Gaussian Process V0.2

May 16, 2021

1 System Setup

[1]:
```
import sys
# !{sys.executable} -m pip install --upgrade pip
# !{sys.executable} -m pip install GPy
# !{sys.executable} -m pip install seaborn
```

[2]:
```
import numpy as np
import os
import matplotlib.pyplot
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
import matplotlib.ticker as mticker
import matplotlib.dates as mdates
import datetime
from tqdm import tqdm
import GPy
from collections import defaultdict
from pathlib import Path
import seaborn as sns
import scipy.stats as stats

mpl.rcParams['legend.frameon'] = False
mpl.rcParams['figure.autolayout'] = True
mpl.rcParams['figure.dpi'] = 300
# mpl.rcParams['axes.spines.right'] = False
# mpl.rcParams['axes.spines.top'] = False


plt.rcParams.update({
    "text.usetex": True,
    "font.family": "sans-serif",
    "font.sans-serif": ["Helvetica"]})

plt.rcParams.update({
    "text.usetex": True,
```

```
    "font.family": "serif",
    "font.serif": ["Palatino"],
})


def utkarshGrid():
    plt.minorticks_on()
    plt.grid(color='grey',
             which='minor',
             linestyle=":",
             linewidth='0.1',
             )
    plt.grid(color='black',
             which='major',
             linestyle=":",
             linewidth='0.1',
             )
```

2 Data Wrangling

[3]:
```
class AllData:
    """Load the data from the set path and prepare it in a useable format.
    """
    def __init__(self):
        self.folder_name = "/bns_m3_3comp" # Change folder name of data as␣
↪required.

    def load_path(self, path_to_dir):
        """ User defined path
        """
        self.folder_path = path_to_dir
        self.path = path_to_dir + self.folder_name
        return None

    def load_raw_data(self):
        """ Loads raw data from given path. Implimentation may different for␣
↪windows and mac/linux users.
        >>> data = AllData()
        >>> data.load_path("/Users/utkarsh/PycharmProjects/SURP2021")
        >>> data.load_raw_data()
        >>> print(data.raw_data.file_name.iloc[0])
        nph1.0e+06_mejdyn0.001_mejwind0.130_phi45.txt
        >>> data.raw_data.file_name.iloc[192] == "nph1.0e+06_mejdyn0.
↪001_mejwind0.090_phi0.txt"
        True
```

```
        >>> data.raw_data.file_name.iloc[192] == "nph1.0e+06_mejdyn0.
↪005_mejwind0.110_phi0.txt"
        False
        >>> data.raw_data.file_name.iloc[192] == "nph1.0e+06_mejdyn0.
↪005_mejwind0.110_phi0.txt"
        False
        """
        resd = defaultdict(list)
        folder_path = Path(self.path)
        for file in folder_path.iterdir():
            with open(file, "r") as file_open:
                resd["file_name"].append(file.name)
        temp_df = pd.DataFrame(resd)
        self.raw_data = temp_df[temp_df.file_name != ".DS_Store"].
↪reset_index(drop=True)
        return None


    def process(self):
        """ Processes the data to a readable reference dataframe.
        >>> data = AllData()
        >>> data.load_path("/Users/utkarsh/PycharmProjects/SURP2021")
        >>> data.load_raw_data()
        >>> data.process()
        >>> print(data.reference_data.mejwind.iloc[68])
        0.03
        >>> print(data.reference_data.mejdyn.iloc[173])
        0.02
        >>> data.reference_data.phi.iloc[55] == 75
        False
        >>> data.reference_data.phi.iloc[57] == 75
        False
        >>> data.reference_data.phi.iloc[56] == 75
        True
        """
        split_series = self.raw_data.file_name.apply(lambda x: x.split('_'))
        temp_df = split_series.apply(pd.Series)
        temp_df["file_name"] = self.raw_data.file_name
        temp_df.columns = ["nph", "mejdyn", "mejwind", "phi", "filename"]
        temp_df["mejdyn"] = temp_df["mejdyn"].str.extract("(\d*\.?\d+)",␣
↪expand=True)
        temp_df["mejwind"] = temp_df["mejwind"].str.extract("(\d*\.?\d+)",␣
↪expand=True)
        temp_df["phi"] = temp_df["phi"].str.extract("(\d*\.?\d+)", expand=True)
        temp_df["nph"] = temp_df["nph"].apply(lambda x: float(x[3:]))
        temp_df[["mejdyn", "mejwind", "phi"]] = temp_df[["mejdyn", "mejwind",␣
↪"phi"]].apply(pd.to_numeric)
```

```
        self.reference_data = temp_df.reset_index(drop=True)
        return None

    def save_reference(self):
        """ Saves the reference data into a file for future use.
        """
        try:
            self.reference_data.to_csv("reference.csv", index = False)
            print("[STATUS] Reference Saved")
        except Exception:
            print("[ERROR] Reference Unsaved")

    def load_reference(self, name):
        """ Loads the saved dataframe to save on computing time.
        >>> data = AllData()
        >>> data.load_reference("reference.csv")
        >>> print(data.reference_data.mejwind.iloc[68])
        0.03
        >>> print(data.reference_data.mejdyn.iloc[173])
        0.02
        >>> data.reference_data.phi.iloc[55] == 75
        False
        >>> data.reference_data.phi.iloc[57] == 75
        False
        >>> data.reference_data.phi.iloc[56] == 75
        True
        """
        self.reference_data = pd.read_csv(name)
```

3 Light Curve Selection

[4]:
```
class LightCurve():
    """ The information regarding KNe light curves and data corresponding to␣
↪KNe light curves.
    """

    def __init__(self, referenceName):
        """ Initializes class, reference is all the light curves, and selected␣
↪represents ones of interest to be narrowed.
        """
        self.reference = pd.read_csv(referenceName)
        self.selected = self.reference.copy()

    def _slice(self, typ, Min, Max):
```

```
        sliced = self.selected[self.selected[typ] >= Min]
        sliced2 = sliced[sliced[typ] <= Max]
        return sliced2

    def select_curve(self, phiRange, mejdynRange, mejwindRange, nphRange =␣
↪[1e6]):
        """ Select a measurment based on the physics limits required.
        >>> data = LightCurve("reference.csv")
        >>> phi_range = [30]
        >>> mejdyn_range = [0.01]
        >>> mejwind_range = [0.11]
        >>> data.select_curve(phiRange = phi_range, mejdynRange = mejdyn_range,␣
↪mejwindRange = mejwind_range)
        >>> print(data.selected.filename.iloc[0])
        nph1.0e+06_mejdyn0.010_mejwind0.110_phi30.txt
        """
        self.selected = self._slice("nph", min(nphRange), max(nphRange))
        self.selected = self._slice("phi", min(phiRange), max(phiRange))
        self.selected = self._slice("mejdyn", min(mejdynRange),␣
↪max(mejdynRange))
        self.selected = self._slice("mejwind", min(mejwindRange),␣
↪max(mejwindRange))
        return None

    def _set_path(self):
        """ Sets the path to the file to be extracted. Chooses first file if␣
↪there are many.
        """
        self.folder_path = os.getcwd() + "/bns_m3_3comp/"
        self.path = self.folder_path + self.selected.filename.iloc[0]
        if len(self.selected.filename) > 1:
            print(f"[WARNING] Many curves in data: First curve has been␣
↪selected. \n[CURVE] {self.selected.filename.iloc[0]}")

        return None

    def extract_curve(self):
        """ Extracts curve based on selected data and converts it into a␣
↪readable format.
        >>> data = LightCurve("reference.csv")
        >>> phi_range = [60]
        >>> mejdyn_range = [0.02]
        >>> mejwind_range = [0.11]
        >>> data.select_curve(phiRange = phi_range, mejdynRange = mejdyn_range,␣
↪mejwindRange = mejwind_range)
        >>> data.extract_curve()
```

```
        >>> data.curve.shape
        (11, 500)
        >>> zBand = 910
        >>> plotDf = data.curve.loc[:, [zBand]]
        >>> print(plotDf.loc[1,zBand][3])
        0.0028678
        >>> print(data.selected.filename.iloc[0])
        nph1.0e+06_mejdyn0.020_mejwind0.110_phi60.txt
        """
        # Obtain path to read curve from.
        self._set_path()

        # Read txt file containig light curve information
        temp0 = pd.read_csv(self.path, header = None, names = ["data"])

        # Set parameters for viewing angles, numbers of wavelengths, and time␣
↪step.
        self.Nobs = int(temp0.data.iloc[0])
        self.Nwave = float(temp0.data.iloc[1])
        self.Ntime = list(map(float, temp0.data.iloc[2].split()))

        # Drop information header and reset index.
        temp1 = temp0.iloc[3:].reset_index(drop = True)

        # Convert data from string to float
        temp1["data"] = temp1["data"].apply(lambda x: list(map(float, x.
↪split())))

        # Obtain wavelength from messy data list. Convert to nm
        temp1.loc[:, 'wavelength'] = temp1.data.map(lambda x: x[0]/10)

        # Remove wavelengths from data vector.
        temp1["data"] = temp1["data"].apply(lambda x: x[1:])

        # Pivot to order the table by wavelengths
        temp1 = temp1.pivot(columns = "wavelength", values = "data")

        # Concatenate all rows to remove NA values to get a neat, readable␣
↪dataframe.
        final = pd.concat([temp1[col].dropna().reset_index(drop=True) for col␣
↪in temp1], axis=1)

        # Rename axis titles.
        final.index.name = "iobs"
        final.columns.name = "wavelength"
        self.curve = final
```

```
        return None

    def _odd(self,x):
        """Rounds to nearest odd numbers
        >>> data = LightCurve("reference.csv")
        >>> data._odd(3)
        3
        >>> data._odd(2.5)
        3
        >>> data._odd(2)
        3
        >>> data._odd(1.999)
        1
        """
        return 2 * int(x/2) + 1

    def simple_plot(self, wv):
        print("[STATUS] Plotting...")
        self.time_arr = np.linspace(int(self.Ntime[1]), int(self.Ntime[2]),␣
↪int(self.Ntime[0]), endpoint = True)
        self.wavelength = 10*self._odd(wv/10)

        viewing_angles = np.linspace(0, 1, data.Nobs, endpoint = True)
        plt.figure()
        plt.gca().set_prop_cycle("color", sns.color_palette("coolwarm_r",self.
↪Nobs))
        for i,j in self.curve.loc[:, [self.wavelength]].iterrows():
            ang = round(np.degrees(np.arccos(viewing_angles[i])), 2)
            plt.plot(self.time_arr, j.values[0], label = f"{ang}"r"$^o$",␣
↪linewidth = 1)
        plt.xlabel("Time (Days)")
        plt.ylabel(r"Flux $Erg s^{-1} cm^{-2}A^{-1}$")
        plt.title(f"Lights curves for {self.Nobs} viewing angles at {self.
↪wavelength}nm")
        utkarshGrid()
        plt.legend(title = r"$\Phi$")
        return None
```

