Sprawozdanie z laboratorium Przetwarzanie Sygnałów i Obrazów

Ćwiczenie numer:

Temat: Analiza widma sygnałów

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

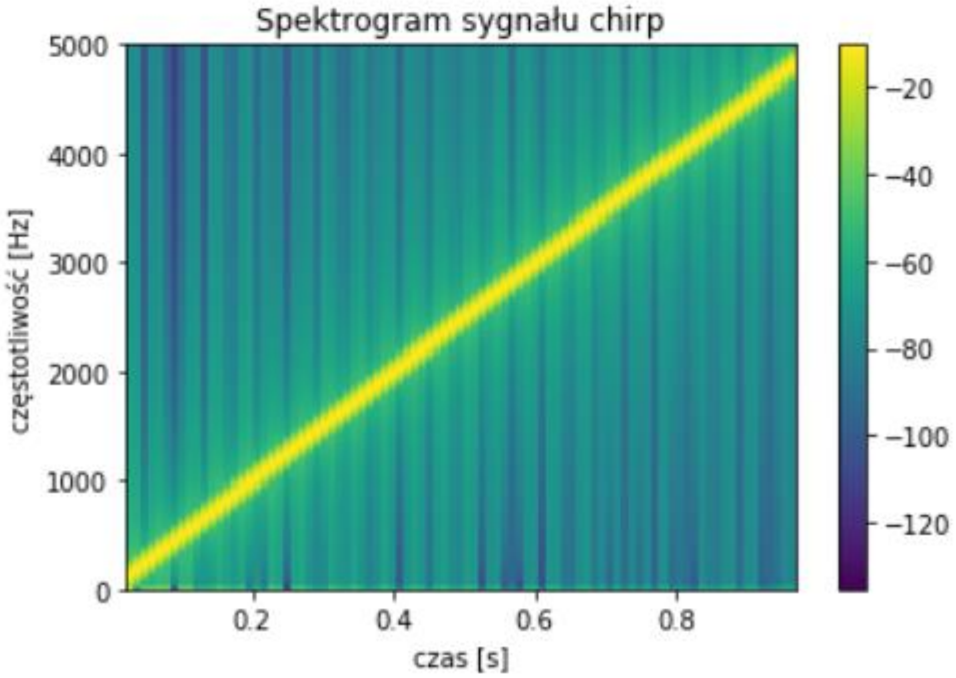
Data wykonania ćwiczenia
20.11.2021r.

....

Data i podpis prowadzącego

Teoria

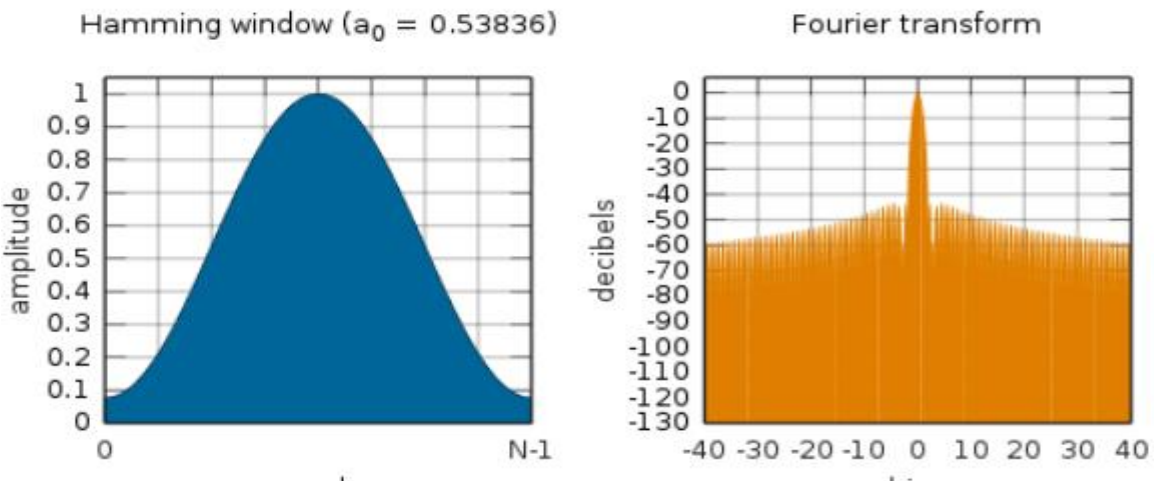
Spektrogram to wizualizacja widma częstotliwościowego sygnału. Oś pionowa to skala częstotliwości, oś pozioma to czas, kolor najczęściej odpowiada za amplitudę określonej częstotliwości w danym czasie.



