DATA624_Homework6

John Ferrara

2025-03-15

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

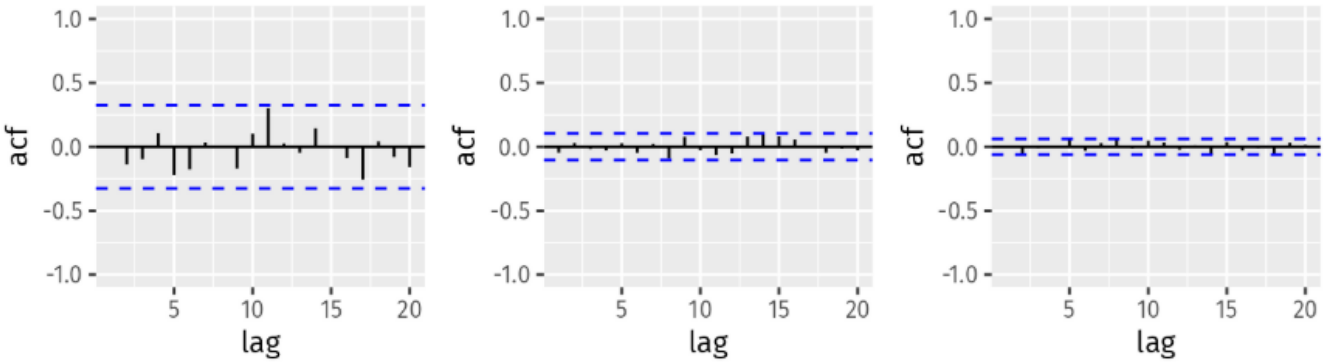


Figure 9.32: Left: ACF for a white noise series of 36 numbers. Middle: ACF for a white noise series of 360 numbers. Right: ACF for a white noise series of 1,000 numbers.

Figure 9.32

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

In this figure there are several sub figures. It seems that the left most figure contains white noise, as there doesn't seem to be any discernible pattern in variances being plotted here. The middle image, does not seem to be white noise, although its possible. There could be somewhat of a pattern in this middle sub figure, while super subtle, the pattern could to be similar to that of a sine wave with varying degrees of strength. For this image, the strength of the sine-like pattern seems to oscillate. That is it starts off small, gets a but stronger, and then diminishes again. Lastly, the right-most sub figure seems to be white noise. While there are portions of this sub figure that could be somewhat of a pattern, overall it seems to be just noise. None of the plotted values in these charts exceed the 95% confidence interval lines.

b) Why are the critical values at different distances from the mean of zero? Why are the autocorrelations different in each figure when they each refer to white noise?

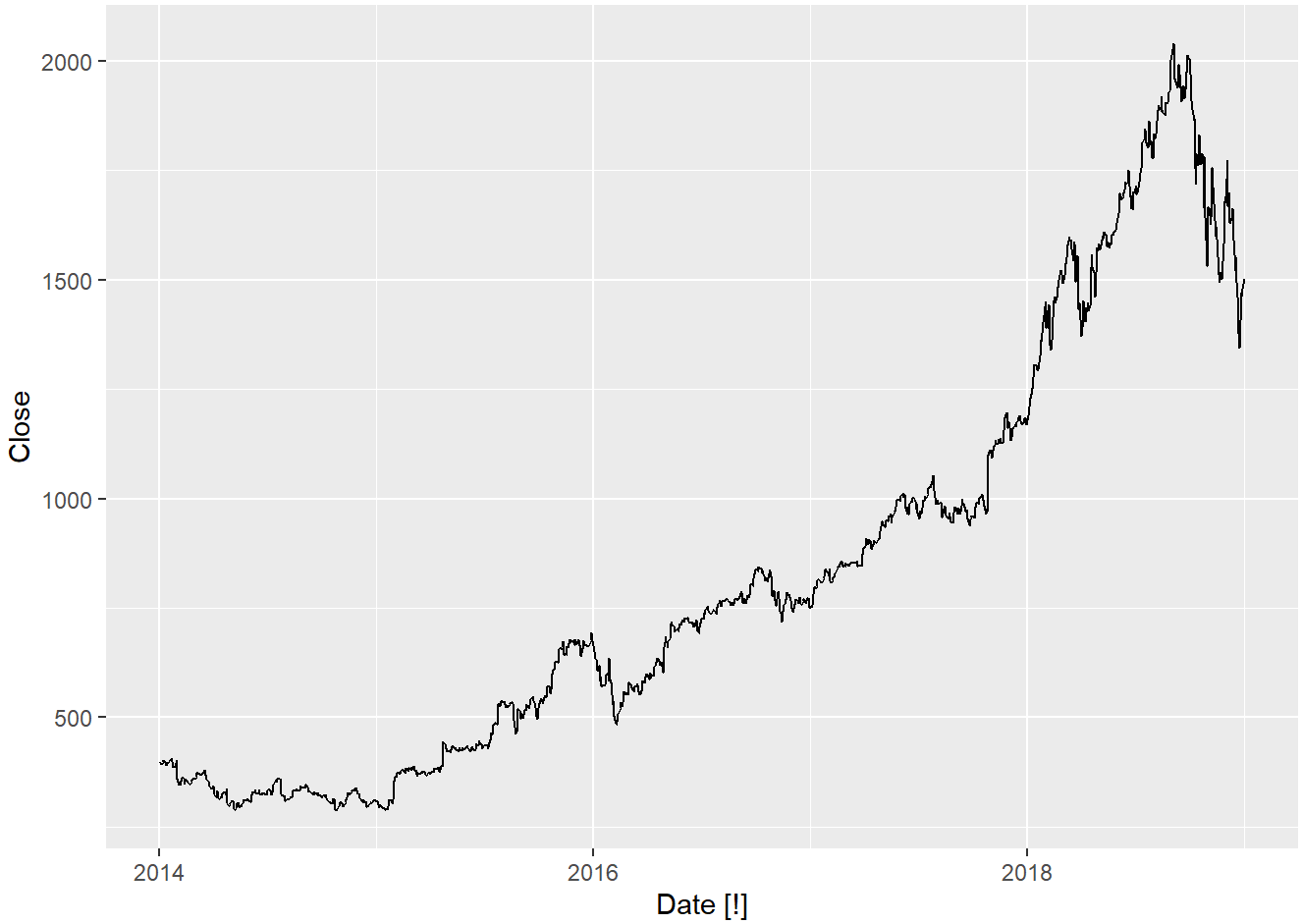
The critical values are at different distances from the mean of zero because of the dataset they are plotted from. Going from left to right, the dataset consisted of 36 random numbers 360 random numbers, and then 1000 random numbers. As the pool of random numbers got larger, the confidence interval where 95% of the data sits got more and more narrow. With a smaller pool of numbers there is the potential for much more variation in the numbers distribution, while larger random numbers would follow more of a normal distribution pattern which would influence the autocorrelation patterns. The autocorrelation patters are different even though they each refer to white noise because the autocorrelation is the numbers within a data set being

compared to other numbers in the same data set. So the variations in critical values and auto correlations are going to be different for each grouping of random numbers regardless of whether there is a pattern in the data or not.

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.

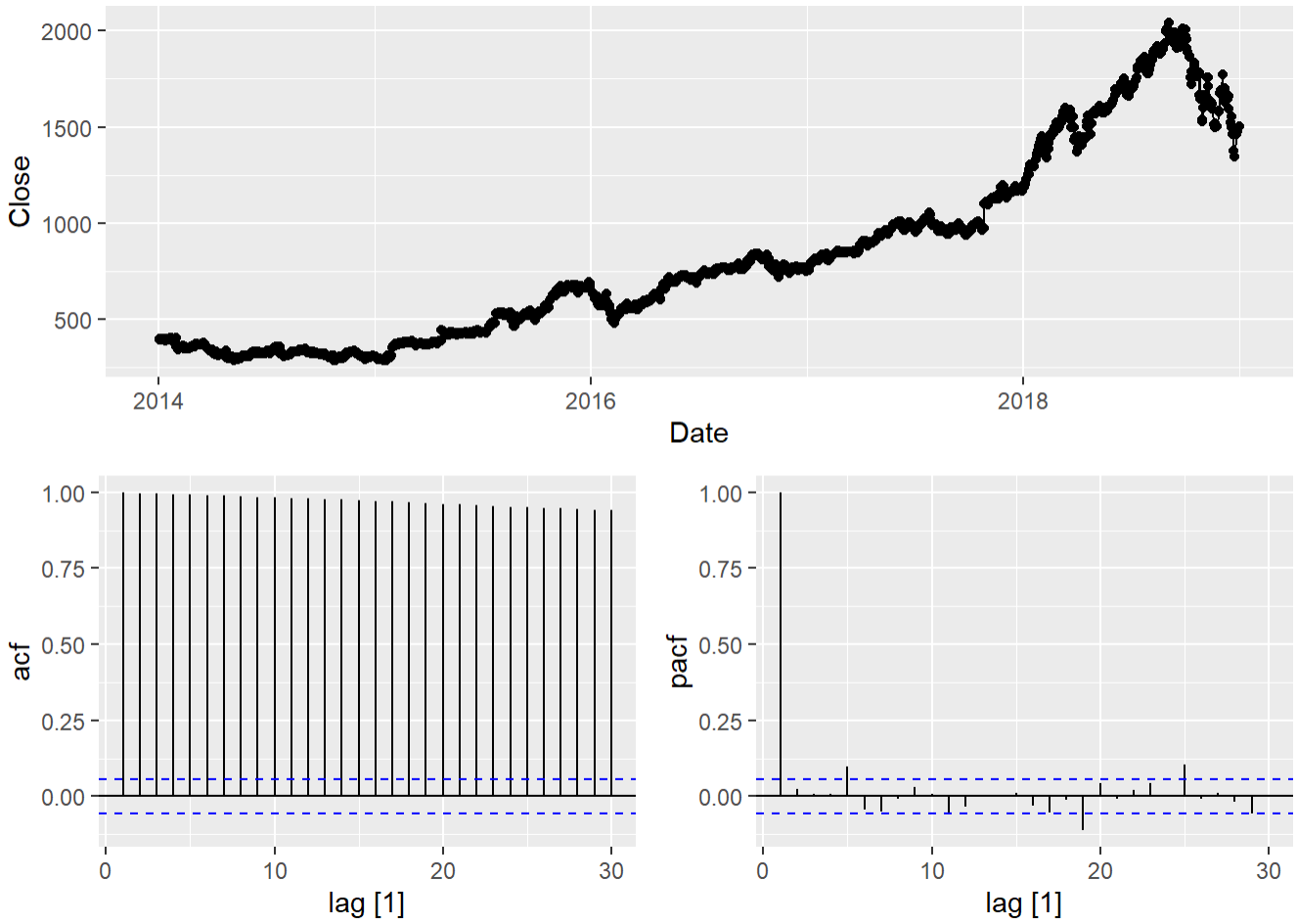
```
#table with just amazon
amazon <- gafa_stock |> filter(Symbol =='AMZN')|> select(-Open, -High, -Adj_Close, -Low, -Vol
ume)
print(amazon |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = Close`
```



```
## Here is the ACF and PACF plots for the raw non-differentiated or otherwise transformed dat
a.
print(amazon |> gg_tsdisplay(Close,plot_type ="partial"))
```

```
## Warning: Provided data has an irregular interval, results should be treated with caution.
Computing ACF by observation.
## Provided data has an irregular interval, results should be treated with caution. Computing
ACF by observation.
```



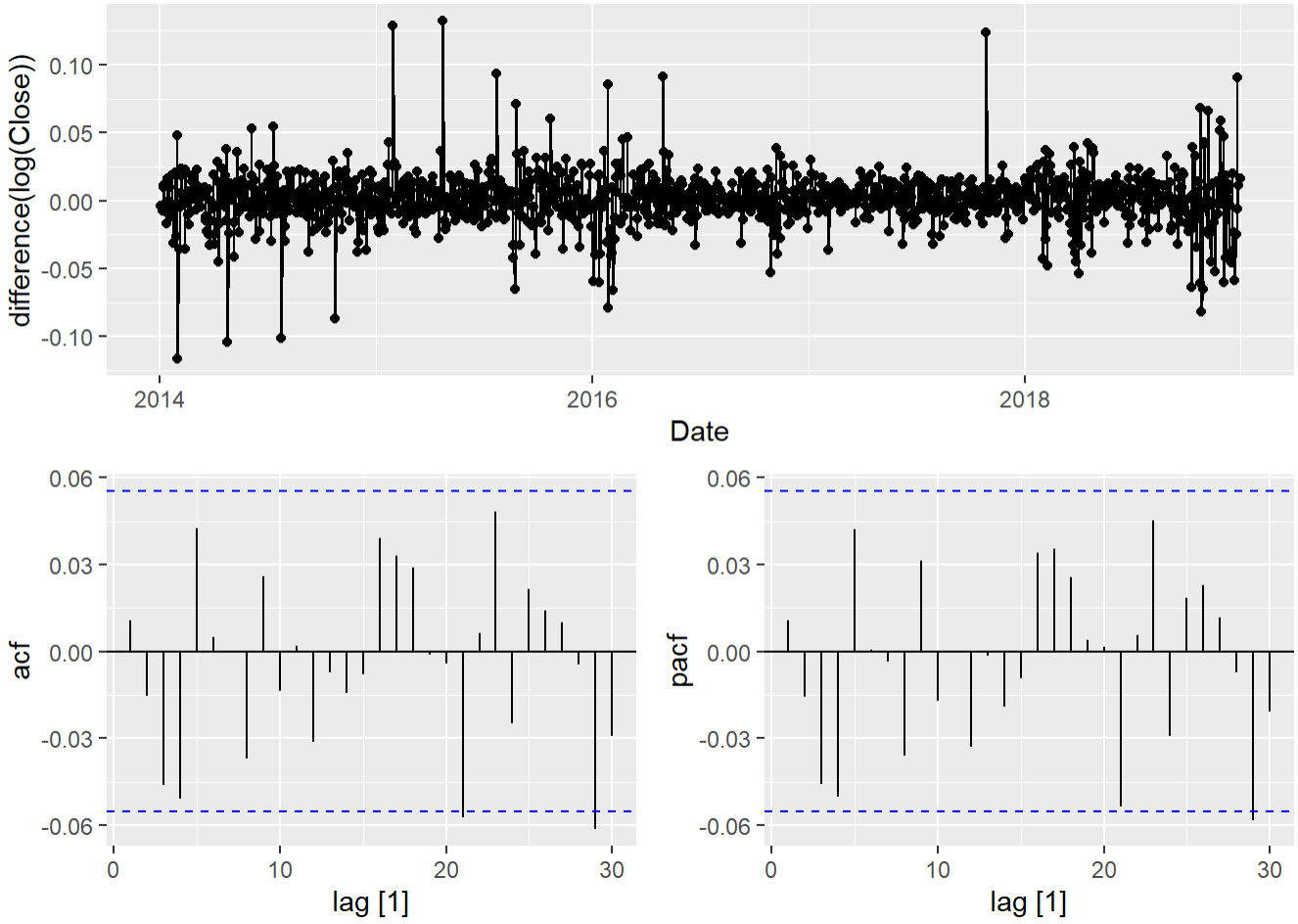
After looking at these plots, firstly the data is not stationary. According to the first plot, there is a trend in the raw data that needs to be flattened out to have the same type o f varying values for different sub time frames in this larger time frame. Additionally, when looking at the ACF chart all of the autocorrelation values start at 1 and slowly decrease jus t under 1. All of these values are very much so outside of the Confidence interval lines. Thi s means there is a fairly consistent trend in the data, thus not stationary. Lastly, for the PACF chart there is a very large spike value in the beginning of the data,and 2 to 3 smaller spikes that are also outside the confidence interval band. This supports the non-stationary n ature of the data.

```
## No seasonality, but trends present in the close prices of the stock.
print(amazon |> gg_tsdisplay(difference(log(Close)),plot_type ="partial"))
```

```
## Warning: Provided data has an irregular interval, results should be treated with caution.
Computing ACF by observation.
## Provided data has an irregular interval, results should be treated with caution. Computing
ACF by observation.
```

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

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



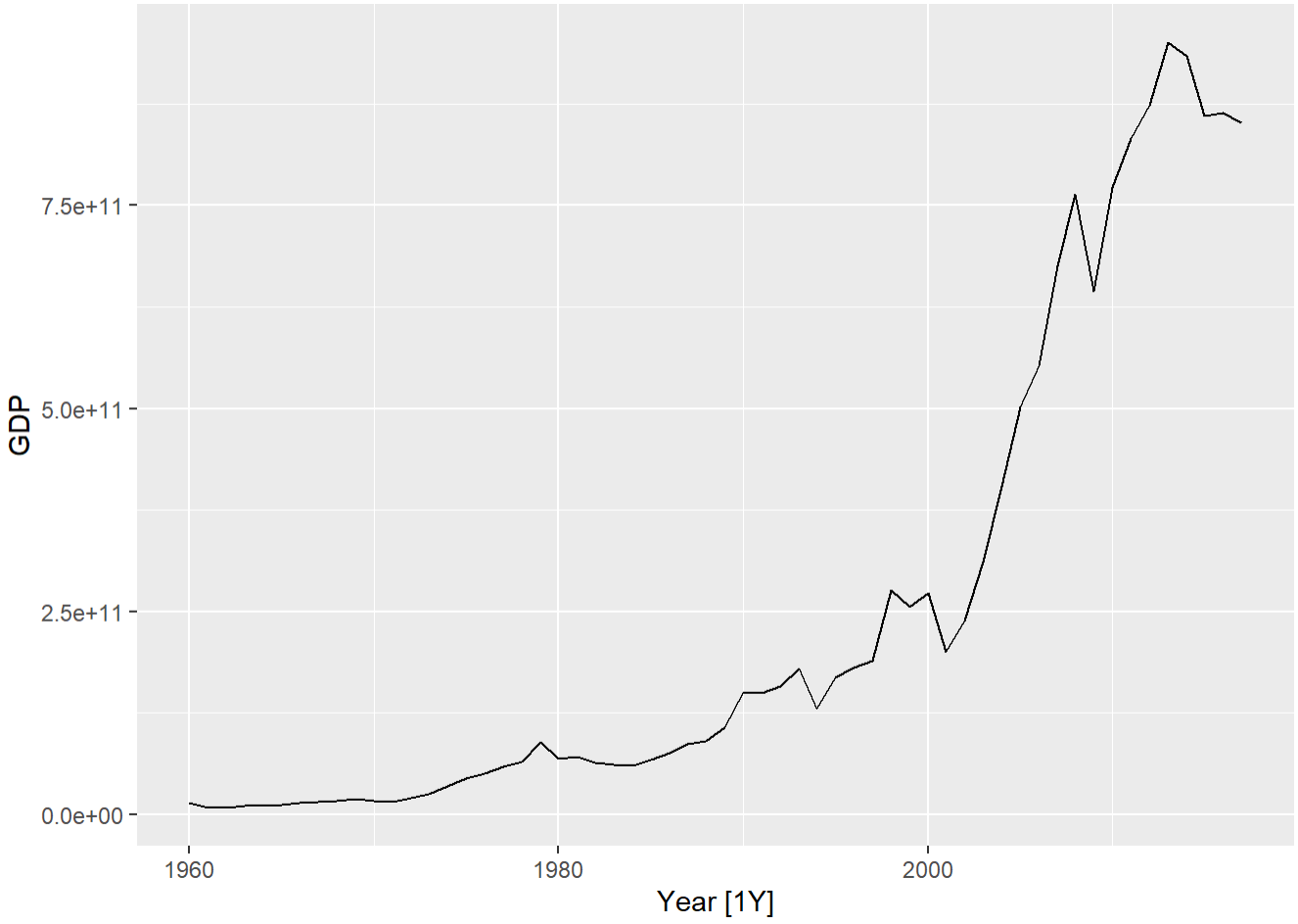
 # Lastly, you can see here, with the logged and differentiated chart that the data is now sta tionary. There is no trend in the top chart, which now looks like white noise. The ACF and PA CF charts both look much more reasonable with nearly all of the values being within the 95% c onfidence interval range.

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.

```
turkey_economy <- global_economy |> filter(Country =='Turkey')

print(turkey_economy|> autoplot())
```

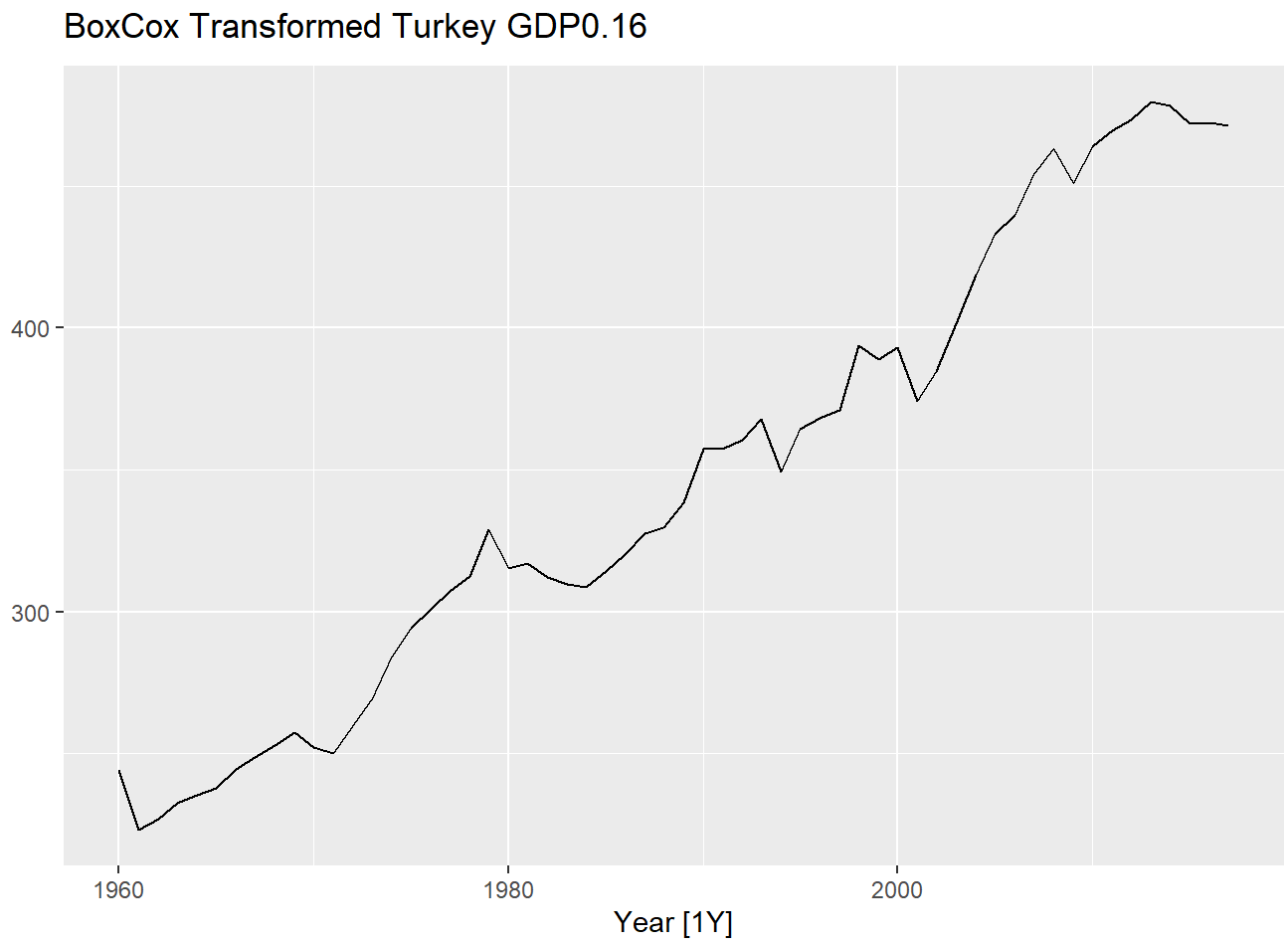
```
## Plot variable not specified, automatically selected `.vars = GDP`
```




```
lambda <- turkey_economy |>
  features(GDP, features = guerrero) |>
  pull(lambda_guerrero)
print(lambda) ## 0.157 Close to a log transformation lambda
```

```
## [1] 0.1572187
```