4 Script

[5]:
```
# if __name__ == "__main__":
import doctest
doctest.testmod()
```

[5]:
```
TestResults(failed=0, attempted=45)
```

[6]:
```
# Parse Data
initial = AllData()

# Only needs to be done the first time
initial.load_path("/Users/utkarsh/PycharmProjects/SURP2021")
initial.load_raw_data()
initial.process()
initial.save_reference()
initial.reference_data
```

```
[STATUS] Reference Saved
```

[6]:

	nph	mejdyn	mejwind	phi	\
0	1000000.0	0.001	0.13	45	
1	1000000.0	0.010	0.05	15	
2	1000000.0	0.001	0.01	75	
3	1000000.0	0.005	0.09	75	
4	1000000.0	0.020	0.11	60	
..	…	…	…	…	
191	1000000.0	0.005	0.11	0	
192	1000000.0	0.001	0.09	0	
193	1000000.0	0.020	0.07	0	
194	1000000.0	0.010	0.05	30	
195	1000000.0	0.001	0.13	60	

	filename
0	nph1.0e+06_mejdyn0.001_mejwind0.130_phi45.txt
1	nph1.0e+06_mejdyn0.010_mejwind0.050_phi15.txt
2	nph1.0e+06_mejdyn0.001_mejwind0.010_phi75.txt
3	nph1.0e+06_mejdyn0.005_mejwind0.090_phi75.txt
4	nph1.0e+06_mejdyn0.020_mejwind0.110_phi60.txt
..	…
191	nph1.0e+06_mejdyn0.005_mejwind0.110_phi0.txt
192	nph1.0e+06_mejdyn0.001_mejwind0.090_phi0.txt
193	nph1.0e+06_mejdyn0.020_mejwind0.070_phi0.txt
194	nph1.0e+06_mejdyn0.010_mejwind0.050_phi30.txt
195	nph1.0e+06_mejdyn0.001_mejwind0.130_phi60.txt

```
[196 rows x 5 columns]
```

[7]:
```
# Selecting a singular light curve

data = LightCurve("reference.csv")
phi_range = [30,60]
mejdyn_range = [0.01, 0.02]
```

```
mejwind_range = [0.11]
data.select_curve(phiRange = phi_range,
                  mejdynRange = mejdyn_range,
                  mejwindRange = mejwind_range)
data.selected
```

[7]:

	nph	mejdyn	mejwind	phi	filename
4	1000000.0	0.02	0.11	60	nph1.0e+06_mejdyn0.020_mejwind0.110_phi60.txt
88	1000000.0	0.01	0.11	60	nph1.0e+06_mejdyn0.010_mejwind0.110_phi60.txt
106	1000000.0	0.01	0.11	45	nph1.0e+06_mejdyn0.010_mejwind0.110_phi45.txt
123	1000000.0	0.02	0.11	30	nph1.0e+06_mejdyn0.020_mejwind0.110_phi30.txt
159	1000000.0	0.01	0.11	30	nph1.0e+06_mejdyn0.010_mejwind0.110_phi30.txt
186	1000000.0	0.02	0.11	45	nph1.0e+06_mejdyn0.020_mejwind0.110_phi45.txt

[8]:
```
data.extract_curve()
data.curve.head(2) # iobs goes from 0 --> 10 (11 rows)
```

```
[WARNING] Many curves in data: First curve has been selected.
[CURVE] nph1.0e+06_mejdyn0.020_mejwind0.110_phi60.txt
```

[8]:

wavelength	10.0
iobs	
0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength	30.0
iobs	
0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength	50.0
iobs	
0	[4.0387e-15, 1.2203e-06, 2.4324e-07, 7.9271e-0…
1	[2.9736e-15, 1.3601e-06, 2.3303e-07, 8.1683e-0…

