Climate Change Models

Lauren Fie

Arcadia University

April 24, 2020

Abstract

As a result of the changing climate, global temperatures and global mean sea levels (GMSL) have been increasing rapidly. The complex physical systems surrounding this growth make it difficult to form an accurate model. This paper looks at a simplified model proposed and supported by Aral, Guan, and Chang[1]. This model consists of a system of ordinary differential equations that are simplified and solved theoretically below, then applied using python to calculate precise values and form predictions.

1 Introduction

Over time, both the sea level the global temperature have changed. Three million years ago, they were both higher than current levels by 25 to 35 meters and 2-3°C, respectively[2]. However, just 20,000 years ago the sea level was 140 meters lower and the temperature was 4-7 °C cooler than current numbers[2]. Based on past trends and the state of the environment, the sea level is predicted to rise for centuries, posing threats to those who live near water [3]. The particular measurement for sea level that is of importance is the global mean sea level (GMSL) since it still has the greatest uncertainty surrounding it [3].The global mean sea level can be found two ways: through averaging all the sea level measurements from individual locations, or through functions that mimic sea level patterns[3]. Scientists and mathematicians have created models and estimates to predict the rises in sea level and temperature with varying levels of success. Some have tried to form these models separately or as functions of each other, but it has become evident that a system of equations is necessary to form the most accurate model[1].

One study that led to this realization was Rahmstorf 2007[2]. Rahmstorf found a proportional relationship between the rate of sea level rise and temperature and used that relationship to make predictions for future sea level rise[1]. In doing so, Rahmstorf disregarded the actual physical system and prioritized finding a simple mathematical formula. In reality, there is not a basic proportional relationship between temperature and sea level; there are many other factors that come into play and affect the way they change. As a result, Rahmstorf's predictions were much higher than those of the Intergovernmental Panel

on Climate Change (IPCC)[1]. There were a few other issues with Rahmstorf's model that contributed to the discrepancies.

First: in order to smooth the oscillations in the data, Rahmstorf used a five year averaging technique, turning the 122 data points into 24. While this may have made the data easier to work with, it sacrificed accuracy and introduced more room for error.

Second: Rahmstorf made some assumptions that further limited his model. His formula assumed that temperature would serve as a known variable and sea level rise would be proportional to it. This prevented him from considering the effects sea level has on temperature. This resulted in his model not accounting for the way changes in water vapor can affect the temperature as the sea level rises[1]. Since changes in sea leavel and temperature affect each other, a system of equations was necessary. In order to set up the necessary system of equations, an independent third variable was introduced: time.

Aral, Guan, and Chang [1] created a system of differential equations that describe the change in temperature and sea level over time in terms of each other. They solved this system of linear differential equations and found a 90% confidence interval around the predicted temperature and sea level rise. This solved the obvious problem in Rahmstorf's study by considering the way the two variables impact one another. This also solved an additional issue in Rahmstorf's study by using statistics to account for uncertainty in the data measurements. Furthermore, in order to avoid the same inaccuracy that Rahmstorf's model had, this model used two year averaging to smooth out oscillation without losing as much precision as five year averaging.They used their models to form predictions between the years 1950 and 2001 and compared their predicted values with known data. They found their sea level model to be very accurate and their temperature model to be close, though less accurate than the sea level predictions [1]. They then used their model to predict the rise of temperature and sea level in year 2100, estimating a 1.33°C increase in temperature and a 423.77mm rise in sea level. These estimates match those by the IPCC.

In this paper I will use the same data Aral, Guan, and Chang did [6][5][7] and attempt to reach and prove their results.

2 Solving the System of Differential Equations

The system of differential equations Aral, Guan, and Chang [1] discovered is given below. In these formulas, T(t) is the temperature as a function of time, H(t) is the sea level rise as a function of time, and $a_{11}, a_{12}, a_{21}, a_{22}, c_1$, and c_2 are constants.

$$\begin{cases} \frac{dT(t)}{dt} = a_{11}T(t) + a_{12}H(t) + c_1 \\ \frac{dH(t)}{dt} = a_{21}T(t) + a_{22}H(t) + c_2 \end{cases}$$

These are relatively simple first order differential equations. However, to reduce them even further into a form that is easiest to work with, we can break it down such that:

$$X(t) = (T(t), H(t))^\tau, \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \mathbf{C} = \begin{Bmatrix} c_1 \\ c_2 \end{Bmatrix}$$

For better visualization, the above system of equations can be written as:

$$\begin{bmatrix} \frac{dT(t)}{dt} \\ \frac{dH(t)}{dt} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} T(t) \\ H(t) \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

For simplification in solving the equation, we will rewrite it once more as:

$$\frac{dX}{dt} = \mathbf{A}X(t) + \mathbf{C}$$

Instead of solving the differential equation the traditional way, they further reduced the problem using basic algebra.

$$\frac{dX}{dt} \approx \frac{\Delta X}{\Delta t}$$

$$\frac{\Delta X}{\Delta t} = \mathbf{A}X(t) + \mathbf{C}$$

$$\Delta X = (\mathbf{A}X(t) + \mathbf{C}) * \Delta t$$

$$\Delta X = \mathbf{A}X(t)\Delta t + \mathbf{C}\Delta t$$

Since $\frac{\Delta X}{\Delta t}$ is the change in X over time, they decided to use a discrete system instead of a continuous one and move in one year increments. That allowed them to write the function as $X(k+1)$ so it can predict the year after the starting point.

$$\Delta X = X(k+1) - X(k)$$

Plug that into the reduced formula above, replacing X(t) with X(k), the discrete time variable:

$$X(k+1) - X(k) = \mathbf{A}X(k)\Delta t + \mathbf{C}\Delta t$$

Solve for $X(k+1)$:

$$X(k+1) = \mathbf{A}X(k)\Delta t + \mathbf{C}\Delta t + X(k)$$

Rearrange the equation to make it easier to factor out like terms:

$$X(k+1) = \mathbf{A}X(k)\Delta t + X(k) + \mathbf{C}\Delta t$$

Factor out $X(k)$:

$$X(k+1) = (\mathbf{A}\Delta t + \mathbf{I})X(k) + \mathbf{C}\Delta t$$

In order to simplify the appearance of the equation, define Φ and Ω such that:

$$\Phi = (\mathbf{A}\Delta t + \mathbf{I}) = (\mathbf{I} + \mathbf{A}\Delta t)$$

$$\Omega = \mathbf{C}\Delta t$$

So we have:

$$X(k+1) = \Phi X(k) + \Omega$$

Since we have two formulas, one for temperature and one for sea level, we can expand this new formula as:

$$x_i(k+1) = \{x_1(k), x_2(k), 1\} \begin{Bmatrix} \varphi_{i1} \\ \varphi_{i2} \\ \omega_i \end{Bmatrix}$$

for $i = 1, 2$ where $i = 1$ gives $x(k+1)$ for temperature and $i = 2$ gives $x(k+1)$ for sea level.
φ_{ij} is in row i and column j of the matrix given by Φ.
ω_i is the value in position i of the vector given by Ω.
To make this easier to work with, it can even further be expanded to:

$$\begin{bmatrix} T(k+1) \\ H(k+1) \end{bmatrix} = \begin{bmatrix} \varphi_{11} & \varphi_{12} \\ \varphi_{21} & \varphi_{22} \end{bmatrix} \begin{bmatrix} T(k) \\ H(k) \end{bmatrix} + \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}$$

This can be reduced to:

$$\begin{bmatrix} T(k+1) \\ H(k+1) \end{bmatrix} = \begin{bmatrix} \varphi_{11} & \varphi_{12} & \omega_1 \\ \varphi_{21} & \varphi_{22} & \omega_2 \end{bmatrix} \begin{bmatrix} T(k) \\ H(k) \\ 1 \end{bmatrix}$$

For years 1 through n-1 we can use the following array in our predictions for year n:

$$\Lambda = \begin{bmatrix} T(1) & H(1) & 1 \\ T(2) & H(2) & 1 \\ . & . & . \\ . & . & . \\ . & . & . \\ T(n-1) & H(n-1) & 1 \end{bmatrix}$$

Let $\mathbf{Y}_1$ hold the predicted values for temperature change and $\mathbf{Y}_2$ hold predicted values for sea level rise, stored as:

$$\mathbf{Y}_1 = \{T(2), T(3), ..., T(n)\}^\tau$$
$$\mathbf{Y}_2 = \{H(2), H(3), ..., H(n)\}^\tau$$

Let Φ_i be the values used to make the predicted values as close to the actual data as possible. So Φ_i can be defined as:

$$\Phi_i = \{\varphi_{i1}, \varphi_{i2}, \omega_i\}^\tau$$

So if Φ_i is able to get the values to be accurate, we should have:

$$\mathbf{Y}_i = \Lambda \Phi_i$$

which can be expanded using the previously defined matrices. For temperature:

$$\begin{bmatrix} T(2) \\ T(3) \\ . \\ . \\ . \\ T(n) \end{bmatrix} = \begin{bmatrix} T(1) & H(1) & 1 \\ T(2) & H(2) & 1 \\ . & . & . \\ . & . & . \\ . & . & . \\ T(n-1) & H(n-1) & 1 \end{bmatrix} \begin{bmatrix} \varphi_{11} \\ \varphi_{12} \\ \omega_1 \end{bmatrix}$$

Similarly, for sea level:

$$\begin{bmatrix} H(2) \\ H(3) \\ . \\ . \\ . \\ H(n) \end{bmatrix} = \begin{bmatrix} T(1) & H(1) & 1 \\ T(2) & H(2) & 1 \\ . & . & . \\ . & . & . \\ . & . & . \\ T(n-1) & H(n-1) & 1 \end{bmatrix} \begin{bmatrix} \varphi_{21} \\ \varphi_{22} \\ \omega_2 \end{bmatrix}$$

We want the values of Φ_i to use the given data to predict future values as accurately as possible. To ensure that these values are good predictors we can use the least squares fitting method.

