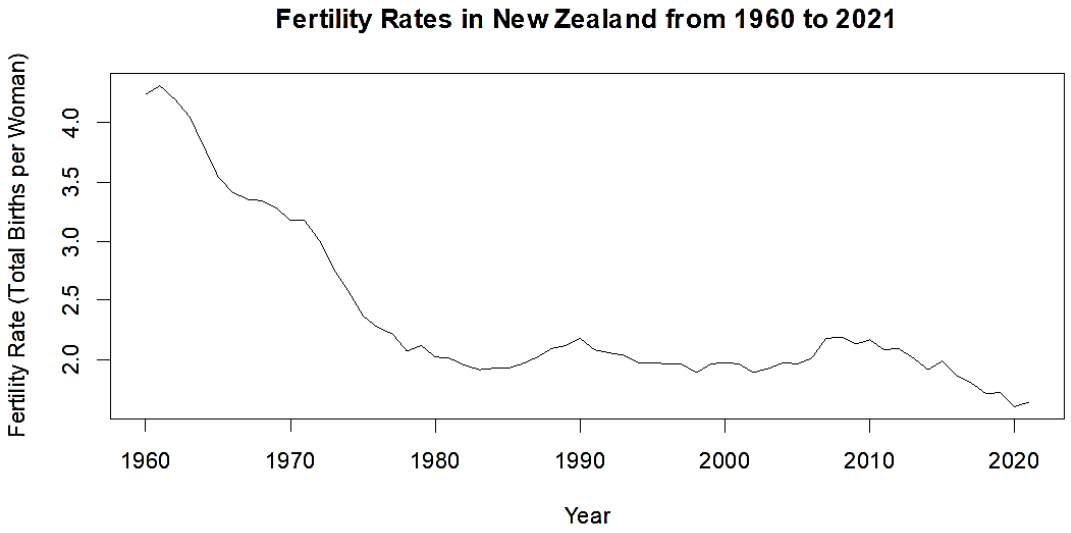
Introduction

This report aims to investigate the historical trends and future projections of fertility rates in New Zealand. Data on the said country's fertility rates from 1960 to 2021 was obtained from The World Bank's website (The World Bank, 2024). In the context of this report, the total fertility rate represents how many children a woman would have if she were to survive through her reproductive years and give birth according to the age-specific fertility rates of a given year. Several statistical methods have been employed for the purpose of forecasting future fertility rates, and these methods will be discussed further in the report. All statistical analyses and forecasts in this study were conducted using the R programming language. The corresponding code and its outputs are included in the *Appendix*. For clarity, each output is presented immediately following its respective code, enclosed within a grey border to distinguish it from the codes.

Section (a)



The time series plot above reveals a clear declining trend in the number of children born per woman from 1960 to 2021 among the New Zealanders. Initially high rates observed around 1960 gradually decrease, with a pronounced drop during the earlier decades, but stabilizing to a lower, more variable pattern post the 1980s. The trend component is the most evident feature of the series where it signals a long-term reduction in fertility rates. While the plot does show some fluctuations, these fluctuations are likely to be irregular fluctuations rather than cyclical given the absence of cyclic patterns to them. Seasonality is not present in the data as well, given that the data is on an annual basis. For seasonality to be observable, the data has to be on a more granular interval, for instance, on a daily, monthly, or quarterly basis. Statistical test precisely the Mann-Kendall Trend test was conducted to verify the existence of the trend component.

The **Mann-Kendall Trend test** is as follows:

$$H_0: Trend\ component\ is\ not\ present\ in\ the\ series$$

$$H_1: Trend\ component\ is\ present\ in\ the\ series$$

Given that the p-value (3.4338e-12) from the Mann-Kendall Trend test is below 0.05, the null hypothesis is rejected. Consequently, there is compelling proof that a trend component is present in time series at the 5% significance threshold.

Section (b)

The three methods chosen to perform forecasting are namely, Simple Exponential Smoothing (SES) method, Holt's Linear Trend model, and Cubic Trend model.

- **Simple Exponential Smoothing Method**
  SES is used when the time series data requires smoothing. It is best suited when there are no significant trends or seasonal patterns present because it merely adjusts the forecasts according to the level of the series alone. While SES is generally not recommended for data with trend components, just like the one analysed in this report, it was chosen because of its simplicity which is meant to serve as a benchmark to examine the impact of adding complexity (trend component in this case) in the other two models.

  The formula and its components are as follows (OTexts, 2023-a):
  - Equation for Smoothing: $l_t = \alpha y_t + (1 - \alpha) l_{t-1}$
  - Equation for Forecast: $\hat{y}_{t+1} = l_t$

  Where:
  - y_t represents the actual observations at time t
  - l_t is the estimated or smoothed level at time t
  - α is the smoothing constant, dictating how quickly the impact of past data diminishes

- **Holt's Linear Trend Model**
  The model is a continuation of the SES method where it is designed precisely to address data with trend components. Contrary to the SES method, which is best suited for data with no trend or seasonal patterns, the Holt's Linear Trend model is able to handle data with trends because there are two components for smoothing, namely the level equation and the trend equation, incorporated into the model. That said, since the fertility data has a trend component to it, and since the Holt's model is able to adjust its forecast to both the level and trend in the data, the model was thereby chosen.

The formula and its components are as follows (OTexts, 2023-b):

- Equation for Level Estimation: $l_t = \alpha y_t + (1-\alpha)(l_{t-1} + b_{t-1})$
- Equation for Trend Adjustment:$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1}$
- Equation for Forecast: $\hat{y}_{t+h} = l_t + hb_t$

Where:

- y_t represents the actual observations at time t
- l_t indicates the level estimated for time t
- b_t is the trend estimated for time t
- α and β are the parameters used for smoothing the level and trend, respectively
- h is the time frame over which forecasts are projected

- **Cubic Trend Model**

As part of the family of the polynomial regression models within the broader category of regression analysis, the Cubic Trend model is a type of advanced forecasting tools that integrates a cubic polynomial to map out the data. The inclusion of both quadratic (ct^2) and cubic terms (dt^3) enables the model to adeptly capture complex, non-linear trends in the data better than models with lower orders (Linear Trend or Quadratic Trend model). Such capabilities are particularly valuable when forecasting variables like fertility rates, where the interaction of multiple external factors can create dynamic and intricate patterns. Given the specific distribution of New Zealand's fertility rates, which notably follows a cubic trajectory, the Cubic Trend Model is especially well-suited, thereby justifying its use in the analysis of this report.

The formula and its components are as follows (PennState Eberly College of Science, 2018):

- Equation for the Model: $y_t = a + bt + ct^2 + dt^3 + \varepsilon_t$

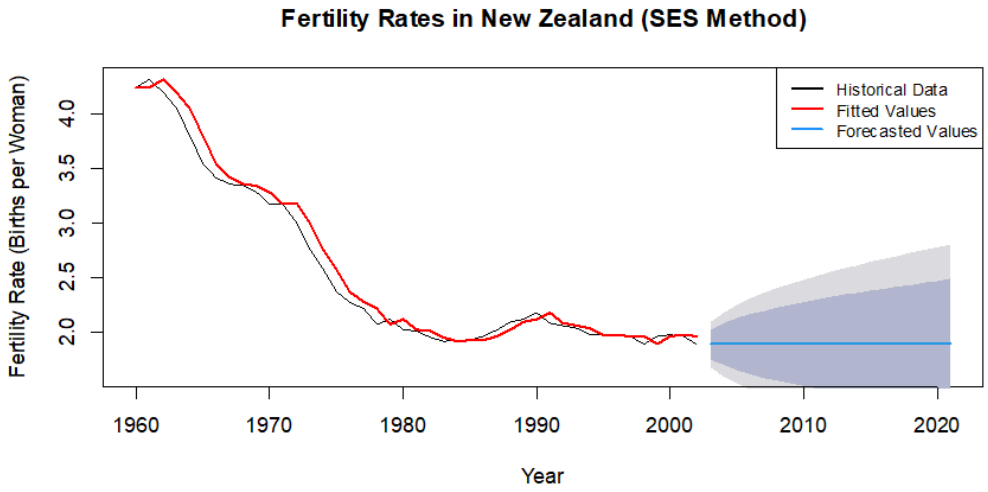
Where:

- y_t represents the actual observations at time t
- t denotes the variable representing time
- a, b, c, d are the coefficients derived through the model fitting process with the observed data
- ϵ_t represents the error at time t

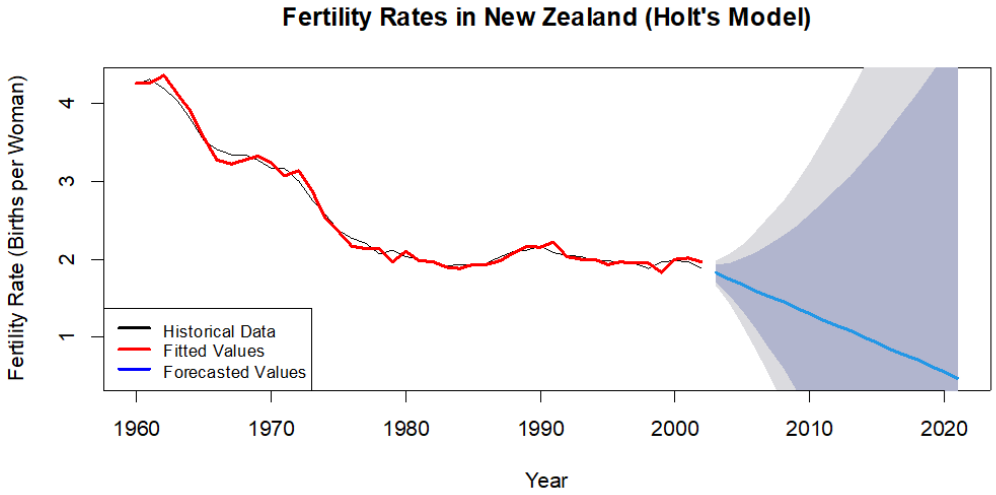
Section (c)

The dataset has been divided into training and testing subsets, with 70% allocated for training and 30% for testing. Detailed forecasts using the methods outlined in *Section (b)* are presented below.

