CSC110 Project Report: The Effect of Climate Change on Flooding Susceptibility in Coastal Regions (Vancouver Area)

Lorena Buciu, Kevin Yang, Ricky Yi, Rafee Rahman

Monday, December 14, 2020

Introduction

How has climate change affected the sea levels in the Vancouver area over the years? Based on this, which areas project to be susceptible to flooding 280 years in the future?

Climate change has a number of devastating impacts on the environment. One of the issues that arise as a result of climate change is the global acceleration of sea-level rise. Sea-level rise (SLR), measured in metres, is frequently observed by researchers and has been known to be increasing for years. Changes in global temperature result in glaciers melting as well as the warming of ocean water, causing it to become less dense and expand, overall resulting in sea-level rise (Hausfather, 2019). Coastal regions are most heavily impacted by these changes since an increased sea-level will begin to take up land and potentially endanger those living nearby and increase the risk of floods and damage to agricultural land. Sea-level rise also has implications for inland areas since the displacement of civilians would mean an increased impact on even non-coastal regions.

A study from Nature Communications finds that approximately 300 million people live in areas at risk of flooding (Kulp & Strauss, 2019). Countries including China, Bangladesh, and Vietnam are already facing heavy floods due to sea-level rise. Based on sea-level projections for 2050, land in these areas will fall below the elevation threshold for annual coastal floods (Jennings, 2019). Therefore, it is important to predict when in the future the sea-levels in those areas will pose a risk to those people if we do not slow down the effects of climate change.

The goal of our research is to predict and visualize the effects of sea-level changes due to climate change in coastal regions, focusing on the greater Vancouver area. We have decided to analyze areas along the coastline to determine which are most at risk of flooding in the future by comparing the estimated future sea-level to the elevation of coastal regions. According to the 2016 Census, Vancouver has one of the highest population densities in Canada, so climate change and, by extension, SLR puts the many people living there at risk. Researchers are also worried that sea-level rise in this region may be 3 times worse than expected (Kulp & Strauss, 2019).

Dataset Description

A dataset that we have used is from the NOAA / NESDIS / Laboratory for Satellite Altimetry Website. It is a CSV file that contains the sea level change in the north pacific ocean, given in millimetres. This file is named pacificocean_sea_level.csv.

The data ranges from the year 1992 to 2020 (there are multiple satellite captures for each year) and the values were obtained by the TOPEX/Poseidon, Jason-1, Jason-2, and Jason-3 satellites. The dataset contains 4 columns, one for the year, and 3 for each satellite and the value that it recorded at a point during the year. An example of data that is in this dataset is: A capture during the year 2020 by the Jason-3 satellite is a change of 140.61mm.

We have cleaned this CSV dataset by calculating the mean sea level change for each year instead of having multiple captures for one year, putting all the satellite data together, and using this data we can predict the change in future years. We created a new CSV file called data_predictions.csv that contains two columns for the year and the corresponding sea level, including the values we predicted with our Theil-Sen regression model.

Another dataset we have used is Vancouver surface elevation data from the Topographic Information section of the Government of Canada website. It is an ASC file (but we can easily read it as a .txt file) that contains digital surface model (DSM) data based on a selected area of the Vancouver region we are analyzing. The value 0 in this dataset represents the elevation at sea level. We can compare this elevation data to the sea level data so that we can determine flooding susceptibility. This data was interpreted into a grid of latitude and longitude values through calculations which allowed us to plot the data points on a plotly mapbox.

After interpreting this DSM data, the program will create a new CSV file called below_sea_level.csv which contains all of the points which are below the given sea level for a certain year. This file contains 3 columns, latitude, longitude, and elevation.

The program also creates a dataset named sarimax_model_data.csv, necessary for displaying the SARIMAX prediction model. This dataset has two columns, year (contains dates as datetime.datetime objects) and the `mean_sea_level` predicted by that model.

