Problem Set 6

Matt Krasnow

Due Friday, November 1, 2024 at 11:59pm

Problem set policies. *Please provide concise, clear answers for each question while making sure to fully explain your reasoning. For problems asking for calculations, show your work in addition to clearly indicating the final answer. For problems involving* `R`*, be sure to include the code and output in your solution.*

Please submit the PDF of your knit solutions to Gradescope and be sure to assign which pages of your solution correspond to each problem. Make sure that the PDF is fully readable to the graders; e.g., make sure that lines don't run off the page margin.

We encourage you to discuss problems with other students (and, of course, with the teaching team), but you must write your final answer in your own words. Solutions prepared "in committee" are not acceptable. If you do collaborate with classmates on a problem, please list your collaborators on your solution. Be aware that simply copying answers found online, whether human-generated or machine-generated, is a violation of the Honor Code.

Question 1

The file `pregnancydata.csv` includes several variables to model the birthweight of babies (measured through an online survey). Those variables are defined below. Use this dataset in R to answer the questions below:

`id`: a unique identifier of the mother
`weight`: birthweight of the newborn baby, in ounces
`pregnancylength`: the length of the pregnancy, in days
`country`: where the birth took place; categories are United States (US), United Kingdom (UK), Canada (Can), and Other
`motherage`: age of mother at childbirth, in years
`multiples`: whether the baby was a 1=singleton or 2=twin
`sex`: sex of the baby: girl or boy
`induced`: a binary indicator for whether labor was induced with oxytocin
`cesarean`: a binary indicator for whether a cesarean (c-section) was performed
`previousbirths`: the number of births by the mother previous to this recorded one (from 0 to 10)

(a) Fit a regression model to predict weight from country and use the `relevel` command to make the "Other" group the reference group (call this **Model 1**). Interpret the results and provide a visual to support your conclusions.

```
# Load necessary libraries
library(ggplot2)
library(dplyr)


pregnancydata <- read.csv("data/pregnancydata.csv")


str(pregnancydata)
```

```
## 'data.frame':    9065 obs. of  10 variables:
##  $ id             : int  158 2738 2739 2740 2741 2742 2745 2748 2749 2750 ...
##  $ weight         : num  116 103 101 125 133 101 106 105 120 114 ...
##  $ pregnancylength: int  278 270 283 278 289 276 262 269 274 282 ...
##  $ country        : chr  "US" "US" "US" "US" ...
##  $ motherage      : int  30 26 25 29 28 22 30 30 25 15 ...
##  $ multiples      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ sex            : chr  "girl" "girl" "girl" "boy" ...
##  $ induced        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ cesarean       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ previousbirths : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
summary(pregnancydata)
```

```
##        id            weight       pregnancylength   country
##  Min.   :  158   Min.   : 29.28   Min.   :158.0   Length:9065
##  1st Qu.: 5732   1st Qu.:111.00   1st Qu.:273.0   Class :character
##  Median : 8531   Median :122.00   Median :280.0   Mode  :character
##  Mean   : 8455   Mean   :121.99   Mean   :278.1
##  3rd Qu.:11168   3rd Qu.:133.00   3rd Qu.:286.0
##  Max.   :13798   Max.   :196.00   Max.   :327.0
##    motherage       multiples         sex                induced
##  Min.   :13.00   Min.   :1.000   Length:9065        Min.   :0.0000
##  1st Qu.:23.00   1st Qu.:1.000   Class :character   1st Qu.:0.0000
```

```
##  Median :27.00   Median :1.000   Mode  :character   Median :0.0000
##  Mean   :26.98   Mean   :1.001                      Mean   :0.2375
##  3rd Qu.:31.00   3rd Qu.:1.000                      3rd Qu.:0.0000
##  Max.   :59.00   Max.   :2.000                      Max.   :1.0000
##     cesarean      previousbirths
##  Min.   :0.0000   Min.   : 0.0000
##  1st Qu.:0.0000   1st Qu.: 0.0000
##  Median :0.0000   Median : 0.0000
##  Mean   :0.1316   Mean   : 0.3733
##  3rd Qu.:0.0000   3rd Qu.: 1.0000
##  Max.   :1.0000   Max.   :10.0000
```

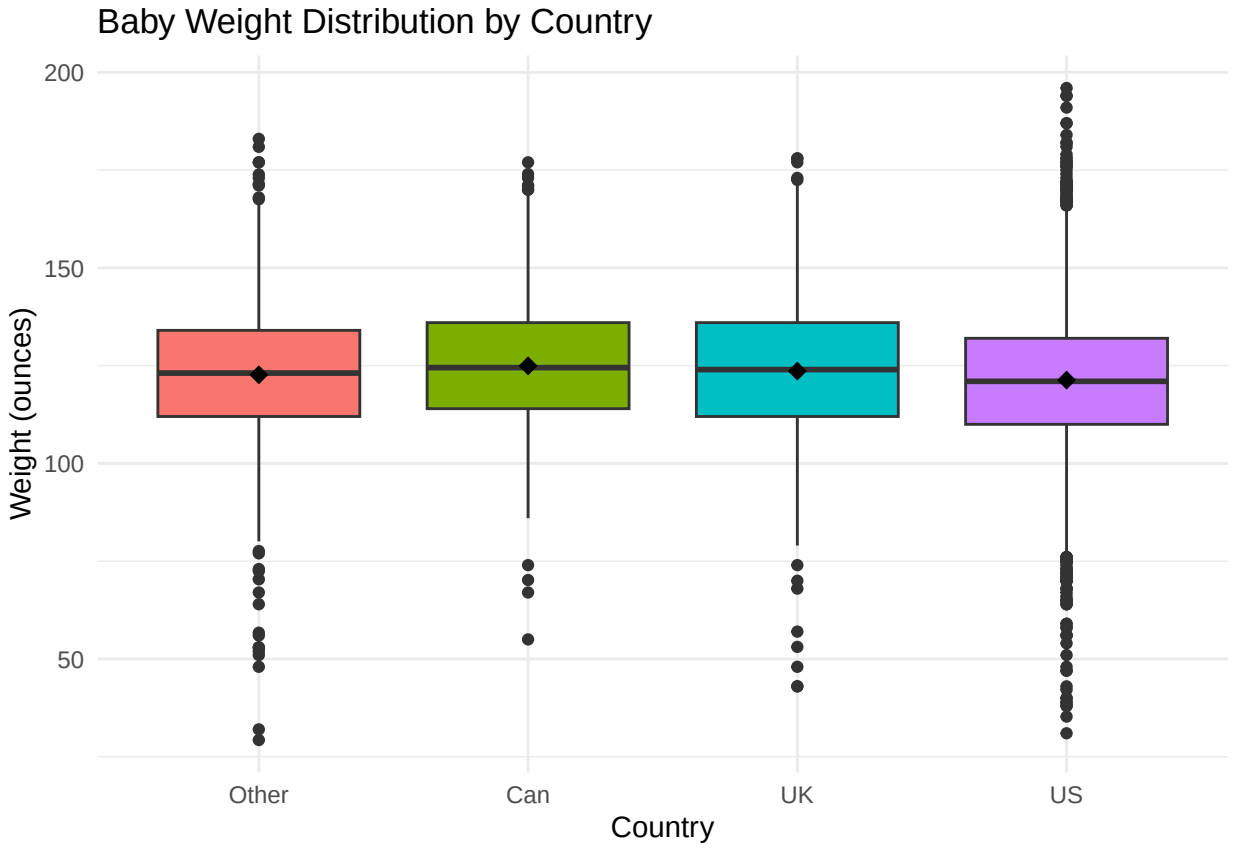
```
pregnancydata$country <- relevel(factor(pregnancydata$country), ref = "Other")

# Verify the releveling
levels(pregnancydata$country)
```

```
## [1] "Other" "Can"   "UK"    "US"
```

```
model1 <- lm(weight ~ country, data = pregnancydata)


summary(model1)
```

```
##
## Call:
## lm(formula = weight ~ country, data = pregnancydata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -93.380 -11.311  -0.311  11.383  74.689
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 122.6570     0.6083 201.631   <2e-16 ***
## countryCan    2.2965     0.9712   2.365   0.0181 *
## countryUK     0.9596     0.7868   1.220   0.2227
## countryUS    -1.3458     0.6480  -2.077   0.0378 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.82 on 9061 degrees of freedom
## Multiple R-squared:  0.004002,   Adjusted R-squared:  0.003672
## F-statistic: 12.13 on 3 and 9061 DF,  p-value: 6.366e-08
```

```
# boxplot
ggplot(pregnancydata, aes(x = country, y = weight, fill = country)) +
  geom_boxplot() +
  theme_minimal() +
  labs(title = "Baby Weight Distribution by Country",
       x = "Country",
       y = "Weight (ounces)",
       fill = "Country") +
  theme(legend.position = "none") +
  stat_summary(fun = mean, geom = "point", shape = 18, size = 3, color = "black")
```



