# Guide to Time Series Analysis with Python — 3: Autoregressive Process

[[Buse Köseoğlu](https://buse-koseoglu13.medium.com/?source=post_page-----3f16857d3e8a--------------------------------)](https://buse-koseoglu13.medium.com/?source=post_page-----3f16857d3e8a--------------------------------)

[Buse Köseoğlu](https://buse-koseoglu13.medium.com/?source=post_page-----3f16857d3e8a--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F5e1a64153870&operation=register&redirect=https%3A%2F%2Fbuse-koseoglu13.medium.com%2Fguide-to-time-series-analysis-with-python-3-autoregressive-process-3f16857d3e8a&user=Buse+K%C3%B6seo%C4%9Flu&userId=5e1a64153870&source=post_page-5e1a64153870----3f16857d3e8a---------------------post_header-----------)

4 min read

·

Aug 8

111

In the previous article, we examined the moving average process. This time we will examine another statistical model, the autoregressive process.

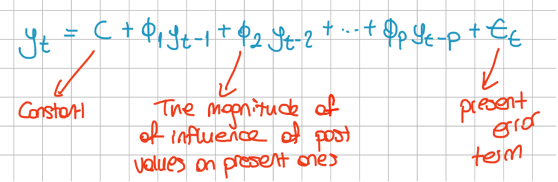
You can find previous articles below.  
[1.Guide to Time Series Analysis with Python — 1: Analysis Techniques and Baseline Model](https://medium.com/@buse-koseoglu13/guide-to-time-series-analysis-with-python-1-analysis-techniques-and-baseline-model-59e12fdd2eba)

[2.Guide to Time Series Analysis with Python — 2: Moving Average Process](https://medium.com/@buse-koseoglu13/guide-to-time-series-analysis-with-python-2-moving-average-process-784328325e5f)

If you’re ready, let’s start.

# What is Autoregressive Process (AR)?

Autoregressive process is a statistical model used for time series to predict the future by looking at historical data. Just as the moving average process uses the effect of past errors, the autoregressive process says that the past values in the time series affect the present. We can see how this is used in the equation below. Here, the parameter we want to determine is the “p” parameter. This p-value decides how far back we go. It is important to find the optimum number.



# Specifying the p Parameter

We use a method like the method in the moving average process to determine this parameter. But this time we will use PACF (Partial Autocorrelation Function) plot instead of ACF plot. PACF gives the partial correlation of the time series with its lagged values.  
If we examine the roadmap below, we can see how we can reach the result step by step. Here, first of all, we need to see if the data fits the moving average process. If it does not fit, then we need to check whether it is suitable for the autoregressive process with the PACF graph.

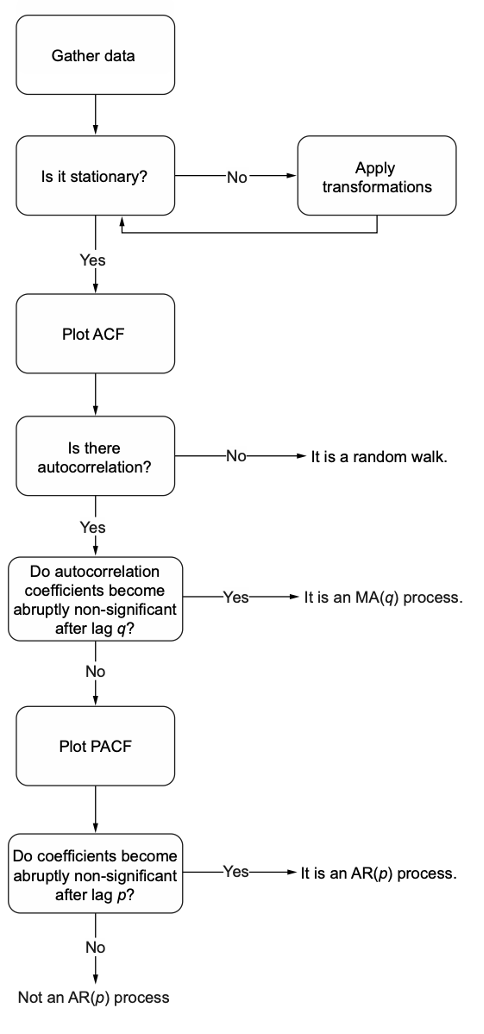
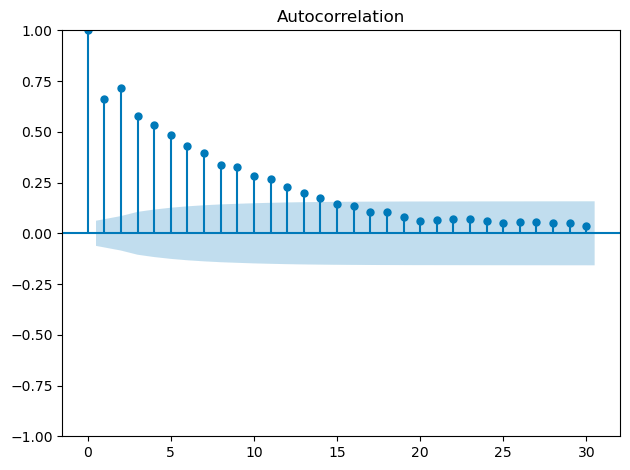


Figure from Time series forecasting in python — Marco Peixerio

Let’s say we have stationary data. Then, let’s draw an ACF plot with this data, and as a result, we get the following graph.

