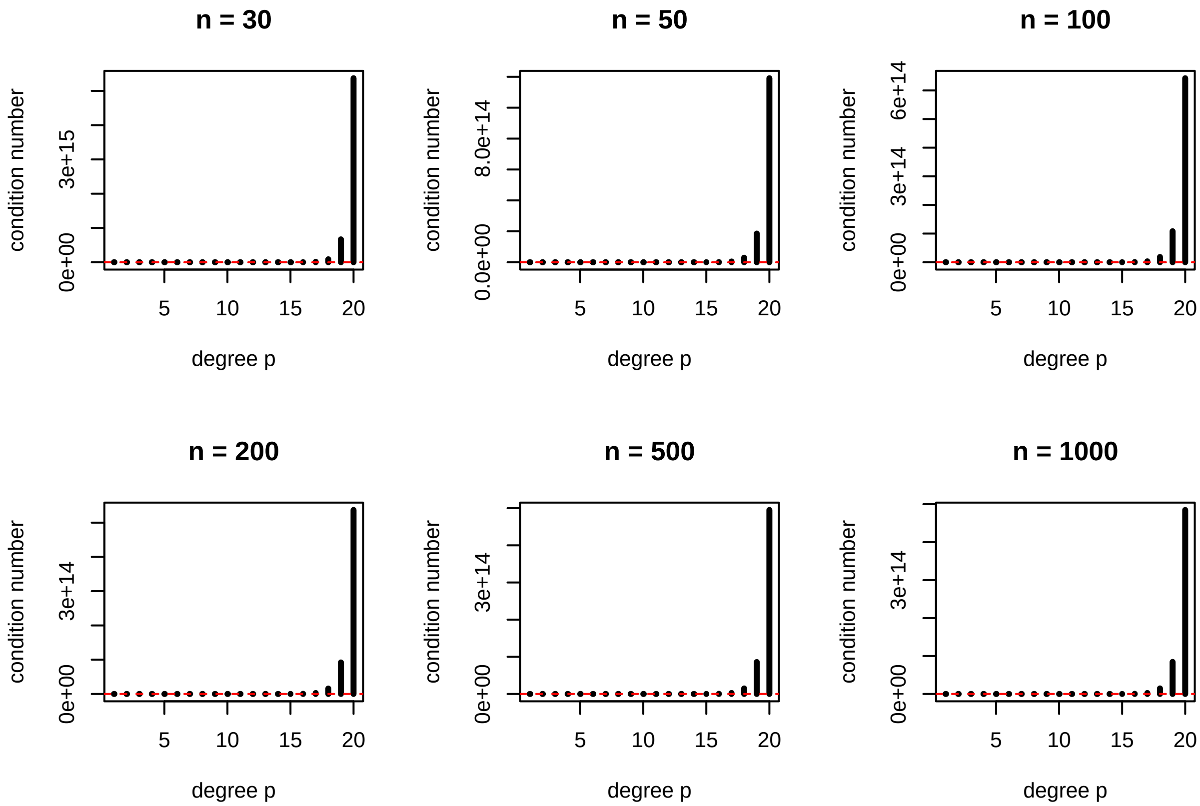
HW3

Sandra Villamar and Shobhit Dronamraju

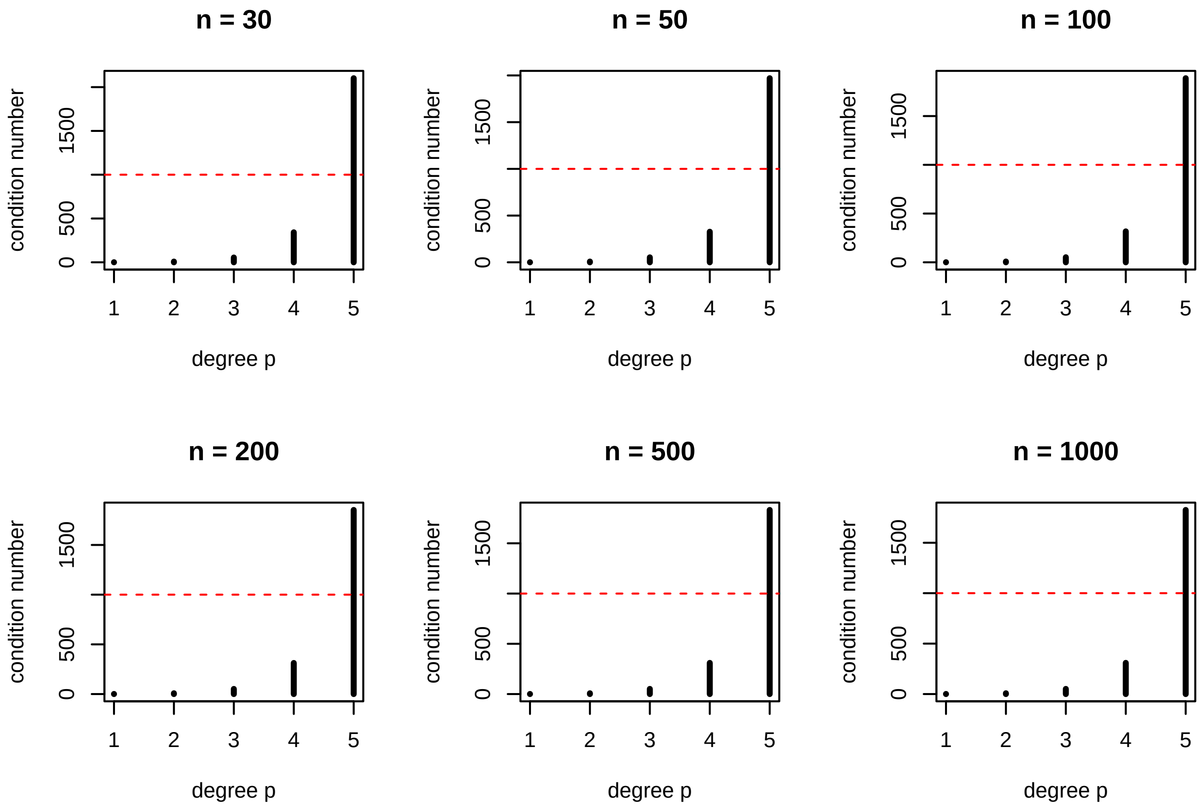
2023-02-02

Problem 1

```
ps = 1:20
ns = c(30, 50, 100, 200, 500, 1000)
Ks = matrix(, nrow = length(ns), ncol = length(ps))

for(n_i in 1:length(ns)){
  n = ns[n_i]
  for(p_i in 1:length(ps)){
    p = ps[p_i]

    # generate data
    xi = 1:n / (n + 1)
    X = poly(xi, degree = p, raw = TRUE)
    X = cbind(rep(1, n), X)  # add degree 0

    # compute condition number:
    # max(eigval) / min(eigval) of X^TX
    X[,-1] = scale(X[,-1])  # standardize X
    d = svd(X)$d
    K = max(d) / min(d)
    Ks[n_i, p_i] = K
  }
}

# plot
par(mfrow=c(2,3))
for(n_i in 1:length(ns)){
  plot(Ks[n_i,], type='h', lwd=3,
       xlab='degree p', ylab='condition number',
       main=sprintf("n = %i", ns[n_i]))
  abline(h = 1000, lty=2, col='red') # threshold for suspects
}
```




```r
# plot zoomed in version to see threshold crossing
par(mfrow=c(2,3))
for(n_i in 1:length(ns)){
  plot(Ks[n_i,1:5], type='h', lwd=3,
       xlab='degree p', ylab='condition number',
       main=sprintf("n = %i", ns[n_i]))
  abline(h = 1000, lty=2, col='red') # threshold for suspects
}
```



We do not see much variability between the different cases of n. On the other hand, for every case of n, we see that the larger the degree p gets, the higher the condition number. Since we standardized X, we use the rule of thumb threshold of 1000 and we see that by degree 5, every case becomes ill-conditioned. This makes sense, as the higher the degree, the more variance you are introducing into the model. This means small errors in the input can lead to high errors in the output, which is exactly what the condition number is measuring.