Interpretation:

There is a significant difference–the country is a predictor of the birthweight. Hoever, the R^2 value is very small, indicating that it explains a very small percentage of the variation in the data. This shows us that our model of just country may not be sufficient. It has statistical signficance, but doesn't really have much practical significance because R^2 is so low–the model is not useful.

interpretation: - canadian babies are on average 2.2965 ounces heaver than the other category - american babies are on average .95 ounces heaver than the other category - british babies are on average 1.3458 ounces lighter than the other category

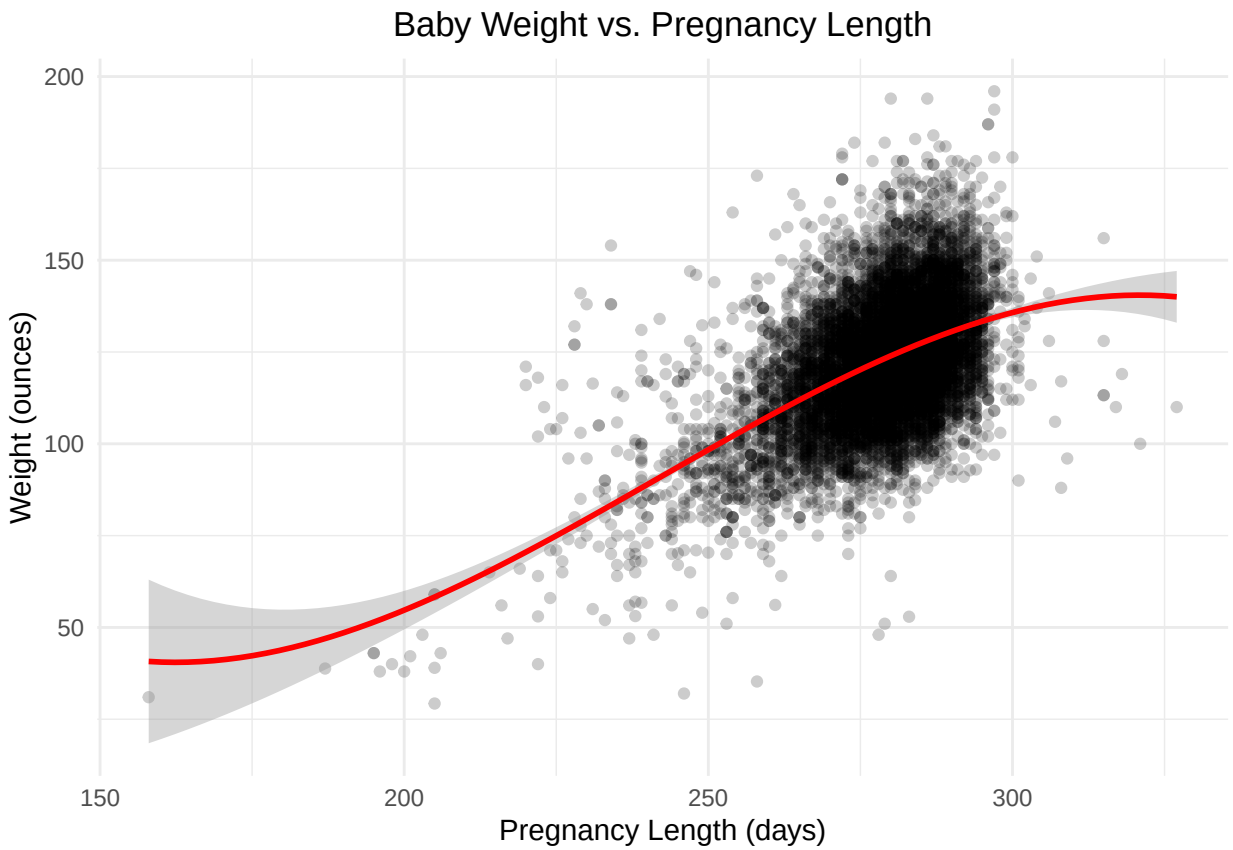
(b) Build a 3^{rd} order polynomial regression model to predict weight from `pregnancylength` (call this **Model 2**). Interpret the output and provide a visual to support the results of the model.

```
model2 <- lm(weight ~ poly(pregnancylength, 3), data = pregnancydata)


summary(model2)
```

```
##
## Call:
## lm(formula = weight ~ poly(pregnancylength, 3), data = pregnancydata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -74.486 -10.087  -0.761   9.364  70.727
##
## Coefficients:
```

```
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  121.986      0.161 757.591  < 2e-16 ***
## poly(pregnancylength, 3)1  866.039     15.331  56.491  < 2e-16 ***
## poly(pregnancylength, 3)2  -68.993     15.331  -4.500 6.87e-06 ***
## poly(pregnancylength, 3)3  -64.594     15.331  -4.213 2.54e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.33 on 9061 degrees of freedom
## Multiple R-squared:  0.2627, Adjusted R-squared:  0.2625
## F-statistic:  1076 on 3 and 9061 DF,  p-value: < 2.2e-16
```

```r
ggplot(pregnancydata, aes(x = pregnancylength, y = weight)) +
  geom_point(alpha = 0.2) +  # Plot raw data with transparency
  geom_smooth(method = "lm", formula = y ~ poly(x, 3),
              color = "red", se = TRUE) +  # Add polynomial fit with confidence interval
  theme_minimal() +
  labs(title = "Baby Weight vs. Pregnancy Length",
       x = "Pregnancy Length (days)",
       y = "Weight (ounces)") +
  theme(plot.title = element_text(hjust = 0.5))
```




```r
# Create sequence of pregnancy lengths
pred_data <- data.frame(
  pregnancylength = seq(min(pregnancydata$pregnancylength),
                        max(pregnancydata$pregnancylength),
                        length.out = 100)
)
```

```
predictions <- predict(model2, newdata = pred_data, interval = "confidence")
pred_data$fit <- predictions[,"fit"]
```

Interpretation

- the model parameters are signficant,
- the R^2 of the model is higher than model 1
- this indicates that there is a non linear relationship between pregnancy length and weight.
- it has an S shape
- birth weight increases as pregancy length increases, but levels out eventually
- still, most of the variation of the weight is unexplained by the model
- the spread of the data is very interesting–we suspect that there might be another factor that is more important than pregnancy length

