W271 Lab 2

Akin Akinlabi, Eduardo Gonzalez, Karanveer Lamba, Rui Sun, Victor Ndayambaje

2025-03-16

Introduction

The Taylor Rule serves as a guideline for central banks, particularly the U.S. Federal Reserve, in adjusting interest rates based on inflation and output gaps. This study examines the empirical relevance of the Taylor Rule by analyzing quarterly U.S. macroeconomic data from 1980 to 2023. The dataset includes the Federal Funds Rate, an inflation measure based on Personal Consumption Expenditures (PCE) inflation, and the output gap, which measures the difference between actual and potential GDP. The data was sourced from FRED using the fredr package in R.. The primary objective is to examine the empirical relevance of the Taylor Rule and assess the potential for cointegration and error correction modeling in explaining Federal Reserve policy decisions. This paper presents the data retrieval process, initial exploratory data analysis, and preprocessing steps. Key economic events, such as the Volcker disinflation in the 1980s, the 2008 Financial Crisis, and the COVID-19 pandemic, are contextualized within the dataset. Additionally, statistical techniques, including outlier detection, seasonality assessment, and stationarity testing, are applied to ensure the robustness of the analysis.

Task 1: Data Retrieval and Initial EDA (Quarterly Frequency)

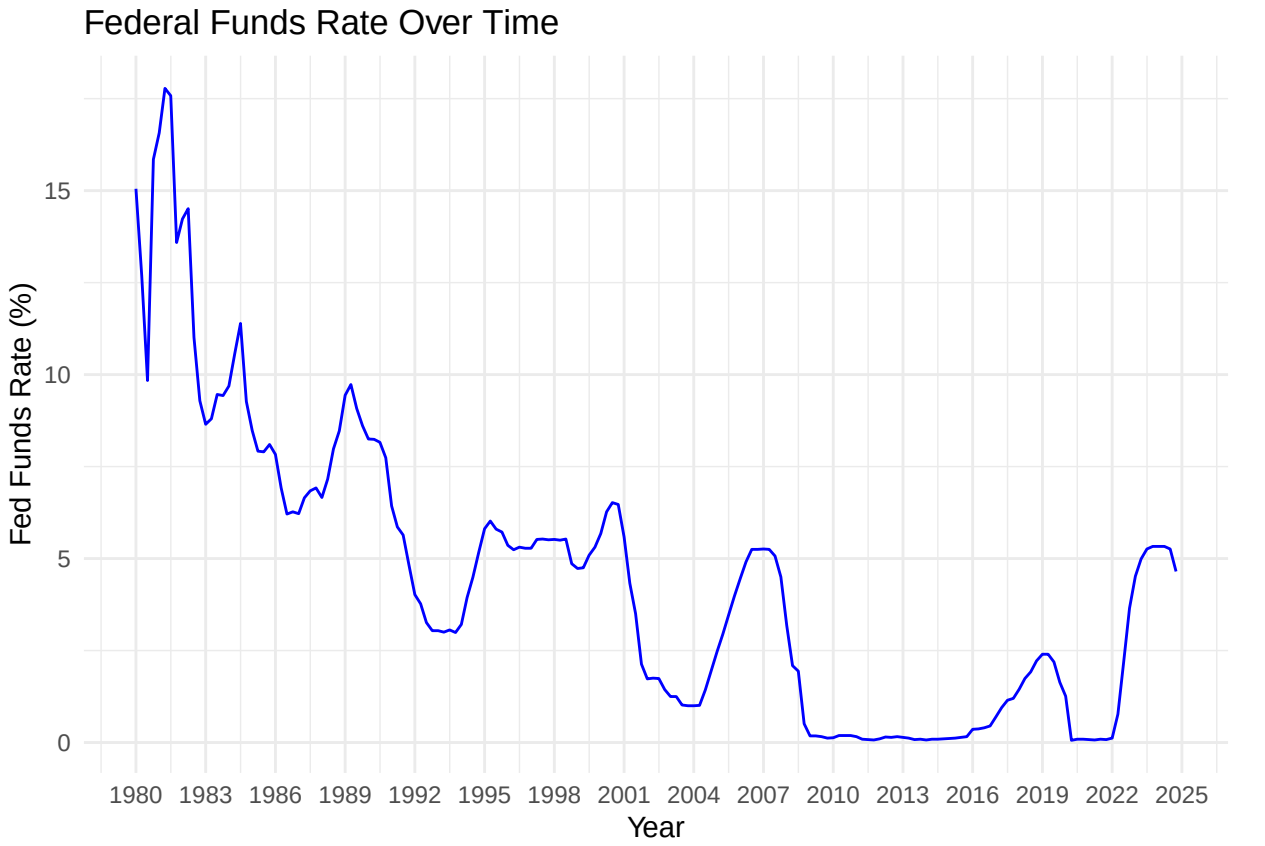
```
# I've used the fredr package in R to pull data - https://cran.r-project.org/web/packages/fredr/vignett

# Set FRED API Key object
fredr_set_key("c58e9c0793ba6ec253e8588464dd1eca")

# Set date range
start_date <- as.Date("1980-01-01")
end_date <- Sys.Date()

# Download Federal Funds Rate (FEDFUNDS)
fed_funds <- fredr(series_id = "FEDFUNDS", observation_start = start_date, frequency = "q") %>%
  select(date, fed_rate = value)

# Download PCE Price Index (PCEPI) for Inflation
pcepi <- fredr(series_id = "PCEPI", observation_start = start_date, frequency = "q") %>%
  select(date, pce_index = value)

# Download Real GDP (GDPC1)
real_gdp <- fredr(series_id = "GDPC1", observation_start = start_date, frequency = "q") %>%
  select(date, real_gdp = value)

# Download Potential GDP (GDPPOT)
potential_gdp <- fredr(series_id = "GDPPOT", observation_start = start_date, frequency = "q") %>%
  select(date, potential_gdp = value)
```

```r
pcepi <- pcepi %>%
  arrange(date) %>%
  mutate(inflation = round((pce_index / lag(pce_index) - 1) * 100, 2))
```

```r
output_gap <- real_gdp %>%
  inner_join(potential_gdp, by = "date") %>%
  mutate(output_gap = ((real_gdp - potential_gdp) / potential_gdp) * 100)
```

```r
df <- fed_funds %>%
  inner_join(pcepi, by = "date") %>%
  inner_join(output_gap, by = "date")
```

```r
# Federal Funds Rate over time
ggplot(df, aes(x = date, y = fed_rate)) +
  geom_line(color = "blue") +
  labs(title = "Federal Funds Rate Over Time", x = "Year", y = "Fed Funds Rate (%)") +
  scale_x_date(date_breaks = "3 years", date_labels = "%Y") +  # Set 2-year gaps on X-axis
  theme_minimal()
```



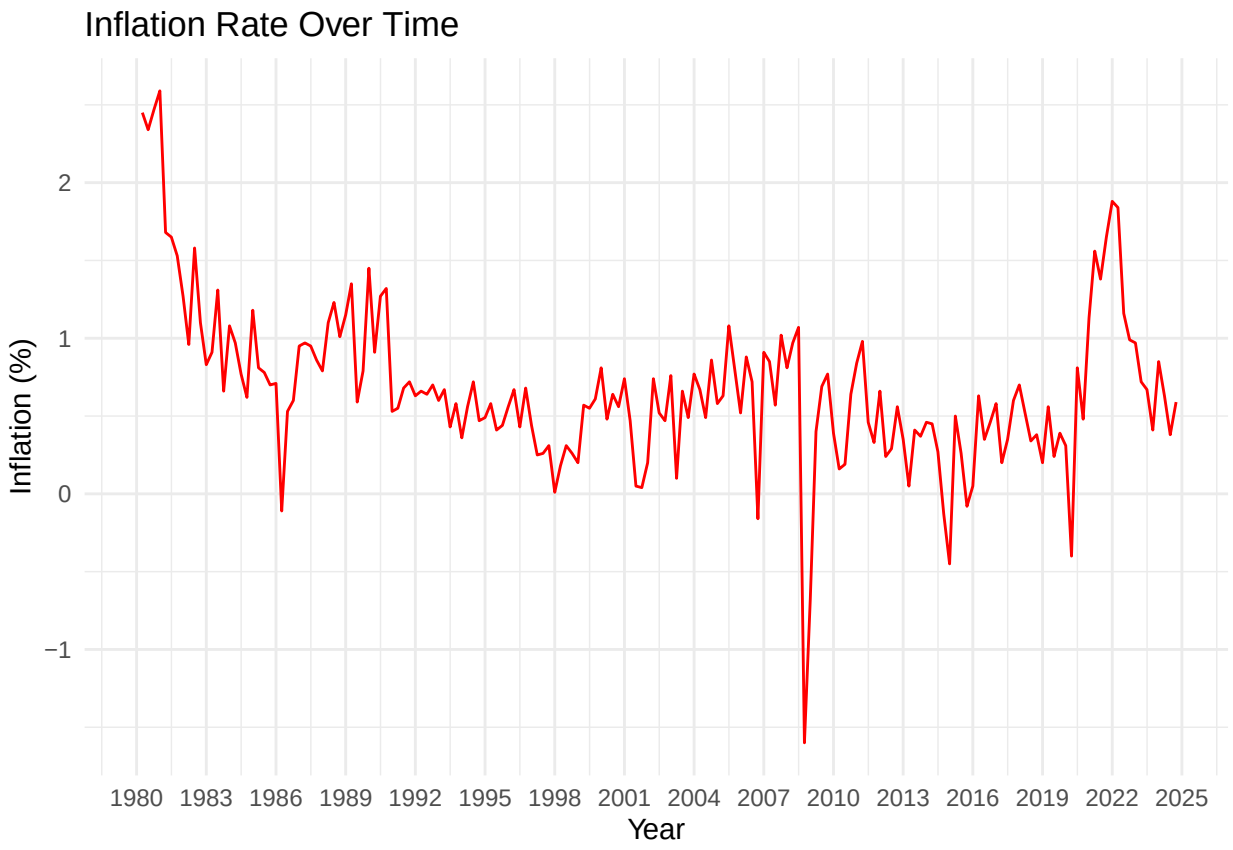
Federal Funds Rate Plot Interpretation

- 1980s: Interest rates were at historically high levels (above 15%) due to the Federal Reserve's aggressive tightening to combat inflation.
- 2000s: The rate remained relatively stable before being cut sharply after the 2008 Financial Crisis, when the Fed reduced rates to near-zero to stimulate the economy.
- 2020: Another near-zero rate policy followed during the COVID-19 pandemic to counter economic downturns.

