AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Instrukcja A4, Python-Analiza przebiegów czasowych

Plik *dane_pomiarowe.csv* zawiera pewien przebieg czasowy. Pierwsza kolumna oznaczona jako „t” zawiera wektor czasu (wartości wyrażone w sekundach) natomiast kolumna oznaczona jako „a” zawiera wektor przyśpieszeń.

1. **Wczytywanie danych z pliku przy użyciu biblioteki pandas:**

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
#Odczyt danych z pliku .csv za pomoca biblioteki pandas
df = pd.read_csv (r'dane_pomiarowe.csv', sep="\t")
#Rzutowanie danych z kolumny z czasem i kolumny z przyśpieszeniem na wektory np.array
t =np.array(df['t'])
a =np.array(df['a'])
```

Za pomocą biblioteki pandas narysuj wykres a(t) z otrzymanych danych.

2. **Przykład zapisywania danych do pliku przy użyciu biblioteki pandas:**

```python
import pandas as pd
import numpy as np
t = np.linspace(-10,10,100) #generuje wektor 100 liczb rzeczywistych od -10 do 10
y = t*t
data = {"t":t,"y":y} #tworzy słownik klucz-wartość przechowujący dane
dataframe = pd.DataFrame(data) #tworzy pandas dataframe
dataframe.to_csv("nazwa_pliku.csv" , index=False, sep="\t")
```

Proszę sprawdzić do czego służy index=False oraz sep=”\t”

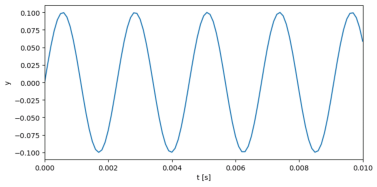
3. **Proszę zapoznać się z poniższym przykładem funkcji dopasowania:**

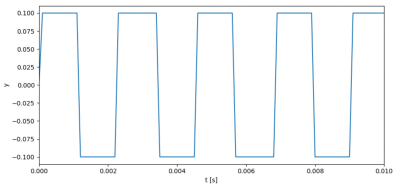
```python
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
#Generowanie punktów funkcji kwadratowej
a=2
b = -7
amp = 10 #amplitida szumu
x = np.linspace(-10,10,100) #dziedzina funkcji
y= a*x**2+b + amp*(np.random.rand(len(x))-0.5) #zbiór wartości x=y**2 plus
szum

# Fitowanie funkcji
def func(x, a, b):
       return a*x**2+b #funkcja fitująca parabolę

p0=[1, 1] #wektor inicjalizujący

fit_params, covariance_matrix = curve_fit(func, x, y,p0=p0) #fitowanie

# Parametry fitowania
print("parametry fitowania: \na=", fit_params[0], "\nb=", fit_params[1]) #
Wyświetlanie wykresów


plt.scatter(x,y) #Wygenerowane punkty


plt.plot(x, func(x, *fit_params), 'r') #Wykres funkcji dopasowania


plt.show()
```

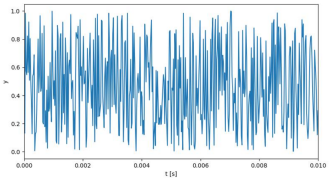
1. Proszę porównać wartości parametrów a i b z funkcji dopasowania z podanymi w kodzie, skąd wynikają różnice?
2. Jak wpływa amplituda szumu na dopasowanie, w jakich granicach fitowanie jest możliwe?
3. Proszę spróbować zmienić wartości a i b i porównać wyniki otrzymanych nowych parametrów z fitowania.
4. Proszę zamiast funkcji kwadratowej wygenerować funkcję liniową i funkcję 3 rzędu i następnie wykonać fitowanie.

4. Transformata Fouriera

Bazując na poprzednim zadaniu proszę wygenerować **sygnał sinusoidalny** o częstotliwości 440 Hzi dowolnej amplitudzie, **sygnał prostokątny** o tej samej częstotliwości i amplitudzie (*Podpowiedź: można np. skorzystać z funkcji np.sign(), która zwraca znak lub z odszukać gotową funkcję służącą do tego celu dostępną w scipy*) oraz **biały szum** (*np.random.rand)* a następnie proszę narysować jego przebieg.





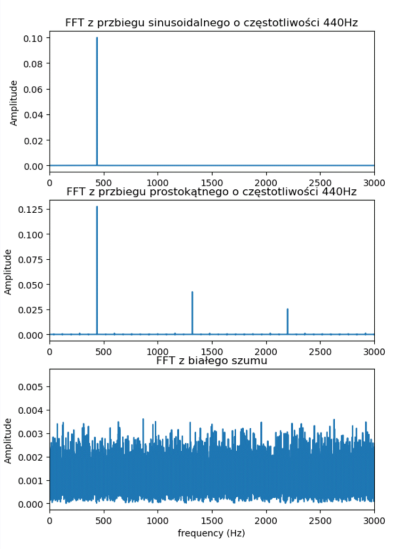


Proszę zapoznać się z poniższym przykładem analizy częstotliwościowej (widma sygnału) przy użyciu Transformaty Fouriera z biblioteki scipy.fft :

Python

