Transformata Fouriera

Mateusz Kojro

1 Podstawa teoretyczna

1.1 Transformata Fouriera

Transformacje Fourierowskie to dziedzina transformacji pozwalających na przekształcanie funkcji z dziedziny czasu (np. przebiegi natężenia dźwięku w czasie) na funkcje w dziedzinie częstotliwości (np. natężenia dźwięku dla poszczególnych częstotliwości). Jednowymiarową transformatę możemy zapisać jako funkcje $f : \mathbb{R} \to \mathbb{C}$ za pomocą wzoru[3]:

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) \exp\left(-i\omega x\right)\, dx, \quad \forall \omega \in \mathbb{R} \tag{1}$$

gdzie i oznacza jednostkę urojoną a jeżeli x oznacza wartości należące do dziedziny badanej funkcji (W przykładzie badania natężenia dźwięku od czasu będzie miał jednostkę czasu), $f(x)$ jest wartością badanej funkcji dla danego x a ω oznacza częstotliwość (w przypadku gdy x jest czasem mierzonym w sekundach ω będzie miało jednostkę Hz) q

1.2 Odwrotna transformata Fouriera

W niektórych sytuacjach możliwe jest odwrócenie transformaty w celu uzyskania oryginalnego sygnału za pomocą tzw. odwrotnej transformaty Fouriera opisanej wzorem[3]:

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\omega) \exp\left(i\omega x\right)\, d\omega, \quad \forall x \in \mathbb{R} \tag{2}$$

gdzie $\hat{f}$ oznacza wynik transformaty f ouriera dla funkcji f

1.3 Dyskretne transformaty Fouriera

Dyskretyzacja transformaty Fouriera pozwala na zastosowanie tradycyjnej transformaty do analizy sygnalów mierzonych przez instrumenty (instrument pomiarowy generować będzie dyskretne próbki danych a nie ciągłą funkcje). Dyskretna transformatę możemy opisać za pomocą sumy przekształcającej ciąg próbek jakiegoś sygnału $[x_0, x_1, \ldots, x_{N-1}]$ gdzie $x_i \in \mathbb{R}$ w ciąg harmonicznych tego sygnału oznaczanych: $[X_0, X_1, \ldots, X_{N-1}]$ gdzie $X_n \in \mathbb{C}$ danej wzorem [5] :

$$X_k = \sum_{n=0}^{N-1} x_n \exp\left(\frac{-ikn2\pi}{N}\right),\ 0 \leq k \leq N-1 \tag{3}$$

gdzie k to numer badanej harmonicznej a N to liczba próbek w sygnale.

1.4 Transformaty wielowymiarowe

Transformata Fouriera może zostać uogólniona do n wymiarów korzystając z wzoru [3]:

$$\hat{f}(k) = \frac{1}{(2\pi)^{\frac{n}{2}}} \int f(r) \exp\left(-ikr\right) d^n r \tag{4}$$

w którym $k = [k_1, k_2, \ldots, k_n]$

a jej odwrotność do

$$f(r) = \frac{1}{(2\pi)^{\frac{n}{2}}} \int \hat{f}(k) \exp(ikr) d^n k \tag{5}$$

1.5 Szybka transformata Fouriera

Tzw szybkie transformacje Fouriera to zbiór algorytmów majacych na celu przspieszenie działania standardowej DFT [2]. Najczęściej stosowanym algorytmem z tej rodziny jest algorytm Cooley'aTukey'a umożliwiający poprawienie złożoności czasowej DFT z $O(n^2)$ do $O(n \log n)$[4] wykorzystując jej wewnętrzną symetrię i uzywając standardowej DFT tylko dla próbek o długości mniejszej od pewnego zadanego maksimum. Jego ograniczeniem jest wymaganie aby długość sygnału była potęgą 2[5].

2 Analiza sygnału za pomocą transformaty Fouriera

Jednowymiarowa sykretna transformata Fouriera moze zostac wykorzystana do analizy i modyfikacji funkcji przebiegu czasowego sygnalu. Dobry przykładem takiego zastosowania jest wykorzystanie DFT podczas analizy i obróbki sygnalu dźwiękowego. Umozliwia ona miedzy innymi na analize spektrum czestotliwosci w celu separacji sygnalow skladowych. Natomiast w polaczeniu z IDFT moze zostac wykorzytana w celu zmiany sygnalu wejsciowgo (np. w celu usuniecia szumu na danej czestotliwosci lub wzmocnienia sygnalu na innej)