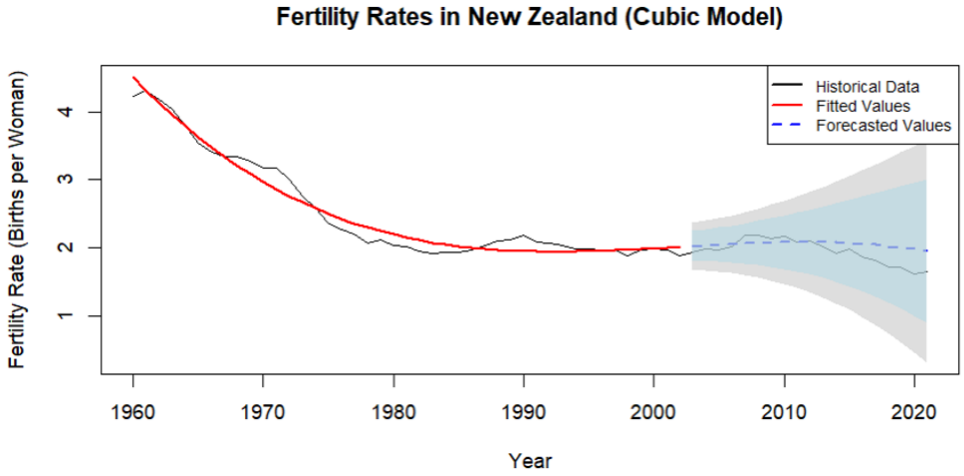
- **Simple Exponential Smoothing Method**



- **Holt's Linear Trend Model**



- **Cubic Trend Model**



Section (d)

The two measures of prediction error chosen are Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).

Reasons for choosing both measures are as follows:

- **Easy to Understand MAE:**
  MAE simply averages out the absolute errors, making it easy for people who are not experts in technical fields to grasp. This clarity is crucial when explaining how well a model performs to decision-makers who need straightforward, practical information. This is particularly important in areas like social sciences where forecasting, such as predicting fertility rates, is involved.

- **All Errors are Equal for MAE:**
  MAE does not penalize larger errors more than it does with smaller ones. In situations where each error in forecasting can significantly impact decisions, like in planning, MAE makes sure every error counts the same.

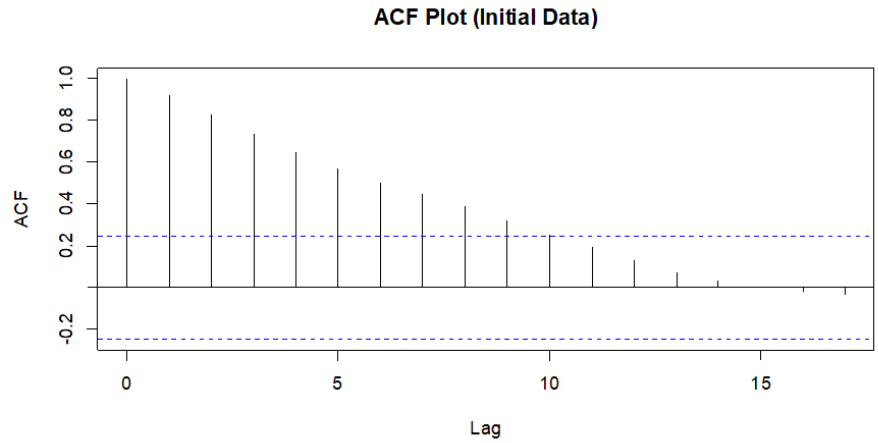
- **RMSE Focuses on Large Errors:**
  RMSE squares the errors, averages them, and then takes the square root. This approach makes RMSE very sensitive to larger errors, which is very important when forecasting things like fertility rates. Big mistakes in such forecasts can lead to major misjudgements in planning needed healthcare, education, and social services. RMSE helps highlight these big errors, giving policymakers critical information to make better choices about distributing resources and planning long-term, reducing the risk of too much or too little service based on population changes.

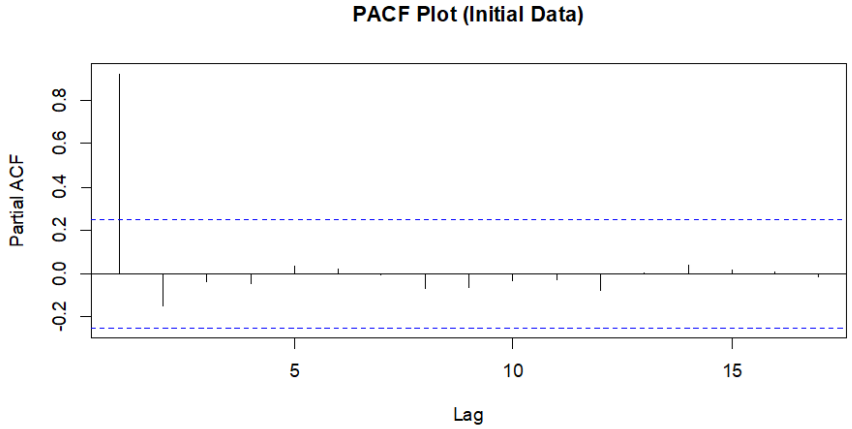
- **Both Metrics Together Give a Balanced Picture:**
  While MAE offers a straightforward look at average model performance, RMSE shows how big errors can impact predictions. Using both gives forecasters a complete view where RMSE shows how often big errors happen, and MAE shows how on-point or off

the predictions generally are daily. This combination helps forecasters know exactly where their predictions stand.

Upon evaluating all three methods stated in *Section (b)*, the Cubic Trend model is the most appropriate forecast model given that it has the lowest RMSE (0.1767227) and MAE (0.1405531) values among the rest of the two models, for the testing set.

Section (e)





Based on the plots displayed above, the Autocorrelation Function (ACF) plot shows a data pattern of dies down slowly, which is a typical indication of a non-stationary time series, possibly due to the presence of a trend component. The PACF plot showing a significant drop after the first lag suggests an autoregressive component of order 1 (AR(1)). In essence, it means that an AR(1) model could be a good fit for the data, capturing the dependency that each value has on its predecessor. To check for the validity of these findings, formal statistical tests, precisely the Unit Root test and the Kwiatkowski–Phillips–Schmidt–Shin (KPSS) test were conducted. Given that the time series data of New Zealand's fertility rates is believed to have a deterministic trend, a downward slope to be precise, the Trend Stationarity version of the KPSS test will be performed instead of its counterpart – Level Stationarity.

The **Unit Root test** is as follows:

$$H_0: The\ time\ series\ does\ not\ exhibit\ stationarity$$

$$H_1: The\ time\ series\ exhibits\ stationarity$$

Given that the p-value (0.1401) from the Unit Root test exceeds 0.05, the null hypothesis is not rejected. There is inadequate proof to assert that the time series exhibits stationarity at the 5% significance threshold.

The **KPSS test (Trend Stationarity)** is as follows:

$$H_0: The\ time\ series\ exhibits\ trend\ stationarity$$

$$H_1: The\ time\ series\ does\ not\ exhibit\ trend\ stationarity$$

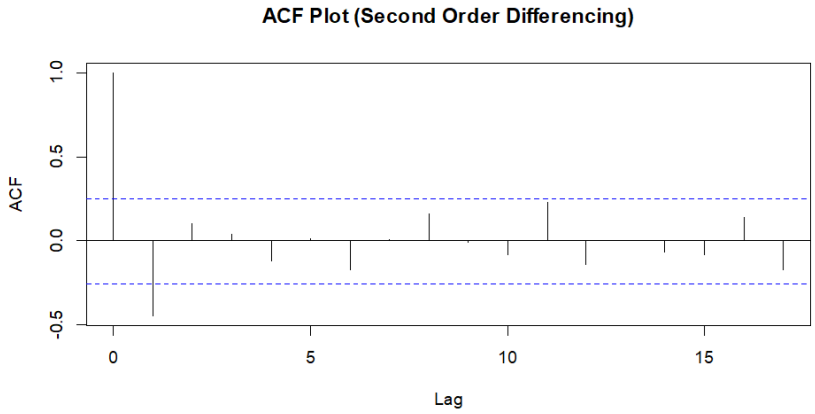
Given that the p-value (0.01) from the KPSS test falls below 0.05, the null hypothesis is rejected. This provides adequate proof to establish that the time series des not exhibit trend stationarity at the 5% significance threshold.

