ARE 212 Midterm

Anaya Hall

March 14, 2018

Question One: "Linkages among climate change, crop yields and Mexico - US cross-border migration"

Load OLS functions

```
# Function to turn given data into matrix for use in OLS function
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)
}

# Function for OLS coefficient estimates and measures of fit
b_ols <- function(data, y_data, X_data, intercept=TRUE) {

  require(dplyr)
    # 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 beta hat ------
    b <- solve( t(X) %*% X ) %*% t(X) %*% y
    # Change the name of 'ones' to 'intercept'
    if(intercept == T){
        rownames(b) <- c("intercept", X_data) }
    else
        rownames(b) <- c(X_data)

    y_hat <- X %*% b
    e <- y - y_hat

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

```r
    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

    results <- data.frame(
      # The rows have the coef. names
      x_var = rownames(b),
      # Estimated coefficients
      coef = as.vector(b) %>% round(3)
    )

    res <<- e
    adjr2 <<- R2adj

    # Return beta_hat & adjusted r2
    #print(paste("Adj R2: ", R2adj))
    return(results)
}
```

Load Data (cleaned and prepared in Excel)

```
## Parsed with column specification:
## cols(
##   state = col_character(),
##   cornyield = col_double(),
##   cornwheatyield = col_double(),
##   remi = col_double(),
##   ppt = col_double(),
##   mat = col_double(),
##   mst = col_double(),
##   y1995 = col_integer(),
##   y2000 = col_integer()
## )
```

Inspect data

```r
summary(feng_data)
```

```
##     state              cornyield       cornwheatyield        remi
##  Length:64          Min.   :-0.9510   Min.   :-0.9510   Min.   :-4.900
##  Class :character   1st Qu.: 0.1767   1st Qu.: 0.1777   1st Qu.: 3.200
##  Mode  :character   Median : 0.5840   Median : 0.5860   Median : 5.000
##                     Mean   : 0.5463   Mean   : 0.5937   Mean   : 4.928
```

```
##                     3rd Qu.: 0.9185   3rd Qu.: 0.9832   3rd Qu.: 7.125
##                     Max.   : 1.9720   Max.   : 1.9230   Max.   :12.000
##       ppt              mat             mst            y1995
##  Min.   :0.1800   Min.   :14.37   Min.   :15.90   Min.   :0.0
##  1st Qu.:0.5250   1st Qu.:17.70   1st Qu.:21.24   1st Qu.:0.0
##  Median :0.7600   Median :20.93   Median :24.11   Median :0.5
##  Mean   :0.8719   Mean   :20.90   Mean   :23.77   Mean   :0.5
##  3rd Qu.:1.0650   3rd Qu.:24.11   3rd Qu.:27.12   3rd Qu.:1.0
##  Max.   :2.3700   Max.   :27.20   Max.   :29.37   Max.   :1.0
##      y2000
##  Min.   :0.0
##  1st Qu.:0.0
##  Median :0.5
##  Mean   :0.5
##  3rd Qu.:1.0
##  Max.   :1.0
```

1. **Estimate model (1) via OLS by regressing emigration rate on log of yields and a time period fixed effect. Report coefficient on yield and adjusted R^2. Does this match the results in the first column of table #1?**

```
##       x_var  coef
## 1 intercept 2.636
## 2 cornyield 0.829
## 3     y1995 3.679
```

Adjusted R^2: 0.321155954568775

Coefs: intercept: 2.636
cornyield: 0.829
y1995: 3.679

These don't really match the results in the paper

```
##            x_var  coef
## 1      intercept 2.611
## 2 cornwheatyield 0.818
## 3          y1995 3.662
```

Adjusted R^2: 0.32147846639082"

intercept 2.611
cornwheatyield 0.818
y1995 3.662

These don't really match the results in the paper

2. **Estimate model (1) again via fixed effects and FWT. Report coefficient on yield and adjusted R^2. Does this match the results in the third column of table #1?**

```
resid_ols <- function(data, y_var, X_vars, intercept = T) {
  # Require the 'dplyr' package
  require(dplyr)
  # Create the y matrix
  y <- to_matrix(the_df = data, vars = y_var)
  # Create the X matrix
  X <- to_matrix(the_df = data, vars = X_vars)
  # If 'intercept' is TRUE, then add a column of ones
  if (intercept == T) {
```

