Analiza Szeregów Fouriera

Zaawansowane metody obliczeniowe

Autor:
Oskar Swat

3 grudnia 2023 r.

Wstęp

W ramach niniejszej pracy przeprowadzam analizę i implementację Szeregów Fouriera, znanych również jako Transformata Fouriera. Jest to technika matematyczna, która umożliwia dekompozycję funkcji na sumę sinusów i cosinusów o różnych częstotliwościach. Szeregi Fouriera są powszechnie wykorzystywane w analizie sygnałów, kompresji danych, obrazowania medycznego i wielu innych dziedzinach nauki i technologii.

Cel pracy

Celem tej pracy jest zrozumienie i zaimplementowanie algorytmów Szeregów Fouriera dla sygnałów jedno- i dwuwymiarowych. Przedstawiam różnice pomiędzy metodą dyskretnej transformacji Fouriera (DFT) a szybką transformacją Fouriera (FFT), analizując ich złożoność obliczeniową i efektywność.

Zakres pracy

Praca obejmuje analizę i implementację Szeregów Fouriera, ze szczególnym uwzględnieniem transformacji jedno- i dwuwymiarowej. Skupiam się na porównaniu dwóch głównych metod: dyskretnej transformacji Fouriera (DFT) oraz szybkiej transformacji Fouriera (FFT) w kontekście sygnałów o skończonej długości. Przedstawiam również zastosowanie tych transformacji w języku Python przy użyciu biblioteki NumPy.

Metodyka

Implementacja projektu została przeprowadzona w języku Python, wykorzystując bibliotekę NumPy do manipulacji danymi oraz Matplotlib do wizualizacji wyników. Metodologia projektu opiera się na analizie teoretycznej Szeregów Fouriera, zrozumieniu różnic między transformacją FFT a DFT, a następnie praktycznej implementacji algorytmów w oparciu o wytyczne udostępnione przez doktora habilitowanego Tomasza Gwizdałłę.

Sposób realizacji

Inicjalizacja

Początkowe parametry, takie jak wymiarowość, liczba elementów oraz elementy, są wczytywane do programu za pomocą standardowego wejścia/wyjścia danych. Dane wejściowe przechowywane są w trzech plikach: `dane_12.in`, `dane_12_a.in`, `dane2_12.in`, gdzie `dane1` odpowiada danym jednowymiarowym, `dane2` danym dwuwymiarowym, a liczba 12 oznacza numer pliku, wyznaczony przez prowadzącego.

Przykładowe dane wejściowe

Poniżej znajdują się przykładowe dane wejściowe dla obu przypadków wymiarowości:

Dane Jednowymiarowe (`dane_12.in`)

```
1     # Wymiarowość (1 dla jednowymiarowych, 2 dla dwuwymiarowych)
10    # Liczba elementów
1.0   # Element 1
2.0   # Element 2
3.0   # Element 3
4.0   # Element 4
5.0   # Element 5
6.0   # Element 6
7.0   # Element 7
8.0   # Element 8
9.0   # Element 9
10.0  # Element 10
```

Dane Dwuwymiarowe (`dane2_12.in`)

```
2     # Wymiarowość (1 dla jednowymiarowych, 2 dla dwuwymiarowych)
3 4   # Liczba elementów w wymiarze N i M
1.0 2.0 3.0 4.0   # Wiersz 1
5.0 6.0 7.0 8.0   # Wiersz 2
9.0 10.0 11.0 12.0 # Wiersz 3
```

Działanie Programu

1. Utworzenie układu o zadanych parametrach

Program inicjalizuje układ, wczytując parametry takie jak wymiarowość, liczba elementów, oraz elementy, z plików `dane_12.in`, `dane_12_a.in`, `dane2_12.in`. Dane wejściowe przechowywane są w tablicach `Fi` oraz `C`.

