Brendan Artley

Summary

This context provides a comprehensive guide to understanding ARIMA-

Time Series Forecasting with ARIMA , SARIMA and SARIMAX

A deep-dive on the gold standard of time series forecasting

Time Series Graph — By Isaac Smith

Time series forecasting is a difficult problem with no easy answer. There are countless statistical models that claim to outperform each other, yet it is never clear which model is best.

That being said, ARMA-based models are often a good model to start with. They can achieve decent scores on most time-series problems and are well-suited as a baseline model in any time series problem.

This article is a comprehensive, beginner-friendly guide to help you understand ARIMA-based models.

The ARIMA model acronym stands for “Auto-Regressive Integrated Moving Average” and for this article we will will break it down into AR, I, and MA.

Autoregressive Component — AR(p)

The autoregressive component of the ARIMA model is represented by AR(p), with the p parameter determining the number of lagged series that we use.

$$v_t = c + \sum^{p} \alpha_n v_{t-n} + \epsilon_t$$

AR Formula — By Author

AR(0): White Noise

If we set the p parameter as zero (AR(0)), with no autoregressive terms. This time series is just white noise. Each data point is sampled from a distribution with a mean of 0 and a variance of sigma-squared. This results in a sequence of random numbers that can’t be predicted. This is really useful as it can serve as a null hypothesis, and protect our analyses from accepting false-positive patterns.

AR(1): Random Walks and Oscillations

timestamp adjusted by a multiplier, and then adding white noise. If the multiplier is 0 then we get white noise, and if the multiplier is 1 we get a random walk. If the multiplier is between $0 < \alpha_1 < 1$, then the time series will exhibit mean reversion. This means that the values tend to hover around 0 and revert to the mean after regressing from it.

AR(p): Higher-order terms

Increasing the p parameter even further is just means going further back and adding more timestamps adjusted by their own multipliers. We can go as far back as we want, but as we get further back it is more likely that we should use additional parameters such as the moving average (MA(q)).

Moving Average — MA(q)

"This component is not a rolling average, but rather the lags in the white noise." — Matt Sosna

MA(q)

MA(q) is the moving average model and q is the number of lagged forecasting error terms in the prediction. In an MA(1) model, our forecast is a constant term plus the previous white noise term times a multiplier, added with the current white noise term. This is just simple probability + statistics, as we are adjusting our forecast based on previous white noise terms.

ARMA and ARIMA architectures are just the AR (Autoregressive) and MA (Moving Average) components put together.

ARMA

The ARMA model is a constant plus the sum of AR lags and their multipliers, plus the sum of the MA lags and their multipliers plus white noise. This equation is the basis of all the models that come next and is a framework for many forecasting models across different domains.

ARIMA

ARIMA Formula — By Author

The ARIMA model is an ARMA model yet with a preprocessing step included in the model that we represent using I(d). I(d) is the difference order, which is the number of transformations needed to make the data stationary. So, an ARIMA model is simply an ARMA model on the differenced time series.

The ARIMA model is great, but to include seasonality and exogenous variables in the model can be extremely powerful. Since the ARIMA model assumes that the time series is stationary, we need to use a different model.

SARIMA

SARIMA Formula — By Author

Enter SARIMA (Seasonal ARIMA). This model is very similar to the ARIMA model, except that there is an additional set of autoregressive and moving average components.The additional lags are offset by the frequency of seasonality (ex. 12 — monthly, 24 — hourly).

SARIMA models allow for differencing data by seasonal frequency, yet also by non-seasonal differencing. Knowing which parameters are best can be made easier through automatic parameter search frameworks such as pmdarina.

ARIMAX and SARIMAX

Sarimax Formula — By Author

Above is the the of the SARIMAX model. This model takes into account exogenous variables, or in other words, use external data in our forecast. Some real-world examples of exogenous variables include gold price, oil price, outdoor temperature, exchange rate.

It is interesting to think that all exogenous factors are still technically indirectly modeled in the historical model forecast. That being said, if we include external data, the model will respond much quicker to its affect than if we rely on the influence of lagging terms.

