36401 Homework 5

```
library(tidyverse)
```

Problem 1

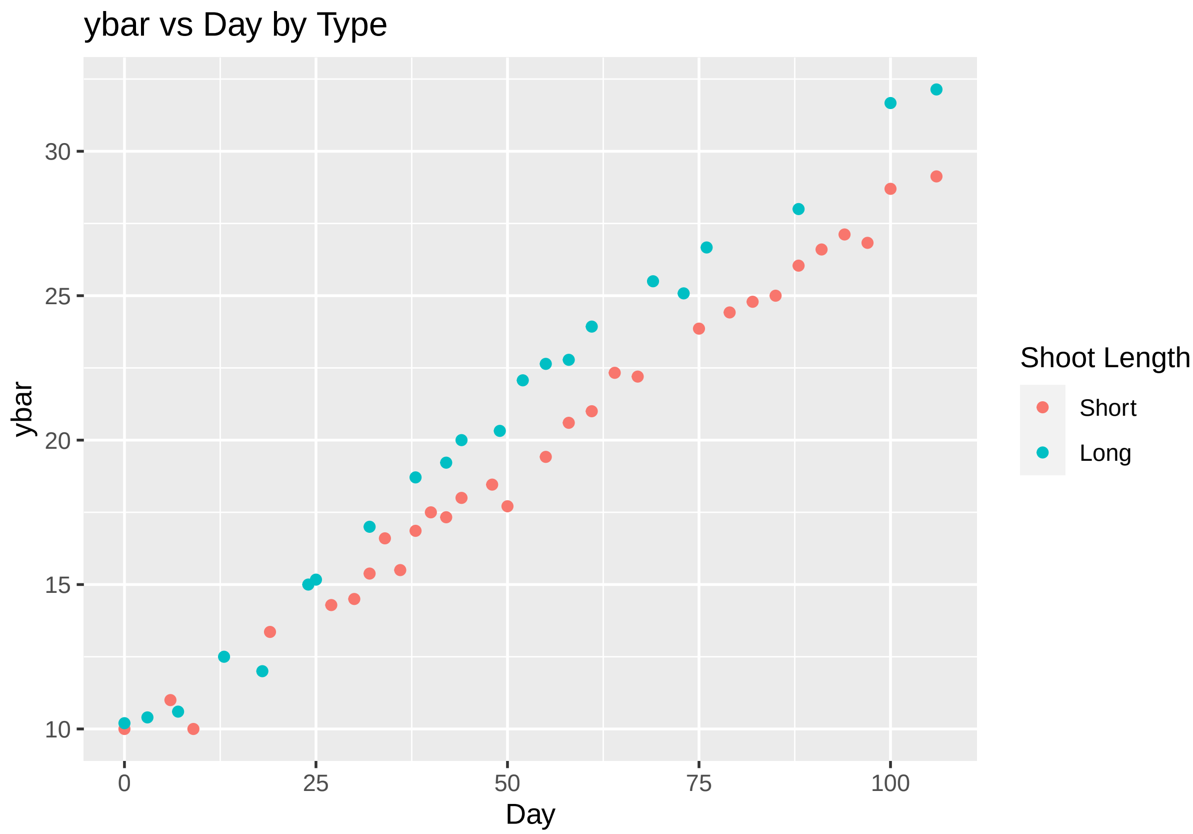
```
library(alr4)
```

```
attach(allshoots)
names(allshoots)
```

```
## [1] "Day"  "n"    "ybar" "SD"   "Type"
```

```
help(allshoots)
```

a.
```
ggplot(data = allshoots, aes(x = Day, y = ybar, color = as.factor(Type))) +
  geom_point() +
  scale_color_discrete(name = "Shoot Length", labels = c("Short", "Long")) +
  labs(title = "ybar vs Day by Type")
```



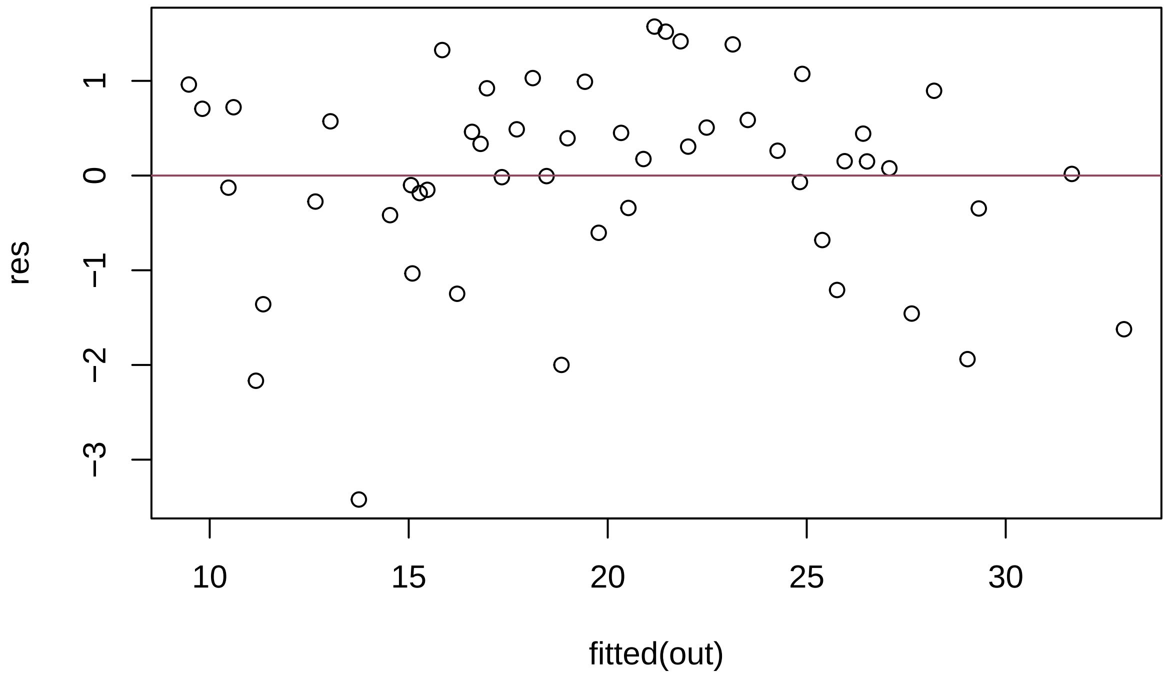
While there seems to be a positive linear correlation between `Day` and `ybar` for both types, observations with long shoots seem to have a higher slope than those with short shoots.

