Project Report

Name : Sushant Ravva, Btech IIT Jodhpur

Summer Project @INCOIS

DATE : 20/06/2023 - 31/07/2023

TIME SERIES ANALYSIS USING ARIMA, SARIMA, and SARIMAX MODEL AND OPTIMISING IT USING PARTICLE SWARM OPTIMIZATION

Part - I (Air Passenger Dataset)

AIM AND DATASET

The objective of the project is to perform a time series analysis on the Airline Passenger dataset and forecast the number of passengers for the next 10 years using models like **ARIMA**, **SARIMA,** and **SARIMAX**. We shall also try to use PSO-trained Artificial Neural Networks.

We are going to use the Air Passenger Dataset. This dataset contains the number of air travel passengers from the start of 1949 to the end of 1960. This dataset has a **positive trend** and **seasonality**.

Time Series

A time series is a data sequence ordered (or indexed) by time. It is discrete, and the interval between each point is constant.

Stationarity

In the context of time series analysis, stationarity refers to a property of a time series where its statistical properties remain constant over time. A stationary dataset has constant mean, constant

variance, and constant auto covariance. When a time series is stationary, it allows for more reliable and accurate forecasting, as the underlying patterns and relationships in the data are expected to remain consistent in the future. If a time series is not stationary, it can exhibit trends, seasonality, or other forms of non-stationarity. In such cases, it is often necessary to transform or preprocess the data to achieve stationarity before applying certain time series models or analysis techniques.

Rolling Mean and Standard Deviation

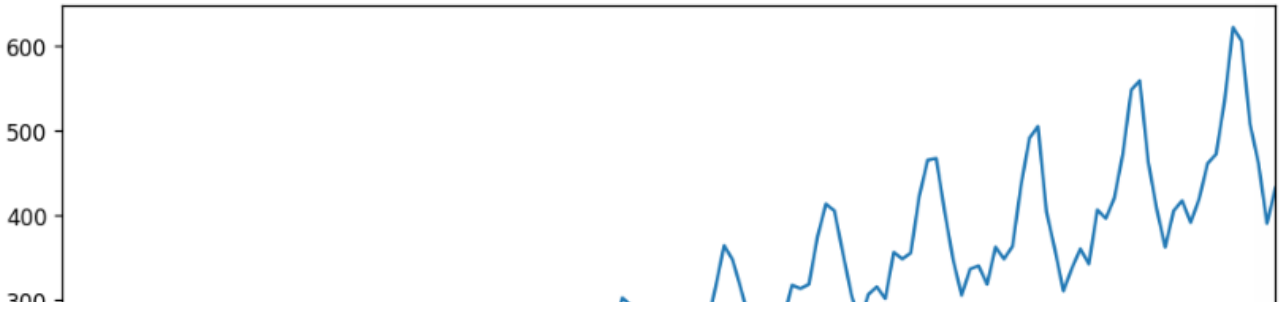
- **Rolling Mean**: The rolling mean, also known as the moving average, is calculated by taking the average of a fixed number of consecutive data points within a defined window and moving it along the time series. At each time point, the mean of the data points within the window is calculated.
- **Rolling Standard Deviation**: The rolling standard deviation is calculated by taking the standard deviation of a fixed window of data points as it moves along the time series. It measures the variability or dispersion of the data points within the window at each time point. The rolling standard deviation can provide insights into the volatility or stability of the time series over time. High values indicate greater variability or fluctuations, while lower values suggest a more stable or predictable pattern.

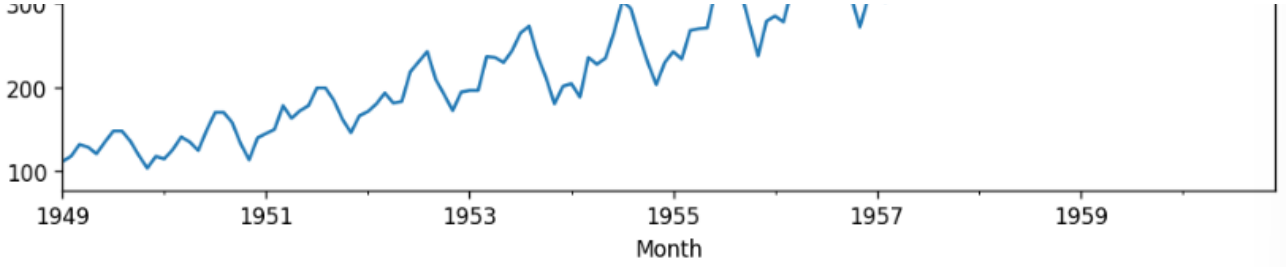
Properties and Types of Series

1. **Trend**: A long-term increase or decrease in the data. This can be seen as a slope roughly going through the data which doesn't need to be linear.
2. **Seasonality**: A time is said to be seasonal when it is affected by seasonal factors (day, week, month, year, etc.).
3. **Cyclicality**: Cyclicality in time series refers to the presence of recurring patterns or fluctuations that repeat at regular intervals over time. Unlike seasonal patterns, which are short-term and regular fluctuations that repeat within a year (e.g., quarterly sales patterns), cyclical patterns have longer durations and are often not as predictable.
4. **Residuals**: Each time series can be decomposed into two parts:- forecast and residuals.

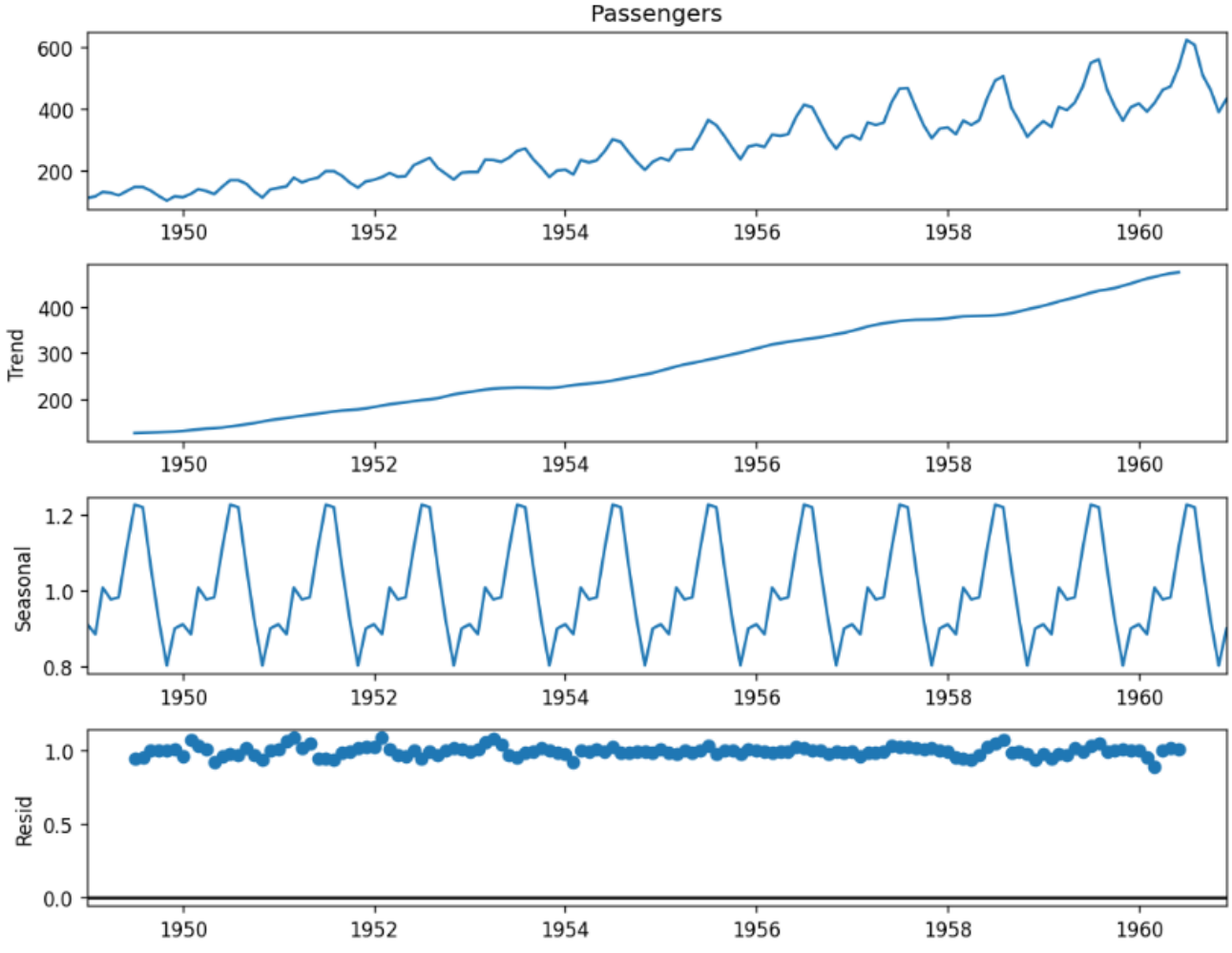
Value of series at time t = Predicted value at time t + Residual at time t

- **The plot Airline Passenger dataset is shown below**





- **We shall then decompose the series for further analysis**



- **Observations :**
  - **Positive trend.**
  - **Seasonal pattern.**

For a time series features are dependent on time. If the time series is not stationary the predictions deviate from the original values and increase the error. Thus we shall now make the series stationary.

To make the data stationary we shall use 5 different methods. To test stationarity we will use Augmented Dickey Fuller Test.

Augmented Dickey-Fuller Test

Null Hypothesis: It assumes that the time series is non-stationary.

Alternate Hypothesis: If the null hypothesis is rejected, then the time series is stationary.

The output of the Augmented Dickey-Fuller Test includes:

- Test Statistic
- p-value
- Number of Lags Used
- Number of Observations Used
- Critical Value (1%)
- Critical Value (5%)
- Critical Value (10%)

For the **Null Hypothesis** to be rejected and accepting that the time series is stationary, this is the necessary requirement:

- p-value < 0.05

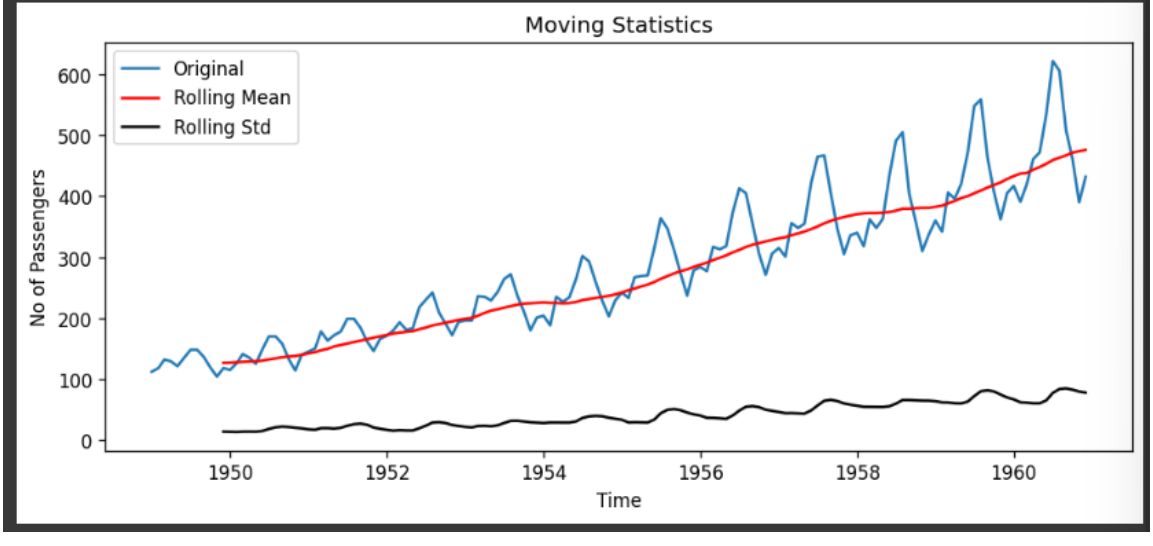
Method Used	**P value (From ADF Test)**
Original Data	0.991880
Log Transformation	0.422367
Subtracting Moving Avg	0.022235
Subtracting exponential Avg	0.005737
Subtracting previous value (before log transformation)	0.054213
Subtracting previous value (Of logged data)	0.071121
Subtracting 2nd previous value	0.021919
Seasonal decomposition (Residual Component)	2.885059e-08

As we can see, all the methods have given pretty good results and have made the series stationary to a pretty good extent. But the differencing method (subtracting previous value before and after log transformation of the data) is widely used. So we shall also be using these two methods and move ahead.

The effect of double-differencing would be too heavy on the data and might result in poor performance.

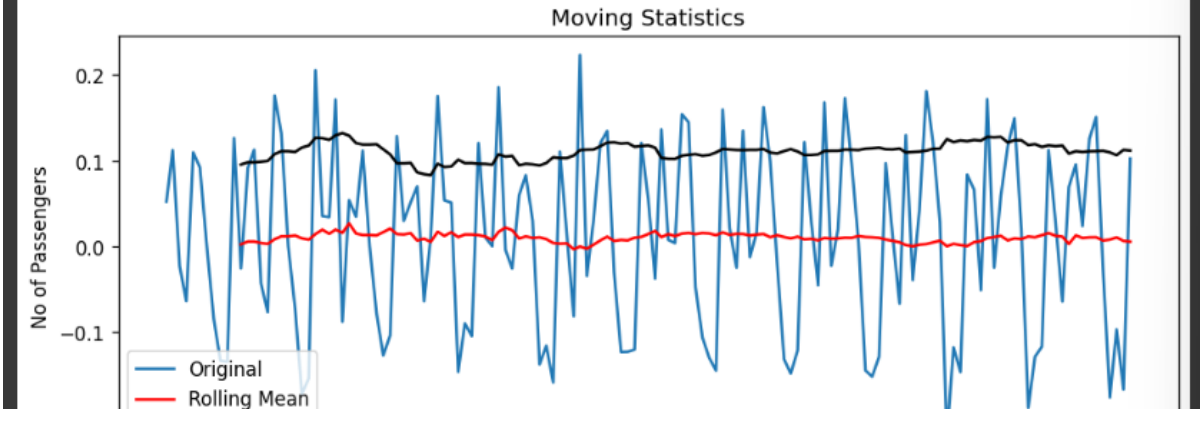
- Original Data

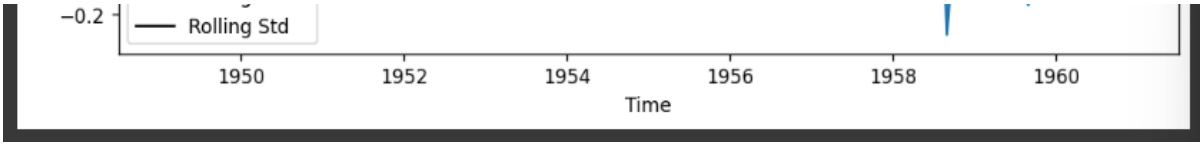
```
Results of Dickey-Fuller Test:
Test Statistic                  0.815369
p-value                         0.991880
#Lags Used                     13.000000
Number of Observations Used    130.000000
Critical Value (1%)             -3.481682
Critical Value (5%)             -2.884042
Critical Value (10%)            -2.578770
dtype: float64
Fail to reject the null hypothesis. The data is non-stationary.
------------------------------------------------------------------------
```



- Subtracting previous value (Of logged data)

