ECOM90004 Assignment 3

Josh Copeland (SID: 1444772)

2025-10-22

```
#Packages used:

# library(tidyverse)
# library(lubridate)
# library(forecast)
# library(urca)
#library(rugarch)
```

```
data <- read_csv("Google.csv", show_col_type = FALSE) %>%
  select(date = Date, price = Google) %>%
  mutate(price_log = log(price)) %>%
  mutate(returns = price_log - lag(price_log)) %>%
  na.omit()

data_est <- data %>% slice(1:(n() - 10)) # Estimation sample

data_fcst <- data %>% slice((n() - 9):n()) # Testing sample
```

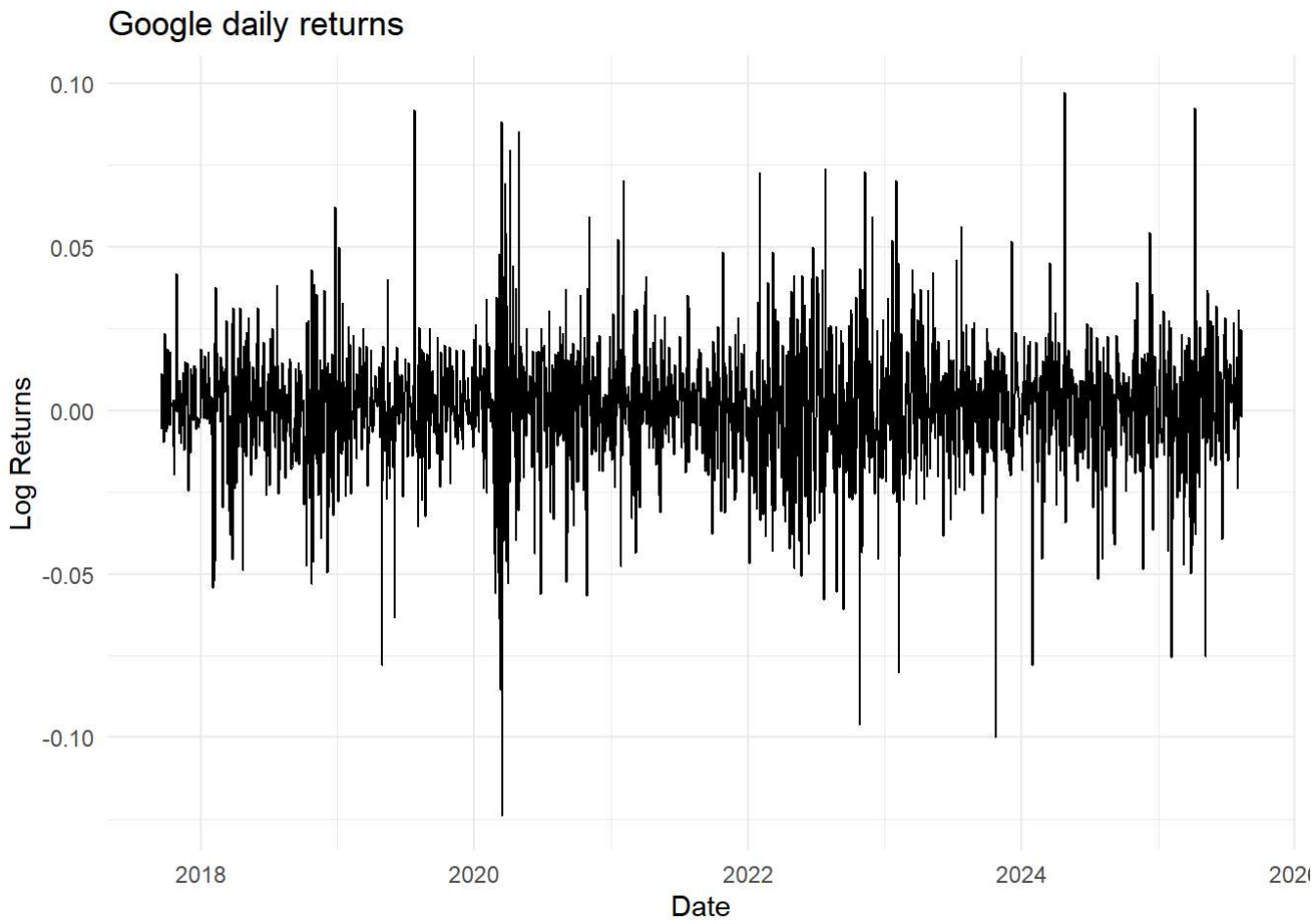
Question 1

The analysis below indicates:

