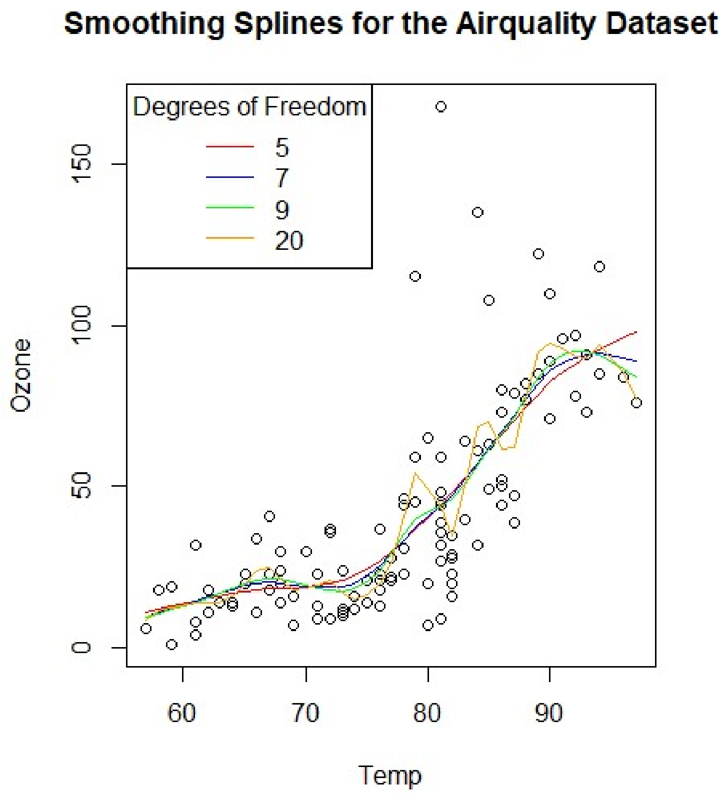
Refer to the Air Quality data described previously, and the analyses we have done with Ozone as the response variable, and the five explanatory variables (including the two engineered features).

1. Use smoothing splines to model the relationship between Ozone and Temp:
(a) On one graph, plot the data along with fits of smoothing splines with 5, 7, 9, and 20 DF.

i. **Present the plot. Be sure to add a legend and use different colours for the different functions**

```
library(dplyr)
library(MASS)   # For ridge regression
library(glmnet) # For LASSO
source("Helper Functions.R")
data = na.omit(airquality[, 1:4])
data$TWcp = data$Temp*data$Wind
data$TWrat = data$Temp/data$Wind


### Let's start with smoothing splines. Fit various degrees of freedom.
### We use the smooth.spline function to fit a smoothing spline. This function
### takes x and y specified as separate vectors. We can also set degrees of
### freedom using the df input.
fit.smooth.5 = smooth.spline(x = data$Temp, y = data$Ozone, df = 5)
fit.smooth.7 = smooth.spline(x = data$Temp, y = data$Ozone, df = 7)
fit.smooth.9 = smooth.spline(x = data$Temp, y = data$Ozone, df = 9)
fit.smooth.20 = smooth.spline(x = data$Temp, y = data$Ozone, df = 20)

### Plot the data, and add a legend to distinguish our splines
### We create legends using the legend() function. The first input is the
### position, which can be numeric (this is hard to get right), or a name, like
### "topright" or "center". The next inputs specify what we want our legend to
### say. See this tutorial's video for more details.

with(data, plot(Temp, Ozone,
                main = "Smoothing Splines for the Airquality Dataset"))
legend("topleft", title = "Degrees of Freedom", legend = c("5", "7", "9", "20"),
       col = c("red", "blue", "green", "orange"), lty = 1)


lines(fit.smooth.5, col = "red")
lines(fit.smooth.7, col = "blue")
lines(fit.smooth.9, col = "green")
lines(fit.smooth.20, col = "orange")
```



ii. If you had to choose one model, **which would it be? Why?**

```
#I would choose smoothing spliens with DF=5 because it's not too wigly and shows the trends well
```

