Midterm Exam

Please give complete solutions, showing your work and explaining clearly. Lack of details can result in point deduction.

All figures are given at the end of the document.

NAME: Enmin Zhou
DUE DATE: March 12, 10:30AM

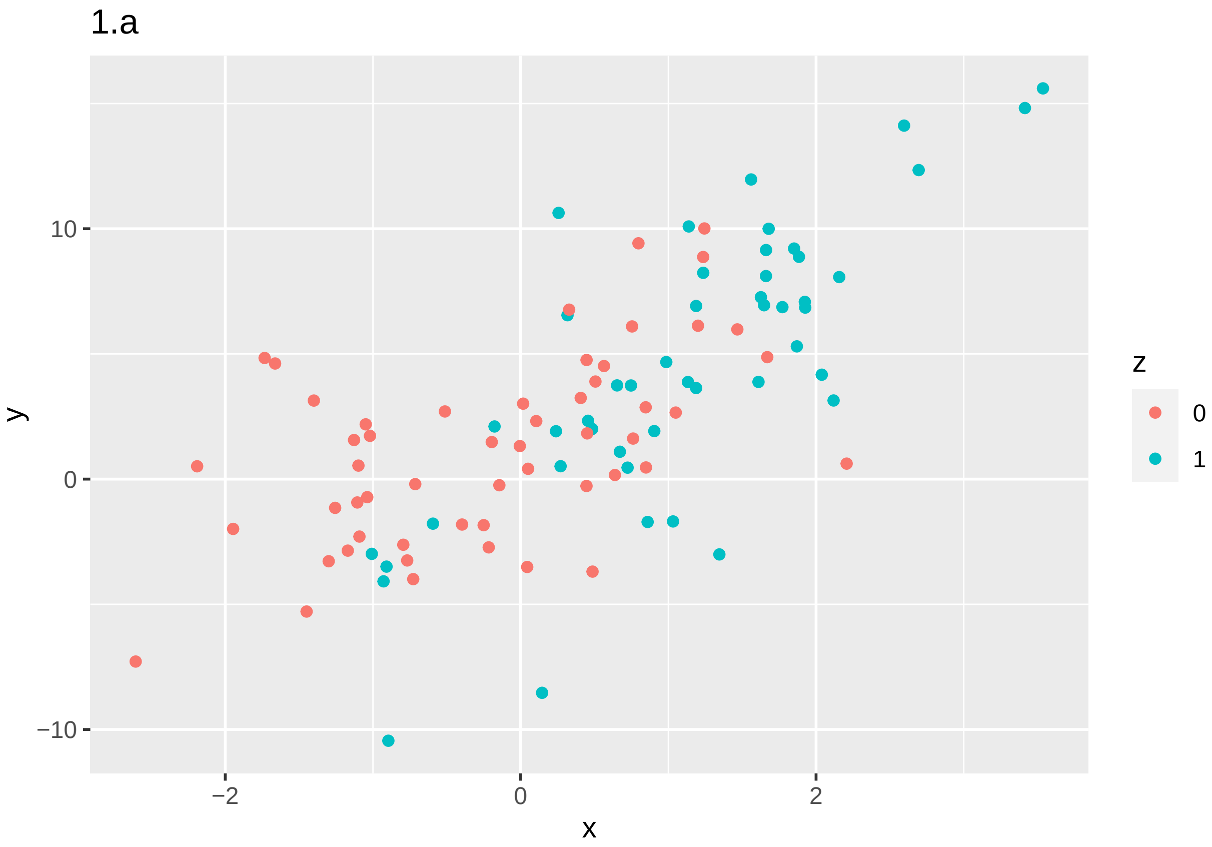
Question 1: Regression with interactions.

Simulate 100 data points from the model $y = b_0 + b_1 x + b_2 z + b_3 xz + error$, with a continuous predictor x and a binary predictor z, coefficients $b = c(1, 2, -1, 2)$, and errors drawn independently from a normal distribution with mean 0 and standard deviation 3, as follows. For each data point i, first draw z_i, equally likely to take on 0 and 1. Then draw x_i from a normal distribution with mean z_i and standard deviation 1. Then draw the error from its normal distribution and compute y_i.

a) (10 points) Display your simulated data as a graph of y vs. x, using two types of colors for z=0 and 1. Make sure to plot with ggplot in R. Be careful with the labels and the axes. Describe your results.