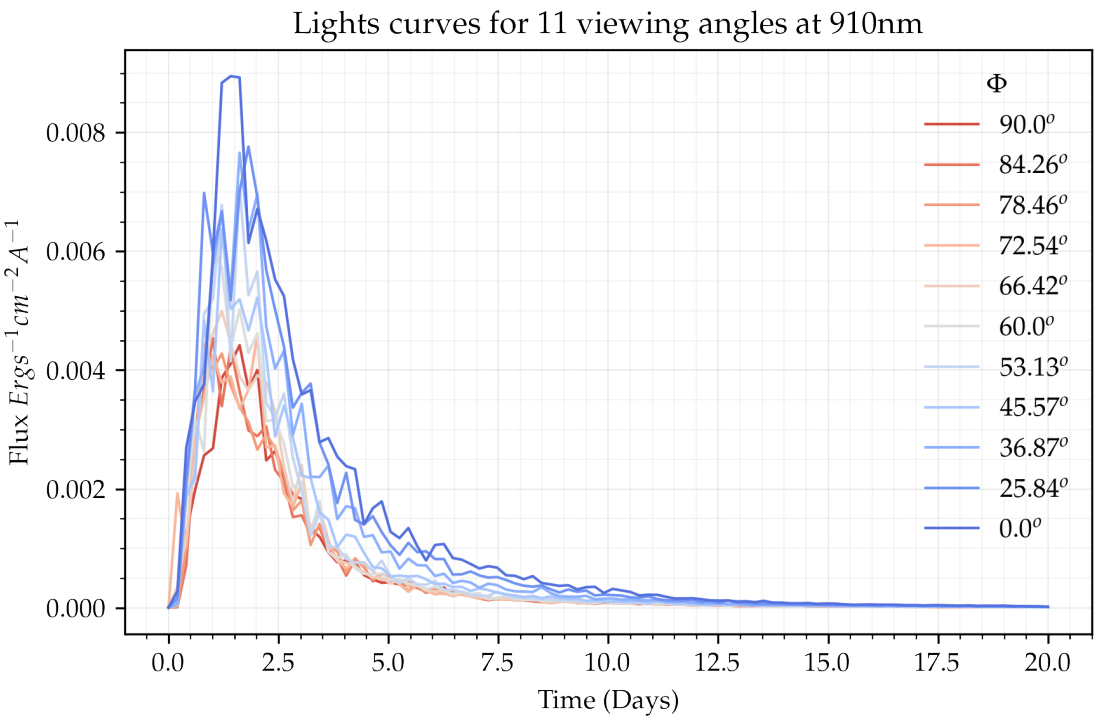
wavelength	70.0
iobs	
0	[1.4257e-12, 3.9016e-05, 1.2179e-05, 3.1122e-0…
1	[2.43e-12, 3.9696e-05, 1.2017e-05, 3.0895e-06,…

```
wavelength                                                 90.0    \
iobs
0           [3.2954e-10, 0.00017416, 0.00010168, 3.2372e-0…
1           [5.3195e-10, 0.00017769, 0.00010233, 3.2907e-0…

wavelength                                                110.0   \
iobs
0           [3.4215e-08, 0.0027166, 0.00045816, 0.00013985…
1           [5.1259e-08, 0.0027128, 0.00046453, 0.00014821…

wavelength                                                130.0   \
iobs
0           [1.2376e-06, 0.00091114, 0.00071721, 0.0004897…
1           [1.769e-06, 0.00090998, 0.00073516, 0.00041236…

wavelength                                                150.0   \
iobs
0           [2.0326e-05, 0.0029708, 0.0013356, 0.0011571, …
1           [2.2727e-05, 0.0030483, 0.0013946, 0.0011474, …

wavelength                                                170.0   \
iobs
0           [8.5044e-05, 0.0047244, 0.0021553, 0.0011496, …
1           [8.6299e-05, 0.0048463, 0.0020631, 0.0011407, …

wavelength                                                190.0   …  \
iobs                                                               …
0           [0.00015567, 0.0015238, 0.0059957, 0.0013954, …  …
1           [0.00014803, 0.0015737, 0.0059852, 0.0017754, …  …

wavelength                                                9810.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 2.4802e-08, 0.0, 0.0…
1           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength                                                9830.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 2.9992e-08, 0.0, 0.0, 0.0…
1           [0.0, 0.0, 0.0, 0.0, 0.0, 1.2769e-08, 0.0, 0.0…

wavelength                                                9850.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1           [0.0, 0.0, 0.0, 0.0, 4.4778e-05, 0.0, 0.0, 0.0…

wavelength                                                9870.0  \
```

```
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength                                                9890.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.8119e-08…
1           [0.0, 0.0, 0.0, 0.0, 2.8966e-08, 0.0, 0.0, 0.0…

wavelength                                                9910.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.7845e-08…

wavelength                                                9930.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength                                                9950.0  \
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …
1           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength                                                9970.0  \
iobs
0           [0.0, 0.0, 0.0, 3.1167e-08, 0.0, 0.0, 0.0, 0.0…
1           [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, …

wavelength                                                9990.0
iobs
0           [0.0, 0.0, 0.0, 0.0, 0.0, 3.827e-05, 0.0, 0.0,…
1           [0.0, 0.0, 0.0, 3.1445e-08, 0.0, 0.0, 0.0, 0.0…

[2 rows x 500 columns]
```

[9]:
```
zBand = 900.0
lBand = 3450
data.simple_plot(zBand)
```

```
[STATUS] Plotting…
```



Plot is based on viewing angle. The plot is also general for any wavelength.

5 Gaussian Process

=== NOT PROPER YET ===

[10]:
```
gp = LightCurve("reference.csv")
phi_range = [45]
mejdyn_range = [0.01]
mejwind_range = [0.11]
gp.select_curve(phiRange = phi_range,
                  mejdynRange = mejdyn_range,
                  mejwindRange = mejwind_range)
gp.extract_curve()
z = gp.curve.T[gp.curve.T.index == 910]
z = z.reset_index(drop = True)
z = z.apply(pd.Series.explode).reset_index(drop = True)
time_arr = np.linspace(gp.Ntime[1], gp.Ntime[2], int(gp.Ntime[0]), endpoint =␣
↪True)
z["time"] = time_arr
z.index.name = "time_step"
print(z.shape)
z
```

(100, 12)

[10]:
iobs	0	1	2	3	4	5
time_step						
0	0.000008	0.000075	0.000068	0.000074	0.000002	0.000002
1	0.000821	0.000162	0.0002	0.00032	0.000109	0.000424
2	0.001215	0.000896	0.0011	0.001182	0.001163	0.000646
3	0.002716	0.002916	0.001421	0.001165	0.002129	0.003567
4	0.001977	0.002169	0.001102	0.002042	0.003423	0.002785
…	…	…	…	…	…	…
95	0.000024	0.000023	0.000023	0.000023	0.000022	0.000023
96	0.000018	0.000019	0.000019	0.000019	0.000019	0.000019
97	0.000016	0.000014	0.000015	0.000014	0.000016	0.000018
98	0.000014	0.000014	0.000014	0.000016	0.000015	0.000016
99	0.000013	0.000012	0.000013	0.000013	0.000013	0.000014

iobs	6	7	8	9	10	time
time_step						
0	0.000002	0.000001	0.000001	0.000001	0.000001	0.000000
1	0.000436	0.00078	0.000874	0.002856	0.000297	0.202020
2	0.001896	0.001691	0.002209	0.001234	0.00303	0.404040
3	0.003088	0.003888	0.005563	0.004848	0.004608	0.606061
4	0.00437	0.005405	0.00737	0.005397	0.003681	0.808081
…	…	…	…	…	…	…
95	0.000024	0.000027	0.000025	0.000027	0.00003	19.191919
96	0.000019	0.000021	0.000025	0.000028	0.000025	19.393939
97	0.000021	0.000023	0.000022	0.000022	0.000023	19.595960
98	0.000015	0.000014	0.000018	0.000022	0.000026	19.797980
99	0.000016	0.000021	0.000019	0.000017	0.000021	20.000000

[100 rows x 12 columns]

[11]:
```
# Q5
# Get time closest to one day.
time_ind = np.argmin(np.abs(time_arr-1)) # Not sure what to do with "one day"

delta = 0
one_day = z.iloc[time_ind - delta: time_ind + delta + 1] # select around one day
del one_day["time"] # dont need time after choosing our time frame
one_day
```

[11]:
iobs	0	1	2	3	4	5
time_step						
5	0.002709	0.002237	0.003472	0.003301	0.004151	0.005073

iobs	6	7	8	9	10
time_step					

```
5          0.004041  0.005399  0.005223  0.005627  0.007347
```

[12]:
```
# Q6, Q7
# Median normalization
def med_norm(df):
    med = np.median(df)
    return df.divide(med) - 1

def med_norm_arr(arr):
    med = np.median(arr)
    return arr/med - 1

normed = med_norm(one_day)
normed
```

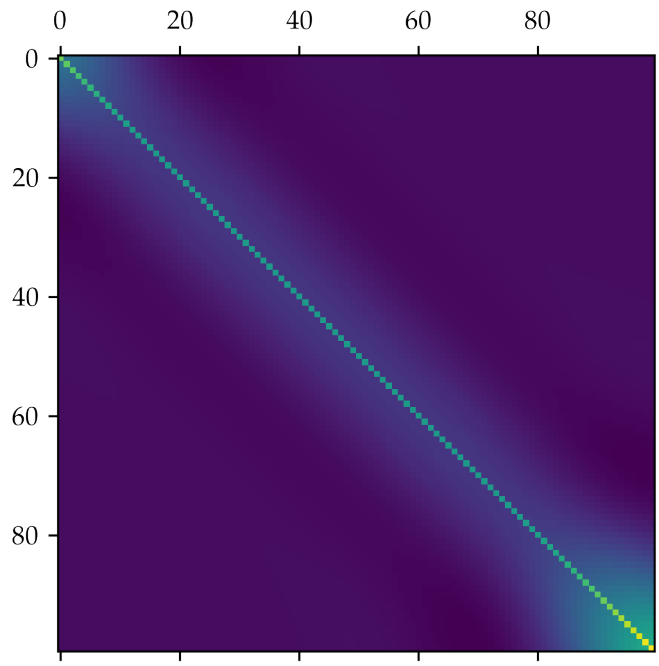
[12]:
```
iobs             0        1         2         3    4         5        6   \
time_step
5         -0.347282 -0.46114 -0.163462 -0.204852  0.0  0.222102 -0.02638

iobs             7         8        9         10
time_step
5          0.300713  0.258215  0.35557  0.770141
```