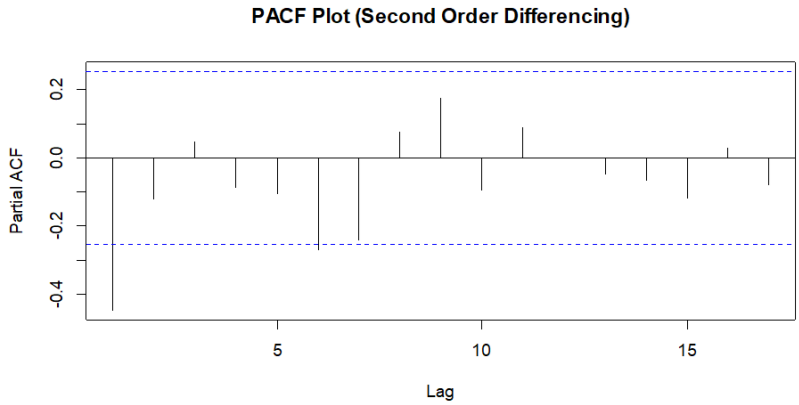
In light of the outputs from the Unit Root test and the KPSS test conducted on the initial time series data, the ndiffs() function from the "forecast" package is called to check for the suggested differencing order to be performed. The function suggested that second order differencing should be performed to convert the time series data from a non-stationary state to a stationary one. While the second order differencing was suggested, increasing the order of the differencing will be done one at a time rather than immediately going straight into the second order differencing. This is because it is always preferred to have a lower differencing order in the context of time series analysis. In other words, after the first differencing order has been applied, both Unit Root test and KPSS test will once again be carried out, to evaluate if a second order differencing is indeed necessary.

From the results of the first order differencing, the p-values for the Unit Root test and the KPSS test were 0.107 and 0.03287, respectively. In other words, it means that albeit having performed first order differencing, the time series still remained non-stationary and non-trend stationary. Given that, a second order differencing is required. After the second order differencing, the time series data is now stationary and trend stationary where p-values were 0.01 and 0.1 for the Unit Root test and KPSS test, respectively. The former (Unit Root test) leads us to reject the null hypothesis and confirm that there is enough proof that the time series is now stationary at the 5% level significance threshold. The latter (KPSS test) leads to the decision to not reject the null hypothesis and confirm that there is not enough proof that the time series is not trend stationary at the 5% significance threshold. In conclusion, a second

order differencing is required to transform the time series data of New Zealand's fertility rate into a stationary time series.

Section (f) (i)





The plots above are the resulting ACF and PACF plots of the time series data after having applied second order differencing to the data. In the ACF plot, while there was a drop right after the first lag, the drop is considered as not a drastic drop. Furthermore, subsequent lags do not hover closely around zero, hence suggesting that the data pattern in the ACF plot is a dies down exponentially pattern. Similarly, in the PACF plot, the drop from the first lag was not too drastic and subsequent lags do not hover closely around zero, once again suggesting that the data pattern in the PACF plot is a dies down exponentially pattern as well. Given these findings, the suggested model according to the Box-Jenkins Methodology would be ARIMA(1,2,1).

To identify the second model, the auto.arima() function from the "forecast" package was called to provide suggestion on the most optimal model for the time series data examined in this report. The output of the function suggested that the ARIMA(0,2,1) model is the best among all of the models that it has evaluated, given that the said model has the lowest AICc value of -128.1283. In fact, the second-best model, with the second lowest AICc value (-126.8624) is the ARIMA(1,2,1) model, essentially confirming our suggestion which was based

on the Box-Jenkins Methodology. To conclude this section, the two models suggested are the ARIMA(1,2,1) model and ARIMA(0,2,1) model.

Section (f) (ii)

The summary outputs for both ARIMA models are detailed below.

Output summary for the **ARIMA (1,2,1)** model:

```
> summary(arima_fertility_121)
Series: fertility
ARIMA(1,2,1)

Coefficients:
         ar1      ma1
      0.2024  -0.8434
s.e.  0.1930   0.1132

sigma^2 = 0.00647:  log likelihood = 66.65
AIC=-127.29   AICc=-126.86   BIC=-121.01

Training set error measures:
                      ME       RMSE        MAE       MPE    MAPE      MASE       ACF1
Training set 0.0006417363 0.07779877 0.06422504 0.1764096 2.83581 0.8706061 0.02054931
```

Output summary for the **ARIMA(0,2,1)** model:

```
> summary(arima_fertility_021)
Series: fertility
ARIMA(0,2,1)

Coefficients:
          ma1
      -0.7222
s.e.   0.1410

sigma^2 = 0.006481:  log likelihood = 66.17
AIC=-128.34   AICc=-128.13   BIC=-124.15

Training set error measures:
                      ME      RMSE        MAE        MPE     MAPE      MASE      ACF1
Training set -0.001266659 0.0785315 0.06404357 0.09027092 2.785506 0.8681462 0.1305416
```

Section (f) (iii)

The respective equations for the suggested ARIMA models are detailed below.

Mathematical Representation of **ARIMA (1,2,1)** model:

$$(1-\phi_1 B)(1-B)^2 Y_t = (1+\theta_1 B)\varepsilon_t$$

$$(1-\phi_1 B)(1-B)(1-B) Y_t = (1+\theta_1 B)\varepsilon_t$$

$$(1-\phi_1 B)(1-2B+B^2) Y_t = (1+\theta_1 B)\varepsilon_t$$

$$(1-2B+B^2-\phi_1 B+2\phi_1 B^2-\phi_1 B^3) Y_t = (1+\theta_1 B)\varepsilon_t$$

$$Y_t - 2Y_{t-1} + Y_{t-2} - \phi_1 Y_{t-1} + 2\phi_1 Y_{t-2} - \phi_1 Y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

$$Y_t = (2+\phi_1) Y_{t-1} - (1+2\phi_1) Y_{t-2} + \phi_1 Y_{t-3} + \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

$$Y_t = 2.2024 Y_{t-1} - 1.4048 Y_{t-2} + 0.2024 Y_{t-3} + \varepsilon_t - 0.8434 \varepsilon_{t-1}$$

Mathematical Representation of **ARIMA (0,2,1)** model:

$$(1-B^2) Y_t = (1+\theta_1 B)\varepsilon_t$$

$$(1-B)(1-B) Y_t = (1+\theta_1 B)\varepsilon_t$$

$$(1-2B+B^2) Y_t = (1+\theta_1 B)\varepsilon_t$$

$$Y_t - 2Y_{t-1} + Y_{t-2} = \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

$$Y_t = 2Y_{t-1} - Y_{t-2} + \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

$$Y_t = 2Y_{t-1} - Y_{t-2} + \varepsilon_t - 0.7222 \varepsilon_{t-1}$$

Section (f) (iv)

For assessing the adequacy of both models, the checkresiduals() function from the "forecast" package was used. The residual analysis consists of two parts precisely the checking of the p-value for the Ljung-Box test, and the checking of the ACF plot of the residuals. Both parts of the residual analysis are detailed below.

The **Ljung-Box test** is as follows:

$$H_0: The\ residuals\ are\ random\ (Model\ is\ Adequate)$$

$$H_1: The\ residuals\ are\ not\ random\ (Model\ is\ Inadequate)$$

As the p-values of 0.3139 for the ARIMA(1,2,1) model and 0.09844 for the ARIMA(0,2,1) model from the Ljung-Box test are both above 0.05, the null hypothesis is not rejected. Therefore, there is not enough evidence to deem both models inadequate at the 5% significance threshold.

From the ACF plot for the residuals of both ARIMA(1,2,1) and ARIMA(0,2,1) model, both plots showed that all autocorrelations are within the confidence bounds for different lags. The bars do not exceed the significance thresholds, thereby suggesting no significant autocorrelation at any of the displayed lags. In other words, it implies that the residuals from both models have the properties of being white noises.

Section (f) (v)

To check for the significance of the coefficient parameters within both models, the coeftest() function from the “lmtest” package was called.

The coefficient test for the **ARIMA(1,2,1)** model is as follows:

- Hypothesis testing for AR1 component

$$H_0: The\ AR1\ component\ equals\ zero$$

$$H_1: The\ AR1\ component\ does\ not\ equal\ zero$$

With the AR1 component's p-value at 0.2944, which is above 0.05, the null hypothesis cannot be rejected. Consequently, there is not enough proof to assert that the AR1 component plays a significant role in the ARIMA(1,2,1) model at the 5% significance threshold.

- Hypothesis testing for MA1 component

$$H_0: The\ MA1\ component\ equals\ zero$$
$$H_1: The\ MA1\ component\ does\ not\ equal\ zero$$

Given that the p-value for the MA1 component is 9.465e-14, falling below 0.05, the null hypothesis is rejected. This provides adequate proof to affirm that the MA1 component is a significant element of the ARIMA(1,2,1) model at the 5% significance threshold.

The coefficient test for the **ARIMA(0,2,1)** model is as follows:

- Hypothesis testing for MA1 component

$$H_0: The\ MA1\ component\ equals\ zero$$
$$H_1: The\ MA1\ component\ does\ not\ equal\ zero$$

Given that the p-value for the MA1 component is 3.049e-07, which is below 0.05, the null hypothesis is rejected. Therefore, there is enough proof to assert that the MA1 component is a significant factor in the ARIMA(0,2,1) model at the 5% significance threshold.

Section (f) (vi)

The MAE and RMSE values for the ARIMA(1,2,1) model are 0.06422504 and 0.07779877 respectively. As for the ARIMA(0,2,1) model, its MAE and RMSE values are 0.06404357 and 0.0785315 respectively. From the evaluation of both models based on the two measures of forecast errors, it is evident that the ARIMA(0,2,1) model has a marginally lower MAE value than the ARIMA(1,2,1) model, but has a slightly higher RMSE values than its counterpart.

