

Part A

We find that in case of OLS estimator, the average value of β_1 is almost the same across different sample sizes (but it is not close to the actual value of 1) whereas the standard deviation decreases with increase in sample size. Now, in case of strong instrument, average value of β_1 approaches its actual value and the standard deviation also decreases with an increase in sample size. Now, in the case of weak instrument, the average value of β_1 is very high for small sample size and also the standard deviation value is too high but it decreases with increase in sample size. So, overall we can conclude that the strong instrument of IV estimator performed the best in terms of evaluating the β_1 value with very less standard deviation value followed by OLS estimator. Weak instrument of IV estimator performed the worst. Lastly we also notice that the increase in sample size (for any of the estimator present) leads to increase in the performance of the estimator and helps in providing a more accurate result.

Simulation results

			Instrument is strong		Instrument is weak	
	β_1		$\beta_{1,IV}$		$\beta_{1,IV}$	
	Average	Standard Deviation	Average	Standard Deviation	Average	Standard Deviation
n=20	1.5033	0.2092	0.9479	0.3753	4.214	262.738
n=100	1.5004	0.0877	0.9923	0.1296	1.2853	43.5204
n=400	1.5003	0.0434	0.9985	0.0631	0.9602	0.3218

Appendix

R code

```
prepare_data <- function(n,r,sxz) {  
  S <- matrix(c(1, 0.5, 0,  
               0.5, 1, sxz,  
               0, sxz, 1),  
             nrow = 3, ncol = 3, byrow = TRUE)  
  mu <- c(u = 0, x = 1, z = 1)  
  set.seed(781)  
  data <- mvtnorm::rmvnorm(n*r, mean = mu, sigma = S)  
  u <- data[,1]  
  x <- data[,2]  
  X <- cbind(1, x)  
  z <- data[,3]  
  Z <- cbind(1, z)  
  y <- -2 + x + u  
  df <- list("X" = X, "y" = y, "Z" = Z)  
  return(df)  
}  
  
OLS <- function(X,y,n,r) {  
  B_ols = matrix(, nrow = 1, ncol = 2)  
  for (i in 1:r) {  
    Xi <- X[(n*(i-1)+1):(n*i), ]  
    yi <- y[(n*(i-1)+1):(n*i)]  
    b_ols <- solve(t(Xi)%*%Xi) %*% t(Xi) %*% yi  
    B_ols <- rbind(B_ols, t(b_ols))  
  }  
  B1_ols <- B_ols[1:r+1,2]  
  cat("n:", n)  
  cat("\nAverage of B1_ols:", mean(B1_ols))
```

```

    cat("\nSD of B1_ols:", sd(B1_ols))
  }
IV <- function(X,Z,y,n,r,sxz) {
  B_IV = matrix(, nrow = 1, ncol = 2)
  for (i in 1:r) {
    Xi <- X[(n*(i-1)+1):(n*i), ]
    Zi <- Z[(n*(i-1)+1):(n*i), ]
    yi <- y[(n*(i-1)+1):(n*i)]
    b_IV <- solve(t(Zi)%*%Xi) %*% t(Zi) %*% yi
    B_IV <- rbind(B_IV, t(b_IV))
  }
  B1_IV <- B_IV[1:r+1,2]
  cat("\nsxz:", sxz)
  cat("\nAverage of B1_IV:", mean(B1_IV))
  cat("\nSD of B1_IV:", sd(B1_IV))
}

r <- 10000
N <- c(20, 100, 400)
Sxz <- c(0.8,0.2)
for (n in N) {
  for (sxz in Sxz) {
    df <- prepare_data(n,r,sxz)
    X <- df$X; y <- df$y; Z <- df$Z
    OLS(X,y,n,r)
    IV(X,Z,y,n,r,sxz)
    cat("\n-----\n")
  }
}

```