(c) Use **Model 2** to estimate the probability that a baby will weigh less than 7 pounds (112 ounces) when born on day 280.

```
# Create new data point
newdata <- data.frame(pregnancylength = 280)

# Get prediction and standard error
pred <- predict(model2, newdata = newdata, se.fit = TRUE)


fit <- pred$fit
se_fit <- pred$se.fit
residual_se <- summary(model2)$sigma


pred_se <- sqrt(se_fit^2 + residual_se^2)


z_score <- (112 - fit) / pred_se

# Calculate probability using normal distribution
# I could use the t distribution, but at 9061 degrees of freedom, this is negligible and it is acceptab
prob <- pnorm(z_score)


print(paste("Predicted weight:", round(fit, 2), "ounces"))
```

```
## [1] "Predicted weight: 123.96 ounces"
```

```
print(paste("Standard error:", round(pred_se, 2), "ounces"))
```

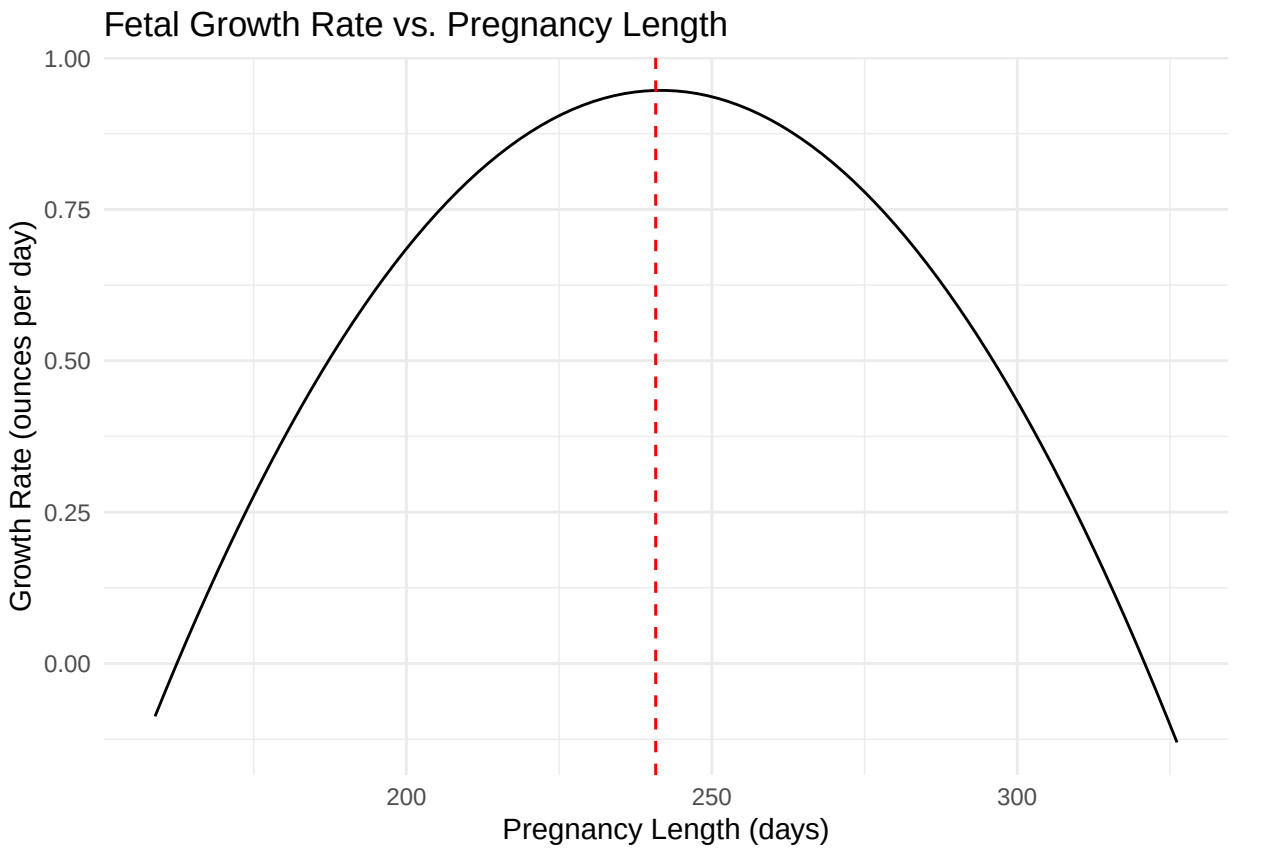
```
## [1] "Standard error: 15.33 ounces"
```

```
print(paste("Probability of weight < 112 ounces:", round(prob, 4)))
```

```
## [1] "Probability of weight < 112 ounces: 0.2177"
```

(d) It is of medical interest to determine at what gestational age a developing fetus is gaining weight the fastest. Use **Model 2** to estimate this *period of fastest growth*.

```r
# Get the raw polynomial terms and fit--not orthogonal
days <- seq(min(pregnancydata$pregnancylength),
            max(pregnancydata$pregnancylength),
            length.out = 100)

# Get predicted weights across pregnancy lengths
pred_data <- data.frame(pregnancylength = days)
predicted_weights <- predict(model2, newdata = pred_data)

# Calculate derivatives
# Use finite differences to approximate growth rate
growth_rate <- diff(predicted_weights)/diff(days)
mid_days <- days[-1] - diff(days)/2  # midpoints for plotting

#  visualization of growth rate
ggplot(data.frame(days = mid_days, rate = growth_rate),
       aes(x = days, y = rate)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Fetal Growth Rate vs. Pregnancy Length",
       x = "Pregnancy Length (days)",
       y = "Growth Rate (ounces per day)") +
  geom_vline(xintercept = mid_days[which.max(growth_rate)],
             linetype = "dashed",
             color = "red")
```



```r
# Find the day of maximum growth
max_growth_day <- mid_days[which.max(growth_rate)]
max_growth_rate <- max(growth_rate)

print(paste("Day of maximum growth:", round(max_growth_day, 1)))
```

```
## [1] "Day of maximum growth: 240.8"
```

```r
print(paste("Maximum growth rate:", round(max_growth_rate, 3), "ounces per day"))
```

```
## [1] "Maximum growth rate: 0.946 ounces per day"
```

Question 2

In this problem, we will attempt to investigate whether the COVID-19-related restrictions imposed by the government had any effect on the reporting of criminal activity in the Boston Police Department (BPD). For this purpose, we will use the `bpd.csv` dataset, which includes the number of daily incident reports filed (`count`) and various weather indicators on those days (`maxtemp` is the only weather variable we will use in this problem).

Note: a state of emergency was declared in Massachusetts on March 10, 2020, and restrictions on non-essential businesses, schools, and MBTA service were mainly put into effect on March 17,2020 (see this City of Boston article for the timeline).

The R chunk below reads in the data and includes some code to create a variable called `dayinyear` in the `bpd` data frame that counts the number of days into the year, starting with 0 for Jan 1.

```
bpd = read.csv('data/bpd.csv')

jan1_19 = as.Date("1/1/19",format="%m/%d/%y")
jan1_20 = as.Date("1/1/20",format="%m/%d/%y")
jan1_21 = as.Date("1/1/21",format="%m/%d/%y")

bpd$dayinyear = as.Date(bpd$date,format="%m/%d/%y") - jan1_19
bpd$dayinyear[bpd$year==2020] =
  as.Date(bpd$date,format="%m/%d/%y")[bpd$year==2020] - jan1_20
bpd$dayinyear[bpd$year==2021] =
  as.Date(bpd$date,format="%m/%d/%y")[bpd$year==2021] - jan1_21
```