[13]:
```
# Q8, Q9
X = np.arange(0, gp.Nobs, 1)
Y = np.array(normed.iloc[0])
X = X.reshape(len(X), 1)
Y = Y.reshape(len(Y), 1)
kernel = GPy.kern.RBF(input_dim=1, variance = 2, lengthscale=2) # Variance
m = GPy.models.GPRegression(X, Y, kernel)
```

[14]:
```
# Q10
predX = np.linspace(-1,12,100).reshape(100, 1)
predY_mean, predY_cov = m.predict(predX,  full_cov=True)

plt.figure()
plt.matshow(predY_cov)
plt.show()
```

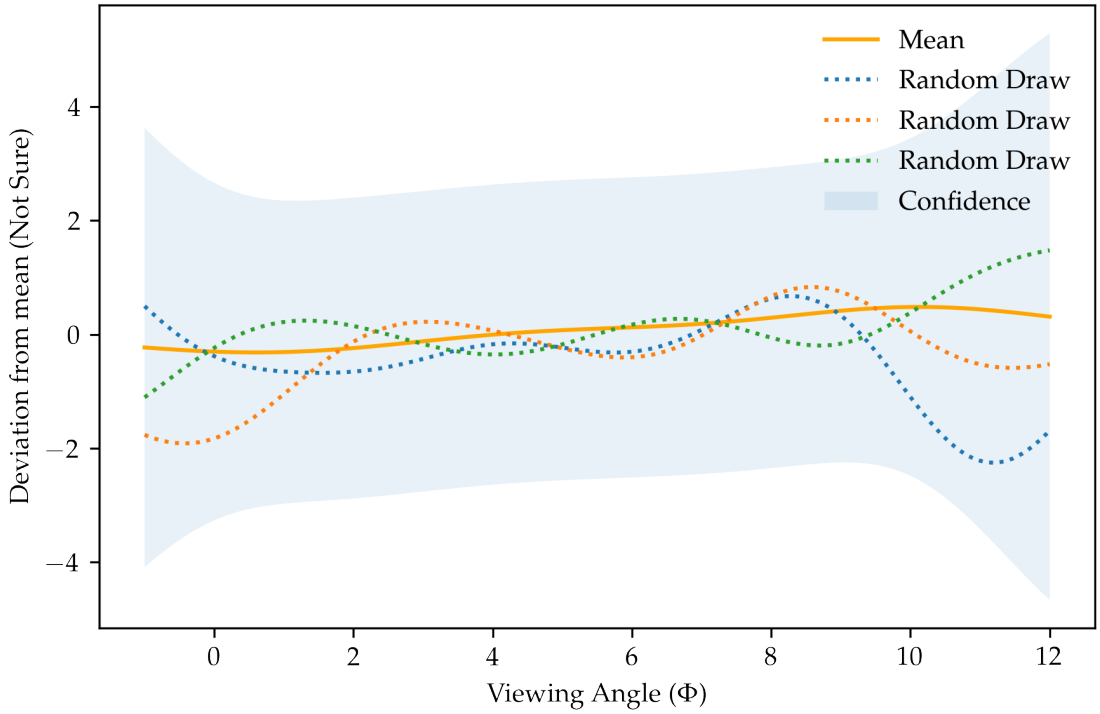
```
<Figure size 1800x1200 with 0 Axes>
```

```
 /Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-
packages/IPython/core/pylabtools.py:132: UserWarning:This figure includes Axes
that are not compatible with tight_layout, so results might be incorrect.
```



[15]:
```
# I DID NOT USE THIS METHOD
# Multidmensional version of scipy normal
# Inject predY_var
# Random Draw from normal
# mu = 0
# sigma = 1
# x = np.linspace(mu - 3*sigma, mu + 3*sigma, 100)
# plt.plot(x, stats.norm.pdf(x, mu, sigma), label = "Unit Gaussian")


def custom_plot(predX):
    # Custom Plotting Function
    posteriorPredY = m.posterior_samples_f(predX, full_cov=True, size=3)
    predY, predY_cov = m.predict(predX, full_cov = True)
    var = np.diag(predY_cov)
    plotX = predX.reshape(1, len(predX))[0]
    plotY = predY.reshape(1, len(predY))[0]

    plt.figure()
    n = 2
    numVar = n * var
    plt.fill_between(plotX, plotY + numVar, plotY - numVar, alpha = 0.1, label␣
 ↪= "Confidence")
    plt.plot(plotX, plotY, label = "Mean", color = "orange")
```

```
    plt.plot(predX, posteriorPredY[:,0, 0], label= "Random Draw", linestyle =␣
↪"dotted")
    plt.plot(predX, posteriorPredY[:,0, 1], label= "Random Draw", linestyle =␣
↪"dotted")
    plt.plot(predX, posteriorPredY[:,0, 2], label= "Random Draw", linestyle =␣
↪"dotted")
    plt.xlabel(r"Viewing Angle ($\Phi$)")
    plt.ylabel("Deviation from mean (Not Sure)")
    plt.legend()
    plt.show()

custom_plot(predX)
```



[16]:
```
# Q11, Q12
display(m)
m.optimize(messages=True)
# m.optimize_restarts(num_restarts = 10)
```

```
<GPy.models.gp_regression.GPRegression at 0x7fab8ca16370>
```

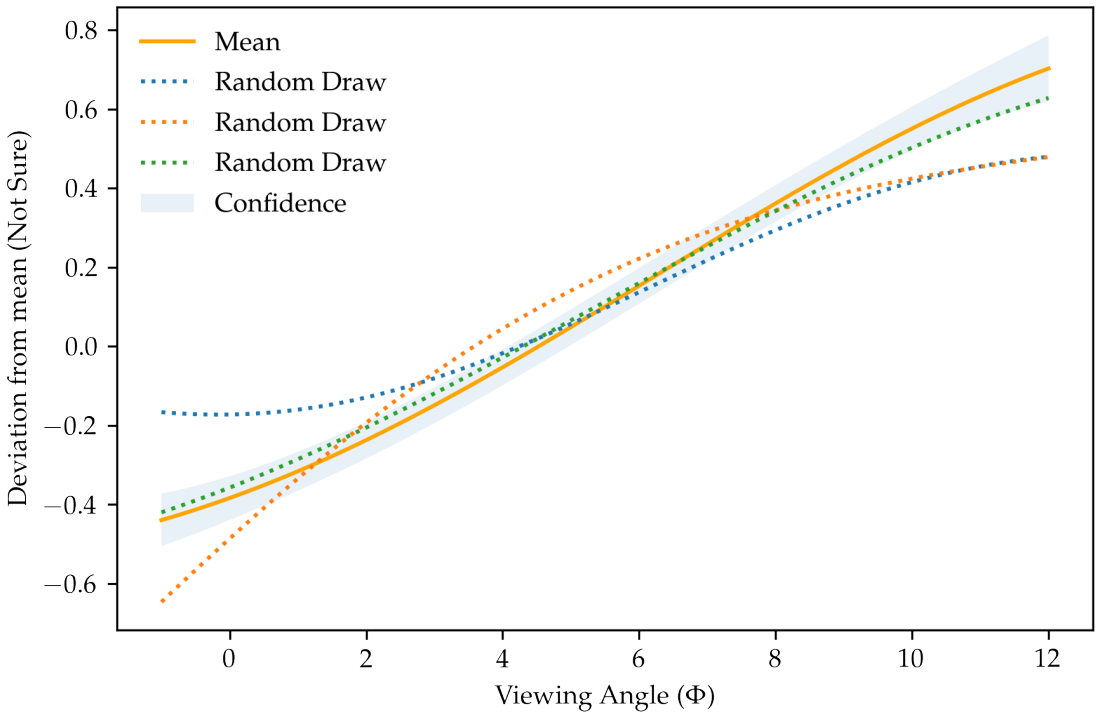
```
Running L-BFGS-B (Scipy implementation) Code:
  runtime   i      f              |g|
    00s00  0004   9.887497e+01   9.774507e+03
    00s02  0011  -1.242646e+00   2.152484e-01
```

```
    00s03  0018  -1.329754e+00   3.420510e-12
    00s04  0019  -1.329754e+00   3.420510e-12
Runtime:     00s04
Optimization status: Converged
```

