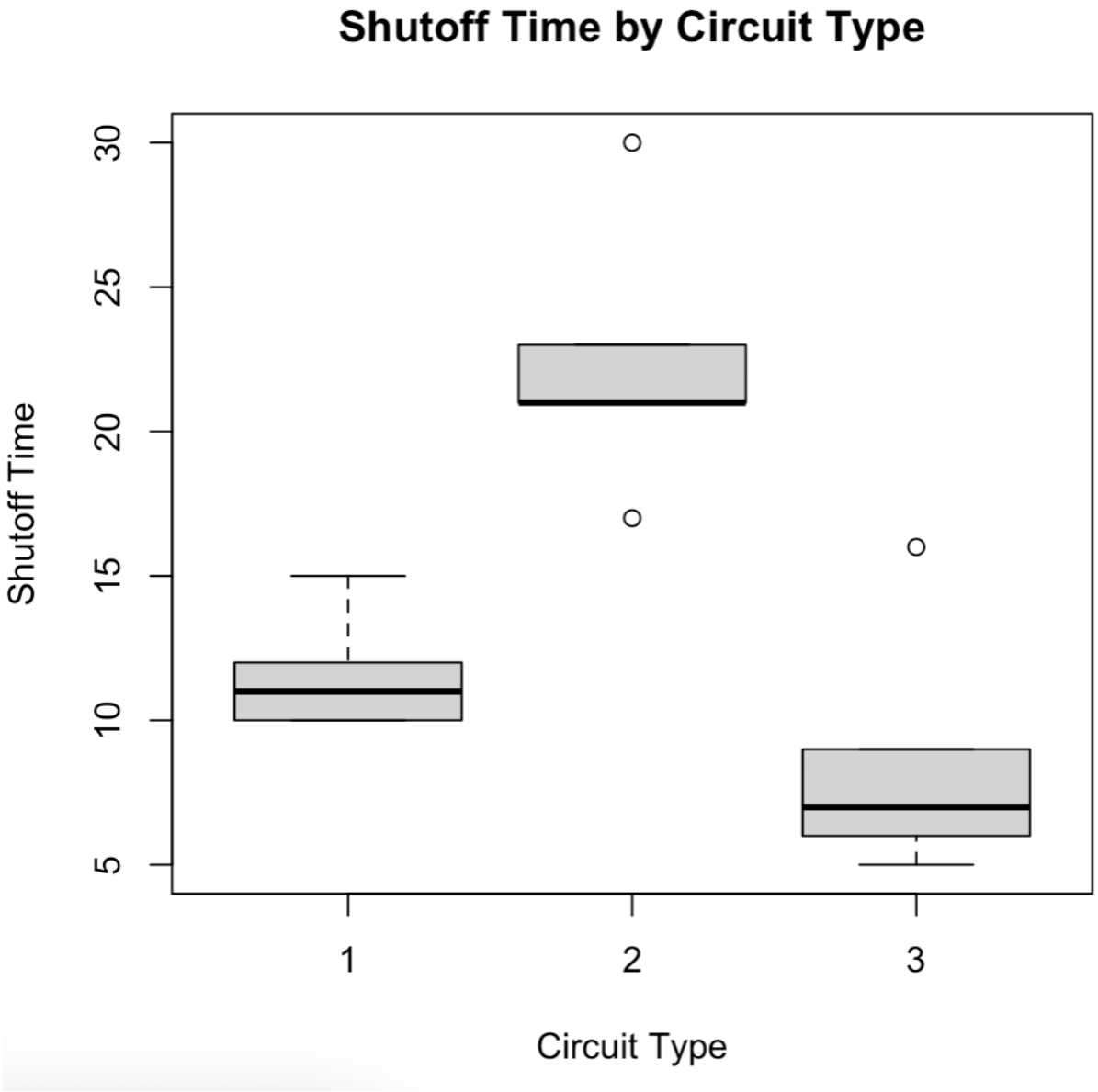
STA 361 **Name:** Ivan Wang

Final Exam (135 pts) **Due:** 5/13 by 6:50 PM

Instructions: Follow all homework guidelines listed in the course syllabus when answering the following questions. All hypotheses should be stated using parameters, when possible. All inferences should be carried out using $\alpha = 0.05$ unless otherwise stated. Upload a PDF version of your answers to the Assignments section of UBLearns, and use the naming convention described at the conclusion of this exam.

1. The data set shutoff25.txt contains the time elapsed before an emergency circuit activated a shutoff mechanism. Three different types of circuits were investigated. We would like to determine whether these three types of circuit have the same average time to shutoff. (25 pts)

a) Produce side-by-side boxplots of shutoff time by circuit type. (2)



b) Describe the class of the objects contained in the Type column. How should these values be treated when fitting an ANOVA model? (2)

```
[1] "integer"
> shutoff25$Type <- factor(shutoff25$Type)
> shutoff25$Type
 [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
Levels: 1 2 3
>
```

It's an integer of three possible values. It represents groups with 3 levels. When we fit a ANOVA model, R will treat factor variables as grouping variables.

c) Fit an appropriate model to address the study question, and present a summarized form of the output. (2)

```
> model <- aov(Time ~ Type, data = shutoff25)
> summary(model)
            Df Sum Sq Mean Sq F value   Pr(>F)
Type         2  526.8  263.40   17.03 0.000313 ***
Residuals   12  185.6   15.47
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
```

d) Perform the desired test, stating the hypotheses, test statistic, p-value, and conclusion in context. (4)

H_o: mu1 = mu2 = mu3 (The mean shutoff times are equal across all circuit types)

H_1: mu1 != mu2 != mu3 (At least one circuit type has a different mean shutoff time.)

Since F-value = 17.03, and P-value = 0.000313, we reject Ho. There is significant difference between mean shutoff times. At least one circuit type has a different mean shutoff time.

e) Reproduce the p-value from part (d) using the pf(.) function. (2)

```
> f_value <- 17.03
> df1 <- 2
> df2 <- 12
> p_value <- pf(f_value, df1, df2, lower.tail = FALSE)
> p_value
[1] 0.0003127115
> 17.03
```

f) What percentage of the variability in shutoff time is explained by circuit type? (1)

```
> ss_between <- anov_table["Type", "Sum Sq"]
> ss_total <- sum(anov_table[["Sum Sq"]])
> ss_between
[1] 526.8
> ss_total
[1] 712.4
>
> r_squared <- (ss_between / ss_total) * 100
> r_squared
[1] 73.94722
```

g) Identify which pairs of circuit types have mean shutoff times that differ significantly. Be sure to use a multiple testing correction when evaluating statistical significance. (4)

```
> TukeyHSD(model)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Time ~ Type, data = shutoff25)

$Type
     diff       lwr      upr     p adj
2-1  10.8   4.16422 17.43578 0.0025465
3-1  -3.0  -9.63578  3.63578 0.4720891
3-2 -13.8 -20.43578 -7.16422 0.0003432
```

2-1 is significant and 3-2 is significant because there P-value is less then 0.05.

Circuit Type 2 has significantly higher mean shutoff time than Type1. Circuit Type 3 is not significantly from Type 1. Circuit Type 3 has a significantly lower mean shutoff than Type2.

h) Test whether the model residuals are normally distributed. (2)

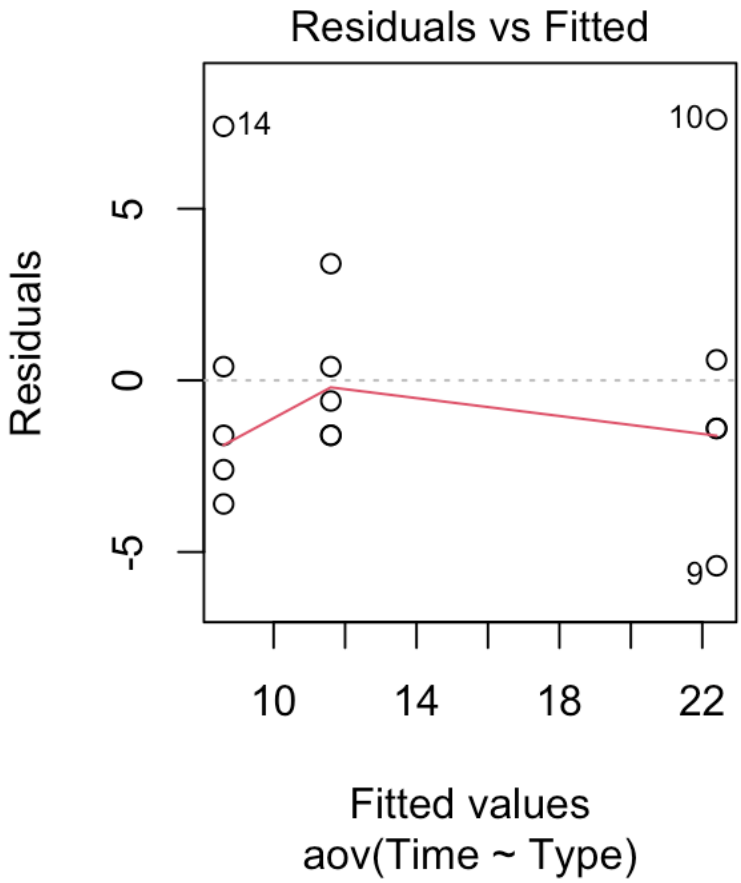
```
> shapiro.test(residuals(model))

        Shapiro-Wilk normality test

data:  residuals(model)
W = 0.87229, p-value = 0.03646
```

