Forecasting COVID-19 Hospitalisations in the UK

Author: Gabriel Berardi

University of Glasgow
School of Mathematics - Data Analytics (ODL)

Submission Date: 16 July 2021

Contents

1. Introduction

The COVID-19 pandemic that started in the beginning of 2020 has revealed the importance of accurately predicting the number of new hospitalizations, to efficiently allocate the limited number of hospital beds and to decide on new policies and restrictions.

In this assignment, we will use data on the daily number of COVID-19 patients admitted to hospital between 23 March 2020 and 22 June 2021 in the UK in order to find a model that can forecast this number for 14 days into the future. The original dataset, that can be downloaded here, has 6 columns: `areaType, areaName, areaCode, date, newAdmissions` and `cumAdmissions.` However, we will only use `date` and `newAdmissions` for the forecasting model. The dataset is valid and there are no missing values present.

In section 2, we will take a closer look at the data and especially the temporal pattern of the number of admissions. In section 3, we will then build two different forecasting models, namely an ARIMA model and a Triple Exponential Model (also known as Holt-Winters Exponential Smoothing). In section 4, we will review the performance of these model and discuss existing limitations as well as suggestions for improving these models.

2. Exploratory Data Analysis (EDA)

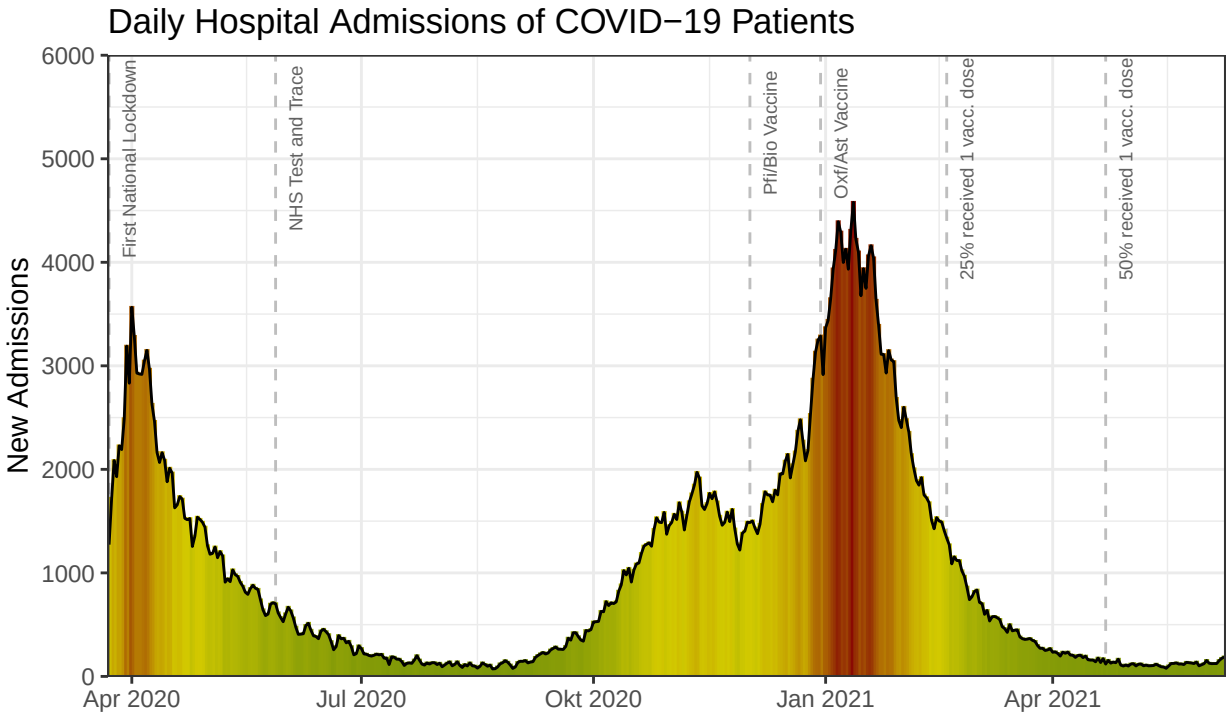
For the purpose of the EDA, we will also keep the `cumAdmissions` column in the dataset. We will remove this column prior to developing the forecasting models. Before start with the EDA, we have to split the data into a training set and a test set. In this case, the training set contains data between 23 March 2020 and 08 June 2021 (443 days), while the test set contains data between 09 June 2021 and 22 June 2021 (14 days). As usual, we will only explore and visualize the training set in the EDA.

