STAT 443: Homework 1

Aronn Grant Laurel (21232475)

30 January, 2025

Question 1

(a)

```
# this is where your R code goes
employee <- read_csv("employee_wages_total_industry.csv")
```

```
## Rows: 335 Columns: 1
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (1): employee_wages
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# head(employee)

employee_ts <- ts(employee$employee_wages,
                  start = c(1997, 1),
                  frequency = 12)

plot(employee_ts,
     main = "Employee Wages Time Series (Jan 1997 - Nov 2024)",
     ylab = "Wages (CAD)",
     xlab = "Year")
```

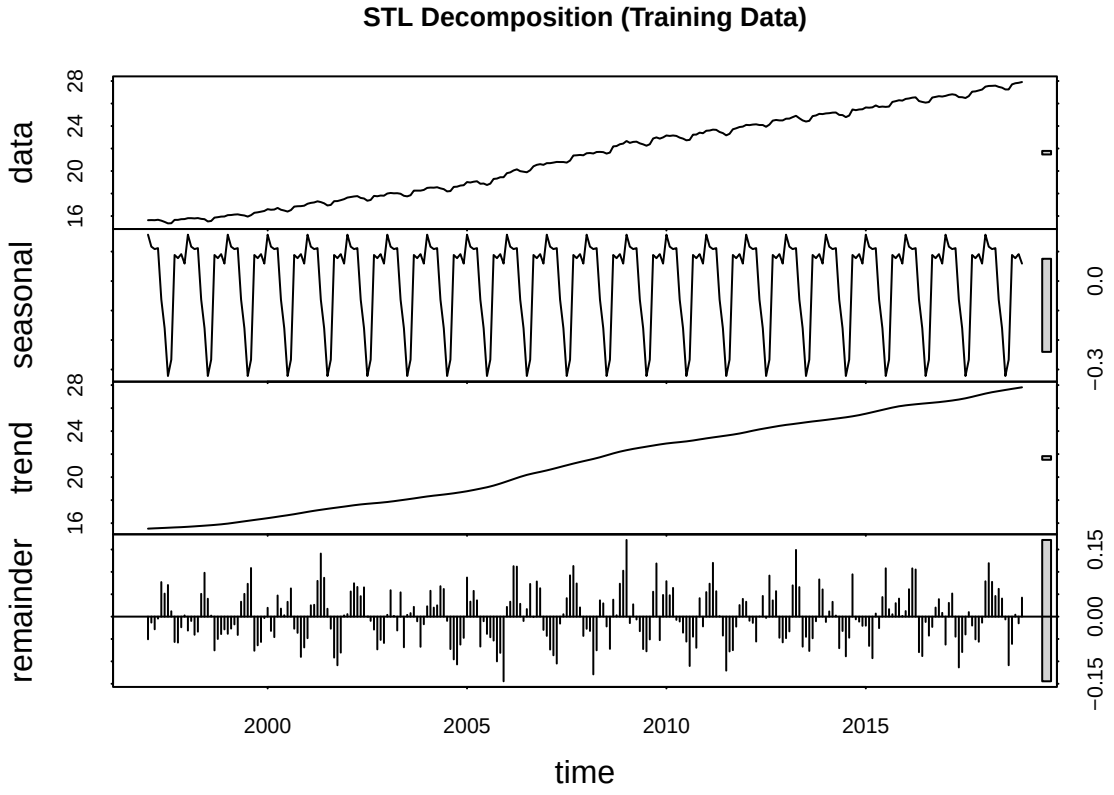
Trend: We can see an steady increase and positive trend in average Wage (CAD) over the years

Seasonality: We can observe seasonal fluctuation every year and a larger fluctuations around after 2020. I believe an Additive model would be more appropriate for our data because we also can observe a fixed variation between seasons.

Stationary: Since we observe an increasing average wage over time and we see a slight variation in the series over time as well which could indicate that it is not stationary.

(b)

```r
# this is where your R code goes
employee_train <- window(employee_ts,
                         start = c(1997, 1),
                         end = c(2018, 12))

employee_test <- window(employee_ts,
                        start = c(2019, 1),
                        end = c(2020, 12))

employee_decomp <- stl(employee_train, s.window = "periodic")

plot(employee_decomp,
     main = "STL Decomposition (Training Data)")
```



(c)

```
trend <- employee_decomp$time.series[, "trend"]
time <- time(trend)

# Fitting Linear Model
trend_lm <- lm(trend ~ time)
summary(trend_lm)
```

