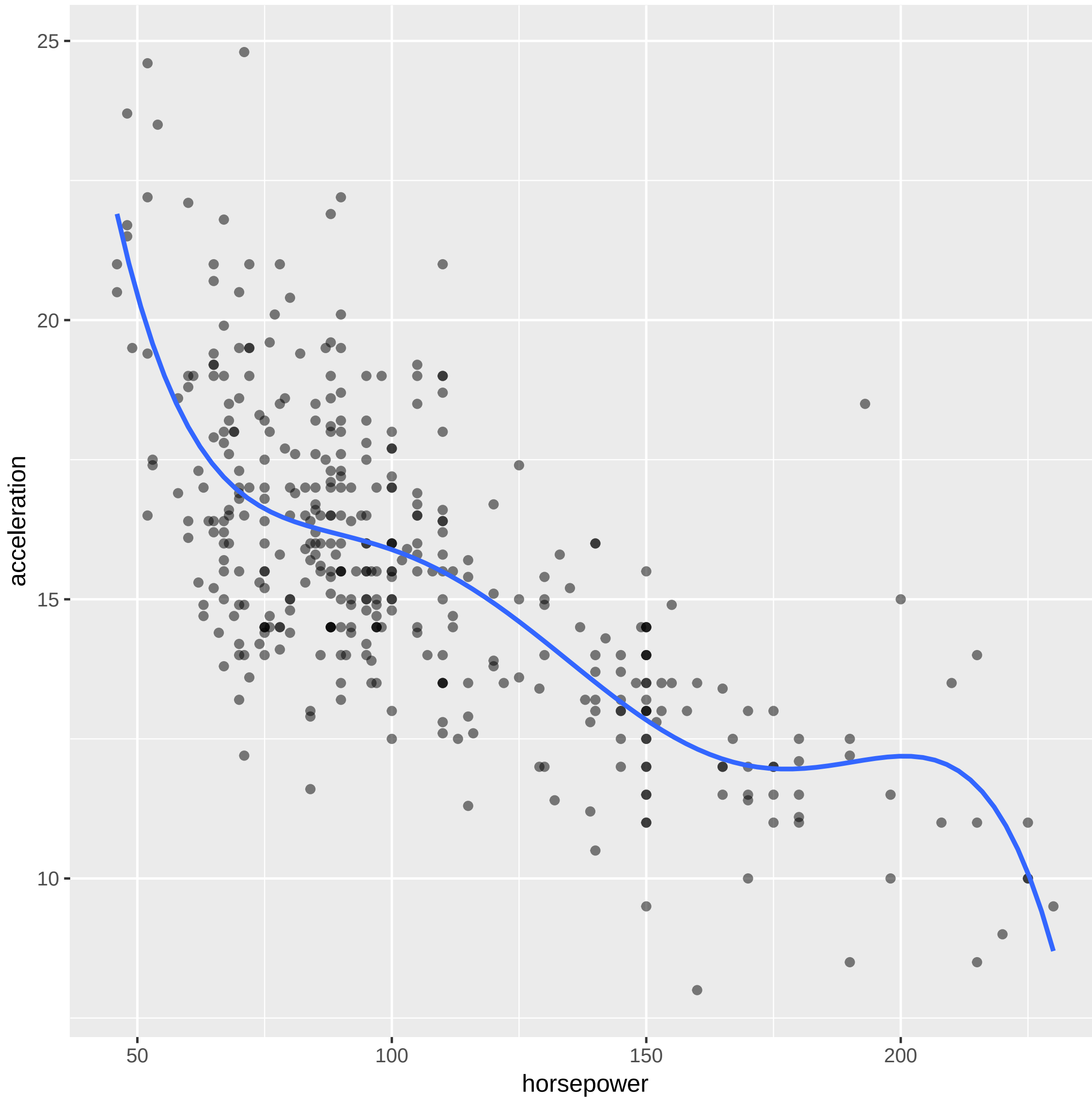
3. **(Chap. 7, ≈# 8, p. 299)** Apply some of the non-linear models discussed in this chapter to the `Auto` data set to predict the vehicle's `acceleration` time based on the `horsepower` of its engine.