- 2022-Present: A rapid rate hike is visible, reflecting the Fed's response to post-pandemic inflation.

```
# Inflation over time
ggplot(df, aes(x = date, y = inflation)) +
  geom_line(color = "red") +
  labs(title = "Inflation Rate Over Time", x = "Year", y = "Inflation (%)") +
  scale_x_date(date_breaks = "3 years", date_labels = "%Y") +  # Set 2-year gaps on X-axis
  theme_minimal()
```

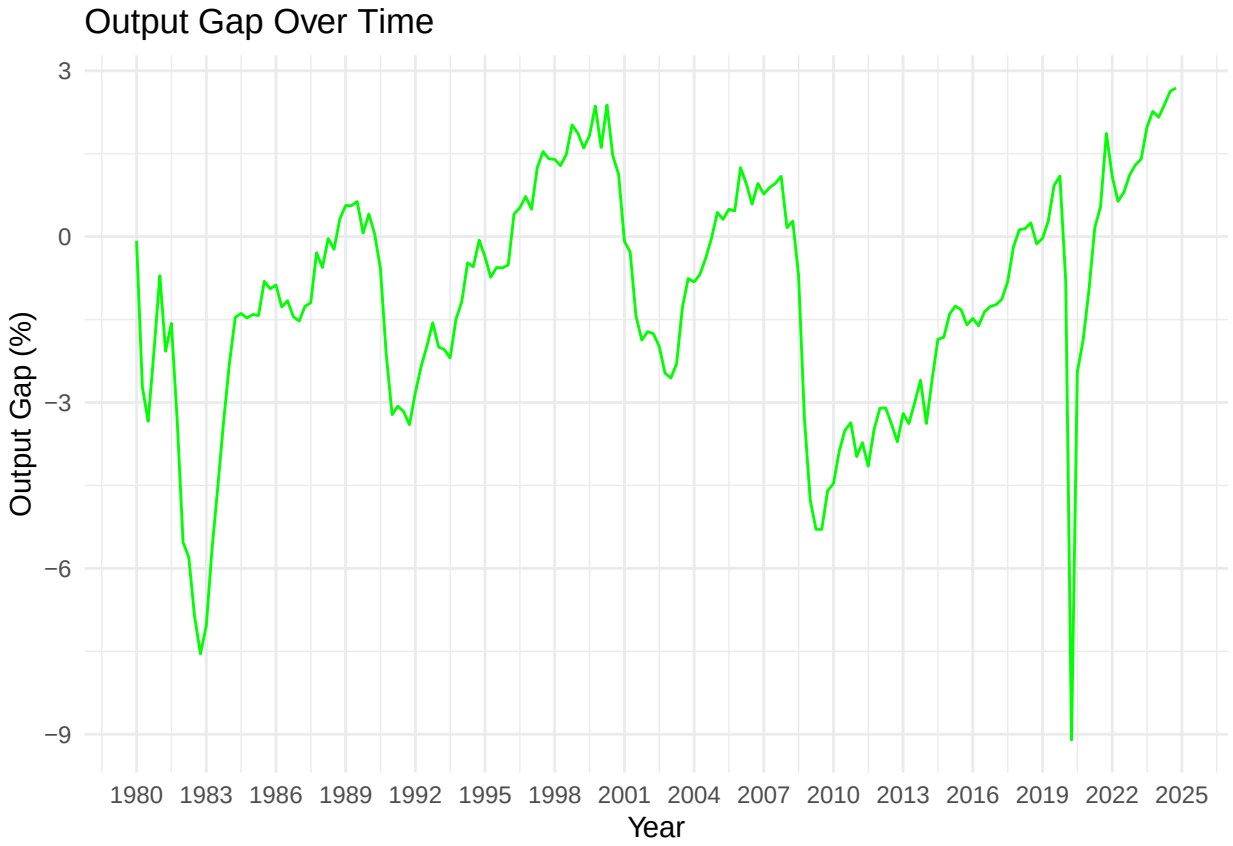
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```



Inflation Rate Plot Interpretation

- 1980s: Inflation started high but steadily declined after the Volcker-led Fed raised interest rates aggressively.
- 2008 Financial Crisis: There was a brief period of deflation (inflation dropping below zero).
- 2020s: A sharp spike in inflation post-COVID due to supply chain disruptions, stimulus spending, and demand recovery.

```
# Output Gap over time
ggplot(df, aes(x = date, y = output_gap)) +
  geom_line(color = "green") +
  labs(title = "Output Gap Over Time", x = "Year", y = "Output Gap (%)") +
  scale_x_date(date_breaks = "3 years", date_labels = "%Y") +  # Set 2-year gaps on X-axis
  theme_minimal()
```



Output Gap Plot Interpretation

- 1980s-1990s: Large negative output gaps were seen during recessions, indicating economic contractions.
- 2008-2009 Financial Crisis: A sharp drop in the output gap occurred, reflecting a deep recession.
- 2020 (COVID-19 Recession): The steepest drop in output gap, followed by a strong recovery.

```
summary(df)
```

```
##       date                fed_rate        pce_index        inflation
##  Min.   :1980-01-01   Min.   : 0.060   Min.   : 37.55   Min.   :-1.6000
##  1st Qu.:1991-03-09   1st Qu.: 1.000   1st Qu.: 61.45   1st Qu.: 0.4100
##  Median :2002-05-16   Median : 4.475   Median : 76.34   Median : 0.6300
##  Mean   :2002-05-17   Mean   : 4.424   Mean   : 78.09   Mean   : 0.6731
##  3rd Qu.:2013-07-24   3rd Qu.: 6.270   3rd Qu.: 95.99   3rd Qu.: 0.8600
##  Max.   :2024-10-01   Max.   :17.780   Max.   :124.48   Max.   : 2.5900
##                                                         NA's   :1
##     real_gdp     potential_gdp     output_gap
##  Min.   : 7182   Min.   : 7347   Min.   :-9.1033
##  1st Qu.:10083   1st Qu.:10331   1st Qu.:-2.2858
##  Median :14490   Median :14766   Median :-0.8488
##  Mean   :14426   Mean   :14563   Mean   :-1.0864
##  3rd Qu.:17884   3rd Qu.:18434   3rd Qu.: 0.4936
##  Max.   :23536   Max.   :22920   Max.   : 2.6877
##
```

This study examines the empirical relevance of the Taylor Rule by analyzing quarterly U.S. macroeconomic data from 1980 to 2023. The dataset includes the Federal Funds Rate, an inflation measure based on

Personal Consumption Expenditures (PCE) inflation, and the output gap, which measures the difference between actual and potential GDP. The data was sourced from FRED using the fredr package in R. The Federal Funds Rate has exhibited significant fluctuations over the past four decades. During the 1980s, interest rates exceeded 15%, reflecting the Federal Reserve's aggressive tightening to combat inflation. In the 2000s, rates remained relatively stable but dropped sharply following the 2008 Financial Crisis, when the Fed cut rates to near-zero to stimulate economic activity. Similarly, in 2020, the Fed once again implemented near-zero interest rates in response to the COVID-19 pandemic. However, starting in 2022, a rapid series of rate hikes occurred as the Fed sought to control post-pandemic inflationary pressures. Inflation trends reflect these monetary policy adjustments. The 1980s saw persistently high inflation, which gradually declined due to restrictive monetary policy. A brief period of deflation occurred during the 2008 Financial Crisis, with inflation falling below zero. However, in the 2020s, inflation spiked dramatically due to post-pandemic supply chain disruptions, stimulus measures, and demand surges. The output gap provides further insights into economic cycles. Large negative output gaps, indicating recessions, were observed in the 1980s, early 1990s, and during the 2008 Financial Crisis. The COVID-19 pandemic caused the steepest drop in the output gap, reflecting an abrupt economic contraction, but a strong recovery followed as the economy reopened.

Task 2: Preprocessing

```
df$quarter <- as.yearqtr(df$date)  # Create a quarter column
df_quarterly <- aggregate(cbind(fed_rate, pce_index, inflation, real_gdp, potential_gdp, output_gap) ~ q
summary(df_quarterly)
```

```
##     quarter        fed_rate        pce_index        inflation
##  Min.   :1980   Min.   : 0.060   Min.   : 38.46   Min.   :-1.6000
##  1st Qu.:1991   1st Qu.: 1.000   1st Qu.: 61.74   1st Qu.: 0.4100
##  Median :2002   Median : 4.460   Median : 76.54   Median : 0.6300
##  Mean   :2002   Mean   : 4.365   Mean   : 78.32   Mean   : 0.6731
##  3rd Qu.:2014   3rd Qu.: 6.245   3rd Qu.: 96.08   3rd Qu.: 0.8600
##  Max.   :2025   Max.   :17.780   Max.   :124.48   Max.   : 2.5900
##     real_gdp     potential_gdp     output_gap
##  Min.   : 7182   Min.   : 7391   Min.   :-9.1033
##  1st Qu.:10087   1st Qu.:10378   1st Qu.:-2.2908
##  Median :14520   Median :14813   Median :-0.8761
##  Mean   :14465   Mean   :14603   Mean   :-1.0920
##  3rd Qu.:17907   3rd Qu.:18455   3rd Qu.: 0.4941
##  Max.   :23536   Max.   :22920   Max.   : 2.6877
```

