ARE 212, Problem Set #2

Anaya Hall & Christian Miller

Part 1: Theory

(for practice only)

Part 2: Applied - Returns to Scale in Electricity Supply

First, load our OLS function created in Problem Set #1. We're including a built in t-test this time around.

```
# Function to convert tibble, data.frame, or tbl_df to matrix
to_matrix <- function(the_df, vars) {
  # Create a matrix from variables in var
  new_mat <- the_df %>%
    #Select the columns given in 'vars'
    select_(.dots = vars) %>%
    # Convert to matrix
    as.matrix()
  # Return 'new_mat'
  return(new_mat)
}

ols <- function(data, y_data, X_data, intercept = T, H0 = 0, two_tail = T, alpha = 0.05) {
  # Function setup ----
    # Require the 'dplyr' package
    require(dplyr)

  # Create dependent and independent variable matrices ----
    # y matrix
    y <- to_matrix (the_df = data, vars = y_data)
    # X matrix
    X <- to_matrix (the_df = data, vars = X_data)
      # If 'intercept' is TRUE, then add a column of ones
      if (intercept == T) {
      X <- cbind(1,X)
      colnames(X) <- c("intercept", X_data)
      }

  # Calculate b, y_hat, and residuals ----
    b <- solve(t(X) %*% X) %*% t(X) %*% y
    y_hat <- X %*% b
    e <- y - y_hat

  # Useful -----
    n <- nrow(X) # number of observations
    k <- ncol(X) # number of independent variables
```

```r
  dof <- n - k # degrees of freedom
  i <- rep(1,n) # column of ones for demeaning matrix
  A <- diag(i) - (1 / n) * i %*% t(i) # demeaning matrix
  y_star <- A %*% y # for SST
  X_star <- A %*% X # for SSM
  SST <- drop(t(y_star) %*% y_star)
  SSM <- drop(t(b) %*% t(X_star) %*% X_star %*% b)
  SSR <- drop(t(e) %*% e)

# Measures of fit and estimated variance ----
  R2uc <- drop((t(y_hat) %*% y_hat)/(t(y) %*% y)) # Uncentered R^2
  R2 <- 1 - SSR/SST # Uncentered R^2
  R2adj <- 1 - (n-1)/dof * (1 - R2) # Adjusted R^2
  AIC <- log(SSR/n) + 2*k/n # AIC
  SIC <- log(SSR/n) + k/n*log(n) # SIC
  s2 <- SSR/dof  # s^2

# Measures of fit table ----
  mof_table_df <- data.frame(R2uc, R2, R2adj, SIC, AIC, SSR, s2)
  mof_table_col_names <- c("$R^2_\\text{uc}$", "$R^2$",
                           "$R^2_\\text{adj}$",
                           "SIC", "AIC", "SSR", "$s^2$")
  mof_table <-  mof_table_df %>% knitr::kable(
    row.names = F,
    col.names = mof_table_col_names,
    format.args = list(scientific = F, digits = 4),
    booktabs = T,
    escape = F
  )

# t-test----
  # Standard error
  se <- as.vector(sqrt(s2 * diag(solve(t(X) %*% X))))
  # Vector of _t_ statistics
  t_stats <- (b - H0) / se
  # Calculate the p-values
  if (two_tail == T) {
  p_values <- pt(q = abs(t_stats), df = dof, lower.tail = F) * 2
  } else {
    p_values <- pt(q = abs(t_stats), df = dof, lower.tail = F)
  }
  # Do we (fail to) reject?
  reject <- ifelse(p_values < alpha, reject <- "Reject", reject <- "Fail to Reject")

  # Nice table (data.frame) of results
  ttest_df <- data.frame(
    # The rows have the coef. names
    effect = rownames(b),
    # Estimated coefficients
    coef = as.vector(b) %>% round(3),
```

```r
      # Standard errors
      std_error = as.vector(se) %>% round(3),
      # t statistics
      t_stat = as.vector(t_stats) %>% round(3),
      # p-values
      p_value = as.vector(p_values) %>% round(4),
      # reject null?
      significance = as.character(reject)
      )

    ttest_table <-  ttest_df %>% knitr::kable(
      booktabs = T,
      format.args = list(scientific = F),
      escape = F
    )

  # Data frame for exporting for y, y_hat, X, and e vectors ----
    export_df <- data.frame(y, y_hat, e, X) %>% tbl_df()
    colnames(export_df) <- c("y","y_hat","e",colnames(X))

  # Return ----
    return(list(n=n, dof=dof, b=b, vars=export_df, R2uc=R2uc,R2=R2,
                R2adj=R2adj, AIC=AIC, SIC=SIC, s2=s2, SST=SST, SSR=SSR,
                mof_table=mof_table, ttest=ttest_table))
}
```