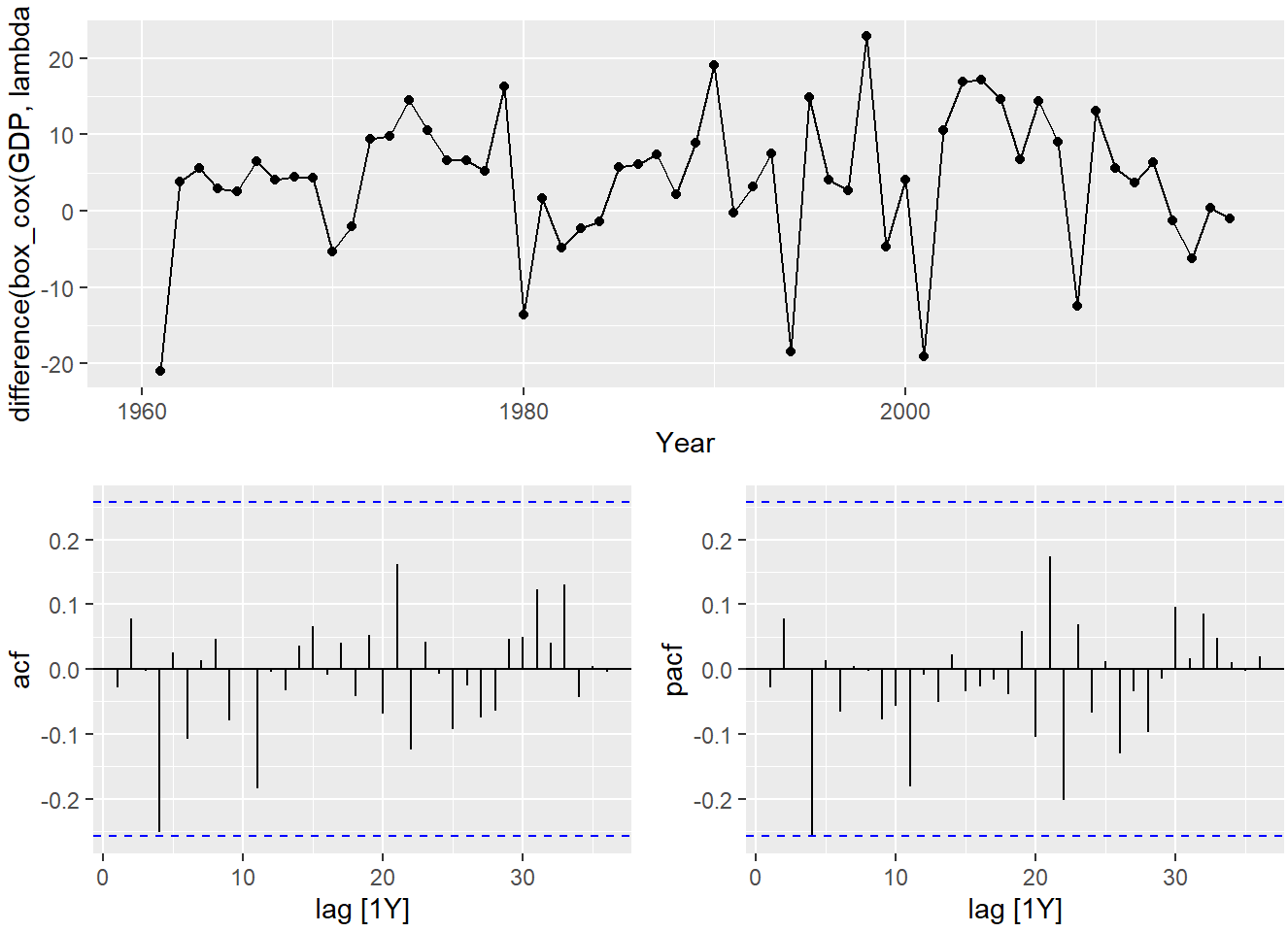
```
## Box Cox Transform successful now to find the differentiation needs
print(turkey_economy |>  autoplot(box_cox(GDP, lambda)) +  labs(y = "",title = latex2exp::TeX
(paste0("BoxCox Transformed Turkey GDP",round(lambda,2)))))
```




```
## There is no seasonality in the data, just a general trend, performing one difference for t
he data. Seeing if second will be needed. Don't think it is, as the data now seems stationary
. The top plot is white noise, and the ACF and PACF plots are within the proper critical valu
e bounds.
print(turkey_economy |> gg_tsdisplay(difference(box_cox(GDP, lambda)),plot_type ="partial", l
ag=36))
```

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

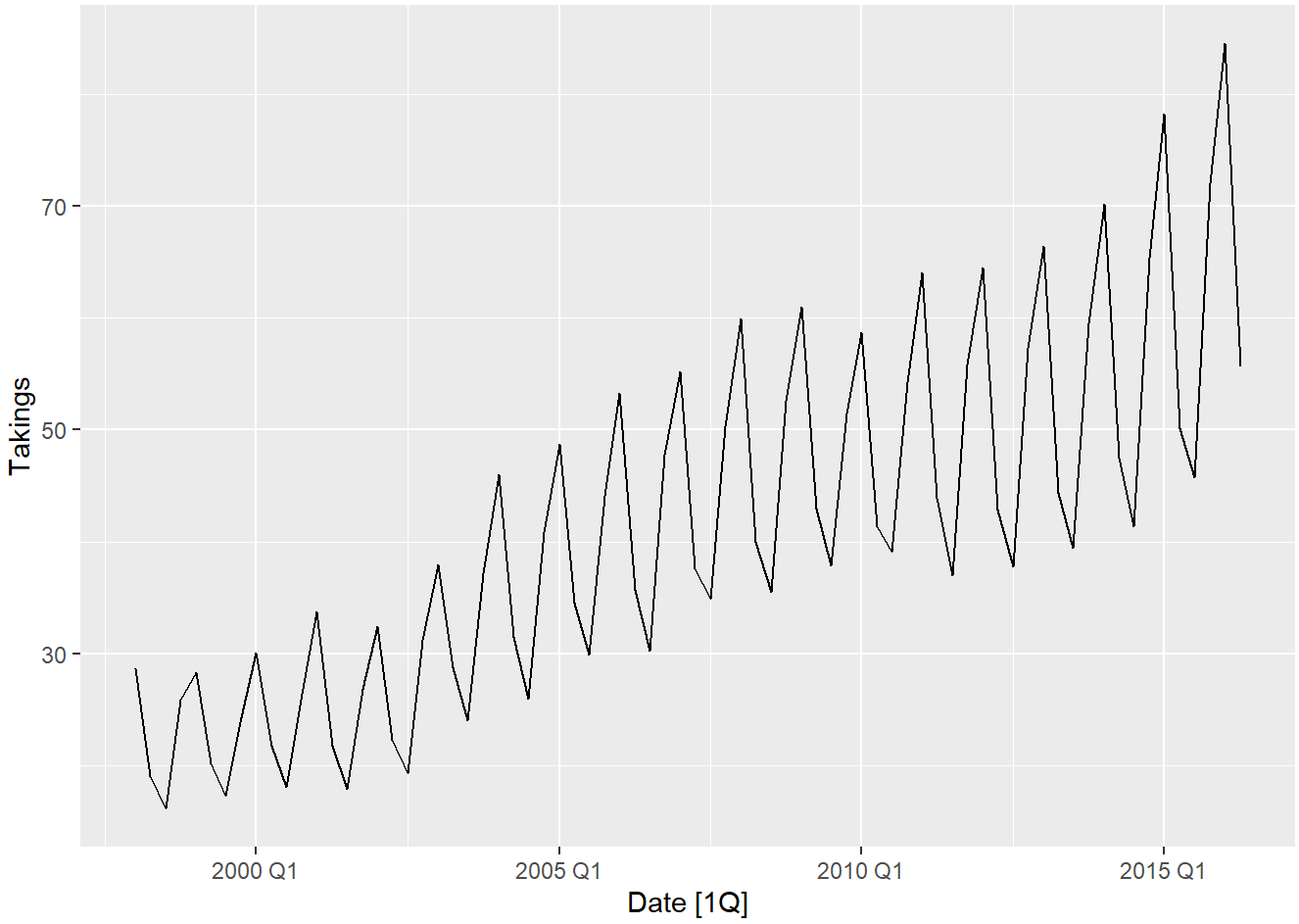
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```



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

```
tasmania <- aus_accommodation |> filter(State == 'Tasmania')
## Seasonality and trend in the data, needs to be smoothed out.
tasmania |> autoplot()
```

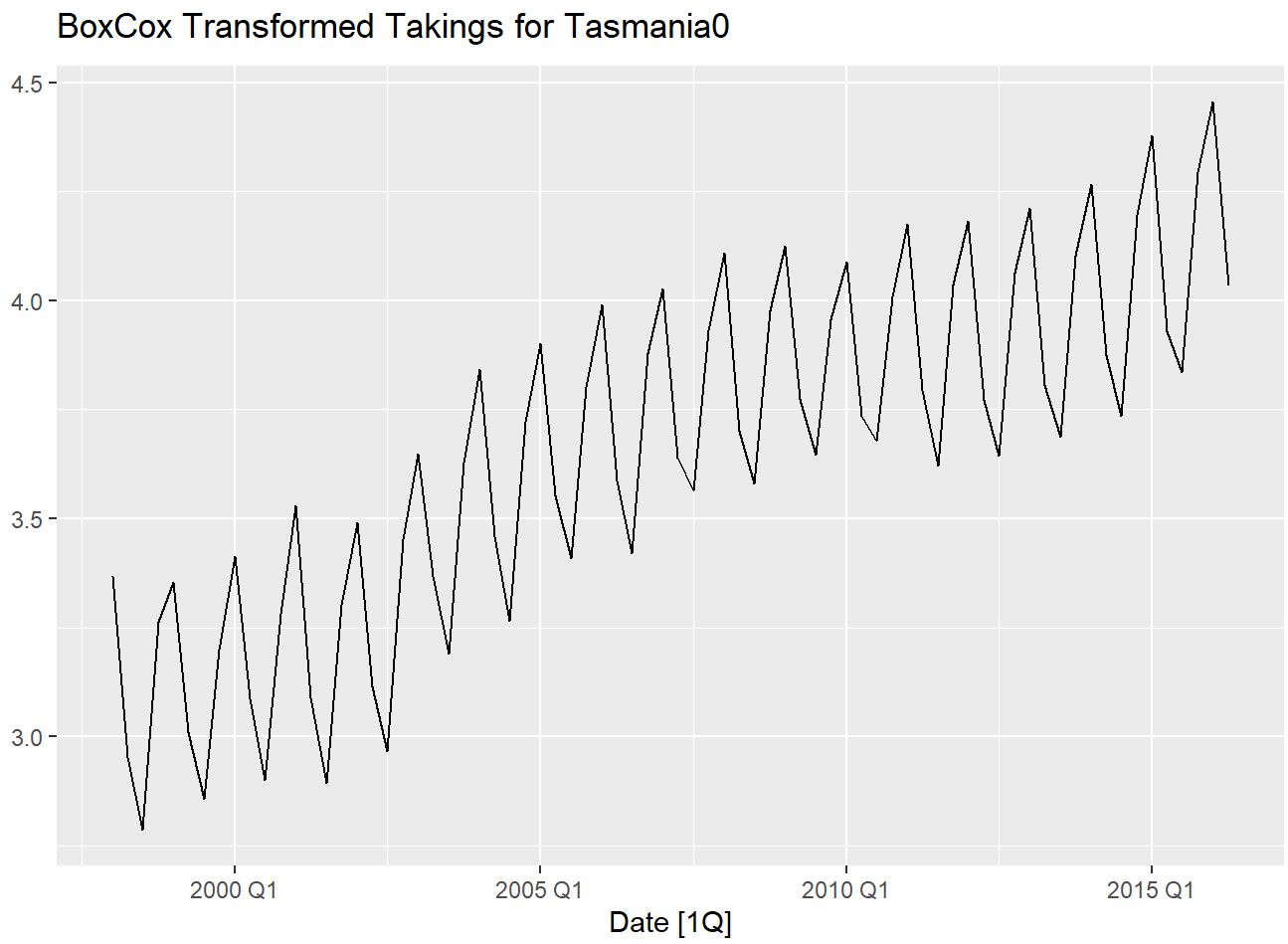
```
## Plot variable not specified, automatically selected `.vars = Takings`
```




```
## Performing the Box Cox transformation
lambda <- tasmania |>
  features(Takings, features = guerrero) |>
  pull(lambda_guerrero)
print(lambda) ## 0.0018 ## logging data is best.
```

```
## [1] 0.001819643
```

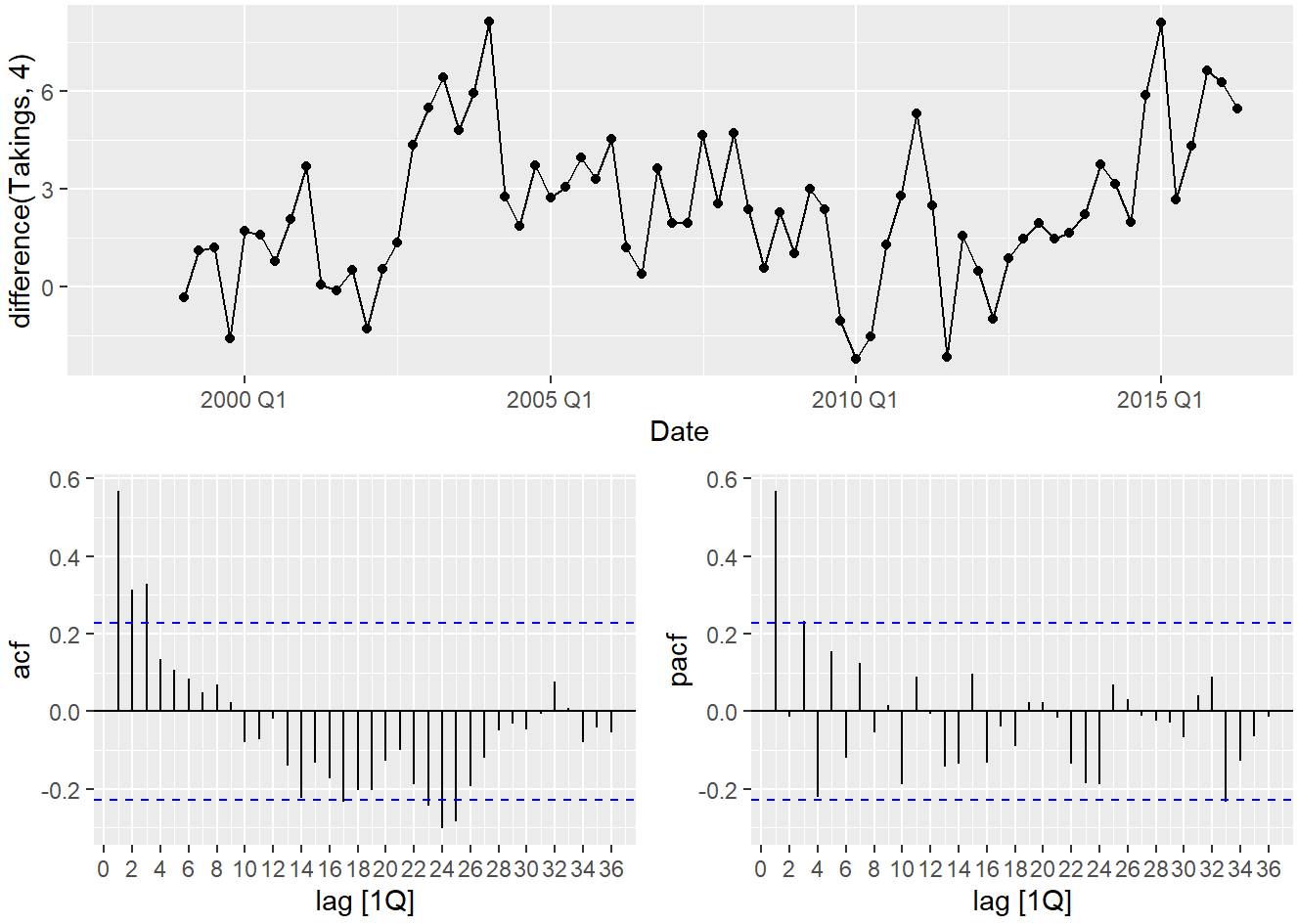
```
## Box Cox Transform successful now to find the differentiation needs
print(tasmania |>  autoplot(box_cox(Takings, lambda)) +  labs(y = "",title = latex2exp::TeX(p
aste0("BoxCox Transformed Takings for Tasmania",round(lambda,2)))))
```




```
## The seasonality is now constant(ish) and the trend looks better for differencing. In order to accommodate the seasonal nature of the data, we need to do a seasonal differencing. Doing a seasonal differencing of 4 because the data is quarterly.

print(tasmania |> gg_tsdisplay(difference(Takings,4), plot_type='partial', lag=36))
```

```
## Warning: Removed 4 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

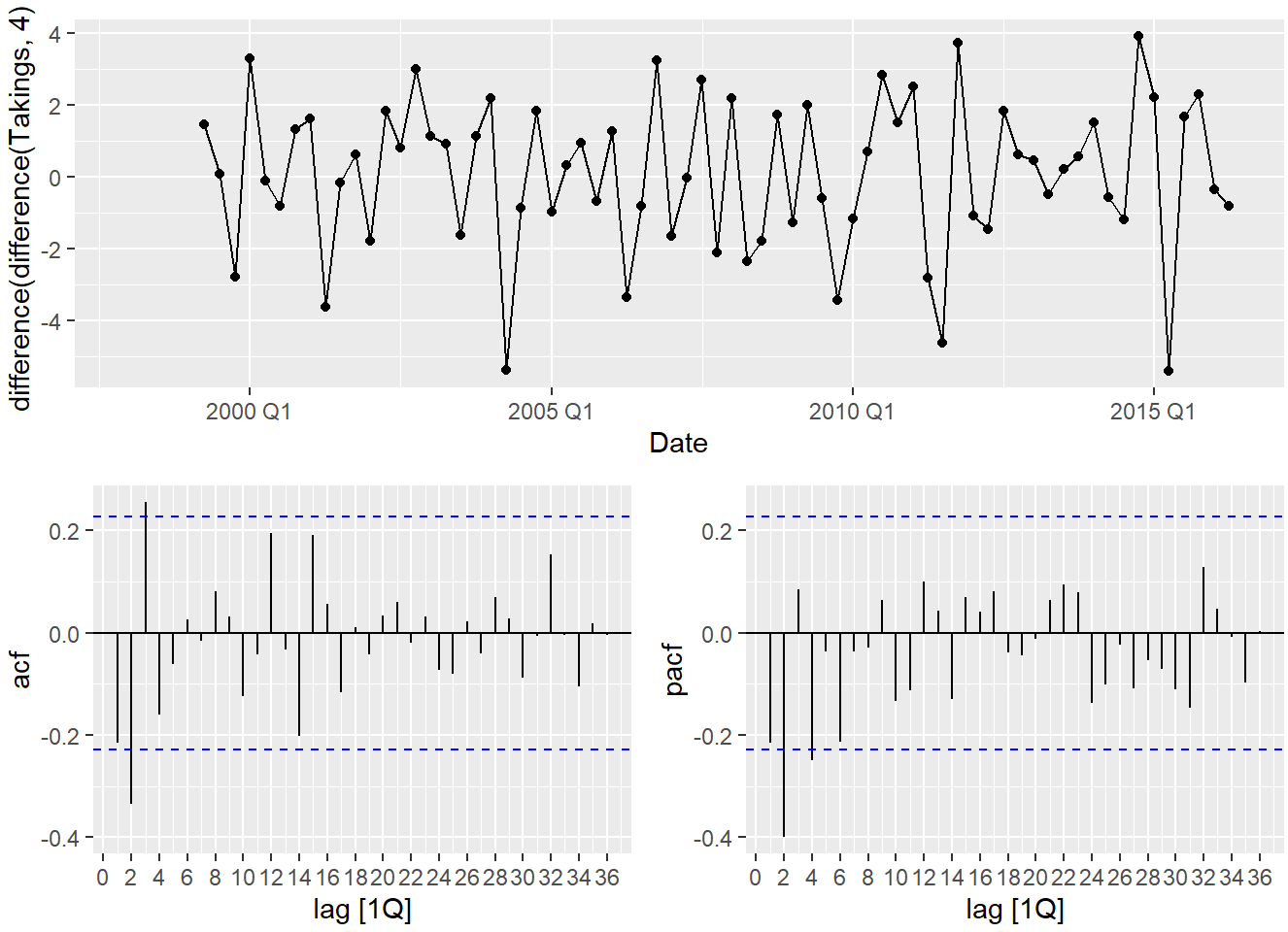
```
## Warning: Removed 4 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
## After looking at the data, it still does not seem completely stationary. The ACF has a tre
nd, the PACF seems to have a slight one as well. There are also multiple values outside of th
e 95% confidence interval boundaries, so there needs to be a second differencing here.

print(tasmania |> gg_tsdisplay(difference(Takings,4)|> difference(), plot_type='partial', lag
=36))
```

```
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 5 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

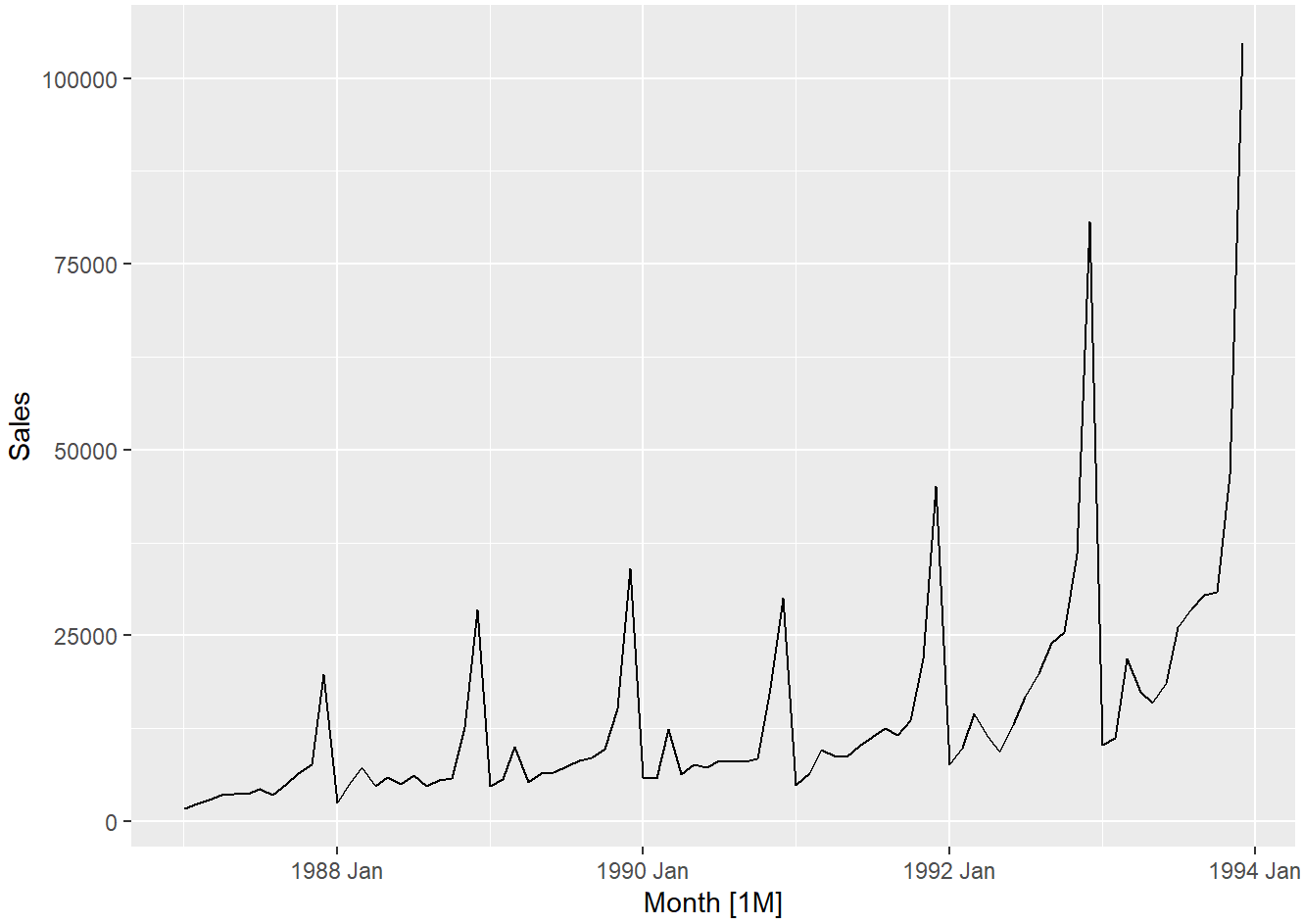



```
## The data now looks much more like white noise, and is probably stationary now.
```

c) Monthly sales from souvenirs.

```
## Taking a look at the data.
print(souvenirs |> autoplot())
```

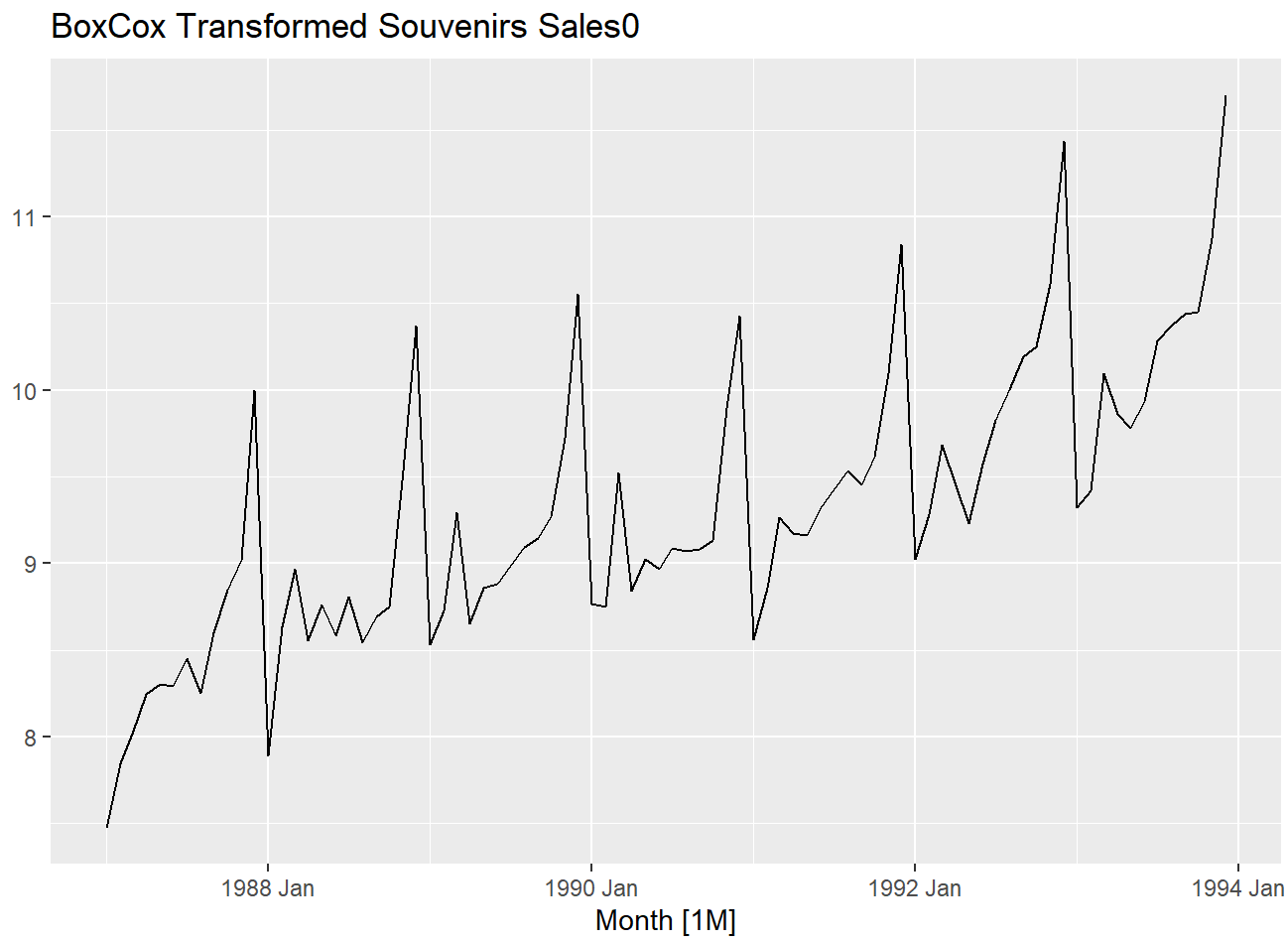
```
## Plot variable not specified, automatically selected `.vars = Sales`
```




```
## There is both a trend and a seasonal pattern here. Using Box Cox to get proper transformat
ion.
## Performing the Box Cox transformation
lambda <- souvenirs |>
  features(Sales, features = guerrero) |>
  pull(lambda_guerrero)
print(lambda) ## 0.00211 ## logging data is best.
```

```
## [1] 0.002118221
```

