Mini-Project 2: Interpolation and Curve Fitting for Linear and Nonlinear Data

Brady Sherry

October 6, 2024

Abstract

In data analysis and modeling, two common techniques/methods used are interpolation and curve fitting. Interpolation involves predicting unknown values within a discrete data set, usually by creating a continuous function that passes through all of the known data points. Similarly, curve fitting attempts to fit a function as closely as possible to given data. The goal is to minimize the difference between the known data points and this predictive model, and this model can then be used to predict data outside of the discrete data set, which is a method known as extrapolation. It is often the case that these models are either linear or can be converted into a linear form, and thus we can solve interpolation or curve fitting problems by utilizing matrix equations. When we cannot "linearize" the data, though, we can use pre-built functions provided in Python to optimize the parameters of a function that we feel would be a good fit for the data.

1 Introduction

In this project, we will be examining several data sets of country populations. Using a world population database provided by data.worldbank.org, I was able to get the population data for every country around the world from 1960 to 2023. After searching through this data, I found some countries that seemed to share similar data trends. First off, Guyana, Germany, and Palau all had wavering population patterns that shifted back and forth from increasing and decreasing intervals. Then, Madagascar, Ethiopia, and Angola all showed similar exponential growth, a pattern characterized by a continuously increasing rate of change over time. Lastly, Nauru had a more obscure data pattern, which I would roughly describe as shifting back and forth between rapidly increasing and slowly increasing intervals. With the use of a Jupyter notebook in Python, I was able to filter the data for these specific countries into a condensed data frame, and Figure 1 shows the first and last 5 rows of this.

	Year	Guyana	Germany	Palau	Madagascar	Ethiopia	Angola	Nauru
0	1960	571990.0	72814900.0	9446.0	5073342.0	21739710.0	5357195.0	4582.0
1	1961	588597.0	73377632.0	9639.0	5206239.0	22281675.0	5441333.0	4753.0
2	1962	604833.0	74025784.0	9851.0	5343117.0	22852158.0	5521400.0	4950.0
3	1963	620703.0	74714353.0	10076.0	5484252.0	23448979.0	5599827.0	5198.0
4	1964	635957.0	75318337.0	10318.0	5630024.0	24073696.0	5673199.0	5484.0
...	...	...	...	...	...	...	...	...
59	2019	798753.0	83092962.0	17916.0	27533134.0	114120594.0	32353588.0	12132.0
60	2020	797202.0	83160871.0	17972.0	28225177.0	117190911.0	33428486.0	12315.0
61	2021	804567.0	83196078.0	18024.0	28915653.0	120283026.0	34503774.0	12511.0
62	2022	808726.0	83797985.0	18055.0	29611714.0	123379924.0	35588987.0	12668.0
63	2023	813834.0	84482267.0	18058.0	30325732.0	126527060.0	36684202.0	12780.0

Figure 1: Population data for the selected countries

2 Problem Statement

The task at hand is to fit these different types of data patterns using interpolation, "linear" curve fitting, and nonlinear curve fitting. For Guyana, Germany, and Palau, we need to select certain points where the data shifts from increasing to decreasing rates, and then use those points to perform a polynomial interpolation. Each country has a similar amount of shifts, so we will select 6 data points from each, which will result in degree 5 polynomials that can be solved using matrix equations. For Madagascar, Ethiopia, and Angola, we will manipulate a general exponential function into

the form of a linear matrix, which will then allow us to optimize the parameters of this exponential function to fit curves for each country. In the case of Nauru, however, we will not be able to manipulate a function into a linear or matrix form for the curve fitting. Instead, we need to find a function that matches the characteristics of Nauru's population pattern and then optimize that specific function's parameters to create the curve. Figure 2 shows the plots of the countries that we will be interpolating, and Figure 3 includes the plots for the curve fitting problems.