We'll also need a function to do an F-test for this Problem Set.

```r
# Joint test function (from Ed's notes).
# Could also write a more complex functions that takes R and r.

F_test <- function(data, y_var, X_vars) {
  # Turn data into matrices
  y <- to_matrix(data, y_var)
  X <- to_matrix(data, X_vars)
  # Add intercept
  X <- cbind(1, X)
  # Name the new column "intercept"
  colnames(X) <- c("intercept", X_vars)
  # Calculate n and k for degrees of freedom
  n <- nrow(X)
  k <- ncol(X)
  # J is k-1
  J <- k - 1
  # Create the R matrix: bind a column of zeros
  # onto a J-by-J identity matrix
  R <- cbind(0, diag(J))

  # Estimate coefficients
  b <- ols(data, y_var, X_vars)
  # Retrieve OLS residuals
  e <- b$vars$e
  # Retrieve s^2
  s2 <- b$s2 %<>% as.numeric()

  # Create the inner matrix R(X'X)^(-1)R'
  RXXR <- R %*% solve(t(X) %*% X) %*% t(R)
  # Calculate the F stat
  f_stat <- t(R %*% b$b) %*% solve(RXXR) %*% (R %*% b$b) / J / s2
  # Calculate the p-value;; why normal and not chi^2
  p_value <- pf(q = f_stat, df1 = J, df2 = n-k, lower.tail = F)
  # Create a data.frame of the f stat. and p-value
  results <- data.frame(
    f_stat = f_stat %>% as.vector(),
    p_value = p_value %>% as.vector())
  return(results)

}
```

Question 1:

Read the data into R. Inspect it. Sort by size (Q (kwh)).

```
nerlove <- readxl::read_excel("nerlove.xls", col_names=T)
summary(nerlove)
```

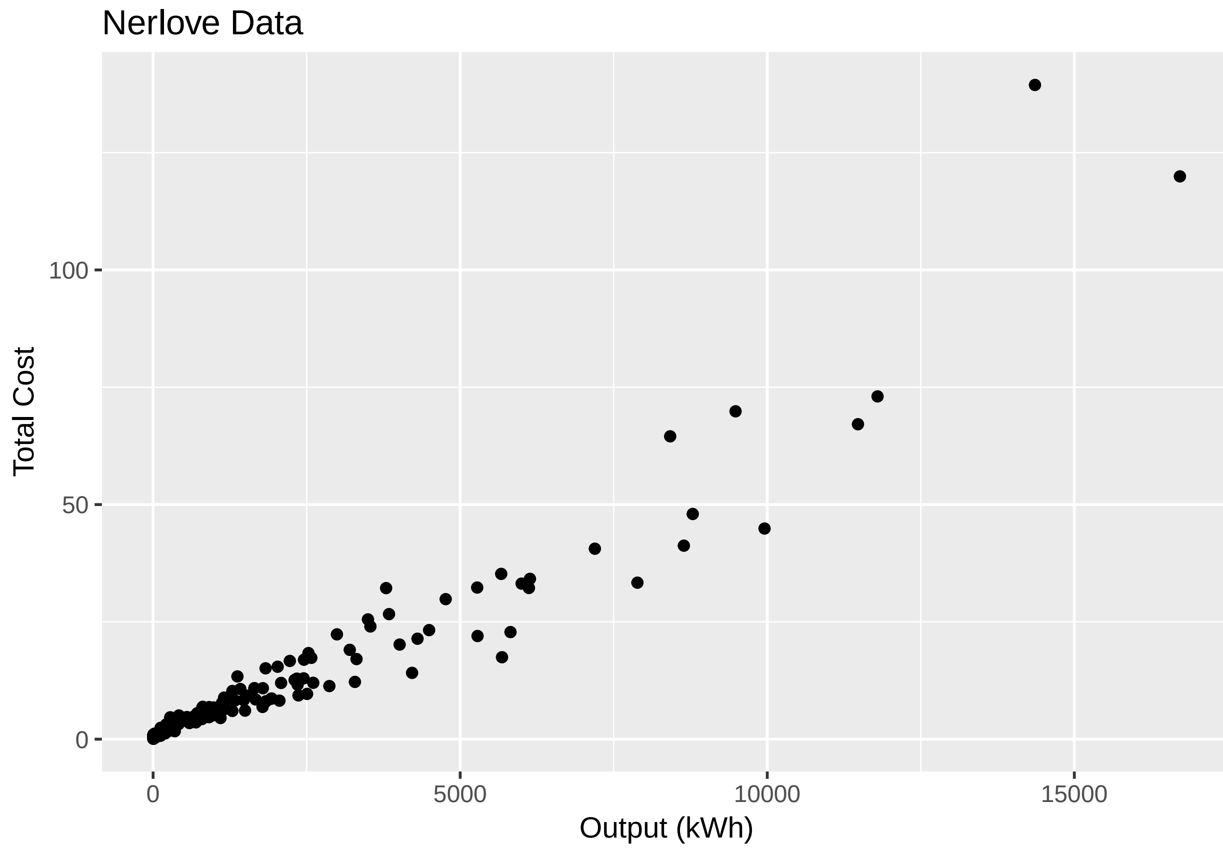
```
##        TC                Q                PL                PF
##  Min.   :  0.082   Min.   :    2   Min.   :  1.450   Min.   :10.30
##  1st Qu.:  2.382   1st Qu.:  279   1st Qu.:  1.760   1st Qu.:21.30
##  Median :  6.754   Median : 1109   Median :  2.040   Median :26.90
##  Mean   : 12.976   Mean   : 2133   Mean   :  3.208   Mean   :26.18
##  3rd Qu.: 14.132   3rd Qu.: 2507   3rd Qu.:  2.190   3rd Qu.:32.20
##  Max.   :139.422   Max.   :16719   Max.   :181.000   Max.   :42.80
##        PK
##  Min.   :138.0
##  1st Qu.:162.0
##  Median :170.0
##  Mean   :174.5
##  3rd Qu.:183.0
##  Max.   :233.0
```

