hw1

1

Prove each of the following. You may use the results from previous parts of the problem in further proofs. For each proof, I have indicated where you should start. For the purposes of these proofs here, assume that X, Y , and Z are continuous random variables. Assume that a, b and c are constants.

a

$$\begin{aligned}
E[aX + bY + c] &= \int\int (ax + by + c)p(x, y)dxdy \\
&= \int\int axp(x, y)dxdy + \int\int byp(x, y)dxdy + \int\int cp(x, y)dxdy \\
&= a\int x\Big[\int p(x, y)dy\Big]dx + b\int y\Big[\int p(x, y)dx\Big]dy + c\int\int p(x, y)dxdy \\
&= a\int xp(x)dx + b\int yp(y)dy + c \cdot 1 \\
&= aE[X] + bE[Y] + c
\end{aligned}$$

b

$$\begin{aligned}
cov(X, Y) &= E\Big[(X - E[X])(Y - E[Y])\Big] \\
&= E\Big[XY - XE[Y] - YE[X] + E[Y]E[X]\Big] \\
&= E[XY] - E[XE[Y]] - E[YE[X]] + E[E[Y]E[X]] \\
&= E[XY] - E[Y]E[X] - E[X]E[Y] + E[Y]E[X]E[1] \qquad \text{since E[X] and E[Y] are just constants} \\
&= E[XY] - E[X]E[Y]
\end{aligned}$$

c

$$
\begin{aligned}
E[Y|X=x] &= \int yp(y|X=x)dy \\
&= \int y\frac{p(y,x)}{p(x)}dy \\
&= \int\int y\frac{p(y,x,z)}{p(x)}dzdy \qquad \text{reverse marginalization} \\
&= \int\int y\frac{p(y|x,z)p(x,z)}{p(x)}dzdy \qquad \text{bayes rule} \\
&= \int \frac{p(x,z)}{p(x)}\Big[\int yp(y|x,z)dy\Big]dz \qquad \text{separating terms} \\
&= \int p(z|x)E[Y|X=x,Z=z]dz \\
&= E\Big[E[Y|X,Z]\ \Big|\ X=x\Big]
\end{aligned}
$$

The transition between the second to last and the last step relies on the fact that since we are conditioning on $X = x$, E[Y|X,Z] is just a function of Z, meaning that the outer expectation is over $p(z|x)$.

d

$$
\begin{aligned}
E[h(X,Y)] &= \int\int h(x,y)p(x,y)dxdy \\
&= \int\int h(x,y)p(y|x)p(x)dxdy \qquad \text{bayes rule} \\
&= \int p(x)\Big[\int h(x,y)p(y|x)dy\Big]dx \qquad \text{rearranging terms} \\
&= \int p(x)E[h(x,y)|X=x] \qquad \text{since we condition on x, E[h(x,y)|x] is over p(y|x)} \\
&= E\Big[E[h(X,Y)|X=x]\Big] \qquad \text{this outer expectation is just over p(x)}
\end{aligned}
$$

e

$$\begin{aligned}
Var(Y) &= E[Y^2] - E[Y]^2 \\
&= E\Big[E[Y^2|X]\Big] - E[Y]^2 \\
&= E\Big[Var(Y|X) + E[Y|X]^2\Big] - E\Big[E[Y|X]\Big]^2 \\
&= E[Var(Y|X)] + E\Big[E[Y|X]^2\Big] - E\Big[E[Y|X]\Big]^2 \\
&= E[Var(Y|X)] + Var(E[Y|X]) \qquad \text{defintion of variance}
\end{aligned}$$

2

Prove each of the following statements. In each, let $X_{(n)}$ and $Y_{(n)}$ be sequences of random variables, and let c and d be constants.

a

Choose some $\varepsilon > 0$. We can write.

$$P\bigg(\Big|(X_{(n)} + Y_{(n)}) - (c+d)\Big| \geq \varepsilon\bigg) = P\bigg(\Big|(X_{(n)} - c) + (Y_{(n)} - d)\Big| \geq \varepsilon\bigg)$$

But if the event $\Big|(X_{(n)} - c) + (Y_{(n)} - d)\Big| \geq \varepsilon$ happened then the event

- "$|X_{(n)} - c| \geq \varepsilon/2$ or $|Y_{(n)} - d| \geq \varepsilon/2$" necessarily happened as well