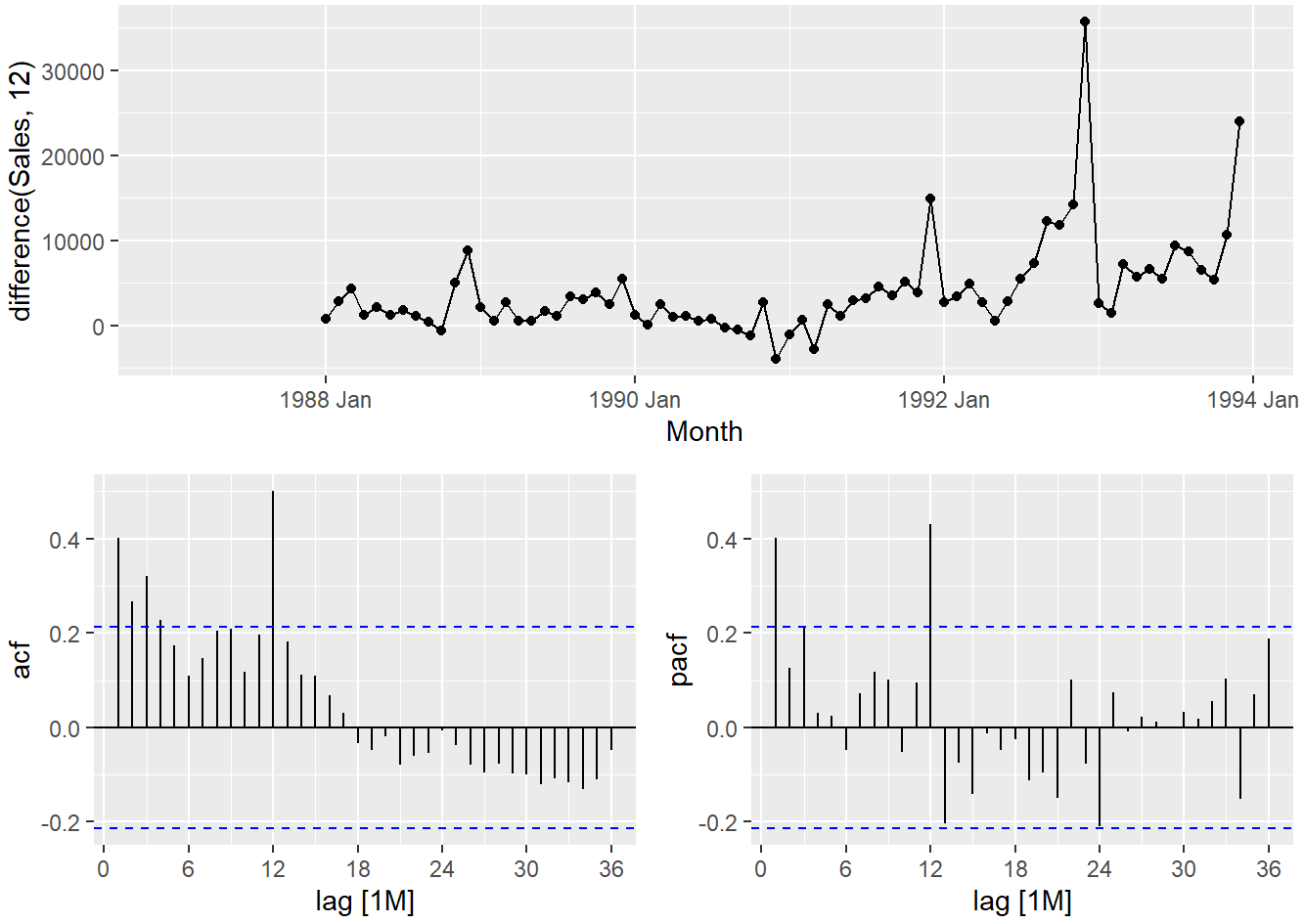
```
## Box Cox Transform successful now to find the differentiation needs
print(souvenirs |>  autoplot(box_cox(Sales, lambda)) +  labs(y = "",title = latex2exp::TeX(pa
ste0("BoxCox Transformed Souvenirs Sales",round(lambda,2)))))
```




```
## eveled out a bit, now doing differencing, monthly for seasonal. (12)
print(souvenirs |> gg_tsdisplay(difference(Sales,12), plot_type='partial', lag=36))
```

```
## Warning: Removed 12 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

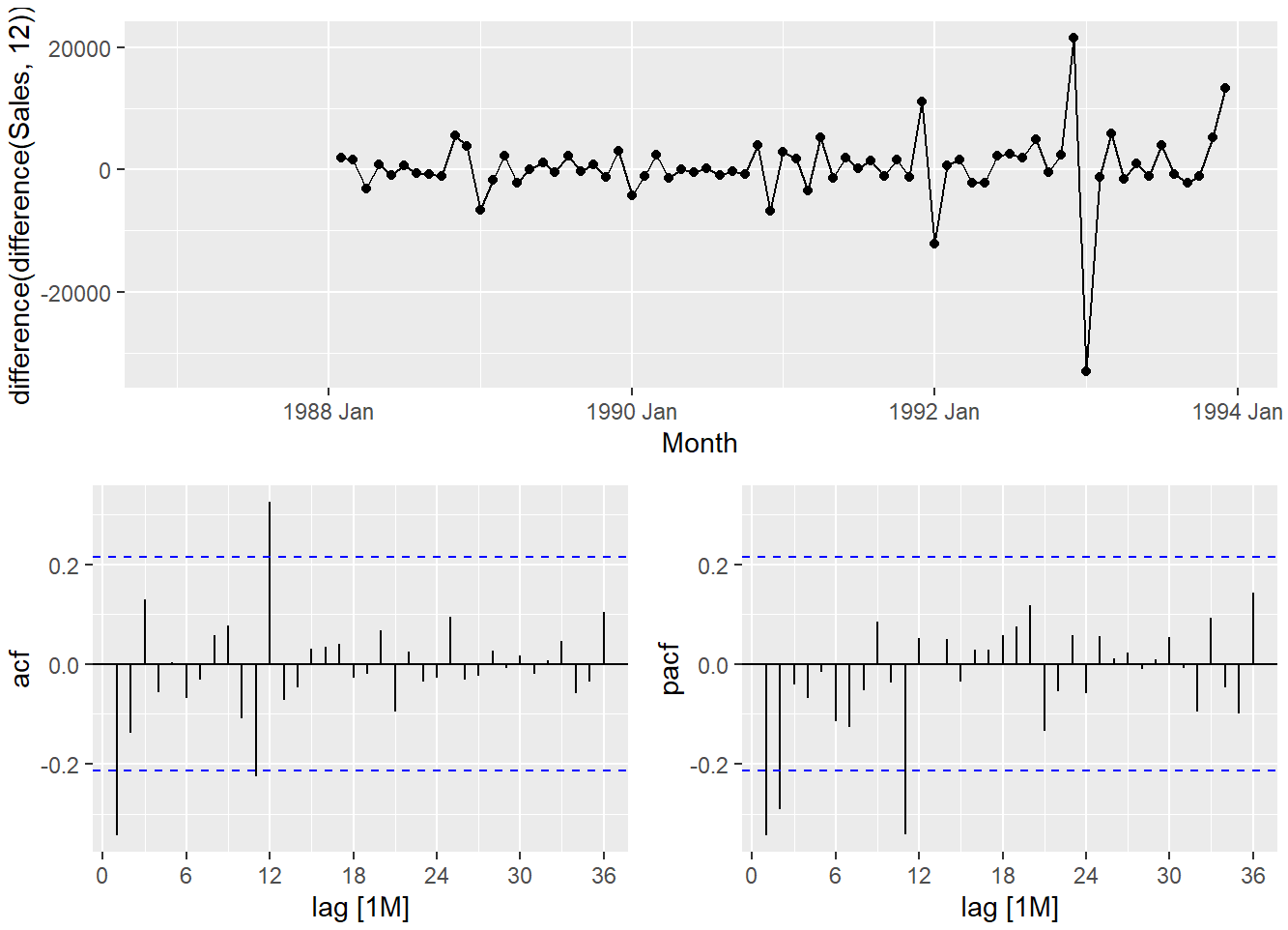
```
## Warning: Removed 12 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
## Still needs more differencing for the trend.
print(souvenirs |> gg_tsdisplay(difference(Sales,12) |> difference(), plot_type='partial', la
g=36))
```

```
## Warning: Removed 13 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 13 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

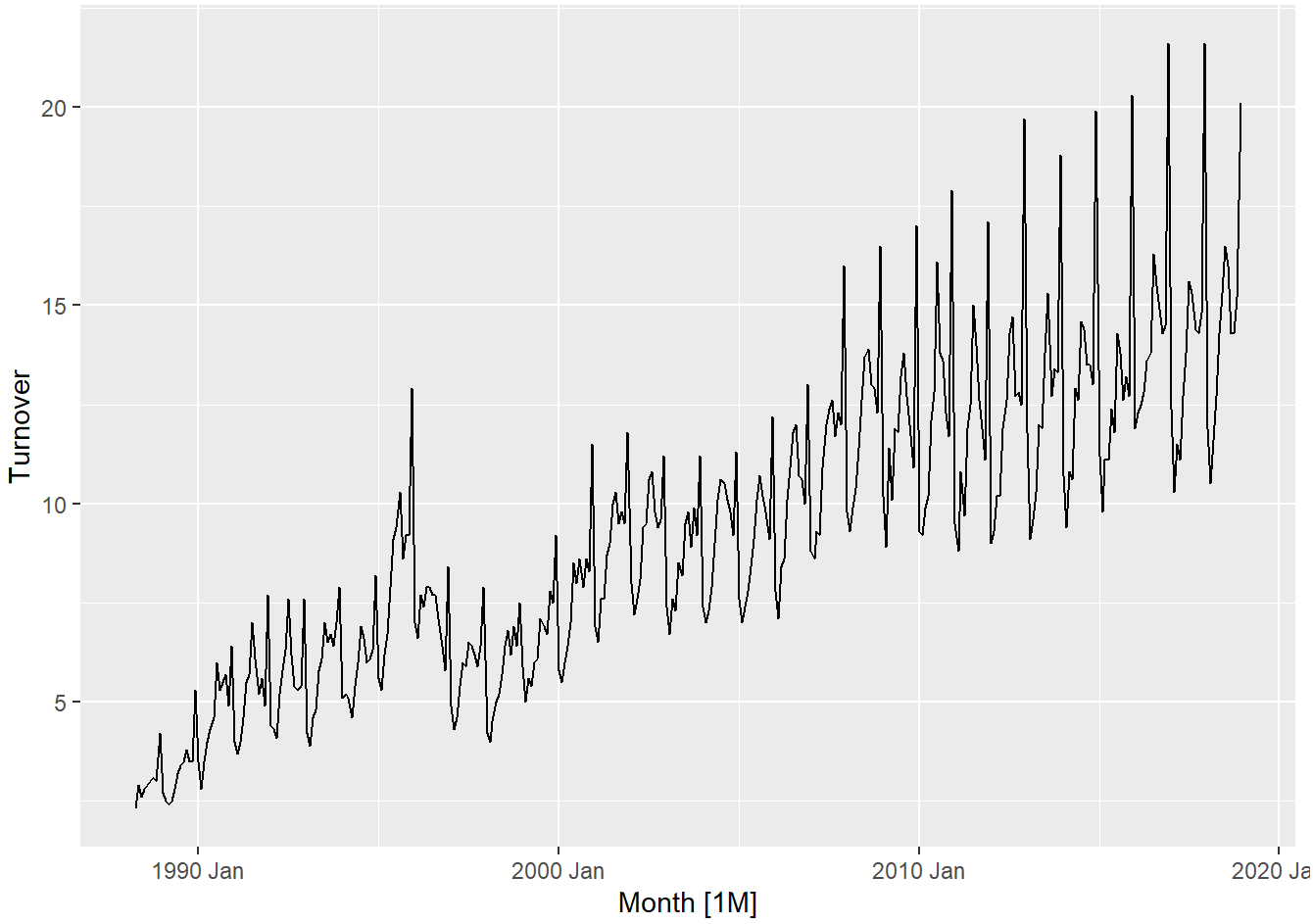



```
## Data finally looks stationary, however there are multiple instances of outliers for the AC
F and the PACF. Similar for the top chart too.
```

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.

```
## CODE FROM CHAPTER 2 Number 7.
set.seed(12345678)
myseries <- aus_retail |>
  filter(`Series ID` == sample(aus_retail$`Series ID`,1))

## Taking a look at the data. There is both a trend and a seasonality in the data. The season
ality is getting more extreme over time, as it seems. These shifts arent constant.
myseries |> autoplot()
```

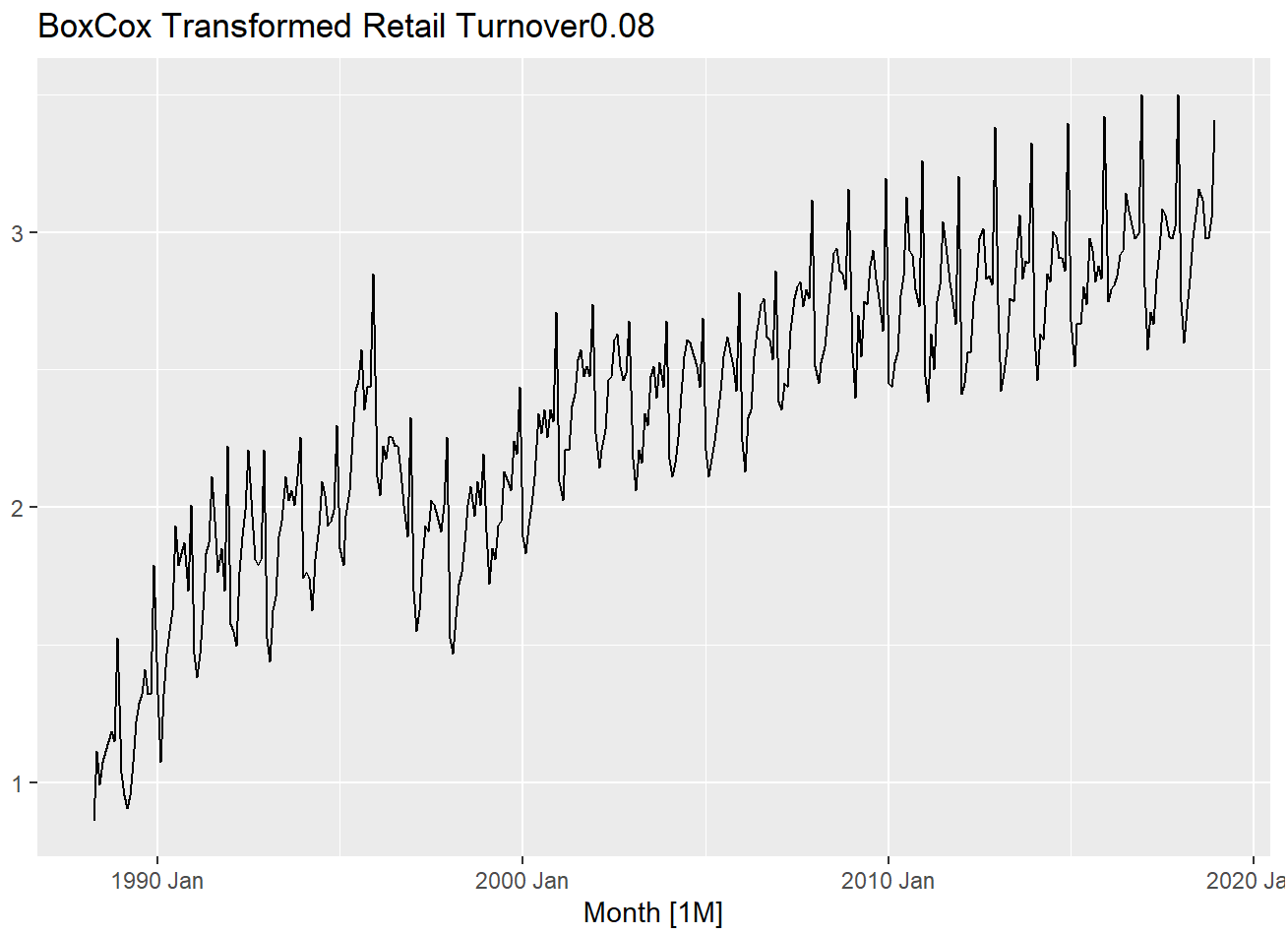
```
## Plot variable not specified, automatically selected `.vars = Turnover`
```




```
### using Box Cox to get ideal transform value.
lambda <- myseries |>
  features(Turnover, features = guerrero) |>
  pull(lambda_guerrero)
print(lambda) ## 0.083 ## inbetween sq root and now transform.
```

```
## [1] 0.08303631
```

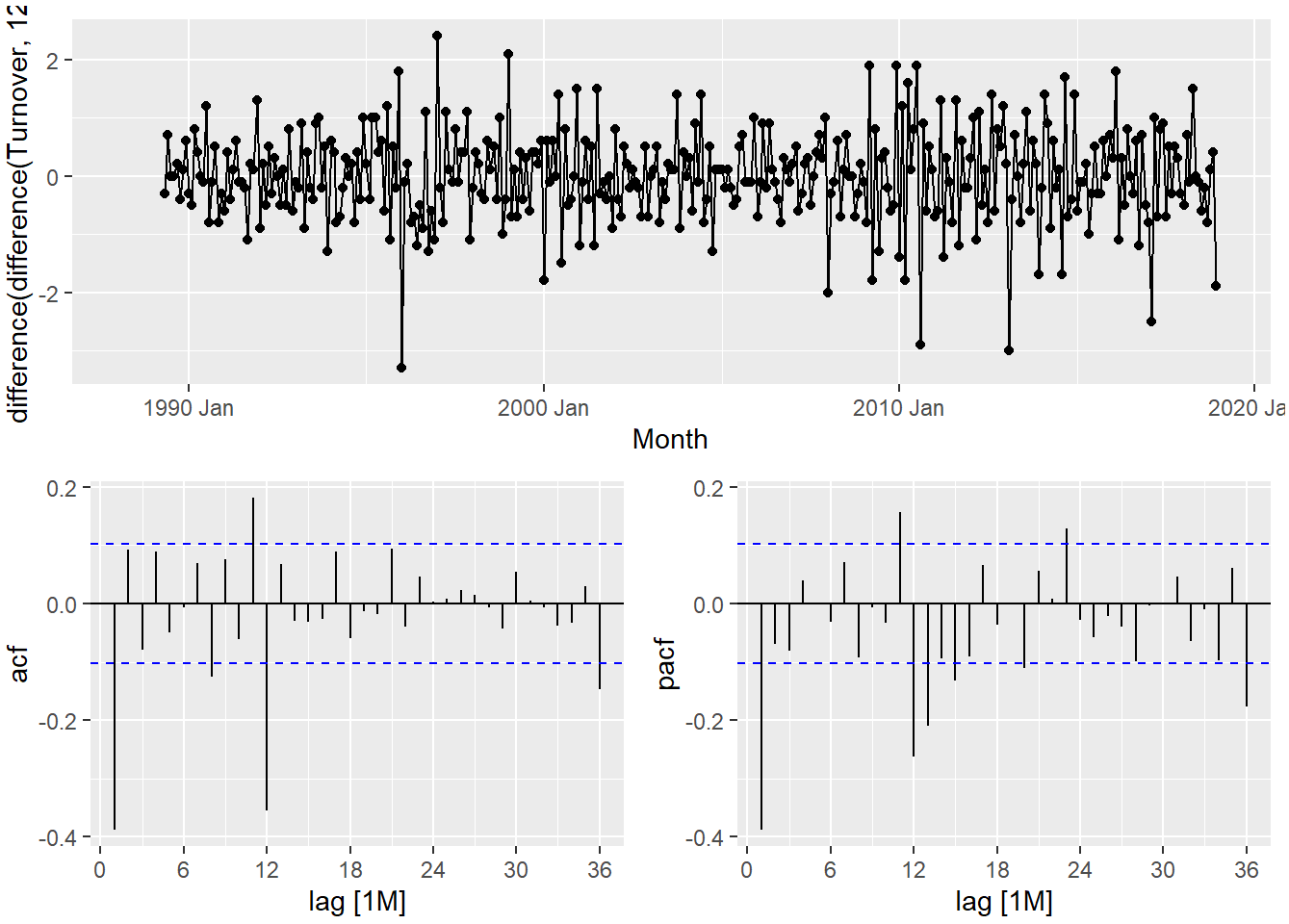
```
## Looks better.
print(myseries |>  autoplot(box_cox(Turnover, lambda)) +  labs(y = "",title = latex2exp::TeX
(paste0("BoxCox Transformed Retail Turnover",round(lambda,2)))))
```




```
## Starting Differencing. Will need 2 levels probably becuase of season and trend. Did with j
ust one & its not a enough, so jumping to 2
print(myseries |> gg_tsdisplay(difference(Turnover,12) |> difference(), plot_type='partial',
lag=36))
```

```
## Warning: Removed 13 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 13 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



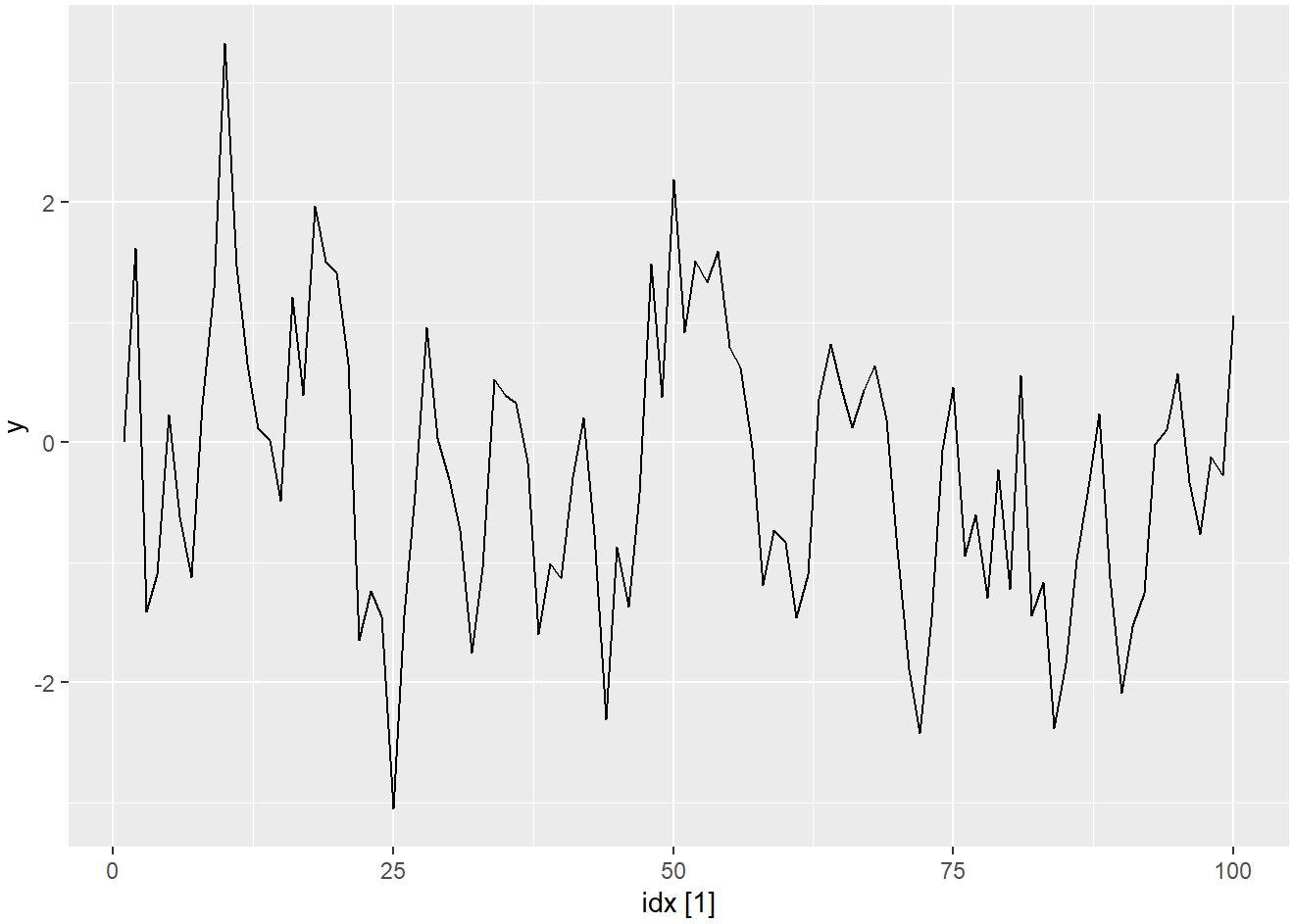

```
## After one transformation there was still a noticable trend in the data, after the second o
ne it seems to be stationary / show white noise.
```

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

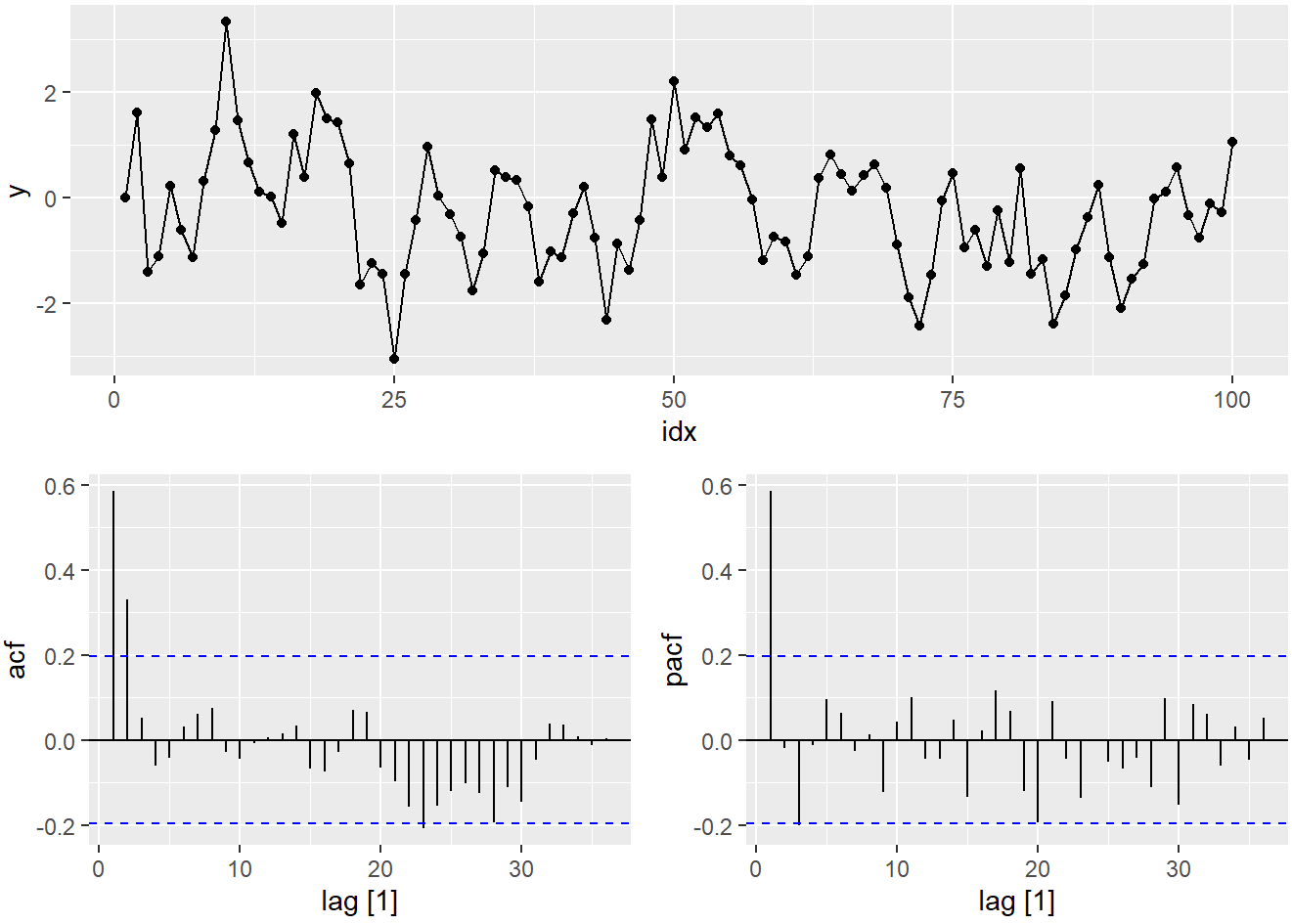
a)Use the following R code to generate data from an AR(1) model with ϕ1 = 0.6 and σ2=1. The process starts with y1=0.