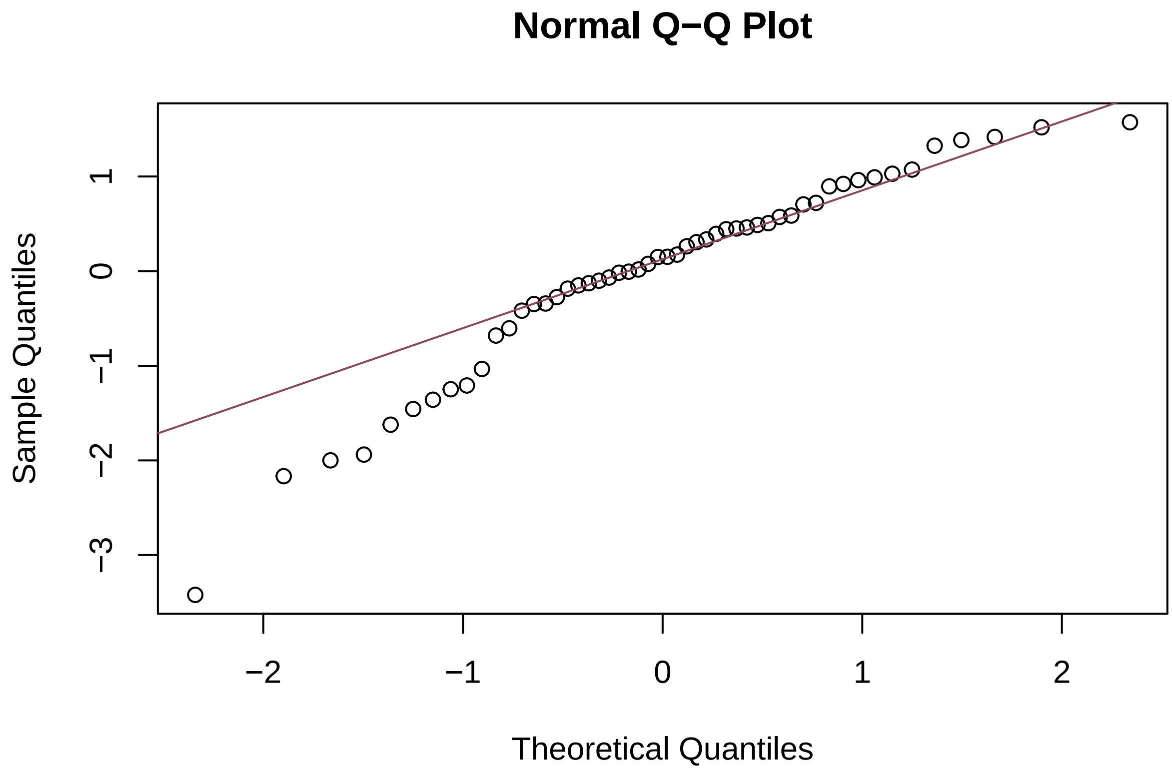
b.
```r
out1 = lm(ybar ~ Day * Type)
summary(out1)
```

