Predicting number of deaths using seasonal models

Introduction

In this short project we will try to predict number of deaths caused by lung diseases. For the time series, we will select the *ldeaths* database from the *datasets* package, which contains data on the number of deaths from lung diseases in the United Kingdom. Given the large number of lung diseases, this database covers the number of deaths from bronchitis, asthma, or emphysema. The database itself is created by merging two databases, *mdeaths* and *fdeaths* which contain the same data separately for males and females. The data were collected from January 1974 to December 1979.

Data Preprocessing and Exploatory Data Analysis

Next we move onto data preprocessing and short exploatory data analysis. If we take a closer look we can see that our data is a time series object. These are vector or matrices with class of *ts* which represent data which has been sampled at equispaced points in time. Next, we will turn *ts* object into *tsibble* object which preserves time indices as the essential data column and makes heterogeneous data structures possible.

```
smrti
```

```
     Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
1974 3035 2552 2704 2554 2014 1655 1721 1524 1596 2074 2199 2512
1975 2933 2889 2938 2497 1870 1726 1607 1545 1396 1787 2076 2837
1976 2787 3891 3179 2011 1636 1580 1489 1300 1356 1653 2013 2823
1977 3102 2294 2385 2444 1748 1554 1498 1361 1346 1564 1640 2293
1978 2815 3137 2679 1969 1870 1633 1529 1366 1357 1570 1535 2491
1979 3084 2605 2573 2143 1693 1504 1461 1354 1333 1492 1781 1915
```

```
  class(smrti)
```

```
[1] "ts"
```

```
  # dataset is given as "ts" object so we will turn it into "tsibble" object

  smrti <- as_tsibble(smrti)
```

As mentioned in Introduction, the data were collected from January 1974. to December 1979. This gives us a total of 72 observations. In order to predict number of deaths in future we shall select test data which will then be used for prediction. Because of that we will observe time period from January 1974. to December of 1978.

```
  smrti_m <- smrti |> filter_index(.~"1978-12")
  smrti_m
```

```
# A tsibble: 60 x 2 [1M]
      index value
      <mth> <dbl>
 1 1974 sij  3035
 2 1974 vlj  2552
 3 1974 ožu  2704
 4 1974 tra  2554
 5 1974 svi  2014
 6 1974 lip  1655
 7 1974 srp  1721
 8 1974 kol  1524
 9 1974 ruj  1596
10 1974 lis  2074
# i 50 more rows
```

Now we move on to some numerical characteristics of ours time series. As we can see, down below, minimal number of monthly deaths from lung diseasses in UK was 1300 people while maximum number of deaths in a single month is equal to 3891. Average number of monthly deaths is 2086 and median is 1920. Also, standard deviation is equal to 617.5449.

```
  summary(smrti_m$value)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1300    1568    1920    2086    2552    3891
```

```
sd(smrti_m$value)
```

```
[1] 617.5449
```

If we take a closer look at graphical representation of observed time series given in Image 1 we can see that there is no linear trend but there is some seasonality. As we can see number of deaths is highest in winter months and they begin to drop with spring and summer months. Autumn brings a renewed increase in number of deaths.