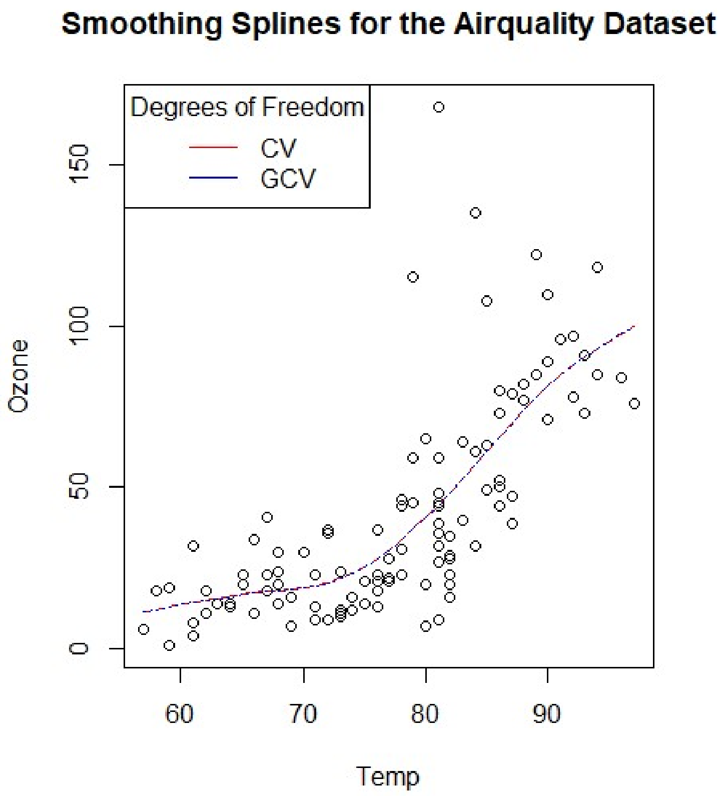
(b) Use cross-validation and generalized cross-validation to choose the optimal smoothing Amount

```
with(data, plot(Temp, Ozone,
                main = "Smoothing Splines for the Airquality Dataset"))
legend("topleft", title = "Degrees of Freedom", legend = c("5", "7", "9", "20"),
       col = c("red", "blue", "green", "orange"), lty = 1)


lines(fit.smooth.5, col = "red")
lines(fit.smooth.7, col = "blue")
lines(fit.smooth.9, col = "green")
lines(fit.smooth.20, col = "orange")

#I would choose smoothing spliens with DF=5 because it's not too wigly and shows the trends well

#b. Use cross-validation and generalized cross-validation to choose the optimal smoothing amount
### We can also fit smoothing splines using CV and GCV. Set cv to TRUE for
### CV and FALSE for GCV
fit.smooth.CV = smooth.spline(x = data$Temp, y = data$Ozone, cv=T)
fit.smooth.GCV = smooth.spline(x = data$Temp, y = data$Ozone, cv=F)

with(data, plot(Temp, Ozone,
                main = "Smoothing Splines for the Airquality Dataset"))
legend("topleft", title = "Degrees of Freedom", legend = c("CV", "GCV"),
       col = c("red", "blue"), lty = 1)
lines(fit.smooth.CV, lty = 2, col="red")
lines(fit.smooth.GCV, lty = 3, col="blue")
```