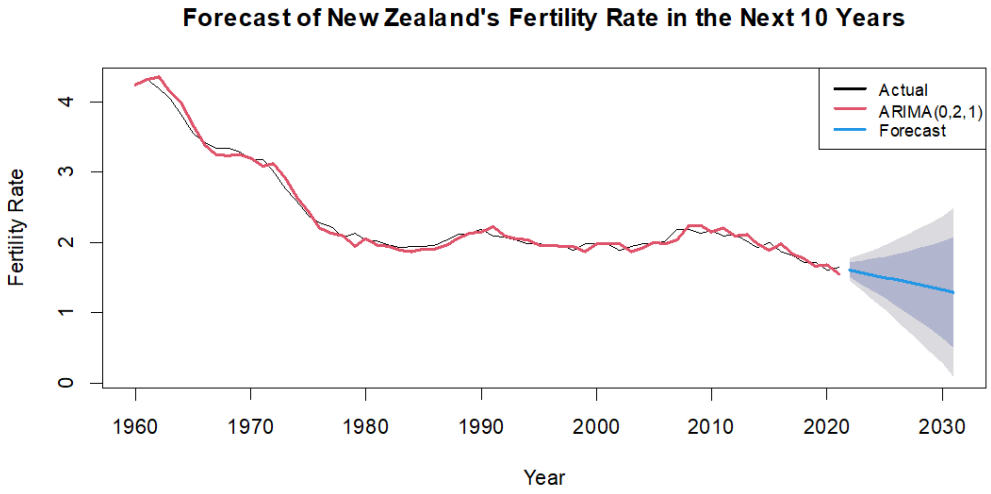
Considering the principles of parsimony and the forecast accuracy metrics presented above, the most suitable model would be the ARIMA(0,2,1) model. The reason is that according to the principle of parsimony, the principle suggests that when two models perform similarly, the simpler model should be preferred because it is likely to generalize better to new data and involves fewer assumptions. In this case, although the ARIMA(1,2,1) model shows a slightly lower RMSE, indicating a marginally better capability at handling larger errors, the ARIMA(0,2,1) model has a lower MAE and only slightly higher RMSE. This difference in RMSE is minimal and does not sufficiently justify the additional complexity of including an autoregressive term in the ARIMA(1,2,1) model. Moreover, the ARIMA(0,2,1) model's slightly better MAE suggests it is generally effective at reducing the average magnitude of errors, which is often more critical for consistent performance across various scenarios. Considering that both models achieve close performance levels, the simpler ARIMA(0,2,1) model is preferable as it likely offers similar predictive capabilities without the risk of overfitting. To conclude, for forecasting purposes where a balance between error magnitude

and model simplicity is crucial, the ARIMA(0,2,1) model stands out as the more practical and robust choice, hence the most appropriate model.

Section (g)

Given that the ARIMA(0,2,1) model was identified as the most suitable model, forecast of New Zealand's fertility rate in the next 10 years is thereby performed using the said model.

```
> forecast_fertility
     Point Forecast     Lo 80    Hi 80      Lo 95    Hi 95
2022       1.604425 1.5012558 1.707594 1.44664135 1.762209
2023       1.568850 1.4014465 1.736253 1.31282855 1.824871
2024       1.533275 1.3013626 1.765187 1.17859565 1.887954
2025       1.497700 1.1984250 1.796974 1.03999857 1.955401
2026       1.462125 1.0919716 1.832278 0.89602431 2.028225
2027       1.426550 0.9818356 1.871264 0.74641827 2.106681
2028       1.390975 0.8680213 1.913928 0.59118649 2.190763
2029       1.355400 0.7505975 1.960202 0.43043459 2.280364
2030       1.319824 0.6296579 2.009991 0.26430582 2.375343
2031       1.284249 0.5053037 2.063195 0.09295473 2.475544
```



References

OTexts. (2023-a). *8.1 Simple exponential smoothing*. https://otexts.com/fpp3/ses.html

OTexts. (2023-b). *7.2 Trend methods*. https://otexts.com/fpp2/holt.html

PennState Eberly College of Science. (2018). *7.7 - Polynomial regression.* https://online.stat.psu.edu/stat462/node/158/

The World Bank. (2024). *Fertility rate, total (births per woman).* https://data.worldbank.org/indicator/SP.DYN.TFRT.IN

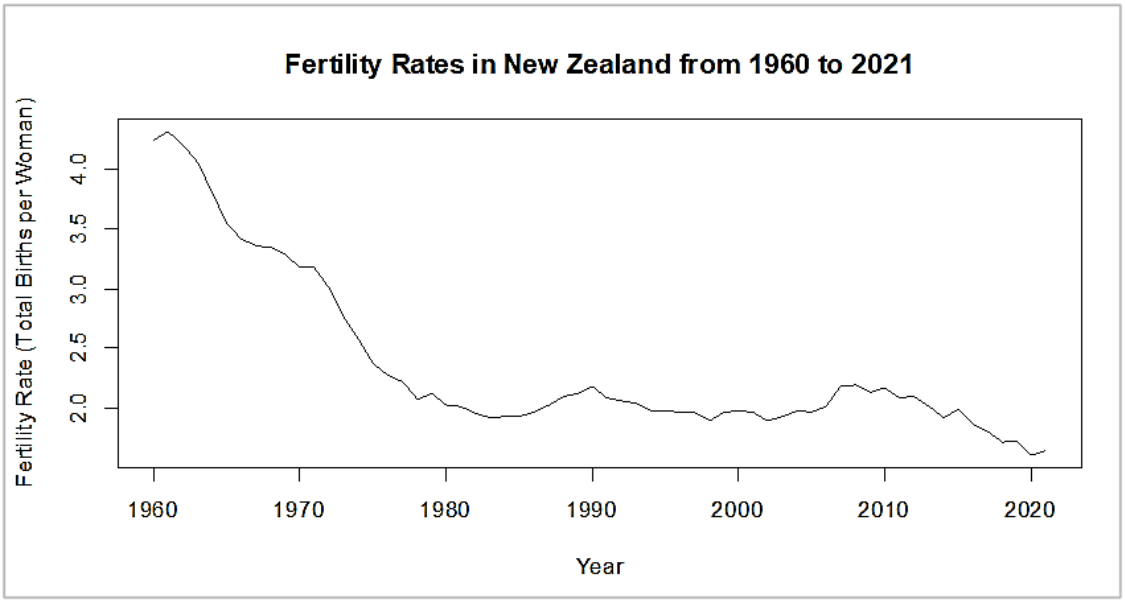
Appendix

Section (a)

```r
#----------------------------------------Preparation of Data-----------------------------------------#

# Install and load packages
install.packages("stats")
install.packages("tidyverse")
install.packages("Kendall")
install.packages("forecast")
install.packages("tseries")
install.packages("lmtest")

library(stats)
library(tidyverse)
library(Kendall)
library(forecast)
library(tseries)
library(lmtest)

# Extract rows for "New Zealand" and exclude columns for Country Name, Country Code, and Indicator Code
fertility_ds <- Assignment_Dataset[Assignment_Dataset$`Country Name` == "New Zealand", ]

# Exclude unnecessary columns
fertility_ds <- fertility_ds[, -c(1, 2, 4)]

# Print the extracted and modified data to verify
print(fertility_ds)
```

```r
# Pivot the data from wide to long format
fertility_fds <- fertility_ds %>%
  pivot_longer(
    cols = -`Indicator Name`,  # Exclude the Indicator Name from the pivoting process
    names_to = "Year",
    values_to = "Fertility_Rate"
  ) %>%
  select(-`Indicator Name`)  # Remove the Indicator Name column from the final output

# Print the reshaped data to check
print(fertility_fds)

# Remove rows where Year is 2022 or 2023
fertility_fds <- fertility_fds[!(fertility_fds$Year %in% c(2022, 2023)), ]

# Print the updated dataset to verify the changes
print(fertility_fds)

# Convert the data into a time series
fertility <- ts(fertility_fds$Fertility_Rate, start=1960, frequency=1)
```

```r
#----------------------------------------Plot Time Series Plot----------------------------------------#

# Plot the time series
plot(fertility, xlab="Year", ylab="Fertility Rate (Total Births per Woman)",
     main="Fertility Rates in New Zealand from 1960 to 2021", type='l')

# Adding a custom x-axis to the time series plot
axis(1, at = seq(0, length(fertility_fds$Year)-1, by = 10),
     labels = seq(1960, 2021, by = 10))
```



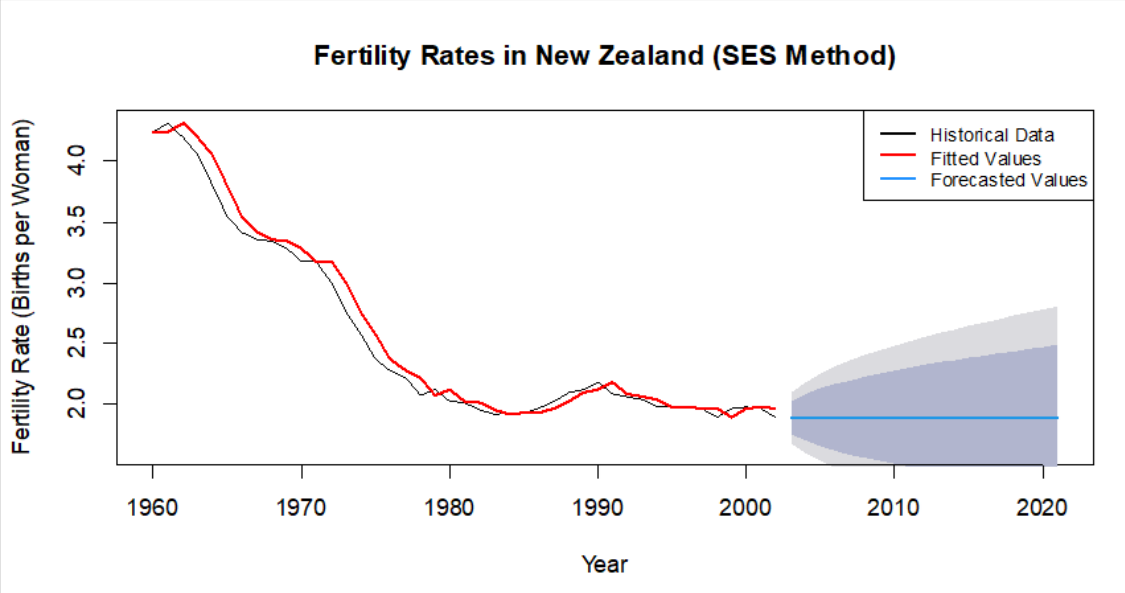

```r
#---------------------------------Statistical test for Trend Component---------------------------------#

# Perform the Mann-Kendall Trend Test to check for the presence of the trend component
# H0: Trend component is not present in the series
# H1: Trend component is present in the series
# Reject H0 if the p-value is less than 0.05

# Perform Mann-Kendall test
MannKendall(fertility)

# Since the p-value (3.4338e-12) is less than 0.05, we reject the null hypothesis. There is
# sufficient evidence to conclude that the trend component is present in the time series
# at 5% level of significance.
```

```
> MannKendall(fertility)
tau = -0.61, 2-sided pvalue =3.4338e-12
```

