# MODELING NON-STATIONARITY AND FINDING EQUILIBRIUM

Project 3 | By Group 361

**Submission No.** 3

**Contributers:**

1. Joel Debo Adriko (adrikodebojoel1@gmail.com)
2. Ronald Gonye (ronny.gonye@gmail.com)
3. Alper Ulku (alperulku1970@gmail.com)

---

# Abstract

Non-stationarity is common in financial time series dataset and handling it properly serves a great deal in creating value out of forecasting activities. However to better deal with non-stationarity, the analyst need to understand various aspects partaining non-stationarity. In this paper we slowly define and explain what non-stanarity is and how to test for various types of stationarity. We also show the damages caused by non-stationarity and a direction on how to handle these damages. We end this paper by handling forecasting using ARIMA model while determining equilibrium model using AIC.

**Keywords:-** Non-stationarity, Stationarity, ARIMA, MA, differencing-method, ADF test, KPSS test, trend stationarity, AIC, forecasting.

# INTRODUCTION

First, it's important that the reader understands that in Project 2 under "Non-Stationarity", we discussed about non-stationarity and we therefore start by briefly reviewing the features of non-stationary dataset. We then talk about types of stationarity (which we never discussed in in Project 2 under "Non-Stationarity"). From there, we shall handle stationarity tests using ADF and KPSS tests (which also we never discussed in in Project 2 under "Non-Stationarity") before demomnstrating damages and directions using Moving Averages and differencing method. Finally we do forecasting.

The equilibrium component of this paper, as discussed below, involves using Akaike's information criterion (AIC) to select the efficient ARIMA model.

In [1]:
```
# import sys
# !{sys.executable} -m pip install pmdarima
```

In [15]:
```
import matplotlib.pyplot as plt
```

```
import numpy as np
import pandas as pd
import statsmodels
import statsmodels.api as sm
import pylab
import statsmodels.formula.api as smf
from pmdarima.arima import auto_arima
from statsmodels.tsa.arima.model import ARIMA
import warnings
warnings.filterwarnings('ignore')
from arch.unitroot import ADF, KPSS

plt.rcParams["figure.figsize"] = (16, 9)  # Figure size and width
```

In [3]:
```
import os
os.getcwd()
os.chdir(
    "F:/Financial Engineer/Financial-econometrics-group-work/project-3"
)
```

In [4]:
```
#Read stock price dataset
df = pd.read_csv("./data/stock_prices.csv")

#Convert date feature into date format
df["Date1"] = pd.to_datetime(df["Date"], format="%Y-%m-%d")

meta_stock_prices = df.loc[:, ["Date1", "META"]].set_index("Date1")
```