Wykres 1 Przykłąd spektrogramu

Okna czasowe to funkcje stosowane wraz z dyskretną transformatą Fouriera. Opisują sposób pobierania próbek z sygnału. Analizując sygnał za pomocą DFT występuje zjawisko przecieku (w sygnale badanym występują kluczowe częstotliwości, które nie zostały wychwycone przez DFT). Aby temu zapobiec, używa się okien czasowych. Mając sygnał **u(x),** należy na niego nałożyć funkcję czasową okna **w(x).** Wynikiem tej operacji będzie nowy sygnał **y(x) = u(x)w(x).** W zależności od tego, które okno wybierzemy pojawią się inne różnice między widmem sygnału u(x) a widmem sygnału g(n). Jednym z nich jest **okno hamminga.** Jego kształt przypomina połowę cyklu fali kosinusoidalnej. Opisuje go wzór:

$$w(n) = 0{,}53836 - 0{,}46164\ \cos\left(\frac{2\pi n}{N-1}\right)$$



Wykres 2 Okno Hamminga

Periodogram obrazuje ciągły rozkład gęstości mocy sygnału w dziedzinie częstotliwości (power spectral density - psd). Może on być wyznaczany z transformaty Fouriera. Skuteczny głównie dla funkcji wyraźnie okresowych. W periodogramie wartość przebiegu widma jest przybliżona jako suma fal sinusoidalnych. Częstotliwości tych fal są wielokrotnościami odwrotności czasu trwania analizowanej próbki.

Wszystkie wyżej wymienione narzędzia pomagają w **analizie widmowej**, którą stosuje się do określenia składowych częstotliwościowych z badanego sygnału.

Zadanie 1

Treść zadania:

Wygenerować/nagrać następujące sygnały (długość 5 sekund każdy, tempo próbkowania fs = 8kHz):
a) szum gaussowski,
b) sygnał sinusoidalny o stałej częstotliwości 1kHz,
c) sygnał o zmiennej częstotliwości w zakresie od 0Hz (0s) do 1kHz (5s) (patrz funkcja chirp),
d) sygnał mowy.
Następnie, dla każdego z sygnałów wykreślić obwiednię mocy w czasie