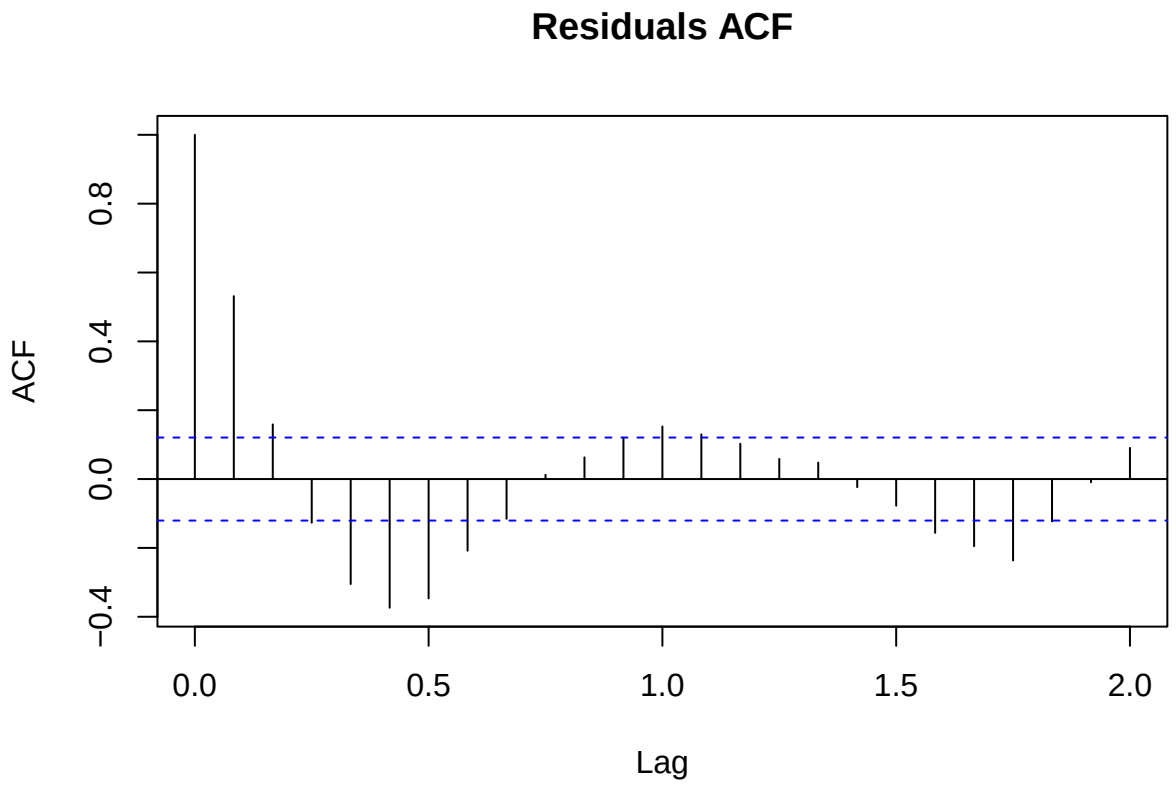
```
##
## Call:
## lm(formula = trend ~ time)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.70034 -0.15901  0.00373  0.21533  0.89784
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.196e+03  6.432e+00  -186.0   <2e-16 ***
## time         6.063e-01  3.203e-03   189.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3305 on 262 degrees of freedom
```

```
## Multiple R-squared:  0.9927, Adjusted R-squared:  0.9927
## F-statistic: 3.582e+04 on 1 and 262 DF,  p-value: < 2.2e-16
```

The fitted model we get: Wage = 13.95 + (0.05655 * time) Since we observe a P-value 2 * 10^(-16) which is much smaller than a small significance level such at 5%, we can say that there is strong evidence that there is a trend at 95%. Intuitively, I would not use this trend component to make predictions because the linear model cannot accurately fit the seasonality as the plot displayed in Q1a).

(d)

```
# this is where your R code goes
error <- employee_decomp$time.series[, "remainder"]
acf(error, main = "Residuals ACF")
```



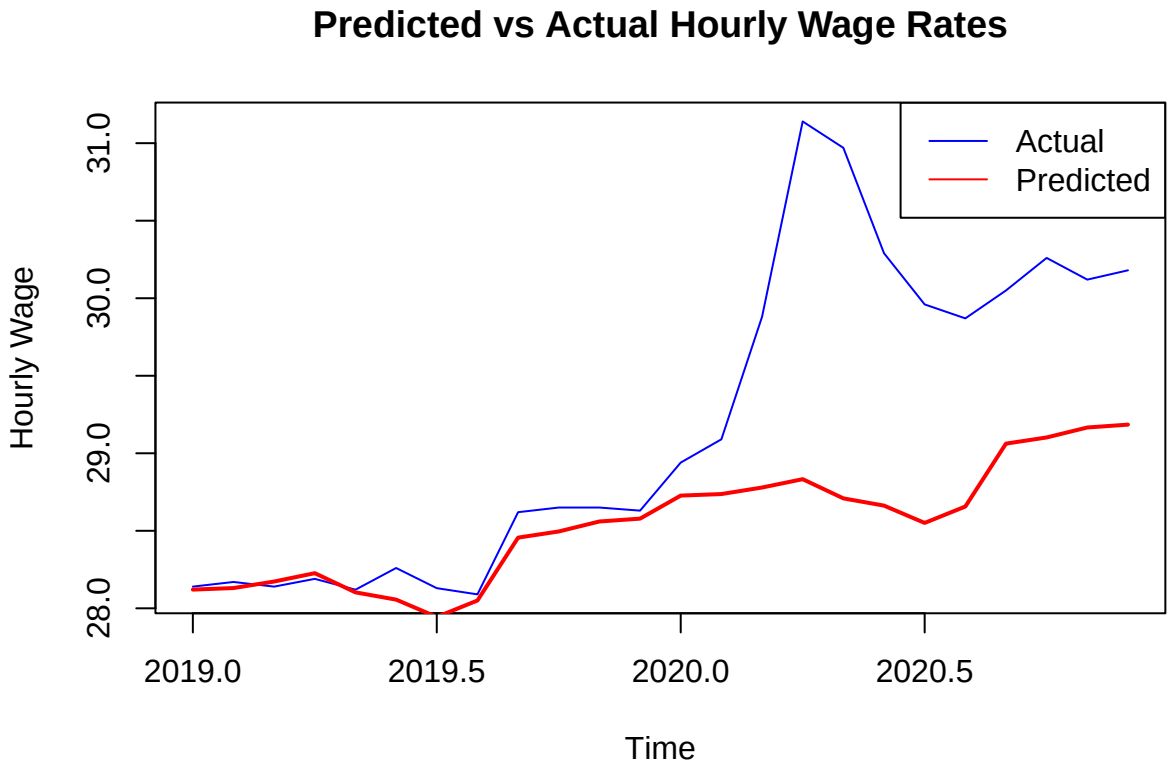
For white noise, we should expect each autocorrelation to be close to 0. Since we see lags that are outside the boundary line and away from 0, I believe it is inconsistent with a white noise process.