Since P-value = 0.036346 < 0.05, we reject Ho. There is no significant evidence that the residuals are normally distributed.

i) Produce a plot of residuals vs fitted values, and comment on the constant variance assumption. (2)



The residuals appear randomly scattered around 0 with no clear pattern or funnel shape, suggesting that the constant variance assumption is reasonable.

j) Based on our previous investigations of the robustness of ANOVA models, explain whether you are concerned about your findings in part (h). (2)

The sample size is small, and the residual vs fitted plot shows heteroscedasticity. As a result, these lead to weaken robustness of the ANOVA model and the results should be further tested.

2. Again we will refer to the data set shutoff25.txt. (21 pts)

a) Use the tapply(.) function to display the three group means. (2)

```
> tapply(shutoff25$Time, shutoff25$Type, mean)
   1    2    3
11.6 22.4  8.6
```

b) Open the "car" library, and use the function leveneTest(.) to examine whether this data set satisfies constant variance. Provide the hypotheses, test statistic, p-value, and conclusion. (4)

```
> library(car)
Loading required package: carData
> leveneTest(Time ~ Type, data = shutoff25)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  2  0.3931 0.6833
      12
>
```

Ho: There is constant variance for the group(homoscedasticity)

Ha: At least one group has different variance.

Since the F-value = 0.3931 and P-value = 0.6833 > 0.05, we fail to reject Ho. There is significant evidence that there is constant variance across all groups (there is homoscedasticity).

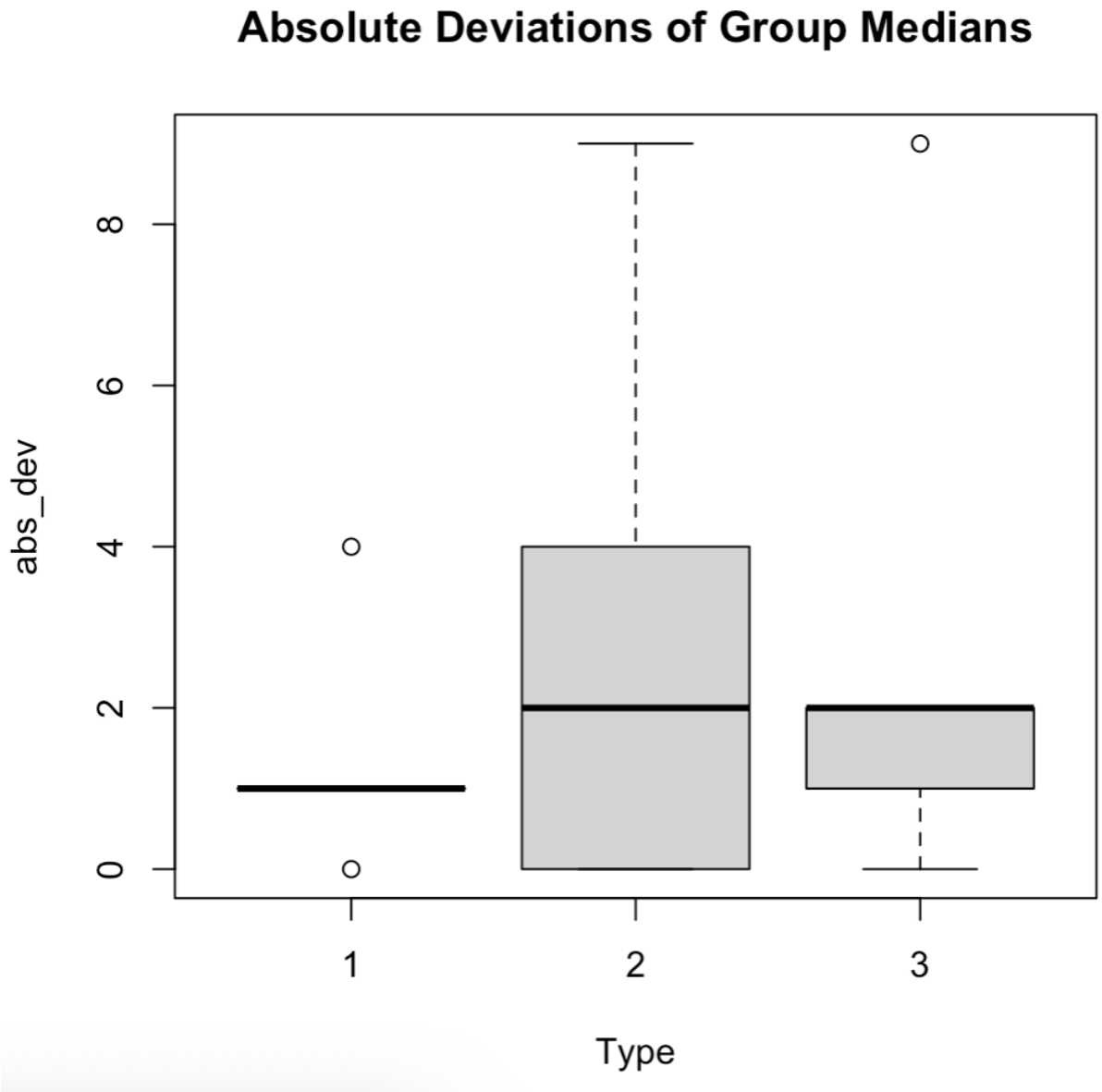
c) The next several steps will reproduce Levene's test. Use the tapply(.) function to store a vector of length three that contains the group **medians**. Print this vector. (2)

```
> medians <- tapply(shutoff25$Time, shutoff25$Type, median)
> medians
 1  2  3
11 21  7
```

d) Add a third column to the original data frame, computed as the absolute value of the response minus its group median, m_i:

$$d_{ij} = |Y_{ij} - m_i|$$

You now possess a set of distances (gaps between the response and its group median). Display these distances using three side-by-side boxplots. (3)



e) Levene's test for non-constant variance is basically the application of one-way ANOVA to a set of absolute distances from group-specific medians. Fit a one-way ANOVA model that uses your newly computed distances as the response variable, and the original grouping variable as the explanatory factor. Demonstrate that your results match the output from leveneTest(.) in part (b). (2)

```
> model <- aov(abs_dev ~ Type, data = shutoff25)
> summary(model)
            Df Sum Sq Mean Sq F value Pr(>F)
Type         2    7.6   3.800   0.393  0.683
Residuals   12  116.0   9.667
```

The F-value and P-value matches the results from (b)

f) Turn your work from parts (c)-(e) into a function (except, of course, the call to leveneTest(.)). The user should provide a vector of responses as the first argument and a vector of group labels as the second argument. The function should create absolute deviations from the group medians, and then execute the one-way ANOVA procedure, outputting (at least) the test statistic and p-value. Your function should run in a fresh R session without loading any external libraries. **Note**: it is not advisable for your function to hard-code group labels in reference to any particular data set. Your function should execute properly on any two vectors (one continuous, one categorical) of equal length. (8)

```
> leveneTest <- function(response, group) {
+   group <- as.factor(group)
+   group_medians <- tapply(response, group, median)
+   abs_dev <- abs(response - group_medians[as.character(group)])
+   model <- aov(abs_dev ~ group)
+   summary(model)[[1]][1, c("F value", "Pr(>F)")]
+ }
>
>
> leveneTest(shutoff25$Time, shutoff25$Type)
            F value Pr(>F)
group        0.3931 0.6833
```