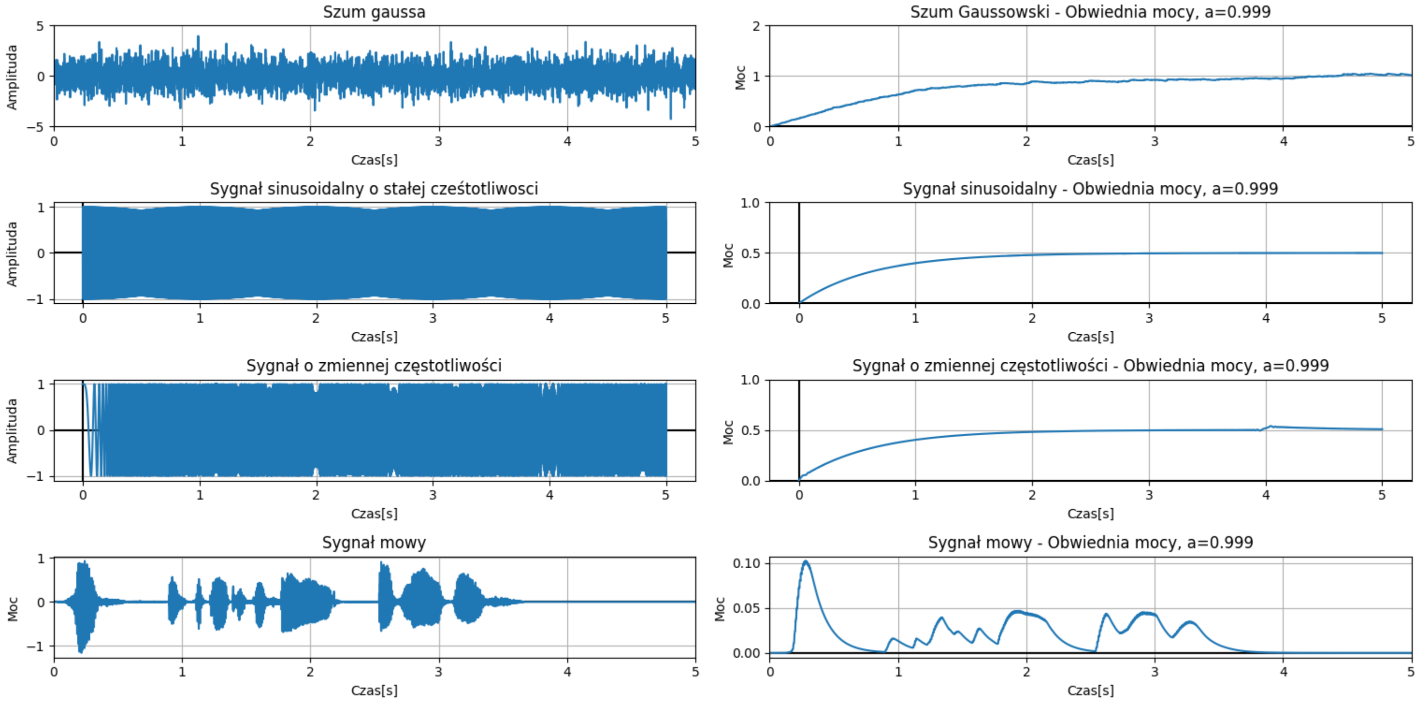
Realizacja w kodzie:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import chirp as chirp
from scipy.io import wavfile

def fun(x,a):
    d1 = len(x)
    F = (1-a)*x**2
    for i in range (1,d1):
        F[i] = a*F[i-1]+F[i]
    return F

def code(a,skala):

