Forecasting with Linear Regression

Nicholas Bradley

# Forecasting with Linear Regression

The focus of this analysis is on methods and approaches for forecasting time series data with linear regression, including methods for decomposing and forecasting the series components (for example the trend and seasonal patterns), handling special events such as outliers and holidays and using external variables as regressors. The following topics are covered:

* Forecasting approaches with linear regression models
* Extracting and estimating series components
* Handling structural breaks, outliers and special events
* Forecasting series with multiseasonality

## The linear regression

For a single independent variable (simple linear regression), the equation is:

For multiple independent variables (multiple linear regression), the equation is:

The model variables for these equations are as follows:

* **y**: This represents the dependent variable or the response variable in your dataset. It’s the variable you are trying to predict or explain.
* **x** (simple linear regression) or **x1, x2, ..., xn** (multiple linear regression)**:** These are the independent variables or predictor variables. They represent the variables that you believe may have an influence on the dependent variable y.
* : This is the intercept term in the linear regression model. It represents the value of the dependent variable y when all the independent variables are set to zero. In other words, it represents the expected value of y when all predictor variables are absent or have no effect.
* : These are the coefficients or regression coefficients. They represent the change in the dependent variable y for a one-unit change in the corresponding independent variable x (in simple linear regression) or x1, x2, ..., xn (in multiple linear regression), holding all other variables constant. They quantify the strength and direction of the relationship between each predictor variable and the response variable.
* : This is the error term or the residual term. It represents the difference between the observed value of the dependent variable y and the value predicted by the regression equation. It captures the variability in y that cannot be explained by the predictor variables included in the model. The error term accounts for factors other than the predictor variables that may affect the dependent variable. It is assumed to follow a normal distribution with mean zero.

## Forecasting with linear regression

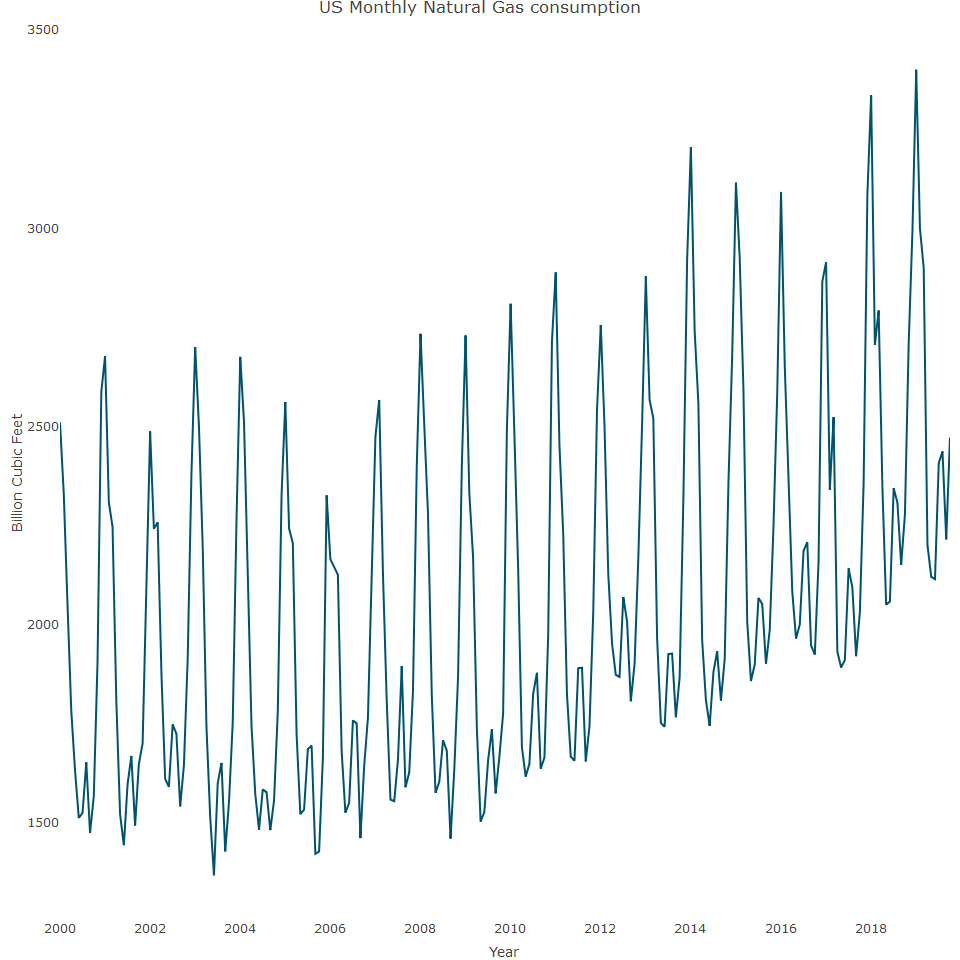
Forecasting with such a model is based on two steps:

* Identifying series sructure, key characteristics, patterns, outliers, and other features
* Transform those features into input variables and regress them with the series to create a forecasting model

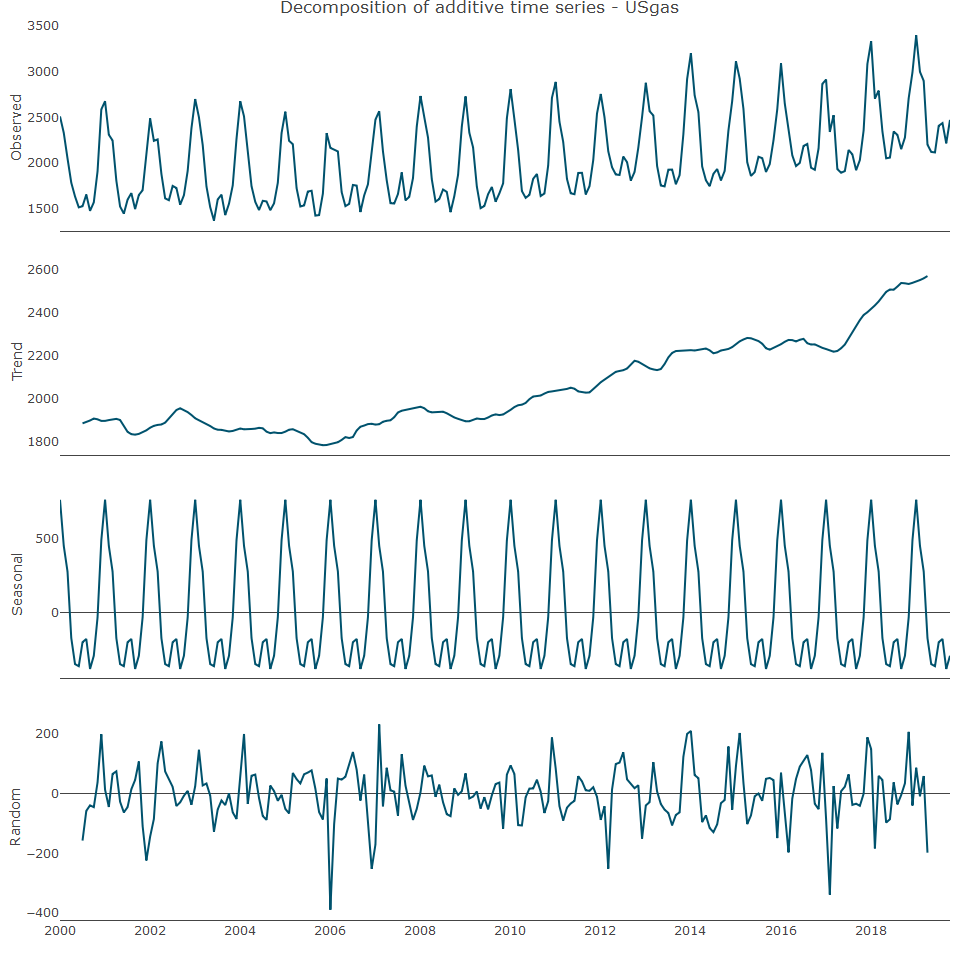
The core components of a linear regression forecasting model are the trend and seasonal components.