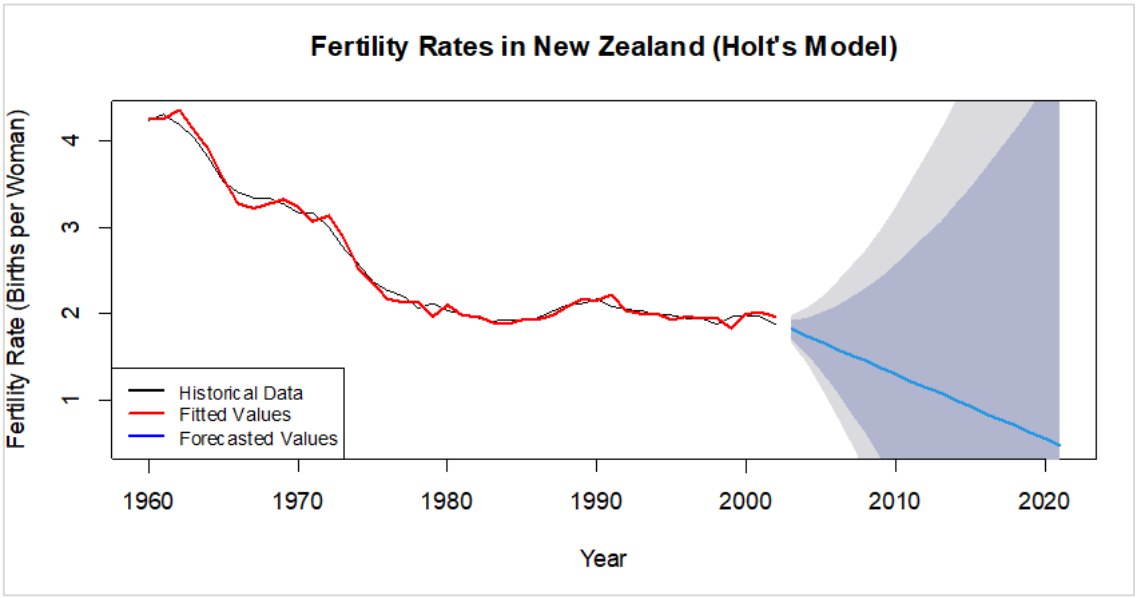
Section (c)

```r
#-------------------------------------------Split Dataset-------------------------------------------#

# Split the dataset into training and testing dataset
train_fertility <- fertility[1:(0.7*length(fertility))]
test_fertility <- fertility[-c(1:(0.7*length(fertility)))]

# Convert the training and testing dataset into time series again
train_fertility <- ts(train_fertility)
test_fertility <- ts(test_fertility)

# Check iif the conversion was successful
print(train_fertility)
print(test_fertility)
```

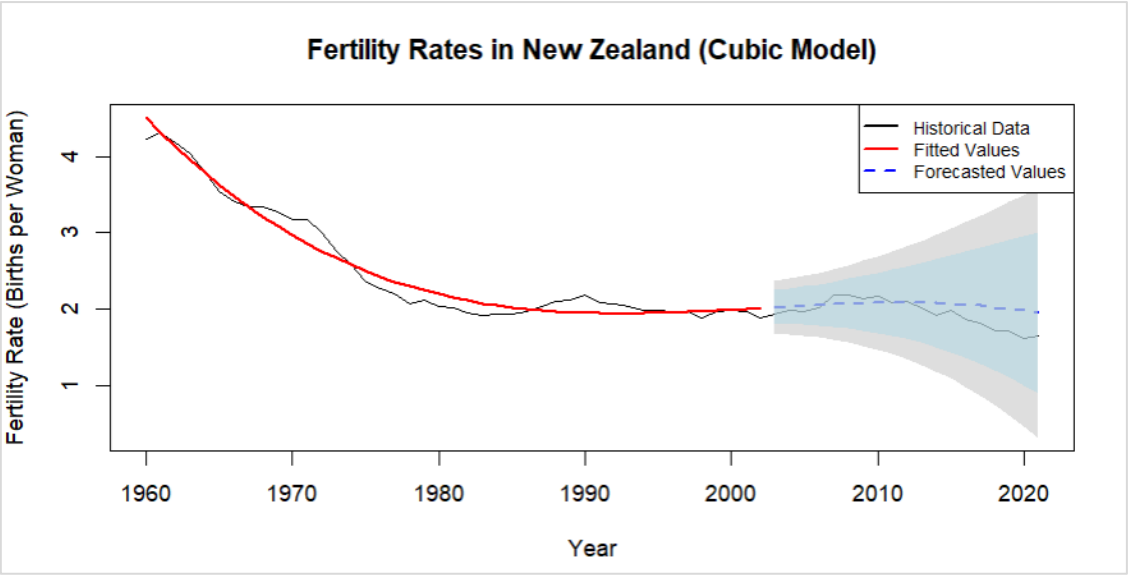
```r
#----------------------------------Simple Exponential Smoothing (SES)----------------------------------#

# Fit the Simple Exponential Smoothing model to the training data
ses_model <- ses(train_fertility, h = length(test_fertility))

# Print the model summary
print(summary(ses_model))

# Generate forecasts
ses_forecast <- forecast(ses_model)

# Determine the range of fertility rates to set the y-axis limits
y_limits_ses <- range(c(fertility, ses_forecast$mean), na.rm = TRUE)

# Adjust the plot to show years starting from 1960 instead of 0
years <- 1960 + seq_along(fertility) - 1
plot(ses_forecast, ylim = y_limits_ses, main="Fertility Rates in New Zealand (SES Method)", xlab="Year",
     ylab="Fertility Rate (Births per Woman)", xaxt='n')
lines(fertility, col="black", lwd=2) # Overlay the complete historical data
lines(fitted(ses_model), col="red", lwd=2) # Overlay the fitted values from the training data

# Add a legend to differentiate the data
legend("topright", legend=c("Historical Data", "Fitted Values", "Forecasted Values"),
       col=c("black", "red", "dodgerblue"), lty=1, lwd=2, cex=0.8)

# Specify x-axis ticks to reflect every decade starting from 1960
axis(1, at=seq(1, length(fertility), by=10), labels=years[seq(1, length(fertility), by=10)])
```



```r
#-------------------------------------Holt's Linear Trend Method-------------------------------------#

# Fit the Holt's linear trend model to the training data
holt_model <- holt(train_fertility, h = length(test_fertility))

# Generate forecasts for the length of the test set
holt_forecast <- forecast(holt_model, h = length(test_fertility))

# Print the model summary
summary(holt_model)

# Determine the range of fertility rates to set the y-axis limits
y_limits <- range(c(fertility, holt_forecast$mean), na.rm = TRUE)

# Adjust the plot to show years starting from 1960 instead of 0
years <- 1960 + seq_along(fertility) - 1
plot(holt_forecast, ylim = y_limits, main="Fertility Rates in New Zealand (Holt's Model)", xlab="Year",
     ylab="Fertility Rate (Births per Woman)", xaxt='n')
lines(fertility, col="black", lwd=2)  # Overlay the complete historical data
lines(fitted(holt_model), col="red", lwd=2)  # Overlay the fitted values from the training data

# Add a legend to differentiate the data
legend("bottomleft", legend=c("Historical Data", "Fitted Values", "Forecasted Values"),
       col=c("black", "red", "blue"), lty=1, lwd=2, cex=0.8)

# Specify x-axis ticks to reflect every decade starting from 1960
axis(1, at=seq(1, length(fertility), by=10), labels=years[seq(1, length(fertility), by=10)])
```



```r
#-----------------------------------------------Cubic Model-----------------------------------------------#

# Prepare the training data with both squared and cubic terms
train_data <- data.frame(year = 1960:(1960 + length(train_fertility) - 1), fertility = train_fertility)
train_data$year_squared <- train_data$year^2
train_data$year_cubed <- train_data$year^3

# Fit a cubic model to the training data
cubic_model <- lm(fertility ~ year + I(year^2) + I(year^3), data=train_data)

# Generate fitted values for the training data
train_data$fitted_values <- predict(cubic_model, newdata=train_data)

# Prepare test data for forecasting, include squared and cubic terms
test_data <- data.frame(year = (1960 + length(train_fertility)):(1960 + length(fertility) - 1))
test_data$year_squared <- test_data$year^2
test_data$year_cubed <- test_data$year^3

# Generate forecasted values and prediction intervals for the testing data
pred_80 <- predict(cubic_model, newdata=test_data, interval="prediction", level=0.80)
pred_95 <- predict(cubic_model, newdata=test_data, interval="prediction", level=0.95)

# Extract forecasted values and intervals
test_data$forecasted_values <- pred_95[, "fit"]
test_data$lwr_80 <- pred_80[, "lwr"]
test_data$upr_80 <- pred_80[, "upr"]
test_data$lwr_95 <- pred_95[, "lwr"]
test_data$upr_95 <- pred_95[, "upr"]

# Plot the fitted and forecasted results with prediction intervals for the testing data
plot(1960:(1960 + length(fertility) - 1), fertility, type='l', col='black',
     main="Fertility Rates in New Zealand (Cubic Model)", xlab="Year",
     ylab="Fertility Rate (Births per Woman)",
     ylim=range(c(fertility, train_data$fitted_values, test_data$forecasted_values,
                  test_data$lwr_95, test_data$upr_95)), xaxt='n')
lines(train_data$year, train_data$fitted_values, col='red', lwd=2)  # Fitted values in red
lines(test_data$year, test_data$forecasted_values, col='blue', lwd=2, lty=2)  # Forecasted values in blue

# Add 95% prediction intervals
polygon(c(test_data$year, rev(test_data$year)), c(test_data$upr_95, rev(test_data$lwr_95)),
        col=adjustcolor("grey", alpha.f=0.5), border=NA)

# Add 80% prediction intervals
polygon(c(test_data$year, rev(test_data$year)), c(test_data$upr_80, rev(test_data$lwr_80)),
        col=adjustcolor("lightblue", alpha.f=0.5), border=NA)

# Add a legend to differentiate the data
legend("topright", legend=c("Historical Data", "Fitted Values", "Forecasted Values"),
       col=c("black", "red", "blue"), lty=c(1, 1, 2, NA, NA), lwd=2, pch=c(NA, NA, NA, 15, 15), cex=0.8)

# Specify x-axis ticks to reflect every decade starting from 1960
axis(1, at=seq(1960, 2021, by=10), labels=seq(1960, 2021, by=10))

# Double check to see if the plots are accurate and informative
print(train_data)
print(test_data)
```



