

# mjon238 Stats 326 Assignment 3

mjon238

13/05/2022

## Problem 1: ETS Modelling

First load the data

```
direct <- "Current Uni Stuff/Stats 326/Assignment 3/"
df <- read_csv("productivity.csv")%>%
  as_tsibble(index = Year)

## Rows: 44 Columns: 2

## -- Column specification -----
## Delimiter: ","
## dbl (2): Year, Productivity

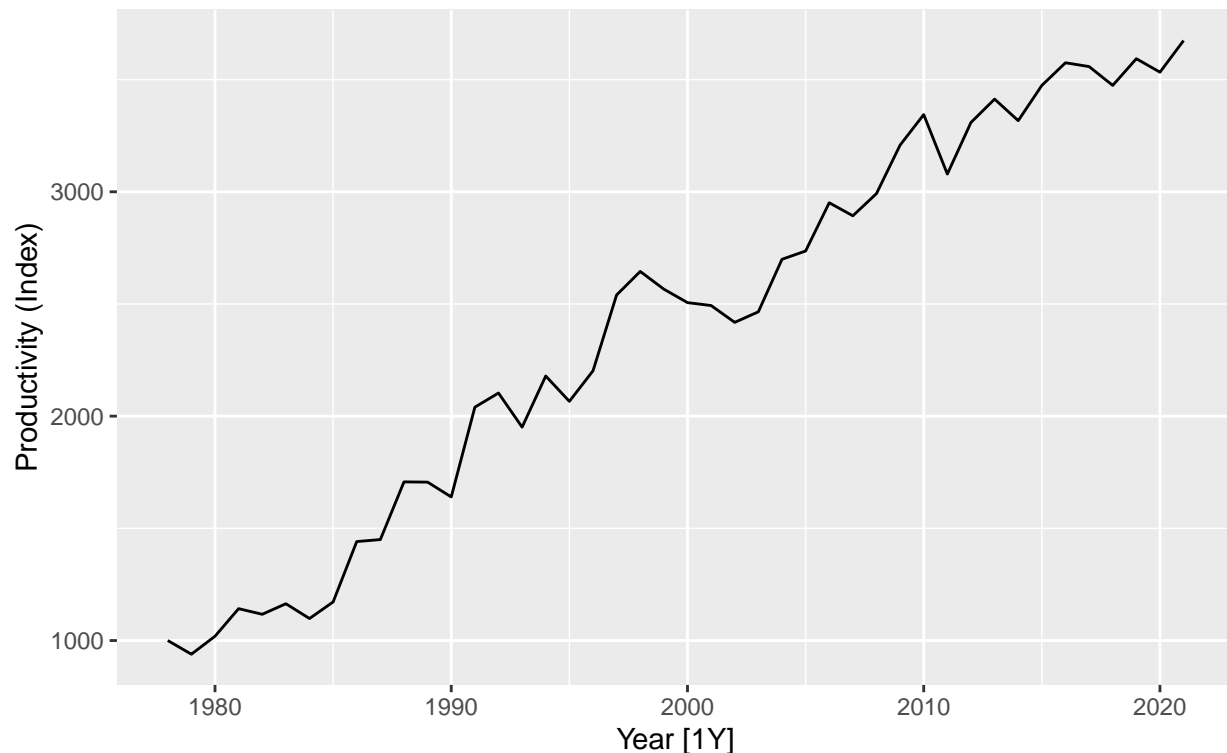
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

## Part 1: Plot the Data

```
df%>%
  autoplot()+
  labs(title = "Labour Productivity for Primary Industries in New Zealand",
        subtitle = "1978 - 2021",
        y = "Productivity (Index)")

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

## Labour Productivity for Primary Industries in New Zealand 1978 – 2021



- There is a linear increasing trend in labour productivity for primary industries in New Zealand.
- The steepest increase is from 1978 to 1998, afterwards productivity continues to increase, however at a slightly slower rate.
- There are cyclical fluctuations in labour productivity, with a number of decreases, notably after 1998 and 2010.
- Productivity does pick-up after these cyclical decreases.

