Ethan Hunt

Dr. Nord

March 3, 2025

CS-100

Projecting Real GDP Using ARIMAX Modelling in Python

Introduction

Professional economists across the world are paid billions to create economic projections, but it is rarely discussed how exactly this is accomplished. While those pushing the frontier may choose a more complex model, the standard model that many practitioners use (that an undergrad would be able to learn) is the ARIMAX model. Using python, we can create projections for real GDP using an ARIMAX model for a given country.

Mathematical Overview of the ARIMA Model

A simple linear regression is a statistical technique used to estimate the relationship between a set of parameters. In layman's terms, we want to figure out the relationship between a set of variables. For a typical univariate regression (with only one parameter), it is mathematically specified as follows:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

where x is the independent variable and y is the dependent variable. The β_1 terms measure the strength of the relationship between x and y, i is used to denote the set of observations we are using, and ϵ is used to account for factors that may influence x but aren't included in the equation. Many often visualize a regression line as a "line of best fit", and, in this way, we can think of β_0 as the constant term (b) in typical slope-intercept (y=mx+b) equation. Additionally, as hinted before when discussing ϵ , we can expand this equation to include multiple parameters to measure the strength of the relationship between two variables.

There are two other classes of models that are important to know: autoregressive (AR) and moving average (MA) models.

An autoregressive model estimates the relationship between the current value of the observed measurement and the past values (Hyndman & Anthanasopoulos, 2018). It can be specified as such:

$$y_i = c + \emptyset y_{t-1} + \emptyset y_{t-2} + \cdots + \emptyset y_{t-p} + \epsilon_t$$

This is like the previous regression equation, except we are using data over time and we measuring relationships with the past value of the same variable. The subscript "t-x" represents how many times increments the backwards a given variable is. For example, if we are trying to predict inflation for 2024, we would predict 2024's inflation with that of 2023's inflation (and so on). The p represents the total number of observations we are using to predict the future observations.

A moving average (MA) model predicts future observation based on the error terms of past sets of observations (Hyndman & Anthanasopoulos, 2018). It is specified such that:

$$y_i = c + \epsilon_t + \theta\epsilon_{t-1} + \theta\epsilon_{t-2} + \cdots + \theta\epsilon_{t-p}$$

In layman's terms, we can think of this as, "how well do unexplained factors in the past predict the current variable?" For example, if we wanted to predict inflation but didn't consider a significant variable such as unemployment, this equation would use unemployment (and other factors in the past) to predict future inflation.

Using these concepts, we can construct an ARIMAX model. This is an acronym for autoregressive integrated moving average model with exogenous variables, and it is fundamentally combining the three previous equations. It can be represented mathematically like:

$$y_i = c + \sum_{i=1}^{p} \emptyset_i y_{t-i} + \sum_{i=1}^{q} \theta_j \epsilon_{t-j} + \sum_{k=1}^{r} \beta_k X_{t,k} + \epsilon_t$$

where $\emptyset$ are the parameters for the AR component, θ represents the parameters for MA component, and β represents the exogenous variables for time t (Majika, 2024). This always predicts a future value based on past values**. We are trying to predict a given value based on other variables in the past, values of the variable of interest in the past, and unexplained variables that we didn't include in our model that come from the past.** In our case, we are using past values of factors related to real GDP (inflation, imports, exports, government revenue, etc), past values of GDP, and the past values of factors not included in the equation (ex: consumer sentiment) to predict GDP in a given year.

Specifying an ARIMA Model

There are a few other important concepts that you need to know. **Stationarity is the idea that our data "has properties do not depend on the time at which the series is observed**" (Hyndman & Anthanasopoulos, 2018). By making our data stationary, we can eliminate the effect of past trends on current data, and this is a requirement to use an ARIMA model. To achieve stationarity, we need to "difference" the data, and we need to continue adding differences until we achieve stationarity (Hyndman

& Anthanasopoulos, 2018). **Differencing is effectively just subtracting the current observation from the previous observation just so we can see the change in the value of the data.** To find out if our data is stationary, we can use an augmented-Dickey-Fuller test (*ARIMA for Time Series Forecasting*, 2025). If the p-value is less than 10%, it means the data is not stationary and needs to be differenced again.