[16]:
```
<paramz.optimization.optimization.opt_lbfgsb at 0x7fab88a300d0>
```

[17]:
```
# predX = np.linspace(0,11,100).reshape(100, 1)
# predY = m.predict(predX)[0] - m.predict(predX)[1] # Should I be applying a␣
 ↪random variance in pred iteratively
# # predY = med_norm_arr(predY) # Why does turning this off make a difference?

# print(m)

# m.plot(samples = 3)
# plt.plot(predX,predY, color = "red", label = "Prediction")
# plt.ylim(-4,4)
# plt.legend()

print(m)

custom_plot(predX)
```

```
Name : GP regression
Objective : -1.3297537698986739
Number of Parameters : 3
Number of Optimization Parameters : 3
Updates : True
Parameters:
  GP_regression.           |                value  |  constraints  |
priors
  rbf.variance             |    0.5048741357115837  |      +ve      |
  rbf.lengthscale          |    10.237731281822857  |      +ve      |
  Gaussian_noise.variance  |  0.019581516431695353  |      +ve      |
```



Q13) Comments about fidelity of the emulator prediction.

[18]:
```
#Q14
hist = []
vec = np.array(normed.iloc[0])
# print("Vector:",vec)

for i in range(11):
# for i in range(1):
    loo = np.delete(vec, i)
    trueY = vec[i:i+1]
    if trueY in loo:
        print("[ERROR] Element left out is in your training data")
        break


    X = np.arange(0, gp.Nobs - 1, 1)
    Y = loo
    X = X.reshape(len(X), 1)
    Y = Y.reshape(len(Y), 1)
    kernel = GPy.kern.RBF(input_dim=1, variance = 1, lengthscale=1)
    mLoo = GPy.models.GPRegression(X, Y, kernel)

    mLoo.optimize #Maybe optimize researts is the correct thing to use here.
```

```
    predX = np.linspace(0,10,100).reshape(100, 1)
    predY_mean, predY_var = mLoo.predict(predX)
    predY_mean = predY_mean.reshape(1, len(predY_mean))[0]
    predY_var = predY_var.reshape(1, len(predY_var))[0]
    print(predY_var) # Variances are the same?
    hist.append((predY_mean - trueY)/np.sqrt(predY_var))

hist
```

```
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
```

```
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
```

```
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
```

```
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
 1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
 1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
 1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
 1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
 1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
 1.69198722 1.73494344 1.77618165 1.81462434]
[1.44892176 1.43113568 1.41794277 1.40868383 1.40258386 1.39885663
 1.39678959 1.395798   1.3954451  1.39543293 1.39557423 1.39575774
 1.39591789 1.39601503 1.39602734 1.39595075 1.39580081 1.39561131
 1.39542682 1.3952903  1.39522973 1.39524884 1.39532599 1.39542189
 1.39549387 1.39551128 1.39546638 1.39537653 1.39527692 1.39520655
 1.39519287 1.39524114 1.39533236 1.39543074 1.39549769 1.39550691
 1.3954546  1.395361   1.39526247 1.39519733 1.39519091 1.39524572
 1.3953404  1.39543809 1.39550074 1.39550395 1.39544648 1.39535071
 1.39525392 1.39519382 1.3951938  1.39525389 1.3953507  1.39544659
 1.39550429 1.39550147 1.39543942 1.39534254 1.39524883 1.39519505
 1.39520234 1.39526798 1.39536634 1.39545894 1.3955095  1.3954984
```

```
       1.39543081 1.39533546 1.3952545  1.39522822 1.39528056 1.39541039
       1.39559187 1.39578296 1.39593863 1.39602359 1.39602029 1.39593093
       1.39577562 1.39559203 1.39544247 1.3954324  1.3957399  1.39665041
       1.39858599 1.40211693 1.40794488 1.41685209 1.42961917 1.44692182
       1.46922331 1.49668228 1.5290936  1.56587464 1.60610069 1.64858449
       1.69198722 1.73494344 1.77618165 1.81462434]
```