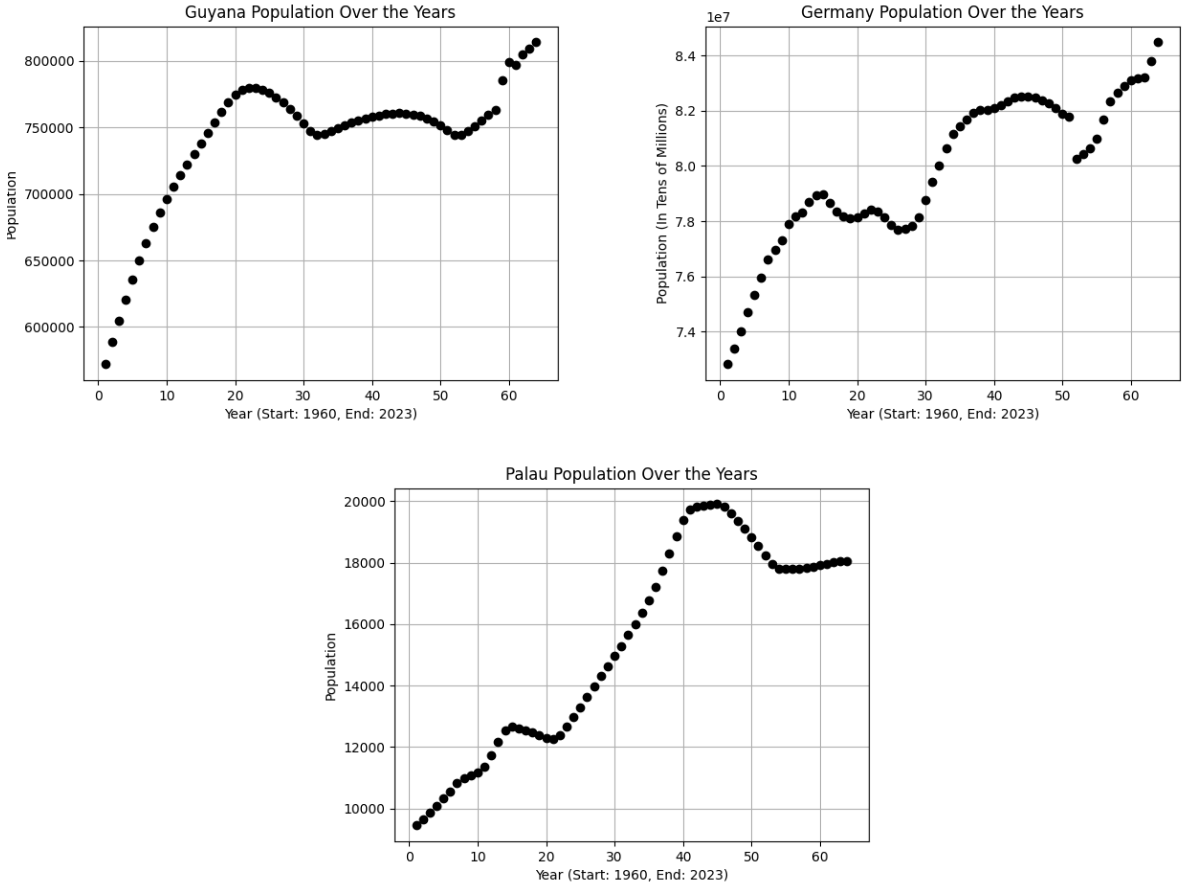

Figure 2: Data plots for Guyana, Germany, and Palau

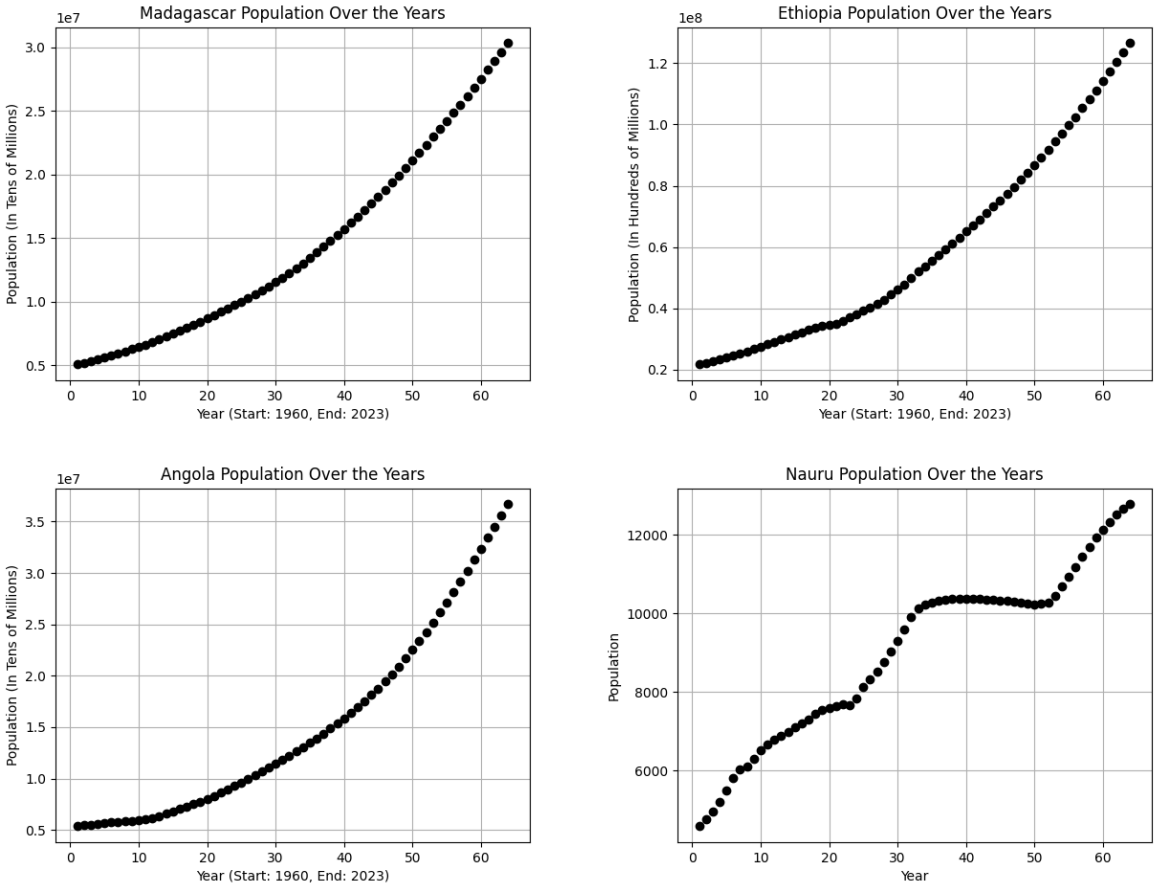

Figure 3: Data plots for Madagascar, Ethiopia, Angola, and Nauru

3 Methodology

We will begin with the interpolation problem. Again, we will be interpolating Guyana, Germany, and Palau as degree 5 polynomials, so I chose 6 data points for each country (see Table 1). For all 3, I chose the first and last point (i.e the years 1960 and 2023), and then selected 4 "turning points" in between, where the data switches between increasing and decreasing intervals. Notice that for the y-values (population counts), the second data points are greater than the first, the third data points are less than the second, and then they continue to alternate back and forth.