Answer 1a: There are two colors of points, red for z=0 and blue for z=1 in the plot. Y ranges aorund -15 to 15 and x ranges around -3 to 3. Red points x-center is around 0 and blue points x-center is around 1. The blues have y-center higher than that of the reds.

```r
rm(list=ls())
set.seed(2020)
errors <- rnorm(100, 0, 3)
zs <- sample(c(0,1), replace=TRUE, size=100)
xs <- c()
ys <- c()
for(i in 1:100){
  x <- rnorm(1, zs[i], 1)
  xs[i] <- x
  ys[i] <- 1 + 2*x - zs[i] + 2*x*zs[i] + errors[i]
}
df <- data.frame("y"=ys, "x"=xs, "z"=zs, "error"=errors)
p1_a <- ggplot(df, aes(x, y, color=as.factor(z))) + geom_point() + labs(colour="z", title="1.a")
p1_a
```



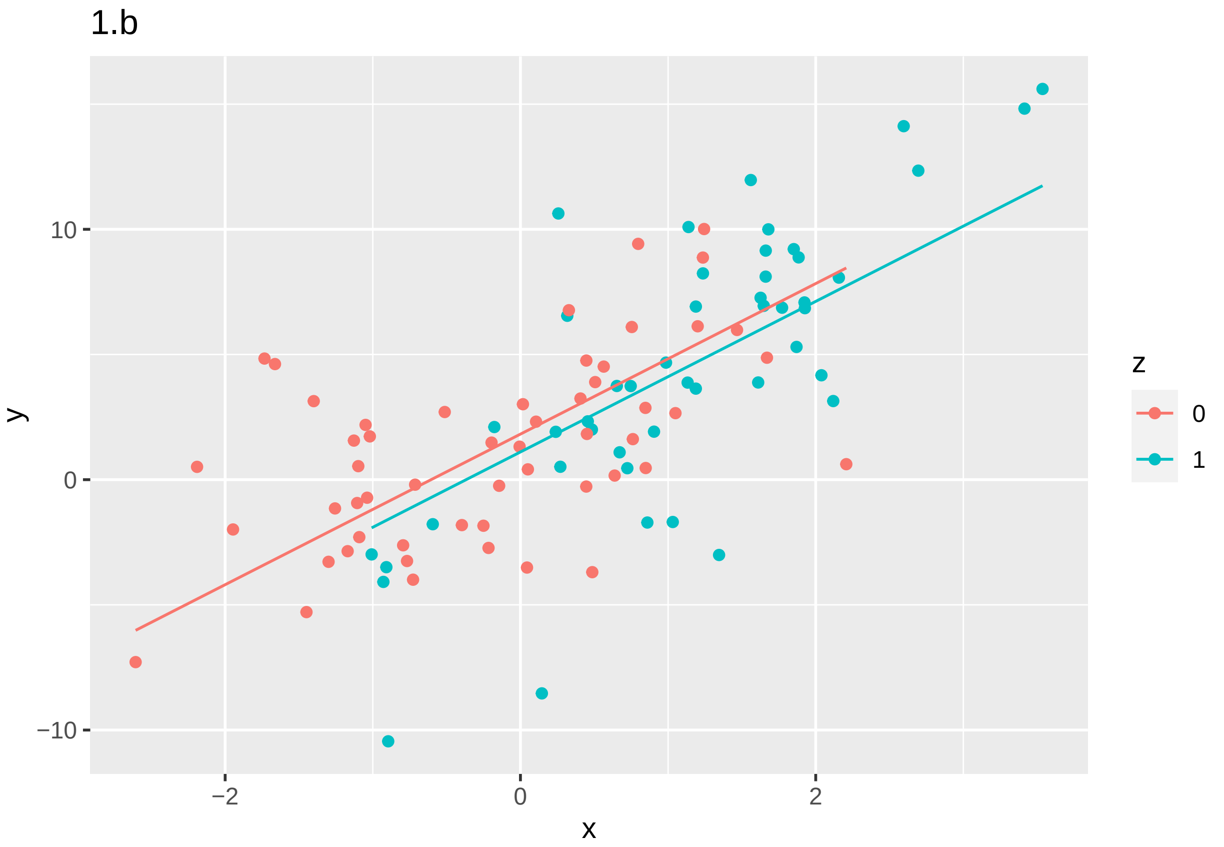
b) (10 points) Fit a regression predicting y from x and z with no interaction. Make a graph (in ggplot) with the data and two parallel lines showing the fitted model. Describe your results.