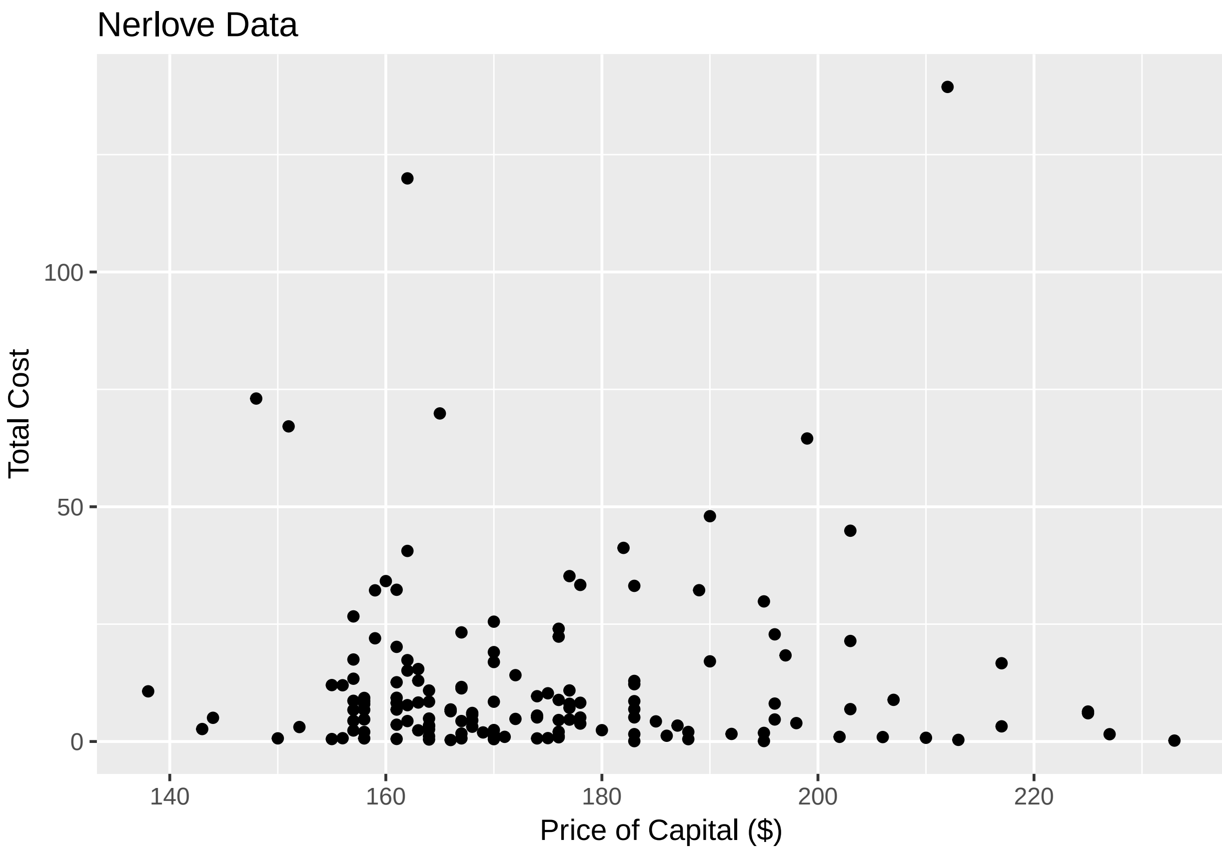
The max on the wage rate should not be that high: we must have an error in the data!