(e)

- the period from January 2019 to December 2020 *

```
# this is where your R code goes

# Predict
prediction <- predict(trend_lm, newdata = data.frame(time = time(employee_test)))
```

```r
seasonal <- employee_decomp$time.series[, "seasonal"]
# Get the last year
last_seasonal <- seasonal[(length(seasonal) - 11) : length(seasonal)]
seasonal_predict <- rep(last_seasonal, length.out = length(employee_test))


# Time Series Object
predicted_values <- ts(prediction + seasonal_predict,
                       start = start(employee_test),
                       end = end(employee_test),
                       frequency = frequency(employee_test))

plot(employee_test,
     col = "blue",
     xlab = "Time",
     ylab = "Hourly Wage",
     main = "Predicted vs Actual Hourly Wage Rates")
lines(predicted_values, col = "red", lwd=2)

# Label
legend("topright",
       legend = c("Actual", "Predicted"),
       col = c("blue", "red"),
       lty = 1)
```



Q: Comment on the performance of your prediction method, explain why or why not the method worked

well for this data, and summarize what you learned from it.

In Comparison with the actual test data values, we see a consistent / fair prediction until 2020 where we end up seeing a significant difference between the actual and predicted result after. The difference could be caused by 'special' events during that period which makes its data inconsistent with the historical trend. Therefore, I believe that more information is required in order for the model to work well with the data.

Q: As a statistician, what other information would you like to add to your forecasts in addition to the point forecasts you produced above?

I would implement Residual analysis where we analyze our residual (actual - prediction) to check for how goodness of fit for our model. Furthermore, I would also apply prediction intervals (~ 95% Confidence interval) to account for uncertainty when predicting new data.

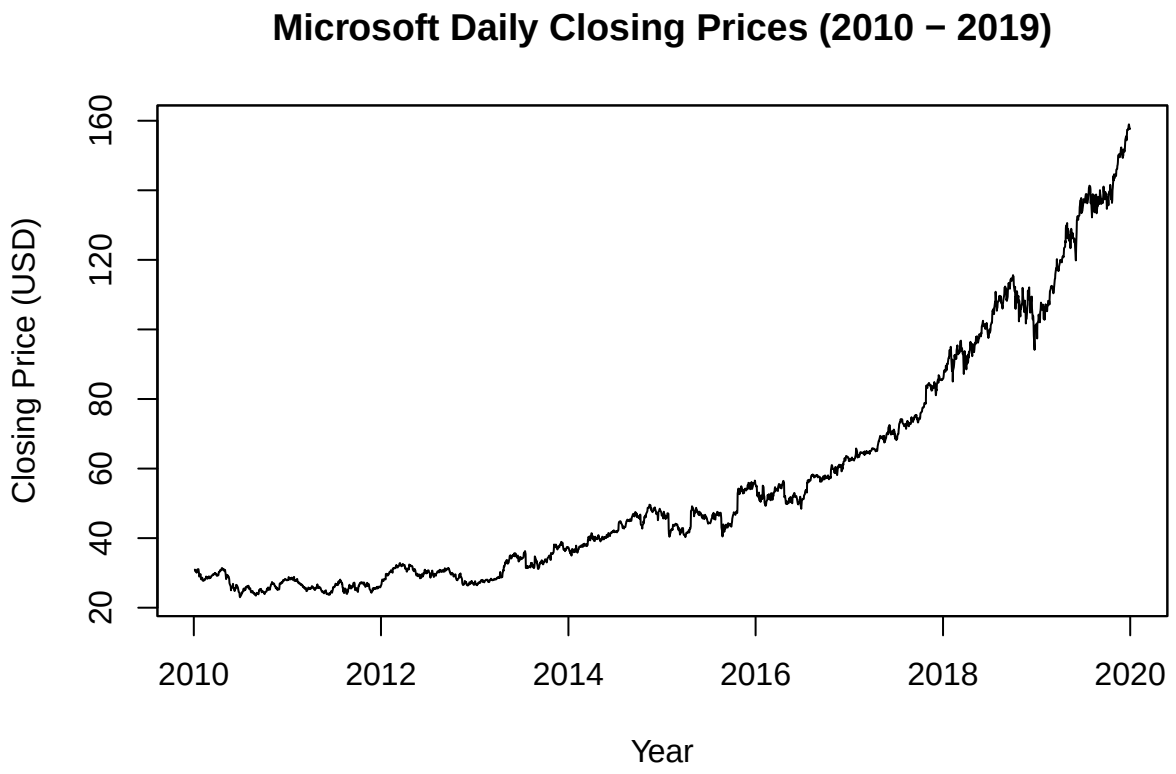
Question 2

(a)

```
# this is where your R code goes
MSFT <- read_csv("MSFT_closing_price.csv")
```

```
## Rows: 2515 Columns: 2
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl  (1): Closing_Price
## date (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
MSFT_zoo <- zoo(MSFT$Closing_Price, as.Date(MSFT$Date))

plot(MSFT_zoo,
     main = "Microsoft Daily Closing Prices (2010 - 2019)",
     xlab = "Year",
     ylab = "Closing Price (USD)")
```