Computational Plan

Part 1: Data Cleaning

Our program is mainly composed of two parts, data processing and then extrapolation. With the datasets being in CSV and ASC format, we needed to read them while at the same time removing any unnecessary data not needed for calculations. Most of these operations are done in the `data_cleaning.py` file. In this file we use the `csv` module to access the contents of the files then use a combination of for loops and module features from pandas and numpy to mutate our data so that it is compatible with our calculation functions. This is perhaps best demonstrated by the `read_csv_data()` function in the `data_cleaning.py` file. This function aims to transform our pacificocean_sea_level.csv file into a dictionary `Dict[str, List[Tuple[str, float]]]` that will work with the rest of the program. To do this, it filters out empty values from the base data set and appends usable data points into the return dictionary.

A major computation performed by our program to transform this new data is seen in the `group_means()` function, also found in the `data_cleaning.py` file. This function iterates through this newly created dictionary, extracting the years from each satellite and computing the means, resulting in a formatted dictionary by mutation done by the helper function `remove_dupes()` with the years as the keys mapping to the mean sea level for that year. This condenses our original pacificocean_sea_level.csv into a simpler, summarized dataset. The program will write our transformed data into a new CSV file called predicted_data.csv for pandas compatibility later on which is done through the usage of the `csv` module.

The most important data processing is seen in the `canada_dsm.py` file. This file interprets the elevation_data.asc file into latitude and longitude points which will be plotted into our map visualization. This file achieves this mainly through the `assign_coords()` function. The original elevation_data.asc file contains values for the number of rows, columns, latitude of the bottommost coordinate, longitude of the leftmost coordinate, and the cell size. Using these initial values, we can use a nested for loop and some calculations to interpret each elevation into a latitude and longitude coordinate, based on its position in an ascii grid.

Part 2: Extrapolation

In the extrapolation portion of our program, we used a combination of two computational models: Theil-Sen Regression and the SARIMAX Model. The Theil-Sen model first determines the inputs in the final display which is then graphically represented by the Sarimax Model.

Theil-Sen Model:

Theil-Sen regression is similar to linear regression in that it can predict future values over time after determining a line of best fit, but the methodology of finding the line is different. Over the years there have been multiple variations of this model developed and the method we implemented will be described. In this method, the slope between each set of data points (x, y) is calculated and the median of the calculated slopes is taken as the slope of the line which can then be used for a y-intercept calculation. In the file theilsen.py, these operations are done through index based for loops seen in the `theil_sen_linear_model` function, to calculate each individual slope which is then appended

to a list of all slopes. The statistics library's median function finds the median of the list which is used to calculate the y-intercept, returning a linear equation.

Sarima Model:

SARIMA, also known as Seasonal Auto Regressive Integrated Moving Average, is a type of time series model that is used for analyzing and forecasting time series data. The sarima model is similar to 'AR', 'MA', and 'ARIMA' models, and uses the same techniques with some variation. Autoregression means that the model uses a dependent relationship between an observation and lagged observations. Lagged observations are a way that time series models are forecasted. Given a time t, the approach is to predict the value at the next time, t + 1, while given the previous time t - 1. The 'I', integrated, is an approach that takes the difference of observations to make the time series data stationary. The 'MA', moving average, depends on the observations and the residual errors of a moving average model. In our data visualization, we saw that the sea level anomaly looked seasonal, meaning that at certain periods of time, there is an increase or decrease in sea level, hence why we used SARIMA. The X at the end of 'SARIMAX' stands for exogenous factors, but we ignored that in our model.

In ARIMA modelling, we use three main arguments. These are known as p, q and d. The p is the number of lags, the d is known as the degree of differencing, and the q is the window/timeframe we are taking the averages from. In statistical practice of ARIMA modelling, we determine whether or not the data is stationary (i.e moving averages are the same). In python, we did this using the Dickey Fuller Test using a built-in function from the statsmodels library called adfuller, which gave us various statistics including the p-value. We also did an ARIMA model simulation to determine the p, q, and d values, using the function auto_arima from the pmdarima library. In regular statistical practice, to do this test on our own we would look at the autocorrelation, and partial autocorrelation of the data but that was a bit more complex.