## Features engineering of the series components

Before creating the regression inputs that represent the series trend and seasonal components, we first have to understand their structure. I can create new features from the US gas series



As can be seen, US gas is a monthly series with a strong monthly seasonal component and fairly stable trend line. We can explore the series component structure with the ts\_decompose function further.



As seen in the above plot, the trend of the series is fairly flat between 2000 and 2010 and has a fairly linear growth moving forward. Therefore, the overall trend between 2000 and 2018 is not strictly linear. This important insight will help to define the trend output for the regression model. Before constructing a linear model, we first need to transform the data from a time series into a data frame object

## ds y  
## 1 2000-01-01 2510.5  
## 2 2000-02-01 2330.7  
## 3 2000-03-01 2050.6  
## 4 2000-04-01 1783.3  
## 5 2000-05-01 1632.9  
## 6 2000-06-01 1513.1

We can now start to create the regression input features, starting with a series trend variable. A basic approach is to index the series observations chronologically.

We can also create a seasonal component variable. As we want to measure the contribution of each frequency unit to the oscilliation of the series, we will use a categorical variable for each frequency unit. In terms of the US gas series, the frequency units represent the months of the year and so, we create a categorical variable with 12 categories, each category corresponding to a specific month.

## ds y trend seasonal  
## 1 2000-01-01 2510.5 1 Jan  
## 2 2000-02-01 2330.7 2 Feb  
## 3 2000-03-01 2050.6 3 Mar  
## 4 2000-04-01 1783.3 4 Apr  
## 5 2000-05-01 1632.9 5 May  
## 6 2000-06-01 1513.1 6 Jun

It is also neccesary to split the series into training and test partitions and the last 12 months can be used as the test data

## Modelling the series trend and seasonal components

We can first model the series trend by regressing the series with the trend variable on the training data

##   
## Call:  
## lm(formula = y ~ trend, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -547.2 -307.4 -153.2 333.1 1052.6   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1751.0074 52.6435 33.26 < 2e-16 \*\*\*  
## trend 2.4489 0.4021 6.09 4.86e-09 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 394.4 on 224 degrees of freedom  
## Multiple R-squared: 0.1421, Adjusted R-squared: 0.1382   
## F-statistic: 37.09 on 1 and 224 DF, p-value: 4.861e-09

The summary reveals that the trend variable is very statistically significant in terms of y variable. It is 0.00000000486, which is much more statistically significant than the 0.05 or 0.01 thresholds.

It is important to add further context to this model through visualisation and we can create a plot that indicates the actual values, predicted values and forecasted values

The function plot\_lm is designed to create an interactive plot using the plotly library in R. This plot visualizes the actual data, the fitted values from a linear regression model, and the forecasted values for a time series dataset. Here’s a breakdown of what the function does:

Function Definition and Parameters Function Name: plot\_lm Parameters: - data: The complete dataset containing the actual values. - train: The training dataset containing the fitted values (yhat). - test: The test dataset containing the forecasted values (yhat). - title (optional): A title for the plot.

Plot Creation Initialize Plot:

* plot\_ly initializes a plotly object using the data parameter.
* x = ~ ds sets the x-axis to the ds column (which likely contains date or time information).
* y = ~ y sets the y-axis to the y column (which contains the actual values).
* type = “scatter” and mode = “line” specify that the plot will be a line plot.
* name = “Actual” labels this trace as “Actual”.

Add Fitted Values:

* add\_lines adds a new line to the plot.
* x = ~ train$ds uses the ds column from the train dataset for the x-axis.
* y = ~ train$yhat uses the yhat column from the train dataset for the y-axis, which contains the fitted values.
* line = list(color = “red”) specifies the line color as red.
* name = “Fitted” labels this trace as “Fitted”.

Add Forecasted Values:

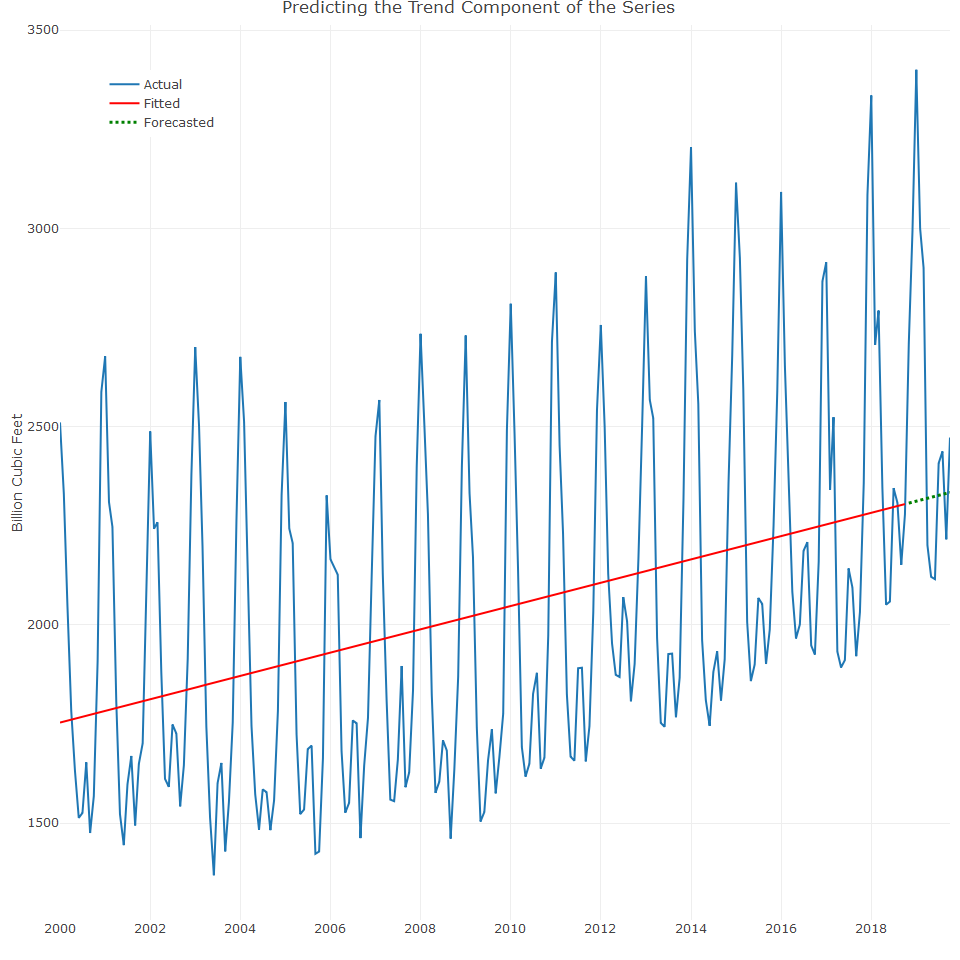
* Another add\_lines adds a new line for the forecasted values.
* x = ~ test$ds uses the ds column from the test dataset for the x-axis.
* y = ~ test$yhat uses the yhat column from the test dataset for the y-axis, which contains the forecasted values.
* line = list(color = “green”, dash = “dot”, width = 3) specifies the line color as green, with a dotted style and width of 3.
* name = “Forecasted” labels this trace as “Forecasted”.