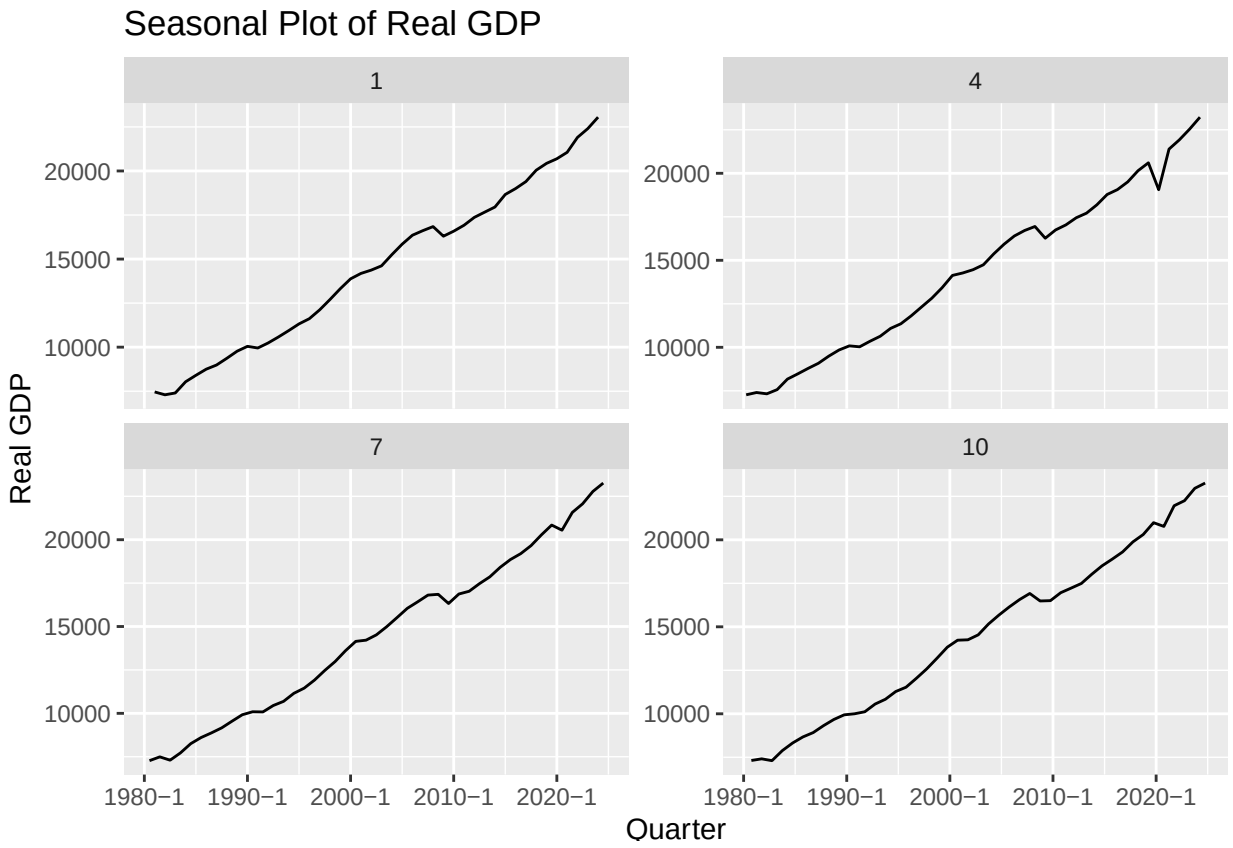
```
# Set percentile thresholds for Winsorizing (1st and 99th percentile)
lower_percentile <- 0.01
upper_percentile <- 0.99

# Function to Winsorize a variable
winsorize <- function(x, lower_percentile, upper_percentile) {
  lower_cutoff <- quantile(x, lower_percentile)
  upper_cutoff <- quantile(x, upper_percentile)

  # Cap values at the 1st and 99th percentiles
  x[x < lower_cutoff] <- lower_cutoff
  x[x > upper_cutoff] <- upper_cutoff

  return(x)
}

# Apply Winsorizing to all variables
```

```r
df_quarterly$fed_rate <- winsorize(df_quarterly$fed_rate, lower_percentile, upper_percentile)
df_quarterly$pce_index <- winsorize(df_quarterly$pce_index, lower_percentile, upper_percentile)
df_quarterly$inflation <- winsorize(df_quarterly$inflation, lower_percentile, upper_percentile)
df_quarterly$real_gdp <- winsorize(df_quarterly$real_gdp, lower_percentile, upper_percentile)
df_quarterly$potential_gdp <- winsorize(df_quarterly$potential_gdp, lower_percentile, upper_percentile)
df_quarterly$output_gap <- winsorize(df_quarterly$output_gap, lower_percentile, upper_percentile)
```

Since some of the original datasets were reported at a monthly frequency, they were converted to quarterly frequency using the as.yearqtr() function in R. Monthly observations were aggregated using the aggregate() function to ensure consistency across all variables. The exploratory analysis revealed extreme fluctuations in interest rates, inflation, and the output gap, suggesting potential outliers. Instead of removing these data points, Winsorizing was applied to cap extreme values at the 99th and 1st percentiles. This approach retains all observations while minimizing the impact of extreme values that could distort statistical analysis. To determine whether seasonal adjustment was necessary, the nsdiffs() function from the forecast package was used to test for seasonal differencing. Additionally, STL decomposition (stl()) was applied to break down the time series into trend, seasonal, and residual components. The seasonal component showed no strong repetitive patterns, and the test result indicated that seasonal differencing was not required. Consequently, no seasonal adjustment was applied to the dataset.

```r
# Seasonal plot of real_gdp
ggplot(df_quarterly, aes(x = quarter, y = real_gdp)) +
  geom_line() +
  facet_wrap(~ factor(month(quarter)), scales = "free_y") +
  labs(title = "Seasonal Plot of Real GDP", x = "Quarter", y = "Real GDP")
```

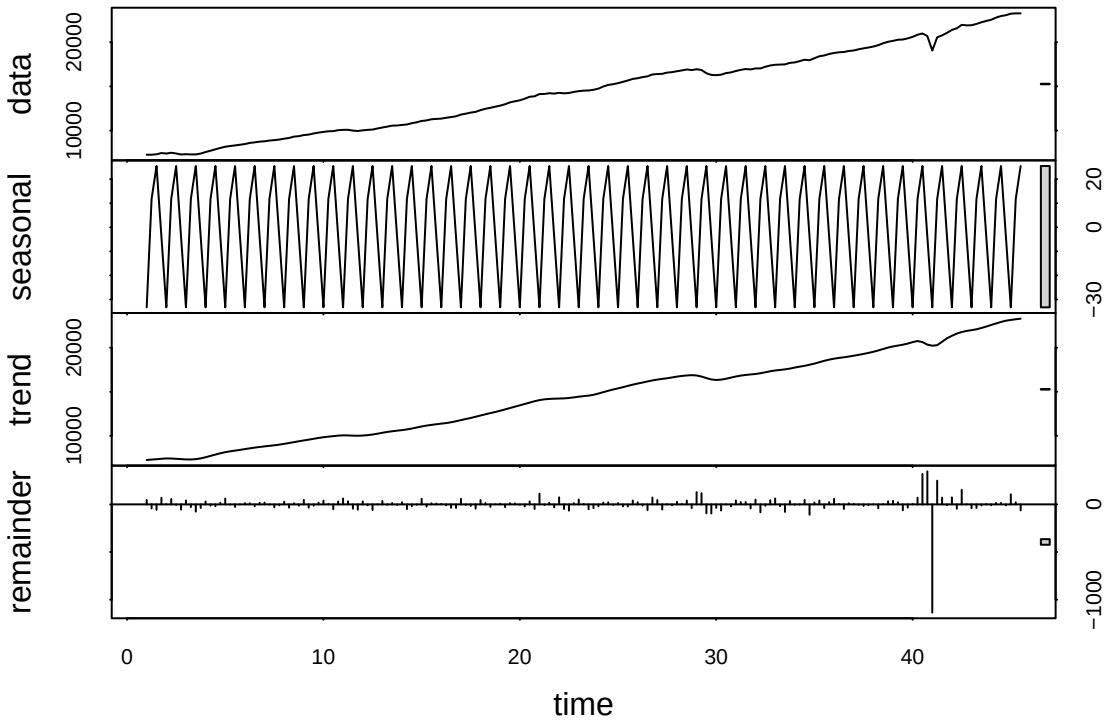
```
## Warning: The `trans` argument of `continuous_scale()` is deprecated as of ggplot2 3.5.0.
## i Please use the `transform` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
# Convert the data to a time series object (quarterly data)
ts_data <- ts(df_quarterly$real_gdp, frequency = 4)

# Test for seasonal differencing
seasonality_test <- nsdiffs(ts_data)

# Print the result
print(seasonality_test)
```

```
## [1] 0
```

```
# Apply STL decomposition
decomp <- stl(ts_data, s.window = "periodic")

# Plot the decomposition to visualize the seasonal component
plot(decomp)
```



Task 3: Stationarity Testing & Potential Structural Breaks

The goal of this task is to assess the stationarity of the time series data, particularly the quarterly real GDP series. Since structural breaks can impact standard stationarity tests, the Augmented Dickey-Fuller (ADF) test is applied with varying deterministic elements (constant, trend) to identify the most appropriate model specification. By starting with the most comprehensive model (including a trend) and progressively simplifying it if no significant trend or drift is detected, this approach ensures a robust evaluation of stationarity. Lag selection is also handled to account for potential autocorrelation, improving the reliability of the results.

```
ts_quarterly <- ts(df_quarterly$real_gdp, start = c(1980, 1), frequency = 4)

# ADF test with constant and trend, specifying 4 lags
adf_trend <- ur.df(ts_quarterly, type = "trend", lags = 4)
summary(adf_trend)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1801.83   -44.65    11.89    63.32   991.42
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.092e+02  2.418e+02   2.519   0.0127 *
## z.lag.1     -8.376e-02  3.851e-02  -2.175   0.0310 *
## tt           7.864e+00  3.443e+00   2.284   0.0236 *
## z.diff.lag1 -1.323e-01  8.030e-02  -1.648   0.1012
## z.diff.lag2 -3.222e-02  8.086e-02  -0.399   0.6908
## z.diff.lag3  3.192e-04  8.016e-02   0.004   0.9968
## z.diff.lag4 -1.829e-02  7.802e-02  -0.234   0.8149
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 189.6 on 167 degrees of freedom
## Multiple R-squared:  0.06962,    Adjusted R-squared:  0.03619
## F-statistic: 2.083 on 6 and 167 DF,  p-value: 0.05779
##
##
## Value of test-statistic is: -2.175 12.5647 3.315
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.99 -3.43 -3.13
## phi2  6.22  4.75  4.07
## phi3  8.43  6.49  5.47
```

```
# ADF test with constant (drift), specifying 4 lags
adf_drift <- ur.df(ts_quarterly, type = "drift", lags = 4)
summary(adf_drift)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1808.38   -36.77     8.39    56.93  1037.36
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 68.089522  49.297487   1.381   0.1691
## z.lag.1      0.003885   0.003309   1.174   0.2421
## z.diff.lag1 -0.188189   0.077445  -2.430   0.0162 *
## z.diff.lag2 -0.082614   0.078760  -1.049   0.2957
## z.diff.lag3 -0.044195   0.078724  -0.561   0.5753
## z.diff.lag4 -0.054586   0.077339  -0.706   0.4813
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 191.9 on 168 degrees of freedom
```

```
## Multiple R-squared:  0.04055,    Adjusted R-squared:  0.012
## F-statistic:  1.42 on 5 and 168 DF,  p-value: 0.2194
##
##
## Value of test-statistic is: 1.1739 15.8407
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.46 -2.88 -2.57
## phi1  6.52  4.63  3.81
```

```
# ADF test with no constant or trend, specifying 4 lags
adf_none <- ur.df(ts_quarterly, type = "none", lags = 4)  # Specify a valid number of lags
summary(adf_none)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1827.36   -26.96    22.84    63.35  1036.06
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## z.lag.1      0.007984   0.001467   5.442 1.83e-07 ***
## z.diff.lag1 -0.182591   0.077546  -2.355   0.0197 *
## z.diff.lag2 -0.075402   0.078798  -0.957   0.3400
## z.diff.lag3 -0.036724   0.078749  -0.466   0.6416
## z.diff.lag4 -0.048466   0.077419  -0.626   0.5321
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 192.4 on 169 degrees of freedom
## Multiple R-squared:  0.2073, Adjusted R-squared:  0.1839
## F-statistic: 8.839 on 5 and 169 DF,  p-value: 1.796e-07
##
##
## Value of test-statistic is: 5.4419
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

The Augmented Dickey-Fuller tests for unit roots were conducted with three different regression models: none, drift, and trend. The trend model, which included both a constant and a linear trend, had a test statistic of -2.18, also showing no significant evidence of stationarity, with a p-value of 0.05779. The second model (drift) included a constant, and the test statistic of 1.17 showed no significant evidence of stationarity,

with a p-value of 0.2194. Finally, the test statistic was 5.44, which is significant at the 1% level, suggesting the series is stationary. This progression from the most general model (trend) to the simplest model (none) supports the conclusion that the series is stationary.

Task 4

In this section, the team estimates the Taylor Rule using OLS on data from 1980Q1 to 2022Q4, reserving 2023Q1 onward for testing. Alpha 1 is used to determine if it satisfies the "Taylor principle", comparing the estimate with theory, assuming the target inflation to be 2% from the https://www.federalreserve.gov/economy-at-a-glance-inflation-pce.html. Model diagnostics include checking for autocorrelation in residuals (ACF, PACF, and residuals plots) and evaluating in-sample RMSE, as demonstrated below. The team then generates forecasts for 2023Q1–2024Q4 and compares out-of-sample RMSE with in-sample performance to assess predictive accuracy and potential overfitting. This provides insights into how well the estimated rule aligns with economic theory and its forecasting reliability.