2.1 Analiza sprawności implementacji DFT i IDFT

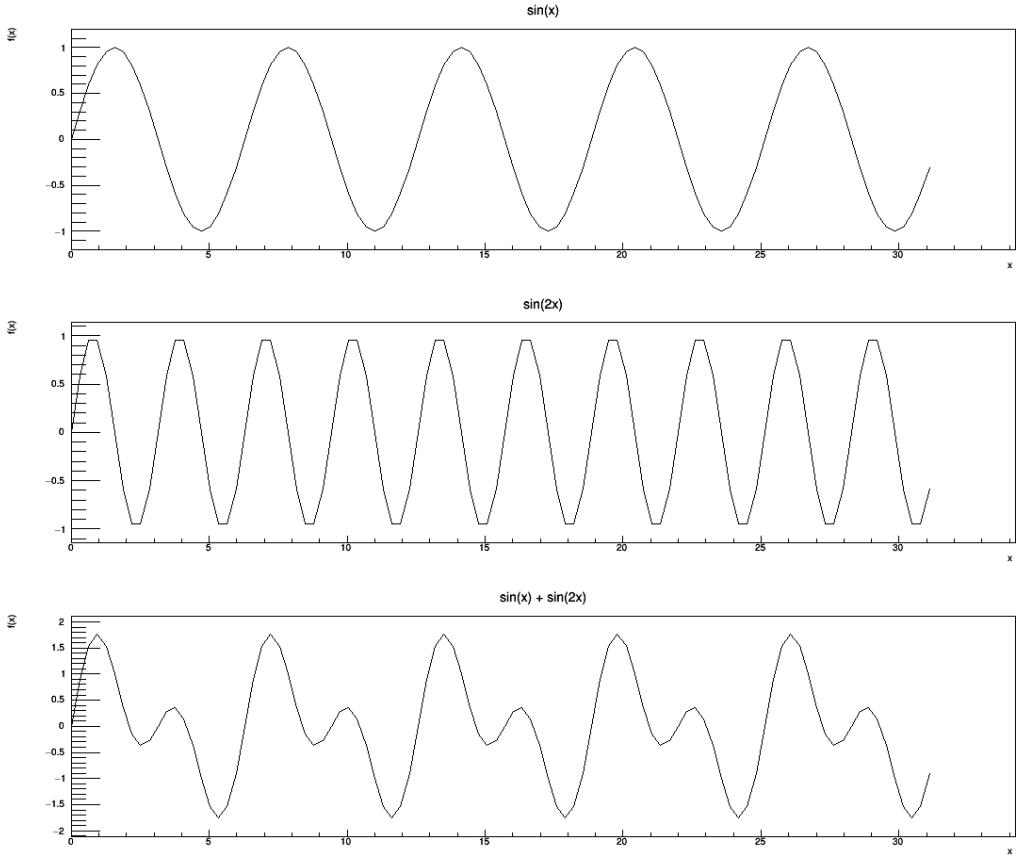
2.1.1 Wykorzystane narzędzia

W celu analizy sygnalu podanego w zadaniu zaimplementowane zostały DFT i IDFT. Wykorzystano język programowania C++ w standardzie 14 (ISO/IEC 14882) kompilowany za pomocą kompilatora MSVC w wersji 19.29.30136

