Problem Set 5

Nima Mohammadi
nimamo@vt.edu

1. Q1: Serial correlation

(See scripts `mod4s3a` and `mod4s3b` for guidance) Consider Greene's gasoline consumption data (on the web under "gasoline" in tab-delimited .txt format). The variables are as follows:

(1) Year = Year, 1953-2004,
(2) GasExp = Total U.S. gasoline expenditure, in thousands
(3) Pop = U.S. total population in thousands
(4) GasP = Price index for gasoline,
(5) Income = Per capita disposable income,
(6) Pnc = Price index for new cars,
(7) Puc = Price index for used cars,
(8) Ppt = Price index for public transportation,
(9) Pd = Aggregate price index for consumer durables,
(10) Pn = Aggregate price index for consumer nondurables,
(11) Ps = Aggregate price index for consumer services.

The textbook analyzes a model using these data in the context of autocorrelation on pp. 649-650. Load the data into R, and specify Greene's model on p. 649 (6th edition), p. 927 (7th edition). Your dependent variable should be log[(GasExp)/(Pop*GasP)]. Your regressors should be:

(1) constant
(2) income = log(Per capita disposable income)
(3) GasP = log(Price index for gasoline)
(4) Pnc = log(Price index for new cars)
(5) Puc = log(Price index for used cars)
(6) time index

```
data <- read.table('/Users/nima/AAEC5126/data/gasoline.txt', sep="\t", header=FALSE)

colnames(data) <- c("Year", "GasExp", "Pop", "GasP","Income", "Pnc", "Puc",
                    "Ppt", "Pd", "Pn", "Ps")
attach(data)
```

To create the last regressor (time index), you need to translate the year - variable into a running index from 1:52. Label this variable "t_i".

```
y <- log(GasExp / (Pop * GasP))

Income <- log(Income)
GasP <- log(GasP)
Pnc <- log(Pnc)
```

```
Puc <- log(Puc)
t_i <- data$Year - min(data$Year) + 1
```

(1) Run a simple OLS model. (Note: Greene's results on p. 650 / 927 are a bit off, but close). Comment on the significance levels of each regressor (ignore the constant term). Are the signs of significant regressors as expected? Explain.

```
X <- cbind(rep(1, nrow(data)), Income, GasP, Pnc, Puc, t_i)
k <- ncol(X)
n <- nrow(X)

#OLS estimator
bols <- solve((t(X)) %*% X) %*% (t(X) %*% y)
e <- y - X %*% bols # residuals
SSR <- (t(e) %*% e)#sum of squared residuals
s2 <- (t(e) %*% e) / (n - k) #estimated variance of "eps"
s2ols <- s2 #for Hausman test below
Vb <- s2[1, 1] * solve((t(X)) %*% X) # estimated VCOV matrix of bols
se = sqrt(diag(Vb)) # get the standard erros for your coefficients
tval = bols / se # get your t-values
```

Table 1. OLS output

variable	estimate	s.e.	t
constant	-26.6753	2.0005	-13.3340
log(income)	1.6246	0.1952	8.3227
log(GasP)	-0.0539	0.0422	-1.2782
log(Pnc)	-0.0834	0.1766	-0.4725
log(Puc)	-0.0847	0.1024	-0.8272
t_i	-0.0139	0.0048	-2.9149

The results show that the coefficients of log(income) and time are significant at 1% and 5% levels. The sign of the coefficient of the variable ”log of Per capita disposable income” is positive which means that a 1% increase in Per capita disposable income leads to 1.625% increase in [(GasExp)/(Pop*GasP)]. Subsequently, whenever per capita disposable income increases people may travel more because they will make more use of their cars and this would lead to an increase of gasoline expenditure. Therefore the sign of significant regressor ”income” is expected. Going ahead in time has different impacts on gas consumption. First, along time technology improves and cars use less gas for the same milage, this suggests negative coefficient. Second, overtime individuals are become wealthier, as percapita income increases, more people can afford cars and this increases the gas usage and suggest positive effect. Therefore, the true sign for time coefficient depends on which effect is larger. However, the results show negative impact of time.

(2) Generate OLS residuals (call them ”**e**”) and plot them against time (year). The pattern should look a lot like the graph on p. 650 / 928. Does it indicate autocorrelation - why or

why not? If so, is it suggestive of positive or negative autocorrelation - explain.