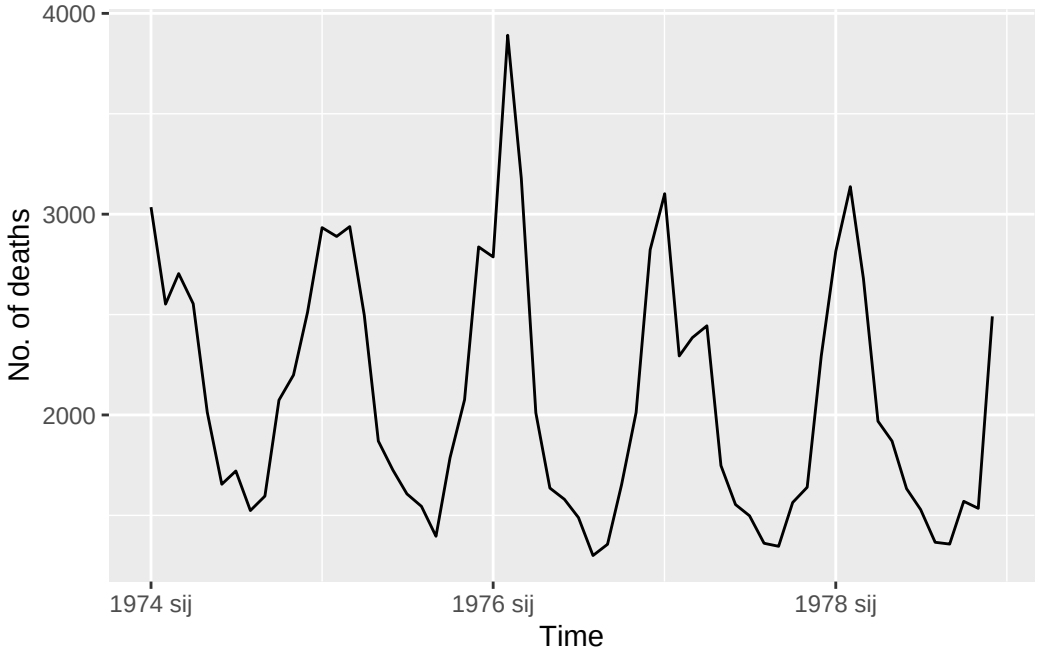


Image 1: Monthly number of deaths from lung diseases in United Kingdom.

This can be better seen if we look at Image 2. We can easily see peaks around February and March of each year followed by strong decrease of deaths up until August. After that, there is a resurgence in the number of deaths as we have already seen.

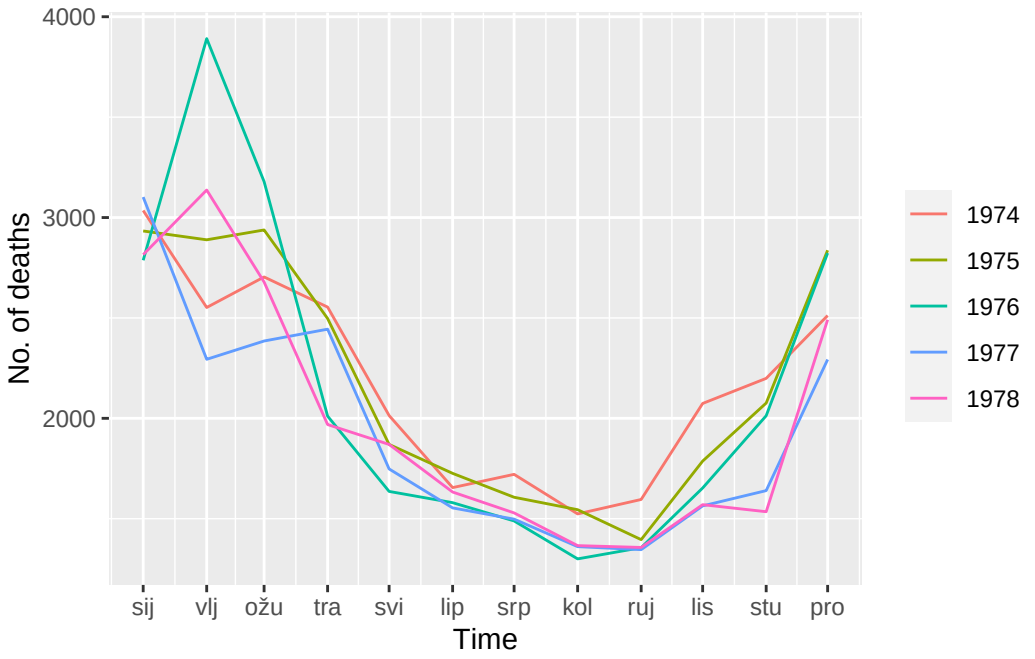


Image 2: Monthly number of deaths from lung diseases in United Kingdom for each year.

In order to successfully model the observed time series, we need to check whether the data is stationary. In order to do this, we will look at the autocorrelation function, whose graphic representation can be seen on Image 3.