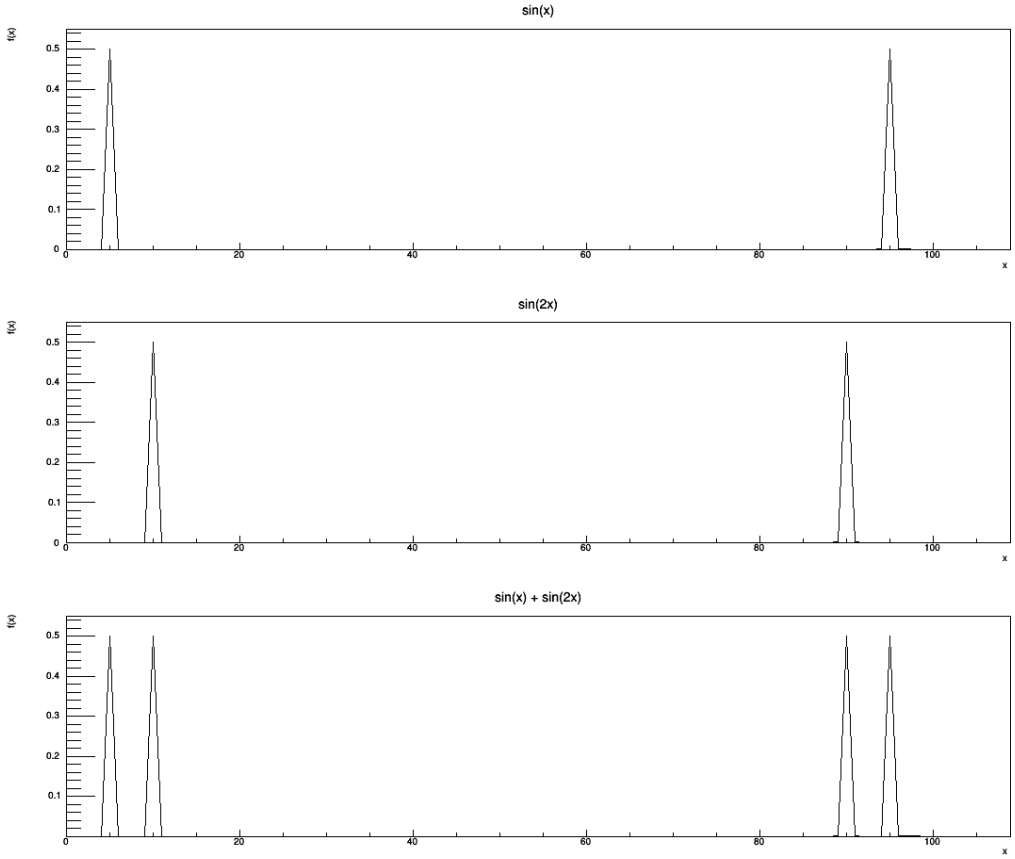
2.1.2 Badanie sygnału o znanych składowych

W celu zbadania poprawnosci implementacji DFT wygenerowano 3 testowe sygnaly na przedziale od 0 do 10π kazdy z nich zawiera 100 probek (ich przebiegi czasowe przedstawione zostaly na rysunku)

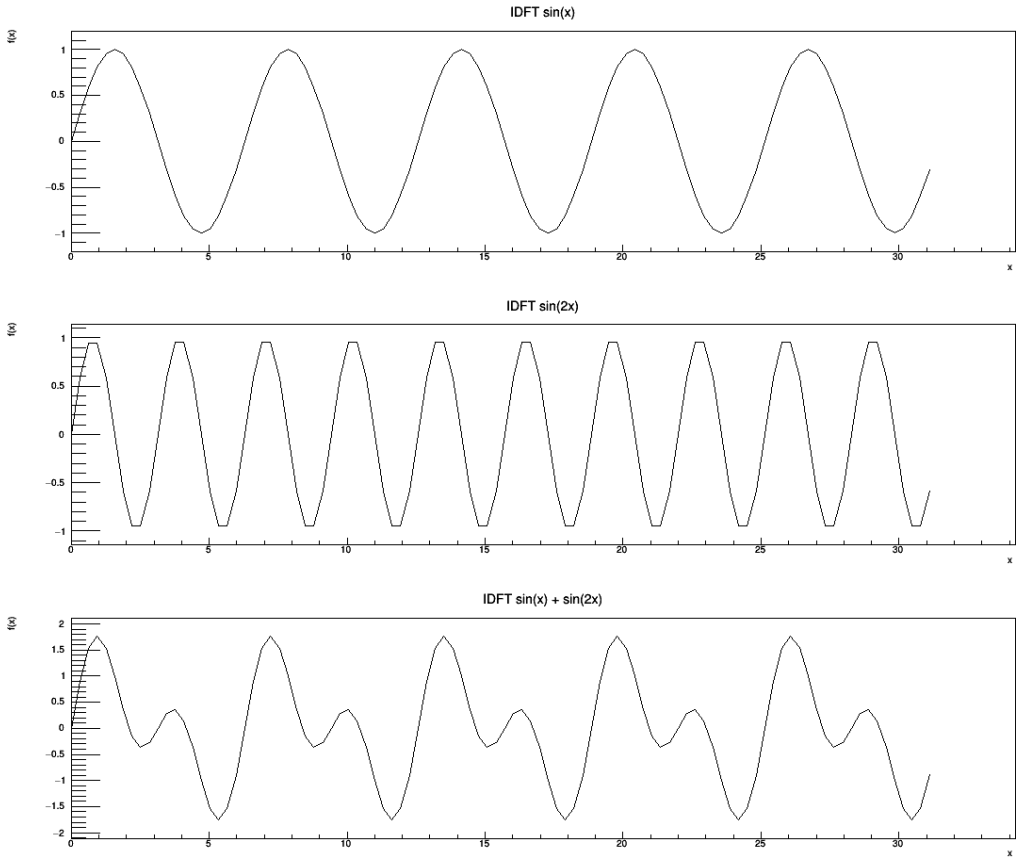
1. Funkcja określona wzorem $f(t) = \sin(t)$
2. Funkcja określona wzorem $f(t) = \sin(2t)$
3. Złożenie funkcji 1 i funkcji 2