First of all, let's look at the summary of the training set:

```
##       date            newAdmissions  cumAdmissions
##  Min.   :2020-03-23   Min.   :  72   Min.   :  4870
##  1st Qu.:2020-07-11   1st Qu.: 179   1st Qu.:129493
##  Median :2020-10-30   Median : 604   Median :174157
##  Mean   :2020-10-30   Mean   :1048   Mean   :245523
##  3rd Qu.:2021-02-17   3rd Qu.:1587   3rd Qu.:432956
##  Max.   :2021-06-08   Max.   :4579   Max.   :468076
```

As we can see, the number of newly admitted COVID-19 patients ranges between 72 and 4,579 with a median of 604. As the median is smaller than the mean, we can see that this column is skewed to the right. Looking at the cumulative number of admissions, we can see that there were already 4,870 patients at the beginning of the dataset.

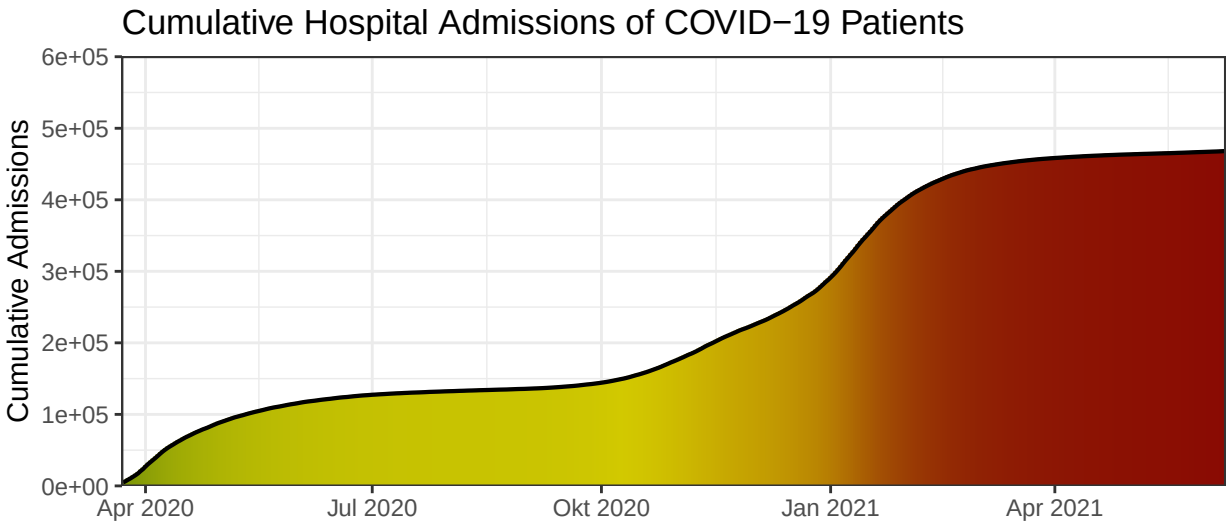
Let's take a first look at a time plot of the daily hospital admissions of COVID-19 patients.



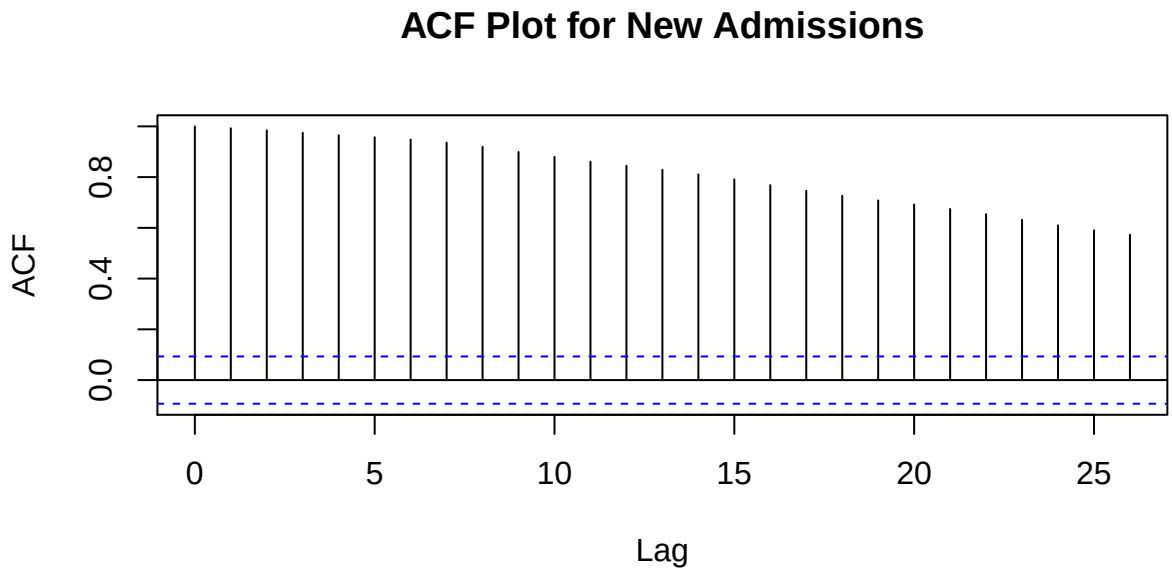
From this time plot, we can see multiple interesting things:

1. There are two peaks in April 2020 and in January 2021
2. In general, it seems that there are fewer COVID-19 patients during the summer months
3. Certain measures, such as lockdowns and vaccinations, influence the number of patients with a certain delay

Another way to visualize this, is by looking at the cumulative number of admitted patients:

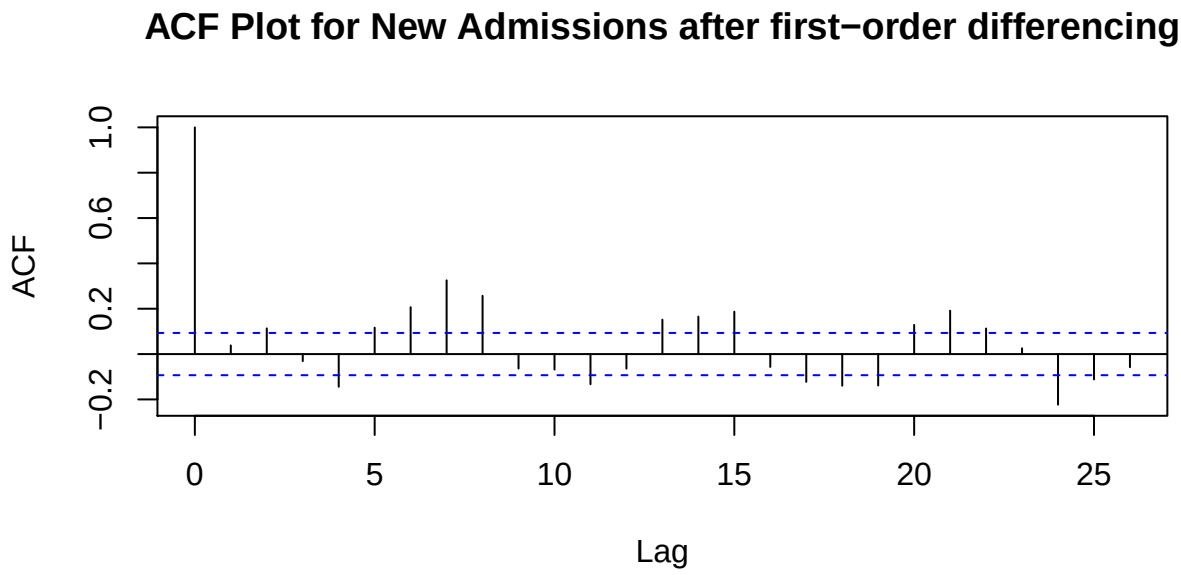


In the next step, we will analyse the temporal structure. Let's begin by looking at a plot of the Auto-Correlation Function (ACF) for different lags:



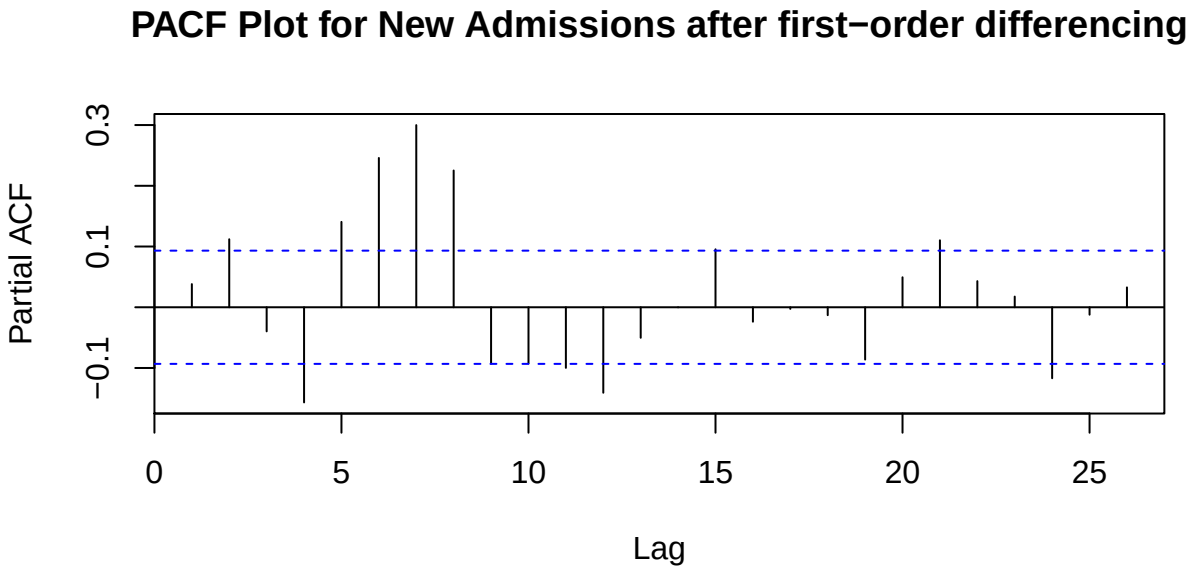
In this ACF plot, we can see that the Auto-Correlation Coefficients (ACC) gradually decrease as the number of lags increase. This suggests that the data is non-stationary, or more specifically, that there is a trend in the data. We can try to make the data stationary by differencing it, which simply means that we take the difference of two subsequent value points within our time series. A dataset with n rows will then result in a new dataset with n-1 rows. We can do this in R using the `diff()` function, in which we can specify the lag.

After applying first-order differencing on the training data, we can draw the ACF plot again:



Looking at the ACF plot after first-order differencing, we can see a clear oscillation indicating a seasonal series. Interestingly, we can see a significant correlation peak at a lag of seven days. This might be evidence of more violations against measures such as social distancing occurring on the weekends.

Let's now look a plot of the Partial Auto-Correlation Function at different lags:



The PACF plot demonstrates a clear cutoff after lag 0 and shows the correlation around the lag of 7 days discussed above even more clearly. We can conclude this short EDA by noticing that a first-order differencing technique seems necessary to make the data stationary.

3. Forecasting Models

Let's now build two forecasting models: an Autoregressive Integrated Moving Average model (ARIMA) and a Triple Exponential model (TEM).

3.1 ARIMA

An ARIMA(p, d, q) model is essentially defined by these three parameters:[1]

- **p:** Describes the order of the Auto-Regressive part of the model
- **d:** Describes the integrative part of the model, i.e. the degree of differencing to turn non-stationary into stationary data
- **q:** Describes the Moving-Average part of the model

[1]Hyndman et al.: Forecasting Functions for Time Series and Linear Models, 2021, Page 11

It is difficult to judge what choice would be appropriate for the three parameters, only by looking at the ACF and PACF plots. The only thing we say is that `d` should be equal to 1, as we have seen that the data is stationary after first-order differencing.

The most efficient way to find an appropriate ARIMA model is to search the space of all possible models, by combining different values for `p, d, q` and comparing a certain criterion - such as AIC, AICc or BIC between the model. This can be done using the `auto.arima()` function from the `forecast` package. The question is, which criterion we should choose for model selection:

- **AIC:** The Akaike Information Criterion can be calculated as $2\,k - 2\,ln(\hat{L})$ with k being the number of predictors in the model and $\hat{L}$ being the maximum value of the likelihood function for the model.
- **AICc:** The small-sample corrected Akaike Information Criterion and can be calculated as $AIC + penalty$. The exact formula for the penalty depends on the statistical model, but in general it will penalize the excessive use of predictors in the model. The AICc should be used if we are dealing with a small dataset. Many researchers suggest to use AICc if the sample size is smaller than 40 times the number of parameters.
- **BIC:** The Bayesian Information Criterion can be calculated as $k\,ln(n) - 2\,ln(\hat{L})$ with k again being the number of predictors, n being the sample size and $\hat{L}$ being the maximum value of the likelihood function of the model. The main difference between the AIC and BIC criterion is that BIC penalizes model complexity more heavily.

