CORAL REEF TRENDS IN THE U.S. VIRGIN ISLANDS

J. BENNETT, M. DAVIS, Z. MEYER, M. SHIELDS, L. TOLMAN

Abstract. In this paper, we attempt to better understand how coral reefs are changing over time. We also seek to better understand how other aspects, specifically global warming and its impact on sea surface temperature, may affect the development and prosperity of coral reefs. Our goal is to provide a predictive model that enables us to accurately predict coral reef coverage based on specific attributes of the reef and its environment. We will use a few different types of models each of which has its own underlying assumptions. Finally, we will address other research that has been conducted in this area and present avenues for further exploration.

1. Background/Motivation

The world is currently undergoing some severe changes and trends in its weather. It has been established by numerous studies and governments that the Earth is warming[7]. Though we are not focused on fully showcasing or verifying these trends, we do recognize that these trends are accompanied by significant side effects all around the world. One area where these trends can have very dangerous effects is on our coral reefs.

Sources showcase that coral reefs are one of the most important ecosystems in the world[6]. Coral reefs provide protection and barrier from storms for coastal lands. Coral reefs are also a huge source of food and new medicines. It is estimated that over half a billion people (over 6% of the human population) rely on coral reefs for income, food, and protection[6].

Coral reefs are incredibly important to numerous species around the globe, including humanity. We seek to better understand how coral reefs are changing over time, given current trends. We hope that this understanding will encourage prevention of future damage and lead us to preserve such a vital part of the ecosystem. Our goal is to explore overall coral coverage at sites in the U.S. Virgin Islands, based off trends over time, bleaching patterns and sea surface temperature (SST). Note that coral reef bleaching is when water temperature is too high for coral and causes it to turn white. When coral is bleached, it is under stress and more likely to die.

Previous research has been conducted in this specific branch of oceanography, using various techniques to predict global and more localized coral

reef coverage patterns through forecasting models.

In 2011, Simon Donner, a professor at the University of British Columbia, published a study evaluating the use of historical sea surface temperature data on coral reef bleaching predictions[2]. He evaluated 3 different methods that used varying periods of how long coral reefs were subject to sea surface temperatures above the local climatological maximum.

Another study on the subject was done in 2014. Researchers, Angang Li and Matthew Reidenbach developed a model derived from the three-dimensional Navier-Stokes equation to simulate local water flow and introduced a heat flux equation to account for local temperature variations[1]. They had the goal of developing a model that predicted sea surface temperatures while being robust to local changes in water flow and water level.

These two research papers display that sea surface temperatures has an impact on coral health and using localized coral data would lead to positive results towards predicting overall coral health and mass coral bleaching events.

We intend to use localized temperature[3] and coral data[4] from the U.S. Virgin Islands, but with the intent of predicting the overall trend of coral health, rather than mass coral bleaching events. Our predictions, if seen as successful, could be used to predict long-term coral health trends in other locations globally based on localized information.

2. Data

Our data set describes the coral health of two locations in the US Virgin Islands, Tektite and Yawzi[4]. The coral coverage (percent of surface area covered by coral) of macro algae, CTB (coral tissue bleaching) and overall coverage are used to describe the health of the coral in these coral reefs. Our data separates each of these locations into quadrants (smaller rectangular regions that together make up the whole location) where coral coverage is measured yearly from 1987 to 2021. This gives us a dataset with measurements for 34 years. There are a few cases where the coral coverage was measured several times a year or there is a missing year. To handle this we averaged all the measurements made each year for an average yearly coral coverage. Where there is missing data we interpolated to fill in the missing years. For the purposes of our models we did feature engineering to calculate the average of all the quadrants to get one measurement for each location each year, namely the average percent of coral coverage in a site. In our models we set the last 5 years aside to serve as a test set. We note that the test sets will not be completely independent of the years before due to the dependence on previous years for coral health. Regardless of this, our goal

is to predict the health of coral in these locations for future years. Thus it makes the most sense to split our data in this manner, rather than by location.

We suspect that sea water temperature may be correlated with the heath of coral reefs. Because of this we also used an additional data set that provides sea water temperature data from the U.S. Virgin Islands every day from 1987 to 2017[3]. With this data, we are able to get an average water temperature measurement per year provided. We can use these measurements to potentially yield more accurate predictions for coral coverage.

3. Methods

Now we will address several methods that we used to attempt to understand and predict change in coral growth overtime. In order to do this, we modified some of our data as described above. We will utilize four main models in our approach. We will use standard time series decomposition, ARIMA, Kalman filter, and VARMAX. Each of these methods will help us get a different understanding of our data and help us predict future time series data which is the goal of our paper.

3.1. **Time Series Decomposition.** Time series decomposition is an excellent way to see underlying seasonality or trends in time series data. Certainly, temperature is seasonal and a case can be made that coral reef coverage could be as well. This model will help us to understand seasonality and true long term trends for both sea surface temperature and coral coverage.

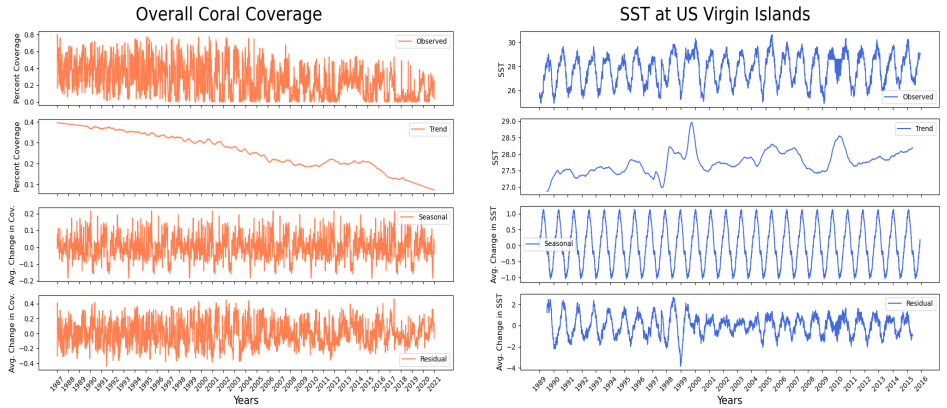
3.2. **ARIMA.** ARIMA is a model that will allow us to predict coral coverage into the future. Mainly, we are using this model to get a better feel for how our coral reefs are changing over time and what future predictions look like. We will utilize a grid search to optimize over p and q as input parameters to our model.

3.3. **Kalman Filtering.** Kalman filtering is another model that we will employ. This model allows for consideration of unmeasured parameters. We are interested in how coral reefs are decreasing overtime, but we do not know nor could we account for all elements that effect this. Thus, we will use Kalman filtering with a grid search to optimize our matrix and covariance parameters.

3.4. **VARMAX.** VARMAX contains similar components to an ARIMA model but it allows us to incorporate several independent and dependent features. Because of this we are able to use the relationship between features to help gain better predictions for each feature. In our data we believe that different types of coverage and sea temperature may have an effect on coral reef health. VARMAX will allow us to better predict future trends of coral reefs using these relationships.

4. Results and Analysis

4.1. **Time Series Decomposition.** In our attempt to better understand the overall trends in sea surface temperature and percentage of coral coverage, we performed a classical decomposition on the local SST (sea surface temperature), overall coral coverage, and the localized coral coverage at Yawzi and Tektite.



(A) Overall Coral (B) Sea Surface Temp.

(C) Yawzi Coral (D) Tektite Coral

Figure 1. Classical Decomposition of SST and Coral Data.

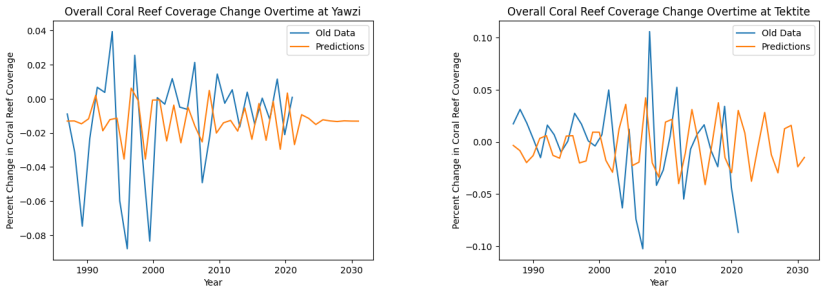
A period of 365 days was selected to use in the classical decomposition. This period was chosen to determine yearly trends and have data that could provide possible correlation over equivalent time periods. Each of the subsets of our data was decomposed under the assumption there was an additive relationship in the moving average.

These decompositions can be seen in Figure 1. The observed and trend subplots in each figure display total coral coverage while the seasonality and residual components show the average change in coral coverage. The overall coral trend shows a decreasing linear trend. We chose to then separate the data further into the specific coral sites. The Yawzi site has a decreasing trend for all time, but the Tektite site only begins to decrease after 2001, beforehand showing trends of stable coral reef ecosystems. There does not appear to be an established seasonal component to coral health. The collective trends display an overall pattern of decreasing coral coverage in the time interval (1987 to 2021) of the collected data.

The SST decomposition shows a sinusoidal pattern of heating and cooling over a similar time period, which correlates with the natural temperature patterns of oceans[5]. The moving-average trend displays a non-linear increase in temperature that has been gradually increasing since the beginning of the data collection.

We now have better justification to attempt to utilize SST data to predict coral reef coverage because of the significant decrease in coral health and increase in temperature over the same periods. We attempt to explore this relationship further in later models and attempt to use this analysis to create accurate forecasting models.

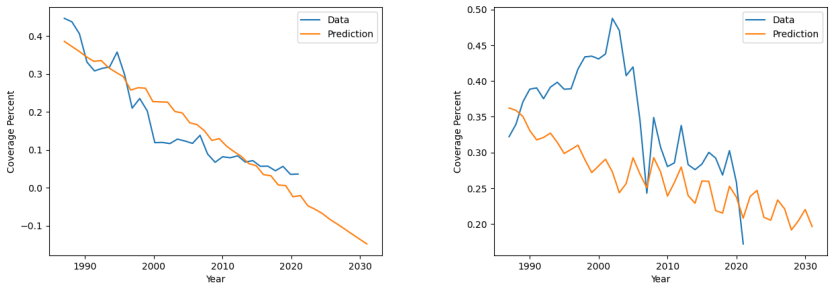
4.2. **ARIMA.** We used an ARIMA model to be able to predict the average change from year to year in each of our two coral reefs. In order to do this we use a grid search over two of the parameters for ARIMA models, p and q. Figures 2A and 2B showcase the change overtime and the model fit as well as its prediction into the next 10 years. We train on the first 30 years of data, then compare our first 5 years of predictions to a test set containing 5 years of data that are on the equivalent time period.



(A) Yawzi (B) Tektite

FIGURE 2. Change in coral coverage over time at the Tektite and Yawzi Sites

Figures 2A and 2B demonstrate that the Tektite site appears to be fairly centered around zero while the Yawzi site seems to be centered around -.01. In order to have more insight into what the coral trends at each site we reconstructed the year over year change to look at the percent coverage in each location. We also did this with the results of our ARIMA model to look at our predictions overtime. These percentages are shown in Figures 3A and 3B



(A) Yawzi (B) Tektite

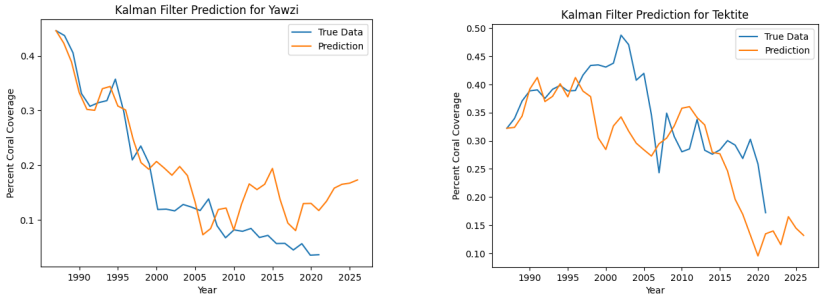
FIGURE 3. Predicted vs actual overall coral coverage as a percentage

The error metrics for our ARIMA models are displayed in Table 1. It's important to note that the R-squared values are extremely low, implying our model does not explain well the variance within our coral data. Between the data and our model, percentage coral coverage at the Tektite site appears to be decreasing slightly. However, with limited time series data, it limits the ability to visualize present trends and create accurate forecasts. On the other hand, percentage coral coverage at the Yawzi site is decreasing at a noticeably faster rate. We will further explore these potential trends, look at different model accuracies, and incorporate other data to further our understanding.

	ARIMA Tektite	**ARIMA Yawzi**
MAE	0.028	0.013
MSE	0.0011	0.0002
RMSE	0.034	0.016
AIC	-86.68	-99.488
R2	0.066	-1.123

TABLE 1. ARIMA model error predictions for coral coverage predictions.

4.3. **Kalman Filtering.** The next model we used to forecast data was a Kalman Filter. This method is particularly useful in this context as it dynamically estimates the state of a system based on noisy measurements over time. Since there is no well defined law about how coral reef coverage is affected over time, we will assume the hidden state space is one-dimensional with no obvious real world application. Using expectation maximization, we found choices of transition matrices that would best fit our data. We calculated error metrics to test its performance. Similar to the previous model, we conducted a grid search to determine the optimal choice parameters for our model, including transition, observation, and covariance matrices. The best of these had a total mean squared error of around 0.01 on the training data. However, the model incorporates noisy data, and can sometimes achieve greatly inaccurate predictions despite optimal choices of parameters. The following figures are the results of conducting a grid search to find parameters that best fit the Kalman filter to our training data, which includes the years 1987 - 2016. The graphs below show the found parameters on our training set and test set, as well as a prediction for several more years.



(A) Yawzi (B) Tektite

FIGURE 4. Coral coverage over time at the Yawzi and Tektite Sites using a Kalman Filter

Table 2 provides a variety of metrics that determine how well the Kalman filter did in predicting the test years. Note the values are relatively promising, with the exception of the R-squared score.

	Kalman Tektite	**Kalman Yawzi**
MAE	0.113	0.064
MSE	0.015	0.0047
RMSE	0.123	0.068
R2	-6.165	-52.62

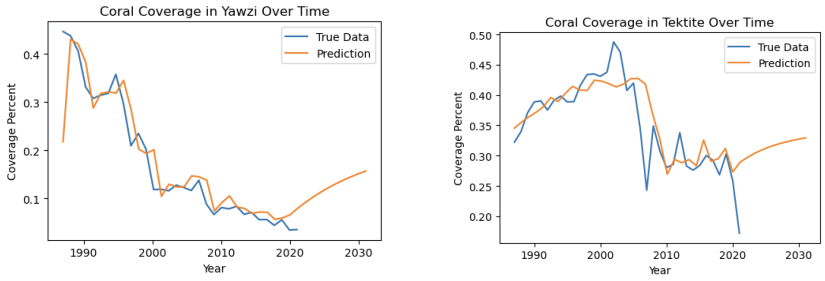
TABLE 2. Kalman model error predictions for coral coverage predictions.

At both sites, the models seem to predict a trend of decreasing coral reef coverage over time, aligning with our findings from the ARIMA model. Interestingly, this trend is much more prominent at the Tektite site as the years progress. The model seems to suggest an increase in coral coverage in coming years at Yawzi, which could provide some preliminary hope for these reefs. However, this is a very simple Kalman filter model that requires further development before conclusions can be made.

We attempted to incorporate sea temperature into this model. However, a grid search proved much more difficult in higher dimensional space and we were unsuccessful in using expectation maximization. Though we were unable to do so, we believe that incorporation of sea temperature into our state space could provide even more promising results. Thus, we suggest that others exploring this topic consider doing so.

4.4. **VARMAX.** We next forecasted our data using a Vector Autoregressive Moving-Average Processes with Exogenous Regressors (VARMAX) model. This is a forecasting model that allows us to consider both dependent and independent features of our data at the same time. VARMAX allows us to use the relationship between these features to better inform our predictions.

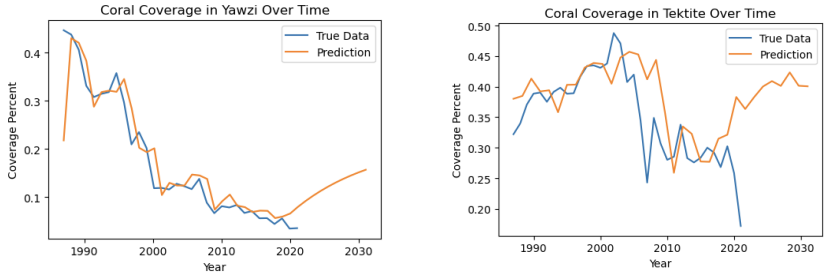
First we implemented our model with the macroalgae and coral coverage of the tektite location, as well as CTB (Coral Tissue Bleaching) as dependent features. We then predicted coral coverage for the Yawzi and Tektite locations and performed a similar grid search as done in the ARIMA model to find the optimal values of p and q for our model. We trained the model on the first 30 years of data and then predicted coverage for the next 10 years. We have the data for the first 5 years of our prediction allowing us to validate our results using various error metrics. We followed a similar process for the Yawzi location. The predictions for the Yawzi and Tektite site are seen in Figures 5A and 5B.



(A) Yawzi (B) Tektite

FIGURE 5. VARMAX predictions using macroalgae and coral coverage and coral tissue bleaching as features for the each location

We are interested to see if ocean temperature information will improve our predictions for coral coverage. We repeated a similar experiment to the one described above now including ocean temperature data as a feature. Our results are seen in Figures 6A and 6B.



(A) Yawzi (B) Tektite

FIGURE 6. VARMAX predictions using macroalgae and coral coverage, coral tissue bleaching, and ocean temperature as features

The error metrics for the multiple variations of our VARMAX model are described in Table 3. We note that due to the high variance in coral coverage data, the R-squared values are incredibly low, so the model will not explain this variance well while forecasting.

	Tektite	**Yawzi**	**Tektite (w/ temp)**	**Yawzi (w/ temp)**
MAE	0.046	0.038	0.107	0.077
MSE	0.004	0.001	0.015	0.009
RMSE	0.066	0.066	0.121	0.097
AIC	-217.823	-189.704	-176.942	-145.734
R2	-1.043	-8378.926	-7.877	-5.106

TABLE 3. VARMAX model error predictions for coral coverage predictions.

We see that both methods for our predictions gave similar results. In this model, temperature did not appear to significantly improve the accuracy of our predictions for coral coverage. We also suspect that both models are likely over fitting the data because we see that our prediction follows the data closely until where our training set ended and our testing set began. This suggests that our model is just learning the training set really well and not actually yielding trust worthy predictions.

5. Ethical Considerations

Though this does not appear to be an ethically challenging project on the surface, there are some aspects to consider. Mostly, it is important to be

mindful in the application of our results. We note that our results are neither robust nor accurate enough to be the sole reason for enacting or refusing to enact social and environmental change, though we hope they provide some inspiration for such possibilities. These results are best used in conjunction with similar research before action should be taken. If one uses these or similar results in making change, we implore them to consider how such policies could impact the biomes and communities near the Yawzi and Tektite sites. Ultimately, our results are meant for our own understanding and hopefully inspire further research into this topic.

6. Conclusion

Overall, we used a variety of different models and methods to help us understand coral reef coverage in our two locations. ARIMA indicates that coral coverage is indeed decreasing significantly over time in Yawzi, while the Kalman Filter and VARMAX mostly follow this trend, accompanied by a hint at the possibility of increasing coverage. Our attempts to introduce temperature did not cause great increases in model accuracy. We do acknowledge our limited understanding between the environmental relationship of coral and temperature. ARIMA seems to showcase a less severe decrease in Tektite. The models used, including time series decomposition, ARIMA, Kalman filtering, and VARMAX, provide valuable insights into coral reef dynamics and offer potential for more accurate predictions. While these findings suggest a decline in coral health, it's crucial to acknowledge the limitations of the study and the ethical considerations involved. The results should be viewed as part of a broader understanding and should not be the sole basis for decision-making. These results should be used along with other considerations for any potential plans moving forward.

References

[1] Li, A., Reidenbach, M.A. Forecasting decadal changes in sea surface temperatures and coral bleaching within a Caribbean coral reef. Coral Reefs 33, 847–861 (2014). https://doi.org/10.1007/s00338-014-1162-1

[2] Donner, S.D. (2011), An evaluation of the effect of recent temperature variability on the prediction of coral bleaching events. Ecological Applications, 21: 1718-1730. https://doi.org/10.1890/10-0107.1

[3] Edmunds, P. J. (2018) Long-term mean daily seawater temperature data from USVI starting in 2003. Biological and Chemical Oceanography Data Management Office (BCO-DMO). (Version 1) Version Date 2018-05-17. http://lod.bco-dmo.org/id/dataset/736809 (Accessed 2024-13-30)

[4] California State University Northridge and P. Edmunds. 2022. Virgin Islands National Park: Coral Reef: Population Dynamics: Scleractinian corals ver 3. Environmental Data Initiative. https://doi.org/10.6073/pasta/dd48dc2f900fd1128461c59245372a93 (Accessed 2024-03-30).

[5] Constantin W. Arnscheidt, Daniel H. Rothman ,Presence or absence of stabilizing Earth system feedbacks on different time scales.Sci. Adv.8,eadc9241(2022).DOI:10.1126/sciadv.adc9241

[6] "Coral Reef Ecosystems." NOAA, National Oceanic and Atmospheric Administration, 1 Feb. 2019, https://www.noaa.gov/education/resource-collections/marine-life/coral-reef-ecosystems. Accessed 30 Mar. 2024.

[7] Hansen, J., R. Ruedy, M. Sato, and K. Lo (2010), Global surface temperature change, Rev. Geophys., 48, RG4004, doi:10.1029/2010RG000345.

vol3_code

April 13, 2024

1 Data Cleaning

In this notebook, we explore and clean our initial dataset. We have a dataframe containing measurements of coral reef cover off of several coasts in the U.S. Virgin islands. The areas are divided into quadrats (areas subdivided into squares) and then are measured by the proportion of the seafloor that is covered in each coral species.