Data Point	Guyana	Germany	Palau
1	(1960, 571,990)	(1960, 72,814,900)	(1960, 9,446)
2	(1981, 779,686)	(1974, 78,967,433)	(1974, 12,662)
3	(1991, 744,096)	(1985, 77,684,873)	(1980, 12,252)
4	(2003, 760,562)	(2003, 82,534,176)	(2004, 19,907)
5	(2012, 743,966)	(2011, 80,274,983)	(2013, 17,805)
6	(2023, 813,834)	(2023, 84,482,267)	(2023, 18,058)

Table 1: Data points selected for interpolation

In order to interpolate these points, we need to normalize the data, so we redefine the year values as an interval of natural numbers from 1 to 64, with the year 1960 corresponding to 1, and the year 2023 corresponding to 64. Then, we are able to set up the polynomial equations in this general form:

$$c_0 + c_1x + c_2x^2 + c_3x^3 + c_4x^4 + c_5x^5 = y$$

Where x is the year, y is the population, and c_0, c_1, c_2, c_3, c_5 are the unknown coefficients. To solve for these coefficients, we set up matrix equations in the form $Ac = y$ for each country, where A is a Vandermonde matrix of the country's x (year) values. For example, here is the matrix equation for Guyana:

$$\begin{bmatrix} 1^0 & 1^1 & 1^2 & 1^3 & 1^4 & 1^5 \\ 22^0 & 22^1 & 22^2 & 22^3 & 22^4 & 22^5 \\ 32^0 & 32^1 & 32^2 & 32^3 & 32^4 & 32^5 \\ 44^0 & 44^1 & 44^2 & 44^3 & 44^4 & 44^5 \\ 53^0 & 53^1 & 53^2 & 53^3 & 53^4 & 53^5 \\ 64^0 & 64^1 & 64^2 & 64^3 & 64^4 & 64^5 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix} = \begin{bmatrix} 571,990 \\ 779,686 \\ 744,096 \\ 760,562 \\ 743,966 \\ 813,834 \end{bmatrix}$$

Once we define these Vandermonde matrices for each country, we can easily solve for the coefficients, c, by utilizing a built-in Python function:

c = numpy.linalg.solve(A,y)

These coefficients can then be plugged in to another Python function that will create the interpolation curve, p:

p = numpy.polyval(numpy.flip(c), x)

The method for "linear" curve fitting is very similar. For this, we will use the exponential data trends of Madagascar, Ethiopia, and Angola, so the first step is to manipulate a general exponential function into a linear form:

$$\begin{aligned} ae^{bx} &= y \\ \implies &\ln(ae^{bx}) = \ln(y) \\ \implies &\ln(a) + \ln(e^{bx}) = \ln(y) \\ \implies &\ln(a) + bx = \ln(y) \end{aligned}$$

Then, we are able to set up our "Vandermonde" matrices for each country with some alterations. In general, we set up our matrix equations like this:

$$\begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ ... & ... \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \ln(a) \\ b \end{bmatrix} = \begin{bmatrix} \ln(y_1) \\ \ln(y_2) \\ \ln(y_3) \\ ... \\ \ln(y_n) \end{bmatrix}$$

Where $x_1, x_2, x_3, ..., x_n$ are the years, and $\ln(y_1), \ln(y_2), \ln(y_3), ..., \ln(y_n)$ are the natural logs of the populations. However, we must define what is known as the "normal equation" before we can solve for the coefficients, a and b. To do this, we can use Python to matrix multiply A^T by A and A^T by $\ln(y)$:

A_norm = A.T @ A
y_norm = A.T @ numpy.log(y)

Then, we use the same Python function as before to solve for the coefficients:

c = numpy.linalg.solve(A_norm,y_norm)

Note that c is an array of length 2, where c[0] = $\ln(a)$ and c[1] = b. Thus, we have found the coefficient for b, but we still must solve for a:

$$\ln(a) = \text{c}[0]$$
$$\implies \text{e}^{\ln(a)} = e^{c[0]}$$
$$\implies \text{a} = \text{e}^{c[0]}$$