```
set.seed(12345678)
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)

## Looking at the random simulation.
print(sim |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



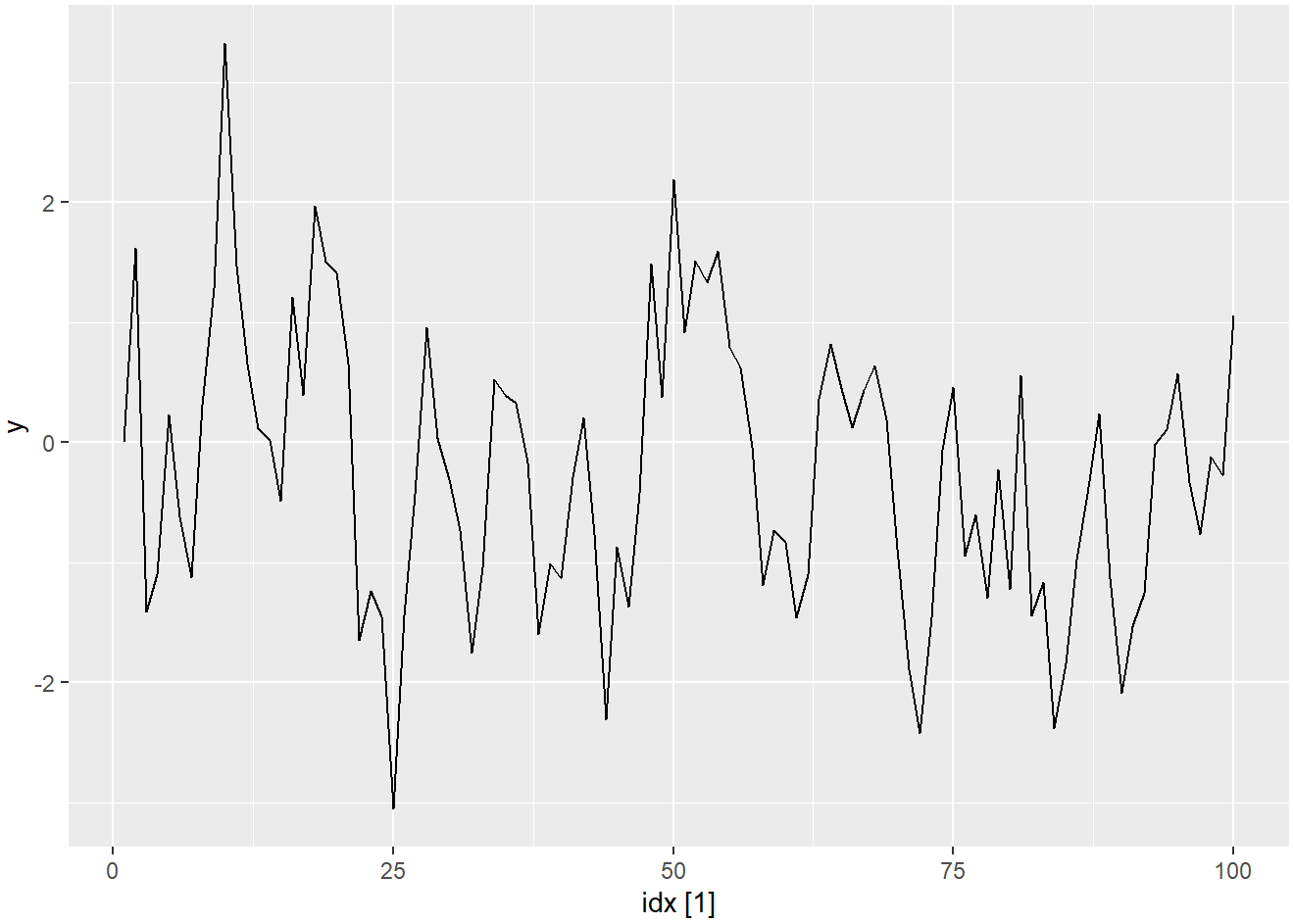

```
## No trend, just variations. No real cyclical / seasonal shifts, bit there is variation. Loo
king at the ACF and PACF to confirm AR(1). There is a gradual decay in the acf which is inlin
e an ar(1) model, but there seems to be a pattern in the 20 - 30 range, but because of the sm
all random sample size it is probably just chance. Then in the PACF there is one main spike a
t the beginning then only smaller values.
print(sim |> gg_tsdisplay(y, plot_type='partial', lag=36))
```



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

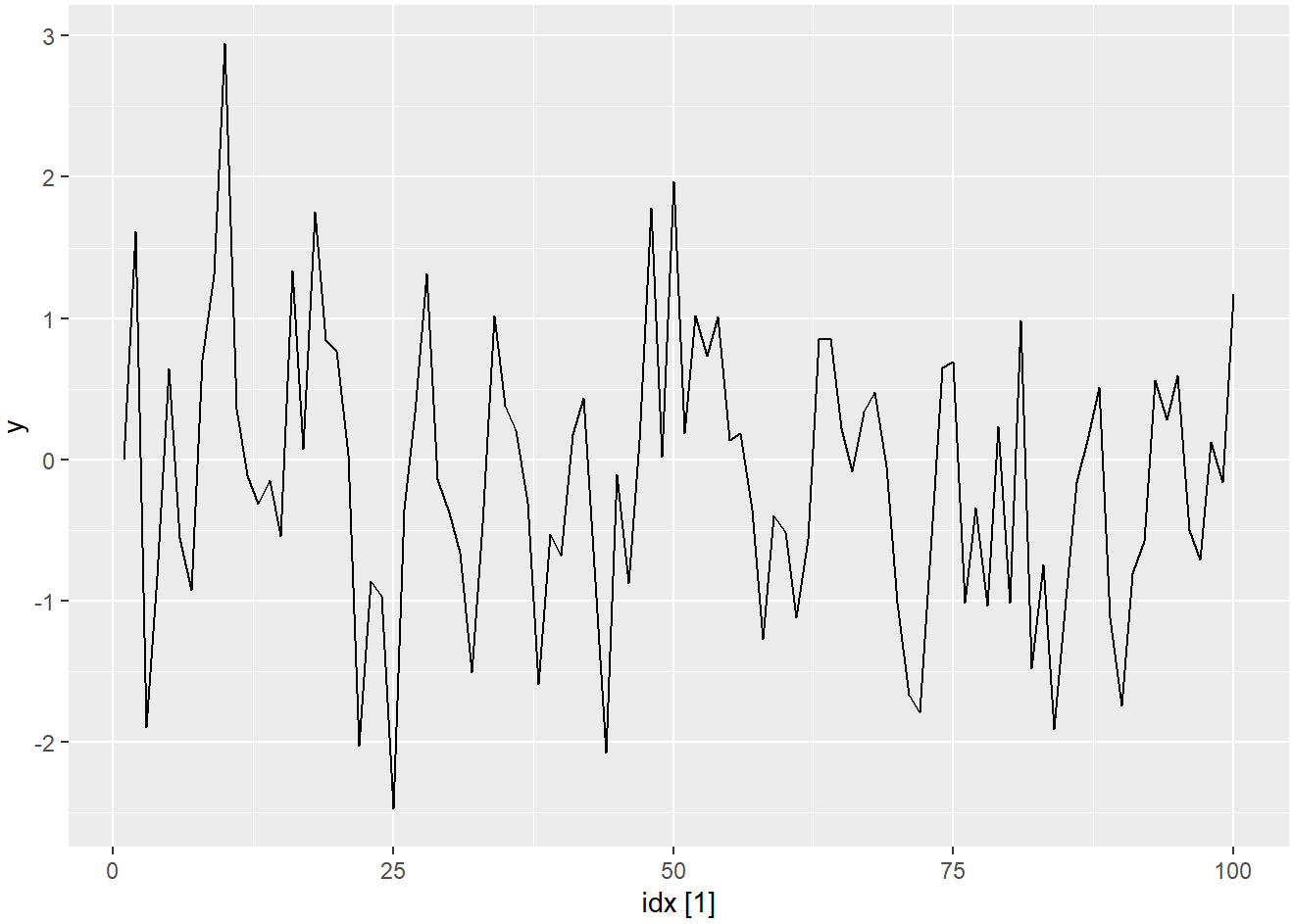
```
## Firstly Doing autoplot of the original sim
print(sim |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



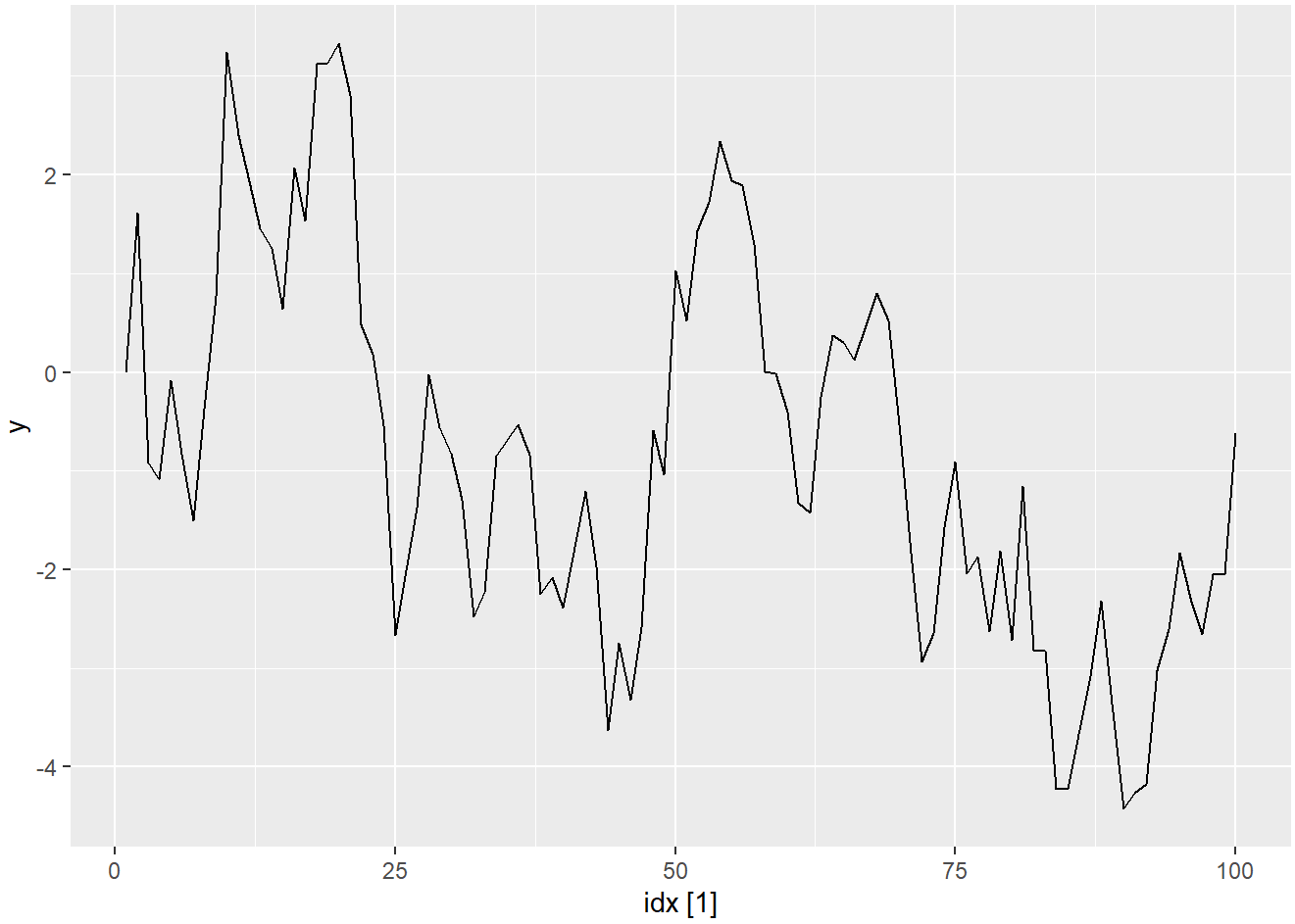

```
### Running again with a different coefficients#
y2 <- numeric(100)
for(i in 2:100)
  y2[i] <- 0.3*y2[i-1] + e[i]
sim2 <- tsibble(idx = seq_len(100), y = y2, index = idx)

## With 0.3
print(sim2 |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



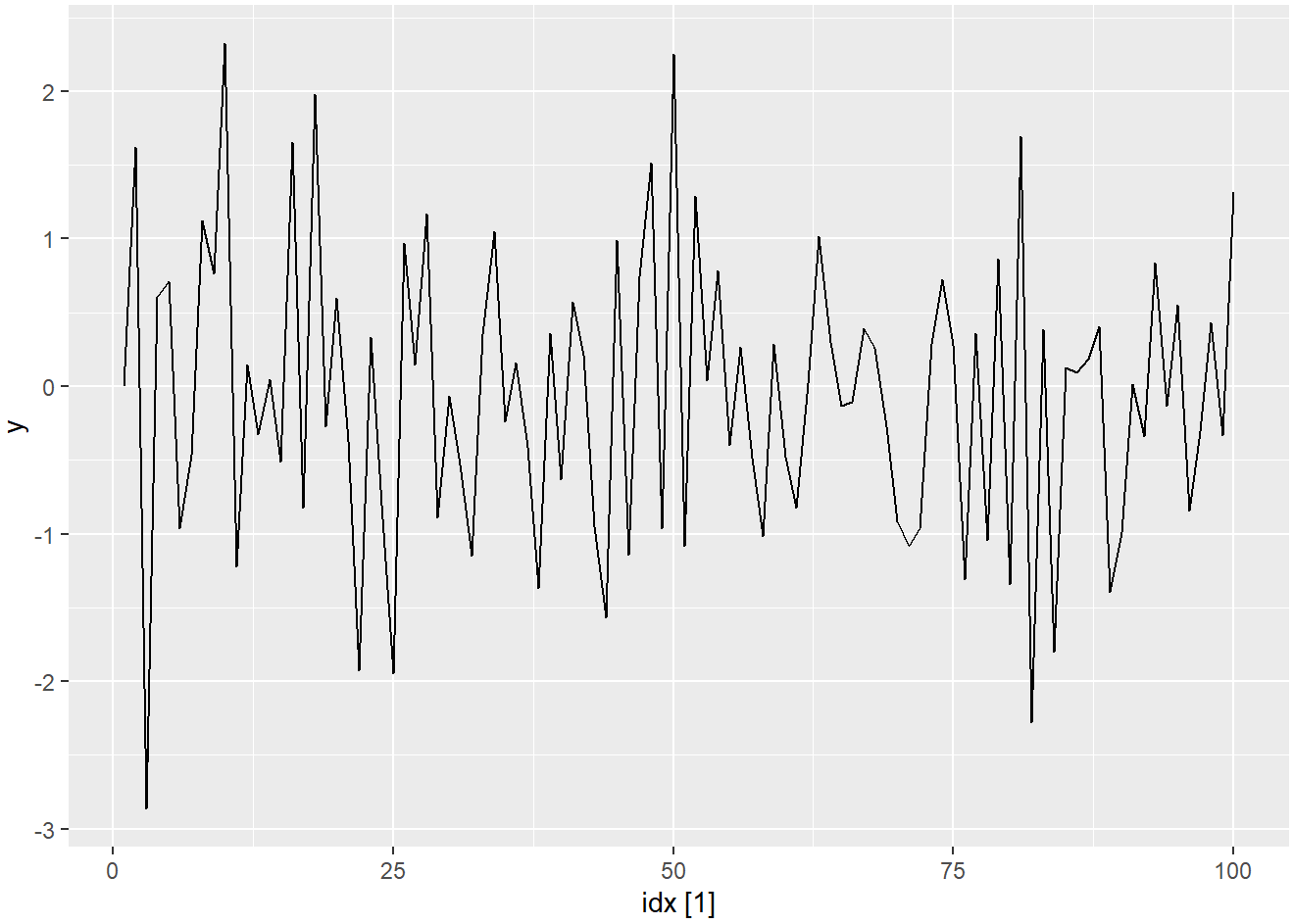

```
## Running agina with different coef.
y3 <- numeric(100)
for(i in 2:100)
  y3[i] <- 0.9*y3[i-1] + e[i]
sim3 <- tsibble(idx = seq_len(100), y = y3, index = idx)

print(sim3 |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



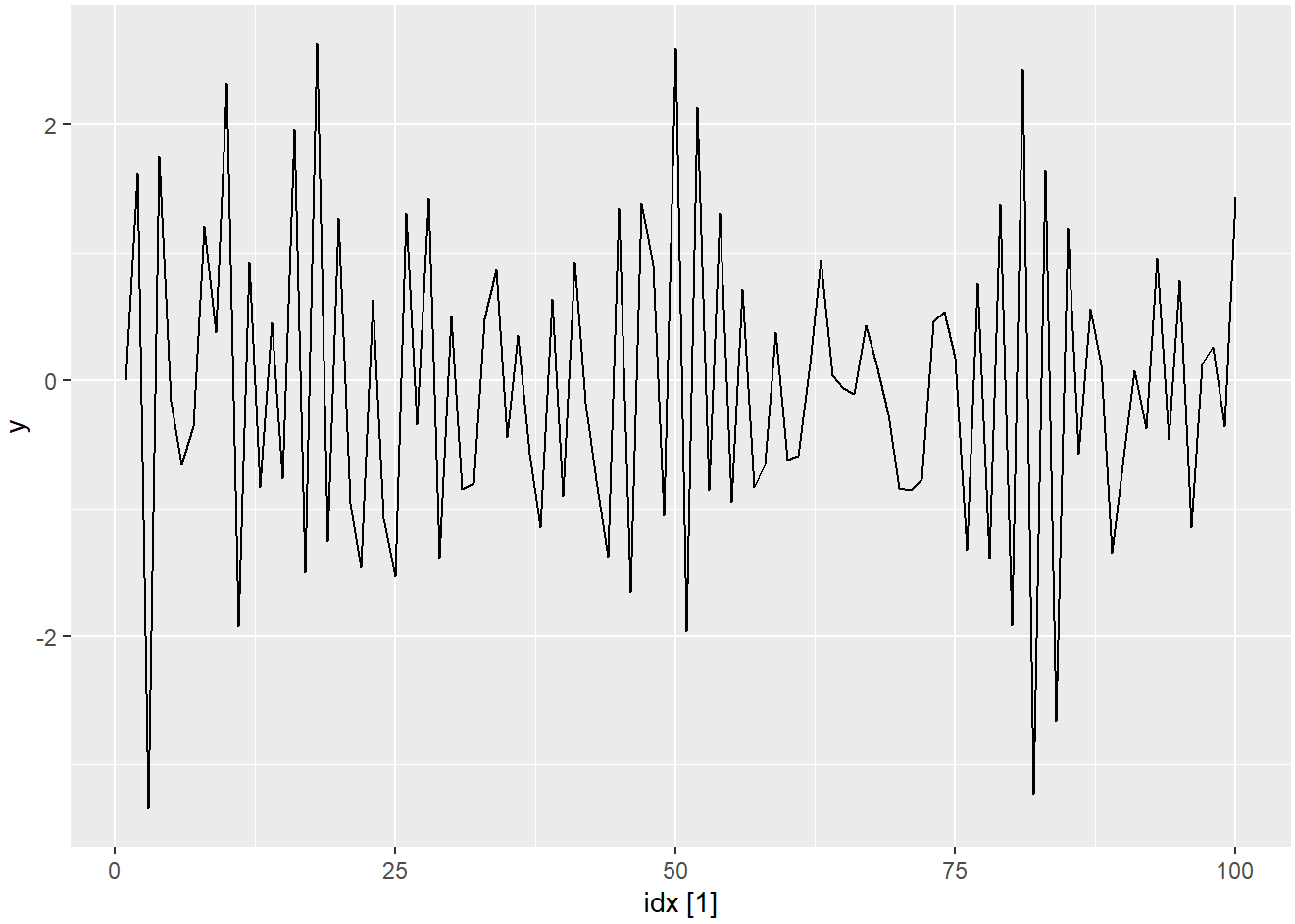

```
### Running again with a different coefficients#
y2_neg <- numeric(100)
for(i in 2:100)
  y2_neg[i] <- -0.3*y2_neg[i-1] + e[i]
sim2_neg <- tsibble(idx = seq_len(100), y = y2_neg, index = idx)

## With -0.3
print(sim2_neg |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



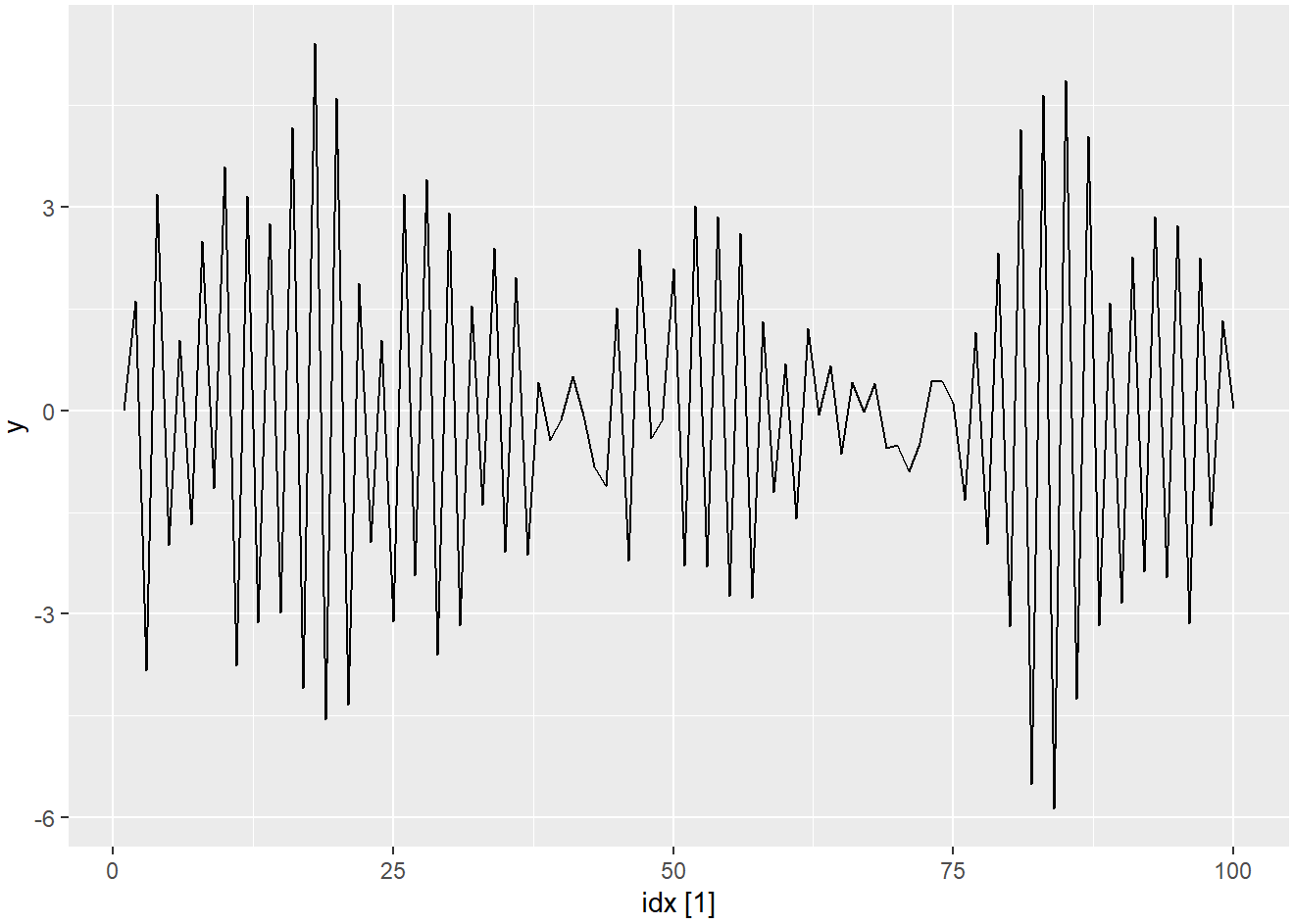

```
## Running again with a different coefficients#
y_neg <- numeric(100)
for(i in 2:100)
  y_neg[i] <- -0.6*y_neg[i-1] + e[i]
sim_neg <- tsibble(idx = seq_len(100), y = y_neg, index = idx)

print(sim_neg |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```