2. Wykonanie Odpowiednich Operacji

W zależności od wybranej transformacji (DFT lub FFT) oraz wymiarowości układu, program wykonuje następujące operacje:

- **DFT (Dyskretna Transformata Fouriera):** Dla jednowymiarowych danych, program stosuje dyskretną transformację Fouriera przy użyciu wzoru (1). Złożoność obliczeniowa jest równa $O(N^2)$.
- **IDFT (Odwrócona Dyskretna Transformata Fouriera):** Dla jednowymiarowych danych, program stosuje odwróconą dyskretną transformację Fouriera przy użyciu wzoru (2). Złożoność obliczeniowa jest równa $O(N^2)$.
- **FFT (Szybka Transformacja Fouriera):** Dla jednowymiarowych danych, program stosuje szybką transformację Fouriera przy użyciu algorytmu Cooleya-Tukeya (3), który znacząco redukuje liczbę operacji, złożoność obliczeniową wynoszącą $O(N \log N)$.
- **IFFT (Odwrócona Szybka Transformata Fouriera):** Dla jednowymiarowych danych, program stosuje odwróconą szybką transformację Fouriera przy użyciu algorytmu Cooleya-Tukeya (4). Złożoność obliczeniowa jest równa $O(N \log N)$.
- **Wypisanie Operacji Dominujących:** Program wypisuje ilość operacji dominujących, tj. ilość podstawowych operacji matematycznych wykonywanych podczas przetwarzania danych. Informacja ta pomaga ocenić efektywność algorytmu.
- **FFT dla Dwuwymiarowych Danych:** Dla dwuwymiarowych danych, program stosuje szybką transformację Fouriera przy użyciu bibliotecznej funkcji `np.fft.fft2`.

3. Badanie szumu

W przypadku jednowymiarowych danych, program wykonuje dodatkowe operacje w celu zbadania szumu:

- Usuwa wartości skrajne z danych `C`,
- Oblicza logarytm z wartości bezwzględnej `C`,
- Pokazuje linię trendu dla danych po usunięciu wartości skrajnych.

4. Pokazanie Wszystkich Wykresów

Program generuje różne wykresy prezentujące dane wejściowe i wyjściowe w formie graficznej. Dla jednowymiarowych danych są to wykresy funkcji przed i po transformacji, modułów wartości, oraz linię trendu. Dla dwuwymiarowych danych dodatkowo generowane są wykresy trójwymiarowe dla danych wejściowych i wyjściowych.

Wzory

DFT:

$$C_n = \sum_{i=0}^{N-1} f_i \cdot \exp\left(-j \cdot \frac{2\pi}{N} \cdot i \cdot n\right), \text{ gdzie } n = 0, 1, \ldots, N-1 \quad (1)$$

IDFT:

$$f_i = \frac{1}{N} \cdot \sum_{n=0}^{N-1} C_n \cdot \exp\left(j \cdot \frac{2\pi}{N} \cdot i \cdot n\right), \text{ gdzie } i = 0, 1, \ldots, N-1 \quad (2)$$

FFT:

$$\begin{aligned} C_n &= \sum_{i=0}^{\frac{N}{2}-1} f_{2i} \cdot \exp\left(-j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) + \exp\left(-j \cdot \frac{2\pi}{N} \cdot n\right) \cdot \sum_{i=0}^{\frac{N}{2}-1} f_{2i+1} \cdot \exp\left(-j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) \\ C_{n+\frac{N}{2}} &= \sum_{i=0}^{\frac{N}{2}-1} f_{2i} \cdot \exp\left(-j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) - \exp\left(-j \cdot \frac{2\pi}{N} \cdot n\right) \cdot \sum_{i=0}^{\frac{N}{2}-1} f_{2i+1} \cdot \exp\left(-j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) \end{aligned} \quad (3)$$

$$\text{gdzie } n = 0, 1, \ldots, \frac{N}{2} - 1$$

IFFT:

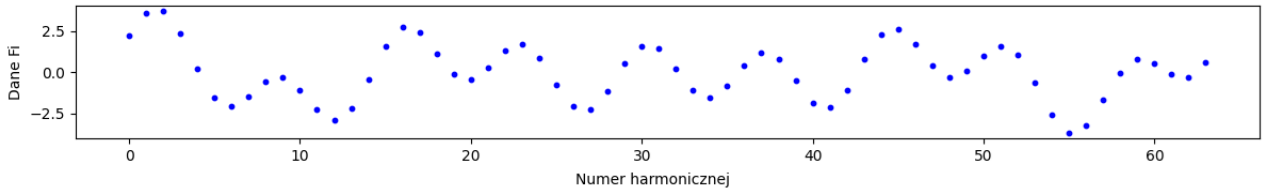
$$\begin{aligned} f_i &= \frac{1}{N} \cdot \sum_{n=0}^{\frac{N}{2}-1} C_{2n} \cdot \exp\left(j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) + \exp\left(j \cdot \frac{2\pi}{N} \cdot n\right) \cdot \frac{1}{N} \cdot \sum_{i=0}^{\frac{N}{2}-1} f_{2i+1} \cdot \exp\left(j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) \\ f_{i+\frac{N}{2}} &= \frac{1}{N} \cdot \sum_{n=0}^{\frac{N}{2}-1} C_{2n} \cdot \exp\left(j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) - \exp\left(j \cdot \frac{2\pi}{N} \cdot n\right) \cdot \frac{1}{N} \cdot \sum_{i=0}^{\frac{N}{2}-1} f_{2i+1} \cdot \exp\left(j \cdot \frac{2\pi}{\frac{N}{2}} \cdot i \cdot n\right) \end{aligned} \quad (4)$$

$$\text{gdzie } i = 0, 1, \ldots, \frac{N}{2} - 1$$

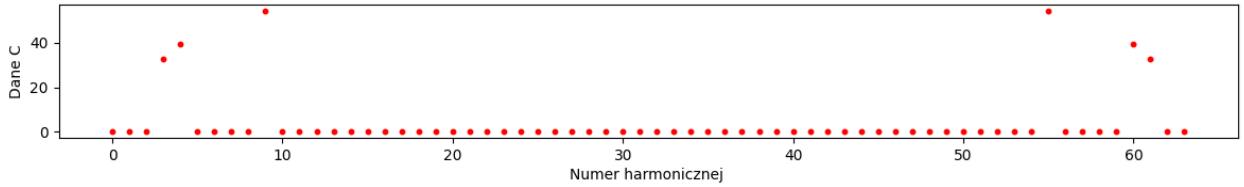
Wyniki

Dane 1 - z pliku dane_12.in

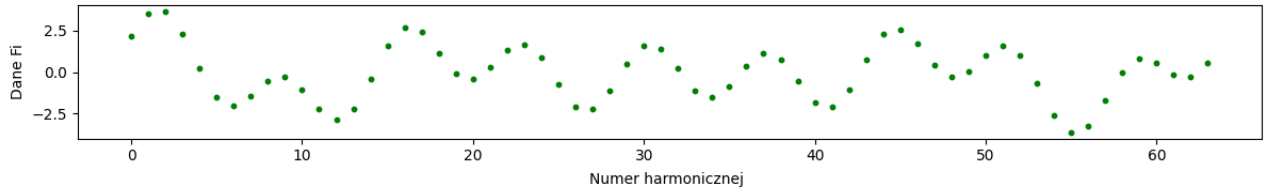
Wyniki dla danych jednowymiarowych z pliku `dane_12.in` przedstawione są na poniższych wykresach. Wykresy DFT oraz FFT dla są identyczne, dlatego przeststawiam je jednorazowo. Ze względu na braku szumu, w tych danych, wykres związany z nim został pominięty.



Rysunek 1: Sygnał wejściowy przed transformatą.



Rysunek 2: Sygnał wyjściowy po transformacie.

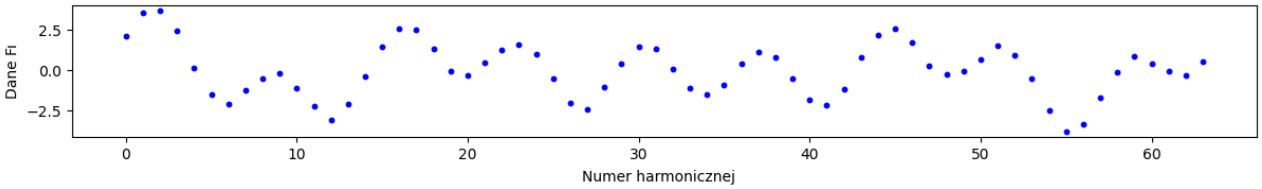


Rysunek 3: Sygnał wyjściowy po odwrotnej transformacie.