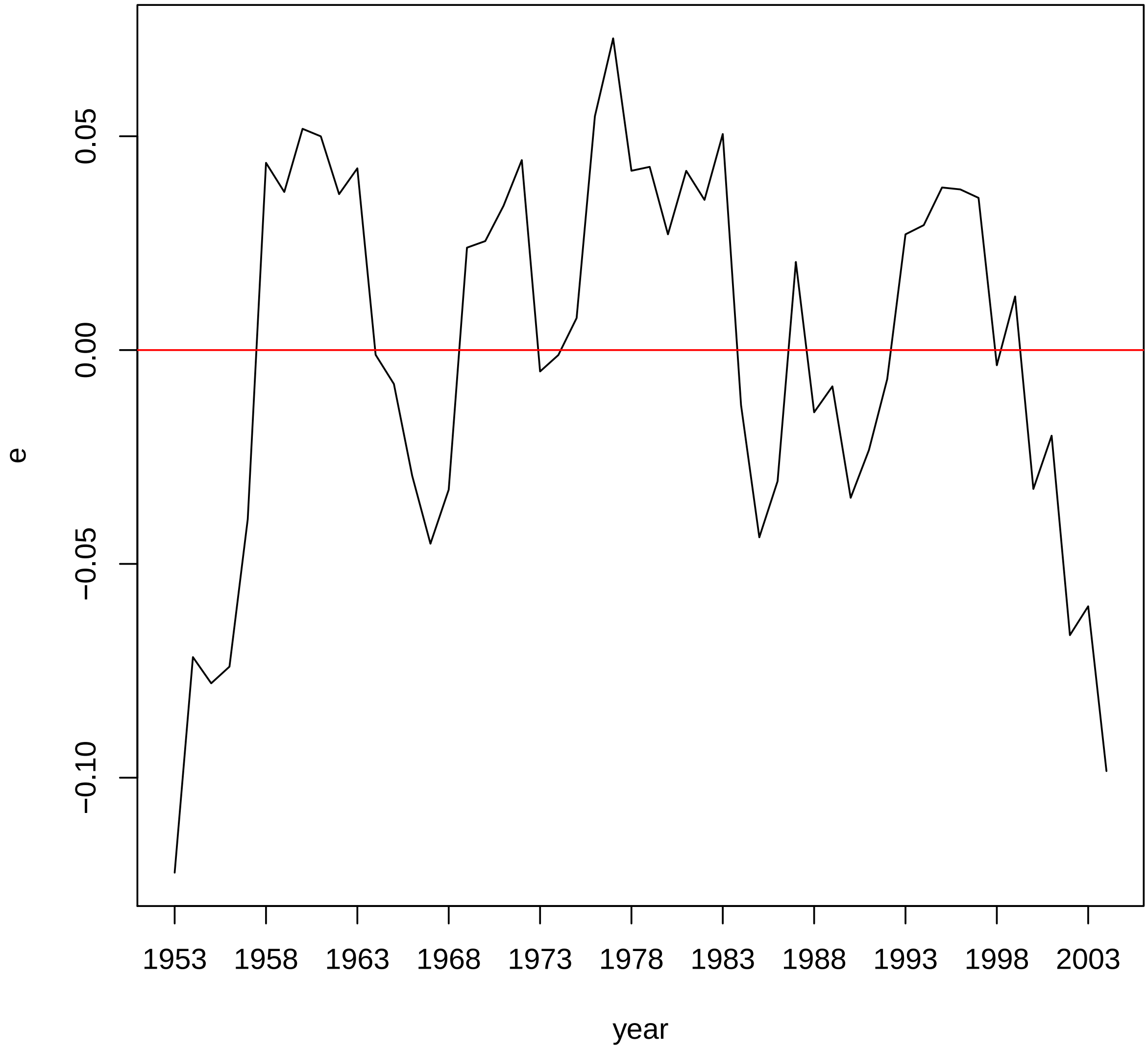

Figure 1. OLS residual plots

The figure shows that the existence of consecutive upward and downward movements in residuals. We know that this plot suggests positive autocorrelation. That is for positive correlation, positive (negative) residuals tend to be followed by positive (negative) residuals.