```
## Running agina with different coef.
y3_neg <- numeric(100)
for(i in 2:100)
  y3_neg[i] <- -0.9*y3_neg[i-1] + e[i]
sim3_neg <- tsibble(idx = seq_len(100), y = y3_neg, index = idx)

print(sim3_neg |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```

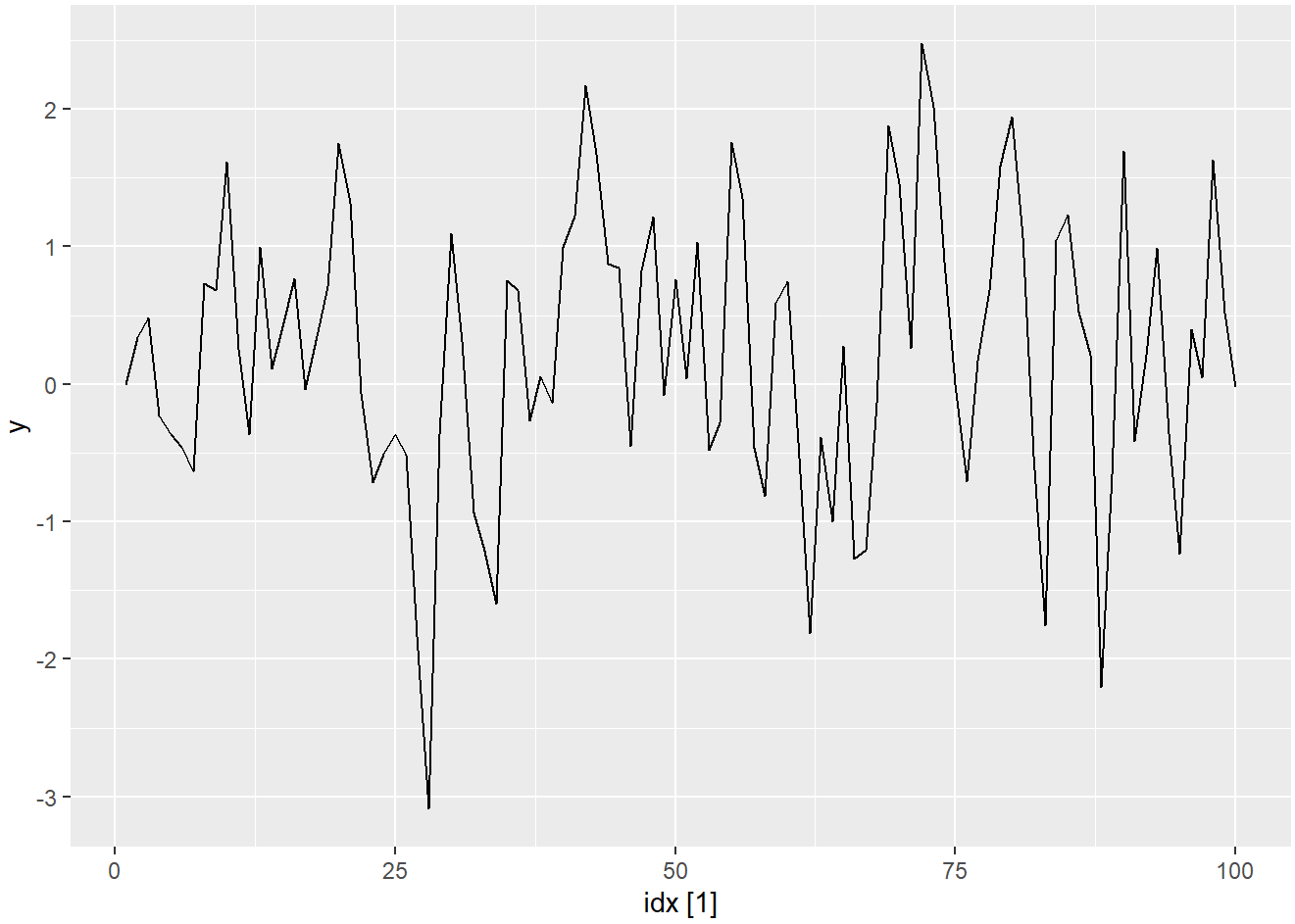



```
## The lower the coefficient (when positive) the variation seems to be a bit more intense, wh
en the positive coefficients get larger the variation in the data seems to be slightly less e
xaggerated. When dealing with negative coefficients, the closer to negative one that the coef
ficient is the more frequency and exaggerated nature of the variations in the data increases.
```

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

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

sim_ma |> autoplot()
```

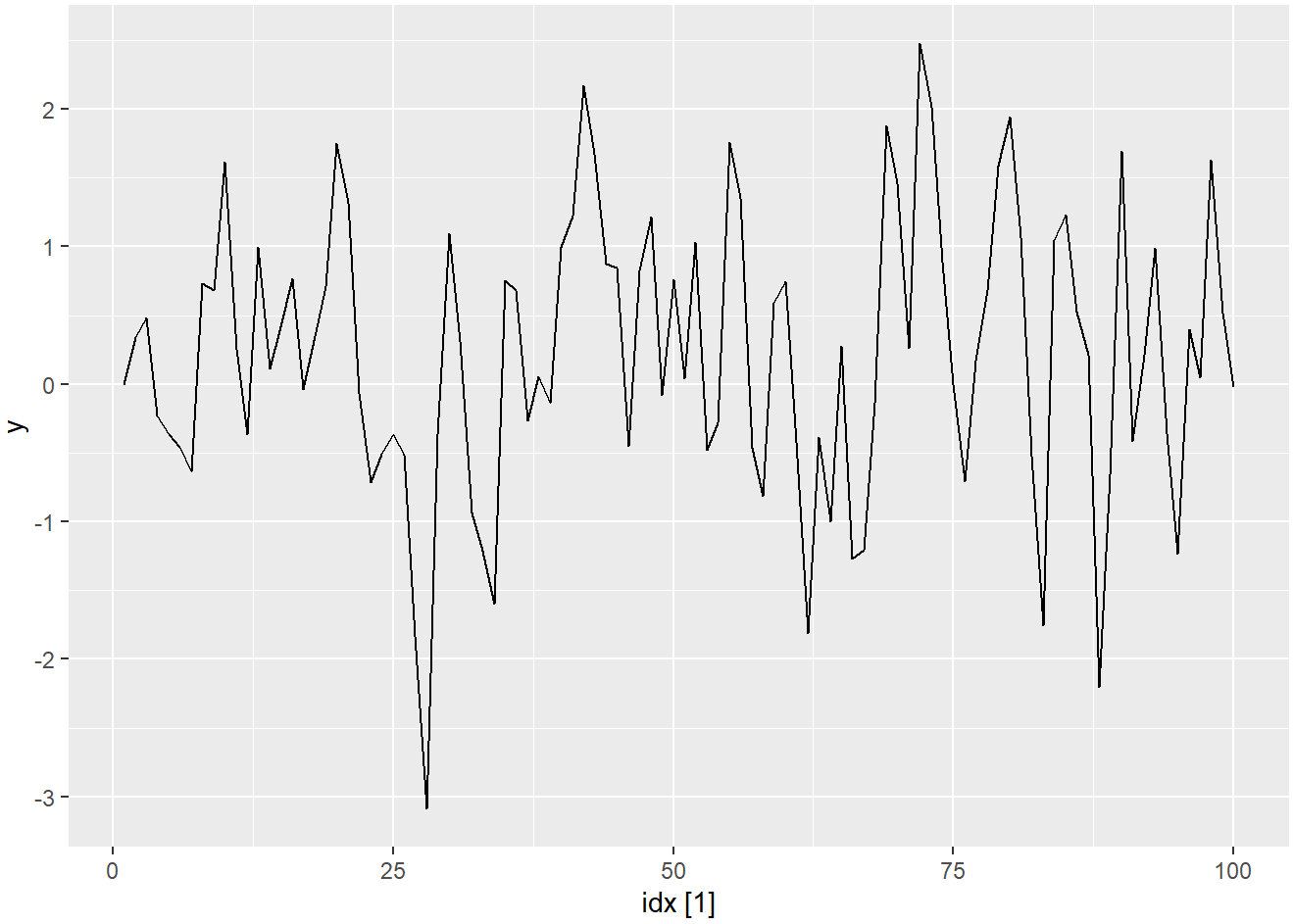
```
## Plot variable not specified, automatically selected `.vars = y`
```



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

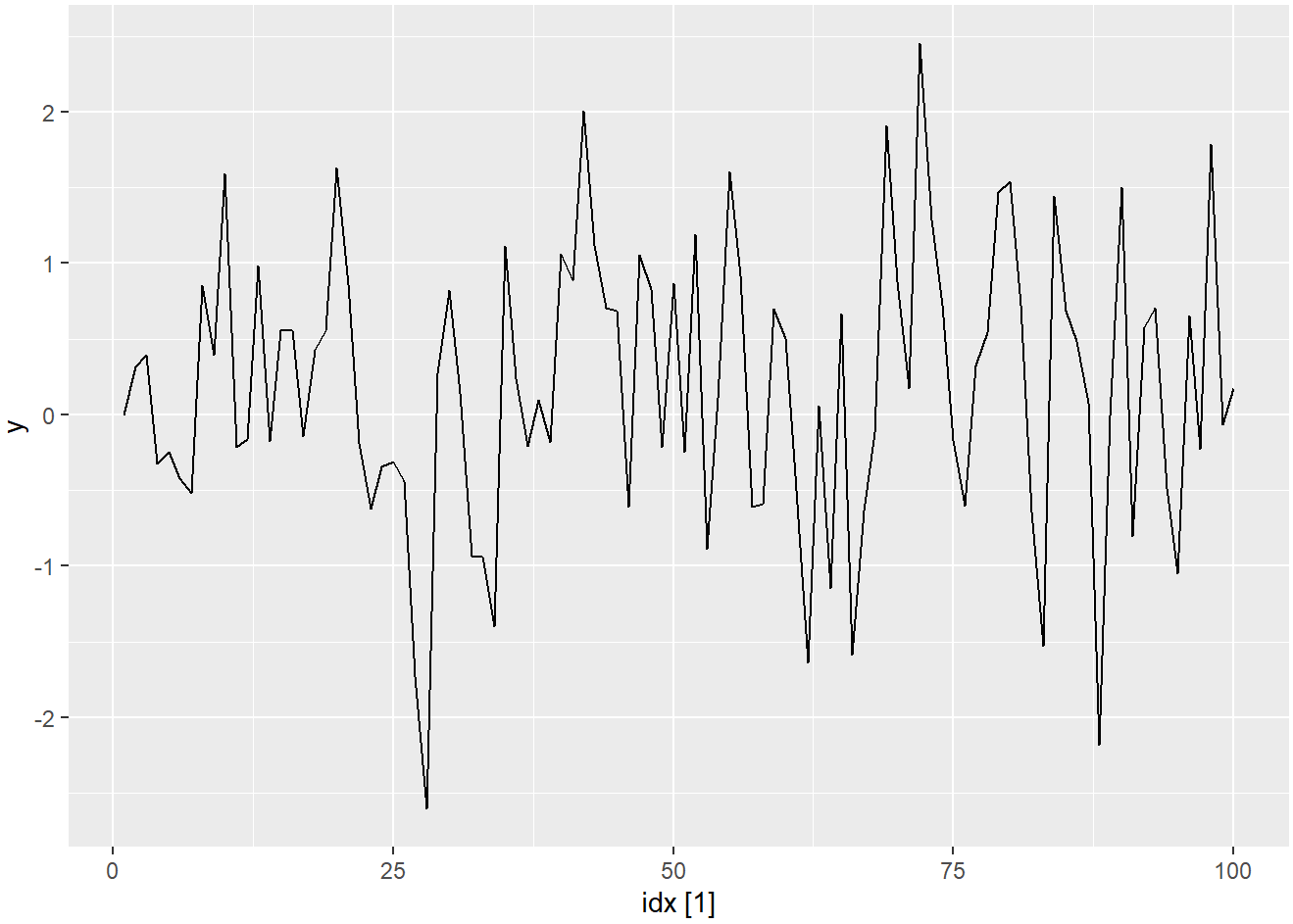
```
# Doing -0.3, -0.6, -0.9, 0.3, 0.6, 0.9
#Original
print(sim_ma |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



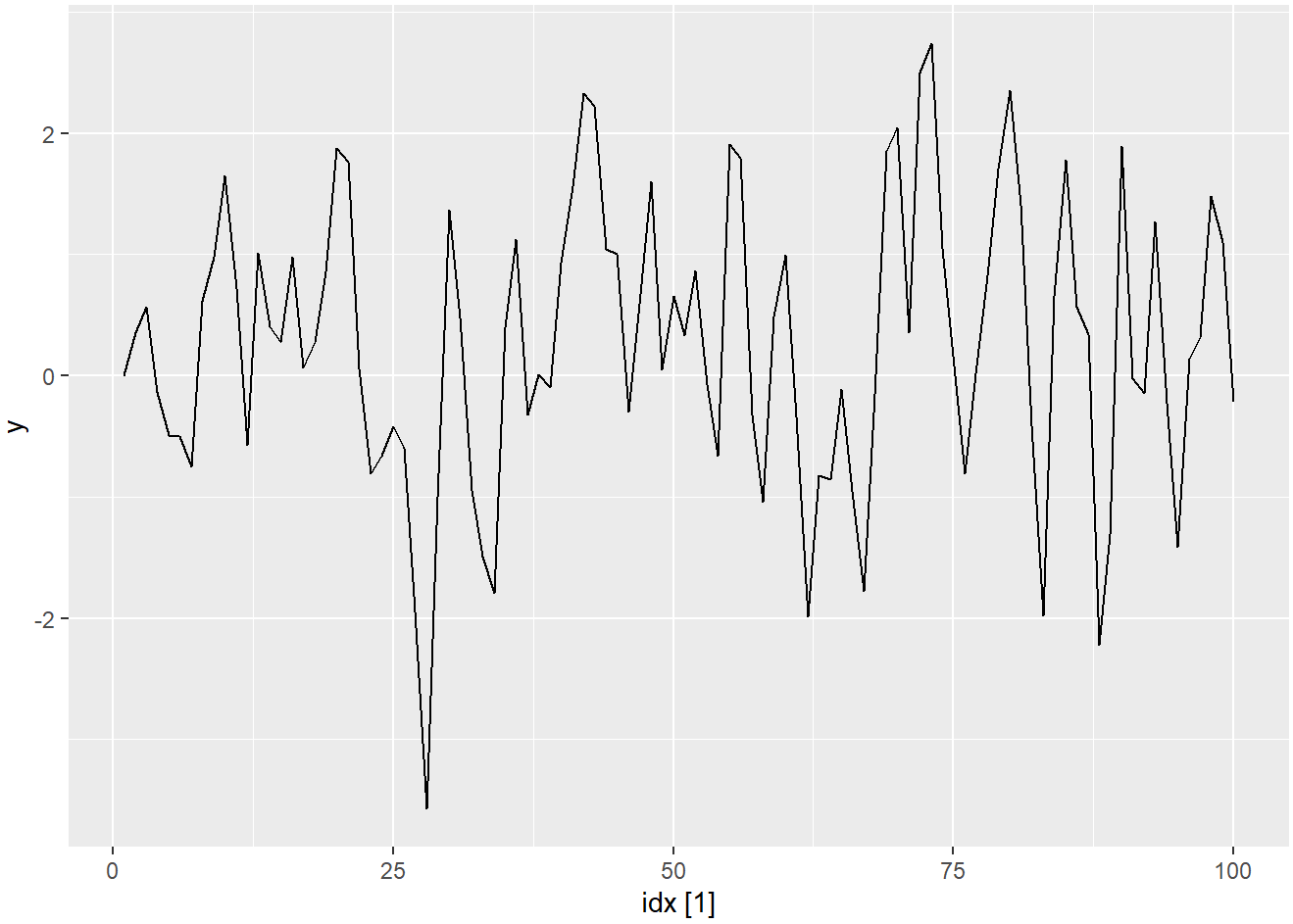

```
### Running again with a different coefficients
yma_2 <- numeric(100)
for(i in 2:100)
  yma_2[i] <- e[i]+0.3*e[i-1]
simma_2 <- tsibble(idx = seq_len(100), y = yma_2, index = idx)

## With 0.3
print(simma_2 |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



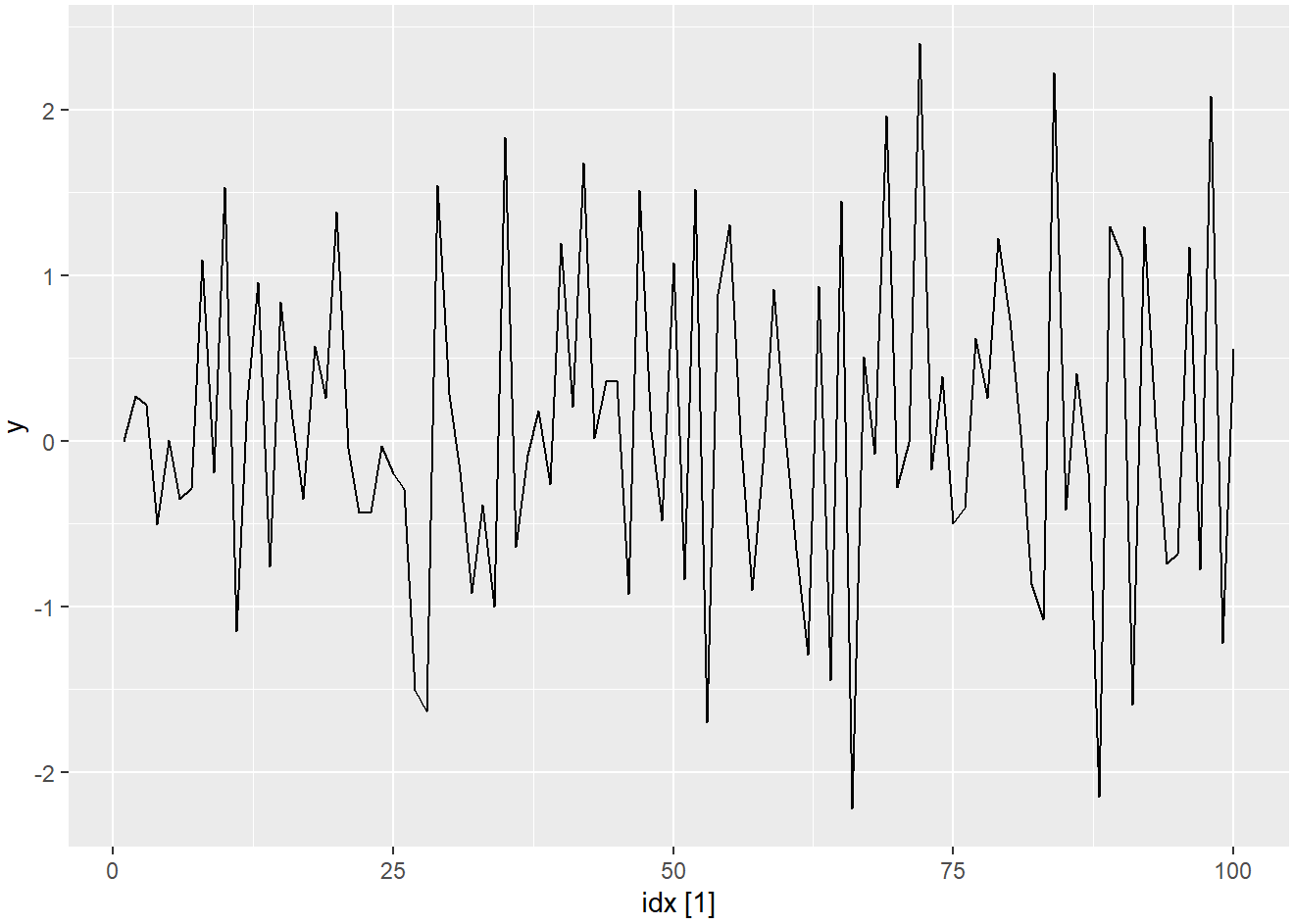

```
# With 0.9
yma_3 <- numeric(100)
for(i in 2:100)
  yma_3[i] <- e[i]+0.9*e[i-1]
simma_3 <- tsibble(idx = seq_len(100), y = yma_3, index = idx)

print(simma_3 |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



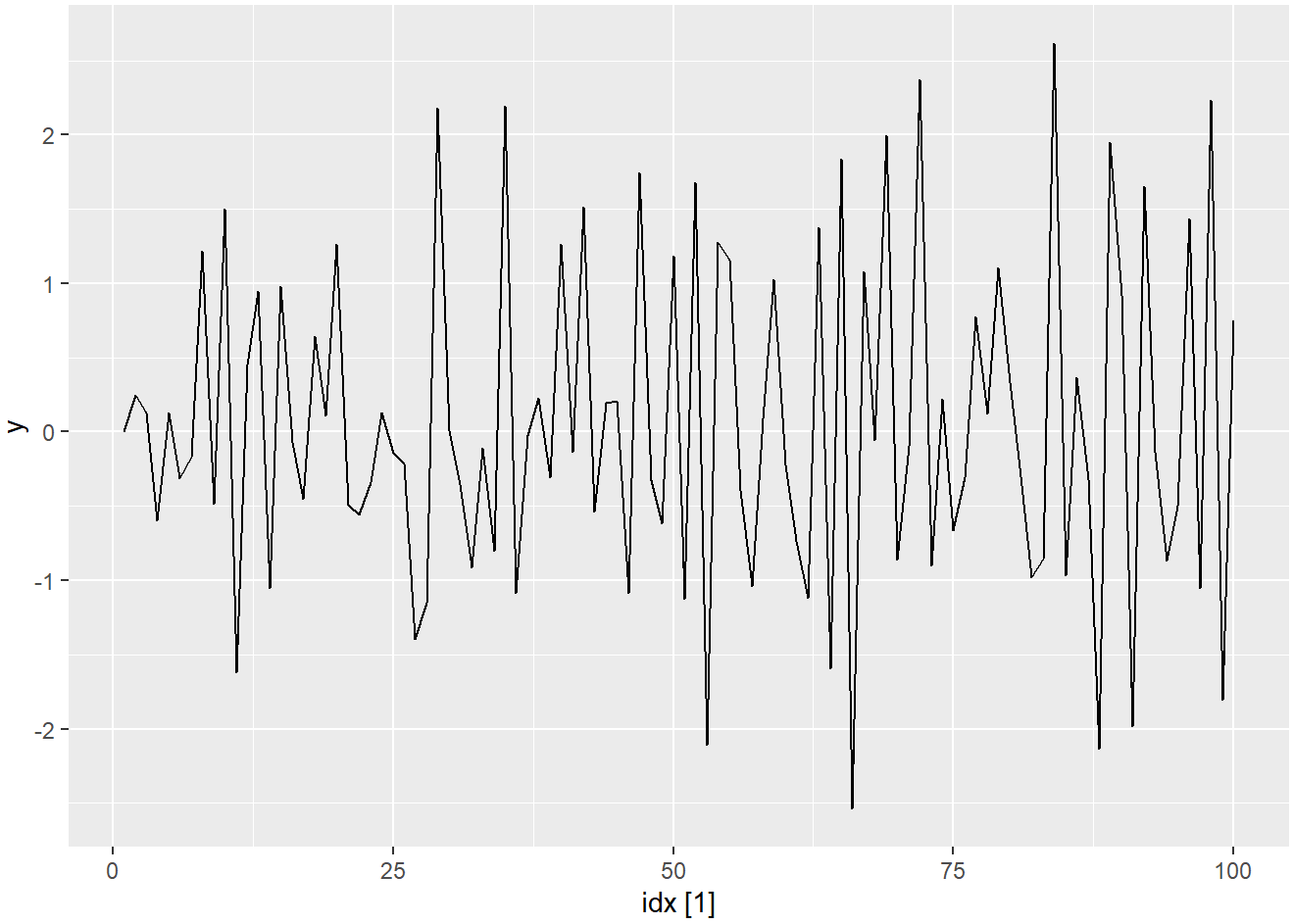

```
### With -0.3
yma_2_neg <- numeric(100)
for(i in 2:100)
  yma_2_neg[i] <- e[i]-0.3*e[i-1]
simma2_neg <- tsibble(idx = seq_len(100), y = yma_2_neg, index = idx)

print(simma2_neg |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



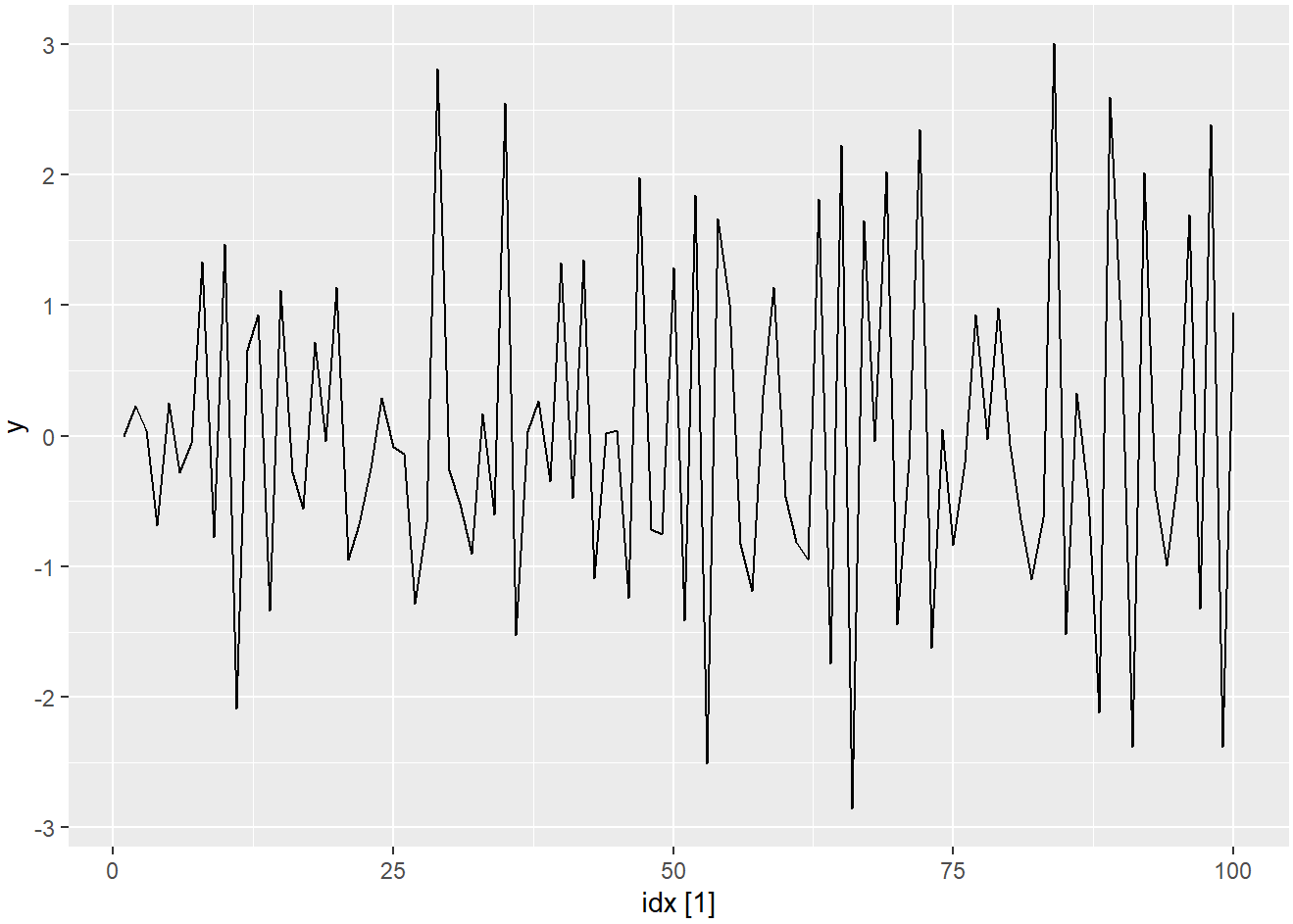