   (a) Use cross-validation to select the optimal degree for the polynomial regression.

```
MSE.hat <- rep(NA,10) # testing MSE
for(i in 1:10){
  ply <- glm( acceleration ~ poly(horsepower,i), data = Auto)
  MSE.hat[i] <- cv.glm( Auto, ply)$delta[2] # adjusted MSE penalize # of predicto
}
plot(MSE.hat, xlab = "degree")
lines(MSE.hat)
```



```
which.min(MSE.hat)

## [1] 5

# plot model
ggplot(Auto, aes(y=acceleration, x=horsepower)) +
  geom_point(alpha = .5) +
  stat_smooth(method = "lm", formula = y ~ poly(x,which.min(MSE.hat)),
              se = FALSE)
```



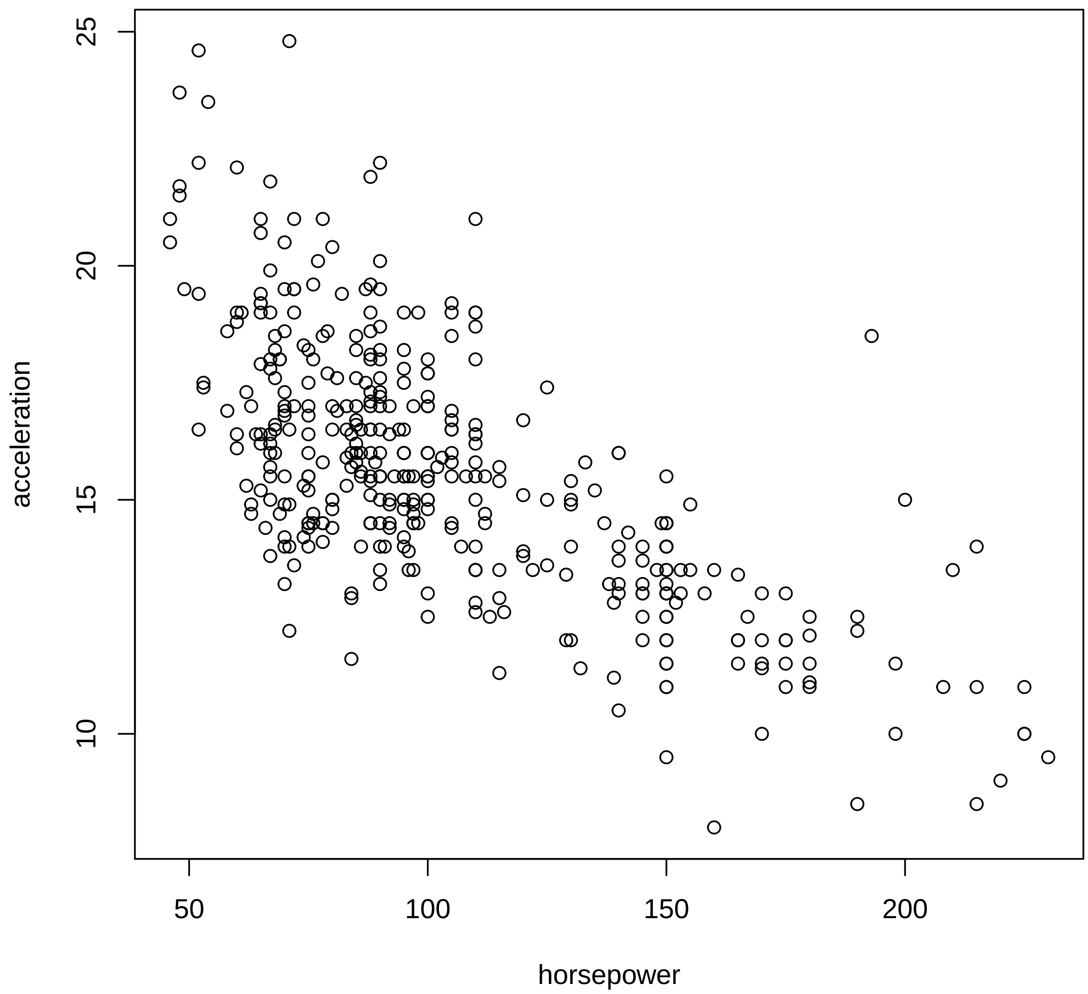

```
# MSE
set.seed(666)
n <- nrow(Auto)
z <- sample(n,n/2)
```

```
ply.train <- glm( acceleration ~ poly(horsepower,i), data = Auto[z,])
mean((acceleration[-z] - predict(ply.train, newx=horsepower[-z]))^2) # test MSE

## [1] 12.17566
```