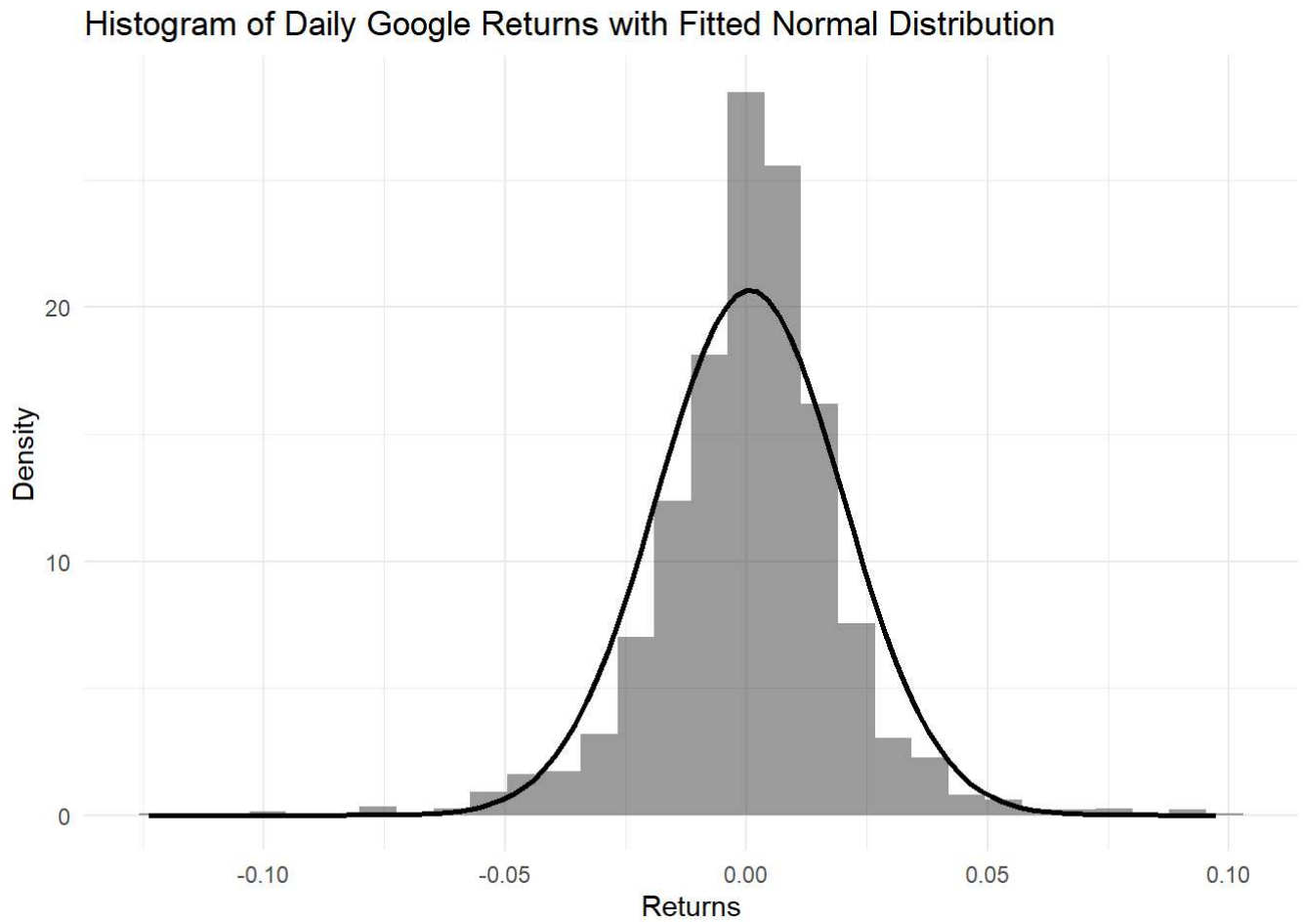
- Although the time series of daily Google returns is mean-reverting, it is unlikely to satsify the definition of a stationary time series given clear patterns of conditional (non-constant) variance.
- The distribution of daily Google Returns looks leptokurtic: over-concentrated at the mean and excess mass in the tails relative to the fitted normal distribution.
- With respect to the descriptive statistics we generate:
    - The mean is approximately zero (0.007) and the standard deviation is 0.0193, indicating an average dialy volatility of roughtly 1.9%. They also shows the distribution is slightly left-skewed (-0.241).
    - This conclusion about leptokurtosis is reinforced by the descriptive statistics we generate, specifically by the Kurtosis value of 7.04, significantly above the value of 3 in a normal distribution.

These conclusions suggest it would not be sufficient to model this time series using an ARMA model given its homoskedasticity assumption, and hence an ARMA-(G)ARCH approach should be investiagted instead.

```
ggplot(data_est, aes(x = date, y = returns)) + geom_line() + labs(title = "Google daily retur
ns",x = "Date", y = "Log Returns")
```




```
ggplot(data_est, aes(returns)) +
  geom_histogram(aes(y = after_stat(density)), bins = 30, alpha = .6) +
  stat_function(fun = dnorm, args = with(data_est, list(mean = mean(returns, na.rm=TRUE), sd
= sd(returns, na.rm=TRUE))), linewidth = 1) +
  labs(title = "Histogram of Daily Google Returns with Fitted Normal Distribution", x = "Retu
rns", y = "Density")
```




```
stats <- data_est %>% summarise(
  Mean = mean(returns, na.rm = TRUE),
  SD = sd(returns, na.rm = TRUE),
  Skewness = mean((returns - mean(returns, na.rm=TRUE))^3, na.rm=TRUE) / sd(returns, na.rm=TR
UE)^3,
  Kurtosis = mean((returns - mean(returns, na.rm=TRUE))^4, na.rm=TRUE) / sd(returns, na.rm=TR
UE)^4
) %>%  print()
```

```
## # A tibble: 1 × 4
##       Mean     SD Skewness Kurtosis
##      <dbl>  <dbl>    <dbl>    <dbl>
## 1 0.000735 0.0193   -0.241     7.04
```

Question 2

The `model_dow` object below create a dummy variable for each day of the week, with coefficients interpreted as deviations from the intercept (Friday). It shows there is no “day of the week” or “calender” effect on Google’s returns because:

- The F-test’s p-value is very large (0.8027), and therefore we cannot reject the null hypothesis that all weekday coefficients are jointly equal to zero.
- All of the inidividual coefficient p-values are very large, meaning their null hypothesises of being equal to zero cannot be rejected at any reasonable significance level.

```
data_est <- data_est %>%
  mutate(weekday = wday(date, label = TRUE, abbr = TRUE))

model_dow <- lm(returns ~ weekday, data = data_est, contrasts = list(weekday = "contr.treatme
nt"))
summary(model_dow)
```

```
##
## Call:
## lm(formula = returns ~ weekday, data = data_est, contrasts = list(weekday = "contr.treatme
nt"))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.123491 -0.009392  0.000645  0.010211  0.096818
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0001940  0.0010003  -0.194    0.846
## weekdayTue   0.0011716  0.0013823   0.848    0.397
## weekdayWed   0.0017079  0.0013855   1.233    0.218
## weekdayThu   0.0009591  0.0013897   0.690    0.490
## weekdayFri   0.0007237  0.0013905   0.520    0.603
##
## Residual standard error: 0.01929 on 1980 degrees of freedom
## Multiple R-squared:  0.0008243,  Adjusted R-squared:  -0.001194
## F-statistic: 0.4084 on 4 and 1980 DF,  p-value: 0.8027
```

Question 3

(a)

An AR(9) model is selected because it minimised AICc among the candidate specifications (testing up to an AR(15) model), which indicates the best balance between model fit and parsimony. Additionally, the Ljung-Box test p-value (0.9647) is large, suggesting there is no remaining autocorrelation in the residuals, supporting this model further.

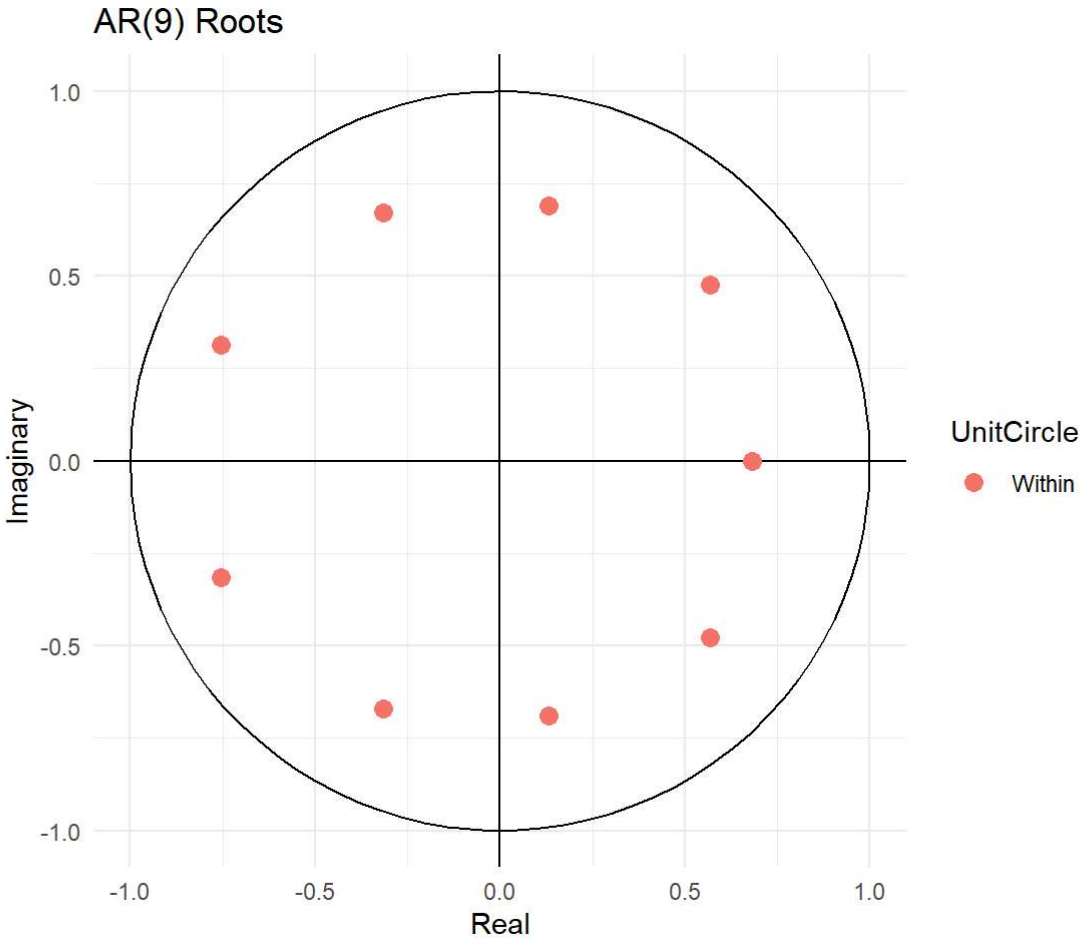
```
pmax <- 15
AICc_vals <- LBp_vals <- numeric(pmax + 1)

for (p in 0:pmax) {
  eq <- Arima(data_est$returns, order = c(p, 0, 0), include.mean = FALSE, method = "ML")
  AICc_vals[p + 1] <- eq$aicc
  LBp_vals[p + 1] <- Box.test(residuals(eq), lag = p + 4, type = "Ljung-Box", fitdf = p)$p.va
lue
}

best_p <- which.min(AICc_vals) - 1
cat("Selected AR order p =", best_p, "\nAICc =", round(min(AICc_vals), 3),
    "\nLjung-Box p-value =", round(LBp_vals[best_p + 1], 4), "\n")
```

```
## Selected AR order p = 9
## AICc = -10061.52
## Ljung-Box p-value = 0.9647
```