[18]:
```
[array([0.0957874842623965, 0.09580452804680838, 0.09721731854528112,
        0.09994137848940067, 0.10384378384800697, 0.10875381565022894,
        0.11447577257285446, 0.12080262475029287, 0.12752957791356143,
        0.1344668797235158, 0.14145124174975443, 0.14835516221375272,
        0.15509339611826164, 0.16162597265041412, 0.16795752435134284,
        0.1741331620626198, 0.1802315407249045, 0.18635598586325983,
        0.19262455563984532, 0.19915975857601517, 0.20607843913285345,
        0.21348218003536426, 0.22144850465041185, 0.23002319623199052,
        0.2392141454295228, 0.24898723346850882, 0.25926478920564955,
        0.2699270637270147, 0.28081691237253903, 0.29174747962793846,
        0.3025122367827274, 0.31289637245140867, 0.3226884278469986,
        0.331691264555836, 0.3397318755523363, 0.34667000083279287,
        0.35240575835761334, 0.35688641160230694, 0.36011200222540113,
        0.36213908323007127, 0.3630814734109174, 0.3631070352991583,
        0.3624299974000339, 0.3612991335918752, 0.3599828893948974,
        0.35875303739486486, 0.357868526860057, 0.3575609172888874,
        0.3580223211979594, 0.35939631383481097, 0.36177192305935824,
        0.36518063489684033, 0.36959631820191197, 0.3749380297917048,
        0.38107573843326525, 0.38783902402424036, 0.3950286918136238,
        0.40243094361971443, 0.40983328851740997, 0.4170408684695324,
        0.4238915124519135, 0.4302678123857297, 0.4361049289005703,
        0.4413935977997278, 0.44617866668551176, 0.4505541392953523,
        0.45465593031340684, 0.45865331478976734, 0.4627395619284114,
        0.4671217401205842, 0.472009411178363, 0.47760201489734305,
        0.4840751369687404, 0.49156638248169904, 0.5001620255997938,
        0.5098857941950338, 0.5206909890458071, 0.5324566483041229,
        0.5449877637300313, 0.5580188348758527, 0.5712195699162462,
        0.5842015621746216, 0.5965254328649578, 0.6077091499693561,
        0.6172396404264378, 0.6245907896281516, 0.6292507544989531,
        0.630759663343639, 0.6287552380875673, 0.6230195388389819,
        0.6135166826478349, 0.6004110633725798, 0.5840593460616235,
        0.5649763519121735, 0.5437820767545785, 0.5211414283997599,
        0.49770833686259147, 0.4740824316381018, 0.4507816134299545,
        0.4282296318447147], dtype=object),
 array([0.2328391677229179, 0.23248099478615236, 0.2327977859281036,
        0.2337757620748258, 0.23537978787651556, 0.237553258510815,
        0.2402208381035329, 0.24329408788866408, 0.24667946928174594,
        0.250287744761149, 0.25404349174708607, 0.2578933541791912,
        0.261811832246524, 0.26580384376366295, 0.26990388218886413,
        0.27417217737341504, 0.2786886660132618, 0.28354570465948453,
```

```
       0.28884032534535253, 0.29466655017653387, 0.30110799002471555,
       0.3082307738342701, 0.3160768473475297, 0.32465782954991873,
       0.3339498492143592, 0.34388999801148856, 0.3543751251268906,
       0.3652635840220758, 0.3763802035307714, 0.38752424958543846,
       0.39847960932071436, 0.4090260536409301, 0.4189503778176636,
       0.4280565251582154, 0.4361743431923089, 0.44316715338273155,
       0.4489385697321868, 0.45343883742783647, 0.45667044256258177,
       0.4586921181033174, 0.4596199674218175, 0.45962449526821025,
       0.4589229224594566, 0.45776707399993743, 0.456428033724732,
       0.4551793328920307, 0.45428052876313196, 0.45396268311532106,
       0.45441666715992923, 0.45578463037207095, 0.4581545494097271,
       0.461557593670391, 0.4659680871853567, 0.47130602821202533,
       0.47744232853665886, 0.4842070272496168, 0.4914006145272732,
       0.498808224863983, 0.5062158767548681, 0.5134273081778308,
       0.5202795231823629, 0.5266551495262425, 0.5324902029176242,
       0.5377767418885805, 0.542560876500777, 0.5469373101759484,
       0.5510418087341327, 0.555042688555813, 0.5591317977155564,
       0.5635148446052454, 0.568400594944136, 0.5739885533207854,
       0.5804552211579103, 0.5879396733475905, 0.5965297518900945,
       0.6062504118820197, 0.6170555648862984, 0.6288241768513827,
       0.6413605471837076, 0.6543978751639519, 0.6676037157842059,
       0.6805860171415719, 0.692899247660476, 0.7040515372772044,
       0.7135153404315986, 0.7207451980982068, 0.7252059642790323,
       0.7264127986644533, 0.7239803239964757, 0.7176735686287733,
       0.7074496012272045, 0.6934783357322649, 0.6761350018115236,
       0.6559642036626506, 0.6336232806899172, 0.6098174852049031,
       0.5852396533006486, 0.5605233459378703, 0.536213172012093,
       0.5127514157285477], dtype=object),
array([-0.05686957968524405, -0.0671463857778855, -0.07642451258782954,
       -0.08450003685613611, -0.09120457592879096, -0.09640680135001306,
       -0.10001073362193699, -0.10195276752080358, -0.10219895194214917,
       -0.10074319716306925, -0.09760625061989421, -0.09283478679165846,
       -0.08649990520047732, -0.07869463207784314, -0.06953046121747633,
       -0.05913331919494085, -0.04763946015088752, -0.03519168660950219,
       -0.021936072420301787, -0.008019185065639832, 0.006414232408410854,
       0.021223058556056523, 0.036270867900573225, 0.051426031499016304,
       0.0665607197947936, 0.08154897347040063, 0.09626435073745246,
       0.11057790367578996, 0.12435726185069472, 0.1374673783149022,
       0.1497730985510805, 0.16114330090243933, 0.17145608862587233,
       0.18060447745904934, 0.18850218622378206, 0.19508936277270672,
       0.20033819340510806, 0.20425824507033793, 0.20690109846307278,
       0.20836349259114165, 0.20878801667418737, 0.20836050196479838,
       0.20730369977566201, 0.20586745081101307, 0.20431614226040612,
       0.2029146289918886, 0.20191389266311963, 0.20153758075658026,
       0.20197032524804495, 0.2033484938849295, 0.20575382308241413,
       0.20921021428673547, 0.21368382361679705, 0.21908643167157346,
       0.2252819566998868, 0.23209587091564823, 0.23932716681531307,
```

```
       0.24676234666129465, 0.25419064378129147, 0.26141936769151297,
       0.26828801522707485, 0.27467975836121933, 0.2805292012543096,
       0.28582585776330827, 0.29061346356626383, 0.2949857743893747,
       0.29907974282656635, 0.3030668815533302, 0.3071433255036798,
       0.3115187905761001, 0.31640446222655694, 0.3219999114348592,
       0.32847939249082725, 0.3359782110537459, 0.34458012560310874,
       0.35430685502928494, 0.36511065646211904, 0.3768706085896292,
       0.3893927357765642, 0.40241354979268584, 0.41560615159067504,
       0.4285879434492626, 0.44092941357270876, 0.4521643574631937,
       0.46180302276020524, 0.469350491626871, 0.4743325182088421,
       0.47632952562416003, 0.4750165195901107, 0.47020305418652725,
       0.4618646272248202, 0.4501567253781819, 0.4354060488329364,
       0.41807934347324344, 0.39873632387045393, 0.3779767713137003,
       0.3563918107199267, 0.334526296693964, 0.31285501555460515,
       0.29177179733316017], dtype=object),
array([-0.02294234878425054, -0.032789348761149265, -0.041598921152543294,
       -0.04915127029273281, -0.055262591863942105, -0.059788811231479326,
       -0.06262582548930652, -0.0637079588979189, -0.06300604207468098,
       -0.06052578216101227, -0.056306336027635163, -0.050418548545302797,
       -0.04296228246606391, -0.03406255545930529, -0.02386460375882669,
       -0.012528296783556492, -0.00022241621083936842,
       0.012880792658515642, 0.02661056745030582, 0.04080195686040496,
       0.05529939572871716, 0.06995928079628609, 0.0846512120149215,
       0.09925761775070846, 0.11367172594705537, 0.1277942194716161,
       0.14152928399571474, 0.15478096851482392, 0.16745073713136704,
       0.17943680157225203, 0.1906353927339825, 0.20094371939686242,
       0.21026412302059538, 0.21850893497188148, 0.2256057225148944,
       0.23150282275555212, 0.2361751419013897, 0.23963004682749484,
       0.2419128335943621, 0.24311088146280097, 0.2433553984906278,
       0.24281978883768943, 0.24171413717719517, 0.24027597315393298,
       0.23875812287322842, 0.23741487515404613, 0.23648780430321448,
       0.23619244761242159, 0.2367067601175829, 0.23816198253308024,
       0.24063632572686333, 0.24415170407132172, 0.2486736217443951,
       0.2541142152165682, 0.2603383740095305, 0.2671727829748219,
       0.27441761212840626, 0.2818603741665613, 0.28929115695516183,
       0.2965180750249857, 0.3033815065087031, 0.3097656517337022,
       0.3156062610994055, 0.3208939845555313, 0.32567350203158313,
       0.33003915642670867, 0.3341280482447061, 0.33811143632618873,
       0.34218494950772715, 0.34655775733481037, 0.35144066078208697,
       0.3570331328475724, 0.3635096265662429, 0.37100584616301013,