Section (d)

```
#-----------------------------------------Accuracy for SES-----------------------------------------#
# Calculate accuracy metrics
ses_accuracy <- accuracy(ses_forecast, test_fertility)
print(ses_accuracy)
```

```
> print(ses_accuracy)
                     ME      RMSE       MAE       MPE     MAPE      MASE      ACF1
Training set -0.05465466 0.1041870 0.07698348 -1.948515 2.943434 0.9768296 0.5471621
Test set      0.05893937 0.1862344 0.16210232  2.179258 8.341094 2.0568874        NA
```

```
#--------------------------------Accuracy for Holt's Linear Trend Model--------------------------------#
# Calculate accuracy metrics
holt_accuracy <- accuracy(holt_forecast, test_fertility)
print(holt_accuracy)
```

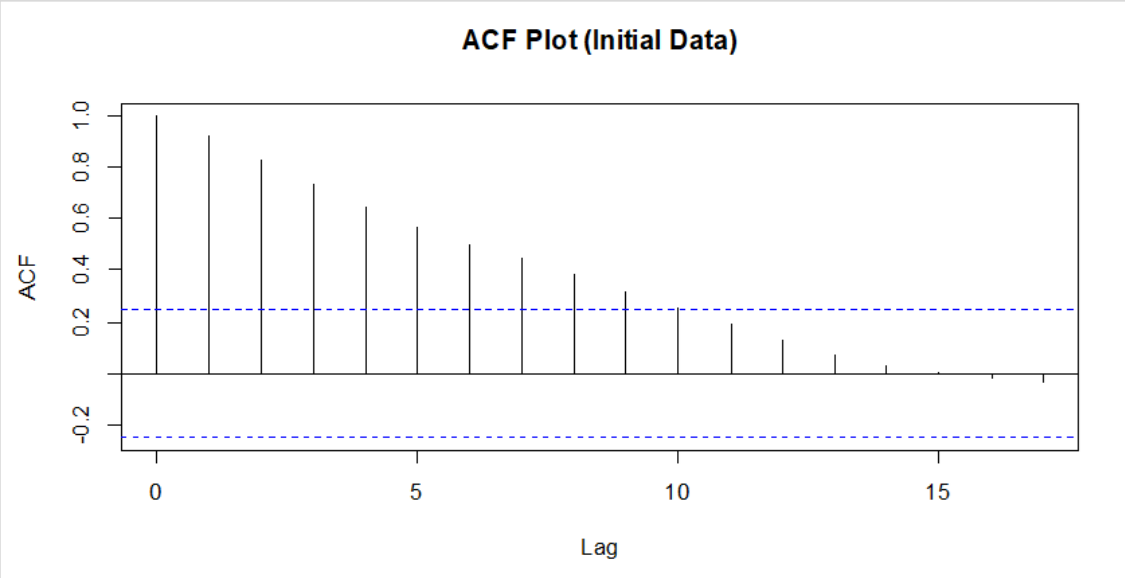
```
> print(holt_accuracy)
                      ME       RMSE        MAE         MPE      MAPE       MASE        ACF1
Training set -0.003145115 0.07776539 0.06251128 -0.03209408  2.470884  0.7931945 -0.04658102
Test set      0.794038401 0.85147706 0.79403840 41.72526222 41.725262 10.0754117          NA
```

```
#---------------------------------------Accuracy for Cubic Model---------------------------------------#
# Calculate accuracy metrics manually for the cubic model
cubic_mae <- mean(abs(test_fertility - test_data$forecasted_values))
cubic_rmse <- sqrt(mean((test_fertility - test_data$forecasted_values)^2))

# Print the accuracy metrics
cat("Cubic Model Accuracy:\n")
cat("MAE:", cubic_mae, "\n")
cat("RMSE:", cubic_rmse, "\n")
```

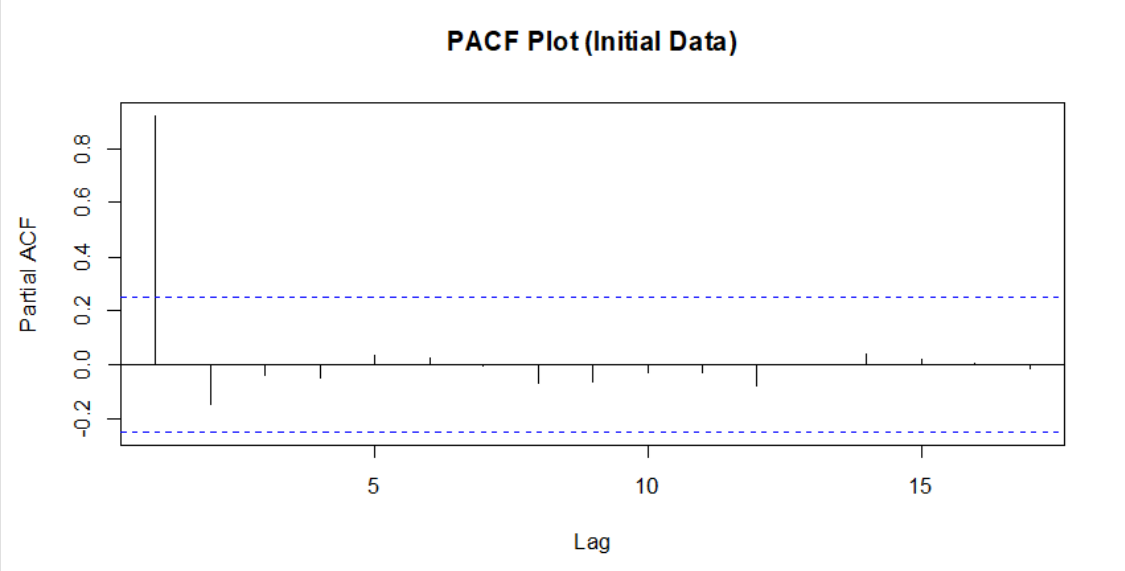
```
> # Print the accuracy metrics
> cat("Cubic Model Accuracy:\n")
Cubic Model Accuracy:
> cat("MAE:", cubic_mae, "\n")
MAE: 0.1405531
> cat("RMSE:", cubic_rmse, "\n")
RMSE: 0.1767227
```