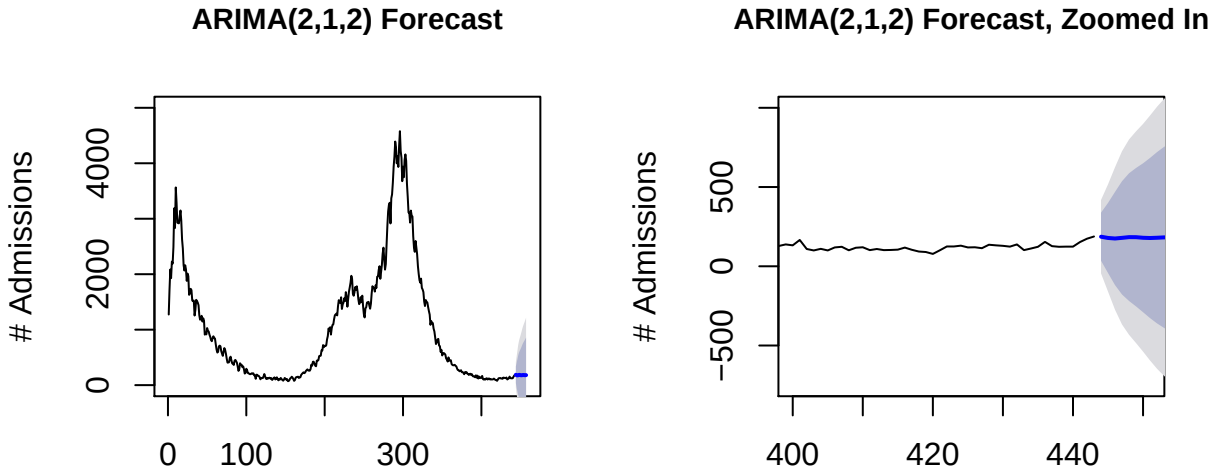
Selecting one of these three criterion introduces some subjectivity into the process. However, if all three criteria yield the same result, we can objectively say that there is string evidence that this is the best model.

Luckily, this is the case with this dataset. Running `auto.arima()` with all default parameters (most notably `stationary = FALSE` and `seasonal = TRUE`) yields an `ARIMA(2,1,2)` model as the best fitting model. Let's look at its summary:

```
## Series: train_df$newAdmissions
## ARIMA(2,1,2)
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##       0.5411  -0.7282  -0.4761  0.9049
## s.e.  0.0559   0.0713   0.0398  0.0473
##
## sigma^2 estimated as 13975:  log likelihood=-2735.02
## AIC=5480.05   AICc=5480.19   BIC=5500.51
##
## Training set error measures:
##                      ME     RMSE     MAE       MPE     MAPE      MASE       ACF1
## Training set -2.053272 117.5453 72.2848 -1.015777 9.681969 0.9640009 -0.0475815
```

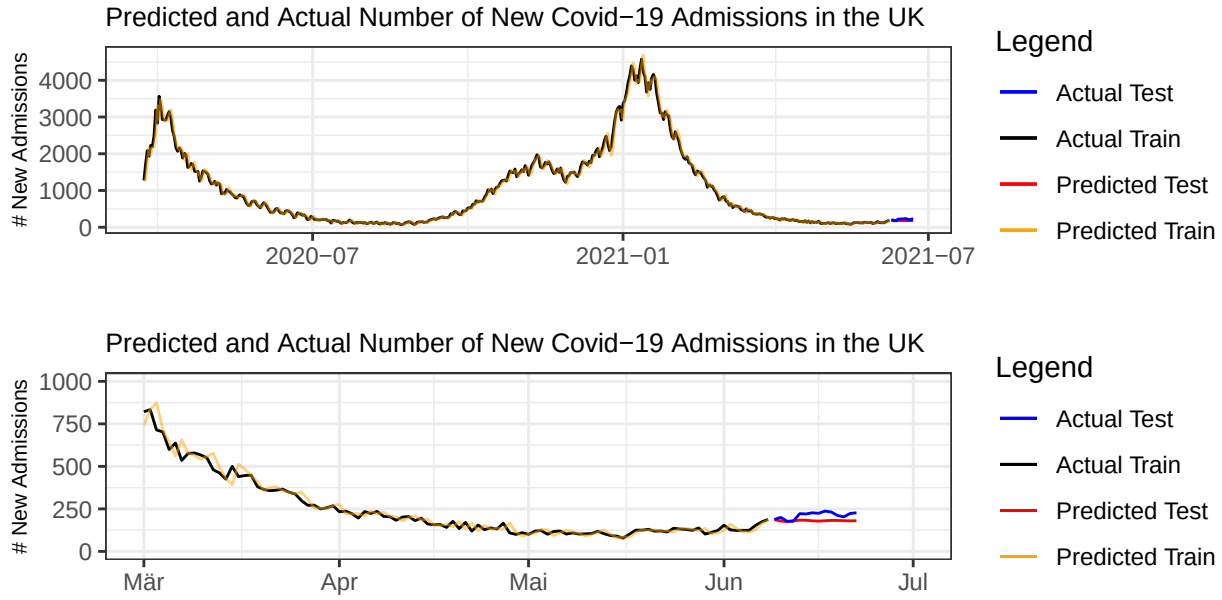
We can see that the Root Mean Squared Error (RMSE) is 117.5 for the training set. We will later compare this to the TEM and both models' performance on the test set.