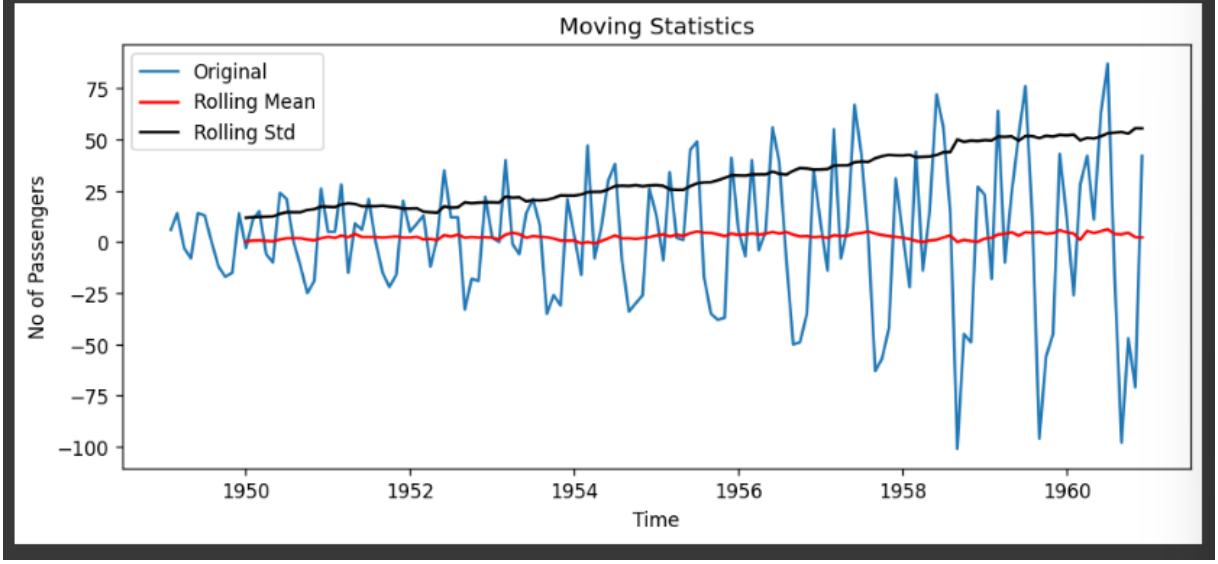
```
Results of Dickey-Fuller Test:
Test Statistic                 -2.717131
p-value                         0.071121
#Lags Used                     14.000000
Number of Observations Used    128.000000
Critical Value (1%)             -3.482501
Critical Value (5%)             -2.884398
Critical Value (10%)            -2.578960
dtype: float64
Fail to reject the null hypothesis. The data is non-stationary.
------------------------------------------------------------------------
```





- Subtracting previous value (before log transformation)

```
Results of Dickey-Fuller Test:
Test Statistic                  -2.829267
p-value                          0.054213
#Lags Used                      12.000000
Number of Observations Used    130.000000
Critical Value (1%)             -3.481682
Critical Value (5%)             -2.884042
Critical Value (10%)            -2.578770
dtype: float64
Fail to reject the null hypothesis. The data is non-stationary.
------------------------------------------------------------------------
```



Autocorrelation and Partial Autocorrelation in time series

Autocorrelation: Autocorrelation is the correlation between two observations at different points in a time series. Values that are separated by an interval might have a strong positive or negative correlation. When these correlations are present, they indicate that past values influence the current value. The number of intervals between the two observations is the lag. For example, the lag between the current and previous observation is one. In mathematical terms, the observations at yt and yt–k are separated by k time units. K is the lag. The autocorrelation function (ACF) assesses the correlation between observations in a time series for a set of lags. The ACF for time series y is given by: Corr (yt,yt−k), k=1,2,3 etc...

The autocorrelation function (ACF) is a valuable tool for analyzing time series data and identifying significant correlations at different time lags. By examining the ACF plot, we can gain insights into

the patterns and characteristics of the time series. This information is crucial for modeling the data effectively.

In the ACF plot, each bar corresponds to a correlation value and its direction. Significant correlations are indicated by bars that extend beyond the threshold line (usually red). By assessing the ACF, we can evaluate the randomness and stationarity of the time series. Moreover, it helps us detect the presence of trends and seasonal patterns in the data. Utilizing these findings, we can make informed decisions to create appropriate models for the time series analysis.

Random data: Autocorrelations should be near 0 for all lags. Non Random data: Have at least 1 significant lag.

When the data is not random, it's a good indication to use time series analysis to model the data appropriately.

The autocorrelation function declines to near zero for a stationary time series, in contrast, the autocorrelation function drops slowly for non-stationary time series.

When trends are present in time series, shorter lags typically have large positive correlations because observations closer in time tend to have similar values.

When the data is seasonal the autocorrelation is large for lags at multiples of seasonal frequency. When a time series has both a trend and seasonality, the ACF plot displays a mixture of both effects.

Partial Autocorrelation: The partial autocorrelation function (PACF) is akin to the ACF but focuses on displaying correlations between two observations that cannot be explained by shorter lags. Specifically, for a given lag, the PACF represents the unique correlation between the two observations after removing the influence of the intervening correlations.

While the ACF helps understand the characteristics of a time series, the PACF is particularly valuable during the process of specifying autoregressive models. It is commonly employed by analysts to determine appropriate regression models when dealing with time series data and constructing Auto Regressive Integrated Moving Average (ARIMA) models.

Modeling

Arima Model

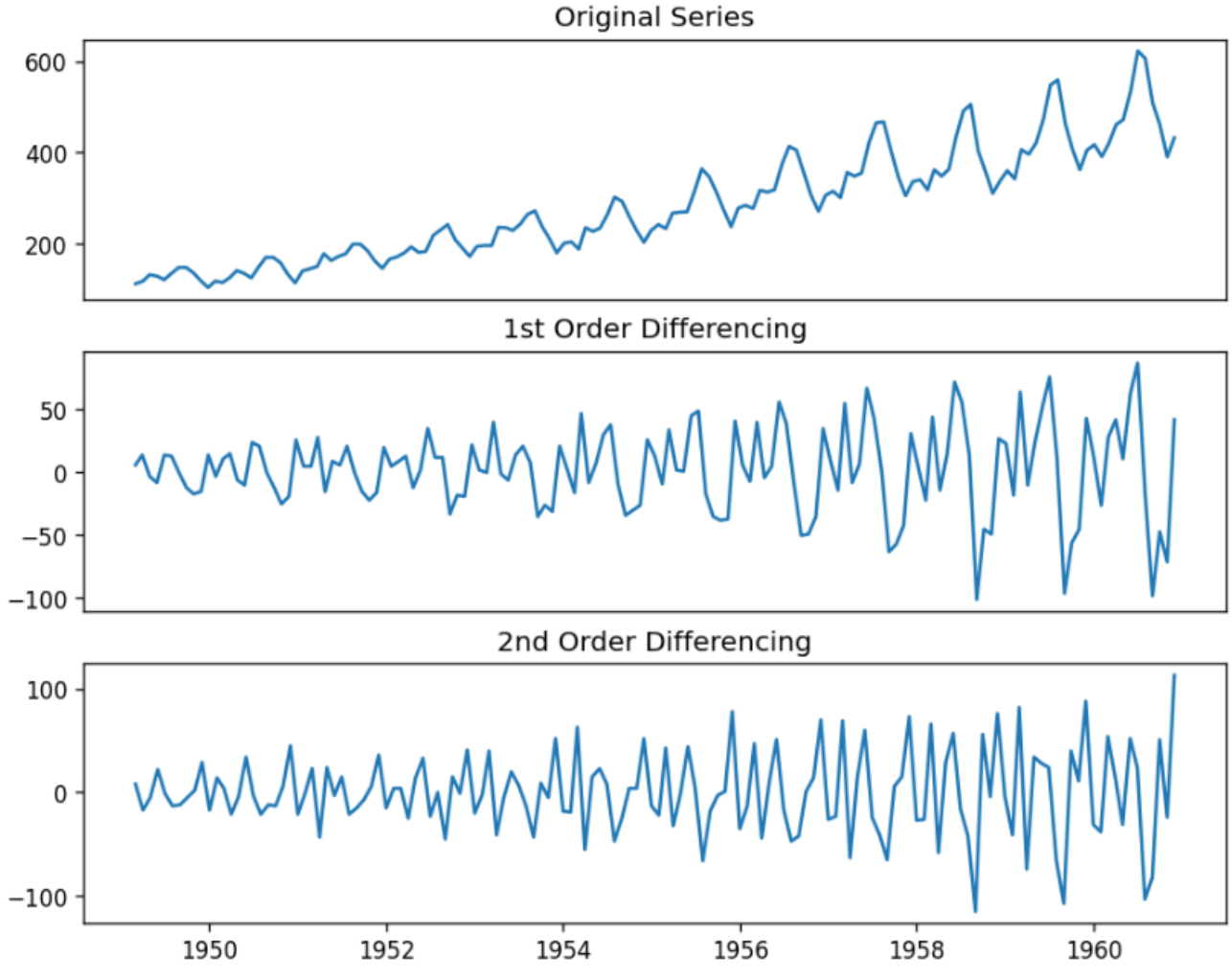
ARIMA models (which include ARMA, AR, and MA models) are a general class of models to forecast stationary time series. ARIMA models are made of three parts:

- A weighted sum of lagged values of the series. (Autoregressive AR part).
- A weighted sum of lagged forecasted errors of the series. (Moving average MA part).
- A difference in the time series (Integrated I part).

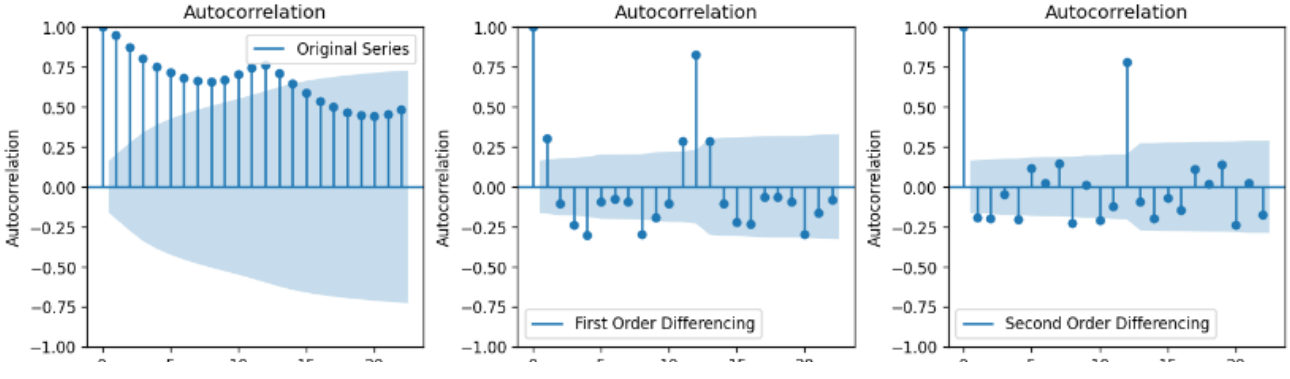
An ARIMA model is often noted as ARIMA(p, d, q) where p represents the order of the AR part, d the order of differencing ("I" part), and q the order of the MA term.

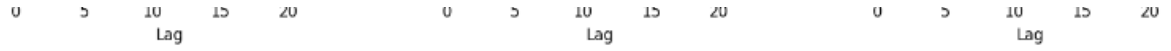
- p is the number of autoregressive terms,
- d is the number of nonseasonal differences,
- and q is the number of lagged forecast errors in the prediction equation.

Finding the value of the d parameter



We can see how the time series has become stationary. One thing which is noticeable here is in first-order differencing we have fewer noises in the data while after 1st order there is an increase in the noise. So we can select 1st order differencing for our model. We can also verify this using an autocorrelation plot.

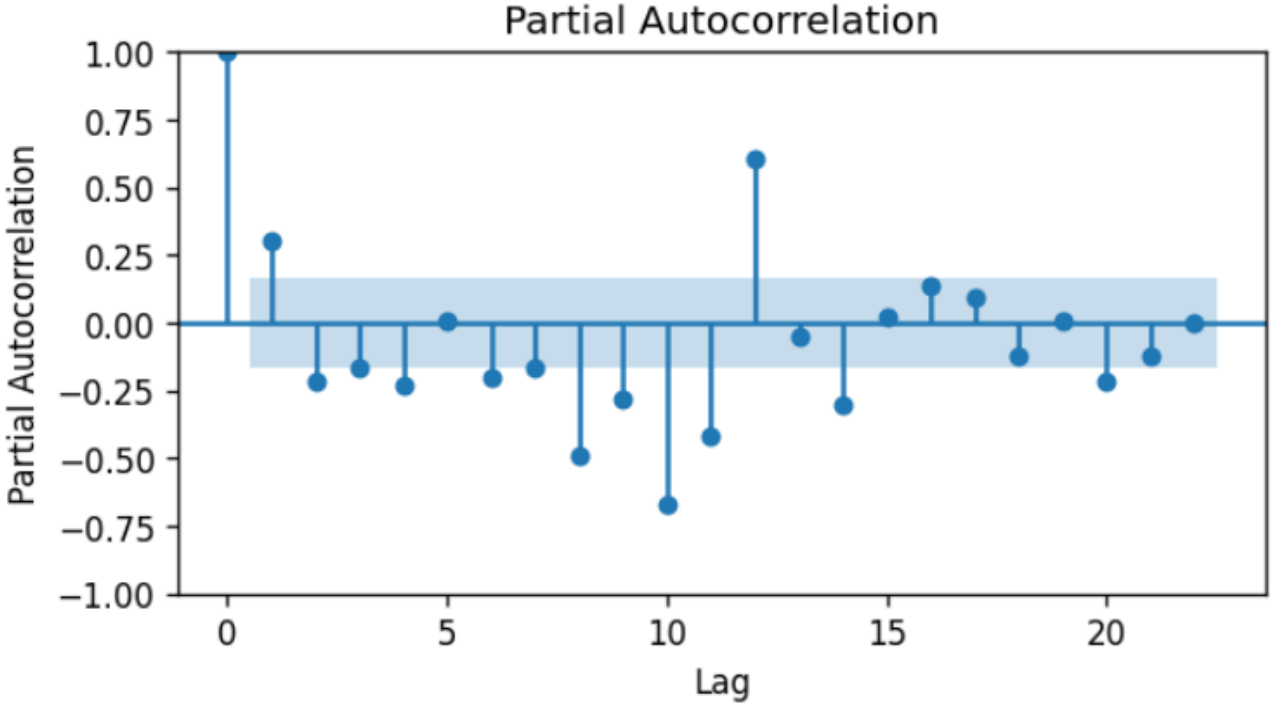




In second order differencing the immediate lag has gone on the negative side representing that in the second order, the series has become over the difference.