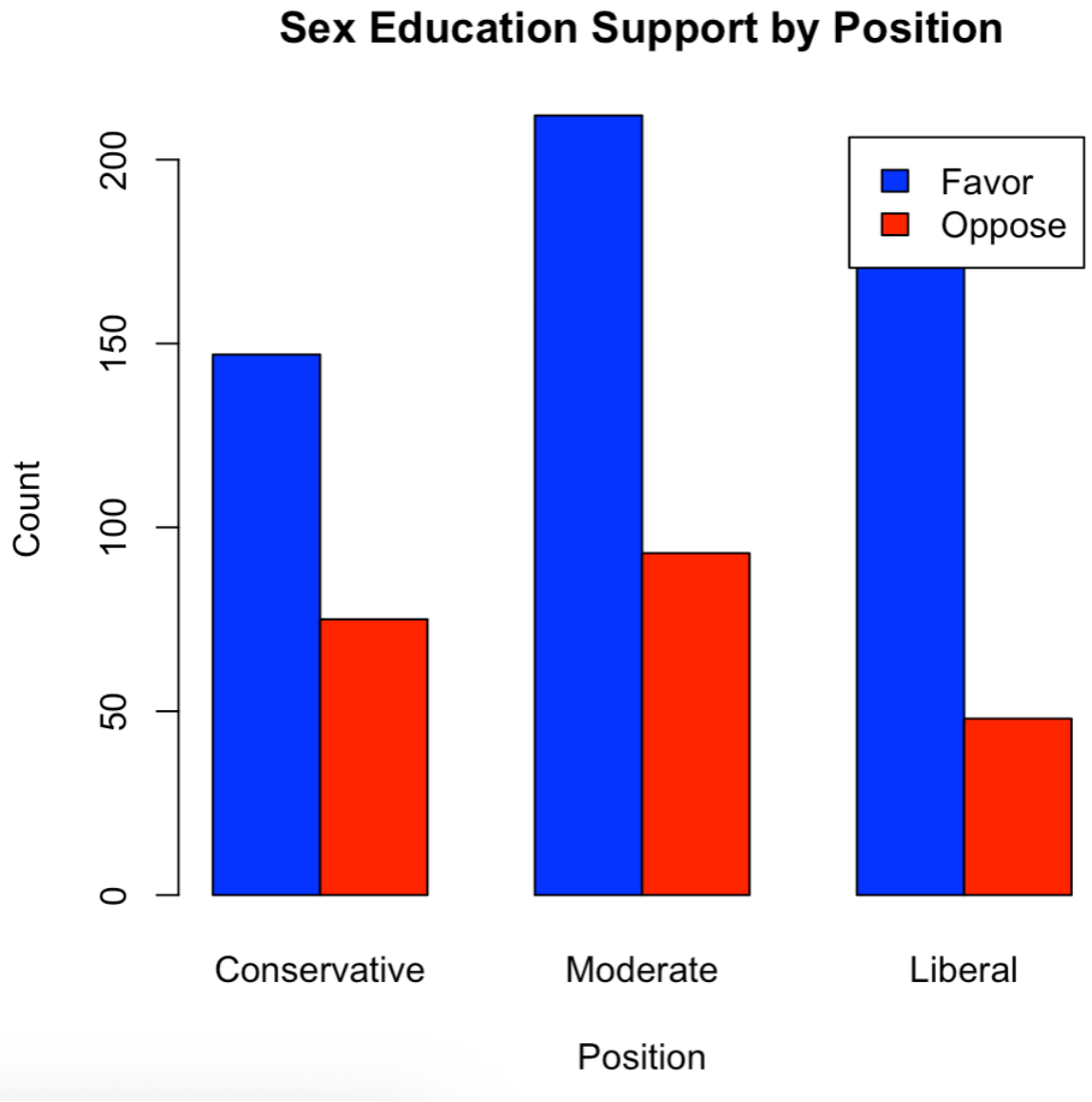
3. The following data are from a study carried out decades ago regarding attitudes toward sex education being instituted in public schools. (16 pts)

Disposition	Sex Education	
	Favor	Oppose
Conservative	147	75
Moderate	212	93
Liberal	204	48

a) Create (and print) a labeled matrix object to store the data. (2)

```
> sex_data <- matrix(c(147, 75, 212, 93, 204, 48),
+                        nrow = 3, byrow = TRUE,
+                        dimnames = list(
+                          position = c("Conservative", "Moderate", "Liberal"),
+                          Opinion = c("Favor", "Oppose")))
>
> sex_data
              Opinion
position       Favor Oppose
  Conservative   147     75
  Moderate       212     93
  Liberal        204     48
```

b) Produce a single barplot that displays all six cell counts. The plot should contain six bars, and category labels should be visible. (4)



c) Carry out the chi-square test for association. Provide the hypotheses, test statistic, p-value, and conclusion in context. (4)

```
> chi_result <- chisq.test(sex_data)
> chi_result

        Pearson's Chi-squared test

data:  sex_data
X-squared = 14.7, df = 2, p-value = 0.0006424
```

Ho: Disposition and Opinions on sex education is independent

Ho: There is an association between disposition and opinion on sex education.

Since X-Squared = 14.7, P-value = 0.0006424 < 0.05, we reject Ho. There is evidence that there is an association between disposition and opinion on sex education.

d) Reproduce the p-value from part (c) using the pchisq(.) function. (2)

```
> 1 - pchisq(chi_result$statistic, df = chi_result$parameter)
   X-squared
0.0006424421
```

e) Examine the standardized residuals from the chi-square test, and describe any way in which the observed data depart significantly from independence. (4)

```
> chi_result$stdres
              Opinion
position            Favor    Oppose
  Conservative -2.383717  2.383717
  Moderate     -1.382342  1.382342
  Liberal       3.742413 -3.742413
>
```

There is a large negative residual for Conservative-Favor and Liberal-Oppose, would suggest that conservatives favor it less than expected and that liberals oppose it less than expected.

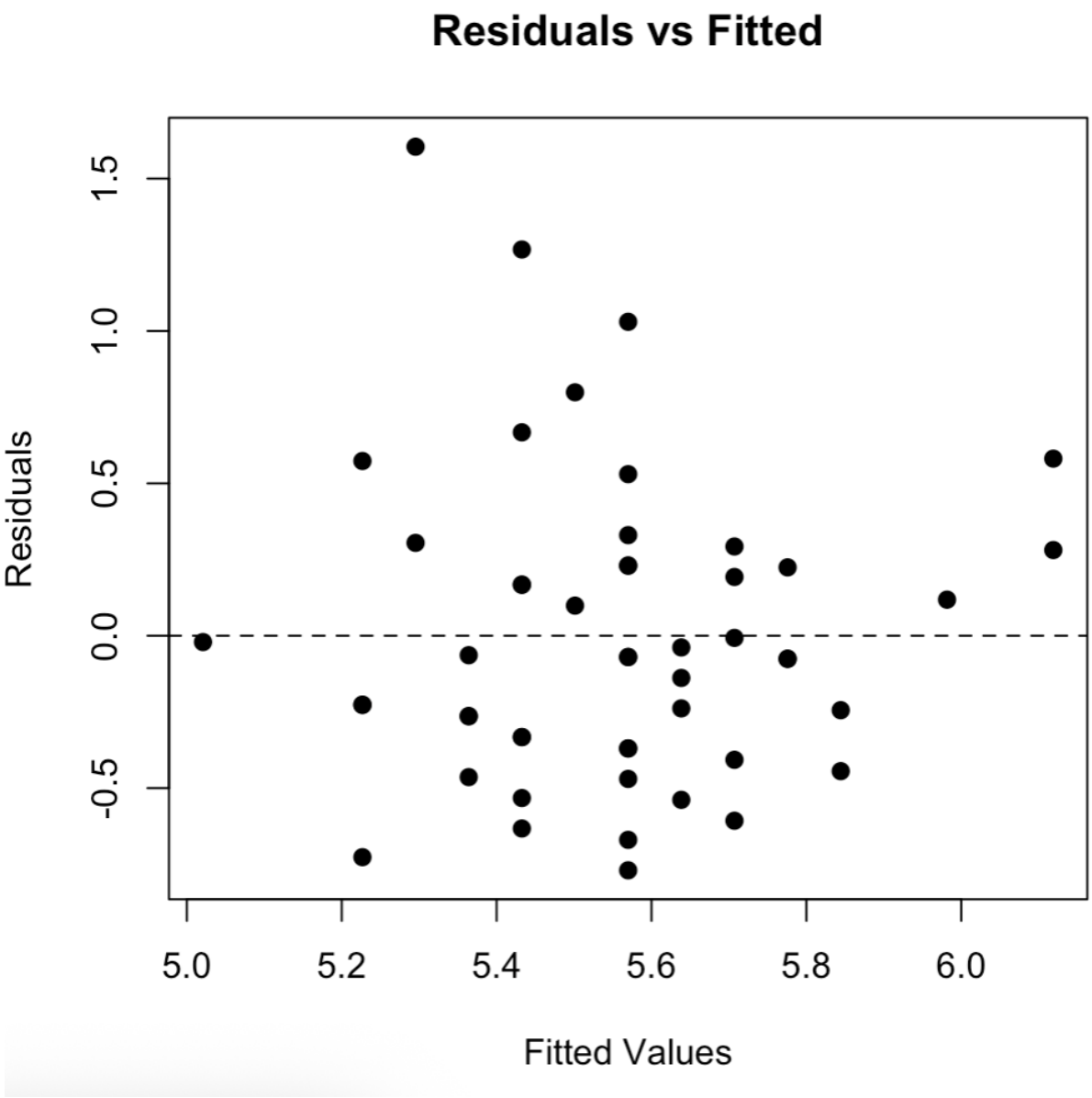
There is a large positive residual for Conservative-Oppose and Liberal-Favor, would suggest that conservative oppose it more than expected, and liberals favor it more than expected.