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

    plt.subplot(4,2,1)
    plt.title('Szum gaussa')
    plt.xlabel('Czas[s]')
    plt.ylabel('Amplituda')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.xlim(0,5000)
    plt.ylim(-5,5)
    plt.xticks([0,1000,2000,3000,4000,5000],[r'0',r'1',r'2',r'3',r'4',r'5'])
    plt.plot(x,y)
```

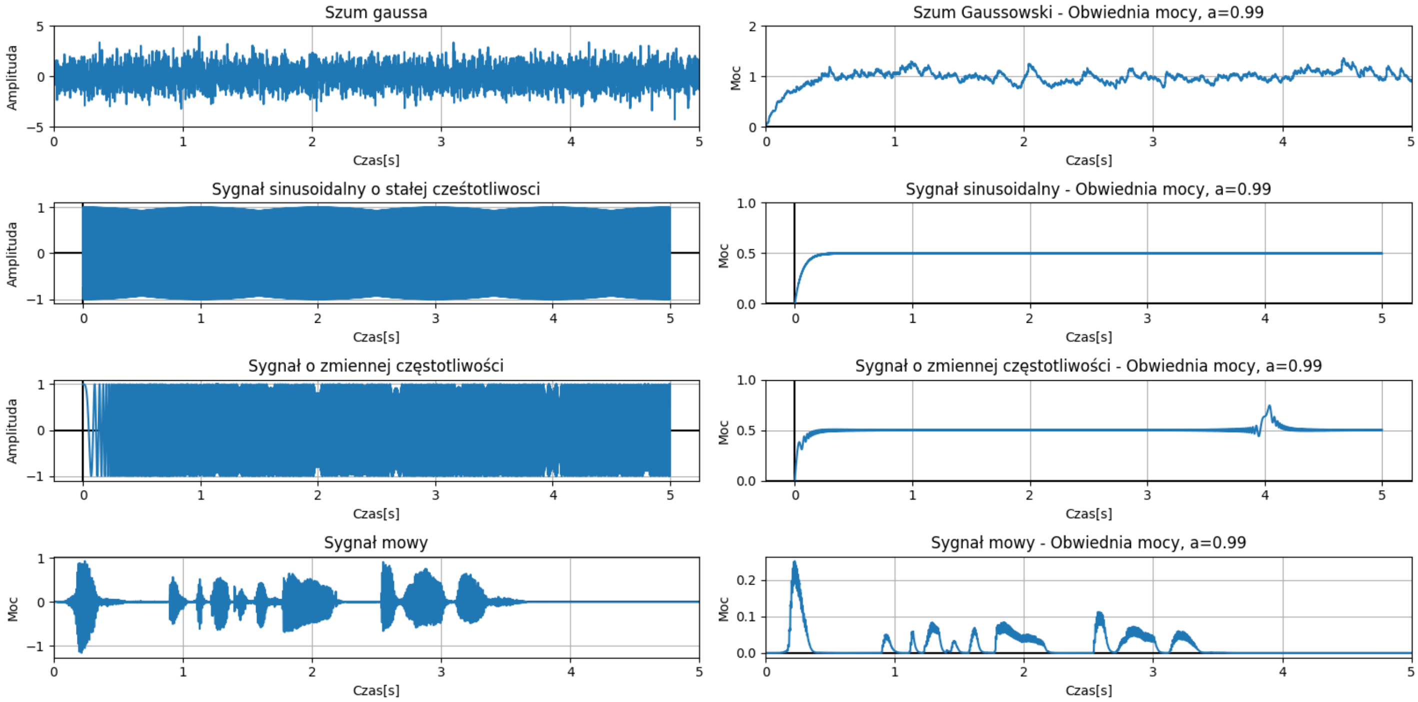
```
    plt.subplot(4,2,2)
    plt.title('Szum Gaussowski - Obwiednia mocy, a='+str(a))
    plt.xlabel('Czas[s]')
    plt.ylabel('Moc')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.xlim(0,5000)
    plt.ylim(0,skala)
    plt.xticks([0,1000,2000,3000,4000,5000],[r'0',r'1',r'2',r'3',r'4',r'5'])
    plt.plot(x,fun(y,a))

    plt.subplot(4,2,3)
    plt.title('Sygnał sinusoidalny o stałej cześtotliwosci')
    plt.xlabel('Czas[s]')
    plt.ylabel('Amplituda')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.plot(t,sin)

    plt.subplot(4,2,4)
    plt.title('Sygnał sinusoidalny - Obwiednia mocy, a='+str(a))
    plt.xlabel('Czas[s]')
    plt.ylabel('Moc')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.ylim(0,1)
    plt.plot(t,fun(sin,a))
    plt.subplot(4,2,5)
    ch = chirp(t,0,c,f)
    plt.title('Sygnał o zmiennej częstotliwości')
    plt.xlabel('Czas[s]')
    plt.ylabel('Amplituda')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.plot(t,ch)

    plt.subplot(4,2,6)
    plt.title('Sygnał o zmiennej częstotliwości - Obwiednia mocy, a='+str(a))
    plt.xlabel('Czas[s]')
    plt.ylabel('Moc')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.ylim(0,1)
    plt.plot(t,fun(ch,a))
```

```
    plt.subplot(4,2,7)
    plt.title('Sygnał mowy')
    plt.xlabel('Czas[s]')
    plt.ylabel('Moc')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.xticks([0,8000,16000,24000,32000,40000],[0,1,2,3,4,5])
    plt.xlim(0,40000)
    plt.plot(mowa)

    plt.subplot(4,2,8)
    plt.title('Sygnał mowy - Obwiednia mocy, a='+str(a))
    plt.xlabel('Czas[s]')
    plt.ylabel('Moc')
    plt.grid(True,which='both')
    plt.axhline(y=0,color='k')
    plt.axvline(x=0,color='k')
    plt.xticks([0,8000,16000,24000,32000,40000],[0,1,2,3,4,5])
    plt.xlim(0,40000)
    plt.plot(fun(mowa,a))