1.0.1 Holding Data for Model Evaluation

We plan to tuck away the data from the most recent couple of years and use that as a testing dataset after our model has been trained and optimized. Unfortunately this data will not be independent of the years before (e.g. 2020 coral reef cover is expected to depend on the cover from 2019). However, we care about predicting coral cover in future years, so splitting our data by time will make the most sense for the goal of our project.

[ ]:
```
import pandas as pd
import numpy as np
from datetime import datetime
from matplotlib import pyplot as plt
from statsmodels.tsa.arima.model import ARIMA
```

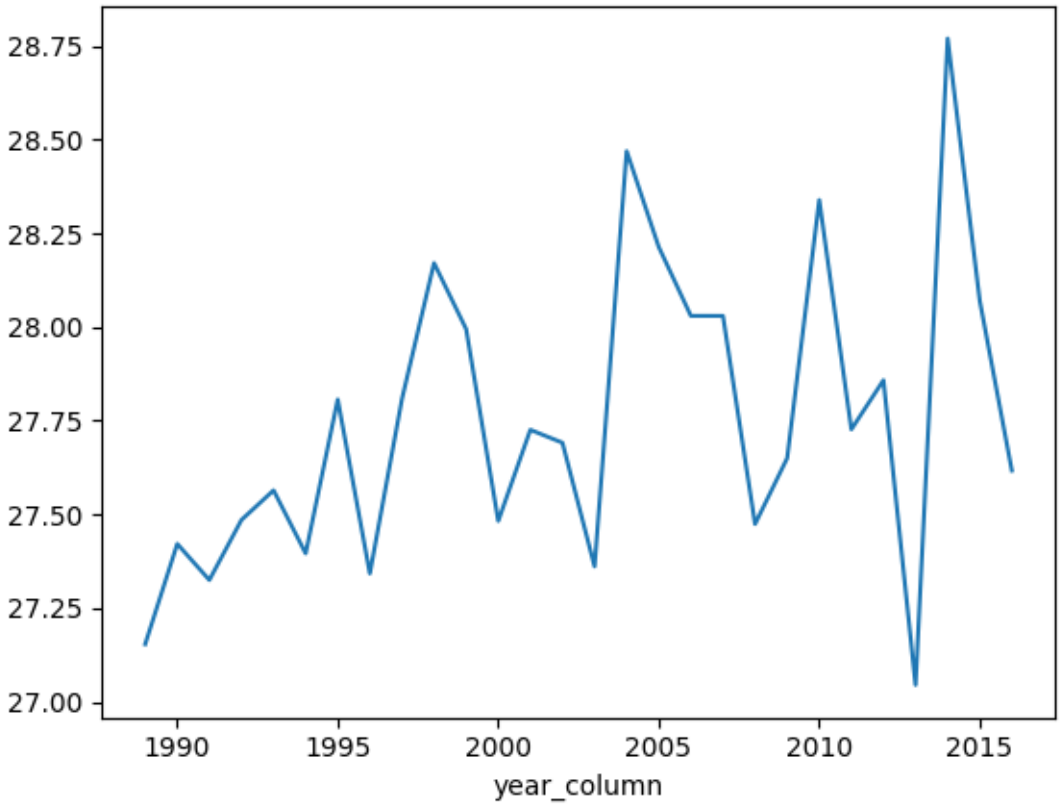
[ ]:
```
sea_df = pd.read_csv("seawatertemp (1).csv", index_col = "Date") #read in the␣
 ↪dataset
sea_df.index = pd.to_datetime(sea_df.index)
sea_df['year_column'] = sea_df.index.year
sea_df = sea_df[sea_df['Site'].isin(['Yawzi_NPS','Yawzi_9m'])]
specific_value_count = sea_df['Temperature'].value_counts()['nd']
# print(specific_value_count)
sea_df = sea_df[sea_df['Temperature'] != 'nd']
sea_df['Temperature'] = sea_df['Temperature'].astype(float)
avg_sea_temp = sea_df.groupby('year_column')['Temperature'].mean()
avg_sea_temp
# avg_sea_temp.plot()
```

```
year_column
1989    27.153627
1990    27.421425
1991    27.324986
1992    27.485410
```

```
1993    27.564384
1994    27.395973
1995    27.806966
1996    27.342350
1997    27.808127
1998    28.170986
1999    27.994685
2000    27.482548
2001    27.725918
2002    27.691342
2003    27.361349
2004    28.469801
2005    28.213955
2006    28.030083
2007    28.029944
2008    27.474306
2009    27.650138
2010    28.339280
2011    27.727008
2012    27.858338
2013    27.045025
2014    28.770303
2015    28.070670
2016    27.617150
Name: Temperature, dtype: float64
```

[ ]:
```
avg_sea_temp.plot()
```

```
<AxesSubplot: xlabel='year_column'>
```



[ ]:
```
df = pd.read_csv("coral_dataset.csv", index_col = "Date") #read in the dataset
```

[ ]:
```
df #print and look at the data set
```

```
              site transect quadrat percentCover_allCoral  \
Date
1987/12/01  Tektite        1    T1Q1                  16.5
1987/12/01  Tektite        1    T1Q2                    10
1987/12/01  Tektite        1    T1Q3                  13.5
1987/12/01  Tektite        1    T1Q4                  15.5
1987/12/01  Tektite        1    T1Q5                    19
…               …      …     …                     …
2021/07/01    Yawzi       nd       6                   0.0
2021/07/01    Yawzi       nd       7                   0.0
2021/07/01    Yawzi       nd       8                   0.5
2021/07/01    Yawzi       nd       9                   0.5
2021/07/01    Yawzi       nd      10                   3.5

           percentCover_macroalgae percentCover_CTB
Date
1987/12/01                     3.5                6
```

```
1987/12/01                      6              74
1987/12/01                    3.5              45
1987/12/01                    6.5            61.5
1987/12/01                      4              48
…                             …              …
2021/07/01                   30.0            50.0
2021/07/01                   28.5            45.5
2021/07/01                   35.0            35.0
2021/07/01                   42.0            42.5
2021/07/01                   34.0            38.5

[2283 rows x 6 columns]
```

[ ]:
```
print(df['percentCover_CTB'].unique()) #check the unique values of the column
```

```
['6' '74' '45' '61.5' '48' '10' '3' '29' '38' '30.5' '31.5' '26' '24'
 '11.5' '14.5' '19' '15.5' '22' '16' '19.3' '31' '39' '30' '19.5' '20.5'
 '44.5' '63' '39.5' '46' '62' '6.8' '38.5' '32.5' '21' '36.5' '25.5'
 '10.5' '18' '12.5' '24.5' '32' '14' '25' '33.5' '40.5' '29.8' '33' '41'
 '9' '34.5' '23' '27.5' '29.5' '7.5' '7' '26.5' '22.5' '16.5' '12' '17'
 '16.1' '17.5' '35.5' '50' '37.5' '27' '18.5' '9.5' '37' '41.5' '51'
 '49.5' '20' '42.5' '21.5' '43.5' '40' '28.5' '28' '23.5' '6.5' '15' '11'
 '34' '8.5' '8' '13' '13.5' '36' '30.3' '32.3' '25.6' '30.9' '17.2' '28.8'
 '5.3' '15.1' '15.6' '25.9' '22.6' '8.9' '10.3' '7.6' '8.7' '8.8' '7.3'
 '5.6' '0' '1.6' '25.3' '40.7' '3.2' '4.5' '2.5' '3.5' '5' '18.7' '18.8'
 '20.8' '8.1' '2.6' '12.3' '14.9' '14.1' '2.7' '15.9' '12.8' '15.7' '6.6'
 '13.8' '1.1' '2.2' '33.2' '23.7' '31.4' '26.2' '1' '7.4' '18.6' '4' '1.5'
 '5.5' '22.3' '11.1' '12.2' '11.2' '24.7' '8.2' '4.1' '7.7' '14.3' '9.7'
 '10.2' '10.7' '5.1' '22.7' '17.9' '2' '9.9' '36.2' '30.8' '20.7' '20.1'
 '23.2' '9.1' '11.4' '12.1' '10.1' '29.4' '24.2' '10.6' '16.6' '12.6'
 '19.1' '18.1' '31.2' '8.6' '11.7' '21.8' '33.7' '13.2' '4.8' '18.4'
 '37.2' '32.1' '13.9' '14.6' '34.7' '36.9' '7.2' '9.2' '16.7' '12.9'
 '29.7' '19.4' '16.4' '10.4' '3.6' '11.9' '11.3' '41.3' '34.2' '24.4'
 '22.2' '19.2' '13.1' '14.4' '9.3' '12.7' '30.4' '7.9' '8.4' '6.3' '15.2'
 '18.9' '22.1' '4.2' '6.4' '17.8' '4.7' '2.1' '2.8' '5.8' '28.9' '40.8'
 '31.6' '37.3' '32.7' '33.1' '25.1' '43.4' '24.3' '58.2' '27.2' '35.3'
 '41.2' '38.8' '32.8' '32.2' '9.4' '55.7' '31.8' '28.6' '16.2' '48.8'
 '43.2' '31.1' '23.6' '30.7' '22.4' '39.3' '39.4' '51.8' '20.4' '23.4'
 '32.6' '16.8' '26.3' '59.7' '45.1' '39.6' '46.4' '40.4' '57.1' '48.2'
 '46.7' '37.8' '31.9' '42.7' '46.3' '53.4' '42.3' '36.4' '35.6' '40.6'
 '56.5' '50.6' '56.7' '43.9' '52.3' '71.8' '53' '39.7' '59' '62.6' '48.3'
 '37.6' '40.3' '50.3' '51.7' '74.3' '63.1' '37.9' '54' '60.3' '42' '45.4'
 '67.9' '39.2' '61.4' '35.9' '46.5' '60.1' '62.1' '29.1' '54.3' '56.4'
 '48.5' '45.6' '37.1' '63.8' '57.7' '23.9' '26.9' '26.1' '21.6' '38.1'
 '29.2' '17.3' '39.8' '71.1' '52.6' '55' '49.7' '43.1' '58.7' '45.5'
 '50.8' '39.1' '53.5' '51.5' '44.7' '57.8' '44.2' '33.8' '27.8' '35.2'
 '23.1' '54.8' '49' '27.1' '49.2' '17.1' '56' '36.7' '57.5' '38.6' '34.8'
 '59.3' '42.1' '35.8' '40.9' '35.4' '41.4' '43.7' '33.3' '34.3' '29.3'
```

```
'27.6' '37.7' '15.8' '55.6' '56.1' '66.3' '62.5' '55.9' '73.3' '48.9'
'56.9' '71.4' '79.4' '70.5' '65.1' '72.3' '58.1' '62.2' '64' '61.9'
'51.6' '26.7' '20.9' '34.4' '39.9' '14.7' '19.7' '9.8' '18.2' '21.7'
'9.6' '13.6' '47.9' '62.3' '47.4' '42.6' '46.1' '63.9' '57.4' '43.6'
'48.7' '41.8' '18.3' '41.1' '28.2' '21.3' '28.4' '20.3' '28.3' '27.4'
'20.6' '19.8' '29.9' '44.8' '44' '57' '82' '19.6' '69' '47' '50.5' '52'
'35' 'nd' '54.5' '55.5' '66' '72.5' '71' '45.9' '25.2' '76.3' '30.2'
'84.5' '87.7' '83.2' '72.8' '72.7' '68.5' '62.8' '47.5' '60' '63.5'
'69.5' '65' '73' '83.7' '79.1' '79.5' '59.5' '20.2' '11.6' '16.9' '13.4'
'52.8' '55.4' '57.3' '60.7' '54.2' '58.5' '23.8' '13.3' '24.1' '27.9'
'34.6' '10.9' '8.3' '14.2' '21.1' '35.7' '31.3' '45.2' '19.9' '42.9'
'26.8' '64.5' '70.4' '85.4' '61.3' '65.3' '51.3' '54.4' '78.2' '83.3'
'7.8' '35.1' '47.7' '30.6' '36.1' '21.9' '36.3' '36.6' '48.1' '24.9'
'25.4' '27.7' '55.1' '37.4' '23.3' '2.3' '1.2' '3.8' '3.7' '1.8' '53.1'
'83.9' '46.2' '41.9' '22.9' '30.1' '33.9' '69.2' '62.4' '80.3' '21.4'
'75.5' '61.7' '59.9' '65.2' '53.3' '53.2' '58.9' '64.4' '60.4' '64.6'
'70.9' '70.1' '78.4' '69.3' '69.6' '56.3' '42.8' '64.9' '66.1' '67.3'
'54.7' '45.3' '16.3' '38.2' '61.8' '40.2' '56.8' '70.7' '43' '28.1'
'72.4' '67.2' '76.5' '59.6' '45.7' '47.2' '75' '60.5' '77.9' '53.8'
'57.6' '55.8' '82.5' '74.4' '52.5' '84' '26.6' '38.3' '80.5' '47.1'
'65.4' '54.6' '81.2' '64.2' '44.4' '44.3' '61.1' '66.5' '17.7' '58.3'
'44.1' '45.8' '68.7' '75.3' '71.9' '73.7' '46.6' '68.9' '85.2' '53.9'
'44.6' '56.2' '41.7' '27.0' '17.6' '26.0' '22.0' '36.0' '42.0' '53.0'
'40.1' '30.0' '25.0' '44.0' '50.0' '35.0']
```

[ ]:
```
print(len(df)) #print the amount of rows of the dataset
#check and print the amount of times there is no data given in the entry for␣
 ↪our three main columns
specific_value_count = df['percentCover_CTB'].value_counts()['nd']
print(specific_value_count)
specific_value_count = df['percentCover_macroalgae'].value_counts()['nd']
print(specific_value_count)
specific_value_count = df['percentCover_allCoral'].value_counts()['nd']
print(specific_value_count)
```

```
2283
19
19
9
```

Based on the previous numbers we can drop all of the rows who have “nd” as the value in our main three columns without losing a bunch of data.

[ ]:
```
df = df[df['percentCover_CTB'] != 'nd'] #drop all of the rows with these␣
 ↪entries and then print out the amount of rows in our dataset
df = df[df['percentCover_macroalgae'] != 'nd']
df = df[df['percentCover_allCoral'] != 'nd']
print(len(df))
```

2264

This leaves us with 2,264 individual rows of data to work with.

[ ]:
```
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
↪'percentCover_allCoral'] #get a list of the columns that we want to cast
df[columns_to_cast] = df[columns_to_cast].astype(float) #cast these columns as␣
↪floats and divide by 100 to make the percents
df[columns_to_cast] = df[columns_to_cast]/100
df['percentCover_other'] = 1 -␣
↪(df['percentCover_CTB']+df['percentCover_macroalgae']) #add in another␣
↪column for other types of coral
```

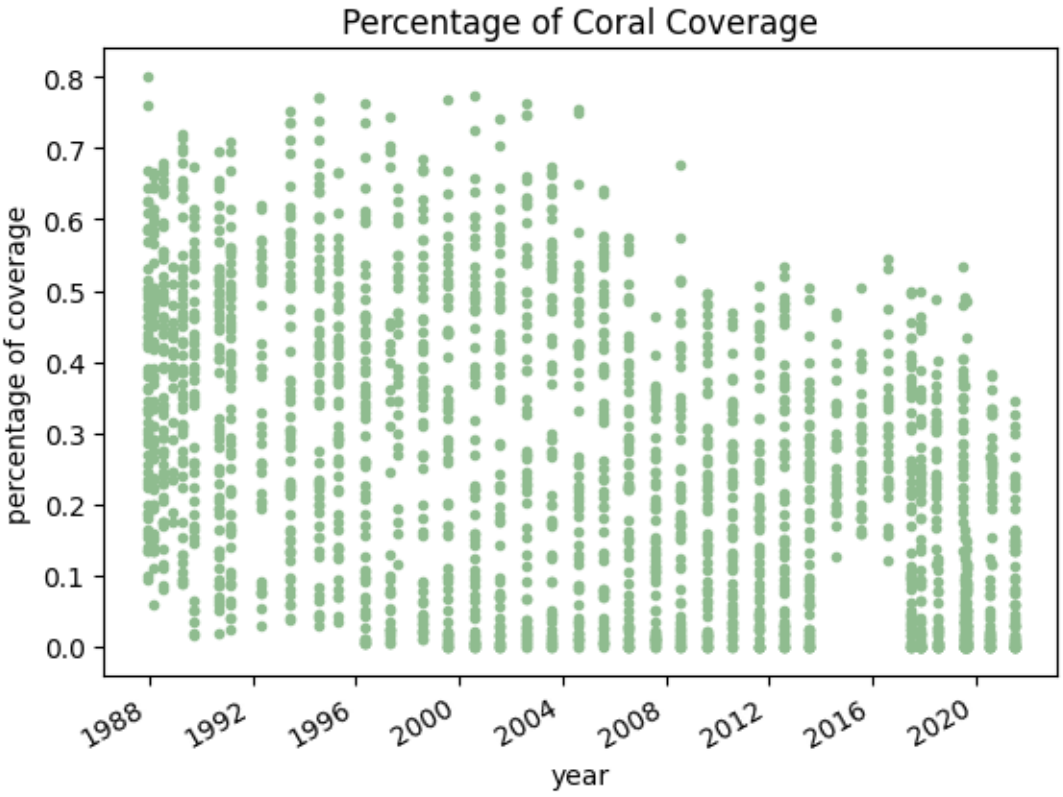
[ ]:
```
# convert the date column to date time objects
df.index = pd.to_datetime(df.index)
```

[ ]:
```
df #print the dataset with the new feature
```

```
              site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Tektite        1    T1Q1                  0.165
1987-12-01  Tektite        1    T1Q2                  0.100
1987-12-01  Tektite        1    T1Q3                  0.135
1987-12-01  Tektite        1    T1Q4                  0.155
1987-12-01  Tektite        1    T1Q5                  0.190
…             …      …     …                      …
2021-07-01    Yawzi       nd       6                  0.000
2021-07-01    Yawzi       nd       7                  0.000
2021-07-01    Yawzi       nd       8                  0.005
2021-07-01    Yawzi       nd       9                  0.005
2021-07-01    Yawzi       nd      10                  0.035

            percentCover_macroalgae  percentCover_CTB  percentCover_other
Date
1987-12-01                    0.035             0.060               0.905
1987-12-01                    0.060             0.740               0.200
1987-12-01                    0.035             0.450               0.515
1987-12-01                    0.065             0.615               0.320
1987-12-01                    0.040             0.480               0.480
…                             …                 …                   …
2021-07-01                    0.300             0.500               0.200
2021-07-01                    0.285             0.455               0.260
2021-07-01                    0.350             0.350               0.300
2021-07-01                    0.420             0.425               0.155
2021-07-01                    0.340             0.385               0.275

[2264 rows x 7 columns]
```

Plot percentage of coral coverage

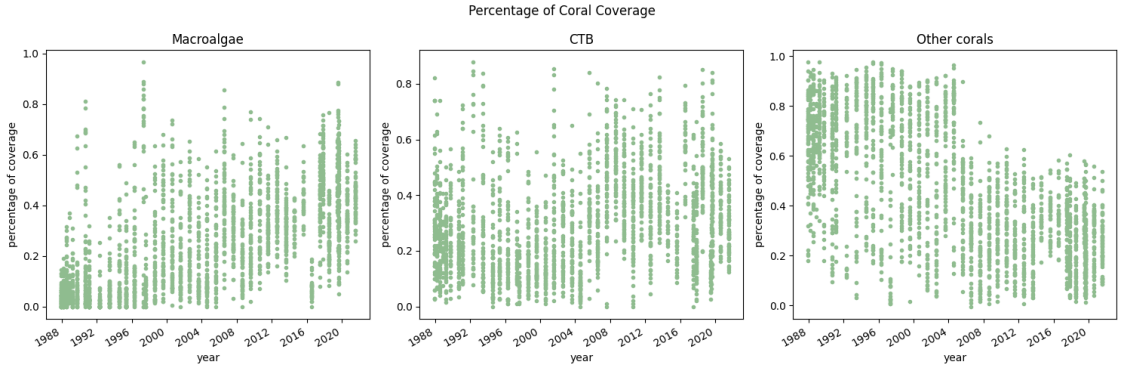
[ ]:
```
df.plot(y = "percentCover_allCoral", linestyle = 'None', marker = '.', color =␣
↪'darkseagreen', title = 'Percentage of Coral Coverage', ylabel='percentage␣
↪of coverage', xlabel = 'year',legend=False)
plt.show()
```



In this plot we can see all the data points of coral reef coverage. While there are seom small outliers, there doesn't appear to be anything extreme enough that we would need to drop that data point.