(b)

The inverse roots of this model all lie strictly within the unit circle, which conforms the stationarity of this model. Therefore, an AR(9) process is a suitable conditional mean model for Google's returns.

```
p <- 9
ar9 <- Arima(na.omit(data_est$returns), order = c(p,0,0), include.mean = FALSE)
autoplot(ar9) + ggtitle("AR(9) Roots")
```



Question 4

(a)

The analysis below model's Google's returns using an AR(9)-GARCH(1,1) process, which can be represented as:

$$\begin{aligned} \text{Mean equation:} \quad & Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_9 Y_{t-9} + U_t, \\ \text{Conditional variance (GARCH(1,1))}: \quad & \sigma_t^2 = \omega + \alpha_1, U_{t-1}^2 + \gamma_1, \sigma_{t-1}^2. \end{aligned}$$

Using the coefficients estimated below, these equations can be rewritten as:

$$Y_t = -0.0239\,Y_{t-1} - 0.0185\,Y_{t-2} - 0.0369\,Y_{t-3} - 0.0300\,Y_{t-4} - 0.0487\,Y_{t-5} - 0.0215\,Y_{t-6} + 0.0399\,Y_{t-7} - 0.0254\,Y_{t-8} + 0.0255\,Y_{t-9} + U_t,$$

$$\sigma_t^2 = 0.0000 + 0.0609\,U_{t-1}^2 + 0.8891\,\sigma_{t-1}^2.$$

```
ar_garch_model <- ugarchfit(
  data = data_est$returns,
  ugarchspec(
    variance.model = list(garchOrder = c(1, 1)),
    mean.model     = list(armaOrder = c(p, 0), include.mean = FALSE)))

coef_table <- round(coef(ar_garch_model), 4) %>%  print()
```

```
##     ar1     ar2     ar3     ar4     ar5     ar6     ar7     ar8     ar9   omega
## -0.0239 -0.0185 -0.0369 -0.0300 -0.0487 -0.0215  0.0399 -0.0254  0.0255  0.0000
##  alpha1   beta1
##  0.0609  0.8891
```

(b)

After fitting an ARMA-(G)ARCH model the Weighted ARCH LM tests examine whether there is any remaining ARCH structure in the residuals. The p-values for these tests are very large (0.6985, 0.9748, 0.9912 at lags 3, 5 and 7 respectively). Therefore we fail to reject the null hypothesis of no remaining ARCH effects in the standardised residuals at any reasonable significance level. Therefore, the AR(9)-GARCH(1,1) model successfully captures the conditional variance dynamics of Google's daily returns as there is no evidence of leftover volatility clustering in the standardised residuals.

```
garch_summary <- capture.output(show(ar_garch_model))
start_idx <- which(garch_summary == "Weighted ARCH LM Tests")
ARCHLM_extract <- garch_summary[start_idx:(start_idx + 6)]
cat(ARCHLM_extract, sep = "\n")
```

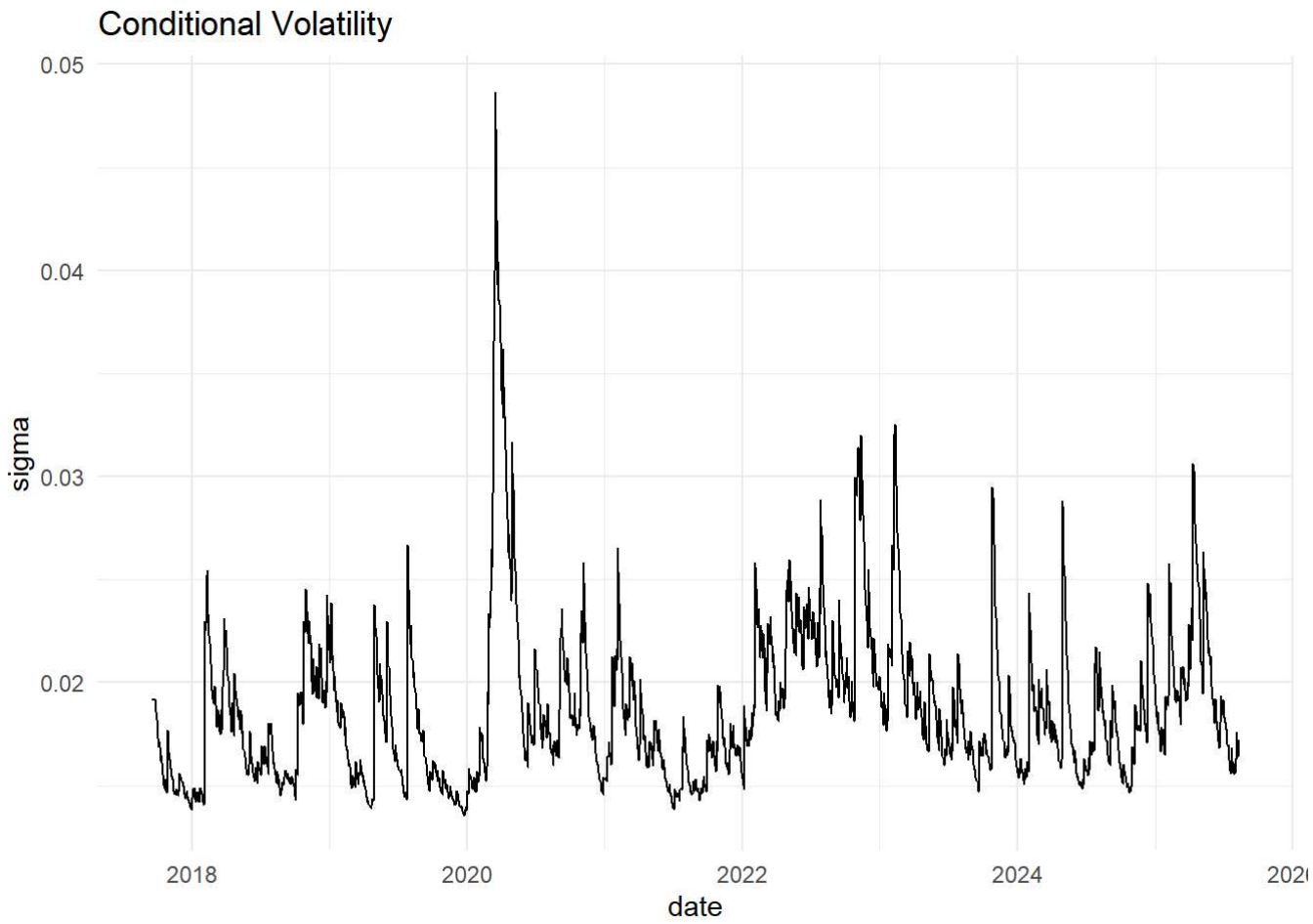
```
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.1501 0.500 2.000  0.6985
## ARCH Lag[5]    0.1591 1.440 1.667  0.9748
## ARCH Lag[7]    0.3289 2.315 1.543  0.9912
```

