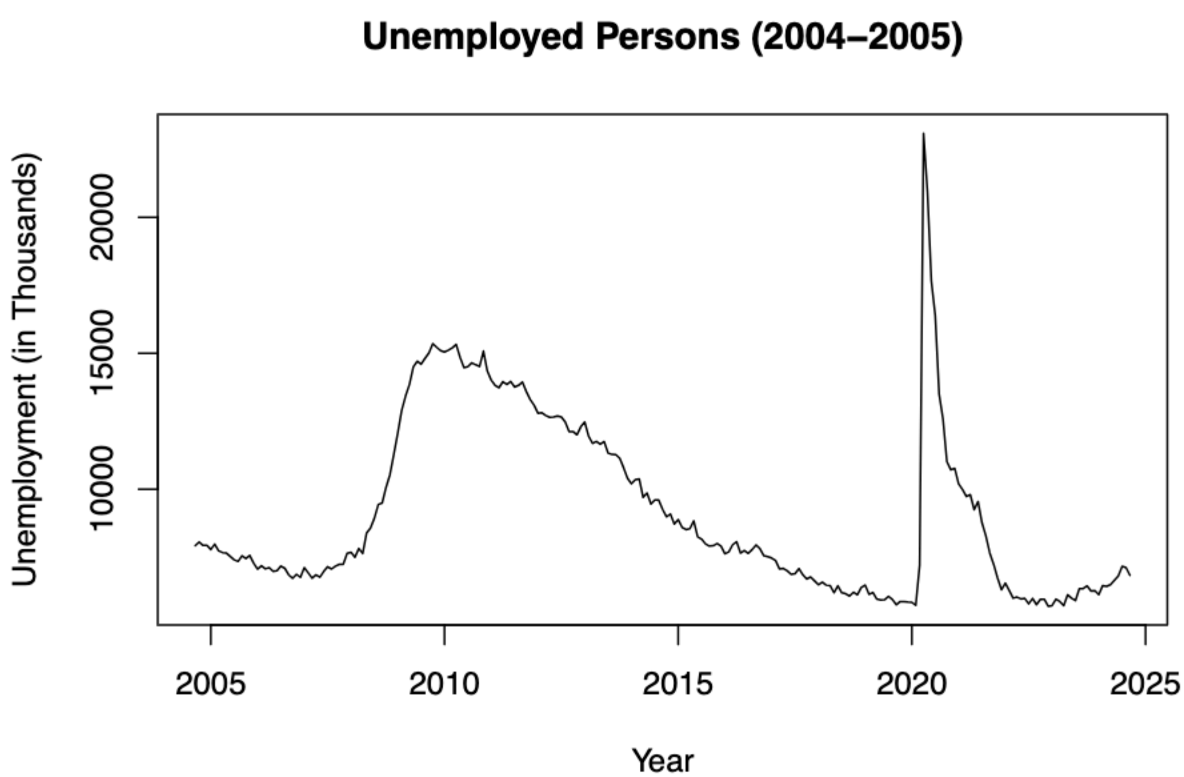
ECON 144 HW 3

Leonard Zhu

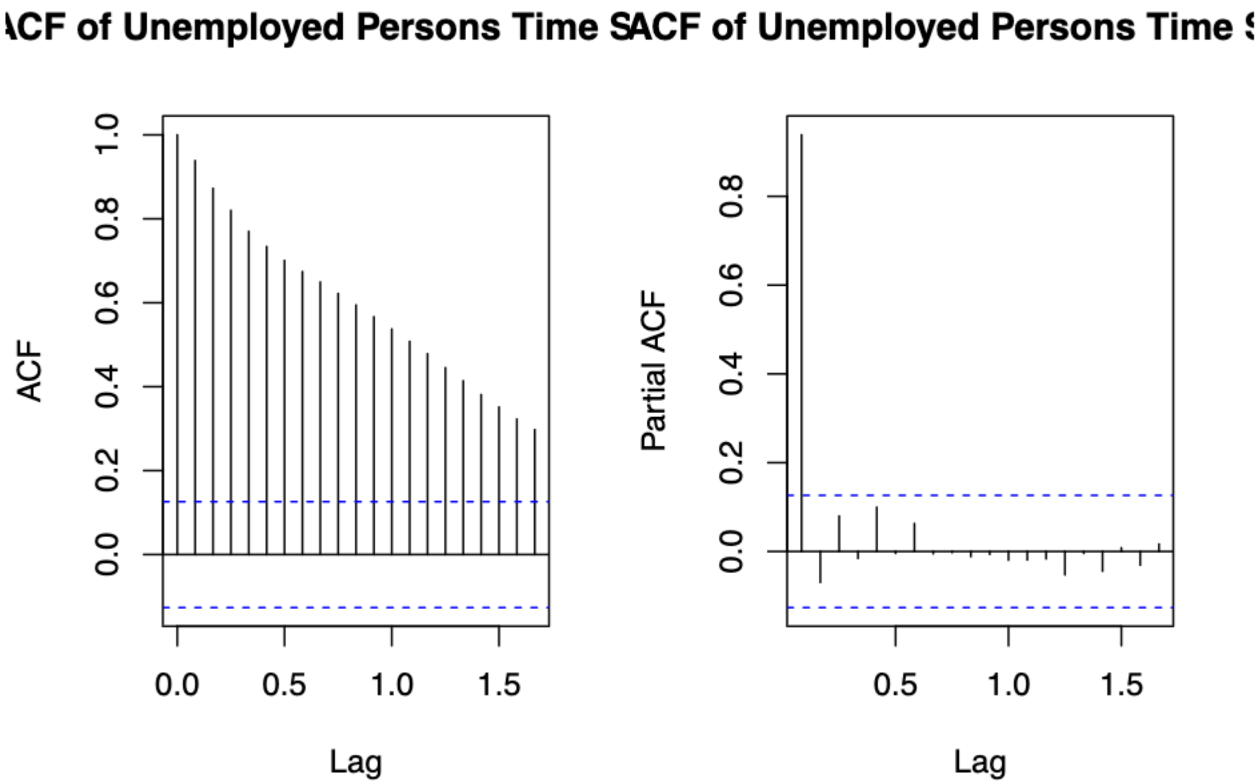
2024-10-31

Problem 7.2

```r
# Load the data
unemployment_data <- read_excel("~/Downloads/unemploy.xlsx")

# Convert the data to a time series object (monthly frequency starting from first date in dataset)
unemployment_ts <- ts(unemployment_data$`Unemployment (in Thousands)`,
                      start = c(2004, 9), frequency = 12)

# Plot the time series
plot(unemployment_ts, main = "Unemployed Persons (2004-2005)", ylab = "Unemployment (in Thousands)", xla
```



```r
par(mfrow=c(1,2))
# Calculate and plot the ACF
acf_values <- acf(unemployment_ts, main = "ACF of Unemployed Persons Time Series", lag.max = 20)

# Calculate and plot the PACF
pacf_values <- pacf(unemployment_ts, main = "PACF of Unemployed Persons Time Series", lag.max = 20)
```




```r