The optimal degree for the polynimial regression is 5.

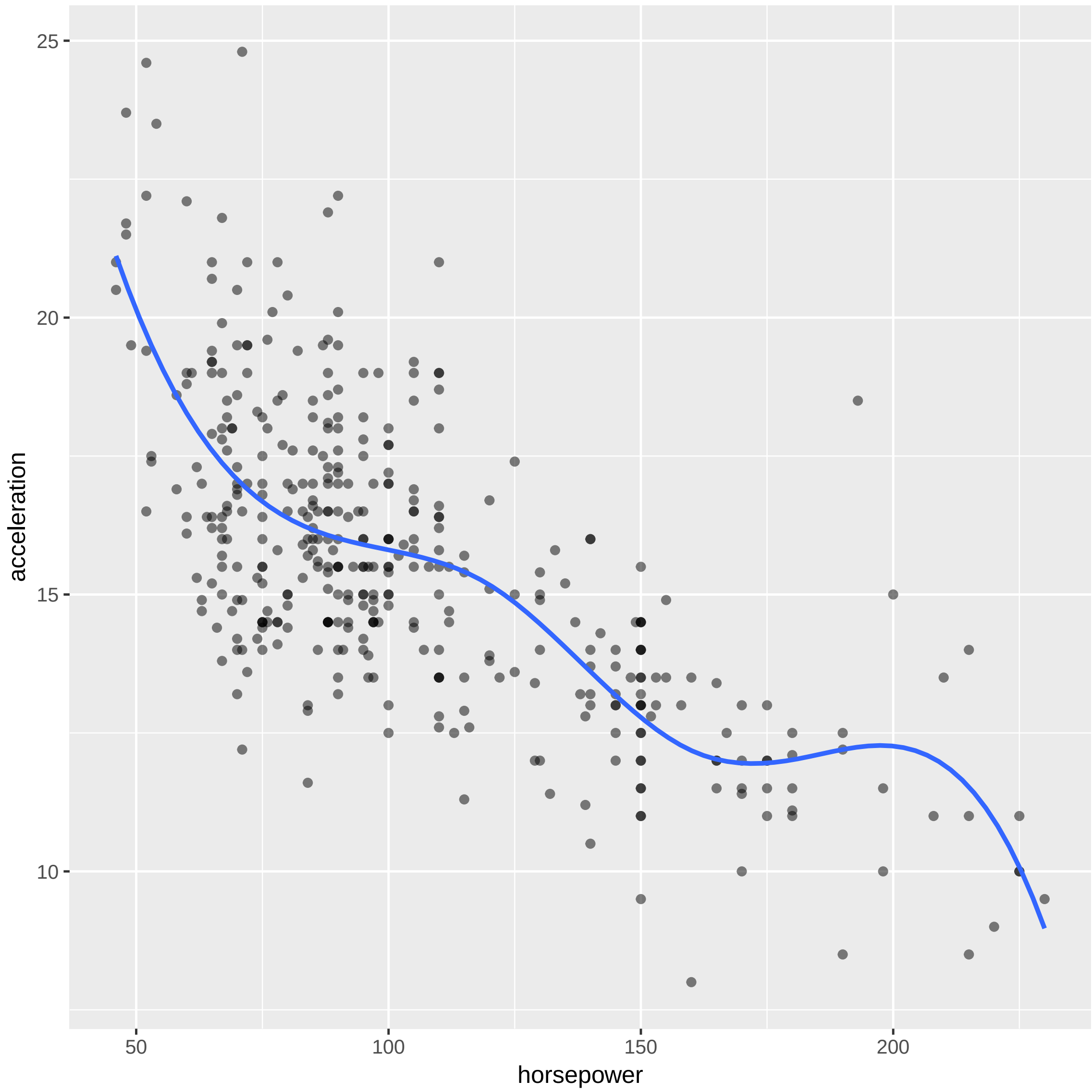
(b) Looking at the scatterplot of acceleration vs horsepower, choose some knots and fit a regression spline.

```
plot(horsepower, acceleration)
```




```
spline <- lm( acceleration ~ bs(horsepower, knots = c(120,160,180)), data = Auto)
# plot model
ggplot(Auto, aes(y=acceleration, x=horsepower)) +
```

```
  geom_point(alpha = .5) +
  stat_smooth(method = "lm", formula = y~ bs(x, knots = c(120,160,180)),
              se = FALSE)
```