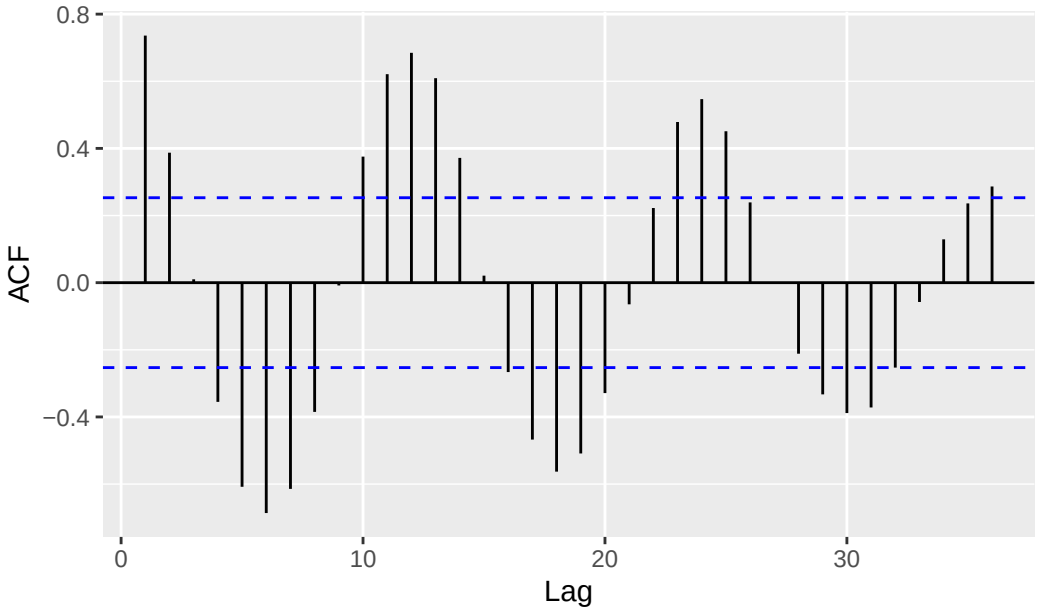


Image 3: Autocorrelation function of monthly number of deaths from lung diseases in United Kingdom.

As mentioned before, we can easily see seasonality among data, and the correlations at steps $12, 24, 36, ...$ very slowly decrease to zero. This tells us that we can doubt the stationarity of data and that we should differentiate it. In order to remove seasonality, we will differentiate the data at step 12. Note that in this case there is no need to test the assumptions about the existence of a unit root, which could consequently also lead to differentiation at the first step, using the extended Dickey - Fuller unit root test and KPSS test because the data does not show any trend. We can verify this by using the *unitroot_ndiffs* function from the *feasts* package, which gives us 0. for the required number of differentiations at the first step.

```
smrti_m <- smrti_m |> mutate(d12 = difference(value, 12))
smrtid12_m<-na.omit(smrti_m)

# How many times should we differentiate at step 1?
smrti_m |> features(value, unitroot_ndiffs)
```

```
# A tibble: 1 x 1
  ndiffs
   <int>
1      0
```

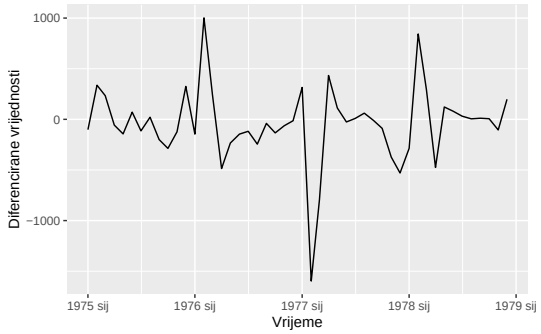
```
  # No need to differentiate at step 1

  smrti_m |>features(value, unitroot_nsdiffs)
```

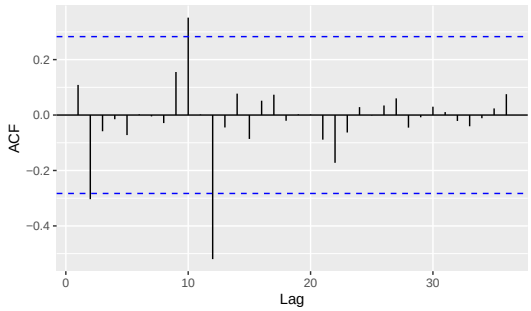
```
# A tibble: 1 x 1
  nsdiffs
    <int>
1       1
```

```
  # It is enough to differentiate once on step 12
```

After differentiation at step 12, Image 4a clearly shows us that there is no more seasonality among the data, and in Image 4b we can see that most of the correlations are not significant, but still we cannot ignore the correlations at steps 10 and 12. Although we have two significant correlations, the *unitroot_nsdiffs* function suggests that one differentiation of the data is sufficient.