But because the first event happening means necessarily that the second happened, then we can say that the first event is a subset of the second event. Of course if $A \subseteq B$ then $P(A) \leq P(B)$ so we can say that

$$P\bigg(\Big|(X_{(n)} + Y_{(n)}) - (c+d)\Big| \geq \varepsilon\bigg) \leq P\bigg(\Big|X_{(n)} - c\Big| \geq \varepsilon/2 \bigcup \Big|Y_{(n)} - d\Big| \geq \varepsilon/2\bigg)$$

We also know that $P(A \cup B) \leq P(A) + P(B)$ so

$$\begin{aligned}
P\bigg(\Big|(X_{(n)} + Y_{(n)}) - (c+d)\Big| \geq \varepsilon\bigg) &\leq P\bigg(\Big|X_{(n)} - c\Big| \geq \varepsilon/2 \bigcup \Big|Y_{(n)} - d\Big| \geq \varepsilon/2\bigg) \\
&\leq P\bigg(\Big|X_{(n)} - c\Big| \geq \varepsilon/2\bigg) + P\bigg(\Big|Y_{(n)} - d\Big| \geq \varepsilon/2\bigg)
\end{aligned}$$

But since we know that $X_{(n)} \xrightarrow{p} c$ and $Y_{(n)} \xrightarrow{p} d$ then we know that

$$P\left(\left|X_{(n)} - c\right| \geq \varepsilon/2\right) + P\left(\left|Y_{(n)} - d\right| \geq \varepsilon/2\right) \to 0 \qquad \text{as } n \to \infty$$

Importantly these are sequences of numbers and not random variables so we can indeed say that since each term individually converges to zero, then so does their sum. We are not using what we were trying to prove.

But now what we have is that for any $\varepsilon > 0$,

$$\lim_{n\to\infty} P\left(\left|(X_{(n)} + Y_{(n)}) - (c + d)\right| \geq \varepsilon\right) = 0$$

which is exactly what it means for $X_{(n)} + Y_{(n)} \xrightarrow{p} c + d$

3

We simply apply Chebyshev's inequality to $X_{(n)}$, choosing any ε, to get

$$P\left(\left|X_{(n)} - E[X_{(n)}]\right| \geq \varepsilon\right) \leq \frac{var(X_{(n)})}{\varepsilon^2}$$

And since we know that $var(X_{(n)}) \to 0$, then we know that

$$\frac{1}{\varepsilon^2} \cdot var(X_{(n)}) \to \frac{1}{\varepsilon^2} \cdot 0 = 0$$

Of course probabilities are bounded below by zero, so we get that the probability expression is squeezed towards zero as n increases. And since we also know that $E[X_{(n)}] \to c$, then when we take the limit of the above expression we get

$$\lim_{n\to\infty} P\left(\left|X_{(n)} - c\right| \geq \varepsilon\right) = 0$$

which is exactly what it means for $X_{(n)} \xrightarrow{p} c$

4

We know that $\sqrt{n}(\hat{\theta}-\theta) \xrightarrow{d} N(0,\sigma^2)$ and $\hat{\sigma} \xrightarrow{p} \sigma$ and we can write

$$\frac{\sqrt{n}(\hat{\theta}-\theta)}{\hat{\sigma}} = \frac{\sigma}{\hat{\sigma}}\frac{\sqrt{n}(\hat{\theta}-\theta)}{\sigma}$$

Next we apply the continuous mapping theorem we can say that

$$\frac{\sigma}{\hat{\sigma}} \xrightarrow{p} \frac{\sigma}{\sigma} \qquad \text{and} \qquad \frac{\sqrt{n}(\hat{\theta}-\theta)}{\sigma} \xrightarrow{d} N(0,1)$$

Next Slutsky's theorem tells us that

$$\frac{\sqrt{n}(\hat{\theta}-\theta)}{\hat{\sigma}} = \frac{\sigma}{\hat{\sigma}}\frac{\sqrt{n}(\hat{\theta}-\theta)}{\sigma} \xrightarrow{d} 1 \cdot N(0,1)$$

And by definition of convergence in distribution we get that

$$\lim_{n\to\infty} P\bigg(-z_{1-\alpha/2} \leq \frac{\sqrt{n}(\hat{\theta}-\theta)}{\hat{\sigma}} \leq z_{1-\alpha/2}\bigg) = 1-\alpha$$

We can then do some rearranging within the parentheses

$$\begin{aligned}
-z_{1-\alpha/2} \leq \frac{\sqrt{n}(\hat{\theta}-\theta)}{\hat{\sigma}} \leq z_{1-\alpha/2} &= -z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \leq \hat{\theta}-\theta \leq z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \\
&= \hat{\theta} - z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \leq -\theta \leq \hat{\theta} + z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \\
&= \hat{\theta} - z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \leq \theta \leq \hat{\theta} + z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \qquad \text{symmetry of normal dist}
\end{aligned}$$

so we have that

$$\lim_{n\to\infty} P\bigg(\hat{\theta} - z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}} \leq \theta \leq \hat{\theta} + z_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{n}}\bigg) = 1-\alpha$$

as desired.