(3) Perform a Breusch-Godfrey multiplier test for AR(1). State the null and alternative hypotheses, the computed p-value and your decision regarding the null (at 5% level of significance).

```
elag <- e[1:(n - 1)]

e0lag <- c(0, elag) # fill first position with 0 */
Xo = cbind(X, e0lag) #augment X with a column of lagged residuals

LM <- n * ((t(e) %*% Xo %*% solve(t(Xo) %*% Xo)
            %*% t(Xo) %*% e) / (t(e) %*% e))
pval = 1 - pchisq(LM, 1)
```

The Breusch-Godfrey test can be used for a set of alternative hypotheses. Each of which describes a different AR(P) process. Here, our test for P=1 we investigate for process of order 1. That is the null hypothesis will be:

- H_0 : No autocorrelation of order P=1
- H_a : Autoregressive or Moving average of order P=1

The test statistic for this test is

$$\text{BG} = T\left(\frac{\varepsilon'\mathbf{X}_0\left(\mathbf{X}_0'\mathbf{X}_0\right)^{-1}\mathbf{X}_0'\varepsilon}{\varepsilon'\varepsilon}\right) \sim \chi^2(P)$$

The BG-statistic for this test is 27.182.
The degrees of freedom for the test are 1.
The corresponding p-value is 0.
So the null hypothesis will be rejected, which is ”there is no autocorrelation” at 5% and 1% level of significance. We cannot say there is no autocorrelation.

(4) Compute the Durbin-Watson statistic. State the null and alternative hypotheses, the appropriate degrees of freedom, the appropriate critical values from the DW Table at $\alpha = 0.05$ (textbook or google on the web), and your decision regarding the null. (the DW value should be the same as the one mentioned in Greene, up two the first 2 decimals).

```
ecurr <- e[2:n]
elag <- e[1:(n - 1)]
d <- (t(ecurr - elag) %*% (ecurr - elag)) / (t(e) %*% e)
```

The test statistic is

$$d = \frac{\left(\varepsilon_t - \varepsilon_{t-1}\right)'\left(\varepsilon_t - \varepsilon_{t-1}\right)}{\varepsilon_t'\varepsilon_t} \approx 2(1-\hat{\rho})$$

- ε_t = vector of original residuals from OLS

- ε_{t-1} = vector of lagged residuals

- $\hat{\rho}$ = estimated autocorrelation coefficient from a regression of ε_t on ε_{t-1}

The null and Alternative hypothesis:

- $H_0 : \rho = 0$ absence of serial correlation

- $H_a : \rho > 0$ positive auotocorrelation

- $H_a : \rho < 0$ negative auotocorrelation

The decision rule using dL, dU:

- For $H_a : \rho > 0$: H_0 is rejected if d<dL; H_0 is not rejected if d>dU. Inconclusive if dL<d<dU.
- For $H_a : \rho < 0$: H_0 is rejected if d>(4-dL); H_0 is not rejected if d<(4-dU). Inconclusive if (4-dU)<d¡(4-dL)

For this specific case our null and alternative hypothesis is:

- $H_0 : \rho = 0$

- $H_a : \rho > 0$

The DW-statistic for this test is 0.424. The sample size is 52. The column space of X is 6.

According to the Durbin-Watson table, the lower bound of d (dL) is 1.34. So d=.424 ¡ dL= 1.34. Thus, we reject the null hypothesis. So, we have positive autocorrelation.

(5) Estimate robust OLS with Newey-West corrected standard errors. (In R, you can use

```
L<-ceiling(n^(1/4))
```

for the lag indicator, where n is the sample size).

```
L <- ceiling(n ^ (1 / 4))
# rounds upwards to nearest integer;
# this would be the generic choice
H <- matrix(0, k, k)

for (j in 1:L) {
  t <- j + 1
  G <- matrix(0, k, k)
  for (i in t:n)  {
    m <- (1 - (j / (L + 1))) * e[i] * e[i - j] *
      (t(X[i, , drop = FALSE]) %*% X[i - j, ] + t(X[i - j, , drop = FALSE]) %*% X[i, ])
    #drop=FALSE forces the transpose to be a column vector
    G <- G + m
  }
  H <- H + G
}
e <- as.vector(e)
S1 <- (t(X) %*% diag(e ^ 2) %*% X) + H
Vb <- solve((t(X)) %*% X) %*% S1 %*% solve((t(X)) %*% X)
se = sqrt(diag(Vb))
tval = bols / se
```

Table 2. Robust OLS output

variable	estimate	s.e.	t
constant	-26.675	2.408	-11.079
log(income)	1.625	0.243	6.676
log(gasP)	-0.054	0.054	-1.003
log(pnc)	-0.083	0.164	-0.509
log(puc)	-0.085	0.104	-0.818
t_i	-0.014	0.007	-1.978