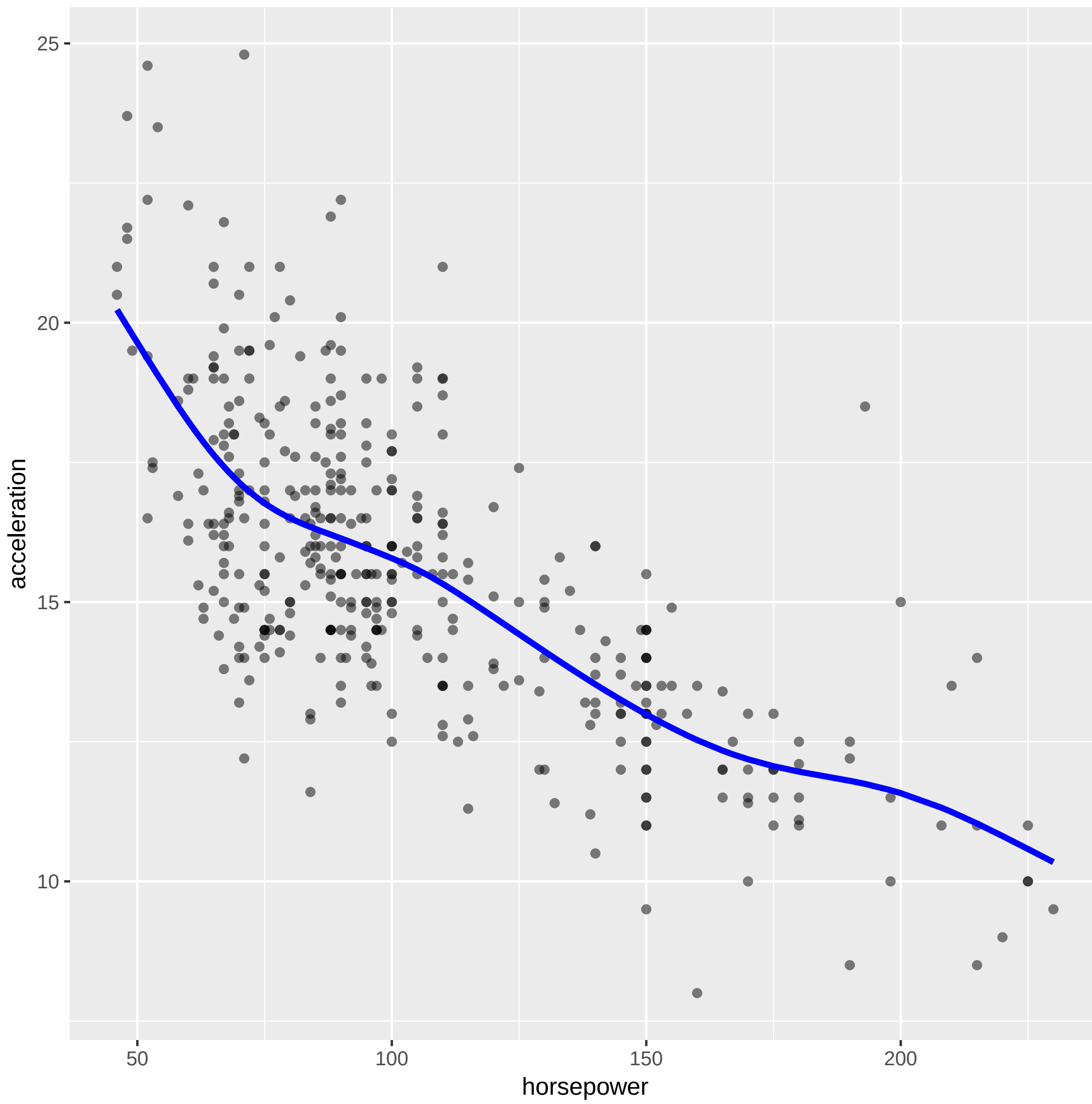

```
# MSE
spline.train <- lm( acceleration ~ bs(horsepower, knots = c(120,160,180)),
                    data = Auto[z,])
mean((acceleration[-z] - predict(spline.train, newx=horsepower[-z]))^2) # test MSE

## [1] 11.88145
```