```r
    # Bind a column of ones to X
    X <- cbind(1, X)
    # Name the new column "intercept"
    colnames(X) <- c("intercept", X_vars)
  }
  # Calculate the sample size, n
  n <- nrow(X)
  # Calculate the residuals
  resids <- (diag(n) - X %*% solve(t(X) %*% X) %*% t(X)) %*% y
  # Return 'resids'
  return(resids)
}
```

```r
step1_resid <- resid_ols(feng_data, "remi", "cornyield", F)

feng_data %<>% mutate(ones = 1)
# Our two regressions
step2a_resid <- resid_ols(feng_data, "ones", "cornyield", F)
step2b_resid <- resid_ols(feng_data, "y1995", "cornyield", F)


df_fwt <- data.frame(
  remi_resid  = as.vector(step1_resid),
  i_resid      = as.vector(step2a_resid),
  fe_resid = as.vector(step2b_resid)
  ) %>% tbl_df()
# The final regression
b_ols(df_fwt, "remi_resid", c("i_resid", "fe_resid"), F)
```

```
##      x_var  coef
## 1  i_resid 2.636
## 2 fe_resid 3.679
```

```r
step1_resid <- resid_ols(feng_data, "remi", "cornwheatyield", F)

feng_data %<>% mutate(ones = 1)
# Our two regressions
step2a_resid <- resid_ols(feng_data, "ones", "cornwheatyield", F)
step2b_resid <- resid_ols(feng_data, "y1995", "cornwheatyield", F)


df_fwt <- data.frame(
  remi_resid  = as.vector(step1_resid),
  i_resid      = as.vector(step2a_resid),
  fe_resid = as.vector(step2b_resid)
  ) %>% tbl_df()
# The final regression
b_ols(df_fwt, "remi_resid", c("i_resid", "fe_resid"), F)
```

```
##      x_var  coef
## 1  i_resid 2.611
## 2 fe_resid 3.662
```

These both match the output from my first regressions, but they do not match the results in the paper

3. **Repeat step 1 without the the fixed effects. Report coefficient on yield and adjusted R^2.**

Do the results look different from what you estimated before? From what is in the paper?

```
b_ols(data = feng_data, y_data = "remi", X_data = c("cornyield"))
```

```
##       x_var  coef
## 1 intercept 4.632
## 2 cornyield 0.541
```

Adjusted R^2: -0.004523002

4. Repeat step 2 without the the fixed effects. Report coefficient on yield and adjusted R^2. Do the results look different from what you estimated before? From what is in the paper?
5. What happened here? What are the consequences?

Question Two: Normality of OLS

Model: $y_i = \beta_o + \beta_1 x_{1i} + \beta_2 x_{2i} + \epsilon_i$

Truth: $\beta_0 = 3$, $\beta_1 = 1$, $\beta_2 =$ -2

Load functions for use in simulation

Generate data function (given a sample size, n)

```
#ADD SWEEP TO FIX COVARIANCE OF X1 & X2 (Should be 0, but isn't right now)
gen_data <- function(sample_size) {
  # Create data.frame with random x and error
  data_df <- data.frame(
    X1 = rnorm(n = sample_size, sd = 5),
    X2 = rnorm(n = sample_size, sd = 5),
    # X <- rnorm(n = sample_size, sd =5) %>% matrix(sample_size, 2),
    e = rnorm(sample_size, sd = 5),
    eta = runif(sample_size, -8.66, 8.66))
  # Calculate y = 3 + 1 x1 - 2 x2 + e; drop 'e'
  data_df %<>% mutate(y_a = 3 + 1 * X1 - 2 * X2 + e,
                      y_b = 3 + 1 * X1 - 2 * X2 + eta) %>% #Populaton
    select(-e, -eta)
  # Return data_df
  return(data_df)

}

 # test <- gen_data(10)
 # test
```

Function for a single simulation of OLS

```
one_sim <- function(sample_size, depvar) {
  # Estimate via OLS
  ols_est <- b_ols(data = gen_data(sample_size),
    y_data = depvar, X_data = c("X1", "X2"))
  # Grab the estimated coefficient on x
  # (the second element of 'coef')
  b2 <- ols_est %$% coef[3]
  # Return a data.frame with b2
```