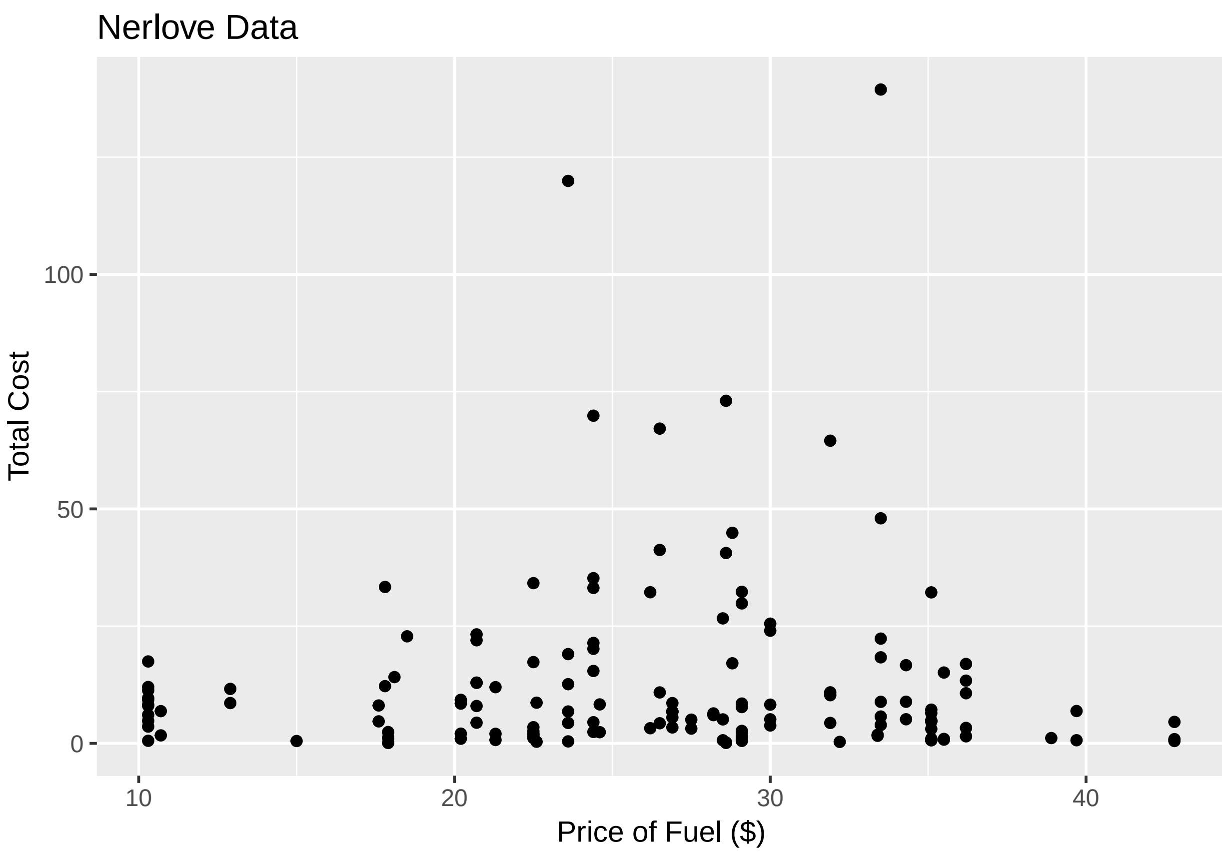
```
# Fix typo in 13th row (missing a decimal!)
nerlove[13, "PL"] <- 1.81

nerlove %>% arrange(Q)
```

```
## # A tibble: 145 x 5
##        TC     Q    PL    PF    PK
##     <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 0.0820    2.  2.09  17.9  183.
##  2 0.661     3.  2.05  35.1  174.
##  3 0.990     4.  2.05  35.1  171.
##  4 0.315     4.  1.83  32.2  166.
##  5 0.197     5.  2.12  28.6  233.
##  6 0.0980    9.  2.12  28.6  195.
##  7 0.949    11.  1.98  35.5  206.
##  8 0.675    13.  2.05  35.1  150.
##  9 0.525    13.  2.19  29.1  155.
## 10 0.501    22.  1.72  15.0  188.
## # ... with 135 more rows
```

Plot the series and make sure your data are read in correctly.







All seems well! Notably, the total cost trends well with the output while the price on labor, fuel, and capital all seem to be fairly random. We'll explore the former relationship more.

Question 2:

Replicate regression I (page 176) in the paper.