## Part 2

### 1) Fit the Two Models

```
dfTrain <- df%>%  
  filter(Year < 2017)  
  
modelLinear <- dfTrain%>%  
  model(Linear = ETS(Productivity ~ error("A") + trend("A") + season("N")))   
  
modelDamped <- dfTrain%>%  
  model(Linear = ETS(Productivity ~ error("A") + trend("Ad") + season("N")))
```

## 2) Interpret Model Parameters and Compare AICc

```
report(modelLinear)
```

```
## Series: Productivity
## Model: ETS(A,A,N)
## Smoothing parameters:
##   alpha = 0.4889942
##   beta  = 0.0001000043
##
## Initial states:
##   l[0]    b[0]
## 880.7263 68.67191
##
## sigma^2: 19414.32
##
##      AIC      AICc      BIC
## 533.7355 535.5536 542.0533
```

```
beta_star = 0.0001000043/0.4889942
beta_star
```

```
## [1] 0.0002045102
```

Parameters in Holts Linear Trend Method

- $\alpha = 0.489$ , which means the level equation is an approximately 50/50 split of the previous observations level and the observations before.
- $\beta^* = 0.0002 \approx 0$ , which means the slope is just the previous estimate of the slope ( $b_t = b_{t-1}$ ).

```
report(modelDamped)
```

```
## Series: Productivity
## Model: ETS(A,Ad,N)
## Smoothing parameters:
##   alpha = 0.5939802
##   beta  = 0.0001001014
##   phi   = 0.98
##
## Initial states:
##   l[0]    b[0]
## 879.452 86.63342
##
## sigma^2: 20988.5
##
##      AIC      AICc      BIC
## 537.6455 540.2705 547.6269
```

```
beta_star = 0.0001001014/0.5939802
beta_star
```

```
## [1] 0.0001685265
```

Parameters in Holts Linear Damped Trend Method

- $\alpha = 0.594$ , which means the level equation has slightly more weighting to the previous observation than the older past.
- $\beta^* = 0.0002 \approx 0$ , which means the slope is just the previous estimate of the slope ( $b_t = \phi b_{t-1}$ ).
- $\phi = 0.98$ , this is the maximum value  $\phi$  can take, which indicates that the trend is approximately linear (as opposed to a decaying slope).

### 3) Compare AICc

Holts Linear Trend Method has an AICc of 535.55 and Holts Linear Damped Trend Method has an AICc of 540.27. Holts Linear Trend Method has a slightly better fit to the training data, but not by much.

## Part 3

### 1) Create Forecasts

```
fcLinear <- modelLinear%>%
  forecast(h = 5)

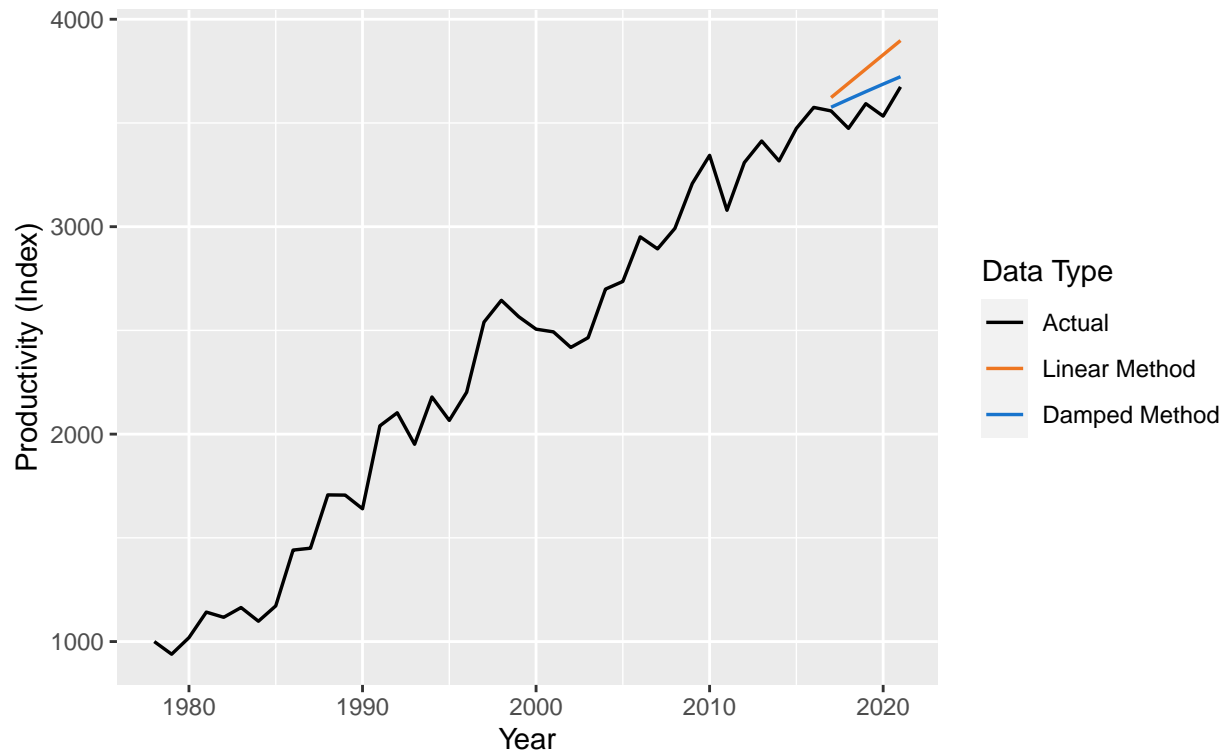
fcDamped <- modelDamped%>%
  forecast(h = 5)
```