Section (e)

```
#----------------------------------------Check for Stationarity----------------------------------------#

# Plot the ACF plot with title
acf(fertility, main="ACF Plot (Initial Data)")
```




```
# Plot the PACF plot with title
pacf(fertility, main="PACF Plot (Initial Data)")
```



```
#-------------------------------------Unit Root Test (Initial Data)-----------------------------------#

# Using the adf.test function to identify if the fertility data is stationary or non-stationary on the
# initial data
adf.test(fertility)
# H0: The time series is not stationary
# H1: The time series is stationary
# Reject H0 if the p-value is less than 0.05
# Since the p-value in the ADF test (0.1401) is greater than 0.05, we do not reject the null hypothesis,
# hence we conclude that the data is not stationary.
```

```
> adf.test(fertility)

        Augmented Dickey-Fuller Test

data:  fertility
Dickey-Fuller = -3.0744, Lag order = 3, p-value = 0.1401
alternative hypothesis: stationary
```

```
#---------------------Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test (Initial Data)--------------------#

# Perform KPSS test for trend stationarity on the initial data
kpss_result <- kpss.test(fertility, null = "Trend")

# Print the results
print(kpss_result)
# H0: The time series is trend stationary
# H1: The time series is not trend stationary
# Reject H0 if the p-value is less than 0.05
# Since the p-value is less than 0.05, we reject the null hypothesis and conclude that the series is
# likely not trend stationary, meaning that it exhibits characteristics of a unit root or some form
# of non-stationarity in the presence of a deterministic trend.
```

```
> print(kpss_result)

        KPSS Test for Trend Stationarity

data:  fertility
KPSS Trend = 0.33773, Truncation lag parameter = 3, p-value = 0.01
```

```
#-------------------------------------------ndiffs Function-------------------------------------------#
# To check what is the recommended differencing order.
ndiffs(fertility)
# 2
```

```
> ndiffs(fertility)
[1] 2
```

```
#-------------------------------------After First Order Differencing----------------------------------#

# Unit Root Test
# Check the ADF test by performing the first order differencing to the dataset.
adf.test(diff(fertility,1))
# H0: The time series is not stationary
# H1: The time series is stationary
# Reject H0 if the p-value is less than 0.05
# Since the p-value in the ADF test (0.107) is greater than 0.05, we do not reject the null hypothesis,
# hence we conclude that the data is not stationary. The second order differencing is required.
```

```
> adf.test(diff(fertility,1))

        Augmented Dickey-Fuller Test

data:  diff(fertility, 1)
Dickey-Fuller = -3.1568, Lag order = 3, p-value = 0.107
alternative hypothesis: stationary
```

```
# KPSS Test
# Check the KPSS test by performing the first order differencing to the dataset.
kpss.test(diff(fertility, 1), null = "Trend")
# H0: The time series is trend stationary
# H1: The time series is not trend stationary
# Reject H0 if the p-value is less than 0.05
# Since the p-value is less than 0.05, we reject the null hypothesis and conclude that the series is
# likely not trend stationary, meaning that it exhibits characteristics of a unit root or some form
# of non-stationarity in the presence of a deterministic trend.
```

```
> kpss.test(diff(fertility, 1), null = "Trend")

        KPSS Test for Trend Stationarity

data:  diff(fertility, 1)
KPSS Trend = 0.16656, Truncation lag parameter = 3, p-value = 0.03287
```

```
#-------------------------------------After Second Order Differencing---------------------------------#

# Unit Root Test
# Second order differencing
adf.test(diff(diff(fertility,1),1))
# H0: The time series is not stationary
# H1: The time series is stationary
# Reject H0 if the p-value is less than 0.05
# Now, the p-value is less than 0.05. Hence we reject the null hypothesis and conclude that the time
# series data is now stationary.
```

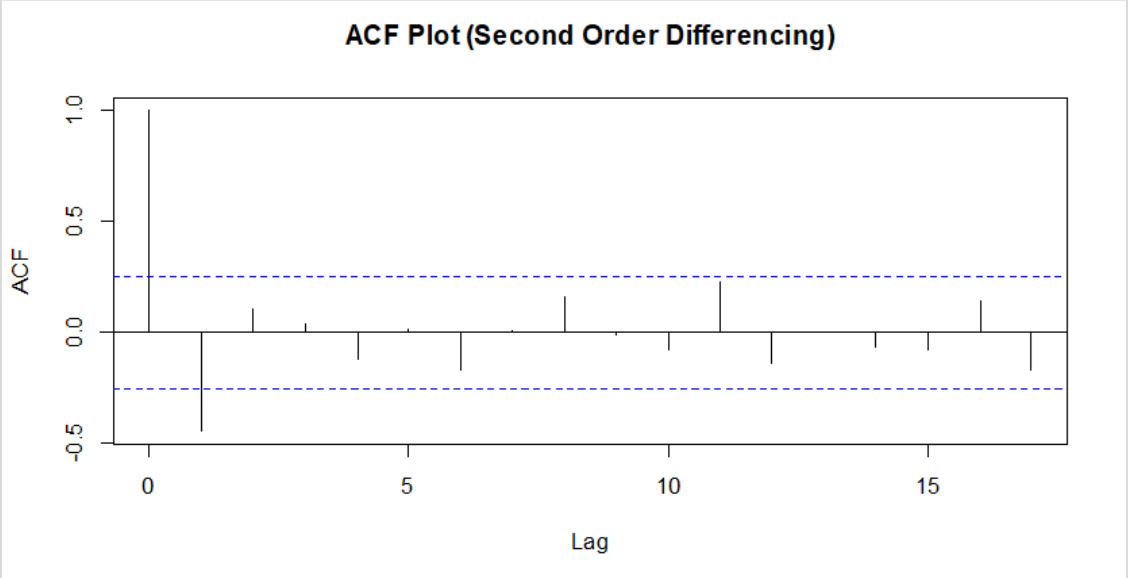
```
> adf.test(diff(diff(fertility,1),1))

        Augmented Dickey-Fuller Test

data:  diff(diff(fertility, 1), 1)
Dickey-Fuller = -5.6374, Lag order = 3, p-value = 0.01
alternative hypothesis: stationary
```

```
# KPSS
# Check the KPSS test by performing the second order differencing to the dataset.
kpss.test(diff(diff(fertility, 1), 1), null = "Trend")
# H0: The time series is trend stationary
# H1: The time series is not trend stationary
# Reject H0 if the p-value is less than 0.05
# Since the p-value is 0.1, which is greater than 0.05, we do not reject the null hypothesis and conclude
# that the series is now trend stationary.
```

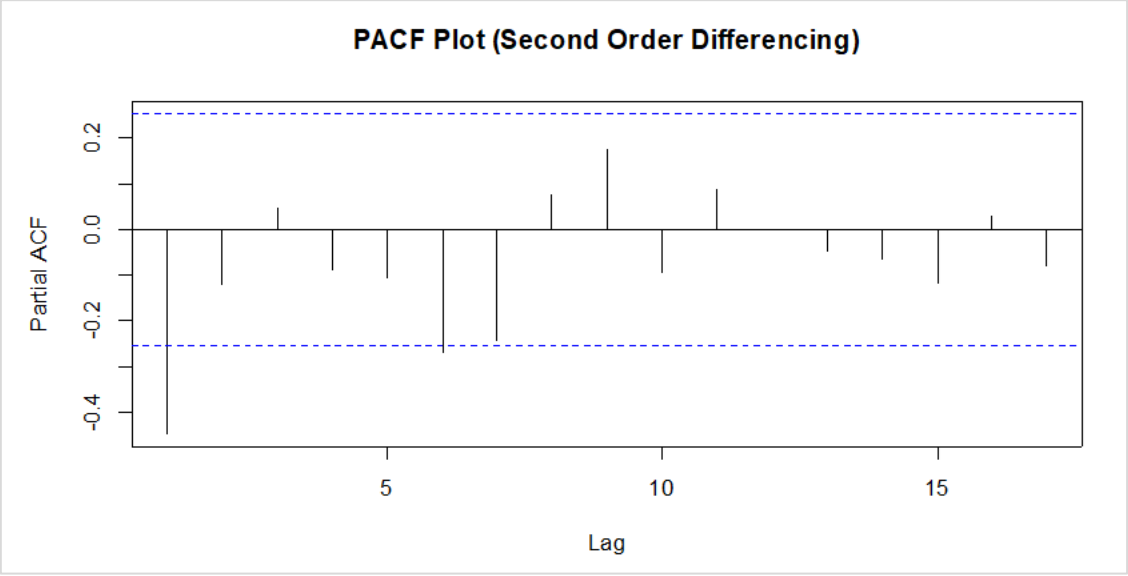
```
> kpss.test(diff(diff(fertility, 1), 1), null = "Trend")

        KPSS Test for Trend Stationarity

data:  diff(diff(fertility, 1), 1)
KPSS Trend = 0.055005, Truncation lag parameter = 3, p-value = 0.1
```

Section (f) (i)

```
#-----------------------------------------Model Suggestion-----------------------------------------#

# Model suggestion based on Box-Jenkins Methodology
# Show the data patterns in ACF plot for the second order differencing with title
acf(diff(diff(fertility,1),1), main="ACF Plot (Second Order Differencing)")
```




```
# Show the data patterns in PACF plot for the second order differencing with title
pacf(diff(diff(fertility,1),1), main="PACF Plot (Second Order Differencing)")
# Both plots suggests the ARIMA(1,2,1) model
```



```
# Check the suggestion by auto.arima()
auto.arima(fertility, trace=TRUE)
# ARIMA(0,2,1)
```

```
> auto.arima(fertility, trace=TRUE)

 ARIMA(2,2,2)                    : -124.5266
 ARIMA(0,2,0)                    : -112.3934
 ARIMA(1,2,0)                    : -125.0228
 ARIMA(0,2,1)                    : -128.1283
 ARIMA(1,2,1)                    : -126.8624
 ARIMA(0,2,2)                    : -126.4347
 ARIMA(1,2,2)                    : -125.7165

 Best model: ARIMA(0,2,1)

Series: fertility
ARIMA(0,2,1)

Coefficients:
          ma1
      -0.7222
s.e.   0.1410

sigma^2 = 0.006481:  log likelihood = 66.17
AIC=-128.34   AICc=-128.13   BIC=-124.15
```

Section (f) (ii)

```
#-------------------------------------Summary Outputs of ARIMA models-----------------------------#

# Estimate the ARIMA(1,2,1) model
arima_fertility_121 <-Arima(fertility,order=c(1,2,1))

# Summary output of ARIMA(1,2,1) model
summary(arima_fertility_121)
```

```
> summary(arima_fertility_121)
Series: fertility
ARIMA(1,2,1)

Coefficients:
         ar1      ma1
      0.2024  -0.8434
s.e.  0.1930   0.1132

sigma^2 = 0.00647:  log likelihood = 66.65
AIC=-127.29   AICc=-126.86   BIC=-121.01

Training set error measures:
                      ME       RMSE        MAE       MPE    MAPE      MASE       ACF1
Training set 0.0006417363 0.07779877 0.06422504 0.1764096 2.83581 0.8706061 0.02054931
```