5

a

Well in OLS we choose β such that $\mathbf{X}^T(\mathbf{Y} - \mathbf{X}\beta) = 0$ ($\hat{\beta}_{OLS}$ solves this equation).

We can write this out as follows

$$\begin{pmatrix} 1 & \dots & 1 \\ X_{1,1} & \dots & X_{n,1} \\ \vdots & \ddots & \vdots \\ X_{1,k-1} & \dots & X_{n,k-1} \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

but immediately we can see that an immediate consequence of this is that

$$1 \cdot e_1 + 1 \cdot e_2 + \dots + 1 \cdot e_n = \sum_{i=1}^{n} e_i = 0$$

so of course $\frac{1}{n}\sum e_i = 0$ as well.

b

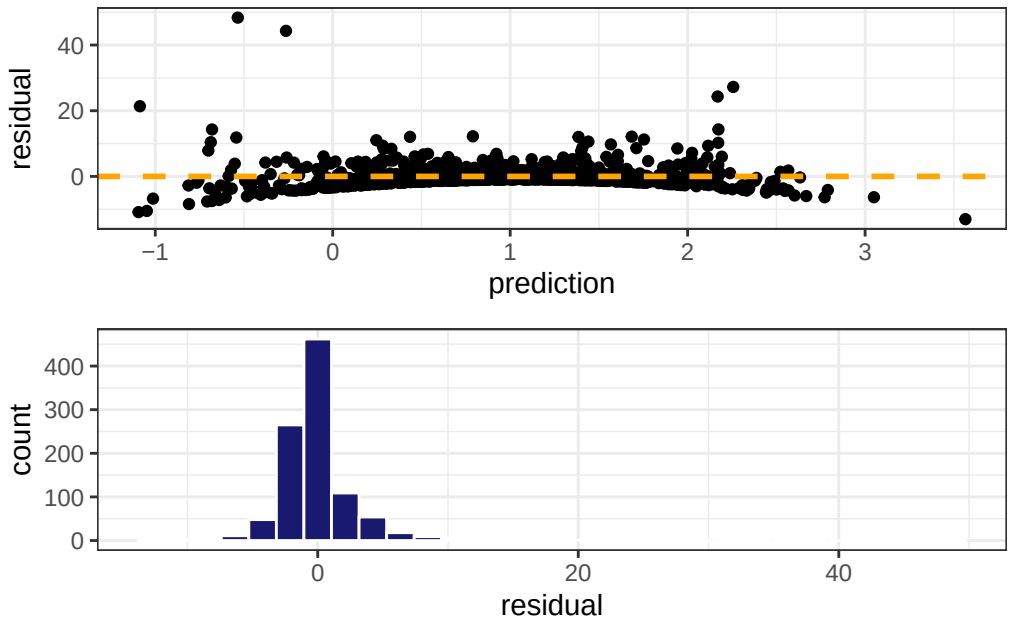
6

```
library(tidyverse)
library(patchwork)
library(moderndive)

dgp <- function(n) {
  x <- rnorm(n)
  epsilon <- (x^2 + 1) * (rchisq(n, df=1)-1)
  y <- 1 + x + epsilon

  df <- tibble(
    y = y,
    x = x
  )
}
```

a

```
set.seed(1)
df <- dgp(1000)
mod <- lm(y ~ x, df)

df_preds <- df %>%
  mutate(
    prediction = predict(mod, newx = x),
    residual = y - prediction
    )
```

```
## linearity
p1 <- df_preds %>%
  ggplot(aes(x, y)) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  theme_bw()

p2 <- df_preds %>%
  ggplot(aes(x = prediction, y = residual)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "orange", linetype = 2, linewidth = 1) +
  theme_bw()

p3 <- df_preds %>%
  ggplot(aes(x = residual)) +
  geom_histogram(fill = "midnightblue", color = "white") +
  theme_bw()


p2 / p3
```