Finding the value of the P parameter

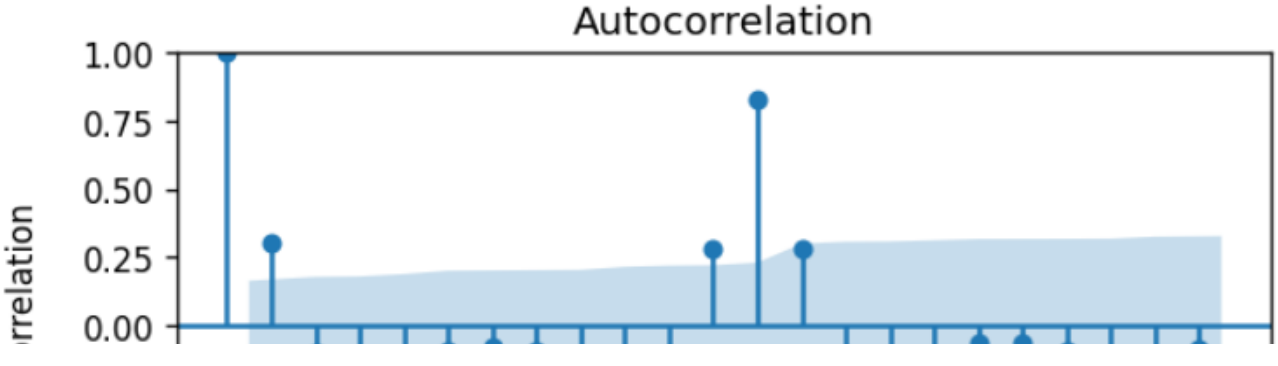
The partial auto-correlation function plot can be used to draw a correlation between the time series and its lag while the contribution from immediate lags can be ignored. Using the PACF plot we can take the order of AR terms to be equal to the lags that can cross a significance limit.

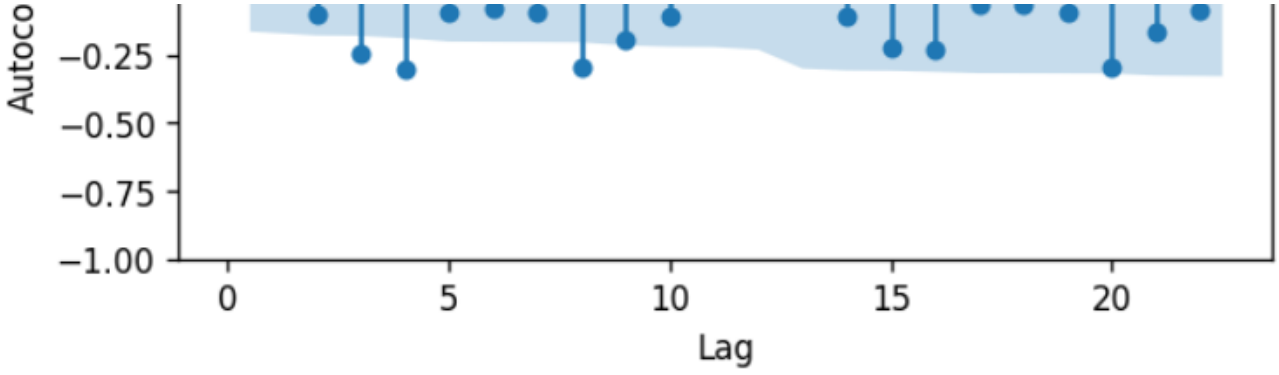


The first lag is significantly out of the limit but the second one is just out of the limit. So we can select the order of p=1.

Finding the value of the q parameter

ACF plot will help us to find the value of q. It will tell how much-moving average is required to remove the autocorrelation from the stationary time series.





Here from the figure it can be seen that 2 of the lags are out of the significance level, so we can say that the optimal value for q is 2.

ARIMA Model Summary

SARIMAX Results

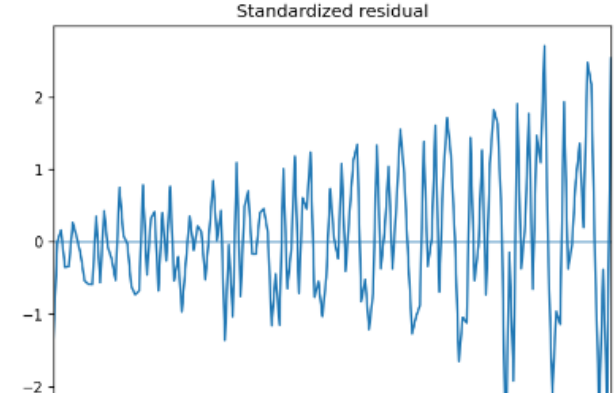
Dep. Variable:	Passengers	No. Observations:	144
Model:	ARIMA(2, 1, 2)	Log Likelihood	-671.673
Date:	Tue, 18 Jul 2023	AIC	1353.347
Time:	05:25:00	BIC	1368.161
Sample:	01-01-1949	HQIC	1359.366
	- 12-01-1960		
Covariance Type:	opg		

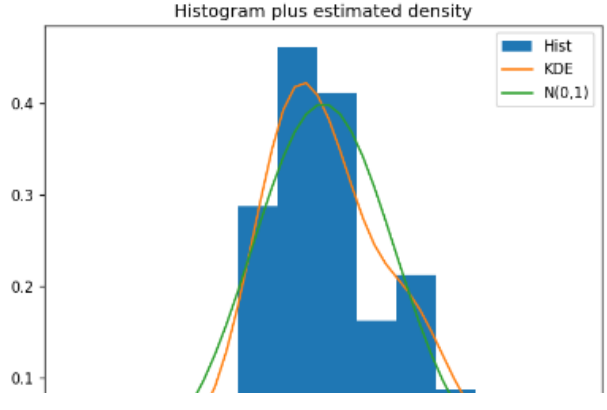
	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	1.6850	0.020	83.061	0.000	1.645	1.725
ar.L2	-0.9549	0.017	-55.420	0.000	-0.989	-0.921
ma.L1	-1.8432	0.124	-14.845	0.000	-2.087	-1.600
ma.L2	0.9953	0.135	7.398	0.000	0.732	1.259
sigma2	665.9646	113.887	5.848	0.000	442.751	889.178

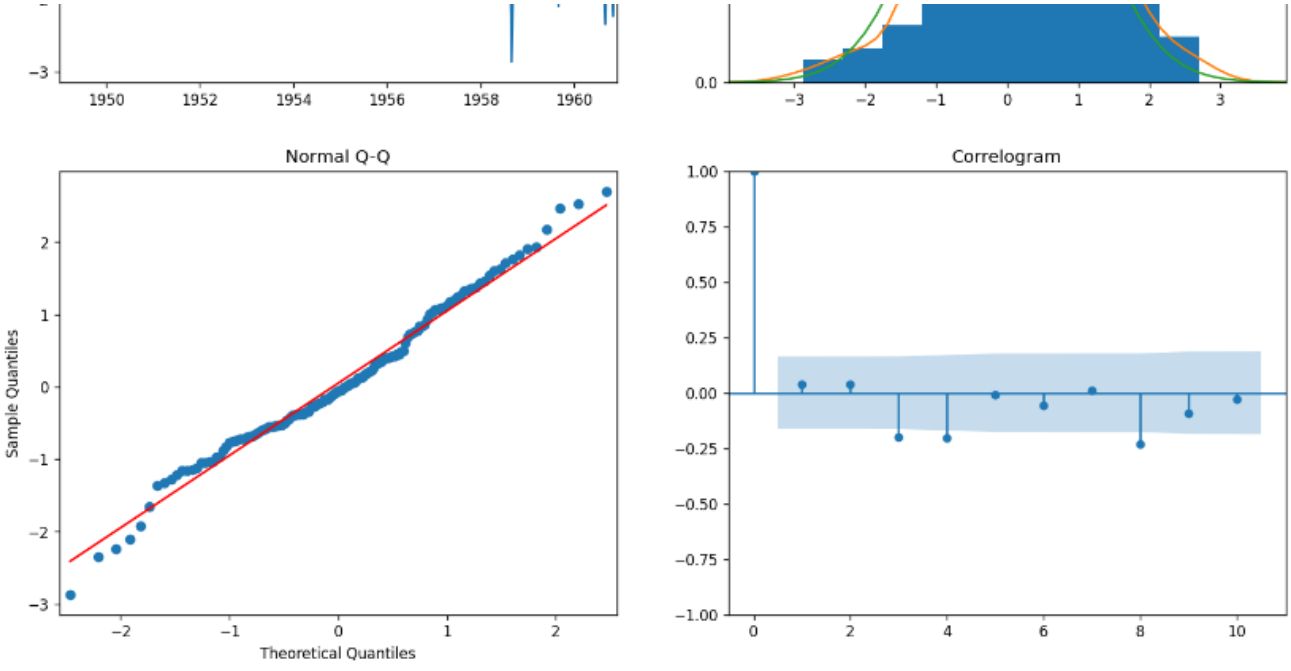
Ljung-Box (L1) (Q):	0.30	Jarque-Bera (JB):	1.84
Prob(Q):	0.59	Prob(JB):	0.40
Heteroskedasticity (H):	7.38	Skew:	0.27
Prob(H) (two-sided):	0.00	Kurtosis:	3.14

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Model Diagnostics







- Four plots result from the plot_diagnostics function. The Standardized residual, Histogram plus KDE estimate, Normal q-q, and the correlogram.
  - **Standardized residual:** There are no obvious patterns in the residuals, with the values having a mean of 0 and a variance of 1.
  - **Histogram and KDE estimate:** The KDE curve should be very similar to the normal distribution. (N(0,1))
  - **Normal Q-Q:** Most of the data points should lie in a straight line.
  - **Correlogram (ACF plot):** The grey area is the confidence band, and if values fall outside of this then they are statistically significant.

In-Sample forecasting: In-sample forecasting, also known as "within-sample forecasting" or "backtesting," involves making predictions or estimating future values based on data that was used to train the model. In this approach, the model uses historical data up until a certain point to make predictions for the same time period. It measures the model's performance on data it has already seen.

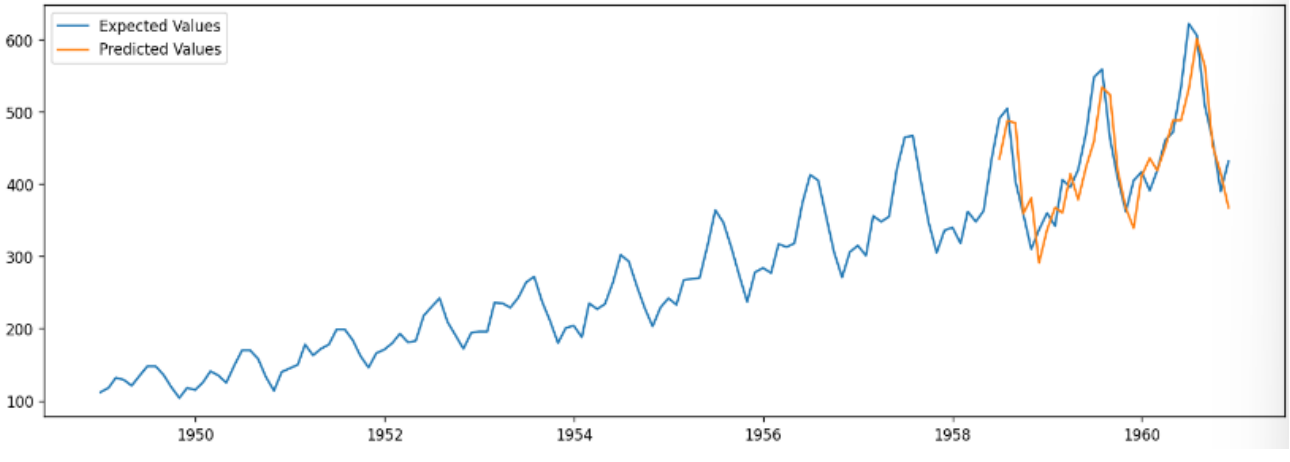
The advantage of in-sample forecasting is that it can provide insights into how well the model fits the training data and how it performs on data it has already encountered. However, it may not accurately reflect the model's ability to generalize to unseen data because the model is essentially memorizing or overfitting the training data.

Test RMSE: 45.3502

```
        ARIMA MODEL : In- Sample Forecasting

predicted = 435.368729, expected = 491.000000
predicted = 487.825559, expected = 505.000000
predicted = 484.761914, expected = 404.000000
predicted = 359.074739, expected = 359.000000
predicted = 380.902398, expected = 310.000000
```

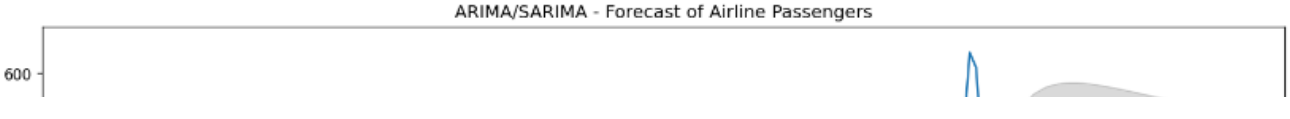
```
predicted = 291.230227, expected = 337.000000
predicted = 336.937875, expected = 360.000000
predicted = 367.624722, expected = 342.000000
predicted = 360.295134, expected = 406.000000
predicted = 414.083702, expected = 396.000000
predicted = 378.520878, expected = 420.000000
predicted = 424.474755, expected = 472.000000
predicted = 459.477027, expected = 548.000000
predicted = 534.244080, expected = 559.000000
predicted = 523.516869, expected = 463.000000
predicted = 417.605516, expected = 407.000000
predicted = 366.683909, expected = 362.000000
predicted = 339.217575, expected = 405.000000
predicted = 411.312454, expected = 417.000000
predicted = 435.937844, expected = 391.000000
predicted = 418.796173, expected = 419.000000
predicted = 451.314963, expected = 461.000000
predicted = 488.383979, expected = 472.000000
predicted = 488.569136, expected = 535.000000
predicted = 532.219019, expected = 622.000000
predicted = 601.265999, expected = 606.000000
predicted = 563.028797, expected = 508.000000
predicted = 452.854957, expected = 461.000000
predicted = 415.004912, expected = 390.000000
predicted = 367.692302, expected = 432.000000
```

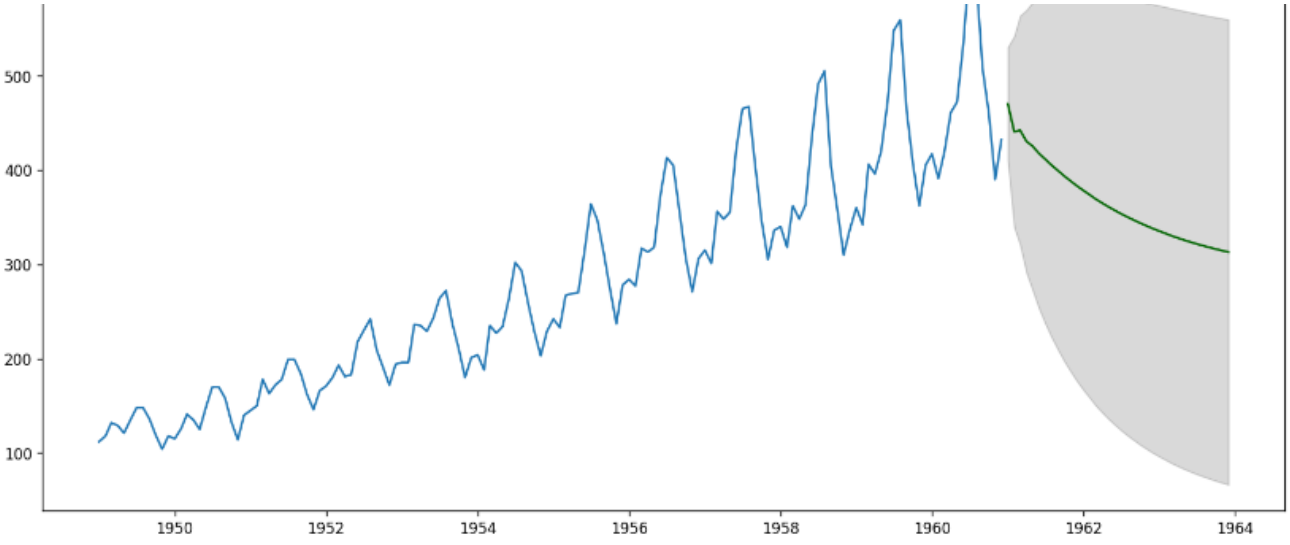


Out-Of Sample Forecasting: Out-of-sample forecasting, also referred to as "holdout forecasting" or "forward testing," involves making predictions or estimating future values based on data that the model has not seen during training. In this approach, the model is evaluated on its ability to generalize to new, unseen data.