i. **How many DF does each method suggest to use?**

-> CV suggests to use 4.588 DF and GCV suggests to use DF 4.56

ii. **Show the fits on one plot with the data.**



iii. **Comment on the quality of each fit.**

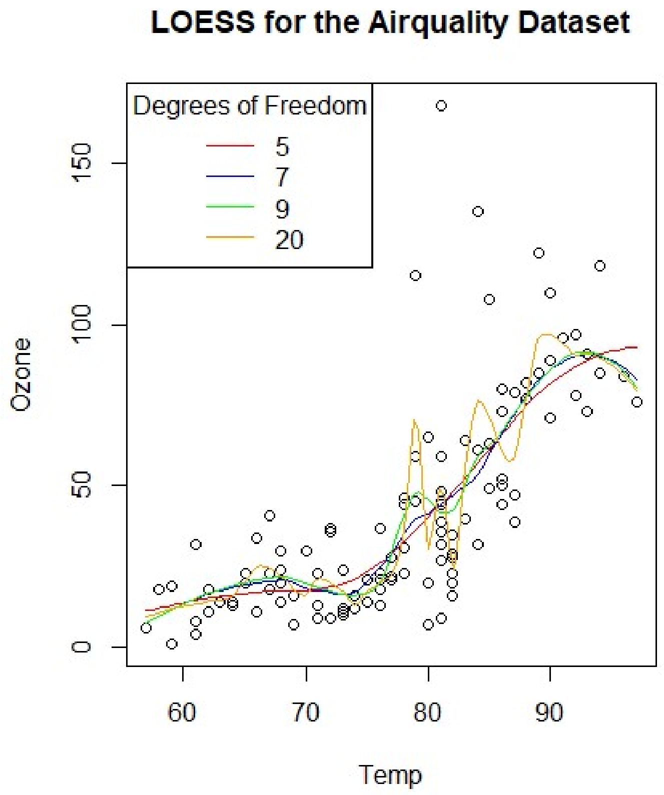
➔ Both are very similar and doing well in that it's not too wiggly but fits the trend.

2. Repeat part (a) from Exercise 1 using LOESS.

(a) On one graph, plot the data along with fits of LOESS with 5, 7, 9, and 20 DF.

i. **Present the plot. Be sure to add a legend and use different colours for the different functions**

```
### Now, let's move on to loess. We fit loess models using the loess()
### function. This function uses data frame & formula syntax instead
### of x & y vectors syntax. We can specify how many degrees of freedom
### to use with the enp.target input.
with(data, plot(Temp, Ozone,
                main = "LOESS for the Airquality Dataset"))
legend("topleft", title = "Degrees of Freedom", legend = c("5", "7", "9", "20"),
       col = c("red", "blue", "green", "orange"), lty = 1)


min.temp = min(data$Temp)
max.temp = max(data$Temp)
vals.temp.raw = seq(from = min.temp, to = max.temp, length.out = 100)
vals.temp = data.frame(Temp = vals.temp.raw)

fit.loess.5 = loess(Ozone ~ Temp, data = data, enp.target = 5)
fit.loess.7 = loess(Ozone ~ Temp, data = data, enp.target = 7)
fit.loess.9 = loess(Ozone ~ Temp, data = data, enp.target = 9)
fit.loess.20 = loess(Ozone ~ Temp, data = data, enp.target = 20)


pred.loess.5 = predict(fit.loess.5, vals.temp)
pred.loess.7 = predict(fit.loess.7, vals.temp)
pred.loess.9 = predict(fit.loess.9, vals.temp)
pred.loess.20 = predict(fit.loess.20, vals.temp)

lines(x = vals.temp$Temp, y = pred.loess.5, col = "red")
lines(x = vals.temp$Temp, y = pred.loess.7, col = "blue")
lines(x = vals.temp$Temp, y = pred.loess.9, col = "green")
lines(x = vals.temp$Temp, y = pred.loess.20, col = "orange")
```



ii. If you had to choose one model, **which would it be? Why?**

I would choose DF=5, because it’s not too wiggly and showing the trend well

```
fit.smooth.CV = smooth.spline(x = data$Temp, y = data$Ozone, cv=T)
fit.smooth.GCV = smooth.spline(x = data$Temp, y = data$Ozone, cv=F)
```