Assignment 4 Question 2 e)

Sheen Thusoo

01/12/2021

Based on your findings in parts (c) and (d), which degree polynomial has the best predictive accuracy? Construct a scatter plot – like the ones from (a) and (b) – but this time create just one plot, and overlay just the polynomial predictor function with the degree you identified as best.

Based on my findings in parts (c) and (d), the degree 5 polynomial has the best predictive accuracy. This is because it has a low APSE value of approximately 74 and it balances the trasde-off between bias and variance. It has a variance of around 3.5 and a bias of around 70. These values are low compared to the rest of the degrees while, when looking at the graph visually, fitting the data well.

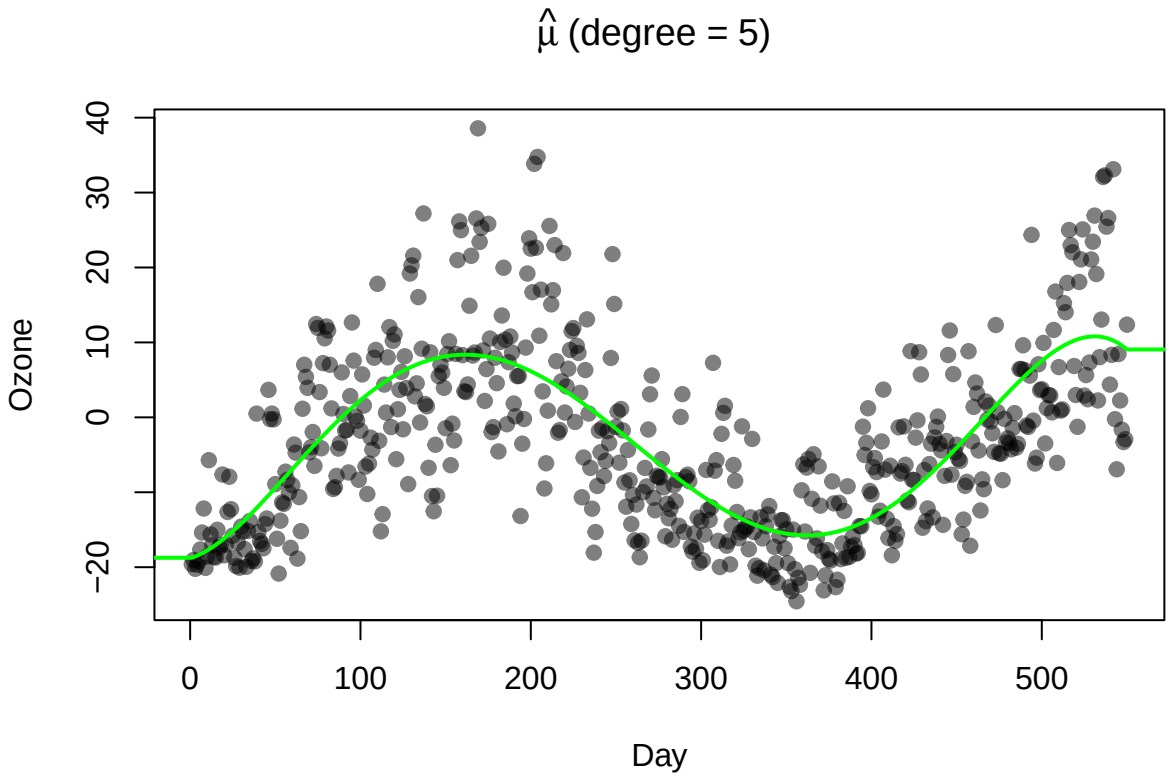
```
data <- read.csv("OzoneData.csv")
```

```
getXYpop <- function(xvarname, yvarname, pop) {
    popData <- pop[, c(xvarname, yvarname)]
    names(popData) <- c("x", "y")
    popData
}

getmuhat <- function(sampleXY, complexity = 1) {
  formula <- paste0("y ~ ",
                    if (complexity==0) {
                      "1"
                      } else
                      paste0("poly(x, ", complexity, ", raw = FALSE)")
                      #paste0("bs(x, ", complexity, ")")
  )

  fit <- lm(as.formula(formula), data = sampleXY)
  tx = sampleXY$x
  ty = fit$fitted.values

  range.X = range(tx)
  val.rY  = c( mean(ty[tx == range.X[1]]),
               mean(ty[tx == range.X[2]]) )

  ## From this we construct the predictor function
  muhat <- function(x){
    if ("x" %in% names(x)) {
      ## x is a dataframe containing the variate named
      ## by xvarname
```

```
      newdata <- x
    } else
      ## x is a vector of values that needs to be a data.frame
    { newdata <- data.frame(x = x) }
    ## The prediction
    ##
    val = predict(fit, newdata = newdata)
    val[newdata$x < range.X[1]] = val.rY[1]
    val[newdata$x > range.X[2]] = val.rY[2]
    val
  }
  ## muhat is the function that we need to calculate values
  ## at any x, so we return this function from getmuhat
  muhat
}


getmuFun <- function(pop, xvarname, yvarname){
  pop   = na.omit(pop[, c(xvarname, yvarname)])

  # rule = 2 means return the nearest y-value when extrapolating, same as above.
  # ties = mean means that repeated x-values have their y-values averaged, as above.
  tauFun = approxfun(pop[,xvarname], pop[,yvarname], rule = 2, ties = mean)
  return(tauFun)
}
```

```
plotfit <- function(muhat, complexity = NULL, color) {

    if (is.null(complexity))
        title = bquote(hat(mu) ~ "(Piecewise)") else title = bquote(hat(mu) ~
                                                                    "(degree =" ~ .(complexity) * ") ")

    plot(data, main = title, xlab = "Day",
        ylab = "Ozone", pch = 19, col = adjustcolor("black", 0.5))

    xlim = extendrange(data[, "Day"])
    curve(muhat, from = xlim[1], to = xlim[2], add = TRUE, col = color,
        lwd = 2, n = 1000)
}
```

```
ozone.data <- getXYpop(xvarname = "Day", yvarname = "Ozone", pop = data)

muhat = getmuhat(ozone.data, complexity=5)
plotfit(muhat, complexity=5, color="green")
```


$\hat{\mu}$ (degree = 5)
Ozone
40
30
20
10
0
−20
0
100
200
300
400
500
Day