# Fit an AR model to the time series data
ar_model <- arima(unemployment_ts, order = c(1, 0, 0))
summary(ar_model)
```

```
##
## Call:
## arima(x = unemployment_ts, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.9372   8949.647
## s.e.  0.0214   1064.070
##
## sigma^2 estimated as 1205216:  log likelihood = -2030.28,  aic = 4066.56
##
## Training set error measures:
##                    ME     RMSE      MAE        MPE     MAPE     MASE       ACF1
## Training set 6.753468 1097.823 350.4473 -0.8500008 3.430979 1.112781 0.07075031
```

Problem 7.5

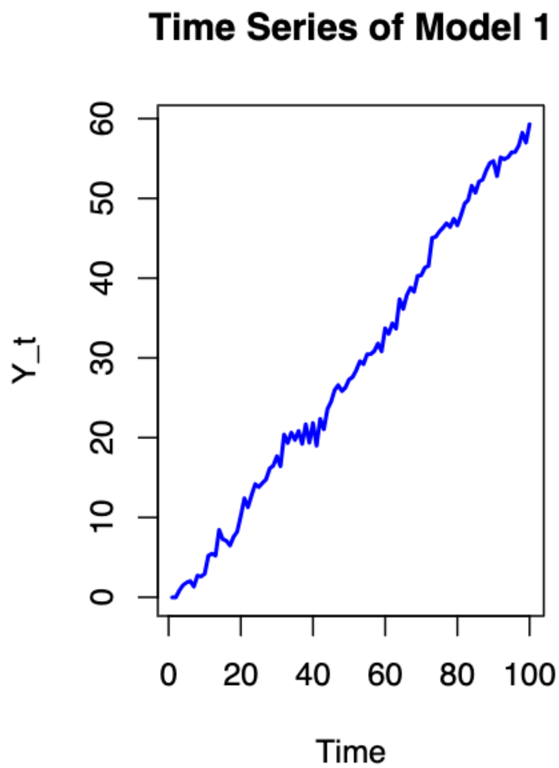
In this analysis, we simulate two autoregressive (AR) processes of order 2 (AR(2)). Each process is defined by different parameters for the lagged terms. We will compare the time series behavior, autocorrelation functions (ACFs), and covariance-stationarity properties of each process.