Answer 1b: The data are the same simulated data from part a. We have a red regression line for red points and a blue regression line for blue points. The blue line is a little bit higher and to the right of the red line because of the blue points distribution is also higher and to the right of the red points distribution. Two lines are parallel since they have the same slope as the value of the binary predictor z does not have any impact on the coefficients on x according to the formula without interaction. The variance of the normal distributions that generate blue points and red points are the same. The difference in the mean causes the difference of the intercept of the regression line.

```
fitb <- lm(y ~ x + z, data=df)
summary(fitb)
```

```
##
## Call:
## lm(formula = y ~ x + z, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.0789  -2.0384   0.1531   2.1383   8.7534
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.8199     0.4989   3.648 0.000428 ***
## x             3.0070     0.3422   8.786 5.58e-14 ***
## z            -0.7132     0.8505  -0.839 0.403773
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.599 on 97 degrees of freedom
## Multiple R-squared:  0.5013, Adjusted R-squared:  0.491
## F-statistic: 48.74 on 2 and 97 DF,  p-value: 2.225e-15
```

```
df_b = cbind(df, pred = predict(fitb))
p1_b <- ggplot(df_b, aes(x, y, color=as.factor(z))) + geom_point() + labs(colour="z", title="1.b") + ge
p1_b
```



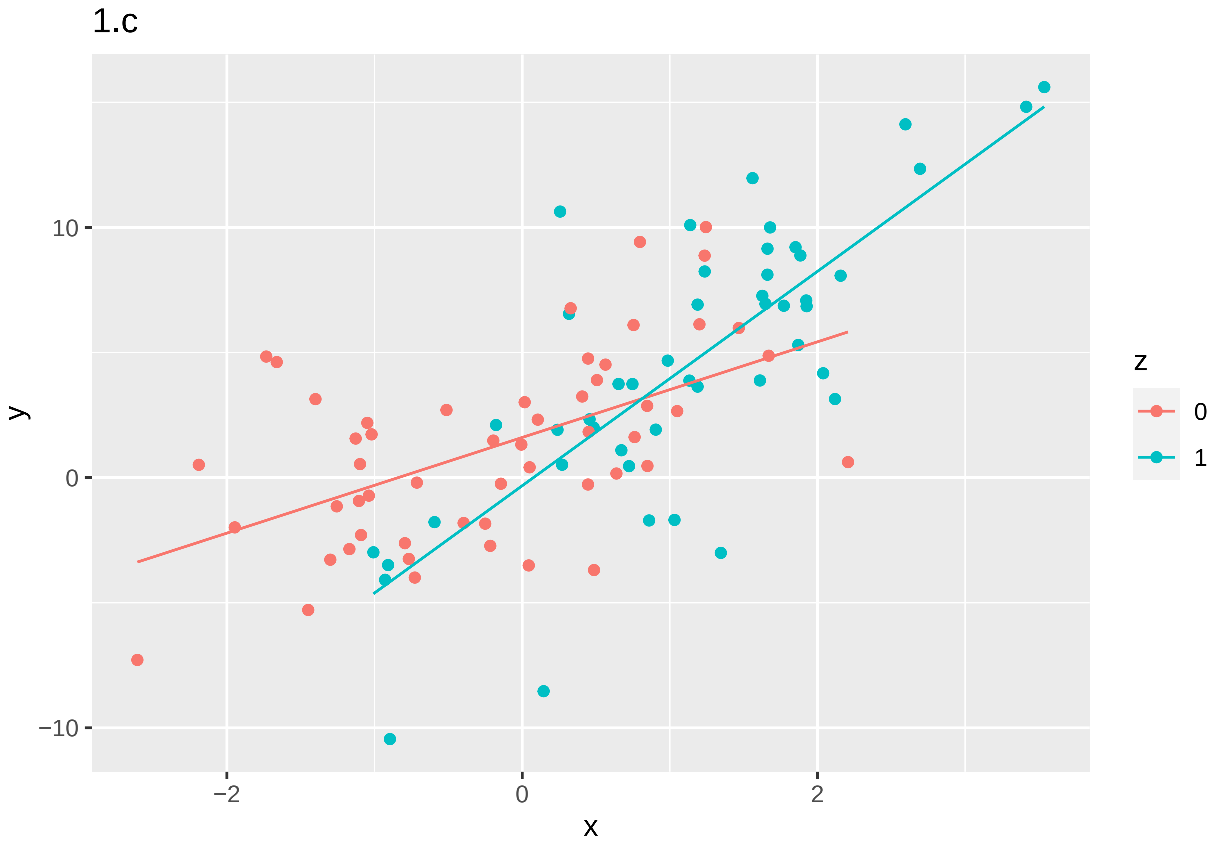
c) (10 points) Fit a regression predicting y from x and z and their interaction. Make a graph (in ggplot) with the data and two lines showing the fitted model. Describe your results.

