Data 624 Homework 6 Chapter 9.11

Enid Roman

2024-10-13

```r
# Load required libraries

library(fpp3)
library(tsibble)
library(ggplot2)
library(tidyverse)
library(forecast)
#install.packages("latex2exp")
library(latex2exp)
#install.packages("imager")
library(imager)
```

1. Figure 9.32 shows the ACFs for 36 random numbers, 360 random numbers and 1,000 random numbers.

a. Explain the differences among these figures. Do they all indicate that the data are white noise?

```r
knitr::include_graphics("C:/Users/enidr/OneDrive/Documents/CUNY SPS DATA 624/
Data 624 Week 7 HW 6/Image 9.32.png")
```

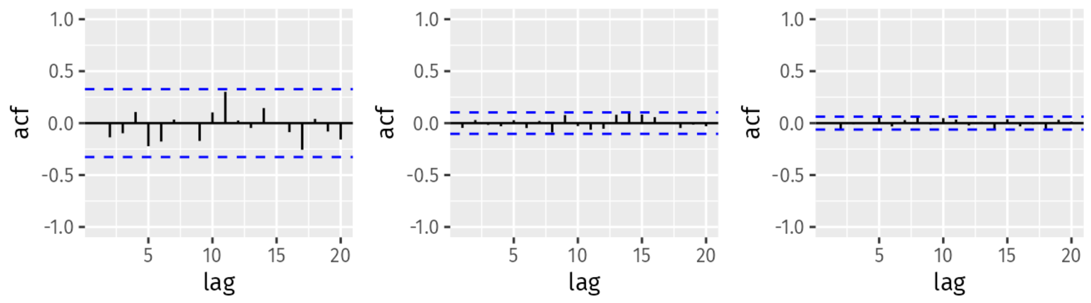


Figure 9.32: Left: ACF for a white noise series of 36 numbers. Middle: ACF for a white noise series of 360 numbers. Right: ACF for a white noise series of 1,000 numbers.

Differences among the plots:

Left plot (36 numbers): This ACF plot shows a lot of variability and some spikes beyond the confidence interval,represented by the dashed blue lines. With only 36 observations, it’s harder to get a clear pattern, and random fluctuations may appear more pronounced. Some of the lag values might seem significant even though the data is white noise, simply because of the small sample size.

Middle plot (360 numbers): The plot looks more stable than the left one, and most of the autocorrelations are close to zero, falling within the confidence intervals. The larger sample size, 360 observations, gives a more accurate picture of the underlying white noise process, so there are fewer false signals of autocorrelation.

Right plot (1,000 numbers): This plot looks very stable, with almost all the points very close to zero and well within the confidence intervals. The large sample size,1,000 observations, gives an even clearer indication that the data is white noise. The randomness is more evenly distributed, and spurious spikes in autocorrelation have been minimized.

White noise determination:

In all three plots, despite some apparent spikes in the smaller sample sizes (left and middle), the data likely represents white noise because, overall, the ACF values hover around zero, and there's no systematic autocorrelation pattern. The deviations in the left plot are likely due to the small sample size.

As the sample size increases, the ACF plots become smoother and show that the autocorrelation values are consistently close to zero, indicating a more precise alignment with white noise properties.

b. Why are the critical values at different distances from the mean of zero?

The critical values (confidence intervals) are at different distances from the mean of zero in each plot because the width of the confidence intervals depends on the sample size. the confidence interval for the autocorrelation is approximately ±1.96/square root of N, where N is the number of observations in the time series.

Smaller sample sizes (e.g., 36 numbers) have wider confidence intervals because there is more uncertainty in estimating autocorrelations.

Larger sample sizes (e.g., 360 or 1,000 numbers) have narrower confidence intervals, reflecting more precise estimates of autocorrelations.

Why are the autocorrelations different in each figure when they each refer to white noise?

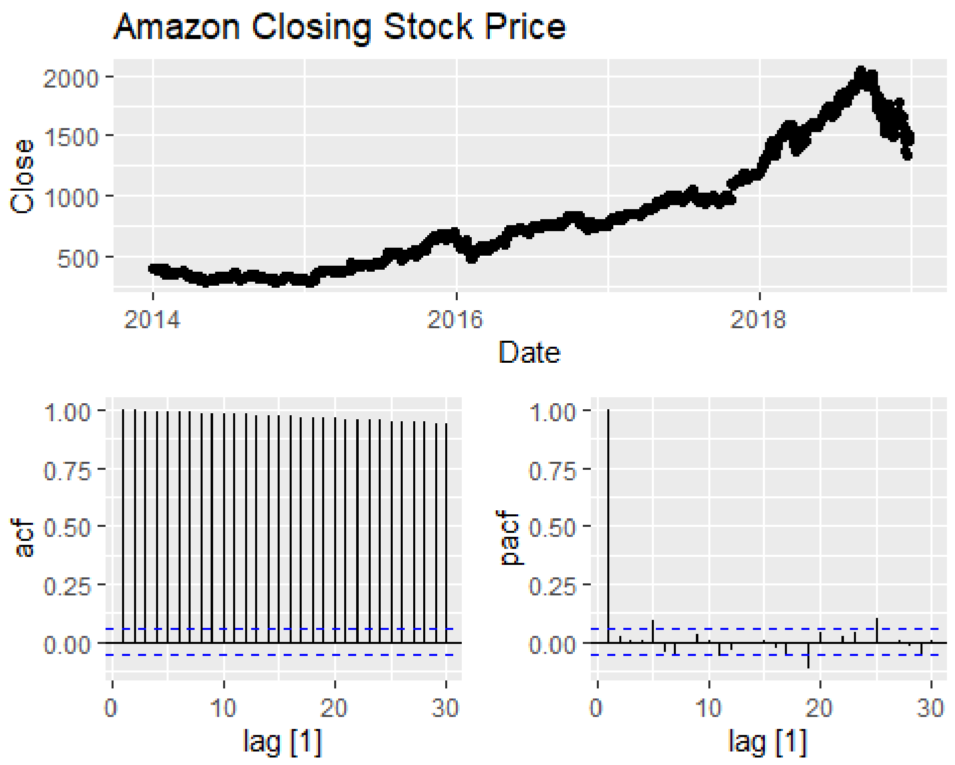
In a white noise process, the true autocorrelations at all non-zero lags should be zero. However, due to sampling variability, the estimated autocorrelations for a finite sample are not exactly zero and fluctuate randomly.

Smaller samples (like the 36-number series) are more likely to show random spikes in autocorrelation, which can deviate from zero due to the limited number of observations. Larger samples (like the 360 or 1,000-number series) tend to have estimated autocorrelations that are much closer to zero because the increased sample size reduces random fluctuations and provides more accurate estimates.

The different sample sizes cause variations in both the width of the critical values and the estimated autocorrelations, even though the underlying process is white noise in all cases.

2. A classic example of a non-stationary series are stock prices. Plot the daily closing prices for Amazon stock (contained in gafa_stock), along with the ACF and PACF. Explain how each plot shows that the series is non-stationary and should be differenced.

```
gafa_stock %>%
  filter(Symbol == "AMZN") %>%
  gg_tsdisplay(Close, plot_type='partial') +
  labs(title = "Amazon Closing Stock Price")
```



```
gafa_stock %>%
  filter(Symbol == "AMZN") %>%
  features(Close, unitroot_ndiffs)

## # A tibble: 1 × 2
##   Symbol ndiffs
##   <chr>   <int>
## 1 AMZN        1
```

The plot of Amazon's daily closing prices over time reveals a clear upward trend. This suggests that the series has a non-constant mean, which is a hallmark of non-stationarity. A stationary series should fluctuate around a constant mean, but the upward trend here suggests that the mean is increasing over time. This kind of trend is typical of non-stationary time series. Differencing can remove this trend by transforming the series into the changes between successive observations, which can stabilize the mean and potentially make the series stationary.

The ACF plot exhibits very high correlations at multiple lags, and the autocorrelations decay very slowly. For a stationary series, the ACF should drop off quickly to near-zero values. In contrast, the slow decay in the ACF suggests a long-term dependence between observations, characteristic of non-stationary data. The slow decay implies that past values have a lasting influence on future values, a sign that differencing is necessary to remove these long-term dependencies and make the series stationary.

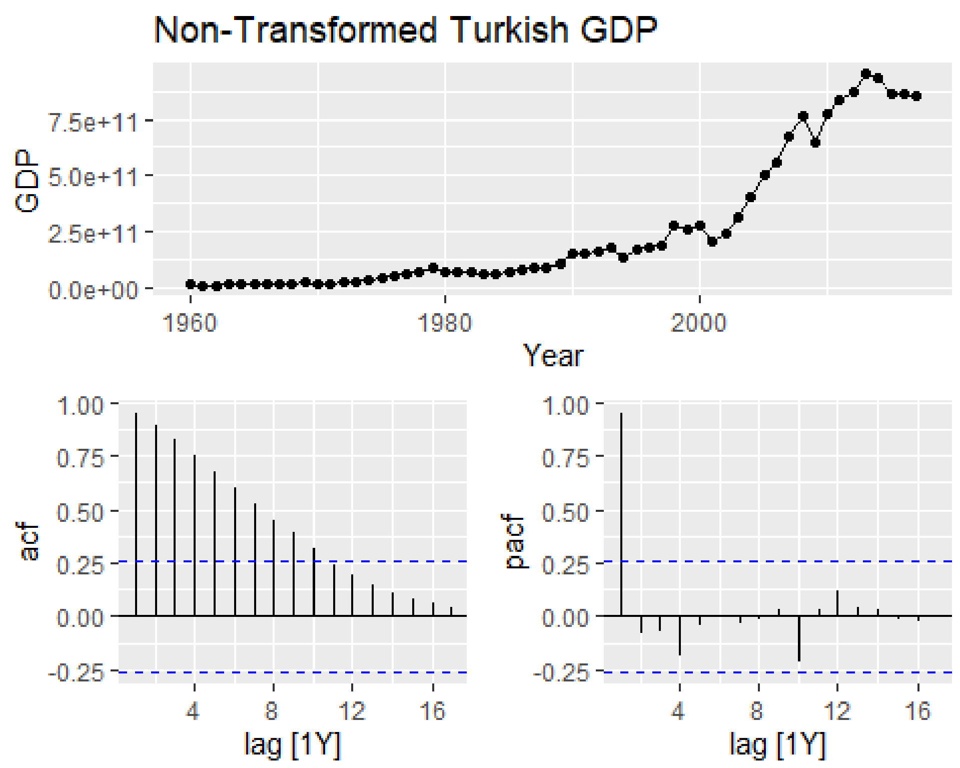
The PACF plot has a significant spike at lag 1, followed by a quick drop-off to near-zero values. A large spike at lag 1 indicates that the first difference between consecutive observations can explain much of the variation in the series. This pattern often suggests that the series has a trend and can be made stationary by first-order differencing. The strong lag-1 autocorrelation points to the need for first-order differencing, which would likely remove the trend and stabilize the mean, making the series more suitable for further time series modeling.

All of these characteristics together indicate that the Amazon stock price series is non-stationary and should be differenced to remove the trend and make the series stationary. The analysis confirms that first-order differencing (1 difference) should be applied to the series, as indicated by the ndiffs = 1 result.

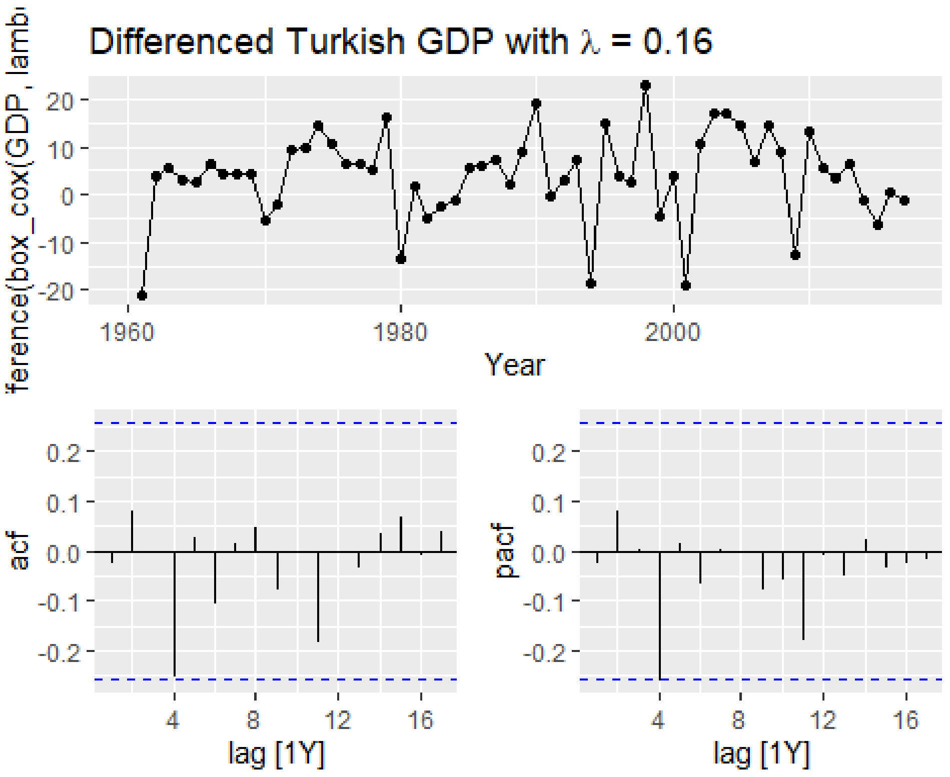
3. For the following series, find an appropriate Box-Cox transformation and order of differencing in order to obtain stationary data.

a. Turkish GDP from global_economy.

```r
# Filter for Turkish GDP and plot the non-transformed series
global_economy %>%
  filter(Country == "Turkey") %>%
  gg_tsdisplay(GDP, plot_type = 'partial') +
  labs(title = "Non-Transformed Turkish GDP")
```



```
# filter and compute features for Turkish GDP
global_economy %>%
  filter(Country == "Turkey") %>%
  features(GDP, list(lambda = guerrero, ndiffs = unitroot_ndiffs))

## # A tibble: 1 × 3
##   Country lambda_lambda_guerrero ndiffs_ndiffs
##   <fct>                    <dbl>         <int>
## 1 Turkey                   0.157             1

# Estimate the lambda using the guerrero method
lambda <- global_economy %>%
  filter(Country == "Turkey") %>%
  features(GDP, features = guerrero) %>%
  pull(lambda_guerrero)

# Apply Box-Cox transformation and differencing, then visualize
global_economy %>%
  filter(Country == "Turkey") %>%
  gg_tsdisplay(difference(box_cox(GDP, lambda)), plot_type = 'partial') +
  labs(title = TeX(paste0("Differenced Turkish GDP with $\\lambda$ = ",
         round(lambda, 2))))
```



The Non-Transforemed Turkish GDP shows a clear upward trend, especially in recent decades. This means the series is non-stationary, as the mean of the series changes over time (increasing trend).

A stationary series should have a constant mean and variance over time. In this plot, the trend suggests that the data are not stationary, and transformations such as differencing may be needed to remove the trend and stabilize the mean.

The ACF plot displays significant autocorrelations across many lags (gradually decaying), indicating that each value in the series is correlated with its preceding values over a long period.

In a stationary series, the ACF should drop off quickly to near-zero values within a few lags. The slow decay in the ACF is characteristic of a non-stationary time series with a trend, which again confirms that differencing is likely necessary to make the series stationary.

The PACF plot shows a strong spike at lag 1, with subsequent lags showing relatively weaker spikes.

To make this series stationary, I would need to apply a first-order difference and a Box-Cox transformation if the variance appears to change significantly over time.

After applying a Box-Cox transformation with lambada = 16 in the Differenced and Transformed Turkish GDP Time Series Plot, The transformation helps stabilize the variance of the series. Differencing removes the trend, making the data more stationary. We no longer see the upward trend present in the original series. The values now fluctuate around a mean of approximately zero, which is a sign of stationarity. The differencing has been effective in removing the trend, and the data seems more stationary now, with no clear upward or downward movement over time.

The ACF plot now shows that all autocorrelations fall well within the 95% confidence bounds (the dashed blue lines), with no significant autocorrelations at any lags.

The ACF should drop to near-zero quickly after the first lag, and that’s exactly what we see here. The absence of significant autocorrelations confirms that the differencing and transformation have made the series stationary.

The PACF plot shows a similar pattern, with no significant spikes at any lags beyond lag 1. The PACF values are small and within the confidence bounds.

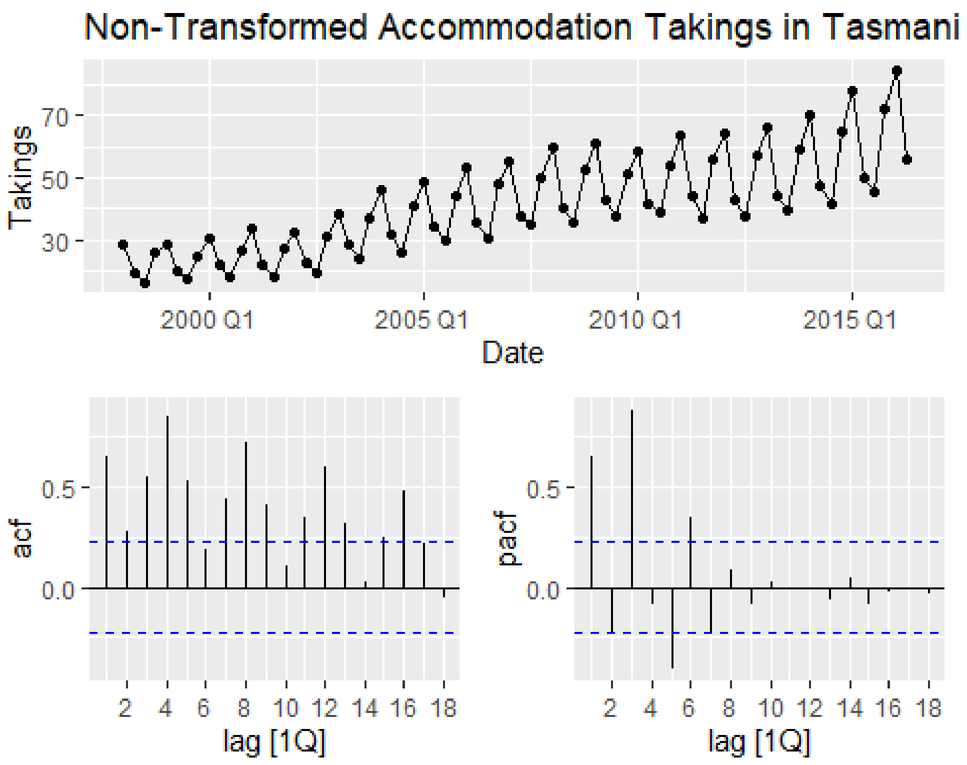
This is another confirmation that the series has been successfully transformed into a stationary one. The lack of significant spikes beyond the first lag indicates that there is no strong autocorrelation structure left in the data, which is typical of stationary data.

The Box-Cox transformation with lambada = 0.16 stabilized the variance, and the first-order differencing effectively removed the trend, resulting in a stationary series. Both the ACF and PACF plots confirm stationarity, as the autocorrelations and partial autocorrelations are close to zero beyond the first lag. The transformed and differenced series is now ready for time series modeling, such as ARIMA, where further analysis can be conducted on the stationary data.

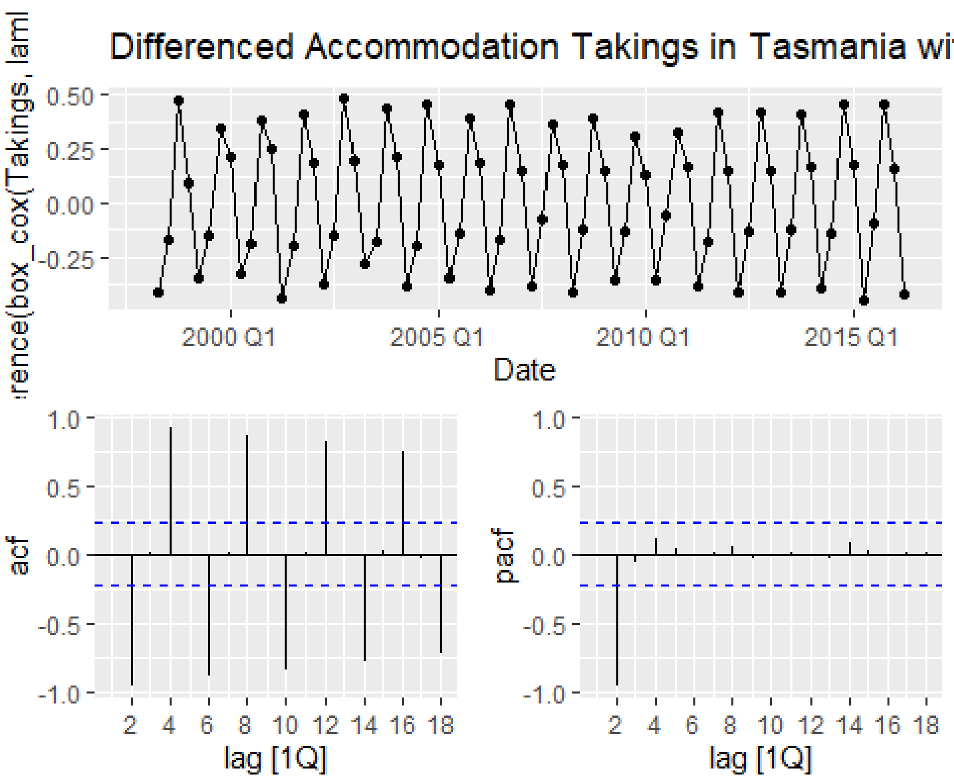
The transformation and differencing steps have successfully removed the trend and stabilized the variance, making the Turkish GDP series stationary for further modeling.

b. Accommodation takings in the state of Tasmania from aus_accommodation.

```
# Filter for accommodation takings in Tasmania and plot the non-transformed s
eries
aus_accommodation %>%
  filter(State == "Tasmania") %>%
  gg_tsdisplay(Takings, plot_type = 'partial') +
  labs(title = "Non-Transformed Accommodation Takings in Tasmania")
```



```
# Filter and compute features for accommodation takings in Tasmania
aus_accommodation %>%
  filter(State == "Tasmania") %>%
  features(Takings, list(lambda = guerrero, ndiffs = unitroot_ndiffs))

## # A tibble: 1 × 3
##   State    lambda_lambda_guerrero ndiffs_ndiffs
##   <chr>                     <dbl>         <int>
## 1 Tasmania                0.00182             1

# Estimate the lambda using the guerrero method
lambda <- aus_accommodation %>%
  filter(State == "Tasmania") %>%
  features(Takings, features = guerrero) %>%
  pull(lambda_guerrero)

# Apply Box-Cox transformation and differencing, then visualize
aus_accommodation %>%
  filter(State == "Tasmania") %>%
  gg_tsdisplay(difference(box_cox(Takings, lambda)), plot_type = 'partial') +
  labs(title = TeX(paste0("Differenced Accommodation Takings in Tasmania with
$\\lambda$ = ",
         round(lambda, 2))))
```



The non-transformed accommodation takings data in Tasmania from 2000 to 2016 is a clear seasonal pattern with regular fluctuations that repeat every year, indicating strong seasonality. The overall trend appears upward, showing growth in accommodation takings over time. The variance increases slightly over time, as the fluctuations (peaks and troughs) grow larger, but the growth in variance is not extreme. The presence of a trend and seasonality suggests that the series is non-stationary, as stationary data should have a constant mean and variance over time without systematic patterns. I will need to remove the trend (through differencing) and stabilize the variance through Box-Cox transformation to make the series stationary.