The advantage of out-of-sample forecasting is that it provides a more realistic assessment of the model's performance in real-world scenarios where it needs to make predictions on unseen data. It helps assess the model's ability to generalize and provides an indication of how well it might perform in the future.

Forecast for the next 36 months





We can see that the plot above doesn't seem to be much accurate forecast. Maybe we need to change the model structure so that it takes seasonality into account.

SARIMA Model

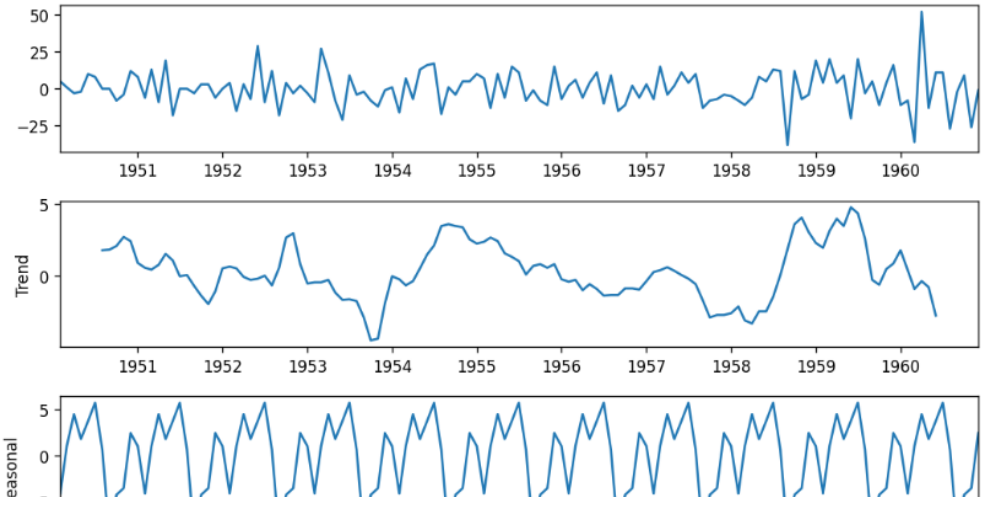
Although the ARIMA model can handle data with trends, it doesn't support time series with seasonal components. An extension to ARIMA that supports the direct modeling of the seasonal component of the series is called SARIMA.

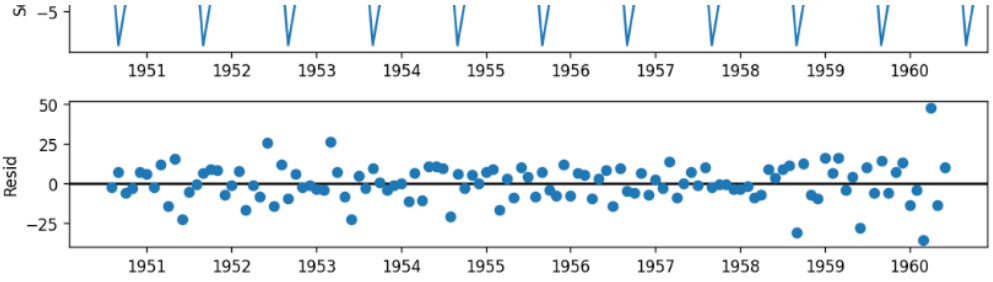
The SARIMA model has 2 kinds of orders **(p,d,q) x (P, D, Q, M). (p,d,q)** is the order that is similar to the order of the ARIMA model.

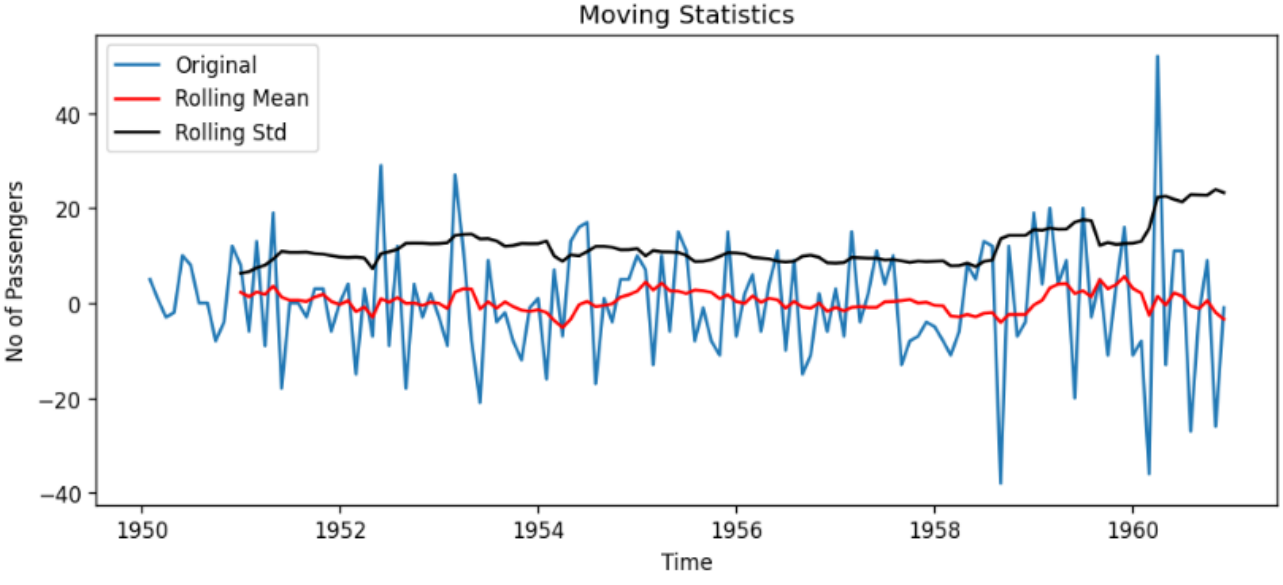
(P, D, Q, M) is known as the Seasonal Order where **(P,D,Q)** are similar to the **(p,d,q)** of the ARIMA model.

Our data is in monthly format and the seasonal period is of 1 year. Hence, we difference the already differenced data by a periodicity, M, value of 12. The seasonality of the data has not completely died down but its values have been dropped.

We will check this seasonally differenced data for stationarity.

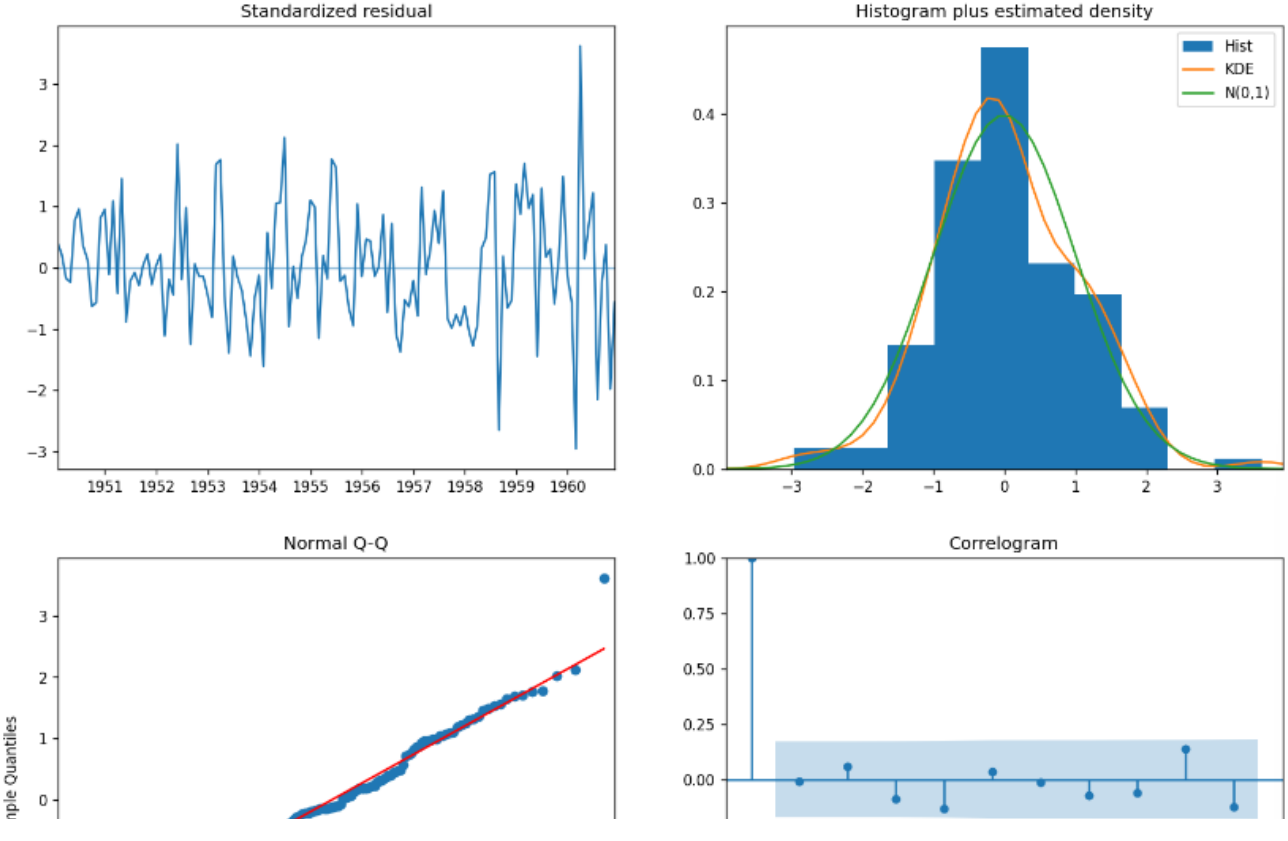


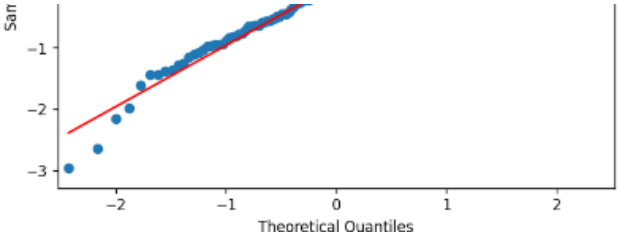


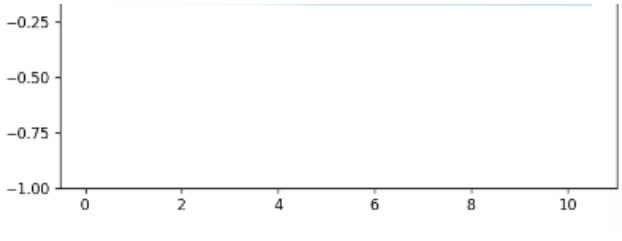


From the ADF test, the rolling mean is 0 and the p-value is <0.05. Hence the series is now stationary.

Model Diagnostics





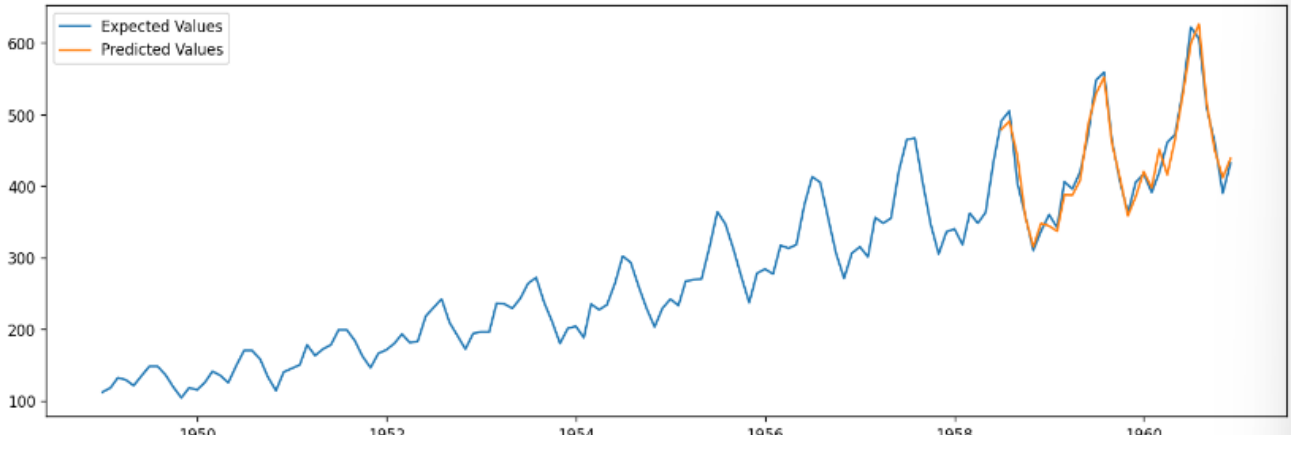


In-Sample forecasting

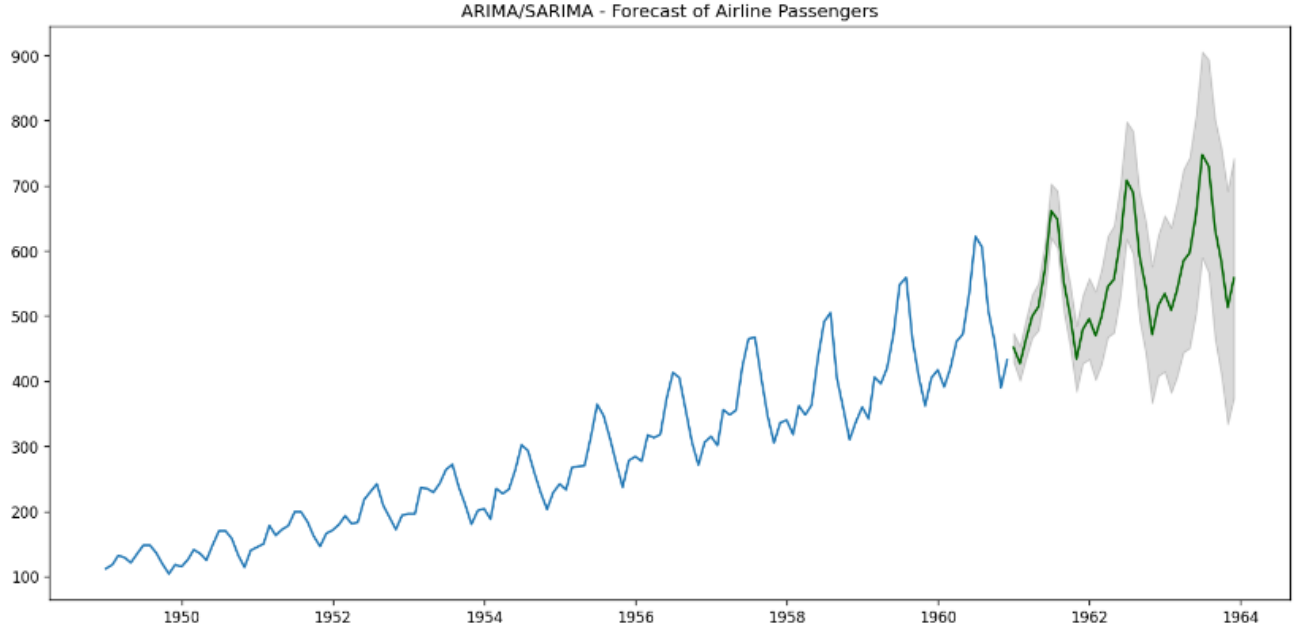
RMSE: 16.9251

```
          SARIMA MODEL : In - Sample Forecastin

predicted = 479.085765, expected = 491.000000
predicted = 490.553509, expected = 505.000000
predicted = 441.276125, expected = 404.000000
predicted = 357.274099, expected = 359.000000
predicted = 315.250484, expected = 310.000000
predicted = 347.831879, expected = 337.000000
predicted = 344.251448, expected = 360.000000
predicted = 336.839209, expected = 342.000000
predicted = 387.593116, expected = 406.000000
predicted = 387.333485, expected = 396.000000
predicted = 408.192789, expected = 420.000000
predicted = 485.988165, expected = 472.000000
predicted = 529.031343, expected = 548.000000
predicted = 551.914006, expected = 559.000000
predicted = 459.061270, expected = 463.000000
predicted = 411.970100, expected = 407.000000
predicted = 358.421153, expected = 362.000000
predicted = 384.945724, expected = 405.000000
predicted = 420.144085, expected = 417.000000
predicted = 397.755393, expected = 391.000000
predicted = 451.335502, expected = 419.000000
predicted = 415.675182, expected = 461.000000
predicted = 465.295980, expected = 472.000000
predicted = 529.835405, expected = 535.000000
predicted = 599.299657, expected = 622.000000
predicted = 626.292198, expected = 606.000000
predicted = 513.891979, expected = 508.000000
predicted = 450.136741, expected = 461.000000
predicted = 411.653931, expected = 390.000000
predicted = 438.411438, expected = 432.000000
```