(6) Compute the Prais-Winsten FGLS estimator. (The results will be a bit different than those given in Greene - he uses a slightly different estimation approach).

```
# Step 1: Get a consistent estimate of rho:
rho <-
  solve(t(elag) %*% elag) %*% t(elag) %*% ecurr #OLS solution for our
# "e vs. e-lag 1 regression model above
#
#Step 2: compose the correlation matrix R
R <- matrix(0, n, n)
up <- seq(1, (n - 1), 1)
down <- seq((n - 1), 1,-1)
int <- c(rho ^ (down), 1, rho ^ (up)) #1 by 2*(n-1)+1
## Warning in rho^(down):  Recycling array of length 1 in array-vector arithmetic
is deprecated.
##  Use c() or as.vector() instead.
## Warning in rho^(up):  Recycling array of length 1 in array-vector arithmetic
is deprecated.
##  Use c() or as.vector() instead.
for (i in 1:n) {
  R[i, ] <- int[(n - (i - 1)):(length(int) - (i - 1))]
}
#
#Step 3: compute FGLS estimator
bgls <- solve((t(X)) %*% solve(R) %*% X) %*% (t(X) %*% solve(R) %*% y)
#
```

```
#Step 4: compute a consistent estimate of sig(eps)
e <- y - X %*% bgls
sige <- (1 / n) * t(e) %*% solve(R) %*% (e)
#
#Step 5: Compute consistent variance-covariance matrix for b_fgls
Om <- sige[1, 1] * R
Vb <- solve((t(X)) %*% solve(Om) %*% X)
se = sqrt(diag(Vb))
tval = bgls / se
```

Table 3. FGLS output

variable	estimate	s.e.	t
constant	-19.557	1.746	-11.203
log(income)	0.855	0.174	4.910
log(gasP)	-0.103	0.037	-2.811
log(pnc)	-0.147	0.150	-0.981
log(puc)	-0.013	0.073	-0.175
t_i	0.004	0.005	0.869

(7) Compare your original OLS estimates, the robust estimates, and the FGLS results and elaborate:

(a) Compare the s.e.’s and t-values between OLS and robust OLS. Are there any noteworthy changes in significance levels?

The estimates for the OLS and the Robust OLS are almost identical. The standard error for the Robust OLS has slightly increased for all the variables except ”pnc”. Thus all the t values are smaller (in absolute term) in Robust OLS except ”pnc”. In both of OLS and Robust OLS ”log(income)” and ”constant’ are significant at 1% and 5% level. But, ”log(gasP)”, ”log(Pnc)” and ”log(Puc)” are insignificant in both. So there is no noteworthy changes in significance levels.
The time coefficient is different. OLS result suggests that time is significant in both 1% and 5% level, but Robust OLS indicates it is not.

(b) Compare the s.e.’s and t-values between the robust OLS and the FGLS model. Are there any noteworthy changes in significance levels?

The standard errors are smaller in FGLS for all variables. As tables show, ”constant” and ”log(income)” are significant in both methods but ”log(Pnc)” nad ”log(Puc)” are insignificant in both. The difference is in ”log(gasP)”. Although, it is significant in FGLS at both 1% and 5%, it is not significant in Robust OLS at both levels.

(c) Assume the main focus of your research is on the effect of ”gas prices” on ”gas consumption”. Overall, which model would you choose?

Since the error term follows AR(1) process, the structure of R matrix and so $E(\varepsilon\varepsilon')$ are known. Therefore, GLS results are the most efficient and we perefer to choose FGSL model.

2. Q2: Estimation of treatment effects via regression

This question uses home sales data from Connecticut (CT) for 1991-1999. All properties are single-family residential homes located within 0.25 miles of the coastline. The total sample size is 6,327. Of these, 2,439 are located in a spezial flood hazard area (SFHA), which means they have been declared to be at higher risk of flooding than the remaining 3,888 properties outside the SFHA. However, these homes are alse enjoying overall nicer amenities, such as proximity to beach, water, and views. Thus, it is ex-ante not clear which effect will dominate - the flood risk effect or the amenity effect.

The outcome variable of interest is sale price, in $1,000. The main objective of this exercise is to estimate the combined SFHA & amenity effect on home prices, and check which effect is stronger.

