Reproducing Regressions from Acemoglu, Johnson, and Robinson (2001)

Bennett Smith-Worthington

In this R markdown file, I verify/test some of the regressions, error estimates, and instrument strengths by using various econometric techniques as demonstrated in the sections below. All notation comes from Bruce Hansen's *Econometrics*, and the notation is on page xxii-xxiii. For any questions, please email be at "bs3248@columbia.edu".

```
## importing data from AJR 2001
data <- read.delim("/Users/bennettsw/Downloads/AJR2001.txt")
```

Regressions for political risk, settler mortality rates

Regression 1: log(GDP per capita) and risk

The equation being estimated is

$$\log(GDP\hat{per}Capita) = \beta risk$$

```
### 12.86 OLS regression
## Independent variable X is `risk' variable, and dependent variable is present day log(GDP):
X = cbind(data$risk, 1)
Y = data$loggdp
B_ols <- solve(t(X)%*%X)%*%t(X)%*%Y
```

Thus, the ordinary least squares estimate of β is:

```
print(B_ols[1])
```

```
## [1] 0.516187
```

Regression 2: Reduced form for log mortality and risk

The equation being estimated is

$$risk = \theta_{rf} \log(mortality) + \hat{u}$$

```
X = cbind(data$logmort0, 1)
Y = data$risk
Theta_rf = solve(t(X)%*%X)%*%(t(X)%*%Y)
```

This returns a reduced form estimate of the coefficient θ (first row) as:

```
print(Theta_rf[1])
```

```
## [1] -0.6132892
```

Regression 3: 2SLS estimation of impact of risk on log(GDP per capita)

By using $\log(mortality)$ as an instrument for $risk$, we have that the equation being estimated is

$$\log(GDP\hat{per}Capita) = \beta_{2SLS} risk$$

```
## Preliminaries: Establishing X,Y,Z
X = cbind(data$risk, 1)
Y = data$loggdp
Z = cbind(data$logmort0, 1)
## Constructing P_Z, X_hat
P_Z = Z%*%solve(t(Z)%*%Z)%*%t(Z) #Hat matrix for log(mortality)
X_hat = P_Z%*%X
B_2SLS <- solve(t(X_hat)%*%X_hat)%*%t(X_hat)%*%Y
```

Using 2SLS returns a coefficient that is roughly double the first regression's coefficient; β_{2SLS} is .1 off of the published result, and is:

```
print(B_2SLS[1])
```

```
## [1] 0.9294897
```

Testing assumptions of homoskedasticity for OLS, Reduced Form, and 2SLS Regressions

```
# OLS Standard Errors
# Steps: contsruct residuals and Qxx to get variance of B_ols,
# then take sqrt and evaluate at 1,1
Y <- data$loggdp
X <- cbind(data$risk, 1)
n <- length(Y)
k <- 2
# Homoskedastic standard error
e_hat <- Y-X%*%B_ols
Q_xx <- t(X)%*%X
Q_xx_hat <- (1/n)*Q_xx
s2 <- (1/(n-k))*(t(e_hat)%*%e_hat)
V_hat_ho <- solve(Q_xx)*s2[1,1]
se_ho <- sqrt(V_hat_ho[1,1])
## Heteroskedastic standard error
omega <- matrix(0,dim(X)[2],dim(X)[2])
# Following for loop estimates variance for each element of the matrix,
# thus creating a heteroskedastic error variance matrix.
for (val in 1:n) {
omega <- omega + ( (1/n) * (e_hat[val]^2) * (X[val,cbind(1,2)]) %*%
t(X[val,cbind(1,2)]) )
}
V_hat_het <- (solve((t(X) %*% X)/n) %*% omega%*% solve((t(X) %*% X)/n))
se_het <- sqrt(V_hat_het[1,1]/n)
print(se_het)
```

```
## [1] 0.05029619
```

Reduced Form Standard Errors

```
X = cbind(data$logmort0, 1)
Y = data$risk
n <- length(Y)
k <- 2
#### Homoskedastic standard error
e_hat <- Y - X%*%Theta_rf
Q_xx <- t(X)%*%X
s2 <- (1/(n-k))*(t(e_hat)%*%e_hat)
V_hat_ho <- solve(Q_xx)*s2[1,1]
se_ho <- sqrt(V_hat_ho[1,1])
print(se_ho)
```

```
## [1] 0.1269412
```

```
### Heteroskedastic standard error
omega <- matrix(0,dim(X)[2],dim(X)[2])
# Following for loop estimates variance for each element of the matrix,
# thus creating a heteroskedastic error variance matrix.
for (val in 1:n) {
omega <- omega + ( (1/n) * (e_hat[val]^2) * (X[val,cbind(1,2)]) %*%
t(X[val,cbind(1,2)]) )
}
# Defining estimated heteroskedastic variance matrix
V_hat_het <- (solve((t(X) %*% X)/n) %*% omega%*% solve((t(X) %*% X)/n))
se_het <- sqrt(V_hat_het[1,1]/n)
print(se_het)
```

```
## [1] 0.1493945
```