    plt.show()
c = 5
fs = 8000
f = 1000
t = np.linspace(0,c,fs)
sin= np.sin(2*np.pi*f*t)
x = np.arange(0,fs,1)
y = np.random.normal(0,1,fs)
rate,sound = wavfile.read('sound.wav')
mowa = sound/(rate*np.pi)

code(0.999,2)

code(0.99,2)

code(0.001,20)
```

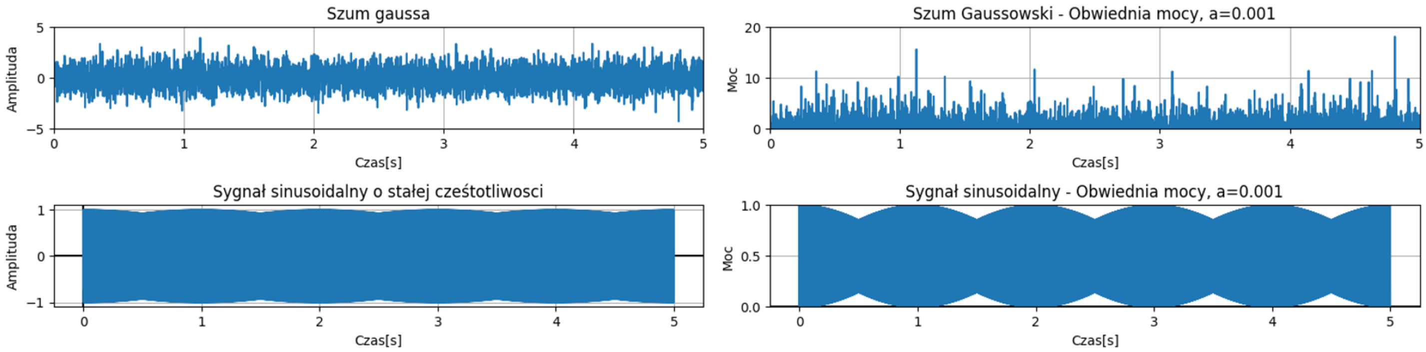
Sygnały i obwiednia mocy, a=0.999

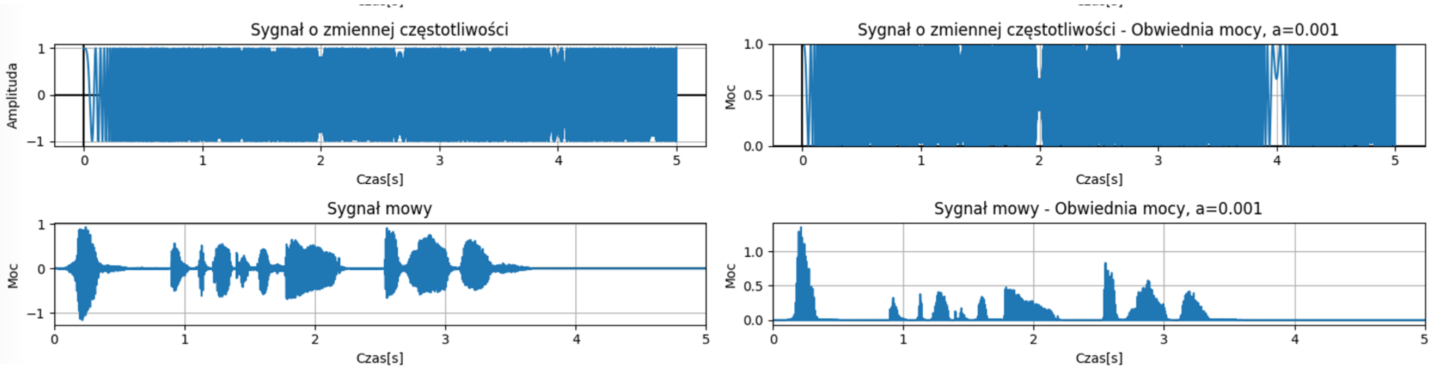


Sygnały i obwiednia mocy, a=0.99



Sygnały i obwiednia mocy, a=0.001





Zadanie 2

Treść zadania:
Wygenerować następujące sygnały (po N = 1024 próbek, każdy):szum gaussowski-w[n] z parametrami μ=0, σ
2 = 1, kombinację sygnałów sinusoidalnych o częstotliwościach f1 = 500Hz i f2 = 1.2kHz zgodnie ze wzorem s[n] = 0.5 sin(2πnf1/fs) + sin(2πnf2/fs) oraz sygnał y[n] = s[n] + 0.1w[n], (we wszystkich przypadkach założyć, że fs = 8kHz). Dla każdego z sygnałów oszacować widmową gęstość mocy (ang. Power Spectral Density - PSD) wykorzystując następujące metody:
a) periodogramu,
a) zmodyfikowanego periodogramu (dla okna Hanna),
a) Welcha (dla okna Hanna dł. 256, 128 i 64 próbki z 50% nakładaniem).
Sporządzić wykresy widmowej gęstości mocy w skali decybelowej (wyniki dla podpunktu c zaprezentować w tym
samym oknie).

Realizacja w kodzie

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import signal as sig
from scipy.signal import chirp as chirp
from scipy.signal import welch as welch
from scipy.signal import spectrogram as spectrogram
from scipy.io import wavfile
N = 1024
fs = 8000
noise =np.random.normal(0,1,N)
x = np.linspace(0,fs,N)
sinone = 0.5*np.sin(2*np.pi*x*500/fs)
sintwo = np.sin(2*np.pi*x*1200/fs)
sinthree = sinone - sintwo
y = sinthree + noise*0.1
f1, p1 = sig.periodogram(noise,fs)
f2, p2 = sig.periodogram(sinthree,fs)
f3, p3 = sig.periodogram(y,fs)
# a
plt.figure(figsize=(8,6),tight_layout=1)
```

```
plt.subplot(3,1,1)
plt.title('Periodiagram - oszacowanie widmowej sęstości mocy')
plt.xlabel('Częstotliwość (x π rad/numer próbki)')
plt.ylabel('Moc/Częstotliowść')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=1.245,color='grey',linestyle='--')
plt.yticks(np.arange(-80,1,20))
plt.plot(f1/N/np.pi, np.log(p1/N))

plt.subplot(3,1,2)
plt.title('Periodiagram - oszacowanie widmowej sęstości mocy')
plt.xlabel('Częstotliwość (x π rad/numer próbki)')
plt.ylabel('Moc/Częstotliowść')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=1.245,color='grey',linestyle='--')
plt.yticks(np.arange(-80,1,20))
plt.plot(f2/N/np.pi, np.log(p2/N))

plt.subplot(3,1,3)
plt.title('Periodiagram - oszacowanie widmowej sęstości mocy')
plt.xlabel('Częstotliwość (x π rad/numer próbki)')
plt.ylabel('Moc/Częstotliowść')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=1.245,color='grey',linestyle='--')
plt.yticks(np.arange(-80,1,20))
plt.plot(f3/N/np.pi, np.log(p3/N))
# b
plt.figure(figsize=(8,6),tight_layout=1)

fhann1,phann1 = sig.periodogram(noise,fs,window='hann')
fhann2,phann2 = sig.periodogram(sinthree,fs,window='hann')
fhann3,phann3 = sig.periodogram(y,fs,window='hann')

plt.subplot(3,1,1)
plt.title('Periodiagram - oszacowanie widmowej sęstości mocy')
plt.xlabel('Częstotliwość (x π rad/numer próbki)')
plt.ylabel('Moc/Częstotliowść')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=1.245,color='grey',linestyle='--')
plt.yticks(np.arange(-30,-9,5))
plt.plot(fhann1/N/np.pi, np.log(phann1/N))
```