(a) Create a binary/dummy variable (call it `restrictions`) to indicate whether that day falls under the time period of state of emergency or restricted business operations in the city of Boston (all dates between and including March 10, 2020 and Friday, May 28, 2020). How many days fall in this time period in the dataset?

```
# Create date objects for restriction period
restriction_start = as.Date("3/10/20", format="%m/%d/%y")
restriction_end = as.Date("5/28/20", format="%m/%d/%y")

# Create the restrictions binary variable
bpd$restrictions = as.numeric(
  as.Date(bpd$date, format="%m/%d/%y") >= restriction_start &
  as.Date(bpd$date, format="%m/%d/%y") <= restriction_end
)

# Count days in restriction period
days_in_restrictions = sum(bpd$restrictions)

# Print result
print(paste("Number of days under restrictions:", days_in_restrictions))
```

```
## [1] "Number of days under restrictions: 80"
```

(b) Calculate the mean number of daily incident reports filed by the BPD during the restriction orders in Boston. Separately calculate the mean number of daily incident reports for a comparison group with the same calendar dates in the pre-pandemic portion of the data. Use these two groups to calculate a reasonable 95% confidence interval for the effect of COVID-19 restrictions on the reporting of crime in the BPD (based on a simple 2-group comparison method and not linear regression).

```
bpd$comparison_group = as.numeric(
  bpd$year == 2019 &
```

```
  as.numeric(format(as.Date(bpd$date, format="%m/%d/%y"), "%m%d")) >=
    as.numeric(format(as.Date("3/10/20", format="%m/%d/%y"), "%m%d")) &
  as.numeric(format(as.Date(bpd$date, format="%m/%d/%y"), "%m%d")) <=
    as.numeric(format(as.Date("5/28/20", format="%m/%d/%y"), "%m%d"))
)

restriction_mean = mean(bpd$count[bpd$restrictions == 1])
comparison_mean = mean(bpd$count[bpd$comparison_group == 1])

t_result = t.test(
  bpd$count[bpd$restrictions == 1],
  bpd$count[bpd$comparison_group == 1]
)

# Print results
cat("Mean daily incidents during restrictions:", round(restriction_mean, 2), "\n")
```

```
## Mean daily incidents during restrictions: 159.12
```

```
cat("Mean daily incidents in comparison period:", round(comparison_mean, 2), "\n")
```

```
## Mean daily incidents in comparison period: 264.21
```

```
cat("\n95% CI for effect of restrictions:",
    round(t_result$conf.int[1], 2), "to",
    round(t_result$conf.int[2], 2), "\n")
```

```
##
## 95% CI for effect of restrictions: -114.8 to -95.37
```

(c) Fit a linear regression model to predict `count` from `maxtemp` and `restrictions` (call it **model1**), and print out the `summary` results. Briefly interpret the coefficient estimates and use this model to estimate the effect of COVID-19 restrictions on the reporting of crime in the BPD (with 95% confidence).

```
model1 = lm(count ~ maxtemp + restrictions, data = bpd)


summary_model1 = summary(model1)

# Print summary
print(summary_model1)
```

```
##
## Call:
## lm(formula = count ~ maxtemp + restrictions, data = bpd)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -146.34  -31.89   -6.60   31.44  120.94
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  181.60815    4.58837  39.580   <2e-16 ***
## maxtemp        0.70066    0.07195   9.739   <2e-16 ***
## restrictions -60.67812    4.92423 -12.322   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42.21 on 1093 degrees of freedom
## Multiple R-squared:  0.1993, Adjusted R-squared:  0.1979
## F-statistic: 136.1 on 2 and 1093 DF,  p-value: < 2.2e-16
```

```
ci_model1 = confint(model1)

cat("\n95% CI for effect of restrictions:",
    round(ci_model1["restrictions",1], 2), "to",
    round(ci_model1["restrictions",2], 2), "\n")
```

```
##
## 95% CI for effect of restrictions: -70.34 to -51.02
```

interpretation:

- this is more accurate because we are controlling for the temperature
- if we were to repeat this process, 95% of the generated intervals would contain the true value
- this means that ther eare fewer incidents during restrictions

(d) Fit a linear regression model to predict `count` from `maxtemp`, `restrictions`, `dayinyear` and all 2-way interactions between these 3 predictors (call it **model2**), and print out the `summary` results. Interpret what this model says about the effect of restrictions when `maxtemp`=0 and `dayinyear`=0 (point estimate only is fine). Also provide a point estimate for `count` on the 91st day of the year in 2020, assuming the temperature was 50 degrees. Do the same for 2019 and compare the difference.

```
# First convert dayinyear to numeric in the original data
bpd$dayinyear = as.numeric(bpd$dayinyear)

# Now fit the model with interactions
model2 = lm(count ~ maxtemp + restrictions + dayinyear +
            maxtemp:restrictions + maxtemp:dayinyear + restrictions:dayinyear,
            data = bpd)

# Print summary
print(summary(model2))
```

```
##
## Call:
## lm(formula = count ~ maxtemp + restrictions + dayinyear + maxtemp:restrictions +
##     maxtemp:dayinyear + restrictions:dayinyear, data = bpd)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -146.051  -31.478   -6.595   30.792  118.593
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              1.925e+02  9.342e+00  20.609  < 2e-16 ***
## maxtemp                  4.720e-01  1.929e-01   2.447  0.01457 *
## restrictions            -7.807e+01  2.719e+01  -2.872  0.00416 **
## dayinyear               -5.861e-02  4.737e-02  -1.237  0.21623
## maxtemp:restrictions     9.130e-01  5.327e-01   1.714  0.08683 .
## maxtemp:dayinyear        1.174e-03  9.534e-04   1.232  0.21828
## restrictions:dayinyear -2.911e-01  2.443e-01  -1.192  0.23371
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42.21 on 1089 degrees of freedom
## Multiple R-squared:  0.2024, Adjusted R-squared:  0.198
## F-statistic: 46.06 on 6 and 1089 DF,  p-value: < 2.2e-16
```

```
# Get coefficient for restrictions (effect when maxtemp=0 and dayinyear=0)
restrictions_effect = coef(model2)["restrictions"]
cat("\nEffect of restrictions when maxtemp=0 and dayinyear=0:",
    round(restrictions_effect, 2), "\n")
```

```
##
## Effect of restrictions when maxtemp=0 and dayinyear=0: -78.07
```

```
# Predictions for day 91, temp 50
# Create prediction data frames with numeric dayinyear
pred_2020 = data.frame(
  maxtemp = 50,
  restrictions = 1,  # during restrictions
  dayinyear = 91
)

pred_2019 = data.frame(
  maxtemp = 50,
  restrictions = 0,  # before restrictions
  dayinyear = 91
)

# Get predictions
pred_2020_value = predict(model2, newdata = pred_2020)
pred_2019_value = predict(model2, newdata = pred_2019)

# Print results
cat("\nPredicted count for day 91, 2020 (with restrictions):",
    round(pred_2020_value, 2), "\n")
```

```
##
## Predicted count for day 91, 2020 (with restrictions): 157.23
```

```
cat("Predicted count for day 91, 2019 (no restrictions):",
    round(pred_2019_value, 2), "\n")
```

```
## Predicted count for day 91, 2019 (no restrictions): 216.14
```

```
cat("Difference (2020 - 2019):",
    round(pred_2020_value - pred_2019_value, 2), "\n")
```

```
## Difference (2020 - 2019): -58.91
```

(e) Perform a formal hypothesis test to determine whether **model2** performs significantly better at predicting `count` than **model1**.

```
# ANOVA test comparing the models
model_comparison = anova(model1, model2)

# Print results
print(model_comparison)
```

```
## Analysis of Variance Table
##
## Model 1: count ~ maxtemp + restrictions
## Model 2: count ~ maxtemp + restrictions + dayinyear + maxtemp:restrictions +
##     maxtemp:dayinyear + restrictions:dayinyear
##   Res.Df     RSS Df Sum of Sq      F Pr(>F)
## 1   1093 1947502
## 2   1089 1940064  4    7437.3 1.0437 0.3834
```

```
# Get R-squared values for context
r2_model1 = summary(model1)$r.squared
r2_model2 = summary(model2)$r.squared

cat("\nR-squared values:\n")
```

```
##
## R-squared values:
```

```
cat("Model 1:", round(r2_model1, 4), "\n")
```

```
## Model 1: 0.1993
```

```
cat("Model 2:", round(r2_model2, 4), "\n")
```

```
## Model 2: 0.2024
```

```
cat("Improvement:", round(r2_model2 - r2_model1, 4), "\n")
```