# DEFINITIONS AND DESCRIPTIONS

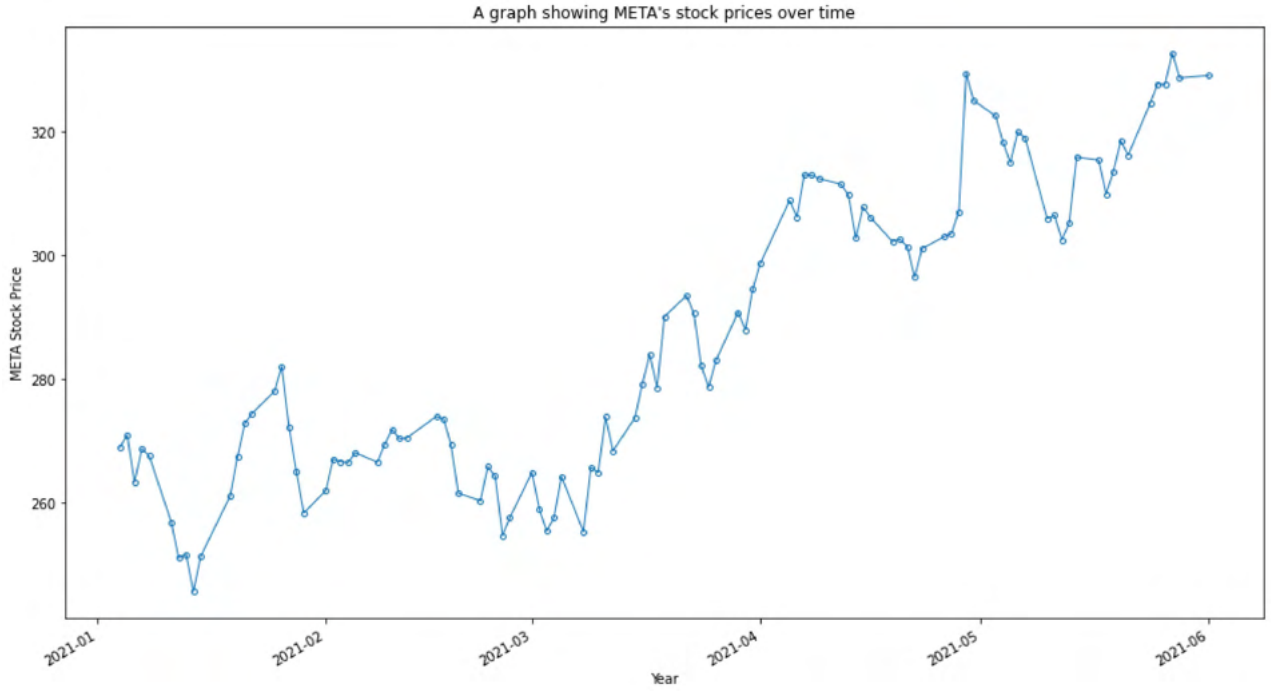
As we defined in Project 2 under "Non-Stationarity", non-stationarity is a condition whereby data values in datasets have a changing mean, variances and covariances over time.

In Project 2 under "Non-Stationarity", we discussed and demonstrated the three behaviours of non-stationary dataset and these are briefly discussed again below using the same META stock prices. Later on, we are going to discuss **trend stationarity** and **unit root** characteristics of non-stationary dataset.

## Demonstration of non-stationary behaviors using META stock prices

In [16]:
```
# Plot Google price time series chart
meta_stock_prices["META"].plot(
    marker="o",
    markersize=4,
    markerfacecolor="none",
    linestyle="-",
    linewidth=1,
    xlabel="Year",
    ylabel="META Stock Price",
```

```
    title="A graph showing META's stock prices over time"
)
plt.show()
```




We can observe same attributes are we noticed with **META's stock prices** in Project 2 under "Non-Stationarity", that is we see:-

1. Cyclic upward movement of stock prices in different seasons for example from mid January 2021 to Febraury 2021, and from April 2021 to May 2021, depicting seasonality.
2. Building on above, we can observe upward trend
3. That the next data values are dependent on the preceeding data values including the effect of shocks. This create a random walk.

# Types of stationarity

According to Singh, we have three types of stationarity and we discuss them below. The discussion of these types of stationarity is not only important in understanding the damages caused by non-stationarity but also how to handle the damage caused by non-stationarity.

## Strict stationarity

According to Singh, strict stationary time series is a time series in which mean, variance and covariance are not part of time function.

## Trend stationarity

Trend stationary dataset is a time series which has no unit root but possesses a trend. Therefore, once we eliminate the trend, the time series dataset become strict stationary. In this sense a trend stationary dataset is a non-stationary dataset.

Note that we discuss unit root below.

### Difference stationarity

According to Singh, when we take the first-difference of a difference-stationary time series, the time series becomes a strict stationary. Similarly, difference-stationary time series is a non-stationary time serie

## Unit Root

According to Herranz, **unit roots** are nonstationary autoregressive (AR) or autoregressive moving-average (ARMA) time series processes with the simplest example as a random walk.

Mathematically a unit root can be defined as below:-

$$x_t = \alpha_0 + \alpha_1 x_{t-1} + v_t \tag{1}$$

Where:-

$$x_t \text{ follows an AR}(1) \text{ process}$$

$$v_t \text{ is a stationary ARMA process with mean} = 0$$

$$|\alpha_1| = 1$$

## Examples of unit roots

We now define two examples of units based on the `equation(1)` and $\alpha_0$ as defined below.