```
df_quarterly <- df_quarterly %>%
  mutate(inflation_gap = inflation - 2)

# Add a second predictor: lagged value of the time series
df_quarterly <- df_quarterly %>%
  mutate(lagged_inflation = lag(inflation, order_by = quarter))
train_set <- df_quarterly %>% filter(quarter < '2022 Q4')
test_set <- df_quarterly %>% filter(quarter >= '2022 Q4')

# Remove NA values
train_set <- na.omit(train_set)
test_set <- na.omit(test_set)

# Fit the ols with lag model
ols_model <- glm(fed_rate ~ lagged_inflation + inflation_gap + output_gap , data = train_set)

# Fit the ols without lag model
ols_model <- glm(fed_rate ~ inflation_gap + output_gap , data = train_set)
# Predict on the test set
#predictions <- predict(ols_model, newdata = test_set)

# Calculate the residuals
residuals <- residuals(ols_model)

# Summarize the model
summary(ols_model)
```

```
##
## Call:
## glm(formula = fed_rate ~ inflation_gap + output_gap, data = train_set)
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   10.04574    0.75675  13.275  < 2e-16 ***
## inflation_gap  4.30803    0.52568   8.195 6.44e-14 ***
## output_gap     0.02714    0.12640   0.215     0.83
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 10.88901)
```

```
##
##     Null deviance: 2541.0  on 168  degrees of freedom
## Residual deviance: 1807.6  on 166  degrees of freedom
## AIC: 888.1
##
## Number of Fisher Scoring iterations: 2
```

```
# Predict on the training set
train_set$Predicted <- predict(ols_model, newdata = train_set)

# Predict on the test set
test_set$Predicted <- predict(ols_model, newdata = test_set)

# Calculate the residuals
residuals <- residuals(ols_model)

# Calculate the RMSE
rmse <- sqrt(mean((train_set$fed_rate - train_set$Predicted)^2))
print(paste("In Sample: Root Mean Squared Error (RMSE):", rmse))
```

```
## [1] "In Sample: Root Mean Squared Error (RMSE): 3.27043042498184"
```

```
rmse <- sqrt(mean((test_set$fed_rate - test_set$Predicted)^2))
print(paste("Out of Sample: Root Mean Squared Error (RMSE):", rmse))
```

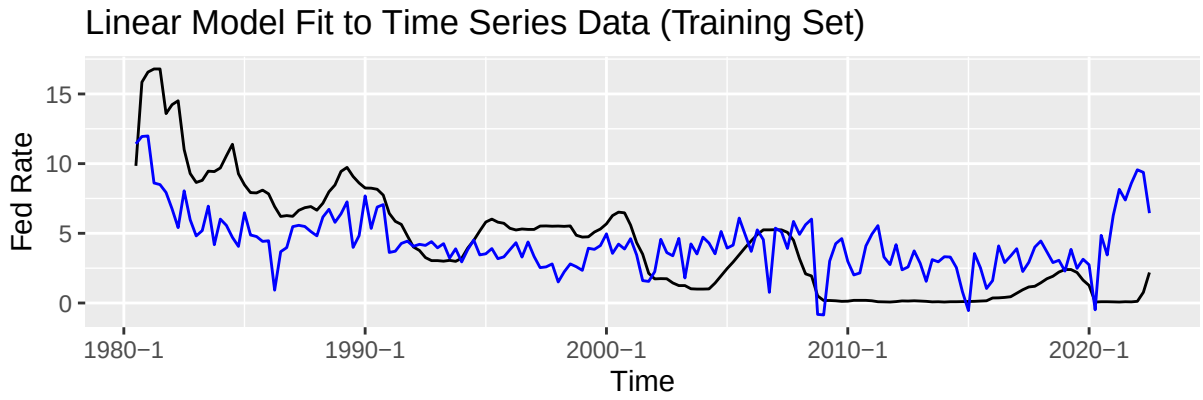
```
## [1] "Out of Sample: Root Mean Squared Error (RMSE): 1.37524946670628"
```

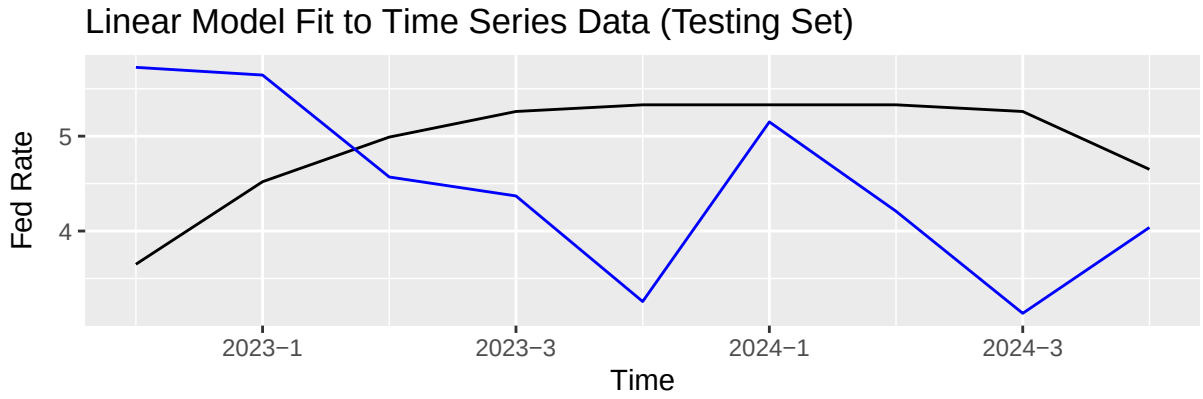
```
library(ggplot2)
library(patchwork)
```

```
##
## Attaching package: 'patchwork'
```

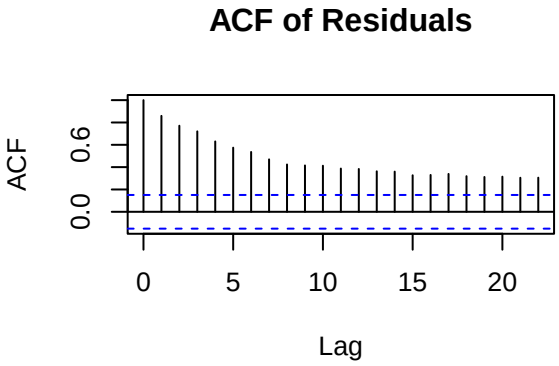
```
## The following object is masked from 'package:MASS':
##
##     area
```

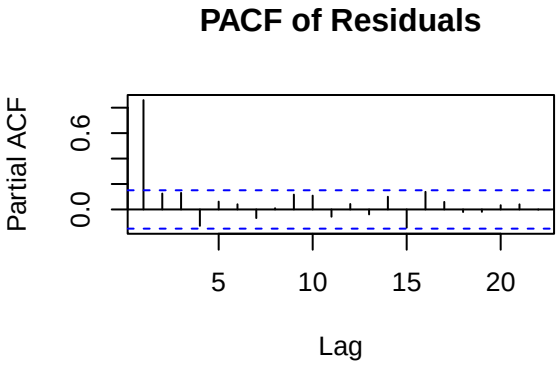
```
# Training set plot
p1 <- ggplot(train_set, aes(x = quarter)) +
  geom_line(aes(y = fed_rate), color = "black") +
  geom_line(aes(y = Predicted), color = "blue") +
  labs(title = "Linear Model Fit to Time Series Data (Training Set)",
       x = "Time",
       y = "Fed Rate")

# Testing set plot
p2 <- ggplot(test_set, aes(x = quarter)) +
  geom_line(aes(y = fed_rate), color = "black") +
  geom_line(aes(y = Predicted), color = "blue") +
  labs(title = "Linear Model Fit to Time Series Data (Testing Set)",
       x = "Time",
       y = "Fed Rate")

# Arrange the plots vertically
p1 / p2
```

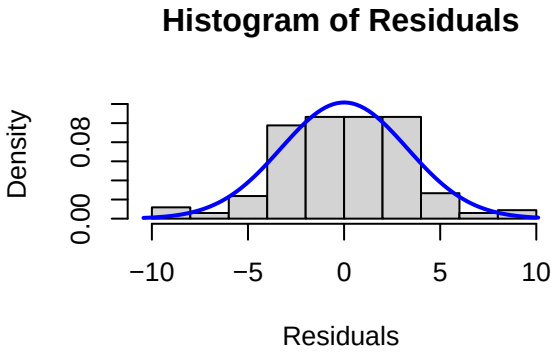





```r
# Calculate the residuals
residuals <- residuals(ols_model)

par(mfrow=c(2,2))

# Plot the residuals using an ACF plot
acf(residuals, main = "ACF of Residuals")
pacf(residuals, main = "PACF of Residuals")


# Plot the histogram of residuals with a normal distribution curve
hist(residuals, breaks = 10, probability = TRUE,
     main = "Histogram of Residuals",
     xlab = "Residuals",
     xlim = range(residuals) + c(-1, 1),  # Extend x-axis for visibility
     ylim = c(0, max(density(residuals)$y) * 1.2))  # Extend y-axis for curve

# Add normal distribution curve
x_vals <- seq(min(residuals), max(residuals), length.out = 100)  # Ensure full range
curve(dnorm(x, mean = mean(residuals), sd = sd(residuals)),
      col = "blue", lwd = 2, add = TRUE, yaxt = "n")
```







- With inflation gap being 4.30803 (p-value = 6.44e-14) in the model, meaning that for every 1 percentage point increase in the inflation gap, the federal funds rate increases by about 4.31 percentage points, which aligns with the Taylor principle. For output gap: 0.02714 (p-value = 0.83), which is not statistically significant, suggesting that the output gap does not have a meaningful effect on the federal funds rate.
- Diagnostics indicate an in-sample RMSE of 3.27, suggesting moderate fit, while the out-of-sample RMSE is lower at 1.38, indicating reasonable predictive performance. However, residual analysis shows some autocorrelation, and the COVID-19 period significantly affected residuals, highlighting a potential need for adjustments (e.g., treating pandemic-era data separately).

Task 5