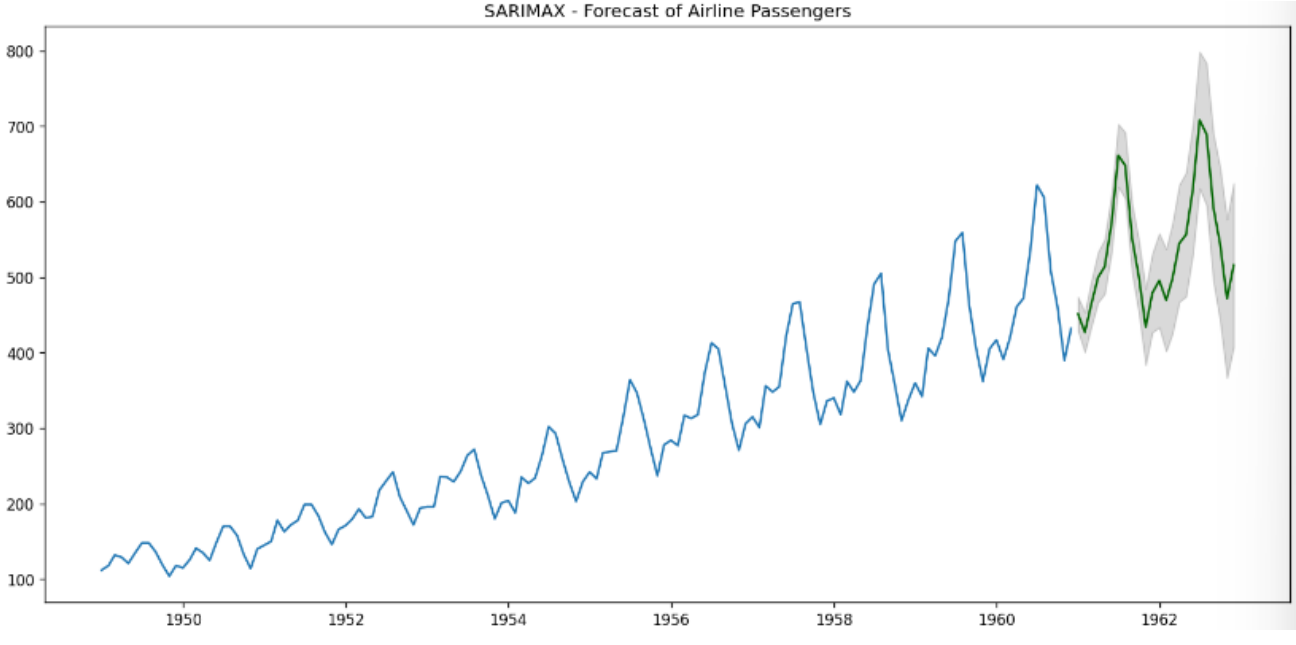
Out-of Sample forecasting



As we can see from the plot below, this seems to be much more accurate than the standard ARIMA model!

SARIMAX Model

Adding in an exogenous variable in the SARIMA model. We simply added the month number as an exogenous variable, but it is not super useful as this is already conveyed through seasonality.



We can see from the following predictions that we are getting some pretty good-looking predictions and the width of the forecasted confidence interval has decreased. This means that the model is more certain of its predictions.

Particle swarm Optimization

Particle Swarm Optimization (PSO) is an optimization algorithm inspired by the social behavior of bird flocking or fish schooling. It was originally proposed by Kennedy and Eberhart in 1995.

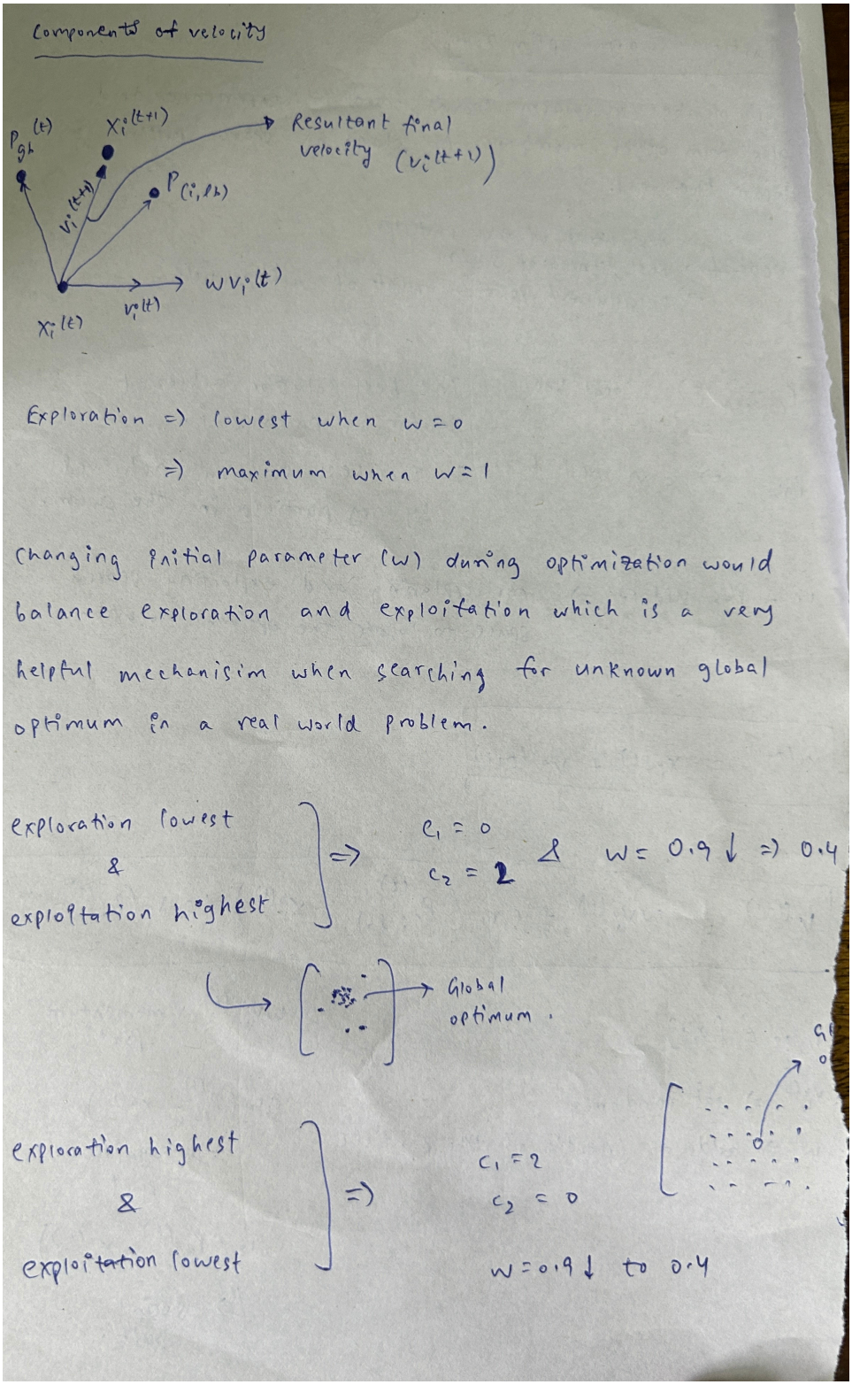
The basic idea behind PSO is to simulate the collective intelligence and cooperation observed in natural systems. The algorithm consists of a population of particles that move through a search space to find the optimal solution to a given problem. Each particle represents a potential solution and is associated with a position and a velocity.

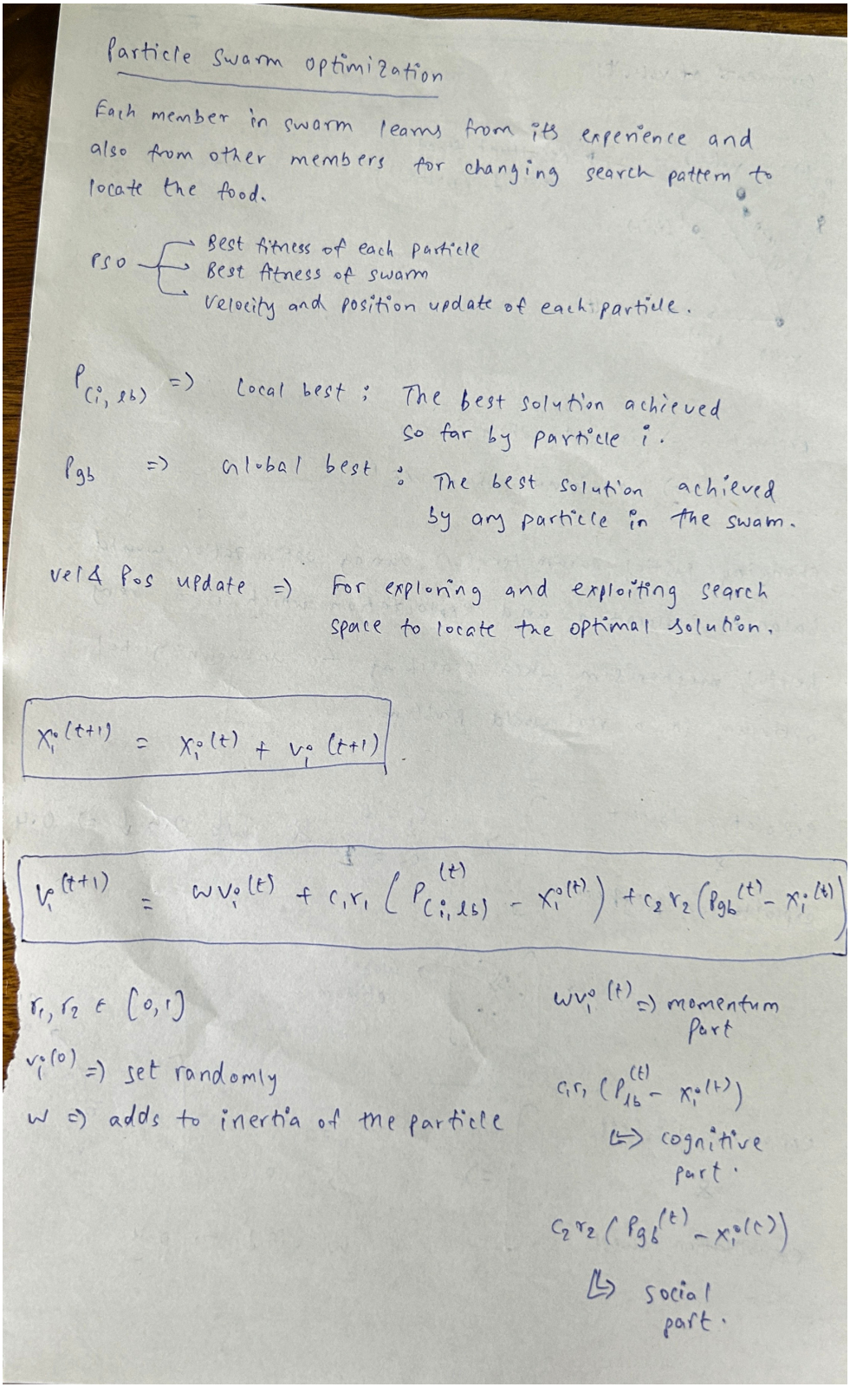
The particles communicate and collaborate with each other to search for the best solution by adjusting their positions and velocities based on their own experience and the experience of the best-performing particle within the population, known as the global best. The movement of each particle is influenced by its current velocity, its best-known position (personal best), and the global best position found so far.

During the optimization process, particles explore the search space by updating their velocities and positions. The updated equations involve both individual learning (based on personal best) and social learning (based on global best). This combination of personal and global information exchange allows particles to converge toward promising regions of the search space.

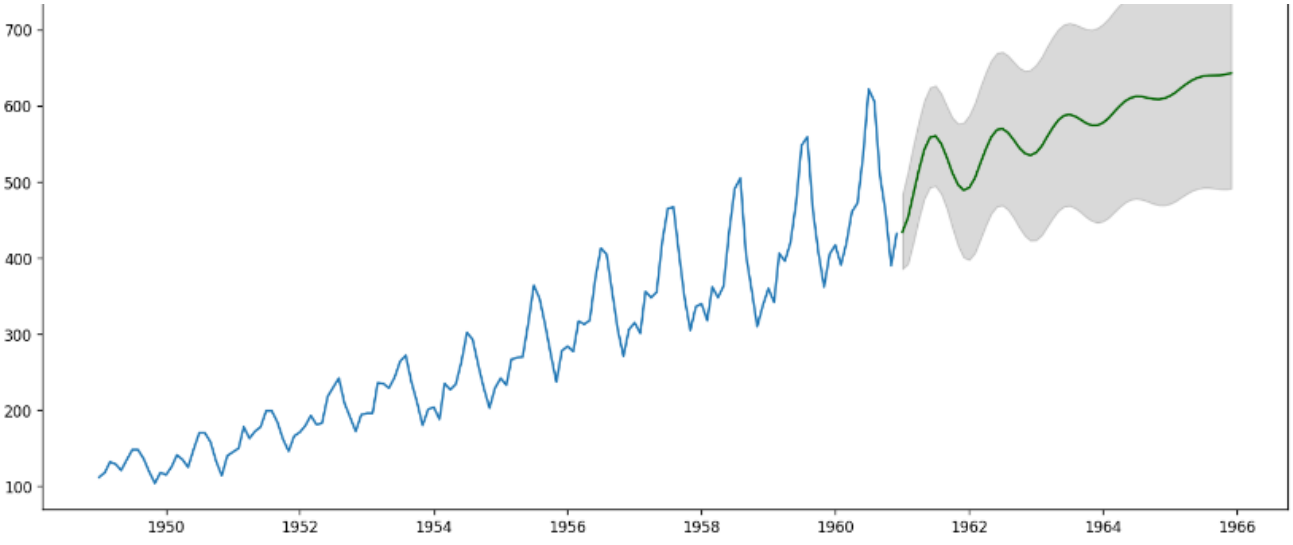
The algorithm continues iterating until a termination criterion is met, such as reaching a maximum number of iterations or achieving a satisfactory solution.