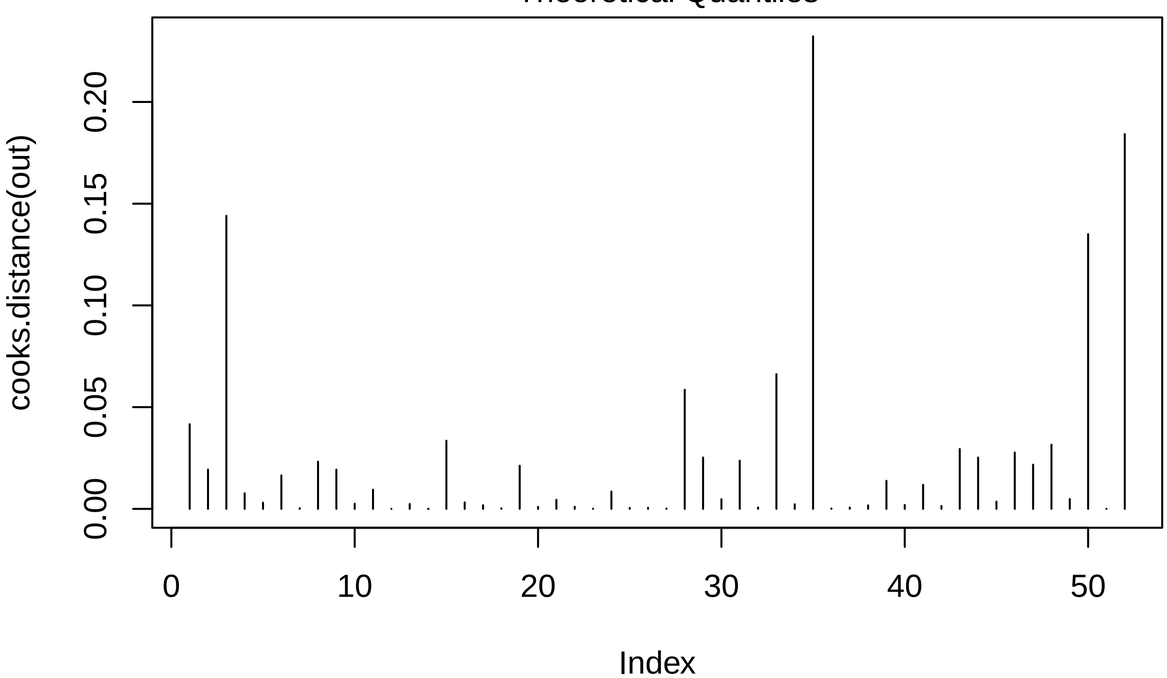
```
##
## Call:
## lm(formula = ybar ~ Day * Type)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.74747 -0.21000  0.08631  0.35212  0.89507
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9.475879   0.230981  41.025  < 2e-16 ***
## Day         0.187238   0.003696  50.655  < 2e-16 ***
## Type        0.339406   0.329997   1.029    0.309
## Day:Type    0.031217   0.005625   5.550 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5917 on 48 degrees of freedom
## Multiple R-squared:  0.9909, Adjusted R-squared:  0.9903
## F-statistic:  1741 on 3 and 48 DF,  p-value: < 2.2e-16
```

The model shows a strong ($R^2 = 99\%$), positive correlation between `Day` and `ybar`, with a significant interaction between `Day` and `Type` as well.

```
plot_resids = function(out, col = "palevioletred4") {
  res = rstudent(out)

  #Residual Plot
  plot(fitted(out), res)
  abline(h = 0, col = col)

  # QQ Plot
  qqnorm(res)
  qqline(res, col = col)

  # Cook's Distance Plot
  plot(cooks.distance(out), type="h")
}
plot_resids(out1)
```







The plot appears to show residuals with approximately equal variance across fitted values of `ybar`. They also seem to be centered around zero and don't follow any clear pattern. The residuals do not seem to be normally distributed, especially the ones in the lower theoretical quantiles. The Cook's distance plot doesn't appear to show any observations having a too large (>1) influence.

```
library(sandwich)
sandwich_confint = function(out_lm, width){
  alpha = (1 - width) / 2
  V = vcovHC(out_lm)
```

```r
  se = sqrt(diag(V))
  z = -qnorm(alpha/2)
  left = out_lm$coef - z*se
  right = out_lm$coef + z*se
  tmp = cbind(out_lm$coef, se, left, right)
  colnames(tmp) = c("Estimate","se","left","right")
  print(tmp)
}

sandwich_confint(out1, 0.80)
```

```
##               Estimate          se        left      right
## (Intercept) 9.47587933 0.255944489  9.05488811 9.89687055
## Day         0.18723815 0.003697817  0.18115578 0.19332052
## Type        0.33940568 0.393775435 -0.30829727 0.98710863
## Day:Type    0.03121657 0.006749807  0.02011413 0.04231901
```

For the intercept, the 80% confidence interval is [9.05488811, 9.89687055].

For `Day`, the 80% confidence interval is [0.18115578, 0.19332052].

For `Type`, the 80% confidence interval is [-0.30829727, 0.98710863].

For `Day:Type`, the 80% confidence interval is [0.02011413, 0.04231901].

c.

```r
out1_w <- lm(ybar ~ Day * Type, weights = n)
summary(out1_w)
```

```
##
## Call:
## lm(formula = ybar ~ Day * Type, weights = n)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2166 -0.8300  0.1597  0.9882  3.3196
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9.488374   0.238615  39.764  < 2e-16 ***
## Day         0.187258   0.003486  53.722  < 2e-16 ***
## Type        0.485380   0.362496   1.339    0.187
## Day:Type    0.030072   0.005800   5.185 4.28e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.675 on 48 degrees of freedom
## Multiple R-squared:  0.9906, Adjusted R-squared:  0.9901
## F-statistic:  1695 on 3 and 48 DF,  p-value: < 2.2e-16
```

```r
sandwich_confint(out1_w, 0.95)
```

```
##               Estimate          se        left      right
## (Intercept) 9.48837412 0.213766173  9.00923804 9.96751020
## Day         0.18725802 0.003049236  0.18042346 0.19409259
## Type        0.48537957 0.369744932 -0.34336773 1.31412687
## Day:Type    0.03007228 0.006281888  0.01599204 0.04415253
```

For the intercept, the 80% confidence interval is [9.00923804, 9.96751020].