4. In our course, we briefly considered fitting linear regression models. Assumptions behind this model include that the model residuals are normally distributed with constant variance. We evaluated the constant variance assumption by visually examining a plot of residuals vs fitted values; we will now consider a formal hypothesis test (known as the Breusch-Pagan test) where H_0 specifies constant variance. (20 pts)

a) Obtain a subset of the iris data: only the 50 observations made on the "virginica" species. Verify that you have done this by showing that the sample mean petal length computed from your subset data frame is 5.552. (1)

```
> virginica <- subset(iris, Species == "virginica")
> mean(virginica$Petal.Length)
[1] 5.552
```

b) Fit a simple linear regression model using petal length as the response variable and sepal width as the single predictor. Present a residual plot. On the basis of this plot, explain whether constant variance seems plausible. (4)



There's no funnel shape or curved pattern which indicates Homoscedasticity(constant variance).

c) Denote the model residuals e_i for $i = 1, \ldots, n$. Append a new column to the data frame from part (a). This column should store the squared residuals, e_i^2. (2)

```
> virginica$resid_sq <- resid(model)^2
> head(virginica)
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species    resid_sq
101          6.3         3.3          6.0         2.5 virginica 0.050293092
102          5.8         2.7          5.1         1.9 virginica 0.069669396
103          7.1         3.0          5.9         2.1 virginica 0.109002854
104          6.3         2.9          5.6         1.8 virginica 0.009758937
105          6.5         3.0          5.8         2.2 virginica 0.052971693
106          7.6         3.0          6.6         2.1 virginica 1.061220977
```

d) Obtain the error sum of squares from the model fit in part (b) as $SSE = \sum_{i=1}^{n} e_i^2$. (1)

```
> SSE <- sum(virginica$resid_sq)
> SSE
[1] 12.52434
```

e) Fit the following regression model:

$$e_i^2 = \beta_0 + \beta_1 \text{Sepal. Width}_i + \varepsilon_i$$

and obtain the regression sum of squares. Call this value SSR^* and report its value. (2)

```
> bp_model <- lm(resid_sq ~ Sepal.Width, data = virginica)
> SSR_star <- sum((fitted(bp_model) - mean(virginica$resid_sq))^2)
> SSR_star
[1] 0.3657103
```

f) The B-P test allows for the possibility that the variance (on log scale) is linearly related to the predictor variable in the model (in this case: $\log(\sigma_i) = \gamma_0 + \gamma_1 \text{Sepal. Width}_i$). The hypotheses for the B-P test are as follows.

$$H_0: \gamma_1 = 0$$
$$H_a: \gamma_1 \neq 0$$

Explain which of these hypotheses is consistent with constant variance. (2)

Ho tells us that variance is constant (homoscedasticity)
Ha tells us that variance depends on Sepal.Width (heteroscedasticity)

Thus Ho is the assumption of constant variance.

g) The Breusch-Pagan test statistic is defined as:

$$X^2_{BP} = \frac{SSR^*/2}{(SSE/n)^2}$$

Report the value of the test statistic. (2)

```
> n <- nrow(virginica)
> X_BP <- (SSR_star / 2) / ((SSE / n)^2)
> X_BP
[1] 2.91432
```

h) When H_0 is true, X^2_{BP} is distributed as a chi-square random variable with degrees of freedom equal to the number of predictor variables included in the original regression model (i.e. the model fit in part (b)). Despite the two-sided alternative hypothesis, the p-value from the B-P test is always the probability in the right tail of the null distribution. Use one of R's probability functions to obtain the p-value corresponding to your test statistic. (2)

```
> p_value <- 1 - pchisq(X_BP, df = 1)
> p_value
[1] 0.0877964
```

i) State the conclusion of the B-P test using the context of the problem. (2)

Since the p-value is greater than 0.05, we reject Ho. There is significant evidence that the residual variance is constant(homoscedasticity is present)

j) The "skedastic" package contains a function called breusch_pagan(.). Show that this function reproduces your results from parts (g) and (h). Note: you will need to set the argument "koenker" to FALSE. (2)

```
> library(skedastic)
>
> breusch_pagan(model, koenker = FALSE)
# A tibble: 1 × 5
  statistic p.value parameter method                          alternative
      <dbl>   <dbl>     <dbl> <chr>                           <chr>
1      2.91  0.0878         1 Breusch-Pagan (non-studentised) greater
>
```

The p-value and test statistics matches from parts (g) and (h)

5. Suppose there is a normally distributed response variable (Y), a uniform random variable (U) generated between 0 and 25, an unseen random process (R) that is loosely correlated with Y, and a predictor variable (X) that investigators observe as a deviation around R. To summarize, the following five random variables are involved: (28 pts)

$$Y \sim N(\mu_Y = 200, \sigma_Y = 15)$$
$$U \sim Unif(0, 25)$$
$$\varepsilon \sim N(\mu_\varepsilon = 0, \sigma_\varepsilon = 10)$$

a) Set the randomization seed to 1985. Generate $n = 40$ observations of Y, U, and ε according to the distributions described above. Present the sample mean of each of the three vectors. (2)

```
> set.seed(1985)
> n <- 40
>
> Y <- rnorm(n, mean = 200, sd = 15)
> U <- runif(n, min = 0, max = 25)
> epsi <- rnorm(n, mean = 0, sd = 10)
>
> mean_y <- mean(Y)
> mean_u <- mean(U)
> mean_epsi <- mean(epsi)
>
> mean_y
[1] 198.7009
> mean_u
[1] 12.23428
> mean_epsi
[1] -1.667541
>
```

b) Create the variables R and X as defined below. Present the sample correlation between X and Y. (2)

$$R = 0.2Y + U$$
$$X = R + \varepsilon$$

```
> R <- 0.2 * Y + U
> X <- R + epsi
> cor_XY <- cor(X, Y)
> cor_XY
[1] 0.5187618
>
```

c) Fit the simple linear regression model using Y and X (Y as the response), and test whether there is a linear relationship present. Provide the hypotheses, test statistic, p-value, and conclusion. (4)

```
> model <- lm(Y ~ X)
> summary(model)

Call:
lm(formula = Y ~ X)

Residuals:
    Min      1Q  Median      3Q     Max
-33.505  -6.545   1.299   8.179  30.326

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 165.8339     9.0681  18.288  < 2e-16 ***
X             0.6533     0.1747   3.741 0.000605 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.18 on 38 degrees of freedom
Multiple R-squared:  0.2691,    Adjusted R-squared:  0.2499
F-statistic: 13.99 on 1 and 38 DF,  p-value: 0.0006048

>
```

H_0: $\beta_1 = 0$ (no linear relationship)

H_1: $\beta_1 \neq 0$ (linear relationship exists)

Since F-statistic = 13.99 and p-value = 0.0006048 < 0.05, we reject Ho. There is significant evidence that an linear relationship exists between X and Y.

d) We will now investigate the power of the test performed above. You will no longer set a randomization seed. Write a program to execute the following tasks $B = 10{,}000$ times:

- Generate $n = 40$ observations of Y, U, and ε.
- Create R and X.
- Regress Y on X.
- Note whether the test for a linear relationship rejects H_0.