(a) after differentiation at step 12.



(b) autocorrelation function of differentiated data.

Image 4: Graphical representation of

Models and Diagnostics

Since we are dealing with seasonal data, we will search for suitable models among SARIMA$(p,d,q)\times(P,D,Q)_{\mathrm{s}}$ processes. In previous analyses, it was very easy to notice that the period s $=12$. Also, considering that we have differentiated the data at a step of 12, when searching for the first and second models, we will fix $d=0,\ D=1$. Additionally, when searching for the second model, we will include a stepwise procedure. Let us note that all models will be chosen based on the smallest Akaike Information Criterion (AIC).

First model

As previously stated, when searching for the first model, we set $d = 0$, $D = 1$, and utilize functions from the *fable* package to identify the optimal model based on the Akaike Information Criterion.

```
m1<- smrti_m |> model(m1 = ARIMA(value ~ pdq(d=0) + PDQ(D=1),stepwise = F))

report(m1)
```

```
Series: value
Model: ARIMA(2,0,0)(1,1,0)[12] w/ drift

Coefficients:
         ar1      ar2     sar1   constant
      0.2773  -0.4121  -0.6040  -122.7094
s.e.  0.1402   0.1570   0.1128    45.4503

sigma^2 estimated as 83964:  log likelihood=-341.07
AIC=692.14   AICc=693.57   BIC=701.5
```

```
# suggested model is SARIMA(2,0,0)x(1,1,0)_12 with drift
# AIC=692.14   AICc=693.57   BIC=701.5
```

Consequently, we acquire the model SARIMA$(2,0,0) \times (1,1,0)_{12}$ with a drift component, yielding an AIC of 692.14. Now, let's focus onto the estimated coefficients for this model which can be seen below.

```
tidy(m1)
```

```
# A tibble: 4 x 6
  .model term     estimate std.error statistic    p.value
  <chr>  <chr>       <dbl>     <dbl>     <dbl>      <dbl>
1 m1     ar1         0.277     0.140      1.98 0.0537
2 m1     ar2        -0.412     0.157     -2.62 0.0116
3 m1     sar1       -0.604     0.113     -5.35 0.00000238
4 m1     constant -123.       45.5       -2.70 0.00955
```

It's easy to notice that all coefficients, except the first one, are significant for this model. Furthermore, looking at the autocorrelation function of residuals in Image 5, we can see that

there is no significant correlations at any lag. Conducting the Ljung-Box test on the residuals, with a p-value of 0.46877, suggests that we can assert the absence of correlated residuals. However, the Shapiro-Wilk test, yielding a p-value of $3.173e-09$, leads us to reject the hypothesis of normality in the residuals' distribution.