b

Linearity holds as shown below

$$\begin{aligned} E[\varepsilon|X] &= E[(X^2+1)\cdot(\chi_1^2-1)\mid X] \\ &= E[(X^2\cdot\chi_1^2+\chi_1^2-X^2-1\mid X] \\ &= E[X^2\cdot\chi_1^2\mid X]+E[\chi_1^2\mid X]-E[X^2\mid X]-E[1\mid X] \\ &= X^2E[\chi_1^2\mid X]+E[\chi_1^2\mid X]-X^2E[1\mid X]-E[1\mid X] \\ &= X^2\cdot 1+1-X^2\cdot 1-1 \\ &= 0 \end{aligned}$$

Equal Variance does not hold as $\text{var}(\varepsilon \mid X=x)$ varies with x as is shown below

$$
\begin{aligned}
\text{var}(\varepsilon \mid X = x) &= E[\varepsilon^2 \mid X = x] \\
&= E\Big[(X^2+1)^2(\chi_1^2 - 1)^2 \mid X = x\Big] \\
&= (x^2+1)^2 E\Big[(\chi_1^2)^2 - 2\chi_1^2 + 1 \mid X = x\Big] \\
&= (x^2+1)^2 \left(E\Big[(\chi_1^2)^2 \mid X = x\Big] - 2E[\chi_1^2 \mid X = x] + 1\right) \\
&= (x^2+1)^2 \left(E\Big[(\chi_1^2)^2 \mid X = x\Big] - 1\right) \\
&= (x^2+1)^2 \left(\text{var}(\chi_1^2 \mid X = x) + \Big(E[\chi_1^2 \mid X = x]\Big)^2 - 1\right) \\
&= (x^2+1)^2 (2 + 1^2 - 1) \\
&= 2 \cdot (x^2+1)^2
\end{aligned}
$$

c

i

```
get_regression_table(mod) %>%
  filter(term == "x") %>%
  select(term, lower_ci, upper_ci)
```

```
# A tibble: 1 x 3
  term  lower_ci upper_ci
  <chr>    <dbl>    <dbl>
1 x        0.462    0.904
```

ii

```
set.seed(100)
df_list <- list()
for(i in 1:1000){
  data <- df %>%
    slice_sample(n = 1000, replace = T)

  df_list[[i]] <- data
}
```

```r
get_coef <- function(data) {
  mod <- lm(y ~ x, data)
  return(coef(summary(mod))[2,1])
}

boot_coefs <- df_list %>%
  map_dbl(.f = get_coef)

quantile(boot_coefs, c(0.025, 0.975))
```

```
     2.5%     97.5%
0.2405628 1.0597162
```

iii

```r
coef(summary(mod))[2, 1] + c(-1, 1)*qnorm(0.975)*sd(boot_coefs)
```

```
[1] 0.2872003 1.0795604
```

iv

```r
library(sandwich)

varBeta <- vcovHC(mod, type="HC")
se <- sqrt(diag(varBeta))[2]

coef(summary(mod))[2, 1] + c(-1, 1)*qnorm(0.975)*se
```

```
[1] 0.2900729 1.0766878
```