For `Day`, the 80% confidence interval is [0.18042346 0.19409259].

For `Type`, the 80% confidence interval is [-0.34336773, 1.31412687].

For `Day:Type`, the 80% confidence interval is [0.01599204, 0.04415253].

The confidence intervals are wider for the weighted regression than for the unweighted regression, though this may also have to do with the confidence intervals having a higher confidence level of 95% vs 80%.

Problem 2

```r
library(alr4)
attach(BigMac2003)
names(BigMac2003)
```

```
##  [1] "BigMac"     "Bread"      "Rice"       "FoodIndex"  "Bus"
##  [6] "Apt"        "TeachGI"    "TeachNI"    "TaxRate"    "TeachHours"
```

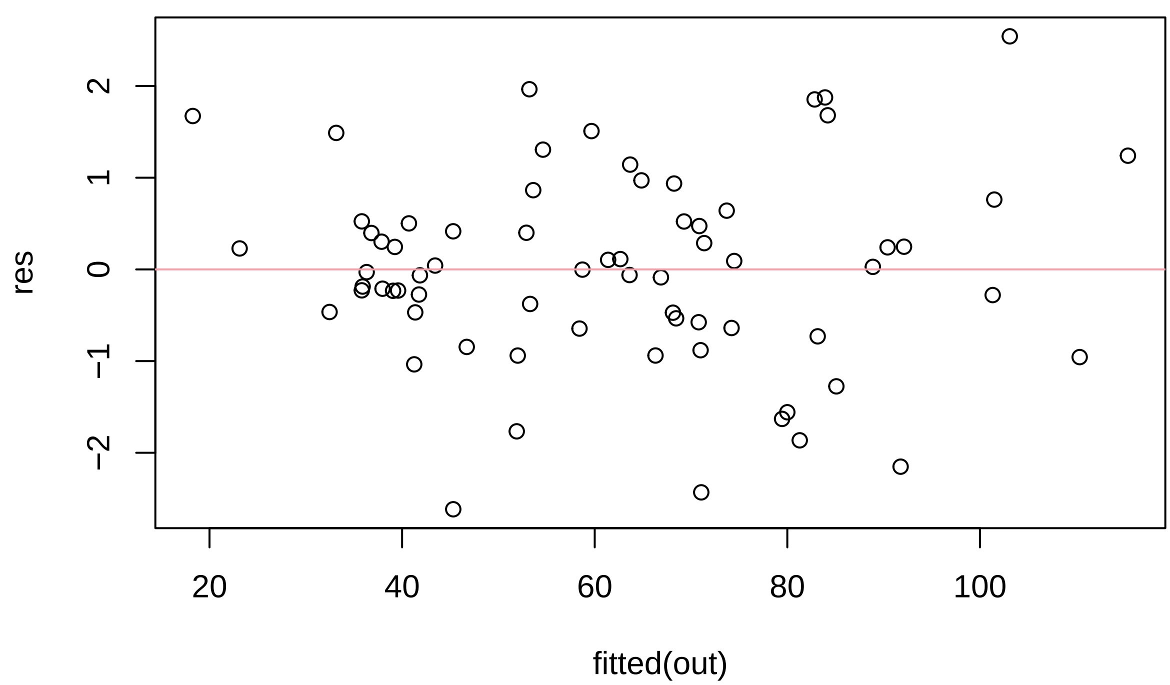
```r
help(BigMac2003)
```

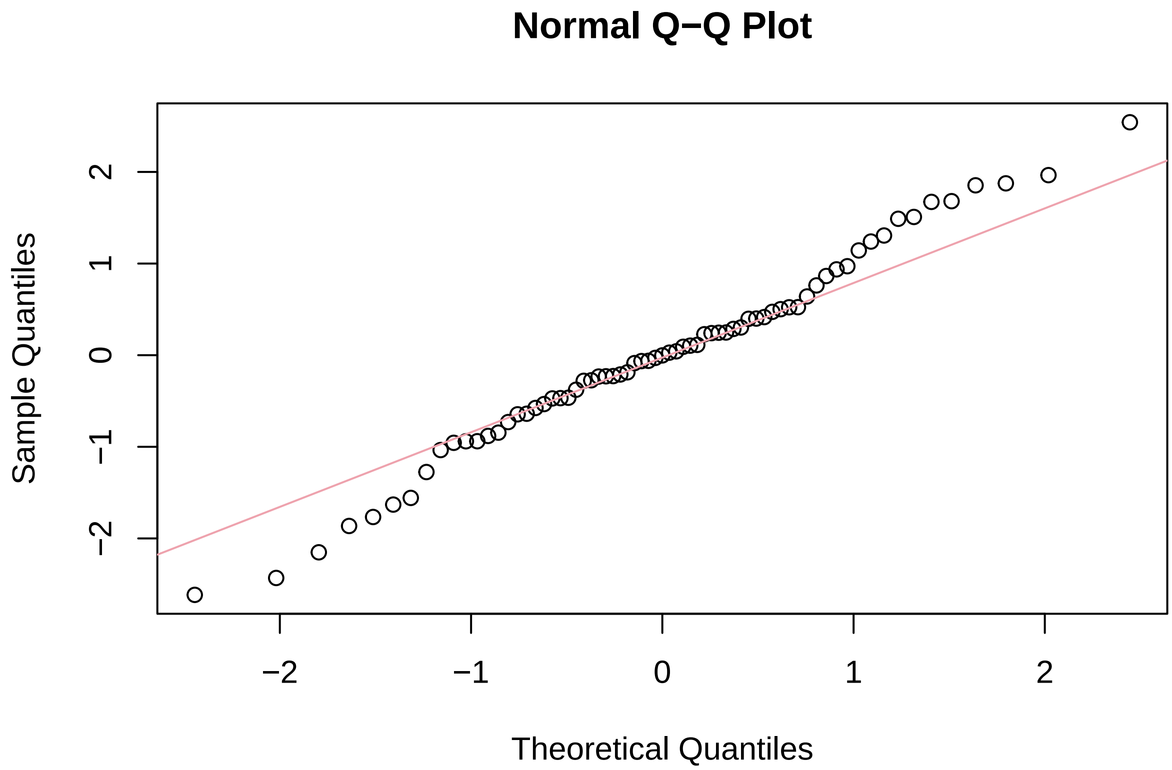
a.
```r
out2 = lm(FoodIndex ~ BigMac + Bread + Rice + Bus + Apt + TeachGI + TeachNI + TaxRate + TeachHours)
summary(out2)
```