We repeat this entire process to find the coefficients, a and b, for each country, and then we plug these coefficients back into our original function to fit the curves:

$$ae^{bx} = y$$

Lastly, consider our nonlinear curve fitting problem with the Nauru data set, which we plotted in Figure 3. The trick here is to choose a function that we feel would match the data's behavior. By examining the plot, we see that certain intervals behave linearly, but other portions follow wavelike patterns similar to that of sin and cos functions. So, a function that we suspect would fit well is defined as follows:

$$f(x) = a + bx + c\sin^2(dx) + e\cos(fx)$$

We now need to find the correct coefficients (a, b, c, d, e, f) of this function in order to fit a curve. After defining our function, f, in Python, we can use another built-in function to optimize its coefficients/parameters:

c_fit = sp.optimize.curve_fit(f, x, y)

Note that x is the year, and y is Nauru's population. All we need to do from here is take the coefficients from this c_fit array and plug them back into our function to fit the curve.

4 Results

After computing the coefficients, we are able to plot our interpolations and curves to see how well they fit the data sets. We will begin with the interpolations for Guyana, Germany, and Palau (See Figure 4 below):

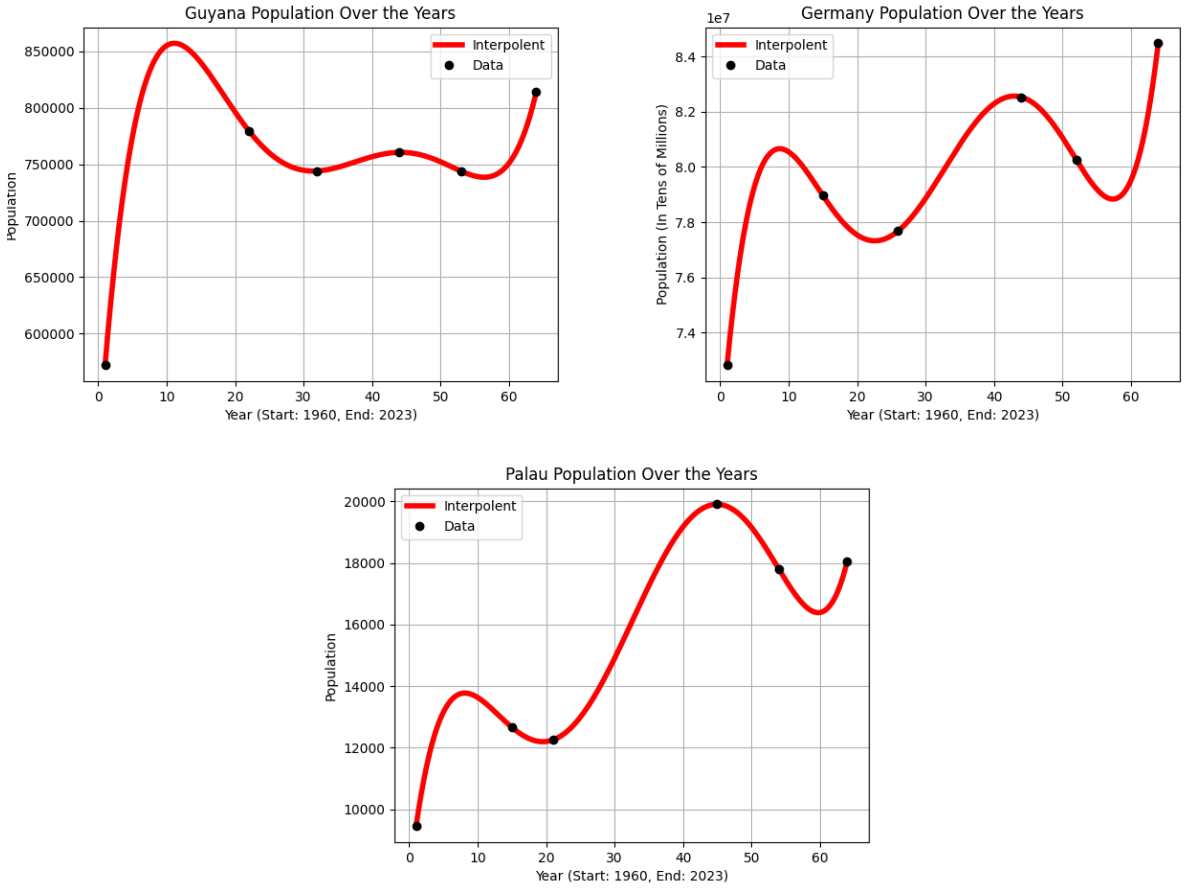

Figure 4: Interpolations for Guyana, Germany, and Palau

These interpolations form smooth curves that loop up and down through each of our selected points, showcasing the wavelike behavior within these data sets. Then, for the curve fitting, we are able to go one step further and use the curves that we computed to make future predictions outside of our data set (extrapolation). For Madagascar, Ethiopia, and Angola, we expect the data to continue to follow the same exponential trend, and for Nauru, we expect a similar linear yet wavelike pattern to continue (see Figure 5 below).