```
  return(data.frame(b2))
}
```

Function for multiple simulations of OLS

```
ols_sim <- function(depvar, n_sims = 1e4, sample_size, seed = 22092008) {
  # require parallel
  require(parallel)
  # Set the seed
  set.seed(seed)
  # Run one_sim n_sims times; convert results to data.frame
  sim_df <- replicate(
    n = n_sims,
    expr = one_sim(sample_size, depvar),
    simplify = F
    ) %>% bind_rows()

   # TRY TO PARALLELIZE TO SPEED UP! ---- get error with bind rows though...
   # sim_df <- mclapply(
   #   X = rep(x = sample_size, times = n_sims),
   #   FUN = one_sim,
   #   # Specify that we want 4 cores
   #   mc.cores = 4
   #   ) %>% bind_rows()
  # Return sim_df
  return(sim_df)
}
```

For each part, repeat for sample sizes: n=[10, 100, 1000, 10000, 20000] and run 1e4 simulations

Part A:

Regress y^a on intercept, x_1 and x_2. Record β_2

```
N <- c(10, 100, 1000, 10000, 20000)

# sim_A <- matrix("list", 5)
# for (n in N){
#   print(n)
#   sim_A[[n]] <- ols_sim(depvar = "y_a", n_sims = 1e4, sample_size = n)
#
# }

# Run for sample sizes: n=[10, 100, 1000, 10000, 20000]
# Run ols_sim for sample size of 10
sima10 <- ols_sim(depvar = "y_a", n_sims = 1e4, sample_size = 10)
```

```
## Loading required package: parallel
```

```
# Run ols_sim for sample size of 100
# sima100 <- ols_sim(depvar = "y_a", n_sims = 1e4, sample_size = 100)
# Run ols_sim for sample size of 1000
# sima1000 <- ols_sim(depvar = "y_a", n_sims = 1e4, sample_size = 1000)
# # Run ols_sim for sample size of 10000
# sima10000 <- ols_sim(depvar = "y_a", n_sims = 1e4, sample_size = 10000)
```

```r
# # Run ols_sim for sample size of 20000
# sima20000 <- ols_sim(depvar = "y_a", n_sims = 1e4, sample_size = 20000)
```

Plot histogram

```r
pa1 <- ggplot(sima10, aes(x=b2)) + geom_density() + labs(title="10 Iterations")
# pa2 <- ggplot(sima100, aes(x=b2)) + geom_density() + labs(title="100 Iterations")
# pa3 <- ggplot(sima1000, aes(x=b2)) + geom_density() + labs(title="1000 Iterations")
# pa4 <- ggplot(sima10000, aes(x=b2)) + geom_density() + labs(title="10000 Iterations")
# pa5 <- ggplot(sima20000, aes(x=b2)) + geom_density() + labs(title="20000 Iterations")

# multiplot(pa1, pa2, pa3, pa4, pa5, cols= 3)
```

Part B:

Regress y^b on x_1 and x_2. Record β_2

```r
# Run for sample sizes: n=[10, 100, 1000, 10000, 20000]
# Run ols_sim for sample size of 10
simb10 <- ols_sim(depvar = "y_b", n_sims = 1e4, sample_size = 10)
# Run ols_sim for sample size of 100
# simb100 <- ols_sim(depvar = "y_b", n_sims = 1e4, sample_size = 100)
# # Run ols_sim for sample size of 1000
# simb1000 <- ols_sim(depvar = "y_b", n_sims = 1e4, sample_size = 1000)
# # Run ols_sim for sample size of 10000
# simb10000 <- ols_sim(depvar = "y_b", n_sims = 1e4, sample_size = 10000)
# # Run ols_sim for sample size of 20000
# simb20000 <- ols_sim(depvar = "y_b", n_sims = 1e4, sample_size = 20000)
```

Plot histogram

```r
pb1 <- ggplot(simb10, aes(x=b2)) + geom_density() + labs(title="10 Iterations")
# pb2 <- ggplot(simb100, aes(x=b2)) + geom_density() + labs(title="100 Iterations")
# pb3 <- ggplot(simb1000, aes(x=b2)) + geom_density() + labs(title="1000 Iterations")
# pb4 <- ggplot(simb10000, aes(x=b2)) + geom_density() + labs(title="10000 Iterations")
# pb5 <- ggplot(simb20000, aes(x=b2)) + geom_density() + labs(title="20000 Iterations")

# multiplot(pb1, pb2, pb3, pb4, pb5, cols= 3)
```

Thanks to Ed for providing A LOT of this code in section!