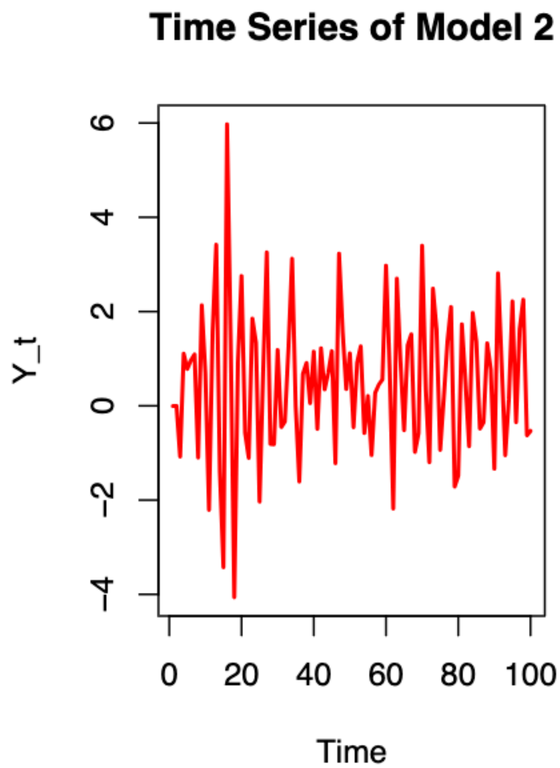
```r
# Define parameters
n <- 100  # Number of observations
y1 <- y2 <- numeric(n)
y1[1:2] <- y2[1:2] <- 0  # Initialize with zeros

# Simulate Model 1
for (t in 3:n) {
  y1[t] <- 1 + 0.3 * y1[t - 1] + 0.7 * y1[t - 2] + rnorm(1, mean = 0, sd = 1)
}

# Simulate Model 2
for (t in 3:n) {
  y2[t] <- 1 - 0.3 * y2[t - 1] - 0.7 * y2[t - 2] + rnorm(1, mean = 0, sd = 1)
}
```

The first model has positive coefficients for the lag terms, suggesting a persistence or positive feedback in the series. In contrast, the second model has negative coefficients, which might indicate oscillations or mean reversion in the process. We expect the differences in lag signs to manifest in distinct time series patterns and autocorrelation structures.

The time series plots provide a visual understanding of each model's behavior over time. By plotting them together, we can observe the effect of positive vs. negative coefficients.

```r
par(mfrow=c(1, 2))  # Two side-by-side plots

# Plot Model 1
plot(y1, type = "l", col = "blue", main = "Time Series of Model 1",
     ylab = "Y_t", xlab = "Time", lwd = 2)

# Plot Model 2
plot(y2, type = "l", col = "red", main = "Time Series of Model 2",
     ylab = "Y_t", xlab = "Time", lwd = 2)
```



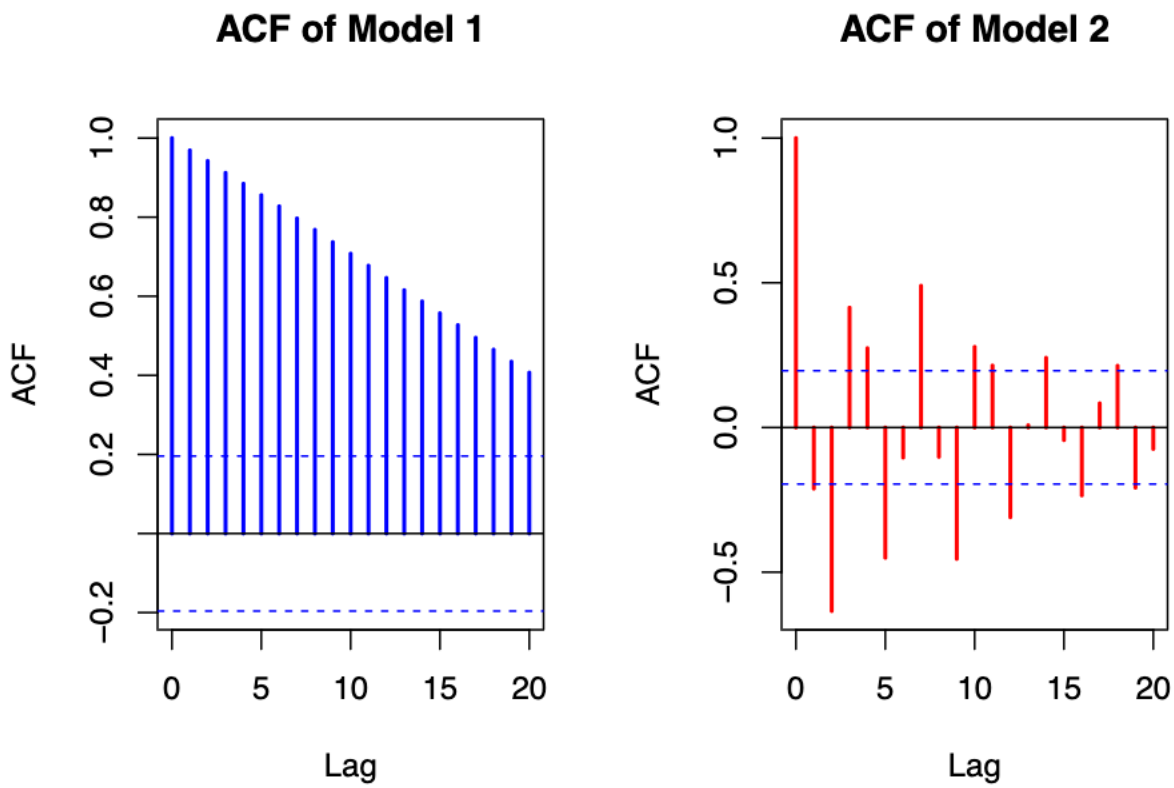


For Model 1, we expect a smoother series with a persistent effect due to the positive coefficients. For Model 2, the negative coefficients might cause oscillatory behavior, as the process could overshoot its mean in each step.