The ACF plot shows significant spikes at multiple lags, especially at lag 1 and around lag 4 and lag 8. The high autocorrelations at early lags indicate a strong relationship between observations close in time, which suggests non-stationarity. The spikes every 4 lags likely correspond to the quarterly seasonality in the data, as accommodation takings might have a strong seasonal component related to tourism trends. In a stationary series, the ACF should drop off quickly to near-zero values after the first few lags. Here, the slow decay and high autocorrelations over multiple lags, especially at seasonal lags, indicate that the series is non-stationary due to both trend and seasonality.

The PACF plot shows a strong spike at lag 1, followed by smaller spikes at subsequent lags. The large spike at lag 1 suggests that the first difference could remove much of the trend in the series. The smaller spikes at subsequent lags, particularly around seasonal lags, lag 4, indicate the presence of seasonality that needs to be addressed. The significant lag-1 spike and the smaller seasonal spikes indicate that the series has both trend and seasonality, which make it non-stationary. First-order differencing and seasonal differencing may be required to remove these effects and make the data stationary.

First-order differencing is likely needed to remove the trend, as indicated by the significant PACF spike at lag 1. Seasonal differencing (quarterly, lag 4) might also be necessary to account for the repeating seasonal pattern in the data. Then stabilize the variance through Box-Cox transformation to make the series stationary.

After applying applying a Box-Cox transformation with lambada = 0, the strong seasonal pattern remains after differencing, with regular peaks and troughs repeating every year. Despite the differencing, the series still appears to have strong seasonal fluctuations, which means that further seasonal differencing might be needed to remove this seasonality. The fluctuations are now centered around zero, which indicates that the trend has been removed, but the seasonal component is still present. Differencing has removed the trend, but the seasonal pattern is still prominent, suggesting the need for seasonal differencing. This indicates that first-order differencing was successful in removing the overall trend but did not completely address the seasonality in the data.

The ACF plot shows significant spikes at seasonal lags, particularly at lag 4, 8, 12, and so on, which corresponds to quarterly seasonality. The significant spikes at lag 4 and its multiples suggest that there is still strong seasonal autocorrelation in the data. The slow decay of autocorrelations is indicative of the seasonal component that has not been fully removed by the first-order differencing. The presence of large spikes at seasonal lags in the ACF suggests that seasonal differencing,likely at lag 4, will be necessary to fully remove the seasonal autocorrelations and make the series stationary.

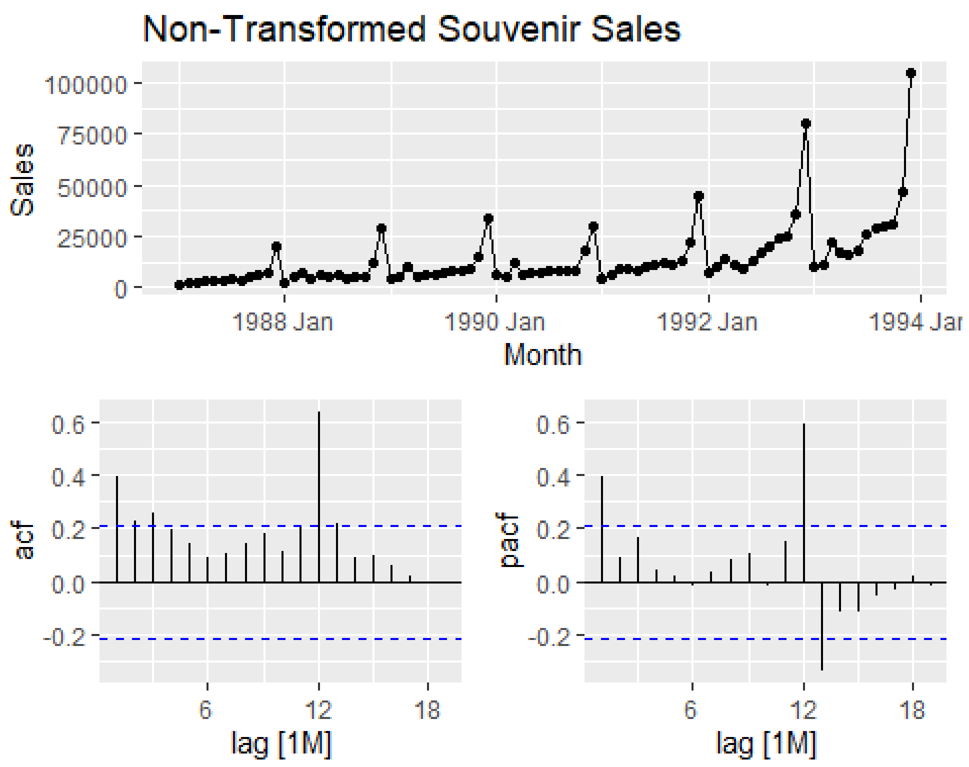
The PACF plot has a significant spike at lag 1 and smaller spikes at subsequent lags, but no significant seasonal pattern. The large spike at lag 1 suggests that the first difference,already applied, is capturing the short-term autocorrelation. This is typical when a series has trend and seasonality. The smaller spikes at higher lags suggest some residual seasonal correlation, though the PACF doesn’t show as clear a seasonal pattern as the ACF. The significant spike at lag 1 confirms that first-order differencing was necessary to remove the trend. However, the combination of ACF and PACF suggests that seasonal differencing is still needed to fully address the seasonality.

Box-Cox Transformation lambada = 0 helped stabilize the variance in the series, although variance was not a significant issue in this case. The first difference has successfully removed the trend, as indicated by the time series plot being centered around zero. The ACF plot shows strong seasonal autocorrelations at lags 4, 8, and beyond, indicating that the series still has a strong seasonal component.

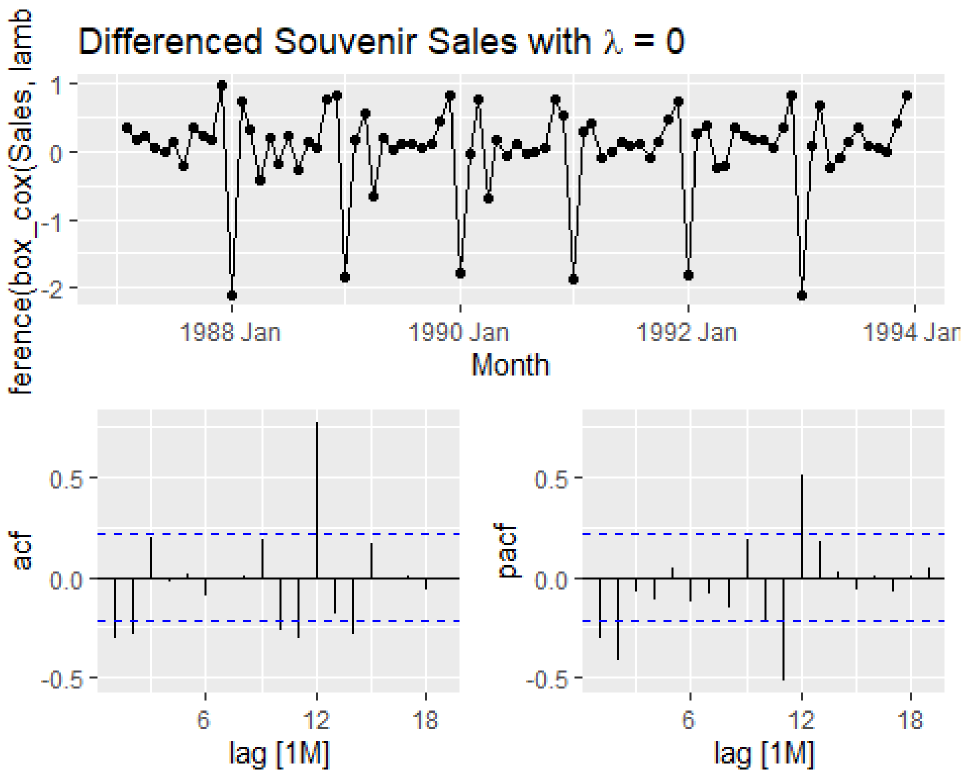
To make the series fully stationary, I would need to apply seasonal differencing at lag 4, quarterly, to remove the remaining seasonal structure. After that, I can re-check the ACF and PACF to ensure that the seasonal pattern has been adequately removed.

c. Monthly sales from souvenirs.

```
# Filter for the souvenir sales and plot the non-transformed series
souvenirs %>%
  gg_tsdisplay(Sales, plot_type = 'partial') +
  labs(title = "Non-Transformed Souvenir Sales")
```



```
# Filter and compute features for souvenir sales
souvenirs %>%
  features(Sales, list(lambda = guerrero, ndiffs = unitroot_ndiffs))

## # A tibble: 1 × 2
##   lambda_lambda_guerrero ndiffs_ndiffs
##                    <dbl>         <int>
## 1                0.00212             1

# Estimate the lambda using the guerrero method for souvenir sales
lambda <- souvenirs %>%
  features(Sales, features = guerrero) %>%
  pull(lambda_guerrero)

# Apply Box-Cox transformation and differencing, then visualize for souvenir
sales
souvenirs %>%
  gg_tsdisplay(difference(box_cox(Sales, lambda)), plot_type = 'partial') +
  labs(title = TeX(paste0("Differenced Souvenir Sales with $\\lambda$ = ",
         round(lambda, 2))))
```



Time Series Plot of Non-Transformed Souvenir Sales represents the raw monthly sales of souvenirs from the mid-1980s to the early 1990s. The series exhibits a repeating pattern with regular peaks and troughs, likely reflecting seasonality. Over time, the series shows a clear upward trend, especially towards the end where sales spike dramatically. The size of the fluctuations, peaks and troughs, appears to grow over time, indicating that the variance is increasing as sales increase. The upward trend shows that the mean is changing over time, a clear sign of non-stationarity. The increasing variance over time suggests the need for a transformation to stabilize it. The strong seasonality suggests a periodic component in the data that needs to be addressed.

The ACF plot displays significant autocorrelations at various lags. There is a strong spike at lag 12, which corresponds to the annual seasonality, 12 months = 1 year. This indicates that sales from the same month in previous years are highly correlated. The slow decay in the ACF suggests the presence of both trend and seasonality. The significant autocorrelations at lag 12 and other lags suggest that the series has a strong seasonal component and possibly a trend that causes long-term dependencies between observations.

The PACF plot shows significant spikes at lag 1 and lag 12, followed by smaller spikes. The spike at lag 1 indicates a short-term autocorrelation, which suggests that first-order differencing may help remove the trend. The spike at lag 12 further confirms the presence of annual seasonality. The strong spike at lag 1 indicates the presence of a trend, and the spike at lag 12 confirms seasonality. These patterns suggest that both differencing and seasonal differencing are needed to remove these effects and make the series stationary.

Since the variance increases over time, applying a Box-Cox transformation will help stabilize the variance. Differencing the series once will help remove the trend, as indicated by the significant spike at lag 1 in the PACF. Given the strong seasonal component at lag 12, applying seasonal differencing (with a period of 12 months) will help remove the seasonal pattern.

The Transformed and Differenced Time Series Plot represents the monthly souvenir sales after applying a logarithmic transformation Box-Cox with lambada = 0 and the first order differencing. The seasonality is still very evident, with repeating peaks and troughs occurring at regular intervals, likely reflecting annual cycles in sales. The series now fluctuates around zero, indicating that the trend has been successfully removed. The amplitude of the fluctuations has been stabilized by the logarithmic transformation, addressing the increasing variance observed in the non-transformed series. The trend has been successfully removed by differencing, but the strong seasonal pattern remains. This suggests that further seasonal differencing, at lag 12, corresponding to annual seasonality, may be necessary to fully remove the seasonality.

The ACF plot now shows smaller autocorrelations, but there are still noticeable spikes at lag 12 and its multiples, lag 12 and lag 24, indicating residual seasonality. The significant spikes at lag 12 suggest that there is still seasonal autocorrelation in the data, even after first-order differencing. The smaller spikes at other lags indicate that the autocorrelation from the trend has been mostly removed, but the seasonality remains. The ACF shows that although first-order differencing was effective in removing the trend, the series still has a strong seasonal component that needs to be addressed. Seasonal differencing at lag 12 should help remove this seasonal autocorrelation.

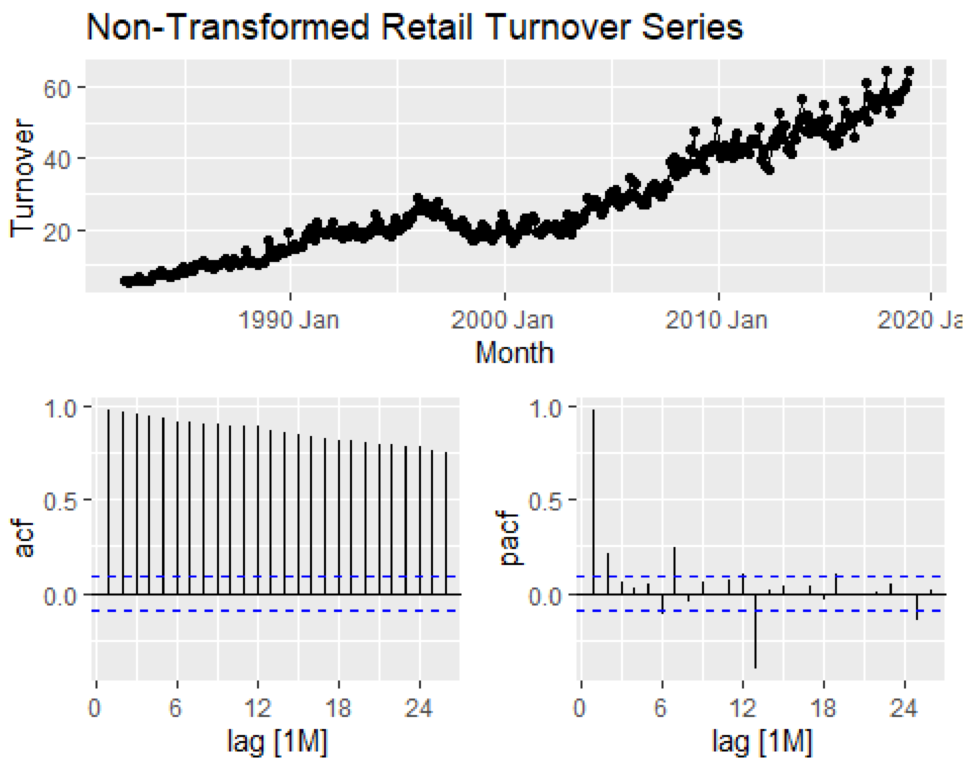
The PACF plot shows a significant spike at lag 12, with smaller spikes at earlier lags. The spike at lag 12 in the PACF plot confirms the seasonal effect, which was also observed in the ACF plot. The smaller spikes at other lags indicate that there is no strong autocorrelation at lower lags, which is expected after removing the trend with first-order differencing. The spike at lag 12 suggests that seasonal differencing at lag 12 (corresponding to annual seasonality) is still required to make the series fully stationary.

Box-Cox transformation lambada = 0 was effective in stabilizing the variance, as seen in the more consistent fluctuations in the transformed and differenced series. The first-order differencing successfully removed the trend, as the differenced series is now centered around zero. The Seasonality, significant spikes at lag 12 in both the ACF and PACF plots indicate strong residual seasonality that still needs to be addressed.

Seasonal differencing at lag 12 should be applied to remove the remaining seasonality. This will make the series more suitable for time series modeling, such as ARIMA or SARIMA, which require stationarity.

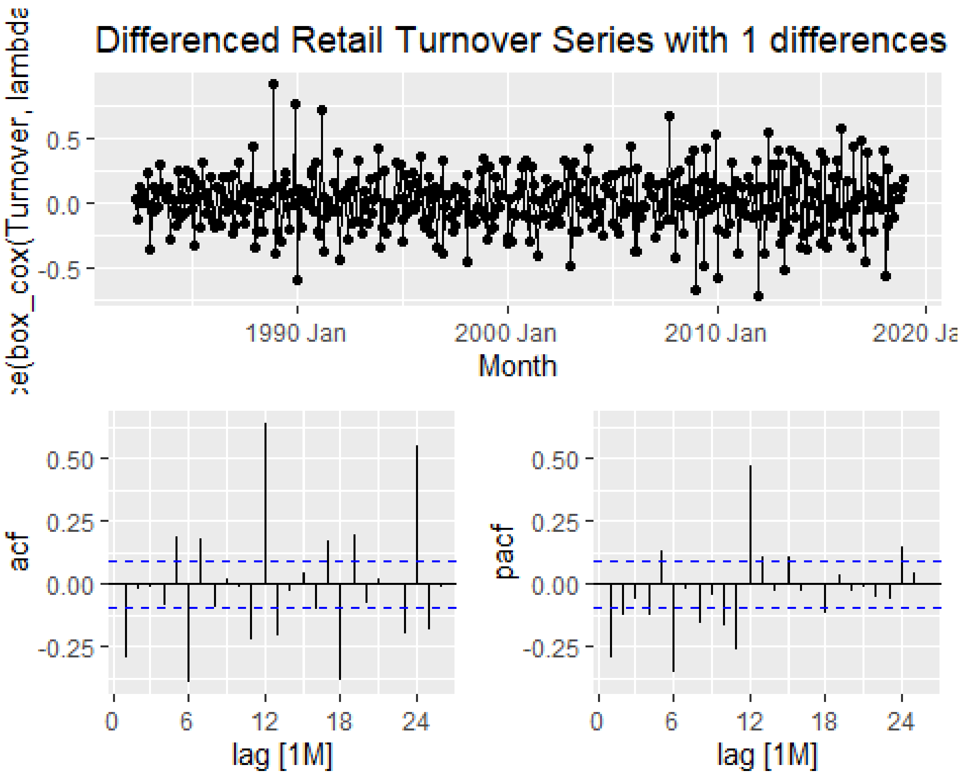
5. For your retail data (from Exercise 7 in Section 2.10), find the appropriate order of differencing (after transformation if necessary) to obtain stationary data.

```
# Set seed for reproducibility
set.seed(1234)

# Filter for a random series from the aus_retail dataset
myseries <- aus_retail %>%
  filter(`Series ID` == sample(aus_retail$`Series ID`, 1))

# Plot the non-transformed series
myseries %>%
  gg_tsdisplay(Turnover, plot_type = 'partial') +
  labs(title = "Non-Transformed Retail Turnover Series")
```



```
# Calculate lambda using the guerrero method and ndiffs (number of difference
s for stationarity)
lambda_diff <- myseries %>%
  features(Turnover, list(lambda = guerrero, ndiffs = unitroot_ndiffs))

# View the lambda_diff table with the computed features
lambda_diff

## # A tibble: 1 × 4
##   State    Industry                                 lambda_lambda_guerrero ndiffs_
```

```
ndiffs
##   <chr>    <chr>                                                <dbl>
<int>
## 1 Tasmania Cafes, restaurants and takeaway…                     0.320
1

# Extract lambda and ndiffs from the lambda_diff table
lambda <- lambda_diff %>% pull(lambda_lambda_guerrero)
n_diffs <- lambda_diff %>% pull(ndiffs_ndiffs)

# Print the extracted lambda and number of differences
lambda

## [1] 0.3196221

n_diffs

## [1] 1

# Apply Box-Cox transformation and differencing, then visualize the transform
ed series
myseries %>%
  gg_tsdisplay(difference(box_cox(Turnover, lambda), n_diffs), plot_type = 'p
artial') +
  labs(title = paste("Differenced Retail Turnover Series with", n_diffs, "dif
ferences and lambda =", round(lambda, 2)))
```



The ACF plot displays significant autocorrelations at various lags, particularly around the first few months, as well as some noticeable spikes around lag 12, indicating potential yearly seasonality (12 months = 1 year). The slow decay of the ACF implies that there is a trend and potential seasonality present in the data. This is further confirmed by the periodicity of the spikes at around 12 lags.

The PACF plot shows strong spikes at lag 1, indicating short-term autocorrelation, and also spikes at lag 12, confirming the annual seasonality. The significant spike at lag 1 suggests that first-order differencing might be necessary to remove the trend, while the spike at lag 12 suggests that seasonal differencing might be needed to handle the yearly seasonality.

Given the increasing variance over time, a Box-Cox transformation with $\lambda=0.32$ was applied to stabilize the variance. This transformation is effective for making the fluctuations more consistent across time. Next, first-order differencing was applied to remove the trend, as indicated by the significant spike at lag 1 in the PACF plot. The plot after differencing shows that the upward trend has been successfully removed, as the series now fluctuates around zero.

The Differenced Retail Turnover Series plot shows the transformed and differenced data. The seasonal pattern is still present, with repeating peaks and troughs, likely reflecting annual cycles in retail turnover. However, the trend has been removed, and the fluctuations appear more stable, thanks to the Box-Cox transformation.