Code Example

Lets look at these models in actions through a simple code example in Python.

```
import pandas as pd
import matplotlib.pylab as plt
%matplotlib inline
from matplotlib.pylab import rcParams

from statsmodels.tsa.stattools import adfuller
!pip install pmdarima --quiet
import pmdarima as pm
```

Loading Data

For this example, we are going to use the Air Passengers 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 annual seasonality.

As soon as the dataset is read, the index is set to the date. This is standard practice when working with time-series data in Pandas, and makes it easier to implement ARIMA, SARIMA, and SARIMAX.

```
#string to date format
df['Month'] = pd.to_datetime(df['Month'],infer_datetime_format=True)
df = df.set_index(['Month'])
df.head(5)
```

arima import data hosted with ❤ by GitHub view raw

	#Passengers

Cell output — By Author

Trend

their youth. On the other hand, someone on a successful weight loss program will see their weight follow a downward trend over time.

Seasonality + Cycles

Any seasonal or repeating patterns with a fixed frequency. Could be hourly, monthly, daily, annually, etc. One example of this is that Winter Jacket sales increase in the winter months and decrease in the summer months. Another example of this could be the balance of your bank account. In the 10 days at the start of every month, your balance follows a downward trend as you pay monthly rent, utilities, and other bill payments.

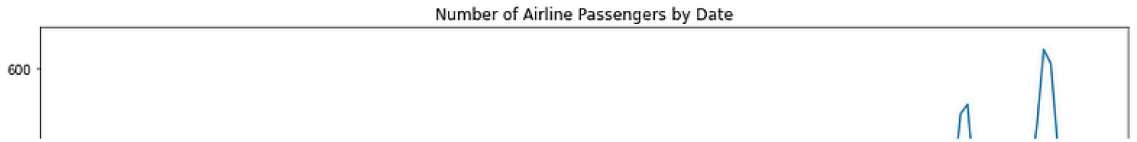
Irregularities + Noise

This is any large spikes or troughs in the data. One example of this could be your heart rate when you run the 400-meter dash. When you start the race your heart rate is similar to what it has been throughout the day, but during the race, it spikes to a much higher level for a small period of time before returning to a normal level.

In the visualization of the airline passenger data below, we can look for these components. At first glance, there looks to be a positive trend and some sort of seasonality or cyclicity in the dataset. There does not appear to be any major irregularities or noise in the data.

```
plt.xlabel('Date')
plt.ylabel('Passengers')
plt.plot(df)
plt.show()
```

arima plot 1 hosted with ❤ by GitHub view raw



Airline Passengers Line Plot — By Author

Rolling Statistics

A rolling average is a great way to visualize how the dataset is trending. As the dataset provides counts by month, a window size of 12 will give us the

We will also include the rolling standard deviation to see how much the data varies from the rolling average.

```
#Determine rolling statistics
df["rolling_avg"] = df["#Passengers"].rolling(window=12).mean() #window size 12 denotes 12
df["rolling_std"] = df["#Passengers"].rolling(window=12).std()

#Plot rolling statistics
plt.figure(figsize=(15,7))
plt.plot(df["#Passengers"], color='#379BDB', label='Original')
plt.plot(df["rolling_avg"], color='#D22A0D', label='Rolling Mean')
plt.plot(df["rolling_std"], color='#142039', label='Rolling Std')
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
```

Rolling Statistics Plot — By Author

The Augmented Dickey-Fuller Test is used to determine if time-series data is stationary or not. Similar to a t-test, we set a significance level before the test and make conclusions on the hypothesis based on the resulting p-value.

Null Hypothesis: The data is not stationary.

Alternative Hypothesis: The data is stationary.

For the data to be stationary (ie. reject the null hypothesis), the ADF test should have:

- p-value <= significance level (0.01, 0.05, 0.10, etc.)

If the p-value is greater than the significance level then we can say that it is likely that the data is not stationary.