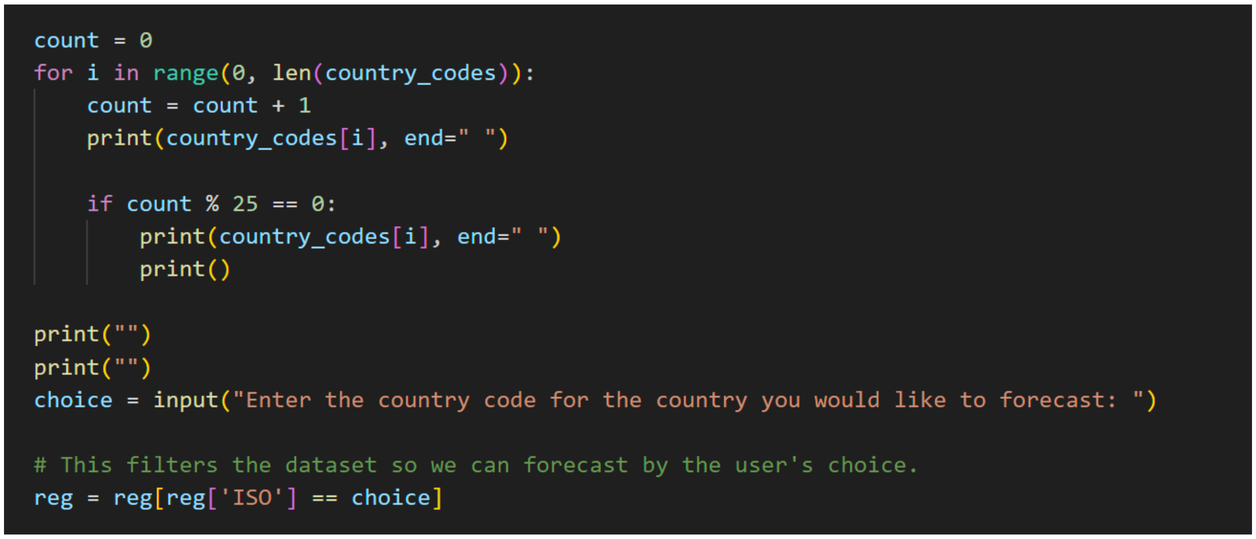
When forecasting the ARIMA model, we also need to specify other terms in the equations to factor in how much of the AR and MA components best predict future values. In other words, we are trying to find out, "how many years in the past best predict future GDP and how many years of unexplained values best predict future GDP?". We can do this using ACF and PACF plots. This portion is a bit complex, but what you need to know is this: where each graph cuts off is the number of terms you need to add into the model, with the PACF plot corresponding to the AR terms and the ACF plot corresponding to the MA terms. While I allowed user functionality to adjust these terms (just in case the plots were weird for a specific country), **I found that an AR term of 2 and MA term of 0 worked well for predicting (real) GDP.**

Using these ideas, we can specify our ARIMA model in terms of (p,d,q). P is the number of AR terms, d is the level of differencing, and q amount of MA terms (Verma, 2022). Functionally, this adjusts the ARIMA equation about to include how many AR and MA component we include and the level of differencing we desire.

