```python
# -*- coding: utf-8 -*-
"""
Created on Fri Jun 18 14:23:59 2021

@author: fredr
"""

import numpy as np
import matplotlib.pyplot as plt

def read_from_file_raw():
    time_pos_array= np.empty((0,2))  #time_pos_array is a 2xN matrix, with pairs of t and x to s
    try:
        with open("A2_volts_cropped3.txt", 'r') as file_obj: #"A2_volts.txt"
            for line in file_obj:
                to_append = np.array(line.split(","), dtype = float)
                if to_append[1]== float(np.nan):
                    print("Not a number")
                    continue
                time_pos_array = np.append(time_pos_array, np.array([to_append]), axis=0)
            time_pos_array = time_pos_array.transpose() #Transpose making it a list of times and
            print("DONE READING")
    except Exception as e:
        print("Error when reading file. Message: \n", e)
    return time_pos_array

def find_indicies(time_pos_array, *time_points):
    time_list = time_pos_array[0]
    index_list = np.empty((0,1), dtype = int)
    for time_point in time_points:
        for i in range(len(time_list)):
            t = float(time_list[i])
            if(time_point < t):
                index_list = np.append(index_list, np.array([int(i)]))
                break
    print("INDEX AND VALUE:")
    [print(i, ": ", time_list[i]) for i in index_list]
    return index_list

def plot(array_1, array_2, label_x, label_y, style = "o-", c = "blue"):
    plt.xlabel(label_x)
    plt.ylabel(label_y)
    plt.plot(array_1, array_2, style, color = c)
    plt.show()
    return 1

def evaluate_spin(time_pos_array, *indicies):
    indicies = indicies[0]
    for index in indicies:
        plot_values = time_pos_array[:,index:(index+100)]
        # plot(plot_values[0], plot_values[1], "x", "t", style = "o", c = "red")
        base_value = time_pos_array[1][index] #The starting value for magnetization
        time = 0
        seconds_on_value = np.empty((0,1), dtype = int) #How long it stays on a value before cha
        for i in range(1, len(plot_values[0])):
            m_value = plot_values[1][i] #float(plot_values[0][i]), float(plot_values[0][i-1])
            if m_value == base_value:
                time += 4.5*10**(-4)
                continue
            elif (m_value == float(np.nan)):
                continue
```

```
            #NEW VALUE FOUND
            seconds_on_value = np.append(seconds_on_value, np.array([time]))
            base_value = m_value
            time = 0
        #For hver verdi i seconds_in_value så har den byttet mellom bunnpunkt og topppunkt, så d
        average_time_between_shits = np.mean(seconds_on_value)
        print("TIME BETWEEN SHIFTS AT {} is: {} \n Gives an angular frequency {}".format(time_po
    return 0

# def evaluate_spin_2(time_pos_array, *indicies):
#     indicies = indicies[0]
#     for index in indicies:
#         plot_values = time_pos_array[:,index:(index+100)]
#         print(plot_values[0])#[:-1]-plot_values[0][1:])
        # rotation_time = np.average(plot_values[0]-plot_values[0][1:-1])
        # rotation_ang_freq = 2*np.pi / rotation_time
        # print("Ang freq at time {} is: {}".format(time_pos_array[0][index], rotation_ang_freq)

def main():
    time_pos_array = read_from_file_raw()
    index_list = find_indicies(time_pos_array, 0, 50) #0, 6853
    evaluate_spin(time_pos_array, index_list)
    return 1

print("Romtek, recording spin.py called as ", __name__)
if __name__ == "__main__":
    main()
```