Answer 1c: We also have two regression lines, but they intersected in the plot in 1.c. Since we predict y from the interaction of x and z, which results in dependent relationship between x and z. Since the binary category predictor z has an impact on the regression together with x, so the fitted lines of two classes of z will have different slops which results in an intersection.

```
fitc <- lm(y ~ x * z, data=df)
df_c = cbind(df, pred=predict(fitc))
p1_c <- ggplot(df_c, aes(x, y, color=as.factor(z))) + geom_point() + labs(colour="z", title="1.c") + ge
p1_c
```

Table 1: Regression output

	Estimate	Std. Error
`Intercept`	1.2	0.2
`x`	1.6	0.4
`z`	2.7	0.3
`x:z`	0.7	0.5



Question 2: Regression with interactions.

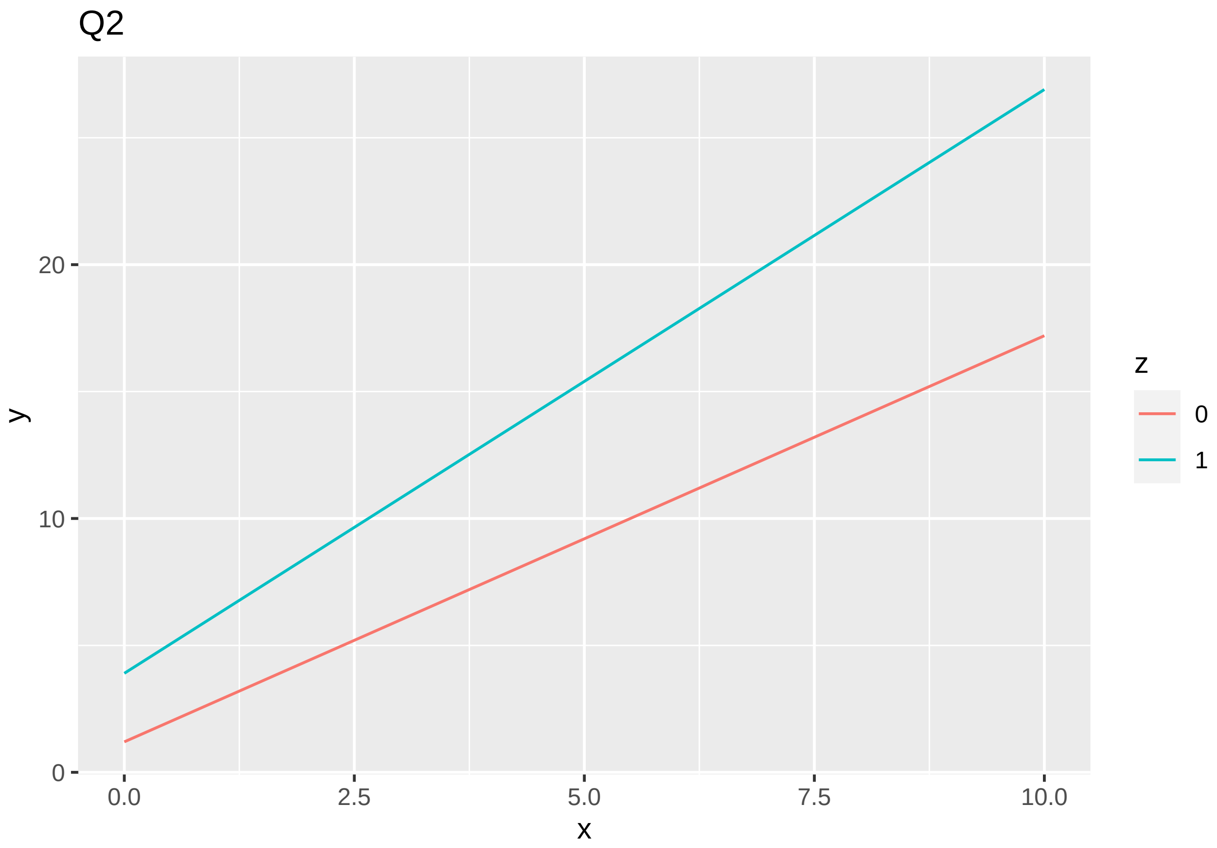
(10 points) Table 1 presents the output from a fitted lienar regression of outcome y on pre-treatment predictor x, binary treatment indicator z, and their interaction:

Are the coefficients significant or not? Write the equation of the estimated regression line of y on x for the treatment group and the control group equation. Graph (in ggplot) the two regression lines assuming x fall in the range $(0, 10)$. Describe your results.

Answer 2: The coefficients for Intercept, x and z are significant, but not for xz (z-score for Intercept is 6, for x is 4, for z is 9 and for xz is 1.4). For treatment group$(z = 1)$, $y = 1.2 + 1.6x + 2.7z + 0.7xz = 3.9 + 2.3x$; for control group$(z = 0)$, $y = 1.2 + 1.6x$. The resulting graph consists of two lines indicated by the coefficients in the Table 1. Two lines have different slopes as x and z interacts in this regression model and changes the slope of y on x.

```
sample_x <- sample(seq(0, 10, by=1), replace = TRUE, size=100)
sample_z <- sample(c(0,1), replace=TRUE, size=100)
```

```r
sample_y <- 1.2 + 1.6*sample_x + 2.7 * sample_z + 0.7*sample_x*sample_z
df2 <- data.frame("y"=sample_y, "x"=sample_x, "z"=sample_z)
p_2 <- ggplot(df2, aes(x, y, color=as.factor(z), group=z)) + geom_line() + labs(colour='z', title='Q2')
p_2
```



Question 3: Simulation study of statistical significance.

(10 points) In this exercise, you will simulate two statistically independent variables to see what happens when we run a regression to predict one from the other. Generate 1000 data points from a normal distribution with mean 0 and standard deviation 1 (call it var1). Then, generate another variable in the same way (call it var2). Run a regression of var2 on var1. Is the slope *statistically significant*? Then, run a simulation repeating this process 100 times. This can be done using a loop. From each simulation, save the z-score (the estimated coefficient of var1 divided by its standard error). If the absolute value of the z-score exceeds 2, the estimate is *statistically significant*. How many of these 100 z-scores exceed 2 in absolute value, thus achieving the conventional statistical significance level? Plot the histogram of the distribution of the z-score in ggplot. Make sure to draw the lines at -2 and 2 in red. Describe your results.

Answer 3: The estimate of slope is 0.01617 while std is 0.11 so the slope is not statistically significant (z-score $<$ 2). There are in total 7 trials with absolute value of z-score exceeding 2. In the histogram, we can see that 4 of the trials have z-score smaller than -2 and 3 of trials have z-score bigger than 2. Values are centered around 0 and roughly symmetric.

```r
var1 <- rnorm(1000, 0, 1)
var2 <- rnorm(1000, 0, 1)
model3 <- lm(var2~var1)
print(summary(model3))
```

