FINC 584 PS3

17 October, 2022

Problem 1.a

```
ak_full <- readxl::read_xlsx(file.path(proj, "AK1991.xlsx"))

x_full <- matrix(data = c(rep(1,nrow(ak_full)), ak_full$edu), ncol = 2)
y_full <- matrix(data = ak_full$logwage)

beta_conventional_full <- solve(t(x_full) %*% x_full)%*%(t(x_full)%*%y_full)

resid_full <- y_full - x_full %*% beta_conventional_full
sigma2_hat_full <- mean(resid_full^2)
beta_conventional_se_full <- sqrt(diag(sigma2_hat_full *
                                         (solve(t(x_full) %*% x_full))))

kable(data.frame(beta_hat = beta_conventional_full,
                 beta_se = beta_conventional_se_full,
                 row.names = c("$\\beta_0$", "$\\beta_1$")),
      format = 'pandoc')
```

	beta_hat	beta_se
β_0	4.995182	0.0044644
β_1	0.070851	0.0003386

Problem 1.b

```
ak<- ak_full %>% slice(1:5000)

## Compute beta_hats for conventional and robust
x <- matrix(data = c(rep(1,nrow(ak)), ak$edu), ncol = 2)
y <- matrix(data = ak$logwage)
beta_conventional <- solve(t(x) %*% x)%*%(t(x)%*%y)
resid <- y - x %*% beta_conventional

## Compute SEs for conventional method
sigma2_hat <- mean(resid^2)
beta_conventional_se <- sqrt(diag(sigma2_hat * (solve(t(x) %*% x))))[2]

## Compute SEs for robust method
sandwich_bread <- solve(t(x) %*% x)
```

```
sandwich_meat <- t(x) %*% diag(as.vector(resid)^2)  %*% x

beta_robust_se <- sqrt(diag(sandwich_bread %*% sandwich_meat %*% sandwich_bread))[2]

## Bootstrap
set.seed(0)
beta_hat_bootstrap <- rep(NA, 5000)

for(i in 1:length(beta_hat_bootstrap)){
  ak_sample <- ak[sample(1:5000, replace = T),]
  beta_hat_bootstrap[i] <- lm(logwage ~ edu, data = ak_sample)$coefficients['edu']
}

beta_bootstrap_se <- sd(beta_hat_bootstrap)

kable(data.frame(beta_hat_1 = c(beta_conventional_se, beta_robust_se,
                 beta_bootstrap_se),
                 row.names = c("Conventional", "Robust", "Bootstrap")),
      )
```

	beta_hat_1
Conventional	0.0029033
Robust	0.0029218
Bootstrap	0.0029199

```
# ggplot(data = data.frame(beta_hat_bootstrap), aes(x = beta_hat_bootstrap)) +
#   geom_histogram(fill = 'steelblue', color = 'black') +
#   labs(title = 'Distribution of beta_hat_1') +
#   theme_bw() +
#   theme(plot.title = element_text(hjust = 0.5))
```

The results are pretty much what I expected. All three standard errors are roughly similar. The robust standard errors are larger than the conventional OLS estimators which is expected. The bootstrap method is similar to the other two methods in terms of efficiency.

Problem 1C

```
confidence_interval <- function(b, b_se){
  return( abs((b - beta_conventional_full[2]))/b_se < qnorm(.975))
}

conventional_method <- function(x,y){

  beta_conventional <- solve(t(x) %*% x)%*%(t(x)%*%y)
  resid <- y - x %*% beta_conventional

  ## Compute SEs for conventional method
  beta_conventional_se <- sqrt(diag(mean(resid^2) * (solve(t(x) %*% x))))[2]
```

```
  return(confidence_interval(beta_conventional[2], beta_conventional_se))
}

robust_method <- function(x,y){

  beta_conventional <- solve(t(x) %*% x)%*%(t(x)%*%y)
  resid <- y - x %*% beta_conventional

  ## Compute SEs for robust method
  sandwich_bread <- solve(t(x) %*% x)
  sandwich_meat <- t(x) %*% diag(as.vector(resid)^2) %*% x

  beta_robust_se <- sqrt(diag(sandwich_bread %*% sandwich_meat %*% sandwich_bread))[2]

  return(confidence_interval(beta_conventional[2], beta_robust_se))

}


bootstrap_method <- function(data,n,m){
  bootstrap_estimates <- rep(NA, m)

  for(i in 1:length(bootstrap_estimates)){
    bootstrap_estimates[i] <- lm(logwage ~ edu,
                                 data = data[sample(1:n, n, replace = T),])$coefficients['edu']
  }

 # return((mean(bootstrap_estimates) - beta_conventional_full[2])/sd(bootstrap_estimates))

 return(confidence_interval(mean(bootstrap_estimates), sd(bootstrap_estimates)))

}


problem1c <- function(n, m, data){

  mc_sample <- data[sample(1:nrow(data), n, replace = T),]

  x <- matrix(data = c(rep(1,n), mc_sample$edu), ncol = 2)
  y <- matrix(data = mc_sample$logwage)

  return(data.frame(n = n,
                    conventional = conventional_method(x,y),
                    robust = robust_method(x,y),
                    bootstrap = bootstrap_method(mc_sample,n,m)
                    ))
}

iter <- 5000

results <- map_dfr(.x = c(rep(50, iter), rep(100, iter), rep(500, iter)), .f = problem1c, data = ak_full

results %>% group_by(n) %>%
```

```
  summarize_all( ~sum(.)*100/iter)
```

```
## # A tibble: 3 x 4
##       n conventional robust bootstrap
##   <dbl>        <dbl>  <dbl>     <dbl>
## 1    50         90.1   92.1      92.6
## 2   100         91.3   94.2      93.9
## 3   500         92.8   95.6      95
```

Problem 1d

$$h(\hat{\beta}) = e^{\hat{\beta}_0+9\hat{\beta}_1} e^{\frac{\sigma^2}{2}} - e^{\hat{\beta}_0+9\hat{\beta}_1} e^{\frac{\sigma^2}{2}}$$

$$h(\hat{\beta}) = e^{\frac{\sigma^2}{2}} \left(e^{\hat{\beta}_0+9\hat{\beta}_1} - e^{\hat{\beta}_0+8\hat{\beta}_1} \right)$$

Thus,

$$\nabla h(\hat{\beta}) = \left(0, e^{\frac{\sigma^2}{2}} \left(9e^{\hat{\beta}_0+9\hat{\beta}_1} - 8e^{\hat{\beta}_0+8\hat{\beta}_1} \right) \right)$$

By the delta method, we have that

$$\sqrt{(n)}(h(\hat{\beta}) - h(\beta)) \rightarrow^D N \left(0, \nabla h(\hat{\beta}) * \Sigma * \nabla (h(\hat{\beta})^T \right)$$

```
e_sigma <- exp(sigma2_hat_full/2)
b0 <- beta_conventional_full[1]
b1 <- beta_conventional_full[2]

del_h <- matrix(c(0, e_sigma*(9*exp(b0 + 9*b1) - 8*exp(b0 + 8*b1))), ncol = 2)
S <- sigma2_hat_full * (solve(t(x_full) %*% x_full))

beta_hat_1d <- e_sigma*(exp(b0 + 9*b1) - exp(b0 + 8*b1))
del_h %*% S %*% t(del_h)
```

```
##             [,1]
## [1,] 0.03219243
```