Explaining the Codebase

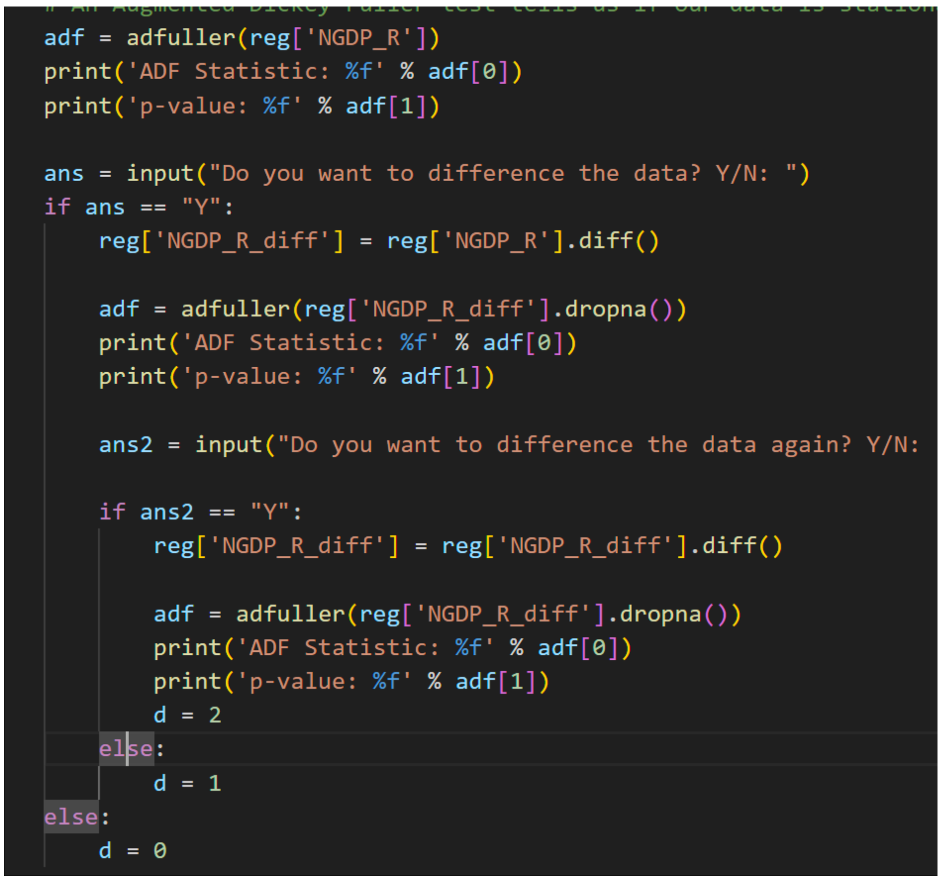
To write this code, I opted for a tutorial by DataCamp, and I used ChatGPT to debug code – but not to write anything fundamental (*ARIMA for Time Series Forecasting*, 2025; Open A.I., 2025).

I opted to use the IMF Economic Outlook dataset for this project, but I had to do some significant shaping with the pandas library to prepare it to function. Since this isn't a data science course, I'll exclude all these details to be viewed in my codebase (with comments), and I became more familiar with data manipulation with pandas from a few online sources (Bobbitt, 2021, 2021; *Pandas Tutorial*, n.d.; *Python Pandas Pivot_table*). Afterwards, I allow the user to select a country from a list of available countries. I loop through the list and print out the country, and I just to a new line every 25 countries. I then filter the dataset so we only forecast the selected country.



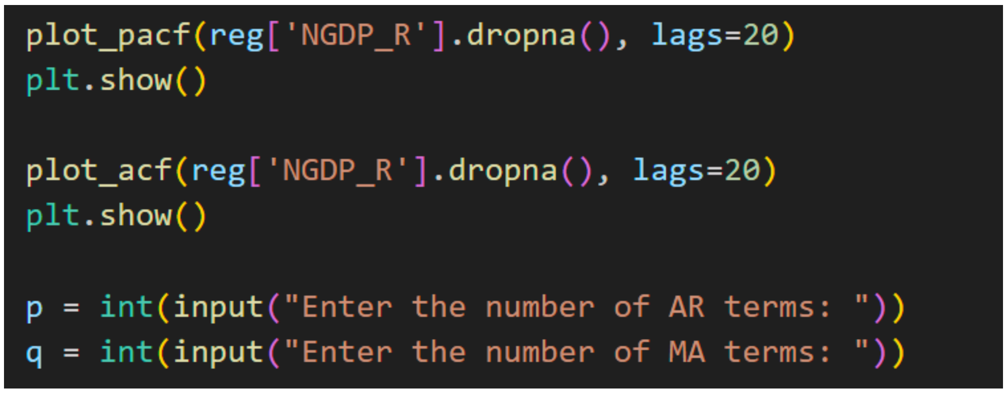
```python
count = 0
for i in range(0, len(country_codes)):
    count = count + 1
    print(country_codes[i], end=" ")

    if count % 25 == 0:
        print(country_codes[i], end=" ")
        print()

print("")
print("")
choice = input("Enter the country code for the country you would like to forecast: ")

# This filters the dataset so we can forecast by the user's choice.
reg = reg[reg['ISO'] == choice]
```