Layout Customization:

* layout sets the overall layout of the plot.
* title = title sets the plot title to the provided title parameter.
* xaxis = list(title = ““) leaves the x-axis title empty.
* yaxis = list(title = “Billion Cubic Feet”) sets the y-axis title to “Billion Cubic Feet”.
* legend = list(x = 0.05, y = 0.95) positions the legend at the specified coordinates within the plot.

Return Value

* The function returns the constructed plotly object p.



Overall, the model was able to capture the general movement of the trend, yet a linear trend may fail to capture the structural break of the trend that occurred around 2010. By means of comparison analysis, we can measure the error rate both for the training and test data sets

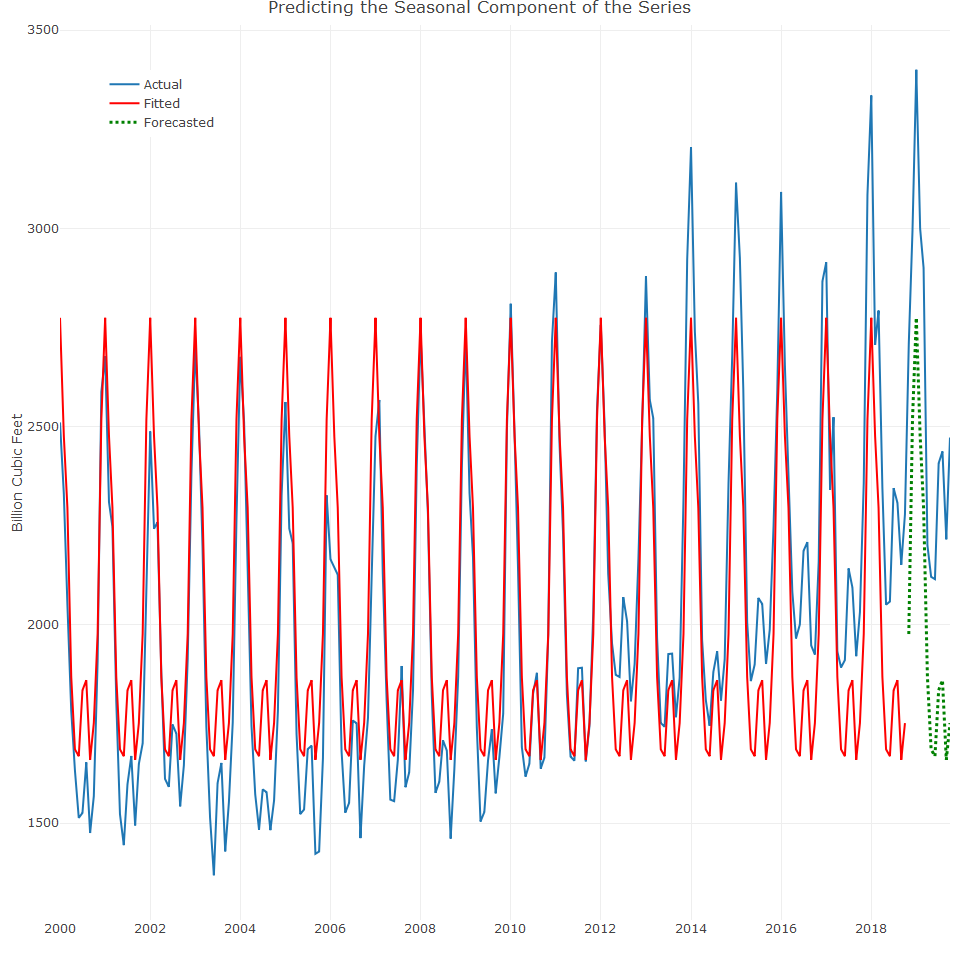
## [1] 0.1644088 0.1299951

We can now model the seasonal component

##   
## Call:  
## lm(formula = y ~ seasonal, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -608.98 -162.34 -50.77 148.40 566.89   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2774.28 49.75 55.759 < 2e-16 \*\*\*  
## seasonalFeb -297.92 70.36 -4.234 3.41e-05 \*\*\*  
## seasonalMar -479.10 70.36 -6.809 9.77e-11 \*\*\*  
## seasonalApr -905.28 70.36 -12.866 < 2e-16 \*\*\*  
## seasonalMay -1088.42 70.36 -15.468 < 2e-16 \*\*\*  
## seasonalJun -1105.49 70.36 -15.711 < 2e-16 \*\*\*  
## seasonalJul -939.35 70.36 -13.350 < 2e-16 \*\*\*  
## seasonalAug -914.12 70.36 -12.991 < 2e-16 \*\*\*  
## seasonalSep -1114.74 70.36 -15.843 < 2e-16 \*\*\*  
## seasonalOct -1022.21 70.36 -14.527 < 2e-16 \*\*\*  
## seasonalNov -797.53 71.33 -11.180 < 2e-16 \*\*\*  
## seasonalDec -256.67 71.33 -3.598 0.000398 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 216.9 on 214 degrees of freedom  
## Multiple R-squared: 0.7521, Adjusted R-squared: 0.7394   
## F-statistic: 59.04 on 11 and 214 DF, p-value: < 2.2e-16

As seen, all the models coefficients are statistically significant. An Adjusted R-squared value of 0.7394 means that approximately 73.94% of the variance in the dependent variable (the variable you are trying to predict or explain) is accounted for by the independent variables (the predictors) included in the model. Before we plot the fitted model and forecast values, let’s update the yhat values with the predict function

Now we can plot it



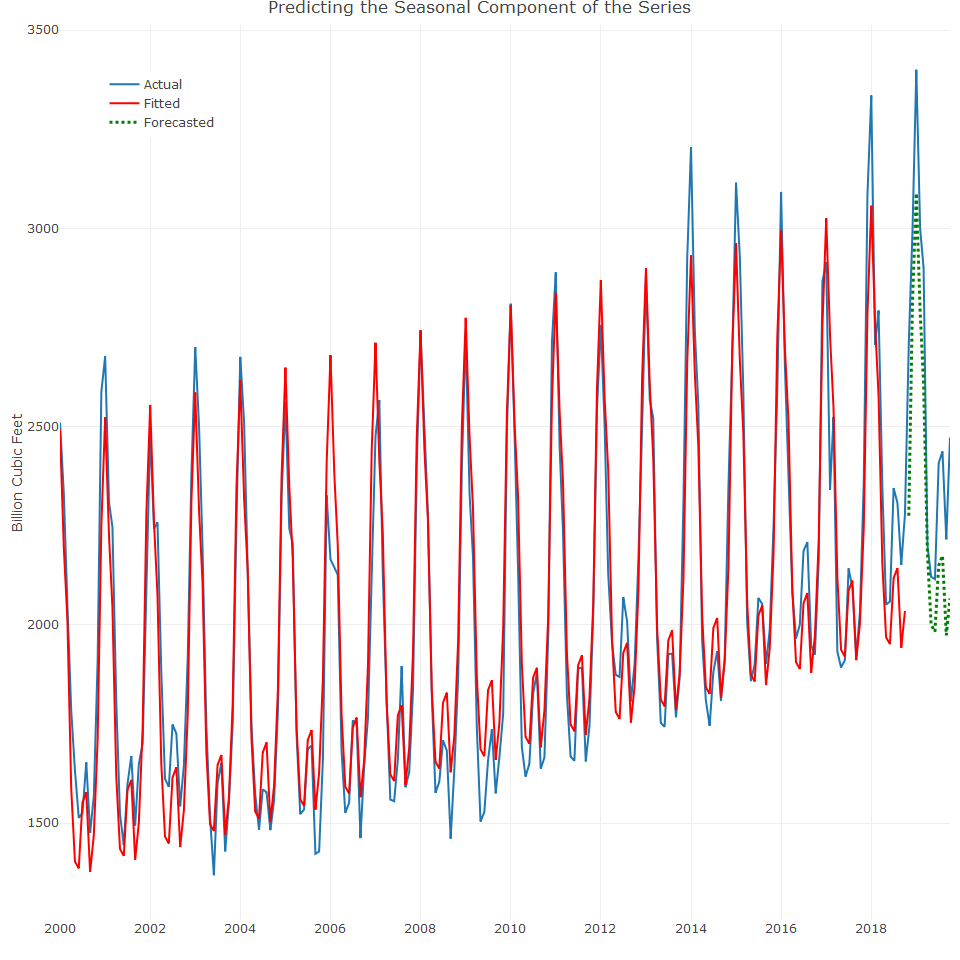
As can be seen, the model does a good job of capturing the structure of the series seasonal pattern. However, as evident, the series trend is missing, but before we add the trend and seasonal components, let’s score the model performance