Present your simulation-based estimate of the power of the test. Provide the full code required to perform your simulation. (4)

```
> B <- 10000
> n <- 40
> alpha <- 0.05
> rejects <- numeric(B)
>
> for (i in 1:B) {
+   Y <- rnorm(n, mean = 200, sd = 15)
+   U <- runif(n, 0, 25)
+   epsilon <- rnorm(n, 0, 10)
+   R <- 0.2 * Y + U
+   X <- R + epsilon
+   model <- lm(Y ~ X)
+   p_val <- summary(model)$coefficients["X", "Pr(>|t|)"]
+   rejects[i] <- (p_val < alpha)
+   rejects[i]
+ }
>
> power_estimate <- mean(rejects)
> power_estimate
[1] 0.3148
> rejects
   [1] 0 0 0 1 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 1 1 1
  [72] 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1
 [143] 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
 [214] 0 1 1 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 1 1 0 0 0 1 0 1 1 0 1 0 1 0 0 0 1 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0
 [285] 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 1 1 1 0 0 0 0 0 0 1 1 0 0 1 0 0 1 1 1 0 1 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 0 1 0 1 0 1 0 0 1 1 1 0 0 1 1 1 0
 [356] 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 0 1 1 1 1 0 1 0 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 1 1 1 0 1 1 1 1 0 0 0 0 1 0 0 0 0 1 1 0
 [427] 1 0 1 0 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 0 0 0 0 1 0 1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1 1 1 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 0 1 1 0
 [498] 0 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1 1 0 0 0 0 1 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 1 1
 [569] 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 1 1 0 1 0 1 0 0 1 0 0 0 0 0
 [640] 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0
 [711] 1 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 1 0 0 0 0 1 0 1 1 0 1 0
 [782] 1 0 1 0 0 0 0 1 0 0 0 1 0 0 0 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 1 1 1 1 1 1 1 0 0 0 1 1 0 1 1 1 0 0 0 1 1
 [853] 0 0 1 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0 0 0 1 1 1 0 1 0 1 0 0 0 0 1 0 1 1 0 0 0 1 0 1 0 0 0
 [924] 0 0 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 1 0 0 1 0
 [995] 0 0 1 1 0 0
 [ reached getOption("max.print") -- omitted 9000 entries ]
```

e) The original setup uses 15 as the population standard deviation of the response variable. Investigate what happens to the power of the test as this value changes. Fill in each entry in the table below, where each value is an estimate based on a simulation of size $B = 10{,}000$. You do not need to provide your code here. (3)

σ_Y	power
10	0.1684
20	0.5057

30	0.8239

f) Return to using $\sigma_Y = 15$. Investigate the impact of the value σ_ε on the power of the test. Again, no code need be presented here, and each simulation should use $B = 10{,}000$. (3)

σ_ε	power
5	0.5417
15	0.192
20	0.1374

g) Return to using $\sigma_Y = 15$ and $\sigma_\varepsilon = 10$. The original definition of the unseen process was:

$$R = 0.2Y + U$$

Investigate the impact that the coefficient on Y (originally 0.2) has on the power of the test. Complete the table below based on simulations of size $B = 10{,}000$. No code needs to be presented. (3)

coefficient	power
0.1	0.113
0.3	0.5979
0.5	0.9469

h) Return to all original settings ($\sigma_Y = 15$, $\sigma_\varepsilon = 10$, coefficient$= 0.2$). Investigate the impact of the sample size n on the power of the test. Again use $B = 10{,}000$; no code required. (3)

n	power
20	0.1729
30	0.2414
50	0.3859

i) Write clear statements summarizing how each item investigated in parts (e)-(h) affects the power of the test. (4)

When we increase population standard deviation, the power increases. When σ_ε increases, the power decreases. When the coefficient on Y increases, the power also increases. When the sample size(n) increases, the power also increases.

6. In this problem, we will simulate aspects of a fun game called Zombie Dice®. While not required, one way to simulate rolling a standard 6-sided die is to generate a uniform random variable between 1 and 7 using the runif(.) function, and then round down by using the floor(.) function.

The following table gives a mapping between ordinary six-sided dice and the yellow variety of Zombie Dice:

Ordinary Dice	Zombie Dice (yellow)
{1, 2}	brain
{3, 4}	escape
{5, 6}	attacked

In this game, each player is a zombie. The goal of the game is to score points by eating brains. According to the chart above, if a 1 or 2 is rolled, you get a nutritious brain to eat. If a 3 or 4 is rolled, you don't get any brains (your intended victim escapes). However, if a 5 or 6 is rolled, the "victim" attacks you. A player rolls dice until they are attacked three times. (25 pts)

a) According to probability theory, what is the probability that a single roll of a yellow die results in a brain? (1)

There are 6 possible outcomes and 2 of them result on brain, so 2/6

b) Write a block of code that randomly chooses an integer from the set {1,2,3,4,5,6} and outputs the text "brain," "escaped," or "attacked" according to the mapping in the problem's description. This will represent one simulated roll of a yellow die. Display output from a single execution of your code block. (3)

```
> roll <- floor(runif(1, min=1, max=7))
> result <- ifelse(roll <= 2, "brain", ifelse(roll <= 4, "escaped", "attacked"))
> result
[1] "escaped"
```

c) Simulate rolling a yellow zombie die 100 times and display the results using the table(.) function. Use these results to estimate the probability for a single yellow die to result in a brain. Also present the numeric difference between your simulation-based result and the theoretical probability from part (a). (3)

```
> results <- floor(runif(100, 1, 7))
> outcomes <- ifelse(results <= 2, "brain", ifelse(results <= 4, "escaped", "attacked"))
> table(outcomes)
outcomes
attacked    brain  escaped
      29       34       37
> mean(outcomes == "brain")
[1] 0.34
> abs(mean(outcomes == "brain") - 2/6)
[1] 0.006666667
```

d) Now simulate rolling a yellow zombie die 100,000 times and tabulate the results. Is the resulting simulation-based estimate of *P*(brain) closer to the true probability than the estimate from part (c)? Should it be? (3)

```
> results <- floor(runif(100000, 1, 7))
> outcomes <- ifelse(results <= 2, "brain", ifelse(results <= 4, "escaped", "attacked"))
> table(outcomes)
outcomes
attacked    brain  escaped
   33656    33312    33032
> mean(outcomes == "brain")
[1] 0.33312
> abs(mean(outcomes == "brain") - 2/6)
[1] 0.0002133333
```

Yes, the estimate should be closer to the true probability than estimated from (c)

e) Suppose that a player is about to roll three yellow zombie dice at the same time. What is the theoretical probability that none of the three victims escape (i.e. they are either eaten, or they fight back)? (1)

P(eaten or fought back) = (4/6 * 4/6 * 4/6) = 0.29629629629

f) Write a simulation in which a player rolls three yellow zombie dice at the same time. Create variables called *roll1*, *roll2*, and *roll3* to store these values. Display a single execution of this program. (2)

```
> roll1 <- floor(runif(1, 1, 7))
> roll2 <- floor(runif(1, 1, 7))
> roll3 <- floor(runif(1, 1, 7))
>
> c(roll1, roll2, roll3)
[1] 2 2 4
```

g) Simulate a player rolling the three dice 100,000 times. Store the results, and use them to estimate the probability that none of the three victims escape. Note that your program should be doing no multiplication. You should use logical operators to observe in simulation how often 0 of 3 victims escape. (3)

```
> trials <- 100000
> none_escaped <- logical(trials)
>
> for (i in 1:trials) {
+   rolls <- floor(runif(3, 1, 7))
+   escapes <- rolls %in% c(3, 4)
+   none_escaped[i] <- sum(escapes) == 0
+ }
> mean(none_escaped)
[1] 0.29643
>
```

h) A player is about to roll three yellow zombie dice. What is the theoretical probability that the turn will end immediately (i.e. the player is attacked by all three victims)? (1)

```
> pattacked <- (2/6)^3
> pattacked
[1] 0.03703704
```

i) Use 100,000 simulated rolls of three yellow zombie dice to estimate the probability that the player's turn ends immediately. Again, your program needs to use logical operators, not multiplication. (3)

```
> all_attacks <- logical(trials)
>
> for (i in 1:trials) {
+   rolls <- floor(runif(3, 1, 7))
+   attacks <- rolls %in% c(5, 6)
+   all_attacks[i] <- sum(attacks) == 3
+ }
>
> mean(all_attacks)
[1] 0.03732
```

j) There are actually zombie dice of three different colors. Green dice represent the most vulnerable victims, while red dice represent victims who are more likely to defend themselves. The following tables describe these other types of dice:

Ordinary Dice	**Zombie Dice (green)**
{1, 2, 3}	brain
{4, 5}	escape
{6}	attacked

Ordinary Dice	**Zombie Dice (red)**
{1}	brain
{2, 3}	escape
{4, 5, 6}	attacked

The complete Zombie Dice set contains six green dice, four yellow dice, and three red dice. As a fun experiment, consider a player rolling all 13 of these dice. We are interested in the extremely unlikely event that all 13 dice result in attacks. Write a new simulation in which you create variables *roll1*, *roll2*, …, *roll13*. Define the random variable $X = \{\#\text{ of attacks when rolling all 13 dice}\}$. Carry out the experiment a million times. In your million simulated trials, does a player ever get attacked by all 13 victims? No multiplication should take place in this problem; only simulation and summarization. (5)

```
> green <- function(n) {
+   vals <- floor(runif(n, 1, 7))
+   ifelse(vals <= 3, "brain", ifelse(vals <= 5, "escaped", "attacked"))
+ }
>
> yellow <- function(n) {
+   vals <- floor(runif(n, 1, 7))
+   ifelse(vals <= 2, "brain", ifelse(vals <= 4, "escaped", "attacked"))
+ }
>
> red <- function(n) {
+   vals <- floor(runif(n, 1, 7))
+   ifelse(vals == 1, "brain", ifelse(vals <= 3, "escaped", "attacked"))
+ }
>
>
> trials <- 1000000
> total_attacks <- integer(trials)
>
> for (i in 1:trials) {
+   res <- c(green(6), yellow(4), red(3))
+   total_attacks[i] <- sum(res == "attacked")
+ }
>
> mean(total_attacks == 13)
[1] 0
```

Naming convention: Upload your PDF file to UBLearns, and use the following template:

Final_<your UBIT name>.pdf

Code:

```
"""
1. The data set shutoff25.txt contains the time elapsed before an emergency circuit
activated a shutoff mechanism. Three different types of circuits were investigated.
We would like to determine whether these three types of circuit have the same average
time to shutoff. (25 pts)
"""
shutoff25 <- read.delim("~/Downloads/shutoff25.txt")
View(shutoff25)

#a)		Produce side-by-side boxplots of shutoff time by circuit type. (2)

boxplot(Time ~ Type,
	data = shutoff25,
	main = "Shutoff Time by Circuit Type",
	xlab = "Circuit Type",
	ylab = "Shutoff Time")

#b)		Describe the class of the objects contained in the Type column. How should these values be treated when fitting an ANOVA model? (2)
class(shutoff25$Type)
shutoff25$Type <- factor(shutoff25$Type)

#It represents groups with 3 levels.
#When we fit a ANOVA model, R will treat factor variables as grouping variables.



"""
c)		Fit an appropriate model to address the study question, and present a summarized form of the output. (2)
"""
model <- aov(Time ~ Type, data = shutoff25)
summary(model)


"""
```

d) Perform the desired test, stating the hypotheses, test statistic, p-value, and conclusion in context. (4)

H_0: mu1 = mu2 = mu3 (The mean shutoff times are equal across all circuit types)

H_1: mu1 != mu2 != mu3 (At least one circuit type has a different mean shutoff time.)

Since F-value = 17.03, and P-value = 0.000313, we reject Ho. There is significant difference between mean shutoff times. At least one circuit type has a different mean shutoff time.

"""

"""

e) Reproduce the p-value from part (d) using the pf(.) function. (2)

"""

```
f_value <- 17.03
df1 <- 2
df2 <- 12
p_value <- pf(f_value, df1, df2, lower.tail = FALSE)
p_value
```

"""

f) What percentage of the variability in shutoff time is explained by circuit type? (1)

"""

```
anov_table <- summary(model)[[1]]
anov_table
ss_between <- anov_table["Type", "Sum Sq"]
ss_total <- sum(anov_table[["Sum Sq"]])
ss_between
ss_total

r_squared <- (ss_between / ss_total) * 100
r_squared
```

"""

g) Identify which pairs of circuit types have mean shutoff times that differ

significantly. Be sure to use a multiple testing correction when evaluating

statistical significance. (4)

"""

TukeyHSD(model)

"""

2-1 is significant and 3-2 is significant because there P-value is less then 0.05.

Circuit Type 2 has significantly higher mean shutoff time than Type1. Circuit Type 3 is not significantly from Type 1. Circuit Type 3 has a significantly lower mean shutoff than Type2.

"""

"""

h) Test whether the model residuals are normally distributed. (2)

"""

shapiro.test(residuals(model))

#Since P-value = 0.036346 < 0.05, we reject Ho. There is no significant evidence that the residuals are normally distributed.

"""

i) Produce a plot of residuals vs fitted values, and comment on the constant variance assumption. (2)

"""

plot(model, which = 1)

#The residuals appear randomly scattered around 0 with no clear pattern or funnel shape, suggesting that the constant variance assumption is reasonable.

"""

j) Based on our previous investigations of the robustness of ANOVA models, explain whether you are concerned about your findings in part (h). (2)

"""

#The sample size is small, and the residual vs fitted plot shows heteroscedasticity. As a result, these lead to weaken robustness of the ANOVA model and the results should be further tested.

"""

2. Again we will refer to the data set shutoff25.txt. (21 pts)

"""

#a) Use the tapply(.) function to display the three group means. (2)

tapply(shutoff25$Time, shutoff25$Type, mean)

"""

b) Open the “car” library, and use the function leveneTest(.) to examine whether this data set satisfies constant variance.

Provide the hypotheses, test statistic, p-value, and conclusion. (4)

"""

install.packages("car")

library(car)

leveneTest(Time ~ Type, data = shutoff25)

"""

c) The next several steps will reproduce Levene’s test. Use the tapply(.) function to store a vector of length three that contains

the group medians. Print this vector. (2)

"""

medians <- tapply(shutoff25$Time, shutoff25$Type, median)

medians

"""

d) Add a third column to the original data frame, computed as the absolute value of the response minus its group median, m_i:

d_ij=|Y_ij-m_i |

You now possess a set of distances (gaps between the response and its group median).

Display these distances using three side-by-side boxplots. (3)

"""

```
shutoff25$abs_dev <- abs(shutoff25$Time - medians[as.character(shutoff25$Type)])
boxplot(abs_dev ~ Type, data = shutoff25, main = "Absolute Deviations of Group Medians")
```

"""

e) Levene's test for non-constant variance is basically the application of one-way ANOVA to a set of absolute distances from group-specific medians. Fit a one-way ANOVA model that uses your newly computed distances as the response variable, and the original grouping variable as the explanatory factor. Demonstrate that your results match the output from leveneTest(.) in part (b). (2)

"""

```
model <- aov(abs_dev ~ Type, data = shutoff25)
summary(model)
```

"""

f) Turn your work from parts (c)-(e) into a function (except, of course, the call to leveneTest(.)).