Regression I:

$$log(TC) - log(P_F) = \beta_0 + \beta_1 Q + \beta_2\Big(log(P_L) - log(P_F)\Big) + \beta_3\Big(log(P_K) - log(P_F)\Big)$$

Equivalent to:

$$log(\frac{TC}{P_F}) = \beta_0 + \beta_1 Q + \beta_2 log(\frac{P_L}{P_F}) + \beta_3 log(\frac{P_K}{P_F})$$

Where:
TC = total production cost,

P_L = wage rate,

P_K = "price" of capital,

P_F = price of fuel,

Q = output (measured in kWh)

In generalized Cobb-Douglas form:

$\beta_1 = \frac{1}{r}$,

$\beta_2 = \frac{a_L}{r}$,

$\beta_3 = \frac{a_K}{r}$

Prepare variables for Regression I.

```
# Create log variables
nerlove %<>% mutate(
  TClog = log(TC),
  Qlog = log(Q),
  PLlog = log(PL),
  PKlog = log(PK),
  PFlog = log(PF)
)

# Create PF scaled variables
nerlove %<>% mutate(
  TCscaled = TClog - PFlog,
  PLscaled = PLlog - PFlog,
  PKscaled = PKlog - PFlog
)
```

Variable names:

$log(\frac{TC}{P_F})$ = "TCscaled"

$log(\frac{P_L}{P_F})$ = "PLscaled"

$log(\frac{P_K}{P_F})$ = "PKscaled"

```r
# Regression I:
# dep var = (log costs - log fuel price) = TCscaled
reg_I <- ols(data = nerlove,y_data = "TCscaled",
             X_data = c("Qlog","PLscaled","PKscaled"),
             intercept = T, H0 = 0, alpha = 0.05)

reg_I$ttest
```

effect	coef	std_error	t_stat	p_value	significance
intercept	-4.691	0.885	-5.301	0.0000	Reject
Qlog	0.721	0.017	41.334	0.0000	Reject
PLscaled	0.593	0.205	2.898	0.0044	Reject
PKscaled	-0.007	0.191	-0.039	0.9692	Fail to Reject

```r
reg_I$mof_table
```

R^2_{uc}	R^2	R^2_{adj}	SIC	AIC	SSR	s^2
0.966	0.9316	0.9301	-1.765	-1.847	21.64	0.1535

Coefficients are pretty close to those in the paper. R^2 matches!

Question 3:

Conduct the hypothesis test using constant returns to scale ($\beta_1 = 1$) as your null hypothesis.

```r
# Re-run regression with null hypothesis H0 = 1
reg_I <- ols(data = nerlove, y_data = "TCscaled",
             X_data = c("Qlog","PLscaled","PKscaled"),
             intercept = T, H0 = 1, alpha = 0.05)

reg_I$ttest
```

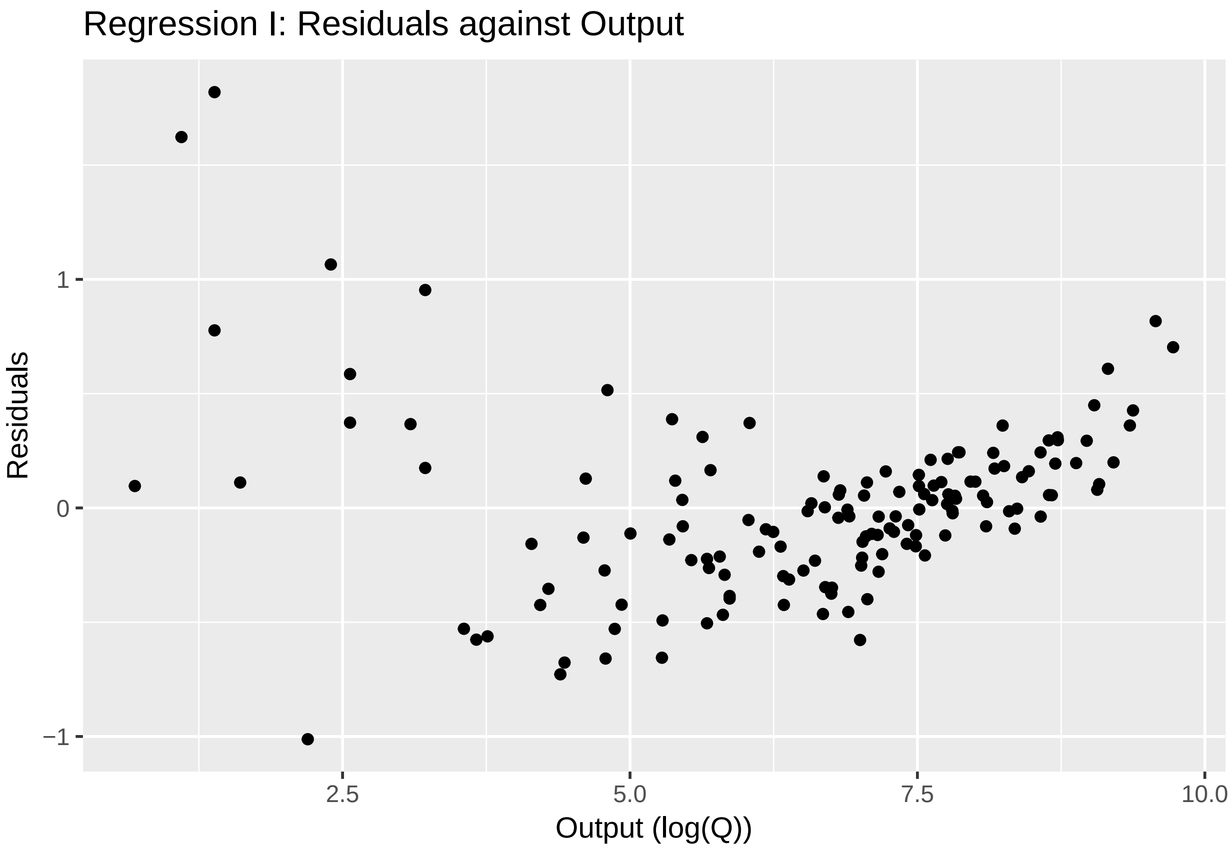
effect	coef	std_error	t_stat	p_value	significance
intercept	-4.691	0.885	-6.431	0.0000	Reject
Qlog	0.721	0.017	-16.020	0.0000	Reject
PLscaled	0.593	0.205	-1.990	0.0485	Reject
PKscaled	-0.007	0.191	-5.282	0.0000	Reject