Analysis of ACFs

The ACF plots help us understand how past values of the series are related to current values. Positive autocorrelation is expected in Model 1 due to the positive lag coefficients, while alternating signs in the ACF might appear for Model 2, where lag terms have negative coefficients.

```r
par(mfrow=c(1, 2))

# ACF of Model 1
acf(y1, main = "ACF of Model 1", col = "blue", lwd = 2)

# ACF of Model 2
acf(y2, main = "ACF of Model 2", col = "red", lwd = 2)
```



- **Time Series Comparison:**
  - **Model 1**: The time series of Model 1, with positive coefficients for the lag terms, exhibits a smoother, more persistent trajectory. The positive feedback implies that each value depends positively on past values, contributing to gradual changes and higher persistence. This behavior may be typical in systems where shocks or trends carry over time.
  - **Model 2**: In contrast, Model 2, with negative coefficients, displays an oscillatory pattern. The alternating sign of coefficients causes the series to overshoot around its mean, creating a "mean-reverting" or cyclic behavior. This type of series is dynamic and prone to rapid reversals, which could be useful in modeling phenomena with natural oscillations.
- **Autocorrelation Function (ACF) Comparison:**
  - **Model 1**: The ACF of Model 1 decays gradually, indicating positive autocorrelation due to the series' tendency to follow recent values closely. This slower decay suggests persistence in past influences, aligning with covariance-stationary properties.
  - **Model 2**: The ACF of Model 2, however, alternates in sign and decays relatively faster. This behavior reflects the process's oscillatory nature, with each value moving in the opposite direction of previous values. Such patterns often indicate strong mean reversion and reduced persistence over time.
- **Covariance-Stationarity**:
  - Both models satisfy covariance-stationarity criteria, as their AR(2) parameters lie within stability bounds. This implies that each process has a stable mean and finite variance, and the relationships between observations remain consistent over time.

These two models exemplify how varying AR(2) coefficients influence the dynamics of a time series, creating either persistence or oscillation. This can inform different analytical applications based on the nature of the observed data.

Problem 7.6

```
# Load CPI data from Excel files
food_data <- read_excel("~/Downloads/fooddata.xlsx")
transportation_data <- read_excel("~/Downloads/transportationdata.xlsx")
housing_data <- read_excel("~/Downloads/housingdata.xlsx")

# Assign column names based on the provided organization
colnames(food_data) <- c("Year", "Period", "Label", "Observation_Value")
colnames(transportation_data) <- c("Year", "Period", "Label", "Observation_Value")
colnames(housing_data) <- c("Year", "Period", "Label", "Observation_Value")

# Preview the data structure
head(food_data)
```

```
## # A tibble: 6 x 4
##   Year  Period Label    Observation_Value
##   <chr> <chr>  <chr>                <dbl>
## 1 1999  M01    1999 Jan              164.
## 2 1999  M02    1999 Feb              164.
## 3 1999  M03    1999 Mar              164.
## 4 1999  M04    1999 Apr              164.
## 5 1999  M05    1999 May              164.
## 6 1999  M06    1999 Jun              164.
```

```
head(transportation_data)
```