We then do an ADF test to determine the level of differencing. The user then picks the level they desire based on test results. The code was based on work by Datacamp (*ARIMA for Time Series Forecasting*, 2025).



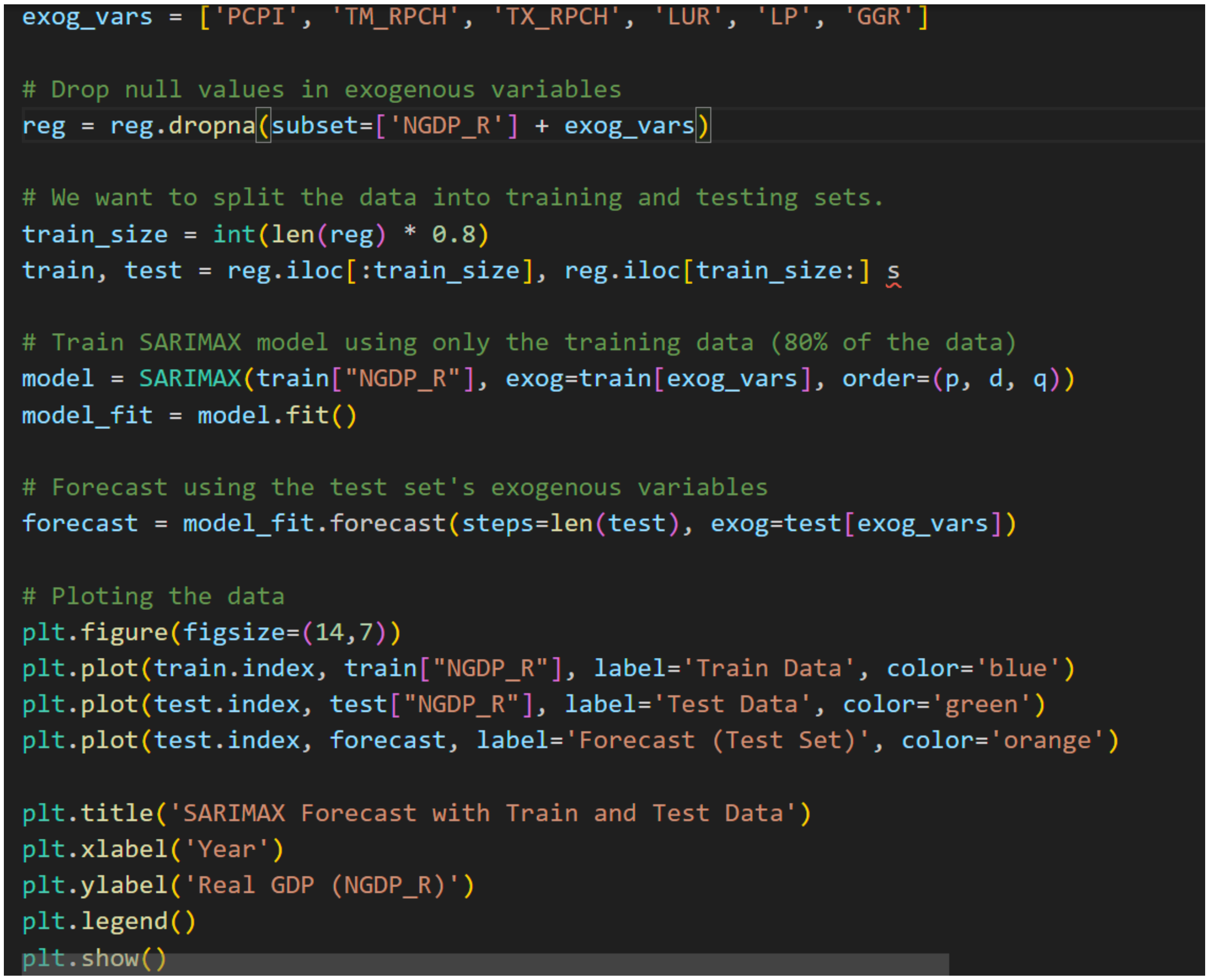
```python
# An Augmented Dickey-Fuller test tells us if our data is station
adf = adfuller(reg['NGDP_R'])
print('ADF Statistic: %f' % adf[0])
print('p-value: %f' % adf[1])

ans = input("Do you want to difference the data? Y/N: ")
if ans == "Y":
    reg['NGDP_R_diff'] = reg['NGDP_R'].diff()

    adf = adfuller(reg['NGDP_R_diff'].dropna())
    print('ADF Statistic: %f' % adf[0])
    print('p-value: %f' % adf[1])

    ans2 = input("Do you want to difference the data again? Y/N:

    if ans2 == "Y":
        reg['NGDP_R_diff'] = reg['NGDP_R_diff'].diff()

        adf = adfuller(reg['NGDP_R_diff'].dropna())
        print('ADF Statistic: %f' % adf[0])
        print('p-value: %f' % adf[1])
        d = 2
    else:
        d = 1
else:
    d = 0
```