3 Least Squares Fitting

The least squares method requires minimizing the sum of squares of the error (SSE). The SSE is found by squaring the error for each value, and that error is found by subtracting the predicted value from the actual value. So in this scenario, our actual values are given by the lefthand side of the two above equations and the predicted values are on the right side of the equation. Starting with temperature, we have:

$$\begin{bmatrix} T(2) \\ T(3) \\ . \\ . \\ . \\ T(n) \end{bmatrix} - \begin{bmatrix} T(1) & H(1) & 1 \\ T(2) & H(2) & 1 \\ . & . & . \\ . & . & . \\ . & . & . \\ T(n-1) & H(n-1) & 1 \end{bmatrix} \begin{bmatrix} \varphi_{11} \\ \varphi_{12} \\ \omega_1 \end{bmatrix}$$

Multiplying the two matrices we have:

$$\begin{bmatrix} T(2) \\ T(3) \\ . \\ . \\ . \\ T(n) \end{bmatrix} - \begin{bmatrix} T(1)\varphi_{11} + H(1)\varphi_{12} + \omega_1 \\ T(2)\varphi_{11} + H(2)\varphi_{12} + \omega_1 \\ ... \\ T(n-1)\varphi_{11} + H(n-1)\varphi_{12} + \omega_1 \end{bmatrix}$$

After subtracting:

$$\begin{bmatrix} T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1 \\ T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} - \omega_1 \\ ... \\ T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1 \end{bmatrix}$$

Now that we've found the error, we need to find the SSE. To do this we need to square each term and find the sum of those squares. Our error is a vector, so in order to find the square

we need to multiply it by its transpose. So:

$$\begin{bmatrix} T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1 \\ T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1 \\ \dots \\ T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1 \end{bmatrix}^{\tau} \begin{bmatrix} T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1 \\ T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1 \\ \dots \\ T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1 \end{bmatrix}$$

This becomes:

$$\begin{bmatrix} T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1 & T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1 & \dots & T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1 \end{bmatrix}$$

$$* \begin{bmatrix} T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1 \\ T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1 \\ \dots \\ T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1 \end{bmatrix}$$

Multiplying two vectors like this results in a single value. We end up with the SSE:

$$\begin{aligned} SSE = {} & (T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1)^2 + (T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1)^2 \\ & + \dots + (T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1)^2 \end{aligned}$$

Next we need to minimize the SSE we just found. We can do this by taking partial derivatives for each variable we need to use to minimize the equation. In this problem, we need to find $\varphi_{11}, \varphi_{12}$, and ω_1. So we have:

$$\begin{aligned} \frac{\delta SSE}{\delta \varphi_{11}} = {} & 2(T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1)(-T(1)) \\ & + 2(T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1)(-T(2)) + \dots \\ & + 2(T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1)(-T(n-1)) \end{aligned} \tag{1}$$

$$\begin{aligned} \frac{\delta SSE}{\delta \varphi_{12}} = {} & 2(T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1)(-H(1)) \\ & + 2(T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1)(-H(2)) + \dots \\ & + 2(T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1)(-H(n-1)) \end{aligned} \tag{2}$$

$$\begin{aligned} \frac{\delta SSE}{\delta \omega_1} = {} & 2(T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1)(-1) + 2(T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1)(-1) \\ & + \dots + 2(T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1)(-1) \end{aligned} \tag{3}$$

In order to minimize, we need to set these derivatives equal to zero and solve for the variable we differentiated with respect to. So for φ_{11} we can set (1) equal to zero as:

$$\begin{aligned} & 2(T(2)-T(1)\varphi_{11}-H(1)\varphi_{12}-\omega_1)(-T(1)) + 2(T(3)-T(2)\varphi_{11}-H(2)\varphi_{12}-\omega_1)(-T(2)) \\ & + \dots + 2(T(n)-T(n-1)\varphi_{11}-H(n-1)\varphi_{12}-\omega_1)(-T(n-1)) = 0 \end{aligned}$$

Factoring out and dividing by -2 gives us:

$$(T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1)(T(1)) + (T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} - \omega_1)(T(2)) \\ + ... + (T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1)(T(n-1)) = 0$$

Distribute to simplify

$$\begin{aligned}&(T(1)T(2) - T(1)^2\varphi_{11} - T(1)H(1)\varphi_{12} - T(1)\omega_1) \\ &+ (T(2)T(3) - T(2)^2\varphi_{11} - T(2)H(2)\varphi_{12} - T(2)\omega_1) + ... \\ &+ (T(n-1)T(n) - T(n-1)^2\varphi_{11} - T(n-1)H(n-1)\varphi_{12} - T(n-1)\omega_1) = 0\end{aligned} \tag{4}$$

Next we want to reorganize this into the form $Ax+b$ where A is a matrix holding coefficients, x is a matrix holding the variables we are solving for $(\varphi_{11}, \varphi_{12}, \omega_1)$ and b holds the values that do not have the variables attached to them. So:

$$\begin{bmatrix}\alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3\end{bmatrix} \begin{bmatrix}\varphi_{11} \\ \varphi_{12} \\ \omega_1\end{bmatrix} + \begin{bmatrix}b_1 \\ b_2 \\ b_3\end{bmatrix} = 0$$

Before we can do this, we need to get equations (2) and (3) to this same state as equation (4). So we set (2) equal to zero and simplify the same way:

$$2(T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1)(-H(1)) + 2(T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} - \omega_1)(-H(2)) \\ + ... + 2(T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1)(-H(n-1)) = 0$$

$$(T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1)(H(1)) + (T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} - \omega_1)(H(2)) \\ + ... + (T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1)(H(n-1)) = 0$$

$$\begin{aligned}&T(2)H(1) - T(1)H(1)\varphi_{11} - H(1)^2\varphi_{12} - H(1)\omega_1 + T(3)H(2) \\ &- T(2)H(2)\varphi_{11} - H(2)^2\varphi_{12} - H(2)\omega_1 + ... + T(n)H(n-1) \\ &- T(n-1)H(n-1)\varphi_{11} - H(n-1)^2\varphi_{12} - H(n-1)\omega_1 = 0\end{aligned} \tag{5}$$

Now we will do the same for (3):

$$2(T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1)(-1) + 2(T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} - \omega_1)(-1) \\ + ... + 2(T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1)(-1) = 0$$

$$(T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1) + (T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} - \omega_1) \\ + ... + (T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1) = 0 \tag{6}$$

So we will be using equations (4), (5), and (6). For better visualization, the three are written below as:

$$T(1)T(2) - T(1)^2\varphi_{11} - T(1)H(1)\varphi_{12} - T(1)\omega_1 + T(2)T(3) - T(2)^2\varphi_{11} - T(2)H(2)\varphi_{12} \\ - T(2)\omega_1 + ... + T(n-1)T(n) - T(n-1)^2\varphi_{11} - T(n-1)H(n-1)\varphi_{12} - T(n-1)\omega_1 = 0 \tag{7}$$

$$\begin{aligned} &T(2)H(1) - T(1)H(1)\varphi_{11} - H(1)^2\varphi_{12} - H(1)\omega_1 + T(3)H(2) \\ &- T(2)H(2)\varphi_{11} - H(2)^2\varphi_{12} - H(2)\omega_1 + ... + T(n)H(n-1) \\ &- T(n-1)H(n-1)\varphi_{11} - H(n-1)^2\varphi_{12} - H(n-1)\omega_1 = 0 \end{aligned} \tag{8}$$

$$\begin{aligned} &T(2) - T(1)\varphi_{11} - H(1)\varphi_{12} - \omega_1 + T(3) - T(2)\varphi_{11} - H(2)\varphi_{12} \\ &- \omega_1 + ... + T(n) - T(n-1)\varphi_{11} - H(n-1)\varphi_{12} - \omega_1 = 0 \end{aligned} \tag{9}$$

First we will find α_1, the coefficients of φ_{11} from equation (7):

$$\alpha_1 = -T(1)^2 - T(2)^2 - ... - T(n-1)^2$$

Next we will find α_2, the coefficients of φ_{12} from equation (7):

$$\alpha_2 = -T(1)H(1) - T(2)H(2) - ... - T(n-1)H(n-1)$$

Then we find α_3, the coefficients of ω_1 from equation (7):

$$\alpha_3 = -T(1) - T(2) - ... - T(n-1)$$

Next we find β_1, the coefficients of φ_{11} in equation (8):

$$\beta_1 = -T(1)H(1) - T(2)H(2) - ... - T(n-1)H(n-1)$$

Then β_2, the coefficients of φ_{12} in equation (8):

$$\beta_2 = -H(1)^2 - H(2)^2 - ... - H(n-1)^2$$

Then we find β_3, the coefficients of ω_1 in equation (8):

$$\beta_3 = -H(1) - H(2) - ... - H(n-1)$$

Finally, we will find the values of the last row in A, starting with γ_1, the coefficients of φ_{11} in equation (9):

$$\gamma_1 = -T(1) - T(2) - ... - T(n-1)$$

Next γ_2, the coefficients of φ_{12} in equation (9):

$$\gamma_2 = -H(1) - H(2) - ... - H(n-1)$$

Finally we can find γ_3, the coefficients of ω_1 in equation (9):

$$-1 - 1 - ... - 1$$

which can be reduced to:

$$-(n-1)$$

Now we can load those values into our matrix A so:

$$\begin{bmatrix} -T(1)^2 - T(2)^2 - ... - T(n-1)^2 & -T(1)H(1) - T(2)H(2) - ... - T(n-1)H(n-1) & -T(1) - T(2) - ... - T(n-1) \\ -T(1)H(1) - T(2)H(2) - ... - T(n-1)H(n-1) & -H(1)^2 - H(2)^2 - ... - H(n-1)^2 & -H(1) - H(2) - ... - H(n-1) \\ -T(1) - T(2) - ... - T(n-1) & -H(1) - H(2) - ... - H(n-1) & -(n-1) \end{bmatrix}$$