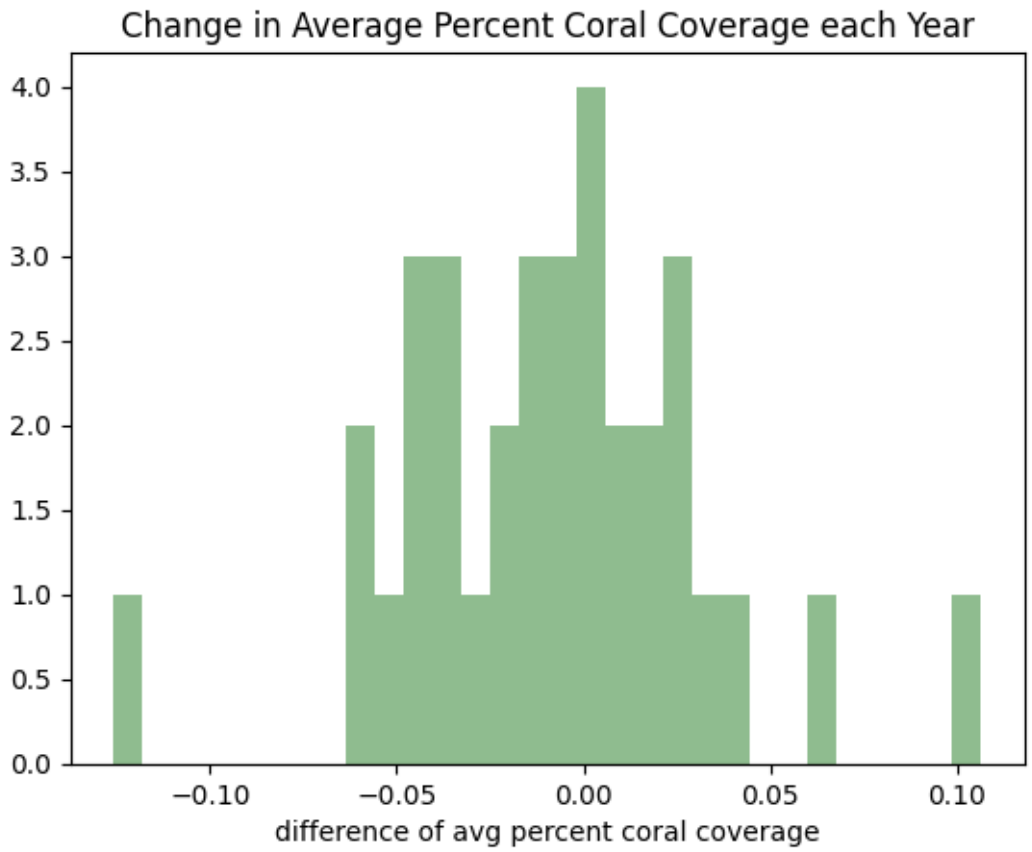
[ ]:
```
#plot coral
fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize = (15,5))
df.plot(ax = ax1,y = ["percentCover_macroalgae"], linestyle = 'None', marker =␣
↪'.', color = ['darkseagreen'], title = 'Macroalgae', ylabel='percentage of␣
↪coverage', xlabel = 'year',legend=False)

df.plot(ax = ax2,y = [ "percentCover_CTB"], linestyle = 'None', marker = '.',␣
↪color = ['darkseagreen'], title = 'CTB', ylabel='percentage of coverage',␣
↪xlabel = 'year',legend=False)

df.plot(ax = ax3,y = ["percentCover_other"], linestyle = 'None', marker = '.',␣
↪color = ['darkseagreen'], title = 'Other corals', ylabel='percentage of␣
↪coverage', xlabel = 'year',legend=False)
```

```
fig.suptitle("Percentage of Coral Coverage")
fig.tight_layout()
plt.show()
```



Plot histogram of difference in average coral coverage each year

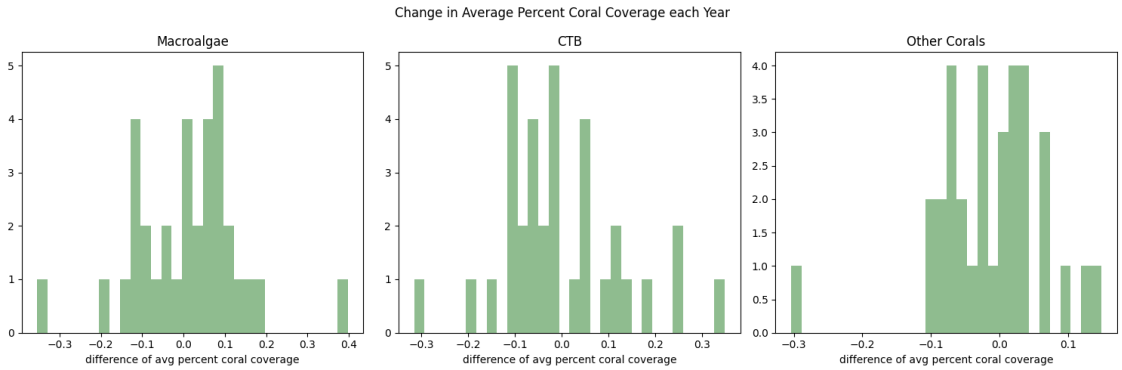
[ ]:
```
years = np.arange(1987,2022,1)
difference = []
for i in range(len(years)-1,0,-1):
    curr = df.loc[str(years[i])]
    past = df.loc[str(years[i-1])]
    difference.append(curr["percentCover_allCoral"].mean() -␣
 ↪past["percentCover_allCoral"].mean())

plt.hist(difference,bins = 30, color = "darkseagreen")
plt.title("Change in Average Percent Coral Coverage each Year")
plt.xlabel("difference of avg percent coral coverage")
plt.show()
```



We see from the histogram that the change in coral reef cover tends to stay near zero from year to year. This supports our assumption that the cover will depend on previous years. We also see there is slightly more years with negative growth, further supporting our concern that coral reefs are getting smaller over time.

[ ]:
```
years = np.arange(1987,2022,1)

#initialize lists for changes in specific types of coral
difference_marco = []
difference_ctb = []
difference_other = []

#calculate the differences from year to year for each coral species
for i in range(len(years)-1,0,-1):
    curr = df.loc[str(years[i])]
    past = df.loc[str(years[i-1])]

    difference_marco.append(curr["percentCover_macroalgae"].mean() -␣
 ↪past["percentCover_macroalgae"].mean())
```

```
    difference_ctb.append(curr["percentCover_CTB"].mean() -␣
 ↪past["percentCover_CTB"].mean())
    difference_other.append(curr["percentCover_other"].mean() -␣
 ↪past["percentCover_other"].mean())

#plot the percent change in coral reef coverage
plt.figure(figsize=(15,5))
plt.subplot(131)
plt.hist(difference_marco,bins = 30, color = "darkseagreen")
plt.title("Macroalgae")
plt.xlabel("difference of avg percent coral coverage")

plt.subplot(132)
plt.hist(difference_ctb,bins = 30, color = "darkseagreen")
plt.title("CTB")
plt.xlabel("difference of avg percent coral coverage")

plt.subplot(133)
plt.hist(difference_other,bins = 30, color = "darkseagreen")
plt.title("Other Corals")
plt.xlabel("difference of avg percent coral coverage")

plt.suptitle("Change in Average Percent Coral Coverage each Year")
plt.tight_layout()
plt.show()
```



[ ]:
```
explore_df = df[df['site'] == 'Tektite']
explore_df = explore_df[explore_df['quadrat'] == 'T1Q1']
explore_df
```

Date	site	transect	quadrat	percentCover_allCoral	\
1987-12-01	Tektite	1	T1Q1	0.165	
1988-03-01	Tektite	1	T1Q1	0.195	

1988-07-01	Tektite	1	T1Q1	0.255
1989-04-01	Tektite	1	T1Q1	0.310
1989-10-01	Tektite	1	T1Q1	0.245
1990-09-01	Tektite	1	T1Q1	0.310
1991-03-01	Tektite	1	T1Q1	0.215
1993-06-01	Tektite	1	T1Q1	0.163
1994-08-01	Tektite	1	T1Q1	0.226
1995-05-01	Tektite	1	T1Q1	0.290
1996-05-01	Tektite	1	T1Q1	0.385
1997-08-01	Tektite	1	T1Q1	0.325
1998-08-01	Tektite	1	T1Q1	0.337
1999-08-01	Tektite	1	T1Q1	0.279
2000-08-01	Tektite	1	T1Q1	0.221
2001-08-01	Tektite	1	T1Q1	0.361
2002-08-01	Tektite	1	T1Q1	0.387
2003-08-01	Tektite	1	T1Q1	0.389
2004-08-01	Tektite	1	T1Q1	0.386
2005-08-01	Tektite	1	T1Q1	0.387
2006-08-01	Tektite	1	T1Q1	0.224
2007-08-01	Tektite	1	T1Q1	0.138
2008-08-01	Tektite	1	T1Q1	0.219
2009-08-01	Tektite	1	T1Q1	0.264
2010-08-01	Tektite	1	T1Q1	0.284
2011-08-01	Tektite	1	T1Q1	0.197
2012-08-01	Tektite	1	T1Q1	0.253
2013-08-01	Tektite	1	T1Q1	0.250
2014-08-01	Tektite	1	T1Q1	0.215
2015-08-01	Tektite	1	T1Q1	0.235
2016-08-01	Tektite	1	T1Q1	0.230
2017-07-01	Tektite	1	T1Q1	0.410
2017-11-01	Tektite	1	T1Q1	0.227
2018-07-01	Tektite	1	T1Q1	0.242
2019-07-01	Tektite	1	T1Q1	0.308
2020-08-03	Tektite	1	T1Q1	0.281
2021-07-01	Tektite	nd	T1Q1	0.161

Date	percentCover_macroalgae	percentCover_CTB	percentCover_other
1987-12-01	0.035	0.060	0.905
1988-03-01	0.005	0.445	0.550
1988-07-01	0.175	0.335	0.490
1989-04-01	0.090	0.300	0.610
1989-10-01	0.065	0.415	0.520
1990-09-01	0.280	0.160	0.560
1991-03-01	0.025	0.330	0.645
1993-06-01	0.062	0.303	0.635
1994-08-01	0.037	0.125	0.838
1995-05-01	0.130	0.210	0.660

1996-05-01	0.102	0.187	0.711
1997-08-01	0.080	0.175	0.745
1998-08-01	0.175	0.165	0.660
1999-08-01	0.234	0.223	0.543
2000-08-01	0.161	0.362	0.477
2001-08-01	0.098	0.294	0.608
2002-08-01	0.225	0.314	0.461
2003-08-01	0.086	0.335	0.579
2004-08-01	0.265	0.127	0.608
2005-08-01	0.249	0.289	0.462
2006-08-01	0.495	0.255	0.250
2007-08-01	0.245	0.597	0.158
2008-08-01	0.219	0.557	0.224
2009-08-01	0.208	0.503	0.289
2010-08-01	0.274	0.311	0.415
2011-08-01	0.323	0.434	0.243
2012-08-01	0.348	0.328	0.324
2013-08-01	0.205	0.480	0.315
2014-08-01	0.380	0.275	0.345
2015-08-01	0.310	0.410	0.280
2016-08-01	0.155	0.556	0.289
2017-07-01	0.574	0.000	0.426
2017-11-01	0.571	0.167	0.262
2018-07-01	0.242	0.479	0.279
2019-07-01	0.444	0.182	0.374
2020-08-03	0.214	0.411	0.375
2021-07-01	0.347	0.383	0.270

[ ]:
```
percent_cover_all_coral = np.array(explore_df['percentCover_allCoral'])
print(percent_cover_all_coral)
```

```
[0.165 0.195 0.255 0.31  0.245 0.31  0.215 0.163 0.226 0.29  0.385 0.325
 0.337 0.279 0.221 0.361 0.387 0.389 0.386 0.387 0.224 0.138 0.219 0.264
 0.284 0.197 0.253 0.25  0.215 0.235 0.23  0.41  0.227 0.242 0.308 0.281
 0.161]
```

1.1 ARMA Model Code

[ ]:
```
def sm_arma(array, p_max=3, q_max=3, n=10):
    """
    Build an ARMA model with statsmodel and
    predict future n values.

    Parameters:
        filename (str): data filename
        p_max (int): maximum order of autoregressive model
        q_max (int): maximum order of moving average model
        n (int): number of values to predict
```

```
    Return:
        aic (float): aic of optimal model
    """
    z = np.diff(array)

    best_aic = np.inf #set our values
    best_model = None
    best_p = None
    best_q = None
    best_std = None
    for p in range(1,p_max+1): #iterate through p and q
        for q in range(1,q_max+1):
            model = ARIMA(z,order=(p,0,q),trend='c').
↪fit(method='innovations_mle') #set the model and then check if best
            aic = model.aic
            if aic < best_aic:
                best_aic = aic
                best_model = model
                best_p = p
                best_q = q
                best_std = model.resid


    final = best_model.predict(start=0,end=len(z)+n) #return desired value␣
↪after prediciton
    return best_aic,final,np.std(best_std)
```

[ ]:
```
aic, final,std = sm_arma(percent_cover_all_coral)
```

[ ]:
```
z = np.diff(percent_cover_all_coral)
```

[ ]:
```
print(len(z))
print(z)
print(len(final))
print(final)
```

```
36
[ 0.03   0.06   0.055 -0.065  0.065 -0.095 -0.052  0.063  0.064  0.095
 -0.06   0.012 -0.058 -0.058  0.14   0.026  0.002 -0.003  0.001 -0.163
 -0.086  0.081  0.045  0.02  -0.087  0.056 -0.003 -0.035  0.02  -0.005
  0.18  -0.183  0.015  0.066 -0.027 -0.12 ]
47
[-1.34488735e-04 -8.14832698e-03 -2.58161958e-02 -4.11108173e-02
 -1.01444411e-02 -3.51044411e-02  8.51466974e-03  2.99003789e-02
 -6.02972483e-04 -2.92209732e-02 -7.00853446e-02 -3.75116528e-02
 -4.06112623e-02 -1.09375694e-02  1.70314778e-02 -5.05782798e-02
 -6.03965365e-02 -5.83741375e-02 -5.41827431e-02 -5.22502072e-02
```

```
 2.95073990e-02  7.03154551e-02  2.75938497e-02  4.26379735e-03
-5.85712055e-03  3.74211857e-02  8.21083814e-03  9.35821697e-03
 2.64601270e-02  1.55477626e-02  1.75083730e-02 -7.34869262e-02
 2.06235054e-02  1.24166458e-02 -2.13007607e-02 -7.16269243e-03
 5.36791961e-02  2.50572290e-02  1.16584714e-02  5.38613161e-03
 2.44987080e-03  1.07532353e-03  4.31858824e-04  1.30634679e-04
-1.03769183e-05 -7.63884608e-05 -1.07290343e-04]
```

[ ]:
```
# time = np.arange(1987,2022,1)
# time2 = np.arange(1987,2022+10,1)
time = np.linspace(1987,2021,len(z))
time2 = np.linspace(1987,2021+10,len(final))
print(len(time))
print(time)
print(len(time2))
print(time2)
```

```
36
[1987.         1987.97142857 1988.94285714 1989.91428571 1990.88571429
 1991.85714286 1992.82857143 1993.8        1994.77142857 1995.74285714
 1996.71428571 1997.68571429 1998.65714286 1999.62857143 2000.6
 2001.57142857 2002.54285714 2003.51428571 2004.48571429 2005.45714286
 2006.42857143 2007.4        2008.37142857 2009.34285714 2010.31428571
 2011.28571429 2012.25714286 2013.22857143 2014.2        2015.17142857
 2016.14285714 2017.11428571 2018.08571429 2019.05714286 2020.02857143
 2021.        ]
47
[1987.         1987.95652174 1988.91304348 1989.86956522 1990.82608696
 1991.7826087  1992.73913043 1993.69565217 1994.65217391 1995.60869565
 1996.56521739 1997.52173913 1998.47826087 1999.43478261 2000.39130435
 2001.34782609 2002.30434783 2003.26086957 2004.2173913  2005.17391304
 2006.13043478 2007.08695652 2008.04347826 2009.         2009.95652174
 2010.91304348 2011.86956522 2012.82608696 2013.7826087  2014.73913043
 2015.69565217 2016.65217391 2017.60869565 2018.56521739 2019.52173913
 2020.47826087 2021.43478261 2022.39130435 2023.34782609 2024.30434783
 2025.26086957 2026.2173913  2027.17391304 2028.13043478 2029.08695652
 2030.04347826 2031.        ]
```

[ ]:
```
explore_df = df[df['site'] == 'Yawzi']
explore_df = explore_df[explore_df['quadrat'] == 'T1Q1']
explore_df
```

```
            site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Yawzi        1    T1Q1                  0.570
1988-03-01  Yawzi        1    T1Q1                  0.490
1988-07-01  Yawzi        1    T1Q1                  0.440
1989-04-01  Yawzi        1    T1Q1                  0.480
1989-10-01  Yawzi        1    T1Q1                  0.510
```

1990-09-01	Yawzi	1	T1Q1	0.520
1991-03-01	Yawzi	1	T1Q1	0.450
1992-05-01	Yawzi	1	T1Q1	0.517
1993-06-01	Yawzi	1	T1Q1	0.648
1994-08-01	Yawzi	1	T1Q1	0.544
1995-05-01	Yawzi	1	T1Q1	0.374
1996-05-01	Yawzi	1	T1Q1	0.687
1997-05-01	Yawzi	1	T1Q1	0.745
1998-08-01	Yawzi	1	T1Q1	0.616
1999-08-01	Yawzi	1	T1Q1	0.320
2000-08-01	Yawzi	1	T1Q1	0.603
2001-08-01	Yawzi	1	T1Q1	0.589
2002-08-01	Yawzi	1	T1Q1	0.539
2003-08-01	Yawzi	1	T1Q1	0.601
2004-08-01	Yawzi	1	T1Q1	0.417
2005-08-01	Yawzi	1	T1Q1	0.441
2006-08-01	Yawzi	1	T1Q1	0.400
2007-08-01	Yawzi	1	T1Q1	0.368
2008-08-01	Yawzi	1	T1Q1	0.324
2009-08-01	Yawzi	1	T1Q1	0.356
2010-08-01	Yawzi	1	T1Q1	0.459
2011-08-01	Yawzi	1	T1Q1	0.452
2012-08-01	Yawzi	1	T1Q1	0.452
2013-08-01	Yawzi	1	T1Q1	0.335
2019-08-14	Yawzi	1	T1Q1	0.490
2019-08-15	Yawzi	1	T1Q1	0.485
2019-08-16	Yawzi	1	T1Q1	0.051
2017-07-01	Yawzi	1	T1Q1	0.500
2017-11-01	Yawzi	1	T1Q1	0.444

Date	percentCover_macroalgae	percentCover_CTB	percentCover_other
1987-12-01	0.005	0.215	0.780
1988-03-01	0.000	0.250	0.750
1988-07-01	0.005	0.290	0.705
1989-04-01	0.035	0.235	0.730
1989-10-01	0.040	0.145	0.815
1990-09-01	0.085	0.075	0.840
1991-03-01	0.000	0.085	0.915
1992-05-01	0.000	0.189	0.811
1993-06-01	0.075	0.140	0.785
1994-08-01	0.040	0.175	0.785
1995-05-01	0.030	0.340	0.630
1996-05-01	0.011	0.097	0.892
1997-05-01	0.000	0.225	0.775
1998-08-01	0.025	0.100	0.875
1999-08-01	0.249	0.110	0.641
2000-08-01	0.005	0.109	0.886

```
2001-08-01                    0.313              0.313              0.374
2002-08-01                    0.202              0.078              0.720
2003-08-01                    0.020              0.121              0.859
2004-08-01                    0.006              0.202              0.792
2005-08-01                    0.000              0.531              0.469
2006-08-01                    0.073              0.395              0.532
2007-08-01                    0.032              0.542              0.426
2008-08-01                    0.000              0.497              0.503
2009-08-01                    0.021              0.543              0.436
2010-08-01                    0.209              0.163              0.628
2011-08-01                    0.156              0.357              0.487
2012-08-01                    0.317              0.226              0.457
2013-08-01                    0.350              0.295              0.355
2019-08-14                    0.240              0.195              0.565
2019-08-15                    0.350              0.150              0.500
2019-08-16                    0.556              0.107              0.337
2017-07-01                    0.116              0.369              0.515
2017-11-01                    0.187              0.359              0.454
```

[ ]:
```
percent_cover_all_coral = np.array(explore_df['percentCover_allCoral'])
print(percent_cover_all_coral)
```

```
[0.57  0.49  0.44  0.48  0.51  0.52  0.45  0.517 0.648 0.544 0.374 0.687
 0.745 0.616 0.32  0.603 0.589 0.539 0.601 0.417 0.441 0.4   0.368 0.324
 0.356 0.459 0.452 0.452 0.335 0.49  0.485 0.051 0.5   0.444]
```

[ ]:
```
aic, final,std = sm_arma(percent_cover_all_coral)
```

[ ]:
```
z = np.diff(percent_cover_all_coral)
```

[ ]:
```
print(len(z))
print(z)
print(len(final))
print(final)
```

```
33
[-0.08  -0.05   0.04   0.03   0.01  -0.07   0.067  0.131 -0.104 -0.17
  0.313  0.058 -0.129 -0.296  0.283 -0.014 -0.05   0.062 -0.184  0.024
 -0.041 -0.032 -0.044  0.032  0.103 -0.007  0.    -0.117  0.155 -0.005
 -0.434  0.449 -0.056]
44
[-0.00457586  0.02432928  0.06623698  0.01681209 -0.04101648 -0.05241812
  0.01643183 -0.02309803 -0.16001424 -0.0699738   0.14091613 -0.11327093
 -0.26007094 -0.07289419  0.25548244  0.0326856  -0.10434916 -0.01874656
 -0.04774091  0.0908686   0.10567232  0.07515865  0.07952902  0.08796979
  0.03474601 -0.0884384  -0.09795479 -0.05995054  0.05587018 -0.0532246
 -0.1030294   0.29805705 -0.01995313 -0.17326996  0.063937    0.05239955
 -0.05059325 -0.01764705  0.01953459 -0.00523348 -0.01518332 -0.00142002
 -0.0006824  -0.00703788]
```

[ ]:
```
# time = np.arange(1987,2022,1)
# time2 = np.arange(1987,2022+10,1)
time = np.linspace(1987,2021,len(z))
time2 = np.linspace(1987,2021+10,len(final))
print(len(time))
print(time)
print(len(time2))
print(time2)
```

```
33
[1987.     1988.0625 1989.125  1990.1875 1991.25   1992.3125 1993.375
 1994.4375 1995.5    1996.5625 1997.625  1998.6875 1999.75   2000.8125
 2001.875  2002.9375 2004.     2005.0625 2006.125  2007.1875 2008.25
 2009.3125 2010.375  2011.4375 2012.5    2013.5625 2014.625  2015.6875
 2016.75   2017.8125 2018.875  2019.9375 2021.    ]
44
[1987.         1988.02325581 1989.04651163 1990.06976744 1991.09302326
 1992.11627907 1993.13953488 1994.1627907  1995.18604651 1996.20930233
 1997.23255814 1998.25581395 1999.27906977 2000.30232558 2001.3255814
 2002.34883721 2003.37209302 2004.39534884 2005.41860465 2006.44186047
 2007.46511628 2008.48837209 2009.51162791 2010.53488372 2011.55813953
 2012.58139535 2013.60465116 2014.62790698 2015.65116279 2016.6744186
 2017.69767442 2018.72093023 2019.74418605 2020.76744186 2021.79069767
 2022.81395349 2023.8372093  2024.86046512 2025.88372093 2026.90697674
 2027.93023256 2028.95348837 2029.97674419 2031.        ]
```