### 2) Create Plot of Point Forecasts

```
#Create data frame, overlay point forecasts on original data
forecasts <- data.frame(Year = c(1978:2021),
                        fcDamped = c(rep(NA,39), fcDamped$.mean),
                        fcLinear = c(rep(NA,39), fcLinear$.mean),
                        Actual = df$Productivity)%>%
  pivot_longer(cols = c(Actual, fcLinear, fcDamped),
               names_to = "Data Type",
               values_to = "Productivity")%>%
  mutate(`Data Type` = factor(`Data Type`, levels = c("Actual", "fcLinear", "fcDamped")))

#Create Plots
ggplot(aes(x = Year, y = Productivity, colour = `Data Type`),
      data = forecasts) +
  geom_line(size = 0.6)+
  labs(y = "Productivity (Index)",
       title = "Labour Productivity for Primary Industries in New Zealand",
       subtitle = "Actual and Forecasts (1978 - 2021))+
  scale_color_manual(labels = c("Actual", "Linear Method", "Damped Method"),
                    values= c("black", "chocolate2", "dodgerblue3"))
```

## Labour Productivity for Primary Industries in New Zealand Actual and Forecasts (1978 – 2021)



### 3) Compute Measures of Accuracy

```
#Holts Linear Trend Method
accuracy(fcLinear, df)
```

```
## # A tibble: 1 x 10
##   .model .type    ME  RMSE  MAE   MPE  MAPE  MASE  RMSSE  ACF1
##   <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Linear Test -194.  208.  194. -5.43  5.43  1.52  1.31 -0.115
```

```
#Holts Linear Damped Trend Method
accuracy(fcDamped, df)
```

```
## # A tibble: 1 x 10
##   .model .type    ME  RMSE  MAE   MPE  MAPE  MASE  RMSSE  ACF1
##   <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Linear Test -83.8  99.4  83.8 -2.37  2.37  0.659  0.628 -0.657
```

Holts Linear Damped Trend provides better forecasts. It has significantly lower MAE and RMSE. It is also fits the data more appropriately.

#### 4) Prediction Intervals

```
fc22 <- modelDamped%>%
  forecast(h = 6)

fc22[6,]%>%
  hilo(level = 95)

## # A tsibble: 1 x 5 [1Y]
## # Key:           .model [1]
##   .model Year Productivity .mean          '95%'
##   <chr>  <dbl>         <dist> <dbl>         <hilo>
## 1 Linear  2022 N(3758, 58049) 3758. [3285.731, 4230.176]95
```

The 95% prediction interval for the year 2022 with the Holt Linear Damped Model is (3285.73, 4230.18).

The model estimates there is a 95% probability the New Zealand Labour Productivity for Primary Industries Index in 2022 will be between 3285.73 and 4230.18.