Unlike traditional optimization methods, PSO does not require the problem to be differentiable.





As we saw the ARIMA model didn't give good results and the forecast didn't seem right. I tried to use the PSO algorithm to get somewhat better results. Yes, the results improved comparatively, however, they aren't as good as the results/forecast done by SARIMA and SARIMAX models.

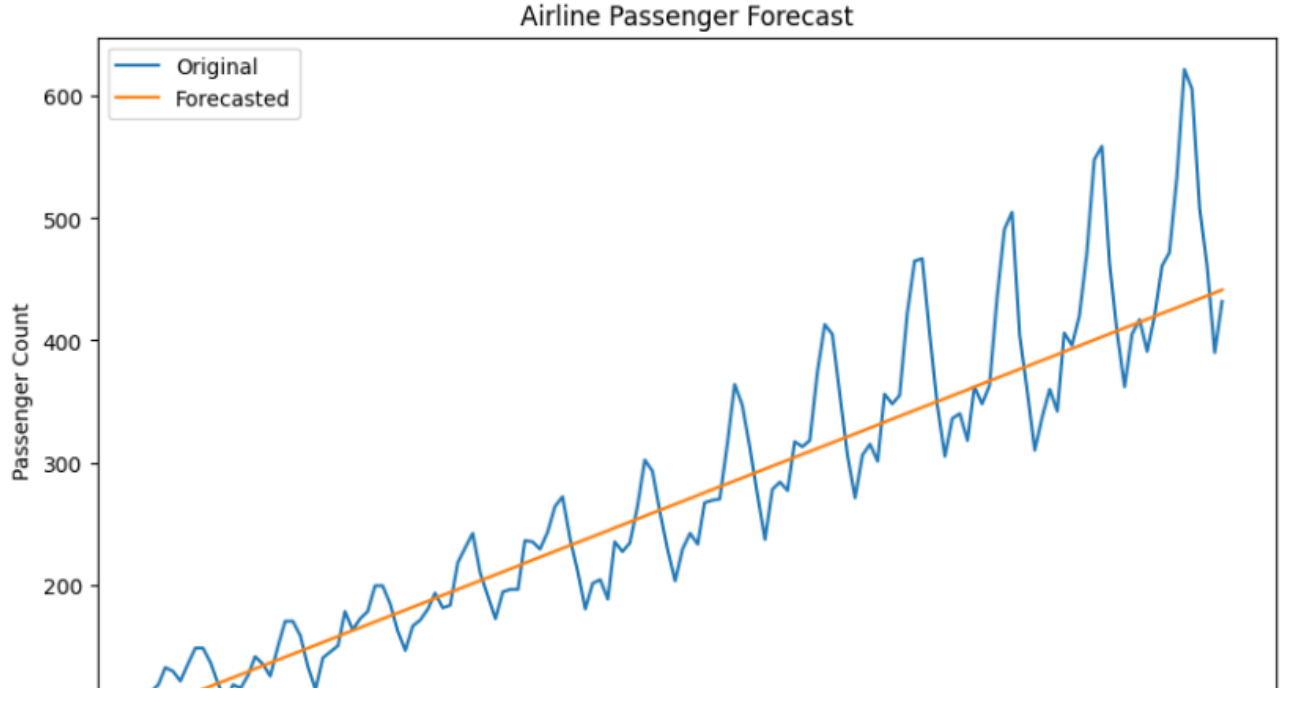


I also tried to perform forecasting using PSO-trained neural networks. The results were disappointing as they could just produce a linear line almost along the slope of the trend. It could neither capture the seasonality nor the cyclicity. This may happen because either the value of the parameters was not set properly or the model is too basic for this task. Probably we might need to increase more number of hidden units of the hidden layer or we might need to increase the number of hidden layers itself.

Firstly I trained a neural network with 1 hidden layer and assigned the **number of hidden units** to be in the range of 1 to 100. I assigned the value of the **Learning rate** to be in the range of 0.001 to 0.1. Lastly the values of momentum to e in the range 0.1 to 0.9.

Then performed the Particle Swarm Optimization to get the most optimal values of these parameters. The results of the optimization were:-

- number of hidden units = 11/12
- learning rate = 0.0956
- Momentum = 0.647



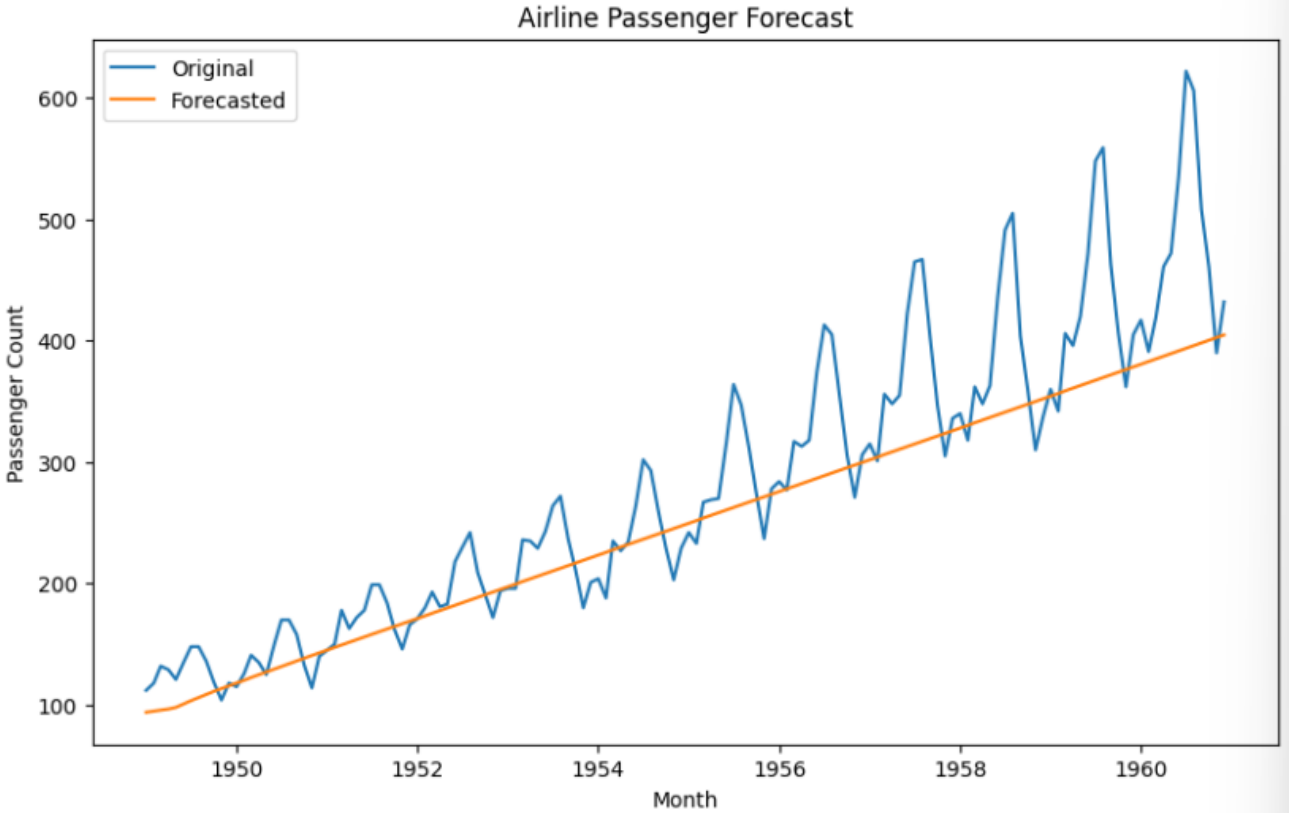
I then tried to use a neural network with 2 hidden layers. Bounds for the parameters → lb/ub: lower/upper bound.

lb = [1, 1, 0.001, 0.1], ub = [100, 100, 0.1, 0.9]

The first indices give the range of the number of units in the first hidden layer. The second indices give the range of the number of hidden units in the second hidden layer. The third indices give the range of the learning rate, and the last indices give the range of the values of momentum. After applying the PSO algorithm, the optimal values of the parameters were:-

- Num of hidden units in the first layer = 82
- Num of hidden units in the second layer = 89
- learning rate = 0.077
- momentum = 0.311

Even after this, the results were not even close, though a kink was observed at the start as compared to the neural network with only one hidden layer.



Conclusion and Summary

In this project, I tried to perform a time series analysis and forecast on the **Air Passenger** dataset using ARIMA, SARIMA, and SARIMAX models.

ARIMA model was not capable to handle the seasonality of the dataset, hence couldn't forecast the seasonal pattern and cyclicity accurately. This is the reason the Arima model didn't give good results and made me choose the SARIMA model, which is specially built to take seasonality into account.

The **Test RMSE: 45.3502** for the ARIMA model.

The ARIMA model took seasonality into account and gave pretty good predictions and forecasts. SARIMA (Seasonal Autoregressive Integrated Moving Average) models are often preferred over simple ARIMA (Autoregressive Integrated Moving Average) models when dealing with time series data that exhibit seasonality. Here are a few reasons why SARIMA may be chosen over ARIMA:

- Seasonal patterns: SARIMA models are specifically designed to capture and model the seasonal patterns present in the data. These models include additional seasonal components, such as seasonal autoregressive (SAR) and seasonal moving average (SMA) terms, which allow for better representation and forecasting of seasonal fluctuations.
- Improved accuracy: By incorporating the seasonal components, SARIMA models can provide more accurate forecasts for seasonal time series data. They account for the repetitive patterns observed at fixed intervals within the data, enabling better capturing of the seasonal variations.
- Enhanced model flexibility: SARIMA models offer greater flexibility in modeling complex time series data with both non-seasonal and seasonal components. They can handle various combinations of autoregressive (AR), integrated (I), and moving average (MA) terms for both non-seasonal and seasonal components, allowing for more nuanced modeling and improved forecasting accuracy.
- Comprehensive forecasting: SARIMA models can generate forecasts that capture both short-term and long-term trends, as well as the seasonal patterns within the data. This makes them suitable for predicting future values of time series data with inherent seasonality, such as air passenger data.
- The choice between SARIMA and ARIMA depends on the characteristics of the specific time series data being analyzed. If the data exhibits clear seasonality, SARIMA models are typically more appropriate.

The **Test RMSE : 16.925** for SARIMA model.

SARIMAX (Seasonal Autoregressive Integrated Moving Average with Exogenous Variables) models are chosen over SARIMA models when there is a need to incorporate exogenous variables that can influence the time series being analyzed. Here are a few reasons why SARIMAX may be chosen over SARIMA:

- Exogenous variables: SARIMAX models allow for the inclusion of exogenous variables that are not part of the time series being forecasted but can affect its behavior. These variables could be external factors, economic indicators, or other relevant factors that can provide additional information to improve forecasting accuracy. By including exogenous variables, SARIMAX models can capture their impact on the time series and potentially enhance the forecasting performance.

- Improved model flexibility: SARIMAX models offer greater flexibility in modeling time series data by incorporating exogenous variables. This flexibility enables the model to account for the effects of external factors that may have a significant influence on the time series behavior. By considering these variables, SARIMAX models can capture additional complexities and variations in the data that would not be possible with SARIMA models alone.
- Enhanced forecasting accuracy: By incorporating exogenous variables, SARIMAX models can potentially improve the accuracy of forecasts. The inclusion of relevant external factors can provide valuable information about the underlying patterns and drivers of the time series, resulting in more precise and reliable predictions. This is particularly beneficial when the exogenous variables have a significant impact on the time series being analyzed.
- Real-world applicability: SARIMAX models are particularly useful in real-world scenarios where the behavior of the time series is influenced by external factors. For example, in forecasting air passenger data, incorporating exogenous variables such as economic indicators, holidays, or events can help capture the impact of these factors on passenger demand and improve forecasting accuracy.

PSO: Particle Swarm Optimization (PSO) is a computational optimization technique inspired by the social behavior of bird flocking or fish schooling. It is used to solve optimization problems, particularly those involving high-dimensional search spaces or non-linear and non-convex objective functions.

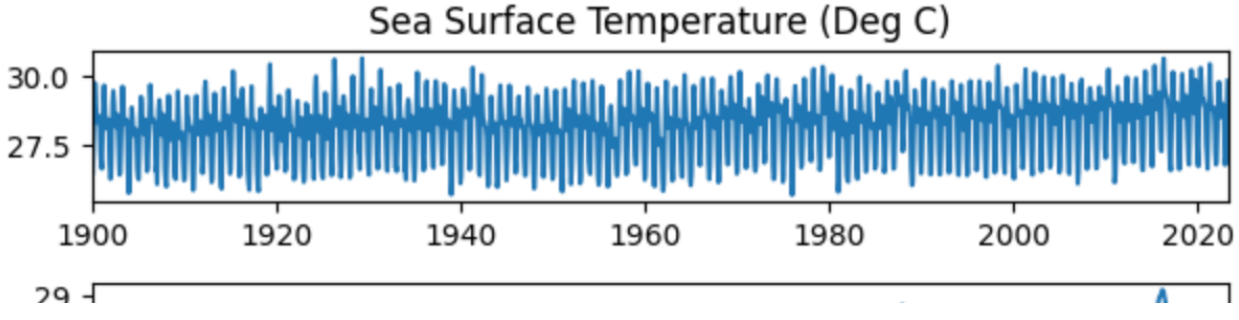
PSO also has an advantage over other optimization algorithms such as gradient descent, Stochastic gradient descent as in PSO differentiation of the objective function is not required.