A deeper look into data from 2012

[ ]:
```
#look at 2012
df_12 = df.loc['2012']
df_12
```

```
              site transect quadrat  percentCover_allCoral  \
Date
2012-08-01  Tektite        1    T1Q1                  0.253
2012-08-01  Tektite        1    T1Q2                  0.216
2012-08-01  Tektite        1    T1Q3                  0.271
2012-08-01  Tektite        1    T1Q4                  0.095
2012-08-01  Tektite        1    T1Q5                  0.345
2012-08-01  Tektite        1    T1Q6                  0.205
2012-08-01  Tektite        1    T1Q7                  0.212
2012-08-01  Tektite        1    T1Q8                  0.171
2012-08-01  Tektite        1    T1Q9                  0.381
2012-08-01  Tektite        1   T1Q10                  0.266
2012-08-01  Tektite        2    T2Q1                  0.286
2012-08-01  Tektite        2    T2Q2                  0.310
2012-08-01  Tektite        2    T2Q3                  0.146
2012-08-01  Tektite        2    T2Q4                  0.372
2012-08-01  Tektite        2    T2Q5                  0.520
```

```
2012-08-01  Tektite        2    T2Q6                0.535
2012-08-01  Tektite        2    T2Q7                0.490
2012-08-01  Tektite        2    T2Q8                0.335
2012-08-01  Tektite        2    T2Q9                0.286
2012-08-01  Tektite        2   T2Q10                0.482
2012-08-01  Tektite        3    T3Q1                0.268
2012-08-01  Tektite        3    T3Q2                0.348
2012-08-01  Tektite        3    T3Q3                0.482
2012-08-01  Tektite        3    T3Q4                0.367
2012-08-01  Tektite        3    T3Q5                0.465
2012-08-01  Tektite        3    T3Q6                0.332
2012-08-01  Tektite        3    T3Q7                0.460
2012-08-01  Tektite        3    T3Q8                0.396
2012-08-01  Tektite        3    T3Q9                0.365
2012-08-01  Tektite        3   T3Q10                0.475
2012-08-01    Yawzi        1    T1Q1                0.452
2012-08-01    Yawzi        1    T1Q2                0.015
2012-08-01    Yawzi        1    T1Q3                0.186
2012-08-01    Yawzi        1    T1Q4                0.136
2012-08-01    Yawzi        1    T1Q5                0.126
2012-08-01    Yawzi        1    T1Q6                0.010
2012-08-01    Yawzi        1    T1Q7                0.145
2012-08-01    Yawzi        1    T1Q8                0.151
2012-08-01    Yawzi        1    T1Q9                0.110
2012-08-01    Yawzi        1   T1Q10                0.075
2012-08-01    Yawzi        2    T2Q1                0.085
2012-08-01    Yawzi        2    T2Q2                0.030
2012-08-01    Yawzi        2    T2Q3                0.035
2012-08-01    Yawzi        2    T2Q4                0.030
2012-08-01    Yawzi        2    T2Q5                0.105
2012-08-01    Yawzi        2    T2Q6                0.020
2012-08-01    Yawzi        2    T2Q7                0.051
2012-08-01    Yawzi        2    T2Q8                0.000
2012-08-01    Yawzi        2    T2Q9                0.020
2012-08-01    Yawzi        2   T2Q10                0.046
2012-08-01    Yawzi        3    T3Q1                0.025
2012-08-01    Yawzi        3    T3Q2                0.080
2012-08-01    Yawzi        3    T3Q3                0.150
2012-08-01    Yawzi        3    T3Q4                0.005
2012-08-01    Yawzi        3    T3Q5                0.000
2012-08-01    Yawzi        3    T3Q6                0.000
2012-08-01    Yawzi        3    T3Q7                0.020
2012-08-01    Yawzi        3    T3Q8                0.000
2012-08-01    Yawzi        3    T3Q9                0.000
2012-08-01    Yawzi        3   T3Q10                0.030

            percentCover_macroalgae  percentCover_CTB  percentCover_other
Date
```

2012-08-01	0.348	0.328	0.324
2012-08-01	0.276	0.427	0.297
2012-08-01	0.347	0.231	0.422
2012-08-01	0.322	0.548	0.130
2012-08-01	0.380	0.245	0.375
2012-08-01	0.260	0.490	0.250
2012-08-01	0.318	0.460	0.222
2012-08-01	0.372	0.432	0.196
2012-08-01	0.310	0.218	0.472
2012-08-01	0.427	0.271	0.302
2012-08-01	0.196	0.492	0.312
2012-08-01	0.380	0.185	0.435
2012-08-01	0.482	0.327	0.191
2012-08-01	0.442	0.171	0.387
2012-08-01	0.305	0.130	0.565
2012-08-01	0.260	0.115	0.625
2012-08-01	0.343	0.152	0.505
2012-08-01	0.380	0.180	0.440
2012-08-01	0.397	0.307	0.296
2012-08-01	0.365	0.102	0.533
2012-08-01	0.414	0.288	0.298
2012-08-01	0.359	0.263	0.378
2012-08-01	0.296	0.171	0.533
2012-08-01	0.347	0.256	0.397
2012-08-01	0.465	0.045	0.490
2012-08-01	0.457	0.156	0.387
2012-08-01	0.369	0.131	0.500
2012-08-01	0.254	0.157	0.589
2012-08-01	0.535	0.085	0.380
2012-08-01	0.335	0.150	0.515
2012-08-01	0.317	0.226	0.457
2012-08-01	0.535	0.430	0.035
2012-08-01	0.492	0.261	0.247
2012-08-01	0.495	0.278	0.227
2012-08-01	0.518	0.281	0.201
2012-08-01	0.615	0.340	0.045
2012-08-01	0.465	0.220	0.315
2012-08-01	0.452	0.281	0.267
2012-08-01	0.510	0.330	0.160
2012-08-01	0.425	0.440	0.135
2012-08-01	0.593	0.322	0.085
2012-08-01	0.545	0.394	0.061
2012-08-01	0.565	0.370	0.065
2012-08-01	0.538	0.360	0.102
2012-08-01	0.205	0.625	0.170
2012-08-01	0.315	0.640	0.045
2012-08-01	0.525	0.348	0.127
2012-08-01	0.291	0.561	0.148

```
2012-08-01                    0.161              0.724              0.115
2012-08-01                    0.205              0.672              0.123
2012-08-01                    0.660              0.225              0.115
2012-08-01                    0.520              0.340              0.140
2012-08-01                    0.370              0.405              0.225
2012-08-01                    0.577              0.378              0.045
2012-08-01                    0.435              0.505              0.060
2012-08-01                    0.230              0.765              0.005
2012-08-01                    0.337              0.603              0.060
2012-08-01                    0.377              0.578              0.045
2012-08-01                    0.320              0.510              0.170
2012-08-01                    0.260              0.685              0.055
```

Finding average over quadrants

[ ]:
```
df
```

```
              site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Tektite        1    T1Q1                  0.165
1987-12-01  Tektite        1    T1Q2                  0.100
1987-12-01  Tektite        1    T1Q3                  0.135
1987-12-01  Tektite        1    T1Q4                  0.155
1987-12-01  Tektite        1    T1Q5                  0.190
…              …      …     …                     …
2021-07-01    Yawzi       nd       6                  0.000
2021-07-01    Yawzi       nd       7                  0.000
2021-07-01    Yawzi       nd       8                  0.005
2021-07-01    Yawzi       nd       9                  0.005
2021-07-01    Yawzi       nd      10                  0.035

            percentCover_macroalgae  percentCover_CTB  percentCover_other
Date
1987-12-01                    0.035             0.060               0.905
1987-12-01                    0.060             0.740               0.200
1987-12-01                    0.035             0.450               0.515
1987-12-01                    0.065             0.615               0.320
1987-12-01                    0.040             0.480               0.480
…                             …                 …                   …
2021-07-01                    0.300             0.500               0.200
2021-07-01                    0.285             0.455               0.260
2021-07-01                    0.350             0.350               0.300
2021-07-01                    0.420             0.425               0.155
2021-07-01                    0.340             0.385               0.275

[2264 rows x 7 columns]
```

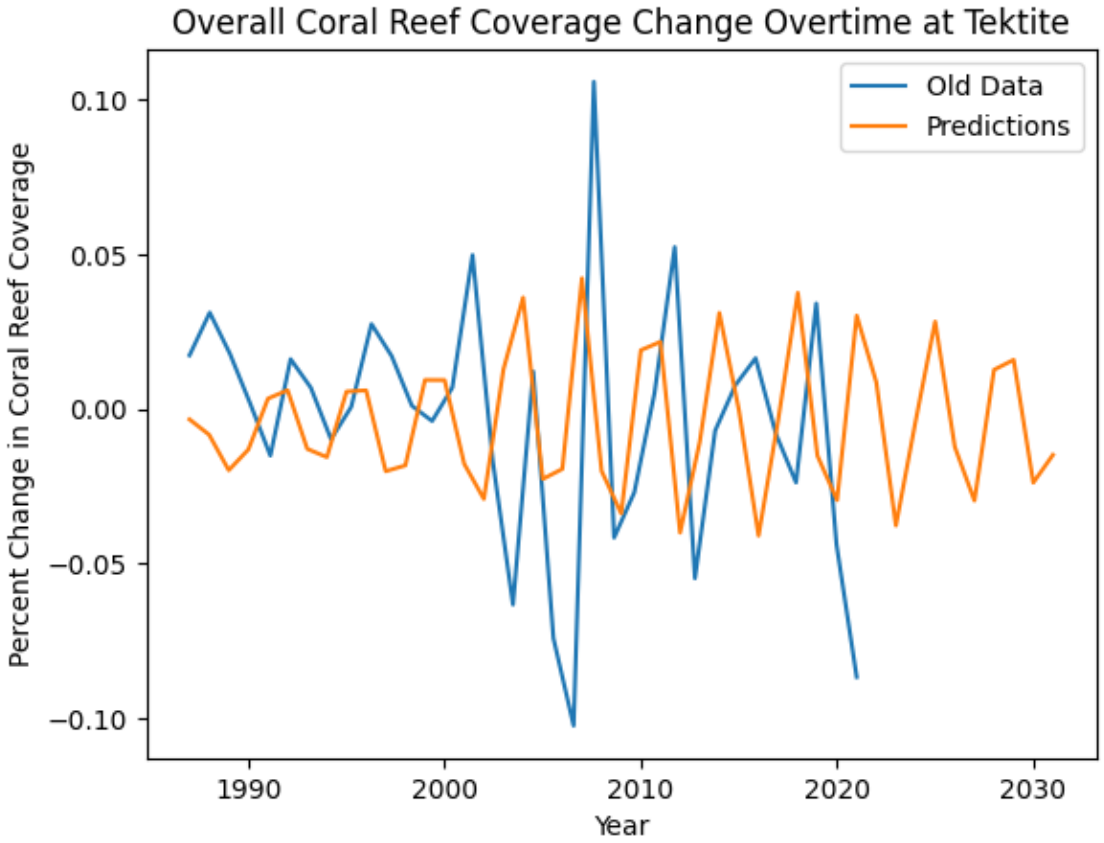
[ ]:
```
print(type(df.index))
```

```
<class 'pandas.core.indexes.datetimes.DatetimeIndex'>
```

[ ]:
```
df['year_column'] = df.index.year
df
```

```
               site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Tektite        1    T1Q1                  0.165
1987-12-01  Tektite        1    T1Q2                  0.100
1987-12-01  Tektite        1    T1Q3                  0.135
1987-12-01  Tektite        1    T1Q4                  0.155
1987-12-01  Tektite        1    T1Q5                  0.190
…              …       …     …                      …
2021-07-01    Yawzi       nd       6                  0.000
2021-07-01    Yawzi       nd       7                  0.000
2021-07-01    Yawzi       nd       8                  0.005
2021-07-01    Yawzi       nd       9                  0.005
2021-07-01    Yawzi       nd      10                  0.035

            percentCover_macroalgae  percentCover_CTB  percentCover_other  \
Date
1987-12-01                    0.035             0.060               0.905
1987-12-01                    0.060             0.740               0.200
1987-12-01                    0.035             0.450               0.515
1987-12-01                    0.065             0.615               0.320
1987-12-01                    0.040             0.480               0.480
…                             …                 …                   …
2021-07-01                    0.300             0.500               0.200
2021-07-01                    0.285             0.455               0.260
2021-07-01                    0.350             0.350               0.300
2021-07-01                    0.420             0.425               0.155
2021-07-01                    0.340             0.385               0.275

            year_column
Date
1987-12-01         1987
1987-12-01         1987
1987-12-01         1987
1987-12-01         1987
1987-12-01         1987
…                   …
2021-07-01         2021
2021-07-01         2021
2021-07-01         2021
2021-07-01         2021
2021-07-01         2021

[2264 rows x 8 columns]
```

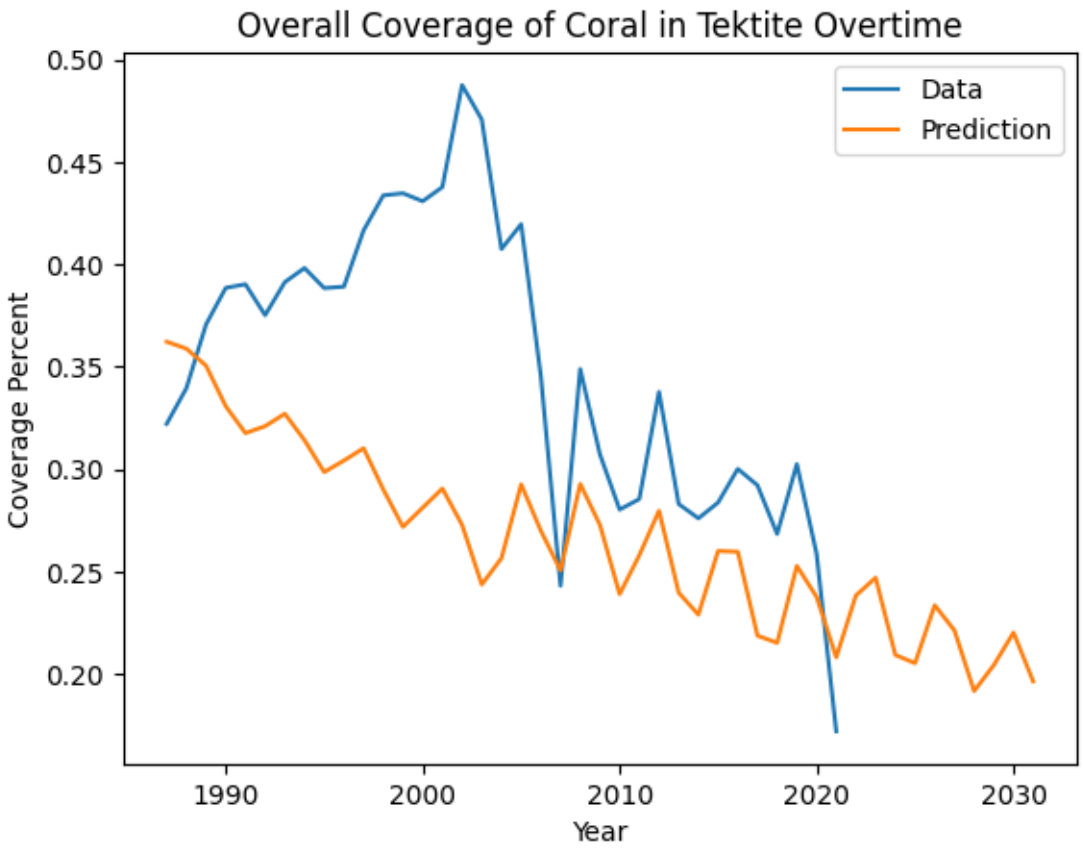
[ ]:
```
df_tektite = df[df['site']=='Tektite']

#create a dataframe of overall coverage at the tektite site
avg_percent_cover_allcoral_tektite = df_tektite.
  ↪groupby('year_column')['percentCover_allCoral'].mean()
aic, final,std = sm_arma(avg_percent_cover_allcoral_tektite)

#plot the differenced dataset as well as the predictions
z = np.diff(avg_percent_cover_allcoral_tektite)
time = np.linspace(1987,2021,len(z))
time2 = np.linspace(1987,2021+10,len(final))
plt.plot(time,z,label='Old Data')
plt.plot(time2,final,label='Predictions')
plt.xlabel('Year')
plt.ylabel('Percent Change in Coral Reef Coverage')
plt.title('Overall Coral Reef Coverage Change Overtime at Tektite')
plt.legend()
plt.show()
```



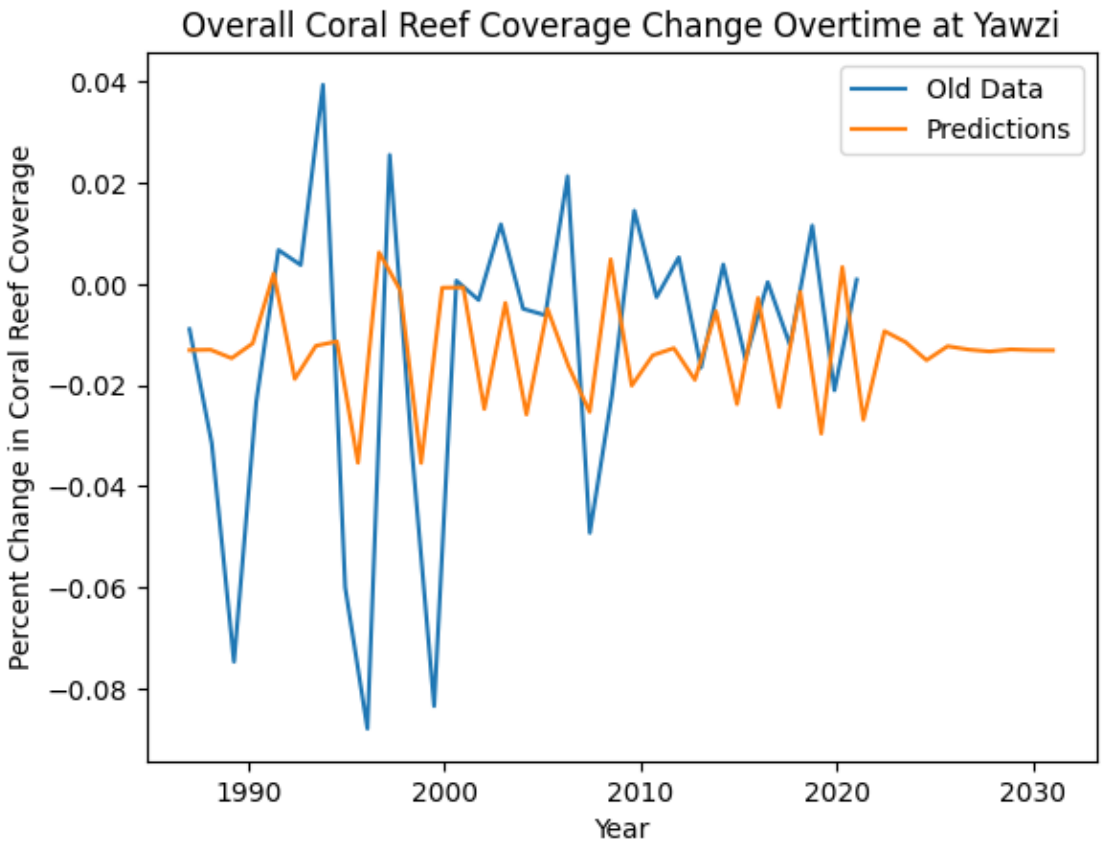
[ ]:
```
avg_percent_cover_allcoral_tektite_prediction = np.array([np.sum(final[:i]) for␣
↪i in range(len(final))])
avg_percent_cover_allcoral_tektite_prediction += np.
↪mean(avg_percent_cover_allcoral_tektite.iloc[:5])
print(avg_percent_cover_allcoral_tektite_prediction)
```

```
[0.36229726 0.35893842 0.35060451 0.33079713 0.3176874  0.32106111
 0.32713903 0.3142278  0.29864262 0.30426879 0.31028745 0.29014455
 0.27189344 0.28126685 0.29061048 0.27286049 0.24374652 0.25658935
 0.29265615 0.26999669 0.2505946  0.29289496 0.27283683 0.23903216
 0.25804377 0.2797686  0.23976714 0.22903824 0.26013433 0.25973636
 0.21877271 0.21526882 0.25284276 0.23771549 0.2082041  0.23842318
 0.24705575 0.20935466 0.20533173 0.23361293 0.22135952 0.19170534
 0.20438692 0.22028419 0.19645699]
```

[ ]:
```
new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_tektite))
new_time2 = np.
↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_tektite_prediction))
plt.plot(new_time,avg_percent_cover_allcoral_tektite,label='Data')
plt.
↪plot(new_time2,avg_percent_cover_allcoral_tektite_prediction,label='Prediction')
plt.title('Overall Coverage of Coral in Tektite Overtime')
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()
plt.show()
```



[ ]:
```
df_yawzi = df[df['site']=='Yawzi']

#create a dataframe of overall coverage at the yawzi site
avg_percent_cover_allcoral_yawzi = df_yawzi.
 ↪groupby('year_column')['percentCover_allCoral'].mean()
aic, final,std = sm_arma(avg_percent_cover_allcoral_yawzi)

#plot the differenced dataset as well as the predictions
z = np.diff(avg_percent_cover_allcoral_yawzi)
time = np.linspace(1987,2021,len(z))
time2 = np.linspace(1987,2021+10,len(final))
plt.plot(time,z,label='Old Data')
plt.plot(time2,final,label='Predictions')
plt.xlabel('Year')
plt.ylabel('Percent Change in Coral Reef Coverage')
plt.title('Overall Coral Reef Coverage Change Overtime at Yawzi')
plt.legend()
plt.show()
```