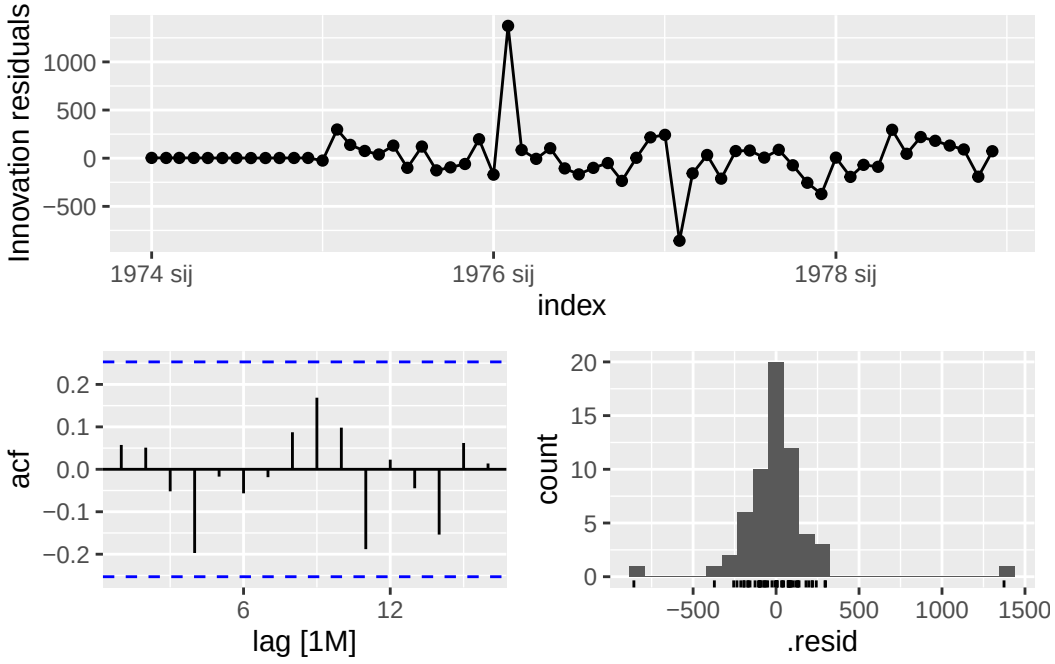


Image 5: Residuals of the first model, their autocorrelation function and histogram.

```
augment(m1) |>
features(.innov, ljung_box, lag = 24, dof = nrow(tidy(m1)))
```

```
# A tibble: 1 x 3
  .model lb_stat lb_pvalue
  <chr>    <dbl>     <dbl>
1 m1        19.8     0.469
```

```
shapiro.test(augment(m1)$.innov)
```

```

    Shapiro-Wilk normality test

data:  augment(m1)$.innov
W = 0.72783, p-value = 3.173e-09
```

Second model

For the second model, we will repeat the analogous procedure as for selecting the first model, but this time we will include the stepwise procedure. As a result, we obtain the model $\text{SARIMA}(0,0,2)\times(1,1,0)_{12}$ also with a drift, with an Akaike Information Criterion of 693.32, making it slightly inferior to the model obtained without the inclusion of the stepwise procedure. Unlike the first model, in this case, we have two coefficients that are not significant, as easily observed in the following code output.

```
m2<- smrti_m |> model(m2=ARIMA(value ~ pdq(d=0)+PDQ(D=1),stepwise=T))
report(m2)
```

```
Series: value
Model: ARIMA(0,0,2)(1,1,0)[12] w/ drift

Coefficients:
         ma1      ma2     sar1   constant
      0.2210  -0.2941  -0.5715  -102.5034
s.e.  0.1567   0.1658   0.1119    42.6348

sigma^2 estimated as 87550:  log likelihood=-341.66
AIC=693.32   AICc=694.74   BIC=702.67
```

```
# suggested model SARIMA(0,0,2)x(1,1,0)_12 with drift

tidy(m2)
```

```
# A tibble: 4 x 6
  .model term     estimate std.error statistic    p.value
  <chr>  <chr>       <dbl>     <dbl>     <dbl>      <dbl>
1 m2     ma1         0.221     0.157      1.41 0.165
2 m2     ma2        -0.294     0.166     -1.77 0.0823
3 m2     sar1       -0.571     0.112     -5.11 0.00000560
4 m2     constant -103.       42.6       -2.40 0.0201
```

Similarly as before, we will analyze the residuals. On Image 6 we see how the residuals are quite similar to those shown on Image 5. We can see that there are no significant correlations, which is confirmed by the Ljung-Box test with a p-value of 0.584472. If we perform the Shapiro-Wilk test, we get a p-value of $2.884\text{e}-09$, so we reject the hypothesis of a normal distribution of residuals.