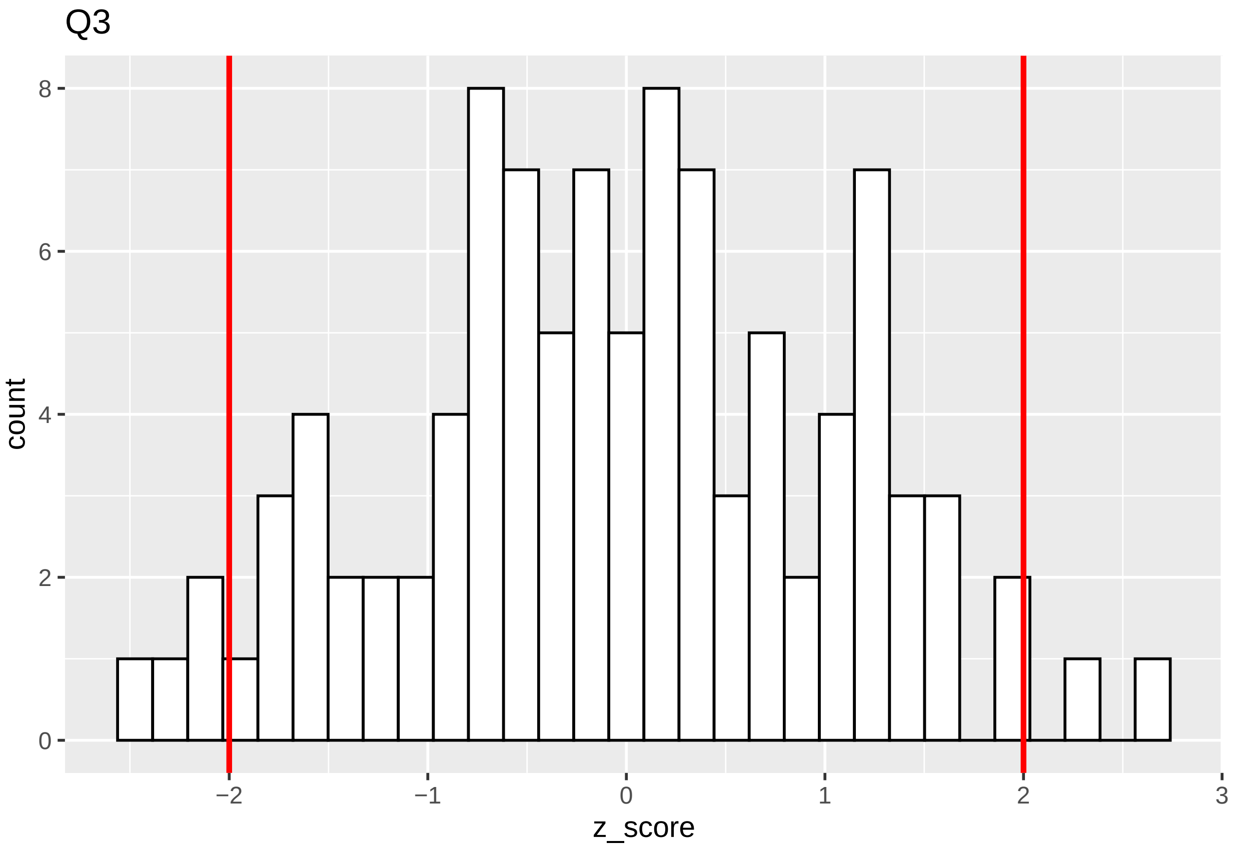
```
##
## Call:
## lm(formula = var2 ~ var1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7761 -0.6335 -0.0417  0.6521  3.7221
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.04004    0.03148  -1.272    0.204
## var1         0.03602    0.03147   1.144    0.253
##
## Residual standard error: 0.9954 on 998 degrees of freedom
## Multiple R-squared:  0.00131,    Adjusted R-squared:  0.0003098
## F-statistic:  1.31 on 1 and 998 DF,  p-value: 0.2527
```

```
z_scores <- c()
set.seed(2020)
for(i in 1:100){
  var1 <- rnorm(1000, 0, 1)
  var2 <- rnorm(1000, 0, 1)
  model3 <- lm(var2~var1)
  estimate <- coef(summary(model3))[2, 'Estimate']
  stderror <- coef(summary(model3))[2, 'Std. Error']
  z_scores[i] <- estimate / stderror
}
exceed = sum(abs(z_scores) > 2)
exceed
```

```
## [1] 7
```

```
df3 <- data.frame("z_score"=z_scores)
p_3 <- ggplot(df3, aes(x=z_score)) + geom_histogram(colour="black", fill="white") +
  geom_vline(aes(xintercept=2), color="red",size=1) + geom_vline(aes(xintercept=-2), color="red",size=1
p_3
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


Q3
count
8
6
4
2
0
−2
−1
0
1
2
3
z_score