Let's first look at a plot of a 14-day forecast of the ARIMA(2,1,2) model:



From the plot above, we can see multiple things. First of all, the predicted values for the 14-day interval is quite stable and only by looking at the graph very closely can we see a small oscillation. We also see that the 95% confidence interval is quite spread out and also includes negative numbers, which obviously doesn't make sense.

Let's take a closer look and plot the actual data versus the predicted values, both for the training and the test set:



From this plot, we can see that the actual values in the test set are more volatile than the predicted values. Nevertheless, the predicted values are quite close to the actual number of newly admitted patients.

The RMSE for the test set is equal to 36.5. This is much lower than the RMSE for the test set, which was 117.5. In this case, this is due to the large difference in the size of the training set (443 observations) and the test set (14 observations). The model seems to fit the data quite well, at least in terms of a 14-day forecast period. Last but not least, we can also calculate the coverage probability, i.e. in how many cases the actual number of admissions lied within the 95% confidence interval of the ARIMA(2,1,2) model. Here, all of the 14 actual values lie within this interval, which is no surprise as the 95% confidence interval was quite spread out.

3.2 Triple Exponential Model (TEM)

The second type of model we are going to use is a Triple Exponential Model (TEM). The reason why this type of model makes sense here, is because we are dealing with univariate, non-stationary data. One characteristic of the TEM, is that its forecasts are weighted averages of past observations, with a geometrically decreasing ratio.[2]

The `ets()` function from the `forecast` package provides a way of automatically selecting an appropriate TEM, if it is not further specified. The most important part of the function is the `model` parameter, which consists of a three-character string that identifies the specific model. The first letter denotes the error type ("A", "M" or "Z"); the second letter denotes the trend type ("N","A","M" or "Z"); and the third letter denotes the season type ("N","A","M" or "Z"). In all cases, "N" = none, "A" = additive, "M" = multiplicative and "Z" = automatically selected.[3] Let's take a look at the model that seems most appropriate for our training data:

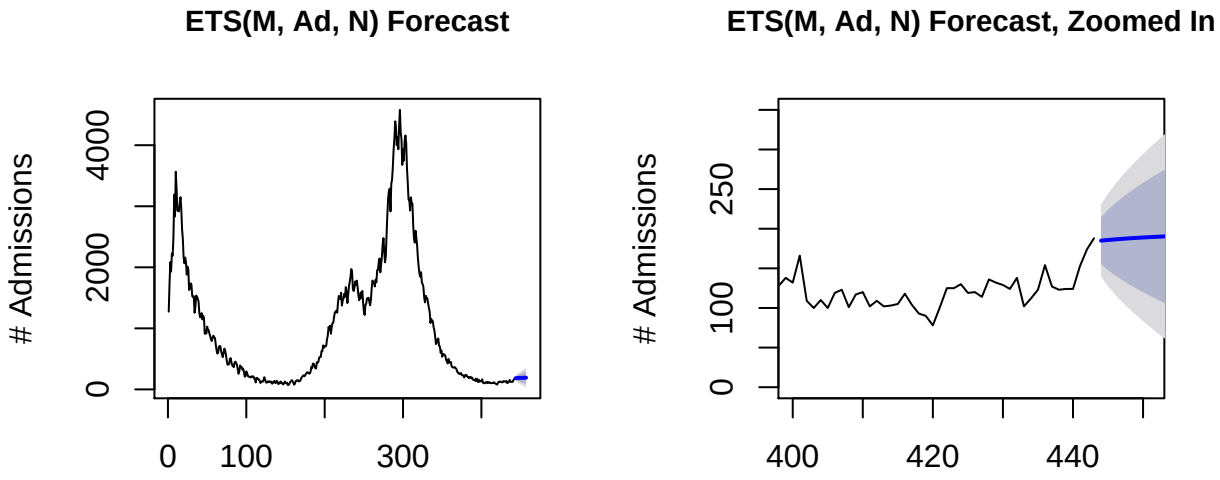
```
## ETS(M,Ad,N)
##
## Call:
##  ets(y = ts(train_df$newAdmissions))
##
##   Smoothing parameters:
##     alpha = 0.7921
##     beta  = 0.0145
##     phi   = 0.9129
##
##   Initial states:
##     l = 1280.0501
##     b = 215.4923
##
##   sigma:  0.1246
##
##      AIC     AICc      BIC
## 6487.690 6487.883 6512.252
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE     MASE      ACF1
## Training set -7.997954 123.3382 76.08606 -1.459213 9.456335 1.014695 0.1990201
```