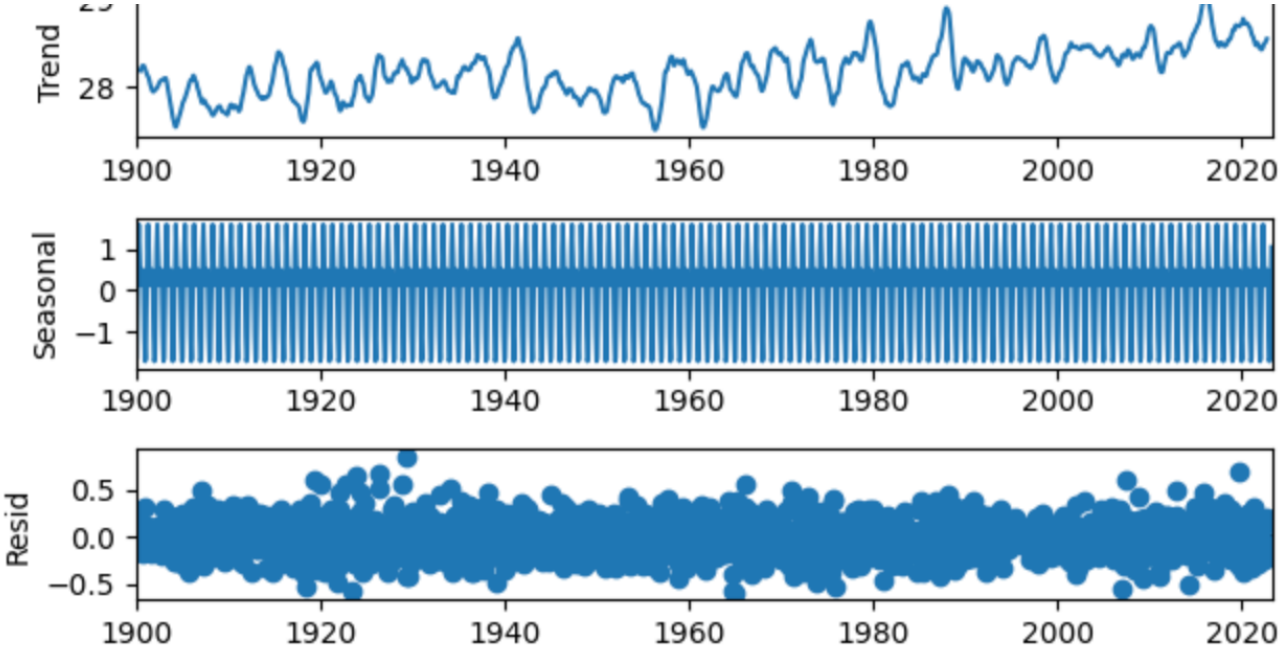
In PSO, a population of particles represents potential solutions within the search space. Each particle's position represents a candidate solution, and its movement is influenced by its own best-known position and the best-known position among all particles in the population. The particles adjust their positions iteratively by following the "best" particles in the search space, aiming to find the global optimum.

Part - 2 (Sea Surface Temperature)

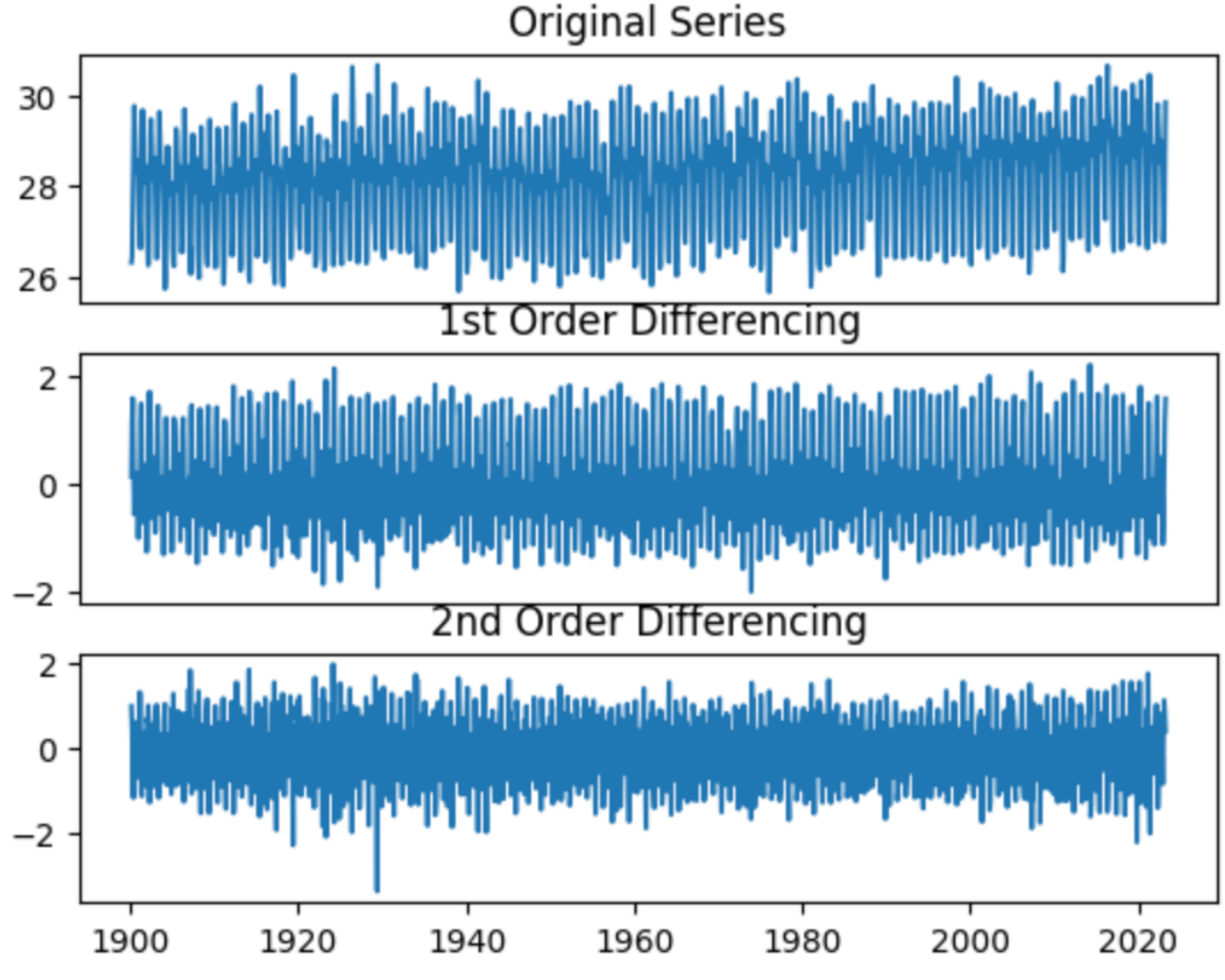
AIM AND DATASET

This dataset contains the monthly sea surface temperature from the year 1900 to 2023. The data is already stationary. Then I decomposed the dataset into its trend, seasonal and residual.



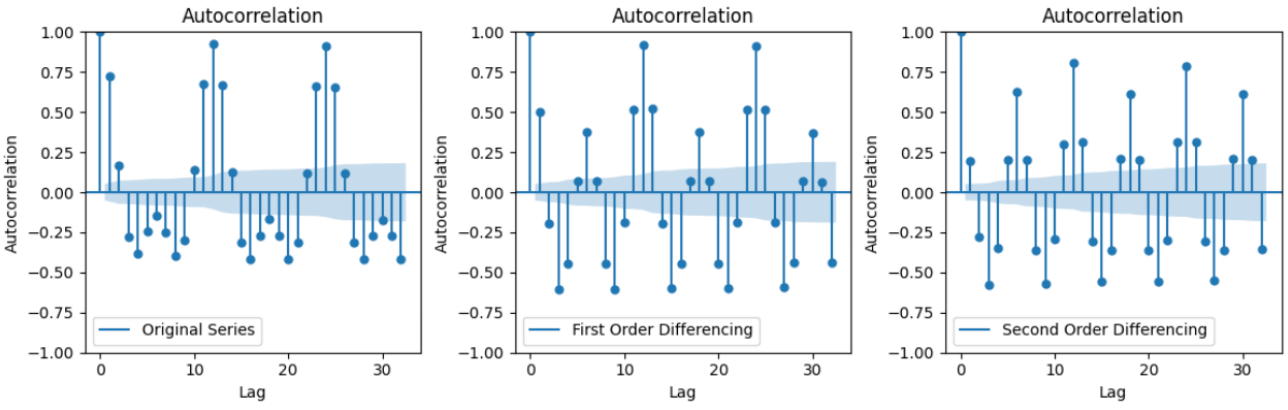


We can see that the time series is seasonal and stationary. Now we shall forecast using the ARIMA, SARIMA, and SARIMAX models. As the dataset is already stationary, there is no need to difference the data.



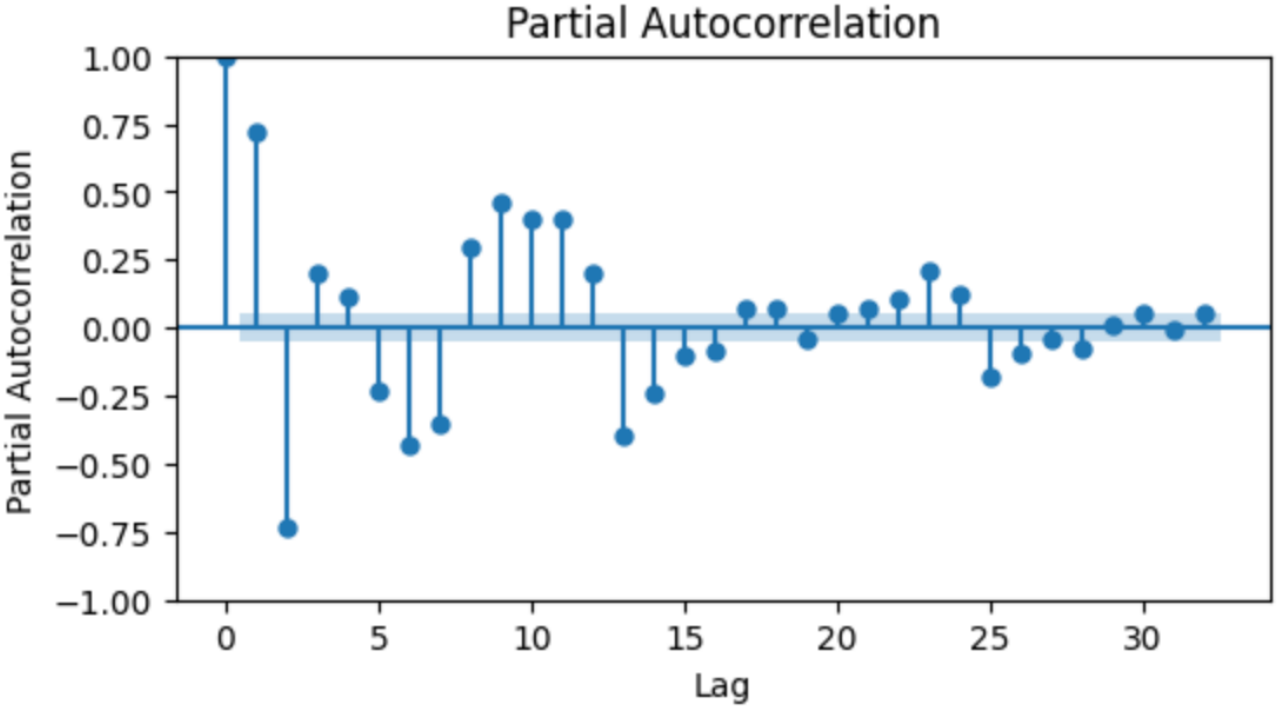
ARIMA MODEL

The value of the d parameter



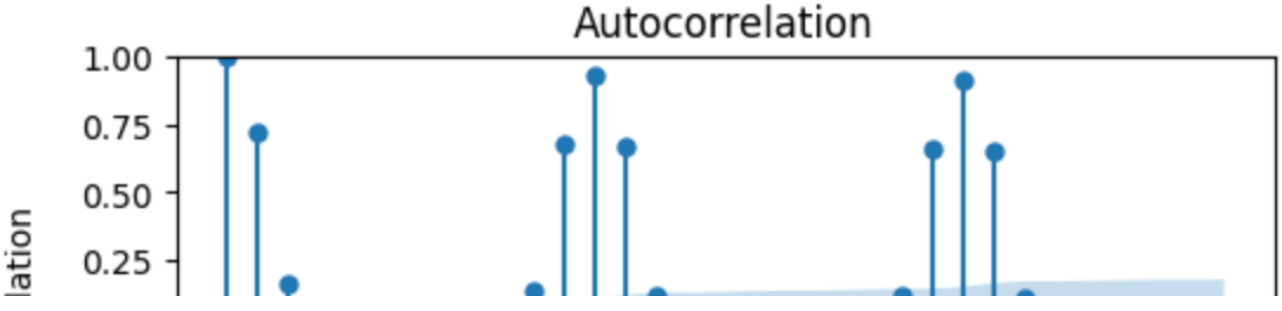
Since the dataset is already Stationary there is no need to difference the data. So, we set the value of d as 0.

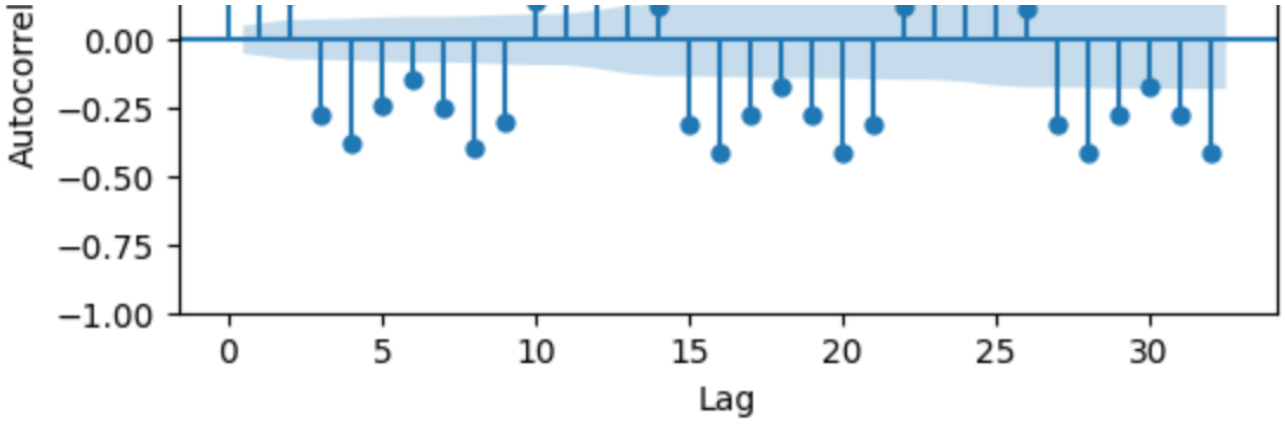
The value of p parameter



Since the starting 3 lags are significantly out of the threshold region we set the value of the p parameter as 3.