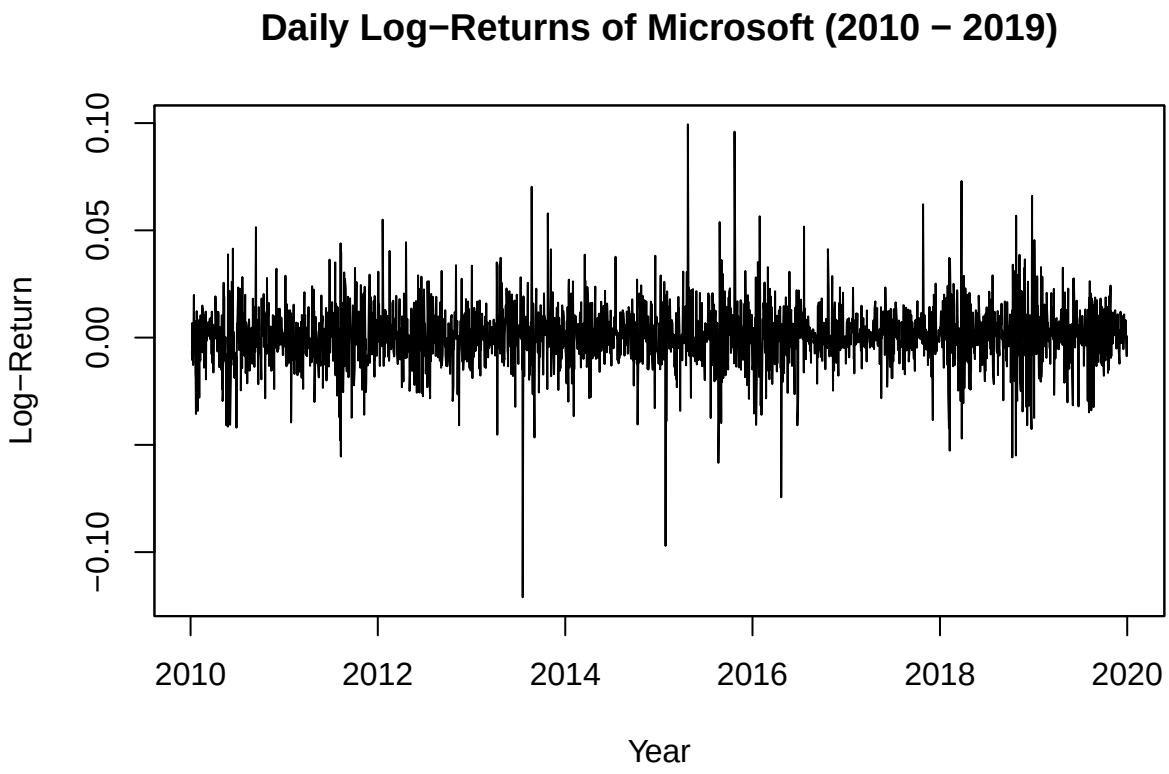


Trend: We see an increasing and positive trend from the above graph, with a sharper increase starting 2016.

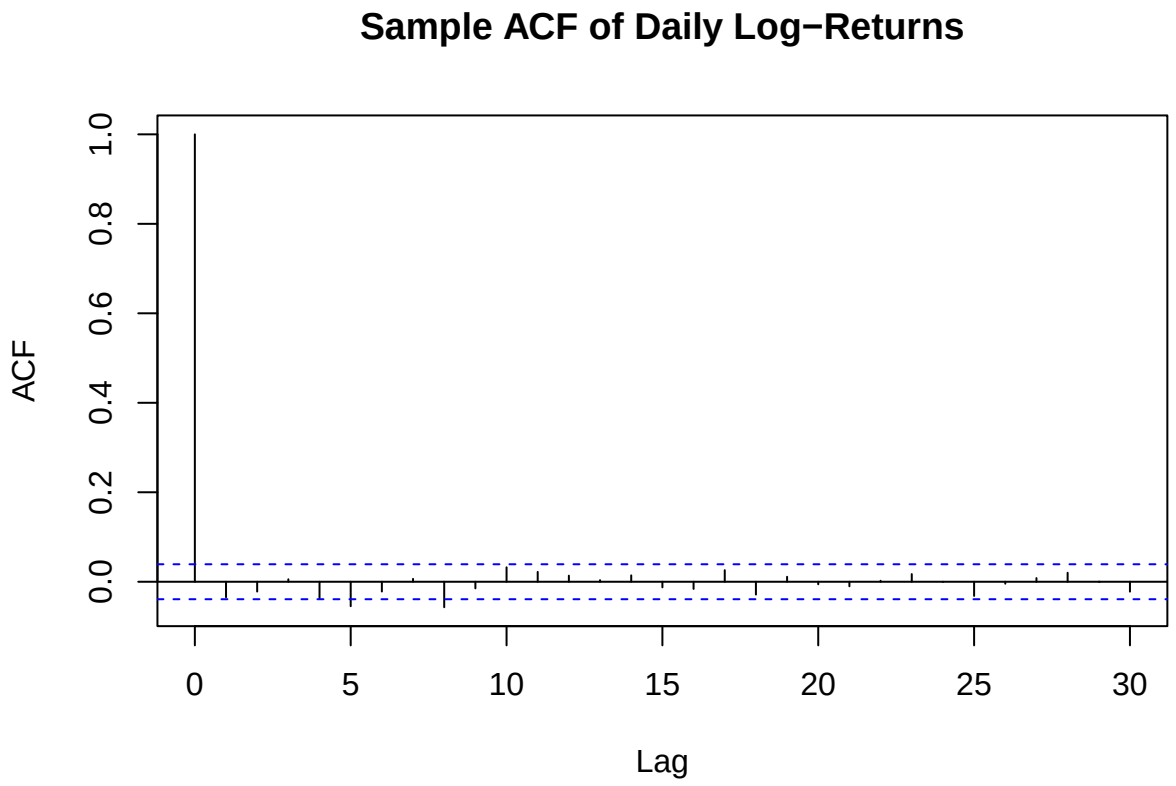
Seasonality: As there are many fluctuations in the season, it is difficult to determine if there is seasonality from the plot above.