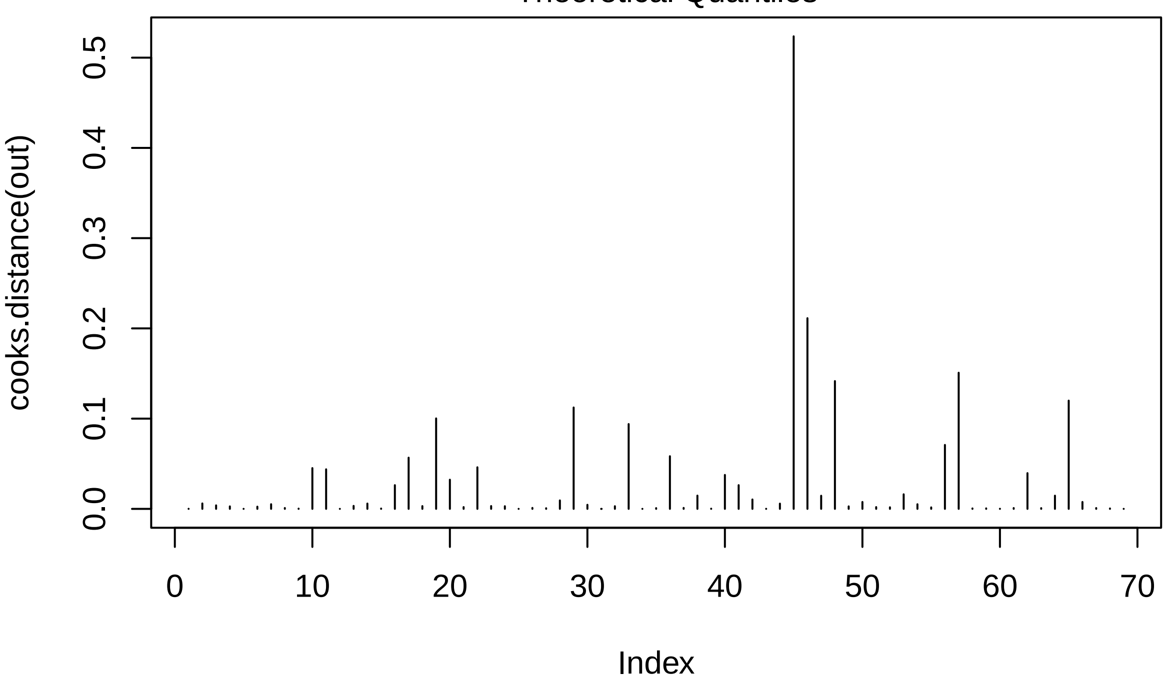
```
##
## Call:
## lm(formula = FoodIndex ~ BigMac + Bread + Rice + Bus + Apt +
##     TeachGI + TeachNI + TaxRate + TeachHours)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.0642  -6.3965  -0.0262   5.6928  26.3002
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.09968   11.19872  -0.098   0.9221
## BigMac      -0.20569    0.07798  -2.638   0.0107 *
## Bread        0.44383    0.10564   4.201 9.11e-05 ***
## Rice         0.26881    0.13597   1.977   0.0527 .
## Bus          3.59014    2.83317   1.267   0.2101
## Apt          0.01825    0.00434   4.204 9.02e-05 ***
```

```
## TeachGI     -0.97768    0.86750  -1.127   0.2643
## TeachNI      2.22275    1.13819   1.953   0.0556 .
## TaxRate      0.26530    0.25724   1.031   0.3066
## TeachHours   0.48015    0.20478   2.345   0.0224 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.86 on 59 degrees of freedom
## Multiple R-squared:  0.7981, Adjusted R-squared:  0.7673
## F-statistic: 25.91 on 9 and 59 DF,  p-value: < 2.2e-16
```

```r
plot_resids = function(out, col = "palevioletred4") {
  res = rstudent(out)

  #Residual Plot
  plot(fitted(out), res)
  abline(h = 0, col = col)

  # QQ Plot
  qqnorm(res)
  qqline(res, col = col)

  # Cook's Distance Plot
  plot(cooks.distance(out), type="h")
}
plot_resids(out2, col = "lightpink2")
```