#### 5) Discuss how you would reduce forecast uncertainty

The formula for the prediction interval in with additive errors is  $y_{T+h|T} \pm z_{\alpha/2} \times \hat{\sigma}_h$ . The primary way to reduce the prediction intervals would be to reduce the forecast standard deviation ( $\hat{\sigma}_h$ ) and therefore the variance,  $\sigma_h^2$ .

The formula for the variance is as follows:

$$\sigma_h^2 = \sigma^2[1 + \alpha^2(h-1) + \frac{\beta\phi h}{(1-\phi)^2}\{2\alpha(1-\phi) + \beta\phi\} - \frac{\beta\phi(1-\phi^h)}{(1-\phi)^2}\{2\alpha(1-\phi)^2 + \beta\phi(1+2\phi-\phi^h)\}]$$

We can reduce  $\sigma_h^2$  by:

- Reducing  $h$ , we can do this by increasing our training data set to include observations up-to 2021.
- Reducing  $\alpha$ , this will increase weighting on the older past.
- Reducing  $\beta = \beta^*\alpha$ , this can be done by reducing alpha or reducing  $\beta^*$ , ie. have the slope change less often.

$\phi$  is a less clear variable.

## Problem 2: ARIMA Modelling

### Part 1: Determining an Appropriate ARIMA Model

#### 1) Construct a KPSS Unit Root Test

Our null hypothesis is  $H_0$ : The data is stationary.

Firstly, with no differencing:

```
dfTrain%>%
  features(`Productivity`, unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##   <dbl>     <dbl>
## 1      1.06      0.01
```

We have a p-value of 0.01, therefore we reject the null hypothesis, the data is non-stationary.

Using differencing of order 1, with 1 lag and the same null hypothesis.

```
dfTrain%>%
  features(difference(Productivity, lag = 1), unitroot_kpss)
```

```
## # A tibble: 1 x 2
##   kpss_stat kpss_pvalue
##   <dbl>     <dbl>
## 1    0.0520      0.1
```

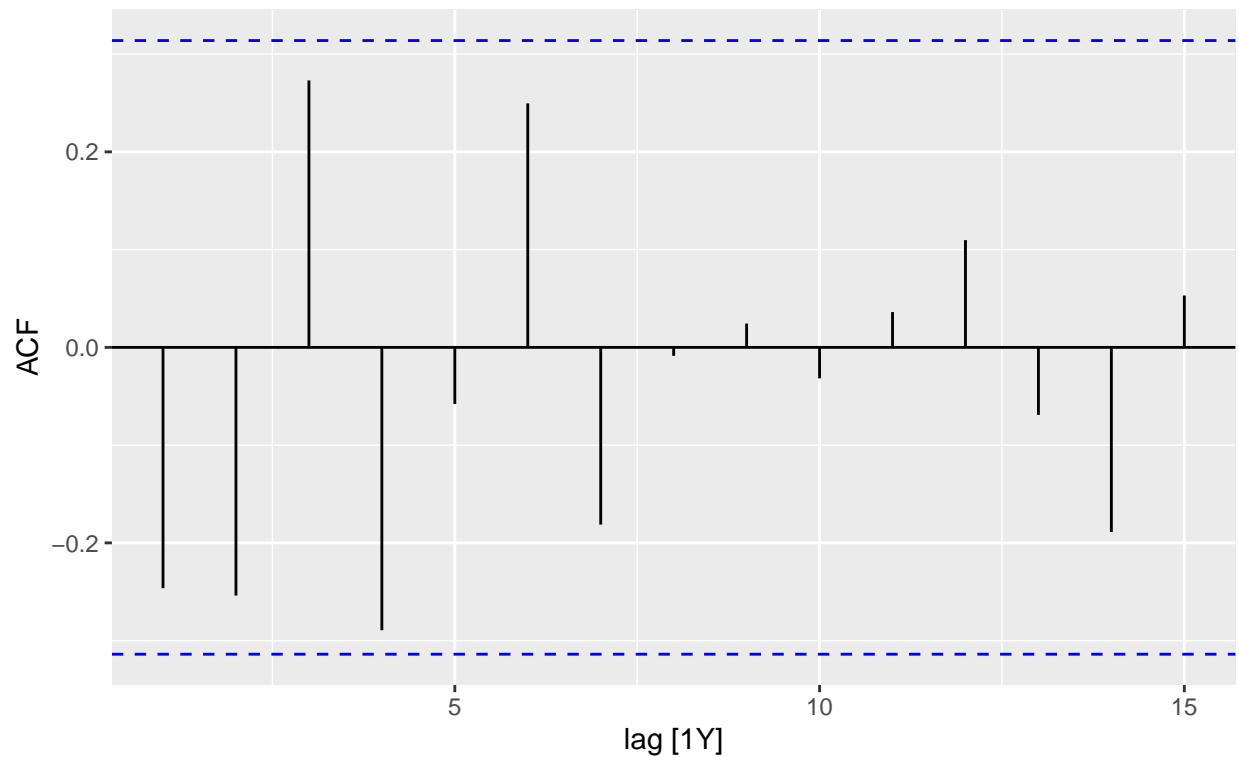
We have a p-value of 0.1, therefore we accept the null hypothesis, the data is stationary.