The user should provide a vector of responses as the first argument and a vector of group labels as the second argument. The function should create absolute deviations from the group medians, and then execute the one-way ANOVA procedure, outputting (at least) the test statistic and p-value. Your function should run in a fresh R session without loading any external libraries. Note: it is not advisable for your function to hard-code group labels in reference to any particular data set. Your function should execute properly on any two vectors (one continuous, one categorical) of equal length. (8)

"""

```
leveneTest <- function(response, group) {
  group <- as.factor(group)
  group_medians <- tapply(response, group, median)
  abs_dev <- abs(response - group_medians[as.character(group)])
  model <- aov(abs_dev ~ group)
```

```
  summary(model)[[1]][1, c("F value", "Pr(>F)")]
}

leveneTest(shutoff25$Time, shutoff25$Type)

"""
3. The following data are from a study carried out decades ago regarding attitudes toward sex education being instituted in public schools. (16 pts)

Disposition                Sex Education
                Favor      Oppose
Conservative    147        75
Moderate  212        93
Liberal   204        48
"""

#a)          Create (and print) a labeled matrix object to store the data. (2)
sex_data <- matrix(c(147, 75, 212, 93, 204, 48),
                   nrow = 3, byrow = TRUE,
                   dimnames = list(
                     position = c("Conservative", "Moderate", "Liberal"),
                     Opinion = c("Favor", "Oppose")))

sex_data

"""
b)          Produce a single barplot that displays all six cell counts. The plot should contain six bars, and category labels should be visible. (4)
"""
barplot(t(sex_data), beside = TRUE,
      col = c("blue", "red"),
      legend.text = colnames(sex_data),
```

```
        names.arg = rownames(sex_data),
        main = "Sex Education Support by Position",
        xlab = "Position",
        ylab = "Count")



"""
c)              Carry out the chi-square test for association. Provide the hypotheses, test statistic, p-value, and conclusion in context. (4)
"""
chi_result <- chisq.test(sex_data)
chi_result


#d)             Reproduce the p-value from part (c) using the pchisq(.) function. (2)
1 - pchisq(chi_result$statistic, df = chi_result$parameter)


"""
e)              Examine the standardized residuals from the chi-square test, and describe any way in which the
observed data depart significantly from independence. (4)
"""
chi_result$stdres




"""
4. In our course, we briefly considered fitting linear regression models. Assumptions behind this model include that the model
residuals are normally distributed with constant variance. We evaluated the constant variance assumption by visually examining a
plot of residuals vs fitted values; we will now consider a formal hypothesis test (known as the Breusch-Pagan test) where
H_0 specifies constant variance. (20 pts)
"""


"""
```

a) Obtain a subset of the iris data: only the 50 observations made on the “virginica” species. Verify that you have done this by showing that the sample mean petal length computed from your subset data frame is 5.552. (1)

"""

```
virginica <- subset(iris, Species == "virginica")
mean(virginica$Petal.Length)
```

"""

b) Fit a simple linear regression model using petal length as the response variable and sepal width as the single predictor. Present a residual plot. On the basis of this plot, explain whether constant variance seems plausible. (4)

"""

```
model <- lm(Petal.Length ~ Sepal.Width, data = virginica)

plot(fitted(model), resid(model),
     main = "Residuals vs Fitted",
     xlab = "Fitted Values",
     ylab = "Residuals",
     pch = 19)
abline(h = 0, lty = 2)

#There's no funnel shape or curved pattern which indicates Homoscedasticity
```

"""

c) Denote the model residuals e_i for i=1,…,n. Append a new column to the data frame from part (a). This column should store the squared residuals, e_i^2. (2)

"""

```
virginica$resid_sq <- resid(model)^2
head(virginica)
```

"""

d) Obtain the error sum of squares from the model fit in part (b) as SSE=∑_(i=1)^n▒e_i^2 . (1)

"""

```
SSE <- sum(virginica$resid_sq)
```

SSE

"""

e) Fit the following regression model:

e_i^2=β_0+β_1 Sepal.Width_i+ε_i

and obtain the regression sum of squares. Call this value SSR^* and report its value. (2)

"""

bp_model <- lm(resid_sq ~ Sepal.Width, data = virginica)

SSR_star <- sum((fitted(bp_model) - mean(virginica$resid_sq))^2)

SSR_star

"""

f) The B-P test allows for the possibility that the variance (on log scale) is linearly related to the predictor variable in the model (in this case: log⁡〖(σ_i)=γ_0+γ_1 Sepal.Width_i 〗). The hypotheses for the B-P test are as follows.

H_0: γ_1=0

H_a: γ_1≠0

Explain which of these hypotheses is consistent with constant variance. (2)

Ho tells us that variance is constant (homoscedasticity)

Ha tells us that variance depends on Sepal.Width (heteroscedasticity)

Thus Ho is the assumption of constant variance.

"""

"""

g) The Breusch-Pagan test statistic is defined as:

X_BP^2=(SSR^*/2)/(SSE/n)^2

Report the value of the test statistic. (2)

"""

n <- nrow(virginica)

X_BP <- (SSR_star / 2) / ((SSE / n)^2)

X_BP_sq

"""

h) When H_0 is true, X_BP^2 is distributed as a chi-square random variable with degrees of freedom equal to the number of predictor variables included in the original regression model (i.e. the model fit in part (b)). Despite the two-sided alternative hypothesis, the p-value from the B-P test is always the probability in the right tail of the null distribution. Use one of R's probability functions to obtain the p-value corresponding to your test statistic. (2)

"""

```
p_value <- 1 - pchisq(X_BP, df = 1)
p_value
```

"""

i) State the conclusion of the B-P test using the context of the problem. (2)

Since the p-value is greater than 0.05, we reject Ho. There is significant evidence that the residual variance is constant(homoscedasticity is present)

"""

"""

j) The "skedastic" package contains a function called breusch_pagan(.). Show that this function reproduces your results from parts (g) and (h). Note: you will need to set the argument "koenker" to FALSE. (2)

"""

```
install.packages("skedastic")
library(skedastic)

breusch_pagan(model, koenker = FALSE)
```

"""

5. Suppose there is a normally distributed response variable (Y), a uniform random variable (U) generated between 0 and 25, an unseen random process (R) that is loosely correlated with Y, and a predictor variable (X) that investigators observe as a

deviation around R. To summarize, the following five random variables are involved: (28 pts)

"""

"""

a) Set the randomization seed to 1985. Generate n=40 observations of Y, U, and ε according to the distributions described above. Present the sample mean of each of the three vectors. (2)

"""

```
set.seed(1985)
n <- 40

Y <- rnorm(n, mean = 200, sd = 15)
U <- runif(n, min = 0, max = 25)
epsi <- rnorm(n, mean = 0, sd = 10)

mean_y <- mean(Y)
mean_u <- mean(U)
mean_epsi <- mean(epsi)

mean_y
mean_u
mean_epsi
```

"""

b) Create the variables R and X as defined below. Present the sample correlation between X and Y. (2)

"""

```
R <- 0.2 * Y + U
X <- R + epsi
cor_XY <- cor(X, Y)
cor_XY
```

"""

c) Fit the simple linear regression model using Y and X (Y as the response), and test whether there is a linear relationship present.

Provide the hypotheses, test statistic, p-value, and conclusion. (4)

"""

```
model <- lm(Y ~ X)

summary(model)
```

"""

d)

We will now investigate the power of the test performed above. You will no longer set a randomization seed. Write a program to execute the following tasks B=10,000 times:

- Generate n=40 observations of Y, U, and ε.
- Create R and X.
- Regress Y on X.
- Note whether the test for a linear relationship rejects H_0.

Present your simulation-based estimate of the power of the test. Provide the full code required to perform your simulation. (4)

"""

```
B <- 10000

n <- 40

alpha <- 0.05

rejects <- numeric(B)


for (i in 1:B) {

 Y <- rnorm(n, mean = 200, sd = 15)

 U <- runif(n, 0, 25)

 epsilon <- rnorm(n, 0, 10)

 R <- 0.2 * Y + U

 X <- R + epsilon

 model <- lm(Y ~ X)

 p_val <- summary(model)$coefficients["X", "Pr(>|t|)"]

 rejects[i] <- (p_val < alpha)

 rejects[i]

}
```

```
power_estimate <- mean(rejects)
power_estimate
rejects
```

"""

e) The original setup uses 15 as the population standard deviation of the response variable. Investigate what happens to the power of the test as this value changes. Fill in each entry in the table below, where each value is an estimate based on a simulation of size B=10,000. You do not need to provide your code here. (3)

"""

```
B <- 10000
n <- 40
alpha <- 0.05
rejects <- numeric(B)

for (i in 1:B) {
  Y <- rnorm(n, mean = 200, sd = 15)
  U <- runif(n, 0, 25)
  epsilon <- rnorm(n, 0, 10)
  R <- 0.2 * Y + U
  X <- R + epsilon
  model <- lm(Y ~ X)
  p_val <- summary(model)$coefficients["X", "Pr(>|t|)"]
  rejects[i] <- (p_val < alpha)
  rejects[i]
}

power_estimate <- mean(rejects)
power_estimate
```

"""

f) Return to using σ_Y=15. Investigate the impact of the value σ_ε on the power of the test.