The data are sorted by treatment (SFHA=1 properties first, followed by SFHA=0 properties), and by sales date within treatment. The variables are as follows:

(1) DQid original property ID
(2) saleyr year of sale
(3) salemonth month of sale
(4) saleday calendar day of sale
(5) saledateE sale date in elapsed days since 1/1/1960
(6) price000 sale price in 1000's of 2014 dollars
(7) SFHA 1= located in SFHA zone
(8) age age of structure, years
(9) sqft00 square footage, in 100's
(10) lot000 lot size, in 1000 sqft
(11) bedrooms total number of bedrooms
(12) bathrooms total number of baths
(13) elev10 elevation in meters at 10 meter resolution
(14) ISMi distance to nearest Interstate, miles
(15) PAMi distance to nearest principal artery, miles
(16) beaMi distance to nearest beach, miles
(17) hidMi distance to nearest high-density development, miles
(18) coaestMi miles to nearest coast or estuary
(19) reslkMi miles to nearest lake, pond, or reservoir
(20) ag10 acreage of ag land w/in 1000m, most current
(21) ind10 acreage of ind. land w/in 1000m, most current
(22) op10 acreage of open land w/in 1000m, most current

The following loads in the data (including all variable names), and saves it immediately in R's internal ("rda") format:

```
rm(list=ls())
data <- read.table('/Users/nima/AAEC5126/data/CTfloodzones.txt',
                   sep = "\t", header = TRUE)
```

(1) Let the dependent variable be "price000," the treatment variable "SFHA,", and let the explanatory data **X** include all variables listed above from "age" to "op10" (15 variables),

in addition to a constant term where needed.

Check for overlap and show results in a table - which explanatory variables raise red flags by exceeding the recommended overlap core of 0.25 (in absolute terms)? Explain in words which group, treated or controls, has relatively larger or smaller values for these red flag variables.

```r
attach(data)
n <- nrow(data)
n1 <- sum(SFHA == 1)
n0 <- n - n1
y <- price000
X <- data[, 8:22]
#
y1 <- y[1:n1]
y0 <- y[(n1 + 1):n]
my1 <- mean(y1)
my0 <- mean(y0)
sy1 <- sd(y1)
sy0 <- sd(y0)
ndiffy <- (my1 - my0) / sqrt(sy1 ^ 2 + sy0 ^ 2)
#
X1 <- X[1:n1, ]
X0 <- X[(n1 + 1):n, ]
mX1 <- colMeans(X1)
mX0 <- colMeans(X0)
sX1 <- apply(X1, 2, sd)
sX0 <- apply(X0, 2, sd)
ndiffX <- as.vector((mX1 - mX0) / sqrt(sX1 ^ 2 + sX0 ^ 2))
#
tt <- data.frame(col1 = c("price000", colnames(data[, 8:22])),
                 col2 = c(ndiffy, ndiffX))
colnames(tt) <- c("variable", "norm.diff")
```

The results for the dependent variable indicates that on average the price for houses located in SFHA zones are higher. As for the independent variables, the normalized differences bigger than 0.25 standard deviations will be substantial. In that case one may want to be suspicious of simple methods like linear regression with a dummy for the treatment variable. When we have poor overlap, then the treatment effect may be correlated with explanatory variables. Regarding the absolute value of calculated normalized differences, variables `elev10` and `coaestMi` violate this condition. The results is logical as we expect zones with lower altitude and nearer to coast to be more likely to be harmed by floods.

(2) Estimate the ATT via difference in means, pooled regression adjustment, and regression adjustment using separate equations, deriving standard errors and t-values as in script `mod5s1`. Show all results in a combined table, as in the lecture script.

TABLE 4. normalized differences

variable	norm.diff
price000	0.24
age	-0.06
sqft00	0.03
lot000	-0.14
bedrooms	0.01
bathrooms	0.11
elev10	-1.12
ISMi	-0.08
PAMi	0.02
beaMi	-0.10
hidMi	-0.09
coaestMi	-0.65
reslkMi	-0.07
ag10	-0.21
ind10	-0.02
op10	-0.20

- Estimation via difference in means:

```
m1 <- (my1 - my0)
sem1 <- sqrt(sy1 ^ 2 / (n1) + sy0 ^ 2 / (n0))
tm1 <- m1 / sem1
#
m1ATT <- m1
sem1ATT <- sem1
tm1ATT <- tm1
```

- Estimation via pooled regression adjustment:

```
X <- cbind(rep(1, n), SFHA, data.matrix(data[, 8:22]))
bols <- solve((t(X)) %*% X) %*% (t(X) %*% y)
e <- as.vector(y - X %*% bols)
S <- diag(e ^ 2)
Vb <- solve((t(X)) %*% X) %*% t(X) %*% S %*% X %*% solve((t(X)) %*% X)
se = sqrt(diag(Vb))
tval = bols / se
#
m2 <- as.vector(bols[2])
sem2 <- as.vector(se[2])
tm2 <- as.vector(tval[2])
#
m2ATT <- m2
sem2ATT <- sem2
tm2ATT <- tm2
```