### Random walk

This is a unit root where $\alpha_0 = 0$

### Random walk with drift

This is a unit root where $\alpha_0 \neq 0$

# DIAGNOSIS AND TESTS

In this section we are goin to show how to test for non-stationarity using two methods: **Augumented Dickey Fuller (ADF) test** and **Kwaitkowski-Phillips-Scmidt-Shin (KPSS) test**. The ADF test will be used to test for unit root while the KPSS test will complement the ADF test by testing for the time trend. Each test is briefly decsribed before carrying out the respective test(s).

## Augumented Dickey Fuller (ADF) test

According to Dickey & Fuller, The Dickey-Fuller test in statistics examines the possibility that a unit root exists in an autoregressive time series model. Depending on the test version being utilized, the alternative hypothesis may vary, but it is typically stationarity or trend-stationarity. The statisticians David Dickey and Wayne Fuller, who created the test in 1979, are remembered by the test's name. (1)

According to Glossary of Economics Research, An augmented Dickey-Fuller test (ADF), used in statistics and economics, examines the possibility that a unit root exists in a time series sample. It is a supplement to the Dickey-Fuller test for a more extensive and intricate collection of time series models. (2)

Augmented Dickey-Fuller (ADF) test statistic is a negative value. The greater the rejection of the unit root hypothesis is, at least at some level of confidence, the more negative it is. The Dickey-Fuller test's testing process is used for the ADF test as well, model equation is as follows:

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t,$$

where alpha is the lag order of the autoregressive process, beta is the coefficient on a time trend, and alpha is a constant. Modeling a random walk with the constraints alpha=0 and beta =0 is equivalent to modeling a random walk without a drift. Using the constraint beta =0 is equivalent to modeling a random walk.

The ADF formulation allows for higher-order autoregressive processes by incorporating lags of the order p. As a result, while using the test, the lag duration p must be established. Examining the t-values on coefficients while testing down from high orders is one strategy that could work. Examining information criteria like the Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), or Hannan-Quinn information criterion is an alternative strategy.

The unit root test is then performed with gamma = 0 as the null hypothesis and gamma < 0 as the alternative hypothesis, so that:

$$\mathrm{DF}_\tau = \frac{\hat{\gamma}}{\mathrm{SE}(\hat{\gamma})}$$

We only care about negative values of our test statistic (DF tau)) because this test is asymmetrical. Gamma = 0 is rejected as the null hypothesis and there is no unit root if the estimated test statistic is less (more negative) than the critical value.

Below we conduct unit root test using **Augumented Dickey Fuller (ADF) test**

# Unit root tests using ADF tests

Below we demonstrate how to test for unit root in a time series to check whether the time series is non-stationary through three tests. We do the test on META stock prices. The null hypothesis of

each test is listed before the test.

Generally, we will use **5% critical value** and compare this against the **ADF's test statistic**. We will accept the null hypothesis $H_0$ if the ADF's test statistic is greater than the 5% critical value.

