In [1]:

```
import os
import glob
import math as m
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import statsmodels.api as sm
import seaborn as sns
import itertools
import sklearn
import re
```

Create File Pathways and see how many files

In [2]:

```
#Path into the chl folder
file_path_chl = "/Users/kenneth/Desktop/repro_database-main/data/environment
#Grabs all file names in the chl folder
file_list_chl = os.listdir(file_path_chl)
file_list_chl.remove(".ipynb_checkpoints")
print(len(set(file_list_chl)))

#Salinity
file_path_sal = "/Users/kenneth/Desktop/repro_database-main/data/environment
#Grabs all file names in the sal folder
file_list_sal = os.listdir(file_path_sal)
print(len(set(file_list_sal)))

#Temperature
file_path_temp = "/Users/kenneth/Desktop/repro_database-main/data/environmer
#Grabs all file names in the temp folder
file_list_temp = os.listdir(file_path_temp)
file_list_temp.remove(".ipynb_checkpoints")
print(len(set(file_list_temp)))
```

```
27
27
91
```

To make sure to find the number of lines of data we have, I checked the number of rows

In [3]:

```
def number_of_lines(file_list, file_path, name):
    num_rows = 0
    for file in file_list:
        loc = file_path + file
        df_temp = pd.read_csv(loc, names=name)
        num_rows += len(df_temp)
    return num_rows
```

In [4]:

```
len_chl = number_of_lines(file_list_chl, file_path_chl, ['date', 'chl'])
print("Chloropyll should be", len_chl, "long.")
len_sal = number_of_lines(file_list_sal, file_path_sal, ['date', 'sal'])
print("Salinity should be", len_sal, "long.")
```

```
len_temp = number_of_lines(file_list_temp, file_path_temp, ['date', 'temp'])
print("Temperature should be", len_temp, "long.")
```

```
Chloropyll should be 1331 long.
Salinity should be 1598 long.
Temperature should be 5931 long.
```

Creating function to combine all corresponding data into one dataset

In [5]:

```
def big_dataset_builder(file_list, file_path):
    big_df = pd.DataFrame()
    counter = 0
    for i in range(len(file_list)):
        loc = file_path + file_list[i]
        nums = int(re.findall(r'\d+', file_list[i])[0])

        df_temp = pd.read_csv(loc, names=['Date', "Value"])
        df_temp["Site_ID"] = [nums] * len(df_temp)
        df_temp = df_temp.reset_index().drop('index', axis = 1)
        #print(df_temp)

        if counter == 0:
            big_df = df_temp
            counter += 1
        else:
            big_df = pd.concat([big_df, df_temp])
    return big_df.reset_index().drop('index', axis = 1)
```

Using this builder, test run on chlorophyll to see if it works:

In [6]:

```
chl = big_dataset_builder(file_list_chl, file_path_chl)
chl
```

Out[6]:

	Date	Value	Site_ID
0	-0.549398	0.151442	318
1	-0.472289	2.170673	318
2	-0.385542	0.353365	318
3	-0.308434	3.180288	318
4	-0.231325	2.170673	318
...	...	...	...
1326	10.741562	7.457273	14
1327	10.875439	7.386420	14
1328	11.274064	7.491223	14
1329	11.461187	7.123253	14
1330	11.667143	7.054298	14

1331 rows × 3 columns

It works so, create 3 dataframes each containing chl, sal, temp

In [7]:

```
#Chloropyll
print("Length of Chloropyll Data:", len(chl))
chl["Data_Type"] = ["Chlorophyll"] * len(chl)
display(chl.head(2))

#Salinity
sal = big_dataset_builder(file_list_sal, file_path_sal)
print("Length of Salinity Data:", len(sal))
sal["Data_Type"] = ["Salinity"] * len(sal)
display(sal.head(2))

#Temperature
temp = big_dataset_builder(file_list_temp, file_path_temp)
print("Length of Temperature Data:", len(temp))
temp["Data_Type"] = ["Temperature"] * len(temp)
display(temp.head(2))
```

```
Length of Chloropyll Data: 1331
```

	Date	Value	Site_ID	Data_Type
0	-0.549398	0.151442	318	Chlorophyll
1	-0.472289	2.170673	318	Chlorophyll

```
Length of Salinity Data: 1598
```

	Date	Value	Site_ID	Data_Type
0	0.358649	33.604009	6	Salinity
1	0.377120	33.320781	6	Salinity

```
Length of Temperature Data: 5931
```

	Date	Value	Site_ID	Data_Type
0	-0.153585	14.225521	97	Temperature
1	-0.158516	12.908090	97	Temperature

Successfully created all 3 datasets with values, time to add them together for later use.