(b)

```
# daily log-returns
log_returns <- diff(log(MSFT_zoo))

plot( log_returns,
      main = "Daily Log-Returns of Microsoft (2010 - 2019)",
      xlab = "Year",
      ylab = "Log-Return")
```



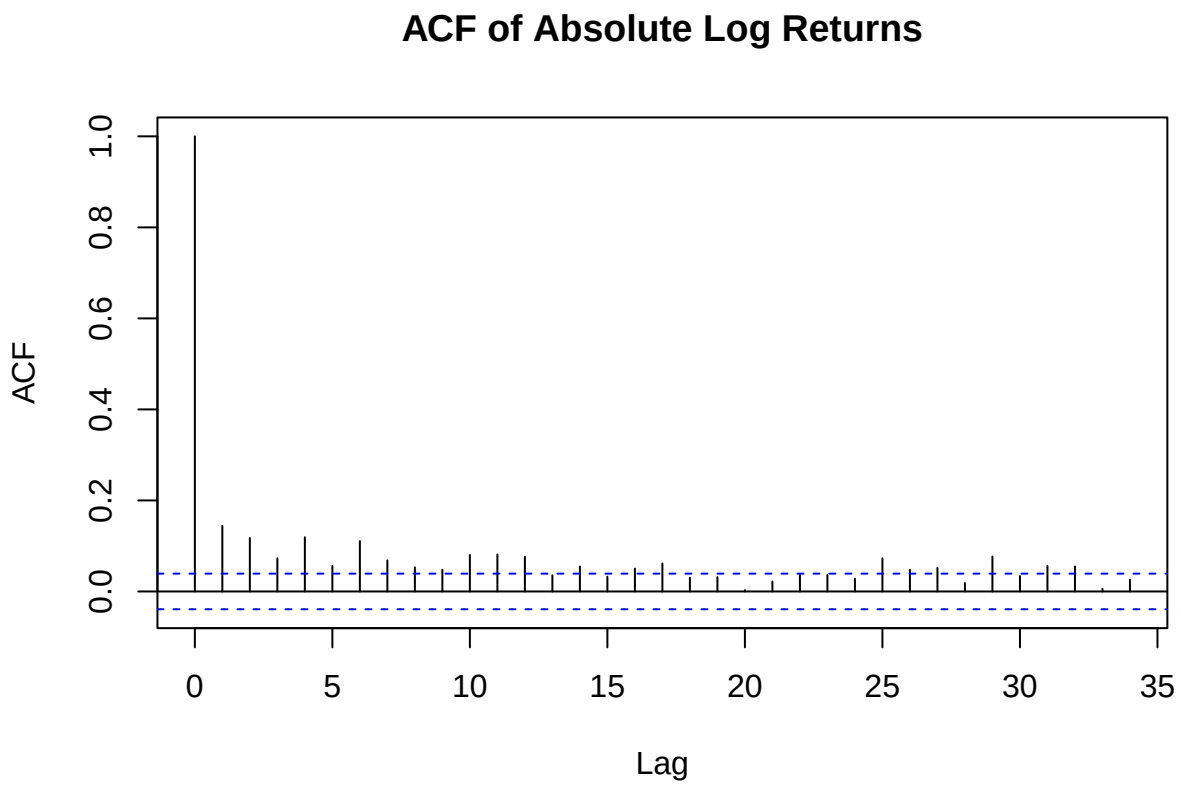

```
acf(coredata(log_returns), # Extract core data (no date)
    main = "Sample ACF of Daily Log-Returns",
    lag.max = 30)
```



Aside from Lag 0, we can see that all the lags from 1-30 are close to 0 (within the significant bound), thus suggesting that our daily log returns may be a white noise process.

(c)

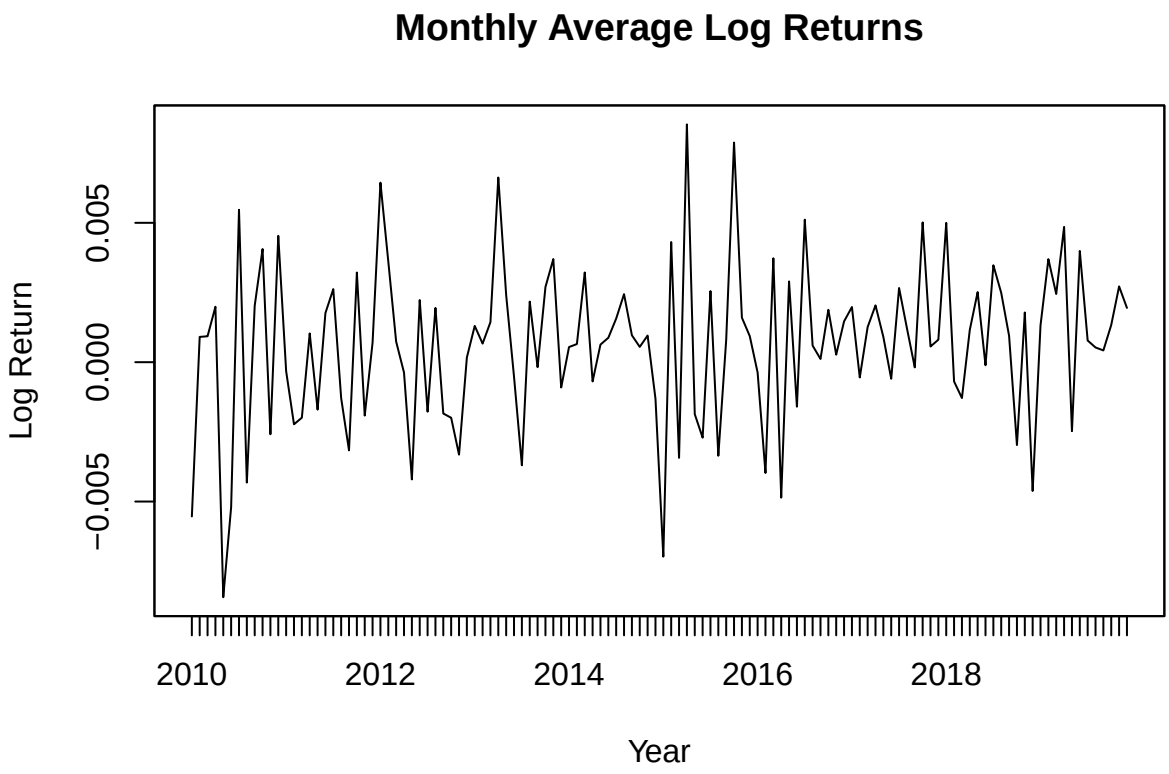
```
# this is where your R code goes
abs_log_returns <- abs(log_returns)

acf(coredata(abs_log_returns),
   main = "ACF of Absolute Log Returns")
```



