Assignment 4 Question 2 a)

Sheen Thusoo

30/11/2021

Generate six scatter plots of the data in a 3×2 grid, where the plots are data as well as polynomials of degrees 1, 2, 5, 10, 15, and 20, respectively overlaid. Use the `getmuhat` function defined in the lectures to estimate these polynomial predictor functions. Use a different colour for each of the different degrees, and use a legend to indicate which degree polynomial is visualized in each plot.

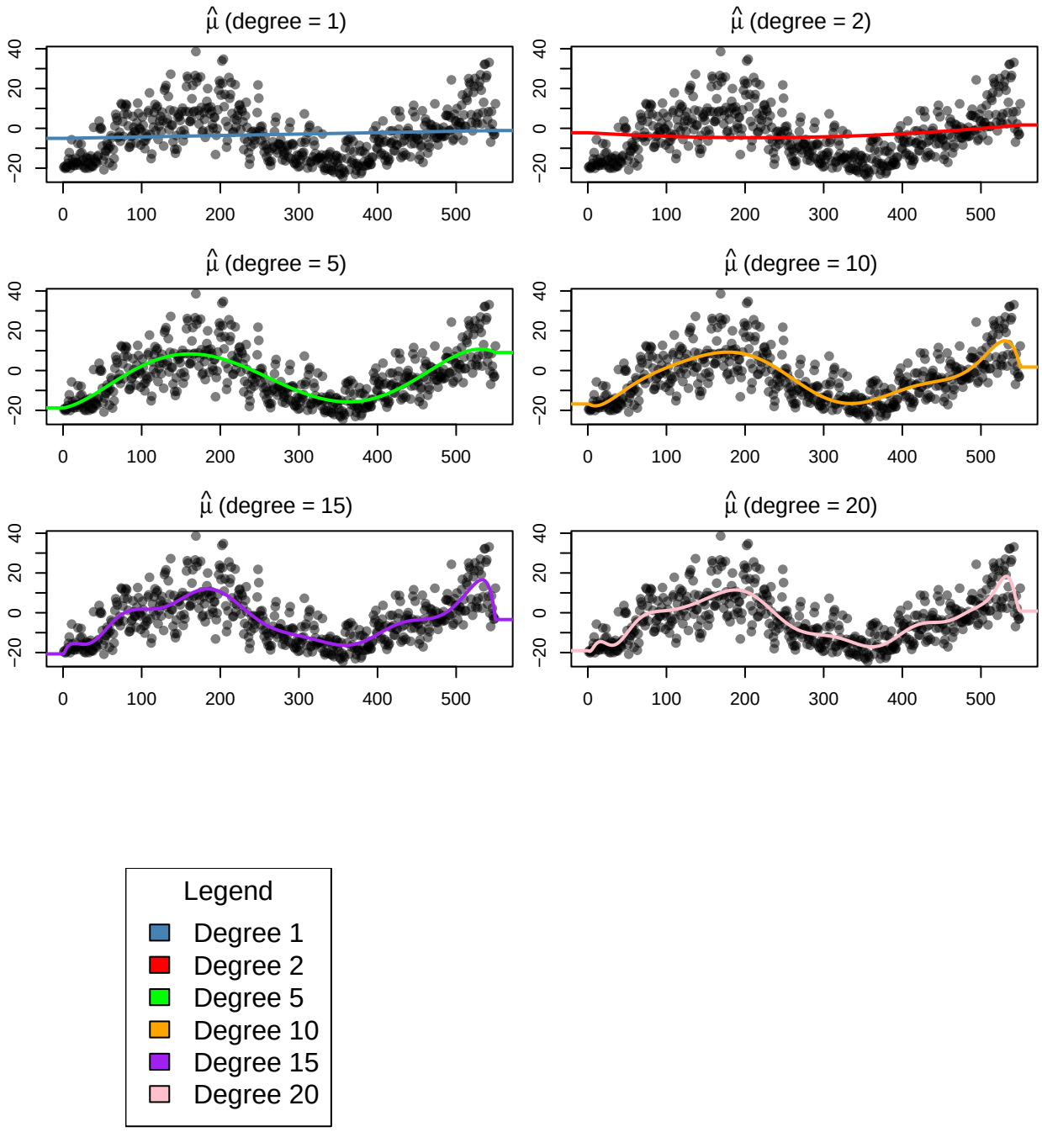
```
data <- read.csv("OzoneData.csv")
```

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

library(splines)

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)

par(mfrow = c(3, 2), mar = 2.5 * c(1, 1, 1, 0.1))

dset = c(1,2,5,10,15,20)
colors = c("steelblue", "red", "green", "orange", "purple", "pink")

muhats = lapply(dset, getmuhat, sampleXY = ozone.data)
for (i in 1:length(dset)) plotfit(muhats[[i]], complexity=dset[i], color=colors[i])
```


$\hat{\mu}$ (degree = 1)
$\hat{\mu}$ (degree = 2)
$\hat{\mu}$ (degree = 5)
$\hat{\mu}$ (degree = 10)
$\hat{\mu}$ (degree = 15)
$\hat{\mu}$ (degree = 20)
−20
0
20
40
0
100
200
300
400
500
Legend
Degree 1
Degree 2
Degree 5
Degree 10
Degree 15
Degree 20