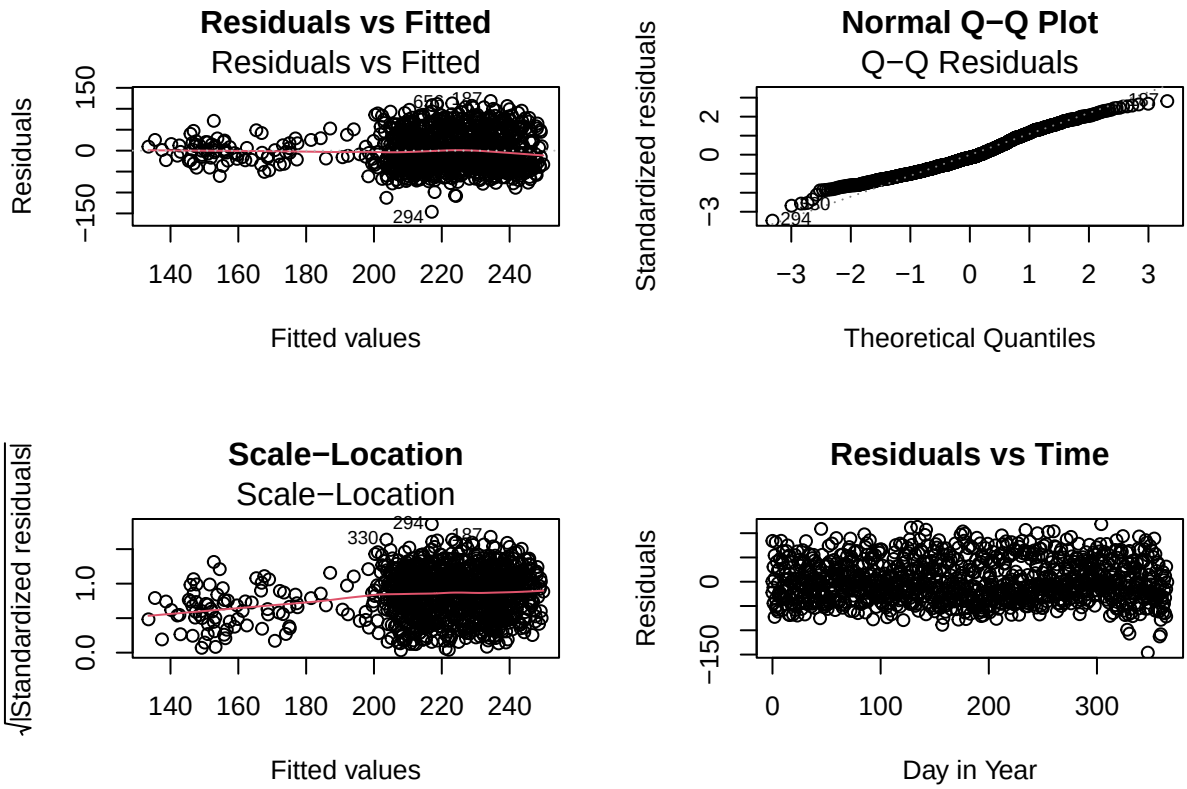
```
## Improvement: 0.0031
```

Interpreation:

- there is not a significant difference between the models at predicting the count.
- we fail to reject the null hypothesis that the models are different from each other

(f) Investigate the assumptions for **model2**. Be sure to include and reference useful visuals.

```
# Create diagnostic plots
par(mfrow=c(2,2))  # Set up 2x2 plot layout

# 1. Residuals vs Fitted (for linearity and homoscedasticity)
plot(model2, which=1, main="Residuals vs Fitted")

# 2. Q-Q plot (for normality)
plot(model2, which=2, main="Normal Q-Q Plot")

# 3. Scale-Location plot (for homoscedasticity)
plot(model2, which=3, main="Scale-Location")

# 4. Residuals vs Day (for independence)
plot(bpd$dayinyear, residuals(model2),
     xlab="Day in Year", ylab="Residuals",
     main="Residuals vs Time")
```




```
# Reset plot layout
par(mfrow=c(1,1))

# Additional numerical diagnostics
# Shapiro-Wilk test for normality
sw_test = shapiro.test(residuals(model2))

# Breusch-Pagan test for heteroskedasticity
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
bp_test = bptest(model2)

# Print numerical test results
cat("\nShapiro-Wilk test for normality:\n")
```

```
##
## Shapiro-Wilk test for normality:
```

```
print(sw_test)
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  residuals(model2)
## W = 0.97927, p-value = 2.088e-11
```

```
cat("\nBreusch-Pagan test for heteroskedasticity:\n")
```

```
##
## Breusch-Pagan test for heteroskedasticity:
```

```
print(bp_test)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model2
## BP = 28.231, df = 6, p-value = 8.498e-05
```

INTERPRETATION

- existence is satisfied because we have count data
- linearity is satistifed by the plots
- independence is satisfied by teh residuals vs time plot (random)
- homosekdasticit: NOT SATISFIED. The BP test was signficant!
- Normality of residuals: NOT SATISFIED. the SW test was significant!

(g) Determine on which four dates **model2** did the worst job at predicting `count`.

```
# Calculate absolute residuals
abs_residuals = abs(residuals(model2))

residual_df = data.frame(
  date = bpd$date,
  actual = bpd$count,
  predicted = fitted(model2),
  residual = residuals(model2),
  abs_residual = abs_residuals
)

# Order by absolute residuals and get top 4
worst_predictions = residual_df[order(-abs_residuals), ][1:4, ]

# Format the
worst_predictions$date = as.Date(worst_predictions$date, format="%m/%d/%y")
worst_predictions = worst_predictions[order(worst_predictions$date), ]


print(worst_predictions[, c("date", "actual", "predicted", "residual", "abs_residual")])
```

```
##           date actual predicted  residual abs_residual
## 656 2019-05-15    336  223.0717  112.9283     112.9283
## 187 2019-11-01    353  234.4065  118.5935     118.5935
## 294 2021-12-14     71  217.0514 -146.0514     146.0514
## 330 2021-12-25     91  203.6782 -112.6782     112.6782
```

(h) Write a short (200-300 word) summary addressing whether there is evidence that COVID-19 reduced the amount of crime in Boston. Be sure to reference the results above (specifically, which approach you think was most reasonable) and mention any biases or confounders that may be present in this relationship.

The best model, model 2, accounts for both temperature effecsts and season thorught time trends with their interactions. thisThis was better than our first model which had larger confidence intervals. However, this difference was NOT. statitically signficant, so we should be careful when interpretting the models. This indicates that they may not actually reflect different populations. When controlling for these factors, we see the deecrease in reported incidents.

We observed a statistically signicant reduction in reported incidents during the restriction perdio. However, we should be wary of potential confounders–did the methods of collecting data change over time? were citizens and police able to report as many?

This is surely not a causal relationship as it was not a RCT. This is a most association or correlation. During the COVID-19 pandemic, there was likely a change in *reported* crime rates. This was a complicated time and it is very difficult to state if there was less crime or if the crime was just less reported.