(c) Fit a smoothing spline, selecting the smoothing parameter by cross-validation.

```r
MSE.hat <- rep(NA,100)
for (k in 1:100){
  d.f <- 2 + k/25
  ss = smooth.spline(horsepower, acceleration, df = d.f) # spline.smooth(x,y)
  MSE.hat[k] <- ss$cv.crit
}
2 + which.min(MSE.hat)/25 # best df
```

```
## [1] 6
```

```r
# plot model
ggplot(Auto, aes(y=acceleration, x=horsepower)) +
  geom_point(alpha = .5) +
  ggformula::geom_spline(df = 2 + which.min(MSE.hat)/25, col = "blue", lwd = 1.3)
```



```r
# MSE
smooth.train <- smooth.spline(horsepower[z], acceleration[z],
                              df = 2 + which.min(MSE.hat)/25)
mean((acceleration[-z] - predict(smooth.train, x=horsepower[-z])$y)^2) # test MSE

## [1] 3.8615
```

For each method, make a plot of the resulting fitted line, and estimate its prediction mean squared error by some cross-validation technique. Which approach resulted in the best prediction accuracy?

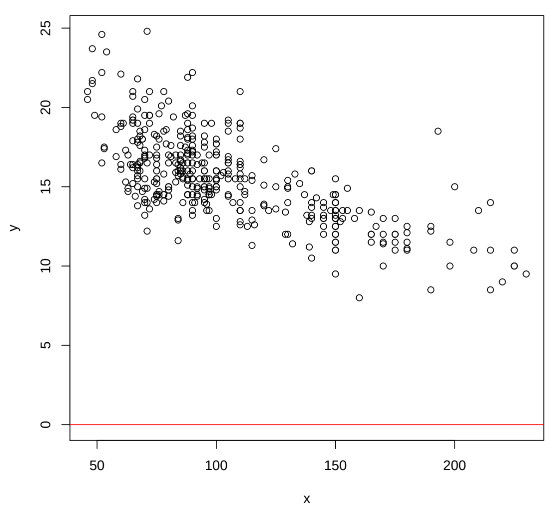
Smoothing spline gives the best prediction accuracy.