After applying the Box-Cox transformation and differencing, the ACF plot shows smaller autocorrelations at most lags. However, there are still significant spikes at lag 12 and its multiples (lag 12, lag 24), indicating that some residual seasonality remains in the data. This suggests that while first-order differencing removed the trend, seasonal differencing may be necessary to remove the remaining seasonality.

The PACF plot shows a clear spike at lag 12, further confirming the seasonal component. There are no strong autocorrelations at the lower lags, which indicates that the first-order differencing effectively removed the trend. The spike at lag 12 suggests that seasonal differencing at lag 12 (corresponding to yearly seasonality) may still be needed to fully eliminate the seasonality and achieve stationarity.

Box-Cox Transformation ($\lambda=0.32$): The transformation was effective in stabilizing the variance, as the fluctuations are now more consistent over time.

First-order Differencing: The trend has been successfully removed, as the series is now centered around zero.

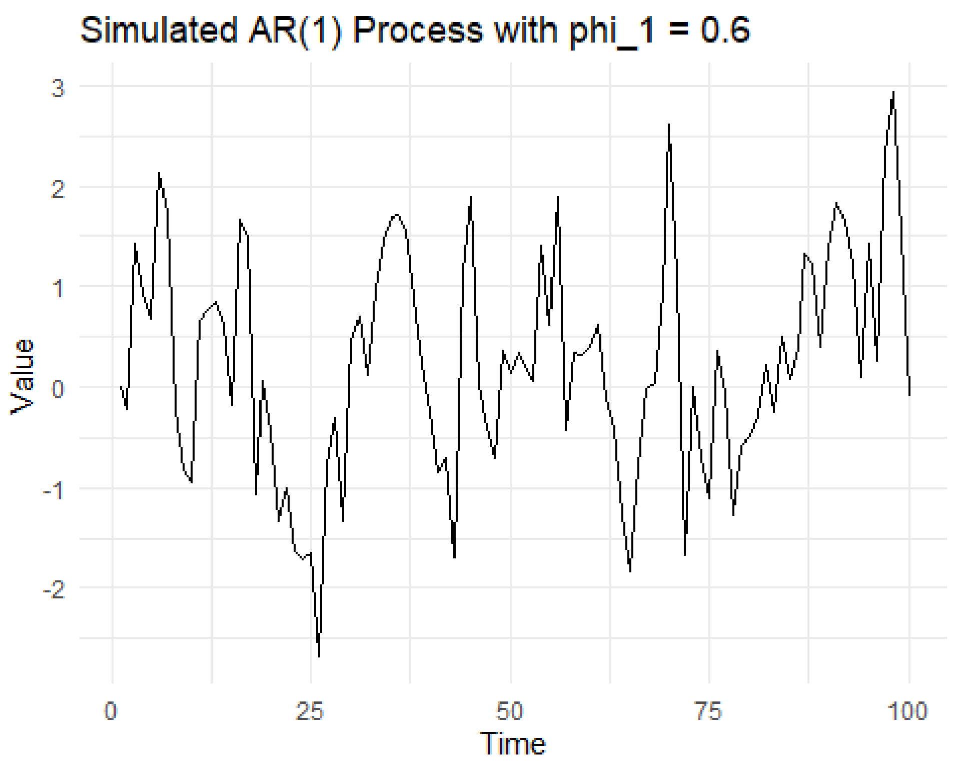
Seasonal Differencing: Based on the significant spikes at lag 12 in both the ACF and PACF plots, there is still a strong seasonal component in the data. Applying seasonal differencing at lag 12 should help remove this remaining seasonality and make the series fully stationary.

Next Steps for Modeling: Once seasonal differencing is applied, the series should be more suitable for time series modeling using methods like ARIMA or SARIMA, which require the data to be stationary. The seasonal differencing will remove the residual seasonality, enabling the model to capture the dynamics of the data more effectively.

6. Simulate and plot some data from simple ARIMA models.

a. Use the following R code to generate data from an AR(1) model with ϕ1=0.6 and σ2=1. The process starts with y1=0.

```
y <- numeric(100)
e <- rnorm(100)
for(i in 2:100)
  y[i] <- 0.6*y[i-1] + e[i]
sim <- tsibble(idx = seq_len(100), y = y, index = idx)

# Set seed for reproducibility
set.seed(123)

# Initialize variables
y <- numeric(100)  # Create a vector to hold 100 data points
e <- rnorm(100)    # Generate 100 random normal errors (with variance 1)

# Simulate AR(1) process with phi_1 = 0.6
y[1] <- 0  # Start the process with y1 = 0
for(i in 2:100) {
  y[i] <- 0.6 * y[i-1] + e[i]
}

# Convert to a tsibble for plotting
sim <- tsibble(idx = seq_len(100), y = y, index = idx)

# Plot the simulated AR(1) process
ggplot(sim, aes(x = idx, y = y)) +
  geom_line() +
  labs(title = "Simulated AR(1) Process with phi_1 = 0.6", x = "Time", y = "V
alue") +
  theme_minimal()
```



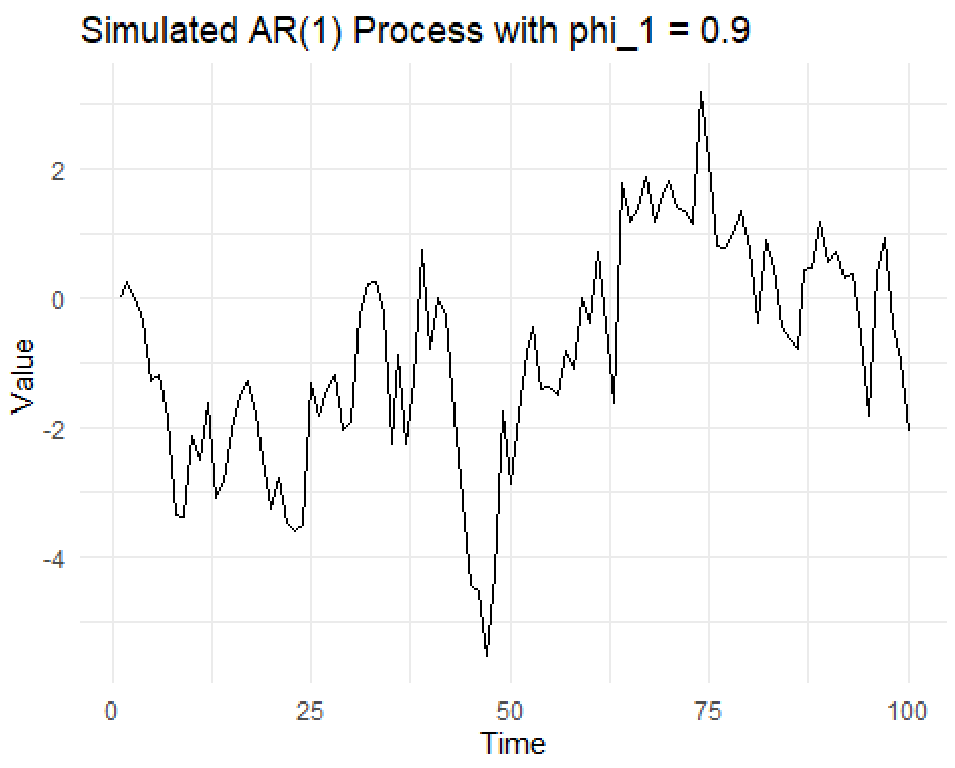
The value ofϕ1=0.6 means that the current value depends moderately on the previous one, resulting in smooth, persistent changes in the series. The process is stationary, so the values fluctuate around zero without any long-term trend. Occasionally, there are sharper changes due to the noise component, but overall, the series moves smoothly over time.

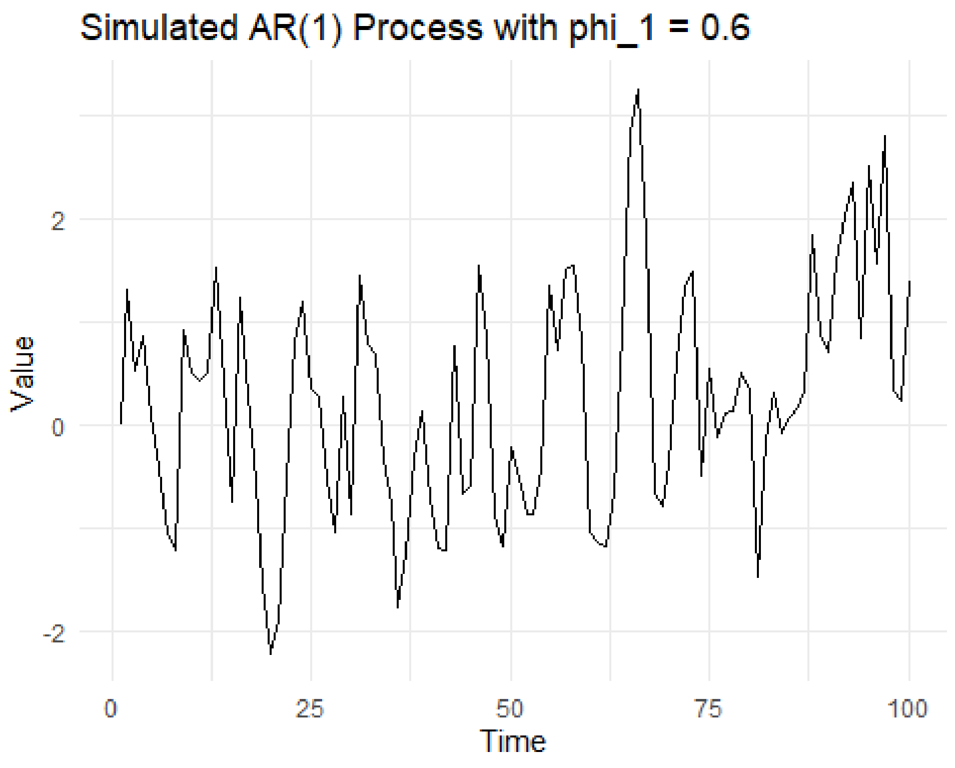
A higher ϕ1 would result in smoother, slower changes with more persistence. A lower ϕ1 would lead to more random, rapid fluctuations. A negative ϕ1 would cause the series to oscillate between highs and lows.

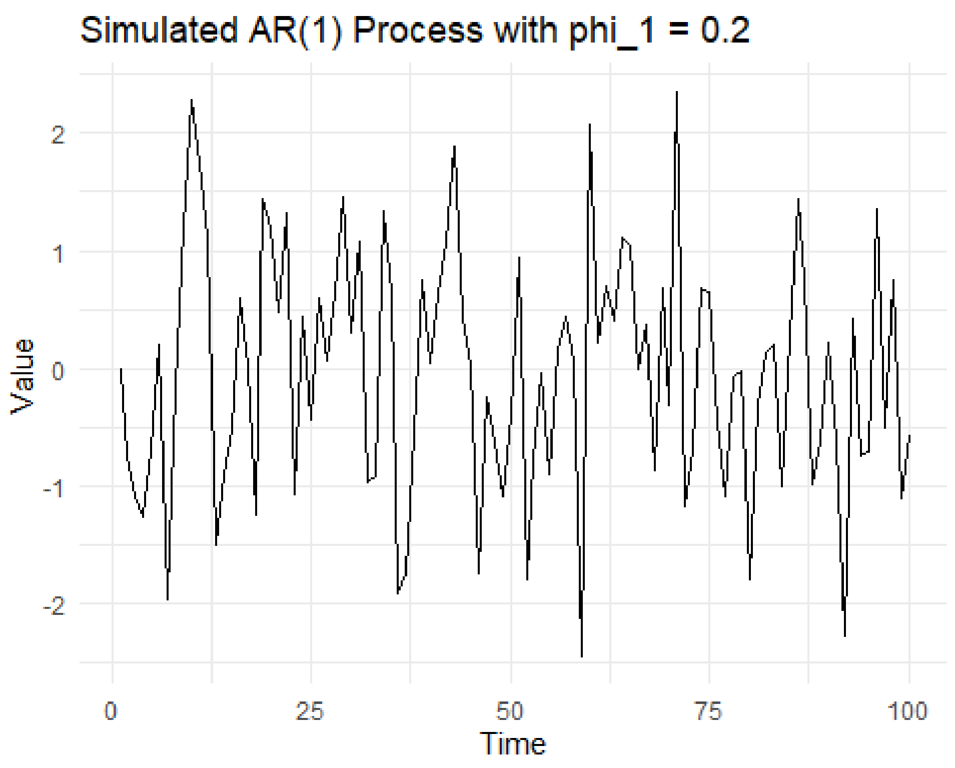
b. Produce a time plot for the series. How does the plot change as you change ϕ1?

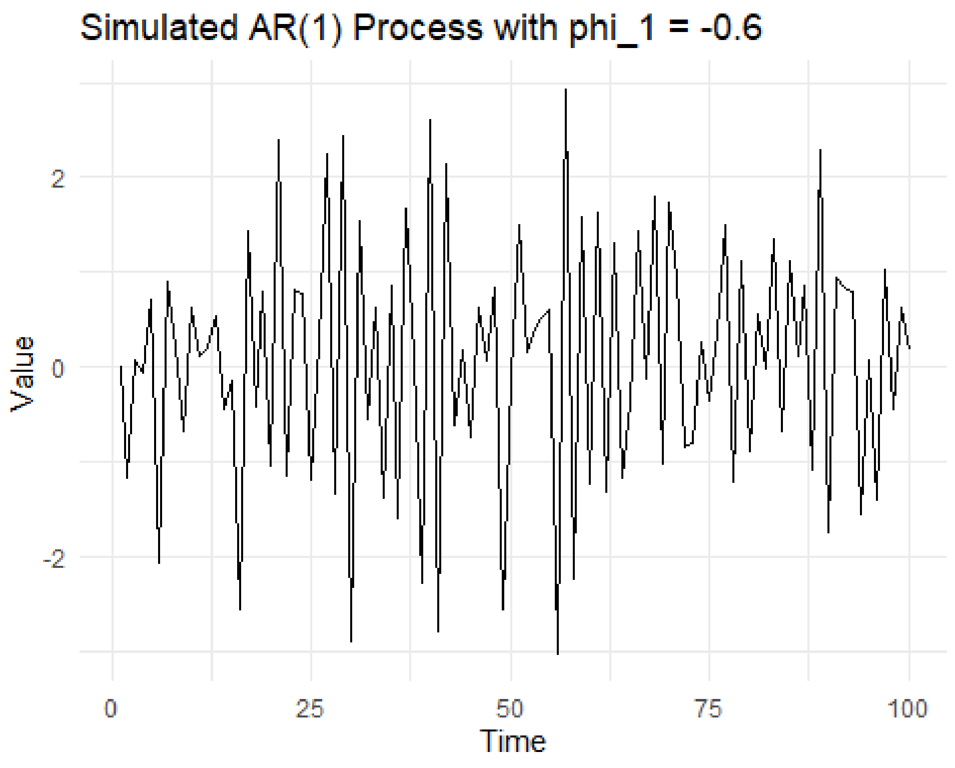
```
# Function to simulate AR(1) process with different phi_1
simulate_ar1 <- function(phi) {
  y <- numeric(100)
  e <- rnorm(100)
  y[1] <- 0  # Initial value
  for(i in 2:100) {
    y[i] <- phi * y[i-1] + e[i]
  }
  tsibble(idx = seq_len(100), y = y, index = idx)
}

# Plot AR(1) series for different phi_1 values
```

```r
phi_values <- c(0.9, 0.6, 0.2, -0.6)
for (phi in phi_values) {
  sim <- simulate_ar1(phi)
  p<- ggplot(sim, aes(x = idx, y = y)) +
    geom_line() +
    labs(title = paste("Simulated AR(1) Process with phi_1 =", phi), x = "Tim
e", y = "Value") +
    theme_minimal()

print(p)  # Print each plot in a loop
}
```


Simulated AR(1) Process with phi_1 = 0.9
Value
Time
2
0
-2
-4
0
25
50
75
100


Simulated AR(1) Process with phi_1 = 0.6
Value
Time
2
0
-2
0
25
50
75
100





In the plot shows a simulated AR(1) process with $\phi 1=0.9$ the series exhibits smooth transitions because each value is highly influenced by the previous one. There are long streaks of consecutive values above or below the mean, reflecting strong autocorrelation.

The changes in the series are slower and less random, as the influence of the noise is weaker compared to the past values. Overall, a high $\phi1$ like 0.9 results in a persistent and smooth time series, with fewer sharp fluctuations.

Comparison to Lower $\phi1$ if $\phi1$ were lower (e.g., 0.6 or 0.2), the series would fluctuate more frequently, with less dependence on the previous values, resulting in more randomness. If $\phi1$ were close to 0, the series would resemble a white noise process with minimal autocorrelation.

The plot here shows the expected behavior for a high $\phi1$, with smoother transitions and stronger persistence in the time series.

The Plot for $\phi1=0.6$, shows a moderate level of autocorrelation where values are influenced by the previous values, but randomness is more prominent than when $\phi1$ is closer to 1. The series demonstrates sharper fluctuations compared to the smoother, more persistent movements seen when $\phi1$ is larger (e.g., 0.9).

Comparison to $\phi1=0.9$ With $\phi1=0.6$, the series is less smooth and transitions more frequently between peaks and troughs than with $\phi1=0.9$. The autocorrelation is still strong enough to maintain some continuity between values, but the influence of past values fades faster than in the case of higher $\phi1$ values.

The plot for the AR(1) process with $\phi1=0.2$ shows each value is only slightly influenced by the previous one, leading to frequent fluctuations. The series appears more erratic, with less smoothness and more rapid changes between values. The process crosses the zero line more often, with sharp changes and less persistence compared to higher $\phi1$ values.

Comparison to Higher $\phi1$ Values, $\phi1=0.6$ or $\phi1=0.9$, this process has much lower smoothness, with more rapid and random changes between values. The series behaves more like a random walk, where each value is influenced only slightly by the previous value. This leads to faster transitions and more noise-dominated behavior.

The plot with $\phi1=-0.6$ shows the series alternates between high and low values, creating a zigzag pattern. Values change direction quickly, with rapid movements between peaks and troughs. The series doesn’t stay consistently above or below the mean, unlike with positive $\phi1$.

Comparison to Positive $\phi1$ creates more smooth and persistent trends, where values move in the same direction over multiple time steps. Negative $\phi1$ results in more volatility and sharp changes, with values frequently reversing direction in an oscillating pattern.

The plot changes significantly as you adjust the value of$\phi1$ in an AR(1) model. Here’s a summary of how the plot evolves with different values of $\phi1$:

1. Higher$\phi1$(e.g.,$\phi1=0.9$):

Smooth transitions: The series becomes smoother with slower, more gradual changes between values. High persistence: Each value is strongly influenced by the previous value, so the series tends to remain in the same direction (upward or downward) for extended periods. Longer streaks: There are long streaks of values above or below the mean (zero), and the series takes longer to revert back to the mean.

2. Moderate $\phi 1$(e.g., $\phi 1$=0.6):

Moderate smoothness: The series exhibits moderate autocorrelation, meaning it is influenced by previous values, but not as strongly as with higher $\phi 1$. More frequent fluctuations: The series changes direction more frequently, and reversions to the mean happen more often compared to higher $\phi 1$. Balance between randomness and persistence: There's a balance between smoothness and randomness, with both noticeable influences on the series.

3. Lower $\phi 1$(e.g., $\phi 1$=0.2):

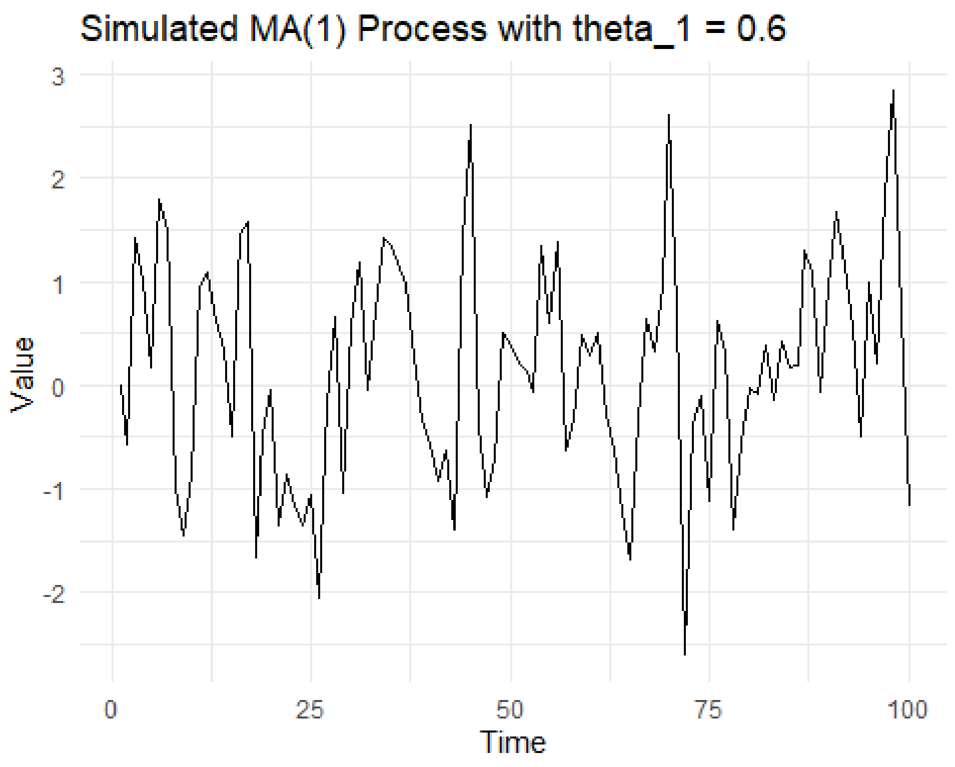
More random fluctuations: The series looks more like white noise, with frequent, abrupt changes between values. Less persistence: Each value is less dependent on the previous one, so the series is less smooth and moves more erratically. Faster reversion to the mean: The series reverts to the mean (zero) quickly, with fewer streaks of values above or below the mean.

4. Negative $\phi 1$(e.g., $\phi 1$=−0.6): Oscillating behavior: The series alternates between high and low values more sharply, creating a zigzag pattern. Negative autocorrelation: A negative $\phi 1$ causes each value to be inversely related to the previous one, so if one value is high, the next is likely to be low, and vice versa.