Now that we have set up A, and we know x, we need to set up b. B will contain all the values for equations (7), (8), (9) that do not have the variables in x.
First we'll find b_1 from equation (7):

$$b_1 = T(1)T(2) + T(2)T(3) + ... + T(n-1)T(n)$$

We will find b_2 from equation (8):

$$b_2 = T(2)H(1) + T(3)H(2) + ... + T(n)H(n-1)$$

Lastly we have b_3 from equation (9):

$$b_3 = T(2) + T(3) + ... + T(n)$$

So now we have our vector b:

$$\begin{bmatrix} T(1)T(2) + T(2)T(3) + ... + T(n-1)T(n) \\ T(2)H(1) + T(3)H(2) + ... + T(n)H(n-1) \\ T(2) + T(3) + ... + T(n) \end{bmatrix} \tag{10}$$

So from our equation $Ax + b = 0$ we need to solve for x, since that holds the variables we are trying to minimize.

$$Ax = -b$$

Since every element in A is negative, we can factor a negative one out and have:

$$-Ax = -b$$

Then we can cancel the negatives out on both sides of the equation and have:

$$Ax = b$$

Then solve for x by moving A to the other side:

$$x = A^{-1}b$$

This exact same process would be done for the sea level values, this time solving for $\varphi_{21}, \varphi_{22}$, and ω_2. We would end up with a solution in the exact same form. The values we have found can be generalized into a form that encompasses both temperature and sea level.
An analysis of A shows that:

$$A = \Lambda^\tau \Lambda$$

And an analysis of b shows that:

$$b = \Lambda^\tau Y_1$$

We can also see that:

$$x = \hat{\Phi}_1$$

So for temperature, we would have:

$$\hat{\Phi}_1 = (\Lambda^\tau \Lambda)^{-1} \Lambda^\tau Y_1$$

This can be generalized as:

$$\hat{\Phi}_i = (\Lambda^\tau \Lambda)^{-1} \Lambda^\tau Y_i$$

So for $i = 1$ we have the values that will best predict temperature and for $i = 2$ we have values to best predict sea level rise. To combine these into one matrix we can define $\hat{\Psi}$ as

$$\hat{\Psi} = [\hat{\Phi}_1, \hat{\Phi}_2]^\tau$$

and we can define $\mathbf{Y}$ as:

$$\mathbf{Y} = [Y_1, Y_2]$$

So then we have:

$$\hat{\Psi} = [(\Lambda^\tau \Lambda)^{-1} \Lambda^\tau \mathbf{Y}]^\tau$$

The output of this matrix will be the numerical values of our best estimators, given as:

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \omega_1 \\ \varphi_{21} & \varphi_{22} & \omega_2 \end{bmatrix}$$

4 Confidence Intervals

Despite the precautions taken to make the models as accurate as possible, there is always going to be some level of uncertainty. Whether this uncertainty was a result of a flaw in the model or a measurement error, it needs to be accounted for. In order to do so, Aral, Guan, and Chang found a 90% confidence interval around their predicted values[1]. This meant that they were 90% sure that the actual values were in these particular intervals around the predicted values.

For this particular data, the best interval to use would be a t-interval. That is because the distribution is approximately normal, with a large sample size, but the standard deviation of the population is unknown. The general form for a t interval is $\bar{x} \pm t_{\frac{\alpha}{2},df}$ multiplied by the standard error of the mean, where df denotes the degrees of freedom.

In our problem, this will look a little different. In place of $\bar{x}$ we will have the predicted values, so $\hat{T}(k)$ for temperature and $\hat{H}(k)$ for sea level. Since we have n-1 data points in our regression and we lose a degree of freedom for each of the values we are estimating (which are $\varphi_{i1}, \varphi_{i2}, \omega_i$) we have $n - 1 - 3 = n - 4$ degrees of freedom. Our standard error, which estimates how far our sample mean is from the population mean, can be found in the following way[8]:
First we must identify our function Y. For this problem we will have two Y functions, one for temperature and one for sea level. These are the same Ys we have been using in the past, defined by:

$$\mathbf{Y}_i = \Lambda \Phi_i$$

For consistency and simplicity, we will be working with $\mathbf{Y}_1$, the temperature formula. That leaves us with :

$$\mathbf{Y}_1 = \begin{bmatrix} T(1) & H(1) & 1 \\ T(2) & H(2) & 1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ T(n-1) & H(n-1) & 1 \end{bmatrix} \begin{bmatrix} \varphi_{11} \\ \varphi_{12} \\ \omega_1 \end{bmatrix}$$

Which we can reduce to:

$$\mathbf{Y}_1 = \begin{bmatrix} T(1)\varphi_{11} + H(1)\varphi_{12} + \omega_1 \\ T(2)\varphi_{11} + H(2)\varphi_{12} + \omega_1 \\ ... \\ T(n-1)\varphi_{11} + H(n-1)\varphi_{12} + \omega_1 \end{bmatrix}$$

We want to get Y into the most simple form, so we can define an x such that:

$$X = \begin{bmatrix} T(1) & H(1) & 1 \\ T(2) & H(2) & 1 \\ ... & ... & ... \\ T(n-1) & H(n-1) & 1 \end{bmatrix}$$

Note that $X = \Lambda$ as defined for the system of differential equations. We can also define β as:

$$\beta = \begin{bmatrix} \varphi_{11} \\ \varphi_{12} \\ \omega_1 \end{bmatrix}$$

So we can rewrite our Y as:

$$\mathbf{Y} = X\beta + \epsilon$$

In this formula, ϵ is the error, which is normally distributed with a mean of 0 and a variance of σ^2.
Let $\hat{\beta}$ be the predicted values of β. then we have our predicted y as:

$$\hat{y} = X_k\hat{\beta}$$

Since we used predicted values to obtain the value used for the confidence interval, we are looking for the variance of the difference between the actual values and the predicted values. In order to do this, we need to know the variances of both the actual values and the predicted values. However, before we can do that we need to find the variances of various parts of these formulas. Since both formulas involve β or $\hat{\beta}$, we'll start there.

4.1 Finding $\hat{\beta}$

In order to work with $\hat{\beta}$ we need to understand and find a formula for it. Remember, $\hat{\beta}$ is the predicted values of β which is given as:

$$\beta = \begin{bmatrix} \varphi_{11} \\ \varphi_{12} \\ \omega_1 \end{bmatrix}$$

Furthermore, we have already minimized the predicted values of β through least squares, so our $\hat{\beta}$ is given by that solution:

$$\hat{\beta} = (\Lambda^\tau\Lambda)^{-1}\Lambda^\tau Y$$

Recall that $X = \Lambda$ and replace the variables to get:

$$\hat{\beta} = (X^\tau X)^{-1} X^\tau Y$$

Since we have defined $Y = X\beta + \epsilon$, we can plug that in:

$$\hat{\beta} = (X^\tau X)^{-1} X^\tau (X\beta + \epsilon)$$

Distribute that through:

$$\hat{\beta} = (X^\tau X)^{-1} X^\tau X\beta + (X^\tau X)^{-1} X^\tau \epsilon$$

Because $X^\tau X$ is being multiplied by its inverse, it goes away, leaving us with:

$$\hat{\beta} = \beta + (X^\tau X)^{-1} X^\tau \epsilon \tag{11}$$

4.2 Expectation of $\hat{\beta}$

In order to properly use $\hat{\beta}$ we need to know its expected value. Using the formula we just found:

$$E[\hat{\beta}] = E[\beta + (X^\tau X)^{-1} X^\tau \epsilon]$$

Because β and $(X^\tau X)^{-1} X^\tau \epsilon$ are independent, their expected values can be separated so:

$$E[\hat{\beta}] = E[\beta] + E[(X^\tau X)^{-1} X^\tau \epsilon]$$

$E[\beta] = \beta$ and once again the constants can be pulled out, so we have:

$$E[\hat{\beta}] = \beta + (X^\tau X)^{-1} X^\tau E[\epsilon]$$

We have previously stated that ϵ is normally distributed with a mean of 0. That means $E[\epsilon] = 0$ so we end up with:

$$E[\hat{\beta}] = \beta$$

This makes $\hat{\beta}$ an unbiased estimator of β.