4. **(For Stat-627 only... Chap. 7, # 2, p. 298)** Suppose that a curve g is computed to smoothly fit a set of n points using the following formula

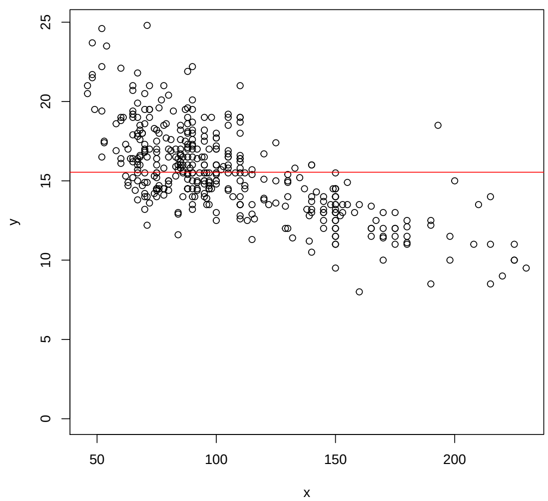
$$\hat{g} = arg \underbrace{min}_{g} \{ \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int [g^{(m)}(x)]^2 dx \}$$

where $g^{(m)}$ is the m-th derivative of g (and $g^{(0)} = g$). Provide example sketches of $\hat{g}$ in each of the following scenarios.

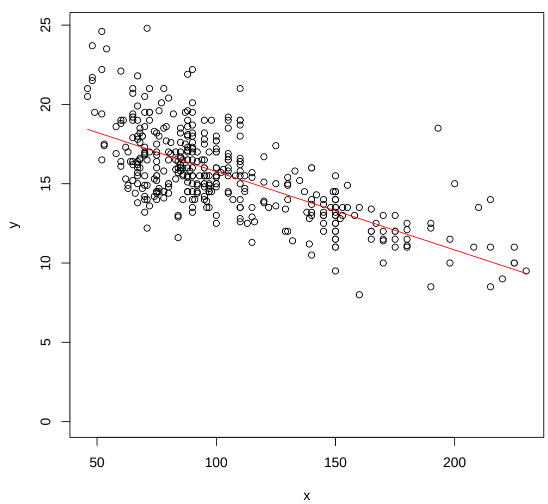
(a) $\lambda = \infty, m = 0$



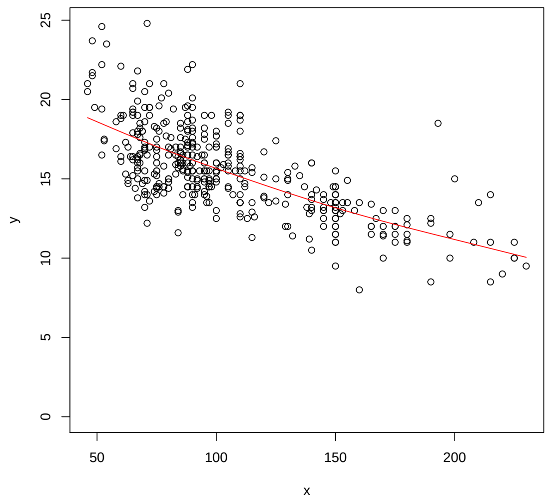
(b) $\lambda = \infty, m = 1$



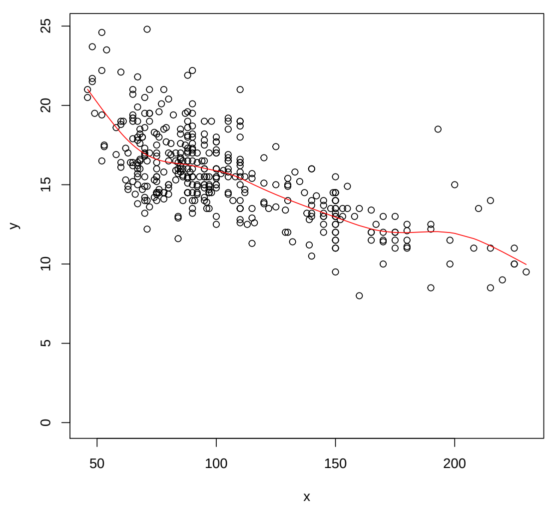
(c) $\lambda = \infty, m = 2$



(d) $\lambda = \infty, m = 3$



(e) $\lambda = 0, m = 3$



This problem does not require you to take evaluate derivatives or integrals. Recall, however, that $g' = g'' = 0$ for a constant, $g' = const$ and $g'' = 0$ for a linear function $g(x)$, and $g'' = const$ for a quadratic function $g(x)$.