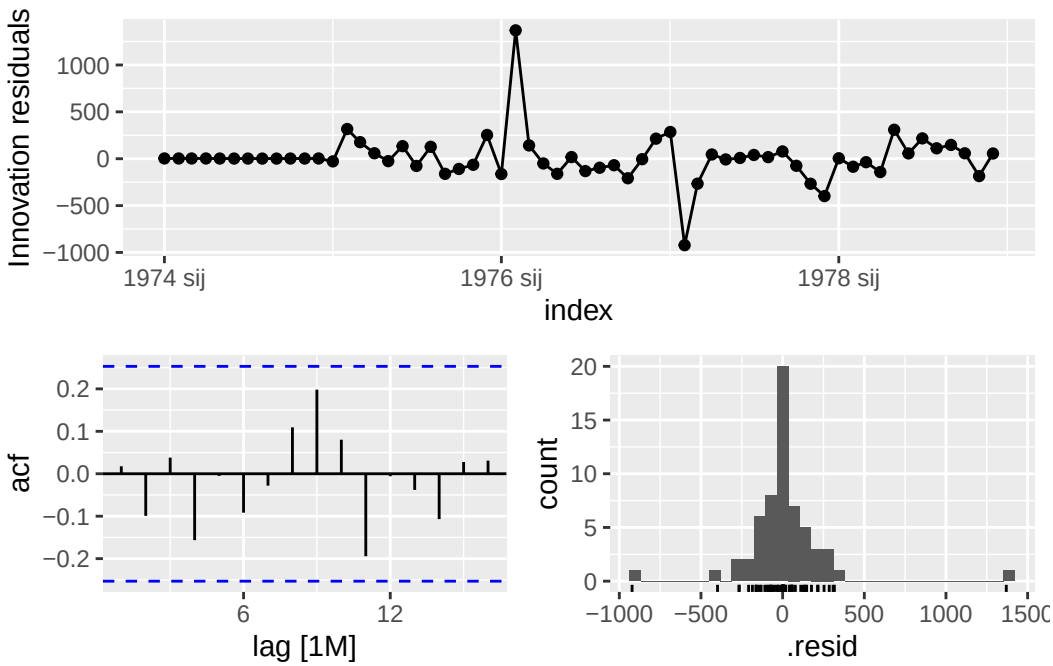


Image 6: Residuals of the second model, their autocorrelation function and histogram.

```
augment(m2) |>
features(.innov, ljung_box, lag = 24, dof = nrow(tidy(m2)))
```

```
# A tibble: 1 x 3
  .model lb_stat lb_pvalue
  <chr>    <dbl>     <dbl>
1 m2        18.0     0.584
```

```
shapiro.test(augment(m2)$.innov)
```

```

    Shapiro-Wilk normality test

data:  augment(m2)$.innov
W = 0.72581, p-value = 2.884e-09
```

Forecasting

Finaly, we can make prediction for each of our models. For that we will use *forecast* function and make prediction for the next 12 months. After we've done that we will compare them and see which one is better.

```
modeli_smrt <- bind_cols(m1, m2)
modeli_smrt
```

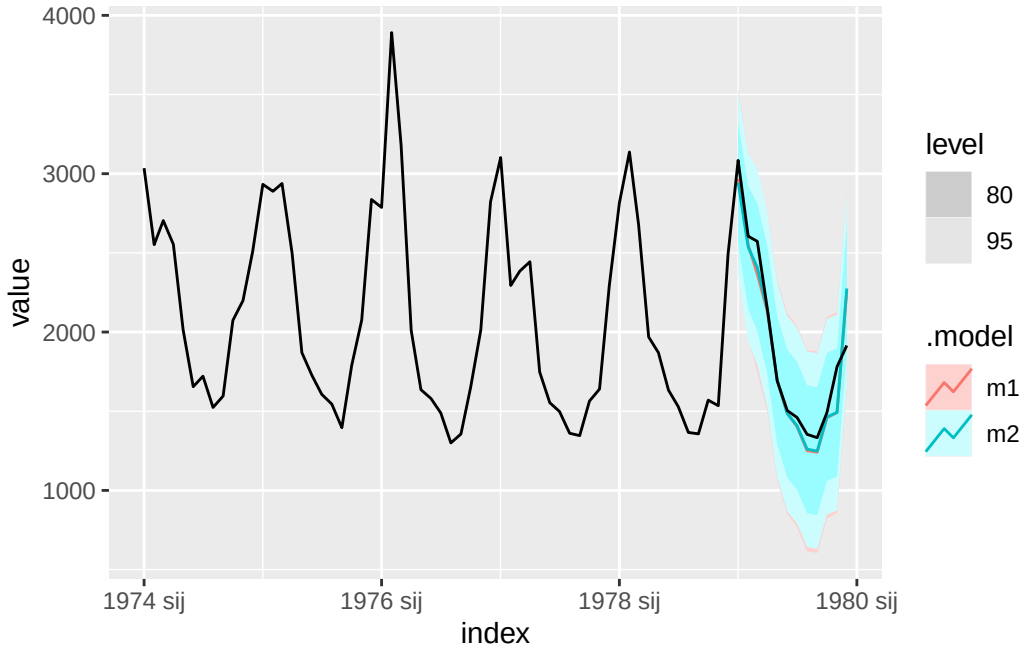
```
# A mable: 1 x 2
                                    m1                                    m2
                                <model>                               <model>
1 <ARIMA(2,0,0)(1,1,0)[12] w/ drift> <ARIMA(0,0,2)(1,1,0)[12] w/ drift>
```

```
pred1 <- modeli_smrt |>
forecast(h = 12)
```