4.3 Variance of $\hat{\beta}$

Now that we have found both a formula for $\hat{\beta}$ and its expected value, we can find its variance. Note that the variance of a variable is the same as the covariance between the variable and itself ($var[x] = cov[x, x]$).
Covariance can be found using the following formula:

$$cov[x, y] = E[(X - E(X))(Y - E(Y))^\tau]$$

Where E is the expectation for the variable in the parenthesis. So plugging in $\hat{\beta}$ gives us:

$$cov[\hat{\beta}, \hat{\beta}] = E[(\hat{\beta} - E(\hat{\beta}))(\hat{\beta} - E(\hat{\beta}))^\tau]$$

Since we just found $E[\hat{\beta}]$ we can plug β in for those values.

$$cov[\hat{\beta}, \hat{\beta}] = E[(\hat{\beta} - \beta)(\hat{\beta} - \beta)^\tau]$$

Substituting the formula we found for $\hat{\beta}$ into this we have:

$$cov[\hat{\beta}, \hat{\beta}] = E[(\beta + (X^\tau X)^{-1} X^\tau \epsilon - \beta)(\beta + (X^\tau X)^{-1} X^\tau \epsilon - \beta)^\tau]$$

Subtracting the βs gives us:

$$cov[\hat{\beta}, \hat{\beta}] = E[((X^\tau X)^{-1} X^\tau \epsilon)((X^\tau X)^{-1} X^\tau \epsilon)^\tau]$$

Multiplying them together and rearranging to put the ϵ values together gives us:

$$cov[\hat{\beta}, \hat{\beta}] = E[(X^\tau X)^{-1} X^\tau \epsilon \epsilon^\tau X (X^\tau X)^{-1}]$$

We factor the Xs out to get:

$$cov[\hat{\beta}, \hat{\beta}] = (X^\tau X)^{-1} X^\tau E[\epsilon \epsilon^\tau] X (X^\tau X)^{-1} \tag{12}$$

Now we need to find $E[\epsilon \epsilon^\tau]$. Remember we assumed ϵ to be normally distributed with a mean (expected value) of 0 and a variance of σ^2. So we have:

$$var[\epsilon] = cov[\epsilon, \epsilon] = E[(\epsilon - E[\epsilon])(\epsilon - E[\epsilon])^\tau]$$

Replacing $E[\epsilon]$ with 0 from our assumptions, we have:

$$cov[\epsilon, \epsilon] = E[\epsilon \epsilon^\tau] = var[\epsilon] = \sigma^2$$

Therefore we can substitute σ^2 into (12), but we have to multiply it by the identity matrix since it is being multiplied by matrices:

$$cov[\hat{\beta}, \hat{\beta}] = (X^\tau X)^{-1} X^\tau (\sigma^2 I) X (X^\tau X)^{-1}$$

Pulling the σ^2 to the front we have:

$$cov[\hat{\beta}, \hat{\beta}] = (\sigma^2)(X^\tau X)^{-1} X^\tau X (X^\tau X)^{-1}$$

Since $X^\tau X$ is being multiplied by its inverse, it cancels out leaving us with a reduced answer:

$$\begin{aligned} cov[\hat{\beta}, \hat{\beta}] &= var[\hat{\beta}] \\ &= (\sigma^2)(X^\tau X)^{-1} \end{aligned} \tag{13}$$

Now that we've found all the necessary parts of our main formulas, we can continue working on the overall variance. First we need to find the variance of y.

4.4 Variance of y

We want to find the variance of Y at year k so we begin by plugging in our formula for y:

$$var[y(k)] = var[X\beta + \epsilon_k]$$

Because $X\beta$ and ϵ_k are independent you can separate them:

$$var[y(k)] = var[X\beta] + var[\epsilon_k]$$

β is an unbiased estimator, which means its variance is 0 so we are left with:

$$var[y] = var[\epsilon_k]$$

Earlier we stated that ϵ is normally distributed with a mean of zero and a variance of σ^2 so we have found that:

$$\begin{aligned} var[y] &= var[\epsilon_k] \\ &= \sigma^2 \end{aligned} \tag{14}$$

Next we need to find the variance of our predicted y so we can find the variance of the difference.

4.5 Variance of $\hat{y}$

From our definition of $\hat{y}$ we know that:

$$var[\hat{y}] = var[X_k\hat{\beta}]$$

Using our formula for covariance, we can find the variance of Y as follows:

$$cov[\hat{y}, \hat{y}] = E[(\hat{y} - E(\hat{y}))(\hat{y} - E(\hat{y}))^\tau]$$

Because we stated that $\hat{y} = X_k\hat{\beta}$ we can substitute that in:

$$cov[\hat{y}, \hat{y}] = E[(X_k\hat{\beta} - E(X_k\hat{\beta}))(X_k\hat{\beta} - E(X_k\hat{\beta}))^\tau]$$

Looking at $E(X_k\hat{\beta})$, we can pull X_k out because it only contains constants. We are left with $X_kE(\hat{\beta})$. We found that $\hat{\beta}$ is an unbiased estimator of β so $E(\hat{\beta}) = \beta$. We can plug that into our variance formula and have:

$$cov[\hat{y}, \hat{y}] = E[(X_k\hat{\beta} - X_k\beta)(X_k\hat{\beta} - X_k\beta)^\tau]$$

Once again, since X contains constants and it is in both terms in each set, it can be factored out. Note that in the second set it must be transposed.

$$cov[\hat{y}, \hat{y}] = E[X_k(\hat{\beta} - \beta)(\hat{\beta} - \beta)^\tau X_k^\tau]$$

Again, because X_k contains constants, $E(X_k) = X_k$ so:

$$cov[\hat{y}, \hat{y}] = X_kE[(\hat{\beta} - \beta)(\hat{\beta} - \beta)^\tau]X_k^\tau$$

The middle of this can be rewritten so we have:

$$cov[\hat{y}, \hat{y}] = X_k E[(\hat{\beta} - E(\hat{\beta}))(\hat{\beta} - E(\hat{\beta}))^\tau] X_k^\tau$$

It is now clear that the middle is the same as $cov[\hat{\beta}, \hat{\beta}]$ so:

$$cov[\hat{y}, \hat{y}] = X_k cov[\hat{\beta}, \hat{\beta}] X_k^\tau$$

We can now plug our formula for $cov[\hat{\beta}, \hat{\beta}]$ (13) into this equation.

$$\begin{aligned} cov[\hat{y}, \hat{y}] &= X_k cov[\hat{\beta}, \hat{\beta}] X_k^\tau \\ &= X_k(\sigma^2)(X^\tau X)^{-1} X_k^\tau \\ &= (\sigma^2) X_k (X^\tau X)^{-1} X_k^\tau \end{aligned} \tag{15}$$

4.6 Variance of the difference between actual and predicted values

Finally we can find the variance of the difference of $y(k) - \hat{y}(k)$:

$$var[y(k) - \hat{y}(k)] = var[\epsilon_k - \hat{y}(k)]$$

We are able to make this substitution because in (14) we showed that $var[y(k)] = var[e_k]$. We can break this variance up as:

$$var[y(k) - \hat{y}(k)] = var[\epsilon_k] + var[\hat{y}(k)] - 2cov[\epsilon_k, \hat{y}(k)]$$

We can assume that ϵ_k is independent from $\hat{y}(k)$ because ϵ_k is independent from $\hat{\beta}$ and $\hat{y}(k) = X\hat{\beta}$. This means $cov[\epsilon_k, \hat{y}(k)] = 0$ so:

$$var[y(k) - \hat{y}(k)] = var[\epsilon_k] + var[\hat{y}(k)]$$

Plugging in the values we found for $var[\epsilon_k]$ and $var[\hat{y_k}]$ from equations (14) and (15) we have:

$$var[y(k) - \hat{y}(k)] = \sigma^2 + (\sigma^2) X_k (X^\tau X)^{-1} X_k^\tau \tag{16}$$

You can factor out a σ^2 and have:

$$var[y(k) - \hat{y}(k)] = \sigma^2 (1 + X_k (X^\tau X)^{-1} X_k^\tau)$$

This is our variance. The square root of this will be our standard error.

4.7 Setting up the Confidence Interval

For our values, X_k will be:

$$\begin{bmatrix} \hat{T}(k) & \hat{H}(k) & 1 \end{bmatrix}$$

So we can define a $\hat{Z}(k)$, such that $X_k = \hat{Z}(k)^\tau$
For our regular X values, we can replace them with Λ as defined in the solution of the system of differential equations. That makes our formula turn into the following:

$$\hat{\sigma}^2 (1 + \hat{Z}(k)^\tau (\Lambda^\tau \Lambda)^{-1} \hat{Z}(k))$$

This is the formula for the variance for both the temperature and sea level. The only difference would be in the sigmas, which will therefore be denoted as σ_T and σ_H. This same process would be followed to find this same formula for the sea level confidence interval. Once we take the square root of the variance and put it in the general form for a t-interval, as mentioned in the beginning of this section, we find the following confidence intervals:

$$\begin{cases} \widehat{T}_{CI}(k) = \widehat{T}(k) \pm t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_T^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))} \\ \widehat{H}_{CI}(k) = \widehat{H}(k) \pm t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_H^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))} \end{cases}$$

In the above formulas, $\hat{T}_{CI}(k)$ and $\hat{H}_{CI}(k)$ are the values of the confidence intervals around the temperature and sea level at year k, respectively. $\hat{T}(k)$ and $\hat{H}(k)$ are the predicted temperature and sea level values at year k. $t_{\frac{\alpha}{2},n-4}$ is the t score of our distribution, where we have n-4 degrees of freedom. Because we are doing a 90% confidence interval, we used a t score of 1.645. $\hat{\sigma}_T^2$ and $\hat{\sigma}_H^2$ are the variances of the predicted temperature and sea level data. $\hat{Z(k)}$ is defined above, and Λ was defined in section 2.

5 Application of Model

The next step is to test to see if the theoretical model holds with the data. I decided to use python in a Jupyter Notebook to test this model since it has packages to handle these operations. Numpy in particular is very useful for the matrix operations.

5.1 Setting up the Problem

Before I could begin solving this problem, I got my data ready to be used. I deleted the columns containing the years of the data from both datasets to make it easier to work with. I also deleted the column headings so I just had the numbers I needed. I saved the altered data as csv files to make them easiest to read in.