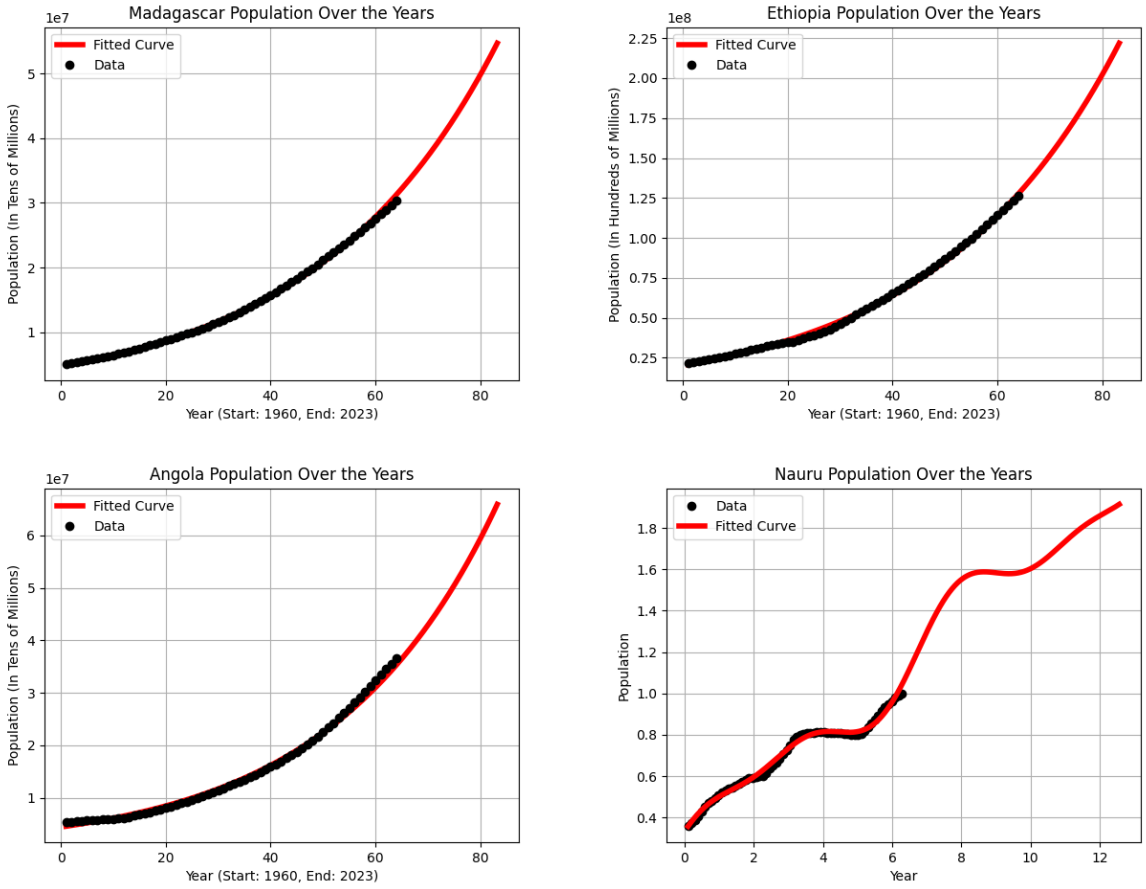

Figure 5: Curve Fits for Madagascar, Ethiopia, Angola, and Nauru

5 Discussion

When creating these models, it is important to test them and see how accurate they are in relation to the data. For interpolation, our model goes directly through each provided data point, so we know that the model will be 100% accurate for those specific points. However, since we didn't factor in any other data when creating this model, the fit will not be perfect for the rest of the data set. In fact, if we plot our interpolation against the original data sets for Guyana, Germany, and Palau, we will see that the model struggles to fit a lot of the data accurately (See Figure 6 below).