In contrast to question 2b), we end up seeing the ACF values for our daily log returns at absolute values with more significant values. The ACF values that goes beyond the boundary line suggests that there exists serial dependence in the series. Hence, we cannot consider this as a white noise process.

(d)

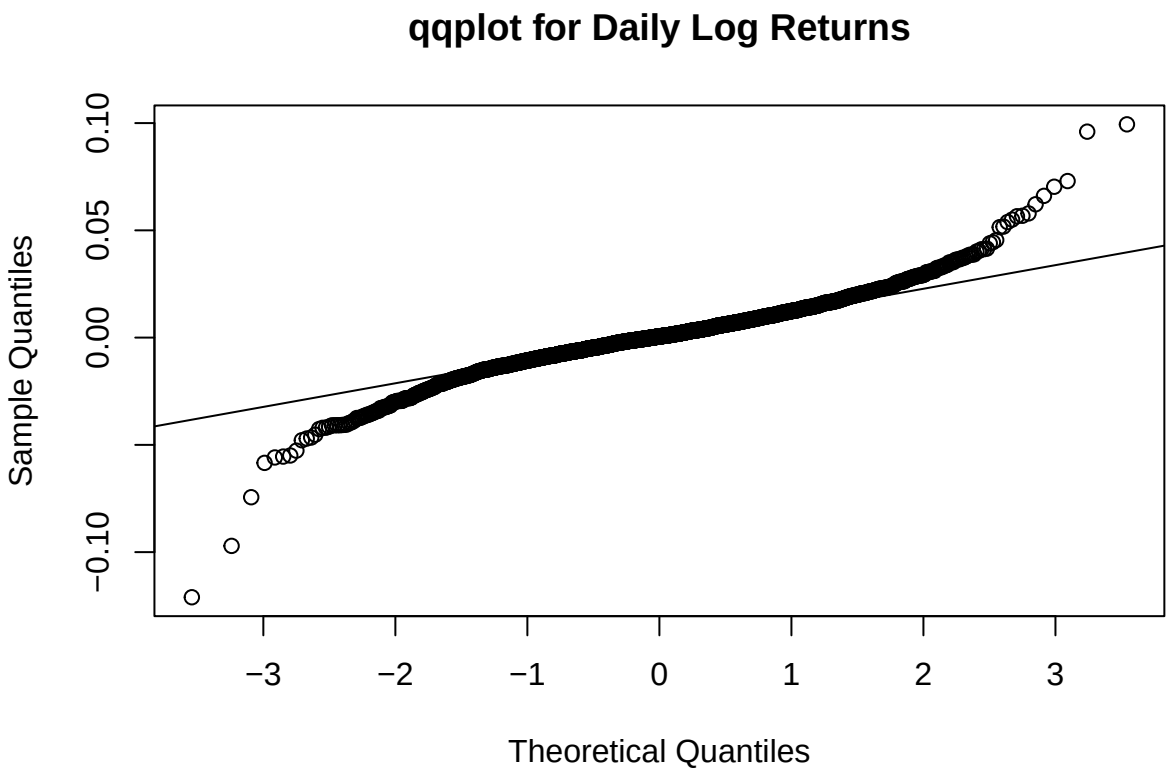
```r
# this is where your R code goes
monthly_returns <- aggregate(log_returns,
                             as.yearmon,
                             FUN = mean)

plot(monthly_returns,
     main = "Monthly Average Log Returns",
     xlab = "Year",
     ylab = "Log Return")
```