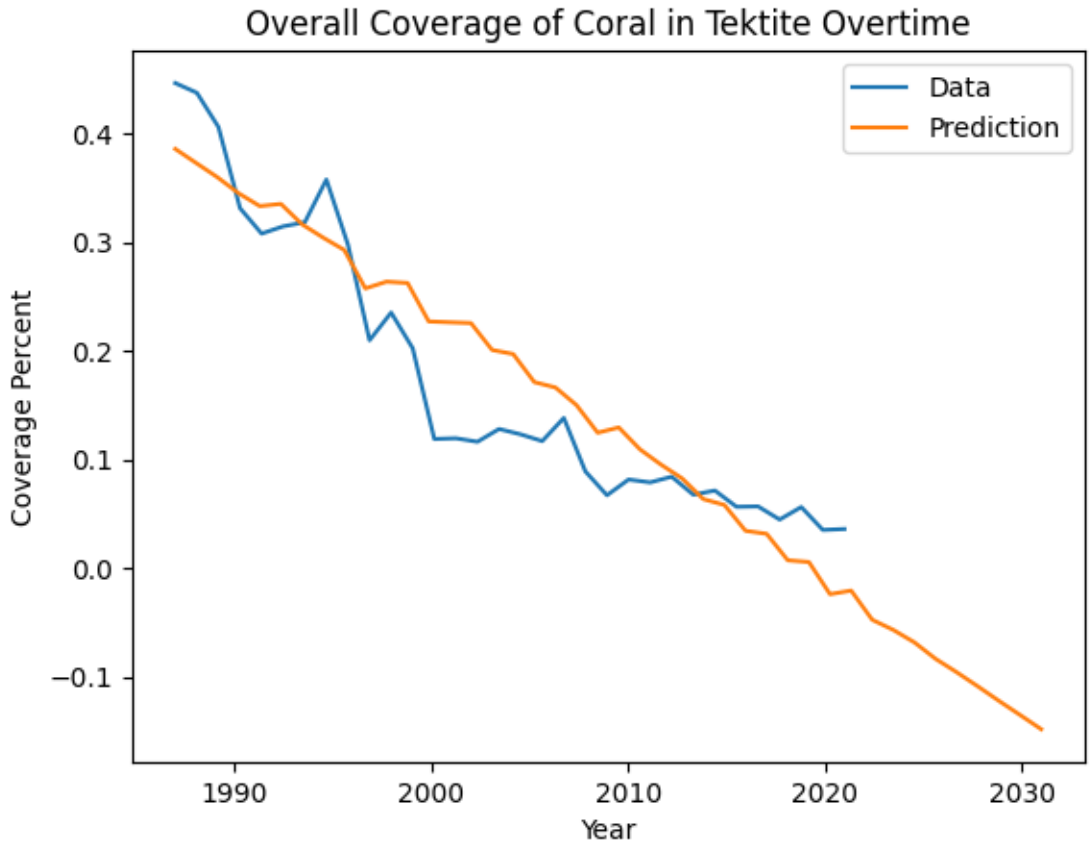


[ ]:
```
avg_percent_cover_allcoral_yawzi_prediction = np.array([np.sum(final[:i]) for i␣
↪in range(len(final))])
avg_percent_cover_allcoral_yawzi_prediction += np.
↪mean(avg_percent_cover_allcoral_yawzi.iloc[:5])
print(avg_percent_cover_allcoral_yawzi_prediction)
```

```
[ 0.38555971  0.37246817  0.35944836  0.34474639  0.33298091  0.33501673
  0.3162277   0.30397839  0.29259392  0.25723772  0.26344368  0.26222808
  0.2268333   0.2260619   0.2253486   0.20061226  0.1968574   0.1710331
  0.16607613  0.14979324  0.12447441  0.12930944  0.10919155  0.09508222
  0.08237367  0.06340405  0.05799597  0.03422338  0.03143972  0.00708826
  0.00549574 -0.02409857 -0.02078671 -0.04766335 -0.05700943 -0.06851678
 -0.08360924 -0.09593488 -0.10893062 -0.12229362 -0.13524803 -0.14834663]
```

[ ]:
```
new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_yawzi))
new_time2 = np.
↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_yawzi_prediction))
plt.plot(new_time,avg_percent_cover_allcoral_yawzi,label='Data')
plt.
↪plot(new_time2,avg_percent_cover_allcoral_yawzi_prediction,label='Prediction')
```

```
plt.title('Overall Coverage of Coral in Tektite Overtime')
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()
plt.show()
```



We also want to look at how ocean temperature affects the growth and loss of coral reefs. We observe an additional dataset recording the temperatures of the ocean in the U.S. Virgin Islands.

2 Classical Decomposition of Coral Coverage

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import acf
from sklearn.linear_model import LinearRegression
import scipy.linalg as la
from matplotlib.dates import YearLocator, MonthLocator, DateFormatter
```

[ ]:
```
coral = pd.read_csv('coral_dataset.csv', index_col='Date')
coral = coral[coral['percentCover_CTB'] != 'nd'] #drop all of the rows with␣
 ↪these entries and then print out the amount of rows in our dataset
coral = coral[coral['percentCover_macroalgae'] != 'nd']
coral = coral[coral['percentCover_allCoral'] != 'nd']
```

[ ]:
```
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
 ↪'percentCover_allCoral'] #get a list of the columns that we want to cast
coral[columns_to_cast] = coral[columns_to_cast].astype(float) #cast these␣
 ↪columns as floats and divide by 100 to make the percents
coral[columns_to_cast] = coral[columns_to_cast]/100
coral['percentCover_other'] = 1 -␣
 ↪(coral['percentCover_CTB']+coral['percentCover_macroalgae']) #add in another␣
 ↪column for other types of coral
```

[ ]:
```
# convert the date column to date time objects
coral.index = pd.to_datetime(coral.index)
coral
```

```
              site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Tektite        1    T1Q1                  0.165
1987-12-01  Tektite        1    T1Q2                  0.100
1987-12-01  Tektite        1    T1Q3                  0.135
1987-12-01  Tektite        1    T1Q4                  0.155
1987-12-01  Tektite        1    T1Q5                  0.190
…              …      …     …                      …
2021-07-01    Yawzi       nd       6                  0.000
2021-07-01    Yawzi       nd       7                  0.000
2021-07-01    Yawzi       nd       8                  0.005
2021-07-01    Yawzi       nd       9                  0.005
2021-07-01    Yawzi       nd      10                  0.035

            percentCover_macroalgae  percentCover_CTB  percentCover_other
Date
1987-12-01                    0.035             0.060               0.905
1987-12-01                    0.060             0.740               0.200
1987-12-01                    0.035             0.450               0.515
1987-12-01                    0.065             0.615               0.320
1987-12-01                    0.040             0.480               0.480
…                             …                 …                   …
2021-07-01                    0.300             0.500               0.200
2021-07-01                    0.285             0.455               0.260
2021-07-01                    0.350             0.350               0.300
2021-07-01                    0.420             0.425               0.155
2021-07-01                    0.340             0.385               0.275

[2264 rows x 7 columns]
```

[ ]:
```
all_coral = coral['percentCover_allCoral']
```

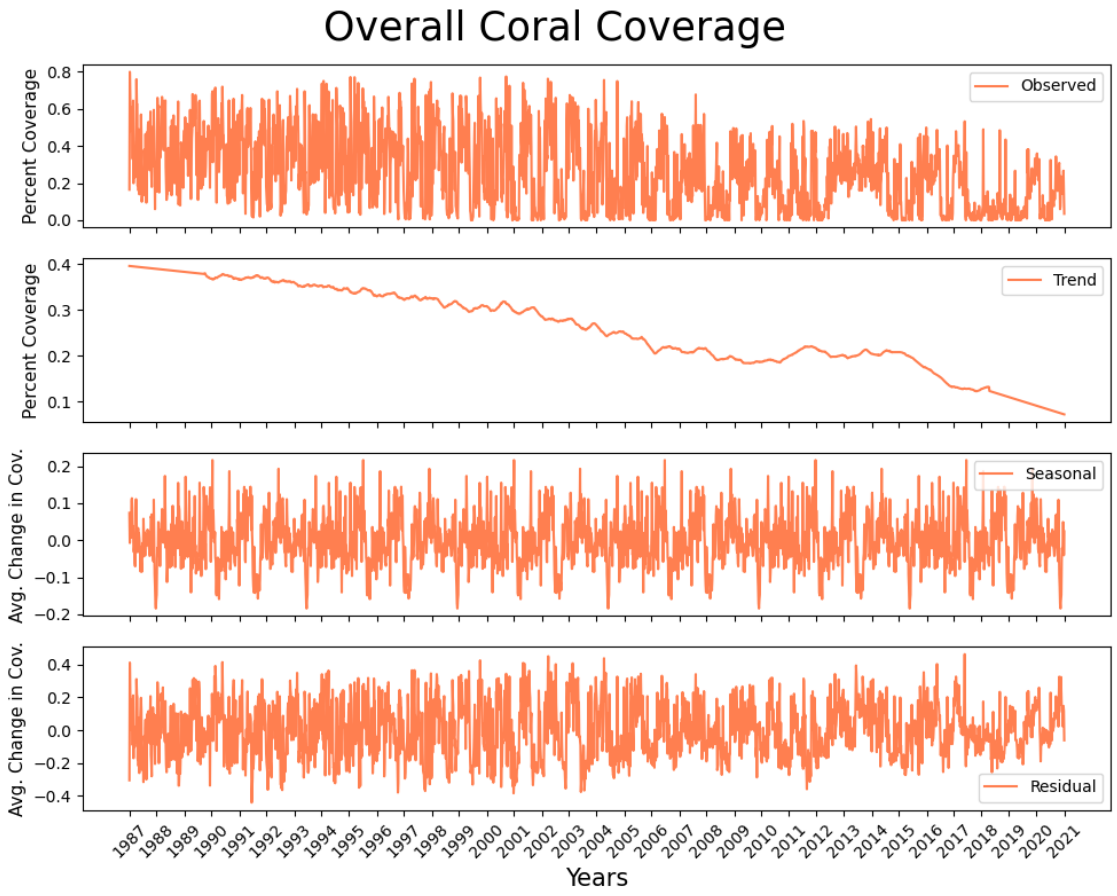
[ ]:
```
all_coral = all_coral.sort_index()
all_coral
```

[ ]:
```
#all_coral_yawzi
all_coral_tekite = coral[coral['site'] == 'Tektite']
all_coral_tekite = all_coral_tekite['percentCover_allCoral']
all_coral_tekite.astype('float')

all_coral_yawzi = coral[coral['site'] == 'Yawzi']
all_coral_yawzi = all_coral_yawzi['percentCover_allCoral']
all_coral_yawzi.astype('float')
```

```
Date
1987-12-01    0.570
1987-12-01    0.230
1987-12-01    0.800
1987-12-01    0.670
1987-12-01    0.481
              …
2021-07-01    0.000
2021-07-01    0.000
2021-07-01    0.005
2021-07-01    0.005
2021-07-01    0.035
Name: percentCover_allCoral, Length: 1116, dtype: float64
```

[ ]:
```
# Decompose the overall coral data
trends = seasonal_decompose(all_coral.sort_index().values, model='additive',␣
 ↪period=365, extrapolate_trend='freq')

# Generate a date range for the x-axis
date_range = pd.date_range(start='1987-12-01', periods=2264, freq='D')

# Calculate indices for each year
start_year = 1987
end_year = 2021
years = range(start_year, end_year + 1)
year_indices = np.linspace(0, len(date_range) - 1, num=len(years), dtype=int)

# Create subplots
fig, axes = plt.subplots(4, 1, figsize=(10, 8), sharex='col')  # 4 rows, 1␣
 ↪column, shared x-axis

# Plot observed component
axes[0].plot(trends.observed, label='Observed', color='coral')
```

```
axes[0].set_ylabel('Percent Coverage', fontsize = 11)
axes[0].legend()

#plot the Trend
axes[1].plot(trends.trend, label='Trend', color='coral')
axes[1].set_ylabel('Percent Coverage', fontsize = 11)
axes[1].legend()

#plot the seasonality
axes[2].plot(trends.seasonal, label='Seasonal', color='coral')
axes[2].set_ylabel('Avg. Change in Cov.', fontsize = 11)
axes[2].legend()

#plot the Residual
axes[3].plot(trends.resid, label='Residual', color='coral')
axes[3].set_ylabel('Avg. Change in Cov.', fontsize = 11)
axes[3].legend()

# Set common x-axis properties
for ax in axes:
    ax.set_xticks(year_indices)
    ax.set_xticklabels(years, rotation=45)

# Add a super title
fig.suptitle('Overall Coral Coverage', fontsize=25)

# Set a common Y-Axis label
#fig.text(0.01, 0.5, 'Percent Coral Coverage', va='center', ha='center',␣
 ↪rotation='vertical', fontsize=15)
plt.xlabel('Years', fontsize = 15)

# Optional: Adjust the layout
plt.subplots_adjust(left=0.15, bottom=0.1, right=0.9, top=0.9, hspace=0.5)
plt.tight_layout()

# Save the figure
plt.savefig('Overall_decomp.png')

# Show the plot
plt.show()
```

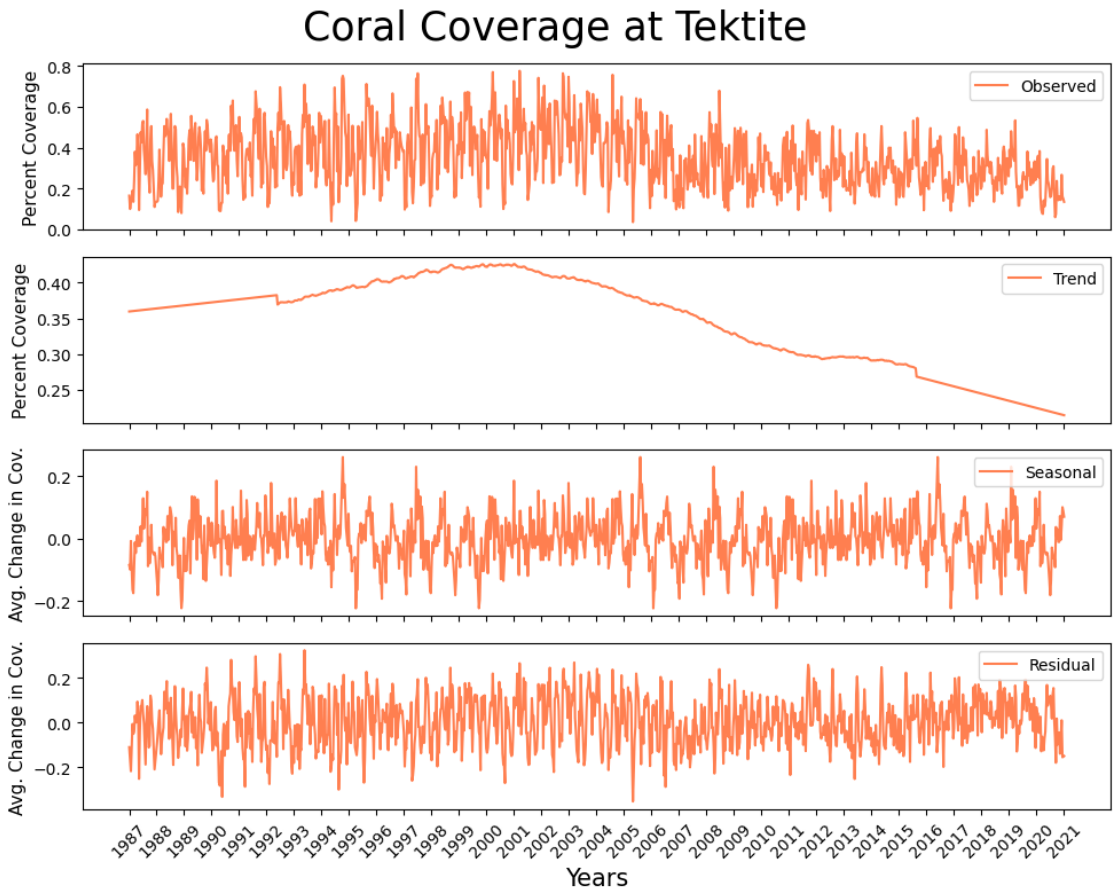


[ ]:
```
all_coral_yawzi.sort_index()
```

```
Date
1987-12-01    0.570
1987-12-01    0.420
1987-12-01    0.225
1987-12-01    0.140
1987-12-01    0.490
              …
2021-07-01    0.060
2021-07-01    0.015
2021-07-01    0.325
2021-07-01    0.045
2021-07-01    0.035
Name: percentCover_allCoral, Length: 1116, dtype: float64
```

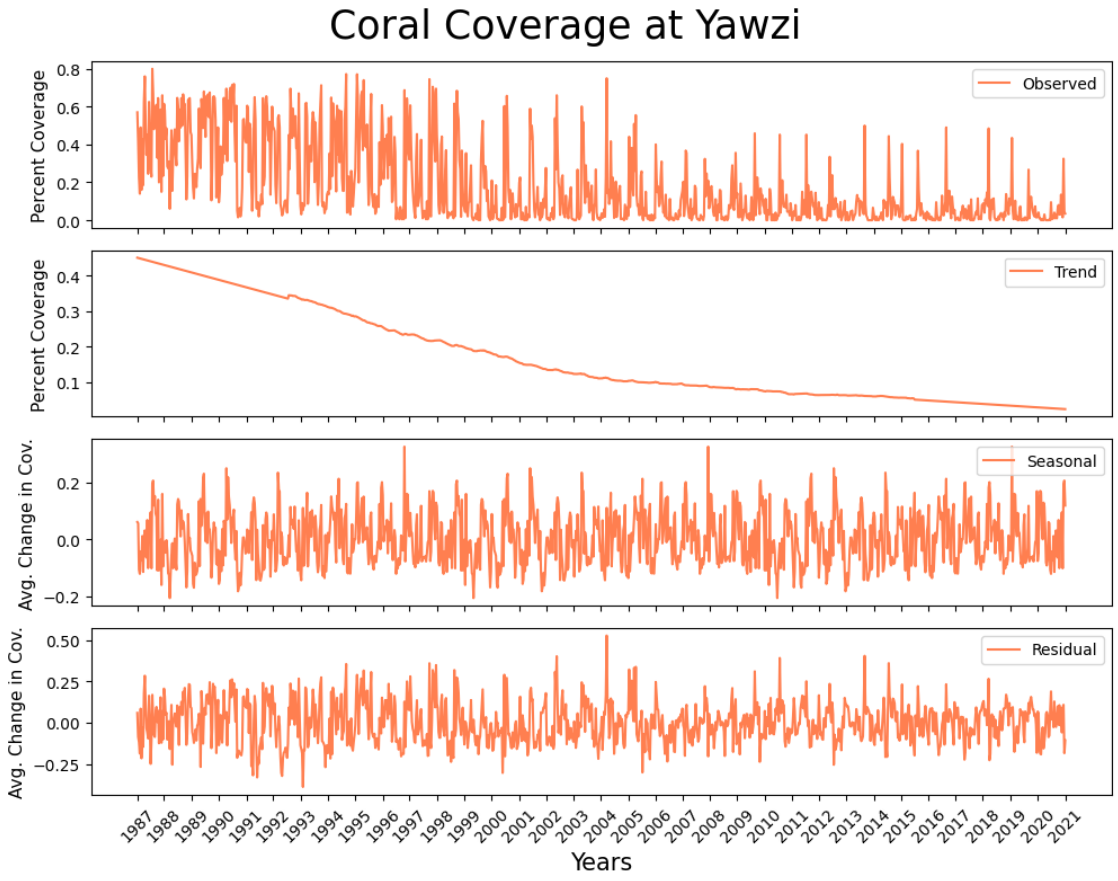
[ ]:
```
# Decompose the coral data at Tektite
trends = seasonal_decompose(all_coral_tekite.sort_index().values,␣
  ↪model='additive', period=365, extrapolate_trend='freq')
```

```
# Generate a date range for the x-axis
date_range = pd.date_range(start='1987-12-01', periods=1148, freq='D')

# Calculate indices for each year
start_year = 1987
end_year = 2021
years = range(start_year, end_year + 1)
year_indices = np.linspace(0, len(date_range) - 1, num=len(years), dtype=int)

# Create subplots
fig, axes = plt.subplots(4, 1, figsize=(10, 8), sharex=True)  # 4 rows, 1␣
 ↪column, shared x-axis

# Plot each component
axes[0].plot(trends.observed, label='Observed', color='coral')
axes[0].set_ylabel('Percent Coverage', fontsize = 11)
axes[0].legend()

axes[1].plot(trends.trend, label='Trend', color='coral')
axes[1].set_ylabel('Percent Coverage', fontsize = 11)
axes[1].legend()

axes[2].plot(trends.seasonal, label='Seasonal', color='coral')
axes[2].set_ylabel('Avg. Change in Cov.', fontsize = 11)
axes[2].legend()

axes[3].plot(trends.resid, label='Residual', color='coral')
axes[3].set_ylabel('Avg. Change in Cov.', fontsize = 11)
axes[3].legend()

# Set common x-axis properties
for ax in axes:
    ax.set_xticks(year_indices)
    ax.set_xticklabels(years, rotation=45)

# Add plot title and descriptions
fig.suptitle('Coral Coverage at Tektite', fontsize=25)
plt.xlabel('Years', fontsize = 15)

#Adjust the layout
plt.subplots_adjust(left=0.15, bottom=0.2, right=0.9, top=0.9, hspace=0.5)
plt.tight_layout()

# Save the figure and display it
plt.savefig('Tektite_decomp.png')
plt.show()
```



[ ]:
```
# Decompose the yawzi coral data
trends = seasonal_decompose(all_coral_yawzi.sort_index().values,␣
 ↪model='additive', period=365, extrapolate_trend='freq')