General Effects of Changing $\phi 1$: Higher $\phi 1$ values create smoother and more persistent series with fewer sharp changes. Lower $\phi 1$ values result in more random and erratic fluctuations, with quicker reversion to the mean. Negative $\phi 1$ introduces alternating patterns where high values are followed by low ones, and vice versa.

c. Write your own code to generate data from an MA(1) model with θ1=0.6 and σ2=1.

```
# Set seed for reproducibility
set.seed(123)

# Initialize variables
n <- 100                  # Number of time points
e <- rnorm(n, mean = 0, sd = 1)  # Generate random errors (e_t) with variance
= 1
theta1 <- 0.6             # Set theta_1 value

# Initialize y, the time series for MA(1) process
y <- numeric(n)

# Simulate the MA(1) process
for(i in 2:n) {
  y[i] <- e[i] + theta1 * e[i-1]  # MA(1) equation: y_t = e_t + theta1 * e_{t
-1}
}

# Convert to a tsibble for easy plotting
sim <- tsibble(idx = seq_len(n), y = y, index = idx)

# Plot the simulated MA(1) process
ggplot(sim, aes(x = idx, y = y)) +
  geom_line() +
  labs(title = "Simulated MA(1) Process with theta_1 = 0.6", x = "Time", y =
"Value") +
  theme_minimal()
```

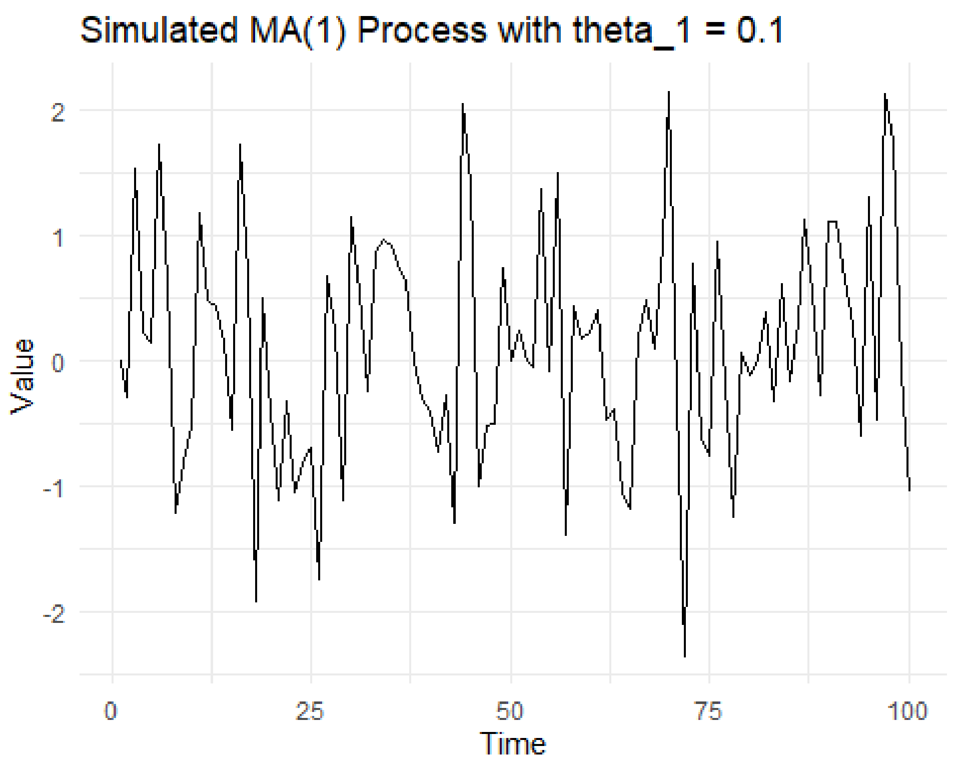


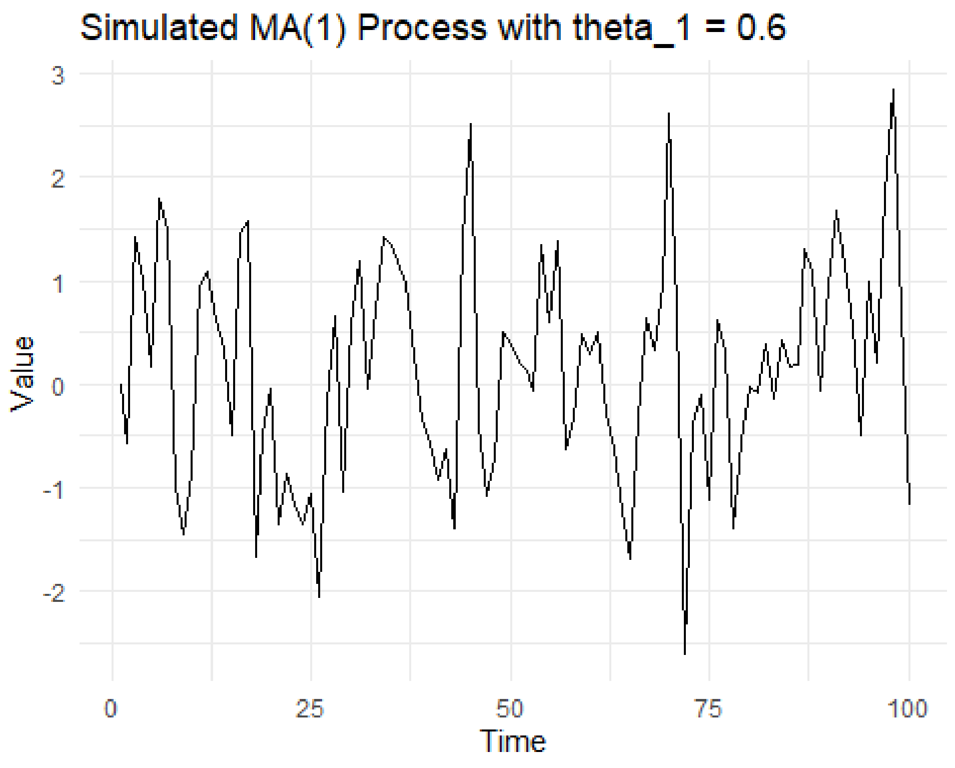
The MA(1) model with $\theta 1$=0.6 creates a series that exhibits short-term correlation between consecutive values, but quickly reverts to randomness, showing less persistence than an AR(1) model. The plot is characterized by random fluctuations and occasional short-term influences from previous shocks, typical of an MA(1) process.

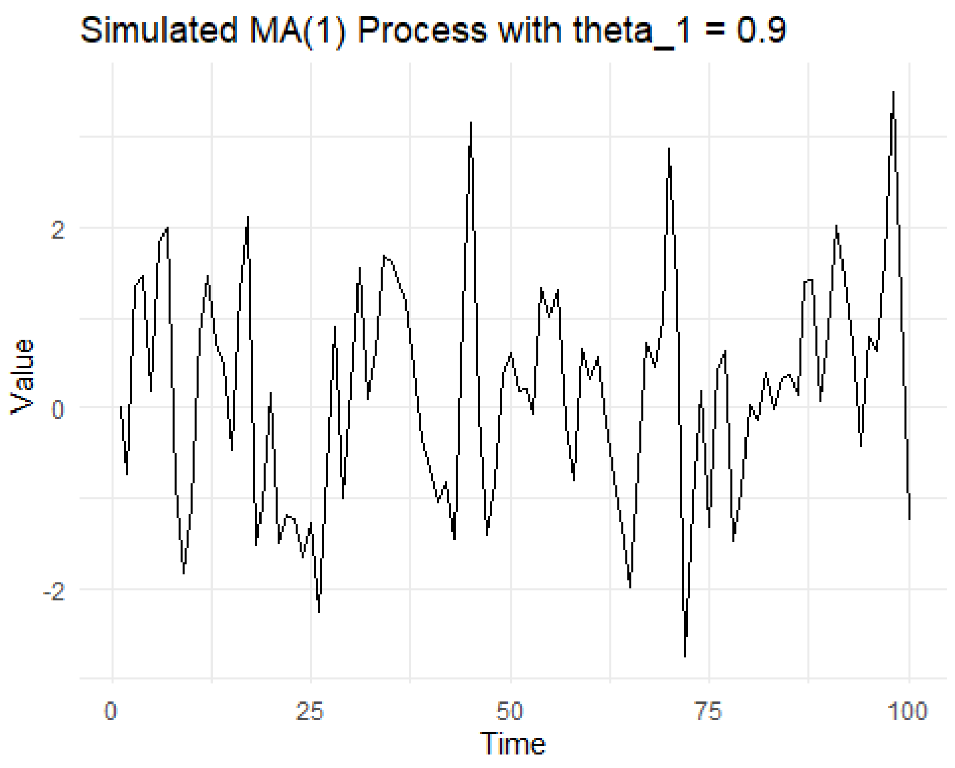
d. Produce a time plot for the series. How does the plot change as you change θ1?

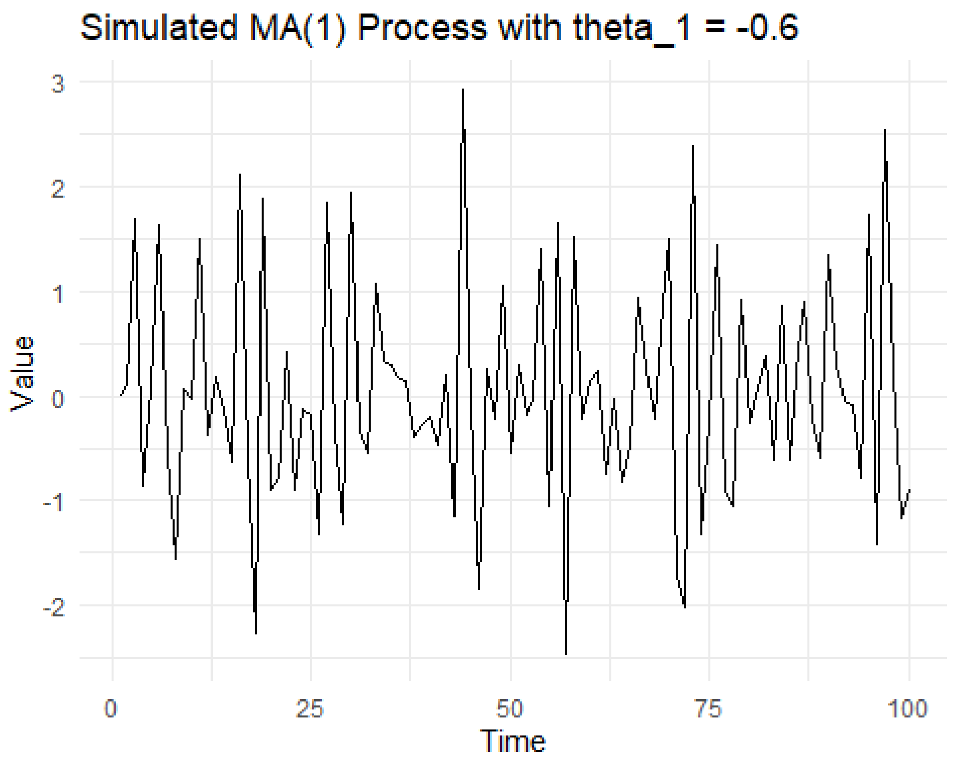
```
# Function to simulate MA(1) process for a given theta_1
simulate_ma1 <- function(theta) {
  set.seed(123)  # Set seed for reproducibility
  n <- 100
```

```r
  e <- rnorm(n, mean = 0, sd = 1)  # Random errors
  y <- numeric(n)

  # Simulate MA(1) process
  for(i in 2:n) {
    y[i] <- e[i] + theta * e[i-1]
  }

  tsibble(idx = seq_len(n), y = y, index = idx)
}

# Plot MA(1) processes for different theta_1 values
theta_values <- c(0.1, 0.6, 0.9, -0.6)

for (theta in theta_values) {
  sim <- simulate_ma1(theta)
  p <- ggplot(sim, aes(x = idx, y = y)) +
    geom_line() +
    labs(title = paste("Simulated MA(1) Process with theta_1 =", theta), x =
"Time", y = "Value") +
    theme_minimal()

  print(p)  # Explicitly call print() when inside a loop or script
}
```


Simulated MA(1) Process with theta_1 = 0.1
Value
Time
2
1
0
-1
-2
0
25
50
75
100


Simulated MA(1) Process with theta_1 = 0.6
Value
Time
3
2
1
0
-1
-2
0
25
50
75
100





The series changes significantly as you adjust the value of $\theta 1$ in an MA(1) process.

1. For $\theta 1$=0.1:

Minimal influence from the previous error e_{t-1}.

The series appears almost like white noise, with frequent, random fluctuations and little correlation between consecutive points.

The process behaves nearly like pure random noise, as the dependence on the past shock is very weak.

2. For $\theta_1=0.6$:

Moderate smoothing occurs, where the current value is influenced by both the current and previous errors.

Short-term correlations are visible, with smoother transitions between points, although the randomness is still present.

The process starts showing more dependence on the previous values, resulting in a smoother series compared to $\theta_1=0.1$.

3. For $\theta_1=0.9$:

The series becomes even smoother, with stronger short-term dependence on the previous error.

Changes between consecutive values are more gradual, and the series shows fewer sharp jumps.

The influence of the previous error is very strong, making the series appear more correlated, with longer sequences of similar values.

4. For $\theta_1=-0.6$:

The series exhibits oscillating behavior, where the influence of the previous error is negative, causing values to alternate more sharply between highs and lows.

This creates a zigzag pattern, where values tend to alternate between positive and negative more frequently.

The negative autocorrelation leads to sharper reversals, making the series switch direction frequently.

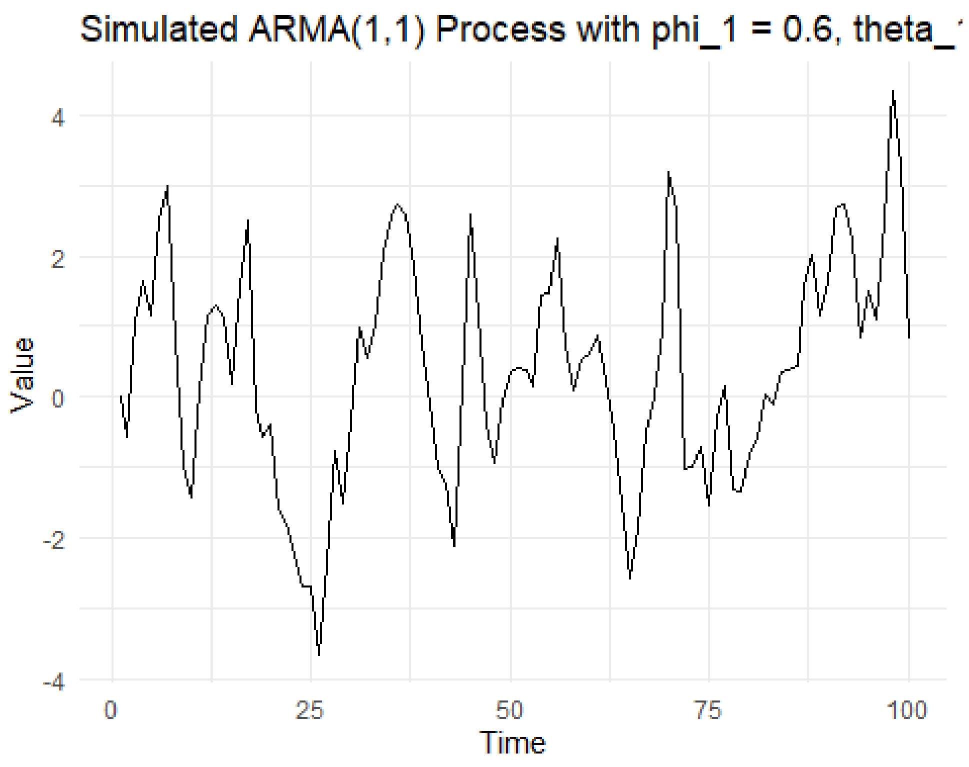
Smaller θ_1 values (like $\theta_1=0.1$) result in more random and erratic behavior, with little influence from the past.

Larger positive θ_1 values (like $\theta_1=0.9$) create smoother, more correlated series, where changes are gradual and more persistent.

Negative θ_1 (like $\theta_1=-0.6$) introduces sharp oscillations, making the series switch direction more frequently.

e. Generate data from an ARMA(1,1) model with ϕ1=0.6, and σ2=1.

```r
# Set seed for reproducibility
set.seed(123)

# Initialize variables
n <- 100  # Number of time points
phi1 <- 0.6  # AR(1) coefficient
theta1 <- 0.6  # MA(1) coefficient
e <- rnorm(n, mean = 0, sd = 1)  # Random errors (e_t) with variance = 1
y <- numeric(n)  # Initialize series

# Simulate ARMA(1,1) process
for (i in 2:n) {
  y[i] <- phi1 * y[i-1] + e[i] + theta1 * e[i-1]  # ARMA(1,1) equation
}

# Convert to tsibble for easy plotting
sim_arma <- tsibble(idx = seq_len(n), y = y, index = idx)

# Plot the simulated ARMA(1,1) process
ggplot(sim_arma, aes(x = idx, y = y)) +
  geom_line() +
  labs(title = "Simulated ARMA(1,1) Process with phi_1 = 0.6, theta_1 = 0.6",
x = "Time", y = "Value") +
  theme_minimal()
```

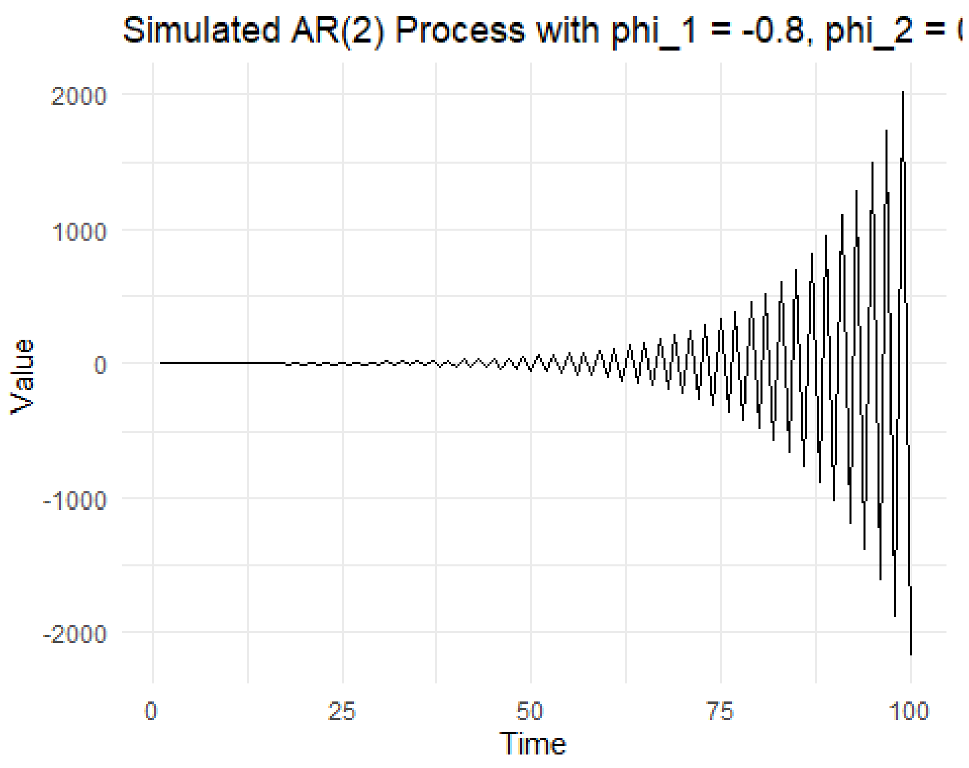


The AR component introduces persistence and smoothness into the series, while the MA component contributes short-term fluctuations and adds randomness from the previous errors.

The plot will show a mix of smoothing (from the AR(1) part) and short-term volatility (from the MA(1) part).

f. Generate data from an AR(2) model with ϕ1=−0.8, ϕ2=0.3 and σ2=1. (Note that these parameters will give a non-stationary series.)

```r
# Set seed for reproducibility
set.seed(123)

# Initialize variables
n <- 100  # Number of time points
phi1 <- -0.8  # AR(1) coefficient
phi2 <- 0.3  # AR(2) coefficient
e <- rnorm(n, mean = 0, sd = 1)  # Random errors (e_t) with variance = 1
y <- numeric(n)  # Initialize series

# Simulate AR(2) process
for (i in 3:n) {
  y[i] <- phi1 * y[i-1] + phi2 * y[i-2] + e[i]  # AR(2) equation
}

# Convert to tsibble for easy plotting
sim_ar2 <- tsibble(idx = seq_len(n), y = y, index = idx)

# Plot the simulated AR(2) process
ggplot(sim_ar2, aes(x = idx, y = y)) +
  geom_line() +
  labs(title = "Simulated AR(2) Process with phi_1 = -0.8, phi_2 = 0.3", x =
"Time", y = "Value") +
  theme_minimal()
```



Given the specific parameters (ϕ1=−0.8 and ϕ2=0.3), the series will likely exhibit oscillations due to the interaction of the AR(1) and AR(2) components.