To build the SARIMA model, some data extrapolation was done. First, we converted the decimal years to date-time values and wrote them into a new csv file. We also combined the topex and jason satellite data into one. We then condensed the data using pandas, which only took the maximum sea level value for one month and also removed duplicate dates. This made the datetime frequency monthly so that the SARIMA, matplot, and auto arima functions could understand it. In some parts, we used matplotlib which helped visualize the techniques that were required for our prediction such as the residual, trend, seasonality and more. Some of these libraries were commented out of the project because it was not necessary for our final visualization.

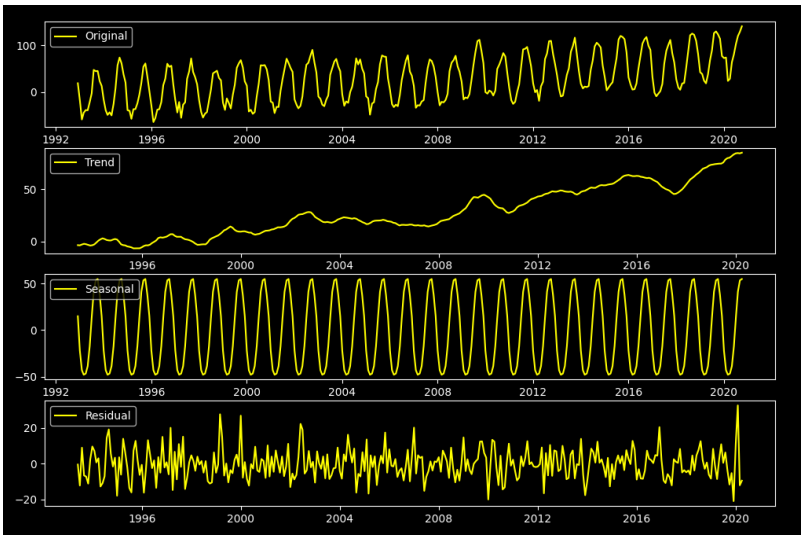

Figure 1: Matplotlib Visualization of Original Data with Residuals, Trends, and Seasonal lines

The auto arima function gave us the best p, q, and d values for our data that would then be used as arguments for the SARIMA model. With our data being seasonal, there is a fourth argument that we added to the SARIMA function, which is from the statsmodels library. This fourth argument is the length of the cycle/where the pattern occurs. We saw that the data was changing about every 12 months, so we used 12 months as our seasonal cycle. Using the data produced from the SARIMA function, we used training data, and testing data to help our forecast

become more accurate. For the training data, we used about 80% of our actual data to train our model. The rest were used in our testing (actual values). To predict the forecast, we used other functions from the statsmodels library including the predict function, which uses mathematical formulas to predict new values. To predict 30 years, we used the start value for the prediction as the end of our dataframe (2020), and the end value of our prediction was the length of our dataframe (2020) added to 30 * 12, which accounts for 30 years. We then plotted the train data, test data, and predicted data to give us a better visualization.

A lot of the mathematical analyses that were used in the functions we imported from the statsmodels library were a bit beyond our scope, which is why we could not implement them ourselves and relied on the libraries. One example is in the SARIMAX function. This function uses the formulas below (from statsmodels in references).

The SARIMA model is specified $(p, d, q) \times (P, D, Q)_s$.

$$\phi_p(L)\tilde{\phi}_P(L^s)\Delta^d\Delta_s^D y_t = A(t) + \theta_q(L)\tilde{\theta}_Q(L^s)\zeta_t$$

In terms of a univariate structural model, this can be represented as

$$y_t = u_t + \eta_t$$
$$\phi_p(L)\tilde{\phi}_P(L^s)\Delta^d\Delta_s^D u_t = A(t) + \theta_q(L)\tilde{\theta}_Q(L^s)\zeta_t$$

where η_t is only applicable in the case of measurement error (although it is also used in the case of a pure regression model, i.e. if p=q=0).