We can see in the ADF test below that the p-value is 0.991880, meaning that it is very likely that the data is not stationary.

```
dftest = adfuller(df['#Passengers'], autolag='AIC')

dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags Used','Number of
for key,value in dftest[4].items():
    dfoutput['Critical Value (%s)'%key] = value

print(dfoutput)
```

ADF Test Output — By Author

ARIMA Model Selection w/ Auto-ARIMA

Using the auto_arima() function from the `pmdarima` package, we can perform a parameter search for the optimal values of the model.

```
#Standard ARIMA Model
ARIMA_model = pm.auto_arima(df['#Passengers'],
                      start_p=1,
                      start_q=1,
                      test='adf', # use adftest to find optimal 'd'
                      max_p=3, max_q=3, # maximum p and q
                      m=1, # frequency of series (if m==1, seasonal is set to FALSE automat
                      d=None,# let model determine 'd'
                      seasonal=False, # No Seasonality for standard ARIMA
                      trace=False, #logs
                      error_action='warn', #shows errors ('ignore' silences these)
                      suppress_warnings=True,
                      stepwise=True)
```

Standard arima hosted with ❤ by GitHub view raw

Model Diagnostics

Four plots result from the plot_diagnostics function. The Standardized residual, Histogram plus KDE estimate, Normal q-q, and the correlogram.

We can interpret the model as a good fit based on the following conditions.

There are no obvious patterns in the residuals, with values having a mean of zero and having a uniform variance.

Histogram plus KDE estimate

The KDE curve should be very similar to the normal distribution (labeled as N(0,1) in the plot)

Normal Q-Q

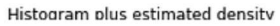
Most of the data points should lie on the straight line

Correlogram (ACF plot)

95% of correlations for lag greater than zero should not be significant. The grey area is the confidence band, and if values fall outside of this then they are statistically significant. In our case, there are a few values outside of this area, and therefore we may need to add more predictors to make the model more accurate

plot_diagnostics hosted with ❤ by GitHub

view raw



We can then use the model to forecast airline passenger counts over the next 24 months.

As we can see from the plot below, this doesn't seem to be a very accurate forecast. Maybe we need to change the model structure so that it takes into account seasonality?

```
def forecast(ARIMA_model, periods=24):
    # Forecast
    n_periods = periods
    fitted, confint = ARIMA_model.predict(n_periods=n_periods, return_conf_int=True)
    index_of_fc = pd.date_range(df.index[-1] + pd.DateOffset(months=1), periods = n_periods

    # make series for plotting purpose
    fitted_series = pd.Series(fitted, index=index_of_fc)
    lower_series = pd.Series(confint[:, 0], index=index_of_fc)
    upper_series = pd.Series(confint[:, 1], index=index_of_fc)

```

ARIMA Forecast — By Author

SARIMA Model

Now let's try the same strategy as we did above, except let's use a SARIMA model so that we can account for seasonality.

```
# Seasonal - fit stepwise auto-ARIMA
SARIMA_model = pm.auto_arima(df["#Passengers"], start_p=1, start_q=1,
                         test='adf',
                         max_p=3, max_q=3,
                         m=12, #12 is the frequncy of the cycle
                         start_P=0,
                         seasonal=True, #set to seasonal
                         d=None,
                         D=1, #order of the seasonal differencing
                         trace=False,
                         error_action='ignore'
```

differences when compared with the standard ARIMA model.

Standardized residual

The Standardized residual is much more consistent across the graph, meaning that the data is closer to being stationary.

Histogram plus KDE estimate

The KDE curve is similar to the normal distribution (not much changed here).

Normal Q-Q

The data points are clustered much closer to the line than in the ARIMA diagnostic plot.

Correlogram (ACF plot)

The grey area is the confidence band, and if values fall outside of this then they are statistically significant. We want all values inside this area. Adding the seasonality component did this! All the points now fall within the 95% confidence interval.

asdasd hosted with ❤ by GitHub

view raw

We can then use the model to forecast airline passenger counts over the next 24 months as we did before.