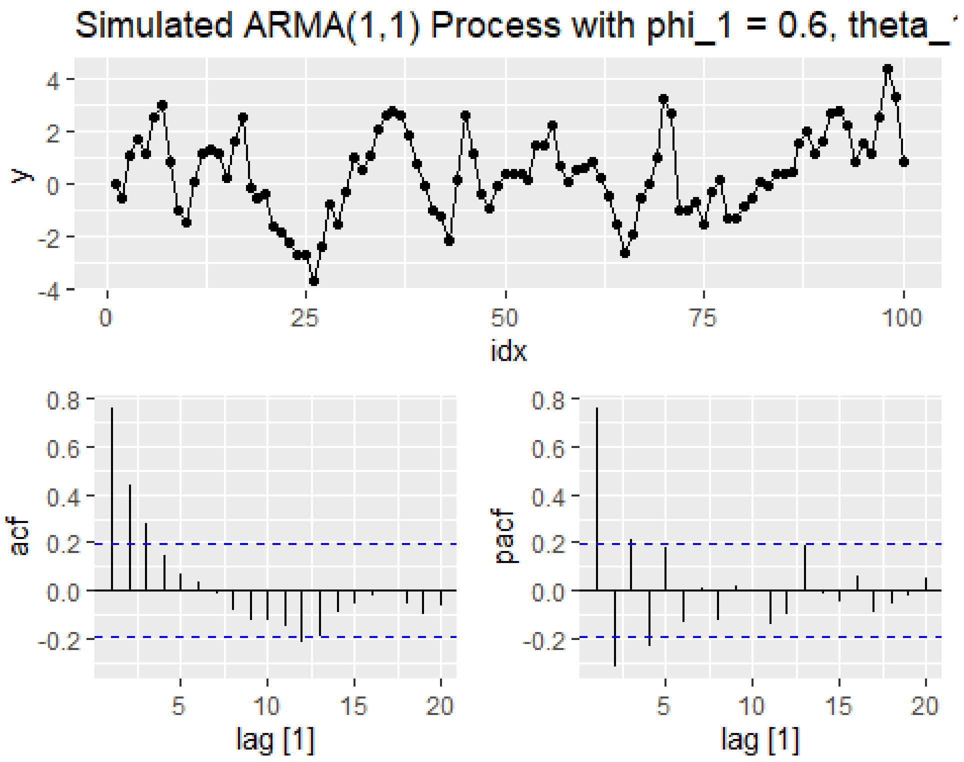
The series may show non-stationary behavior, where the mean and variance change over time, which is common for certain parameter combinations like this one.

The time plot reveals the non-stationary nature of the series with potentially increasing fluctuations or oscillations.

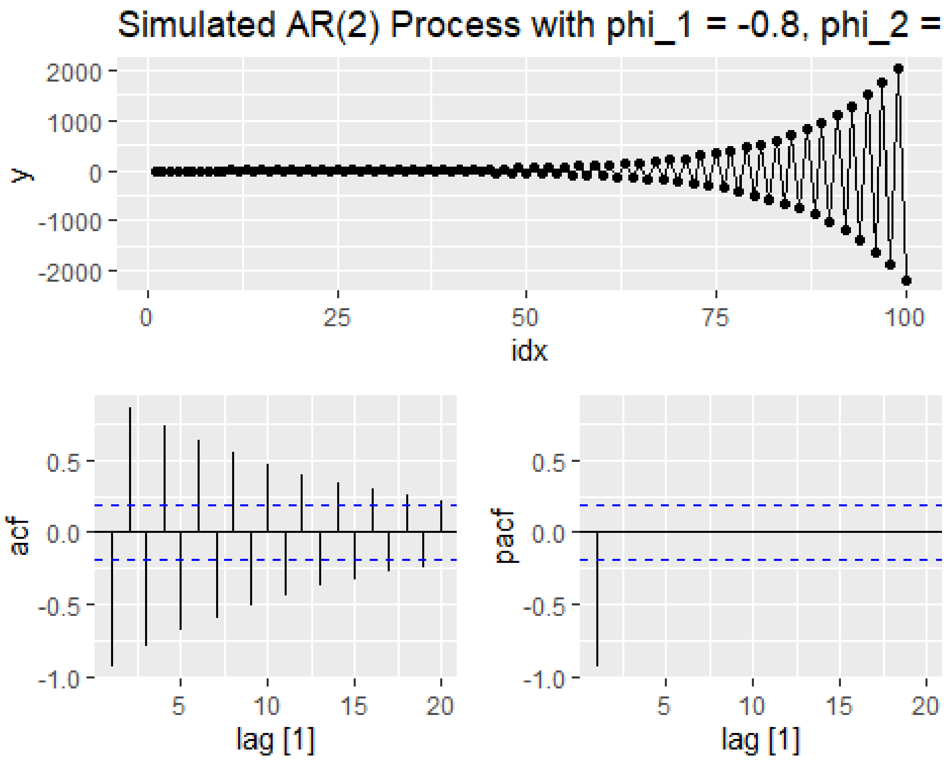
** g. Graph the latter two series and compare them.**

```
# Simulate ARMA(1,1) process with phi_1 = 0.6, theta_1 = 0.6
simulate_arma11 <- function(n) {
  set.seed(123)
  phi1 <- 0.6
  theta1 <- 0.6
  e <- rnorm(n, mean = 0, sd = 1)
  y <- numeric(n)

  for (i in 2:n) {
    y[i] <- phi1 * y[i-1] + e[i] + theta1 * e[i-1]
  }

  tsibble(idx = seq_len(n), y = y, index = idx)
}
```

```r
# Simulate AR(2) process with phi_1 = -0.8, phi_2 = 0.3
simulate_ar2 <- function(n) {
  set.seed(123)
  phi1 <- -0.8
  phi2 <- 0.3
  e <- rnorm(n, mean = 0, sd = 1)
  y <- numeric(n)

  for (i in 3:n) {
    y[i] <- phi1 * y[i-1] + phi2 * y[i-2] + e[i]
  }

  tsibble(idx = seq_len(n), y = y, index = idx)
}

# Number of time points
n <- 100

# Simulate ARMA(1,1) and AR(2) series
arma11_series <- simulate_arma11(n)
ar2_series <- simulate_ar2(n)

# Plot ARMA(1,1) series with time plot, ACF, and PACF using gg_tsdisplay
arma11_series %>%
  gg_tsdisplay(y, plot_type = 'partial') +
  labs(title = "Simulated ARMA(1,1) Process with phi_1 = 0.6, theta_1 = 0.6")
```



```r
# Plot AR(2) series with time plot, ACF, and PACF using gg_tsdisplay
ar2_series %>%
  gg_tsdisplay(y, plot_type = 'partial') +
  labs(title = "Simulated AR(2) Process with phi_1 = -0.8, phi_2 = 0.3")
```



ARMA(1,1) Model ($\phi_1 = 0.6$, $\theta_1 = 0.6$):

Time series plot: The ARMA(1,1) model exhibits a relatively stable and random-looking series. There's moderate autocorrelation present, but the values fluctuate around zero with occasional trends.

ACF plot: The autocorrelation function (ACF) shows a gradual decline after a few significant lags. This is typical for an ARMA(1,1) model, where both autoregressive and moving average terms influence the series.

PACF plot: The PACF quickly cuts off after the first lag, confirming the partial autocorrelation effect introduced by the AR term ($\phi_1 = 0.6$).

AR(2) Model ($\phi_1 = -0.8$, $\phi_2 = 0.3$):

Time series plot: The AR(2) series shows much more volatility as it oscillates and amplifies over time, eventually increasing in magnitude. This non-stationary behavior is due to the choice of the autoregressive parameters.

ACF plot: The ACF shows clear signs of non-stationarity, with slow decay across multiple lags. The oscillatory nature of the ACF is due to the combined influence of ϕ_1 and ϕ_2 in the

AR(2) model . PACF plot: The PACF shows significant values at lags 1 and 2, reflecting the influence of the AR(2) model structure. After lag 2, the partial correlations become much smaller.

The ARMA(1,1) model appears more stable and stationary, while the AR(2) model shows clear signs of non-stationarity, as evidenced by its oscillatory pattern and increasing amplitude over time.

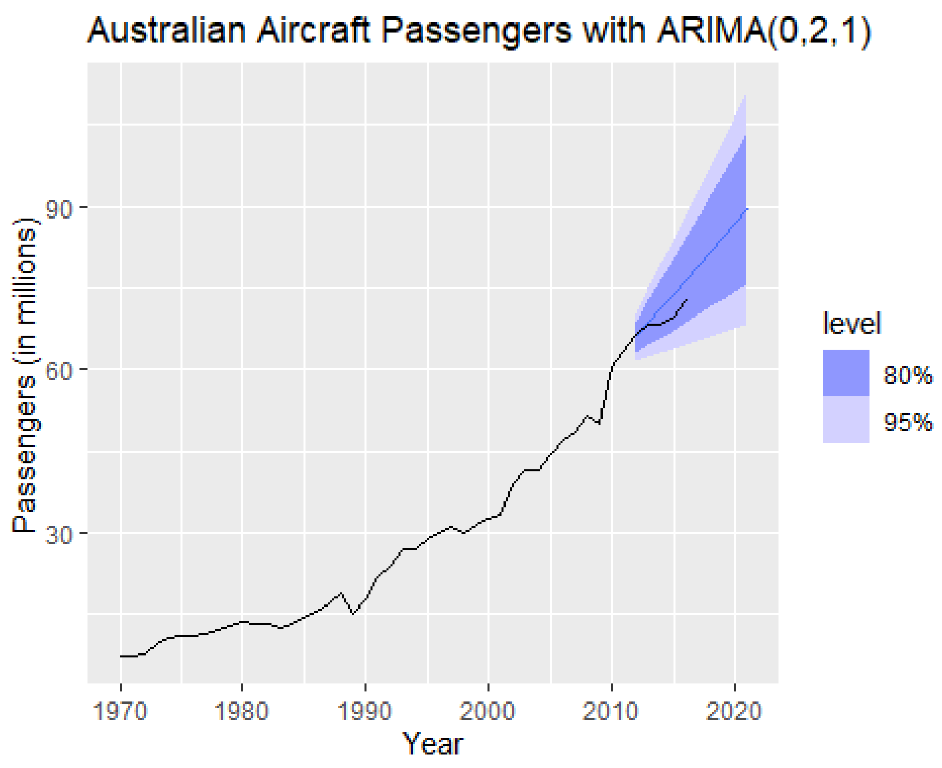
The ACF and PACF for the ARMA(1,1) model display patterns typical for a stationary series, while the AR(2) model shows a slow decay and oscillation in the ACF, indicating non-stationarity.

The non-stationary behavior in the AR(2) series is due to the choice of parameters, which does not satisfy stationarity conditions. Meanwhile, the ARMA(1,1) model remains stationary, with more controlled and predictable behavior.

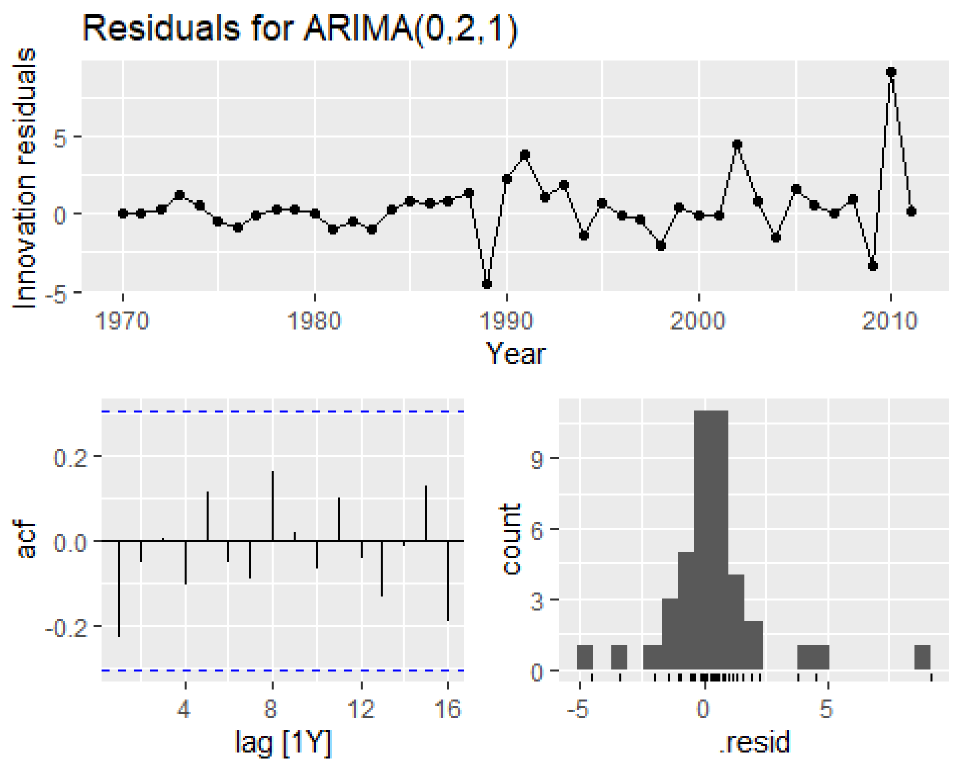
7. Consider aus_airpassengers, the total number of passengers (in millions) from Australian air carriers for the period 1970-2011.

a. Use ARIMA() to find an appropriate ARIMA model. What model was selected. Check that the residuals look like white noise. Plot forecasts for the next 10 periods.

```
fit1 <- aus_airpassengers %>%
  filter(Year < 2012) %>%
  model(ARIMA(Passengers))

report(fit1)

## Series: Passengers
## Model: ARIMA(0,2,1)
##
## Coefficients:
##           ma1
##       -0.8756
## s.e.   0.0722
##
## sigma^2 estimated as 4.671:  log likelihood=-87.8
## AIC=179.61   AICc=179.93   BIC=182.99

fit1 %>%
  forecast(h=10) %>%
  autoplot(aus_airpassengers) +
  labs(title = "Australian Aircraft Passengers with ARIMA(0,2,1)", y = "Passe
ngers (in millions)")
```



```r
fit1 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(0,2,1)")
```



The selected ARIMA model is ARIMA(0,2,1). This means: No autoregressive terms (AR = 0). The data has been differenced twice to make it stationary (D = 2). There is one moving average term (MA = 1).

The coefficients:

MA1 = -0.8756, with a standard error of 0.0722. Sigma^2 (the variance of the residuals) is estimated as 4.671.

From the residuals plot:

The time plot of the residuals shows randomness, with no apparent trends.

The ACF plot shows that none of the lags are significantly different from zero (outside the confidence bands), indicating no significant autocorrelation.

The histogram of the residuals looks roughly normal, with most values centered around zero. These plots suggest that the residuals resemble white noise, indicating that the ARIMA model fits well.

The forecast plot shows the forecasted values for the next 10 periods, along with 80% and 95% prediction intervals. The confidence intervals widen as we forecast further into the future, which is typical for time series forecasting.

b. Write the model in terms of the backshift operator.

For ARIMA(0,2,1):

No autoregressive terms, so no ϕterms. The series is differenced twice, which means $(1-B)2(1-B)$. There is one moving average term $\theta 1=-0.8756$.

$(1-B)2yt=(1+\theta B)\epsilon t$ $(1-B)2yt=(1-0.8756B)\epsilon t$

Expanding the Differencing Term $(1-B)2$:

$(1-B)2=1-2B+B2$

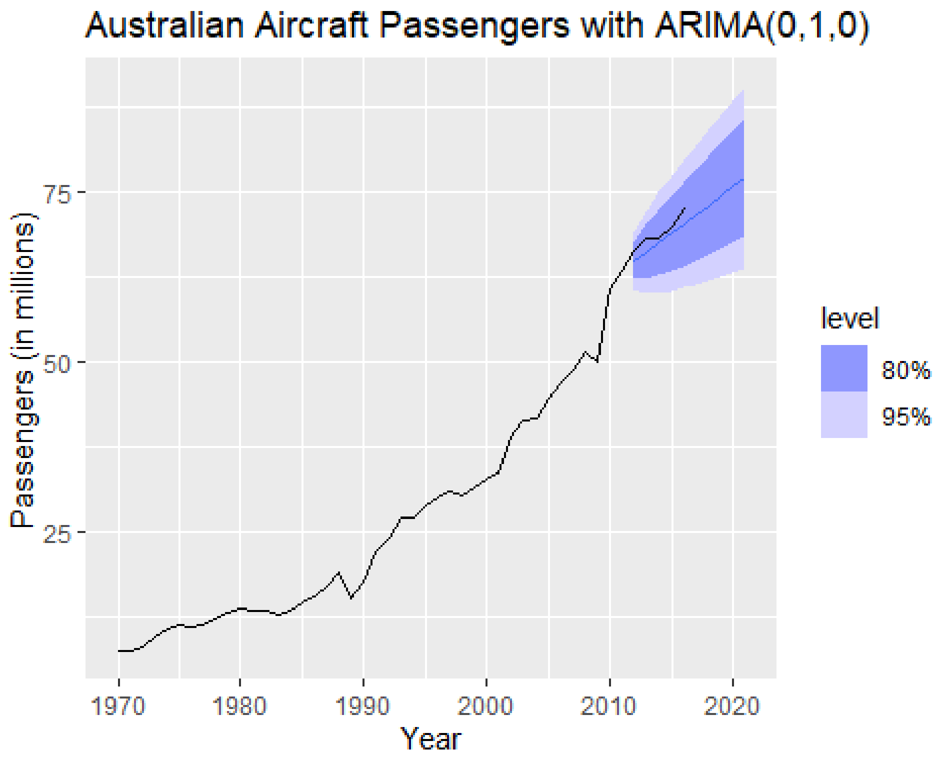
The full ARIMA(0,2,1) model becomes:

$(1-2B+B2)yt=(1-0.8756B)\epsilon t$

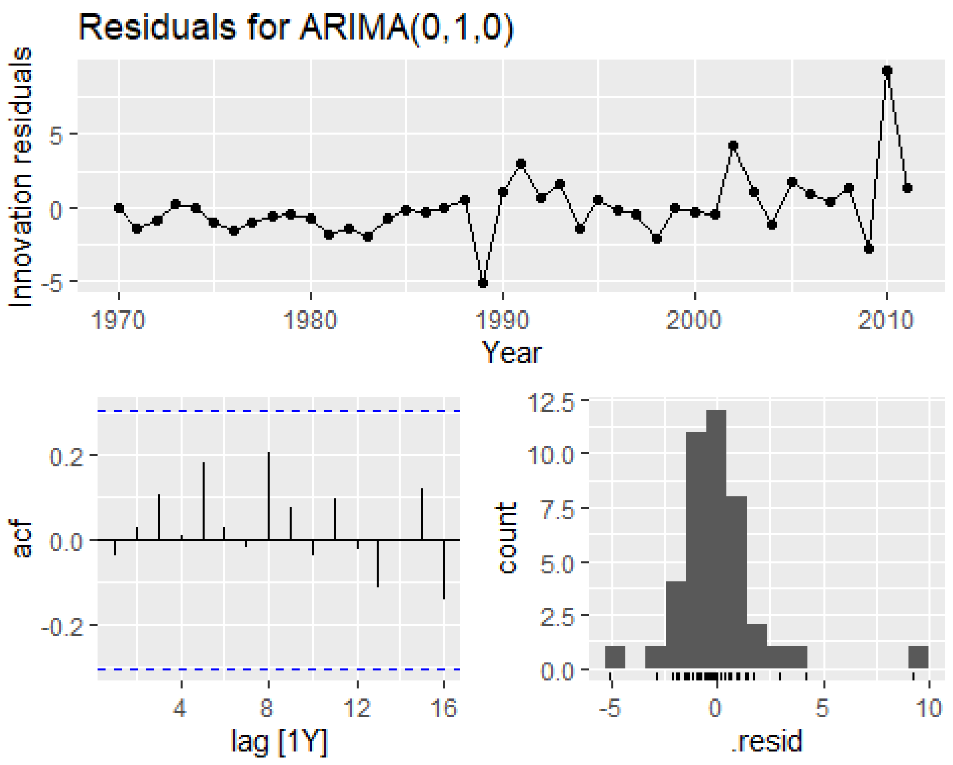
##**b. Plot forecasts from an ARIMA(0,1,0) model with drift and compare these to part a.**

```
fit2 <-aus_airpassengers %>%
  filter(Year < 2012) %>%
  model(ARIMA(Passengers ~ pdq(0,1,0)))

fit2 %>%
  forecast(h=10) %>%
  autoplot(aus_airpassengers) +
```

```r
  labs(title = "Australian Aircraft Passengers with ARIMA(0,1,0)", y = "Passe
ngers (in millions)")
```



```r
fit2 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(0,1,0)")
```



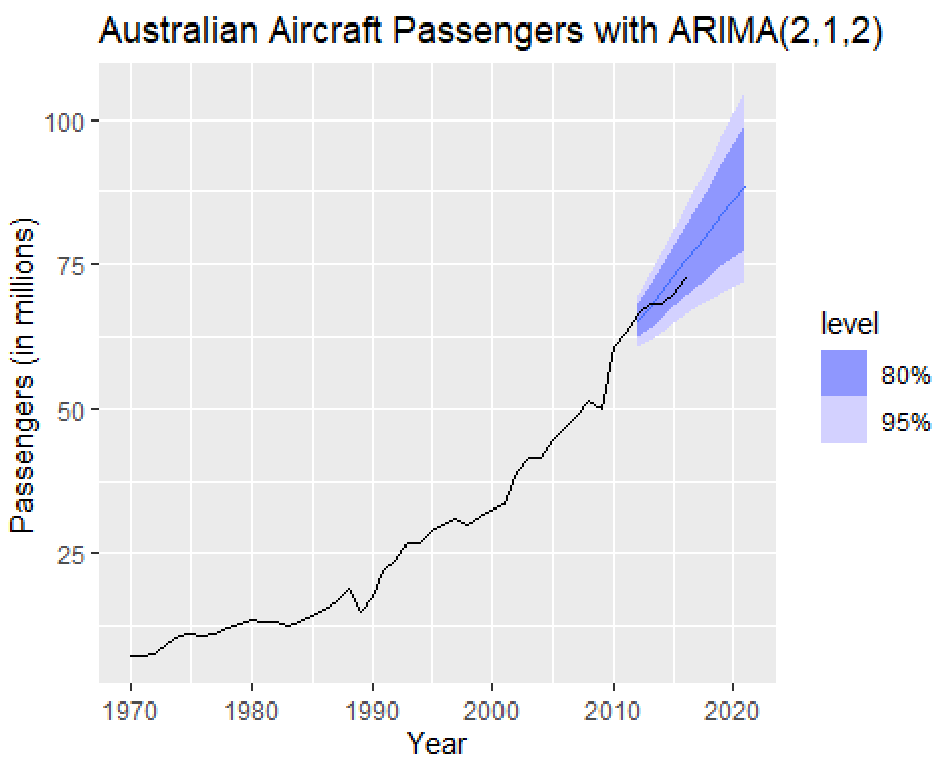
In the ARIMA(0,1,0) model with drift, the forecast plot shows a steady upward trend, with predictions widening over time, similar to the ARIMA(0,2,1) model. However, the ARIMA(0,1,0) model suggests a simpler structure due to only differencing once with a drift term and no moving average component.