## 2) Plotting ACF and PACF

```
dfTrain2 <- dfTrain%>%
  mutate(diff = difference(Productivity, lag = 1))

dfTrain2%>%
  ACF(diff)%>%
  autoplot() +
  labs(title = "ACF Plot for Differenced Data With Order 1, Lag 1",
       subtitle = "Labour Productivity for Primary Industries in New Zealand",
       y = "ACF")
```

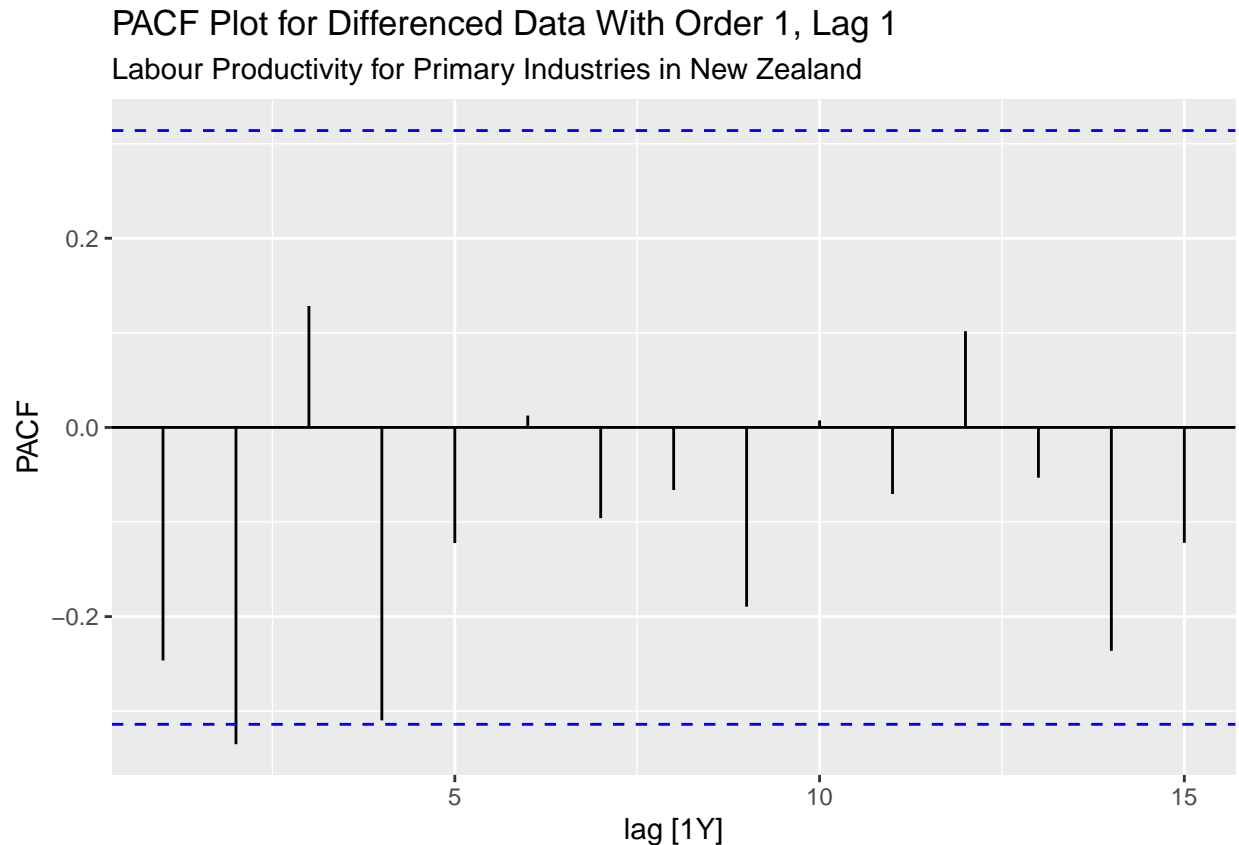
ACF Plot for Differenced Data With Order 1, Lag 1  
Labour Productivity for Primary Industries in New Zealand