## META Stock Price ADF Test without Drift and Trend

$H_0$=There is a unit root in META stock price

In [6]:
```
# AAPL Stock Price, ADF Test without Drift and Trend
adf_none = ADF(meta_stock_prices, trend="n", method="bic")
print("Augmented Dickey-Fuller Unit Root Test\n", adf_none.regression.summary())
print("\nTest statistics and critical values: \n", adf_none)
```

```
Augmented Dickey-Fuller Unit Root Test
                                OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared (uncentered):                   0.009
Model:                            OLS   Adj. R-squared (uncentered):             -0.000
Method:                 Least Squares   F-statistic:                              0.9651
Date:                Tue, 04 Oct 2022   Prob (F-statistic):                       0.328
Time:                        07:57:06   Log-Likelihood:                         -322.17
No. Observations:                 102   AIC:                                      646.3
Df Residuals:                     101   BIC:                                      649.0
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Level.L1       0.0019      0.002      0.982      0.328      -0.002       0.006
==============================================================================
Omnibus:                        5.711   Durbin-Watson:                   2.024
Prob(Omnibus):                  0.058   Jarque-Bera (JB):                5.956
Skew:                           0.348   Prob(JB):                       0.0509
Kurtosis:                       3.958   Cond. No.                         1.00
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a con
stant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specifi
ed.

Test statistics and critical values:
    Augmented Dickey-Fuller Results
====================================
Test Statistic                  0.982
P-value                         0.913
Lags                                0
------------------------------------

Trend: No Trend
Critical Values: -2.59 (1%), -1.94 (5%), -1.61 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

From the ADF test results above, the test statistic for META's stock price without drift and trend is

0.982, which is greater than 5% critical value of -1.94. Therefore we fail to reject $H_0$ and conclude that there is a unit root in **META**'s stock price.

Next, we run the second unit root test: **ADF test with drift without trend.**

## META Stock Price ADF Test with Drift but no Trend

$H_0$=There is a unit root and drift in APPLE stock price

In [7]:
```
# ADF test with drift but no trend
adf_drift = ADF(meta_stock_prices, trend="c", method="bic")
print("Augmented Dickey-Fuller Unit Root Test\n", adf_drift.regression.summary())
print("\nTest statistics and critical values: \n", adf_drift)
```

```
Augmented Dickey-Fuller Unit Root Test
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.005
Model:                            OLS   Adj. R-squared:                 -0.005
Method:                 Least Squares   F-statistic:                    0.4558
Date:                Tue, 04 Oct 2022   Prob (F-statistic):              0.501
Time:                        07:57:06   Log-Likelihood:                -321.87
No. Observations:                 102   AIC:                             647.7
Df Residuals:                     100   BIC:                             653.0
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Level.L1      -0.0163      0.024     -0.675      0.501      -0.064       0.032
const          5.2508      6.927      0.758      0.450      -8.492      18.993
==============================================================================
Omnibus:                        6.543   Durbin-Watson:                   1.999
Prob(Omnibus):                  0.038   Jarque-Bera (JB):                7.589
Skew:                           0.349   Prob(JB):                       0.0225
Kurtosis:                       4.140   Cond. No.                     3.51e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
ed.
[2] The condition number is large, 3.51e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

Test statistics and critical values:
    Augmented Dickey-Fuller Results
=====================================
Test Statistic                 -0.675
P-value                         0.853
Lags                                0
-------------------------------------

Trend: Constant
Critical Values: -3.50 (1%), -2.89 (5%), -2.58 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

From the results, the ADF test statistic is -0.675, and the 5% critical value is -2.89. We also fail to

reject the $H_0$, and conclude that there is a unit root and a drift in META's stock price, since the ADF test statistic is greater that the 5% critical value.

We then run the third test: **the ADF test with both drift and trend**

## META Stock Price ADF Test with Drift and Trend

$H_0$=There is a unit root, drift and trend in META stock price

In [8]:
```
# ADF test with drift and trend
adf_trend = ADF(meta_stock_prices, trend="ct", method="bic")
print("Augmented Dickey-Fuller Unit Root Test\n", adf_trend.regression.summary())
print("\nTest statistics and critical values: \n", adf_trend)
```

```
Augmented Dickey-Fuller Unit Root Test
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.090
Model:                            OLS   Adj. R-squared:                  0.071
Method:                 Least Squares   F-statistic:                     4.880
Date:                Tue, 04 Oct 2022   Prob (F-statistic):            0.00953
Time:                        07:57:07   Log-Likelihood:                -317.31
No. Observations:                 102   AIC:                             640.6
Df Residuals:                      99   BIC:                             648.5
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Level.L1      -0.1547      0.051     -3.031      0.003      -0.256      -0.053
const         38.5181     12.797      3.010      0.003      13.126      63.910
trend          0.1244      0.041      3.044      0.003       0.043       0.205
==============================================================================
Omnibus:                        6.734   Durbin-Watson:                   1.904
Prob(Omnibus):                  0.034   Jarque-Bera (JB):                8.659
Skew:                           0.307   Prob(JB):                       0.0132
Kurtosis:                       4.288   Cond. No.                     6.86e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
ed.
[2] The condition number is large, 6.86e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

Test statistics and critical values:
    Augmented Dickey-Fuller Results
=====================================
Test Statistic                -3.031
P-value                        0.124
Lags                               0
-------------------------------------

Trend: Constant and Linear Time Trend
Critical Values: -4.05 (1%), -3.45 (5%), -3.15 (10%)
Null Hypothesis: The process contains a unit root.
Alternative Hypothesis: The process is weakly stationary.
```

The ADF test for a model with drift and trend has test statistic -3.031 which is more than the 5% critical value of -3.45. This is an indication that we cannot reject $H_0$.