What is the p- value associated with you test statistic? What is your point estimate of returns to scale? Constant? Increasing? Decreasing?

The p-value is 0.000. The point estimate of returns to scale is $\frac{1}{\beta_1} = r =$ 1.3876, hence returns to scale is increasing.

Question 4:

Plot residuals against output.



What do you notice? What does this potentially tell you from an economic perspective?

Evidence of heteroskedasticity: residuals seem to track the log output through parabola (or some general nonlinear fashion). We may want to rethink our specification!

Compute the correlation coefficient of the residuals with output for the entire sample? What does this tell you?

```
# R = cov(xy)/var(x)var(y)
R_I <- (cov(x=reg_I$vars$e, y=reg_I$vars$Qlog)/(var(reg_I$vars$e)*var(reg_I$vars$Qlog))) %>%
  signif(3) %>% as.character()
```

The correlation coefficient is extremely small: -1.08e-14. This tells us that, conversely, there is very little correlation between the output of the firm and the undetermined part of the model. This supports our strict exogeneity assumption.

Question 5:

Divide your sample into 5 subgroups of 29 firms each according to the level of output. Estimate the regression model again for each group separately.

```
# Divide sample into 5 subgroups
d <- split(nerlove,rep(1:5,each=29))
```

Now we have a list, d, with five dataframes '1' through '5' for our five subgroups.

Can you replicate Equations IIIA - IIIE? Calculate the point estimates for returns to scale for each sample. Is there a pattern relating to size of output?

We're going to run five separate regressions and display the coefficients on $ln(y)$ and the corresponding returns to scale. *Regression IIIA code as an example below.

```
# Regression IIIA
reg_IIIA <- ols(data = d$`1`,y_data = "TCscaled",
             X_data = c("Qlog","PLscaled","PKscaled"))
```

```
# Create Returns to scale table
coef <- rbind(reg_IIIA$b[2], reg_IIIB$b[2], reg_IIIC$b[2],
             reg_IIID$b[2], reg_IIIE$b[2])
rts <- rbind(1/reg_IIIA$b[2], 1/reg_IIIB$b[2],
             1/reg_IIIC$b[2], 1/reg_IIID$b[2],
             1/reg_IIIE$b[2])
rts_rownames <- rbind("IIIA", "IIIB","IIIC","IIID","IIIE")

reg_III_df <- data.frame(rts_rownames, coef, rts)

reg_III_table <- reg_III_df %>% knitr::kable(
    col.names = c("Regression","Coefficients", "Returns to Scale"),
    format.args = list(scientific = F, digits = 4),
    booktabs = T,
    escape = F
  )
reg_III_table
```

Regression	Coefficients	Returns to Scale
IIIA	0.4003	2.4982
IIIB	0.6582	1.5194
IIIC	0.9383	1.0658
IIID	0.9120	1.0964
IIIE	1.0444	0.9575

Coefficients roughly match the results of the paper! As the size (output) of the firms get larger they see decreasing, increasing returns to scale. The largest bucket even sees decreasing returns to scale.