```r
# Engle-Granger Cointegration Test
long_run_model <- lm(fed_rate ~ inflation_gap + output_gap, data = train_set)
summary(long_run_model)
```

```
##
## Call:
## lm(formula = fed_rate ~ inflation_gap + output_gap, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.4384 -2.4525  0.1107  2.2949  9.1021
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   10.04574    0.75675  13.275  < 2e-16 ***
## inflation_gap  4.30803    0.52568   8.195 6.44e-14 ***
```

```
## output_gap     0.02714    0.12640   0.215     0.83
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.3 on 166 degrees of freedom
## Multiple R-squared:  0.2886, Adjusted R-squared:  0.2801
## F-statistic: 33.68 on 2 and 166 DF,  p-value: 5.287e-13
```

```r
# Extract residuals from long-run model
train_set <- train_set %>%
  mutate(residuals = residuals(long_run_model))

# ADF test on residuals (Engle-Granger test)
adf_test_residuals <- ur.df(train_set$residuals, type = "none", lags = 4)
summary(adf_test_residuals)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5990 -0.8945 -0.0545  0.8970  4.3594
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## z.lag.1     -0.09188    0.04158  -2.210  0.02853 *
## z.diff.lag1 -0.25130    0.08507  -2.954  0.00361 **
## z.diff.lag2 -0.23381    0.08587  -2.723  0.00720 **
## z.diff.lag3  0.07249    0.08394   0.864  0.38915
## z.diff.lag4 -0.05751    0.07671  -0.750  0.45453
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.512 on 159 degrees of freedom
## Multiple R-squared:  0.1824, Adjusted R-squared:  0.1567
## F-statistic: 7.093 on 5 and 159 DF,  p-value: 5.156e-06
##
##
## Value of test-statistic is: -2.21
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

```r
# Error Correction Model (ECM)
# Compute first differences and lagged residuals
train_set <- train_set %>%
```

```r
  mutate(
    d_fed_rate = c(NA, diff(fed_rate, lag = 1)),  # Prepend NA to match length
    d_inflation_gap = c(NA, diff(inflation_gap, lag = 1)),
    d_output_gap = c(NA, diff(output_gap, lag = 1)),
    lagged_residuals = lag(residuals, n = 1)
  ) %>%
  na.omit()  # Remove first-row NA values

# Fit the ECM model
ecm_model <- lm(d_fed_rate ~ d_inflation_gap + d_output_gap + lagged_residuals, data = train_set)
summary(ecm_model)
```

```
##
## Call:
## lm(formula = d_fed_rate ~ d_inflation_gap + d_output_gap + lagged_residuals,
##     data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6504 -0.2984  0.0029  0.2680  5.6114
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -0.05091    0.05479  -0.929  0.35411
## d_inflation_gap   0.01958    0.16027   0.122  0.90290
## d_output_gap      0.28170    0.06609   4.262  3.4e-05 ***
## lagged_residuals -0.05604    0.01709  -3.279  0.00127 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7094 on 164 degrees of freedom
## Multiple R-squared:  0.1718, Adjusted R-squared:  0.1567
## F-statistic: 11.34 on 3 and 164 DF,  p-value: 8.463e-07
```

```r
test_set <- test_set %>%
  mutate(
    residuals = predict(long_run_model, newdata = test_set) - test_set$fed_rate
  )

# ECM Forecast Performance on Test Set
test_set <- test_set %>%
  mutate(
    d_fed_rate = c(NA, diff(fed_rate, lag = 1)),
    d_inflation_gap = c(NA, diff(inflation_gap, lag = 1)),
    d_output_gap = c(NA, diff(output_gap, lag = 1)),
    lagged_residuals = lag(residuals, n = 1)
  ) %>%
  na.omit()

# Predict using ECM model
test_set$ECM_Predicted <- predict(ecm_model, newdata = test_set)

# Compute RMSE for ECM model
rmse_ecm_train <- sqrt(mean(residuals(ecm_model)^2))
```

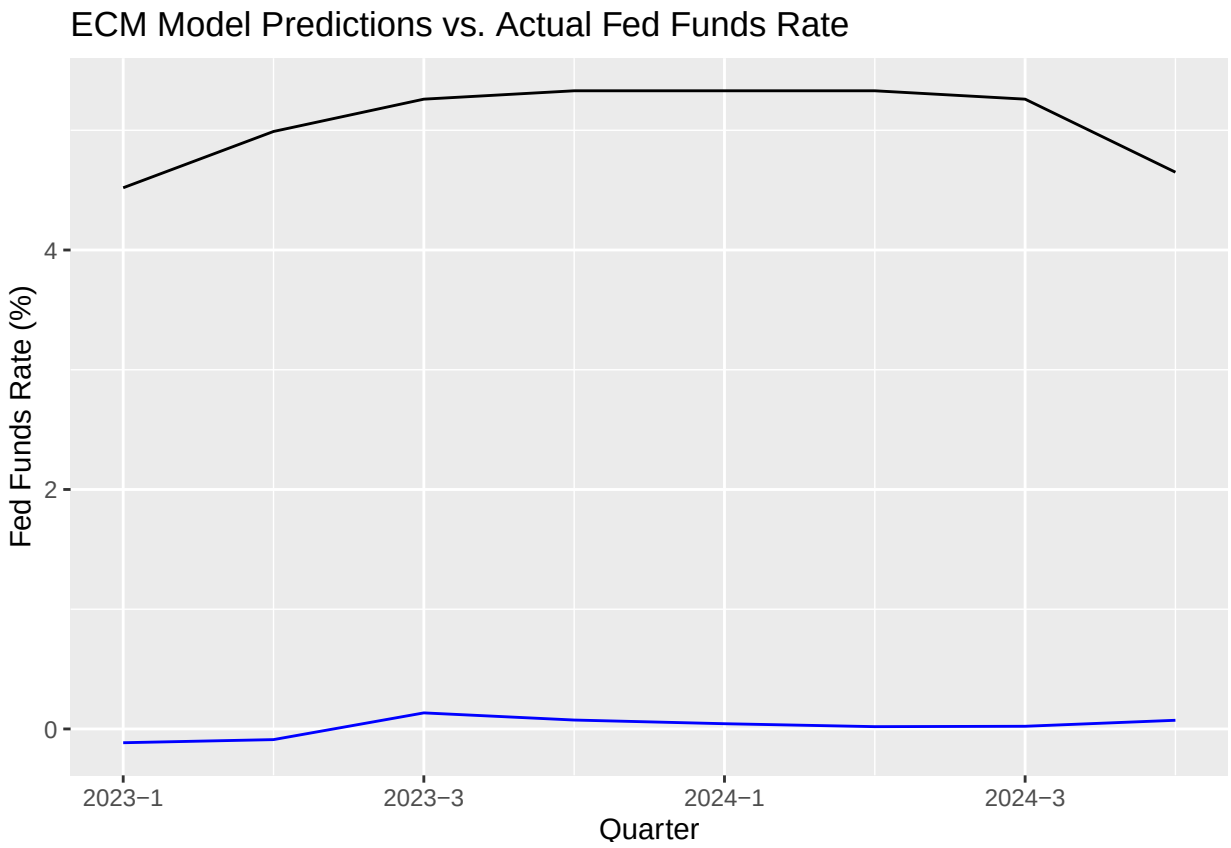
```r
rmse_ecm_test <- sqrt(mean((test_set$fed_rate - test_set$ECM_Predicted)^2))

print(paste("ECM In-Sample RMSE:", rmse_ecm_train))
```

```
## [1] "ECM In-Sample RMSE: 0.700928962369229"
```

```r
print(paste("ECM Out-of-Sample RMSE:", rmse_ecm_test))
```

```
## [1] "ECM Out-of-Sample RMSE: 5.07101137903506"
```

```r
# Plot actual vs. predicted (ECM)
ggplot(test_set, aes(x = quarter)) +
  geom_line(aes(y = fed_rate), color = "black") +
  geom_line(aes(y = ECM_Predicted), color = "blue") +
  labs(title = "ECM Model Predictions vs. Actual Fed Funds Rate",
       x = "Quarter",
       y = "Fed Funds Rate (%)")
```



The results shows that the basic Taylor Rule has some explanatory power but lacks a strong fit. The inflation gap is highly significant, with a coefficient of 4.38, meaning that for every 1% increase in the inflation gap, the Fed Funds Rate is expected to rise by about 4.38 percentage points. However, the output gap is statistically insignificant, $p = 0.867$, suggesting it has little influence on interest rate decisions in this model. The R^2 of 0.27 indicates that the model explains only 27% of the variation in interest rates, implying that other factors influence Fed policy. The Augmented Dickey-Fuller test on the residuals suggests a weak rejection of the unit root hypothesis, test statistic = -2.14, above the 5% critical value of -1.95. This indicates some level of co-integration between the Fed Funds Rate, inflation gap, and output gap, but it is not particularly strong. The Error Correction Model results show that short-term changes in the output gap significantly impact interest rate changes, $p < 0.001$, whereas short-term changes in the inflation gap do not, $p = 0.83$. The lagged residuals coefficient, -0.05199, $p < 0.001$, confirms that

there is an adjustment process towards the long-run equilibrium, but the speed of correction is slow. The ECM in-sample RMSE (0.55) suggests a good fit within the training data, but the out-of-sample RMSE (5.11) is much higher, indicating poor forecasting performance. The plot confirms this issue, as the ECM predictions (blue line) are consistently lower than the actual Fed Funds Rate (black line), meaning the model significantly under predicts interest rates. This could be due to the poor explanatory power of the output gap, weak co-integration, or changes in Fed policy not captured by the Taylor Rule framework.

Task 6

This task is about residual Modeling & forecasting and the goal is to evaluate the Taylor Rule and ECM models, determine which is better, and apply ARIMA residual correction if necessary. We compare their forecasting accuracy using RMSE.