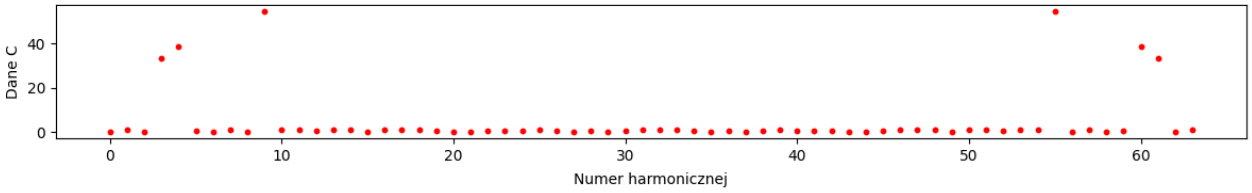
Podczas operacji DFT/IDFT wykonano 4096 operacji dominujących.
Podczas operacji FFT/IFFT wykonano 384 operacji dominujących.

Dane 2 - z pliku dane_12_a.in

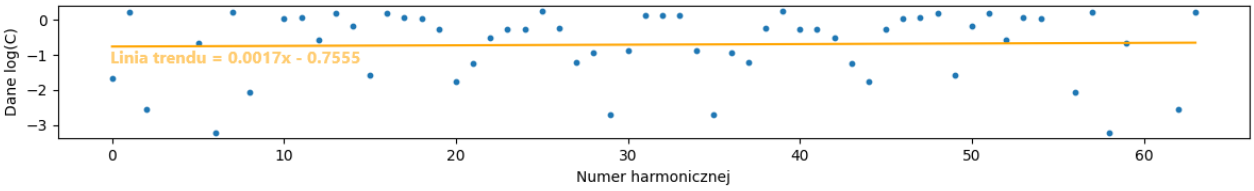
Wyniki dla danych jednowymiarowych z pliku `dane_12_a.in` przedstawione są na poniższych wykresach. Wykresy DFT oraz FFT dla są identyczne, dlatego przeststawiam je jednorazowo. Ze względu na pojawienie się szumu, w tych danych, możemy zaoberowować jak wygląda wykres dla niego.



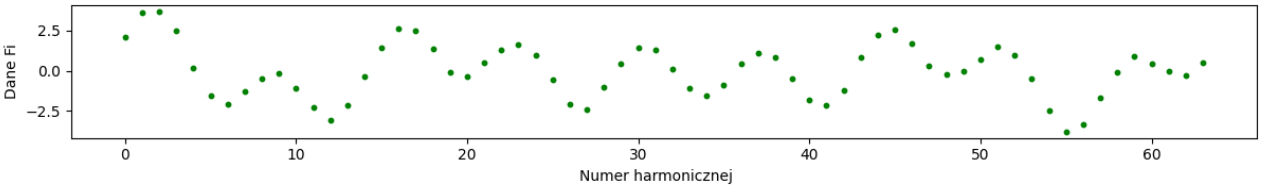
Rysunek 4: Sygnał wejściowy przed transformatą.



Rysunek 5: Sygnał wyjściowy po transformacie.



Rysunek 6: Sygnał wyjściowy po transformacie - bez skrajnych wartości, pokazany z linią trendu oraz zlinearyzowany