After making these adjustments, I loaded the data into two separate Numpy arrays, one for temperature and one for sea level. My data was slightly different than what was used by Aral, Guan, and Chang. In order to correct for this, I had to shift the sea level numbers into cm from mm, so I divided my sea level values by 10. The temperature data was approximately $.44°C$ off so I subtracted .44 from my temperature values to get my data to match. These steps are shown in the screenshot below.

In [1]:
```
#import numpy and abbreviate
#import genfromtxt to read in our data

import numpy as np
from numpy import genfromtxt
import matplotlib.pyplot as plt
import statistics as stat
import math
```

Set up fundamentals

In [2]:
```
np.set_printoptions(suppress=True) #this prevents numpy from using scientific notation
```

read in temperature and sea level data and store them in arrays with values separated by ,s

In [3]:
```
temp = genfromtxt('temp-data.csv',delimiter=',')
sea = genfromtxt('sl-data.csv',delimiter=',')
```

adjust to make match

In [4]:
```
temp = temp - 0.44
sea = sea/10
```

Figure 1: Above is a screenshot of the code I used to set up the problem

Next I defined a moving average function that takes an array and a value n as parameters. This function uses cumsum to find the cumulative sum of each cell in the given array. Then I used slicing to make sure each cell only contained the sum of itself and its neighboring cell(s), depending on the provided n. Then I divided by n, getting the average. I used this to find the two-year moving averages for the sea level and temperature data.

```
def movingavg(a,n):
    sum = np.cumsum(a,dtype = float)
    sum[n:] = sum[n:] - sum[:-n]
    return (sum[n - 1:] / n)
```

```
t = movingavg(temp,2)
h = movingavg(sea,2)
```

Figure 2: Above is a screenshot of the code I used to calculate moving averages

5.2 Finding Ψ for 1880-1950

We are looking to find the values of $\varphi_{11}, \varphi_{12}, \varphi_{21}, \varphi_{22}, \omega_1$, and ω_2 to best model our data. Recall that, using least squares fitting, we determined the values that would give be the best predictors were given by Ψ, defined as:

$$\hat{\Psi} = [(\Lambda^\tau \Lambda)^{-1} \Lambda^\tau \mathbf{Y}]^\tau$$

I needed to define all the various parts of Ψ formula. To do this I used slicing to create arrays just containing data from 1880 to 1949; these are the values T(1), T(2), ... T(n-1) and H(1), H(2), .. H(n-1) which are the components of Λ for 1880-1950.

Then I formed the arrays for my Y values. Recall that Y_i is defined as:

$$\mathbf{Y}_1 = \{T(2), T(3), ..., T(n)\}^\tau$$

$$\mathbf{Y}_2 = \{H(2), H(3), ..., H(n)\}^\tau$$

I deleted the data from after 1950 and the data for 1880 for both temperature and sea level to get these arrays.

Once all of my individual arrays were of the correct size and contained the desired data, I paired them using column stacks. For Λ (also called x) I added ones to the third column of that array. This process is shown below.

```
tto50 = np.delete(t,slice(70,140))
sto50 = np.delete(h,slice(70,135))
```

```
yt50 = np.delete(t,slice(71,140))
yt50 = np.delete(yt50,0)
ys50 = np.delete(h,slice(71,140))
ys50 = np.delete(ys50,0)
```

```
a = np.column_stack((tto50,sto50,))
x = np.insert(a,2,1,axis =1)
```

```
y = np.column_stack((yt50,ys50))
```

Figure 3: Above is a screenshot of the code I used to set up matrices

Since we established Λ and Y, I was able to perform the necessary functions fairly easily. I used numpy.transpose to find Λ^τ and numpy.matmul to find $\Lambda^\tau \Lambda$. Then I used numpy.linalg.inv to find the inverse of that value. I could follow this same process to multiply Λ^τ and Y, then to multiply those results by $\Lambda^\tau \Lambda$.

The results of this are the best values of $\varphi_{11}, \varphi_{12}, \varphi_{21}, \varphi_{22}, \omega_1$, and ω_2 for this model for these years. This process is shown below.

```
xt = x.transpose()
```

```
xtx = np.matmul(xt,x)
```

```
xtxinv = np.linalg.inv(xtx)
```

```
xty = np.matmul(xt,y)
```

```
xtxinvxty = np.matmul(xtxinv,xty)
```

```
xtxinvxtyt = xtxinvxty.transpose()
```

```
xtxinvxtyt
```

```
array([[ 0.83743348,  0.00633697, -0.02550508],
       [ 0.55402455,  0.98634909,  0.31491215]])
```

Figure 4: Above is a screenshot of the code solving for Ψ for 1880-1950

5.3 Finding Ψ for 1880-2001

This exact same process was followed to find the values of Ψ for the years 1880 to 2001. This is shown below.

```
tto01= np.delete(t,slice(121,140))
sto01 = np.delete(h,slice(121,135))
```

```
yt01 = np.delete(t,slice(122,140))
yt01 = np.delete(yt01,0)
ys01 = np.delete(h,slice(122,140))
ys01 = np.delete(ys01,0)
```

```
b = np.column_stack((tto01,sto01,))
x2 = np.insert(b,2,1,axis =1)
```

```
y2 = np.column_stack((yt01,ys01))
```

```
xt2 = x2.transpose()
```

```
xtx2 = np.matmul(xt2,x2)
```

```
xtxinv2 = np.linalg.inv(xtx2)
```

```
xty2 = np.matmul(xt2,y2)
```

```
xtxinvxty2 = np.matmul(xtxinv2,xty2)
```

```
xtxinvxtyt2 = xtxinvxty2.transpose()
```

```
xtxinvxtyt2
```

```
array([[ 0.8292325 ,  0.00800358, -0.01266538],
       [ 0.4673051 ,  0.9865147 ,  0.26720449]])
```

Figure 5: Above is a screenshot of the code solving for Ψ for 1880-2001

5.4 Calculating R^2

With these values, I was able to use past values to predict future ones. We have previously defined Y as:

$$\mathbf{Y}_i = \Lambda\Phi_i$$

Since $\Psi = \Phi^\tau$, we can use the values we found before we transposed them to calculate our predicted Y values.

I defined two new arrays: predicted temperature values and predicted sea level values. I found these arrays using numpy.matmul to multiply our φ and ω values by Λ. I calculated predicted values for both 1880-1950 and 1880-2001. Then I found R^2, the square of the correlation coefficient, for each group to find how much of the data was accurately modeled by the predictions. This is shown below.

```
phi = xtxinvxty
ypred = np.matmul(x,phi)
```

```
temp_pred = ypred[:,0]
tpred50 = np.delete(temp_pred,slice(72,140))
```

```
sea_pred = ypred[:,1]
sea_pred
hpred50 = np.delete(sea_pred,slice(72,145))
```

```
tempcoef = np.corrcoef(tpred50,yt50)[1,0]
tempcoef
```

```
0.8956680311332991
```

```
seacoef = np.corrcoef(hpred50,ys50)[1,0]
seacoef
```

```
0.9894189852874723
```

Figure 6: Above is a screenshot of the code finding the correlation coefficients for 1880-1950

```
phi2 = xtxinvxty2
ypred2 = np.matmul(x2,phi2)
```

```
temp_pred2 = ypred2[:,0]
```

```
sea_pred2 = ypred2[:,1]
```

```
tempcoef2 = np.corrcoef(temp_pred2,yt01)[1,0]
tempcoef2
```

```
0.9591124195809934
```

```
seacoef2 = np.corrcoef(sea_pred2,ys01)[1,0]
seacoef2
```

```
0.9979716326987719
```

Figure 7: Above is a screenshot of the code finding the correlation coefficients for 1880-2001

5.5 Calculating Confidence Intervals

Next I calculated the values of the confidence intervals around each set of predictions. The formula for the confidence intervals is given by:

$$\begin{cases} \widehat{T}_{CI}(k) = \widehat{T}(k) \pm t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_T^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))} \\ \widehat{H}_{CI}(k) = \widehat{H}(k) \pm t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_H^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))} \end{cases}$$

The first step to solving this is to make sure we have all the pieces. We already have the predicted values ($\hat{T}(k)$ and $\hat{H}(k)$). Aral, Guan, and Chang gave $t_{\frac{\alpha}{2},n-2} = 1.645$ for their 90% level of confidence[1], so that is what I used as well. I used the python statistics package to calculate the variance for the predicted values of temperature and sea level. I then created different arrays, one for each of the below:

$$\widehat{T}_{CI}(k) = \widehat{T}(k) + t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_T^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))}$$

$$\widehat{T}_{CI}(k) = \widehat{T}(k) - t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_T^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))}$$

$$\widehat{H}_{CI}(k) = \widehat{H}(k) + t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_H^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))}$$

$$\widehat{H}_{CI}(k) = \widehat{H}(k) - t_{\frac{\alpha}{2},n-4}\sqrt{\hat{\sigma}_H^2(1+\hat{Z}(k)^\tau(\Lambda^\tau\Lambda)^{-1}\hat{Z}(k))}$$

Then I used a for loop to cycle through each year, creating an array Z such that:

$$Z = \begin{bmatrix} T(k) \\ H(k) \\ 1 \end{bmatrix}$$

for each year k. For each year, I calculated the confidence interval values and added them to the appropriate arrays. This work is below.

```
tvar50 = stat.variance(tpred50)
```

```
posconft50 = np.array([])
negconft50 = np.array([])
for i in range(70):
    temporary = np.column_stack((tto50[i],sto50[i],))
    z = np.insert(temporary,2,1,axis =1).transpose()
    ci1 = tpred50[i] + 1.645*math.sqrt(tvar50*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    ci2 = tpred50[i] - 1.645*math.sqrt(tvar50*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    posconft50 = np.append(posconft50,ci1)
    negconft50 = np.append(negconft50,ci2)
```