The ACF Plot tells us the differences series acts like a white-noise series, with no significant lags.

```
dfTrain2%>%  
  PACF(diff)%>%  
  autoplot() +  
  labs(title = "PACF Plot for Differenced Data With Order 1, Lag 1",  
        subtitle = "Labour Productivity for Primary Industries in New Zealand",  
        y = "PACF")
```





The PACF Plot has a significant lag at lag 2, but the rest are insignificant.

### 3) Suggested ARIMA Model

Firstly the ACF Plot, has no significant lags, therefore our q-term will be 0. Secondly the PACF Plot, has a significant second lag, therefore our p-term will be 2. We are using difference data, so our d term is equal to 1.

Hence, I would fit an ARIMA (2,1,0).

In backshift notation this is:  $(1 - \phi_1 B - \phi_2 B^2)(1 - B)y_t = c + \varepsilon_t$

where  $c$  is a constant and  $\varepsilon_t$  is a white noise series.

## Part 2: ARIMA Model Fitting

```
#Fit Models
fitARIMA <- dfTrain%>%
  model(arima210 = ARIMA(Productivity ~ pdq(2,1,0)),
        stepwise = ARIMA(Productivity, stepwise = TRUE),
        search = ARIMA(Productivity, stepwise = FALSE))

fitARIMA
```

```
## # A mable: 1 x 3
```

```
##           arima210           stepwise           search
##           <model>           <model>           <model>
## 1 <ARIMA(2,1,0) w/ drift> <ARIMA(0,1,1) w/ drift> <ARIMA(0,1,1) w/ drift>
```

```
glance(fitARIMA)%>%
  arrange(AICc)%>%
  select(.model:BIC)
```

```
## # A tibble: 3 x 6
##   .model  sigma2 log_lik  AIC  AICc  BIC
##   <chr>    <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 stepwise 18873.   -240.  486.  487.  491.
## 2 search  18873.   -240.  486.  487.  491.
## 3 arima210 18401.   -239.  486.  487.  493.
```

The stepwise selected model and the non-stepwise (search) selected model are identical, ARIMA(0,1,1) with a drift. This model has the lowest AICc.