(c)

This "volatility" was greatest around early 2020, which aligns with the initial outbreak of the COVID-19 pandemic and associated global market turmoil. This lines up with the March 2020 equity crash, when investors reacted to lockdowns and uncertainty in earnings.

```
vol_df <- tibble(date = data_est$date, sd = sigma(ar_garch_model))
ggplot(vol_df, aes(date, sd)) + geom_line() + labs(title="Conditional Volatility", y="sigma")
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



(d)

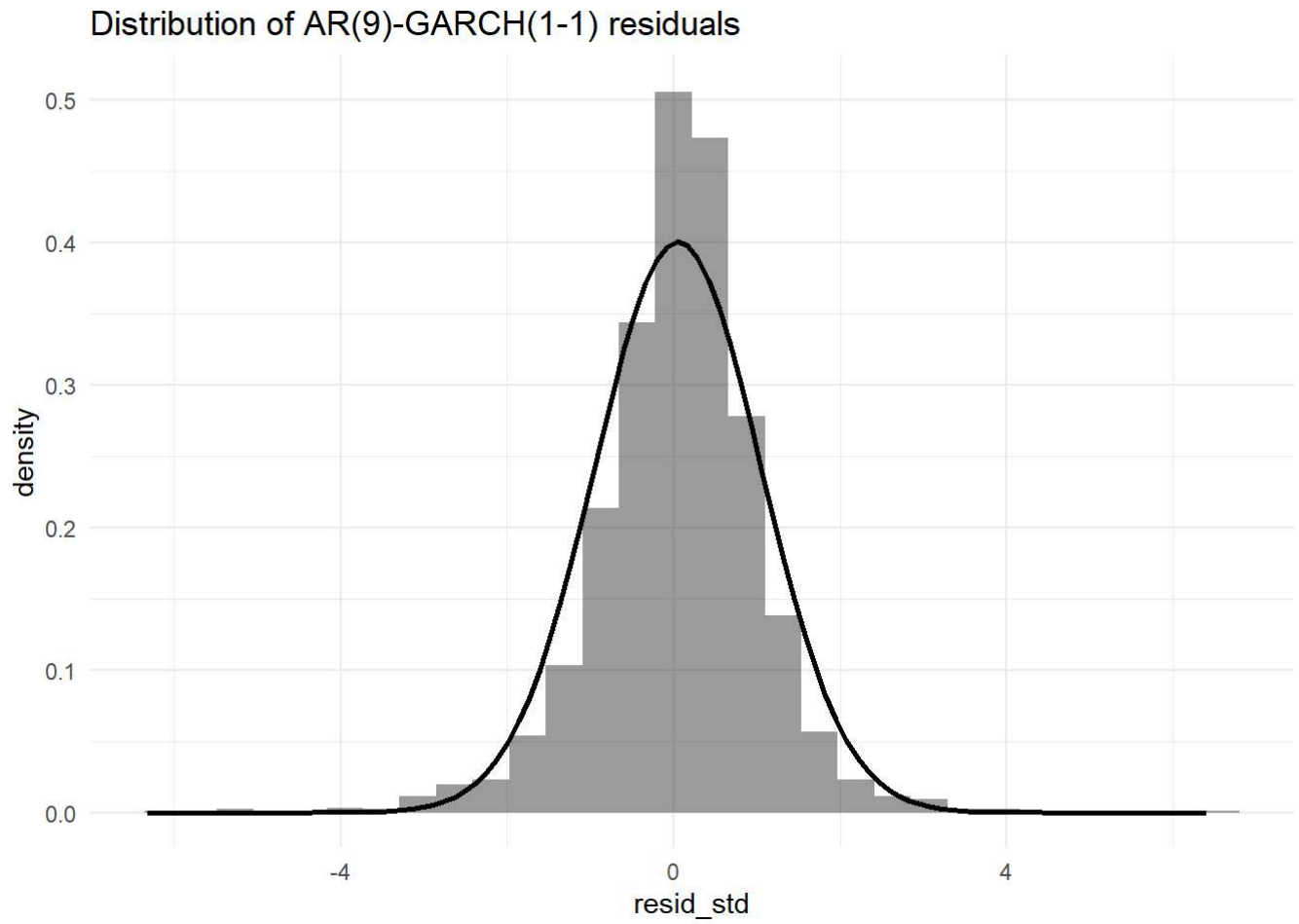
The standardised residuals appear centered around zero with approximately unit variable, suggesting this model has correctly captured the conditional variance element of this process.

However, the residual distribution shows evidence of leptokurtosis, as indicated by the descriptive statistics: skewness of –0.294 and kurtosis of 7.26, which is more than double the Gaussian benchmark of 3.

Overall, this indicated that although volatility clustering has been filtered out, the residuals still exhibit non-normal behaviour in higher moments.

Specifically, it has been unabel to deal with the heavy tails of this distribution, implying prediction intervals based on a normality assumption will understate the probability of extreme negative events (losses because of the slight negative skeweness of this distribution).