The residuals appear to be random, centered around zero, and have a roughly equal variance across fitted values of `FoodIndex`. However, they don't necessarily seem to follow a normal distribution well, with some of the residuals diverging from the normal line in the Normal Q-Q Plot. None of the observations seem to have a too-large influence on the data based on the Cook's Distance plot.

b.
```
sandwich_confint = function(out_lm, width){
  alpha = (1 - width) / 2
```

```
  V = vcovHC(out_lm)
  se = sqrt(diag(V))
  z = -qnorm(alpha/2)
  left = out_lm$coef - z*se
  right = out_lm$coef + z*se
  tmp = cbind(out_lm$coef, se, left, right)
  colnames(tmp) = c("Estimate","se","left","right")
  print(tmp)
}
sandwich_confint(out2, 0.99)
```

```
##               Estimate          se          left       right
## (Intercept) -1.0996831 14.538753550 -41.910455301 39.71108903
## BigMac      -0.2056922  0.121622958  -0.547091936  0.13570756
## Bread        0.4438322  0.147985096   0.028433065  0.85923139
## Rice         0.2688081  0.311288199  -0.604988421  1.14260455
## Bus          3.5901381  4.592260279  -9.300491555 16.48076780
## Apt          0.0182453  0.005549614   0.002667345  0.03382325
## TeachGI     -0.9776817  1.044473704  -3.909554657  1.95419126
## TeachNI      2.2227541  1.397504249  -1.700087512  6.14559573
## TaxRate      0.2652971  0.238558093  -0.404343528  0.93493772
## TeachHours   0.4801551  0.253723001  -0.232053936  1.19236413
```

The 99% confidence interval for the `BigMac` covariate is [-0.5471, 0.1357]. This means that 99% of confidence intervals constructed in the same manner as above would contain the true value of `BigMac`. This is the true amount the Food Price Index increases when it takes 1 more minute of labor to purchase a Big Mac. Since the confidence interval contains zero, we can't be sure at the alpha = 0.005 significance level that `BigMac` is a significant predictor for `FoodIndex`.

c.
```
out2_bm = lm(FoodIndex ~ BigMac)
# summary(out2_bm)
anova(out2_bm, out2)
```

```
## Analysis of Variance Table
##
## Model 1: FoodIndex ~ BigMac
## Model 2: FoodIndex ~ BigMac + Bread + Rice + Bus + Apt + TeachGI + TeachNI +
##     TaxRate + TeachHours
##   Res.Df     RSS Df Sum of Sq     F    Pr(>F)
## 1     67 27532.9
## 2     59  8299.9  8     19233 17.09 8.026e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis is that the all of the coefficients other than `BigMac` are zero. Since the p-value is approximately zero, we have significant evidence to reject the null hypothesis. We have sufficient evidence to show that including all of the other covariates as predictors in addition to `BigMac` decreases error.

d.
```
pred_error = function(out_lm) {
  return(mean((resid(out_lm))^2 / (1 - (hatvalues(out_lm)))^2))
}
```

```
# Prediction error for all covariates
pred_error(out2)
```

```
## [1] 193.0971
```

```
# Prediction error for big mac-only model
pred_error(out2_bm)
```

```
## [1] 462.8564
```

The prediction error for for the model with only `BigMac` as a covariate is much higher than for the model with all of the covariates included. In conjunction with the F test from earlier, it seems clear that the model with all covariates included gives better predictions.

Problem 3

```
library(mlbench)
data(BreastCancer)
df = BreastCancer[complete.cases(BreastCancer), ]
df$Class = as.numeric(df$Class) - 1
df$Cl.thickness = as.factor(df$Cl.thickness)
df$Bl.cromatin = as.factor(df$Bl.cromatin)
```

```
out3 = glm(Class ~ Cl.thickness + Cell.size + Cell.shape + Marg.adhesion +
             Epith.c.size + Bare.nuclei, data = df, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(out3)
```

```
##
## Call:
## glm(formula = Class ~ Cl.thickness + Cell.size + Cell.shape +
##     Marg.adhesion + Epith.c.size + Bare.nuclei, family = "binomial",
##     data = df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.008   0.000   0.000   0.000   1.467
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)         34.1527  6186.2156   0.006    0.996
## Cl.thickness.L      68.8230  5753.4447   0.012    0.990
## Cl.thickness.Q      32.7521  3853.4042   0.008    0.993
## Cl.thickness.C      18.7497  3684.1200   0.005    0.996
## Cl.thickness^4      -1.1942  5693.4669   0.000    1.000
## Cl.thickness^5      18.2303  7137.8670   0.003    0.998
```