```
## -0.6
yma_neg <- numeric(100)
for(i in 2:100)
  yma_neg[i] <- e[i]-0.6*e[i-1]
simma_neg <- tsibble(idx = seq_len(100), y = yma_neg, index = idx)

print(simma_neg |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```




```
## With -0.9.
yma3_neg <- numeric(100)
for(i in 2:100)
  yma3_neg[i] <- e[i]-0.9*e[i-1]
simma3_neg <- tsibble(idx = seq_len(100), y = yma3_neg, index = idx)

print(simma3_neg |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```




```
# There really isn't much variation with the shift in moving average model coefficients, ther
e arent any big changes in the data.
```

e) Generate data from an ARMA(1,1) model with ϕ1=0.6, θ1=0.6 and σ2=1.

```
#Generating new ARMA model.
set.seed(123)
y_arma <- numeric(100)
e_arma <- rnorm(100, sd=1) #σ2=1
for(i in 2:100)
  y_arma[i] <- 0.6*y_arma[i-1] + e_arma[i] + 0.6*e_arma[i-1]
sim_arma <- tsibble(idx = seq_len(100), y = y_arma, index = idx)
```

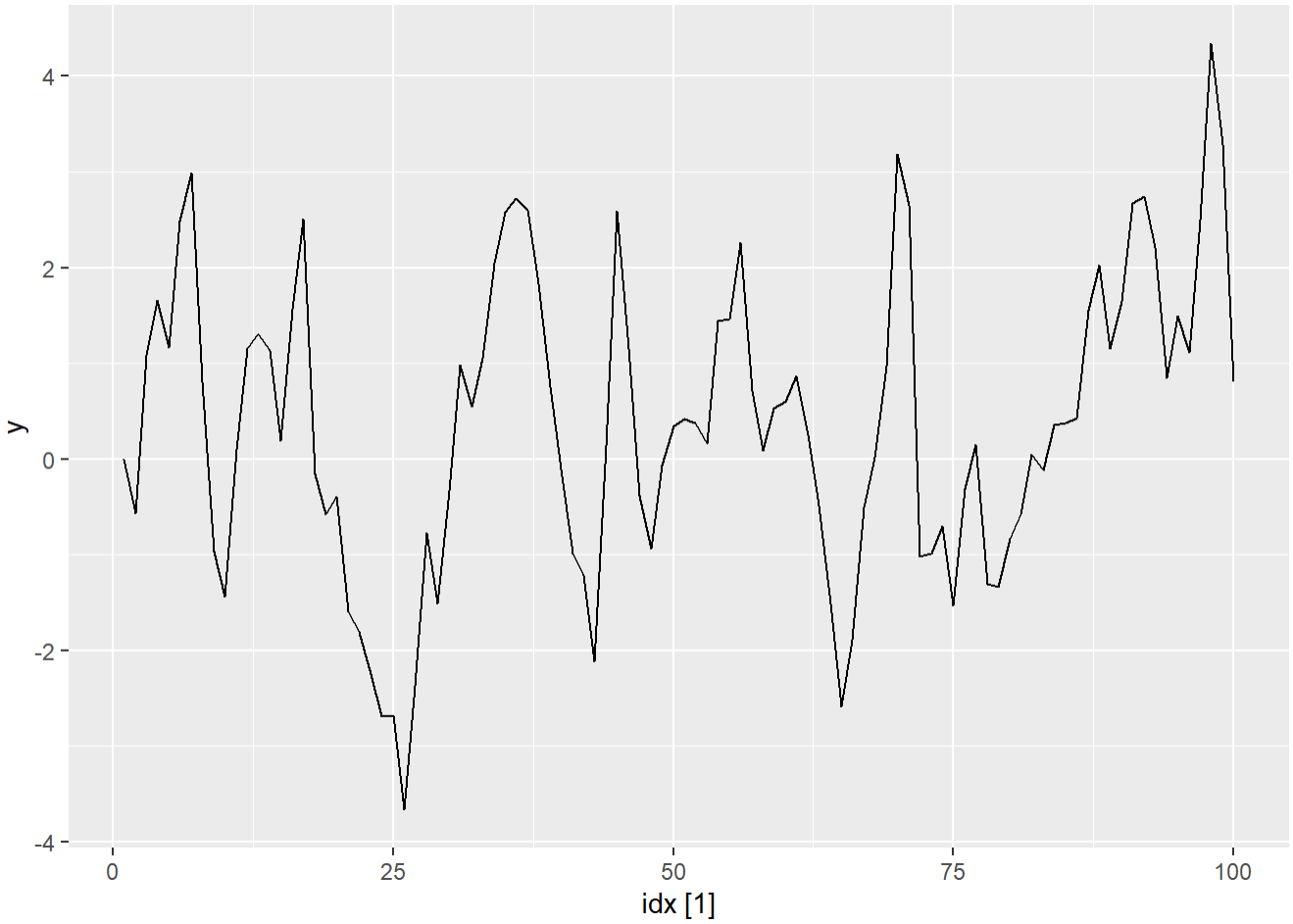
f) Generate data from an AR(2) model with ϕ1=−0.8, ϕ2=0.3 and σ2=1. (Note that these parameters will give a non-stationary series.)

```
## Applying the coefficients into a formula like above.
y_ar2 <- numeric(100)
e_ar2 <- rnorm(100, sd=1)
for(i in 3:100)
  y_ar2[i] <- 0.8*y_ar2[i-1] + 0.3*y_ar2[i-2] + e_ar2[i]
sim_ar2 <- tsibble(idx = seq_len(100), y = y_ar2, index = idx)
```

g) Graph the latter two series and compare them.

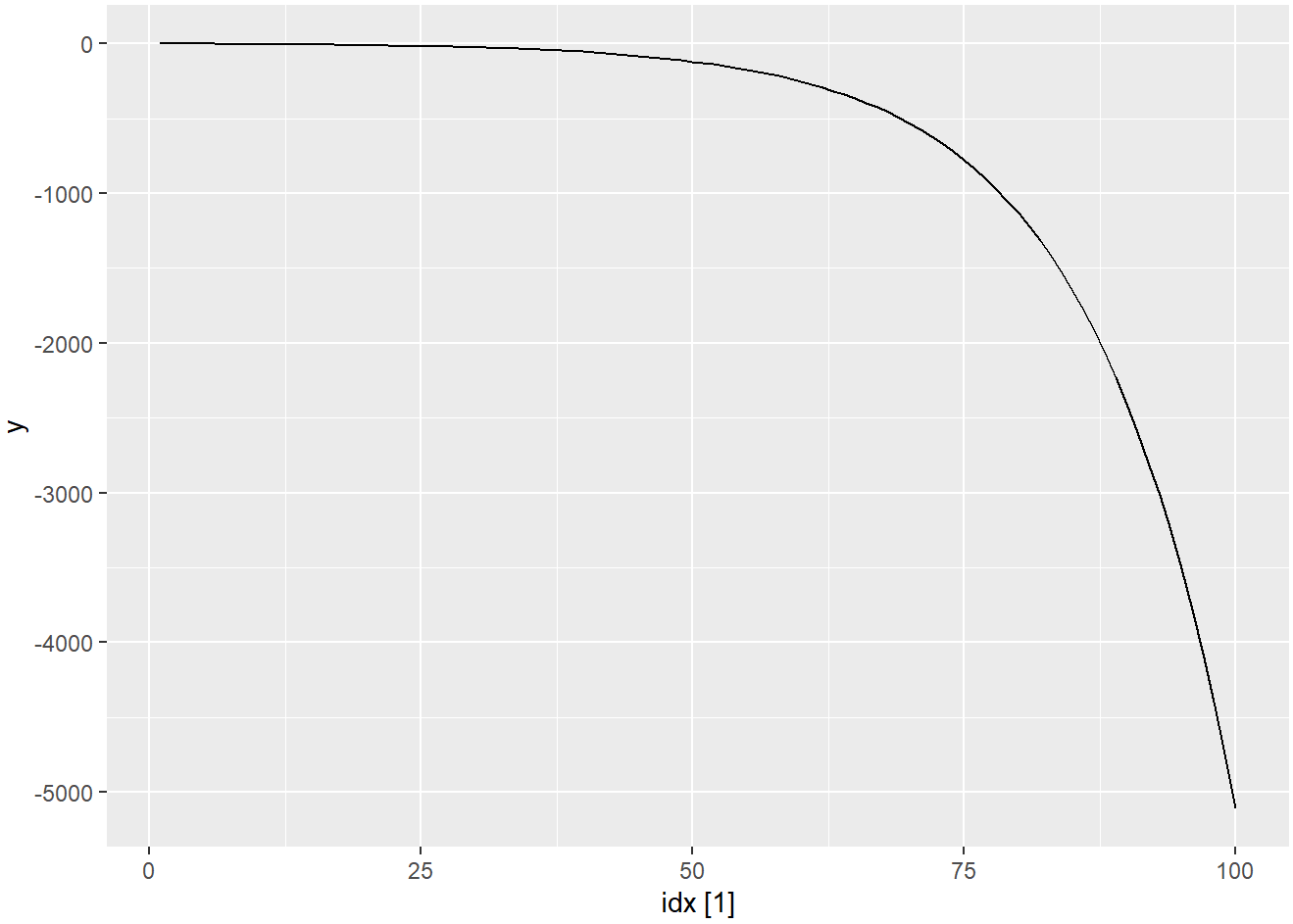
```
#plotting  ARMA(1,1)
print(sim_arma |> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```




```
#plotting AR(2)
print(sim_ar2|> autoplot())
```

```
## Plot variable not specified, automatically selected `.vars = y`
```



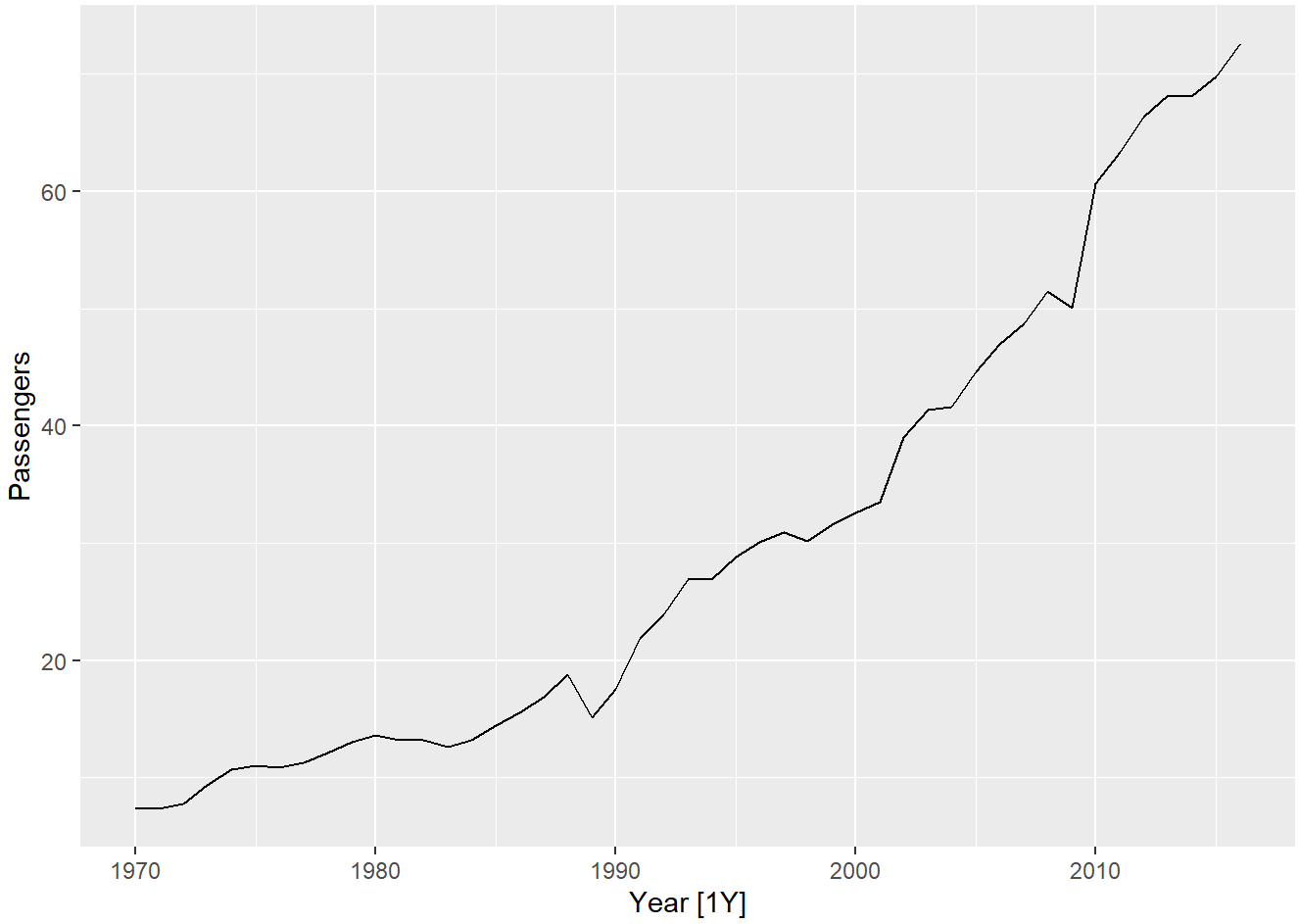

```
### When looking at the plots of both of these charts, it is obvious that the first ARMA(1,1) model does not really have a trend but simply has variations in the data. There doesnt seem t o be a discernable pattern there. However, in the second of the two, the AS(2) model, there i s an obvious, almost exponential trend downwards.
```

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.

```
# plotting first - No real seasonality here, just upward trend over time.
print(aus_airpassengers |> autoplot())
```

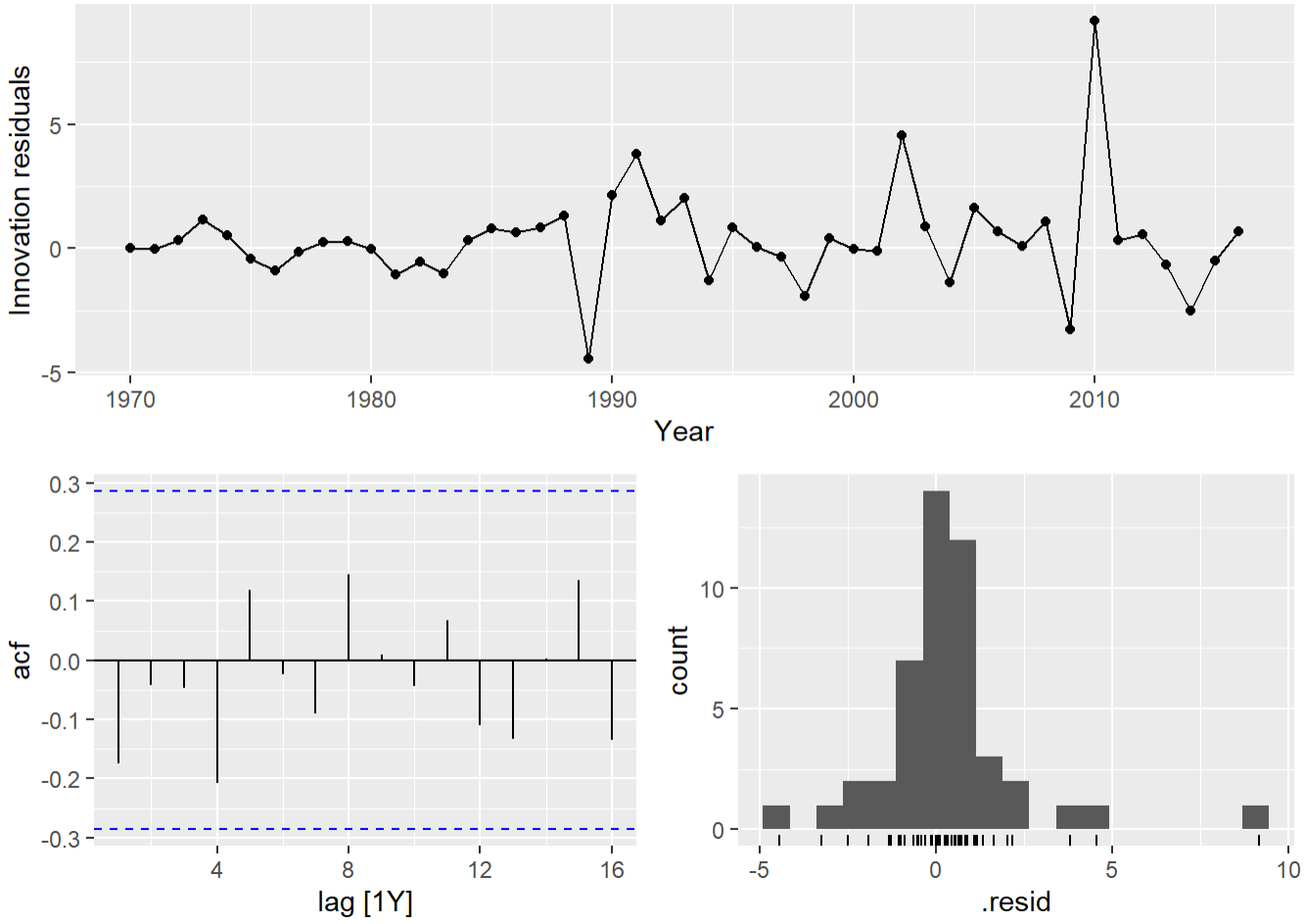
```
## Plot variable not specified, automatically selected `.vars = Passengers`
```



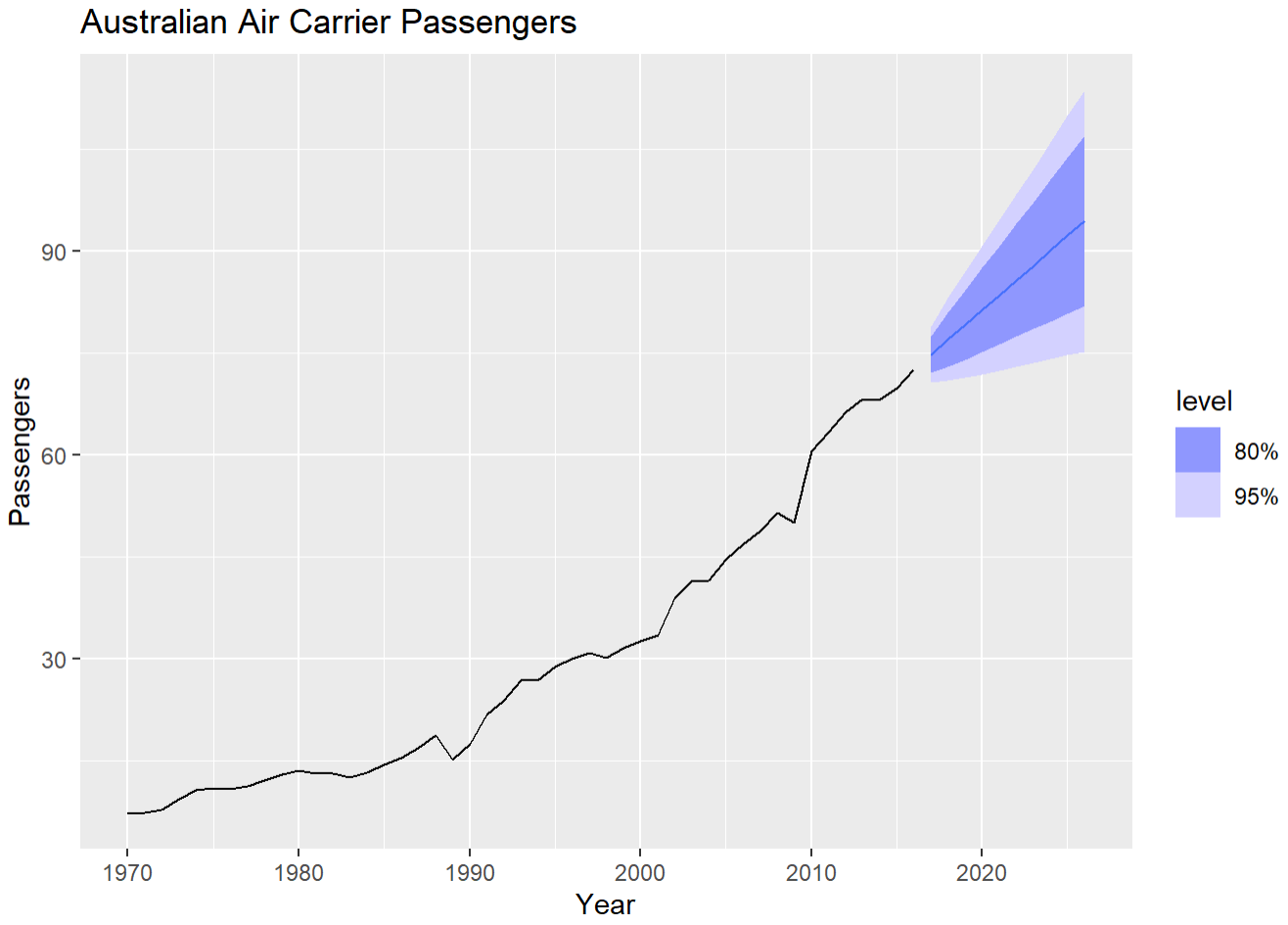

```
## Using built in ARIMA
fit1 <- aus_airpassengers |>
  model(ARIMA(Passengers))

## Printing the decision of the model selector function
print(report(fit1))
```

```
## Series: Passengers
## Model: ARIMA(0,2,1)
##
## Coefficients:
##           ma1
##       -0.8963
## s.e.   0.0594
##
## sigma^2 estimated as 4.308:  log likelihood=-97.02
## AIC=198.04   AICc=198.32   BIC=201.65
## # A mable: 1 x 1
##   `ARIMA(Passengers)`
##               <model>
## 1      <ARIMA(0,2,1)>
```

```
###<OUTPUT>
###Series: Passengers
###Model: ARIMA(0,2,1)

###Coefficients:
###          ma1
###      -0.8963
### s.e.   0.0594

###sigma^2 estimated as 4.308:  log likelihood=-97.02
###AIC=198.04   AICc=198.32   BIC=201.65


## The arima function found that an ARIMA(0,2,1) model was the best fit for the data. This me
ans that no AR modeling techniques were used in the model, and only Moving Averaging was used
. There were also 2 levels of differencing applied. The moving average coefficient that was a
best fit to the data was -0.8963 with a standard error of 0.059. The standard deviation was a
bout 4.3.

## looking at residuals
print(gg_tsresiduals(fit1))
```



```
## The residuals do look like white noise. The acf as no patterns to speak of and the counts are basially normally distributed.

## Predicting the next ten years with the model
print(fit1 |> forecast(h=10) |> autoplot(aus_airpassengers) + labs(y = "Passengers", title = "Australian Air Carrier Passengers"))
```