## [1] 0.08628973 0.21502100

The high error rate on the test data is related to the trend component that was not included in the model. The next step is to join the two components into one model and to forecast the feature values of the series

##   
## Call:  
## lm(formula = y ~ seasonal + trend, data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -514.73 -77.17 -17.70 85.80 336.95   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2488.8994 32.6011 76.344 < 2e-16 \*\*\*  
## seasonalFeb -300.5392 41.4864 -7.244 7.84e-12 \*\*\*  
## seasonalMar -484.3363 41.4870 -11.674 < 2e-16 \*\*\*  
## seasonalApr -913.1334 41.4880 -22.010 < 2e-16 \*\*\*  
## seasonalMay -1098.8884 41.4895 -26.486 < 2e-16 \*\*\*  
## seasonalJun -1118.5855 41.4913 -26.960 < 2e-16 \*\*\*  
## seasonalJul -955.0563 41.4936 -23.017 < 2e-16 \*\*\*  
## seasonalAug -932.4482 41.4962 -22.471 < 2e-16 \*\*\*  
## seasonalSep -1135.6874 41.4993 -27.366 < 2e-16 \*\*\*  
## seasonalOct -1045.7687 41.5028 -25.198 < 2e-16 \*\*\*  
## seasonalNov -808.0016 42.0617 -19.210 < 2e-16 \*\*\*  
## seasonalDec -269.7642 42.0635 -6.413 9.05e-10 \*\*\*  
## trend 2.6182 0.1305 20.065 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 127.9 on 213 degrees of freedom  
## Multiple R-squared: 0.9142, Adjusted R-squared: 0.9094   
## F-statistic: 189.2 on 12 and 213 DF, p-value: < 2.2e-16

The adjusted-squared value is 0.9094, which means that nealy 91% of the variance in the data is explained by the predictors, so the model fits the data very well, which can be seen from the plot of the model



The plot indicates that the model trend is to linear and missing the structural break of the series trend. This is the point where adding a polynomial component for the model could potentially further improve model accuracy

We can check the error scores

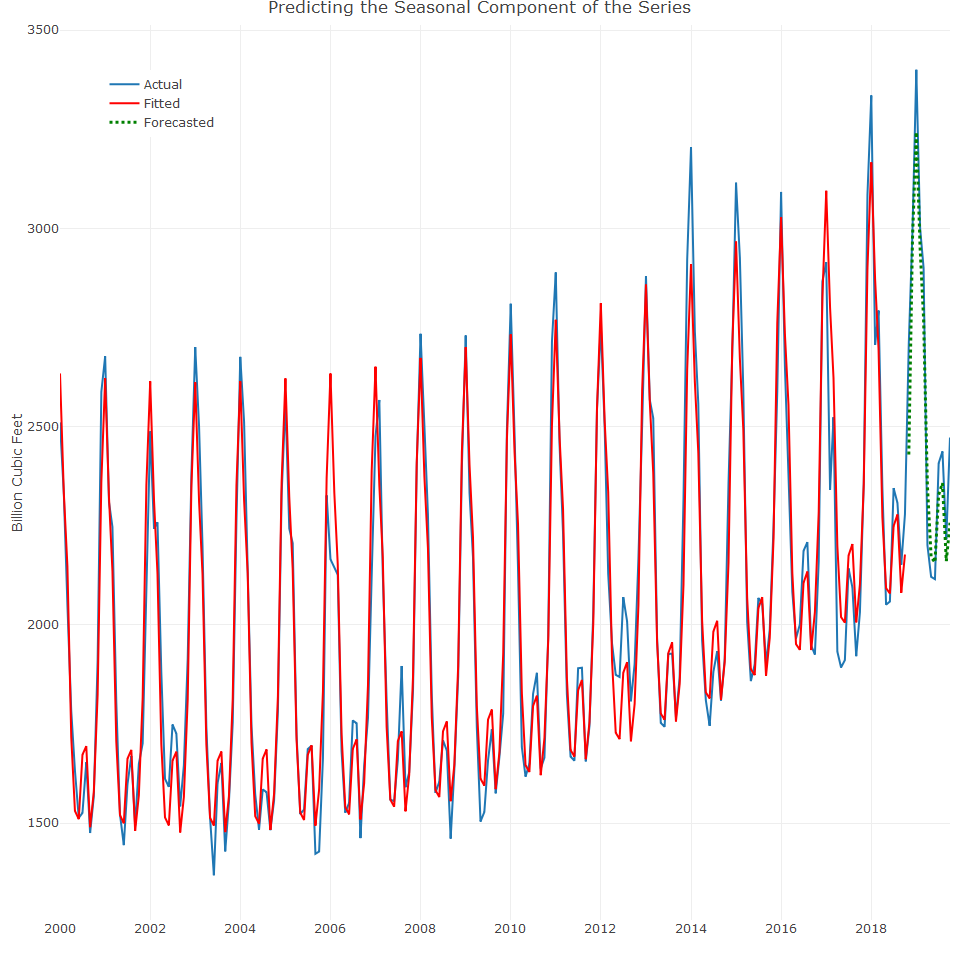
## [1] 0.04774945 0.09143290

Mean Absolute Percentage Error (MAPE) measures the accuracy of a forecasting model by comparing the absolute errors between the predicted and actual values as a percentage of the actual values. A MAPE of 4.77% suggests a relatively high level of accuracy in the forecasting model, indicating that it closely aligns with the training data. A MAPE of 9.14% suggests a moderate level of accuracy in the forecasting model. While it is not as accurate as the first dataset, it still provides reasonably good predictions for the test data.

