Sprawozdanie z laboratorium

Przetwarzanie Sygnałów i Obrazów

Ćwiczenie numer: 3

Temat: Dyskretne przekształcenie Fouriera

Wykonujący ćwiczenie:

- Zaborowska Magda
- Wójtowicz Patryk

Studia dzienne I stopnia

Kierunek: Informatyka

Semestr: III   Grupa zajęciowa: PS 12

Prowadzący ćwiczenie:

mgr inż. Dawid Najda

....

OCENA

Data wykonania ćwiczenia

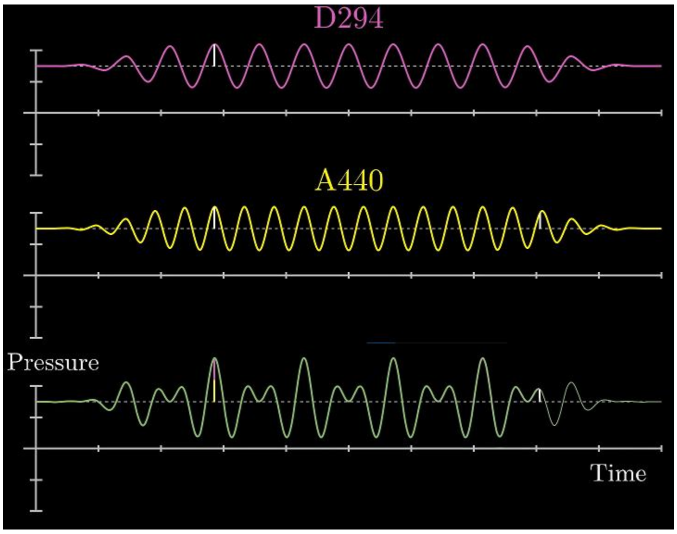
05.11.2021r.

....

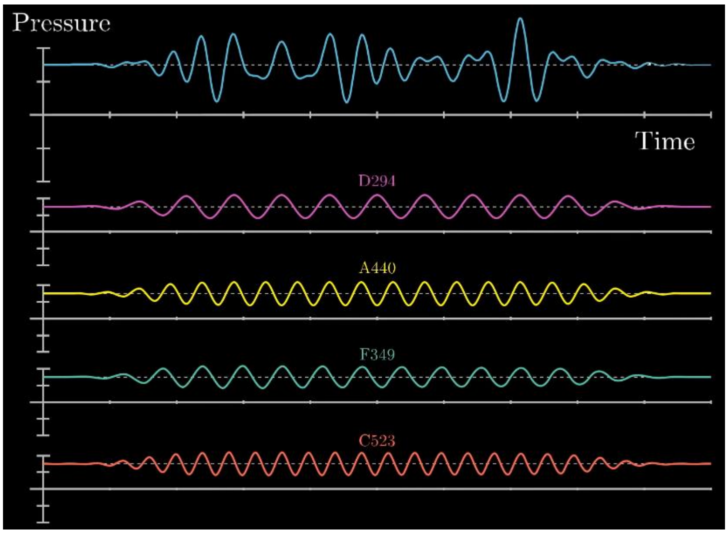
Data i podpis prowadzącego

Teoria

Mając dwa sygnały jesteśmy w stanie je zsumować tworząc nowy sygnał. Co jednak gdybyśmy chcieli się dowiedzieć z jakich sygnałów stworzony jest nasz sygnał pierwotny?



Rysunek 1 Sumowanie sygnałów okresowych.



Rysunek 2 Wydobycie z sygnału nieokresowego jego składowych sygnałów okresowych.

Służy do tego szereg Fouriera. Rozkłada on funkcję okresową na szereg funkcji trygonometrycznych. Zakładając, że funkcja f(x) jest okresowa, a jej dziedziną jest zbiór liczb rzeczywistych, okres funkcji wynosi T oraz jest całkowalna w przedziale $< -\frac{T}{2}; \frac{T}{2} >$, szereg ma postać:

$$S(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos\left(\frac{2n\pi}{T} x \right) + b_n \sin\left(\frac{2n\pi}{T} x \right) \right)$$

Równanie 1 Wzór na szereg Fouriera.

Natomiast współczynniki $\mathbf{a_n}$ i $\mathbf{b_n}$ można obliczyć w sposób:

$$a_n = \frac{2}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(x) \cos\left(\frac{2n\pi}{T} x \right) dx, \qquad b_n = \frac{2}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(x) \sin\left(\frac{2n\pi}{T} x \right) dx,$$

$$\mathrm{a}_0 = \frac{2}{\mathrm{T}} \int_{-\frac{\mathrm{T}}{2}}^{\frac{\mathrm{T}}{2}} \mathrm{f(x)}\, dx$$

Równanie 2 Wzory na współczynniki an, bn.

Sumując ze sobą kolejne wartości obliczone w nawiasie w drugiej części wzoru S(x), otrzymamy coraz dokładniejszy wykres zadanej funkcji. Natomiast zapisując je, otrzymamy rozkład na składowe naszej funkcji.

Transformacja Fouriera służy do przekształcenia funkcji ciągłej o dziedzinie czasu w funkcję ciągłą o dziedzinie częstotliwości. Jej wynikiem jest transformata Fouriera, która pokazuje z jakich częstotliwości składa się pierwotna funkcja. Umożliwia ona opisanie danej funkcji za pomocą ciągłej superpozycji częstotliwości. Dla funkcji f(x) transformację Fouriera określamy wzorem:

$$F(s) = \int_{-\infty}^{+\infty} f(x) \cdot e^{-2\pi i x s} dx$$

Równanie 3. i to jednostka urojona, s to częstotliwość, x to czas.

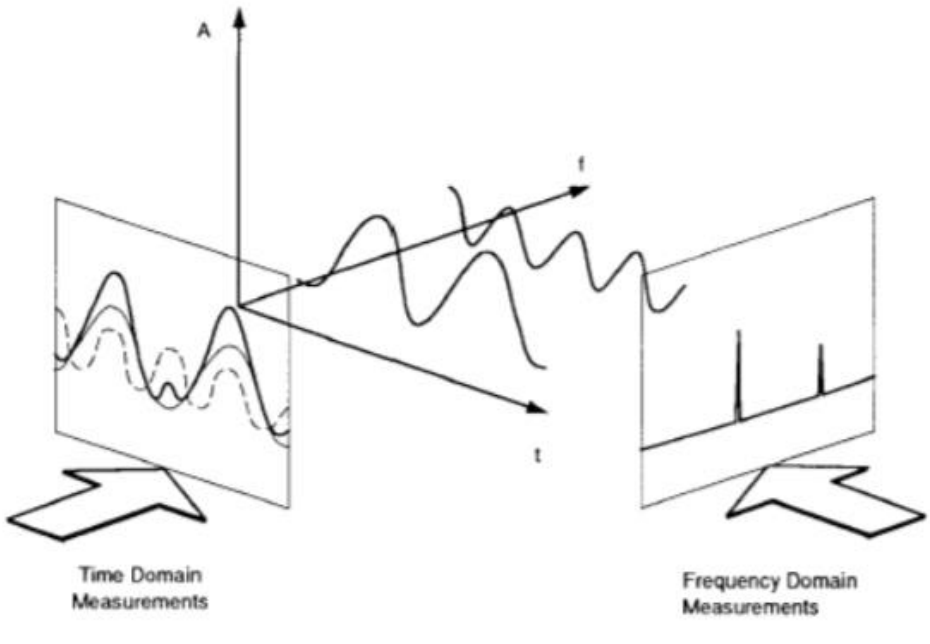
Dyskretna transformata Fouriera jest transformatą Fouriera, ale wyznaczoną dla sygnału dyskretnego, czyli próbkowanego. Jest ona określona następującym wzorem:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk} \qquad k = 0, \ldots, N-1$$

Równanie 4. N to ilość próbek, k to indeks próbki, x_n to wartość próbki, i to jednostka urojona.

Złożoność obliczania DFT z powyższego wzoru wynosi $O(N^2)$. Wynika stąd, że nie jest to wydajny sposób, dlatego powstała szybka transformata Fouriera FFT. Złożoność A=algorytmu FFT jest liniowo-logarytmiczna ($O(N \log_2 N)$). Jest to znacząca różnica. Oryginalna funkcja FTT liczy dla ilości próbek równej $N=2^k$ czyli dla potęg dwójki.

Za pomocą transformacji Fouriera otrzymamy widmo sygnału. Jest to reprezentacja sygnału w dziedzinie częstotliwości. Jest to zarówno wykres transformaty Fouriera jak i sama transformata. Widmo sygnału jest reprezentowane jako prążki o wysokości odpowiadającej amplitudzie poszczególnych składowych sygnału.



Rysunek 3 Po lewej stronie sygnały składowe, po prawej widmo.

Zadanie 1

Treść zadania:

• Wygeneruj dokładnie 1 okres fali sinusoidalnej (32 lub 64 próbki).
• Sporządź wykresy sinusoidy i jej transformaty Fouriera (funkcja fft) w jednym oknie (część rzeczywistą - real,
urojoną - imag, moduł - absolute i kąt - angle).
• Co możesz powiedzieć o symetrii widma zespolonego?