## General conclusion about unit root test

All the tests show the ADF test's statistic greater than the 5% critical value level for all three models; therefore we can conclude that META's stock prices has unit root and is non-stationary dataset. However, we need to test for time trend in the META's stock prices and we do this by using **Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test** below. But first let's define KPSS.

# Kwiatkowski-Phillips-Schmidt-Shin (KPSS) Test

The **KPSS test** complements the unit root test and is used to determine if the time series has a time trend.

The hypotheses of **KPSS test** are listed below:-

> $H_0$ **:** the time series dataset is trend stationary
> $H_1$ **:** the time series dataset is not trend stationary

Generally, we are going to use the **5% critical value** against the **p-value**, and we are going to reject the $H_0$ if the p-value is less than the 5% critical value.

Let's now run a KPSS test for META stock prices to see whether it is a trend stationary time series.

## KPSS test for META's stock prices

In [9]:
```
# KPSS test
print(KPSS(meta_stock_prices, trend="ct", lags=-1))
```

```
    KPSS Stationarity Test Results
=====================================
Test Statistic                  0.131
P-value                         0.074
Lags                               13
-------------------------------------

Trend: Constant and Linear Time Trend
Critical Values: 0.22 (1%), 0.15 (5%), 0.12 (10%)
Null Hypothesis: The process is weakly stationary.
Alternative Hypothesis: The process contains a unit root.
```

The KPSS test results show that the $p$ -value (0.131) is less than 5% critical value (0.15) and there we reject $H_0$ and we conclude that META's stock price is no trend stationary.

# Overall conclusion on the tests

From the ADF tests and KPSS test above we conclude that META's stock prices have unit root and is not trend stationary and we can consequently apply unit root remedies.

# DAMAGES AND DIRECTION

Iordanova explains that, as an industry rule, non-stationary time series data are unpredictable and cannot easily be modeled or forecasted. He (Iordanova) continues to explain that trying to build a model using non-stationary dataset may give results that show an existence of relationship between two variable when in actual sense such a relationship is inexistent.

Below we use the same **moving average** model that we used in Project 2 under "Non-Stationarity", to show the damage done by non-stationary time series. We shall later, contrast this by a transformed stationary dataset

## Moving Average [MA(1)] model with original META stock prices

In [17]:
```
# MA(1) for original META stock prices
meta_original = statsmodels.tsa.arima.model.ARIMA(meta_stock_prices.META, order=(0, 0,

# Plot Original Meta prices vs fitted Meta stock prices
meta_original_residuals = meta_original.resid
meta_original_fitted_values = meta_stock_prices["META"] - meta_original_residuals

meta_stock_prices["META"].plot(linewidth=0.8, label="Original META stock price")
meta_original_fitted_values.plot(linewidth=0.8, label="Fitted META stock price")
plt.xlabel("Time")
plt.ylabel("Price")
plt.legend()
plt.show()
```