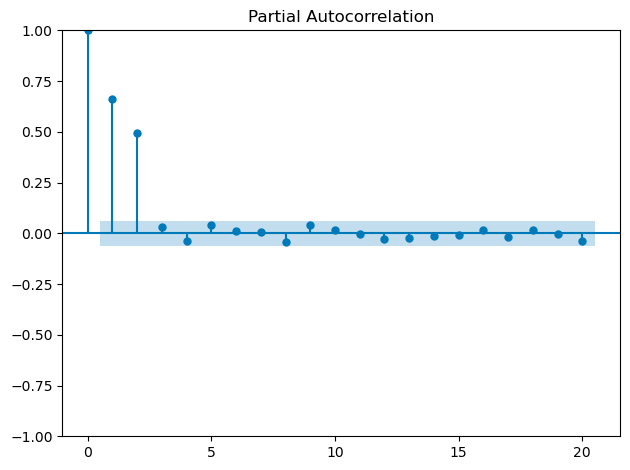
plot\_acf(ar2\_process)  
  
plt.tight\_layout()



ACF Plot

When the above ACF graph is examined, we can see that there are coefficients that are significant after lag 0. When examining this graph in the moving average process, we expected only 1 significant coefficient and the other coefficients to remain within the confidence interval (blue area). Since there is no such situation here, we can say that this data is not suitable for the moving average process and draw the PACF graph.

plot\_pacf(ar2\_process, lags=20)  
  
plt.tight\_layout()



PACF Plot

If we examine the PACF graph above, we can see that the coefficients after lag 2 are non-significant. At the same time, with this graph, we determine that the “p” parameter is also 2, and we have the AR(2) process.

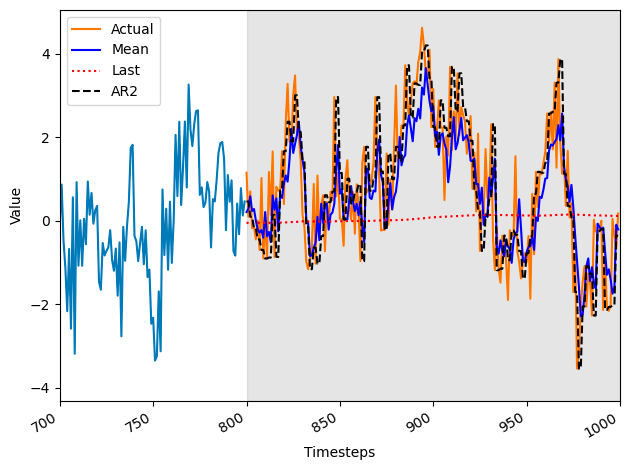
# Modeling AR(2)

The next processes are very similar to the moving average process. We can implement the AR(2) process with only minor changes. After separating the data as train and test, we can use the following function for modeling.

def rolling\_forecast(df: pd.DataFrame, train\_len: int, horizon: int, window: int, method: str) -> list:  
   
 total\_len = train\_len + horizon  
 end\_idx = train\_len  
   
 if method == 'mean':  
 pred\_mean = []  
   
 for i in range(train\_len, total\_len, window):  
 mean = np.mean(df[:i].values)  
 pred\_mean.extend(mean for \_ in range(window))  
   
 return pred\_mean  
  
 elif method == 'last':  
 pred\_last\_value = []  
   
 for i in range(train\_len, total\_len, window):  
 last\_value = df[:i].iloc[-1].values[0]  
 pred\_last\_value.extend(last\_value for \_ in range(window))  
   
 return pred\_last\_value  
   
 elif method == 'AR':  
 pred\_AR = []  
   
 for i in range(train\_len, total\_len, window):  
 model = SARIMAX(df[:i], order=(2,0,0))  
 res = model.fit(disp=False)  
 predictions = res.get\_prediction(0, i + window - 1)  
 oos\_pred = predictions.predicted\_mean.iloc[-window:]  
 pred\_AR.extend(oos\_pred)  
   
 return pred\_AR

The important part here is the SARIMAX(df[:i], order=(2,0,0)) part in the “AR” condition. In the MA(2) process, the order was written as (0,0,2). In AR(2), it is written as (2,0,0).

Actually, the process is that simple. Both MA and AR contain very similar implementations. You can find the codes for both in the[githup repo](https://github.com/busekoseoglu/time-series-forecasting). If we examine the AR model and base model results, we can see the graph below. Here we can see that the autoregressive process captures highs and lows better than the base models and works better than the base model as we expected.



Thank you for reading. See you in the next article.

# References

* Time series forecasting in python — Marco Peixerio
* Tan,Y.-F.;Ong,L.-Y.;Leow, M.-C.; Goh, Y.-X. Exploring Time-Series Forecasting Models for Dynamic Pricing in Digital Signage Advertising. Future Internet 2021, 13, 241. <https://doi.org/10.3390/> fi13100241
* Machine learning with time series — Markus Löning
* <https://www.influxdata.com/time-series-forecasting-methods/>
* <https://builtin.com/data-science/time-series-forecasting-python>
* <https://www.machinelearningplus.com/time-series/time-series-analysis-python/>
* <https://towardsdatascience.com/interpreting-acf-and-pacf-plots-for-time-series-forecasting-af0d6db4061c>
* <https://en.wikipedia.org/wiki/Partial_autocorrelation_function>