Question 3 (Faraway, Chapter 3 Problem 3)

The `cheddar` dataset (in the `faraway` package) contains data on a study of cheddar cheese from the LaTrobe Valley of Victoria, Australia (you might have used this dataset in the our last problem set). Thirty samples of cheddar cheese were analyzed for their content of acetic acid, hydrogen sulfide and lactic acid. Each sample was tasted and scored by a panel of judges and the average taste score produced. Use the `cheddar` dataset to answer the following:

(a) Fit a regression model with taste as the response and the three chemical contents as predictors. Identify the predictors that are statistically significant at the 5% level.

```
library(faraway)
library(dplyr)


data(cheddar)
str(cheddar)
```

```
## 'data.frame':    30 obs. of  4 variables:
##  $ taste : num  12.3 20.9 39 47.9 5.6 25.9 37.3 21.9 18.1 21 ...
##  $ Acetic: num  4.54 5.16 5.37 5.76 4.66 ...
##  $ H2S   : num  3.13 5.04 5.44 7.5 3.81 ...
##  $ Lactic: num  0.86 1.53 1.57 1.81 0.99 1.09 1.29 1.78 1.29 1.58 ...
```

```
summary(cheddar)
```

```
##      taste           Acetic           H2S             Lactic      
##  Min.   : 0.70   Min.   :4.477   Min.   : 2.996   Min.   :0.860  
##  1st Qu.:13.55   1st Qu.:5.237   1st Qu.: 3.978   1st Qu.:1.250  
##  Median :20.95   Median :5.425   Median : 5.329   Median :1.450  
##  Mean   :24.53   Mean   :5.498   Mean   : 5.942   Mean   :1.442  
##  3rd Qu.:36.70   3rd Qu.:5.883   3rd Qu.: 7.575   3rd Qu.:1.667  
##  Max.   :57.20   Max.   :6.458   Max.   :10.199   Max.   :2.010
```

```
cheddar_model <- lm(taste ~ Acetic + H2S + Lactic, data = cheddar)


summary(cheddar_model)
```

```
##
## Call:
## lm(formula = taste ~ Acetic + H2S + Lactic, data = cheddar)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.390  -6.612  -1.009   4.908  25.449
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -28.8768    19.7354  -1.463  0.15540
## Acetic        0.3277     4.4598   0.073  0.94198
## H2S           3.9118     1.2484   3.133  0.00425 **
## Lactic       19.6705     8.6291   2.280  0.03108 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.13 on 26 degrees of freedom
```

```
## Multiple R-squared:  0.6518, Adjusted R-squared:  0.6116
## F-statistic: 16.22 on 3 and 26 DF,  p-value: 3.81e-06
```

```
coef_summary <- summary(cheddar_model)$coefficients
significant_predictors <- coef_summary[coef_summary[,4] < 0.05, ]


print("Significant predictors at 5% level:")
```

```
## [1] "Significant predictors at 5% level:"
```

```
print(significant_predictors)
```

```
##         Estimate Std. Error  t value    Pr(>|t|)
## H2S     3.911841   1.248430 3.133408 0.004247081
## Lactic 19.670543   8.629055 2.279571 0.031079481
```

(b) `Acetic` and `H2S` are measured on a log scale. Fit a linear model where all three predictors are measured on their original scale. Identify the predictors that are statistically significant at the 5% level for this model.

```
cheddar$Acetic_orig <- exp(cheddar$Acetic)
cheddar$H2S_orig <- exp(cheddar$H2S)

# Fit new model with transformed variables
cheddar_model_orig <- lm(taste ~ Acetic_orig + H2S_orig + Lactic, data = cheddar)


summary(cheddar_model_orig)
```

```
##
## Call:
## lm(formula = taste ~ Acetic_orig + H2S_orig + Lactic, data = cheddar)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.209  -7.266  -1.651   7.385  26.335
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.897e+01  1.127e+01  -1.684   0.1042
## Acetic_orig  1.891e-02  1.562e-02   1.210   0.2371
## H2S_orig     7.668e-04  4.188e-04   1.831   0.0786 .
## Lactic       2.501e+01  9.062e+00   2.760   0.0105 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.19 on 26 degrees of freedom
## Multiple R-squared:  0.5754, Adjusted R-squared:  0.5264
## F-statistic: 11.75 on 3 and 26 DF,  p-value: 4.746e-05
```

```
coef_summary <- summary(cheddar_model_orig)$coefficients
significant_predictors <- coef_summary[coef_summary[,4] < 0.05, ]


print("Significant predictors at 5% level:")
```

```
## [1] "Significant predictors at 5% level:"
```

```r
print(significant_predictors)
```

```
##    Estimate  Std. Error     t value    Pr(>|t|)
## 25.00735787  9.06212141  2.75954787  0.01046242
```

(c) Can we use an F-test to compare these two models? Explain. Which model provides a better fit to the data? Explain your reasoning.

We CANNOT use an F-test to compare these models because they are non-nested models The models represent different functional forms of the relationship between predictors and response They are essentially different transformations of the predictor variables

(d) If `H2S` is increased 0.01 for the model used in part **(a)**, what change in the `taste` would be expected?

when H2S increases by 0.01 units on the log scale, we expect taste to increase by approximately 0.039 units

(e) What is the percentage change in `H2S` on the original scale corresponding to an additive increase of 0.01 on the (natural) log scale?

```r
# ratio
ratio <- exp(0.01)

# percentage change
percent_change <- (ratio - 1) * 100
print(percent_change)
```

```
## [1] 1.005017
```

an increase of 0.01 units on the natural log scale corresponds to approximately a 1.005% increase in the original scale.

Question 4 (Based on Faraway, Chapter 3 Problem 6)

The `happy` dataset (in the `faraway` package) contains data on happiness from a sample of 39 students collected in a University of Chicago MBA class (you might have used this dataset in our last problem set). The students were asked about happiness and how this related to their income and social life. Fit a model with `happy` as the dependent variable, and the other four variables as independent variables.

(a) Which predictors are statistically significant at the 1% level.

```
data(happy)
str(happy)
```

```
## 'data.frame':    39 obs. of  5 variables:
##  $ happy: num  10 8 8 8 4 9 8 6 5 4 ...
##  $ money: num  36 47 53 35 88 175 175 45 35 55 ...
##  $ sex  : num  0 1 0 1 1 1 1 0 1 1 ...
##  $ love : num  3 3 3 3 1 3 3 2 2 1 ...
##  $ work : num  4 1 5 3 2 4 4 3 2 4 ...
```

```
head(happy)
```

```
##   happy money sex love work
## 1    10    36   0    3    4
## 2     8    47   1    3    1
## 3     8    53   0    3    5
## 4     8    35   1    3    3
## 5     4    88   1    1    2
## 6     9   175   1    3    4
```

```
happy_model <- lm(happy ~ money + sex + love + work, data = happy)

# Get detailed summary statistics
summary_stats <- summary(happy_model)
print(summary_stats)
```

```
##
## Call:
## lm(formula = happy ~ money + sex + love + work, data = happy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.7186 -0.5779 -0.1172  0.6340  2.0651
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.072081   0.852543  -0.085   0.9331
## money        0.009578   0.005213   1.837   0.0749 .
## sex         -0.149008   0.418525  -0.356   0.7240
## love         1.919279   0.295451   6.496 1.97e-07 ***
## work         0.476079   0.199389   2.388   0.0227 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.058 on 34 degrees of freedom
## Multiple R-squared:  0.7102, Adjusted R-squared:  0.6761
## F-statistic: 20.83 on 4 and 34 DF,  p-value: 9.364e-09
```

```r
coef_pvalues <- summary_stats$coefficients[, "Pr(>|t|)"]
print("P-values for all predictors:")
```

```
## [1] "P-values for all predictors:"
```

```r
print(coef_pvalues)
```

```
##  (Intercept)        money          sex         love         work
## 9.331165e-01 7.491036e-02 7.240174e-01 1.968174e-07 2.265649e-02
```

```r
# Identify significant predictors at 1% level
sig_predictors <- coef_pvalues[coef_pvalues < 0.01]
print("\nPredictors significant at 1% level:")
```

```
## [1] "\nPredictors significant at 1% level:"
```

```r
print(names(sig_predictors))
```

```
## [1] "love"
```