d

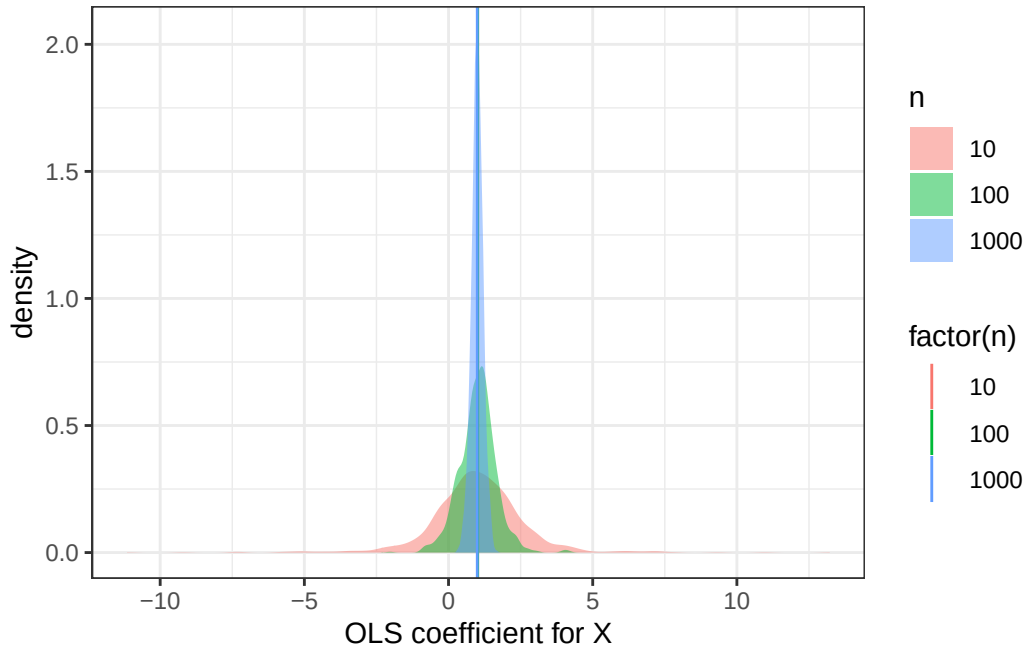
```r
set.seed(4)

df_list <- list()

for(i in 1:1000) {
```

```
  df_list[[i]] <- list(
    df_10 = dgp(10),
    df_100 = dgp(100),
    df_1000 = dgp(1000)
  )
}

getter <- function(df){
  model <- lm(y ~ x, df)
  beta <- coef(summary(model))[2]
  trad_ci <- get_regression_table(model) %>%
    filter(term == "x") %>%
    select(lower_ci, upper_ci) %>%
    mutate(method = "trad")

  varBeta <- vcovHC(model, type="HC")
  se <- sqrt(diag(varBeta))[2]
  endpts <- coef(summary(model))[2, 1] + c(-1, 1)*qnorm(0.975)*se
  robust_ci <- tibble(
    lower_ci = endpts[1],
    upper_ci = endpts[2],
    method = "robust"
  )

  all_ci <- rbind(trad_ci, robust_ci)
  return(list(beta, all_ci))
}

res_list <- df_list %>%
  map(.f = ~map(.x, .f = getter))

coef_df <- data.frame()
ci_df <- data.frame()

for (i in 1:1000){

  coef_10 <- res_list[[i]]$df_10[[1]]
  coef_100 <- res_list[[i]]$df_100[[1]]
  coef_1000 <- res_list[[i]]$df_1000[[1]]

  ci_10 <- res_list[[i]]$df_10[[2]] %>%
```

```
    mutate(n = 10, iter = i)


  ci_100 <- res_list[[i]]$df_100[[2]] %>%
    mutate(n = 100, iter = i)

  ci_1000 <- res_list[[i]]$df_1000[[2]] %>%
    mutate(n = 1000, iter = i)

  res_ci <- rbind(ci_10, ci_100, ci_1000)

  res_coef <- tibble(
    n = c(10, 100, 1000),
    beta = c(coef_10, coef_100, coef_1000),
    iter = rep(i, 3)
  )

  ci_df <- rbind(ci_df, res_ci)
  coef_df <- rbind(coef_df, res_coef)

}
```

looks unbiased and the distributions have lower variance as n increases

```
library(here)
coef_df <- read_csv(here("data", "hw1_coef_df.csv"))
ci_df <- read_csv(here("data", "hw1_ci_df.csv"))

means <- coef_df %>%
  group_by(n) %>%
  summarise(mean = mean(beta))


coef_df %>%
  ggplot(aes(x = beta, fill = factor(n))) +
  geom_density(color = NA, alpha = 0.5) +
  geom_vline(data = means, aes(xintercept = mean, color = factor(n))) +
  theme_bw() +
  labs(
    fill = "n",
    x = "OLS coefficient for X"
  )
```




```
# not sure what to do here
ci_df %>%
  left_join(coef_df, by = c("n", "iter")) %>%
  rowwise() %>%
  mutate(contains = between(beta, lower_ci, upper_ci)) %>%
  ungroup() %>%
  group_by(n, method) %>%
  summarise(coverage = mean(contains))
```

```
`summarise()` has grouped output by 'n'. You can override using the `.groups`
argument.
```

```
# A tibble: 6 x 3
# Groups:   n [3]
      n method coverage
  <dbl> <chr>     <dbl>
1    10 robust        1
2    10 trad          1
3   100 robust        1
4   100 trad          1
5  1000 robust        1
6  1000 trad          1
```