# Generate a date range for the x-axis
date_range = pd.date_range(start='1987-12-01', periods=1116, freq='D')

# Calculate indices for each year
start_year = 1987
end_year = 2021
years = range(start_year, end_year + 1)
year_indices = np.linspace(0, len(date_range) - 1, num=len(years), dtype=int)

# Create subplots
fig, axes = plt.subplots(4, 1, figsize=(10, 8), sharex=True)  # 4 rows, 1␣
 ↪column, shared x-axis

# Plot each component
axes[0].plot(trends.observed, label='Observed', color='coral')
```

```
axes[0].set_ylabel('Percent Coverage', fontsize = 11)
axes[0].legend()

axes[1].plot(trends.trend, label='Trend', color='coral')
axes[1].set_ylabel('Percent Coverage', fontsize = 11)
axes[1].legend()

axes[2].plot(trends.seasonal, label='Seasonal', color='coral')
axes[2].set_ylabel('Avg. Change in Cov.', fontsize = 11)
axes[2].legend()

axes[3].plot(trends.resid, label='Residual', color='coral')
axes[3].set_ylabel('Avg. Change in Cov.', fontsize = 11)
axes[3].legend()

# Set common x-axis properties
for ax in axes:
    ax.set_xticks(year_indices)
    ax.set_xticklabels(years, rotation=45)

# Add a title and plot descriptions
fig.suptitle('Coral Coverage at Yawzi', fontsize=25)
plt.xlabel('Years', fontsize = 15)

#Adjust the layout
plt.subplots_adjust(left=0.15, bottom=0.1, right=0.9, top=0.9, hspace=0.5)
plt.tight_layout()

# Save the figure and show the plot
plt.savefig('Yawzi_decomp.png')
plt.show()
```



[ ]:
```
#read in the temperature data and set the index to datetime objects
sea_df = pd.read_csv("seawatertemp (1).csv", index_col = "Date") #read in the␣
 ↪dataset
sea_df.index = pd.to_datetime(sea_df.index)

#get the years from the dataframe
sea_df['year_column'] = sea_df.index.year
sea_df = sea_df[sea_df['Site'].isin(['Yawzi_NPS','Yawzi_9m'])]
specific_value_count = sea_df['Temperature'].value_counts()['nd']

#find empty values, and convert temperature values to floats
sea_df = sea_df[sea_df['Temperature'] != 'nd']
sea_df['Temperature'] = sea_df['Temperature'].astype(float)

#aggregate temperature per year
avg_sea_temp = sea_df.groupby('year_column')['Temperature'].mean()
avg_sea_temp
```

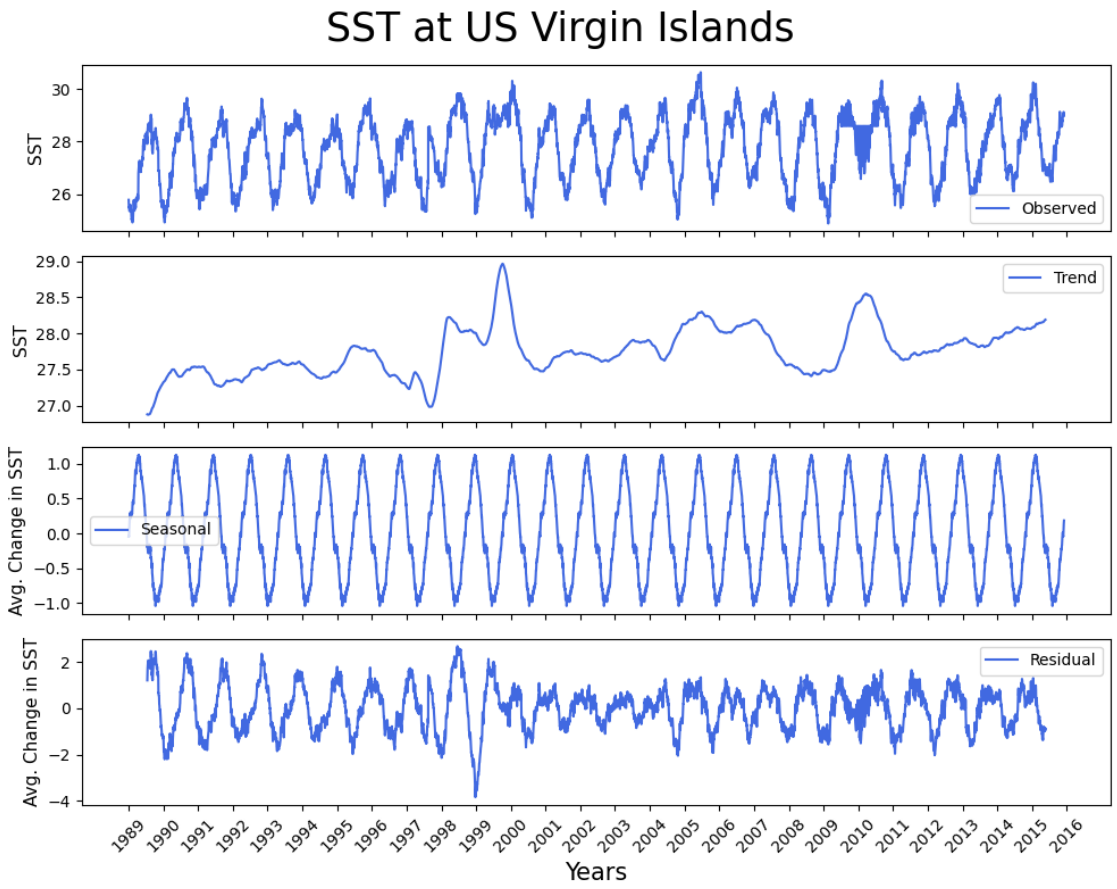
```
year_column
1989     27.153627
```

```
1990    27.421425
1991    27.324986
1992    27.485410
1993    27.564384
1994    27.395973
1995    27.806966
1996    27.342350
1997    27.808127
1998    28.170986
1999    27.994685
2000    27.482548
2001    27.725918
2002    27.691342
2003    27.361349
2004    28.469801
2005    28.213955
2006    28.030083
2007    28.029944
2008    27.474306
2009    27.650138
2010    28.339280
2011    27.727008
2012    27.858338
2013    27.045025
2014    28.770303
2015    28.070670
2016    27.617150
Name: Temperature, dtype: float64
```

[ ]:
```
sea_temp_trends = seasonal_decompose(sea_df_new.values, model = 'additive',␣
↪period = 365)
```

[ ]:
```
# Generate a date range for the x-axis
date_range = pd.date_range(start='1989-01-01', periods=9163, freq='D')
# Calculate indices for each year
start_year = 1989
end_year = 2016
years = range(start_year, end_year + 1)
year_indices = np.linspace(0, len(date_range) - 1, num=len(years), dtype=int)

# Create subplots
fig, axes = plt.subplots(4, 1, figsize=(10, 8), sharex=True)  # 4 rows, 1␣
↪column, shared x-axis

# Plot each component
axes[0].plot(sea_temp_trends.observed, label='Observed', color='royalblue')
axes[0].set_ylabel('SST', fontsize = 11)
```

```
axes[0].legend()

axes[1].plot(sea_temp_trends.trend, label='Trend', color='royalblue')
axes[1].set_ylabel('SST', fontsize = 11)
axes[1].legend()

axes[2].plot(sea_temp_trends.seasonal, label='Seasonal', color='royalblue')
axes[2].set_ylabel('Avg. Change in SST', fontsize = 11)
axes[2].legend()

axes[3].plot(sea_temp_trends.resid, label='Residual', color='royalblue')
axes[3].set_ylabel('Avg. Change in SST', fontsize = 11)
axes[3].legend()

# Set common x-axis properties
for ax in axes:
    ax.set_xticks(year_indices)
    ax.set_xticklabels(years, rotation=45)

# Add title, labels and adjust layout
fig.suptitle('SST at US Virgin Islands', fontsize=25)
plt.subplots_adjust(left=0.15, bottom=0.2, right=0.9, top=0.9, hspace=0.5)
plt.xlabel('Years', fontsize = 15)
plt.tight_layout()

# Save the figure
plt.savefig('new_sea_water.png')

# Show the plot
plt.show()
```



3 ARIMA Train and Test Split

In this notebook, we test our ARIMA Model and find it's accuracy scores

[ ]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import accuracy_score
```

[ ]:
```
coral = pd.read_csv('coral_dataset.csv', index_col='Date')
coral = coral[coral['percentCover_CTB'] != 'nd'] #drop all of the rows with␣
 ↪these entries and then print out the amount of rows in our dataset
coral = coral[coral['percentCover_macroalgae'] != 'nd']
coral = coral[coral['percentCover_allCoral'] != 'nd']
```

[ ]:
```
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
 ↪'percentCover_allCoral'] #get a list of the columns that we want to cast
coral[columns_to_cast] = coral[columns_to_cast].astype(float) #cast these␣
 ↪columns as floats and divide by 100 to make the percents
coral[columns_to_cast] = coral[columns_to_cast]/100
```

```
coral['percentCover_other'] = 1 -␣
↪(coral['percentCover_CTB']+coral['percentCover_macroalgae']) #add in another␣
↪column for other types of coral
```

[ ]:
```
# convert the date column to date time objects
coral.index = pd.to_datetime(coral.index)
coral
```

```
              site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Tektite        1    T1Q1                  0.165
1987-12-01  Tektite        1    T1Q2                  0.100
1987-12-01  Tektite        1    T1Q3                  0.135
1987-12-01  Tektite        1    T1Q4                  0.155
1987-12-01  Tektite        1    T1Q5                  0.190
…              …       …      …                      …
2021-07-01    Yawzi       nd       6                  0.000
2021-07-01    Yawzi       nd       7                  0.000
2021-07-01    Yawzi       nd       8                  0.005
2021-07-01    Yawzi       nd       9                  0.005
2021-07-01    Yawzi       nd      10                  0.035

            percentCover_macroalgae  percentCover_CTB  percentCover_other
Date
1987-12-01                    0.035             0.060               0.905
1987-12-01                    0.060             0.740               0.200
1987-12-01                    0.035             0.450               0.515
1987-12-01                    0.065             0.615               0.320
1987-12-01                    0.040             0.480               0.480
…                             …                 …                   …
2021-07-01                    0.300             0.500               0.200
2021-07-01                    0.285             0.455               0.260
2021-07-01                    0.350             0.350               0.300
2021-07-01                    0.420             0.425               0.155
2021-07-01                    0.340             0.385               0.275

[2264 rows x 7 columns]
```

[ ]:
```
explore_df = coral[coral['site'] == 'Yawzi']
explore_df = explore_df[explore_df['quadrat'] == 'T1Q1']
explore_df
```

```
             site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Yawzi        1    T1Q1                  0.570
1988-03-01  Yawzi        1    T1Q1                  0.490
1988-07-01  Yawzi        1    T1Q1                  0.440
1989-04-01  Yawzi        1    T1Q1                  0.480
```

1989-10-01	Yawzi	1	T1Q1	0.510
1990-09-01	Yawzi	1	T1Q1	0.520
1991-03-01	Yawzi	1	T1Q1	0.450
1992-05-01	Yawzi	1	T1Q1	0.517
1993-06-01	Yawzi	1	T1Q1	0.648
1994-08-01	Yawzi	1	T1Q1	0.544
1995-05-01	Yawzi	1	T1Q1	0.374
1996-05-01	Yawzi	1	T1Q1	0.687
1997-05-01	Yawzi	1	T1Q1	0.745
1998-08-01	Yawzi	1	T1Q1	0.616
1999-08-01	Yawzi	1	T1Q1	0.320
2000-08-01	Yawzi	1	T1Q1	0.603
2001-08-01	Yawzi	1	T1Q1	0.589
2002-08-01	Yawzi	1	T1Q1	0.539
2003-08-01	Yawzi	1	T1Q1	0.601
2004-08-01	Yawzi	1	T1Q1	0.417
2005-08-01	Yawzi	1	T1Q1	0.441
2006-08-01	Yawzi	1	T1Q1	0.400
2007-08-01	Yawzi	1	T1Q1	0.368
2008-08-01	Yawzi	1	T1Q1	0.324
2009-08-01	Yawzi	1	T1Q1	0.356
2010-08-01	Yawzi	1	T1Q1	0.459
2011-08-01	Yawzi	1	T1Q1	0.452
2012-08-01	Yawzi	1	T1Q1	0.452
2013-08-01	Yawzi	1	T1Q1	0.335
2019-08-14	Yawzi	1	T1Q1	0.490
2019-08-15	Yawzi	1	T1Q1	0.485
2019-08-16	Yawzi	1	T1Q1	0.051
2017-07-01	Yawzi	1	T1Q1	0.500
2017-11-01	Yawzi	1	T1Q1	0.444

Date	percentCover_macroalgae	percentCover_CTB	percentCover_other
1987-12-01	0.005	0.215	0.780
1988-03-01	0.000	0.250	0.750
1988-07-01	0.005	0.290	0.705
1989-04-01	0.035	0.235	0.730
1989-10-01	0.040	0.145	0.815
1990-09-01	0.085	0.075	0.840
1991-03-01	0.000	0.085	0.915
1992-05-01	0.000	0.189	0.811
1993-06-01	0.075	0.140	0.785
1994-08-01	0.040	0.175	0.785
1995-05-01	0.030	0.340	0.630
1996-05-01	0.011	0.097	0.892
1997-05-01	0.000	0.225	0.775
1998-08-01	0.025	0.100	0.875
1999-08-01	0.249	0.110	0.641

2000-08-01	0.005	0.109	0.886
2001-08-01	0.313	0.313	0.374
2002-08-01	0.202	0.078	0.720
2003-08-01	0.020	0.121	0.859
2004-08-01	0.006	0.202	0.792
2005-08-01	0.000	0.531	0.469
2006-08-01	0.073	0.395	0.532
2007-08-01	0.032	0.542	0.426
2008-08-01	0.000	0.497	0.503
2009-08-01	0.021	0.543	0.436
2010-08-01	0.209	0.163	0.628
2011-08-01	0.156	0.357	0.487
2012-08-01	0.317	0.226	0.457
2013-08-01	0.350	0.295	0.355
2019-08-14	0.240	0.195	0.565
2019-08-15	0.350	0.150	0.500
2019-08-16	0.556	0.107	0.337
2017-07-01	0.116	0.369	0.515
2017-11-01	0.187	0.359	0.454

[ ]:
```
percent_cover_all_coral = np.array(explore_df['percentCover_allCoral'])
```

[ ]:
```
def sm_arma(array, p_max=3, q_max=3, n=10):
    """
    Build an ARMA model with statsmodel and
    predict future n values.

    Parameters:
        filename (str): data filename
        p_max (int): maximum order of autoregressive model
        q_max (int): maximum order of moving average model
        n (int): number of values to predict

    Return:
        aic (float): aic of optimal model
    """
    #take a difference quotient and separate into train and test sets
    z = np.diff(array)
    train = z[:30]
    test = z[30:]

    #initialize attributes of our data
    best_aic = np.inf
    best_model = None
    best_p = None
    best_q = None
    best_std = None
```

```
    #perform a grid search on p and q
    for p in range(1,p_max+1):
        for q in range(1,q_max+1):

            #fit the model on the training data
            model = ARIMA(train,order=(p,0,q),trend='c').
↪fit(method='innovations_mle') #set the model and then check if best
            aic = model.aic

            #check if AIC has decreased
            if aic < best_aic:
                best_aic = aic
                best_model = model
                best_p = p
                best_q = q
                best_std = model.resid

    #return desired value after prediciton
    final = best_model.predict(start=0,end=len(z)+n)
    predictions = final[30:30+len(test)]

    #calculate error metrics to determine success of model
    mae = mean_absolute_error(test, predictions)
    mse = mean_squared_error(test, predictions)
    rmse = np.sqrt(mse)
    r2 = r2_score(test, predictions)

    return best_aic,final,np.std(best_std), [mae, mse, rmse, r2]
```

[ ]:
```
percent_cover_all_coral = np.array(explore_df['percentCover_allCoral'])
```

[ ]:
```
aic, final,std, scores = sm_arma(percent_cover_all_coral)
```

[ ]:
```
MAE,MSE,RMSE,R2= scores[0],scores[1],scores[2],scores[3]
```

[ ]:
```
z = np.diff(percent_cover_all_coral)
time = np.linspace(1987,2021,len(z))
time2 = np.linspace(1987,2021+10,len(final))
```

[ ]:
```
coral['year_column'] = coral.index.year
coral
```

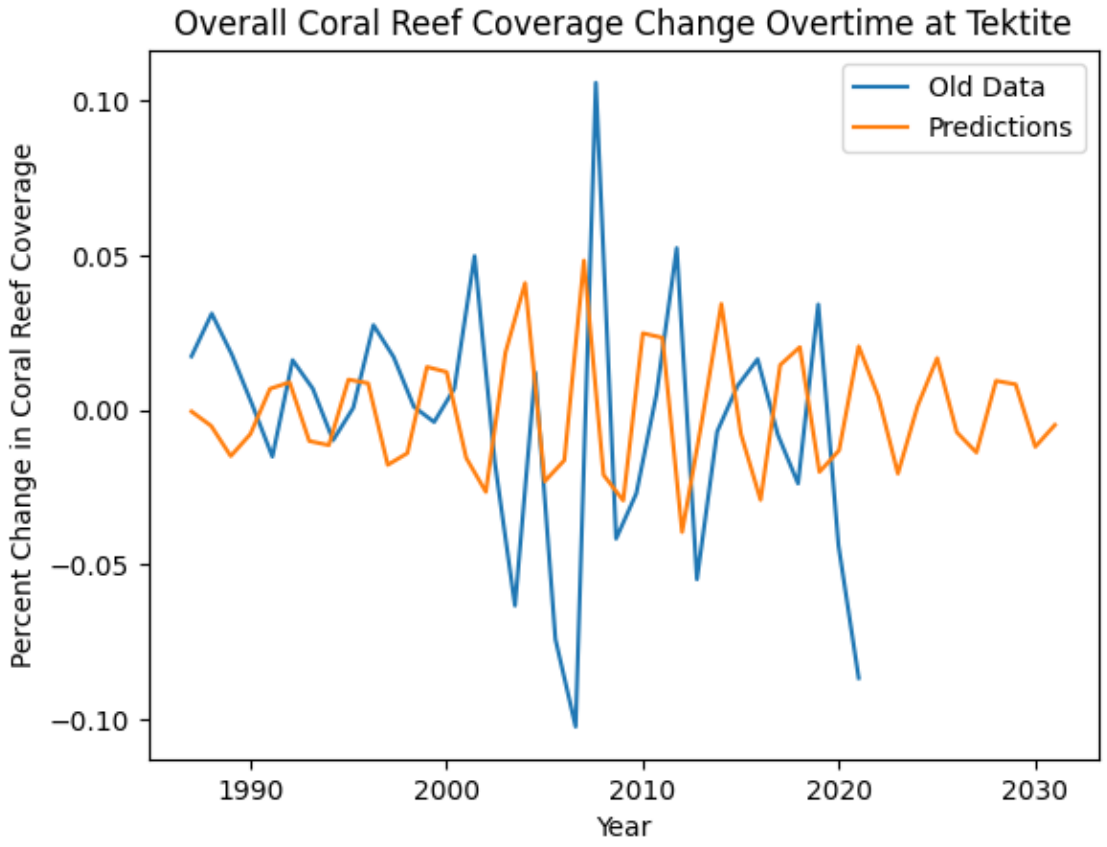
```
              site transect quadrat  percentCover_allCoral  \
Date
1987-12-01  Tektite        1    T1Q1                  0.165
1987-12-01  Tektite        1    T1Q2                  0.100
1987-12-01  Tektite        1    T1Q3                  0.135
```

```
1987-12-01  Tektite        1    T1Q4                0.155
1987-12-01  Tektite        1    T1Q5                0.190
…             …      …      …                    …
2021-07-01    Yawzi       nd       6                0.000
2021-07-01    Yawzi       nd       7                0.000
2021-07-01    Yawzi       nd       8                0.005
2021-07-01    Yawzi       nd       9                0.005
2021-07-01    Yawzi       nd      10                0.035

            percentCover_macroalgae  percentCover_CTB  percentCover_other  \
Date
1987-12-01                    0.035             0.060               0.905
1987-12-01                    0.060             0.740               0.200
1987-12-01                    0.035             0.450               0.515
1987-12-01                    0.065             0.615               0.320
1987-12-01                    0.040             0.480               0.480
…                             …                 …                   …
2021-07-01                    0.300             0.500               0.200
2021-07-01                    0.285             0.455               0.260
2021-07-01                    0.350             0.350               0.300
2021-07-01                    0.420             0.425               0.155
2021-07-01                    0.340             0.385               0.275

            year_column
Date
1987-12-01         1987
1987-12-01         1987
1987-12-01         1987
1987-12-01         1987
1987-12-01         1987
…                   …
2021-07-01         2021
2021-07-01         2021
2021-07-01         2021
2021-07-01         2021
2021-07-01         2021

[2264 rows x 8 columns]
```

3.0.1 Tektite ARIMA with error scores

[ ]:
```
df_tektite = coral[coral['site']=='Tektite']
avg_percent_cover_allcoral_tektite = df_tektite.
 ↪groupby('year_column')['percentCover_allCoral'].mean()
aic, final,std,scores = sm_arma(avg_percent_cover_allcoral_tektite)
z = np.diff(avg_percent_cover_allcoral_tektite)
time = np.linspace(1987,2021,len(z))
```

```
time2 = np.linspace(1987,2021+10,len(final))
plt.plot(time,z,label='Old Data')
plt.plot(time2,final,label='Predictions')
plt.xlabel('Year')
plt.ylabel('Percent Change in Coral Reef Coverage')
plt.title('Overall Coral Reef Coverage Change Overtime at Tektite')
plt.legend()
plt.show()
```