We can fit the polynomial term

##   
## Call:  
## lm(formula = y ~ seasonal + trend + I(trend^2), data = train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -468.47 -54.66 -2.21 63.11 294.32   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.635e+03 3.224e+01 81.738 < 2e-16 \*\*\*  
## seasonalFeb -3.004e+02 3.540e+01 -8.487 3.69e-15 \*\*\*  
## seasonalMar -4.841e+02 3.540e+01 -13.676 < 2e-16 \*\*\*  
## seasonalApr -9.128e+02 3.540e+01 -25.787 < 2e-16 \*\*\*  
## seasonalMay -1.099e+03 3.540e+01 -31.033 < 2e-16 \*\*\*  
## seasonalJun -1.118e+03 3.540e+01 -31.588 < 2e-16 \*\*\*  
## seasonalJul -9.547e+02 3.540e+01 -26.968 < 2e-16 \*\*\*  
## seasonalAug -9.322e+02 3.541e+01 -26.329 < 2e-16 \*\*\*  
## seasonalSep -1.136e+03 3.541e+01 -32.070 < 2e-16 \*\*\*  
## seasonalOct -1.046e+03 3.541e+01 -29.532 < 2e-16 \*\*\*  
## seasonalNov -8.001e+02 3.590e+01 -22.286 < 2e-16 \*\*\*  
## seasonalDec -2.618e+02 3.590e+01 -7.293 5.95e-12 \*\*\*  
## trend -1.270e+00 4.472e-01 -2.840 0.00494 \*\*   
## I(trend^2) 1.713e-02 1.908e-03 8.977 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 109.1 on 212 degrees of freedom  
## Multiple R-squared: 0.9379, Adjusted R-squared: 0.9341   
## F-statistic: 246.1 on 13 and 212 DF, p-value: < 2.2e-16

Adding the polynomial term did not significantly improve the goodness of fit. However, as seen in the plot below, it did capture the structural break of the trend over time



## [1] 0.03688770 0.04212618

These MAPE scores are very good for the training and test data sets

## The tslm function

This function automatically transforms a time series object into a data frame and there are several advantages:

* Efficiency - it does not require transforming the object to a data frame and feature engineering
* The output object supports all of the functionality of the forecast and TSstudio packages

##   
## Call:  
## tslm(formula = train.ts ~ season + trend + I(trend^2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -468.47 -54.66 -2.21 63.11 294.32   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.635e+03 3.224e+01 81.738 < 2e-16 \*\*\*  
## season2 -3.004e+02 3.540e+01 -8.487 3.69e-15 \*\*\*  
## season3 -4.841e+02 3.540e+01 -13.676 < 2e-16 \*\*\*  
## season4 -9.128e+02 3.540e+01 -25.787 < 2e-16 \*\*\*  
## season5 -1.099e+03 3.540e+01 -31.033 < 2e-16 \*\*\*  
## season6 -1.118e+03 3.540e+01 -31.588 < 2e-16 \*\*\*  
## season7 -9.547e+02 3.540e+01 -26.968 < 2e-16 \*\*\*  
## season8 -9.322e+02 3.541e+01 -26.329 < 2e-16 \*\*\*  
## season9 -1.136e+03 3.541e+01 -32.070 < 2e-16 \*\*\*  
## season10 -1.046e+03 3.541e+01 -29.532 < 2e-16 \*\*\*  
## season11 -8.001e+02 3.590e+01 -22.286 < 2e-16 \*\*\*  
## season12 -2.618e+02 3.590e+01 -7.293 5.95e-12 \*\*\*  
## trend -1.270e+00 4.472e-01 -2.840 0.00494 \*\*   
## I(trend^2) 1.713e-02 1.908e-03 8.977 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 109.1 on 212 degrees of freedom  
## Multiple R-squared: 0.9379, Adjusted R-squared: 0.9341   
## F-statistic: 246.1 on 13 and 212 DF, p-value: < 2.2e-16

## Modelling single events and non seasonal events

In some cases, time series data may contain unusual patterns that are either re-occuring over time or not, including the following:

* **Outliers:** A single event or events that are out of the normal patterns of the series
* **Structural break:** A significant event that changes the historical pattern of the series. A common example is a change in the growth of the series.
* **Non-seasonal re-occurring events:** An event that repeats from cycle to cycle, but the time at which they occur changes from cycle to cycle. A common example of such an event is the Easter holidays.

To capture the structural break around the year 2010, we can create binary variable where rows before 2010 are indicated with the value 0 and those after are given the value of 1.

##   
## Call:  
## tslm(formula = USgas ~ season + trend + I(trend^2) + s\_break,   
## data = USgas\_df)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -469.25 -50.68 -2.66 63.63 275.89   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.661e+03 3.200e+01 83.164 < 2e-16 \*\*\*  
## season2 -3.054e+02 3.448e+01 -8.858 2.61e-16 \*\*\*  
## season3 -4.849e+02 3.448e+01 -14.062 < 2e-16 \*\*\*  
## season4 -9.272e+02 3.449e+01 -26.885 < 2e-16 \*\*\*  
## season5 -1.108e+03 3.449e+01 -32.114 < 2e-16 \*\*\*  
## season6 -1.127e+03 3.450e+01 -32.660 < 2e-16 \*\*\*  
## season7 -9.568e+02 3.450e+01 -27.730 < 2e-16 \*\*\*  
## season8 -9.340e+02 3.451e+01 -27.061 < 2e-16 \*\*\*  
## season9 -1.138e+03 3.452e+01 -32.972 < 2e-16 \*\*\*  
## season10 -1.040e+03 3.453e+01 -30.122 < 2e-16 \*\*\*  
## season11 -7.896e+02 3.497e+01 -22.577 < 2e-16 \*\*\*  
## season12 -2.649e+02 3.498e+01 -7.571 9.72e-13 \*\*\*  
## trend -1.928e+00 4.479e-01 -4.304 2.51e-05 \*\*\*  
## I(trend^2) 1.862e-02 1.676e-03 11.113 < 2e-16 \*\*\*  
## s\_break 6.060e+01 2.836e+01 2.137 0.0337 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 109 on 223 degrees of freedom  
## Multiple R-squared: 0.9423, Adjusted R-squared: 0.9387   
## F-statistic: 260.3 on 14 and 223 DF, p-value: < 2.2e-16

As seen in the above output, the structural break variable has significance of 0.03

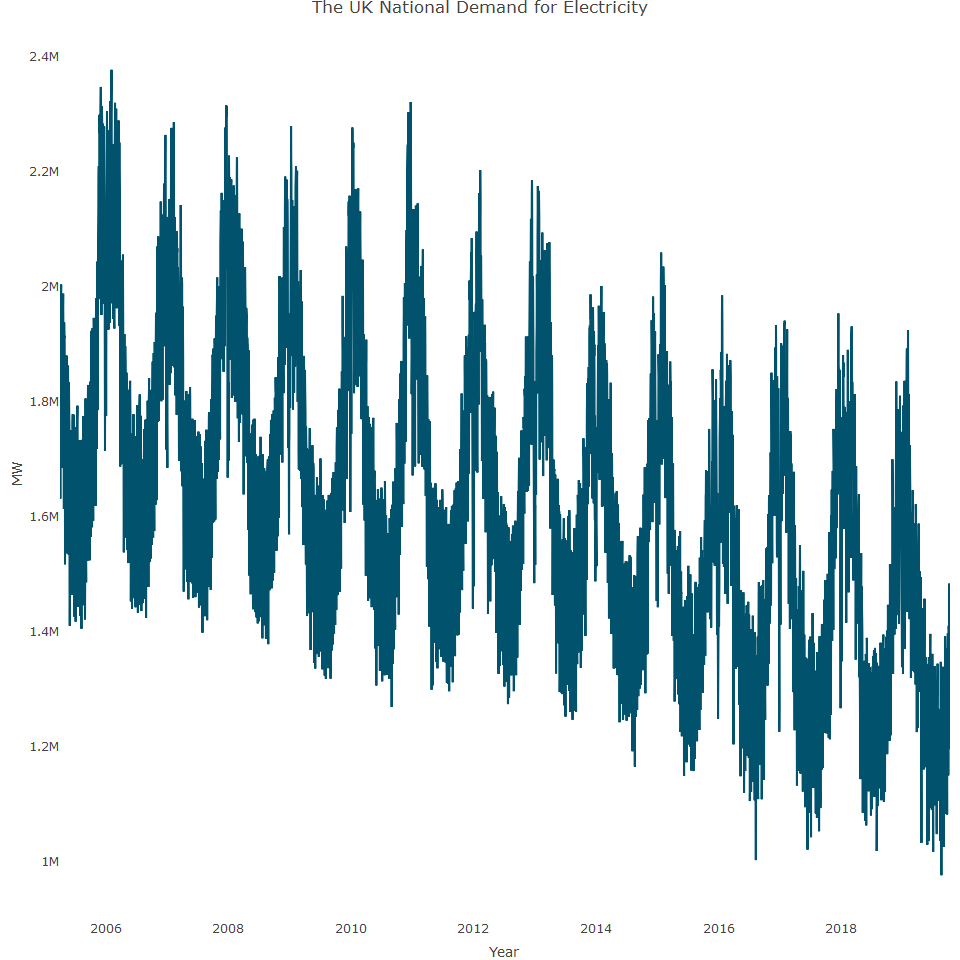
## Forecasting a series with multiseasonality components