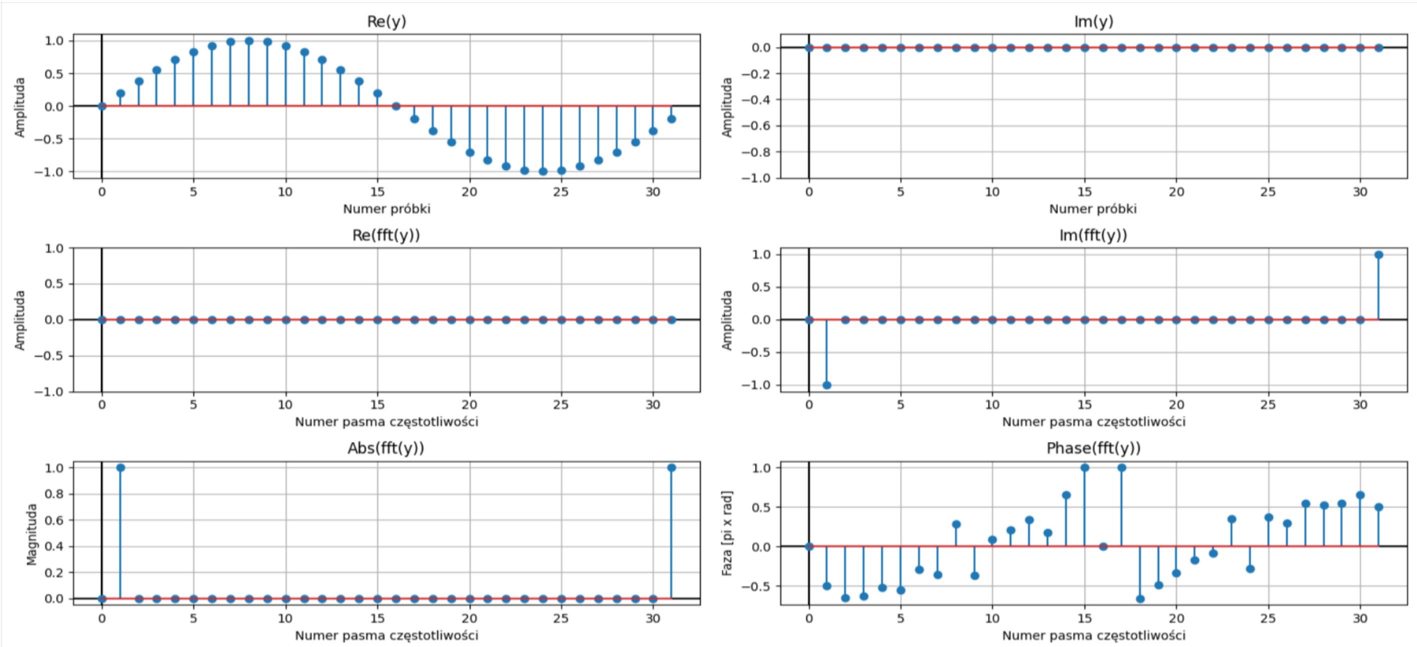
A) Realizacja w kodzie:

```
import numpy as np
import matplotlib.pyplot as plt

n = 32 #ilość próbek
x = np.arange(n) # tablica w n komurkami
y = np.sin(2*np.pi*x/n) # wzór funkcji bazowej
z = 2*np.fft.fft(y)/n #transformata Fouriera dla funkcji y

plt.figure(figsize=(8,6),tight_layout=1) # tworzenie figury na której wyświetlą sie wykresy

plt.subplot(3,2,1) #oznaczenie położenie wykresu
plt.title('Re(y)') #tytuł
plt.xlabel('Numer próbki') #nazwa osi OX
plt.ylabel('Amplituda') #nazwa osi OY
plt.axhline(y=0,color = "k") #pogrubienie osi OX
plt.axvline(x=0,color = "k") #pogrubienie osi OY
plt.stem(np.real(y),use_line_collection=10) #stworzenie wykresy typu stem
plt.grid(True,which='both') #włączenie siatki
```

```
# Tworzenie części zespolonej wykresu y
plt.subplot(3,2,2)
plt.title('Im(y)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,0.1)
plt.stem(np.imag(y),use_line_collection=1)
plt.grid(True,which='both')

# Tworzenie wykresu części rzeczywistej FFT(y)
plt.subplot(3,2,3)
plt.title('Re(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,1)
plt.stem(np.real(z),use_line_collection=1)
plt.grid(True,which='both')

# Tworzenie wykresu części zespolonej FFT(y)
plt.subplot(3,2,4)
plt.title('Im(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(z),use_line_collection=1)
plt.grid(True,which='both')
```

```
# Tworzenie wykresu modułu FFT(y)
plt.subplot(3,2,5)
plt.title('Abs(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(z),use_line_collection=1)
plt.grid(True,which='both')

# Tworzenie wykresu fazy FFT(y)
plt.subplot(3,2,6)
plt.title('Phase(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(z)/np.pi,use_line_collection=1)
plt.grid(True,which='both')

plt.show()
```

Wynik:



Dla transformaty Fouriera funkcji sinus(x) w części urojonej zauważalna jest symetria względem punktu (16, 0), natomiast dla modułu funkcji FFT(sin x) widoczna jest symetria względem osi X w punkcie x=16.

Zadanie 2

Treść zadania:

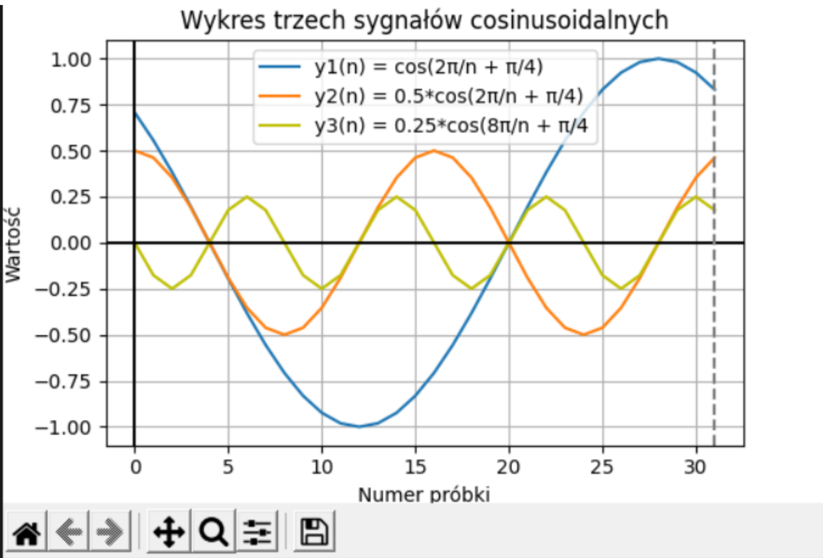
Wygeneruj następujące sygnały:

y1[n] = cos (2πn/N +π/4), y2[n] = 0.50 cos (4πn/N), y3[n] = 0.25 cos (8πn/N +π/2)

1. Wyznacz ich transformaty Fouriera.
2. Obliczenia powtórz dla sygnału y4 = y1 + y2 + y3.
3. Jaki jest związek pomiędzy amplitudą, fazą, liczbą okresów poszczególnych sygnałów a wartościami widma zespolonego?
4. Jak zachowuje się funkcja fft w stosunku do sumy sygnałów?

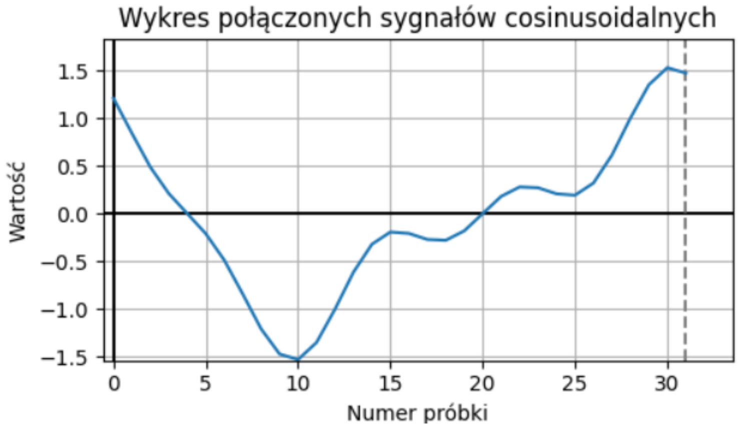
Realizacja założeń:

```python
import numpy as np
import matplotlib.pyplot as plt

n = 32
x = np.arange(n)
y1 = np.cos(2*np.pi*x/n+(np.pi/4))
y2 = 0.5*np.cos(4*np.pi*x/n)
y3 = 0.25*np.cos(8*np.pi*x/n+(np.pi/2))

plt.figure(figsize=(8,6))

plt.plot(x,y1)
plt.plot(x,y2)
plt.plot(x,y3,'y')
plt.grid(True,which='both')
plt.title('Wykres trzech sygnałów cosinusoidalnych')
plt.xlabel('Numer próbki')
plt.ylabel('Wartość')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.axvline(x=n-1,color = "grey",linestyle = "--")
plt.legend(['y1(n) = cos(2π/n + π/4)','y2(n) = 0.5*cos(2π/n + π/4)','y3(n) = 0.25*cos(8π/n + π/4'])

plt.show()
```



Stworzenie zsumowanego sygnału:

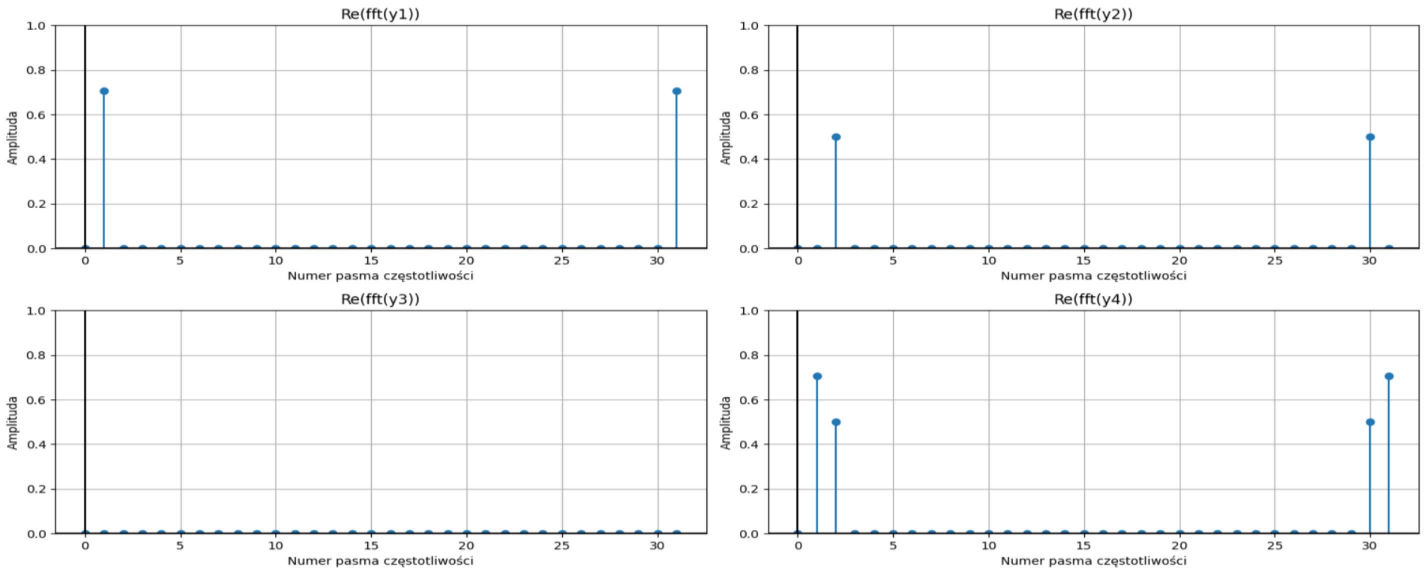
```python
# Podpunkt 2 , obliczenia powtórz dla sygnału y4 = y1 + y2 +y3
y4 = y1 + y2 + y3

plt.figure(figsize=(8,6))
plt.grid(True,which='both')
plt.title('Wykres połączonych sygnałów cosinusoidalnych')
plt.xlabel('Numer próbki')
plt.ylabel('Wartość')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.axvline(x=n-1,color = "grey",linestyle = "--")
plt.plot(x,y4)
plt.show()
```



Obliczenie transformat Fouriera oraz tworzenie wykresów części rzeczywistej widma sygnałów:

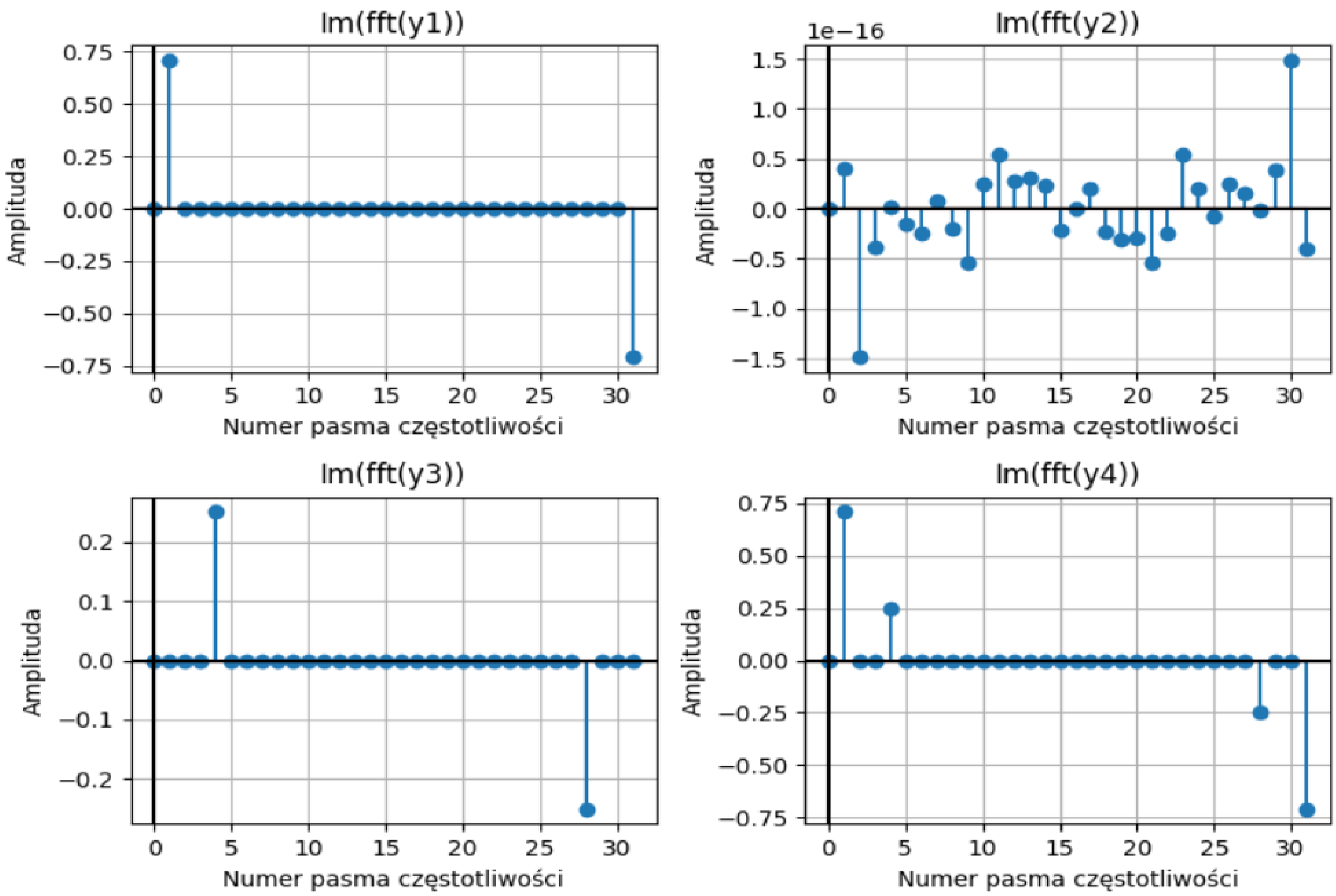
```python
# Podpunkt 1, obliczenia tranformatory
# Fouriera dla poszczególnych funkcji
f1 = 2*np.fft.fft(y1)/n
f2 = 2*np.fft.fft(y2)/n
f3 = 2*np.fft.fft(y3)/n
f4 = 2*np.fft.fft(y4)/n

#Część rzeczywista
plt.figure(figsize=(8,6),tight_layout=1)

plt.subplot(2,2,1)
plt.grid(True,which='both')
plt.stem(np.real(f1),use_line_collection=1)
plt.ylim(0,1)
plt.title('Re(fft(y1))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,2)
plt.grid(True,which='both')
plt.stem(np.real(f2),use_line_collection=1)
plt.ylim(0,1)
plt.title('Re(fft(y2))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
```

```python
plt.subplot(2,2,3)
plt.grid(True,which='both')
plt.stem(np.real(f3),use_line_collection=1)
plt.ylim(0,1)
plt.title('Re(fft(y3))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,4)
plt.grid(True,which='both')
plt.stem(np.real(f4),use_line_collection=1)
plt.ylim(0,1)
plt.title('Re(fft(y4))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.show()
```



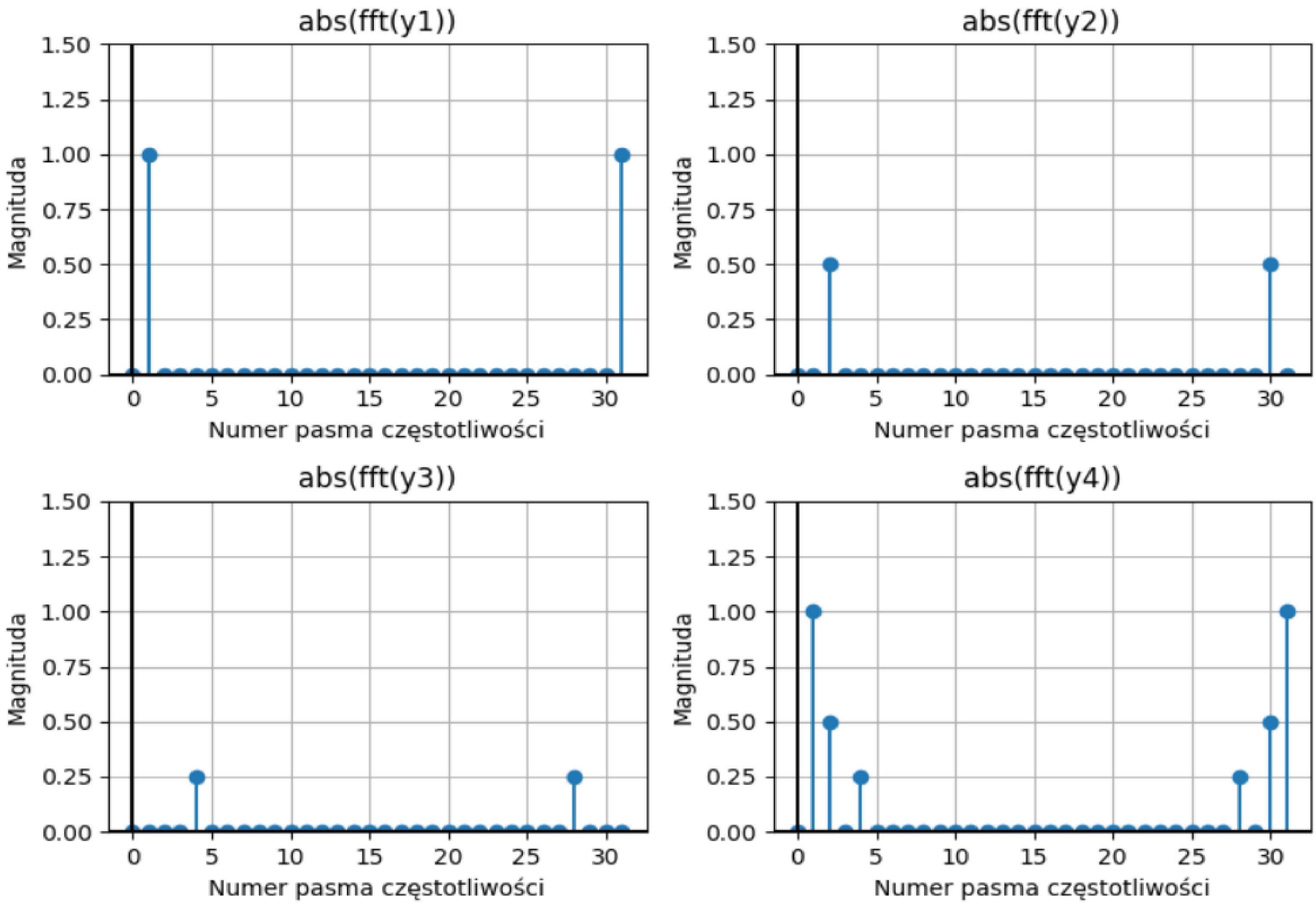
Tworzenie wykresów części urojonej widma sygnałów:

```
#Część urojona

plt.figure(figsize=(8,6),tight_layout=1)

plt.subplot(2,2,1)
plt.grid(True,which='both')
plt.stem(np.imag(f1),use_line_collection=1)
plt.title('Im(fft(y1))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,2)
plt.grid(True,which='both')
plt.stem(np.imag(f2),use_line_collection=1)
plt.title('Im(fft(y2))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")


plt.subplot(2,2,3)
plt.grid(True,which='both')
plt.stem(np.imag(f3),use_line_collection=1)
plt.title('Im(fft(y3))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,4)
plt.grid(True,which='both')
plt.stem(np.imag(f4),use_line_collection=1)
plt.title('Im(fft(y4))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.show()
```



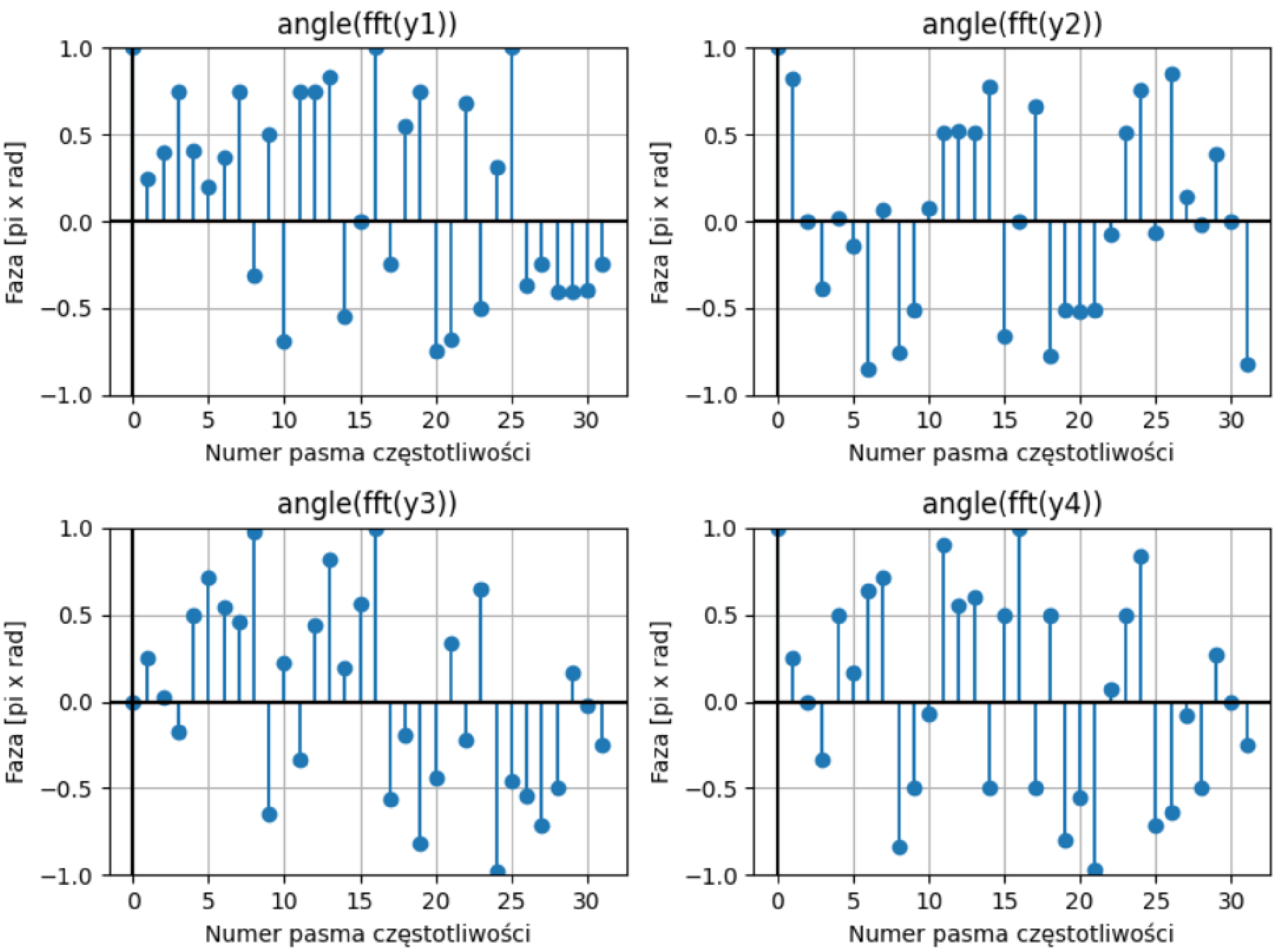
Tworzenie wykresów modułu widma sygnałów:

```python
# Moduł

plt.figure(figsize=(8,6),tight_layout=1)

plt.subplot(2,2,1)
plt.grid(True,which='both')
plt.stem(np.absolute(f1),use_line_collection=1)
plt.ylim(0,1.5)
plt.title('abs(fft(y1))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,2)
plt.grid(True,which='both')
plt.stem(np.absolute(f2),use_line_collection=1)
plt.ylim(0,1.5)
plt.title('abs(fft(y2))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")



plt.subplot(2,2,3)
plt.grid(True,which='both')
plt.stem(np.absolute(f3),use_line_collection=1)
plt.ylim(0,1.5)
plt.title('abs(fft(y3))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,4)
plt.grid(True,which='both')
plt.stem(np.absolute(f4),use_line_collection=1)
plt.ylim(0,1.5)
plt.title('abs(fft(y4))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.show()
```