When comparing the residuals between the ARIMA(0,1,0) and the ARIMA(0,2,1), the residuals for ARIMA(0,1,0) still show some autocorrelation in the ACF plot, indicating the model may not be capturing all the autocorrelation present. This may suggest that the ARIMA(0,2,1) model was better suited as it modeled more of the structure in the data.

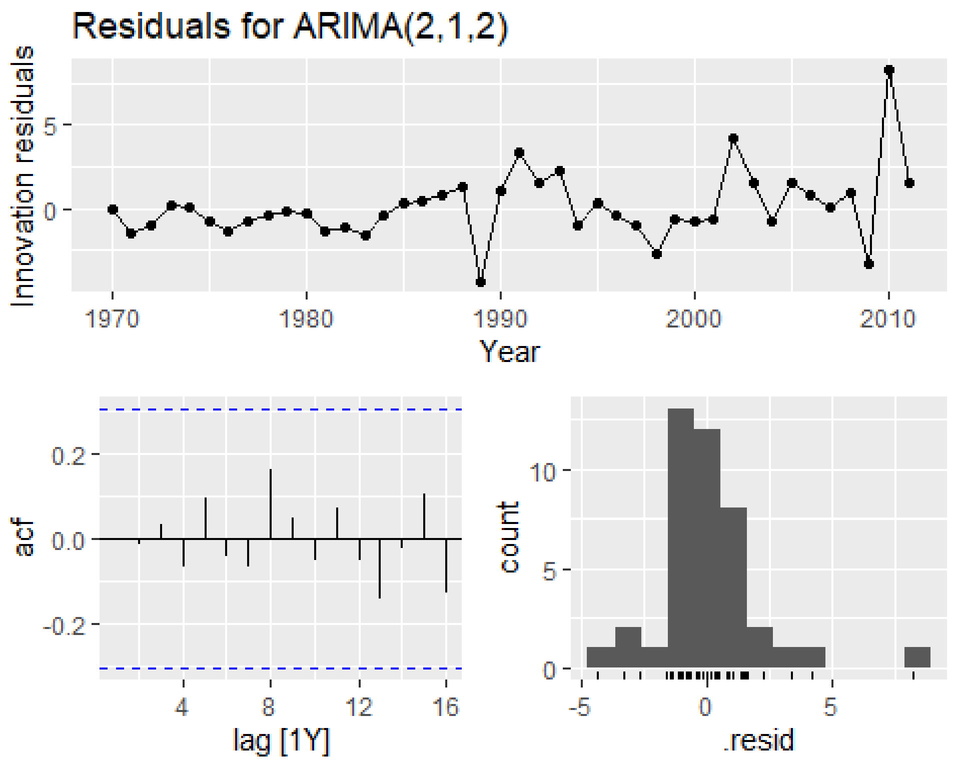
While the ARIMA(0,1,0) model captures the trend, it might not handle the underlying noise as well as ARIMA(0,2,1), based on residual diagnostics and the ACF.

##** d. Plot forecasts from an ARIMA(2,1,2) model with drift and compare these to parts a and c. Remove the constant and see what happens.**

```
fit3 <-aus_airpassengers %>%
  filter(Year < 2012) %>%
  model(ARIMA(Passengers ~ pdq(2,1,2)))

fit3 %>%
  forecast(h=10) %>%
  autoplot(aus_airpassengers) +
  labs(title = "Australian Aircraft Passengers with ARIMA(2,1,2)", y = "Passe
ngers (in millions)")
```



```r
fit3 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(2,1,2)")
```



ARIMA(0,2,1):

In part (a), this model was fitted with a second difference and one MA term. The forecast for this model showed a steady continuation of the existing trend with a slight increase in uncertainty over the forecast horizon.

The residuals looked approximately like white noise, as shown by the ACF and PACF plots.

ARIMA(0,1,0) with Drift:

In part (c), this model captured the increasing trend with a first difference and drift, suggesting that the series has a constant linear growth.

Residuals for this model also looked relatively uncorrelated, but there may be some patterns suggesting the model may not fully capture the data.

ARIMA(2,1,2):

This model introduces more complexity with two AR terms and two MA terms. It captures both autoregressive behavior and moving average effects.

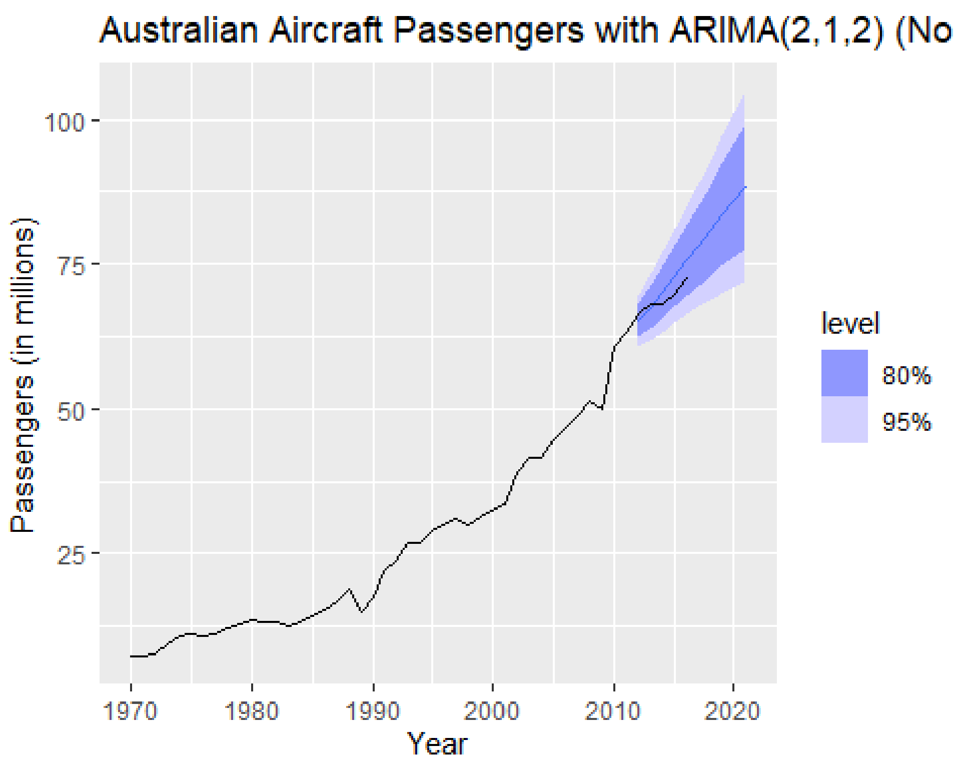
The forecast suggests a continuation of the trend but with slightly wider confidence intervals than ARIMA(0,2,1), indicating more uncertainty.

The residuals plot shows a good fit, but you would want to check the ACF and PACF to ensure that they indicate white noise (i.e., no significant autocorrelation).

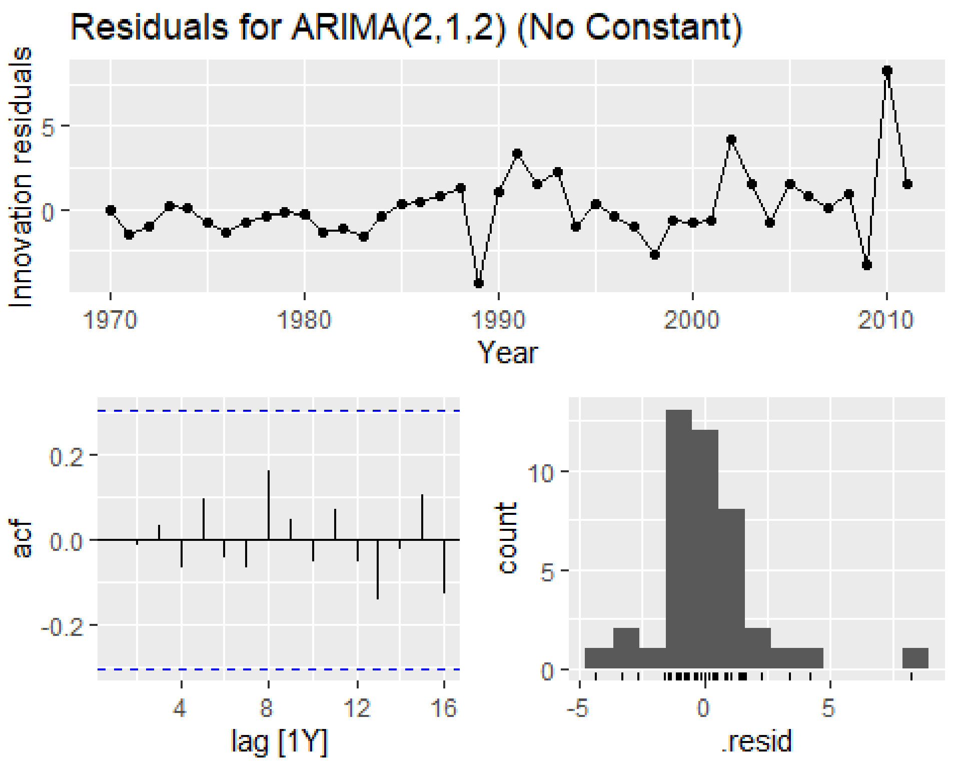
The ARIMA(2,1,2) model may capture more nuances in the data, offering a more refined fit, while both ARIMA(0,2,1) and ARIMA(0,1,0) give slightly simpler trend forecasts.

While ARIMA(2,1,2) may be more complex and potentially provide a better fit, it’s essential to balance model complexity with interpretability and check whether the residuals confirm that the model is well-fitted.

```r
# Fit the ARIMA(2,1,2) model without drift (constant term)
fit_no_constant <- aus_airpassengers %>%
  filter(Year < 2012) %>%
  model(ARIMA(Passengers ~ pdq(2,1,2) + PDQ(0,0,0)))

# Forecast the next 10 periods without the constant
fit_no_constant %>%
  forecast(h = 10) %>%
  autoplot(aus_airpassengers) +
  labs(title = "Australian Aircraft Passengers with ARIMA(2,1,2) (No Constant
)",
       y = "Passengers (in millions)")
```



```r
# Plot the residuals for the model without the constant
fit_no_constant %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(2,1,2) (No Constant)")
```

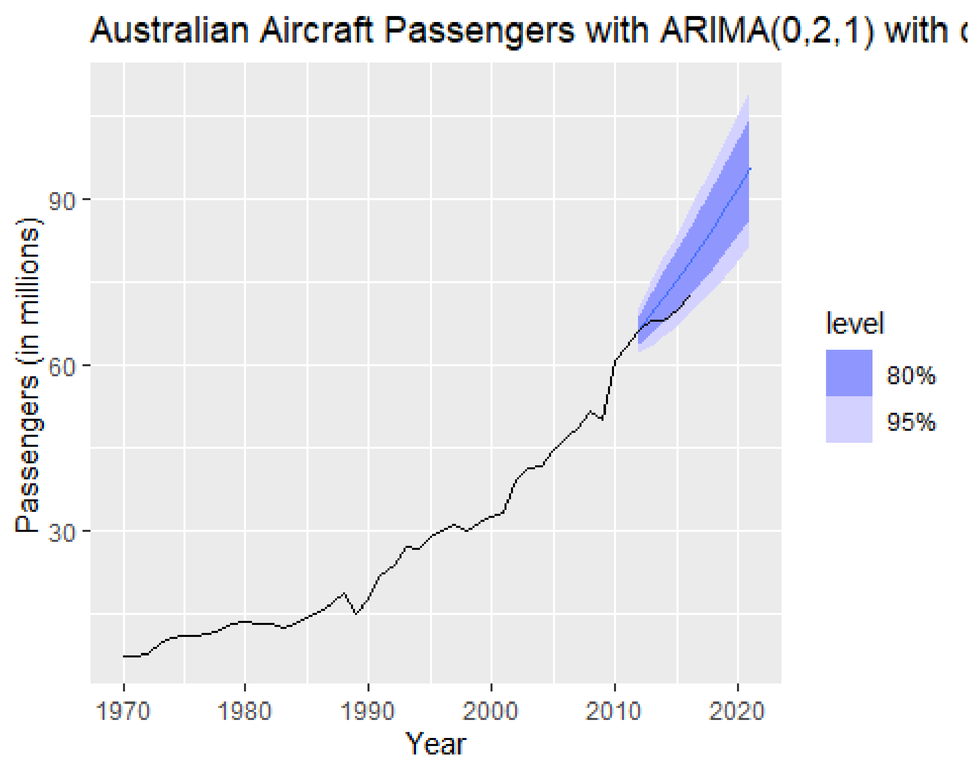


After removing the constant from the ARIMA(2,1,2) model, the forecast still captures the general upward trend, but the uncertainty bands (the blue shaded region) appear slightly wider compared to the model with the constant. This suggests that the removal of the constant introduces more uncertainty in the forecasts.

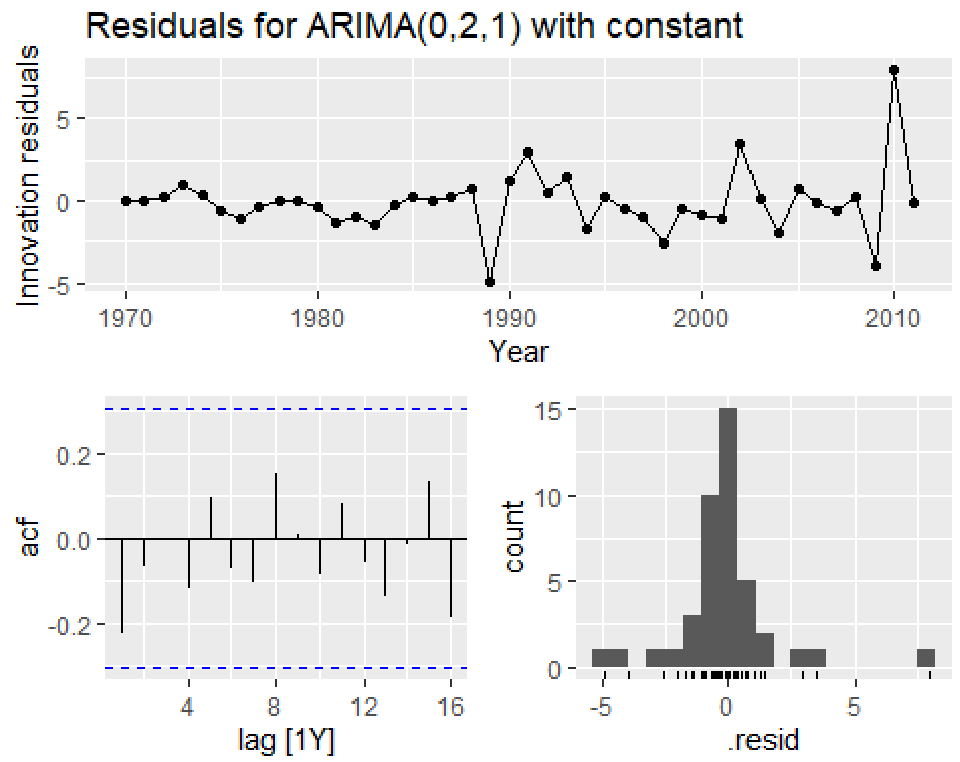
In the residuals plot, there is no significant difference between this and the model with the constant— the residuals still seem to behave like white noise, suggesting that the model fits reasonably well even without the constant term. However, the lack of the constant slightly alters the prediction dynamics and reduces the ability to explicitly account for a fixed trend component in the data.

##e. Plot forecasts from an ARIMA(0,2,1) model with a constant.

```
fit5 <-aus_airpassengers %>%
  filter(Year < 2012) %>%
  model(ARIMA(Passengers ~ 1 + pdq(0,2,1)))

fit5 %>%
  forecast(h=10) %>%
  autoplot(aus_airpassengers) +
  labs(title = "Australian Aircraft Passengers with ARIMA(0,2,1) with constan
t", y = "Passengers (in millions)")
```



```
fit5 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(0,2,1) with constant")
```



Model Fit and Forecast: The ARIMA(0,2,1) model with a constant induced a quadratic or higher-order trend in the forecasts. This is because adding a constant to an ARIMA model with two differences (d=2) leads to an upward trend in the forecast, which is noticeable in your plot. The model fits the data but shows an exaggerated upward trend due to the constant.

Residuals: The residuals do not appear to be white noise, suggesting that the model may not fully capture the dynamics of the data. There is some pattern in the residuals, especially at the end, indicating potential issues with the fit. The histogram also shows a slight skew in the residuals distribution, and the ACF plot of residuals shows non-zero autocorrelations at some lags, suggesting that the model may be underfitting.

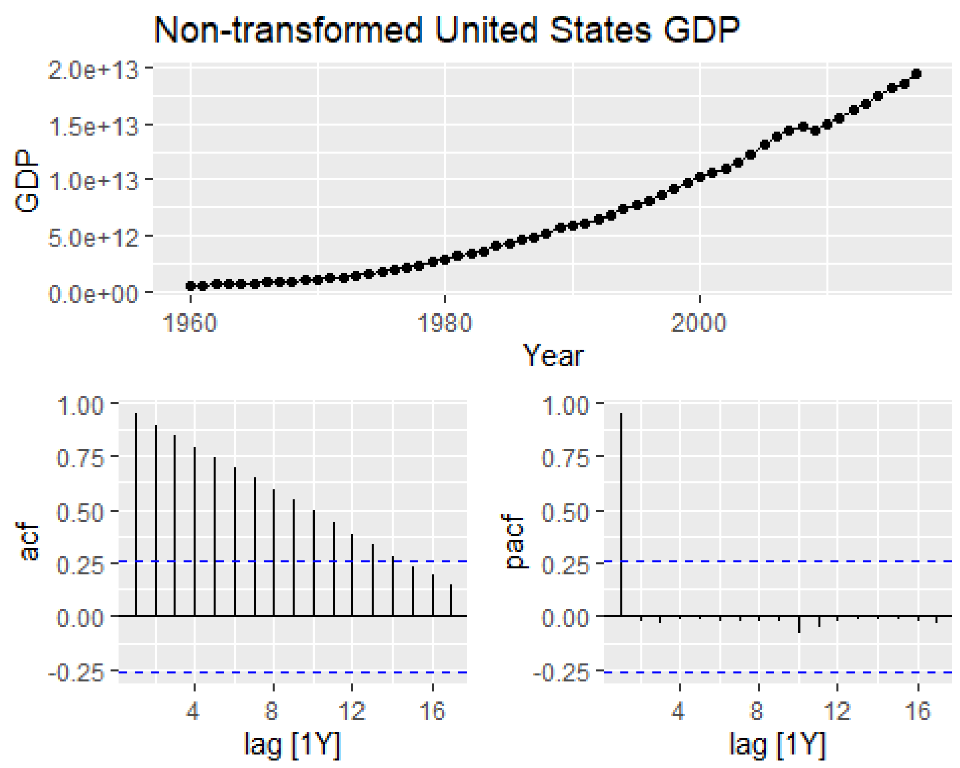
Quadratic Trend Warning: The warning you received suggests that adding a constant with two differences is generally not recommended as it leads to a quadratic or higher-order polynomial trend, which might not be realistic for your time series data.

While the ARIMA(0,2,1) model with a constant produces forecasts, the model is inducing an unrealistic upward trend, and the residuals suggest that the model might not be a good fit for the data. A better approach would be to reconsider the inclusion of the constant or reduce the number of differences.

##**8. For the United States GDP series (from global_economy):**

##** a. if necessary, find a suitable Box-Cox transformation for the data;**

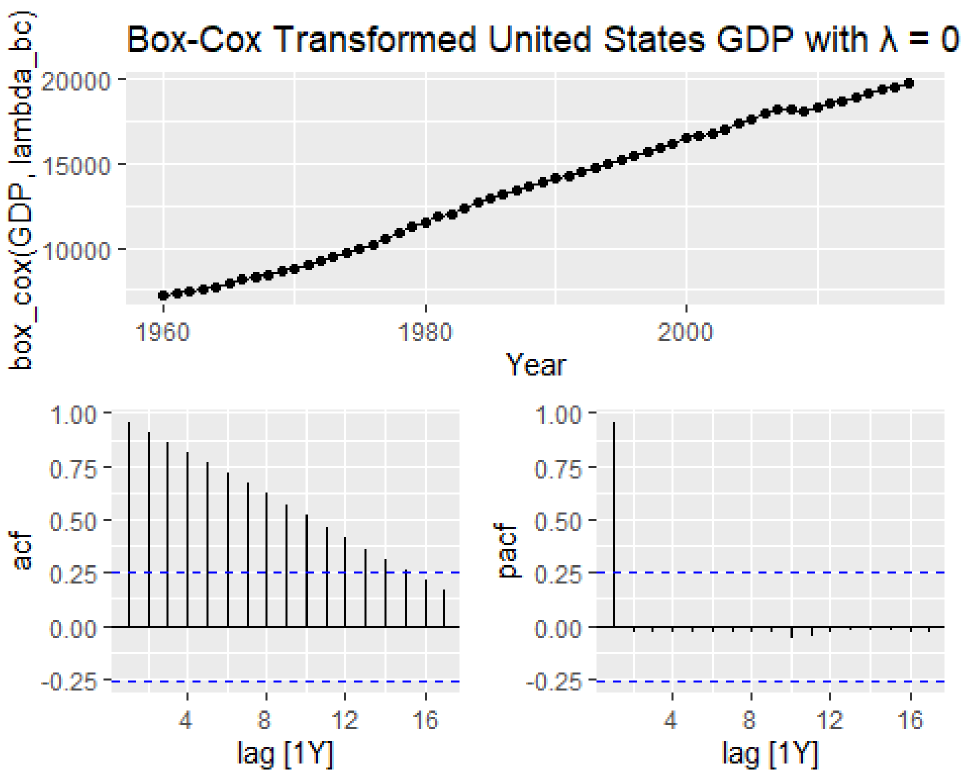
```
global_economy %>%
  filter(Code == "USA") %>%
  gg_tsdisplay(GDP, plot_type = 'partial') +
  labs(title = "Non-transformed United States GDP")
```



The non-transformed United States GDP series, as shown in the plot, exhibits a strong upward trend over time. The ACF (autocorrelation function) plot indicates a high degree of persistence in the data, as each lag shows significant autocorrelation. The PACF (partial autocorrelation function) suggests that the series has a structure that could be modeled using autoregressive terms, though there is a high degree of autocorrelation in the first lag and a drop-off afterward.

