Solutions L9

```
knitr::opts_chunk$set(echo = T, message = F,
  fig.width = 7, fig.height=3.5)
options(scipen=999)
```

```
library(splines)
```

```
################################
### Import and process data ###
################################

### Import and clean the air quality data
data("airquality")
AQ.raw = na.omit(airquality[,1:4])

### Construct new variables
AQ = AQ.raw
AQ$TWcp = with(AQ.raw, Temp * Wind)
AQ$TWrat = with(AQ.raw, Temp / Wind)

########################
### Helper Functions ###
########################

### Create function to compute MSPEs
get.MSPE = function(Y, Y.hat){
  return(mean((Y - Y.hat)^2))
}

### Create function which constructs folds for CV
### n is the number of observations, K is the number of folds
get.folds = function(n, K) {
  ### Get the appropriate number of fold labels
  n.fold = ceiling(n / K) # Number of observations per fold (rounded up)
  fold.ids.raw = rep(1:K, times = n.fold)
  fold.ids = fold.ids.raw[1:n]

  ### Shuffle the fold labels
  folds.rand = fold.ids[sample.int(n)]

  return(folds.rand)
}
```

Applications

Question 1

```
### Fit polynomial regression
fit.poly = lm(Ozone ~ poly(Temp, degree = 3), data = AQ)

### Fit cubic splines with various DF
fit.cub.5 = lm(Ozone ~ bs(Temp, df = 5), data = AQ)
fit.cub.7 = lm(Ozone ~ bs(Temp, df = 7), data = AQ)
fit.cub.9 = lm(Ozone ~ bs(Temp, df = 9), data = AQ)
fit.cub.20 = lm(Ozone ~ bs(Temp, df = 20), data = AQ)

### Create grid in Temp
data.temp = data.frame(Temp = seq(57, 97, by = 0.5))

### Get predictions
pred.poly = predict(fit.poly, data.temp)
pred.cub.5 = predict(fit.cub.5, data.temp)
pred.cub.7 = predict(fit.cub.7, data.temp)
pred.cub.9 = predict(fit.cub.9, data.temp)
pred.cub.20 = predict(fit.cub.20, data.temp)

### Construct plot
with(AQ, plot(Temp, Ozone, xlab = "Temperature"))
lines(data.temp$Temp, pred.poly, col = 1, lwd = 2)
lines(data.temp$Temp, pred.cub.5, col = 2, lwd = 2)
lines(data.temp$Temp, pred.cub.7, col = 3, lwd = 2)
lines(data.temp$Temp, pred.cub.9, col = 4, lwd = 2)
lines(data.temp$Temp, pred.cub.20, col = 5, lwd = 2)

### Add legend
legend("topleft", legend = c("Cubic Reg", "Cubic Spline - 5", "Cubic Spline - 7", "Cubic Spline - 9", "
```