Figure 8: Finding the confidence intervals for temperature for 1880-1950

```
hvar50 = stat.variance(hpred50)
```

```
posconfh50 = np.array([])
negconfh50 = np.array([])
for i in range(70):
    temporary = np.column_stack((tto50[i],sto50[i],))
    z = np.insert(temporary,2,1,axis =1).transpose()
    ci1 = hpred50[i] + 1.645*math.sqrt(hvar50*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    ci2 = hpred50[i] - 1.645*math.sqrt(hvar50*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    posconfh50 = np.append(posconfh50,ci1)
    negconfh50 = np.append(negconfh50,ci2)
```

Figure 9: Finding the confidence intervals for sea level for 1880-1950

```
tvar01 = stat.variance(temp_pred2)
```

```
posconft01 = np.array([])
negconft01 = np.array([])
for i in range(121):
    temporary = np.column_stack((tto01[i],sto01[i],))
    z = np.insert(temporary,2,1,axis =1).transpose()
    ci1 = temp_pred2[i] + 1.645*math.sqrt(tvar01*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    ci2 = temp_pred2[i] - 1.645*math.sqrt(tvar01*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    posconft01 = np.append(posconft01,ci1)
    negconft01 = np.append(negconft01,ci2)
```

Figure 10: Finding the confidence intervals for temperature for 1880-2001

```
hvar01 = stat.variance(sea_pred2)
```

```
posconfh01 = np.array([])
negconfh01 = np.array([])
for i in range(121):
    temporary = np.column_stack((tto01[i],sto01[i],))
    z = np.insert(temporary,2,1,axis =1).transpose()
    ci1 = sea_pred2[i] + 1.645*math.sqrt(hvar01*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    ci2 = sea_pred2[i] - 1.645*math.sqrt(hvar01*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    posconfh01 = np.append(posconfh01,ci1)
    negconfh01 = np.append(negconfh01,ci2)
```

Figure 11: Finding the confidence intervals for sea level for 1880-2001

5.6 Graphs

I created several graphs from my results; some show predicted values and actual values, and others show the data points, the prediction, and the confidence interval. I used matplotlib to make the graphs, and used np.linspace to set the x-axis to be the desired years. For example:

```
to1950 = np.linspace(1880,1950,70)
plt.plot(to1950, tto50, 'ko', label="temperature data")
plt.plot(to1950, tpred50, label = "predicted temperatures")
plt.plot(to1950, posconft50, '--', label = "confidence interval")
plt.plot(to1950, negconft50, '--',label = "confidence interval")
plt.legend(loc= 'best')
```

Figure 12: The above screenshot shows how I created my first graph

5.7 2100 Prediction

The graphs and my correlation coefficients suggest my prediction model was very accurate so I felt confident using my values to predict future changes.

I used the data for the temperature and sea level in 2001 and my matrix Ψ from 1880-2001 to predict the values for 2002 using the same $\mathbf{Y}_i = \Lambda\Phi_i$ formula. My newly calculated 2002 values and the same Ψ values were then multiplied to 2003. This process continued in a loop to calculate values up until 2100 so I could compare my predictions to those by the IPCC. This is shown below.

```
xk = np.array([])
xk = np.append(xk,tto01[120])
xk = np.append(xk,sto01[120])
xk = np.append(xk,1)
predvalues = np.array([])
for i in range(99):
    prediction = np.matmul(xk,phi2)
    predvalues = np.append(predvalues, prediction)
    xk = prediction
    xk = np.append(xk,1)
predvalues = np.reshape(predvalues, (99,2))
predvalues
```

Figure 13: The above screenshot shows how I predicted the sea level and temperature change in 2100

The output of this code gives a 99 by 2 array with the first column holding the temperature values and the second column holding sea level values. The final row gives the predictions for 2100.

5.8 Confidence Intervals Around Predictions

In order to find the confidence intervals around the predicted values, I created separate arrays to hold the predicted temperature values and the predicted sea level values. I then found the variances of these arrays. I set up a new X array of the predicted values, and performed the usual operations to find $(X^\tau X)^{-1}$ or $(\Lambda^\tau \Lambda)^{-1}$. In a for loop, I set up Z and calculated the confidence interval for each row of the predicted arrays. Once this was completed, I printed out the confidence interval values for year 2100. This process is seen below.

```
predictedtemps = predvalues[:,0]
predictedtempvar = stat.variance(predictedtemps)
```

```
predictedsea = predvalues[:,1]
predictedseavar = stat.variance(predictedsea)
```

```
x = np.insert(predvalues,2,1,axis=1)
xt = x.transpose()
xtx = np.matmul(xt,x)
xtxinv = np.linalg.inv(xtx)
```

```
posconftemppred = np.array([])
negconftemppred = np.array([])
posconfseapred = np.array([])
negconfseapred = np.array([])

for i in range(99):
    temporary = np.column_stack((predictedtemps[i],predictedsea[i],))
    z = np.insert(temporary,2,1,axis =1).transpose()
    citemp1 = predictedtemps[i] + 1.645*math.sqrt(predictedtempvar*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    citemp2 = predictedtemps[i] - 1.645*math.sqrt(predictedtempvar*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    posconftemppred = np.append(posconftemppred,citemp1)
    negconftemppred = np.append(negconftemppred,citemp2)
    cisea1 = predictedsea[i] + 1.645*math.sqrt(predictedseavar*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    cisea2 = predictedsea[i] - 1.645*math.sqrt(predictedseavar*(1 + np.matmul(np.matmul(z.transpose(), xtxinv),z)))
    posconfseapred = np.append(posconfseapred,cisea1)
    negconfseapred = np.append(negconfseapred,cisea2)
print("The temperature in 2100 is between",negconftemppred[98],"and",posconftemppred[98])
print("The sea level in 2100 is between",negconfseapred[98],"and",posconfseapred[98])
```

Figure 14: The above screenshot shows how I found the confidence interval of my prediction

6 Results

Below is the table with the results of $\hat{\Psi}$. The results in each cell are formatted as follows:

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \omega_1 \\ \varphi_{21} & \varphi_{22} & \omega_2 \end{bmatrix}$$

	Aral, Guan, and Chang	My results
1880-1950	$\begin{bmatrix} 0.8337 & 0.0072 & 0.0083 \\ 0.3441 & 0.9960 & 0.2234 \end{bmatrix}$	$\begin{bmatrix} 0.8374 & 0.00634 & -0.0255 \\ 0.5540 & 0.9863 & 0.3149 \end{bmatrix}$
1880-2001	$\begin{bmatrix} 0.8074 & 0.0068 & -0.0110 \\ 0.4115 & 0.9956 & 0.2585 \end{bmatrix}$	$\begin{bmatrix} 0.8292 & 0.0080 & -0.0127 \\ 0.4673 & 0.9865 & 0.2672 \end{bmatrix}$

Table 1: Table containing $\hat{\Psi}$ values

It can be seen that my values differ from theirs to varying degrees. In most cases, they're off by a few hundredths or thousandths, though some are more substantial. There are several reasons this could have occurred.

First, it could be a result of differences in data. It was quite difficult to track down the data used in their paper and it's possible that I ended up finding different data that didn't match theirs, even with the alterations I made to correct for that possibility. Another possible cause lies in the software used to perform the calculations. Perhaps their software or mine rounded somewhere, causing the slight differences. There is also the possibility that I made some error in the set up of the problem. I double checked my moving average function and the values in my arrays to ensure that the right data was included, but did not find any errors that would result in different values.

I used these values to create my predicted values and found the correlation between the predictions and the data. My R^2 values, in table 2 below, were excellent and very close to 1, proving a strong correlation. If my numbers differed due to a miscalculation or an error in the set up of my problem, there would not be such a strong correlation, so I believe the problem was most likely in data differences.

However, this does not account for the large difference between my R^2 values and those found by Aral, Guan, and Chang. If the only difference lied in data, their values should have led them to the same correlation that I found. This leads me to wonder if it is indeed a software issue. They do not mention what software programs they used to create their models so I am unsure if there was indeed a problem with rounding or something within that program. The differences in the R^2 values can be seen below.

	Temperature	Sea Level
1880-1950	0.8022	0.97895
1880-2001	0.91987	0.99594

Table 2: Table containing my R^2 values

	Temperature	Sea Level
1880-1950	0.6	0.8
1880-2001	0.5	0.9

Table 3: Table containing Aral, Guan, and Chang's R^2 values

The correlations within my data can be visualized via graphs. I have created several graphs, as shown below. First are line graphs of the actual data and the predicted data. In these figures you can see how close the predicted values are to the actual data.

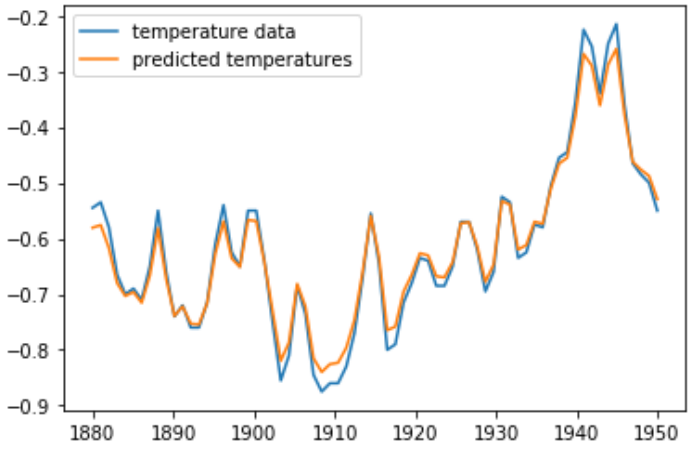

Figure 15: Above is a graph of my predicted and actual temperature values for 1880-1950.