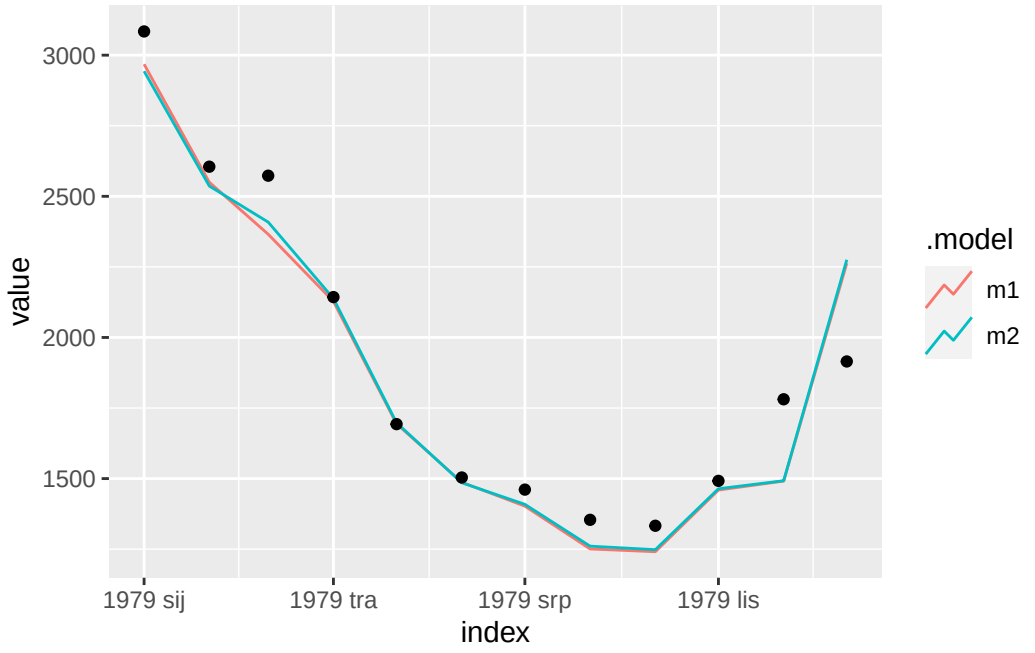
In **?@fig-7-1** we see the forecast for the next 12 months based on two selected models with actual values displayed. For simplicity we will focus on **?@fig-7-2** where we see the actual values and predictions for each model, but without the prior values.

```
#| echo: false
#| label: fig-7
#| fig-cap: "Graphical representation of"
#| fig-subcap:
#|    - "prediction for both models."
#|    - "detailed prediction for both models."
#| layout: [[45,-10, 45], [100]]

pred1 |> autoplot() +
autolayer(smrti, value)
```




```
pred1 |> autoplot(level = NULL) +
geom_point(data = smrti |> filter_index("1979.01" ~ .), aes(y = value))
```



We can easily see a great similarity in the forecast between the first (m1) and second (m2) model, which almost perfectly follow real values in the period from April to June, but in the other months they tend to deviate.

```
fabletools::accuracy(pred1,smrti)
```

```
# A tibble: 2 x 10
  .model .type    ME  RMSE   MAE   MPE  MAPE  MASE RMSSE   ACF1
  <chr>  <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
1 m1     Test   53.2  155.  111.  2.88  5.92 0.457 0.405 -0.392
2 m2     Test   48.3  154.  109.  2.57  5.76 0.448 0.402 -0.386
```

In the table above, we see some coefficients that are used when determining the accuracy of a particular model, and we obtained them using the accuracy function. We observe that the SARIMA$(0,0,2)\times(1,1,0)_{12}$ model proved to be the most accurate. We can see the graphic representation at Image 7 (b) in blue and its formula is given with

$$(1+0.5715B^{12})X_t = (1-0.2210B+0.2941B^2)Z_t - 102.5034.$$