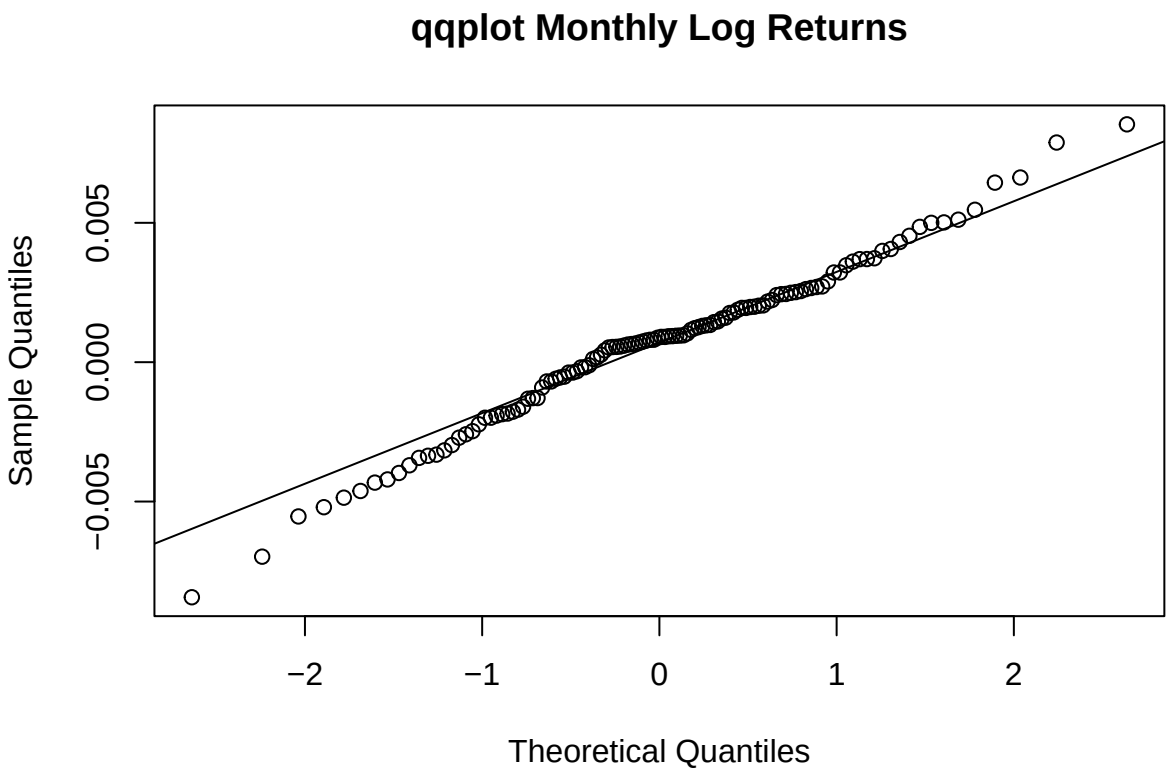
By aggregating our value, we smoothened the fluctuations which would exhibit a weaken overall serial dependence compared to the daily return series.

(e)

```
# this is where your R code goes
qqnorm(log_returns, main = "qqplot for Daily Log Returns")
qqline(log_returns)
```




```
qqnorm(monthly_returns, main = "qqplot Monthly Log Returns")
qqline(monthly_returns)
```



The qqplot for daily log returns exhibits a left heavy-tailed behavior while our monthly mean returns is closer to a straight line, suggesting that it can be approximated well with a normal distribution in comparison to our qqplot for the daily log returns.

Question 3

a) Compute the mean and variance of r1 and r2 values from your simulation study

```
set.seed(123)

# Part i)
n <- 1000
m <- 10000
z <- matrix(rnorm(n * m), nrow = n, ncol = m)

# Part ii)
r1 <- numeric(m)
r2 <- numeric(m)


for (i in 1:m) {
  r1[i] <- acf(z[, i], lag.max = 1, plot = FALSE)$acf[2]
  r2[i] <- acf(z[, i], lag.max = 2, plot = FALSE)$acf[3]
}
```

```r
# this is where your R code goes
mean_r1 <- mean(r1)
mean_r1
```

```
## [1] -0.001014699
```

```r
mean_r2 <- mean(r2)
mean_r2
```

```
## [1] -0.0009236813
```

```r
var_r1 <- var(r1)
var_r1
```

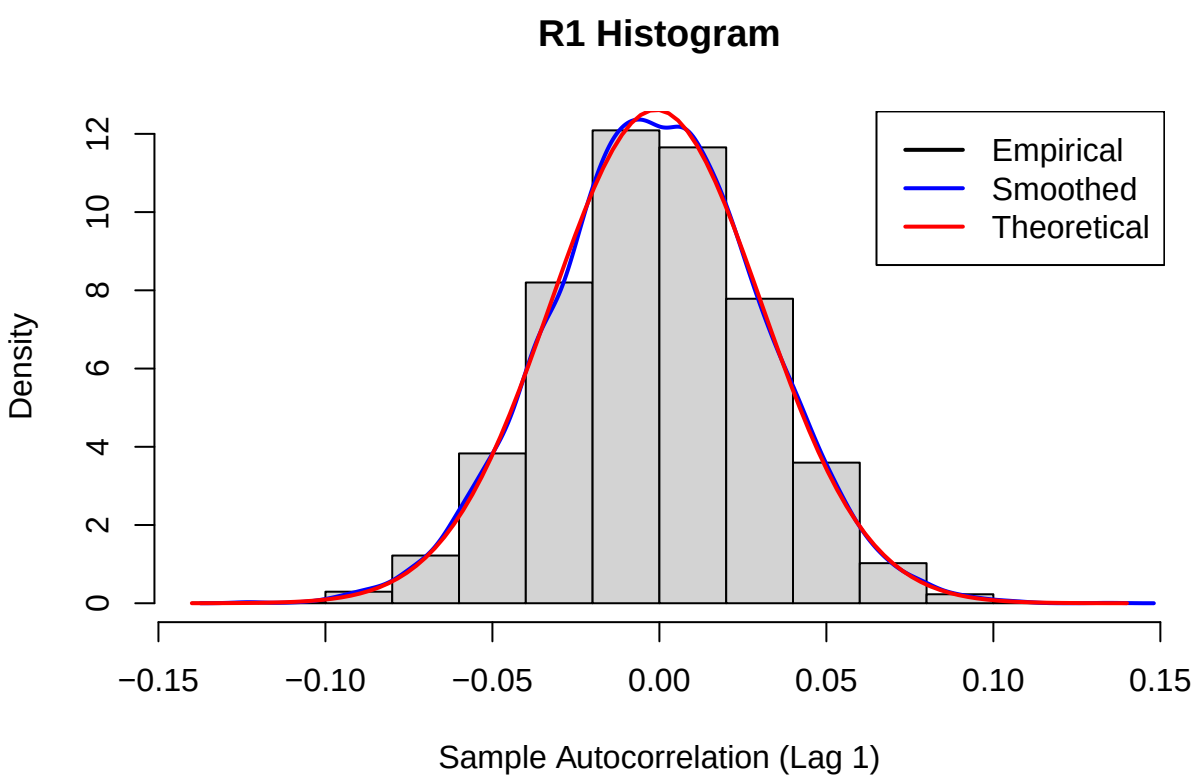
```
## [1] 0.001003194
```

```r
var_r2 <- var(r2)
var_r2
```