```
## Cl.thickness^6      1.4076  5862.0001   0.000    1.000
## Cl.thickness^7     -6.8861  3950.9825  -0.002    0.999
## Cl.thickness^8    -35.9175  3708.0154  -0.010    0.992
## Cl.thickness^9     -2.2800  2446.8855  -0.001    0.999
## Cell.size.L         1.8264 44041.0827   0.000    1.000
## Cell.size.Q        -6.7347 22447.2883   0.000    1.000
## Cell.size.C        28.1968 19186.6306   0.001    0.999
## Cell.size^4        33.2267 46489.3609   0.001    0.999
## Cell.size^5        37.2971 56045.7054   0.001    0.999
## Cell.size^6        -2.7392 47892.2153   0.000    1.000
## Cell.size^7        17.0590 30932.5500   0.001    1.000
## Cell.size^8        23.0155 14943.9837   0.002    0.999
## Cell.size^9       -37.0380  8483.8289  -0.004    0.997
## Cell.shape.L       45.0476 43425.9201   0.001    0.999
## Cell.shape.Q      -16.4212 22048.4713  -0.001    0.999
## Cell.shape.C        2.7423 19143.4087   0.000    1.000
## Cell.shape^4       10.0018 45536.4308   0.000    1.000
## Cell.shape^5      -26.8050 54985.5267   0.000    1.000
## Cell.shape^6      -29.6416 46869.3868  -0.001    0.999
## Cell.shape^7      -28.8389 30100.3992  -0.001    0.999
## Cell.shape^8        0.9091 14313.2477   0.000    1.000
## Cell.shape^9      -16.0036  4462.6534  -0.004    0.997
## Marg.adhesion.L    59.3809 12141.6199   0.005    0.996
## Marg.adhesion.Q    27.2403  6991.9637   0.004    0.997
## Marg.adhesion.C   -15.6598  7770.2763  -0.002    0.998
## Marg.adhesion^4   -27.0696 12650.6173  -0.002    0.998
## Marg.adhesion^5   -12.8821 16674.2950  -0.001    0.999
## Marg.adhesion^6   -12.0361 13802.6775  -0.001    0.999
## Marg.adhesion^7    -1.7179  9863.3184   0.000    1.000
## Marg.adhesion^8     8.1805 10220.5200   0.001    0.999
## Marg.adhesion^9     7.2190  6838.1522   0.001    0.999
## Epith.c.size.L      6.7477 19045.6944   0.000    1.000
## Epith.c.size.Q     13.7847  9483.2697   0.001    0.999
## Epith.c.size.C      3.0502  7624.2626   0.000    1.000
## Epith.c.size^4     13.4066 19370.3879   0.001    0.999
## Epith.c.size^5    -20.1535 23772.9094  -0.001    0.999
## Epith.c.size^6     -7.4764 20297.7247   0.000    1.000
## Epith.c.size^7      8.0911 13107.4216   0.001    1.000
## Epith.c.size^8     24.9940  6884.7061   0.004    0.997
## Epith.c.size^9     21.7454  2909.4821   0.007    0.994
## Bare.nuclei2        1.7273     3.9297   0.440    0.660
## Bare.nuclei3       27.5362    19.9990   1.377    0.169
## Bare.nuclei4       32.4423    23.3202   1.391    0.164
## Bare.nuclei5       17.5589    13.7607   1.276    0.202
## Bare.nuclei6       33.7788 27318.7180   0.001    0.999
## Bare.nuclei7       27.5633    19.2736   1.430    0.153
## Bare.nuclei8       -6.1103     6.9944  -0.874    0.382
## Bare.nuclei9       57.4281 19077.2296   0.003    0.998
## Bare.nuclei10      12.0700     8.0537   1.499    0.134
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 884.350  on 682  degrees of freedom
## Residual deviance:  36.473  on 628  degrees of freedom
```

```
## AIC: 146.47
##
## Number of Fisher Scoring iterations: 22
```

The fitted model is shown above.

```
p = predict(out3,type="link")
y = df$Class
n = length(y)
yhat = rep(0,n)
yhat[p >= .5] = 1
T = table(y,yhat)
training_error = (T[1,2] + T[2,1])/sum(T)
print(training_error)
```

```
## [1] 0.01756955
```

The proportion of misclassifications is 0.0176.

Problem 4

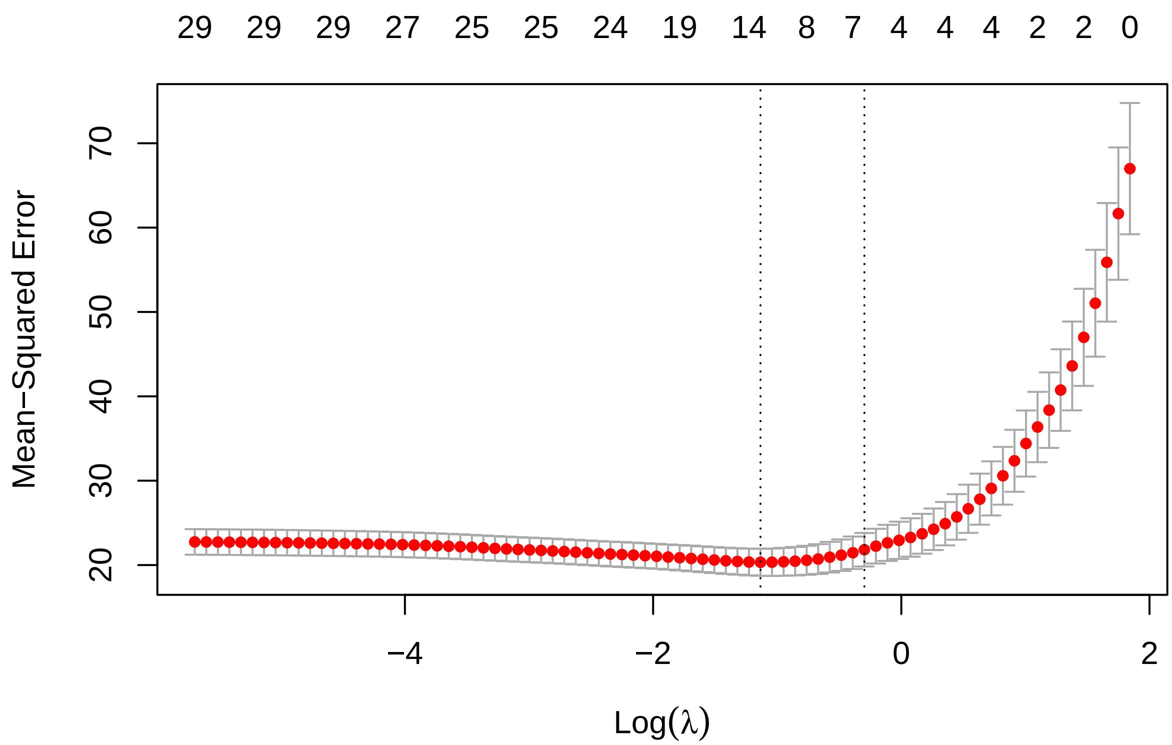
```
library(mlbench)
data(Ozone)
help(Ozone)
```