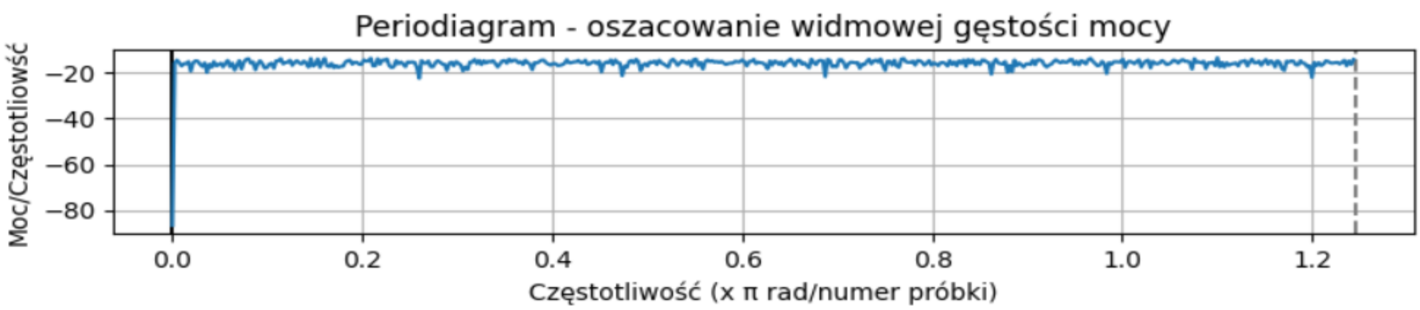
```python
plt.subplot(3,1,2)
plt.title('Periodiagram - oszacowanie widmowej sęstości mocy')
plt.xlabel('Częstotliwość (x π rad/numer próbki)')
plt.ylabel('Moc/Częstotliowść')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=1.245,color='grey',linestyle='--')
plt.yticks(np.arange(-50,-5,10))
plt.plot(fhann2/N/np.pi, np.log(phann2/N))

plt.subplot(3,1,3)
plt.title('Periodiagram - oszacowanie widmowej sęstości mocy')
plt.xlabel('Częstotliwość (x π rad/numer próbki)')
plt.ylabel('Moc/Częstotliowść')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=1.245,color='grey',linestyle='--')
plt.yticks(np.arange(-35,-5,5))
plt.plot(fhann3/N/np.pi, np.log(phann3/N))
plt.show()
```