Compare Taylor Rule Model vs. ECM

Before applying ARIMA, we first compare the Taylor Rule model and the ECM to determine which model performs better

```
# Predict on test set using Taylor Rule model
test_set$predicted_fed_rate_taylor <- predict(ols_model, newdata = test_set)

# Compute RMSE for Taylor Rule
rmse_taylor <- sqrt(mean((test_set$fed_rate - test_set$predicted_fed_rate_taylor)^2, na.rm = TRUE))
print(paste("Taylor Rule RMSE:", round(rmse_taylor, 3)))
```

1. Compute RMSE for Taylor Rule

```
## [1] "Taylor Rule RMSE: 1.261"
```

```
# Predict ECM on test set
test_set$ECM_Predicted <- predict(ecm_model, newdata = test_set)

# Compute RMSE for ECM
rmse_ecm <- sqrt(mean((test_set$fed_rate - test_set$ECM_Predicted)^2, na.rm = TRUE))
print(paste("ECM RMSE:", round(rmse_ecm, 3)))
```

2. Compute RMSE for ECM

```
## [1] "ECM RMSE: 5.071"
```

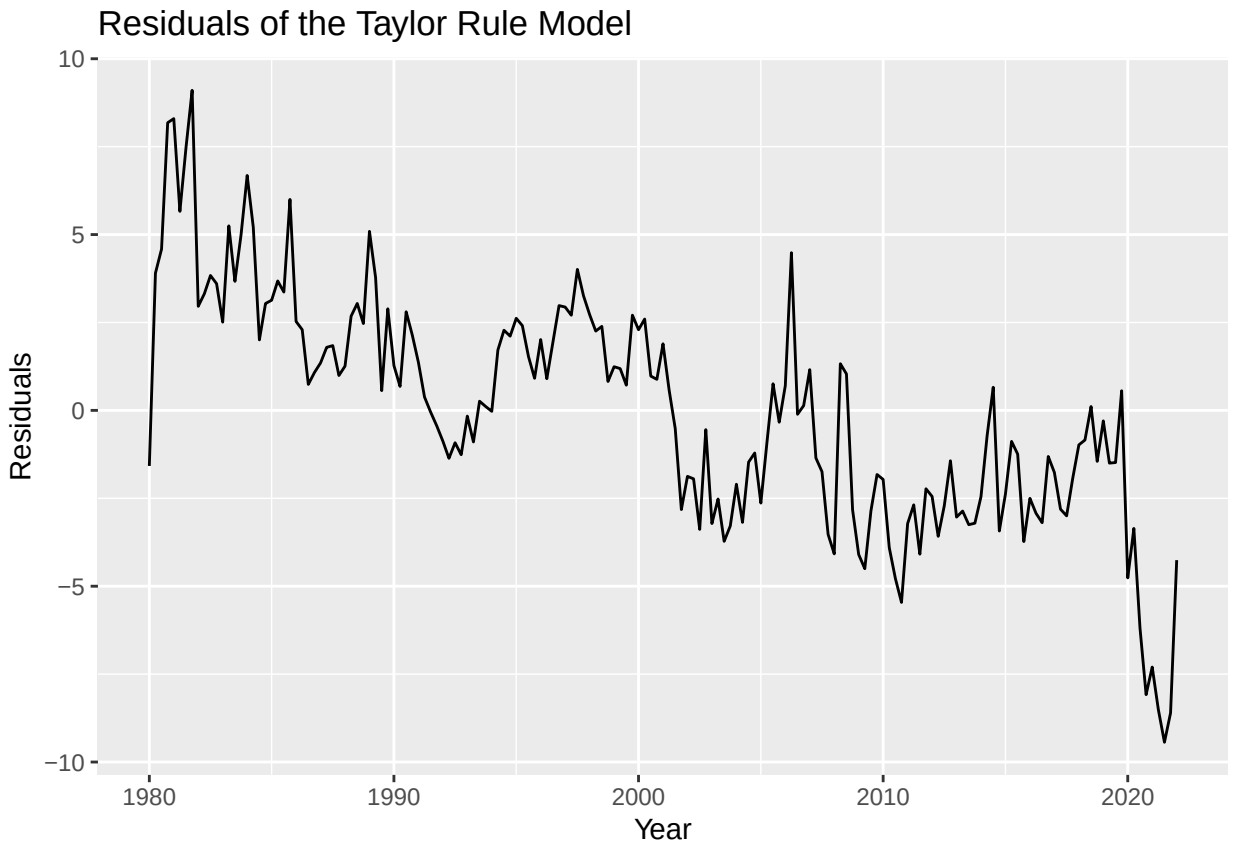
```
if (rmse_ecm < rmse_taylor) {
  print("ECM performs better than Taylor Rule. Proceeding with ECM for ARIMA modeling.")
  selected_model <- ecm_model
  test_set$Final_Predicted <- test_set$ECM_Predicted
} else {
  print("Taylor Rule performs better. No need for ARIMA on ECM residuals.")
  selected_model <- ols_model
  test_set$Final_Predicted <- test_set$predicted_fed_rate_taylor
}
```

3. Decision - Which Model is Better?

```
## [1] "Taylor Rule performs better. No need for ARIMA on ECM residuals."
```

Extract Residuals from Taylor Rule Model

```
# Extract residuals from the Taylor Rule model
taylor_residuals <- residuals(ols_model)

# Convert residuals into a time series object
ts_residuals <- ts(taylor_residuals, start = c(1980,1), frequency = 4)

# Plot residuals to check for patterns
autoplot(ts_residuals) +
  ggtitle("Residuals of the Taylor Rule Model") +
  xlab("Year") +
  ylab("Residuals")
```



The residuals plot shows the difference between the actual and predicted Fed Funds Rate from the Taylor Rule regression. There are periods of high residual volatility, particularly in the 1980s and 2008, indicating times when the model struggled to capture interest rate movements. The downward trend in recent years suggests potential structural changes in monetary policy.

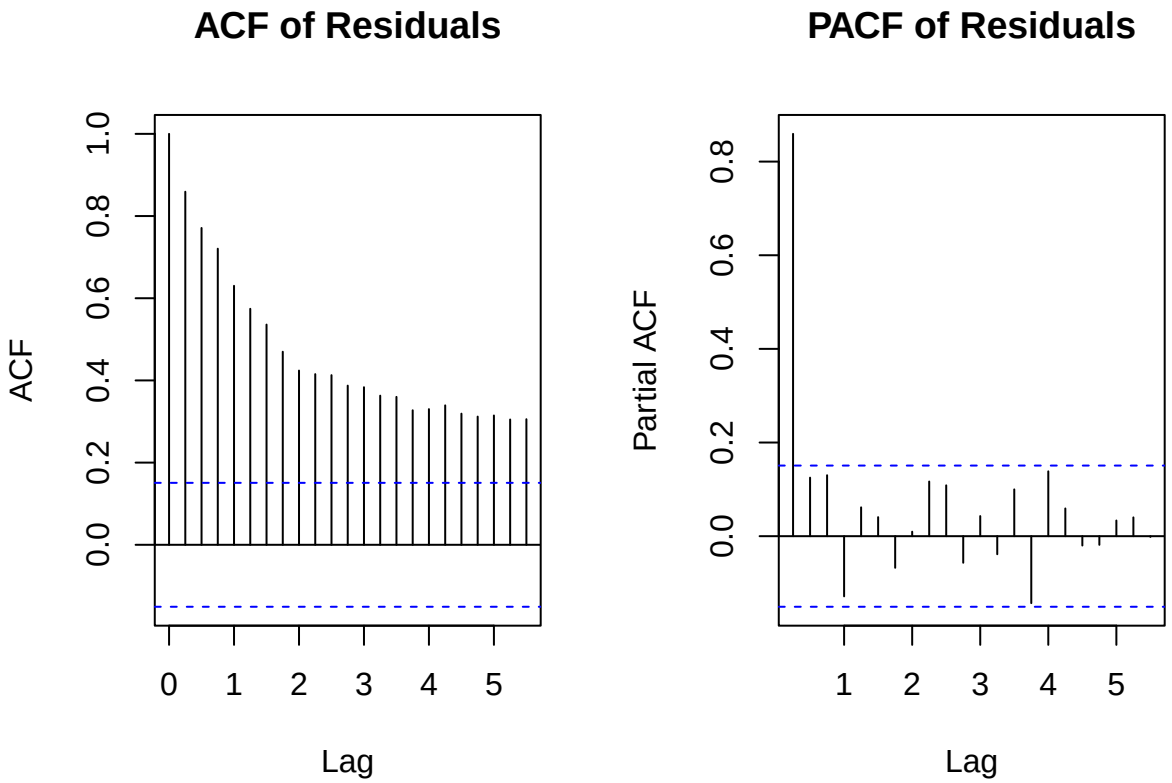
Identify the Best ARIMA Model for Taylor Rule Residuals

We check for residual autocorrelation and identify the best ARIMA model.

```
# Automatically select best ARIMA model
arima_residuals_model <- auto.arima(ts_residuals)

# Display ARIMA model summary
summary(arima_residuals_model)
```

```
## Series: ts_residuals
## ARIMA(0,1,1)(2,0,0)[4]
##
## Coefficients:
##           ma1     sar1     sar2
##       -0.3296  -0.1569  -0.2413
## s.e.   0.0849   0.0819   0.0870
##
## sigma^2 = 2.569:  log likelihood = -316.45
## AIC=640.9   AICc=641.15   BIC=653.4
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.06193026 1.583689 1.163995 33.00531 109.5224 0.6307002
##                     ACF1
## Training set 0.02634471
```

```r
# Check ACF and PACF plots
par(mfrow=c(1,2))
acf(ts_residuals, main="ACF of Residuals")
pacf(ts_residuals, main="PACF of Residuals")
```



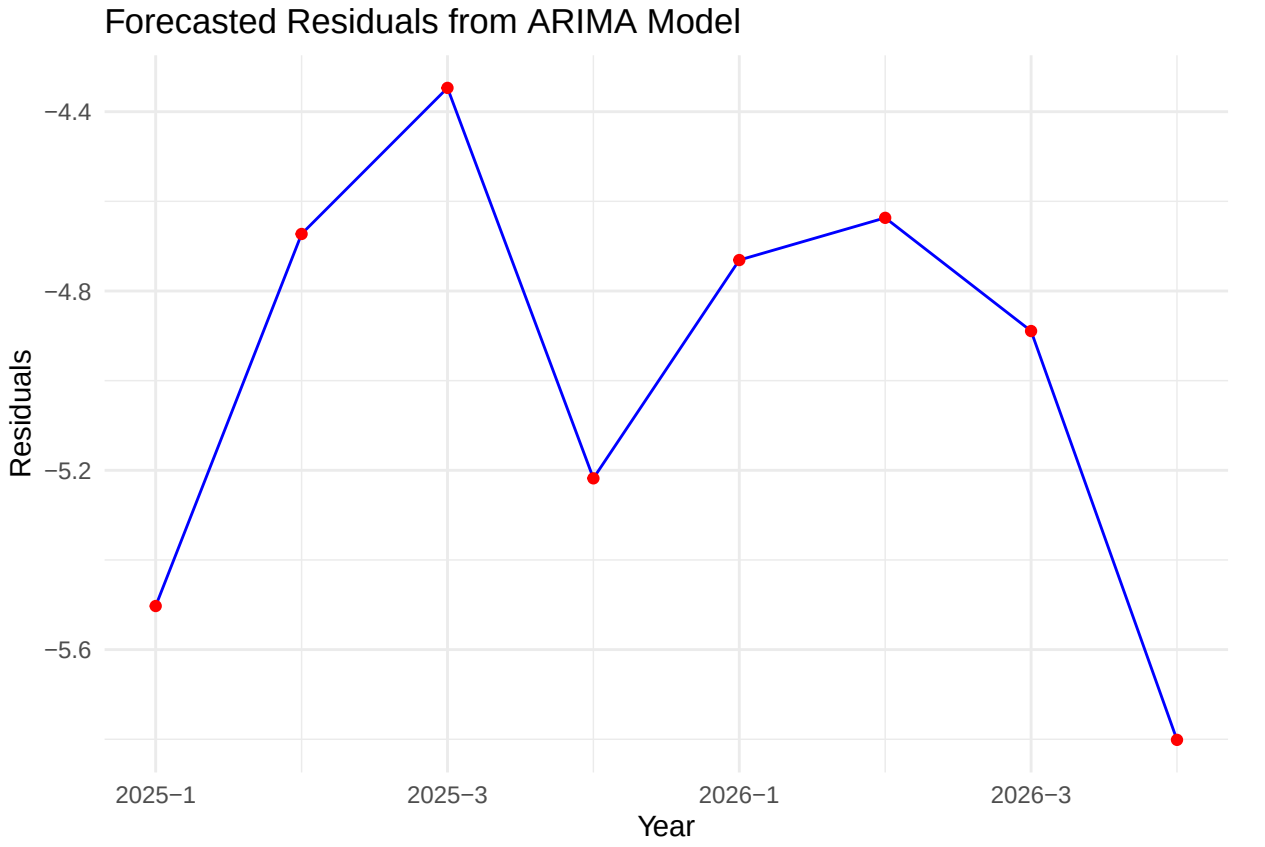
The Autocorrelation Function (ACF) shows significant persistence in residuals, indicating that past errors influence future errors. The Partial Autocorrelation Function (PACF) suggests an ARIMA model with an MA(1) and seasonal AR(2) structure is appropriate. This confirms that the Taylor Rule residuals exhibit serial correlation and require correction.