Question 6:

Create "dummy variables" for each industry. Interact them with the output variable to create five "slope coefficients".

```
# create group categorical var
for (i in 1:5) {
  d[[i]] %<>% mutate(gvar=i)}

# unsplit
df <- rbind(d[[1]], d[[2]], d[[3]], d[[4]], d[[5]])

# create dummies
df %<>%
    mutate(
      g1 = ifelse(gvar==1, 1, 0),
      g2 = ifelse(gvar==2, 1, 0),
      g3 = ifelse(gvar==3, 1, 0),
      g4 = ifelse(gvar==4, 1, 0),
      g5 = ifelse(gvar==5, 1, 0)) %>% select(-gvar)

# interact with output variable
df %<>%
    mutate(
      lQ_A = Qlog*g1,
      lQ_B = Qlog*g2,
      lQ_C = Qlog*g3,
      lQ_D = Qlog*g4,
      lQ_E = Qlog*g5)
```

Run a model, letting the intercept and slope coefficient on output differ across plants, but let the remainder of the coefficients be pooled across plants.

```
reg_IV <- ols(data = df, y_data = "TCscaled",
              X_data = c("lQ_A", "lQ_B", "lQ_C", "lQ_D", "lQ_E",
                         "PLscaled", "PKscaled", "g1", "g2", "g3", "g4", "g5"),
              intercept = F)

reg_IV$mof_table
```

R^2_{uc}	R^2	R^2_{adj}	SIC	AIC	SSR	s^2
0.9802	0.9602	0.9569	-2.033	-2.279	12.58	0.09457

Regression	Coefficients	Returns to Scale
IVA	0.3969	2.520
IVB	0.6482	1.543
IVC	0.8848	1.130
IVD	0.9087	1.100
IVE	1.0627	0.941

Are there any noticeable changes in returns to scale from the previous part?

We got roughly the same point estimates on returns to scale as the previous part! There is a slight bit of variation which is from the pooled labor and capital price effect. I would imagine we'd get even closer to the previous result if we interacted our industry dummy variable with each of them.

Question 7:

Conduct a statistical test comparing the first model you estimate to the last model you estimated. (Hint: Is one model a restricted version of the other?). Would separate t-test have given you the same results?

Regression I is the restricted model, Regression IV is unrestricted model. This being, Regression I says that all five industries all have the same slope on output and all have the same intercept, thus placing restrictions.

$F = \frac{(SSR_R - SSR_U)/J}{SSR_U/(n-k)}$

```
# Calculate F_stat using SSR approach
ssr_u <- reg_IV$SSR
dof <- reg_IV$dof

ssr_r <- reg_I$SSR
j <- 8 #four restrictions on both slope and intercept

f_statistic <- ((ssr_r - ssr_u) / j) / (ssr_u / dof)

# Calculate the p-value
p_value <- pf(q = f_statistic, df1 = j, df2 = dof, lower.tail = F)
# Create a data.frame of the f stat. and p-value
f_test <- data.frame(
  f_statistic = f_statistic %>% as.vector(),
  p_value = p_value %>% as.vector())

f_test
```

```
##   f_statistic      p_value
## 1     11.9797 8.901198e-13
```

As you can see, we got an incredibly low p-value between the first and last model, menaing the differentiated slope on output and intercept coefficients are jointly, statistically significant. This SSR approach is different than just doing an F-Test on the last regression because it is saying, all the variables (including the pooled price on labor and capital) are jointly, statistically significant from just the mean of the scaled log of total cost. I don't even want to begin thinking about running a myriad of t-test on the 8 individual restrictions of our first model. Theory tells us we are likely to get different results because the F-test correctly ajusts the joint confidence intervals while seperate t-test would be a ridgid overlapping of confidence intervals that may(not) correctly approximate the F-test.

Question 8:

To see whether returns to scale declined with output, Nerlove tested a nonlinear specification by including $ln(y)^2$ as a regressor. Conduct a statistical test you feel is appropriate to test this hypothesis.