       0.3796059909935429, 0.38933210939844165, 0.4001365797317396,
       0.4118983710847787, 0.42442319038534637, 0.4374470276474007,
       0.45064216704614063, 0.4636246646464876, 0.47596276156005585,
       0.48718667490630074, 0.4968013954393533, 0.504304981695804,
       0.5092147278897049, 0.51110199309639, 0.5096333989666383,
       0.5046123207823903, 0.4960117057706372, 0.483989057541334,
       0.46887783351512996, 0.4511556131233487, 0.4313956911153729,
```

```
       0.4102125182006351, 0.38821136439711185, 0.3659494198987087,
       0.3439111799601304, 0.3224972117033642], dtype=object),
array([-0.19227868151975694, -0.20280244653116866, -0.21199957575613412,
       -0.2197000615903952, -0.22578883819056092, -0.23020223455406147,
       -0.2329216921651019, -0.2339674608510429, -0.2333938628578896,
       -0.23128630723251198, -0.22775913260999994, -0.22295288707630573,
       -0.21702985328594646, -0.210167291725178, -0.20254867694257633,
       -0.19435382779703966, -0.18574910347755924, -0.17687875154519236,
       -0.1678581889543976, -0.15876964982909378, -0.14966037365415197,
       -0.14054337182934448, -0.1314007564056681, -0.12218956317940953,
       -0.1128498835545433, -0.10331490715941397, -0.09352218957533626,
       -0.08342515728813717, -0.07300363239011, -0.06227209478076663,
       -0.05128456444728733, -0.040135381958493206, -0.02895570969083772,
       -0.017906125902630833, -0.007166096137036308, 0.003078688289004233,
       0.012650082154306578, 0.02139137960997866, 0.029180340466769917,
       0.03594030705459752, 0.041648333578253434, 0.04633935033261204,
       0.05010569978541285, 0.053091887114709316, 0.055484963349011464,
       0.057501452795047286, 0.05937204622357698, 0.06132539543091422,
       0.06357231810648355, 0.06629161730255233, 0.06961856317600834,
       0.07363686479803512, 0.07837466464361804, 0.08380474106658835,
       0.08984876452029991, 0.09638518194806038, 0.1032601186787775,
       0.11030055246615617, 0.11732887382537752, 0.12417777951172507,
       0.1307043001774503, 0.13680173840344562, 0.14240847514329263,
       0.14751299440601415, 0.15215497725611482, 0.15642276869988958,
       0.16044779321371108, 0.16439654860213435, 0.16846070812356104,
       0.1728457267051923, 0.17775828365199992, 0.18339294378665721,
       0.18991855517142853, 0.1974650603373888, 0.2061115168259211,
       0.21587616156407974, 0.22670928851452898, 0.23848952037954763,
       0.2510237339843771, 0.26405048132614073, 0.2772463496396298,
       0.29023450982926396, 0.3025948911747226, 0.3138760395738186,
       0.32360958486475166, 0.33132893644314576, 0.3365937923668201,
       0.3390208359842775, 0.3383185716364437, 0.3343212648673889,
       0.3270147199821594, 0.31654662024753216, 0.303217077472953,
       0.28745009605990446, 0.26975175856461386, 0.2506638811443606,
       0.2307216742344094, 0.2104212139755832, 0.19019887896815282,
       0.1704218308394889], dtype=object),
array([-0.3770840302679547, -0.3887729842530343, -0.3988359848655284,
       -0.4071278592264397, -0.41357065630029693, -0.4181422456646152,
       -0.42086343524713443, -0.4217875619479661, -0.42099451191444726,
       -0.41858896107085997, -0.4147010524321799, -0.40948711510171576,
       -0.4031283672046111, -0.395826528272727, -0.3877964410618521,
       -0.379256729979858, -0.3704199355947856, -0.3614834404610538,
       -0.35262202727452524, -0.3439823607321, -0.33567929050317535,
       -0.327793739427114, -0.3203720354654128, -0.3134267420700621,
       -0.3069391907991976, -0.3008639137598219, -0.2951349775247827,
       -0.2896738762815021, -0.2843982485507011, -0.2792303657363256,
       -0.2741042233258779, -0.26897022192964043, -0.2637968465967751,
```

```
       -0.2585693329945315, -0.25328587640824596, -0.24795232566109335,
       -0.2425764147530237, -0.23716243384276503, -0.23170693099323783,
       -0.22619570544398854, -0.22060211829586754, -0.21488666197605538,
       -0.20899777710084555, -0.2028740076991577, -0.19644764305591172,
       -0.18964992504917713, -0.18241767543680046, -0.17470085890065026,
       -0.16647024656450468, -0.15772411220351, -0.14849289015560366,
       -0.138840987764319, -0.12886541086808187, -0.11869137795380492,
       -0.10846549811270512, -0.09834726505255212, -0.0884995820932551,
       -0.0790788859025114, -0.07022531190826893, -0.06205332407289235,
       -0.05464330795470443, -0.04803471713550746, -0.04222137308448762,
       -0.0371493988937679, -0.032718043114712665, -0.02878339579421291,
       -0.0251647866095514, -0.021653512677265258, -0.01802345268441502,
       -0.014043051123337813, -0.00948808932623356, -0.004154617603383605,
       0.002128567573150l476, 0.009488454330908051, 0.017997828342760345,
       0.02766719985821386, 0.038439089981278436, 0.050185145335733976,
       0.06270649362735277, 0.07573754350036681, 0.08895309274738473,
       0.10197828155073095, 0.114400814385584, 0.12578510987055205,
       0.1356885615480214, 0.14368060493627927, 0.14936533619180675,
       0.15240762381734135, 0.15256094076358867, 0.14969301625633752,
       0.14380388365927227, 0.13503109460270019, 0.1236392641249656,
       0.10999503067327791, 0.09453232316912998, 0.07771487247423738,
       0.060002509651488414, 0.04182552799526711, 0.023568510365693116,
       0.0055626751943684246], dtype=object),
array([-0.1706703330384252, -0.18106383505543358, -0.1901403906332812,
       -0.1977191820010065, -0.2036728104010832, -0.2079252074354304,
       -0.21044680482809394, -0.21124955441058027, -0.21038335377714384,
       -0.20793408661814722, -0.20402240504795568, -0.19880189794382142,
       -0.19245544896336697, -0.18518919790067412, -0.17722426218185397,
       -0.16878695061532917, -0.16009844324295214, -0.15136483270073,
       -0.14276816891007713, -0.1344588944798595, -0.12654991288672937,
       -0.11911250339284293, -0.1121743172718795, -0.10571967394287023,
       -0.09969227396974312, -0.09400026377235002, -0.08852336625157722,
       -0.08312158039580339, -0.07764478694548654, -0.07194250105331851,
       -0.06587300904547043, -0.05931123280952446, -0.052154877884859445,
       -0.044328696274429334, -0.03578695313929676, -0.026514346137339447,
       -0.01652564750967265, -0.005864250922059 8596, 0.00540030686180154,
       0.017175819231548862, 0.02935176537082703, 0.04180426090177809,
       0.05440114672546172, 0.06700670757876584, 0.07948563677987258,
       0.09170613034299623, 0.10354228678894051, 0.11487617246019584,
       0.12559990080418124, 0.1356178790380835, 0.144849101018806,
       0.15322915777198567, 0.16071161185727806, 0.1672685600685958,
       0.17289050484914004, 0.17758591802517867, 0.181380974126968,
       0.18431979978545202, 0.18646528278145366, 0.18790014194305185,
       0.18872772750087635, 0.18907199895342822, 0.1890763147735622,
       0.1889009702468263, 0.18871969492223206, 0.18871545384418967,
       0.18907585096756663, 0.18998826055697055, 0.1916346154475056,
       0.19418565867841606, 0.19779447079892226, 0.20258921553260756,
```

```
       0.20866525677219713, 0.21607703236278697, 0.2248302750547842,
       0.234875309202541, 0.2461021847564183, 0.2583382971077424,
       0.271348861072913, 0.2848402000845311, 0.2984654107366343,
       0.31183177334521645, 0.32450948839840726, 0.33604196915658463,
       0.3459588134383349, 0.3537932701266426, 0.35910594177078153,
       0.36151517154873125, 0.3607320267882891, 0.356594685300979,
       0.34909471092768346, 0.33838768243812106, 0.3247836373347122,
       0.3087179941004201, 0.2907088701257688, 0.2713097498263489,
       0.2510662535602715, 0.2304829652057653, 0.2100025331229485,
       0.1899961029465658], dtype=object),
array([-0.44239210092397746, -0.45446308234209054, -0.46480254200014065,
       -0.47327657174274296, -0.47982344589854314, -0.4844394228565098,
       -0.48716347097379653, -0.48806529606789983, -0.48723873892874553,
       -0.4848001785432051, -0.48088984195506695, -0.4756732674158897,
       -0.46934056106939553, -0.46210218938772857, -0.45418136102808965,
       -0.44580409196354764, -0.4371885199548639, -0.4285349095214823,
       -0.42001726771419434, -0.41177687831545295, -0.4039176216232174,
       -0.39650280886477174, -0.38955340087100554, -0.3830477642060181,
       -0.37692336960751893, -0.37108091394839005, -0.3653911806690747,
       -0.3597045658815479, -0.353862686508311, -0.3477110038270782,
       -0.34111110585039595, -0.33395131924469135, -0.32615468826443733,
       -0.31768395210654216, -0.3085437560523563, -0.2987807167038731,
       -0.28848199695390264, -0.2777727646574872, -0.2668124883268508,
       -0.25578969810338154, -0.24491478865795238, -0.23441070134172526,