In terms of this model, regression with SARIMA errors can be represented easily as

$$y_t = \beta_t x_t + u_t$$
$$\phi_p(L)\tilde{\phi}_P(L^s)\Delta^d\Delta_s^D u_t = A(t) + \theta_q(L)\tilde{\theta}_Q(L^s)\zeta_t$$

this model is the one used when exogenous regressors are provided.

Note that the reduced form lag polynomials will be written as:

$$\Phi(L) \equiv \phi_p(L)\tilde{\phi}_P(L^s)$$
$$\Theta(L) \equiv \theta_q(L)\tilde{\theta}_Q(L^s)$$

Figure 2: Formulas From Statsmodels Reference

Part 3: Reporting Results

As described in the introduction, the main form of reporting takes the form of an interactive map of the Vancouver area. Based on the extrapolated values we computed, the user of our program is able to move a slider to select a year and the extrapolated results will be displayed on a map. There is also another slider which allows the user to select a sea level and see which areas are going to be affected by that selected sea level. The dash framework allowed us to establish an interactive front-end visualization, using HTML and dash core components to program a dynamic slider.

To determine which points are below the given sea level, we have the `check_elevation()` function in the `canada_dsm.py` file. This function creates a list of lists containing the latitude, longitude, and elevation − sea_level which are below the given sea level, by checking if the elevation is less than the sea level with a while loop. These values are then written to a CSV file called below_sea_level.csv, which is used as the dataset for our `display_map()` function in `models.py`. The map is displayed using plotly's mapbox visualization object. First, we can use pandas to convert this new CSV file into a pandas dataframe with `pd.read_csv('below_sea_level.csv')` which makes it easier to use with plotly's mapbox. By importing plotly.express we used `px.scatter_mapbox()` and passed in the necessary parameters to display these coordinates that are below the sea level.

To create our graphical visualizations, we used plotly's `go.scatter()`, and `add_trace()` to display the points necessary for our regular model and predictive models.

Program Instructions

Before the program can be run, there are 2 necessary datasets to download:

- elevation_data.asc (dataset containing the elevation for the vancouver area on a grid) from CLICK TO DOWNLOAD

    1. Place it in the project directory (no subfolders)

- Dataset containing satellite sea level rise measures from CLICK TO DOWNLOAD
    1. After downloading please name this file to “pacificocean_sea_level.csv”
    2. Again, place it in the project directory (no subfolders)

After downloading the relevant datasets, please install the necessary python modules from requirements.txt (**Important**: requirement.txt installs numpy 1.19.3 because the latest version raises an error on some devices. If you are getting a windows runtime error when using numpy please make sure the correct version is installed). Then, run the main.py file and click on the link displayed in the console output.

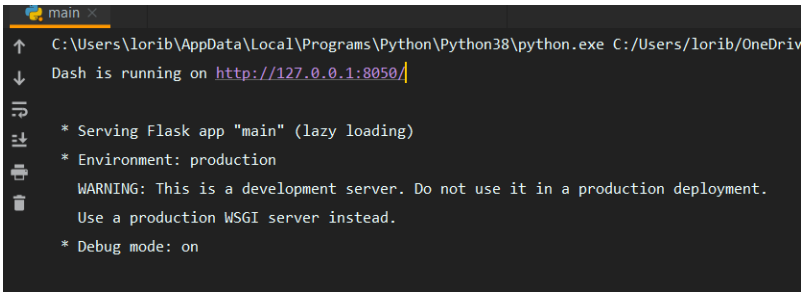

Figure 3: Console Output

This will take you to the Dash webpage we created where you can view and interact with the visualizations. Our main interaction is being able to adjust the 2 sliders underneath the map we created to see how the data points change at a specific year, or with a predetermined sea level input.