Tworzenie wykresów fazy widma sygnałów:

```python
# Kąt

plt.figure(figsize=(8,6),tight_layout=1)

plt.subplot(2,2,1)
plt.grid(True,which='both')
plt.stem(np.angle(f1)/np.pi,use_line_collection=1)
plt.ylim(-1,1)
plt.title('angle(fft(y1))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,2)
plt.grid(True,which='both')
plt.stem(np.angle(f2)/np.pi,use_line_collection=1)
plt.ylim(-1,1)
plt.title('angle(fft(y2))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")



plt.subplot(2,2,3)
plt.grid(True,which='both')
plt.stem(np.angle(f3)/np.pi,use_line_collection=1)
plt.ylim(-1,1)
plt.title('angle(fft(y3))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.subplot(2,2,4)
plt.grid(True,which='both')
plt.stem(np.angle(f4)/np.pi,use_line_collection=1)
plt.ylim(-1,1)
plt.title('angle(fft(y4))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")

plt.show()
```



Zadanie 3

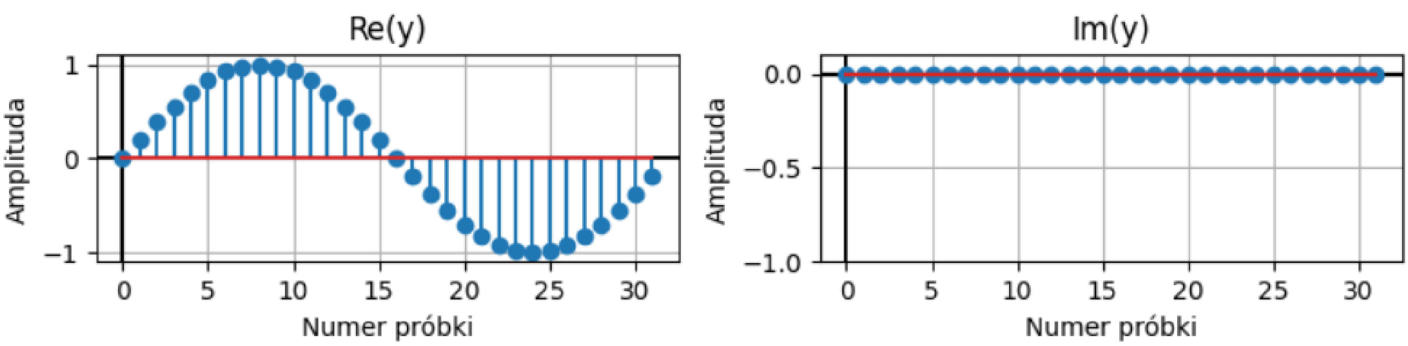
Treść zadania:

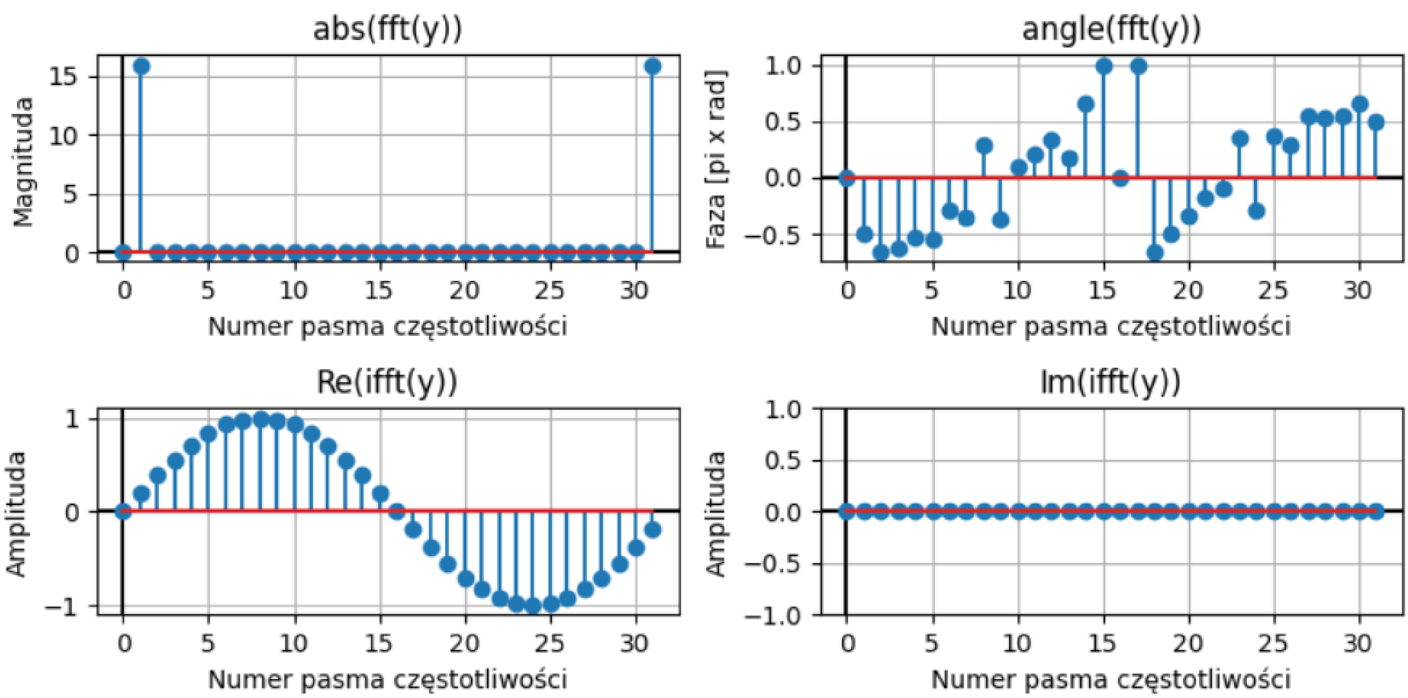
- Wyznacz odwrotną transformatę Fouriera (funkcja ifft) dla widm zespolonych FFT otrzymanych w zadaniach 3.1 i 3.2.
- Jak zinterpretujesz otrzymane wyniki?

Realizacja odwrotnej transformaty dla zadanie 3.1:

```
import numpy as np
import matplotlib.pyplot as plt

#zadanie 1
n = 32
x = np.arange(n)
y = np.sin(2*np.pi*x/n)
yfft = np.fft.fft(y) # fft - szybka transformata
yifft = np.fft.ifft(yfft) # ifft - odwrotna transformata
plt.figure(figsize=(8,6),tight_layout=1)
plt.subplot(3,2,1)
plt.title('Re(y)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(y),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,2)
plt.title('Im(y)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,0.1)
plt.stem(np.imag(y),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,3)
plt.title('abs(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(yfft),use_line_collection=1)
plt.grid(True,which='both')




plt.subplot(3,2,4)
plt.title('angle(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(yfft)/np.pi,use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,5)
plt.title('Re(ifft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1.1,1.1)
plt.stem(np.real(yifft),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,6)
plt.title('Im(ifft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,1)
plt.stem(np.imag(yifft),use_line_collection=1)
plt.grid(True,which='both')

plt.show()
```

Wynik:

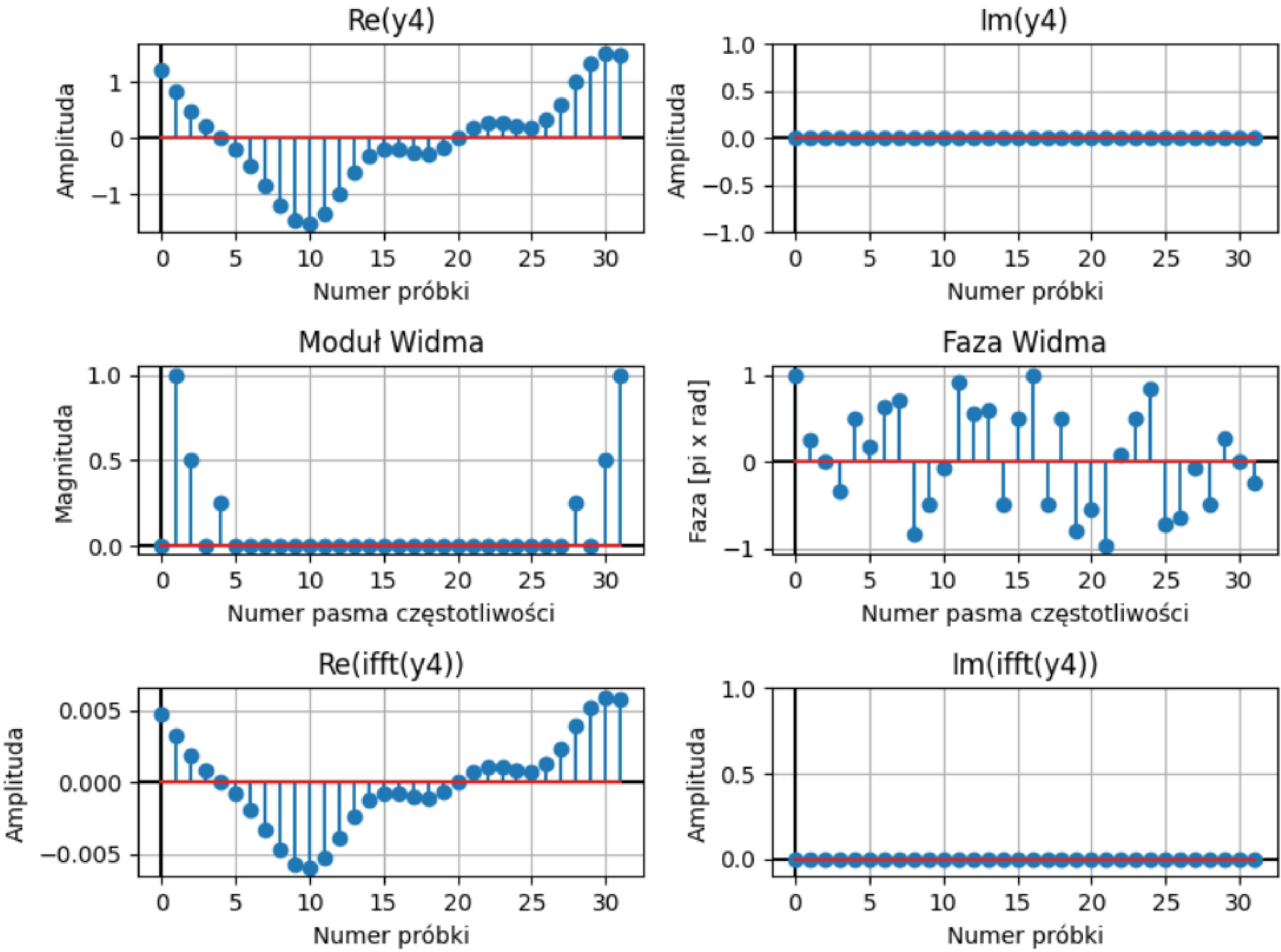




Realizacja odwrotnej transformaty dla zadanie 3.2:

```python
#zadanie 2
y1 = np.cos(2*np.pi*x/n+(np.pi/4))
y2 = 0.5*np.cos(4*np.pi*x/n)
y3 = 0.25*np.cos(8*np.pi*x/n+(np.pi/2))
y4=y1+y2+y3

f4 = 2*np.fft.fft(y4)/n
if4 = 2*np.fft.ifft(f4)/n

plt.figure(figsize=(8,6),tight_layout=1)
plt.subplot(3,2,1)
plt.title('Re(y4)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(y4),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,2)
plt.title('Im(y4)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,1)
plt.stem(np.imag(y4),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,3)
plt.title('Moduł Widma')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(f4),use_line_collection=1)
plt.grid(True,which='both')




plt.subplot(3,2,4)
plt.title('Faza Widma')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(f4)/np.pi,use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,5)
plt.title('Re(ifft(y4))')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(if4),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(3,2,6)
plt.title('Im(ifft(y4))')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-0.1,1)
plt.stem(np.imag(if4),use_line_collection=1)
plt.grid(True,which='both')

plt.show()
```

Wynik:



Zadanie 4

Treść zadania:

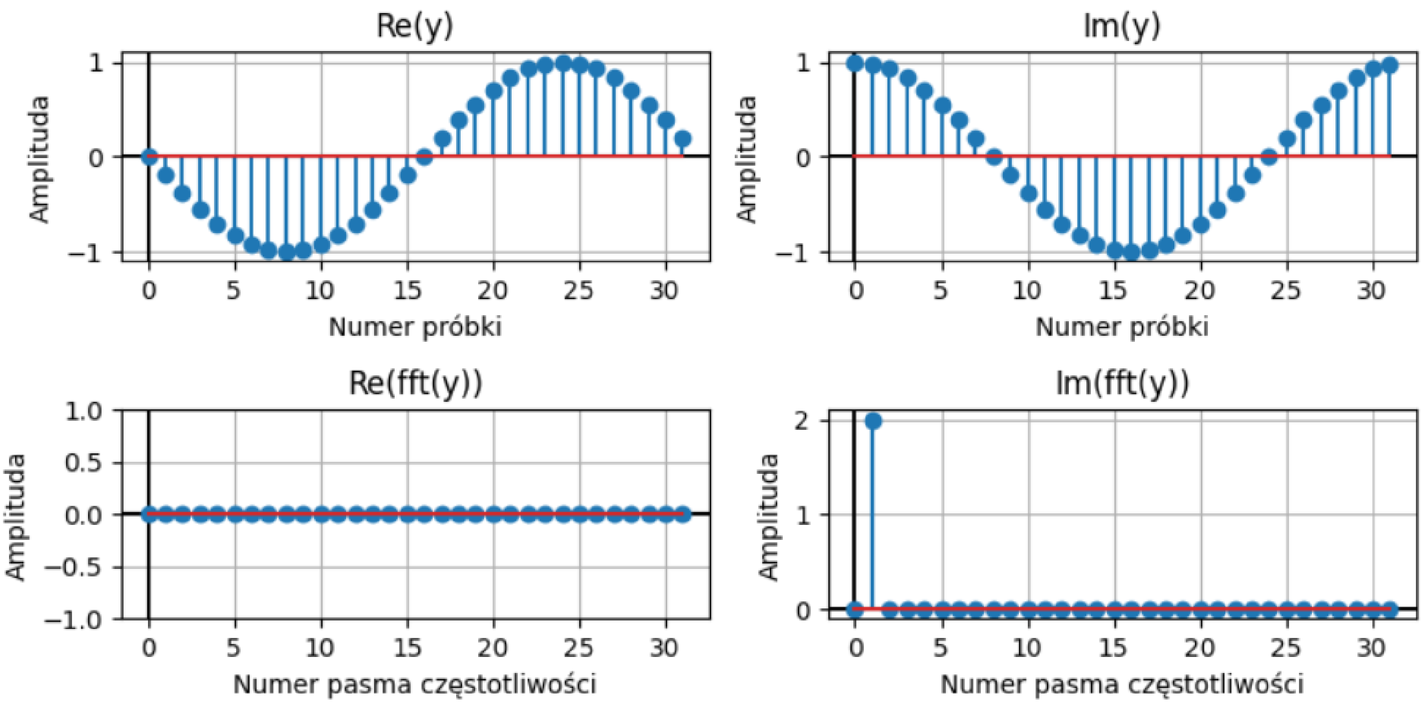
• Powtórz zadanie 3.1 dla sinusoidy zespolonej y[n] = exp (j(ωn + φ)) , gdzie ω – pulsacja unormowana oraz ϕ – przesunięcie fazowe.

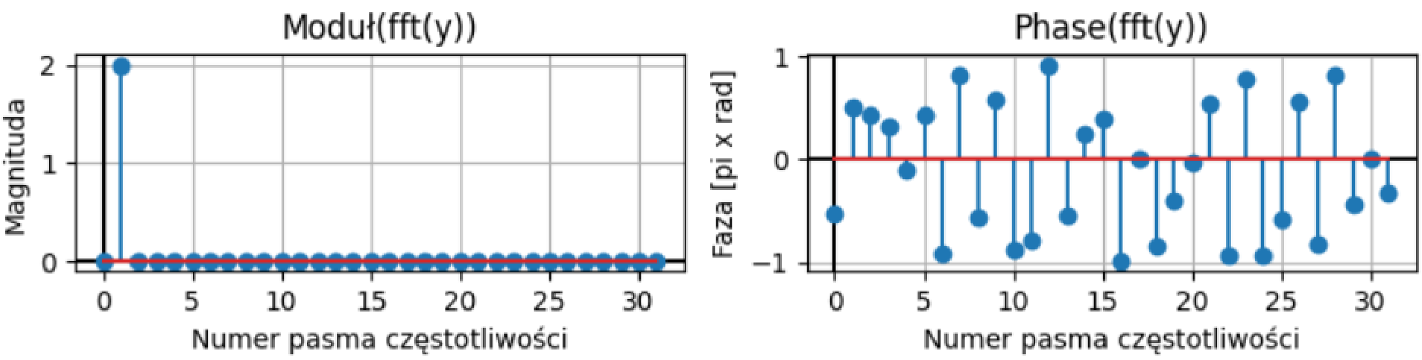
• Jakie są różnice (w symetrii) w stosunku do widma sygnału rzeczywistego?

Realizacja w kodzie:

```python
import numpy as np
import matplotlib.pyplot as plt

n = 32
x = np.arange(n)
y = np.exp(1j*(2*np.pi*1/n*x+np.pi/2)) #wzór funkcji
z = 2*np.fft.fft(y)/n #FFT dla funkcji

plt.figure(figsize=(8,6),tight_layout=1)
#cz. rzeczywista funkcji y
plt.subplot(3,2,1)
plt.title('Re(y)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(y),use_line_collection=1)
plt.grid(True,which='both')
#cz. urojona funkcji y
plt.subplot(3,2,2)
plt.title('Im(y)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(y),use_line_collection=1)
plt.grid(True,which='both')
#cz. rzeczywista funkcji FFT
plt.subplot(3,2,3)
plt.title('Re(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,1)
plt.stem(np.real(z),use_line_collection=1)
plt.grid(True,which='both')

#cz. urojona funkcji FFT(y)
plt.subplot(3,2,4)
plt.title('Im(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(z),use_line_collection=1)
plt.grid(True,which='both')
#moduł funkcji FFT(y)
plt.subplot(3,2,5)
plt.title('Moduł(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(z),use_line_collection=1)
plt.grid(True,which='both')
#kąt funkcji FFT(y)
plt.subplot(3,2,6)
plt.title('Phase(fft(y))')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(z)/np.pi,use_line_collection=1)
plt.grid(True,which='both')

plt.show()
```

Wynik:





Zadanie 5

Treść zadania:

Napisz własną funkcję realizującą dyskretną transformatę Fouriera – DFT oraz jej transformatę odwrotną - IDFT. Wykorzystaj postać macierzową przekształcenia oraz funkcję generującą macierz typu DFT (ang. DFT matrix), np. from scipy.linalg import dft. Porównaj wyniki z zadania 3.1 i 3.2 z wynikami własnej funkcji transformaty Fouriera

Stworzenie własnych funkcji DFT IDFT:

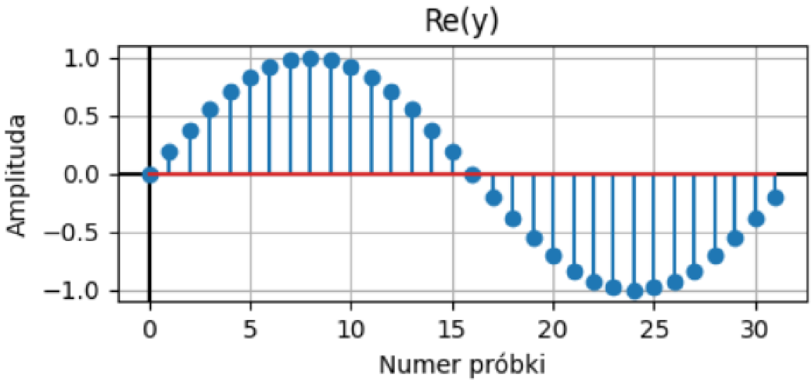
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import dft as dft

#stworzenie własnych funkcji DFT oraz IDFT
def DFT(y,n):
    m = dft(n)
    return y@m

def IDFT(f,n):
    m = dft(n)
    odw = np.conj(m)/n
    return f@odw
```

Tworzenie zmiennych:

```
n = 32
x = np.arange(n) #zekres 32 próbek
y = np.sin(2*np.pi*x/n) #funkcja sinusa
z = 2*np.fft.fft(y)/n #funkcja FTT z sinusa
ydft = 2*DFT(y,n)/n #funkcja własna DFT z sinusa

```

Wykres funkcji bazowej:

```
plt.title('Re(y)')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(y),use_line_collection=1)
plt.grid(True,which='both')

plt.show()
```

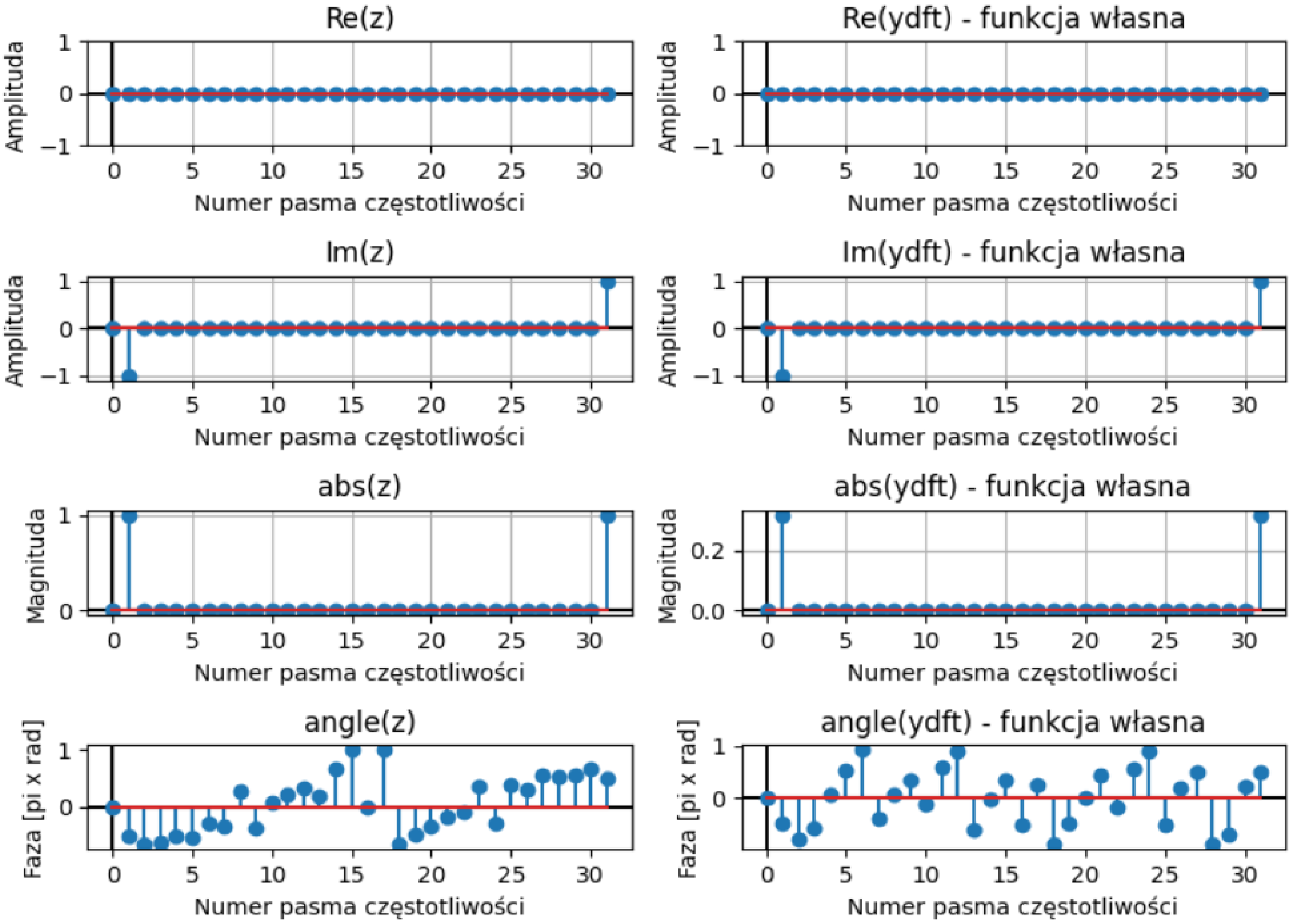


Porównanie własnej funkcji z funkcją z zadania 3.1:

```
plt.subplot(4,2,1)
plt.title('Re(z)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(z),use_line_collection=1)
plt.ylim(-1,1)
plt.grid(True,which='both')

plt.subplot(4,2,2)
plt.title('Re(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,1)
plt.stem(np.real(ydft),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,3)
plt.title('Im(z)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(z),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,4)
plt.title('Im(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(ydft),use_line_collection=1)
plt.grid(True,which='both')
```

```
plt.subplot(4,2,5)
plt.title('abs(z)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(z),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,6)
plt.title('abs(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(ydft)/np.pi,use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,7)
plt.title('angle(z)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(z)/np.pi,use_line_collection=1)

plt.subplot(4,2,8)
plt.title('angle(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(ydft)/np.pi,use_line_collection=1)

plt.show()
```

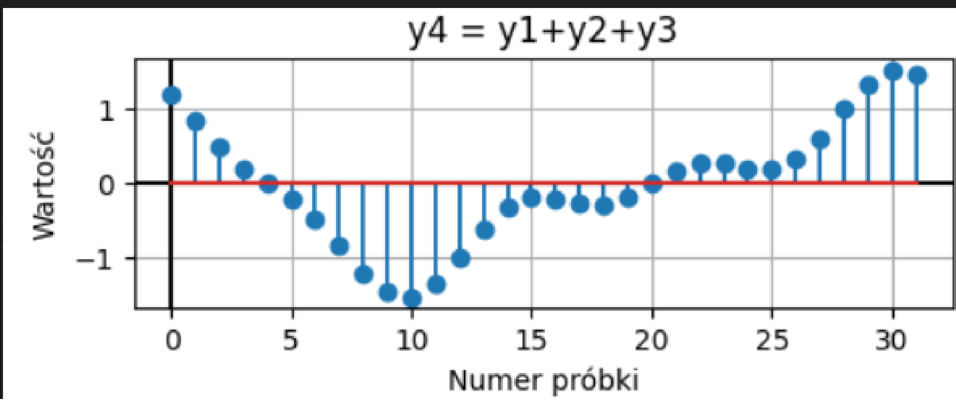
Wynik:



Deklaracja zmiennych z zadania 3.2 oraz stworzenie wykresy sumy sygnałów:

```
y1 = np.cos(2*np.pi*x/n+(np.pi/4)) #pierwsza funkcja z zadania 3.2
y2 = 0.5*np.cos(4*np.pi*x/n) #druga funkcja z zadania 3.2
y3 = 0.25*np.cos(8*np.pi*x/n+(np.pi/2)) #trzecia funkcja z zadania 3.2
y4 = y1 + y2 + y3   #suma funkcji
f4 = 2*np.fft.fft(y4)/n #FFT z sumy funkcji
ydft = 2*DFT(y4,n)/n #własne DFT z sumy funkcji
```

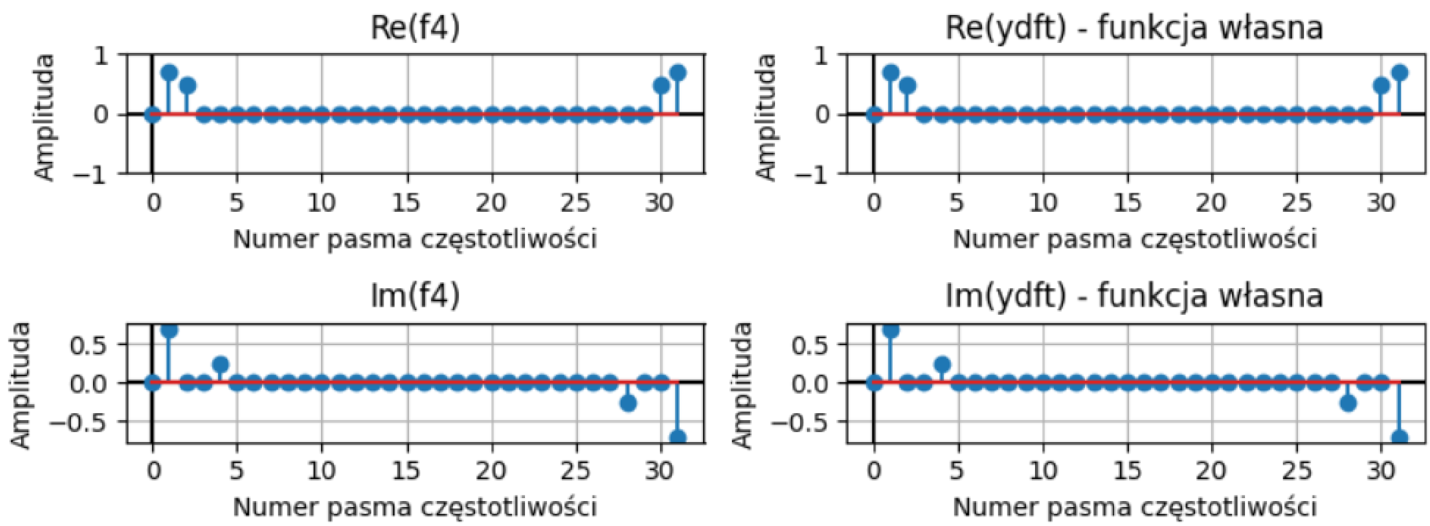
```
plt.title('y4 = y1+y2+y3')
plt.xlabel('Numer próbki')
plt.ylabel('Wartość')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(y4,use_line_collection=1
plt.grid(True,which='both')

plt.show()
```

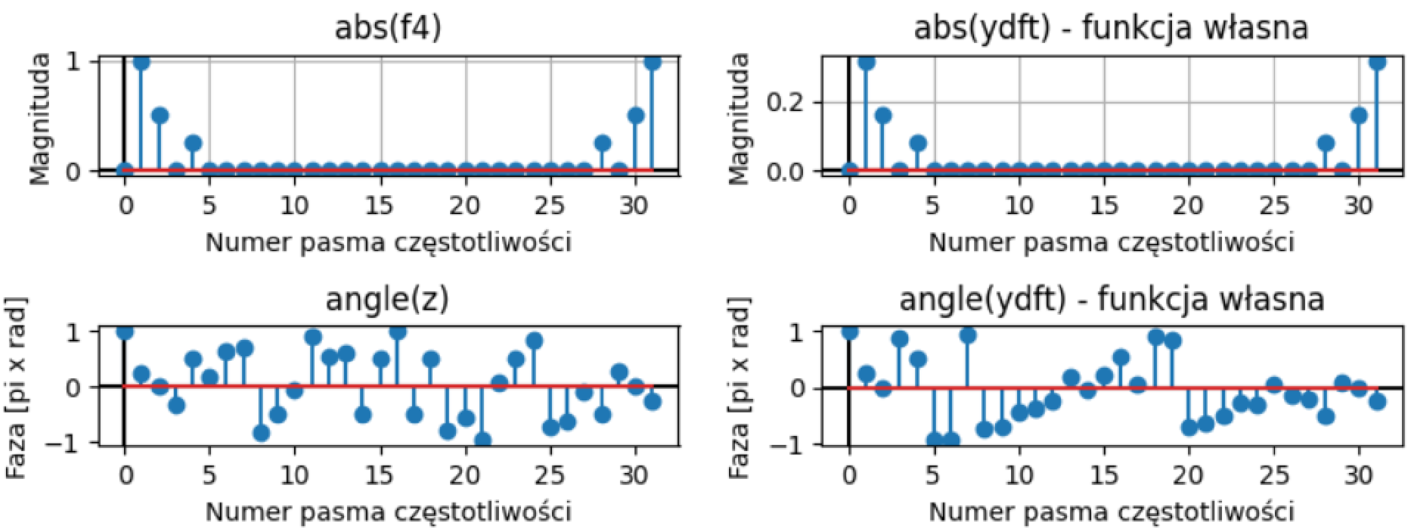


Porównanie własnej funkcji z funkcją z zadania 3.2:

```python
plt.subplot(4,2,1)
plt.title('Re(f4)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.real(f4),use_line_collection=1)
plt.ylim(-1,1)
plt.grid(True,which='both')

plt.subplot(4,2,2)
plt.title('Re(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.ylim(-1,1)
plt.stem(np.real(ydft),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,3)
plt.title('Im(f4)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(f4),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,4)
plt.title('Im(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.imag(ydft),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,5)
plt.title('abs(f4)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(f4),use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,6)
plt.title('abs(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.abs(ydft)/np.pi,use_line_collection=1)
plt.grid(True,which='both')

plt.subplot(4,2,7)
plt.title('angle(z)')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(f4)/np.pi,use_line_collection=1)

plt.subplot(4,2,8)
plt.title('angle(ydft) - funkcja własna')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Faza [pi x rad]')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(np.angle(ydft)/np.pi,use_line_collection=1)

plt.show()
```

Wyniki





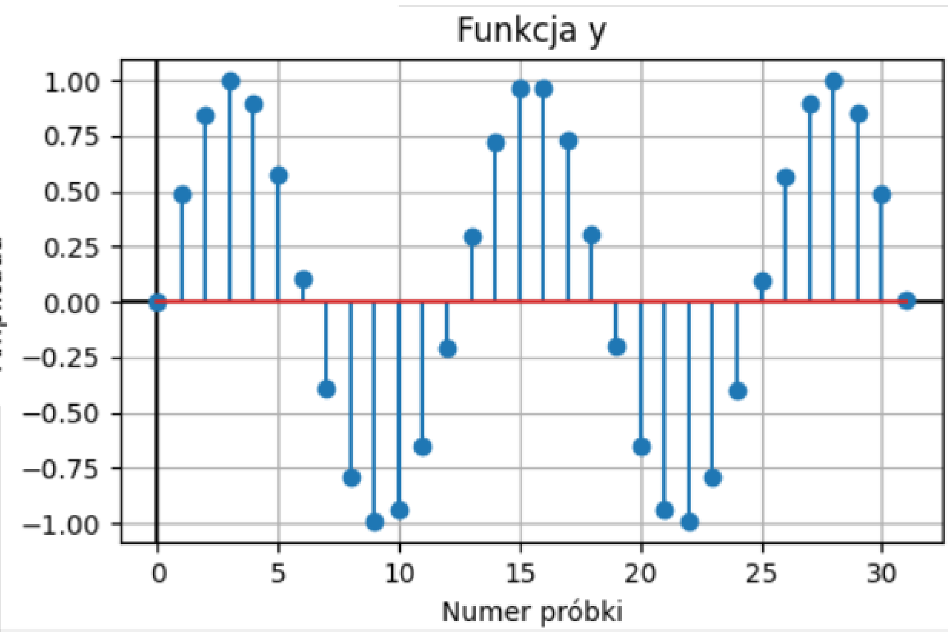
Zadanie 6

Treść zadania:

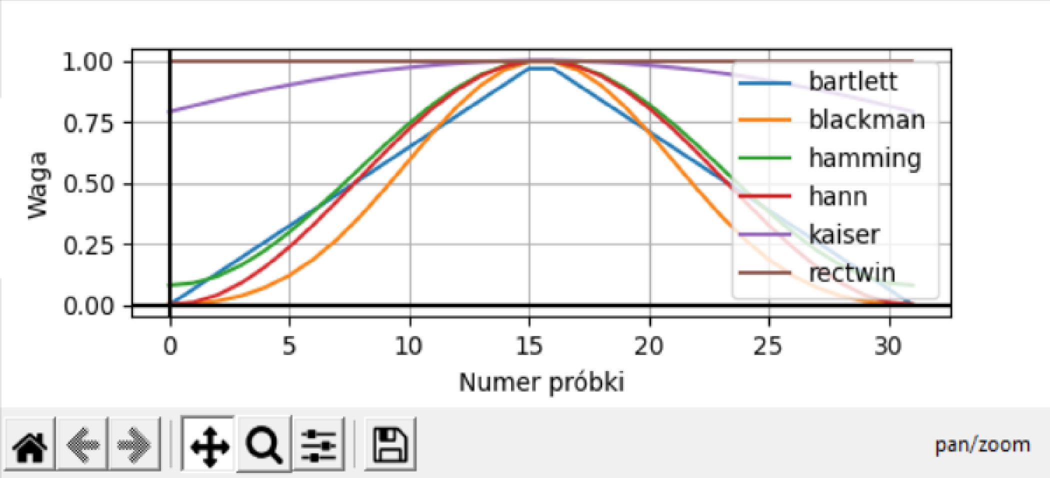
Wygeneruj 2.5 okresu sinusoidy. Sprawdź jak wygląda moduł widma zespolonego FFT sygnału.
• Skąd biorą się zniekształcenia?
• Porównaj z FFT sygnału wymnożonego przez funkcję okna. Zastosuj funkcję okna typu: bartlett, blackman, hamming, hann, kaiser, rectangular (okno prostokątne).
• Porównaj parametry okien.

Wygenerowanie 2,5 okresu sinusoidy:

```python
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sig

n = 32
x = np.arange(n)
y = np.sin(((5*np.pi)+0.5)*x/n)

plt.title('Funkcja y')
plt.xlabel('Numer próbki')
plt.ylabel('Amplituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(y,use_line_collection=1)
plt.grid(True,which='both')

plt.show()
```



Funkcje okna:

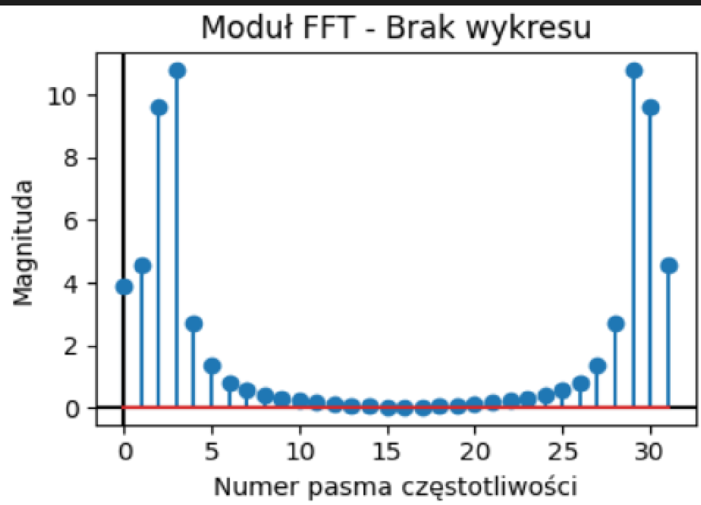


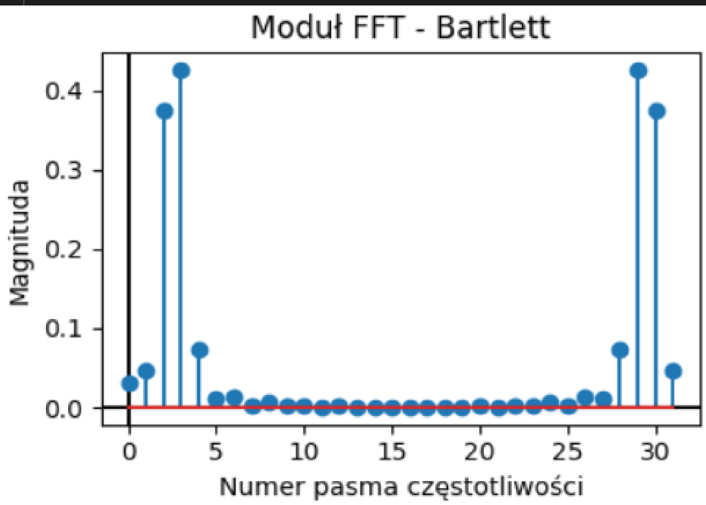

```
plt.xlabel('Numer próbki')
plt.ylabel('Waga')
plt.plot(np.bartlett(n))
plt.plot(np.blackman(n))
plt.plot(np.hamming(n))
plt.plot(sig.hann(n))
plt.plot(np.kaiser(n,1))
plt.plot(np.kaiser(n,0))
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.grid(True,which='both')
plt.legend(["bartlett","blackman","hamming","hann","kaiser","rectwin"],loc="upper right")
plt.show()
```

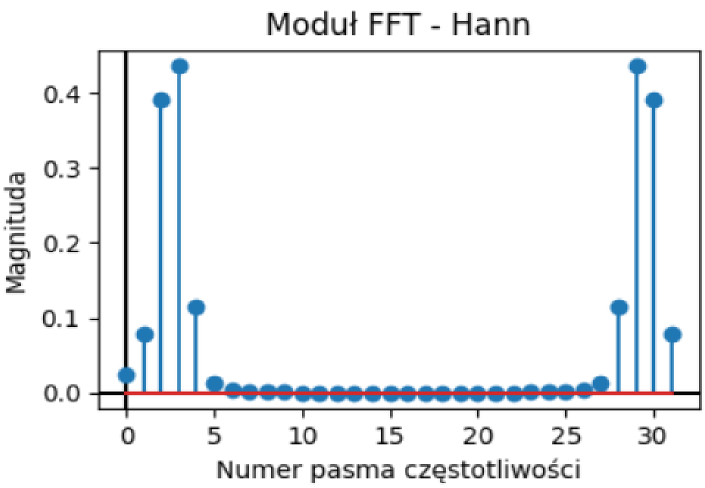
Porównanie FFT sygnału wymnożonego przez funkcję okna:

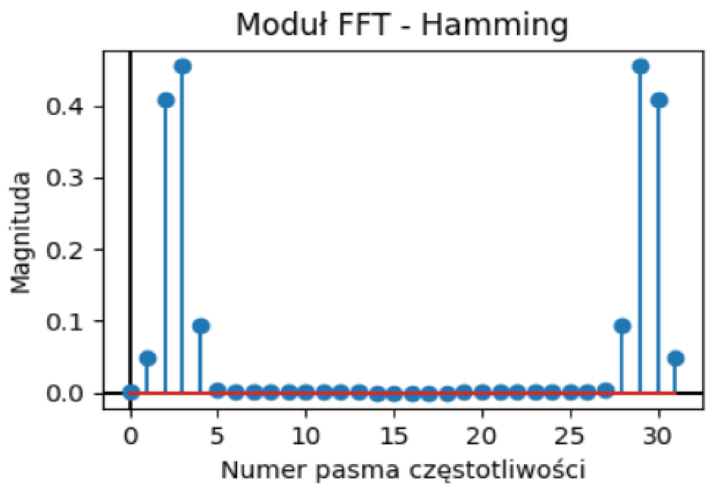
```
z = np.fft.fft(y)
plt.subplot(2,2,1)
plt.title('Moduł FFT - Brak wykresu')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z),use_line_collection=1)

z1 = np.bartlett(n)*y
z12 = 2*np.fft.fft(z1)/n
plt.subplot(2,2,2)
plt.title('Moduł FFT - Bartlett')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z12),use_line_collection=1)
```