Linear regression has advantages over time series models like ARIMA or Holt Winters as it provides various customization options and enables complex time series to be modelled, such as that with multiseasonality. We can use the UK Grid time series in the modelling

## The UKgrid series is a xts object with 1 variable and 5304 observations  
## Frequency: daily   
## Start time: 2005-04-01   
## End time: 2019-10-08

## [,1]  
## 2005-04-01 1920069  
## 2005-04-02 1674699  
## 2005-04-03 1631352  
## 2005-04-04 1916693  
## 2005-04-05 1952082  
## 2005-04-06 1964584

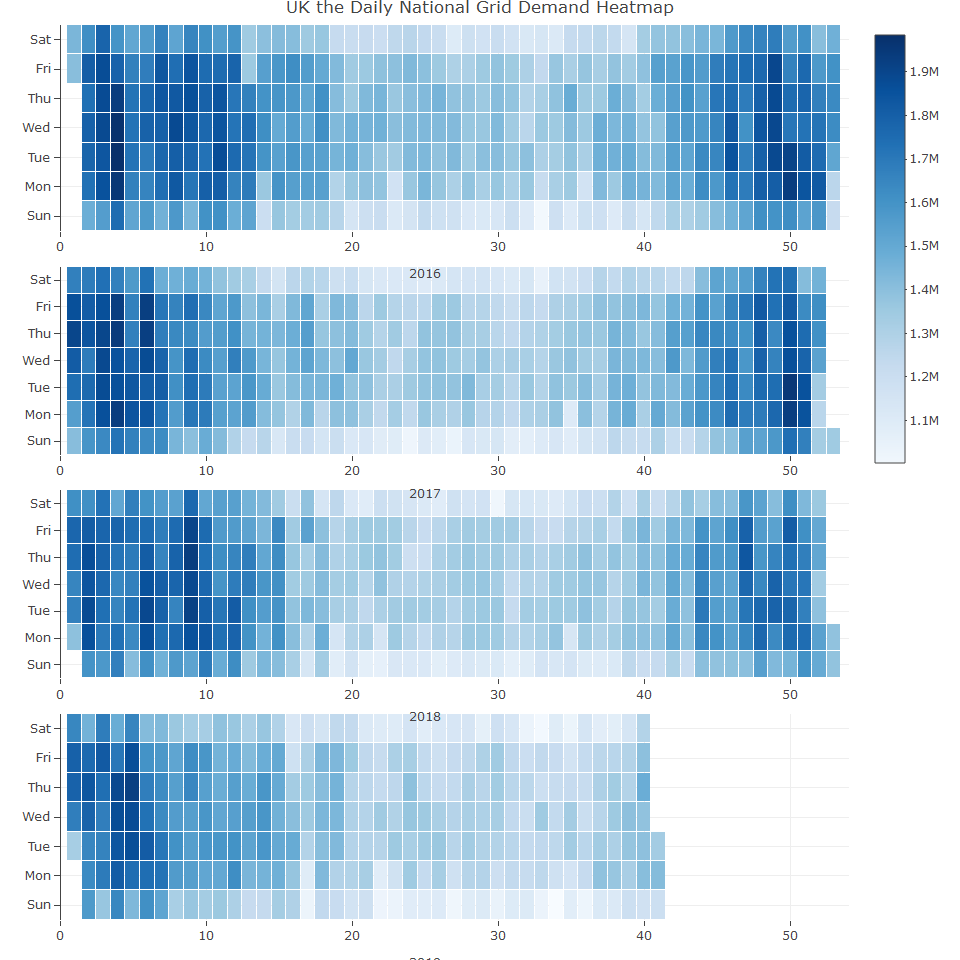
We can plot it



As evident, there is a clea downward trend and there is a strong seasonal pattern.It also has multiple seasonality patterns:

* **Daily:** A cycle of 365 days a yea
* **Day of the week:** A seven day cycle
* **Monthly:** Effected from the weather

Evidence for these can be seen in the below heatmap



As evident, the overall demand increases throughut the winter weeks (calendar weeks 1-12 and 44-52). In addition,the demand increases during the working days of the week, and decreases over the weekend

## Preprocessing and feature engineering of the UK grid series

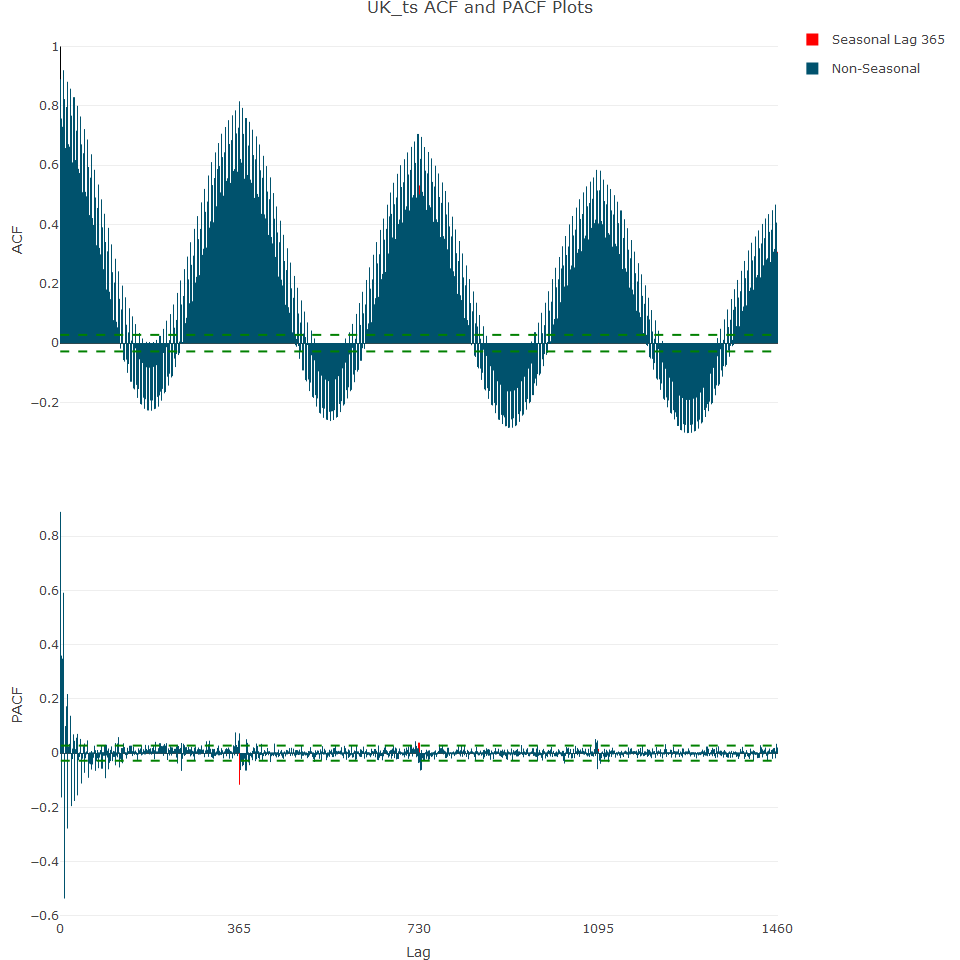
In order to capture the seasonal components of the series, we will set the series as daily frequency and create the following variables

* Day of the week indicator
* Month of the year indicator

It is also reasonable to assume that the series has a strong correlation with the seasonal lags, and so we create a lag variable of 365 observations

## 'data.frame': 4939 obs. of 5 variables:  
## $ TIMESTAMP: Date, format: "2006-04-01" "2006-04-02" ...  
## $ ND : num 1718405 1691341 1960325 2023886 2026204 ...  
## $ wday : Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<..: 7 1 2 3 4 5 6 7 1 2 ...  
## $ month : Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 4 4 4 4 4 4 4 4 4 4 ...  
## $ lag365 : num 1920069 1674699 1631352 1916693 1952082 ...