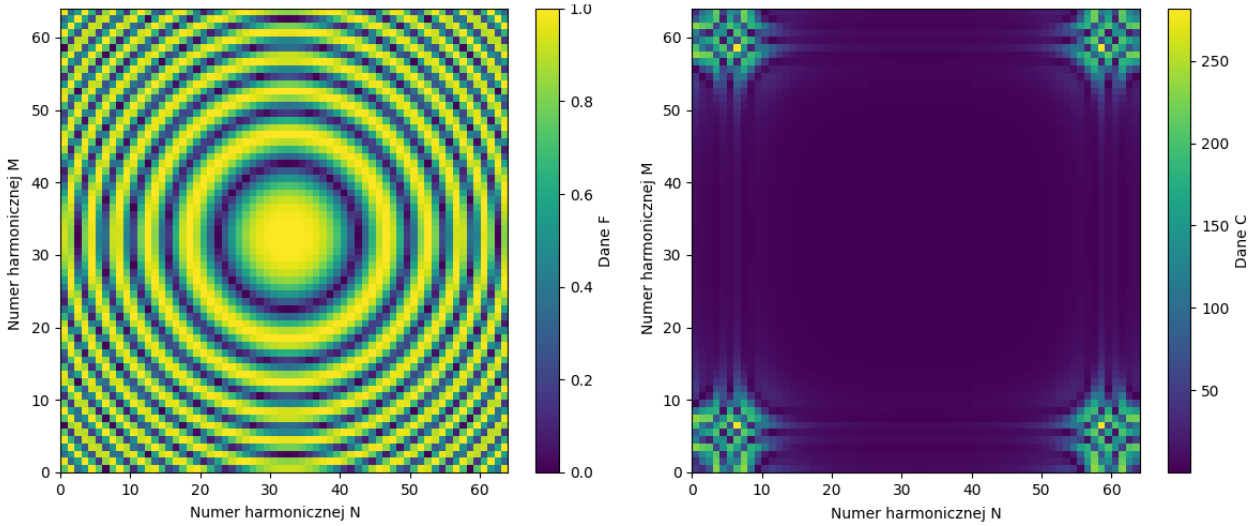


Rysunek 7: Sygnał wyjściowy po odwrotnej transformacie.

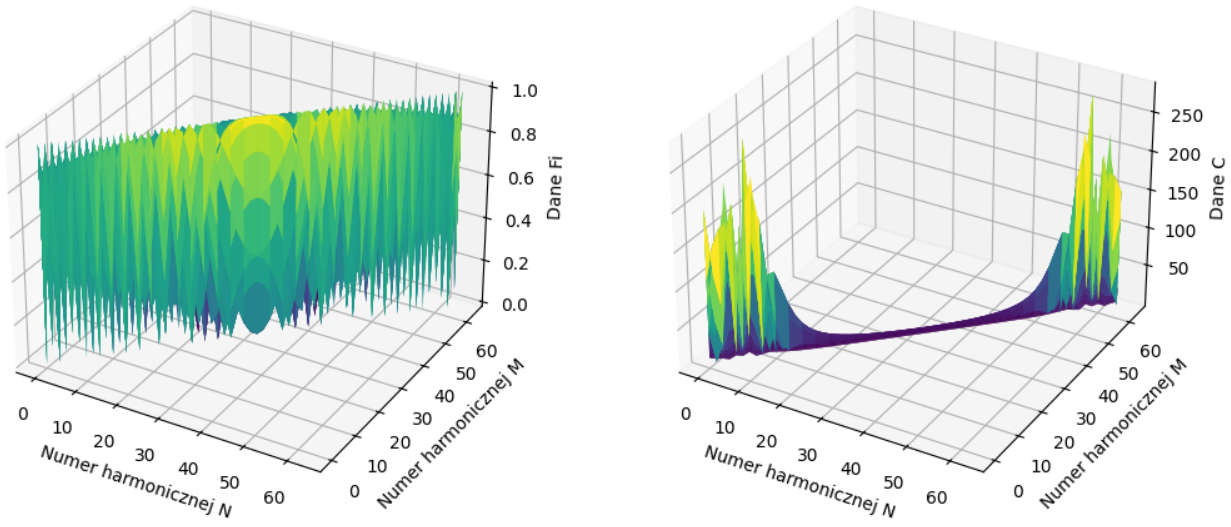
Podczas operacji DFT/IDFT wykonano 4096 operacji dominujących.
Podczas operacji FFT/IFFT wykonano 384 operacji dominujących.
Obliczony wzór linii trendu to $y = 0.0017x - 0.7555$

Dane 3 - z pliku dane2_12.in

Wyniki dla danych dwuwymiarowych z pliku `dane2_12.in` przedstawione są na poniższych wykresach. Zostały one zobrazowane za pomocą szybkiej tranformaty, ponieważ jest ona wykorzytana z wbudowanej biblioteki.