[2]Hyndman: Forecasting - principles and practice, 2013, Page 171

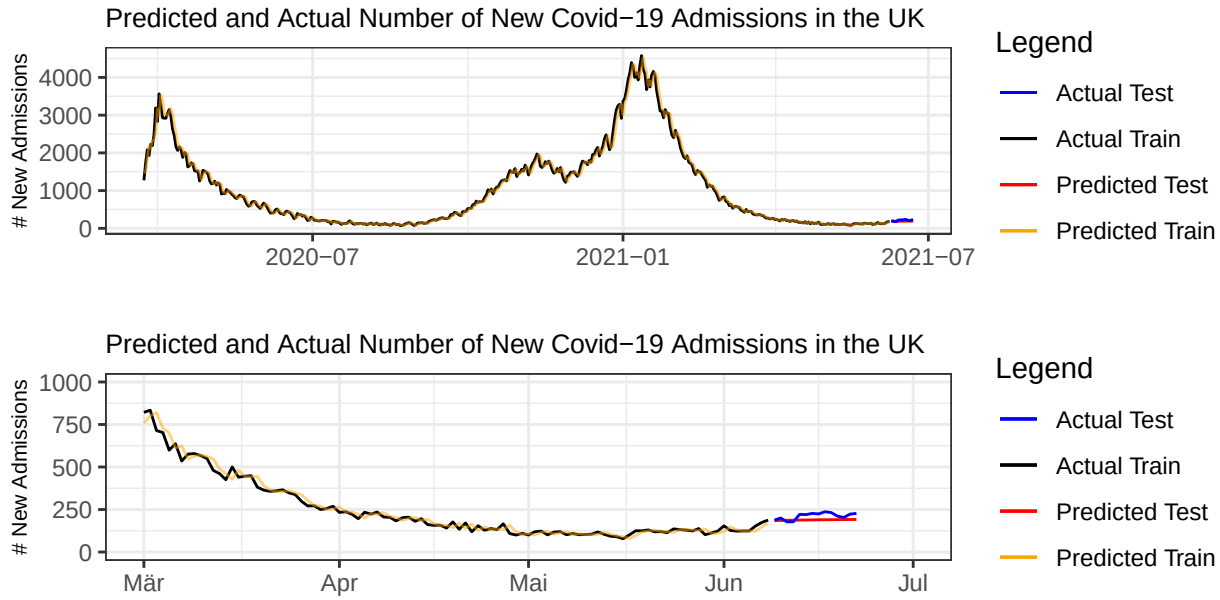
[3]Hyndman: Forecasting Functions for Time Series and Linear Models, 2021, Page 43

As we can see, the best TEM seems to be an ETS(M, Ad, N) model, which means a model with a multiplicative error, an additive trend and no seasoanlity. Both the AIC and the RMSE for the training set are higher in the TEM compared to the ARIMA model discussed above.

Let's take a look at the 14-day forecast of the ETS(M, Ad, N) model:



As we can see from the plot, the 95% confidence interval of the TEM seems to be much narrower than the one from the ARIMA model. Also, all values within the confidence interval are positive, which is also preferable. Let's also look at the actual data versus the predicted values for the training and the test set:



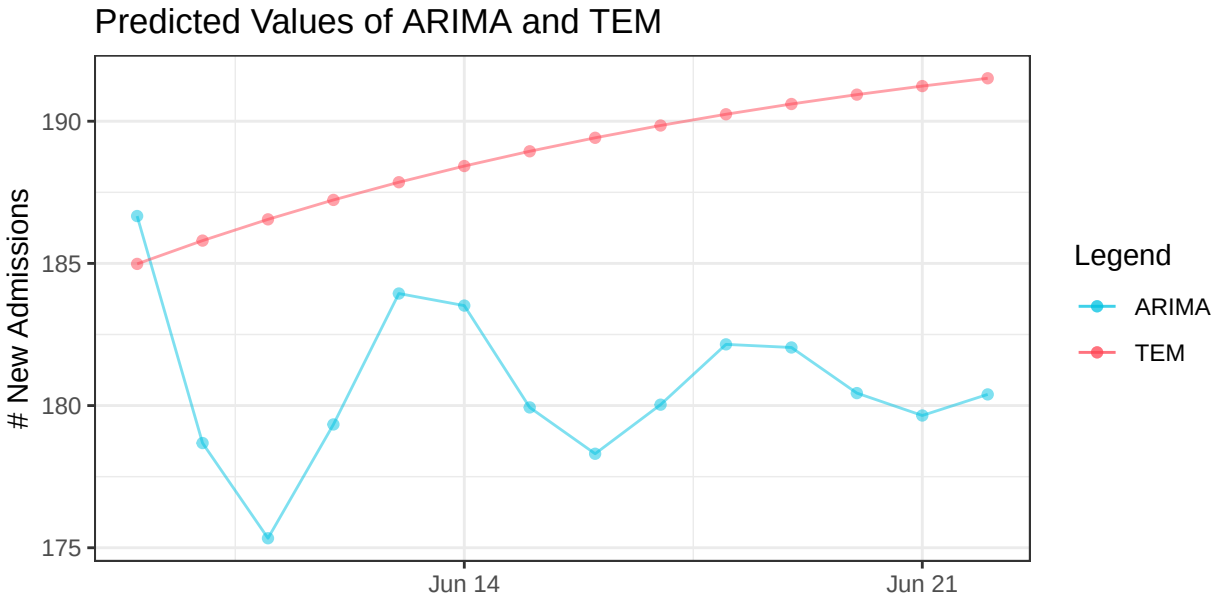
Similar as before, the predicted values are much less volatile than the actual values in the 14-day forecast.

In the case of the TEM, the RMSE for the test set is equal to 29.3. Again, this is much lower than the RMSE for the test set which was 123.3. Just as in the case of the ARIMA model, this is due to the large difference in the size of the training set (443 observations) and the test set (14 observations). All of the 14 actual values lie within the predicted 95% confidence interval, which means that the coverage probability is 100%. Since the confidence interval is much narrower than the one from the ARIMA model, this metric is equally more relevant.

Let's now move on and further discuss these results, and also look at existing limitations and suggestions for further improvement.

4. Discussion

First, we will look at a direct comparison of the 14-day forecast between the two models:



As we can see, the structure of the prediction of both models is quite different. While the TEM looks like a series with gradually decreasing growth rates, the ARIMA model is oscillating with decreasing variability.

There are three reasons why one might prefer the TEM over the ARIMA model in practice:

1. The TEM has a lower RMSE
2. The 95 % confidence interval of the TEM is narrower
3. All values in the 95 % confidence interval are positive, as should be the case for count data

Of course, there exist a number of limitations in the two forecasting methods discussed here. However, arguably the largest limitation lies within the data that we have used to predict the number of new hospital admissions of COVID-19 patients in the UK. The dataset we have worked with is univariate and only takes the past number of COVID-19 patients into account. Obviously, there are many other variables that we could include into a forecasting model that would likely significantly improve the accuracy, such as

- Data on current regulations and lockdowns
- Data on average encounters between people
- Mobility data
- Vaccination progress and distribution among the population
- PCR testing data
- Data on the emergence of new virus variants

and many more.

Apart from the data itself, there are also measurements that could be taken to improve predictions using the dataset at hand. For example, one might argue that only more recent data should be used to train the forecasting models, as the inherent structure of the pandemic today is very different from 2020. Also, we could try to further tweak the ARIMA and ETS model, for example by using other optimization methods, or by expanding the search space for different parameters. Last but not least, there are also alternative models that we could fit to our data, such as the Pattern Sequence-Based Forecasting Algorithm, better known as `PSF`. This is a forecasting method based on the assumption that there exist pattern sequences in the time series, and it was proposed in 2008.[4]

This concludes this assignment. As we have seen, we are able to perform quite accurate predictions of the COVID-19 admissions using the univariate data. However, certain limitations exist that are worth dealing with to further improve the validity of these predictions.

[4]Martínez-Álvarez et al: A Labeled-Based Forecasting Algorithm and Its Application to Electricity Price Time Series In Data Mining, 2008, Pages 453–461