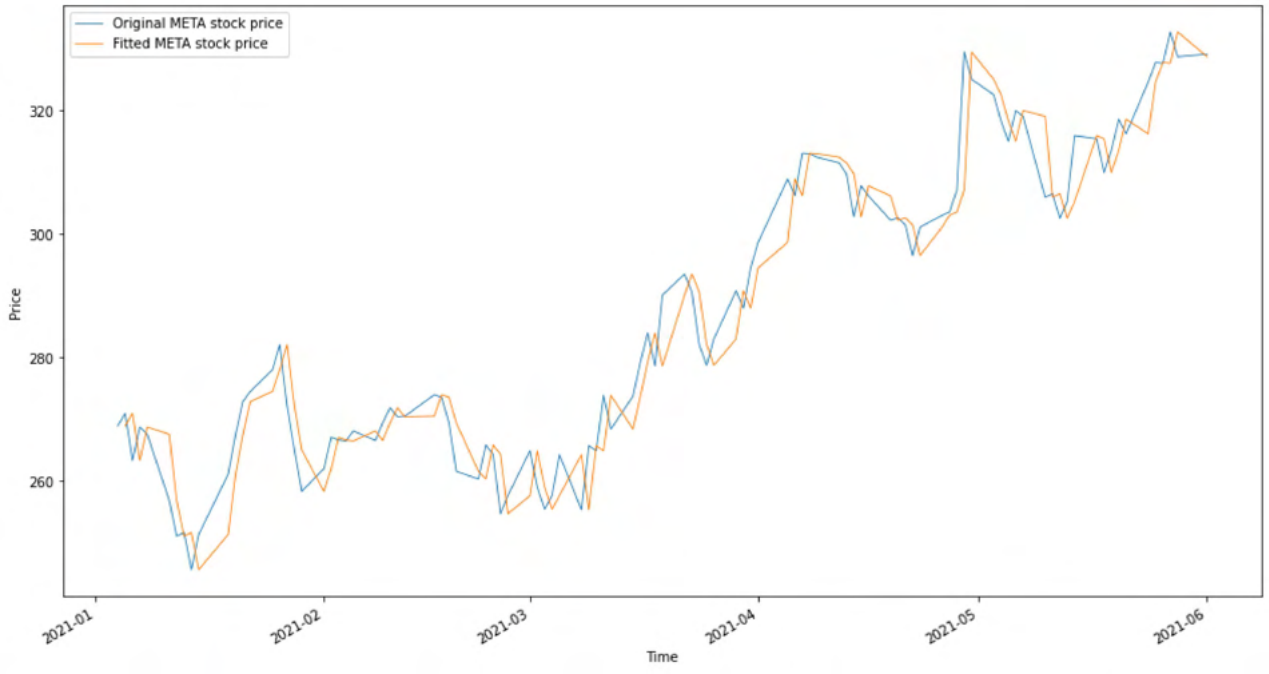

The figure above shows that the plot from fitted META stock price values does not follow closely the plot from META stock prices and this is the non-stationary dataset. Below we run a Moving Average[ MA(1) ] again but this time with the first difference of META's stock prices.

# Moving Average [MA(1)] model with first difference of META stock prices

In [18]:
```
# MA(1) for first difference of META stock prices
meta_original = statsmodels.tsa.arima.model.ARIMA(meta_stock_prices.META, order=(0, 1,

# Plot Meta prices vs fitted Meta stock prices
#Note that we have to slice out the first element since first difference makes us to lo
meta_original_residuals = meta_original.resid[1:]
meta_original_fitted_values = (meta_stock_prices["META"] - meta_original_residuals)[1:]

meta_stock_prices["META"].plot(linewidth=0.8, label="Original META stock price")
meta_original_fitted_values.plot(linewidth=0.8, label="Fitted META stock price")
plt.xlabel("Time")
plt.ylabel("Price")
plt.legend()
plt.show();
```




The figure above shows that the curve from the plots of **fitted META stock prices** closely follow the curve from the plot of original META prices. This shows that taking the first difference of META's dataset transforms the dataset into a stationary dataset and fixes the damage.

# Conclusion about the direction

As we discussed in Project 2 under "Non-Stationarity", for us to remove the unit root from a dataset we take the first difference of the dataset. This method is called a **differencing method**, and building on what we discussed above about the *types of stationarity*, our original non-stationary time series is called **difference-stationary time series**. Therefore, we can conclude that our original META's stock price dataset is a **difference-stationary time series**.

Below, under `Deployment and finding equilibrium` we continue to apply the differencing method and apply other tools to find the best parameters.