```
resid_std <- as.numeric(residuals(ar_garch_model, standardize = TRUE))

ggplot(tibble(resid_std), aes(resid_std)) +
  geom_histogram(aes(y = after_stat(density)), bins = 30, alpha = 0.6) +
  stat_function(fun = dnorm, args = list(mean = mean(resid_std), sd = sd(resid_std)), linewid
th = 1) +labs(title="Distribution of AR(9)-GARCH(1-1) residuals")
```




```
tibble(
  Mean = mean(resid_std),
  SD = sd(resid_std),
  Skewness = mean((resid_std - mean(resid_std))^3) / sd(resid_std)^3,
  Kurtosis = mean((resid_std - mean(resid_std))^4) / sd(resid_std)^4
)
```

```
## # A tibble: 1 × 4
##     Mean    SD Skewness Kurtosis
##    <dbl> <dbl>    <dbl>    <dbl>
## 1 0.0502 0.996   -0.294     7.26
```

Question 5

(a)

```
fc <- forecast(ar9, h = 10)
fc_ar9 <- tibble(H = 1:10,
                 Forecast = as.numeric(fc$mean),
                 Lo95 = as.numeric(fc$lower[,"95%"]),
                 Hi95 = as.numeric(fc$upper[,"95%"]),
                 Actual = as.numeric(data_fcst$returns),
                 Error = Actual - Forecast)

print(fc_ar9)
```

```
## # A tibble: 10 × 6
##        H   Forecast    Lo95   Hi95   Actual     Error
##    <int>      <dbl>   <dbl>  <dbl>    <dbl>     <dbl>
##  1     1 -0.00156   -0.0391 0.0359  0.0116   0.0131
##  2     2  0.0000371 -0.0375 0.0376 -0.00681 -0.00685
##  3     3 -0.00360   -0.0412 0.0340  0.00484  0.00845
##  4     4  0.00238   -0.0352 0.0400  0.00472  0.00234
##  5     5 -0.00163   -0.0392 0.0360 -0.00196 -0.000335
##  6     6  0.00205   -0.0356 0.0397 -0.00953 -0.0116
##  7     7 -0.00104   -0.0387 0.0366 -0.0112  -0.0102
##  8     8  0.00174   -0.0360 0.0395  0.00216  0.000411
##  9     9 -0.0000202 -0.0378 0.0378  0.0312   0.0313
## 10    10 -0.000462  -0.0383 0.0374  0.0116   0.0120
```

(b)

```
garch_fc <- ugarchforecast(ar_garch_model, n.ahead = 10)
fc_garch <- tibble(
  H = 1:10,
  Forecast = as.numeric(fitted(garch_fc)),
  SD_Forecast = as.numeric(sigma(garch_fc)),
  Lo95 = Forecast - 1.96 * SD_Forecast,
  Hi95 = Forecast + 1.96 * SD_Forecast,
  Actual = as.numeric(data_fcst$returns),
  Error = Actual - Forecast)
print(fc_garch)
```