In backshift notation this is:  $(1 - B)y_t = c + (1 + \theta_1 B)\varepsilon_t$

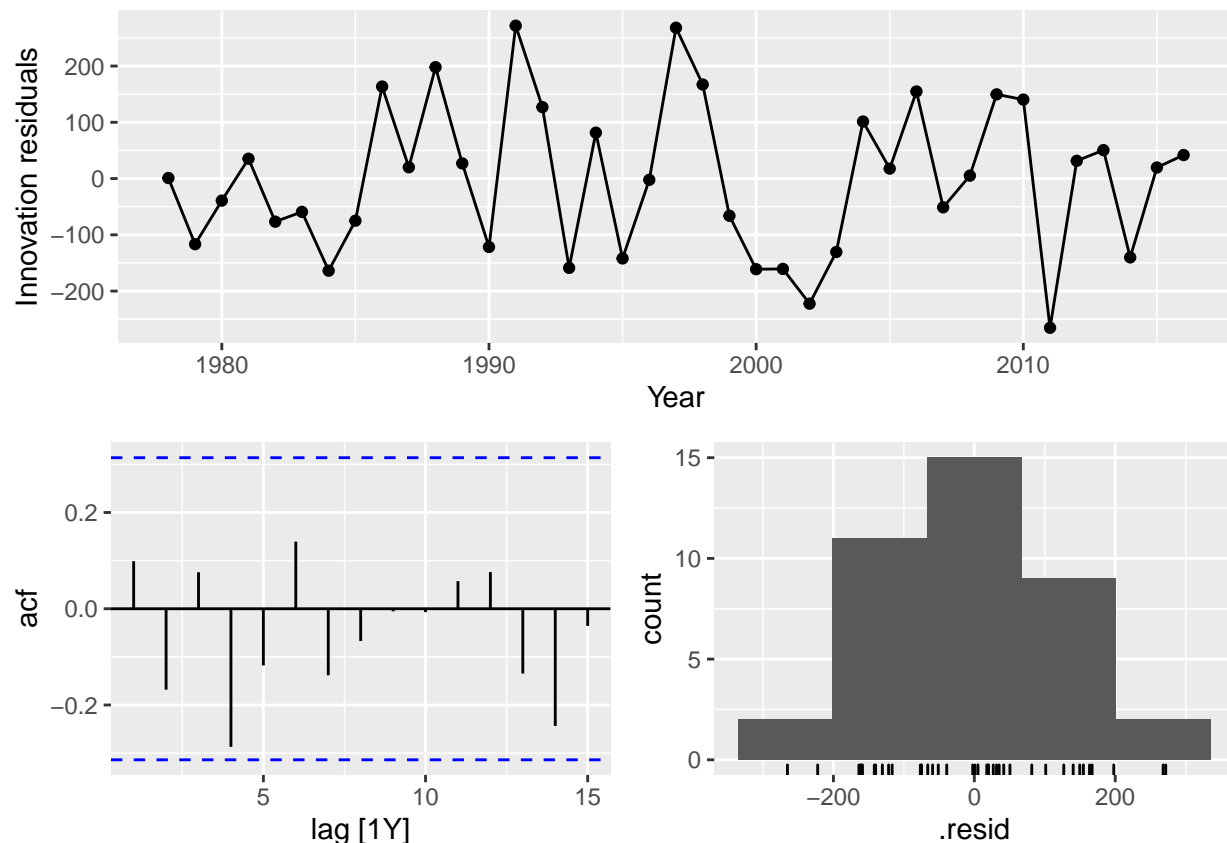
Where  $c$  is a constant and  $\varepsilon_t$  is a white noise series.

## Part 3: Model Checking and Forecasting

### 1) Residuals

```
bestARIMA <- dfTrain%>%
  model(ARIMA(Productivity ~ pdq(0,1,1)))

bestARIMA%>%
  gg_tsresiduals()
```



The residuals look, overall, very good

- They are normally distributed.
- Have approximately zero mean and constant variance.
- They resemble a white-noise series.

I have no concerns about the model, we can accept the ARIMA(0,1,1) as our final model.

## 2) Forecasts

```
fc <- bestARIMA%>%
  forecast(h = 5)
```

```
fc
```

```
## # A tibble: 5 x 4 [1Y]
## # Key:   .model [1]
##   .model      Year Productivity .mean
##   <chr>      <dbl>      <dist> <dbl>
## 1 ARIMA(Productivity ~ pdq(0, 1, 1)) 2017 N(3623, 18873) 3623.
## 2 ARIMA(Productivity ~ pdq(0, 1, 1)) 2018 N(3692, 23805) 3692.
## 3 ARIMA(Productivity ~ pdq(0, 1, 1)) 2019 N(3761, 28737) 3761.
## 4 ARIMA(Productivity ~ pdq(0, 1, 1)) 2020 N(3830, 33669) 3830.
## 5 ARIMA(Productivity ~ pdq(0, 1, 1)) 2021 N(3899, 38601) 3899.
```

### 3) Plotting

```
fc%>%  
  autoplot(df, level = c(90, 99)) +  
  labs(title = "5 Year Forecast Using an ARIMA(0,1,1) with Drift",  
        subtitle = "Labour Productivity for Primary Industries in New Zealand",  
        y = "Productivity (Index)")
```