- Estimation via regression adjustment using seperate equations:

```
X <- cbind(rep(1, n), data.matrix(data[, 8:22]))
#eliminate train as a explanatory variable
y1 <- as.matrix(y[1:n1])
y0 <- as.matrix(y[(n1 + 1):n])
X1 <- X[1:n1, ]
X0 <- X[(n1 + 1):n, ]
#
b1 <- solve((t(X1)) %*% X1) %*% (t(X1) %*% y1)
b0 <- solve((t(X0)) %*% X0) %*% (t(X0) %*% y0)
#
#ATE
y1p <-
  X %*% b1 #create predictions for treated outcome for ALL observations
y0p <-
  X %*% b0 #create predictions for UNtreated outcome for ALL observations
m3 <- mean(y1p - y0p)
#ATT
m3ATT <- mean(y1p[1:n1] - y0p[1:n1])
#
#
#run bootstrap to get s.e.'s (see Wooldridge, p. 918)
################################
R <- 1000 #number of bootstrap replications
out <- rep(0, R) #will collect ATE result for each replication
outATT <- rep(0, R) #will collect ATT result for each replication
com1 <- cbind(y1, X1) #glue y1 and X1 together
com0 <- cbind(y0, X0)

for (i in 1:R) {
  int1 <- com1[sample(nrow(com1), n1, replace = TRUE),]
  #sample n1 id's with replacement (this allows for multiple entries)
  y1r <- int1[, 1]
  X1r <- int1[, 2:dim(com1)[2]]
  b1 <- solve((t(X1r)) %*% X1r) %*% (t(X1r) %*% y1r)
  #
  int0 <- com0[sample(nrow(com0), n0, replace = TRUE),]
  #sample n1 id's with replacement (this allows for multiple entries)
  y0r <- int0[, 1]
  X0r <- int0[, 2:dim(com0)[2]]
  b0 <- solve((t(X0r)) %*% X0r) %*% (t(X0r) %*% y0r)
  #
  Xr <- rbind(X1r, X0r)
  y1rp <- Xr %*% b1
  y0rp <- Xr %*% b0
  #ATE
  out[i] <- mean(y1rp - y0rp)
  #ATT
  outATT[i] <- mean(y1rp[1:n1] - y0rp[1:n1])
}
sem3 <- sd(out)
tm3 <- m3 / sem3
#
sem3ATT <- sd(outATT)
tm3ATT <- m3ATT / sem3ATT
```

Table 5. Combined estimation results for **ATE**

estimator	estimate	s.e.	t-value
difference in means	95.372	7.430	12.837
pooled regression	42.578	6.595	6.456
separate regressions	9.586	20.328	0.472

Table 6. Combined estimation results for **ATT**

estimator	estimate	s.e.	t-value
difference in means	95.372	7.430	12.837
pooled regression	42.578	6.595	6.456
separate regressions	57.770	7.478	7.725

(3) Comment on your results - are they similar or not? Which effect appears to be stronger - the risk effect or the amenity effect? Which estimate would you pick if you had to choose among those three? Provide some rationale.

Comparing the results depicted on the two table above: For ”different in means”, ATT and ATE are identical. This is obviously by design and we would not expect this method to distinguish between the two. The same goes for ”pooled regression” estimator. For our most general model, separate regression model adjustment, we get separate estimates with very different point estimates across groups. The third model (with the least assumptions) is the one we can usually trust over the first two estimators. We need to take into account that the t-value of the third does not suggest significance, whereas it does for the first two models. Overall, the results shows the positive influence of the property being located in SFHA zone on its price.

3. Q3: Estimation of treatment effects via matching

Use the same data as for Q2, and lecture script `mod5s3` for guidance. Use 1-neighbor matching ($M = 1$) without forcing exact matches ($Me = 0$), and use all variables in $\mathbf{X}$ described above for both matching and the regression adjustment.

```
detach()
rm(list=ls())
data <- read.table('/Users/nima/AAEC5126/data/CTfloodzones.txt',
                   sep = "\t", header = TRUE)
attach(data)
```