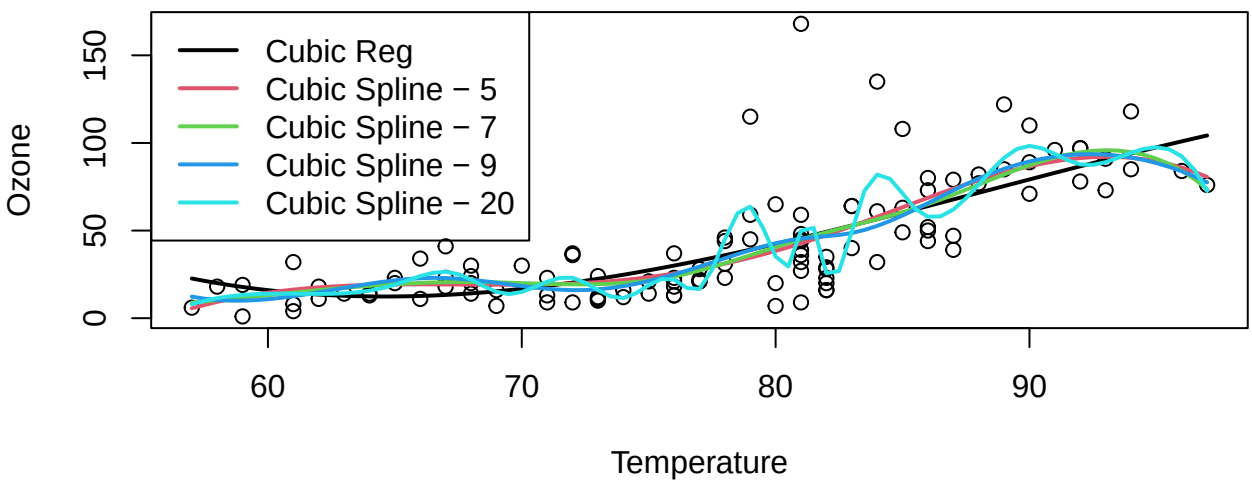

Figure 1: Cubic Regression and Cubic Splines

See Figure 1 for a comparison of cubic splines. The cubic regression model appears to have the most bias.

The cubic spline with 20 degrees of freedom appears to be overfitting because it is very wiggly. If I had to choose a single model, I would use the spline with 7 DF (the green curve) because it does a good job of following the trend that I see in the dataset without following the errors too much.

Question 2

Based on the results from question 1, I will use 5, 7, and 9 degrees of freedom. I think that these cover a reasonable range of flexibility without being too extreme toward bias or variance.

```
### Fit natural splines with various DF
fit.nat.5 = lm(Ozone ~ ns(Temp, df = 5), data = AQ)
fit.nat.7 = lm(Ozone ~ ns(Temp, df = 7), data = AQ)
fit.nat.9 = lm(Ozone ~ ns(Temp, df = 9), data = AQ)

### Get predictions
pred.nat.5 = predict(fit.nat.5, data.temp)
pred.nat.7 = predict(fit.nat.7, data.temp)
pred.nat.9 = predict(fit.nat.9, data.temp)

### Construct plot
with(AQ, plot(Temp, Ozone, xlab = "Temperature"))
lines(data.temp$Temp, pred.nat.5, col = 2, lwd = 2)
lines(data.temp$Temp, pred.nat.7, col = 3, lwd = 2)
lines(data.temp$Temp, pred.nat.9, col = 4, lwd = 2)

### Add legend
legend("topleft", legend = c("Natural Spline - 5", "Natural Spline - 7", "Natural Spline - 9"), col = 2
```

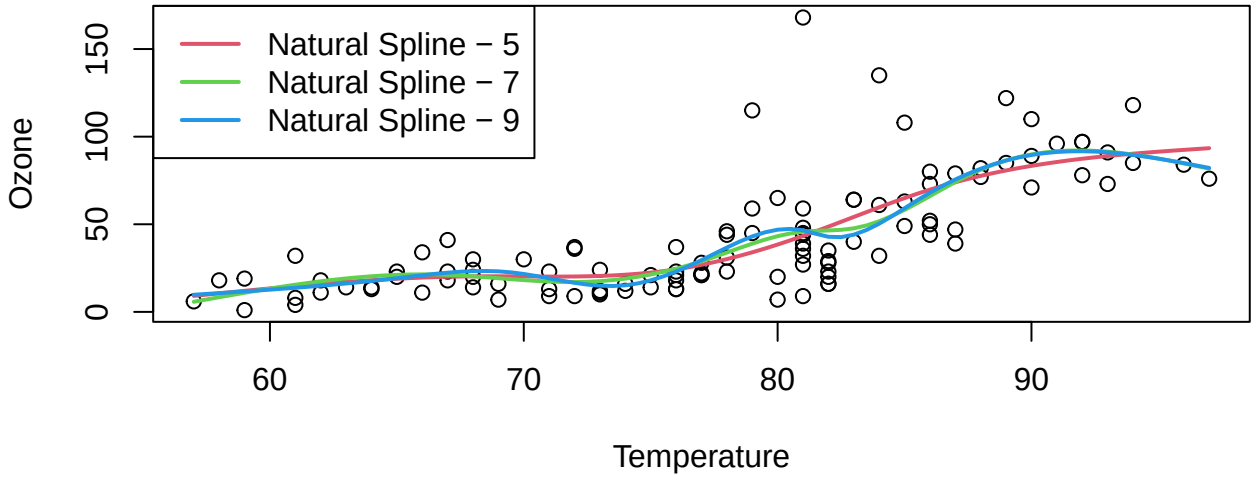

Figure 2: Natural Splines

See Figure 2 for natural splines fit with my chosen degrees of freedom. I now prefer the fit with 5 degrees of freedom, since the 7 DF fit curves down more than I would like around where temperature equals 82 degrees.