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

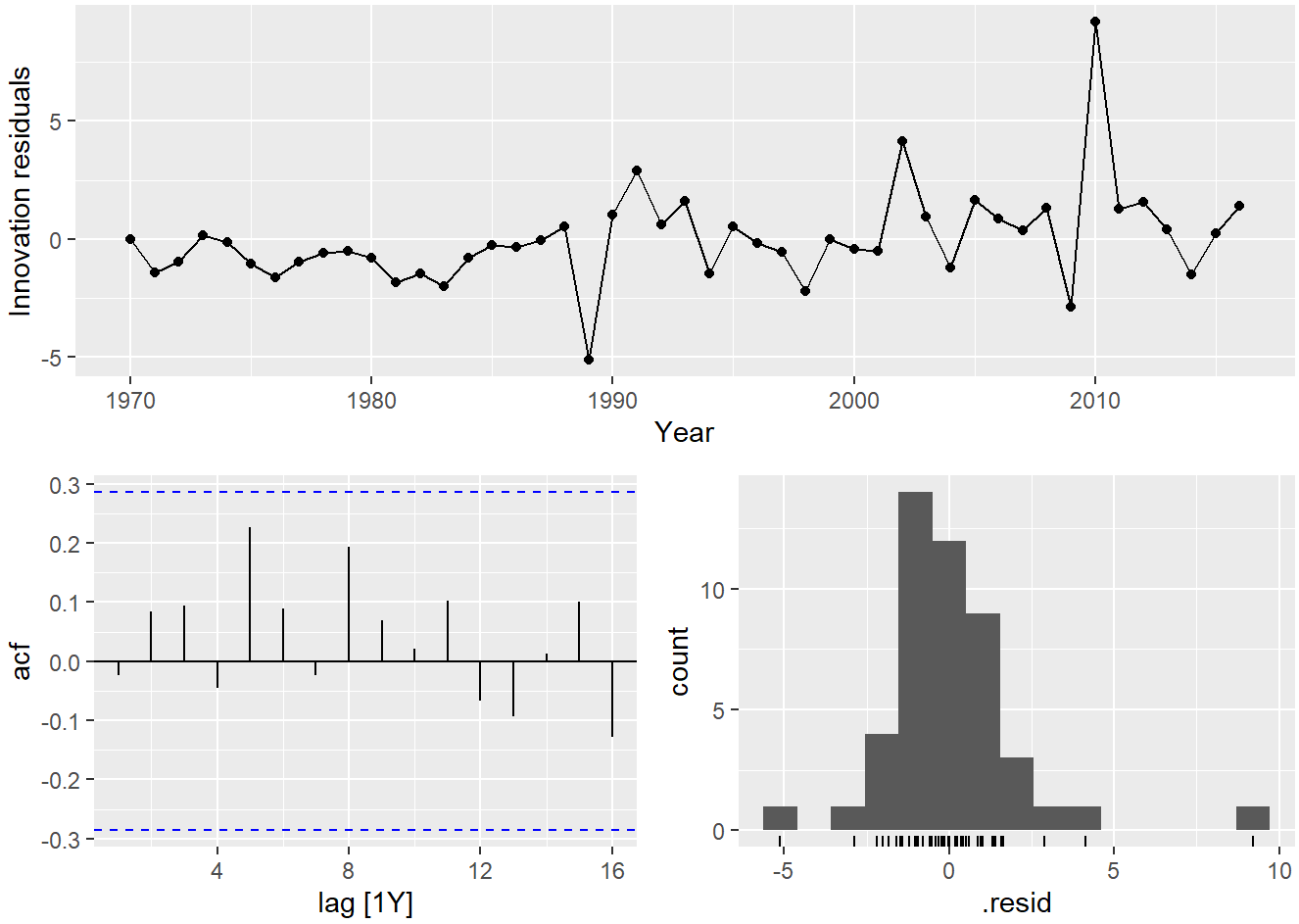
```
#ARIMA(0,2,1)
#p=0 ## No AR
#d=2 ## number of differences
#q=1 ## MA one t diff.

## Replacing with backshift filling from textbook
#(1-B)^2y[t] = -0.8963(B)ε[t]
```

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

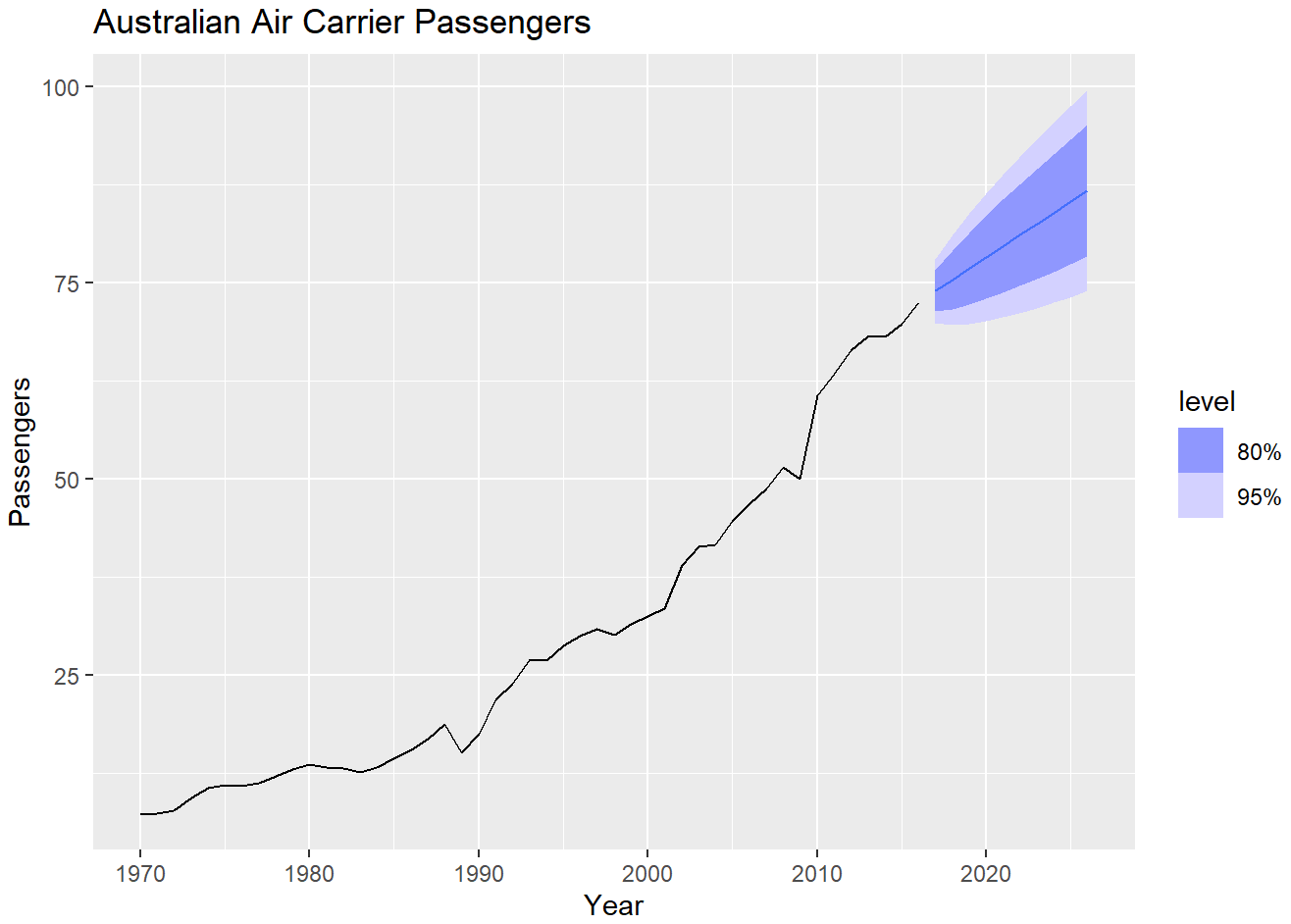
```
## Using built in ARIMA
fit2 <- aus_airpassengers |>  model(ARIMA(Passengers ~ 1+ pdq(0,1,0) )) #

## Printing the decision of the model selector function
print(report(fit2))
```

```
## Series: Passengers
## Model: ARIMA(0,1,0) w/ drift
##
## Coefficients:
##       constant
##         1.4191
## s.e.    0.3014
##
## sigma^2 estimated as 4.271:  log likelihood=-98.16
## AIC=200.31   AICc=200.59   BIC=203.97
## # A mable: 1 x 1
##   `ARIMA(Passengers ~ 1 + pdq(0, 1, 0))`
##                                   <model>
## 1               <ARIMA(0,1,0) w/ drift>
```

```
#Coefficients:
#          ar1      ar2     ma1     ma2  constant
#      -0.5518  -0.7327  0.5895  1.0000    3.2216
#s.e.   0.1684   0.1224  0.0916  0.1008    0.7224

#sigma^2 estimated as 4.031:  log likelihood=-96.23
#AIC=204.46   AICc=206.61   BIC=215.43

#Residuals
print(gg_tsresiduals(fit2))
```




```
## When comparing to part a, the residual counts seem a little more normally distributed. The ACF chart has less negative values, but still is white noise. Similarly, The top chart seems to be pretty similar as well.

## Predicting the next ten years with the model
print(fit2 |> forecast(h=10) |> autoplot(aus_airpassengers) + labs(y = "Passengers", title = "Australian Air Carrier Passengers"))
```



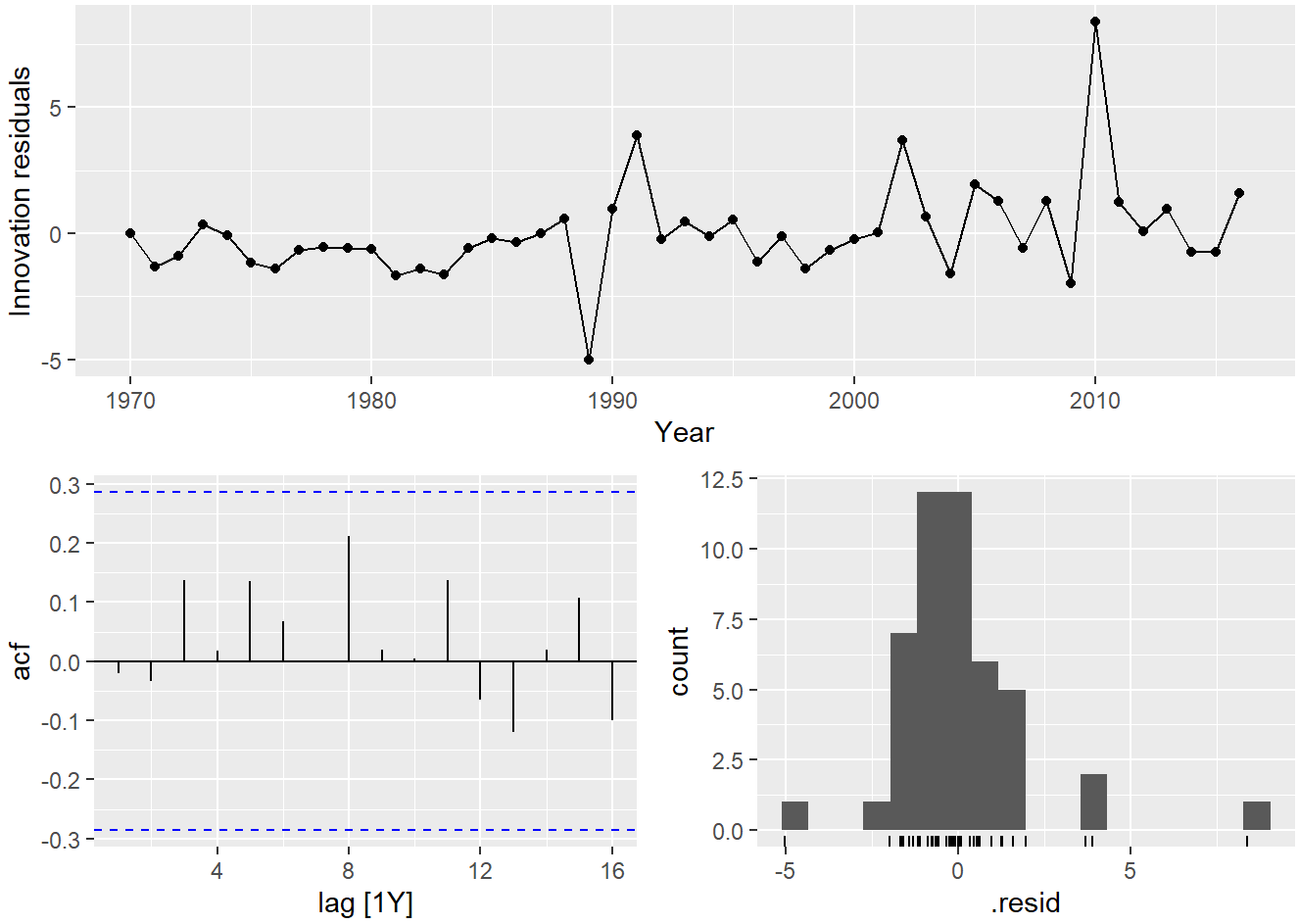

```
## When compared to part a, the forecast is slightly less upwards with this forecast being a bit more blunted in terms of predicted increase in passengers over the next ten years.
```

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.

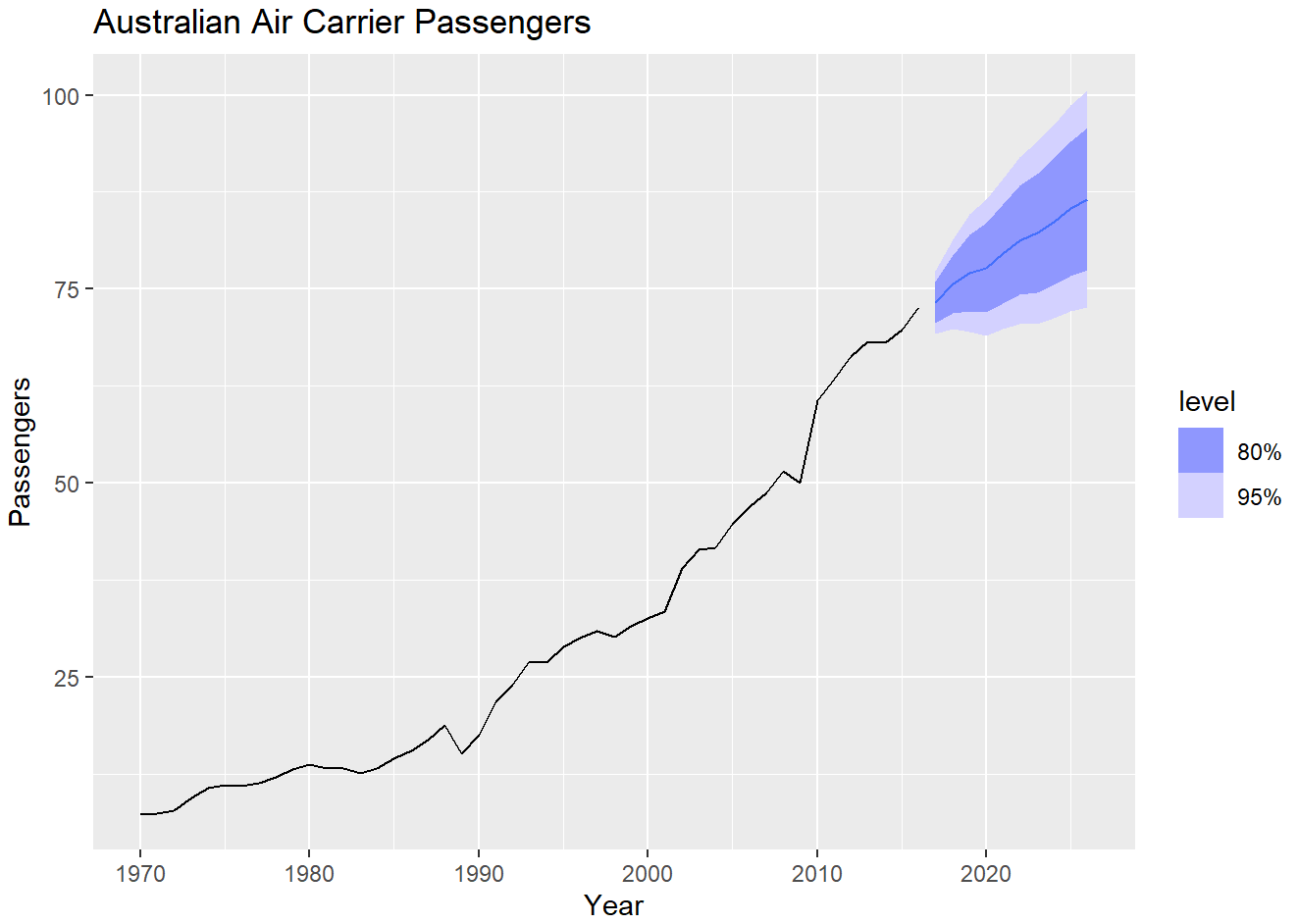
```
fit3 <- aus_airpassengers |>   model(ARIMA(Passengers ~ 1+ pdq(2,1,2)))

print(report(fit3))
```

```
## Series: Passengers
## Model: ARIMA(2,1,2) w/ drift
##
## Coefficients:
##           ar1      ar2     ma1     ma2  constant
##       -0.5518  -0.7327  0.5895  1.0000    3.2216
## s.e.   0.1684   0.1224  0.0916  0.1008    0.7224
##
## sigma^2 estimated as 4.031:  log likelihood=-96.23
## AIC=204.46   AICc=206.61   BIC=215.43
## # A mable: 1 x 1
##   `ARIMA(Passengers ~ 1 + pdq(2, 1, 2))`
##                                   <model>
## 1               <ARIMA(2,1,2) w/ drift>
```

```
#Coefficients:
#          ar1      ar2     ma1     ma2  constant
#      -0.5518  -0.7327  0.5895  1.0000    3.2216
#s.e.   0.1684   0.1224  0.0916  0.1008    0.7224

#sigma^2 estimated as 4.031:  log likelihood=-96.23
#AIC=204.46   AICc=206.61   BIC=215.43

print(gg_tsresiduals(fit3))
```




```
# The residual distribution is less normal than those in part a and c, That is the main diffe
rence i see in the residuals plot.

#Forecast
print(fit3 |> forecast(h=10) |> autoplot(aus_airpassengers) + labs(y = "Passengers", title =
"Australian Air Carrier Passengers"))
```




```
# The forecast with this model is less straight. The link itself is more "wiggled" which mean
s it is more susceptible to fluctuations in the data.


#### Removign the Constant to see any shift

fit3_nc <- aus_airpassengers |>   model(ARIMA(Passengers ~ pdq(2,1,2)))
```

```
## Warning: It looks like you're trying to fully specify your ARIMA model but have not said i
f a constant should be included.
## You can include a constant using `ARIMA(y~1)` to the formula or exclude it by adding `ARIM
A(y~0)`.
```

```
## Warning: 1 error encountered for ARIMA(Passengers ~ pdq(2, 1, 2))
## [1] Could not find an appropriate ARIMA model.
## This is likely because automatic selection does not select models with characteristic root
s that may be numerically unstable.
## For more details, refer to https://otexts.com/fpp3/arima-r.html#plotting-the-characteristi
c-roots
```

```
## When i remove the constant from the model i get the following warning along with a null mo
del.

#Warning: It looks like you're trying to fully specify your ARIMA model but have not said if
a constant should be included.
#You can include a constant using `ARIMA(y~1)` to the formula or exclude it by adding `ARIMA
(y~0)`.Warning: 1 error encountered for ARIMA(Passengers ~ pdq(2, 1, 2))
#[1] Could not find an appropriate ARIMA model.
```

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

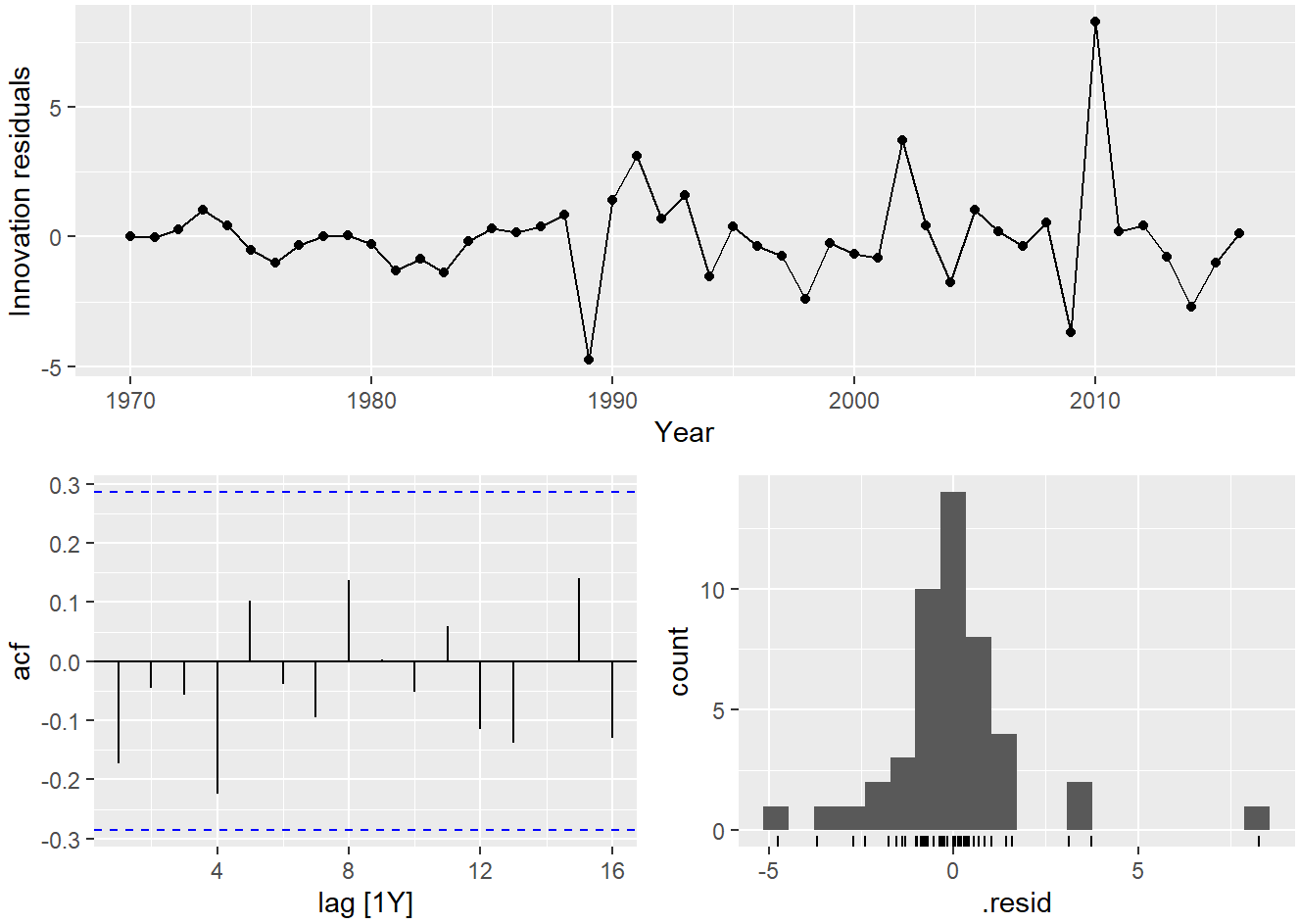
```
fit4 <- aus_airpassengers |>  model(ARIMA(Passengers ~ 1+ pdq(0,2,1) )) #
```

```
## Warning: Model specification induces a quadratic or higher order polynomial trend.
## This is generally discouraged, consider removing the constant or reducing the number of di
fferences.
```