The tslm function requires a time series object, so we convert it and review the series correlation with its lags from the past four years.



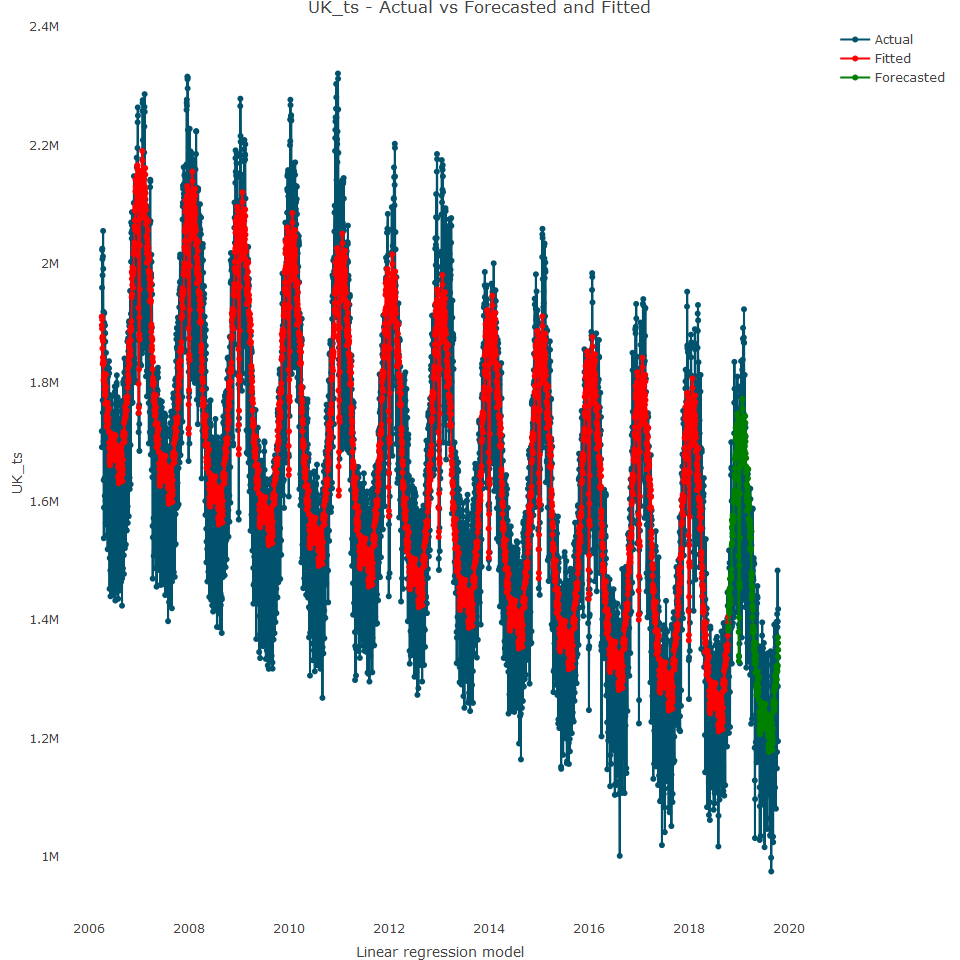
The ACF plot shows the series has a strong relationship with the seasonal lags, in particular, lag 365, the first lag. We can now split the input series into training and test data sets. The goal is to forecast the next 365 observations.

## Training and testing the forecasting model

We can use the training to train the following models:

* **Baseline Model:** Regressing the series with the seasonal and trend components. As we set the series frequency to 365, the seasonal feature of the series refers to the daily seasonality.
* **Multiseasonal model:** Adding the day of the week and month of the year indicators for capturing the multiseasonality of the series
* **A multiseasonal model with a seasonal lag:** Using, in addition to the seasonal indicators, the seasonal lag variable.

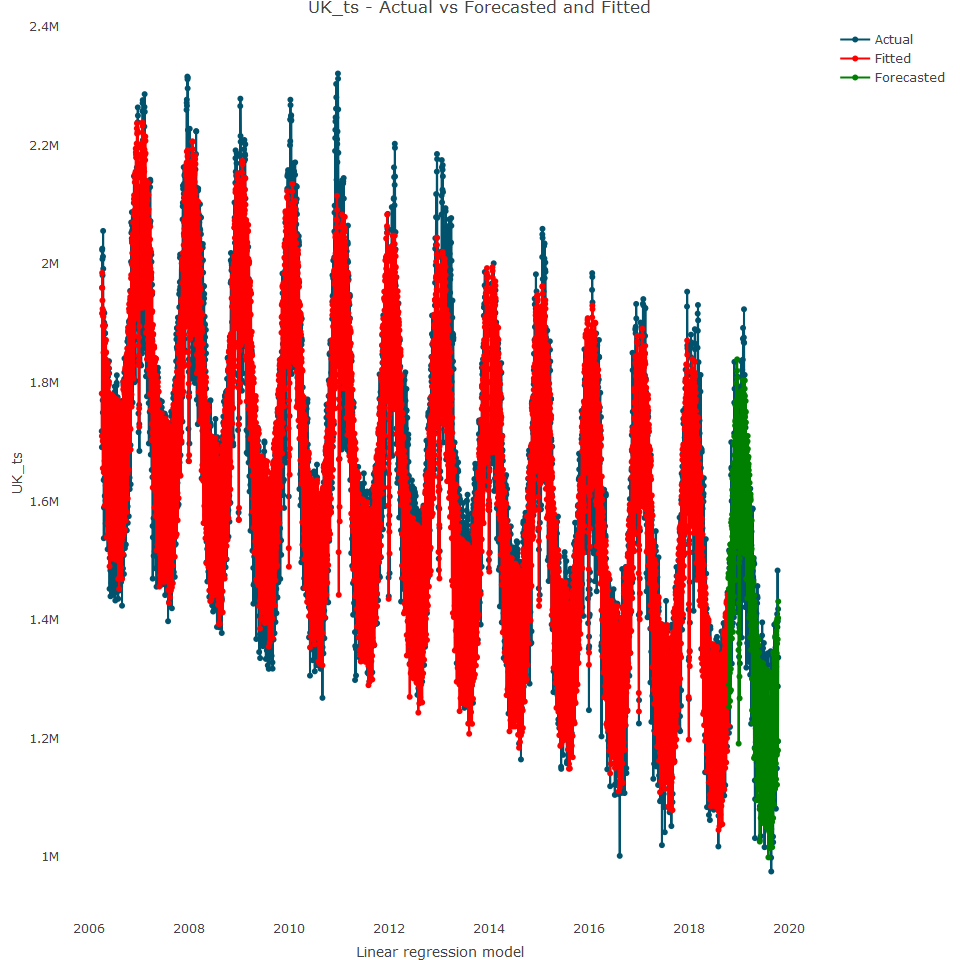
We start with the baseline model, regressing the series with its seasonal and trend components:



The plot reveals that the baseline model is doing a great job of capturing the series trend and the day of the year seasonality. However, it fails to capture the oscillation that related to the day of the week.