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

```
1  forecast(SARIMA_model)
```

asdasd hosted with ❤ by GitHub view raw

SARIMA Forecast — By Author

SARIMAX Model Selection

Now let's practice adding in an exogenous variable. In this example, I am simply going to add the month number as an exogenous variable, but this is not super useful as this is already conveyed through the seasonality.

Note that we are adding additional square brackets around the data being passed into the SARIMAX model.

```
3
4  # SARIMAX Model
5  SARIMAX_model = pm.auto_arima(df[['#Passengers']], exogenous=df[['month_index']],
6                                start_p=1, start_q=1,
7                                test='adf',
8                                max_p=3, max_q=3, m=12,
9                                start_P=0, seasonal=True,
10                               d=None, D=1,
11                               trace=False
```

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.

```
1  def sarimax_forecast(SARIMAX_model, periods=24):
2      # Forecast
3      n_periods = periods
4
5      forecast_df = pd.DataFrame({"month_index":pd.date_range(df.index[-1], periods = n_perio
6                      index = pd.date_range(df.index[-1] + pd.DateOffset(months=1), periods =
7
8      fitted, confint = SARIMAX_model.predict(n_periods=n_periods,
9                                              return_conf_int=True,
10                                             exogenous=forecast_df[['month_index']])
11     index_of_fc = pd.date_range(df.index[-1] + pd.DateOffset(months=1), periods = n_periods
```

SARIMAX Forecast — By Author

Closing Thoughts

Please find the code for this article here.

Putting ideas into my own words and implementing ARIMA models hands-on is the best way to learn. Hopefully this article can motivate others to do the same.

ARIMA model architectures provide more explainability than RNN's, yet RNN's are known to generate more accurate predictions. Now I have a good grasp on the ARIMA model architecture, I need to look into LSTM and RNN deep learning models for forecasting time series data!

Further Reading

sources. Thank you to all for sharing!

A Deep Dive on Arima Models — by Matt Sosna ← MUST READ!

Time Series For beginners with ARIMA — by @Arindam Chatterjee

Arima Model for Time Series Forecasting — by @Prashant Banerjee

StatsModels ADF Documentation

Removing Trends and Seasonality Article — by Jason Brownlee

A Gentle Introduction to SARIMA — by Jason Brownlee

Time Series Forecasting | Arima | Machine Learning | Statistics | Time Series Analysis

---

Chris Yan

Understanding SARIMAX: An Seasonal Time Series Forecasting Technique

SARIMAX, or Seasonal AutoRegressive Integrated Moving Average with eXogenous factors, is a powerful extension of the ARIMA model that...

5 min read

Ahmad Waleed

Forecasting Time Series Data with SARIMAX: A Step-by-Step Guide

Time series forecasting plays a pivotal role in fields like finance, economics, and weather prediction, providing insights that drive...

5 min read

sophiamsac

ARIMA vs. SARIMA vs. SARIMAX

Exploring Time Series Forecasting Models

5 min read

Chris Kuo/Dr. Dataman

Automatic ARIMA!

Automatic model selection and multi-step forecasting

Jonas Dieckmann

Getting Started Predicting Time Series Data with Facebook Prophet

This article aims to take away the entry barriers to get started with time series analysis in a hands-on tutorial using Prophet

9 min read

Ganesh Bajaj

Time Series Analysis: Interpretation of ACF and PACF Plots

Autocorrelation (ACF) and Partial Autocorrelation (PACF) plots are powerful tools for uncovering hidden patterns in time series data...

5 min read

Kishan A

ARIMA vs SARIMA vs SARIMAX vs Prophet for Time Series Forecasting

Time series forecasting is a crucial tool in various industries like retail, finance, and healthcare, allowing businesses and researchers...

5 min read

Irina (Xinli) Yu, Ph.D.

environmental science, and operations...

5 min read

L Data PR

ARIMA Models in R part-1

Hello Guys,

4 min read

 Abhishek Jha

Understanding Statistical Significance: Myths and Realities in Data Science

Scientific debates around statistical significance testing have intensified due to the replication crisis. While some researchers argue for...

5 min read