2SLS Standard Errors

```
X = cbind(data$risk, 1)
Y = data$loggdp
Z = cbind(data$logmort0, 1)
n <- length(Y)
k <- 2
e_hat <- Y - X%*%B_2SLS
Q_xx <- t(X)%*%X
Q_xz <- t(X) %*% Z
Q_zz <- (t(Z) %*% Z)
Q_zx <- (t(Z) %*% X)
s2 <- (1/(n-k))*(t(e_hat)%*%e_hat)
# Defining homoskedastic variance matrix
V_hat_ho <- s2[1,1] * solve( (Q_xz/n)%*% solve(Q_zz/n) %*% ((Q_zx)/n))
se_ho <- sqrt(V_hat_ho[1,1]/n) # standard error for homoskedasticity
# Homoskedastic standard error:
print(se_ho)
```

```
## [1] 0.1560901
```

```
### Heteroskedastic standard error
omega <- matrix(0,2,2)
# The following loop enters the heteroskedastic error variance matrix by
```

```r
# entering the estimate for variance in each element of the matrix.
for (val in 1:n) {
omega <- omega + ( (1/n) * (e_hat[val]^2) * (Z[val,cbind(1,2)]) %*%
t(Z[val,cbind(1,2)]) )
}
A <- solve( ((Q_xz)/n) %*% solve((Q_zz)/n) %*% ((Q_zx)/n) ) %*% ((t(X) %*% Z)/n) %*% solve( (Q_zz)/n)
V_hat_het <- A %*% omega %*% t(A) # Defining heteroskedastic variance matrix
# Heteroskedastic standard error:
print((V_hat_het[1,1]/n)**.5)
```

```
## [1] 0.1700872
```

Since all of these homoskedastic errors found in this section are the errors printed in AJR (2001), one can conclude that the **authors assumed homoskedastic errors.**

Calculating 2SLS Coefficient via Indirect Least Squares

```r
X <- cbind(data$risk, 1)
Y <- data$loggdp
Z <- cbind(data$logmort0, 1)
B_ILS <- solve(t(Z) %*% X) %*% (t(Z) %*% Y)
# Indirect Least Squares coefficient:
print(B_ILS[1])
```

```
## [1] 0.9294897
```

Thus the indirect least squares (ILS) estimate is the same as the coefficient estimated via 2SLS

Calculating 2SLS Coefficient via Two Stage Approach

```r
P_Z = Z%*%solve(t(Z)%*%Z)%*%t(Z)
X_hat = P_Z%*%X
## Beta_TSLS
B_TSLS <- solve(t(X_hat)%*%X_hat)%*%t(X_hat)%*%Y
print(B_TSLS[1])
```

```
## [1] 0.9294897
```

Therefore, we see that the coefficient estimated by the two-stage strategy gives the same result.

Calculating 2SLS Coefficient via Control Function Approach

```r
## See p. 370 of Bruce Hansen's Econometrics book to read about this approach.
X <- cbind(data$risk)
Y <- data$loggdp
Z <- cbind(data$logmort0, 1)
B_OLS <- solve(t(Z)%*%Z)%*%t(Z)%*%X
X_hat <- Z%*%B_OLS
u_hat <- X - X_hat
X<- cbind(data$risk, u_hat, 1)

B_OLS <- solve(t(X)%*%X)%*%(t(X)%*%Y)
print(B_OLS[1])
```

```
## [1] 0.9294897
```

Therefore, it is clear that the control function approach gives the same coefficients (first and third) ## Checking for Weak or Strong Instruments via Stock-Yogo Test

```
X <- data$risk
A <- cbind(data$logmort0, data$logmort0^2, 1)
B0 <- (solve(t(A) %*% A) %*% t(A) %*% X)
ehat <- X - A %*% B0
e_tilde2 <- sum((X-mean(X))^2)
e_hat2 <- (t(ehat) %*% ehat)[1,1]
F <- ((e_tilde2-e_hat2)/(2))/(e_hat2/(n-3))
print(F)
```

```
## [1] 18.42273
```

Since the F-test is greater than 10, then logic of Stock-Yogo indicates that $\log(mortality)$ is not a weak instrument.