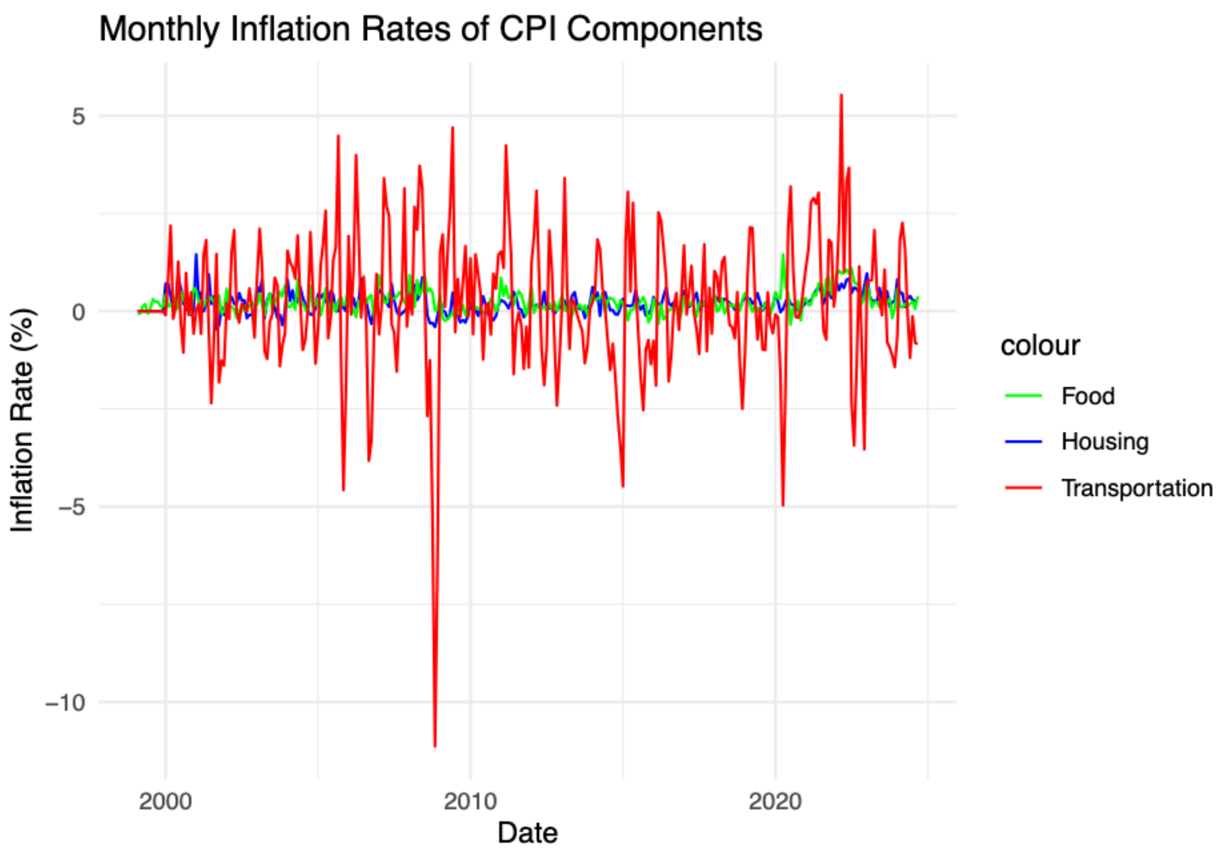
```
## # A tibble: 6 x 4
##   Year  Period Label    Observation_Value
##   <chr> <chr>  <chr>                <dbl>
## 1 1999  M12    1999 Dec             100
## 2 2000  M01    2000 Jan              99.9
## 3 2000  M02    2000 Feb             101.
## 4 2000  M03    2000 Mar             103
## 5 2000  M04    2000 Apr             103.
## 6 2000  M05    2000 May             103.
```

```
head(housing_data)
```

```
## # A tibble: 6 x 4
##   Year  Period Label    Observation_Value
##   <chr> <chr>  <chr>                <dbl>
## 1 1999  M12    1999 Dec              100
## 2 2000  M01    2000 Jan              101.
## 3 2000  M02    2000 Feb              101.
## 4 2000  M03    2000 Mar              102.
## 5 2000  M04    2000 Apr              102.
## 6 2000  M05    2000 May              102.
```

```r
# Define a function to process each dataset
process_cpi_data <- function(data) {
  data %>%
    mutate(
      Date = as.Date(paste(Year, substr(Period, 2, 3), "01", sep = "-"), format = "%Y-%m-%d"),
      Index = Observation_Value
    ) %>%
    arrange(Date) %>%
    mutate(
      inflation = (Index - lag(Index)) / lag(Index) * 100
    ) %>%
    select(Date, inflation) %>%
    na.omit()
}

# Process each dataset for inflation rates
food_data_processed <- process_cpi_data(food_data)
transportation_data_processed <- process_cpi_data(transportation_data)
housing_data_processed <- process_cpi_data(housing_data)
```

```r
# Use full_join to combine dataframes by Date
inflation_data <- full_join(food_data_processed, transportation_data_processed, by = "Date", suffix = c
  full_join(housing_data_processed, by = "Date") %>%
  rename(Housing = inflation)

# Fill in NA values with 0 or the mean, as per your analysis requirements
inflation_data[is.na(inflation_data)] <- 0  # or use the mean/infinity based on your needs

# 4. Visualize Inflation Rates

# Plot
ggplot(inflation_data, aes(x = Date)) +
  geom_line(aes(y = Housing, color = "Housing")) +
  geom_line(aes(y = inflation_Food, color = "Food")) +
  geom_line(aes(y = inflation_Transportation, color = "Transportation")) +
  labs(title = "Monthly Inflation Rates of CPI Components", y = "Inflation Rate (%)") +
  theme_minimal() +
  scale_color_manual(values = c("Housing" = "blue", "Food" = "green", "Transportation" = "red"))
```



Problem 7.8

```
cpi_excl_data <- read_excel("~/Downloads/CPILFESL.xls")
cpi_general_data <- read_excel("~/Downloads/CPIAUCSL.xls")

# Preview the data structure
str(cpi_excl_data)
```

```
## tibble [813 x 2] (S3: tbl_df/tbl/data.frame)
##  $ observation_date: POSIXct[1:813], format: "1957-01-01" "1957-02-01" ...
##  $ CPILFESL        : num [1:813] 28.5 28.6 28.7 28.8 28.8 28.9 29 29 29.1 29.2 ...
```