```
z2 = sig.hann(n)*y
z22 = 2*np.fft.fft(z2)/n
plt.subplot(2,2,3)
plt.title('Moduł FFT - Hann')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z22),use_line_collection=1)

z3 = np.hamming(n)*y
z32 = 2*np.fft.fft(z3)/n
plt.subplot(2,2,4)
plt.title('Moduł FFT - Hamming')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z32),use_line_collection=1)

plt.show()
```



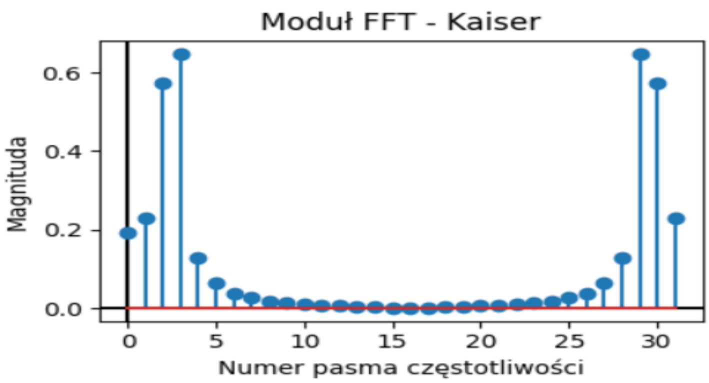


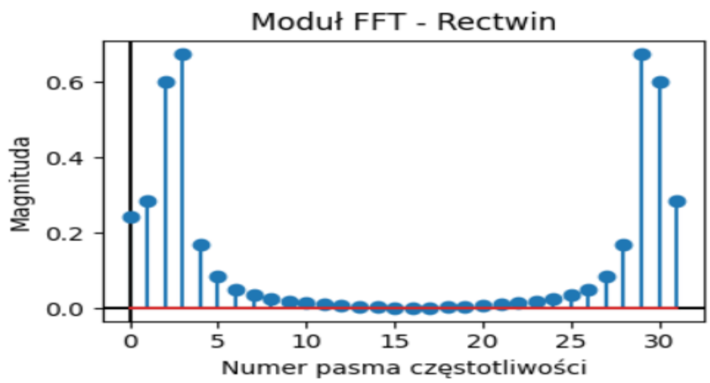


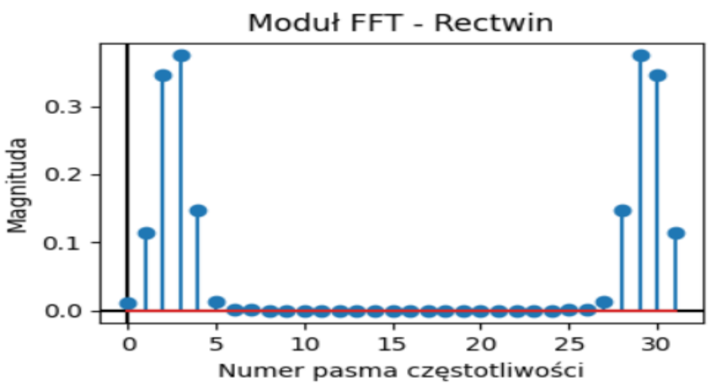



```python
z4 = np.kaiser(n,1)*y
z42 = 2*np.fft.fft(z4)/n
plt.subplot(2,2,1)
plt.title('Moduł FFT - Kaiser')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z42),use_line_collection=1)

z5 = np.kaiser(n,0)*y
z52 = 2*np.fft.fft(z5)/n
plt.subplot(2,2,2)
plt.title('Moduł FFT - Rectwin')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z52),use_line_collection=1)
```

```python


z6 = np.blackman(n)*y
z62 = 2*np.fft.fft(z6)/n
plt.subplot(2,2,3)
plt.title('Moduł FFT - Rectwin')
plt.xlabel('Numer pasma częstotliwości')
plt.ylabel('Magnituda')
plt.axhline(y=0,color = "k")
plt.axvline(x=0,color = "k")
plt.stem(abs(z62),use_line_collection=1)

plt.show()
```



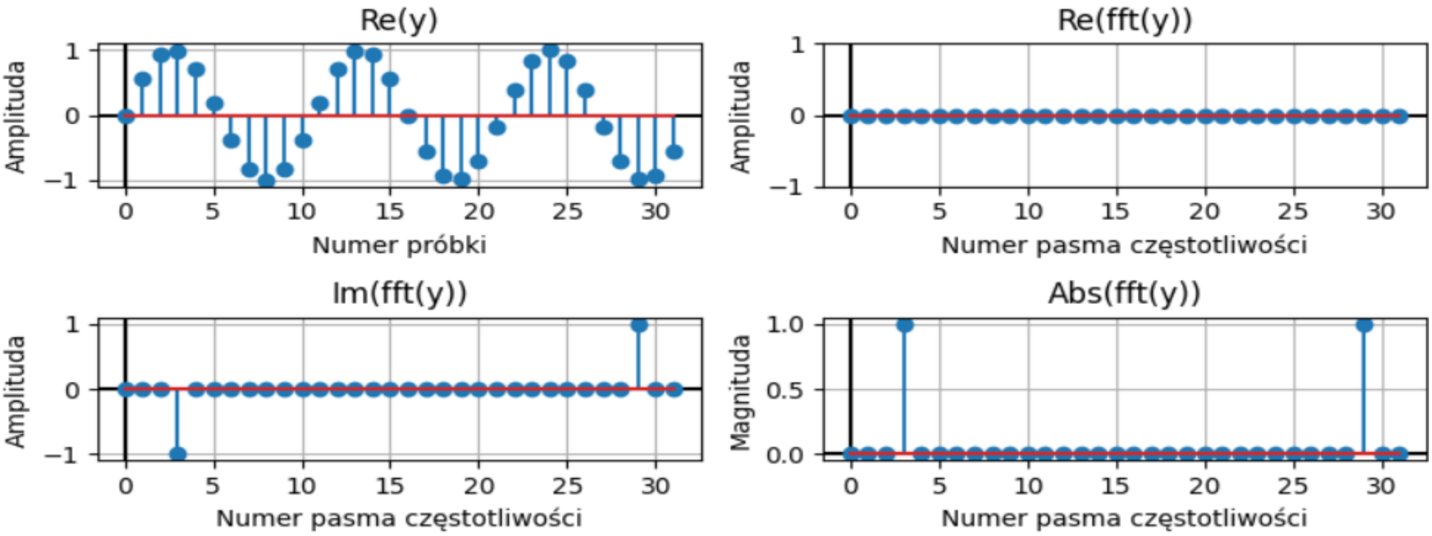




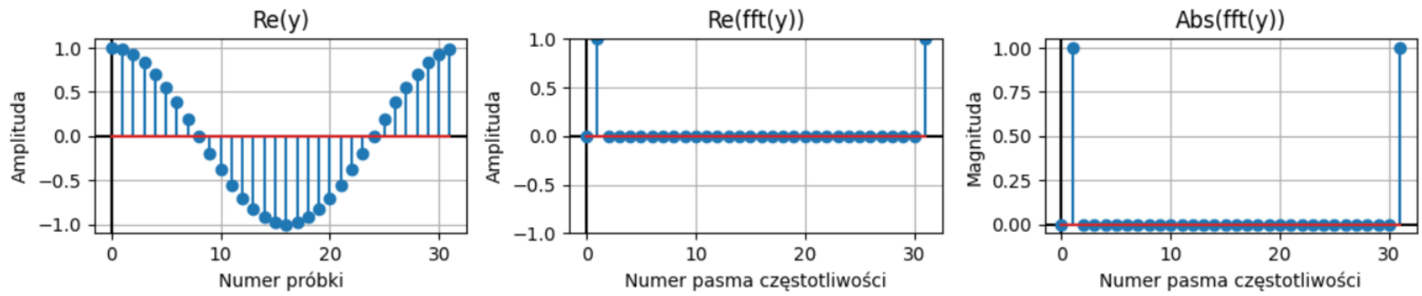
Interpretacja wyników i wnioski

Wynalezienie szeregu Fouriera jest niezwykle istotnym odkryciem. Pozwala rozłożyć dowolną okresową funkcję na szereg jej składowych. Dzięki temu jesteśmy w stanie przeanalizować ogromną ilość sygnałów, które na pierwszy rzut oka wydaję się być bardzo skomplikowane. Transformacja Fouriera również ułatwia analizowania sygnałów. Za jej pomocą jesteśmy w stanie przekształcić funkcję z dziedziną czasu w funkcję z dziedziną częstotliwość. Za pomocą graficznego przedstawienia widma jesteśmy w stanie określić na przykład z ilu częstotliwości składa się nasz początkowy sygnał. Używając języka Python korzystamy z szybkiej transformaty Fouriera. Dzieje się tak, ponieważ ma ona dużo lepszą złożoność obliczeniową (O (n log n)).

W pierwszym zadaniu badano transformatę Fouriera funkcji sinus. Zauważono, że dla częstotliwości f=1 na wykresie części urojonej oraz modułu powstały dwa „piki”. Dla części urojonej zachodzi symetria względem punktu (n/2 , 0), natomiast dla modułu widoczna jest symetria względem prostej x=n/2. Dodatkowo sprawdzono co się dzieje w przypadku zwiększenia częstotliwości. Okazało się, że „piki” przesuwają się do punktu (n/2 , 0).



Zbadano również jak zachowuje się widmo w przypadku funkcji cosinus. Otóż w części rzeczywistej jak i modułu pojawiają się dwa „piki’, w obu przypadkach są one symetryczne względem prostej x=n/2.



W zadaniu drugim porównano widma sygnałów składowych z widmem ich sumy. Zauważono, zauważono, że widmo sumy sygnałów jest połączeniem widm składowych.

W kolejnym zadaniu należało wyznaczyć odwrotną transformatę Fouriera dla widm FFT z poprzednich zadań. Skorzystano z funkcji ifft z pakietu NumPy. Działa ona poprawnie. Widma powstałe w wyniku odwrotnej transformaty pokrywają się z wykresami początkowymi sygnału. Oznacza to, że mając transformatę Fouriera, jesteśmy w stanie powrócić do sygnału początkowego.

W czwartym zadaniu porównywano widma sinusoidy zespolonej z tą z zadania 3.1. Różnią się one znacząco. W części urojonej w zadaniu 4 otrzymano tylko jeden „pik”, podobnie w przypadku modułu. Natomiast w zadaniu 1 pojawiają się dwa „piki”. Nie jesteśmy zatem w stanie mówić tu o symetrii. Zauważono również różnice w części urojonej obu funkcji y. W przypadku zadania 1 jest ona dla każdej próbki równa zeru, natomiast w sygnale sinusoidy zespolonej częścią urojoną jest cosinus.

W zadaniu piątym porównując wykresy otrzymane w zadaniu 3.1 oraz 3.2 i wykresów z użyciem funkcji własnej okazuje się, że wykresy są takie same z wyjątkiem fazy widma, które nieznacznie się różni. Oznacza to, że własna funkcja dobrze zastępuje polecenie np.fft.fft() i można ją poprawnie używać do generowania wykresów.

W kolejnym zadaniu badano zniekształcenia modułu widma zespolonego FFT. Zniekształcenia te powstają w wyniku zastosowania różnych funkcji typów okna przy tworzeniu wykresu. Każda funkcja typu okna różni się od drugiej w swojej składni, dlatego powstają też drobne zniekształcenia. Nie różnią się one bardzo od modułu FFT bez okna, lecz zwracają nieco inne amplitudy oraz częstotliwość w wykresach modułu z oknem różni się nieco od tego bez okna. Parametry w wykresach modułu z różnymi oknami są do siebie podobne.

Źródła

[1] http://wikidyd.iem.pw.edu.pl/attachments/LAPPD(2f)Cw2/c2_aws_instr.pdf
[2] http://aragorn.pb.bialystok.pl/~boldak/DIP/CPO-W06-v05-50pr.pdf
[3] https://pl.wikipedia.org
[4] https://home.agh.edu.pl/~zobmat/2020/II_mich_mar/wstep.html
[5]http://prac.im.pwr.edu.pl/~agniesz/rachunek_prawd_MAEW104/wyklady/R_Pr_MAEW104_wyklad2_szeregi_Fouriera.pdf
[6] http://wikidyd.iem.pw.edu.pl/attachments/LAPPD(2f)Cw2/c2_aws_instr.pdf
[7] https://eti.pg.edu.pl/documents/176593/26763380/Wykl_AlgorOblicz_10.pdf
[8] https://www.youtube.com/watch?v=spUNpyF58BY