```
Ozone = Ozone[complete.cases(Ozone), ]
#Ozone = subset(Ozone.m, select = -c(V2, V3))
Ozone$V1 = as.numeric(Ozone$V1)
Ozone$month = Ozone$V1
Ozone$oz = Ozone$V4
Ozone$pressureh = Ozone$V5
Ozone$wind = Ozone$V6
Ozone$humidity = Ozone$V7
Ozone$tempS = Ozone$V8
Ozone$tempE = Ozone$V9
Ozone$invHeight = Ozone$V10
Ozone$pressureg = Ozone$V11
Ozone$invTemp = Ozone$V12
Ozone$visibility = Ozone$V13
# Ozone.x = select(Ozone, select = -oz)
# Ozone.x = select(Ozone, select = -V1:V13)
# just take named columns
I = c(14, 16:24)
X = data.matrix(Ozone[,I])
Y = data.matrix(Ozone[,15])

n = nrow(X)
fake = rnorm(20*n)
fake = matrix(fake,n,20)
X = cbind(X,fake)
```

a.
```r
library(glmnet)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-2
```

```r
out4 = cv.glmnet(X, Y, alpha = 1)
plot(out4)
```




```r
coefs4 = coef(out4, s="lambda.min")
print(coefs4)
```

```
## 31 x 1 sparse Matrix of class "dgCMatrix"
##                        s1
## (Intercept) -17.146166247
## month        -0.226098431
## pressureh     .
## wind          .
## humidity      0.098872830
## tempS         0.127823414
## tempE         0.308953574
```

```
## invHeight    -0.000261416
## pressureg     .
## invTemp       .
## visibility   -0.002346410
##               0.047788463
##               .
##               .
##               0.029818168
##               .
##               .
##               .
##               .
##               .
##               .
##              -0.565161779
##               .
##               .
##               .
##               .
##              -0.067671907
##               .
##               0.327494032
##               .
##               .
```

```
  # plot(b, type="h")
  # lambda1 = out$lambda.min
  # fit = glmnet(X,Y,alpha=1,lambda=lambda1)

  # let's predict some new data
  # a = max(abs(Ynew))
  # newfit = predict(fit,newx=Xnew,xlim=c(-a,a),ylim=c(-a,a))
  # plot(Ynew,newfit)
  # abline(a=0,b=1)

# lasso_reg(X, Y)
```

The variables in the final model are month, humidity, temperature at Sandburg, temperature at El Monte, and visibility. Since the model started with 29 variables and went up to 30 according to the plot, it seems that the fake variables were in the model first since there were only 10 real variables.

b.
```
lasso = function(X,Y){
     tmp = cv.glmnet(X,Y)
     lambdaCV = tmp$lambda.min
     out = glmnet(X,Y,lambda=lambdaCV)
     return(out)
}

lasso_CI = function(X,Y,j){
     ### confidence interval for beta[j]
     ### in high dimensional regression
     Z = X[,-j]
     X = X[,j]
```

```r
    tmp1 = lasso(Z,Y)
    beta = as.matrix(as.numeric(tmp1$beta))
    fitted = Z %*% beta
    R = Y - fitted
    tmp2 = lasso(Z,X)
    beta = as.matrix(as.numeric(tmp2$beta))
    fitted = Z %*% beta
    S = X - fitted
    beta.hat = sum(R*S)/sum(S^2)
    sigma = sqrt(mean((R - beta.hat*S)^2))
    se = sigma/sqrt(sum(X^2))
    return(list(beta.hat=beta.hat,se=se))
}
lci = lasso_CI(X, Y, 5)
```

```r
z = -qnorm((1 - 0.9)/2)
lower = lci$beta.hat - z * lci$se
upper = lci$beta.hat + z * lci$se
paste("[", lower, ", ", upper, "]", sep = "")
```

```
## [1] "[0.364491614174825, 0.380672146895094]"
```

So the confidence interval is $\hat{\beta}_8 \pm z_{\alpha/2}\hat{se} = [0.310,\, 0.326]$.