(1) Find matches and check for overlap (="balance") - show the overlap results in a table. How do these scores compare to those from the unmatched data in Q2? Are there any noteworthy improvements (lower overlap score, in absolute terms) for some variables that would indicate that the data are now better balanced?

```
y <- as.matrix(price000)
n <- nrow(y)
w <- as.matrix(SFHA)
#
f0 <- find(w == 0)
f1 <- find(w == 1)
#
y0 <- as.matrix(y[f0])
y1 <- as.matrix(y[f1])
n0 <- nrow(y0)
n1 <- nrow(y1)
#
X <- as.matrix(data[, 8:22])
X1 <- as.matrix(X[f1, ])
X0 <- as.matrix(X[f0, ])

k <- ncol(X)
#
M <- 1 #min. number of matches per treated obs.
Me <- 0 #number of variables that must match exactly
```

```
y0hat = rep(n1, 1)  # will collect counterfactual estimates
IMatch <- vector('list', n1) #collects matching info for each treatment obs
JMivec = rep(0, n1) #collects number of matches used for each treatment obs
KMlvec = rep(0, n)
#collects weighted counts for how often each control is used as match
# will be zero for treated obs's
#
Vi <- 1 / as.matrix(diag(var(X))) #vector of inverted variances
```

```
pen <- c(rep(1, (k - Me)), rep(1000, Me))# penalties for variance terms
Vi <- Vi * pen  #penalize for exact matches
#
for (i in 1:n1) {
  xi <- as.matrix(X1[i, ])
  int1 <- (repmat(xi, n0, 1) - X0) ^ 2 #squared differences, k by n0
  int2 <- repmat(Vi, n0, 1)
  int <- as.matrix(sqrt(rowSums(int1 * int2))) #n0 by 1 matching scores
  #
  Imat <- cbind(f0, y0, int)
  Imat <- Imat[order(int), ] #sort in order of lowest to highest matching score
  int <- Imat[, 3]
  #find >= M observations with lowest distance, allowing for ties
  g <- 1 #counter for unique values - we need exactly M
  j <- 1 #counts over observations
  #
  while (g < (M + 1)) {
    d <- int[j] - int[j + 1]
    if (d != 0)
      g <- g + 1
    j = j + 1
  }
  #
  y0hat[i] = mean(Imat[1:(j - 1), 2])
  IMatch[[i]] = Imat[1:(j - 1),]
  JMivec[i] <- j - 1
  #
  f <- Imat[1:(j - 1), 1] #set of indices for controls
  KMlvec[f] <- KMlvec[f] + (1 / (j - 1))
}
```

```
chosen <- find(KMlvec > 0)
#index vector for controls that were selected as a match at least once
X0m <- as.matrix(X[chosen, ]) #covariate for matched controls
y0m <- as.matrix(y[chosen]) #outcome for matched controls
#
my1 <- mean(y1)
my0 <- mean(y0m)
sy1 <- sd(y1)
sy0 <- sd(y0m)
ndiffy <- (my1 - my0) / sqrt(sy1 ^ 2 + sy0 ^ 2)
#
mX1 <- colMeans(X1)
mX0 <- colMeans(X0m)
sX1 <- apply(X1, 2, sd)
sX0 <- apply(X0m, 2, sd)
```

```
ndiffX <- as.vector((mX1 - mX0) / sqrt(sX1 ^ 2 + sX0 ^ 2))
```

Table 7. normalized differences treated vs. chosen controls

variable	norm.diff
price000	0.16
age	-0.03
sqft00	-0.04
lot000	-0.11
bedrooms	-0.02
bathrooms	0.01
elev10	-1.39
ISMi	-0.09
PAMi	-0.08
beaMi	-0.03
hidMi	-0.03
coaestMi	-0.28
reslkMi	-0.02
ag10	-0.06
ind10	-0.02
op10	-0.11

Now, the table shows that all the overlap score are below 0.25 and therefore we can expect for ATE and ATT to be well-identified, using these explanatory variables and the matched control observations.

(2) Compute the uncorrected and corrected ATTs, along with consistent standard errors and t-values following script `mod5s3`. Report this output in a table.

Computing uncorrected estimator & percentage of exact matches:

```
ATT <- mean(y1 - y0hat) #correct, same as Matlab

# of exact matches
#################################
exact <- rep(0, n1) #collects counts of exact matches for education
#
for (i in 1:n1) {
  #Initial code: fi<-IMatch[[i]][,1]
  #Important note: With M=1, IMatch will usually only have one row,
  #but R doesn't understand that this is a matrix construct with one row,
  #and thus creates an error message when you call the first column of that "matrix."
  #BETTER:
  fiprep <- matrix(IMatch[[i]], ncol = 3) # now it gets it,
  # a row with 3 columns (but multiple rows still OK)
  fi <- fiprep[, 1]
  #id's of matched observations
  int2 <- X1[i, k] - X[fi, k] #difference in educ
  #
  fAll <- find(int2 == 0)
  exact[i] = length(fAll)
}

pAll <- sum(exact) / sum(JMivec)
```