```
## # A tibble: 10 × 7
##        H   Forecast SD_Forecast    Lo95   Hi95   Actual    Error
##    <int>      <dbl>       <dbl>   <dbl>  <dbl>    <dbl>    <dbl>
##  1     1 -0.00115        0.0168 -0.0340 0.0317  0.0116   0.0127
##  2     2 -0.000232       0.0169 -0.0333 0.0329 -0.00681 -0.00658
##  3     3 -0.00213        0.0170 -0.0355 0.0312  0.00484  0.00697
##  4     4  0.0000347      0.0171 -0.0335 0.0336  0.00472  0.00468
##  5     5 -0.000493       0.0172 -0.0343 0.0333 -0.00196 -0.00147
##  6     6  0.00131        0.0173 -0.0327 0.0353 -0.00953 -0.0108
##  7     7 -0.000577       0.0174 -0.0347 0.0336 -0.0112  -0.0106
##  8     8  0.000750       0.0175 -0.0336 0.0351  0.00216  0.00140
##  9     9 -0.0000299      0.0176 -0.0345 0.0345  0.0312   0.0313
## 10    10 -0.000116       0.0177 -0.0348 0.0345  0.0116   0.0117
```

(c)

The AR(9)-GARCH(1,1) model is marginally preferred by virture of a lower RMSE.

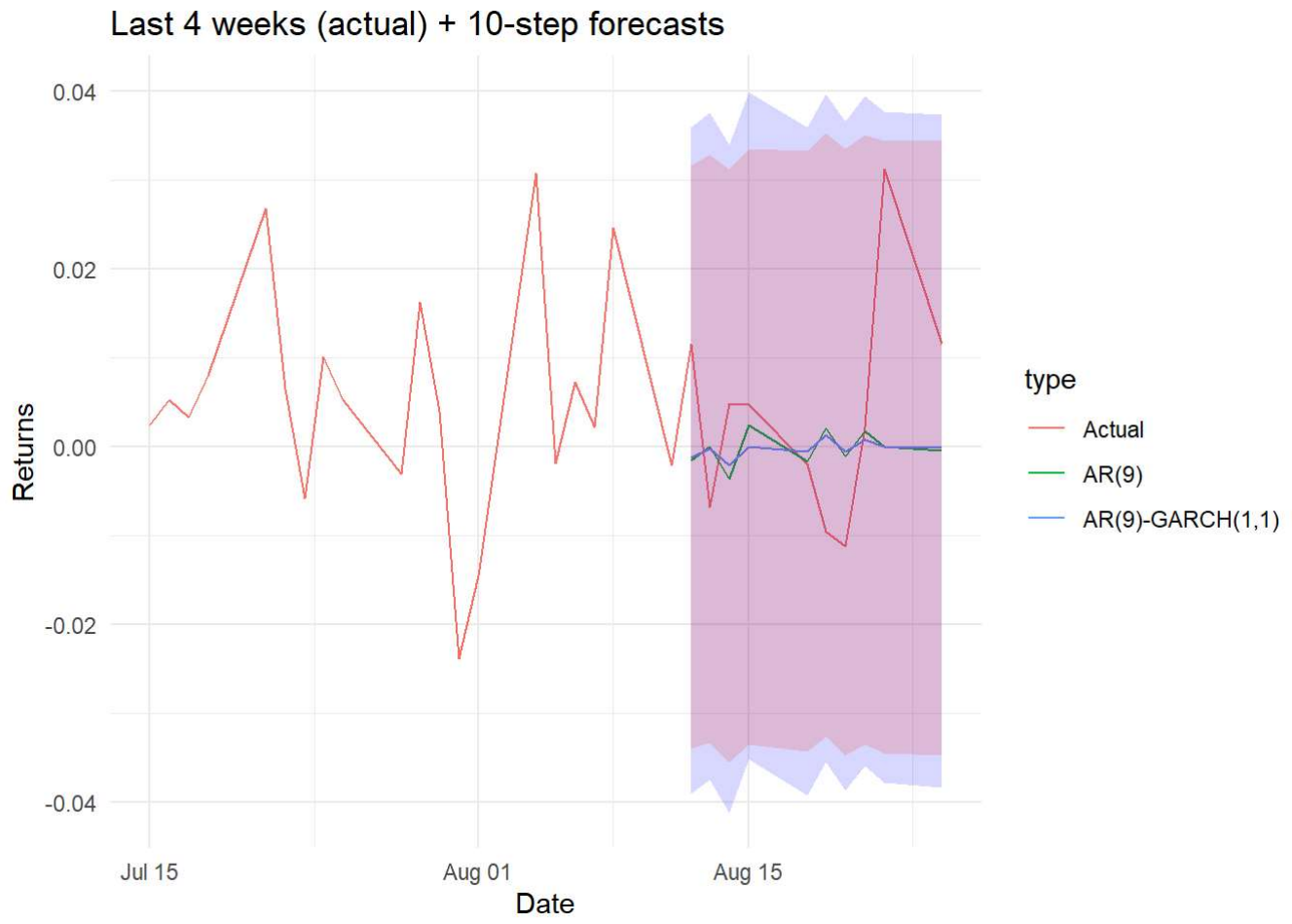
```
rmse_ar9 <- sqrt(mean(fc_ar9$Error^2))
rmse_garch <- sqrt(mean(fc_garch$Error^2))
tibble(Model = c("AR(9)", "AR(9)-GARCH(1,1)"),
       RMSE = c(rmse_ar9, rmse_garch))
```

```
## # A tibble: 2 × 2
##   Model              RMSE
##   <chr>             <dbl>
## 1 AR(9)            0.0129
## 2 AR(9)-GARCH(1,1) 0.0127
```

(d)

- Both models produce similar point forecasts.
- However, their uncertainty assessments (as proxied by forecast intervals) differ. The AR model produces intervals which widen gradually with horizon due to accumulating error preidctions (based on the homoskedasticity assumption) whereas the AR-GARCH model's intervals are wider overall as a reaction to the elevated volatility observed in the final part of the sample.