powinnismy wiec otrzymac maxima transformaty sygnałów w $x = 5$ dla sygnalu 1 i $x = 10$ dla sygnalu 2 o wartości około $f(x) = 0.5$. Dla ich zlozenia transformata powinna natomiast wygladac jak sum tych wykresow. Wyniki przedstawione na rysunku ? Z duza dokldanoscia zgadzaja sie z oczekiwanymi wynikami. Co argumentuje poprawnosc implementacji dft.

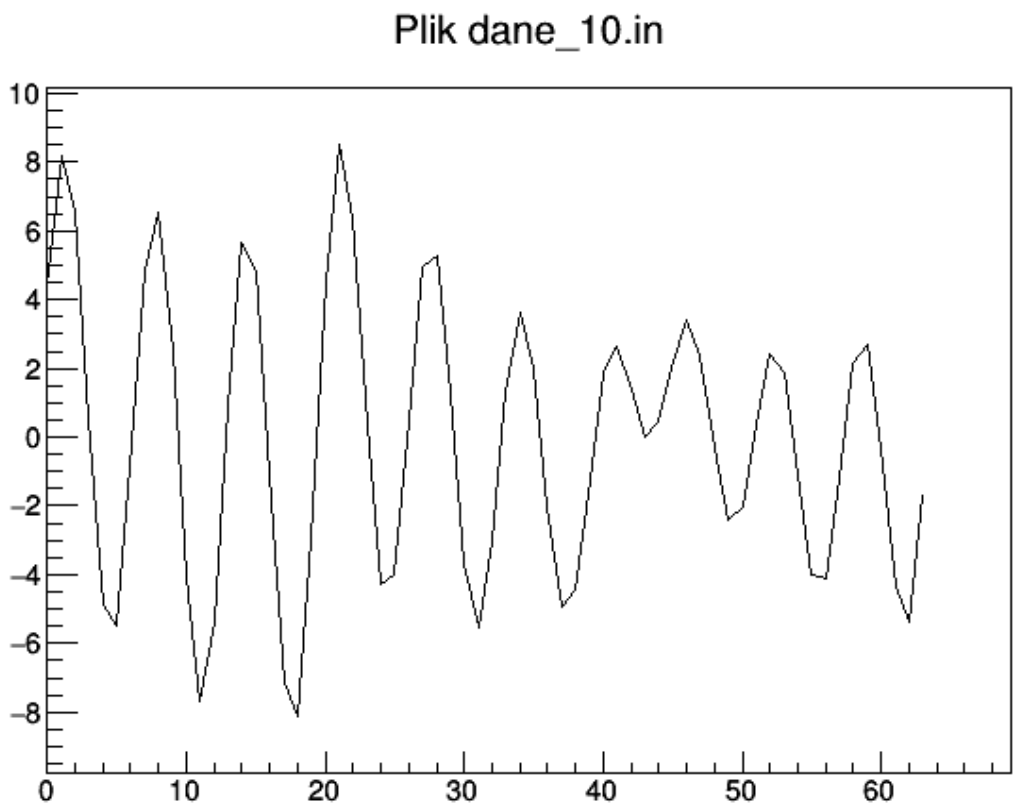


W celu sprawdzenia poprawnosci IDFT wyniki tej ww. transformacji zostana nastepnie poddane transfomacji odwrotnej a uzyskany sygnal powinien byc zblizony do sygnalu oryginalnego. Wyniki IDFT porownane z sygnalem orginlanym przedstawiono na