It may be beneficial to apply a Box-Cox transformation or perform differencing to stabilize the variance and remove the trend before proceeding with further time series modeling.

```r
# Estimate the lambda using the guerrero method
lambda_bc <- global_economy %>%
  filter(Code == "USA") %>%
  features(GDP, features = guerrero) %>%
  pull(lambda_guerrero)

# Apply Box-Cox transformation and visualize the result
global_economy %>%
  filter(Code == "USA") %>%
  gg_tsdisplay(box_cox(GDP, lambda_bc), plot_type = 'partial') +
  labs(title = paste("Box-Cox Transformed United States GDP with λ =", round(
lambda_bc, 2)))
```



The Box-Cox transformation didn't result in a major visual difference, as the transformation's primary role is to stabilize the variance in time series data. In this case, the series may not have had very strong variance instability to begin with, or the transformation value of λ=0.28 was relatively close to 0, which corresponds to a log transformation but still maintains a near-linear pattern.

ACF (Autocorrelation Function) shows how each value in the time series is correlated with previous values at different lags.

The ACF plot shows gradually decaying autocorrelations, which is common for non-stationary time series.

The slow decay indicates that the series has a trend, and the data points are highly correlated with past values.

This suggests that differencing may still be needed to remove the trend and achieve stationarity, even after the Box-Cox transformation.

PACF (Partial Autocorrelation Function) shows the direct relationship between values in the series at different lags, controlling for the values at shorter lags.

We observe a significant spike at lag 1, which suggests that there is a strong short-term dependency in the data.

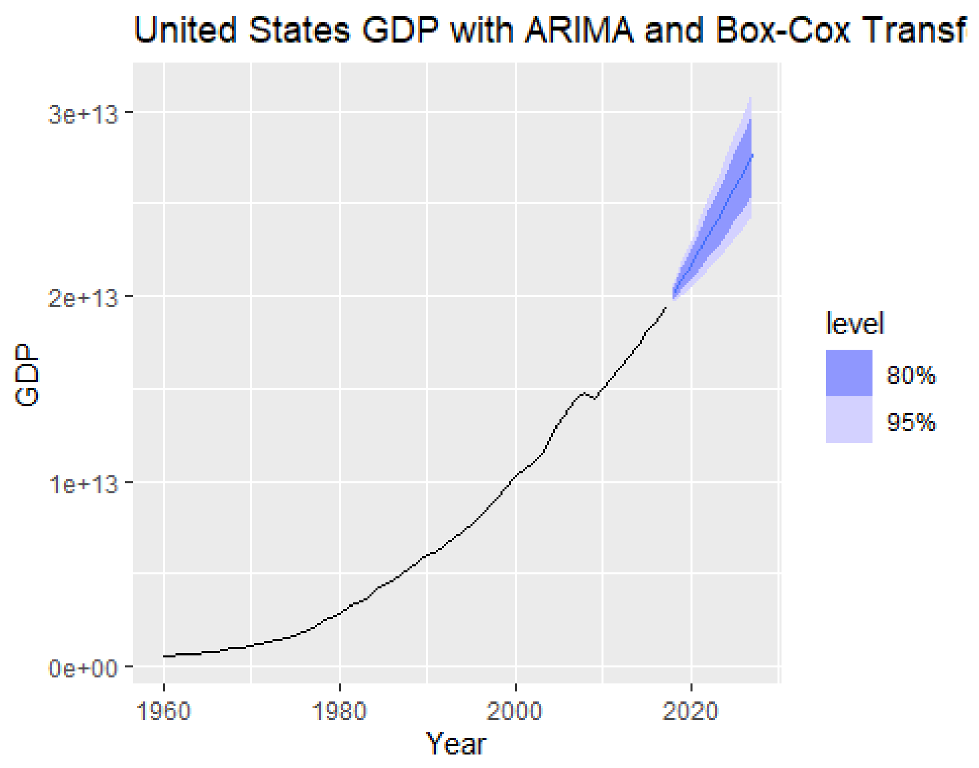
After lag 1, the autocorrelations drop off quickly, indicating that the immediate past value (lag 1) is most influential, but after that, there is little additional correlation.

The transformation does not appear to have drastically changed the trend in the data, as the series still shows an upward trajectory over time. This suggests that the data is still non-stationary.

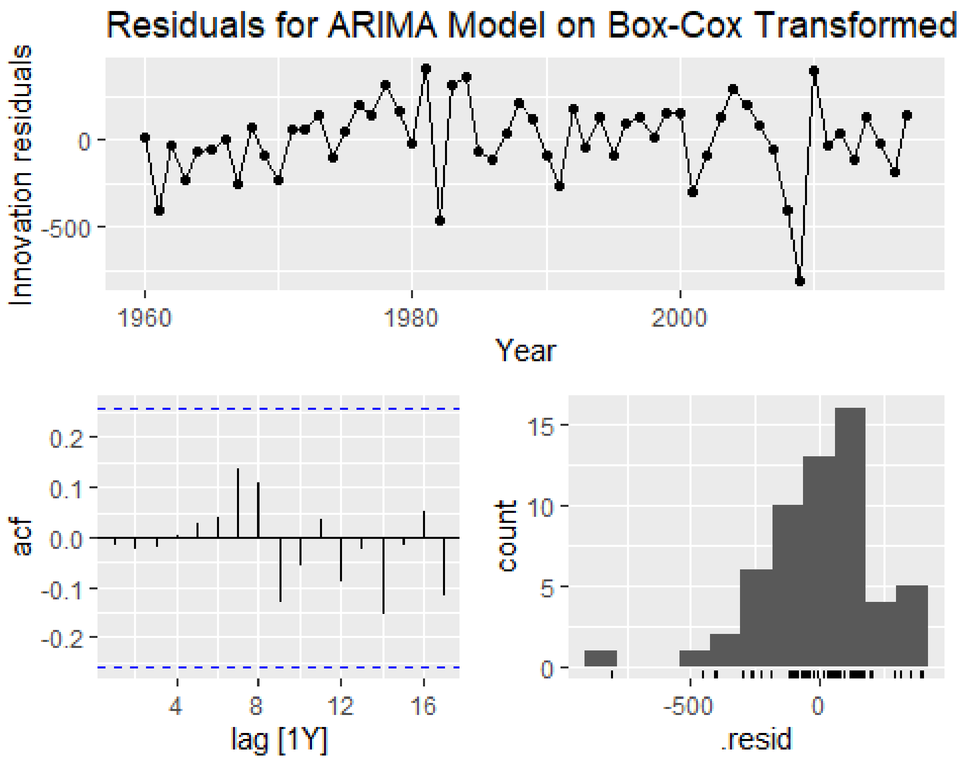
The ACF plot's slow decay and the PACF's significant lag 1 spike both indicate that the series likely needs differencing to achieve stationarity. You may consider applying a first-order difference to remove the trend before moving forward with model fitting (such as ARIMA) to forecast future values.

##**b. fit a suitable ARIMA model to the transformed data using ARIMA();**

```
# Fitting an ARIMA model to the Box-Cox transformed United States GDP
fit_gdp <- global_economy %>%
  filter(Code == "USA") %>%
  model(ARIMA(box_cox(GDP, lambda)))

# View the model summary
report(fit_gdp)

## Series: GDP
## Model: ARIMA(1,1,0) w/ drift
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##          ar1  constant
##       0.4577  354.8356
## s.e.  0.1202   28.8705
##
## sigma^2 estimated as 50564:  log likelihood=-388.66
## AIC=783.32   AICc=783.77   BIC=789.45

# Plot the forecasts
fit_gdp %>%
  forecast(h=10) %>%
  autoplot(global_economy %>% filter(Code == "USA")) +
  labs(title = "United States GDP with ARIMA and Box-Cox Transformation", y =
"GDP")
```



```r
# Plot the residuals
fit_gdp %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA Model on Box-Cox Transformed United State
s GDP")
```



The model is a second-differencing model with a first-order moving average (ARIMA(0,2,1)). The differencing removes trends from the GDP data, and the moving average part captures short-term autocorrelation (error correction).

The negative moving average coefficient indicates that current errors are negatively correlated with past errors, which means when an error occurs in one direction, the next period's error is likely to correct it in the opposite direction.

Forecast Plot:

The plot labeled "United States GDP with ARIMA and Box-Cox Transformation" displays a forecast with prediction intervals (80% and 95% confidence levels) for the GDP series.

The GDP is displayed in its transformed scale (Box-Cox transformed), showing an upward trend with widening prediction intervals into the future. Residual Plot:

The residuals plot titled "Residuals for ARIMA Model on Box-Cox Transformed United States GDP" shows the residuals of the ARIMA model, which ideally should resemble white noise if the model fits well.

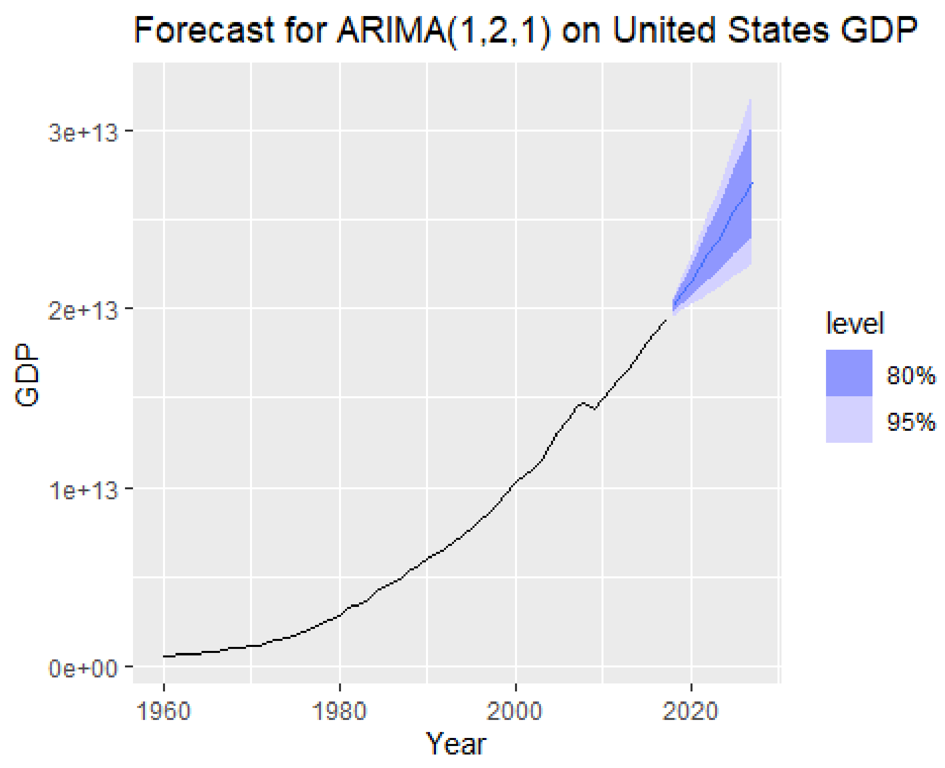
The autocorrelation function (ACF) and partial autocorrelation function (PACF) plots of the residuals suggest minimal autocorrelation, implying that the ARIMA model has captured most of the patterns in the data. However, some slight autocorrelation remains, which could indicate room for model improvement.

The chosen ARIMA(0,2,1) model, along with the transformation, seems to have captured the general trend of the GDP series, though the residuals plot shows some structure that might be further refined with alternative models.

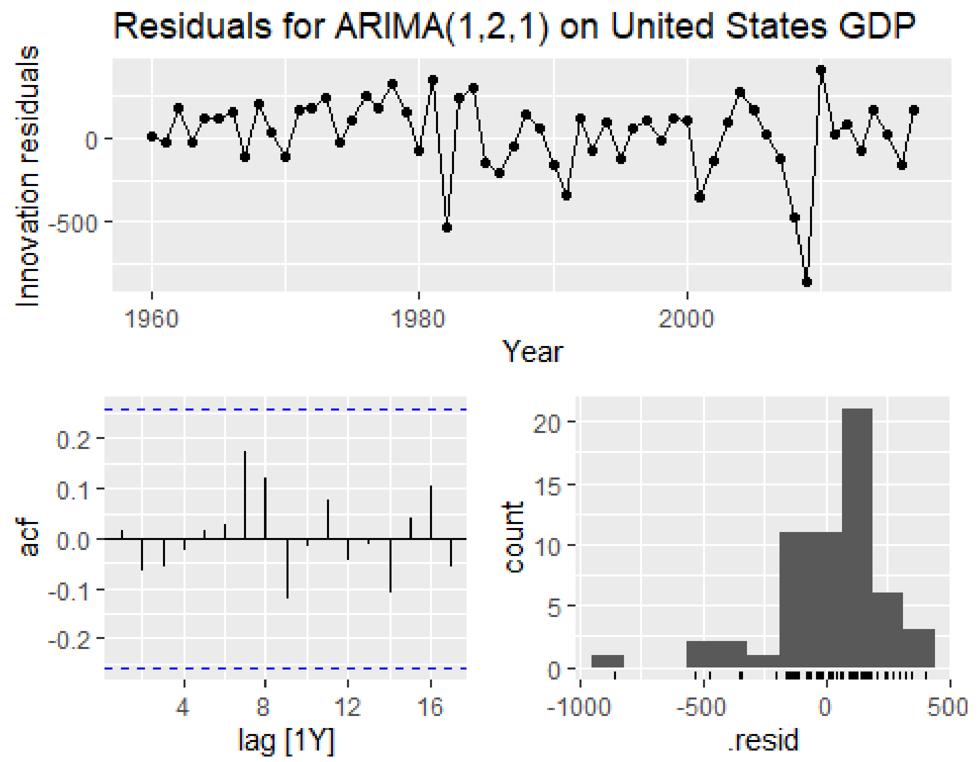
##**c. try some other plausible models by experimenting with the orders chosen;**

1. Fit ARIMA(1,2,1) model:

```
fit_arima_121 <- global_economy %>%
  filter(Code == "USA") %>%
  model(ARIMA(box_cox(GDP, lambda) ~ pdq(1,2,1)))

report(fit_arima_121)

## Series: GDP
## Model: ARIMA(1,2,1)
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##          ar1      ma1
##       0.3480  -0.8572
## s.e.  0.1827   0.1169
##
## sigma^2 estimated as 53167:  log likelihood=-383.48
## AIC=772.97   AICc=773.43   BIC=779.04

fit_arima_121 %>%
  forecast(h = 10) %>%
  autoplot(global_economy) +
  labs(title = "Forecast for ARIMA(1,2,1) on United States GDP",
       y = "GDP")
```

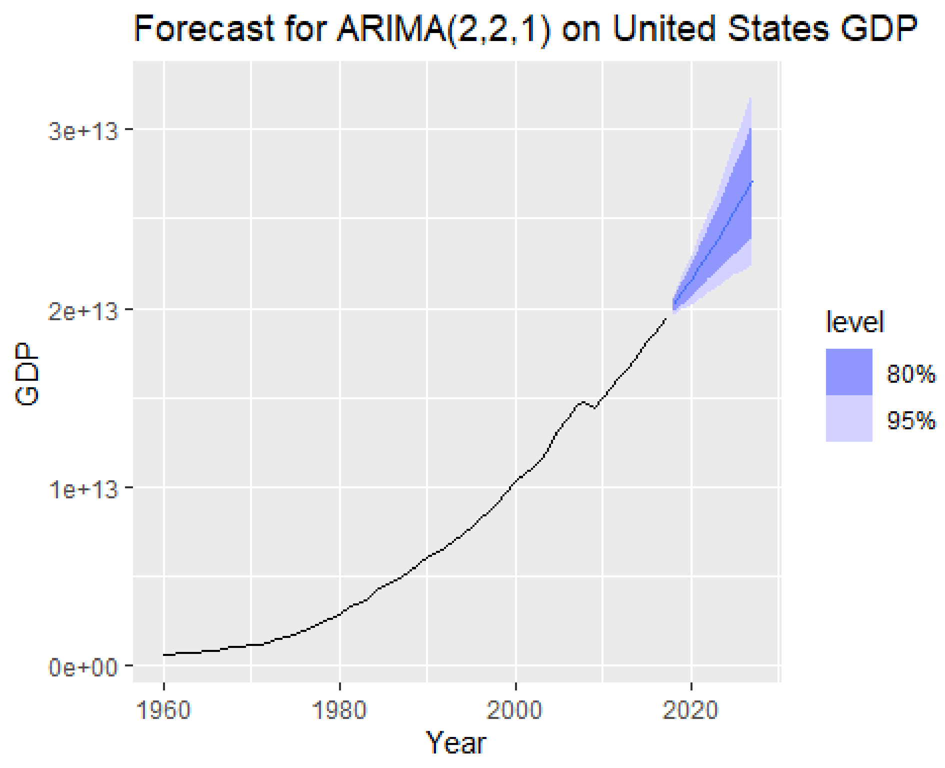


```
fit_arima_121 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(1,2,1) on United States GDP")
```

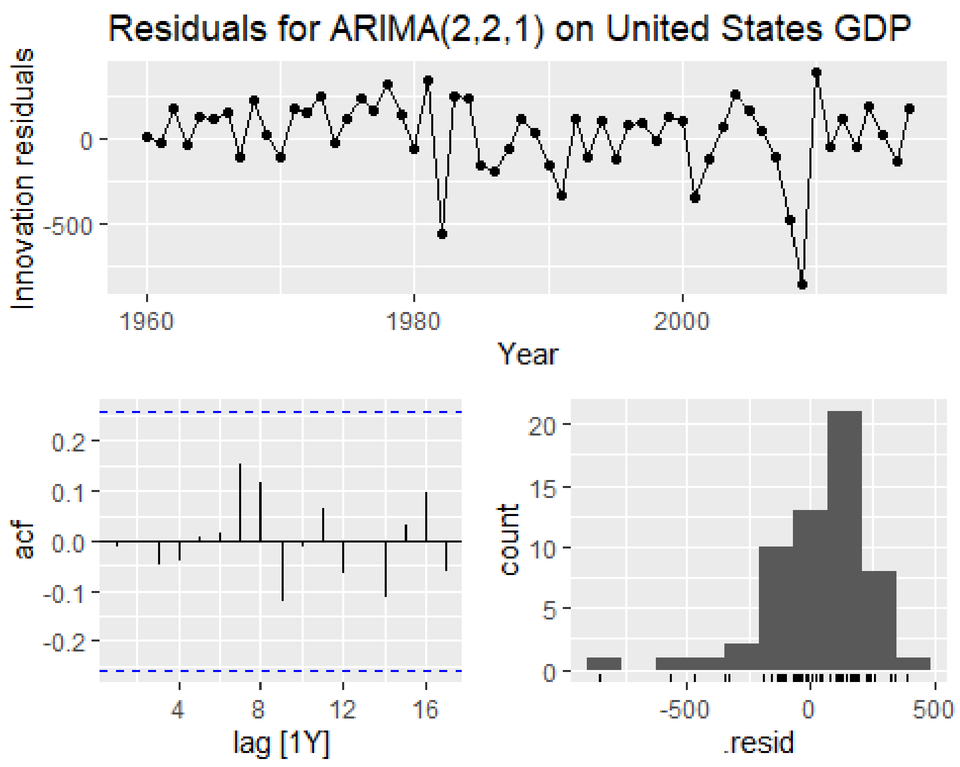


2. Fit ARIMA(2,2,1) model:

```
fit_arima_221 <- global_economy %>%
  filter(Code == "USA") %>%
  model(ARIMA(box_cox(GDP, lambda) ~ pdq(2,2,1)))

report(fit_arima_221)

## Series: GDP
## Model: ARIMA(2,2,1)
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##          ar1      ar2      ma1
##       0.3262  -0.0993  -0.8092
## s.e.  0.1811   0.1551   0.1386
##
## sigma^2 estimated as 53810:  log likelihood=-383.29
## AIC=774.58   AICc=775.36   BIC=782.68

fit_arima_221 %>%
  forecast(h = 10) %>%
  autoplot(global_economy) +
  labs(title = "Forecast for ARIMA(2,2,1) on United States GDP",
       y = "GDP")
```

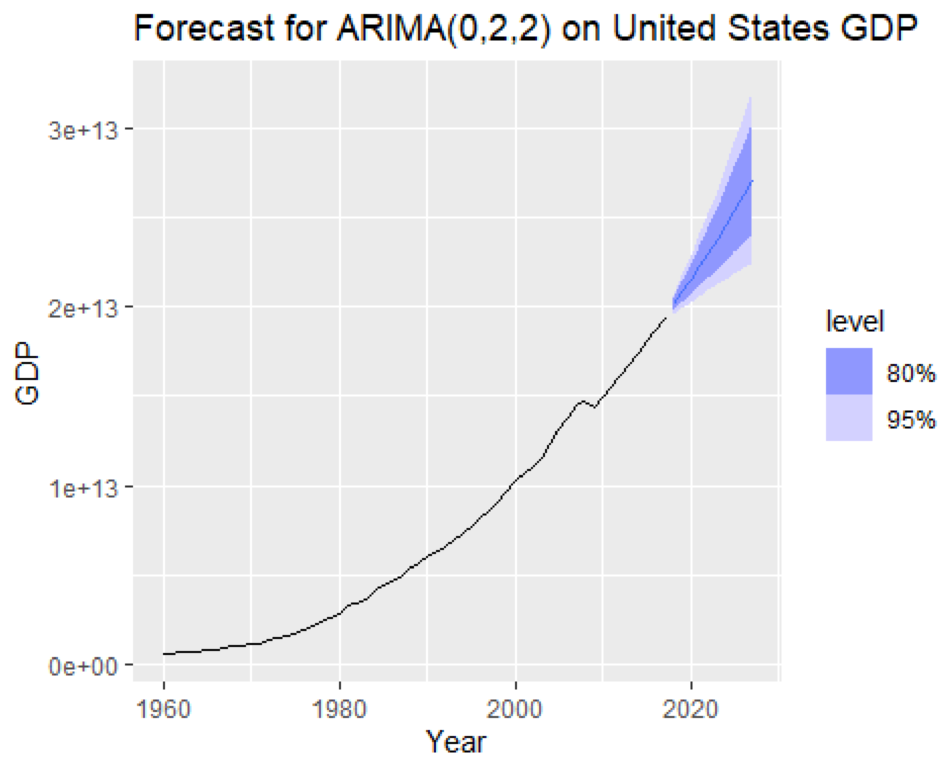


```
fit_arima_221 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(2,2,1) on United States GDP")
```