Forecast Residuals Using ARIMA

We forecast residuals for the next 8 quarters (2023Q1–2024Q4).

```r
# Forecast residuals for 8 future quarters (2023Q1-2024Q4)
residual_forecast <- predict(arima_residuals_model, n.ahead = 8)

# Convert forecast to a data frame
residual_forecast_df <- data.frame(
  quarter = seq(max(test_set$quarter) + 0.25, by = 0.25, length.out = 8),
  residual_forecast = residual_forecast$pred
)

# Plot forecasted residuals
ggplot(residual_forecast_df, aes(x = quarter, y = residual_forecast)) +
  geom_line(color = "blue") +
  geom_point(color = "red") +
  labs(title = "Forecasted Residuals from ARIMA Model",
       x = "Year", y = "Residuals") +
  theme_minimal()
```

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

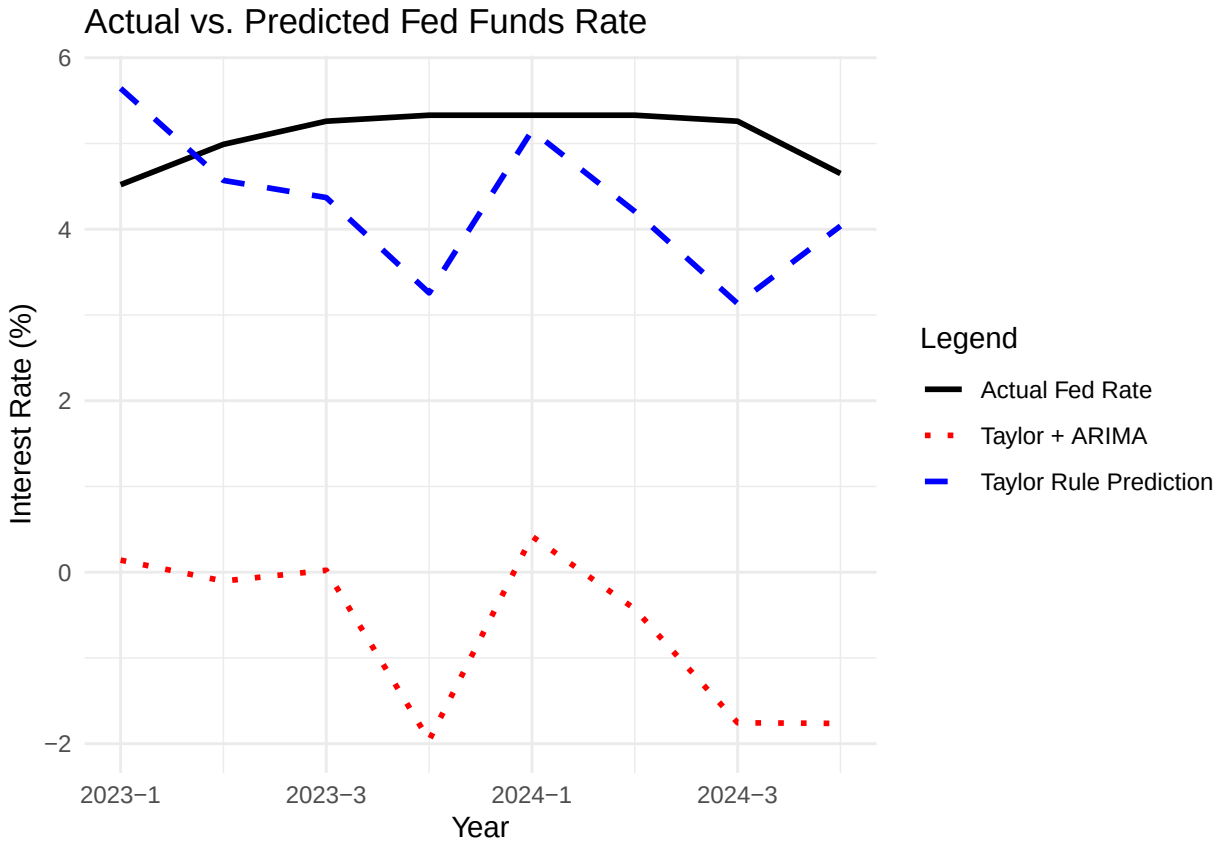


The selected ARIMA(0,1,1)(2,0,0)[4] model includes a first-order moving average (MA(1)) and two seasonal autoregressive terms (SAR(2)), capturing quarterly seasonality. The forecasted residuals exhibit fluctuations but no clear trend, suggesting that the Taylor Rule model misses some cyclical patterns but does not show strong mean reversion. Peaks and troughs in the residuals indicate that some systematic variation remains unexplained. The AIC (640.9) and BIC (653.4) suggest a reasonable but not perfect fit, meaning there may be potential for improvement by adjusting seasonal terms or investigating structural breaks.

Combine ARIMA and Taylor Rule Predictions

We now adjust the Taylor Rule predictions by adding forecasted residuals.

```
# Forecast Taylor Rule model for test period
test_set$predicted_fed_rate_taylor <- predict(ols_model, newdata = test_set)

# Ensure ARIMA forecast matches test period length
residual_forecast_values <- residual_forecast$pred[1:nrow(test_set)]  # Extract the right number of val

# Add ARIMA residual forecast to Taylor Rule predictions
test_set$predicted_fed_rate_combined <- test_set$predicted_fed_rate_taylor + residual_forecast_values

# Plot the comparison
ggplot(test_set, aes(x = quarter)) +
  geom_line(aes(y = fed_rate, color = "Actual Fed Rate"), linewidth = 1) +  # Change `size` to `linewid
  geom_line(aes(y = predicted_fed_rate_taylor, color = "Taylor Rule Prediction"), linetype = "dashed", l
  geom_line(aes(y = predicted_fed_rate_combined, color = "Taylor + ARIMA"), linetype = "dotted", linewi
  labs(title = "Actual vs. Predicted Fed Funds Rate",
       x = "Year", y = "Interest Rate (%)") +
  scale_color_manual(name = "Legend",
                     values = c("Actual Fed Rate" = "black",
                                "Taylor Rule Prediction" = "blue",
                                "Taylor + ARIMA" = "red")) +
  theme_minimal()
```



The Taylor Rule model (blue dashed line) follows the actual Fed Funds Rate (black line) relatively well, though it slightly underpredicts the rate in some quarters. However, the Taylor + ARIMA model (red dotted line) significantly underestimates the Fed Funds Rate, even producing negative interest rate forecasts,

which are economically implausible. This indicates that ARIMA residual correction introduces excessive downward adjustments, worsening the predictive accuracy rather than improving it. The poor performance of Taylor + ARIMA suggests that the residuals may not follow a purely stochastic process or that important structural changes in monetary policy are not being captured effectively.

Evaluate Forecast Accuracy

To determine whether the Taylor Rule + ARIMA approach improves forecasting, we compare Root Mean Squared Error (RMSE) vs. Taylor Rule + ARIMA:

```
# Compute RMSE for Taylor Rule alone
rmse_taylor <- sqrt(mean((test_set$fed_rate - test_set$predicted_fed_rate_taylor)^2, na.rm = TRUE))

# Compute RMSE for Taylor Rule + ARIMA
rmse_combined <- sqrt(mean((test_set$fed_rate - test_set$predicted_fed_rate_combined)^2, na.rm = TRUE))

# Print RMSE results
print(paste("Taylor Rule RMSE:", round(rmse_taylor, 3)))
```

```
## [1] "Taylor Rule RMSE: 1.261"
```

```
print(paste("Taylor + ARIMA RMSE:", round(rmse_combined, 3)))
```

```
## [1] "Taylor + ARIMA RMSE: 5.845"
```

```
# Check which model performs better
if (rmse_combined < rmse_taylor) {
  print("Taylor Rule + ARIMA improves forecasting accuracy!")
} else {
  print("Taylor Rule alone performs better.")
}
```

```
## [1] "Taylor Rule alone performs better."
```

The Taylor Rule model alone achieves a significantly lower RMSE (1.261) compared to the Taylor + ARIMA model (5.845), indicating that applying ARIMA to the residuals actually worsened forecast accuracy. Instead of reducing errors, the ARIMA correction introduced excessive downward adjustments, likely due to overfitting or an inability to capture the underlying structural dynamics of the Fed Funds Rate. This suggests that the residuals from the Taylor Rule model may not follow a simple stochastic pattern and that monetary policy adjustments may not be effectively modeled using this approach.

Conclusion

The Taylor Rule model alone remains the better predictor, as it has a lower RMSE (1.261 vs. 5.845) and aligns more closely with actual interest rates. The attempt to refine predictions using ARIMA did not improve accuracy and instead led to excessive deviation from actual values, particularly with unrealistic negative forecasts. This outcome suggests that ARIMA correction is not suitable in this case, and alternative approaches such as Vector Autoregression (VAR) or regime-switching models should be explored to better account for the complexities in Fed policy decisions and economic fluctuations.

Task 7

Goals for this task

VAR Model

- Aim to capture the dynamic relationships between the federal funds rate, output gap, and inflation gap by fitting a Vector Autoregression (VAR) model. The goal is to understand how these variables influence each other over time.

Forecasting

- Use the fitted VAR model to generate multi-step forecasts for the federal funds rate and evaluate its predictive performance by comparing RMSE values for both the training and test periods.