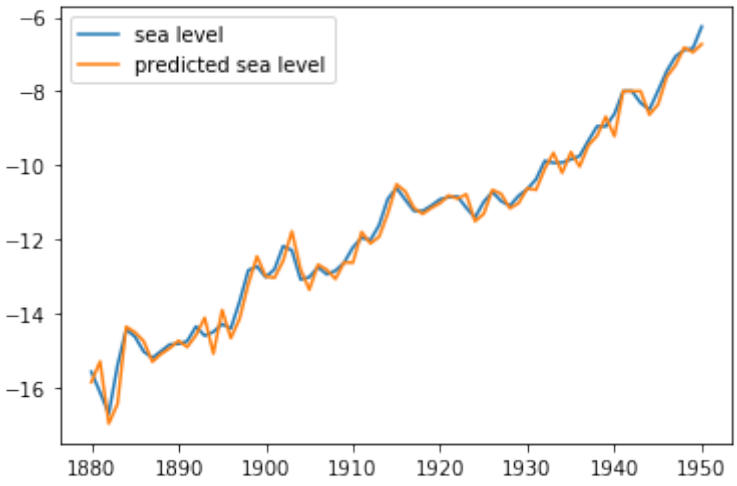

Figure 16: Above is a graph of my predicted and actual sea level values for 1880-1950.

With these visualization of my predicted values, it is obvious that my predictions did not form a straight line, but rather took the shape of the data. This explains why my values differed from Aral, Guan, and Chang's so dramatically. Their versions of these graphs are pictured below.

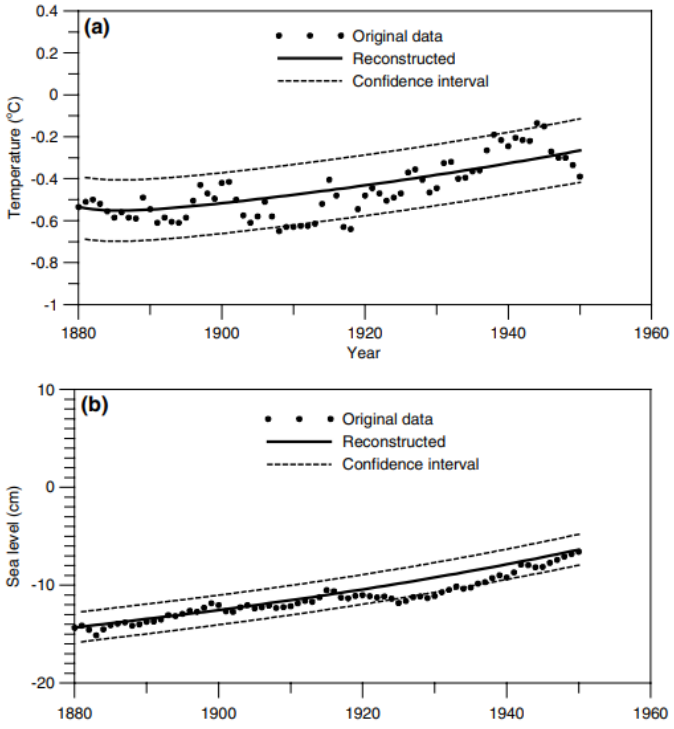

Fig. 1. Temperature and sea level from 1880 to 1950 reconstructed by the dynamic system model by using data set (1880–1950) and 90% confidence interval band

Figure 17: Above is the graphs of Aral, Guan, and Chang's data from 1880-1950.

The same can be seen with the graphs from 1880-2001.

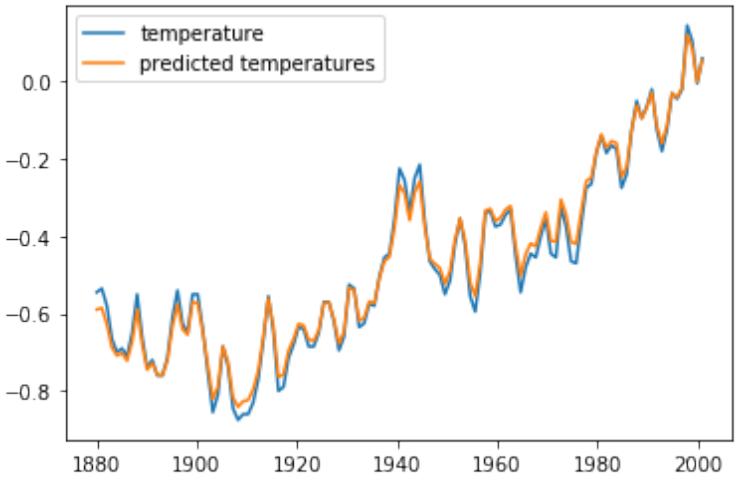

Figure 18: Above is a graph of my predicted and actual temperature values for 1880-2001.

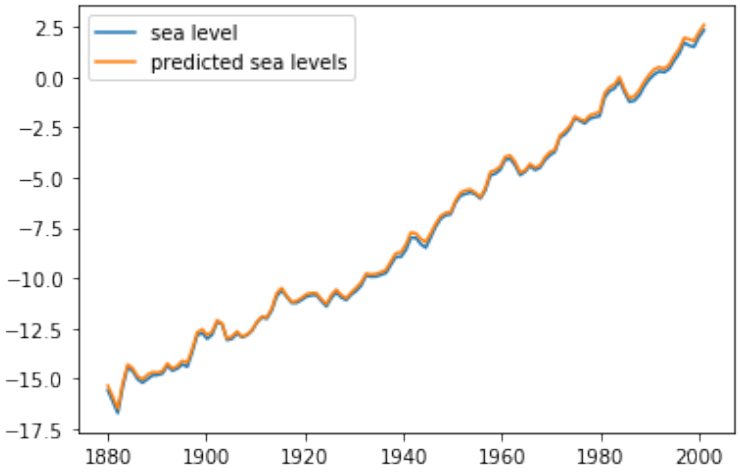

Figure 19: Above is a graph of my predicted and actual sea level values for 1880-2001.

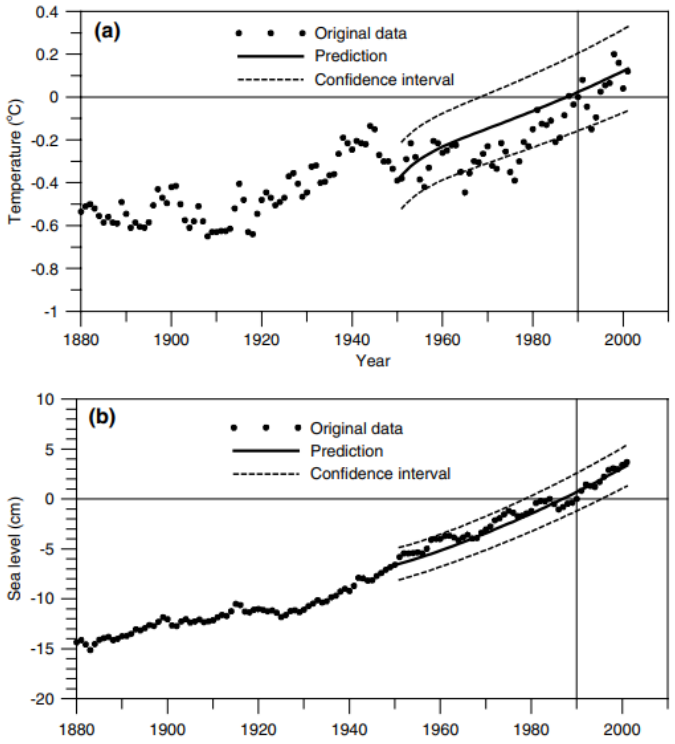

Fig. 2. Temperature and sea level from 1950 to 2001 predicted by the dynamic system model by using data set (1880–1950) and 90% confidence interval band

Figure 20: Above is the graphs of Aral, Guan, and Chang's data from 1880-2001.

In addition, I created graphs with the confidence intervals. My confidence intervals took the shape of the graph and contain all the data points.