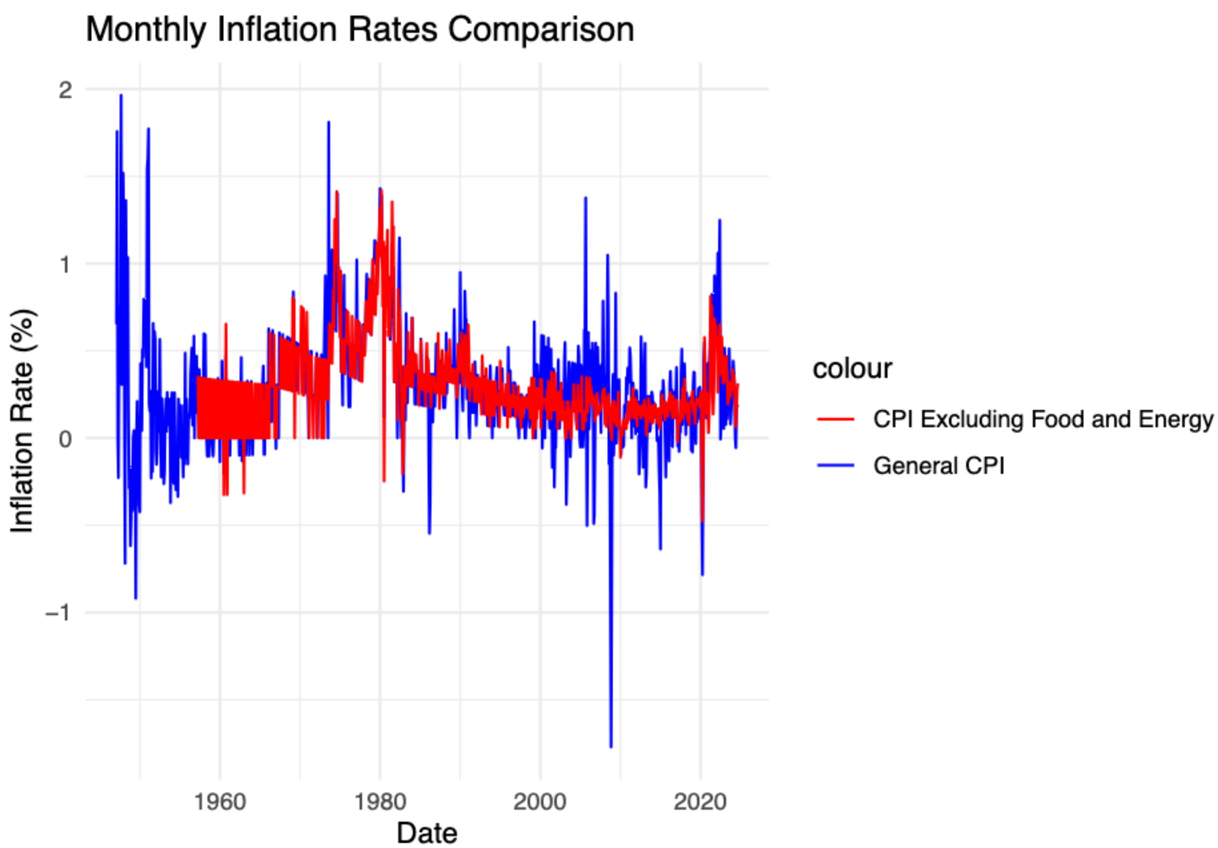
```
str(cpi_general_data)
```

```
## tibble [933 x 2] (S3: tbl_df/tbl/data.frame)
##  $ observation_date: POSIXct[1:933], format: "1947-01-01" "1947-02-01" ...
##  $ CPIAUCSL        : num [1:933] 21.5 21.6 22 22 21.9 ...
```

```
# Convert dates to Date format and calculate inflation rate as % change from the previous month
cpi_general_data <- cpi_general_data %>%
  mutate(Date = as.Date(observation_date),
```

```
         Inflation = (CPIAUCSL - lag(CPIAUCSL)) / lag(CPIAUCSL) * 100) %>%
  na.omit()

cpi_excl_data <- cpi_excl_data %>%
  mutate(Date = as.Date(observation_date),
         Inflation = (CPILFESL - lag(CPILFESL)) / lag(CPILFESL) * 100) %>%
  na.omit()
```

```
ggplot() +
  geom_line(data = cpi_general_data, aes(x = Date, y = Inflation, color = "General CPI")) +
  geom_line(data = cpi_excl_data, aes(x = Date, y = Inflation, color = "CPI Excluding Food and Energy"))
  labs(title = "Monthly Inflation Rates Comparison", y = "Inflation Rate (%)") +
  scale_color_manual(values = c("General CPI" = "blue", "CPI Excluding Food and Energy" = "red")) +
  theme_minimal()
```




```
# Fit AR(2) model for each series
model_cpi_general <- Arima(cpi_general_data$Inflation, order = c(2, 0, 0))
model_cpi_excl <- Arima(cpi_excl_data$Inflation, order = c(2, 0, 0))

summary(model_cpi_general)
```