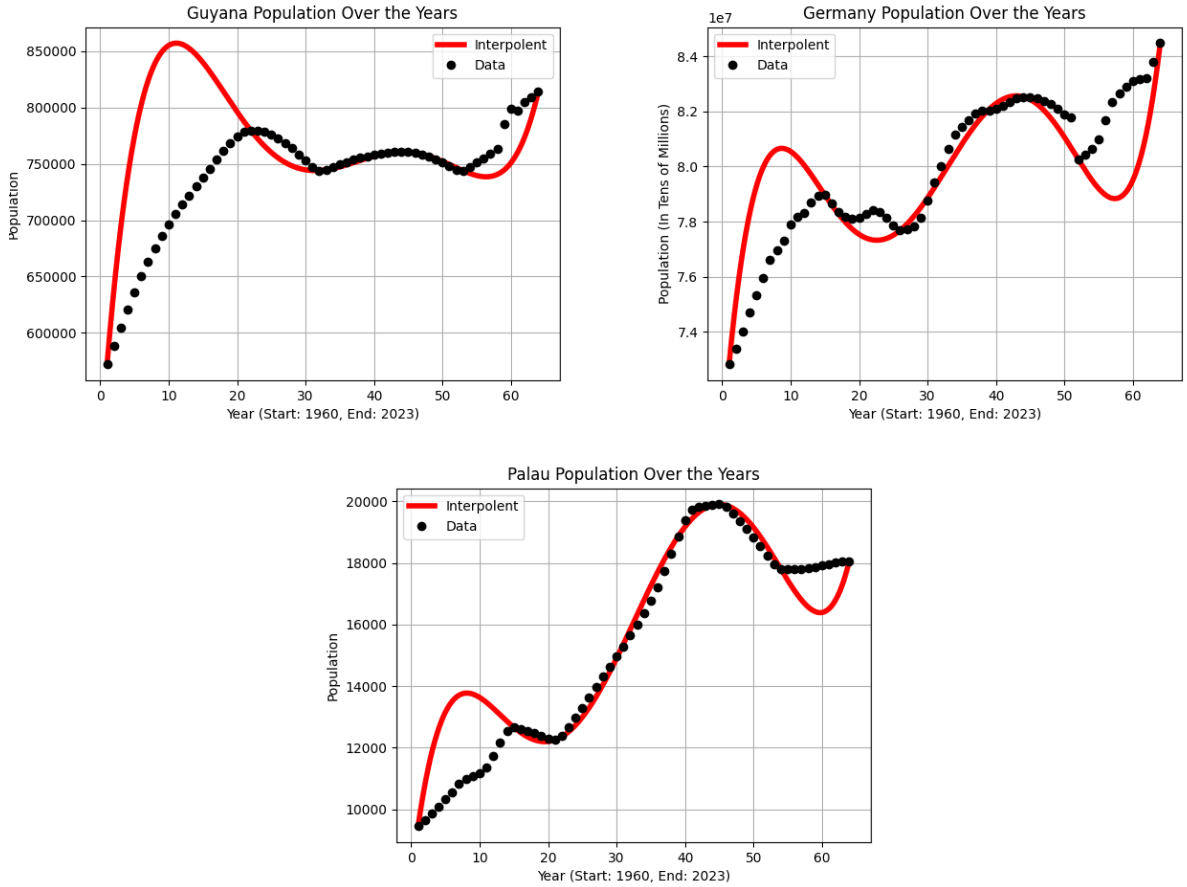

Figure 6: Interpolations vs. Original Data Sets

By adding more points to our interpolation, we may have been able to make these models more accurate, but we do see some areas that are fit relatively well, such as the years 20-50 for Palau. In our curve fits, though (see Figure 5), we optimized our coefficients for the entire data sets, rather than just a few select points. So, we should find that most of the data corresponds very well to our curves, and we can check just how accurate the curves are by calculating R^2 values for each country. The formula for R^2 is as follows:

$$\mathrm{SS}_{\mathrm{res}} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$
$$\mathrm{SS}_{\mathrm{tot}} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$
$$R^2 = 1 - \frac{\mathrm{SS}_{\mathrm{res}}}{\mathrm{SS}_{\mathrm{tot}}}$$

Where y_i is the actual value, $\hat{y}_i$ is the predicted value, and $\bar{y}$ is the mean of the actual values. In simple terms, R^2 tells you how well the model's predictions match the actual data, on a scale from 0 to 1. We can use Python to calculate and compare this R^2 value for Madagascar, Ethiopia, and Angola, the countries that we fit the same general exponential function to. (See Figure 7)

```python
#calculate predicted values
predicted_mad = a_mad * np.exp(b_mad * years)
predicted_eth = a_eth * np.exp(b_eth * years)
predicted_ang = a_ang * np.exp(b_ang * years)

#calculate R^2 for each country
def calculate_r_squared(y_actual, y_predicted):
    ss_total = np.sum((y_actual - np.mean(y_actual))**2)
    ss_residual = np.sum((y_actual - y_predicted)**2)
    r_squared = 1 - (ss_residual / ss_total)
    return r_squared

r_squared_mad = calculate_r_squared(mad, predicted_mad)
r_squared_eth = calculate_r_squared(eth, predicted_eth)