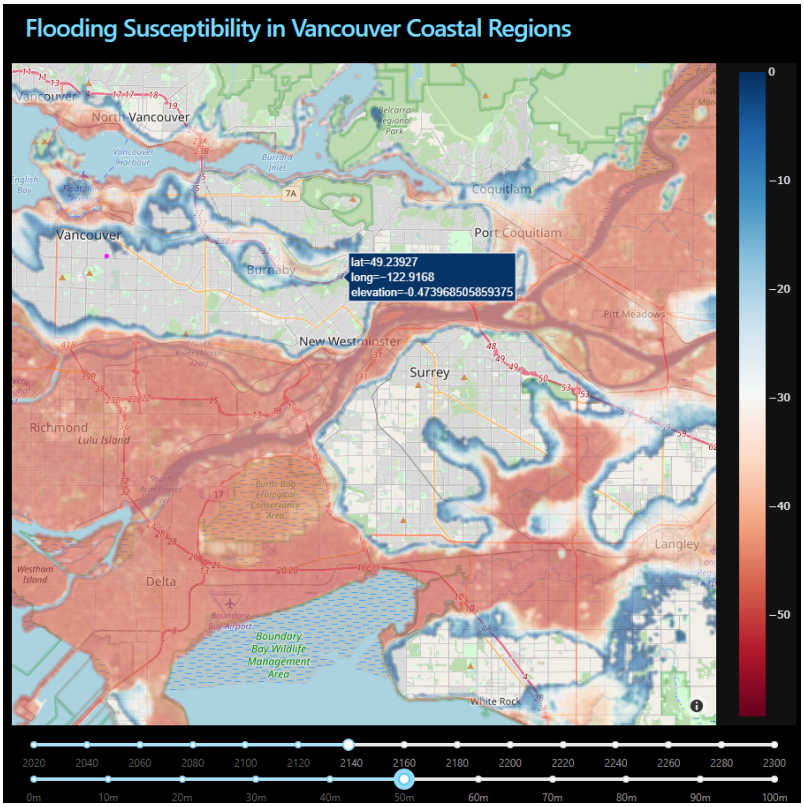

Figure 4: Front-end Output

Project Changes

Based on the feedback received from our proposal, we have made some minor changes to our project plan to streamline the process. For example we realized the vagueness of our original research question and thus adjusted it to be more focused by specifying a certain time period to analyze. We also realized the repetitiveness in only performing a linear regression calculation on our data seeing as we have done the same thing on Assignment 1. After this was brought to our attention, we looked into other methods to model our calculated data and decided on a Theil-Sen regression and SARIMAX prediction model. The Theil-Sen regression is what determines our predictions but the SARIMAX model provides a nice moving average prediction model for visualization. We also realized the complexity of having a dropdown menu and other forms of interactive output, due to our limited expertise in front-end development. We stuck to having 2 sliders, one for the year and one for the sea level and a color bar which determines flood severity based on elevation.

Discussion

Our computational exploration does a sufficient job in answering which coastal areas in Vancouver are the most susceptible for flooding 280 years into the future. This is displayed by our mapbox visualization and data interpretation of the annual mean sea level changes. If we move the slider all the way to 2300, we can see which areas are going to be the most impacted, due to the greater frequency of data points which are below sea level. By analyzing this map we see that the most heavily impacted areas are the coastlines around Richmond, Delta, and Surrey, due to their lower elevations.

Additionally, our graphical models display the changes in sea level over the years. We can see that there is a positively increasing trend, which means that due to climate change, sea levels are indeed beginning to rise. Our predictive model shows how high the changes will be in the future, urging us as Earth's inhabitants to take action in order to disrupt this trend.

One of the main limitations was with finding an appropriate dataset for the sea level changes. It was very hard to find a dataset in a format that would be easily interpreted by our program, hence the many data filtrations/transformations we needed to apply to this dataset. It was also hard to find a dataset that dates back to an earlier year in the past, as in earlier than 1992. This affects our prediction model because the model was trained over a shorter period of time, so the projected trend may be less representative of reality.

Another limitation was with the plotly library, some of the functions were difficult to use because some parameters for our figures would not seem to work as desired or described by their official documentation, so we needed to research workarounds for these issues by using other functions, which consumed a lot of time of the project. However, researching documentations for modules and seeking solutions for errors online is very typical in a software engineering environment, so it was good for us to get this experience. We used dash to display our plotly figures, the main challenge from this was dealing with the front end components. Since we are not experienced with front end development it was quite difficult to work with the dash framework, which results in an interactive display that is more basic than we intended. However, we still made an effort to research how to work with this framework and were able to implement a slider which provides an interactive experience with our figures.