Rysunek 8: Sygnał wejściowy (lewy) oraz wyjściowy (prawy) przedstawione na dwuwymiarowej osi.



Rysunek 9: Sygnał wejściowy (lewy) oraz wyjściowy (prawy) przedstawione na trójwymiarowej osi.

Wykresy po lewej strone (Rysunek 8 i 9) nazywane są wzorcami. Wzorzec sygnału odnosi się do struktury i charakterystyki sygnału przed poddaniem go transformacji Fouriera. Natomiast wykresy po prawej stronie (Rysunek 8 i 9) nazywane są widmami. Widmo sygnału to reprezentacja sygnału w dziedzinie częstotliwości po zastosowaniu transformacji Fouriera. W widmie sygnału możemy zidentyfikować różne składowe częstotliwości, nazywane harmonikami. Każdy harmoniczny reprezentuje konkretną częstotliwość obecną w sygnale. Zmiany w wzorcu sygnału, takie jak dodawanie, usuwanie lub modyfikacje składowych, wpływają na widmo sygnału. Widmo reaguje na zmiany w strukturze sygnału, co jest istotne w kontekście analizy sygnałów dynamicznych.

Zależność między wzorcem a widmem jest kluczowa dla zrozumienia, jak transformacja Fouriera umożliwia analizę struktury sygnału w kontekście jego składowych częstotliwości. Widmo sygnału jest narzędziem, które pozwala na charakteryzację i interpretację sygnałów w dziedzinie częstotliwości, co ma zastosowanie w wielu dziedzinach, takich jak przetwarzanie sygnałów, telekomunikacja, czy analiza obrazów.

Podsumowanie

Program został stworzony w celu realizacji transformacji Fouriera dla danych jedno- i dwuwymiarowych. Wykorzystano implementacje zarówno DFT (Dyskretnej Transformaty Fouriera), jak i FFT (Szybkiej Transformaty Fouriera).

Wyniki

W przypadku danych jednowymiarowych, zarówno DFT, jak i FFT, zaprezentowały skuteczność w analizie sygnałów, a uzyskane wykresy potwierdzają zgodność wyników obu metod. W przypadku danych dwuwymiarowych, skorzystano z funkcji FFT z wbudowanej biblioteki.

Szum

Otrzymane wyniki przedstawione na dwu- i trójwymiarowych wykresach obrazują skuteczność transformacji w przestrzeni dwuwymiarowej. Możemy określić szum sygnału na podstawie obliczonego współczynnika linii trendu. W moim przypadku wyniósł on 0.0017, a więc nasza linia trendu rośnie bardzo powoli. Taki współczynnik wskazuje iż jest to szum biały.

Operacje Dominujące

W analizie wydajności programu, zidentyfikowano ilość operacji dominujących przeprowadzonych podczas transformacji. Dla jednowymiarowej DFT wykonano 4096 operacji, podczas gdy dla FFT liczba ta wyniosła 384. Są to liczby które zgadzają się z wcześniej podanymi złożonościami obliczeniowymi. FFT jest metodą znacznie szybszą.

Wnioski

Program jest efektywnym narzędziem do przeprowadzania transformacji Fouriera dla danych jedno- i dwuwymiarowych. Wykorzystanie szybkich algorytmów FFT pozwala na osiągnięcie dobrych wyników nawet dla dużych zbiorów danych.

Literatura

- 3Blue1Brown, "Czym jest transformata Fouriera? Wizualne wprowadzenie. ",
  https://www.youtube.com/watch?v=spUNpyF58BY
- Reducible, "The Fast Fourier Transform (FFT): Most Ingenious Algorithm Ever? ",
  https://youtu.be/h7apO7q16V0?si=aRDbTw-xUFpEhfLI
- AGH, "Transformacje",
  https://shorturl.at/hAZ27