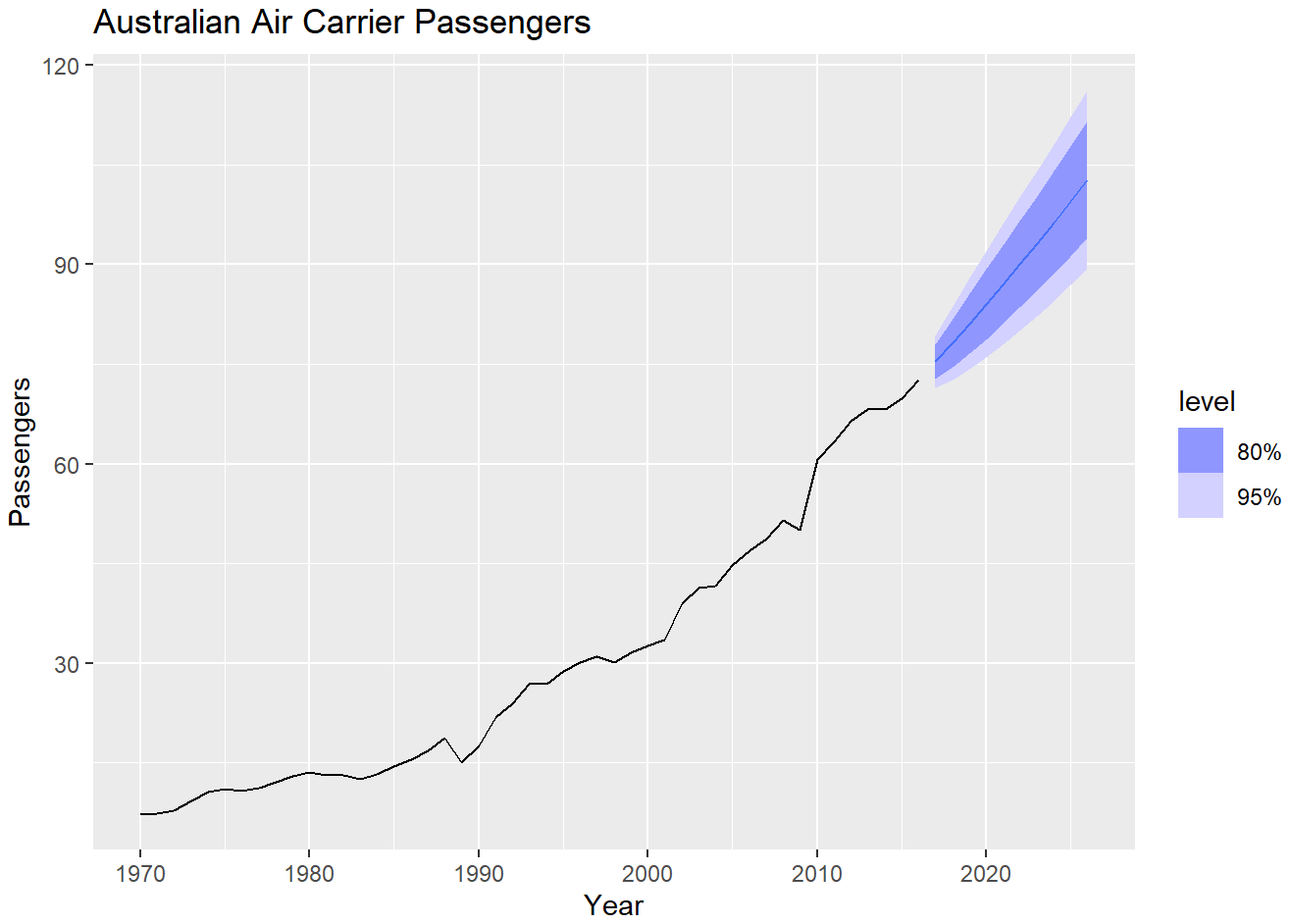
```
## pritnign report
print(report(fit4))
```

```
## Series: Passengers
## Model: ARIMA(0,2,1) w/ poly
##
## Coefficients:
##           ma1  constant
##       -1.0000    0.0571
## s.e.   0.0585    0.0213
##
## sigma^2 estimated as 3.855:  log likelihood=-95.1
## AIC=196.21   AICc=196.79   BIC=201.63
## # A mable: 1 x 1
##   `ARIMA(Passengers ~ 1 + pdq(0, 2, 1))`
##                                   <model>
## 1                 <ARIMA(0,2,1) w/ poly>
```

```
#Output
#Coefficients:
#          ma1  constant
#      -1.0000    0.0571
#s.e.   0.0585    0.0213

#sigma^2 estimated as 3.855:  log likelihood=-95.1
#AIC=196.21   AICc=196.79   BIC=201.63


## Residuals
print(gg_tsresiduals(fit4))
```




```
# The redsidual distribtuion has multiple outliers which may be problematic. The other charts seem to be ok though.

## Forecast
print(fit4 |> forecast(h=10) |> autoplot(aus_airpassengers) + labs(y = "Passengers", title = "Australian Air Carrier Passengers"))
```



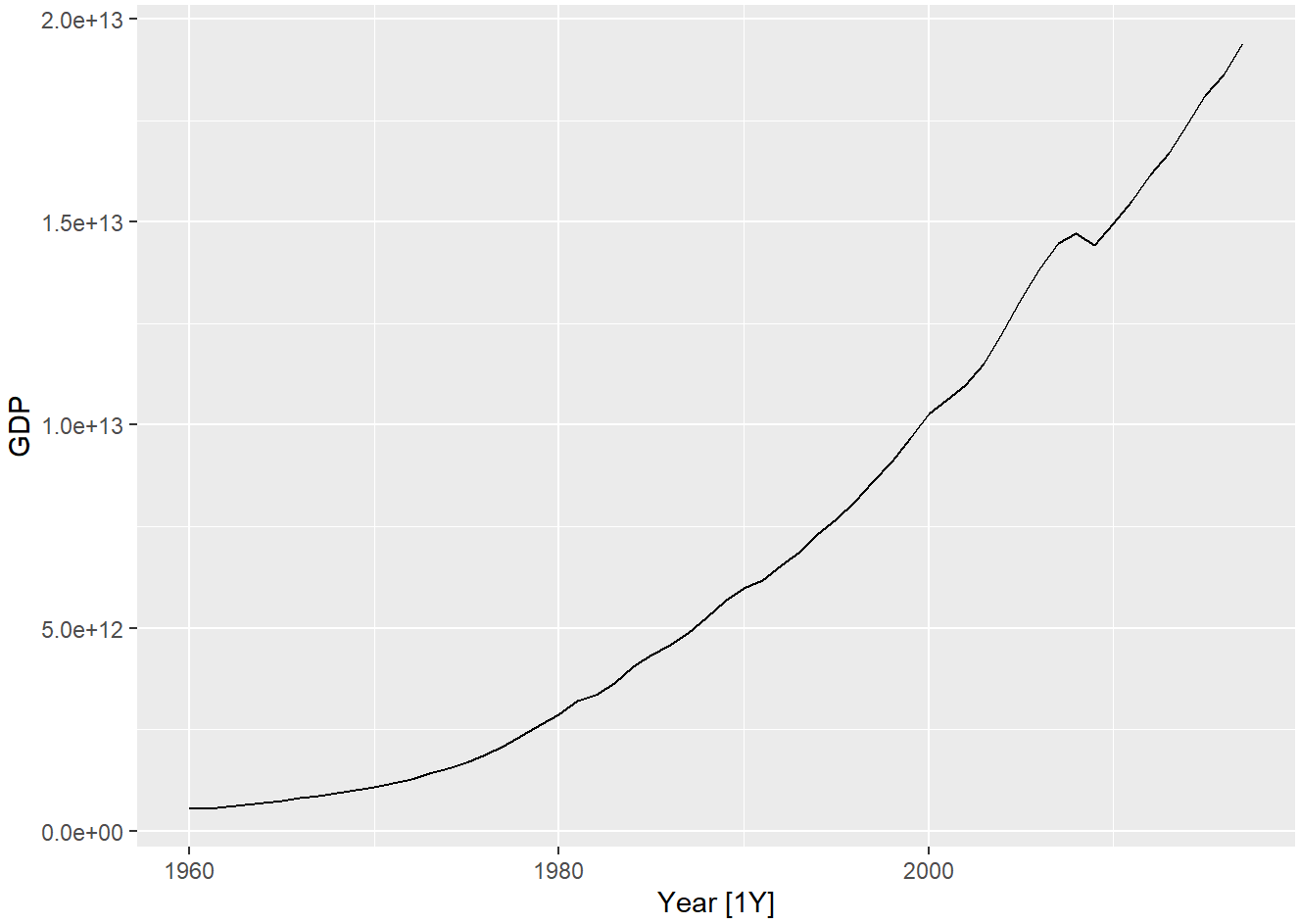

```
##The forecast for this model increases more than the previous ones. Particularly, the confid
ence intervals go up to nearly 120 on the y axis, where in the other models it stays lower. T
he confidence interval bands are also narrower, which means the slope of the overall predicti
on is much more steep than the other models.
```

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

a) if necessary, find a suitable Box-Cox transformation for the data;

```
#print(unique(global_economy$Country))
us_gdp <- global_economy|> filter(Country == "United States")

## Viewing the data
us_gdp |> autoplot()
```

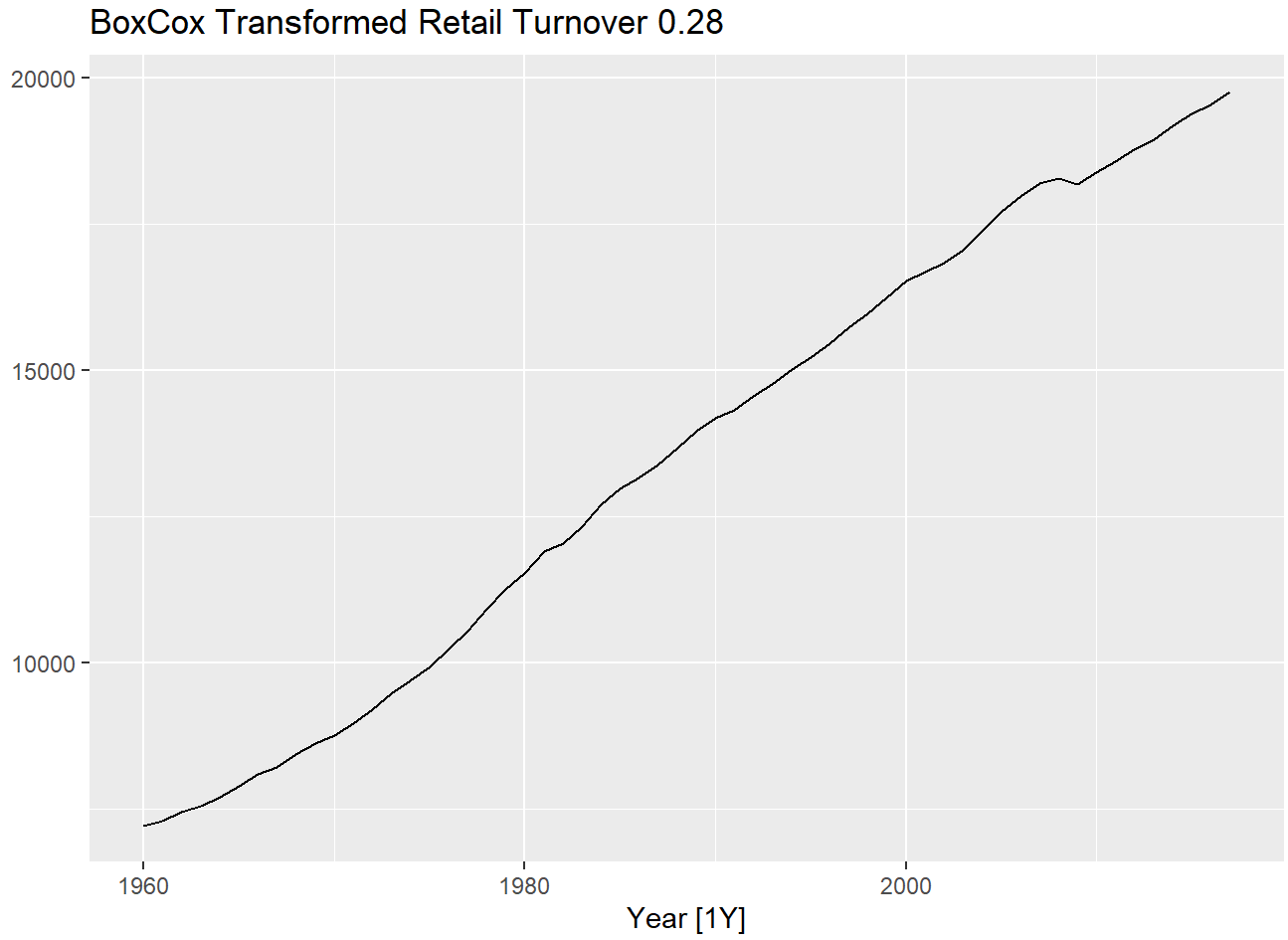
```
## Plot variable not specified, automatically selected `.vars = GDP`
```




```
## Mainly a trend upwards, no seasonality or cyclical anything.

### using Box Cox to get ideal transform value.
lambda <- us_gdp |>
  features(GDP, features = guerrero) |>
  pull(lambda_guerrero)
print(lambda) ## 0.2619
```

```
## [1] 0.2819443
```

```
## Looks better.
print(us_gdp |>  autoplot(box_cox(GDP, lambda)) +  labs(y = "",title = latex2exp::TeX(paste0
("BoxCox Transformed Retail Turnover ",round(lambda,2)))))
```



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

```
#Getting the ideal model using ARIMA
fit_usgdp <- us_gdp |>  model(ARIMA(box_cox(GDP, lambda)))
## Model chosen was ARIMA (1,1,0)
report(fit_usgdp)
```

```
## Series: GDP
## Model: ARIMA(1,1,0) w/ drift
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##          ar1  constant
##       0.4586  118.1822
## s.e.  0.1198    9.5047
##
## sigma^2 estimated as 5479:  log likelihood=-325.32
## AIC=656.65   AICc=657.1   BIC=662.78
```

```
#<OUTPUT>
#Coefficients:
#         ar1  constant
#      0.4586  118.1822
#s.e.  0.1198    9.5047

#sigma^2 estimated as 5479:  log likelihood=-325.32
#AIC=656.65   AICc=657.1   BIC=662.78
```

c) try some other plausible models by experimenting with the orders chosen;

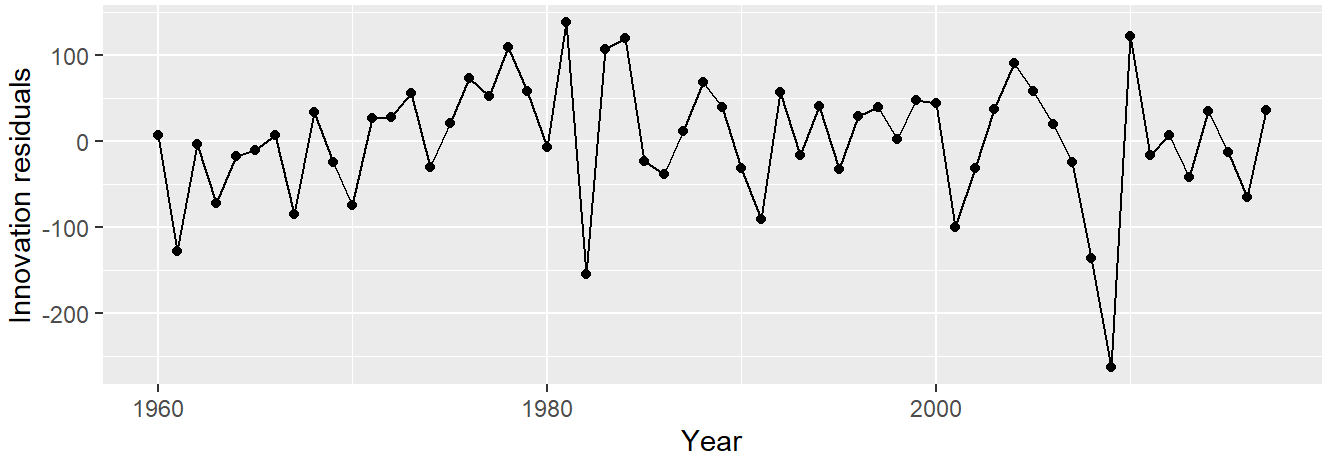
```
## Going to do ARIMA(2,1,0); so one differencing and 2 look back wieghtings
fit_usgdp2 <- us_gdp |>  model(ARIMA(box_cox(GDP, lambda) ~ pdq(2,1,0)))
report(fit_usgdp2)
```

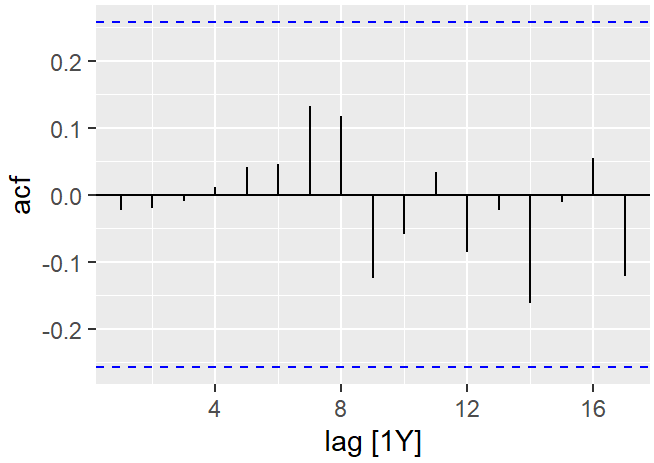
```
## Series: GDP
## Model: ARIMA(2,1,0) w/ drift
## Transformation: box_cox(GDP, lambda)
##
## Coefficients:
##          ar1     ar2  constant
##       0.4554  0.0071  117.2907
## s.e.  0.1341  0.1352    9.5225
##
## sigma^2 estimated as 5580:  log likelihood=-325.32
## AIC=658.64   AICc=659.41   BIC=666.82
```

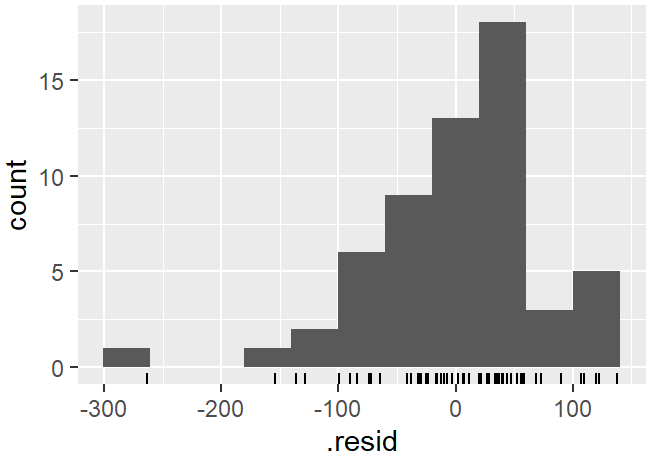
```
## <OUTPUT>
#Coefficients:
#         ar1     ar2  constant
#      0.4554  0.0071  117.2907
#s.e.  0.1341  0.1352    9.5225

#sigma^2 estimated as 5580:  log likelihood=-325.32
#AIC=658.64   AICc=659.41   BIC=666.82
```

d) choose what you think is the best model and check the residual diagnostics;

```
## Choosing the first model as the numbers when looking at the AIC, the BIC, and the AICc, ar
e the most ideal. The second model i tried not that much different, but got worse results wit
h these benchmarks.

#Looking at residuals
gg_tsresiduals(fit_usgdp)
```



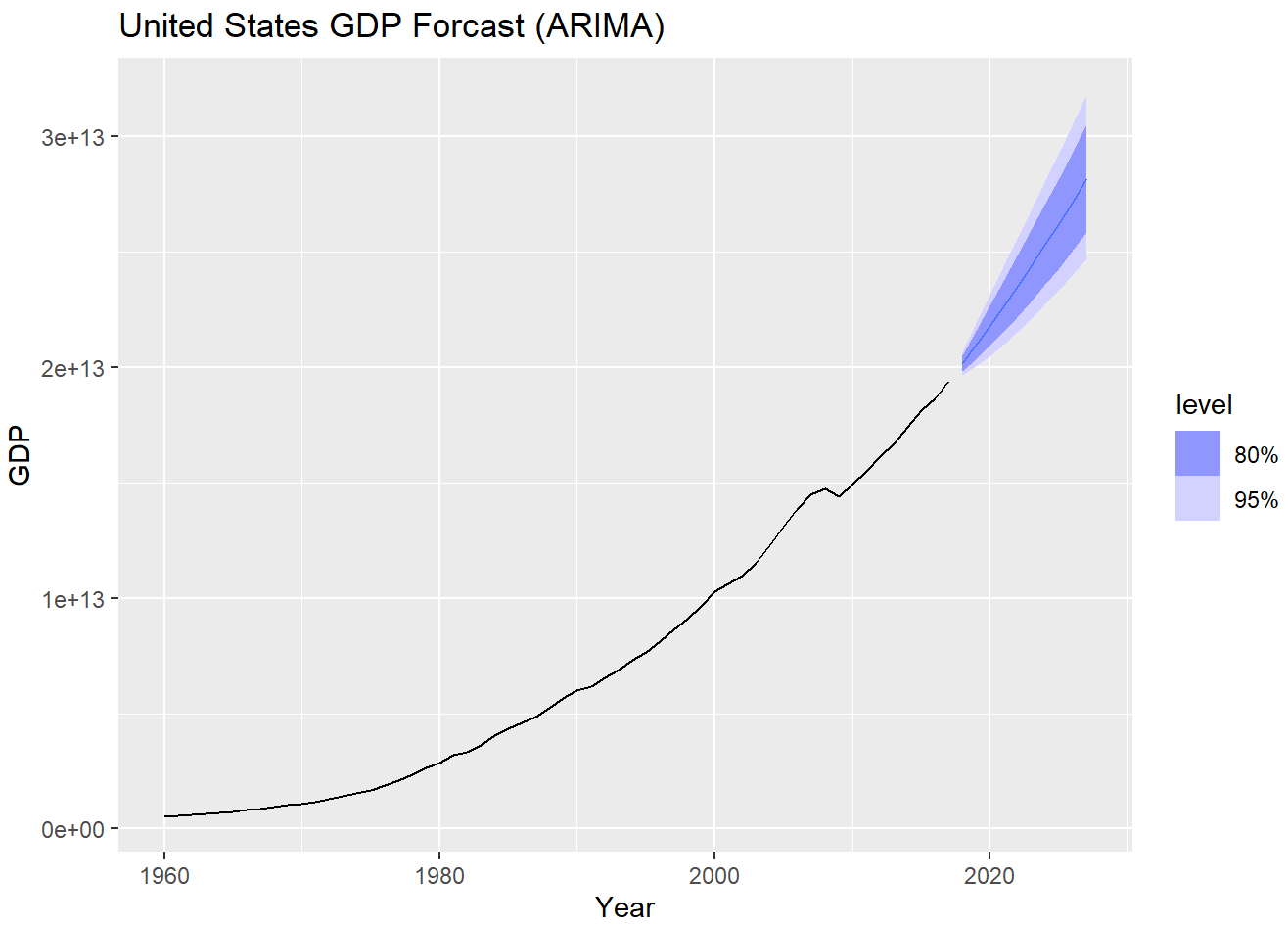





```
## These plots seem to check out, the data is stationary and white noise. There are no discer
nible patterns in the top chart, or the acf chart. The residual count chart does seem to have
a slight patterns, but could just be chance. The residuals are mostly normal looking.
```

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

```
# Forcasting with the chosen models.
print(fit_usgdp |> forecast(h=10) |> autoplot(us_gdp) + labs(y = "GDP", title = "United State
s GDP Forcast (ARIMA)"))
```



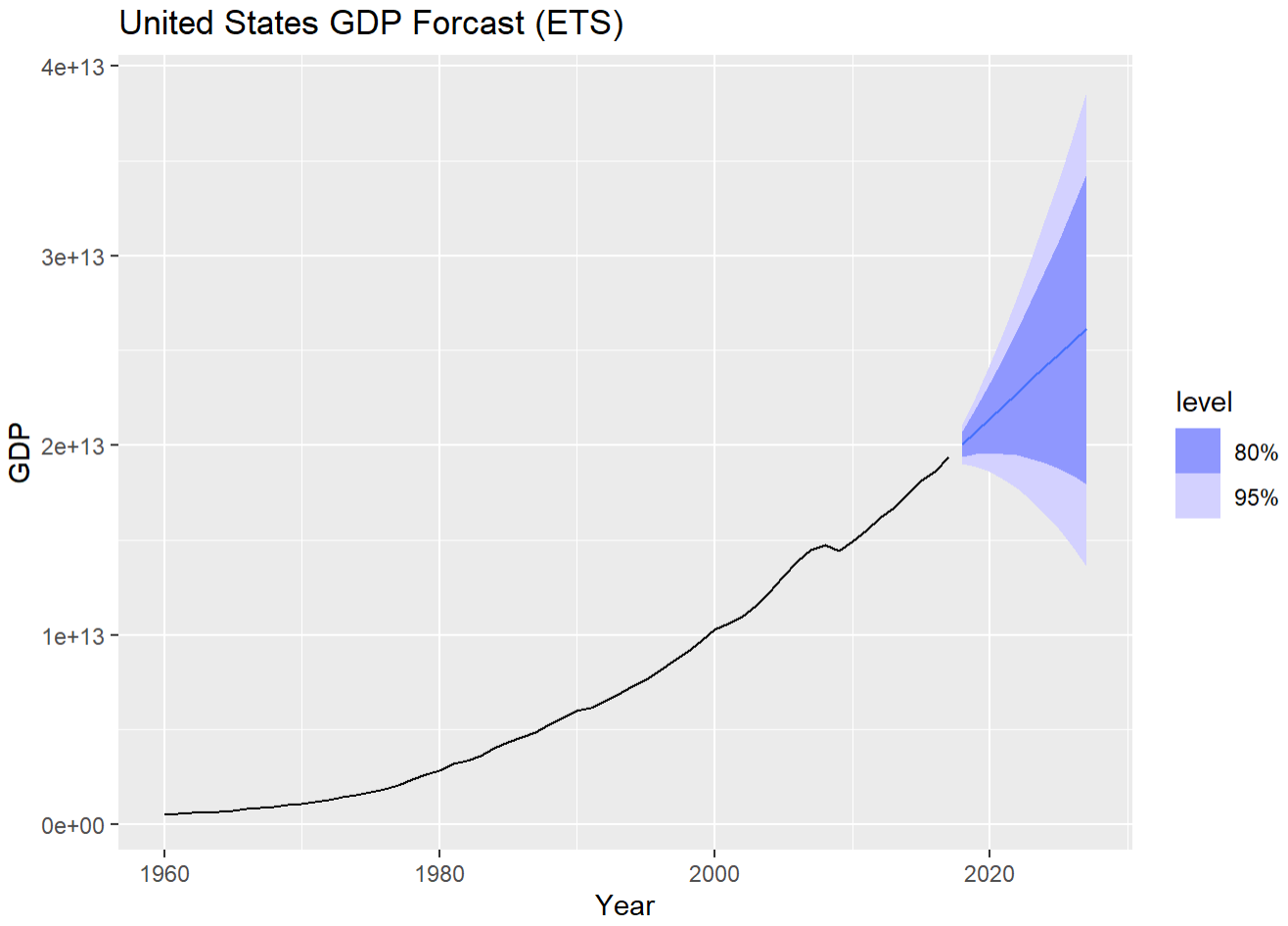

```
## Yes, the forecast does seem reasonable. It very much appears to continue the trend already existant in the data.
```

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

```
us_gdp_ets <- us_gdp |>  model(ETS(GDP))

print(report(us_gdp_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
## # A mable: 1 x 2
## # Key:     Country [1]
##   Country         `ETS(GDP)`
##   <fct>              <model>
## 1 United States <ETS(M,A,N)>
```

```
print(us_gdp_ets |> forecast(h=10) |> autoplot(us_gdp) + labs(y = "GDP", title = "United Stat
es GDP Forcast (ETS)"))
```



```
# The results from the ETS model versus the ARIMA model are different. The ETS forecast does not trend as upward as the ARIMA model does. In addition to that, the confidence intervals for the ARIMA model was much more narrow., while the ETS model seemed much more uncertain as to what was possible.
```