Again, no code need be presented here, and each simulation should use B=10,000. (3)

"""

```
B <- 10000
n <- 40
alpha <- 0.05
rejects <- numeric(B)

for (i in 1:B) {
  Y <- rnorm(n, mean = 200, sd = 15)
  U <- runif(n, 0, 25)
  epsilon <- rnorm(n, 0, 20)
  R <- 0.2 * Y + U
  X <- R + epsilon
  model <- lm(Y ~ X)
  p_val <- summary(model)$coefficients["X", "Pr(>|t|)"]
  rejects[i] <- (p_val < alpha)
  rejects[i]
}

power_estimate <- mean(rejects)
power_estimate
```

"""

g) Return to using σ_Y=15 and σ_ε=10. The original definition of the unseen process was:

R=0.2Y+U

Investigate the impact that the coefficient on Y (originally 0.2) has on the power of the test. Complete the table below based on simulations of size B=10,000. No code needs to be presented. (3)

"""

```
B <- 10000
n <- 40
alpha <- 0.05
rejects <- numeric(B)
```

```
for (i in 1:B) {
  Y <- rnorm(n, mean = 200, sd = 15)
  U <- runif(n, 0, 25)
  epsilon <- rnorm(n, 0, 10)
  R <- 0.5 * Y + U
  X <- R + epsilon
  model <- lm(Y ~ X)
  p_val <- summary(model)$coefficients["X", "Pr(>|t|)"]
  rejects[i] <- (p_val < alpha)
  rejects[i]
}

power_estimate <- mean(rejects)
power_estimate



"""
h)  Return to all original settings (σ_Y=15, σ_ε=10, coefficient=0.2). Investigate the impact of the sample size n on the power of the test.
Again use B=10,000; no code required. (3)
"""
B <- 10000
n <- 50
alpha <- 0.05
rejects <- numeric(B)

for (i in 1:B) {
  Y <- rnorm(n, mean = 200, sd = 15)
  U <- runif(n, 0, 25)
  epsilon <- rnorm(n, 0, 10)
  R <- 0.2 * Y + U
  X <- R + epsilon
  model <- lm(Y ~ X)
  p_val <- summary(model)$coefficients["X", "Pr(>|t|)"]
  rejects[i] <- (p_val < alpha)
```

```
  rejects[i]
}

power_estimate <- mean(rejects)
power_estimate
```

"""

i) Write clear statements summarizing how each item investigated in parts (e)-(h) affects the power of the test. (4)

When we increase population standard deviation, the power increases. When σ_ε increases, the power decreases.

When the coefficient on Y increases, the power also increases. When the sample size(n) increases, the power also increases.

"""

"""

6. In this problem, we will simulate aspects of a fun game called Zombie Dice®. While not required, one way to simulate rolling a standard 6-sided die is to generate a uniform random variable between 1 and 7 using the runif(.) function, and then round down by using the floor(.) function.

The following table gives a mapping between ordinary six-sided dice and the yellow variety of Zombie Dice:

Ordinary Dice	Zombie Dice (yellow)
{1, 2}	brain
{3, 4}	escape
{5, 6}	attacked

In this game, each player is a zombie. The goal of the game is to score points by eating brains. According to the chart above, if a 1 or 2 is rolled, you get a nutritious brain to eat. If a 3 or 4 is rolled, you don't get any brains (your intended victim escapes). However, if a 5 or 6 is rolled, the "victim" attacks you. A player rolls dice until they are attacked three times. (25 pts)

"""

#a) According to probability theory, what is the probability that a single roll of a yellow die results in a brain? (1)

#There are 6 possible outcomes and 2 of them result on brain, so 2/6

"""

b) Write a block of code that randomly chooses an integer from the set {1,2,3,4,5,6} and outputs the text “brain,” “escaped,” or “attacked” according to the mapping in the problem’s description. This will represent one simulated roll of a yellow die. Display output from a single execution of your code block. (3)

"""

```
roll <- floor(runif(1, min=1, max=7))
result <- ifelse(roll <= 2, "brain", ifelse(roll <= 4, "escaped", "attacked"))
result
```

"""

c) Simulate rolling a yellow zombie die 100 times and display the results using the table(.) function. Use these results to estimate the probability for a single yellow die to result in a brain. Also present the numeric difference between your simulation-based result and the theoretical probability from part (a). (3)

"""

```
results <- floor(runif(100, 1, 7))
outcomes <- ifelse(results <= 2, "brain", ifelse(results <= 4, "escaped", "attacked"))
table(outcomes)
mean(outcomes == "brain")
abs(mean(outcomes == "brain") - 2/6)
```

"""

d) Now simulate rolling a yellow zombie die 100,000 times and tabulate the results. Is the resulting simulation-based estimate of P(brain) closer to the true probability than the estimate from part (c)? Should it be? (3)

"""

```
results <- floor(runif(100000, 1, 7))
outcomes <- ifelse(results <= 2, "brain", ifelse(results <= 4, "escaped", "attacked"))
```

```
table(outcomes)

mean(outcomes == "brain")

abs(mean(outcomes == "brain") - 2/6)
```

```
"""
e)  Suppose that a player is about to roll three yellow zombie dice at the same time. What is the theoretical probability that none
of the three victims escape (i.e. they are either eaten, or they fight back)? (1)

P(eaten or fought back) = (4/6 * 4/6 * 4/6) = 0.29629629629
"""
```

```
"""
f)  Write a simulation in which a player rolls three yellow zombie dice at the same time. Create variables called roll1,
roll2, and roll3 to store these values. Display a single execution of this program. (2)
"""
roll1 <- floor(runif(1, 1, 7))

roll2 <- floor(runif(1, 1, 7))

roll3 <- floor(runif(1, 1, 7))

c(roll1, roll2, roll3)
```

```
"""
g)  Simulate a player rolling the three dice 100,000 times. Store the results, and use them to estimate the probability that
none of the three victims escape. Note that your program should be doing no multiplication. You should use logical
operators to observe in simulation how often 0 of 3 victims escape. (3)
"""
trials <- 100000

none_escaped <- logical(trials)
```

```
for (i in 1:trials) {
  rolls <- floor(runif(3, 1, 7))
  escapes <- rolls %in% c(3, 4)
  none_escaped[i] <- sum(escapes) == 0
}
mean(none_escaped)
```

"""

h) A player is about to roll three yellow zombie dice. What is the theoretical probability that the turn will end immediately (i.e. the player is attacked by all three victims)? (1)

"""

```
pattacked <- (2/6)^3
pattacked
```

"""

i) Use 100,000 simulated rolls of three yellow zombie dice to estimate the probability that the player's turn ends immediately. Again, your program needs to use logical operators, not multiplication. (3)

"""

```
all_attacks <- logical(trials)

for (i in 1:trials) {
  rolls <- floor(runif(3, 1, 7))
  attacks <- rolls %in% c(5, 6)
  all_attacks[i] <- sum(attacks) == 3
}

mean(all_attacks)
```

"""

j) There are actually zombie dice of three different colors. Green dice represent the most vulnerable victims, while red dice represent victims who are more likely to defend themselves. The following tables describe these other types of dice:

The complete Zombie Dice set contains six green dice, four yellow dice, and three red dice. As a fun experiment, consider a player rolling all 13 of these dice. We are interested in the extremely unlikely event that all 13 dice result in attacks. Write a new simulation in which you create variables roll1, roll2, …, roll13. Define the random variable X={# of attacks when rolling all 13 dice}. Carry out the experiment a million times. In your million simulated trials, does a player ever get attacked by all 13 victims? No multiplication should take place in this problem; only simulation and summarization. (5)

"""

```
green <- function(n) {
  vals <- floor(runif(n, 1, 7))
  ifelse(vals <= 3, "brain", ifelse(vals <= 5, "escaped", "attacked"))
}

yellow <- function(n) {
  vals <- floor(runif(n, 1, 7))
  ifelse(vals <= 2, "brain", ifelse(vals <= 4, "escaped", "attacked"))
}

red <- function(n) {
  vals <- floor(runif(n, 1, 7))
  ifelse(vals == 1, "brain", ifelse(vals <= 3, "escaped", "attacked"))
}



trials <- 1000000
total_attacks <- integer(trials)

for (i in 1:trials) {
  res <- c(green(6), yellow(4), red(3))
  total_attacks[i] <- sum(res == "attacked")
}

mean(total_attacks == 13)
```