In [8]:

```
env_df = pd.concat([chl, sal, temp])
env_df = env_df.rename(columns = {'Date': 'Normalized Date'})
env_df
```

Out[8]:

	Normalized Date	Value	Site_ID	Data_Type
0	-0.549398	0.151442	318	Chlorophyll
1	-0.472289	2.170673	318	Chlorophyll
2	-0.385542	0.353365	318	Chlorophyll
3	-0.308434	3.180288	318	Chlorophyll
4	-0.231325	2.170673	318	Chlorophyll
...	...	...	...	...
5926	0.884478	15.272427	71	Temperature
5927	0.950274	14.582943	71	Temperature
5928	0.998773	14.585183	71	Temperature
5929	1.026338	10.292826	71	Temperature
5930	1.057564	11.679310	71	Temperature

8860 rows × 4 columns

Fix all Negative Dates:

In [9]:

```
fix_neg = [1 + i if i < 0 else i for i in env_df['Normalized Date']]
print(all(i >= 0 for i in fix_neg))
env_df['Normalized Date'] = fix_neg
env_df
```

```
True
```

Out[9]:

	Normalized Date	Value	Site_ID	Data_Type
0	0.450602	0.151442	318	Chlorophyll
1	0.527711	2.170673	318	Chlorophyll
2	0.614458	0.353365	318	Chlorophyll
3	0.691566	3.180288	318	Chlorophyll
4	0.768675	2.170673	318	Chlorophyll
...	...	...	...	...
5926	0.884478	15.272427	71	Temperature
5927	0.950274	14.582943	71	Temperature
5928	0.998773	14.585183	71	Temperature
5929	1.026338	10.292826	71	Temperature
5930	1.057564	11.679310	71	Temperature

8860 rows × 4 columns

In [10]:

```
# new_dates = []
# for i in env_df["Normalized Date"]:
#     while i > 1.0833333:
#         i -= 1
#     new_dates.append(m.floor(i * 12))
# env_df["Normalized Date"] = new_dates
# env_df
```

Export this to a csv file:

In [11]:

```
file_path_df = '/Users/kenneth/Desktop/repro_database-main/data/env_data.csv
env_df.to_csv(file_path_df, sep=',', index=False, encoding='utf-8')
```

Let's try to make the data look a little better in a different data frame.

In [12]:

```
piv_env = env_df.pivot_table(values = 'Value', index = ["Site_ID", "Normaliz
piv_env = piv_env.reset_index()
piv_env = piv_env.rename_axis(None, axis=1).fillna(0)
piv_env
```

Out[12]:

	Site_ID	Normalized Date	Chlorophyll	Salinity	Temperature
0	1	4.000000	0.0	28.783536	0.000
1	1	5.000000	0.0	28.819698	0.000
2	1	6.000000	0.0	35.105508	0.000
3	1	7.000000	0.0	33.390196	0.000
4	1	8.000000	0.0	34.424729	0.000
...	...	...	...	...	...
8697	343	0.582143	0.0	0.000000	30.868
8698	343	0.664286	0.0	0.000000	29.865
8699	343	0.832143	0.0	0.000000	25.898
8700	343	0.917857	0.0	0.000000	21.254
8701	343	1.085714	0.0	0.000000	18.967

8702 rows × 5 columns

Export the dataframe for further use:

In [13]:

```
file_path_df = '/Users/kenneth/Desktop/repro_database-main/data/piv_env_data
piv_env.to_csv(file_path_df, sep=',', index=False, encoding='utf-8')
```

Combined All Environmental Data!!

Time to combine families, reprodata, and env_data

In [14]:

```
env_data = pd.read_csv('/Users/kenneth/Desktop/repro_database-main/data/env_
families = pd.read_csv('/Users/kenneth/Desktop/repro_database-main/data/fami
reprodata = pd.read_csv('/Users/kenneth/Desktop/repro_database-main/data/rep

env_data = env_data.rename(columns= {"Site_ID": "SiteID"})
```

Lets check the data to see how we can combine them

In [15]:

```
print(env_data.shape)
env_data.head(3)
```

```
(8860, 4)
```

Out[15]:

	Normalized Date	Value	SiteID	Data_Type
0	0.450602	0.151442	318	Chlorophyll
1	0.527711	2.170673	318	Chlorophyll
2	0.614458	0.353365	318	Chlorophyll

In [16]:

```
print(families.shape)
families.head(3)
```

```
(232, 4)
```

Out[16]:

	Unnamed: 0	db	query	family
0	1	itis	Abra alba	Semelidae
1	2	itis	Abra nitida	Semelidae
2	3	itis	Abra tenuis	Semelidae

In [17]:

```
print(reprodata.shape)
reprodata.head(3)
```

```
(541, 19)
```

Out[17]:

	SiteID	Study	Species	Locality	LatDeg	LatMin	LongDeg	LongMin	spaw
0	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0	
1	2	Adachi 1979	Ruditapes phillipinarum	Inage, Japan	35	36.0	140	3.0	
2	3	Ahn et al 2003	Laternula elliptica	Marian Cove, Antarctica	-62	13.0	-58	47.0	

Combine all the information from above to create and export the large dataframe

In [18]:

```
df = reprodata.merge(env_data, on= 'SiteID', how= 'left')
df = df.drop(columns = ["Unnamed: 18"])

family_dic = {}
for i in np.arange(len(families)):
    row = families.iloc[i, :].values
    family_dic[row[2]] = row[3]

family_col = []
for i in np.arange(len(df)):
    row = df.iloc[i, :].values
    if row[2] in family_dic.keys():
        family_col.append(family_dic[row[2]])
    else:
        family_col.append(np.NaN)
df["Family"] = family_col
df
```

Out[18]:

	SiteID	Study	Species	Locality	LatDeg	LatMin	LongDeg	LongMin
0	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
1	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
2	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
3	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
4	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
...	...	...	...	...	...	...	...	...
9992	560	Cledon et al 2004`	Tagelus plebeius	Mar Chiquita, Argentina	-37	44.0	-57	26.0
9993	561	Drescher et al 2019	Rangia cuneata	Johnson Bayou, MS	30	20.0	-89	14.0
9994	562	Cain 1975	Rangia cuneata	James River, Virginia	37	13.0	-76	43.0
9995	563	Jovanovich and Marion 1989	Rangia cuneata	Dog River, Mobile, Alabama	30	34.0	-88	5.0
9996	564	Fairbanks 1963	Rangia cuneata	Lake Ponchartrain, LA	30	7.0	-90	6.0

9997 rows × 22 columns

We will change up the dataset a bit to make it better

In [19]:

```
file_path_df = '/Users/kenneth/Desktop/repro_database-main/clams.csv'
df.to_csv(file_path_df, sep=',', index=False, encoding='utf-8')
```

Now lets try to do the same for the pivoted table:

In [20]:

```
piv_env_data = pd.read_csv('/Users/kenneth/Desktop/repro_database-main/data/
piv_env_data = piv_env_data.rename(columns= {"Site_ID": "SiteID"})
print(piv_env_data.shape)
piv_env_data.head(3)
```

```
(8702, 5)
```

Out[20]:

	SiteID	Normalized Date	Chlorophyll	Salinity	Temperature
0	1	4.0	0.0	28.783536	0.0
1	1	5.0	0.0	28.819698	0.0
2	1	6.0	0.0	35.105508	0.0

In [21]:

```
df2 = reprodata.merge(piv_env_data, on= 'SiteID', how= 'left')
df2.drop(columns = ["Unnamed: 18"])

family_dic = {}
for i in np.arange(len(families)):
    row = families.iloc[i, :].values
    family_dic[row[2]] = row[3]

family_col = []
for i in np.arange(len(df2)):
    row = df2.iloc[i, :].values
    if row[2] in family_dic.keys():
        family_col.append(family_dic[row[2]])
    else:
        family_col.append(np.NaN)
df2["Family"] = family_col
df2
# family_dic = {}
# family_col = []
# for i in range(len(df2)):
#     row = df2.iloc[i, :]
#     if df2.iloc[i, 2] in family_dic:
#         family_col.append(family_dic[df2.iloc[i, 2]][0])
#     else:
#         family_col.append(0)
# df2["Family"] = family_col
# df2
```

Out[21]:

	SiteID	Study	Species	Locality	LatDeg	LatMin	LongDeg	LongMin
0	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
1	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
2	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
3	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
4	1	Abraham 1953	Meretrix casta	Adyar river mouth	13	1.0	80	16.0
...	...	...	...	...	...	...	...	...
9817	560	Cledon et al 2004`	Tagelus plebeius	Mar Chiquita, Argentina	-37	44.0	-57	26.0
9818	561	Drescher et al 2019	Rangia cuneata	Johnson Bayou, MS	30	20.0	-89	14.0
9819	562	Cain 1975	Rangia cuneata	James River, Virginia	37	13.0	-76	43.0
9820	563	Jovanovich and Marion 1989	Rangia cuneata	Dog River, Mobile, Alabama	30	34.0	-88	5.0
9821	564	Fairbanks 1963	Rangia cuneata	Lake Ponchartrain, LA	30	7.0	-90	6.0

9822 rows × 24 columns

Let's export this file as well:

In [22]:

```
file_path_df = '/Users/kenneth/Desktop/repro_database-main/piv_clams.csv'
df2.to_csv(file_path_df, sep=',', index=False, encoding='utf-8')
```

The big Dataset has been created! Now time to do actual Data Science