The main obstacle we faced in designing this program was implementing the prediction model algorithm. This was difficult because it was hard to find a model that was appropriate for our situation. We also wanted to implement this algorithm without as much help from an external library as possible, to increase the complexity of our calculations.

The next steps for our exploration would be to incorporate temperature data as well for a stronger analysis. If we displayed figures with temperature against sea level rise we would be able to see the relationship between climate change and sea level rise directly, rather than assuming it as background knowledge. If we used the relationship between global temperature change and sea level change to train our prediction model rather than sea level change and time, maybe our model would have been more accurate.

In conclusion, our computational exploration allows us to see the coastal areas in the Vancouver region most susceptible to flooding up to 280 years into the future, due to our trained prediction model computations, data interpretations, and geographical visualizations, despite the challenges and limitations faced.

References

Canada, N. (2020, October 16). Government of Canada. Retrieved November 06, 2020, from https://www.nrcan.gc.ca/topographic-information/10785

Bhattiprolu, Sreenivas. "163 - An Introduction to Time Series Forecasting - Part 3 Using ARIMA in Python." YouTube, YouTube, 1 Oct. 2020, www.youtube.com/watch?v=TSfdvQ0ARwI

Brownlee, Jason. How to Create an ARIMA Model for Time Series Forecasting in Python, 9 Jan. 2017, http://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/

Dash Documentation User Guide. (n.d.). Retrieved December 13, 2020, from https://dash.plotly.com/

Explainer: How climate change is accelerating sea level rise. (2019, September 30). Retrieved November 05, 2020, from https://www.carbonbrief.org/explainer-how-climate-change-is-accelerating-sea-level-rise

Kulp, S., Strauss, B. (2019, October 29). New elevation data triple estimates of global vulnerability to sea-level rise and coastal flooding. Retrieved November 05, 2020, from https://www.nature.com/articles/s41467-019-12808-z

Laboratory for Satellite Altimetry / Sea Level Rise. (n.d.). Retrieved November 06, 2020, from https://www.star.nesdis.noaa.gov/socd/lsa/SeaLevelRise/LSA_SLR_timeseries.php

Little, S. (2019, November 01). Sea level rise may be '3 times worse' than expected. Here's how it could impact Metro Vancouver. Retrieved November 05, 2020, from https://globalnews.ca/news/6102304/sea-level-rise-metro-vancouver/

(n.d.). Retrieved December 13, 2020, from https://numpy.org/doc/

Pandas. (n.d.). Retrieved December 13, 2020, from https://pandas.pydata.org/

Plotly Python Graphing Library. (n.d.). Retrieved November 05, 2020, from https://plotly.com/python/

Report: Flooded Future: Global vulnerability to sea level rise worse than previously understood. (2019, October 29). Retrieved November 06, 2020, from https://www.climatecentral.org/news/report-flooded-future-global-vulnerability-to-sea-level-rise-worse-than-previously-understood

Rising Sea Levels Threaten to Flood Southern Vietnam by 2100. (n.d.). Retrieved December 13, 2020, from https://www.voanews.com/east-asia-pacific/rising-sea-levels-threaten-flood-southern-vietnam-2100

SciPy. (n.d.). Retrieved December 13, 2020, from https://docs.scipy.org/doc/scipy/reference/

Statsmodels.tsa.statespace.sarimax.SARIMAX Documentation. (n.d.). Retrieved December 14, 2020, from https://www.statsmodels.org/stable/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html

Theil, Henri. "A Rank-Invariant Method of Linear and Polynomial Regression Analysis" Retrieved December 9 2020, https://link.springer.com/chapter/10.1007/978-94-011-2546-8$_2$0

Vancouver has the highest population density in Canada: Census. (2017, December 28). Retrieved November 05, 2020, from https://biv.com/article/2017/02/vancouver-has-highest-population-density-canada-ce