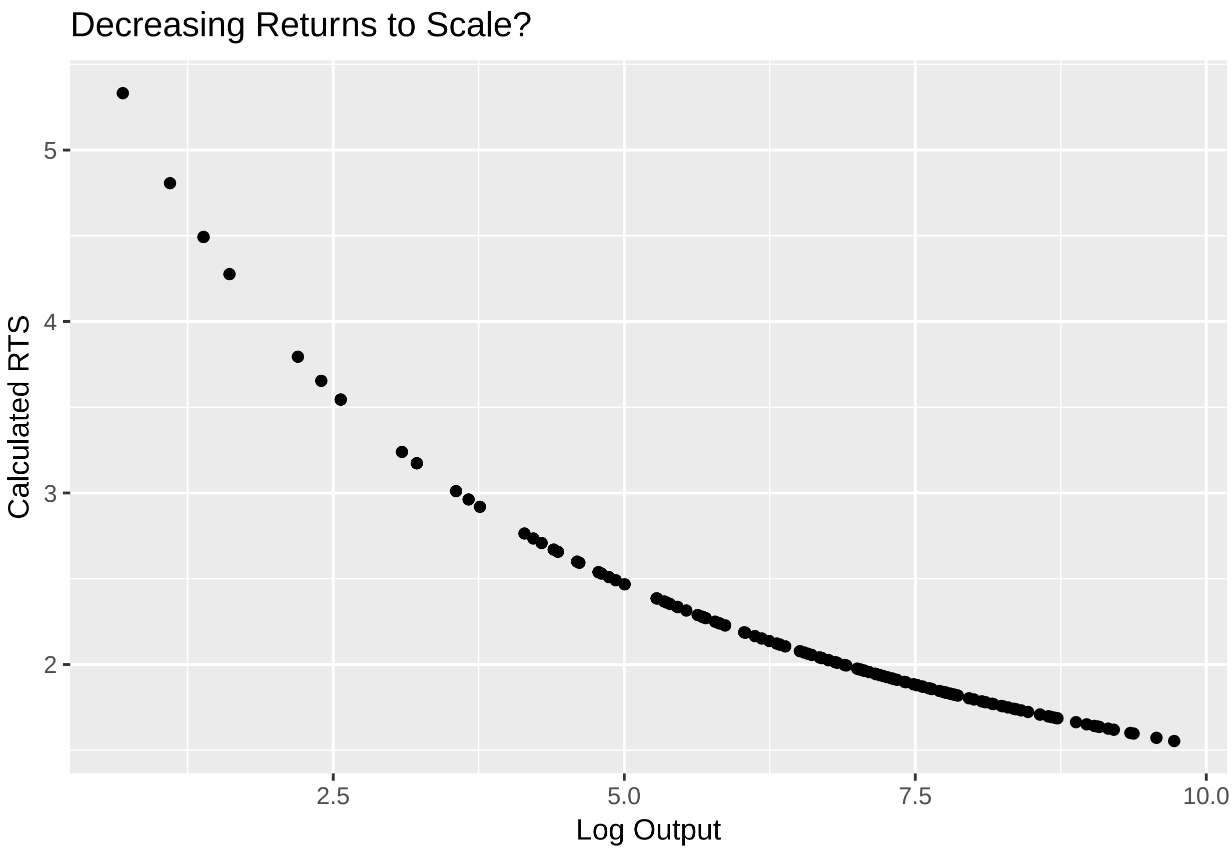
It follows that returns to scale (r) $= \frac{1}{\alpha+\beta ln(y)}$, where α is the cofficient on $ln(y)$ and β is the coefficient on $ln(y)^2$ in the regression below.

```r
df %<>% mutate(Qlog_sq = (Qlog)^2)

reg_nl <- ols(data = df, y_data = "TCscaled",
              X_data = c("Qlog","Qlog_sq", "PLscaled","PKscaled"),
              intercept = T, H0 = 0, alpha = 0.05)

reg_nl$ttest
```

effect	coef	std_error	t_stat	p_value	significance
intercept	-3.765	0.702	-5.365	0.0000	Reject
Qlog	0.153	0.062	2.466	0.0149	Reject
Qlog_sq	0.051	0.005	9.418	0.0000	Reject
PLscaled	0.481	0.161	2.984	0.0034	Reject
PKscaled	0.074	0.150	0.494	0.6218	Fail to Reject

We're going to save these results and first plot our calculated returns to scale variable against log output to see if there is a trend.

```r
# Calculated RTS
df %<>% mutate(rts_nl = 1/(reg_nl$b[2] + reg_nl$b[3]*df$Qlog))

ggplot(df, aes(x=Qlog, y=rts_nl)) +
  geom_point() +
  labs(title="Decreasing Returns to Scale?", x="Log Output", y="Calculated RTS")
```



It would seem that our returns to scale decreases as output increases. Whether this is statisically significant is the question. We'll run a regression of our calculated returns to scale on the output and it's square; then, we'll run our F-test function comparing this function to just the mean of RTS. This will adquately measure for joint significance.

```
reg_nl2 <- ols(data = df, y_data = "rts_nl",
               X_data = c("Qlog","Qlog_sq"),
               intercept = T, H0 = 0, alpha = 0.05)

reg_nl2$ttest
```

effect	coef	std_error	t_stat	p_value	significance
intercept	5.438	0.037	146.445	0	Reject
Qlog	-0.800	0.013	-60.146	0	Reject
Qlog_sq	0.043	0.001	37.188	0	Reject

```
reg_nl2_f <- F_test(data = df, y_var = "rts_nl", X_vars = c("Qlog","Qlog_sq"))
F_p <- reg_nl2_f[2] %>% signif(4) %>% as.character()
```

Here, it would seem that for every one percent increase of output for firms, we see a -0.008 unit decrease in the firms returns to scale; and the output and its square are independently significant (notwithstanding the robust variance issue). Also, the p-value on our F-test is 1.984e-141 which tells us that output (and its square) is jointly, statistically significant on returns to scale.