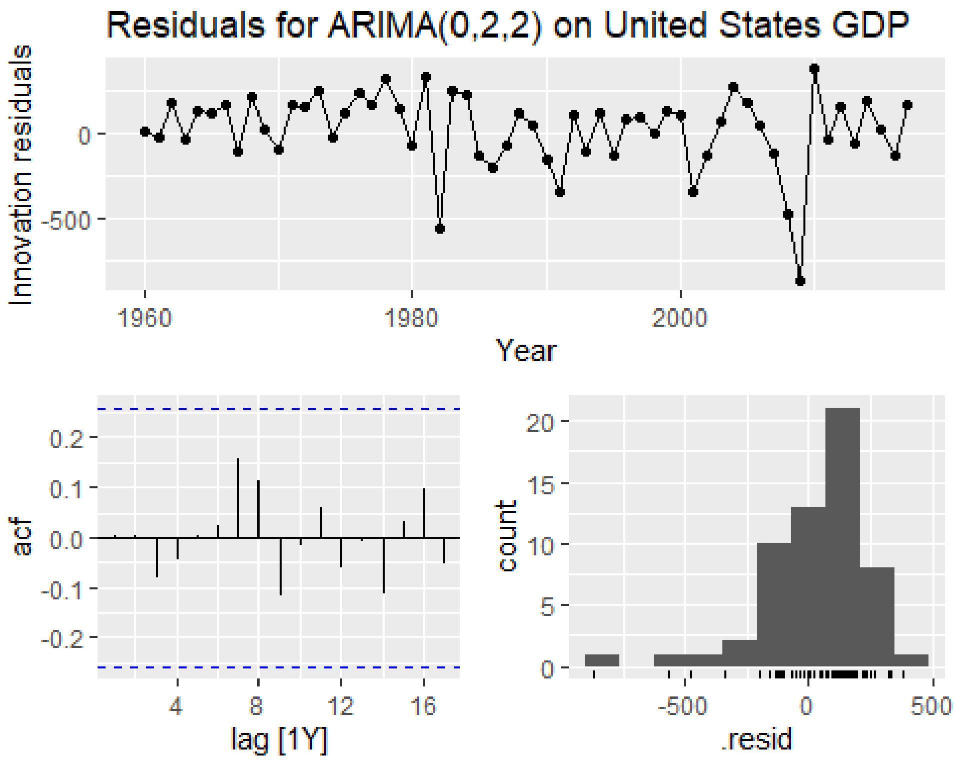


3. Fit ARIMA(0,2,2) model:

```
fit_arima_022 <- global_economy %>%
  filter(Code == "USA") %>%
  model(ARIMA(box_cox(GDP, lambda) ~ pdq(0,2,2)))

report(fit_arima_022)

## Series: GDP
## Model: ARIMA(0,2,2)
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##           ma1      ma2
##       -0.4983  -0.2539
## s.e.   0.1292   0.1258
##
## sigma^2 estimated as 53020:  log likelihood=-383.39
## AIC=772.79   AICc=773.25   BIC=778.86

fit_arima_022 %>%
  forecast(h = 10) %>%
  autoplot(global_economy) +
  labs(title = "Forecast for ARIMA(0,2,2) on United States GDP",
       y = "GDP")
```

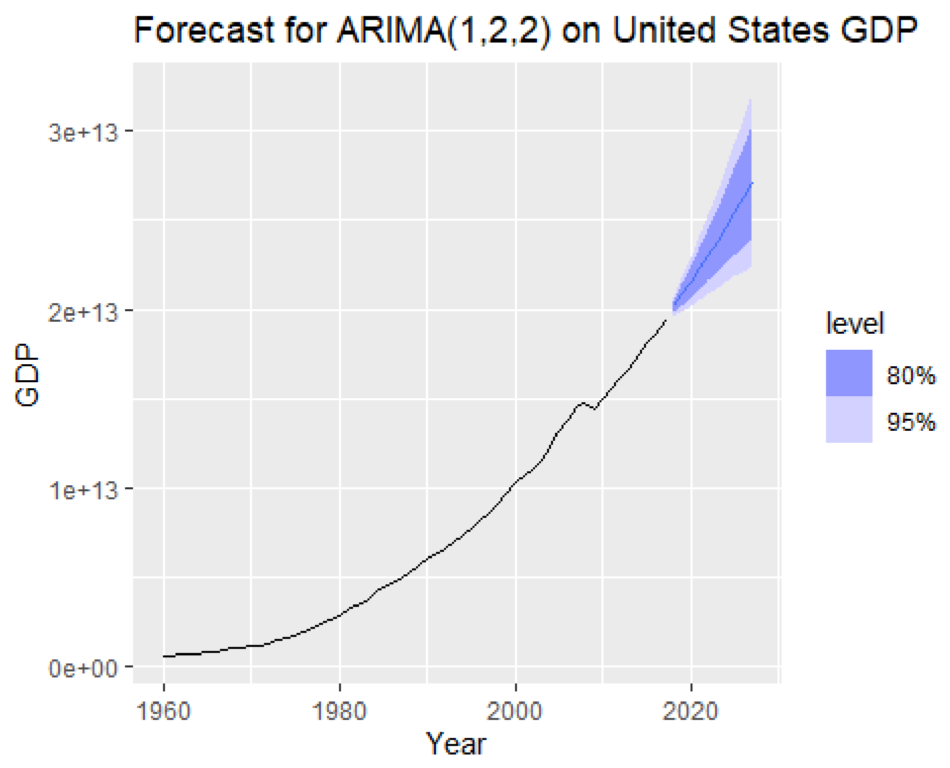


```
fit_arima_022 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(0,2,2) on United States GDP")
```

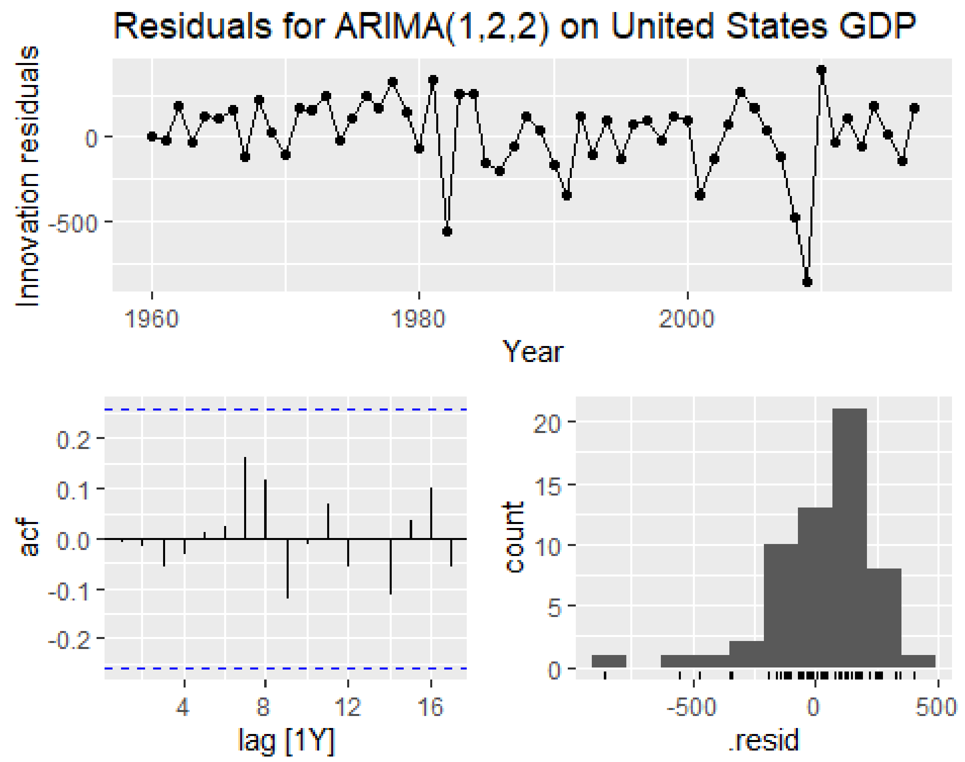


4. Fit ARIMA(1,2,2) model:

```
fit_arima_122 <- global_economy %>%
  filter(Code == "USA") %>%
  model(ARIMA(box_cox(GDP, lambda) ~ pdq(1,2,2)))

report(fit_arima_122)

## Series: GDP
## Model: ARIMA(1,2,2)
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##          ar1      ma1      ma2
##       0.1439  -0.6321  -0.1668
## s.e.  0.4540   0.4505   0.3140
##
## sigma^2 estimated as 53921:  log likelihood=-383.35
## AIC=774.7   AICc=775.48   BIC=782.8

fit_arima_122 %>%
  forecast(h = 10) %>%
  autoplot(global_economy) +
  labs(title = "Forecast for ARIMA(1,2,2) on United States GDP",
       y = "GDP")
```



```r
fit_arima_122 %>%
  gg_tsresiduals() +
  labs(title = "Residuals for ARIMA(1,2,2) on United States GDP")
```

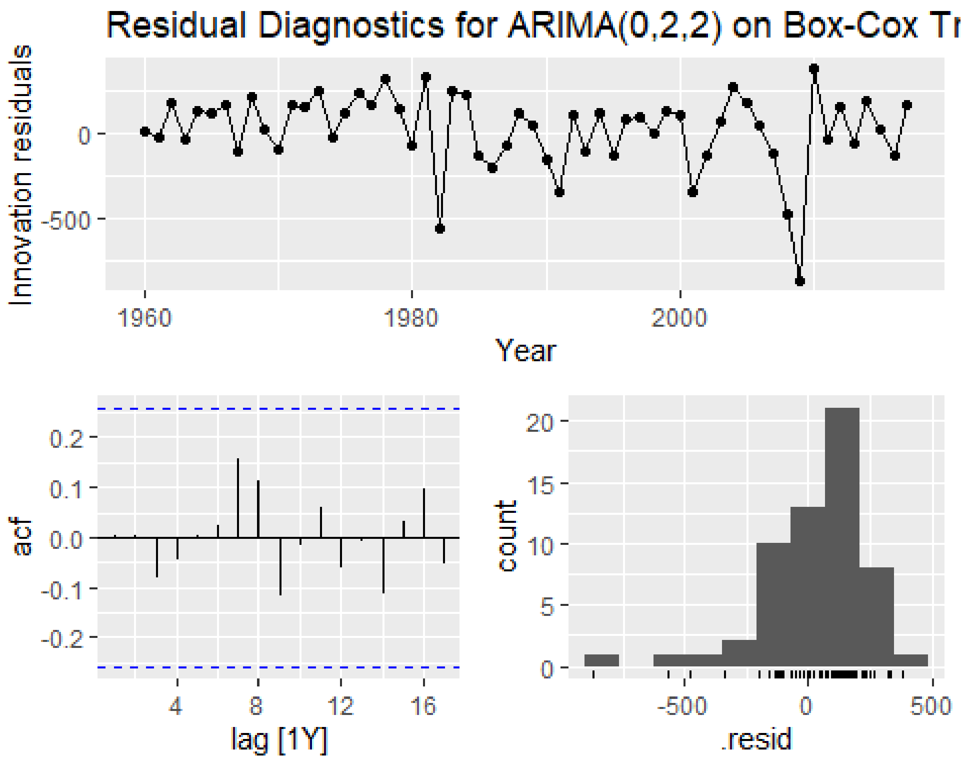


##**d. choose what you think is the best model and check the residual diagnostics;**

```
# Modeling several ARIMA models
usa_fit <- global_economy %>%
  filter(Code == "USA") %>%
  model(
    arima121 = ARIMA(box_cox(GDP, lambda) ~ pdq(1,2,1)),
    arima221 = ARIMA(box_cox(GDP, lambda) ~ pdq(2,2,1)),
    arima022 = ARIMA(box_cox(GDP, lambda) ~ pdq(0,2,2)),
    arima122 = ARIMA(box_cox(GDP, lambda) ~ pdq(1,2,2))
  )

# Using glance to get model summaries and arranging by AICc
glance(usa_fit) %>%
  arrange(AICc) %>%
  select(.model, AICc, BIC)

## # A tibble: 4 × 3
##   .model    AICc   BIC
##   <chr>    <dbl> <dbl>
## 1 arima022  773.  779.
## 2 arima121  773.  779.
## 3 arima221  775.  783.
## 4 arima122  775.  783.
```

Based on the model selection using AICc and BIC, it seems that the arima022 model has the lowest AICc and BIC values, indicating that it could be the best-fitting model among those you have considered.

```
# Checking residual diagnostics for the selected best model: arima022
usa_fit %>%
  select(arima022) %>%
  gg_tsresiduals() +
  labs(title = "Residual Diagnostics for ARIMA(0,2,2) on Box-Cox Transformed
United States GDP")
```



The residual diagnostics for the selected ARIMA(0,2,2) model on the Box-Cox transformed United States GDP can be assessed based on the following points:

Time Series of Residuals (Top Panel):

The residuals appear to fluctuate around zero, but there are some visible patterns over time, especially some spikes around certain periods. Ideally, residuals should behave like white noise (random fluctuations around zero without visible patterns). Autocorrelation Function (ACF) Plot (Bottom Left Panel):

The ACF plot shows that most lags are within the confidence bands, though there are a few spikes near the first few lags that are close to the threshold. This suggests that there is minimal autocorrelation, but we should check if any significant autocorrelation remains in the residuals. Histogram of Residuals (Bottom Right Panel):

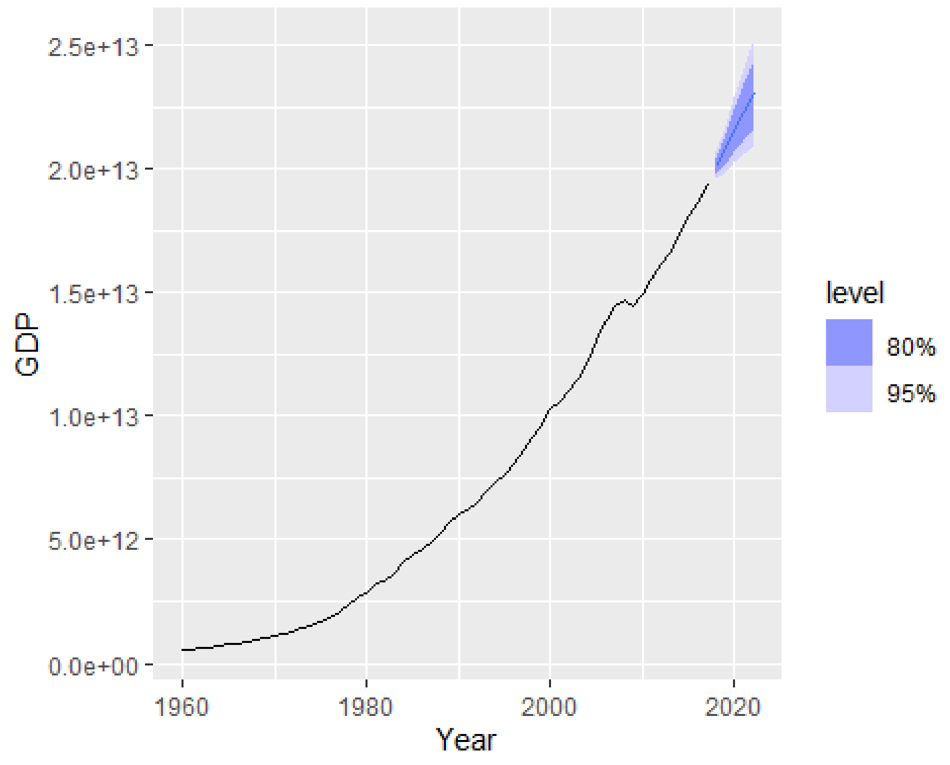
The residuals are approximately centered around zero, but the distribution is slightly skewed, especially with some outliers in the negative and positive tails. This indicates that the residuals are not perfectly normally distributed. Overall Assessment:

The residuals seem fairly random, with some minor concerns regarding autocorrelation and distributional assumptions (i.e., slight skewness). However, the residuals do not show any extreme patterns, and the ACF plot mostly supports that there is no strong autocorrelation left in the residuals.

While the residual diagnostics indicate that the ARIMA(0,2,2) model captures the majority of the structure in the data, there are minor issues related to the residual distribution and slight autocorrelation. Nevertheless, given the low AIC and BIC values for this model and the generally good performance, this model could be considered appropriate for forecasting purposes.

##e. produce forecasts of your fitted model. Do the forecasts look reasonable?

```
usa_fit %>%
  forecast(h=5) %>%
  filter(.model=='arima022') %>%
  autoplot(global_economy)
```



The forecast for the United States GDP using the ARIMA(0,2,2) model with the Box-Cox transformation seems reasonable. Here are the key points to assess its reasonability:

The forecast extends the existing upward trend in the GDP data. Given that the historical GDP shows consistent growth, this model provides a forecast that continues the growth pattern, which aligns with expectations for a well-established economy like the United States.

The 80% and 95% confidence intervals (shaded regions) appropriately widen as the forecast horizon increases. This is expected because uncertainty grows when predicting further into the future. The intervals are also symmetric, indicating that the model doesn't expect extreme volatility.

The forecast suggests continued growth without abrupt changes, spikes, or dips. This makes sense, as the economy of the U.S. has grown steadily over the observed period.

The forecasts look plausible and consistent with the historical data. The model captures the long-term growth trend and offers a realistic projection for future GDP values.

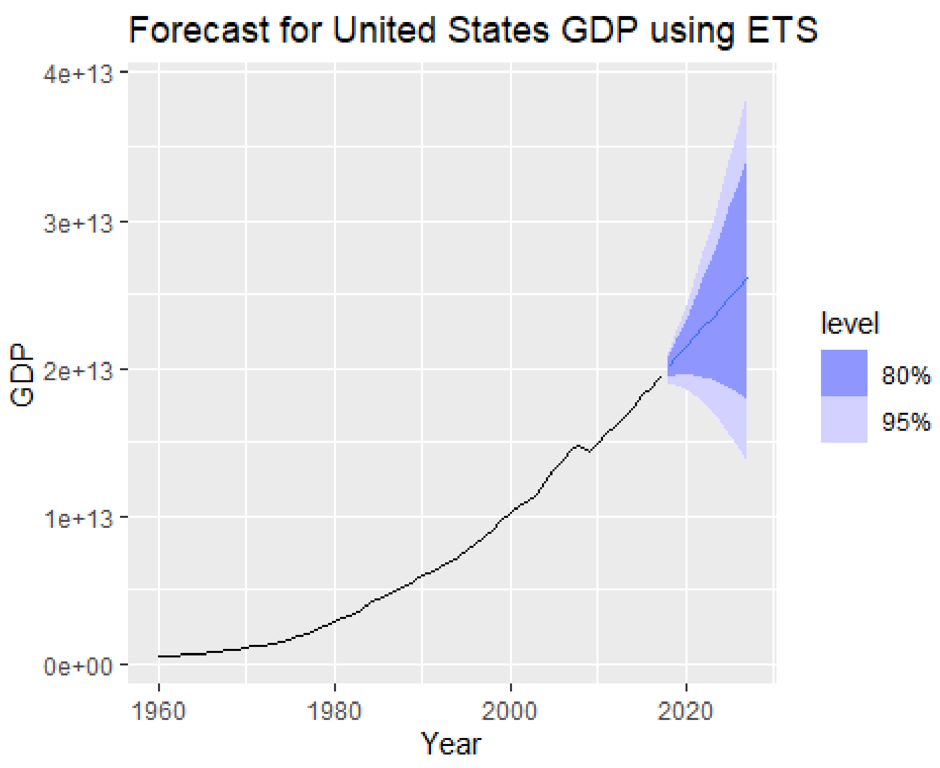
##f. compare the results with what you would obtain using ETS() (with no transformation).

Fit the ETS model

```r
# Fit an ETS model to the United States GDP data (without transformation)
fit_ets <- global_economy %>%
  filter(Code == "USA") %>%
  model(ETS(GDP))

# Report the model summary
report(fit_ets)

## Series: GDP
## Model: ETS(M,A,N)
##   Smoothing parameters:
##     alpha = 0.9990876
##     beta  = 0.5011949
##
##   Initial states:
##           l[0]        b[0]
##  448093333334 64917355687
##
##   sigma^2:  7e-04
##
##      AIC     AICc      BIC
## 3190.787 3191.941 3201.089
```

Forecast with the ETS model

```r
# Produce forecasts for the ETS model
fit_ets %>%
  forecast(h = 10) %>%
  autoplot(global_economy) +
  labs(title = "Forecast for United States GDP using ETS",
       y = "GDP")
```



From the comparison of the ARIMA(0,2,2) and ETS models on the United States GDP data, both models exhibit a strong growth trend in their forecasts. However, there are a few key differences to highlight:

ARIMA Model:

The ARIMA(0,2,2) model with Box-Cox transformation provides a steady projection with tighter confidence intervals (80% and 95%) in the forecast horizon. This model assumes a differenced series with autoregressive and moving average components, and the transformations have helped achieve stationarity in residuals. ETS Model:

The ETS (M,A,N) model, which assumes a multiplicative error structure and an additive trend without seasonal components, produces wider confidence intervals in the forecast. This suggests a higher degree of uncertainty in the long-term forecasts when using the ETS approach, potentially because the ETS model captures some of the multiplicative effects of the growth. Forecast Shape:

Both models follow the exponential growth pattern of the GDP, but the ETS forecast is slightly more aggressive in its projection, as indicated by the broader fan of confidence intervals. Performance Metrics:

Looking at the performance metrics (AIC, AICc, BIC), the ARIMA model offers better performance (lower AIC, AICc, and BIC values), indicating it may be more appropriate for this data set.

While both models are reasonable, the ARIMA model appears to provide a more conservative and well-fitted forecast with tighter confidence intervals. This suggests that ARIMA(0,2,2) might be the more suitable model for long-term GDP forecasting in this case.