```
from scipy.fft import fft
import matplotlib.pyplot as plt
import numpy as np
sampling = 44100 #częstotliwość próbkowania,https://en.wikipedia.org/wiki/44,100_Hz

time = 5 #czas trwania wygenerowanego pliku

t = np.linspace(0,time,time*sampling) # wektor czasu uwzględniający zadaną długość i częstotliwość próbkowania
A = 0.1 # amplituda (od 0 do 1)
f = 440 # częstotliwość w Hz
data = A*np.sin(2*np.pi*f*t) #generuje przebieg


def TranformataFouriera(t,y):
        N = len(t)
        dt = t[1] - t[0]
        yf = 2.0 / N * np.abs(fft(y)[0:N // 2])
        xf = np.fft.fftfreq(N, d=dt)[0:N // 2]
        return xf, yf


xf, yf = TranformataFouriera(t,data) #obliczamy Transformatę Fouriera dla wygenerowanego przebiegu

plt.plot(xf, yf)
plt.xlim(0,1000) # ogranicza wykres do zakresu od 0 do 1kHz
plt.xlabel("frequency (Hz)")
plt.ylabel("amplitude")
```

```
plt.grid()
plt.show()
```



Jako wynik działania powyższego przykładowego kodu otrzymujemy górny wykres. Reprezentuje on częstotliwość przebiegu sinusoidalnego jako szpilkę na częstotliwości 440Hz. Drugi przykład przedstawia transformatę Fouriera dla przebiegu prostokątnego o tej samej częstotliwości (widoczna częstotliwość podstawowa oraz jej nieparzyste harmoniczne). Trzeci przykład przedstawia transformatę Fouriera dla białego szumu wygenerowanego za pomocą funkcji np.random.rand()

Zadanie 1. Zmodyfikuj dane tak aby otrzymać drugi i trzeci przypadek.

Zadanie 2. Proszę wygenerować przebieg trójkątny o częstotliwości 100 Hz i amplitudzie 0.1 i dodać do niego biały szum następnie proszę wykreślić FFT dla takiej sumy sygnałów.

<u>Zadanie domowe</u> ,

Napisz klasę generator, posiadającą metody pozwalające na generowanie następujących przebiegów czasowych (https://en.wikipedia.org/wiki/Waveform):

- Sine (f, A) **(10%)**
- Square (f, A) **(10%)**
- Sawtooth (f, A) **(10%)**
- Triangle (f, A) **(10%)**
- WhiteNoise (A) **(10%)**

gdzie A – amplituda, f – częstotliwość

Podczas tworzenia obiektu, klasa powinna przyjmować częstotliwość próbkowania oraz długość trwania przebiegu w sekundach.

Klasa powinna mieć metody pozwalające na:

- Narysowanie wykresu przebiegu czasowego pobierając od użytkownika zakres czasowy np. (0-10 ms) **(10%)**
- Obliczenie i narysowanie transformaty Fouriera dla danego sygnału, **(10%)**
- Zapisanie przebiegu czasowego oraz transformaty Fouriera do pliku .csv, **(5%)**
- Zapisanie przebiegu czasowego do pliku .wav **(5%)**

Należy stworzyć program wyposażony w odpowiednie menu pozwalające na demonstrację działania stworzonej klasy, program powinien być zapętlony. **(20%)**