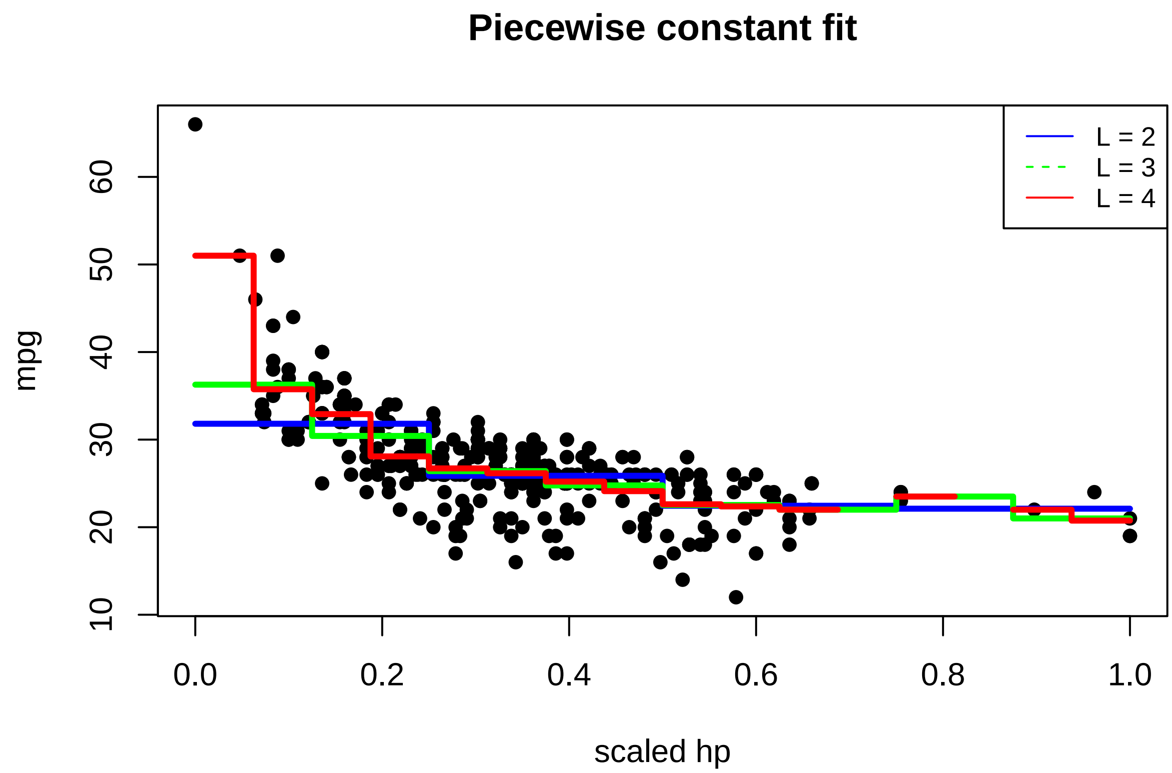
Problem 2A

```
piecewiseConstant <- function(x, y, L, plot = TRUE){

  K = seq(0, 1, length.out=2^L+1)  # interval points
  pts = rep(NA, 2*(2^L))  # values on x axis
  vals = rep(NA, 2*(2^L))  # values on y axis

  for(j in 1:length(K)){
    I = (K[j] < x) & (x <= K[j+1])  # interval
    if(any(I)){
      fit = lm(y[I] ~ 1)  # fit constant fn
      pts[2*j-1] = K[j]
      pts[2*j] = K[j+1]
      vals[2*j-1] = coef(fit)
      vals[2*j] = coef(fit)
    }
```

```
  }

  if(plot){
    plot(x, y, pch = 16, main="Piecewise constant fit", cex = 1)
    lines(pts, vals, col="blue", lwd = 3)
  }

  return(list("pts"=pts, "vals"=vals))
}
```

Problem 2B

```
load("../data/04cars.rda")  # load cars dataset to "dat"
dat = dat[,c(13,15,16,18,19)]  # extract selected variables
dat = dat[complete.cases(dat),]  # extract complete cases
names(dat) = c("hp","mpg","wt","len","wd")  # abbreviate names

x = (dat$hp - min(dat$hp)) / (max(dat$hp) - min(dat$hp))  # scale to [0,1]

plot(x, dat$mpg, pch = 16, main="Piecewise constant fit", cex = 1,
     xlab="scaled hp", ylab="mpg")
Ls = 2:4
colors = c("blue", "green", "red")
for(i in 1:length(Ls)){
  fit = piecewiseConstant(x, dat$mpg, Ls[i], plot = FALSE)
  lines(fit$pts, fit$vals, col=colors[i], lwd = 3)
}
legend("topright", legend=c("L = 2", "L = 3", "L = 4"),
       col=colors, lty=1:2, cex=0.8)
```



The greater the L, the more intervals we have, which leads to a finer model that captures more of the variance. This effect must be balanced, though, because too much variance could lead to overfitting while too little variance will not capture the trend of the observations well enough.