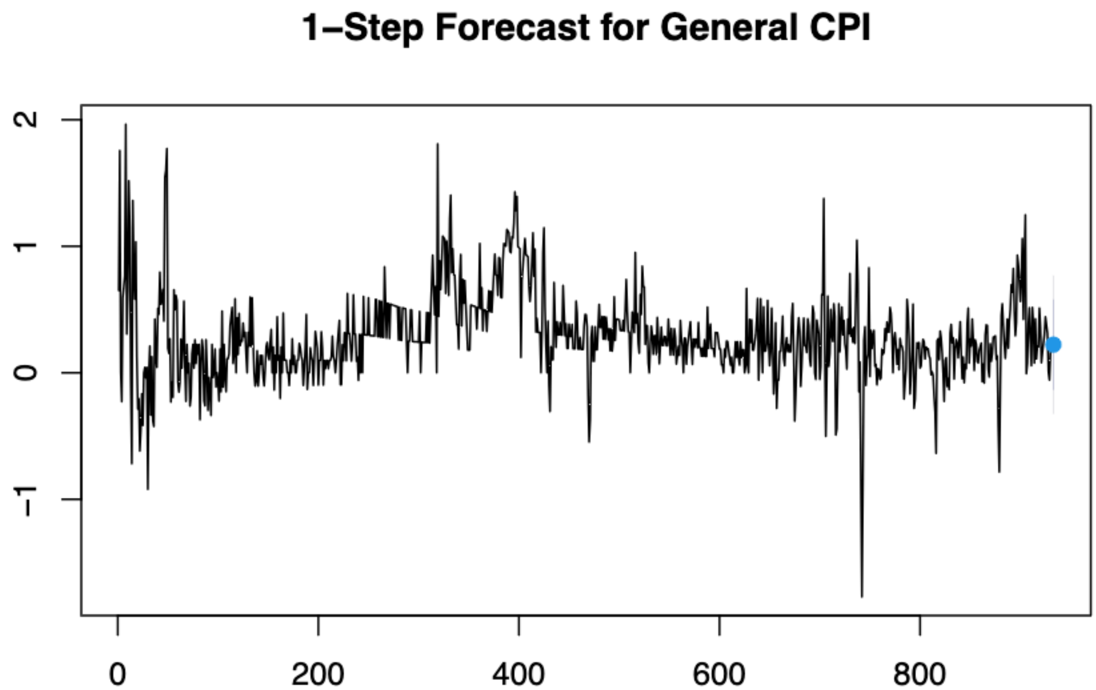
```
## Series: cpi_general_data$Inflation
## ARIMA(2,0,0) with non-zero mean
##
```

```
## Coefficients:
##          ar1     ar2    mean
##       0.5072  0.1181  0.2901
## s.e.  0.0325  0.0329  0.0243
##
## sigma^2 = 0.07775:  log likelihood = -130.88
## AIC=269.77   AICc=269.81   BIC=289.12
##
## Training set error measures:
##                         ME      RMSE       MAE MPE MAPE     MASE         ACF1
## Training set -0.0005964804 0.2783892 0.1932691 NaN  Inf 0.908802 -0.01695277
```

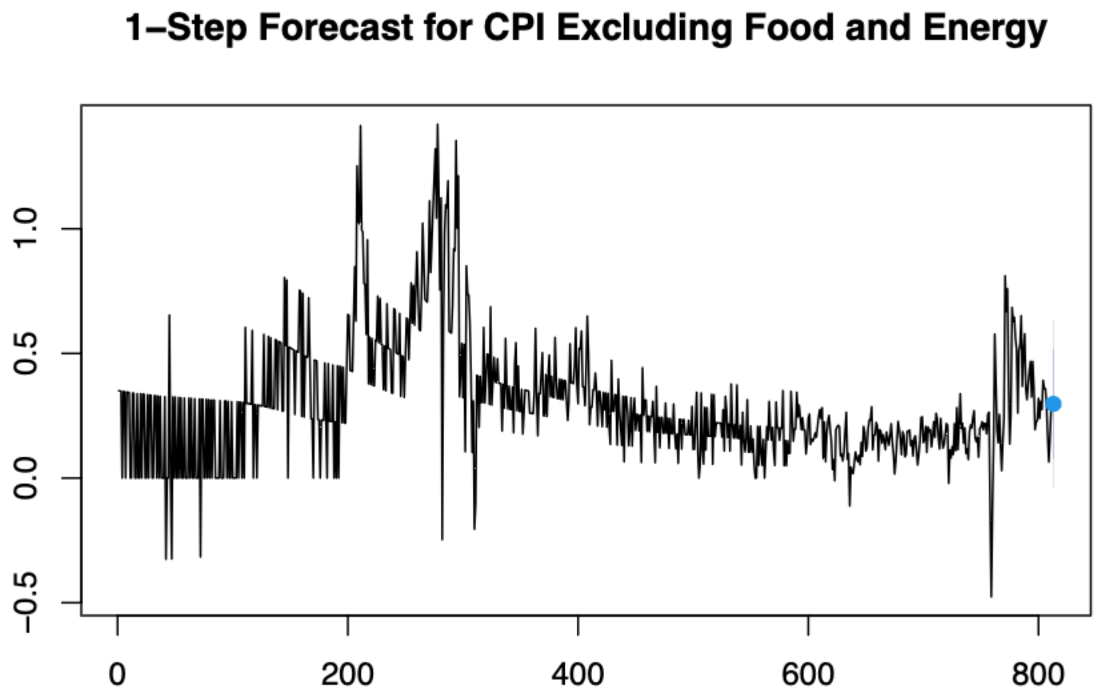
```r
summary(model_cpi_excl)
```

```
## Series: cpi_excl_data$Inflation
## ARIMA(2,0,0) with non-zero mean
##
## Coefficients:
##          ar1     ar2    mean
##       0.4042  0.3810  0.2989
## s.e.  0.0324  0.0324  0.0278
##
## sigma^2 = 0.02951:  log likelihood = 279.19
## AIC=-550.38   AICc=-550.33   BIC=-531.58
##
## Training set error measures:
##                         ME      RMSE       MAE MPE MAPE      MASE         ACF1
## Training set -3.160774e-05 0.1714779 0.1239182 NaN  Inf 0.8830726 -0.06520764
```