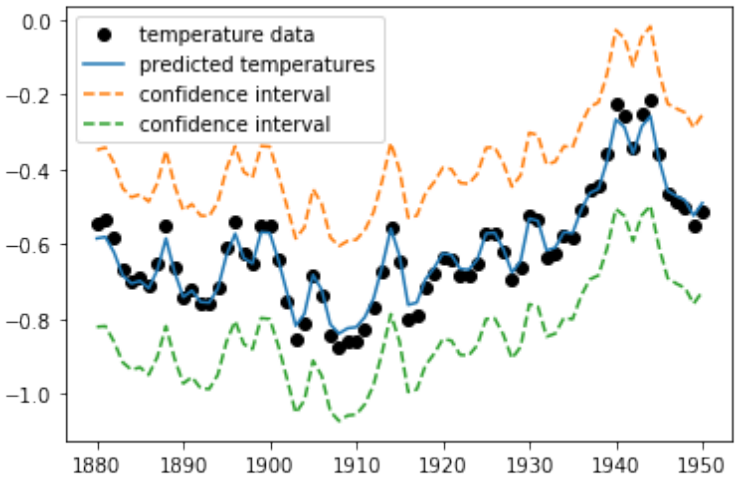

Figure 21: Above is a graph showing the predicted values, actual values, and the confidence intervals for the temperature data for the years 1880-1950

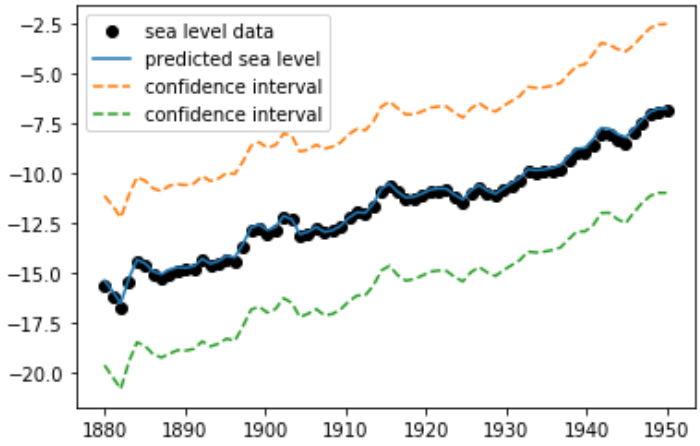

Figure 22: Above is a graph showing the predicted values, actual values, and the confidence intervals for the sea level data for the years 1880-1950

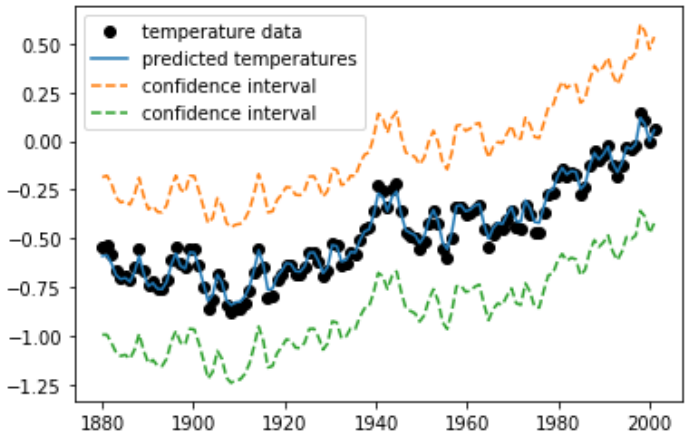

Figure 23: Above is a graph showing the predicted values, actual values, and the confidence intervals for the temperature data for the years 1880-2001

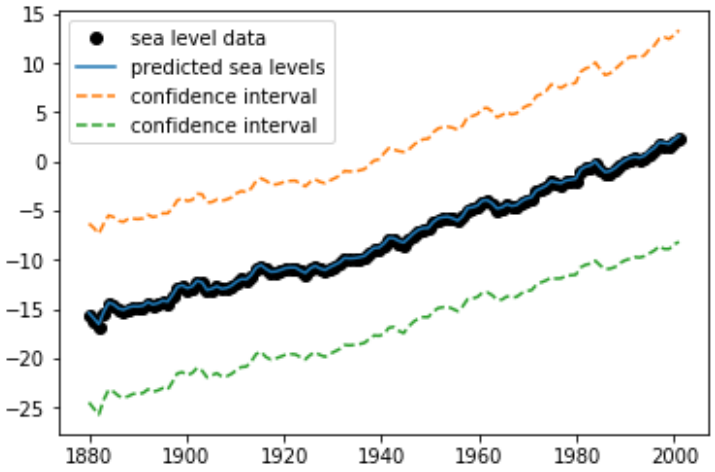

Figure 24: Above is a graph showing the predicted values, actual values, and the confidence intervals for the sea level data for the years 1880-2001

With the accuracy of these predictions in mind, I estimated the temperature and sea level values for year 2100. I found a temperature increase of $1.4663°C$ and a sea level change of 0.35523 m. Aral, Guan, and Chang predicted a $1.33°C$ increase in temperature and a .42377mm increase in sea level[1]. I compared these values to the ones given in the IPCC Third Assesement Report, the same reference used by Aral, Guan, and Chang [1] and Rahmstorf [2]. The IPCC predicted a temperature change from 1990 (about when our data hits 0) to 2100 of $1.4°C$ to $5.8°C$, so my prediction falls at the lower end of that range[4]. This report also predicts a rise in sea level ranging from 0.09 m to 0.88 m, so my predicted value falls within that range as well[4].

Aral, Guan, and Chang's sea level value was within range just fine, but their temperature value was slightly below it. They calculated the values of their confidence interview around the prediction. For their temperature, they found the interval: $[1.12, 1.54]°C$[1]. Their upper values were then within the IPCC range. I calculated the confidence interval around my own predicted values and found $[0.749, 2.183]°C$. My confidence interval was much wider than theirs, but still fell at the much lower end of the IPCC range.

For the interval around the predicted sea level in 2100, Aral, Guan, and Chang found [399.66, 447.89]mm or [.39966, .44780]m increase[1]. I found a [.19364, .516817]m increase. Both of these interval fall within the bounds given by the IPCC.

This confirmed what my correlation coefficients and graphs showed. While their values formed fairly accurate predictions, my values got closer to the actual system and the reliable estimates given by the IPCC.

7 Conclusions

The system of differential equations proposed here can be used to find decently accurate predictions of temperature change and sea level rise. Aral, Guan, and Chang [1] had decent

accuracy with their linear prediction. However, by taking the shape of the data, my predictions had much stronger correlation coefficients. This difference in shape accounts for the discrepancies between our Φ values and correlation coefficients.

The correlation coefficients I found were extremely close to 1, meaning my predictions almost perfectly modeled the data. If the temperature and sea level continue to change in the way they have in the past, this model can be used to accurately predict future temperature and sea level values. Since my predicted values fall within the ranges provided by the IPCC[4], it is likely that they are at least close to what we can expect.

Of course, it is difficult to say that the past is an effective way to model the future. Looking at the graphs of the data and predictions, it can be seen that the rate at which these values change can vary dramatically. Sometimes there is a dramatic jump and sometimes it seems to increase more slowly. Will the future data experience a lot of extreme jumps, or will it increase at a more consistent rate? It is hard to say. Either way, it is clear that these values are increasing and are likely to continue increasing, which is bad news for the environment.

One factor that makes temperature and sea level rise so hard to model is the complicated way different aspects of the environment change as a result. A major advantage of this model is that it formed accurate predictions without taking the complex physical model into consideration. The downside of such a simple model is that there is no guarantee it will maintain its accuracy if the physical system causes the rate of growth to change. Currently, the system of differential equations proposed by Aral, Guan, and Chang[1] is a simple, effective, and accurate model of the climate change.

While my predicted values were extremely close to the actual data, my long term predictions were pretty close to the lower bound of the predictions by the IPCC[4]. That begs the question: what can be done to increase the accuracy of this model any further?

First, looking at the graphs of the data, it can be seen that there is quite a bit of oscillation in the data despite the two year moving average. Rahmstorf's five year moving average smoothed the data out to the point of losing precision and accuracy[1], but perhaps two year averaging doesn't smooth out the oscillations enough. To test this, I would try these calculations using a three-year moving average. This wouldn't be too difficult since the moving average function in my program is generalized and can be used with any number of years (n). I would just have to verify that my arrays contain the correct years and perhaps adjust the point where they cut the arrays. The actual math would remain the same, so most of my existing code would still work just fine.

In addition, I would like to solve the continuous system in addition to the discrete system. Recall that the original differential equations were given by:

$$\frac{dX}{dt} = \mathbf{A}X(t) + \mathbf{C} \text{ where } X(t) = \begin{bmatrix} T(t) \\ \\ H(t) \end{bmatrix}, \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ \\ a_{21} & a_{22} \end{bmatrix}, \mathbf{C} = \begin{Bmatrix} c_1 \\ \\ c_2 \end{Bmatrix}$$

We converted this into a discrete system by increasing by one year intervals. We then solved the discrete system. It is possible that solving the continuous system would increase out accuracy even further.

Despite these potential future improvements, this model still fell within the ranges provided by the IPCC, making them fairly accurate and reliable. This accuracy was supported

further by the correlation between the data and the prediction, and was visualized on several graphs. All in all, it was shown that a simple system of differential equations can serve as an accurate model of the rise of temperatures and sea levels as a result of climate change.

8 The Bibliography

References

[1] Mustafa M. Aral; Jiabao Guan; and Biao Chang. Dynamic system model to predict global sea-level rise and temperature change. *Journal of Hydrologic Engineering*, 17(2), 2012.

[2] Stefan Rahmstorf. A semi-empirical approach to projecting future sea-level rise. *Science*, 315, 2007.

[3] John A. Church; Neil J. White. Sea-level rise from the late 19th to the early 21st century. *Surv Geophys*, 32, 2011.

[4] IPCC. *Climate Change 2001: Synthesis Report*. Cambridge University Press, Cambridge, United Kingdom, and New York, NY, USA, 2001.

[5] A.; Woodworth P.L.; Rickards L.J.; Tamisiea M.E.; Bradshaw E.; Foden P.R.; Gordon K.M.; Jevrejeva S.; Pugh J. Holgate, S.J.; Matthews. New data systems and products at the permanent service for mean sea level. *Journal of Coastal Research*, 29(3):493–504, 2013.

[6] Permanent Service for Mean Sea Level (PSMSL). Tide gauge data, 2020. Retrieved 24 Feb 2020 from http://www.psmsl.org/data/obtaining/.

[7] NOAA National Centers for Environmental Information. Climate at a glance: Global time series. Retrieved 24 Feb 2020 from https://www.ncdc.noaa.gov/cag/.

[8] Charles Zaiontz. Multiple regression using matrices. Retrieved 27 Feb 2020 from http://www.real-statistics.com/multiple-regression/multiple-regression-analysis/multiple-regression-using-matrices/.