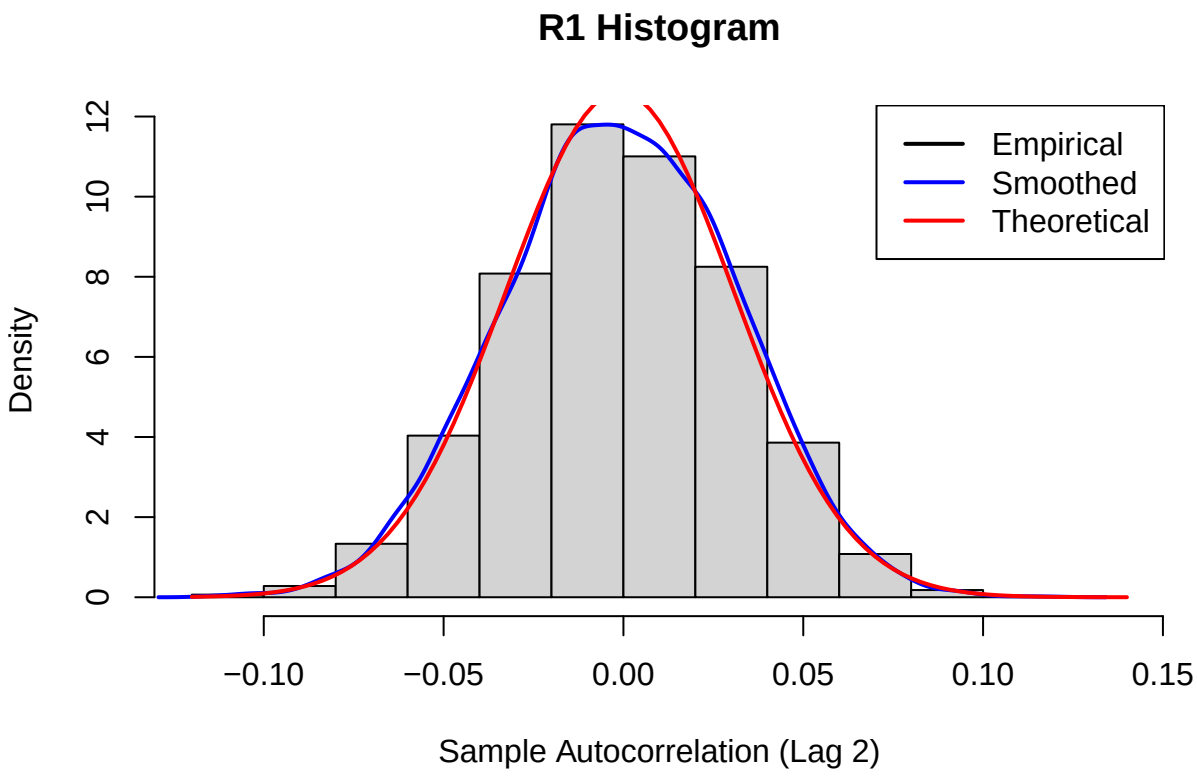
```
## [1] 0.001029984
```

b) In two separate figures, plot the two histograms for the sample of r1 and r2 values from the simulation study (function hist()), add the smoothed version of the histogram (function density()) and the theoretical asymptotic normal density (function dnorm()). Make sure your plots are well-presented, including a suitable title, axes labels, curves of different type or colour, and a legend.

```r
# this is where your R code goes

# Plotting r1 histogram
hist(r1,
     freq = FALSE,
     main = "R1 Histogram",
     xlab = "Sample Autocorrelation (Lag 1)")

# empirical density estimate
lines(density(r1), col = "blue", lwd = 2)

# theoretical normal curve
curve(dnorm(x, mean = -1/n, sd = sqrt(1/n)), add = TRUE, col = "red", lwd = 2)

legend("topright",
       legend = c("Empirical", "Smoothed", "Theoretical"),
       col = c("black", "blue", "red"),
       lty = 1,
       lwd = 2)
```




```r
# this is where your R code goes

# Plotting r2 histogram
hist(r2,
     freq = FALSE,
     main = "R1 Histogram",
     xlab = "Sample Autocorrelation (Lag 2)")

# empirical density estimate
lines(density(r2), col = "blue", lwd = 2)

# theoretical normal curve
curve(dnorm(x, mean = -1/n, sd = sqrt(1/n)), add = TRUE, col = "red", lwd = 2)

legend("topright",
       legend = c("Empirical", "Smoothed", "Theoretical"),
       col = c("black", "blue", "red"),
       lty = 1,
       lwd = 2)
```



c) Comment whether there is an agreement between the empirical estimates of the bias, variance and sampling density of the estimator of the autocorrelation at lag h and their theoretical approximation.

We can observe an agreement of values for r1 and r2 to their corresponding theoretical approximation which is approximately normal.

Looking at our calculations, we see our empirical r1 and r2 yielding variances approximately 0.001 which is very close to our theoretical approximation 1/1000. Similarly to their Mean, we see our empirical mean at around -0.001 which is close to -1/1000 where our n = 1000. similar