Computing consistent standard errors for uncorrected estimator

```
# compute sighat
################################
sumterm = rep(0, n1)

for (i in 1:n1) {
  #outi=IMatch[[i]] #same problem as above
  outi <- matrix(IMatch[[i]], ncol = 3)
  JMi <- nrow(outi) #number of obs's used for matching
  y0l <- outi[, 2]
  int <-
    sum((y1[i] - y0l - ATT) ^ 2) #y0l is JMi by 1, the other terms are scalars
  sumterm[i] <- (1 / JMi) * int
}

# compute variance
###############################
sighat <- (1 / (2 * n1)) * sum(sumterm)
VarATT <- (sighat / n1 ^ 2) * (n1 + sum(KMlvec ^ 2))
seATT <- sqrt(VarATT) #correct, same as Matlab
tATT <- ATT / seATT
```

Computing corrected estimator:

```
# run auxiliary regression
############################
Xfull <- cbind(rep(1, n), X[, 1:(k - Me)])
#add constant, drop variables that must match exactly
X1full = cbind(rep(1, n1), X1[, 1:(k - Me)])
X0full = cbind(rep(1, n0), X0[, 1:(k - Me)])
#
kfull <- ncol(Xfull)
#
fK <- find(KMlvec != 0) #use only matched obs
Kaux = KMlvec[fK]
yaux = sqrt(Kaux) * y[fK]
#weighting by (weighted) number of time an obs. was matched
Xaux <- repmat(sqrt(Kaux), 1, kfull) * Xfull[fK, ]
#
```

```
baux <- solve((t(Xaux)) %*% Xaux) %*% (t(Xaux) %*% yaux)
y1pred <- X1full %*% baux
#
# Compute estimator
#########################
y0hat <- rep(0, n1) # will collect counterfactual estimates

for (i in 1:n1) {
  fiprep <- matrix(IMatch[[i]], ncol = 3) #same correction as above
  fi <- fiprep[, 1]  # id's of matched observations
  yl <- fiprep[, 2]  # outcomes for matched controls
  y0lpred <- Xfull[fi, ] %*% baux #predictions for matched controls
  y0hat[i] <- mean(yl - y0lpred + y1pred[i])
  #last element is a scalar, but R gets it
}
#
ATTc <- mean(y1 - y0hat) #same as Matlab
```

Computing consistent standard errors for corrected estimator:

```
# compute sighat
################################
sumterm = rep(0, n1)

for (i in 1:n1) {
  outi <- matrix(IMatch[[i]], ncol = 3)
  JMi <- nrow(outi) #number of obs's used for matching
  y0l <- outi[, 2]
  int <-
    sum((y1[i] - y0l - ATTc) ^ 2) #y0l is JMi by 1, the other terms are scalars
  sumterm[i] <- (1 / JMi) * int
}

# compute variance
###############################
sighat <- (1 / (2 * n1)) * sum(sumterm)
VarATTc <- (sighat / n1 ^ 2) * (n1 + sum(KMlvec ^ 2))
seATTc <- sqrt(VarATTc)
tATTc <- ATTc / seATTc
```

(3) How does the corrected ATT compare to its uncorrected counterpart? Which one would you choose and why?

The table shows us that we demanded at least one match per treated, but no exact match needed. The uncorrected ATT is around 70k in 2014 dollars, and about 28k for the corrrected ATT.

(4) Across all ATT estimates from Q2 and Q3, which one would you choose and why?

Table 8. Combined estimation results for ATT

estimator	estimate	s.e.	t-value
min. # matches	1.000	-	-
# vars, exact match	0.000	-	-
% exact matches	0.000	-	-
ATT, uncorrected	70.072	10.386	6.747
ATT, corrected	28.361	10.507	2.699

The Matching model is the state-of-the-art model and via tweaking the parameters we can force the number of variables that may be matched via penalizing the distance of the variable of interest. For example, we may force nearest neighbors to have the exact same number of bedrooms and bathrooms as the target property. Also notice that only after performing Matching, our check for overlap turned out to be satisfactory. While it is still not perfect overlap, it is certainly a considerable improvement.

(5) In sum, what can you conclude regarding the challenge of estimating flood risk effects on home prices in presence of (unobserved) coastal amenities? Which effect is likely going to dominate? What additional data might help to directly control for amenities in the econometric model?

Separating treated from control homes at a given time-of-sale can be beneficial. We have excluded this data from our analysis. Also the analysis did not take into account the school zones. We observed that SFHA has a positive influence on the price level of homes, and that Matching has the potential to provide us a more adequate estimate.