r_squared_ang = calculate_r_squared(ang, predicted_ang)

#print R^2 values
print(f"R^2 for Madagascar: {r_squared_mad:.4f}")
print(f"R^2 for Ethiopia: {r_squared_eth:.4f}")
print(f"R^2 for Angola: {r_squared_ang:.4f}")
```

Figure 7: R^2 calculations for Madagascar, Ethiopia, and Angola

After running this code, we find the R^2 values of 0.9991, 0.9984, and 0.9959 for Madagascar, Ethiopia, and Angola, respectively. We see that Madagascar has the highest value by a slight margin, but they are all very close to 1, which means that our curves fit each of the data sets extremely well. Repeating the same process for our nonlinear curve fit with Nauru, we find an R^2 value of 0.9924, which is once again very close to 1. Usually this high of accuracy means that our model will be great at predicting future values as well, but I am a lot more skeptical about the Nauru model since it doesn't follow as consistent of a pattern as the exponential models do.

6 Conclusion

In conclusion, interpolation and curve fitting are both very useful methods when attempting to model both linear and nonlinear data. Interpolation is particularly useful if we want to create a smooth curve that directly accounts for all of our known data, although this becomes increasingly difficult when dealing with larger, more complex data sets. Alternatively, curve fitting allows us to optimize the coefficients of a wide variety of functions and model any kind of data patterns that we want, including linear, exponential, trigonometric, and more. If I were to expand on this project, I would attempt to fit curves for Guyana, Germany, and Paulu, or at least try to include more points for their interpolations. These data sets were very complex, but I feel that we would be able to fit them relatively accurately if we examined them further.