The value of q parameter



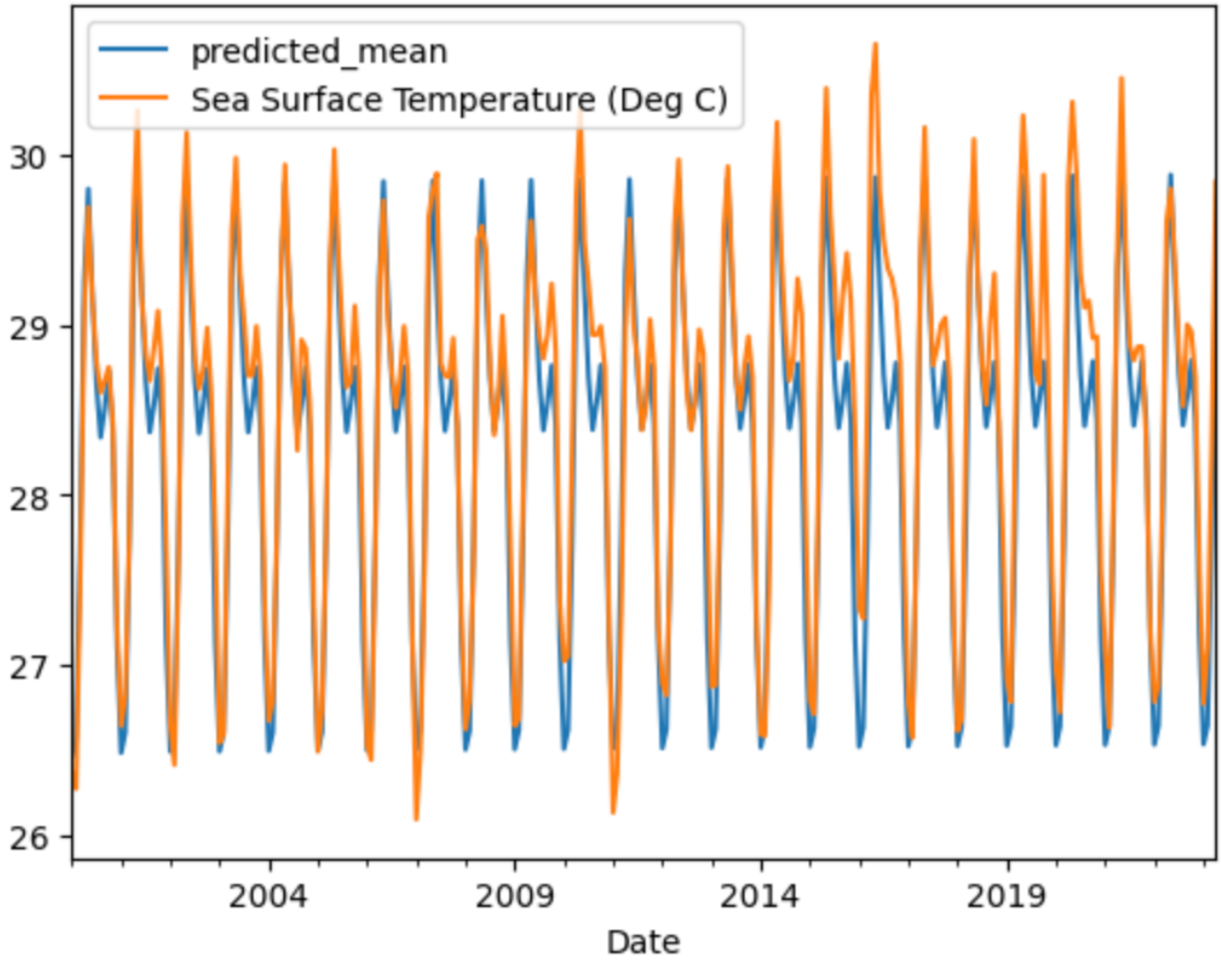


Between the first 12 lags, 5 lags are significantly out of the threshold level. Hence we set the value of the q parameter as 5.

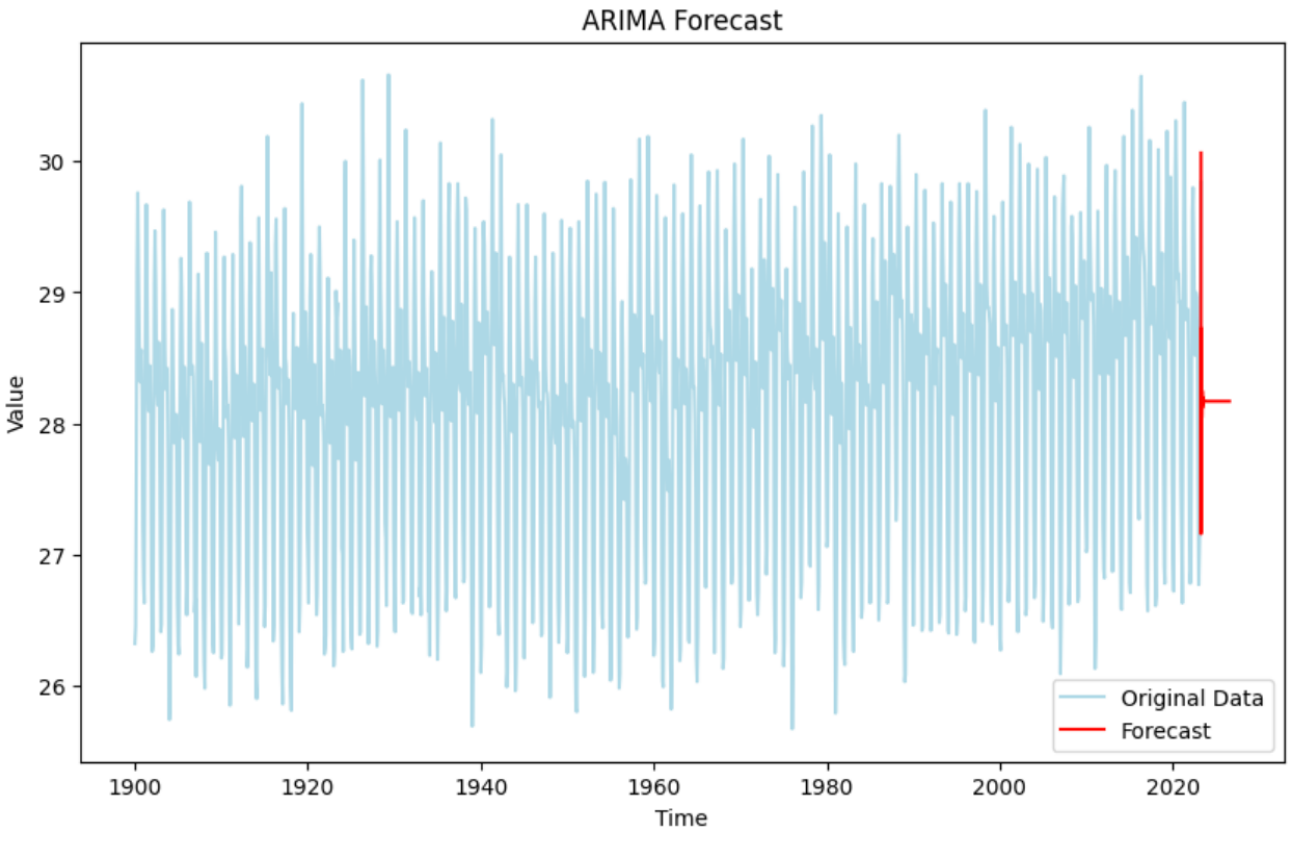
Hence, we get the order for the Arima model as (3, 0, 5).

Then I used the step-wise function to find out the seasonal order for Sarima model and cross-validate the order of the Arima model.

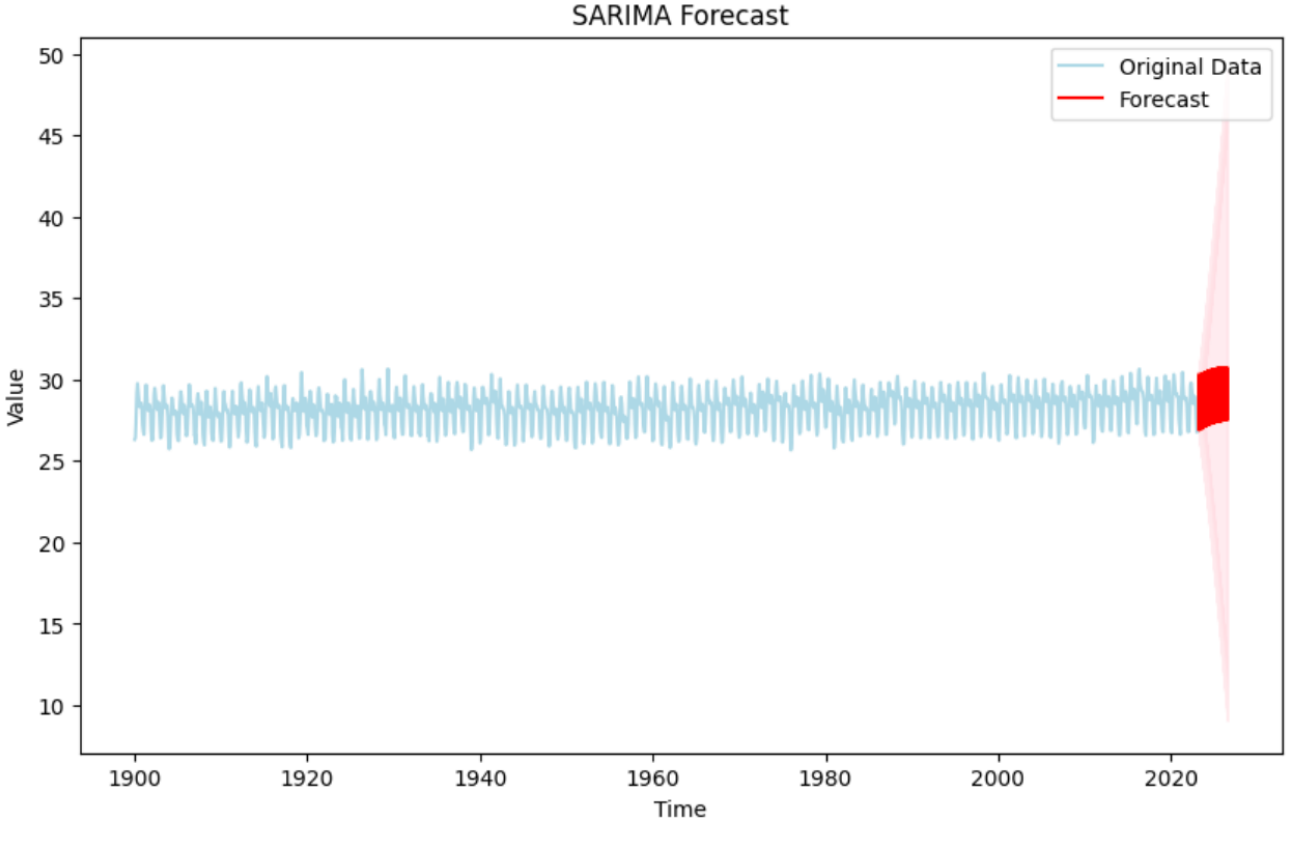
From the analysis, we obtain the values of the parameters of the Arima model as p = 3, d = 0, q = 5 and seasonal order parameters P = 2, D = 0, Q = 2 and M = 12.



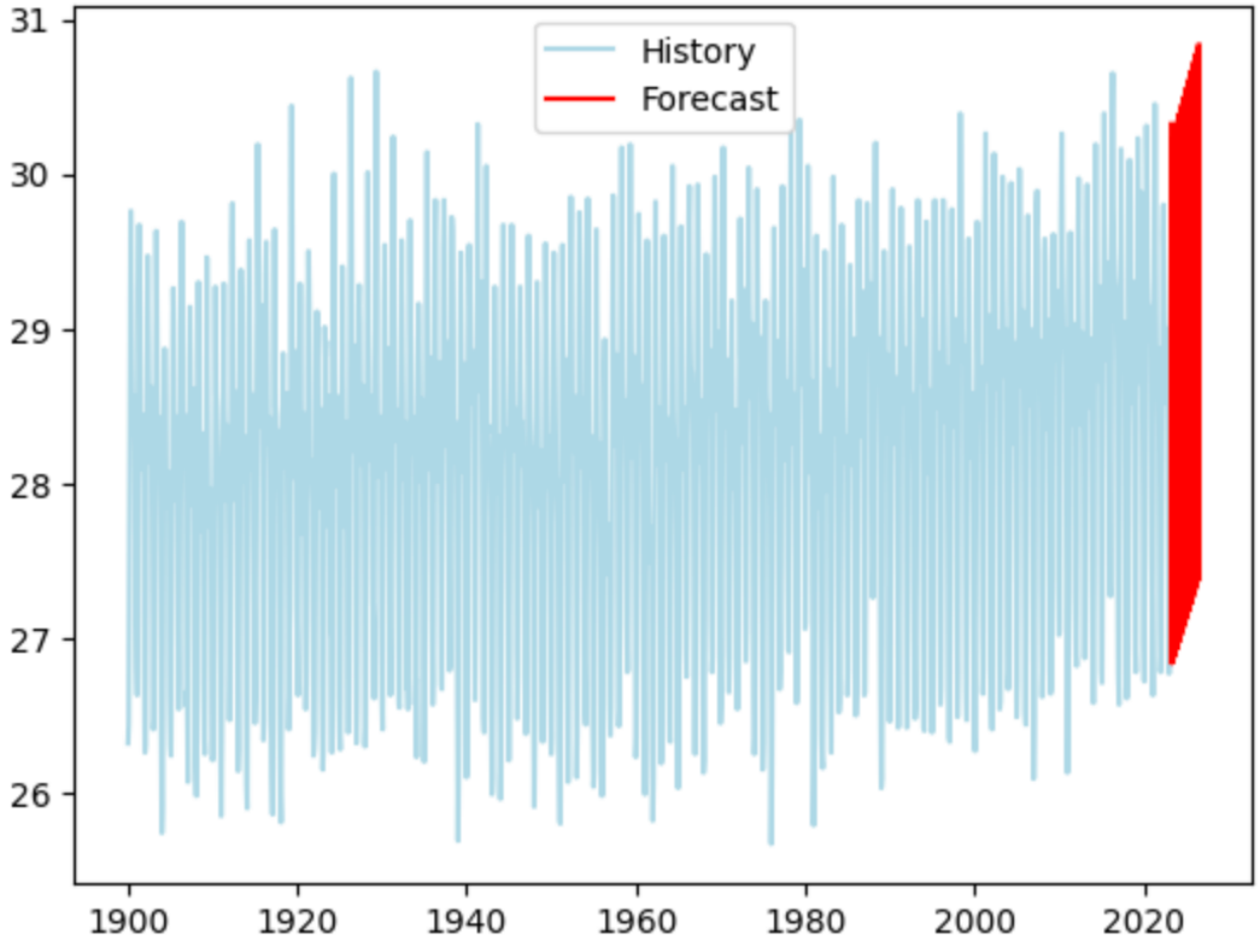
ARIMA FORECAST



SARIMA FORECAST



SARIMAX FORECAST



As expected the Sarimax and Sarima model have performed better than the Arima model.

ACKNOWLEDGEMENT

Thank you Director @INCOIS for giving me this opportunity. I learned a lot of new things and gained lots of knowledge through this project.

I would like to express my heartfelt gratitude and extend my sincerest appreciation to my mentor for the project Dr. Venkata Shesu R, Scientist E, ODICT. This project helped me to implement my coding language skills and learn new libraries. I am truly fortunate to have had the opportunity to work under their guidance. Thank you sir for believing in me and giving me this great opportunity.

I would also like to extend my sincere appreciation to J V S N Raju, his profound knowledge, guidance, and willingness to share insights have been immensely beneficial and have broadened my understanding of the subject matter. The discussions were very insightful and were of great help.

I am also really grateful to V Venu Gopala Rao for his constant guidance and assistance throughout the duration of this project. This project was not possible without him. Thank you for your encouragement.

References:

1) https://towardsdatascience.com/time-series-in-python-exponential-smoothing-and-arima-processes-2c67f2a52788

2) https://analyticsindiamag.com/quick-way-to-find-p-d-and-q-values-for-arima/#:~:text=To%20make%20a%20better%20explanation,I%20and%20q%20is%20MA.

3) https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/

4) https://www.analyticsvidhya.com/blog/2020/10/how-to-create-an-arima-model-for-time-series-forecasting-in-python/

5) file:///C:/Users/Sushant/Downloads/Effectiveness_of_PSO_Based_Neural_Network_for_Seasonal_Time_Series_Forecasting%20(1).PDF

6) https://towardsdatascience.com/time-series-in-python-part-2-dealing-with-seasonal-data-397a65b74051

7) https://statisticsbyjim.com/time-series/autocorrelation-partial-autocorrelation/