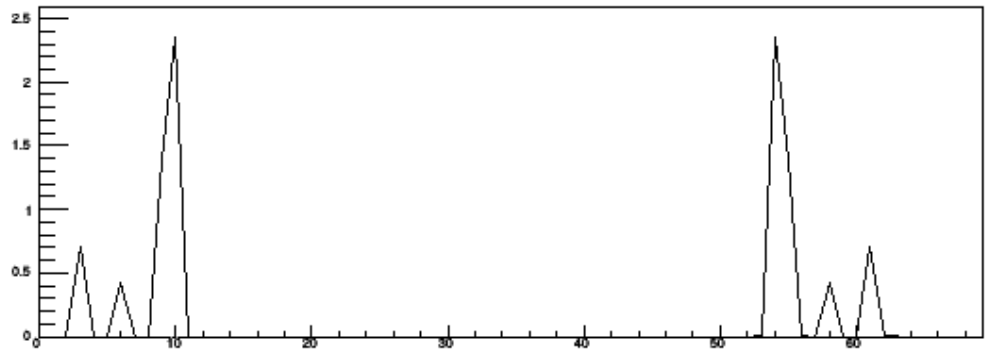
2.1.3 Porównanie wyników z implementacja biblioteki SciPy

Droga zastosowana metoda badania poprawnosci implementacji jest porownanie wynikow wygenerowanych dla danego zestawu danych z wynikiami otrzymanymi po aplikowaniu implementacji dft i idft znajdujacych sie w bibilotece SciPy [1]. Wykorzytsany do tego zostanie plik dane_10.in.

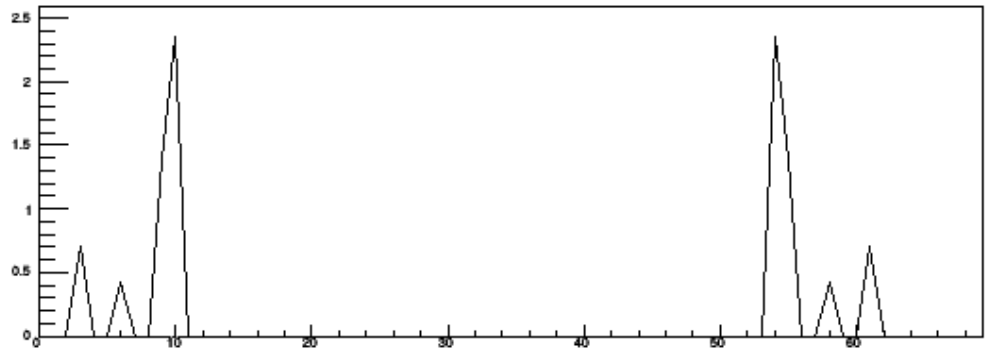


Rysunki ? jasno obrazuja bardzo wysoki stopien podobienstwa pomiedzy wynikami testowanej implementacji i implementacji w bibilotece python (wyniki fft w pythonie poddane zostaly normalizaci poprzez podzielnie wszystkich wartości przez ilosc probek poniewaz testowana implementacja stosuje taki zabieg)