```
pdat <- bind_rows(
  tail(data_est,20) %>% transmute(date, y = returns, type = "Actual"),
  data_fcst %>% transmute(date, y = returns, type = "Actual"),
  fc_ar9 %>% mutate(date = data_fcst$date) %>% transmute(date, y = Forecast, type = "AR(9)"),
  fc_garch %>% mutate(date = data_fcst$date) %>% transmute(date, y = Forecast, type = "AR(9)-
GARCH(1,1)")
)

ggplot()+
  geom_line(data = pdat, aes(date, y, color = type))+
  geom_ribbon(data = fc_ar9 %>% mutate(date = data_fcst$date), aes(date, ymin = Lo95, ymax =
Hi95), alpha = .15, fill = "blue")+
  geom_ribbon(data = fc_garch %>% mutate(date = data_fcst$date), aes(date, ymin = Lo95, ymax
= Hi95), alpha = .15, fill = "red")+
  labs(title = "Last 4 weeks (actual) + 10-step forecasts", x = "Date", y = "Returns")
```



(e)

The boostrap prediction intervals are centred closer to the mean forecasts obtained in part (b), with median values closer to zero (consistent with the model's tiny conditional mean).

However, the intervals themselves are narrower and slightly more stable across horizons. This is because this empirical approach to producing intervals produces is less dispersed than the theoretical normal assumption used in part (b).

Both approaches yield similar point forecasts, but the boostrap approach is more robust to the heavy-tailed/slight negative skewness of the residual distribution observed in Q4(d). This gives a more realistic measure of forecasts uncertainty under a violation of normality assumptions.

```
boot <- ugarchboot(ar_garch_model, method = "partial", n.ahead = 10, n.bootpred = 5000)

boot_pi <- tibble(
  H = 1:10,
  Median = apply(boot@forc@forecast$series, 2, median),
  Lo95 = apply(boot@forc@forecast$series, 2, quantile, probs = 0.025),
  Hi95 = apply(boot@forc@forecast$series, 2, quantile, probs = 0.975)
) %>% print()
```

```
## # A tibble: 10 × 4
##        H    Median     Lo95    Hi95
##    <int>     <dbl>    <dbl>   <dbl>
##  1     1 -0.000174 -0.00191 0.00118
##  2     2 -0.000174 -0.00191 0.00118
##  3     3 -0.000174 -0.00191 0.00118
##  4     4 -0.000174 -0.00191 0.00118
##  5     5 -0.000174 -0.00191 0.00118
##  6     6 -0.000174 -0.00191 0.00118
##  7     7 -0.000174 -0.00191 0.00118
##  8     8 -0.000174 -0.00191 0.00118
##  9     9 -0.000174 -0.00191 0.00118
## 10    10 -0.000174 -0.00191 0.00118
```

(f)

In question 4(d) the standardised residuals were found to be negatively skewed and leptokurtic. Therefore, we should expect the bootstrap prediction intervals to deviate from the symmetric Gaussian bands used in part (b).

The additional residual statistics defined below confirm this:

- The empirical lower 2.5% quantile (-2.13) is more extreme than the normal benchmark (-1.96), while the upper 97.5% quantile (1.85) is slightly lower than the normal benchmark (1.96). This indicated a heavier left tail and morem ass near zero consistent with negative skewness and high kurtosis.
- These explain the observed differences in interval shape and width. The normal intervals from part (b) are wide and symmetric (approximately 0.066 - 0.069) while the boostrap intervals are tighter (consistent 0.003 width). The narrower, stable boostrap intervals reflect the fact taht most resampled residuals fall close to zero thanks to the very large peaking of the empirical distribution (much greater than under normality).
- skewness = −0.29, kurtosis = 7.26, and the lower 2.5% quantile (q_emp_2.5) = −2.13 vs −1.96 under Normality (q_norm_2.5). These indicate a heavier left tail and more mass near zero, which explain why the bootstrap intervals are both tighter overall and slightly asymmetric on the lower side.

```
# Comparing prediction intervals
comp <- tibble(H = 1:10,
  Analytic_Width = fc_garch$Hi95 - fc_garch$Lo95,
  Bootstrap_Width = boot_pi$Hi95 - boot_pi$Lo95) %>% print()
```

```
## # A tibble: 10 × 3
##        H Analytic_Width Bootstrap_Width
##    <int>          <dbl>           <dbl>
##  1     1         0.0657         0.00309
##  2     2         0.0662         0.00309
##  3     3         0.0667         0.00309
##  4     4         0.0671         0.00309
##  5     5         0.0675         0.00309
##  6     6         0.0679         0.00309
##  7     7         0.0683         0.00309
##  8     8         0.0687         0.00309
##  9     9         0.0690         0.00309
## 10    10         0.0693         0.00309
```

```
# Additional statistics
e <- as.numeric(residuals(ar_garch_model, standardize = TRUE)); m <- mean(e); s <- sd(e)
skew <- mean((e - m)^3) / s^3
kurt <- mean((e - m)^4) / s^4
tails <- tibble(
  q_emp_2.5 = quantile(e, .025), q_emp_97.5 = quantile(e, .975),
  q_norm_2.5 = qnorm(.025),       q_norm_97.5 = qnorm(.975),
  Skewness = skew, Kurtosis = kurt) %>% print()
```

```
## # A tibble: 1 × 6
##   q_emp_2.5 q_emp_97.5 q_norm_2.5 q_norm_97.5 Skewness Kurtosis
##       <dbl>      <dbl>      <dbl>       <dbl>    <dbl>    <dbl>
## 1     -2.13       1.85      -1.96        1.96   -0.294     7.26
```