(b) Implement a permutation procedure to test the significance of the `money` predictor. Do not use any existing packages here, write your own code instead.

```r
original_model <- lm(happy ~ money + sex + love + work, data = happy)
observed_t <- summary(original_model)$coefficients["money", "t value"]
print(paste("Observed t-statistic for money:", observed_t))
```

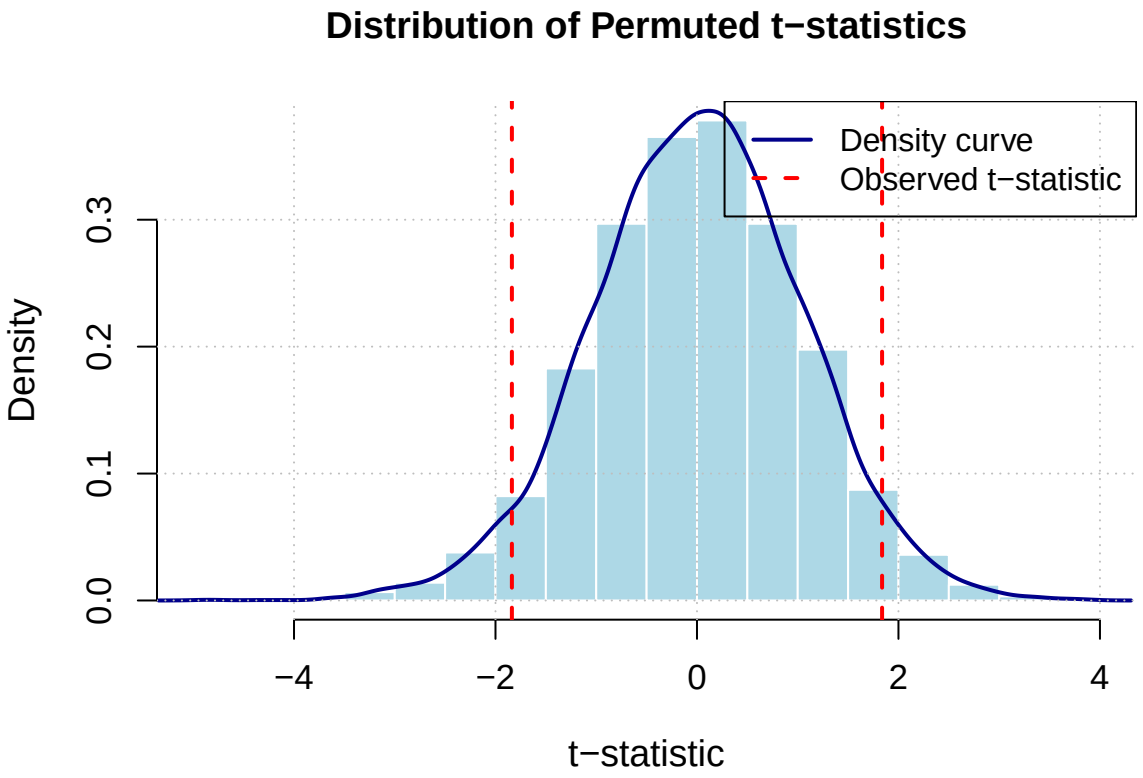
```
## [1] "Observed t-statistic for money: 1.83735710968689"
```

```r
perm_test <- function(n_perm = 10000) {
  # Store t-statistics
  t_stats <- numeric(n_perm)

  for(i in 1:n_perm) {
    perm_data <- happy
    perm_data$money <- sample(happy$money, replace = FALSE)

    perm_model <- lm(happy ~ money + sex + love + work, data = perm_data)

    t_stats[i] <- summary(perm_model)$coefficients["money", "t value"]
  }

  p_value <- mean(abs(t_stats) >= abs(observed_t))

  return(list(
    p_value = p_value,
    t_stats = t_stats,
    observed_t = observed_t
  ))
}

set.seed(139)
results <- perm_test(10000)

print(paste("Permutation test p-value:", results$p_value))
```

```
## [1] "Permutation test p-value: 0.0802"
```

(c) Create a histogram of the permutation t-statistics. Make sure to use the probability rather than frequency version of the histogram.

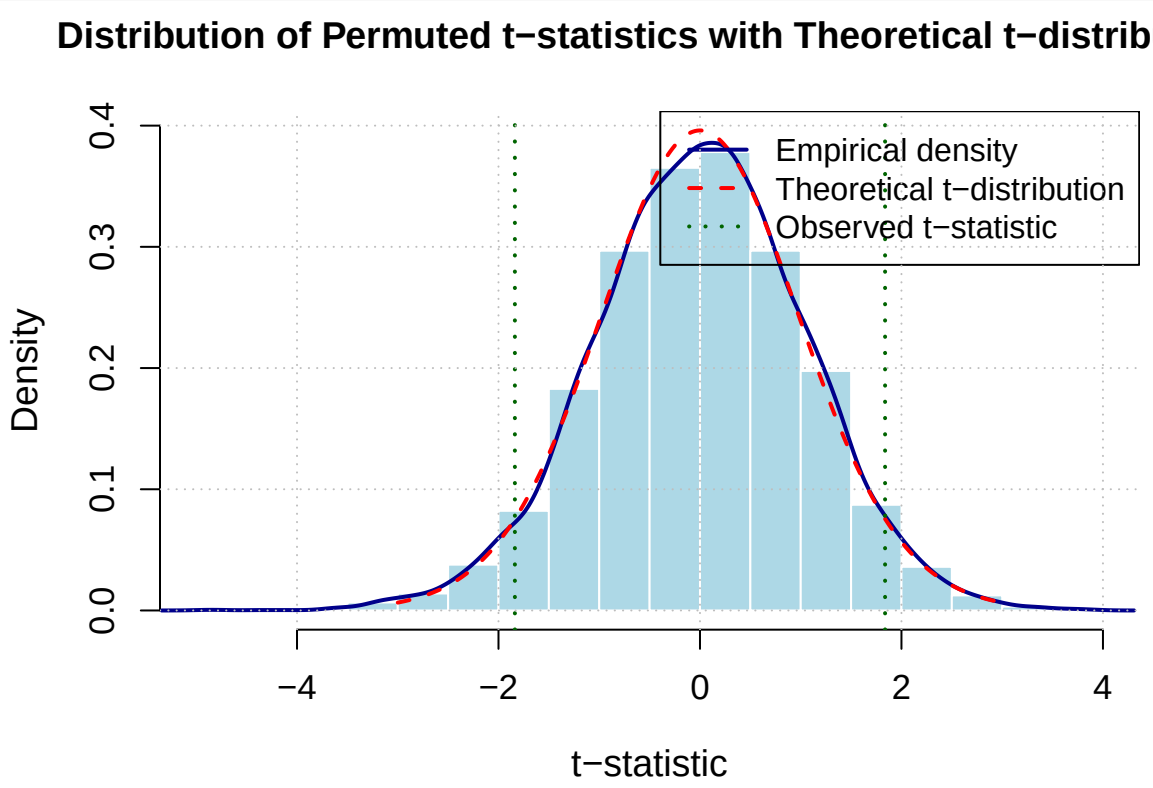
```r
par(mar = c(5, 5, 4, 2))
hist(results$t_stats,
     breaks = 30,
     probability = TRUE,
     main = "Distribution of Permuted t-statistics",
     xlab = "t-statistic",
     ylab = "Density",
     col = "lightblue",
     border = "white",
     cex.lab = 1.2,
     cex.axis = 1.1)


lines(density(results$t_stats), col = "darkblue", lwd = 2)


abline(v = results$observed_t, col = "red", lwd = 2, lty = 2)
abline(v = -results$observed_t, col = "red", lwd = 2, lty = 2)


legend("topright",
       legend = c("Density curve", "Observed t-statistic"),
       col = c("darkblue", "red"),
       lwd = c(2, 2),
       lty = c(1, 2))


grid(nx = NULL, ny = NULL, col = "gray", lty = "dotted")
```