## ME RMSE MAE MPE MAPE MASE  
## Training set -4.781286e-12 121551.4 100439.64 -0.5721337 6.294111 0.8418677  
## Test set -1.740215e+04 123156.6 96785.27 -1.8735336 7.160573 0.8112374  
## ACF1 Theil's U  
## Training set 0.5277884 NA  
## Test set 0.5106681 1.071899

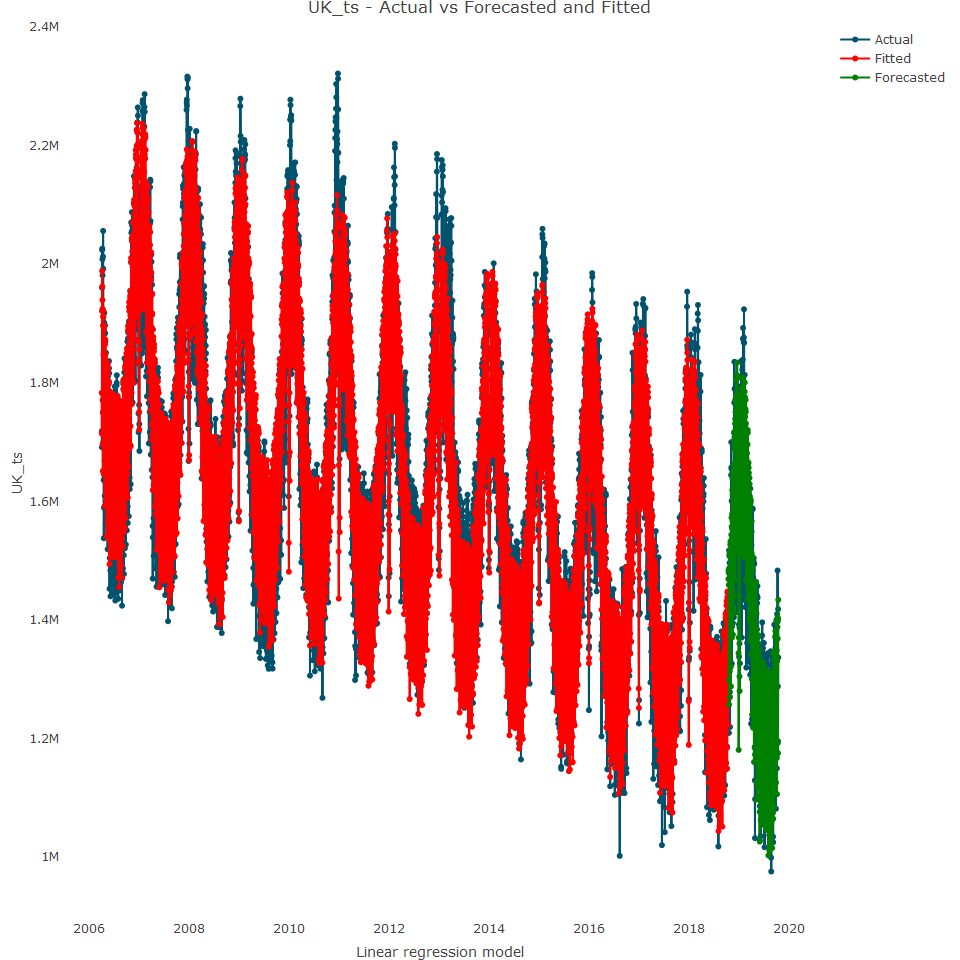
The MAPE score is 6.29% and 7.16% for the train and test set respectively. We can try to improve the accuracy by adding the day of the week and the month of the year to the model



This model has captured the trend and multiseasonality of the series, which can also e observed by looking at the MAPE scores which have dropped to 3.16% and 4.68% for training and test respectively.

## ME RMSE MAE MPE MAPE MASE  
## Training set 8.172823e-12 70245.98 52146.79 -0.1738708 3.167605 0.4370853  
## Test set -1.764563e+04 80711.71 65373.21 -1.3715505 4.682071 0.5479470  
## ACF1 Theil's U  
## Training set 0.7513664 NA  
## Test set 0.6075598 0.68445

Let’s now add the lag variable to the model:



It’s hard to see from that plot alone any improvement from the second model, so let’s review the MAPE scores. The MAPE score for the training is more or less the same, while for test, the score is slightly worse.

## ME RMSE MAE MPE MAPE MASE  
## Training set -9.836939e-13 69904.92 52006.75 -0.1717563 3.163385 0.4359116  
## Test set -1.754061e+04 81783.55 65957.66 -1.3613252 4.722083 0.5528457  
## ACF1 Theil's U  
## Training set 0.7500146 NA  
## Test set 0.6094666 0.6925767

## Model selection

We need to decide between the second and third model. The first question is to ask whether the lag variable in the third model is statistically significant?

## Estimate Std. Error t value Pr(>|t|)  
## lag365 -0.06038328 0.01545495 -3.907051 9.490321e-05

The p-value is statistically significant. We can further check for statistical significance by running an ANOVA test

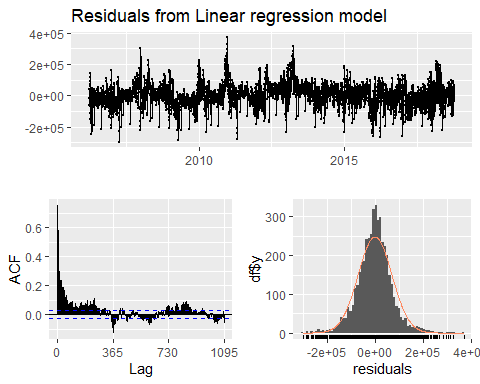
## Analysis of Variance Table  
##   
## Response: train\_ts  
## Df Sum Sq Mean Sq F value Pr(>F)   
## season 364 1.4279e+14 3.9227e+11 73.5348 < 2.2e-16 \*\*\*  
## trend 1 7.2634e+13 7.2634e+13 13615.7078 < 2.2e-16 \*\*\*  
## wday 6 4.5009e+13 7.5016e+12 1406.2214 < 2.2e-16 \*\*\*  
## month 11 1.3721e+11 1.2473e+10 2.3382 0.007266 \*\*   
## lag365 1 8.1432e+10 8.1432e+10 15.2650 9.49e-05 \*\*\*  
## Residuals 4190 2.2352e+13 5.3345e+09   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The ANOVA test also reveals it as significant

It may be that the second model is more accurate (has a better MAPE score) just by chance. Therefore, the backtesting of both models could help to validate one over the other. We select the final model as the second model:

## Residuals Analysis

Just before we finalise the model, let’s analyse the model residuals



##   
## Breusch-Godfrey test for serial correlation of order up to 730  
##   
## data: Residuals from Linear regression model  
## LM test = 3141.8, df = 730, p-value < 2.2e-16

Some autocorrelation exists between the residuals series and their lags, which indicates that the model did not capture all the patterns or information that exists in the series. One way to address this is to identify additional variables that can explain the variation in the residuals,but it can be hard to identify additional variables.

## Finalizing the forecast

Lets finalize the model and forecast the future 365 observations.