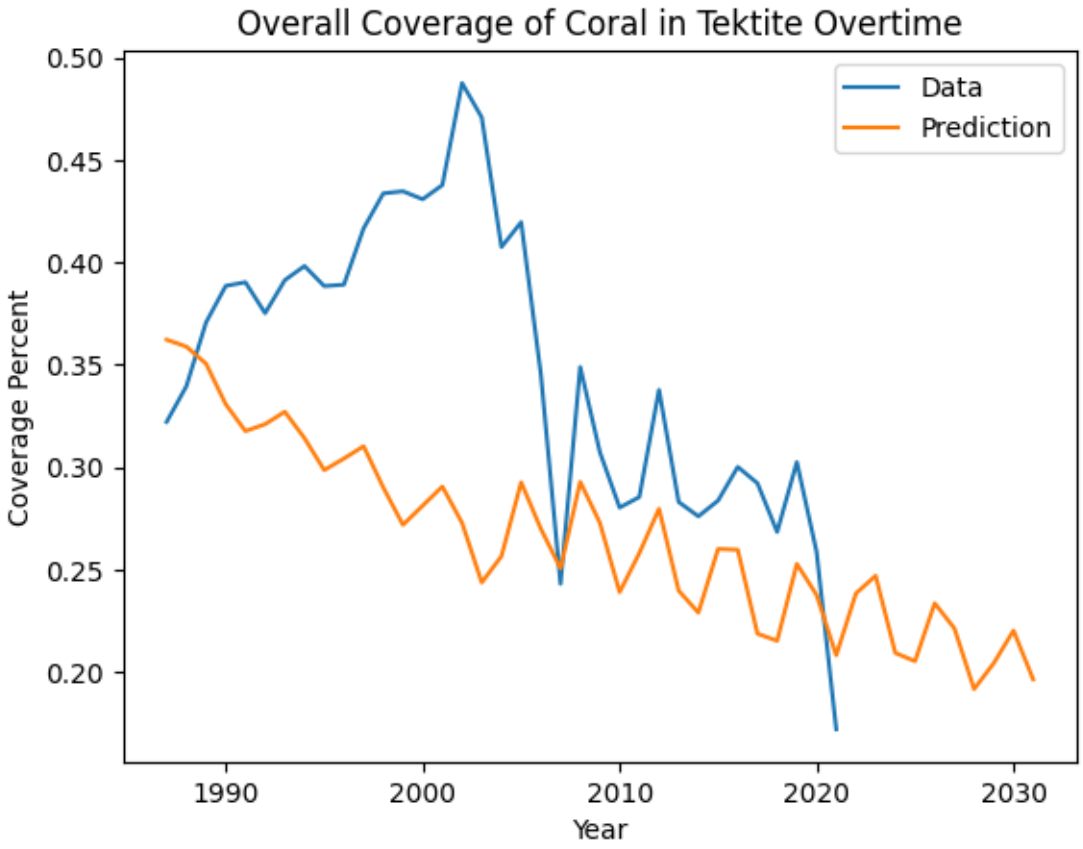
[ ]:
```
MAE,MSE,RMSE,R2= scores[0],scores[1],scores[2],scores[3]
```

[ ]:
```
aic
```

-86.68223450031562

[ ]:
```
avg_percent_cover_allcoral_tektite_prediction = np.array([np.sum(final[:i]) for␣
↪i in range(len(final))])
avg_percent_cover_allcoral_tektite_prediction += np.
↪mean(avg_percent_cover_allcoral_tektite.iloc[:5])
```

[ ]:
```
new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_tektite))
new_time2 = np.
 ↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_tektite_prediction))
plt.plot(new_time,avg_percent_cover_allcoral_tektite,label='Data')
plt.
 ↪plot(new_time2,avg_percent_cover_allcoral_tektite_prediction,label='Prediction')
plt.title('Overall Coverage of Coral in Tektite Overtime')
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()
plt.show()
```



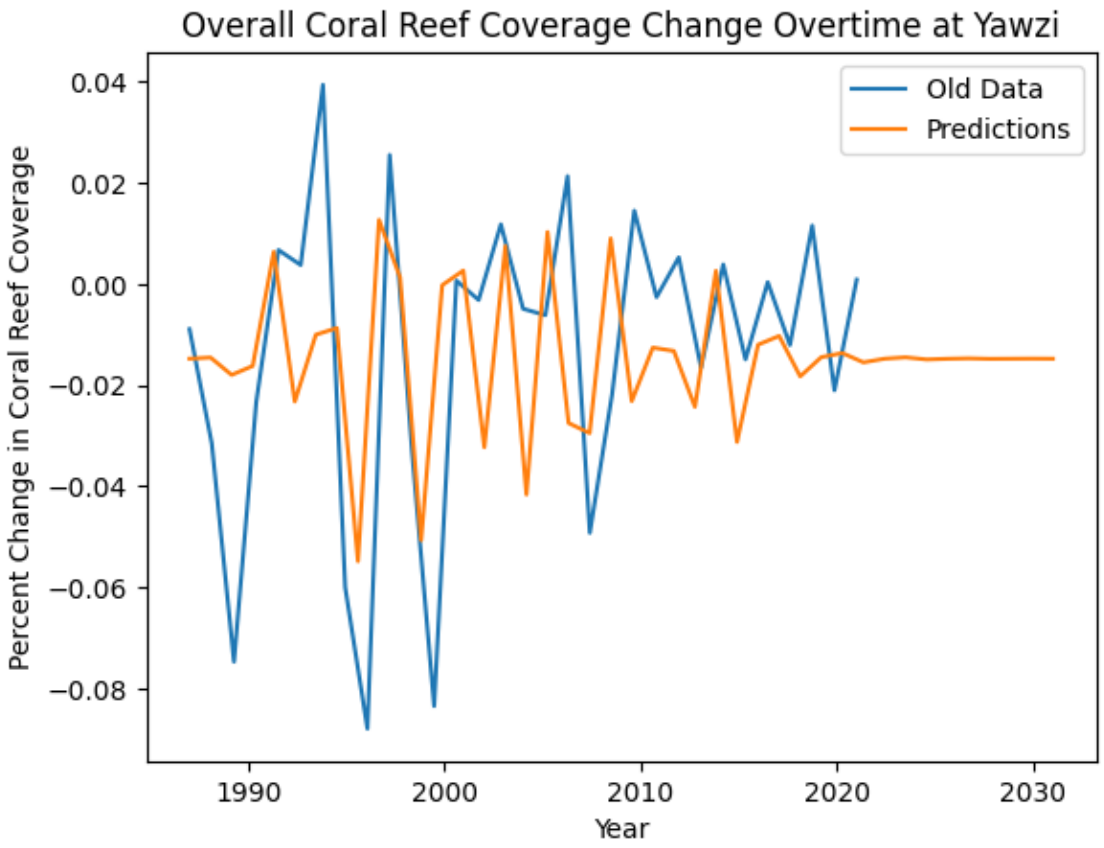
3.0.2 Yawzi ARIMA with Error Metrics

[ ]:
```
avg_percent_cover_allcoral_yawzi
```

```
year_column
1987    0.446172
1988    0.437217
1989    0.405708
```

```
1990    0.331034
1991    0.307667
1992    0.314350
1993    0.318067
1994    0.357400
1995    0.297433
1996    0.209533
1997    0.235000
1998    0.202133
1999    0.118700
2000    0.119333
2001    0.116133
2002    0.127900
2003    0.122967
2004    0.116767
2005    0.138033
2006    0.088833
2007    0.066933
2008    0.081400
2009    0.078767
2010    0.084000
2011    0.067433
2012    0.071267
2013    0.056333
2017    0.056667
2018    0.044567
2019    0.056075
2020    0.035033
2021    0.035867
Name: percentCover_allCoral, dtype: float64
```

[ ]:
```
df_yawzi = coral[coral['site']=='Yawzi']
avg_percent_cover_allcoral_yawzi = df_yawzi.
  ↪groupby('year_column')['percentCover_allCoral'].mean()
aic, final,std,scores = sm_arma(avg_percent_cover_allcoral_yawzi)
z = np.diff(avg_percent_cover_allcoral_yawzi)
time = np.linspace(1987,2021,len(z))
time2 = np.linspace(1987,2021+10,len(final))
plt.plot(time,z,label='Old Data')
plt.plot(time2,final,label='Predictions')
plt.xlabel('Year')
plt.ylabel('Percent Change in Coral Reef Coverage')
plt.title('Overall Coral Reef Coverage Change Overtime at Yawzi')
plt.legend()
plt.show()
```



[ ]:
```
MAE,MSE,RMSE,R2= scores[0],scores[1],scores[2],scores[3]
```

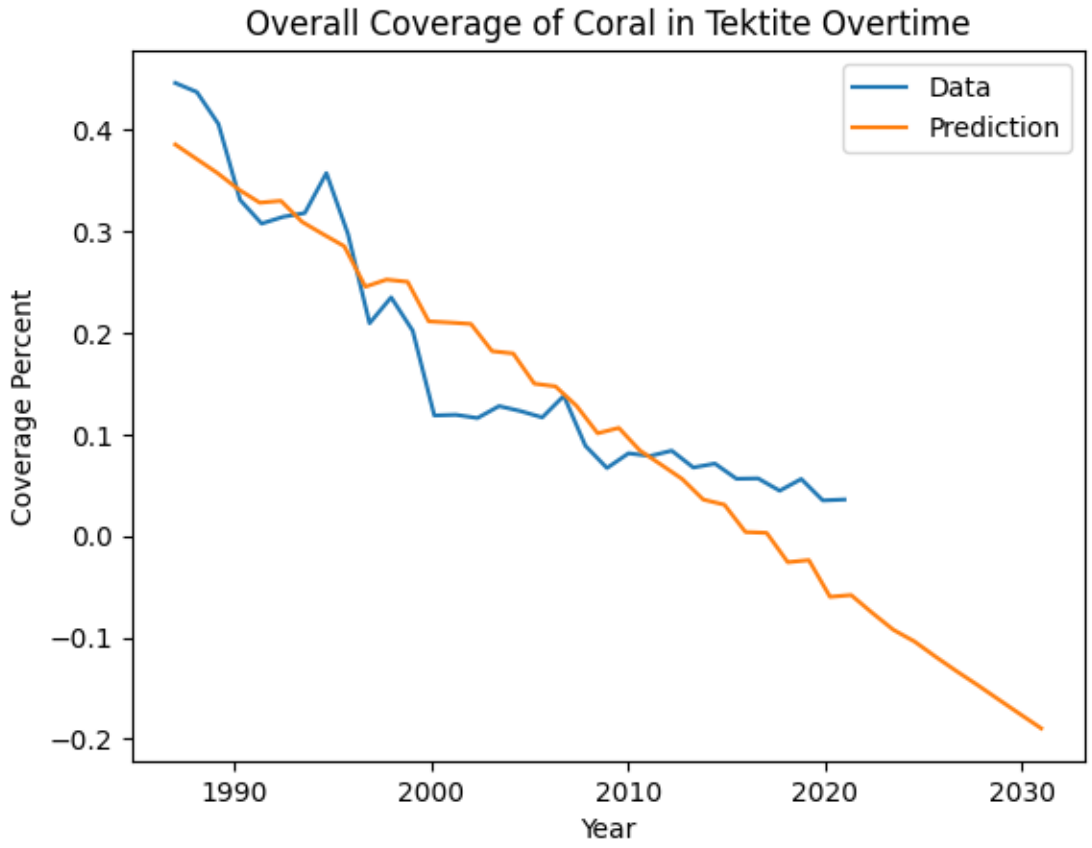
[ ]:
```
aic
```

```
-99.48838625249137
```

[ ]:
```
avg_percent_cover_allcoral_yawzi_prediction = np.array([np.sum(final[:i]) for i␣
↪in range(len(final))])
avg_percent_cover_allcoral_yawzi_prediction += np.
↪mean(avg_percent_cover_allcoral_yawzi.iloc[:5])
```

[ ]:
```
new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_yawzi))
new_time2 = np.
↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_yawzi_prediction))
plt.plot(new_time,avg_percent_cover_allcoral_yawzi,label='Data')
plt.
↪plot(new_time2,avg_percent_cover_allcoral_yawzi_prediction,label='Prediction')
plt.title('Overall Coverage of Coral in Tektite Overtime')
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
```

```
plt.legend()
plt.show()
```



4 Kalman Filtering Code

In this notebook we will explore our intiail dataset. The dataset contains data about coral coverage in two different main locations over multiple decades.

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.statespace.kalman_filter import KalmanFilter
```

[ ]:
```
df = pd.read_csv("coral_dataset.csv") #read in the dataset
```

[ ]:
```
df #print and look at the data set
```

```
           Date     site transect quadrat percentCover_allCoral  \
0    1987/12/01  Tektite        1    T1Q1                  16.5
1    1987/12/01  Tektite        1    T1Q2                    10
```

```
2     1987/12/01  Tektite        1    T1Q3                  13.5
3     1987/12/01  Tektite        1    T1Q4                  15.5
4     1987/12/01  Tektite        1    T1Q5                    19
…           …       …       …      …                     …
2278  2021/07/01    Yawzi       nd       6                   0.0
2279  2021/07/01    Yawzi       nd       7                   0.0
2280  2021/07/01    Yawzi       nd       8                   0.5
2281  2021/07/01    Yawzi       nd       9                   0.5
2282  2021/07/01    Yawzi       nd      10                   3.5

     percentCover_macroalgae percentCover_CTB
0                        3.5               6
1                          6              74
2                        3.5              45
3                        6.5            61.5
4                          4              48
…                        …              …
2278                    30.0            50.0
2279                    28.5            45.5
2280                    35.0            35.0
2281                    42.0            42.5
2282                    34.0            38.5

[2283 rows x 7 columns]
```

[ ]:
```
df = df[df['percentCover_CTB'] != 'nd'] #drop all of the rows with these␣
 ↪entries and then print out the amount of rows in our dataset
df = df[df['percentCover_macroalgae'] != 'nd']
df = df[df['percentCover_allCoral'] != 'nd']
print(len(df))
```

```
2264
```

[ ]:
```
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
 ↪'percentCover_allCoral'] #get a list of the columns that we want to cast
df[columns_to_cast] = df[columns_to_cast].astype(float) #cast these columns as␣
 ↪floats and divide by 100 to make the percents
df[columns_to_cast] = df[columns_to_cast]/100
df['percentCover_other'] = 1 -␣
 ↪(df['percentCover_CTB']+df['percentCover_macroalgae']) #add in another␣
 ↪column for other types of coral
```

[ ]:
```
# convert the date column to date time objects
df.index = pd.to_datetime(df.index)
```

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import inv, norm
```

[ ]:
```
df = pd.read_csv("coral_dataset.csv", index_col = "Date") #read in the dataset
df = df[df['percentCover_CTB'] != 'nd'] #drop all of the rows with these␣
 ↪entries and then print out the amount of rows in our dataset
df = df[df['percentCover_macroalgae'] != 'nd']
df = df[df['percentCover_allCoral'] != 'nd']
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
 ↪'percentCover_allCoral'] #get a list of the columns that we want to cast
df[columns_to_cast] = df[columns_to_cast].astype(float) #cast these columns as␣
 ↪floats and divide by 100 to make the percents
df[columns_to_cast] = df[columns_to_cast]/100
df['percentCover_other'] = 1 -␣
 ↪(df['percentCover_CTB']+df['percentCover_macroalgae']) #add in another␣
 ↪column for other types of coral
df.index = pd.to_datetime(df.index)
df['year_column'] = df.index.year

df_tektite = df[df['site']=='Tektite']
avg_percent_cover_allcoral_tektite = df_tektite.
 ↪groupby('year_column')['percentCover_allCoral'].mean()

tek = np.diff(avg_percent_cover_allcoral_tektite)
```

[ ]:
```
df = pd.read_csv("coral_dataset.csv", index_col = "Date") #read in the dataset
df = df[df['percentCover_CTB'] != 'nd'] #drop all of the rows with these␣
 ↪entries and then print out the amount of rows in our dataset
df = df[df['percentCover_macroalgae'] != 'nd']
df = df[df['percentCover_allCoral'] != 'nd']
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
 ↪'percentCover_allCoral'] #get a list of the columns that we want to cast
df[columns_to_cast] = df[columns_to_cast].astype(float) #cast these columns as␣
 ↪floats and divide by 100 to make the percents
df[columns_to_cast] = df[columns_to_cast]/100
df['percentCover_other'] = 1 -␣
 ↪(df['percentCover_CTB']+df['percentCover_macroalgae']) #add in another␣
 ↪column for other types of coral
df.index = pd.to_datetime(df.index)
df['year_column'] = df.index.year

df_tektite = df[df['site']=='Yawzi']
avg_percent_cover_allcoral_yawzi = df_tektite.
 ↪groupby('year_column')['percentCover_allCoral'].mean()

yaw = np.diff(avg_percent_cover_allcoral_tektite)
```

[ ]:
```
class KalmanFilter(object):
    def __init__(self,F,Q,H,R,u):
```

```
        """
        Initialize the dynamical system models.

        Parameters
        ----------
        F : ndarray of shape (n,n)
            The state transition model.
        Q : ndarray of shape (n,n)
            The covariance matrix for the state noise.
        H : ndarray of shape (m,n)
            The observation model.
        R : ndarray of shape (m,m)
            The covariance matrix for observation noise.
        u : ndarray of shape (n,)
            The control vector.
        """
        #save attributes
        self.F=F
        self.Q=Q
        self.H=H
        self.R=R
        self.u=u

    def evolve(self,x0,N):
        """
        Compute the first N states and observations generated by the Kalman␣
↪system.

        Parameters
        ----------
        x0 : ndarray of shape (n,)
            The initial state.
        N : integer
            The number of time steps to evolve.

        Returns
        -------
        states : ndarray of shape (n,N)
            The i-th column gives the i-th state.
        obs : ndarray of shape (m,N)
            The i-th column gives the i-th observation.
        """
        #initialize states matrix
        n=len(x0)
        m=len(self.H@x0)
        states=np.zeros((n,N))
        states[:,0]=x0
```

```
        obs=np.zeros((m,N))
        obs[:,0]=self.H@(states[:,0])
        for i in range(1,N):
            #get noise
            w=np.random.multivariate_normal(np.zeros_like(states[:,0]), self.Q)
            v=np.random.multivariate_normal(np.zeros_like(obs[:,0]), self.R)
            #update states
            states[:,i]=self.F@states[:,i-1]+self.u+w
            #update obs
            obs[:,i]=self.H@states[:,i]+v
        return states, obs

    def estimate(self,x0,P0,z, return_norms = False):
        """
        Compute the state estimates using the kalman filter.

        Parameters
        ----------
        x0 : ndarray of shape (n,)
            The initial state estimate.
        P0 : ndarray of shape (n,n)
            The initial error covariance matrix.
        z : ndarray of shape(m,N)
            Sequence of N observations (each column is an observation).

        Returns
        -------
        out : ndarray of shape (n,N)
            Sequence of state estimates (each column is an estimate).
        """
        #initialize states matrix
        n=len(x0)
        m,N=z.shape
        states=np.zeros((n,N))
        P=P0
        states[:,0]=x0
        #iterate to get states
        for i in range(1,N):
            #predict phase
            xkk1=self.F@states[:,i-1]+self.u
            Pkk1=self.F@P@self.F.T+self.Q
            #update step
            y=z[:,i]-self.H@xkk1
            S=self.H@Pkk1@self.H.T+self.R
            K=Pkk1@self.H.T@inv(S)
            states[:,i]=xkk1+K@y
            P=(np.eye(n)-K@self.H)@P
```

```
        return states


    def predict(self,x,k):
        """
        Predict the next k states in the absence of observations.

        Parameters
        ----------
        x : ndarray of shape (n,)
            The current state estimate.
        k : integer
            The number of states to predict.

        Returns
        -------
        out : ndarray of shape (n,k)
            The next k predicted states.
        """
        #initialize states matrix
        n=len(x)
        states=np.zeros((n,k))
        states[:,0]=self.F@x+self.u
        #iterate to get k next steps
        for i in range(1,k):
            states[:,i]=self.F@states[:,i-1]+self.u
        return states

    def rewind(self,x,k):
        """
        Predict the states from time 0 through k-1 in the absence of␣
↪observations.

        Parameters
        ----------
        x : ndarray of shape (n,)
            The state estimate at time k.
        k : integer
            The current time step.

        Returns
        -------
        out : ndarray of shape (n,k)
            The predicted states from time 0 up through k-1 (in that order).
        """
        #get F^-1
        Finv=inv(self.F)
```

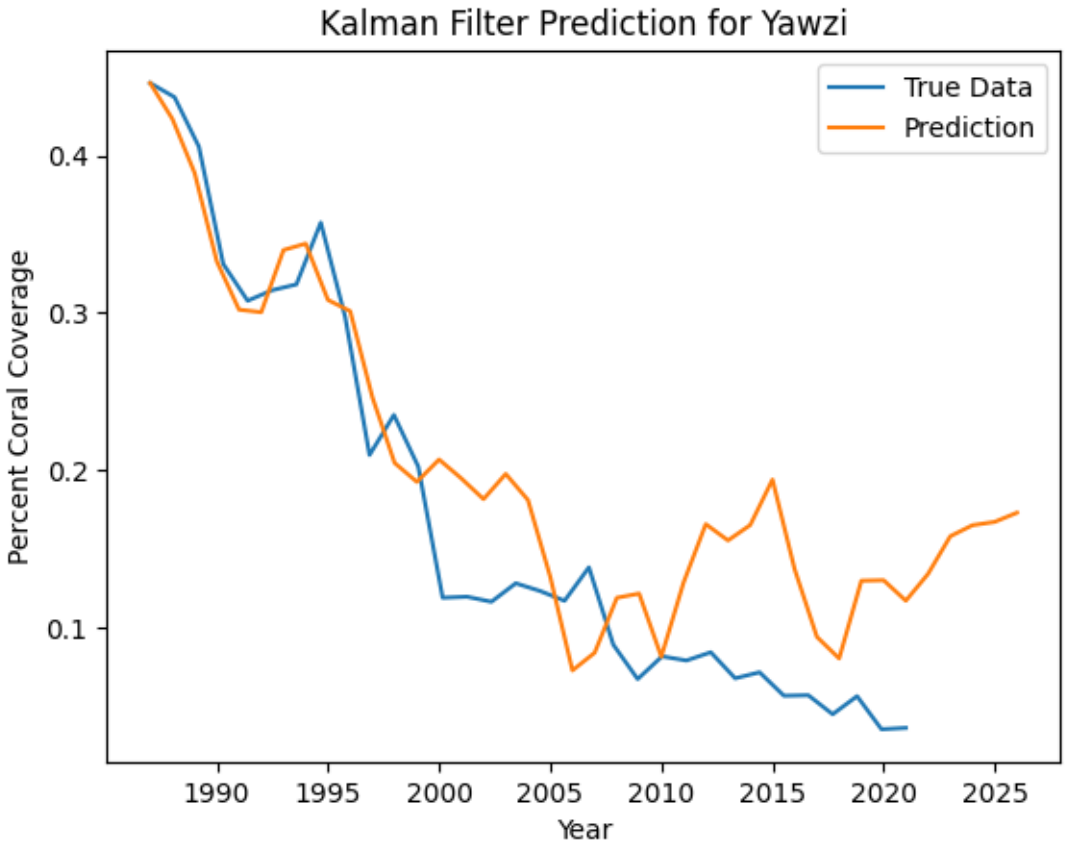
```
        n=len(x)
        states=np.zeros((n,k))
        #get most previous step
        states[:,-1]=Finv@(x-self.u)
        #iterate to get k steps
        for i in range(k-2,-1,-1):
            states[:,i]=Finv@(states[:,i+1]-self.u)
        return states
```