(d) Overlay an appropriate t-distribution over your histogram. Hint: Use `grid <- seq(-3, 3, length = 300)` to create a grid of values, then use the `dt` function to compute the t-density over this grid and the `lines` function to superimpose the result.

```r
df <- nrow(happy) - 5

grid <- seq(-3, 3, length = 300)

par(mar = c(5, 5, 4, 2))


hist(results$t_stats,
     breaks = 30,
     probability = TRUE,
     main = "Distribution of Permuted t-statistics with Theoretical t-distribution",
     xlab = "t-statistic",
     ylab = "Density",
     col = "lightblue",
     border = "white",
     cex.lab = 1.2,
     cex.axis = 1.1,
     ylim = c(0, max(dt(0, df), max(density(results$t_stats)$y))))

lines(density(results$t_stats), col = "darkblue", lwd = 2)

lines(grid, dt(grid, df), col = "red", lwd = 2, lty = 2)


abline(v = results$observed_t, col = "darkgreen", lwd = 2, lty = 3)
abline(v = -results$observed_t, col = "darkgreen", lwd = 2, lty = 3)
```

```r
legend("topright",
       legend = c("Empirical density",
                  "Theoretical t-distribution",
                  "Observed t-statistic"),
       col = c("darkblue", "red", "darkgreen"),
       lwd = c(2, 2, 2),
       lty = c(1, 2, 3))

grid(nx = NULL, ny = NULL, col = "gray", lty = "dotted")
```



(e) Implement a bootstrap procedure to compute 90% and 95% confidence intervals for β_{money}. Do not use any existing packages here, write your own code instead. Does zero fall within these confidence intervals? Are these results consistent with the previous tests?

```r
bootstrap_beta <- function(data, n_boot = 10000) {

  beta_money <- numeric(n_boot)


  n <- nrow(data)

  for(i in 1:n_boot) {

    boot_indices <- sample(1:n, size = n, replace = TRUE)
    boot_data <- data[boot_indices, ]


    boot_model <- lm(happy ~ money + sex + love + work, data = boot_data)
```

```
    beta_money[i] <- coef(boot_model)["money"]
  }


  ci_90 <- quantile(beta_money, c(0.05, 0.95))
  ci_95 <- quantile(beta_money, c(0.025, 0.975))

  return(list(
    beta_money = beta_money,
    ci_90 = ci_90,
    ci_95 = ci_95
  ))
}


boot_results <- bootstrap_beta(happy)


original_model <- lm(happy ~ money + sex + love + work, data = happy)
original_beta <- coef(original_model)["money"]


print("Original beta coefficient for money:")
```

```
## [1] "Original beta coefficient for money:"
```

```
print(original_beta)
```

```
##       money
## 0.009578074
```

```
print("\n90% Confidence Interval:")
```

```
## [1] "\n90% Confidence Interval:"
```

```
print(boot_results$ci_90)
```

```
##          5%         95%
## 0.001783694 0.021910148
```

```
print("\n95% Confidence Interval:")
```

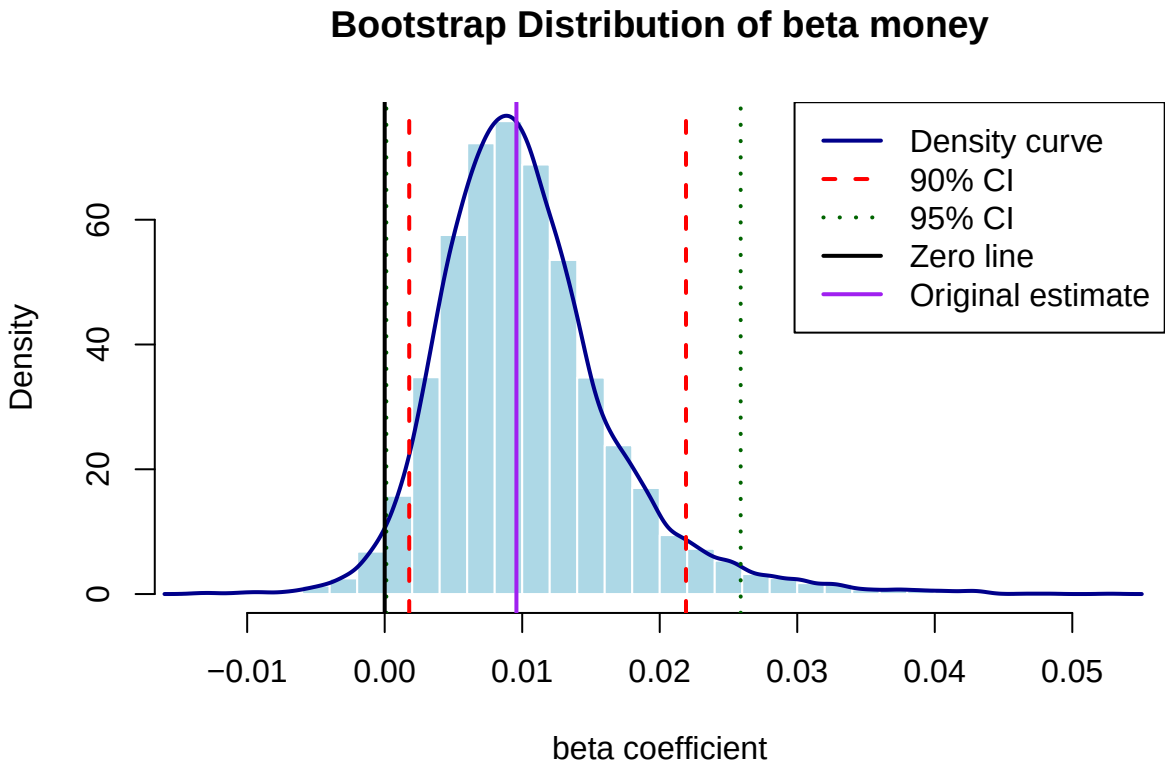
```
## [1] "\n95% Confidence Interval:"
```

```
print(boot_results$ci_95)
```

```
##         2.5%        97.5%
## 0.0001189152 0.0258870804
```

```
hist(boot_results$beta_money,
     breaks = 30,
     probability = TRUE,
     main = "Bootstrap Distribution of beta money",
     xlab = "beta coefficient",
     ylab = "Density",
     col = "lightblue",
     border = "white")
```

```r
lines(density(boot_results$beta_money), col = "darkblue", lwd = 2)


abline(v = boot_results$ci_90, col = "red", lwd = 2, lty = 2)
abline(v = boot_results$ci_95, col = "darkgreen", lwd = 2, lty = 3)
abline(v = 0, col = "black", lwd = 2, lty = 1)  # Add line at zero
abline(v = original_beta, col = "purple", lwd = 2)


legend("topright",
       legend = c("Density curve",
                  "90% CI",
                  "95% CI",
                  "Zero line",
                  "Original estimate"),
       col = c("darkblue", "red", "darkgreen", "black", "purple"),
       lwd = 2,
       lty = c(1, 2, 3, 1, 1))
```



the intervals do not contain 0, so the coefficients are significant at the 95% level. this is consistent with our previous findings with the permutation test. we should reject the null hypothesis for