       -0.2245017831049028, -0.21540157891383047, -0.207300590339887,
       -0.2003550491669522, -0.19467755981801318, -0.1903301788512306,
       -0.1873202351712986, -0.18559899909886804, -0.18506316800763434,
       -0.1855590161049166, -0.1868889424992702, -0.18882006176291283,
       -0.19109443543842675, -0.1934405308433422, -0.19558546452097092,
       -0.19726748286781978, -0.19824793848070774, -0.19832180621540574,
       -0.19732568328700748, -0.19514236557219433, -0.1917015296558996,
       -0.18697668094437542, -0.1809791498580221, -0.17375032080854308,
       -0.16535335086150457, -0.15586541218911437, -0.14537112495252716,
       -0.1339575139203783, -0.12171064080018987, -0.10871404294960388,
       -0.09504915597330292, -0.08079788005554618, -0.06604727104212467,
       -0.05089599227835871, -0.03546174720364481, -0.019888580376368824,
       -0.0043528296388277935, 0.010933304373961037, 0.02572119746118043,
       0.039728421738731826, 0.05264443311795936, 0.0641392340988519,
       0.07387520063497974, 0.0815225479916461, 0.08677881976635188,
       0.0893919934081922, 0.08918527748004405, 0.08607987941372253,
       0.08011083168815993, 0.07143131194233572, 0.06030313623775126,
       0.04707459291166793, 0.03215006810419925, 0.01595758321718359,
       -0.001080107183863204 2, -0.01856726848154973, -0.03614872946234837,
       -0.05351617184724368], dtype=object),
array([-0.40708590981578796, -0.41893816167513076, -0.42911275999077586,
       -0.43746976438412727, -0.4439389952269931, -0.4485074083342271,
       -0.45120517654739534, -0.45209462305360926, -0.4512640077107745,
```

```
       -0.44882587607454605, -0.44491803103685545, -0.4397045551258015,
       -0.4333746749145807, -0.42613829723184943, -0.41821828467796673,
       -0.4098405180123106, -0.401223235654015, -0.3925670222158986,
       -0.38404633246654246, -0.37580287151269176, -0.3679407502195916,
       -0.3605232112320316, -0.35357084604445504, -0.3470614683564772,
       -0.34093201488104563, -0.33508288747593395, -0.3293849763505877,
       -0.32368923733554517, -0.3178382269232962, -0.311678561923637,
       -0.30507301355185273, -0.2979109796649892, -0.290116422713908,
       -0.28165291087611327, -0.27252595586614337, -0.2627831931165222,
       -0.25251298089927976, -0.24184173912563564, -0.2309299745677946,
       -0.21996666448322671, -0.20916165445011886, -0.1987359944143489,
       -0.18891057455297566, -0.17989383915547072, -0.17186958850192924,
       -0.16498586797152326, -0.15934574962981254, -0.15500055327986373,
       -0.1519458299663531, -0.15012027114635132, -0.1494075810671728,
       -0.1496412156718089, -0.15061173798484465, -0.1520763963907626,
       -0.1537704353268079, -0.15541960118700876, -0.15675326709864104,
       -0.157517511176675, -0.15748732666069135, -0.15647697949698755,
       -0.15434748281914512, -0.1510103463367862, -0.14642721123566504,
       -0.1406055978174637, -0.13359158111216163, -0.12546058081846934,
       -0.11630751860465936, -0.10623740671043502, -0.0953571301450052,
       -0.08376892189782618, -0.07156588758372161, -0.05882989312440146,
       -0.04563209220524129, -0.03203623398072158, -0.018104603879304945,
       -0.003906045973521503, 0.010474895806400026, 0.024928958994312005,
       0.039314544988594415, 0.053452550629945327, 0.06712428204012563,
       0.08007265865540734, 0.09200667604176394, 0.10260915563803932,
       0.11154810627611374, 0.11849229495324214, 0.12313148995310044,
       0.1252009732072973, 0.1245082928061226, 0.12095830118507613,
       0.11457121930150667, 0.1054887695746371, 0.09396573305057053,
       0.08034795469743185, 0.065041332620484, 0.04847818049674617,
       0.031086955330820513, 0.013269241747025056, -0.0046147551436710055,
       -0.022253095249663738], dtype=object),
array([-0.4879646622729152, -0.5003180133817281, -0.5108704186046749,
       -0.5194957848297838, -0.5261433265817759, -0.530821334830933,
       -0.5335801502622011, -0.5344990181573201, -0.5336789799940336,
       -0.531241342014131, -0.5273294166716613, -0.5221105497989453,
       -0.5157758789810649, -0.5085364529557342, -0.50061574845479,
       -0.4922397388179096, -0.48362617917465395, -0.4749746396668051,
       -0.4664582517057595, -0.4582174586656272, -0.4503555717194477,
       -0.442935774428904, -0.43597937996766184, -0.429465478162747,
       -0.4233324198184261, -0.41748170491232045, -0.41178468409648206,
       -0.40609206673530945, -0.40024566277613555, -0.3940912420017463,
       -0.38749106104148884, -0.3803346269381013, -0.37254666226041494,
       -0.36409188721321883, -0.35497689966412355, -0.3452498614801131,
       -0.3349987426788513, -0.3243485646474597, -0.3134576105005972,
       -0.30251219022676856, -0.2917194606131213, -0.281298059111314,
       -0.2714668066359497, -0.2624322544764598, -0.2543761904761411,
       -0.2474442784942345, -0.2417368143177936, -0.23730225772249863,
```

```
       -0.23413386682846596, -0.23216948752691344, -0.23129434933006113,
       -0.23134657470596506, -0.23212501339918268, -0.23339896975513777,
       -0.23491939315549953, -0.23643111085376253, -0.23768563421710007,
       -0.23845391304404523, -0.2385381625027607, -0.23778164580541586,
       -0.23607522187768185, -0.2333596946080765, -0.2296235488410829,
       -0.22489639869317438, -0.21923917635210793, -0.2127325351238028,
       -0.20546502502570377, -0.19752236978143337, -0.1889787838557126,
       -0.17989088772756434, -0.17029451772690254, -0.16020459603051726,
       -0.14961815845946647, -0.13852053130621025, -0.12689442487390717,
       -0.1147313550888584, -0.10204437622133812, -0.08888072911125963,
       -0.07533282213964891, -0.06154606942757554, -0.047722516259535906,
       -0.03411976080740955, -0.021045215186001304, -0.0088460291765494,
       0.0021050413116705793, 0.011427748195886019, 0.018756409728977658,
       0.02376536862241314, 0.026196109834292176, 0.025882711309806693,
       0.02277108358346477, 0.016927516732468412, 0.008533869896409238,
       -0.002130128785795364, -0.014713813828515863,
       -0.028826201387738577, -0.04406481319767244, -0.06003887603114627,
       -0.07638534245829347, -0.09277813951334844], dtype=object),
array([-0.8323747494457912, -0.8468615779765332, -0.8590223466368938,
       -0.8687899327728172, -0.8761961499795468, -0.8813401635173472,
       -0.8843582290302008, -0.8854016983657268, -0.884626102468098,
       -0.8821901297568889, -0.8782606455315319, -0.8730189967685358,
       -0.8666645742531424, -0.8594134103905159, -0.8514917164014189,
       -0.8431259675680958, -0.8345319443986825, -0.8259049405399719,
       -0.8174124373482625, -0.8091894119354026, -0.8013355765582075,
       -0.793913548076557, -0.7869472566346657, -0.7804206133241199,
       -0.7742772117057051, -0.7684222832403347, -0.7627280423837762,
       -0.7570429307831626, -0.7512042923742476, -0.7450530115650874,
       -0.7384479880104876, -0.7312782725509115, -0.7234713065938249,
       -0.714996784570471, -0.7058667887974313, -0.696133591145197,
       -0.6858866065535868, -0.675249435502018, -0.664377030756048,
       -0.6534521882451878, -0.6426801667761254, -0.6322804607486079,
       -0.6224754929911189, -0.6134769546622512, -0.6054713182649862,
       -0.5986064065632879, -0.5929807321249104, -0.588636742538063,
       -0.5855583400209474, -0.5836723283911467, -0.5828529574730846,
       -0.5829285823398087, -0.5836896327384143, -0.5848974965666869,
       -0.5862943758150374, -0.5876144551439192, -0.5885966587994399,
       -0.588998814069526, -0.588612314340913, -0.5872756595008093,
       -0.5848848798378239, -0.581399064842381, -0.576840038058972,
       -0.5712863950717838, -0.5648632363222081, -0.5577295865635966,
       -0.5500655110076078, -0.5420604021625678, -0.533903115360831,
       -0.5257739250956379, -0.5178379129470371, -0.510239450073572,
       -0.5030977826887462, -0.49650413016719824, -0.49052091944782,
       -0.4851836587102798, -0.48050549610445337, -0.47648384941289673,
       -0.47310784731743344, -0.4703649241797764, -0.46824493310032755,
       -0.4667406446536586, -0.46584439264819555, -0.4655416993581937,
       -0.4658036872469758, -0.4665807028373017, -0.4677996390120577,
```

```
-0.46936681719840073, -0.47117696984313506, -0.4731270515444507,
-0.4751318151609027, -0.47713705407430307, -0.47912675041279656,
-0.4811221289594386, -0.4831731061487784, -0.48534476212787697,
-0.4877024277161511, -0.49029863126569684, -0.4931639664090386,
-0.496302586483753], dtype=object)]
```

[19]:
```
plt.figure()
plt.hist(hist, bins = 2)
mu = 0
sigma = 1
x = np.linspace(mu - 3*sigma, mu + 3*sigma, 100)
plt.plot(x, stats.norm.pdf(x, mu, sigma), label = "Unit Gaussian")
plt.legend()
plt.show()
```