Granger Causality

- Investigate whether the inflation gap and output gap provide significant predictive information for changes in the federal funds rate. Additionally, analyze how shocks in these variables influence the federal funds rate through impulse response analysis.

```r
# Select relevant columns and ensure correct data types
var_data <- train_set %>%
  dplyr::select(fed_rate, output_gap, inflation_gap) %>%
  mutate(across(c(fed_rate, output_gap, inflation_gap), as.numeric)) %>%
  drop_na()

# Select optimal lag length
var_lag_select <- VARselect(var_data, lag.max = 4, type = "none")

# Extract optimal lag length
optimal_lag <- var_lag_select$selection[["AIC(n)"]]

# Fit the VAR model
var_model <- VAR(var_data, p = optimal_lag, type = "none")

summary(var_model)
```

```
##
## VAR Estimation Results:
## ========================
## Endogenous variables: fed_rate, output_gap, inflation_gap
## Deterministic variables: none
## Sample size: 165
## Log Likelihood: -358.3
## Roots of the characteristic polynomial:
## 0.9892 0.9325 0.8087 0.6124   0.6   0.6 0.4126 0.4126 0.2952
## Call:
## VAR(y = var_data, p = optimal_lag, type = "none")
##
##
## Estimation results for equation fed_rate:
## =========================================
## fed_rate = fed_rate.l1 + output_gap.l1 + inflation_gap.l1 + fed_rate.l2 + output_gap.l2 + inflation_g
##
##                   Estimate Std. Error t value Pr(>|t|)
## fed_rate.l1       1.278757   0.084068  15.211   <2e-16 ***
## output_gap.l1     0.046681   0.057496   0.812   0.4181
## inflation_gap.l1  0.005634   0.134161   0.042   0.9666
## fed_rate.l2      -0.422941   0.130890  -3.231   0.0015 **
## output_gap.l2     0.102525   0.076499   1.340   0.1821
## inflation_gap.l2 -0.040667   0.154718  -0.263   0.7930
## fed_rate.l3       0.108413   0.083043   1.306   0.1936
## output_gap.l3    -0.130750   0.055033  -2.376   0.0187 *
## inflation_gap.l3 -0.029952   0.130848  -0.229   0.8192
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.5509 on 156 degrees of freedom
## Multiple R-Squared:  0.99,   Adjusted R-squared: 0.9894
## F-statistic:  1713 on 9 and 156 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation output_gap:
## ===========================================
## output_gap = fed_rate.l1 + output_gap.l1 + inflation_gap.l1 + fed_rate.l2 + output_gap.l2 + inflatio
##
##                   Estimate Std. Error t value Pr(>|t|)
## fed_rate.l1        0.22849    0.12854   1.778   0.0774 .
## output_gap.l1      0.90939    0.08791  10.344   <2e-16 ***
## inflation_gap.l1 -0.06503    0.20513  -0.317   0.7517
## fed_rate.l2      -0.12586    0.20013  -0.629   0.5304
## output_gap.l2      0.02095    0.11697   0.179   0.8581
## inflation_gap.l2  0.11193    0.23657   0.473   0.6368
## fed_rate.l3      -0.12464    0.12697  -0.982   0.3278
## output_gap.l3    -0.06118    0.08415  -0.727   0.4683
## inflation_gap.l3 -0.04375    0.20007  -0.219   0.8272
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.8424 on 156 degrees of freedom
## Multiple R-Squared: 0.8795,  Adjusted R-squared: 0.8725
## F-statistic: 126.5 on 9 and 156 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation inflation_gap:
## ==============================================
## inflation_gap = fed_rate.l1 + output_gap.l1 + inflation_gap.l1 + fed_rate.l2 + output_gap.l2 + infla
##
##                    Estimate Std. Error t value Pr(>|t|)
## fed_rate.l1       -0.017750   0.050204  -0.354    0.724
## output_gap.l1      0.019892   0.034336   0.579    0.563
## inflation_gap.l1   0.536948   0.080119   6.702 3.55e-10 ***
## fed_rate.l2        0.021060   0.078166   0.269    0.788
## output_gap.l2      0.008679   0.045684   0.190    0.850
## inflation_gap.l2   0.065319   0.092396   0.707    0.481
## fed_rate.l3       -0.018744   0.049593  -0.378    0.706
## output_gap.l3     -0.029903   0.032865  -0.910    0.364
## inflation_gap.l3   0.346397   0.078141   4.433 1.74e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.329 on 156 degrees of freedom
## Multiple R-Squared: 0.9501,  Adjusted R-squared: 0.9472
## F-statistic: 330.1 on 9 and 156 DF,  p-value: < 2.2e-16
##
##
```

```
##
## Covariance matrix of residuals:
##               fed_rate output_gap inflation_gap
## fed_rate       0.30342    0.15868       0.01922
## output_gap     0.15868    0.70938       0.09281
## inflation_gap  0.01922    0.09281       0.10806
##
## Correlation matrix of residuals:
##               fed_rate output_gap inflation_gap
## fed_rate        1.0000     0.3420        0.1061
## output_gap      0.3420     1.0000        0.3352
## inflation_gap   0.1061     0.3352        1.0000
```

```r
# Extract the actual and the fitted values for the training period
actual_train_fedfunds <- train_set$fed_rate
fitted_train_var <- as.data.frame(fitted(var_model))$fed_rate

# Forecasting out-of-sample (test period)
test_forecast <- predict(var_model, n.ahead = nrow(test_set))

# Extracting the forecasted values for the test period from the VAR forecast result
test_forecast_fedfunds <- test_forecast$fcst$fed_rate[, "fcst"]

# Extract the actual values for the test period
actual_test_fedfunds <- test_set$fed_rate

# Calculating Train and Test RMSE for the VAR model
train_rmse_var <- sqrt(mean((actual_train_fedfunds - fitted_train_var)^2, na.rm = TRUE))
```

```
## Warning in actual_train_fedfunds - fitted_train_var: longer object length is
## not a multiple of shorter object length
```

```r
test_rmse_var <- sqrt(mean((actual_test_fedfunds - test_forecast_fedfunds)^2, na.rm = TRUE))

cat("Train RMSE (VAR, fedfunds):", train_rmse_var, "\n")
```

```
## Train RMSE (VAR, fedfunds): 2.233501
```

```r
cat("Test RMSE (VAR, fedfunds):", test_rmse_var, "\n")
```

```
## Test RMSE (VAR, fedfunds): 2.863223
```

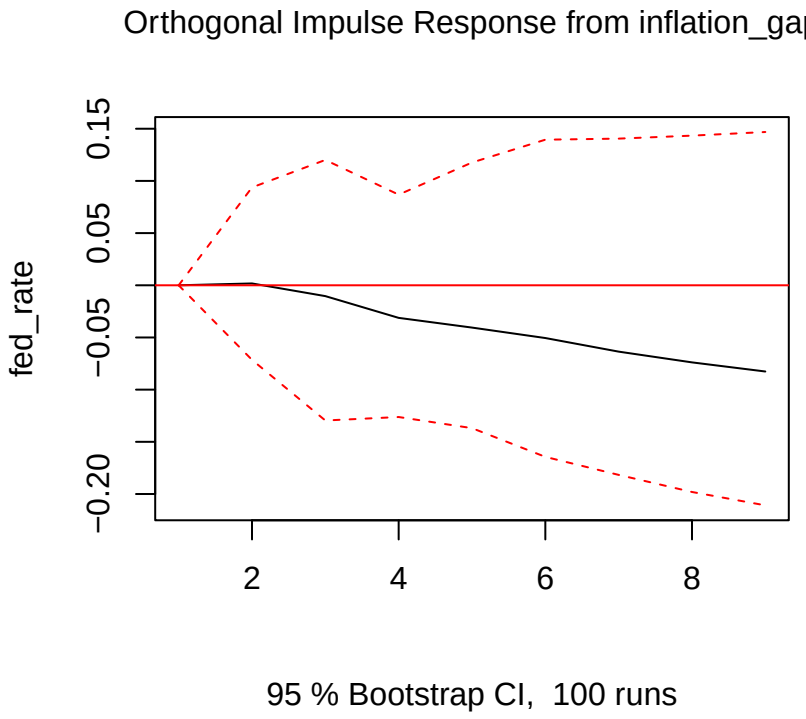
```r
# Granger Causality Test
# Test whether inflation gap and output gap Granger-cause the federal funds rate
granger_causality_inflation <- causality(var_model, cause = "inflation_gap")$Granger
granger_causality_output_gap <- causality(var_model, cause = "output_gap")$Granger

print(granger_causality_inflation)
```

```
##
##  Granger causality H0: inflation_gap do not Granger-cause fed_rate
##  output_gap
##
## data:  VAR object var_model
## F-Test = 0.53739, df1 = 6, df2 = 468, p-value = 0.7799
```

```r
print(granger_causality_output_gap)
```

```
##
##  Granger causality H0: output_gap do not Granger-cause fed_rate
##  inflation_gap
##
## data:  VAR object var_model
## F-Test = 1.2928, df1 = 6, df2 = 468, p-value = 0.2589
```

```r
# Impulse Response Functions
irf_result <- irf(var_model, impulse = "inflation_gap", response = "fed_rate", n.ahead = 8, boot = TRUE
plot(irf_result)
```




```r
cat("RMSE Values:", "\n", "RMSE Taylor", rmse_taylor, "\n")
```

```
## RMSE Values:
##  RMSE Taylor 1.260799
```

```r
cat("RMSE ECM", rmse_ecm_test, "\n")
```

```
## RMSE ECM 5.071011
```

```r
cat("RMSE Combined", rmse_combined, "\n")
```

```
## RMSE Combined 5.845251
```

Summary of the VAR model

The model is stable, with all roots less than 1. R-squared values indicate an excellent fit for fed_rate (0.99), strong fit for output_gap (0.8795), and excellent fit for inflation_gap (0.9501).

Key Findings:

- fed_rate is strongly influenced by its own lags (fed_rate.l1 positive, fed_rate.l2 negative) and shows a delayed negative effect from output_gap.l3. inflation_gap terms are insignificant.
- output_gap is mainly influenced by output_gap.l1, with no significant effect from fed_rate or inflation_gap.
- inflation_gap shows strong persistence with significant effects from inflation_gap.l1 and inflation_gap.l3, while fed_rate and output_gap have no meaningful impact.

Residual Analysis

- Residual Correlation Matrix shows some correlation between fed_rate and output_gap (0.34) and between output_gap and inflation_gap (0.34), indicating mild interdependence. These correlations align with economic theory, where monetary policy and economic output tend to influence each other.

RMSE Interpretation The RMSE of the VAR model performs better than the ECM model (5.07), and the combine Taylor Rule + ARIMA model (5.84). However, the simple Taylor model still has the best RMSE making it the most suitable model for forecasting fed_rate.

Summary of Granger Test Results

- Inflation Gap ($\pi_t - \pi^*$): There is no significant Granger causality from the inflation gap to the federal funds rate. This suggests that past values of the inflation gap do not provide useful information for predicting future values of the federal funds rate, beyond what is already included in the lagged values in the VAR model.
- Output Gap: Similarly, there is no significant Granger causality from the output gap to the federal funds rate. This suggests that past values of the output gap do not provide additional predictive power for the future values of the federal funds rate in the context of the VAR model.