The user can then interpret the PACF and ACF plots to determine the p and q terms for the ARIMA (*ARIMA for Time Series Forecasting*, 2025).



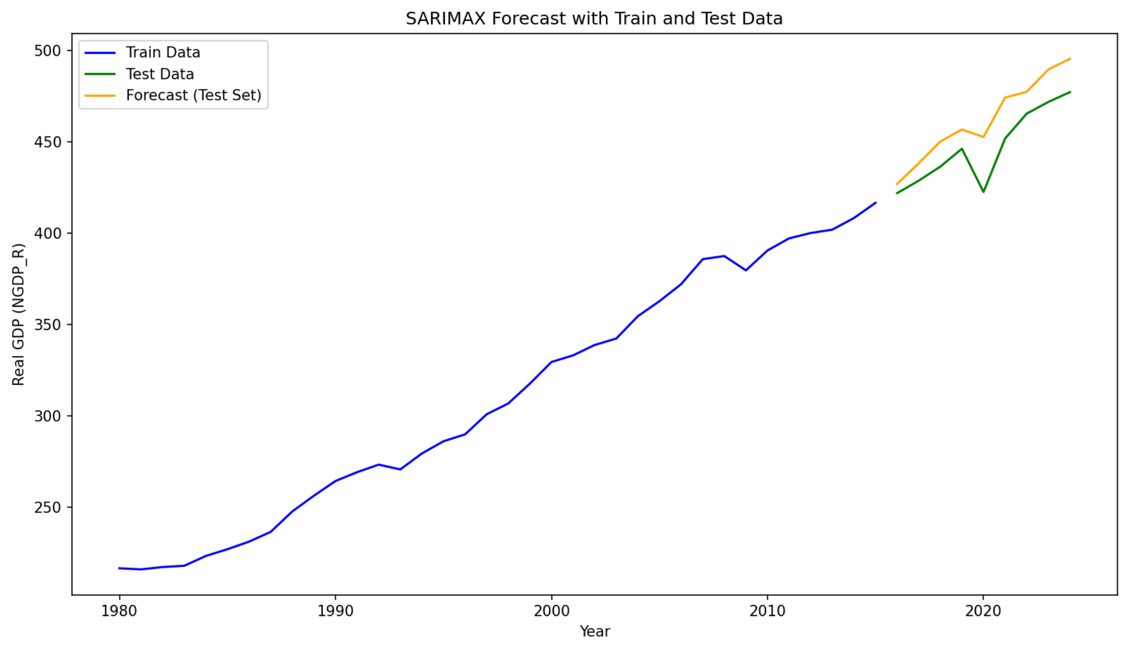
```python
plot_pacf(reg['NGDP_R'].dropna(), lags=20)
plt.show()

plot_acf(reg['NGDP_R'].dropna(), lags=20)
plt.show()

p = int(input("Enter the number of AR terms: "))
q = int(input("Enter the number of MA terms: "))
```