# DEPLOYMENT AND FINDING EQUILIBRIUM

In this section we are going to demonstrate two concepts: finding equilibrium and deployment of our model. The finding equilibrium component will be demonstrated by using the **Akaike's information criterion (AIC)** to select the efficient ARIMA model for our META stock prices. While the deployment component will be demonstrated by using the selected ARIMA model and forecasting future META stock movements in the next 100 days with 80% probability.

All these are grouped under the heading: *ARIMA Model for META's stock prices*

## ARIMA Model for META's Stock Price

### ARIMA Model Selection Report

We going to use Akaike's information criterion (AIC) to select the efficient ARIMA model. The AIC metric measures the goodness of fit for a model and the lower the AIC, the better the model.

The reports for AIC are shown below:

In [12]:
```
# Efficient ARIMA model Selection
model_selection = auto_arima(
    np.log(meta_stock_prices).dropna(),  # stepwise=False,
    start_p=0,
    start_d=0,
    start_q=0,
    max_p=3,
    max_d=3,
    max_q=3,
    trace=True,
    with_intercept=False,
    return_valid_fits=True,
)
```

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=-505.723, Time=0.17 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=-503.724, Time=0.04 sec
 ARIMA(0,1,1)(0,0,0)[0]             : AIC=-503.724, Time=0.10 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=-501.739, Time=0.10 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=-504.720, Time=0.06 sec

Best model:  ARIMA(0,1,0)(0,0,0)[0]
Total fit time: 0.658 seconds
```

From the above results,ARIMA(0,1,0) is the best model since ut has the lowest AIC.

### ARIMA(0,1,0) Model Estimation

In [13]:

```
# Best ARIMA Model for META stock price
arima_model = ARIMA(
    np.log(meta_stock_prices), order=(0, 1, 0), trend="n"
).fit()  # This is the best model in Python implementation
print(arima_model.summary())
```

```
                               SARIMAX Results
==============================================================================
Dep. Variable:                    META   No. Observations:                  103
Model:                 ARIMA(0, 1, 0)   Log Likelihood                 253.862
Date:                Tue, 04 Oct 2022   AIC                           -505.723
Time:                        07:57:09   BIC                           -503.098
Sample:                             0   HQIC                          -504.661
                                  - 103
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
sigma2         0.0004   5.14e-05      7.844      0.000       0.000       0.001
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):                 1.04
Prob(Q):                              0.95   Prob(JB):                         0.59
Heteroskedasticity (H):               0.99   Skew:                             0.18
Prob(H) (two-sided):                  0.97   Kurtosis:                         3.35
===================================================================================

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

# Model forecast

Let's now forecast META's stock movement in the next 100 days with 80% probability

In [19]:

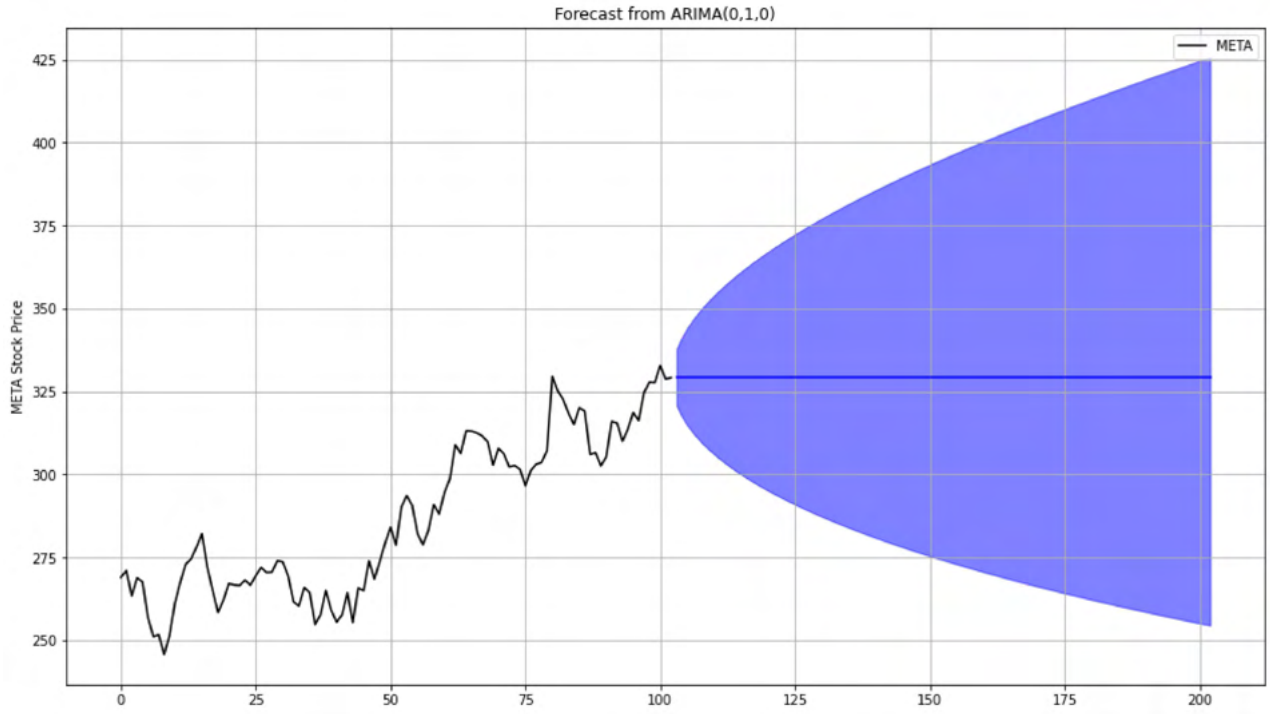
```
# Forecast Plot of ARIMA(0,1,0) with 95% Confidence Interval

# Plot META data
ffx = meta_stock_prices.copy()
ffx.index = [i for i in range(len(ffx))]  # Set numeric index
ffx.plot(ylabel="META Stock Price", title="Forecast from ARIMA(0,1,0)", color="k")

# get forecast data for next 100 steps
forecast = arima_model.get_forecast(steps=100)
forecast_mean = np.exp(forecast.predicted_mean)  # mean of forecast data
conf_int80 = forecast.conf_int(alpha=0.2)  # 80% confidence interval

# plot mean forecast and 80% confidence intervals
plt.plot(forecast_mean, c="b")

plt.fill_between(
    conf_int80.index,
    np.exp(conf_int80["lower META"]),
    np.exp(conf_int80["upper META"]),
    color="b",
    alpha=0.5,
)
plt.grid()
plt.show()
```




We can observe from the diagram above that there is an 80% chance that META's stock price will move within a range of about $257 and $425 for the next 100 days.

# CONCLUSION

Non-stationarity is a common issue is time-series forecasting. There are two ways of fixing non-stationarity depending on the type of stationarity: we can detrend the original time series or take a first difference of the time series. As a standard, it's always advised to first test for stationarity before building your model. And in a case a non-stationarity is detected, a transformation is recommended.

# References

Adriko, Joel Debo, et al. "Project 2| Group 361." World Quant University | Group Project, 18 Sept. 2022.

Dickey, David A., and Wayne A. Fuller. "Distribution of the Estimators for Autoregressive Time Series with a Unit Root." Journal of the American Statistical Association, vol. 74, no. 366a, 1979, pp. 427–431., https://doi.org/10.1080/01621459.1979.10482531.

Herranz, Edward. "Unit Root Tests." Wiley Interdisciplinary Reviews: Computational Statistics, vol. 9, no. 3, 2017, https://doi.org/10.1002/wics.1396.

Iordanova, Tzveta. "An Introduction to Non-Stationary Processes." Investopedia, Investopedia, 8 Feb. 2022, https://www.investopedia.com/articles/trading/07/stationary.asp.

Singh, Aishwarya. "An Introduction to Non Stationary Time Series in Python." Analytics Vidhya, 24 May 2020, https://www.analyticsvidhya.com/blog/2018/09/non-stationary-time-series-python/.

"Augmented Dickey–Fuller Test." Glossary of Economics Research, https://web.archive.org/web/20090302082540/http://econterms.com/glossary.cgi?action=%2B%2BSearch%2B%2B&query=augmented%2Bdickey-fuller.