[ ]:
```
F=np.linspace(.05,2.5,20)
Q=np.linspace(0.001,0.02,4)
H=np.linspace(0.001,3,20)
R=np.linspace(0.0001,.5,4)
best_mse=np.inf
for f in F:
    for q in Q:
        for h in H:
            for r in R:
                kf=KalmanFilter(np.array([[f]]),np.array([[q]]),np.
 ↪array([[h]]),np.array([[r]]),u=np.zeros(1))
                states, obs= kf.evolve([avg_percent_cover_allcoral_yawzi.
 ↪iloc[0]], 30)
                mse = np.mean((obs - np.
 ↪array(avg_percent_cover_allcoral_yawzi)[-5])**2)
                if mse < best_mse:
                    best_mse = mse
                    best_params = (f,q,h,r)
                    beststates=states
                    bestobs=obs
```

[ ]:
```
f,q,h,r = (0.9526315789473685, 0.001, 0.1588421052631579, 0.0001)
kf=KalmanFilter(np.array([[f]]),np.array([[q]]),np.array([[h]]),np.
 ↪array([[r]]),u=np.zeros(1))
states, bestobs = kf.evolve([avg_percent_cover_allcoral_yawzi.iloc[0]], 40)
```

[ ]:
```
print(best_mse)
print(best_params)
x = np.linspace(1987,2021,len(avg_percent_cover_allcoral_yawzi))
x1 = np.linspace(1987,2026,len(states[0]))
plt.plot(x,avg_percent_cover_allcoral_yawzi, label = 'True Data')
plt.plot(x1, states[0], label = 'Prediction')
plt.title('Kalman Filter Prediction for Yawzi')
plt.ylabel('Percent Coral Coverage')
plt.xlabel('Year')
plt.legend()
plt.show()
```

```
0.010863811043616518
(0.9526315789473685, 0.001, 1.263736842105263, 0.0001)
```



[ ]:
```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

[ ]:
```
print(f"MSE for Yawzi Test␣
  ↪{mean_squared_error(avg_percent_cover_allcoral_yawzi[-5:],states[0][30:
  ↪-5])}")
print(f"RMSE for Yawzi Test {np.
  ↪sqrt(mean_squared_error(avg_percent_cover_allcoral_yawzi[-5:],states[0][30:
  ↪-5]))}")
print(f"MAE for Yawzi Test␣
  ↪{mean_absolute_error(avg_percent_cover_allcoral_yawzi[-5:],states[0][30:
  ↪-5])}")
print(f"R2 for Yawzi Test {r2_score(avg_percent_cover_allcoral_yawzi[-5:
  ↪],states[0][30:-5])}")
```

```
MSE for Yawzi Test 0.004714895231061715
RMSE for Yawzi Test 0.06866509470656627
MAE for Yawzi Test 0.06436685871918293
```

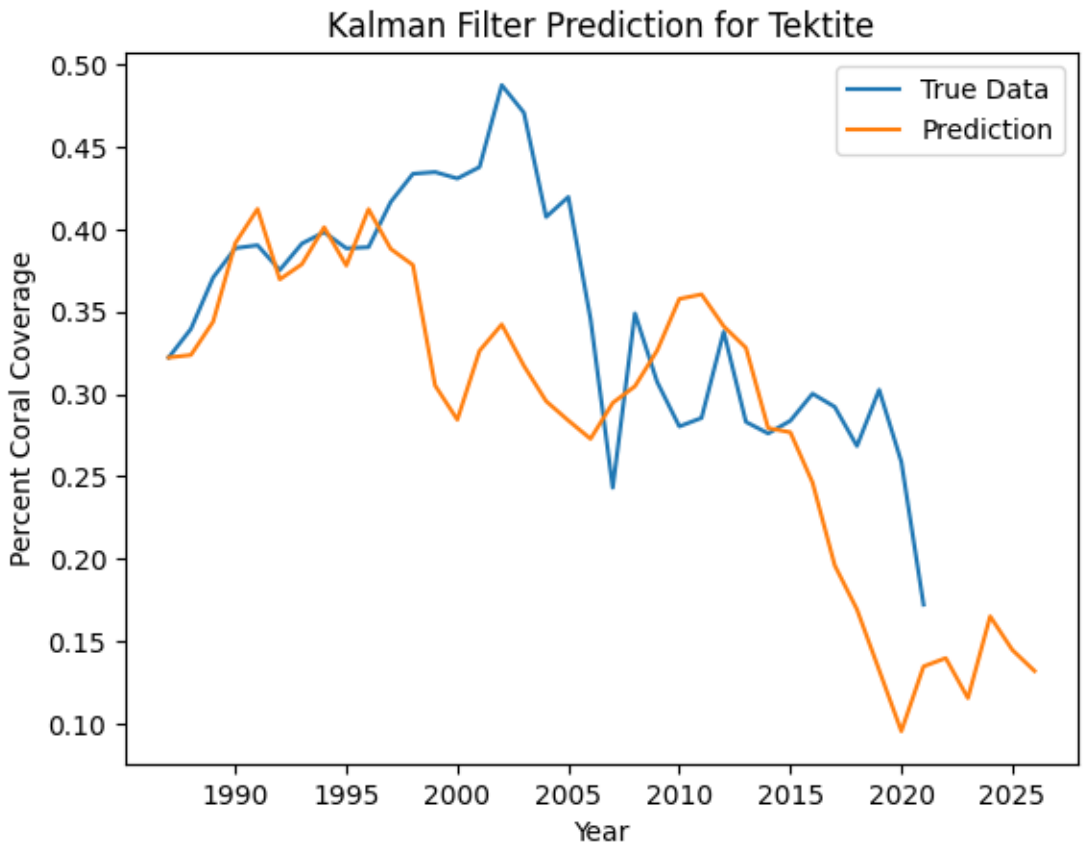
R2 for Yawzi Test -52.6212451241687

[ ]:
```
F=np.linspace(.05,1.5,20)
Q=np.linspace(0.001,0.03,4)
H=np.linspace(0.001,2,20)
R=np.linspace(0.0001,.5,4)
best_mse=np.inf
for f in F:
    for q in Q:
        for h in H:
            for r in R:
                kf=KalmanFilter(np.array([[f]]),np.array([[q]]),np.
 ↪array([[h]]),np.array([[r]]),u=np.zeros(1))
                states, obs= kf.evolve([avg_percent_cover_allcoral_tektite.
 ↪iloc[0]], 30)
                mse = np.mean((obs - np.
 ↪array(avg_percent_cover_allcoral_tektite)[-5])**2)
                if mse < best_mse:
                    best_mse = mse
                    best_params = (f,q,h,r)
                    beststates=states
                    bestobs=obs
```

[ ]:
```
f,q,h,r = best_params
kf=KalmanFilter(np.array([[f]]),np.array([[q]]),np.array([[h]]),np.
 ↪array([[r]]),u=np.zeros(1))
states, bestobs = kf.evolve([avg_percent_cover_allcoral_tektite.iloc[0]], 40)
```

[ ]:
```
print(best_mse)
print(best_params)
x = np.linspace(1987,2021,len(avg_percent_cover_allcoral_tektite))
x1 = np.linspace(1987,2026,len(bestobs[0]))
plt.plot(x,avg_percent_cover_allcoral_tektite,label = 'True Data')
plt.plot(x1, states[0], label = 'Prediction')
plt.title('Kalman Filter Prediction for Tektite')
plt.ylabel('Percent Coral Coverage')
plt.xlabel('Year')
plt.legend()
plt.show()
```

```
0.0039343621682250895
(0.9657894736842105, 0.001, 0.9478947368421052, 0.0001)
```



[ ]:
```
print(f"MSE for Tektite Test␣
  ↪{mean_squared_error(avg_percent_cover_allcoral_tektite[-5:],states[0][30:
  ↪-5])}")
print(f"RMSE for Tektite Test {np.
  ↪sqrt(mean_squared_error(avg_percent_cover_allcoral_tektite[-5:],states[0][30:
  ↪-5]))}")
print(f"MAE for Tektite Test␣
  ↪{mean_absolute_error(avg_percent_cover_allcoral_tektite[-5:],states[0][30:
  ↪-5])}")
print(f"R2 for Tektite Test {r2_score(avg_percent_cover_allcoral_tektite[-5:
  ↪],states[0][30:-5])}")
```

```
MSE for Tektite Test 0.015251066164043328
RMSE for Tektite Test 0.12349520704887024
MAE for Tektite Test 0.1133635766891387
R2 for Tektite Test -6.165586205377412
```

5 VARMAX Code

In this notebook, we experiment with predictions using a VARMAX model, something we have learned outside of class.

5.1 Cleaning the data

[ ]:
```
df = pd.read_csv("coral_dataset.csv", index_col = "Date") #read in the dataset
specific_value_count = df['percentCover_CTB'].value_counts()['nd']
specific_value_count = df['percentCover_macroalgae'].value_counts()['nd']
specific_value_count = df['percentCover_allCoral'].value_counts()['nd']
df = df[df['percentCover_CTB'] != 'nd'] #drop all of the rows with these␣
 ↪entries and then print out the amount of rows in our dataset
df = df[df['percentCover_macroalgae'] != 'nd']
df = df[df['percentCover_allCoral'] != 'nd']
columns_to_cast = ['percentCover_CTB', 'percentCover_macroalgae',␣
 ↪'percentCover_allCoral'] #get a list of the columns that we want to cast
df[columns_to_cast] = df[columns_to_cast].astype(float) #cast these columns as␣
 ↪floats and divide by 100 to make the percents
df[columns_to_cast] = df[columns_to_cast]/100
df['percentCover_other'] = 1 -␣
 ↪(df['percentCover_CTB']+df['percentCover_macroalgae']) #add in another␣
 ↪column for other types of coral
# convert the date column to date time objects
df.index = pd.to_datetime(df.index)
```

5.2 Format dataframes for model

[ ]:
```
df['year_column'] = df.index.year

df_tektite = df[df['site']=='Tektite']
avg_percent_cover_allcoral_tektite = df_tektite.
 ↪groupby('year_column')[['percentCover_allCoral','percentCover_macroalgae',␣
 ↪'percentCover_CTB']].mean()
df_yawzi = df[df['site']=='Yawzi']
avg_percent_cover_allcoral_yawzi = df_yawzi.
 ↪groupby('year_column')[['percentCover_allCoral','percentCover_macroalgae',␣
 ↪'percentCover_CTB']].mean()
```

5.3 Varmax model without temp

[ ]:
```
from statsmodels.tsa.statespace.varmax import VARMAX
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
def varmax(array, name, p_max=3, q_max=3, n=10):
    """
    Build an ARMA model with statsmodel and
    predict future n values.
```

```
    Parameters:
        filename (str): data filename
        p_max (int): maximum order of autoregressive model
        q_max (int): maximum order of moving average model
        n (int): number of values to predict

    Return:
        aic (float): aic of optimal model
    """
    train = array[:30]
    test = array[30:]
    # z = np.diff(array)

    best_aic = np.inf #set our values
    best_model = None
    best_p = None
    best_q = None
    best_std = None
    for p in range(1,p_max+1): #iterate through p and q
        for q in range(1,q_max+1):
            model = model = VARMAX(train, order=(p,q), seasonal_order=(0, 0, 0,␣
↪0),mle_regression = True,filter_concentrated = True)#set the model and then␣
↪check if best
            model_fit = model.fit(disp=True)
            aic = model_fit.aic
            # aic = model.score()
            # aic = model.score_obs()
            if aic < best_aic:
                print("new")
                best_aic = aic
                best_model = model_fit
                best_p = p
                best_q = q
                # best_std = model.resid


    final = best_model.predict(start=0,end=len(train)+n) #return desired value␣
↪after prediciton
    predictions = final[30:30+len(test)]
    print(len(final), len(predictions))
    mae = mean_absolute_error(test[name], predictions[name])
    mse = mean_squared_error(test[name], predictions[name])
    rmse = np.sqrt(mse)
    r2 = r2_score(test[name], predictions[name])

    return best_aic,final, [mae, mse, rmse, r2]
```

5.4 Using the model for tektite and yawzi

[ ]:
```
aic_tek, avg_percent_cover_allcoral_tektite_prediction, scores_tek =␣
 ↪varmax(avg_percent_cover_allcoral_tektite,'percentCover_allCoral')
aic_yaw, avg_percent_cover_allcoral_Yawzi_prediction, scores_yaw =␣
 ↪varmax(avg_percent_cover_allcoral_yawzi,'percentCover_allCoral')
```

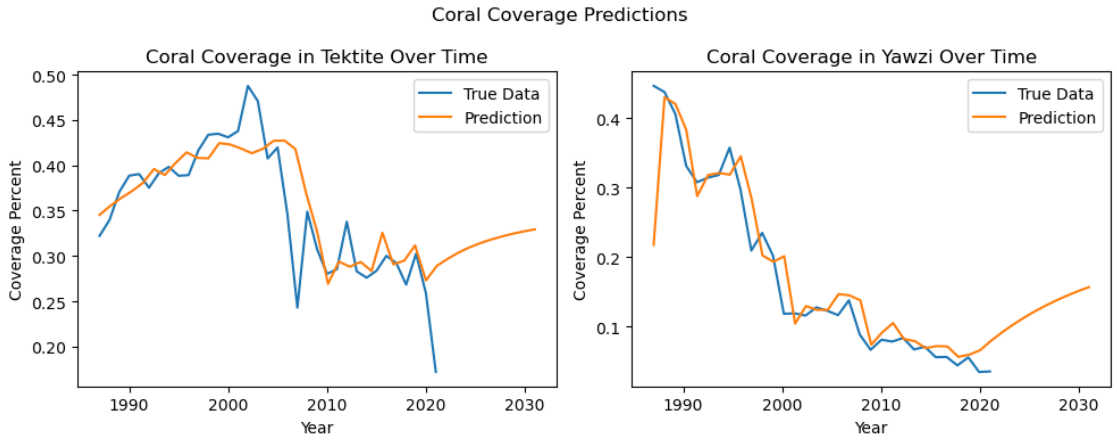
[ ]:
```
# print scores
print("Tektite")
print(f"aic: {aic_tek}")
print(f"scores: {scores_tek}")

print("Yawzi")
print(f"aic: {aic_yaw}")
print(f"scores: {scores_yaw}")
```

```
Tektite
aic: -217.8230022795396
scores: [0.045589966471170515, 0.004347603694286463, 0.06593636094209676,
-1.0426853259397117]
Yawzi
aic: -189.70391296596947
scores: [0.03761063463535112, 0.00145484820469039, 0.03814247245119789,
-8378.925659016539]
```

5.5 Plot results

[ ]:
```
new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_tektite))
new_time2 = np.
 ↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_tektite_prediction))
['percentCover_allCoral','percentCover_macroalgae', 'percentCover_CTB']
plt.figure(figsize = (10,4))
plt.subplot(121)
plt.title('Coral Coverage in Tektite Over Time')
plt.
 ↪plot(new_time,avg_percent_cover_allcoral_tektite['percentCover_allCoral'],label='True␣
 ↪Data')
plt.
 ↪plot(new_time2,avg_percent_cover_allcoral_tektite_prediction['percentCover_allCoral'],label
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()

new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_yawzi))
new_time2 = np.
 ↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_Yawzi_prediction))
plt.subplot(122)
plt.title('Coral Coverage in Yawzi Over Time')
```

```
plt.
  ↪plot(new_time,avg_percent_cover_allcoral_yawzi['percentCover_allCoral'],label='True␣
  ↪Data')
plt.
  ↪plot(new_time2,avg_percent_cover_allcoral_Yawzi_prediction['percentCover_allCoral'],label='
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()

plt.suptitle("Coral Coverage Predictions")
plt.tight_layout()
plt.legend()
plt.show()
```



5.6 Varmax model including temp

[ ]:
```
from statsmodels.tsa.statespace.varmax import VARMAX
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
def varmax(array, name, p_max=3, q_max=3, n=10):
    """
    Build an ARMA model with statsmodel and
    predict future n values.

    Parameters:
        filename (str): data filename
        p_max (int): maximum order of autoregressive model
        q_max (int): maximum order of moving average model
        n (int): number of values to predict

    Return:
        aic (float): aic of optimal model
```

```
    """
    train = array[:23]
    test = array[23:]
    # z = np.diff(array)

    best_aic = np.inf #set our values
    best_model = None
    best_p = None
    best_q = None
    best_std = None
    for p in range(1,p_max+1): #iterate through p and q
        for q in range(1,q_max+1):
            model = model = VARMAX(train, order=(p,q), seasonal_order=(0, 0, 0,␣
↪0),mle_regression = True,filter_concentrated = True)#set the model and then␣
↪check if best
            model_fit = model.fit(disp=True)
            aic = model_fit.aic
            # aic = model.score()
            # aic = model.score_obs()
            if aic < best_aic:
                print("new")
                best_aic = aic
                best_model = model_fit
                best_p = p
                best_q = q
                # best_std = model.resid


    final = best_model.predict(start=0,end=len(train)+n) #return desired value␣
↪after prediciton
    predictions = final[len(train):len(train)+len(test)]
    print(predictions)
    print(len(final), len(predictions))
    mae = mean_absolute_error(test[name], predictions[name])
    mse = mean_squared_error(test[name], predictions[name])
    rmse = np.sqrt(mse)
    r2 = r2_score(test[name], predictions[name])

    return best_aic,final, [mae, mse, rmse, r2]
```

5.7 loading/cleaning temp data

[ ]:
```
sea_df = pd.read_csv("seawatertemp (1).csv", index_col = "Date") #read in the␣
 ↪dataset
sea_df.index = pd.to_datetime(sea_df.index)
sea_df['year_column'] = sea_df.index.year
sea_df = sea_df[sea_df['Site'].isin(['Yawzi_NPS','Yawzi_9m'])]
```

```
specific_value_count = sea_df['Temperature'].value_counts()['nd']
sea_df = sea_df[sea_df['Temperature'] != 'nd']
sea_df['Temperature'] = sea_df['Temperature'].astype(float)
year_means = sea_df.groupby('year_column')['Temperature'].mean()
```

5.8 formating data to run model

[ ]:
```
tek_temp = avg_percent_cover_allcoral_tektite.join(year_means)
tek_temp = tek_temp[2:]

yaw_temp = avg_percent_cover_allcoral_yawzi.join(year_means)
yaw_temp = tek_temp[2:]
```

5.9 running model for tektite and yawzi

[ ]:
```
aic_tek, avg_percent_cover_allcoral_tektite_prediction, scores_tek =␣
 ↪varmax(tek_temp,'percentCover_allCoral')
aic_yaw, avg_percent_cover_allcoral_yawzi_prediction, scores_yaw =␣
 ↪varmax(yaw_temp,'percentCover_allCoral')
```

[ ]:
```
# print scores
print("Tektite")
print(f"aic: {aic_tek}")
print(f"scores: {scores_tek}")

print("Yawzi")
print(f"aic: {aic_yaw}")
print(f"scores: {scores_yaw}")
```

```
Tektite
aic: -176.94236839014184
scores: [0.107187507531975, 0.014749074286818393, 0.12144576685425636,
-7.877664878410419]
Yawzi
aic: -145.7338022757362
scores: [0.07743817692092156, 0.009464714668263985, 0.09728676512385426,
-5.105991653765369]
```

5.10 Plot results

[ ]:
```
new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_tektite))
new_time2 = np.
 ↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_tektite_prediction))

plt.figure(figsize = (10,4))
plt.subplot(121)
plt.title('Coral Coverage in Tektite Over Time')
```

```
plt.
  ↪plot(new_time,avg_percent_cover_allcoral_tektite['percentCover_allCoral'],label='True␣
  ↪Data')
plt.
  ↪plot(new_time2,avg_percent_cover_allcoral_tektite_prediction['percentCover_allCoral'],label
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()

new_time = np.linspace(1987,2021,len(avg_percent_cover_allcoral_yawzi))
new_time2 = np.
  ↪linspace(1987,2021+10,len(avg_percent_cover_allcoral_Yawzi_prediction))
plt.subplot(122)
plt.title('Coral Coverage in Yawzi Over Time')
plt.
  ↪plot(new_time,avg_percent_cover_allcoral_yawzi['percentCover_allCoral'],label='True␣
  ↪Data')
plt.
  ↪plot(new_time2,avg_percent_cover_allcoral_Yawzi_prediction['percentCover_allCoral'],label='
plt.xlabel('Year')
plt.ylabel('Coverage Percent')
plt.legend()

plt.suptitle("Coral Coverage Predictions Without Temperature")
plt.tight_layout()
plt.legend()
plt.show()
```