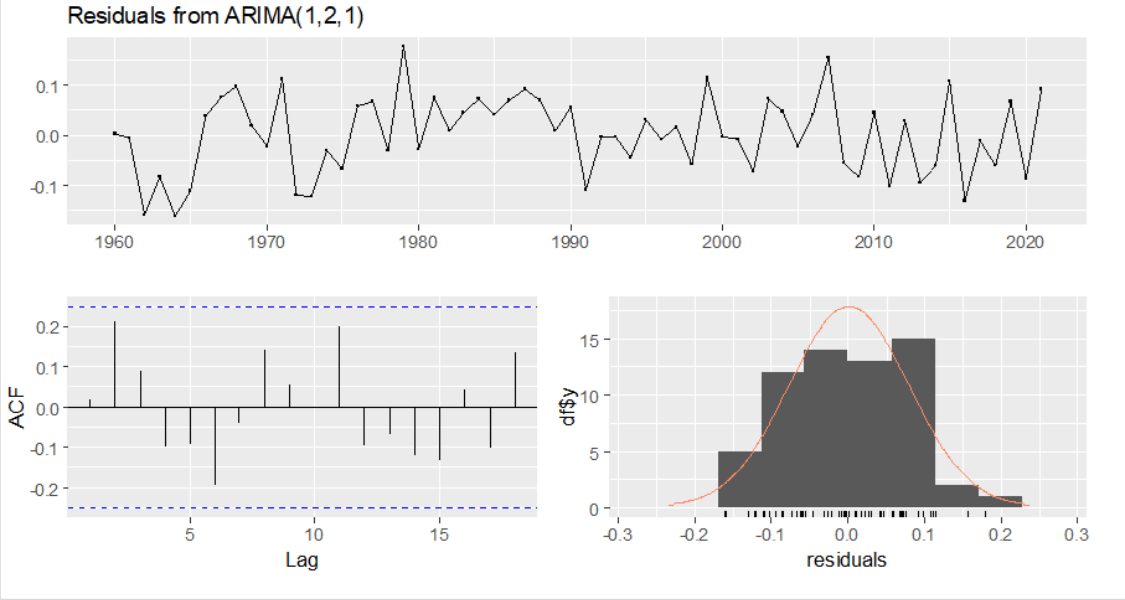
```
# Estimate the ARIMA(0,2,1)
arima_fertility_021 <-Arima(fertility,order=c(0,2,1))

# Summary output of ARIMA(0,2,1) model
summary(arima_fertility_021)
```

```
> summary(arima_fertility_021)
Series: fertility
ARIMA(0,2,1)

Coefficients:
          ma1
      -0.7222
s.e.   0.1410

sigma^2 = 0.006481:  log likelihood = 66.17
AIC=-128.34   AICc=-128.13   BIC=-124.15

Training set error measures:
                       ME      RMSE        MAE        MPE     MAPE      MASE      ACF1
Training set -0.001266659 0.0785315 0.06404357 0.09027092 2.785506 0.8681462 0.1305416
```

Section (f) (iv)

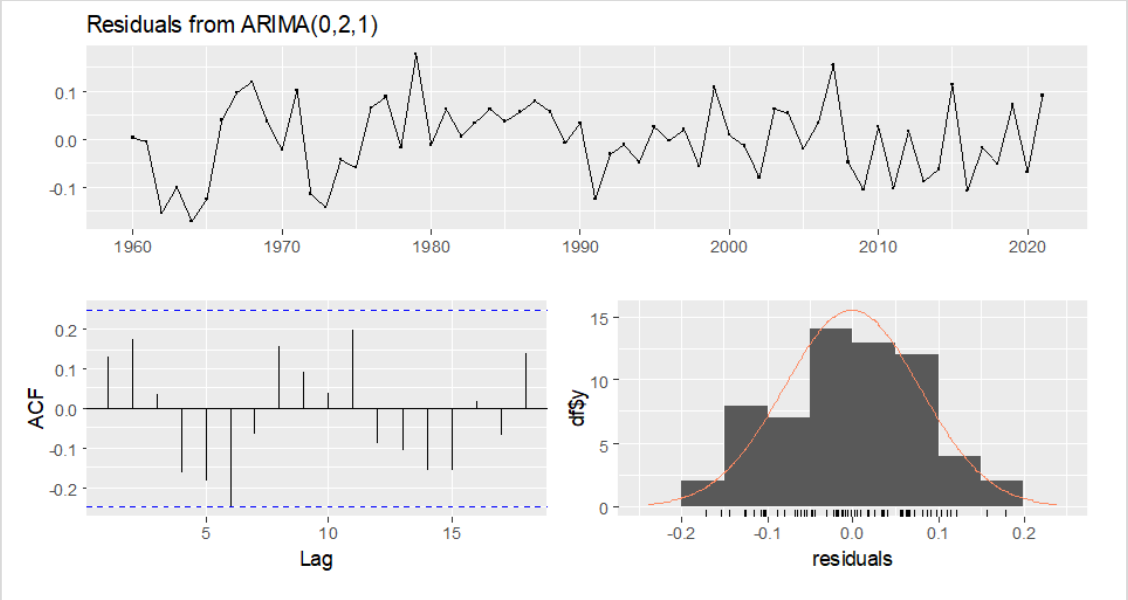
```
#-----------------------------------------------Model Adequacy-----------------------------------------------#

# Model adequacy of ARIMA(1,2,1) model
checkresiduals(arima_fertility_121)
# H0: The model is adequate
# H1: The model is inadequate
# Reject H0 if p-value is less than 0.05
# Since the p-value(0.3139) is greater than 0.05, we do not reject the null hypothesis. There is
# insufficient evidence to conclude that the model is inadequate at 5% level of significance.
```

```
> checkresiduals(arima_fertility_121)

        Ljung-Box test

data:  Residuals from ARIMA(1,2,1)
Q* = 9.3471, df = 8, p-value = 0.3139

Model df: 2.   Total lags used: 10
```




```
# Model adequacy of ARIMA(0,2,1) model
checkresiduals(arima_fertility_021)
# H0: The model is adequate
# H1: The model is inadequate
# Reject H0 if p-value is less than 0.05
# Since the p-value(0.09844) is greater than 0.05, we do not reject the null hypothesis. There is
# insufficient evidence to conclude that the model is inadequate at 5% level of significance.
```

```
> checkresiduals(arima_fertility_021)

        Ljung-Box test

data:  Residuals from ARIMA(0,2,1)
Q* = 14.736, df = 9, p-value = 0.09844

Model df: 1.   Total lags used: 10
```



Section (f) (v)

```
#---------------------------------------Significance of Coefficients-----------------------------------#

# Coefficient significance of ARIMA(1,2,1) model
coeftest(arima_fertility_121)
# From the summary output, the ar1 parameter is the only parameter that is insignificant. ma1 on the other
# hand was found to be a significant parameter.
```

```
> coeftest(arima_fertility_121)

z test of coefficients:

    Estimate Std. Error z value  Pr(>|z|)
ar1  0.20236    0.19300  1.0485    0.2944
ma1 -0.84341    0.11324 -7.4482 9.465e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
# Coefficient significance of ARIMA(0,2,1) model
coeftest(arima_fertility_021)
# From the summary output, the ma1 parameter was found to be a significant parameter.
```

```
> coeftest(arima_fertility_021)

z test of coefficients:

    Estimate Std. Error z value  Pr(>|z|)
ma1 -0.72216    0.14104 -5.1204 3.049e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Section (f) (vi)

```
#-------------------------------------------------Model Accuracy-------------------------------------------#

# Model accuracy of ARIMA(1,2,1) model
accuracy(arima_fertility_121)
# MAE: 0.06422504
# RMSE: 0.07779877
```

```
> accuracy(arima_fertility_121)
                      ME       RMSE        MAE       MPE     MAPE      MASE       ACF1
Training set 0.0006417363 0.07779877 0.06422504 0.1764096 2.83581 0.8706061 0.02054931
```

```
# Model accuracy of ARIMA(0,2,1) model
accuracy(arima_fertility_021)
# MAE: 0.06404357
# RMSE: 0.0785315
```

```
> accuracy(arima_fertility_021)
                      ME      RMSE        MAE        MPE     MAPE      MASE      ACF1
Training set -0.001266659 0.0785315 0.06404357 0.09027092 2.785506 0.8681462 0.1305416
```

Section (g)

```
#------------------------------------------------Generate Forecast-----------------------------------------------#

# Generate the forecast for the next 10 years
forecast_fertility <- forecast(arima_fertility_021, h=10)
forecast_fertility
```

```
> forecast_fertility
     Point Forecast     Lo 80    Hi 80      Lo 95    Hi 95
2022       1.604425 1.5012558 1.707594 1.44664135 1.762209
2023       1.568850 1.4014465 1.736253 1.31282855 1.824871
2024       1.533275 1.3013626 1.765187 1.17859565 1.887954
2025       1.497700 1.1984250 1.796974 1.03999857 1.955401
2026       1.462125 1.0919716 1.832278 0.89602431 2.028225
2027       1.426550 0.9818356 1.871264 0.74641827 2.106681
2028       1.390975 0.8680213 1.913928 0.59118649 2.190763
2029       1.355400 0.7505975 1.960202 0.43043459 2.280364
2030       1.319824 0.6296579 2.009991 0.26430582 2.375343
2031       1.284249 0.5053037 2.063195 0.09295473 2.475544
```

```
plot(forecast_fertility, main="Forecast of New Zealand's Fertility Rate in the Next 10 Years", xlab="Year",
     ylab="Fertility Rate")
lines(forecast_fertility$fitted, col=2, lwd=2)
legend("topright", c("Actual", "ARIMA(0,2,1)", "Forecast"), col=c(1,2,4), lwd=2, cex=0.8)
```