We then specify the names of the exogenous variables in a list. These variables are other factors that influence real GDP and we used in a paper by Xie et al. (2024). We drop the null values (or else we get an error, and then create a train and test dataset to see performance. We then create the ARIMAX model using our p,d, and q values, and then we graph this along historical, train, and test set values.



```
exog_vars = ['PCPI', 'TM_RPCH', 'TX_RPCH', 'LUR', 'LP', 'GGR']

# Drop null values in exogenous variables
reg = reg.dropna(subset=['NGDP_R'] + exog_vars)

# We want to split the data into training and testing sets.
train_size = int(len(reg) * 0.8)
train, test = reg.iloc[:train_size], reg.iloc[train_size:] s

# Train SARIMAX model using only the training data (80% of the data)
model = SARIMAX(train["NGDP_R"], exog=train[exog_vars], order=(p, d, q))
model_fit = model.fit()

# Forecast using the test set's exogenous variables
forecast = model_fit.forecast(steps=len(test), exog=test[exog_vars])

# Ploting the data
plt.figure(figsize=(14,7))
plt.plot(train.index, train["NGDP_R"], label='Train Data', color='blue')
plt.plot(test.index, test["NGDP_R"], label='Test Data', color='green')
plt.plot(test.index, forecast, label='Forecast (Test Set)', color='orange')

plt.title('SARIMAX Forecast with Train and Test Data')
plt.xlabel('Year')
plt.ylabel('Real GDP (NGDP_R)')
plt.legend()
plt.show()
```

As a result of all this hard work, we get a graph like this. This is the predicted forecasted values for Belizean real GDP for the years 2016 to 2024 (train) against the actual values for 2016-2024 real GDP (test). **The ARIMA model uses the observations of each previous year to predict the observation for the subsequent year.**



Bibliography

ARIMA for Time Series Forecasting: A Complete Guide. (2025). Datacamp. https://www.datacamp.com/tutorial/arima

Bobbitt, Z. (2021, November 1). *Pandas: How to Reshape DataFrame from Wide to Long*. Statology. https://www.statology.org/pandas-wide-to-long/

Hyndman, R., & Anthanasopoulos, G. (2018). *Forecasting: Principles and Practice (2nd ed)* (2nd ed.). OTexts. https://otexts.com/fpp2/

Majika, M. (2024). *ARIMAX: Time Series Forecasting with External Variables*. Linkedin. https://www.linkedin.com/pulse/arimax-time-series-forecasting-external-variables-marcin-majka-64yxf/

OpenAI. (2025). *ChatGPT* (Mar 12 version) [Large language model]. https://chat.openai.com/chat

Pandas Tutorial. (n.d.). Retrieved March 12, 2025, from https://www.w3schools.com/python/pandas/default.asp

Python Pandas.pivot_table().GeeksforGeeks. https://www.geeksforgeeks.org/python-pandas-pivot_table/

Verma, Y. (2022, May 24). *Quick way to find p, d and q values for ARIMA*. Analytics India Magazine. https://analyticsindiamag.com/ai-trends/quick-way-to-find-p-d-and-q-values-for-arima/

Xie, H., Xu, X., Yan, F., Qian, X., & Yang, Y. (2024). *Deep Learning for Multi-Country GDP Prediction: A Study of Model Performance and Data Impact* (No. arXiv:2409.02551; Version 1). arXiv. https://doi.org/10.48550/arXiv.2409.02551