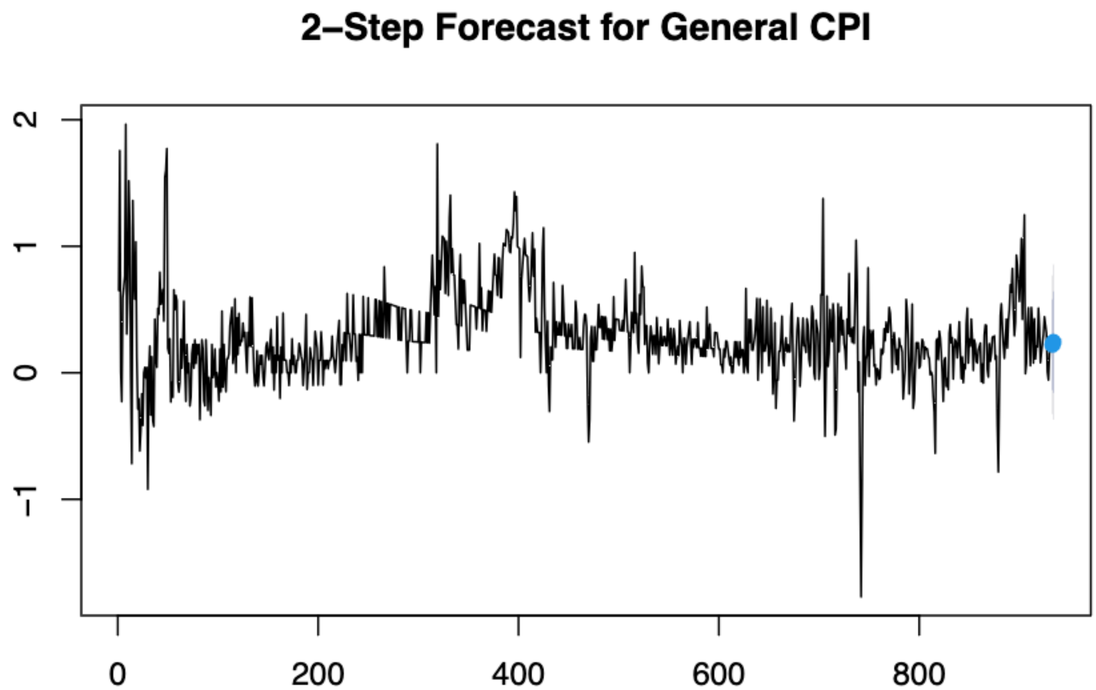
```r
# 1-step and 2-step forecasts
forecast_cpi_general_1 <- forecast(model_cpi_general, h = 1)
forecast_cpi_excl_1 <- forecast(model_cpi_excl, h = 1)

forecast_cpi_general_2 <- forecast(model_cpi_general, h = 2)
forecast_cpi_excl_2 <- forecast(model_cpi_excl, h = 2)

# Plot density forecasts
plot(forecast_cpi_general_1, main = "1-Step Forecast for General CPI")
```




```r
plot(forecast_cpi_excl_1, main = "1-Step Forecast for CPI Excluding Food and Energy")
```




```
plot(forecast_cpi_general_2, main = "2-Step Forecast for General CPI")
```




```r
plot(forecast_cpi_excl_2, main = "2-Step Forecast for CPI Excluding Food and Energy")
```

2–Step Forecast for CPI Excluding Food and Energy