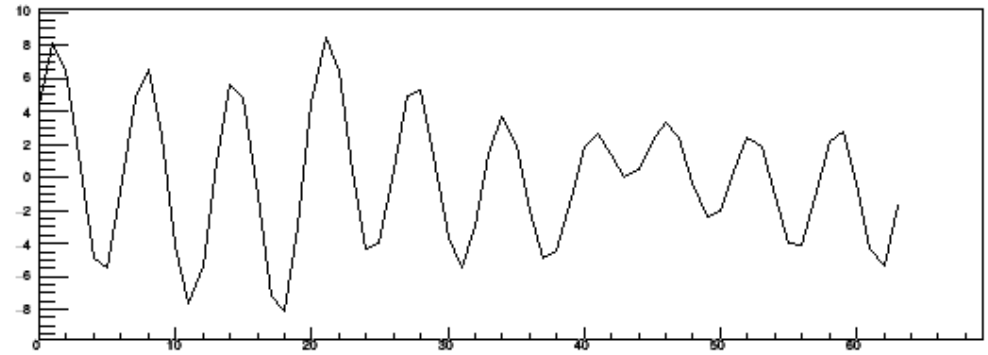
DFT Testowana implementacja



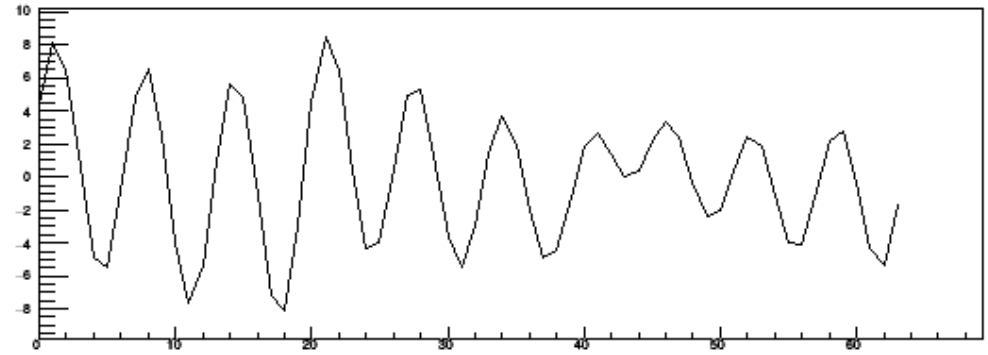
DFT SciPy



IDFT Testowana implementacja

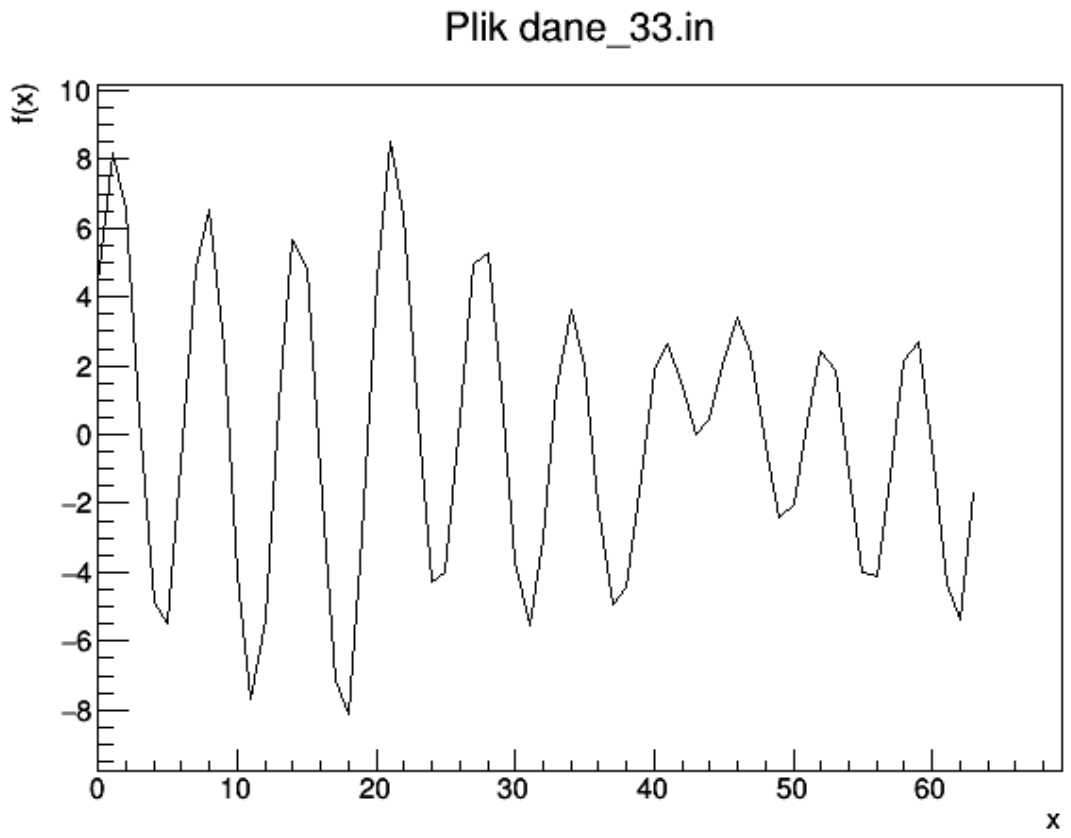


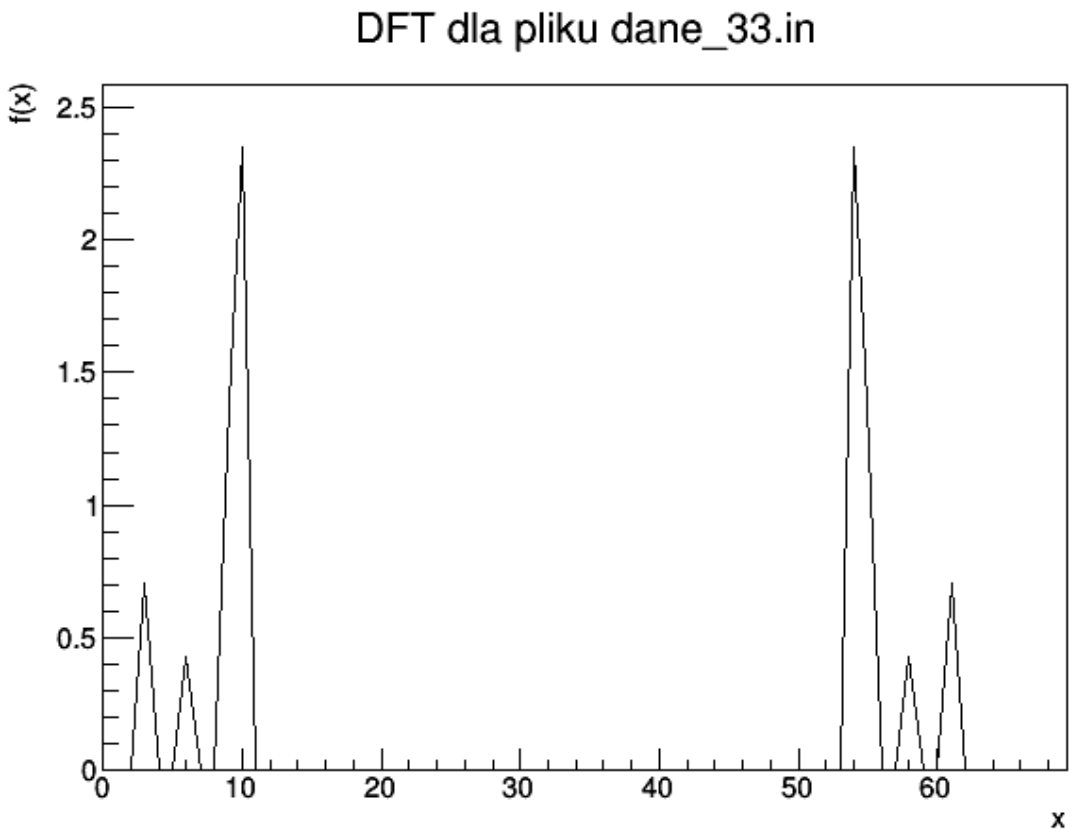
IDFT SciPy

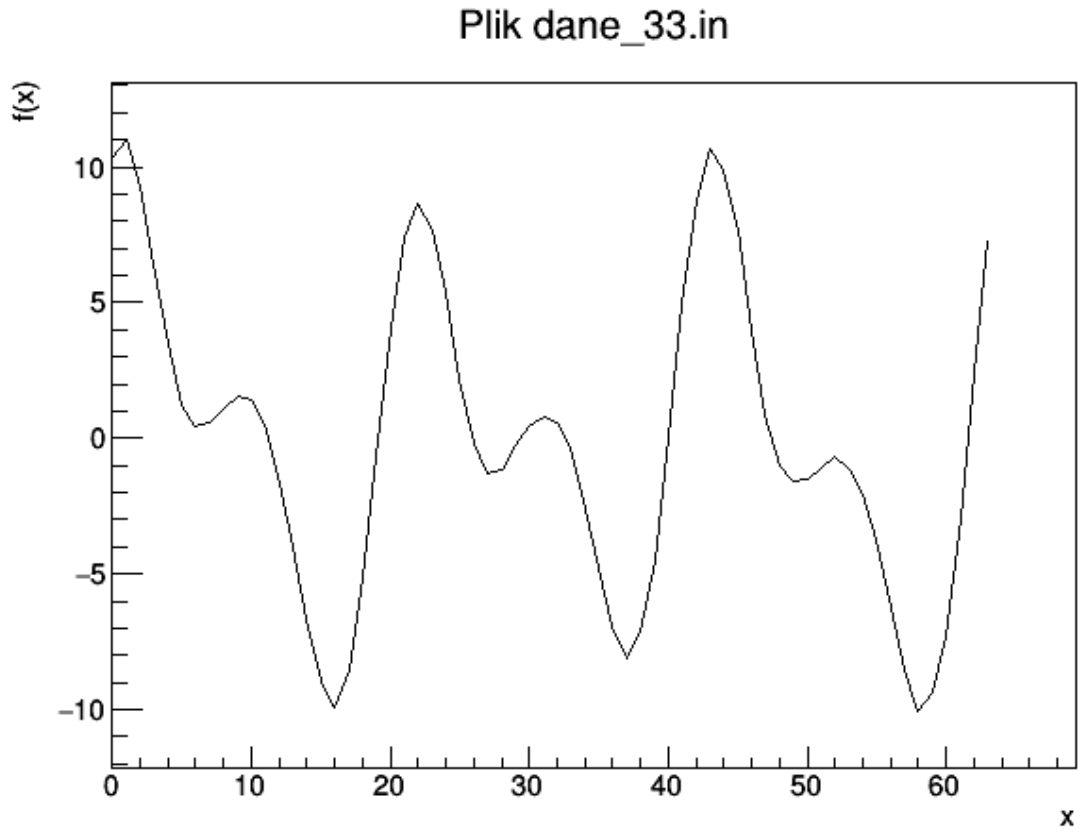


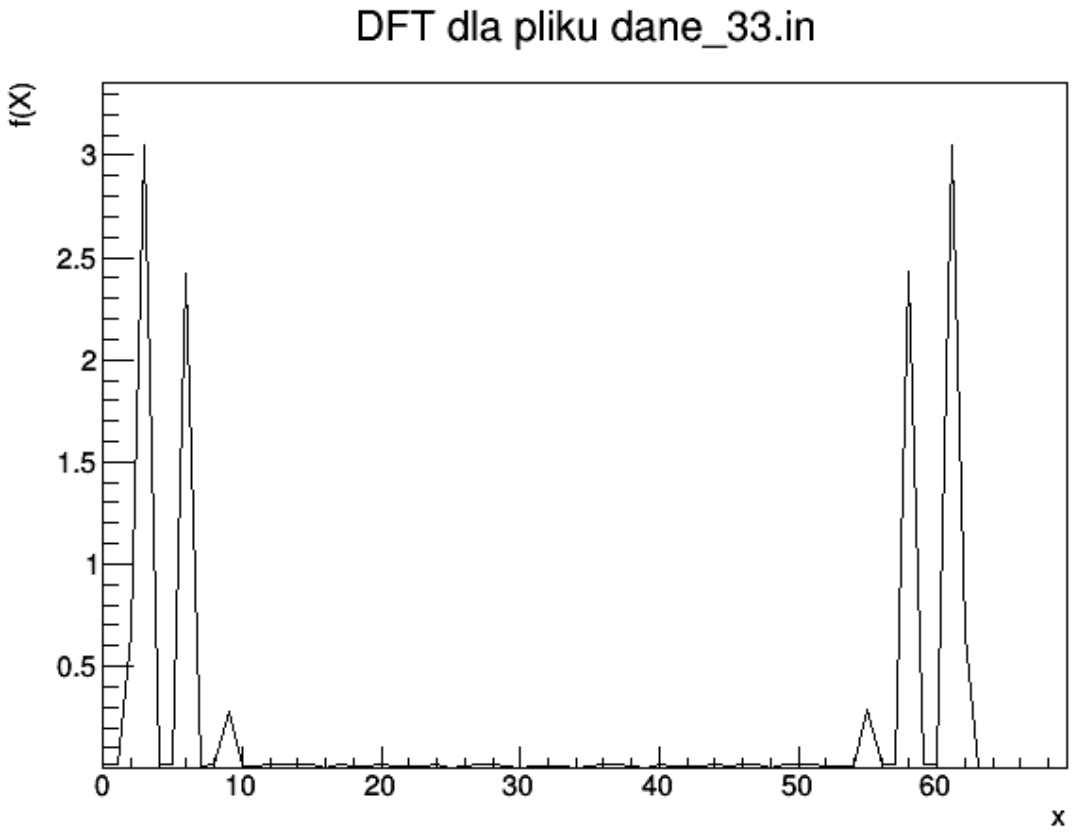
2.2 Analiza zadanego sygnału

Na rysunkach ? przedstawiono przebiegi czasowe zadanych sygnalow (10, 33 i dwu wymiarowy 6) ktore nastepnie poddane zostana analizie.




DFT dla pliku dane_33.in
f(x)
2.5
2
1.5
1
0.5
0
0
10
20
30
40
50
60
x


Plik dane_33.in
f(x)
10
5
0
−5
−10
0
10
20
30
40
50
60
x



2.2.1 Sygnal 1 wymiarowy

2.2.2 Sygnal 2 wymiarowy

2.3 Wnioski i podsumowanie

References

[1] The SciPy community. *Fourier Transforms*. URL: https://www.scipy.org/doc/scipy/reference/fft.html.

[2] *Cooley Tukey FFT algorithm*. URL: https://en.wikipedia.org/wiki/Cooley%E2%80%93Tukey_FFT_algorithm.

[3] S.J Bence K.F Riley M.P Hobson. *Mathematical Methods for Physics and Engineering*. Cambridge University Press, 2018.

[4] M. Marchewczyk. *Transformacje Fouriera*. URL: https://home.agh.edu.pl/~zobmat/2020/II_mich_mar/.

[5] A. Bayen Q. Kong T. Siauw. *Python Programming and Numerical Methods*. URL: https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html.