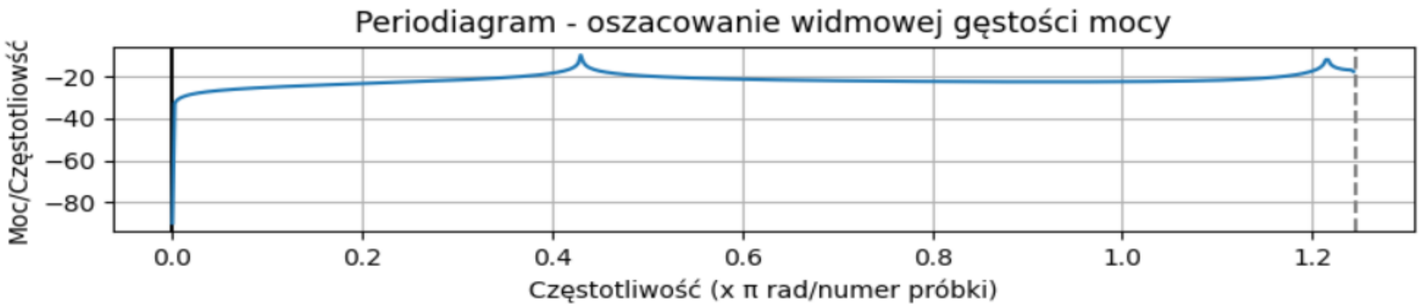
```python
# c
plt.figure(figsize=(8,6),tight_layout=1)
welchf1,welchp1 = welch(noise,fs,window='hanning',nperseg=256,noverlap=128)
welchf2,welchp2 = welch(noise,fs,window='hanning',nperseg=128,noverlap=64)
welchf3,welchp3 = welch(noise,fs,window='hanning',nperseg=64,noverlap=32)

plt.subplot(3,1,1)
plt.title('Szum gaussowski - Periodiagram Welscha')
plt.xlabel('Częstotliwość')
plt.ylabel('PSD (db/Hz)')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=4000,color='grey',linestyle='--')
plt.plot(welchf1,np.log(welchp1/N),label = 'NFFT=256')
plt.plot(welchf2,np.log(welchp2/N),label = 'NFFT=128')
plt.plot(welchf3,np.log(welchp3/N),label = 'NFFT=64')
plt.legend(loc='upper right')
```

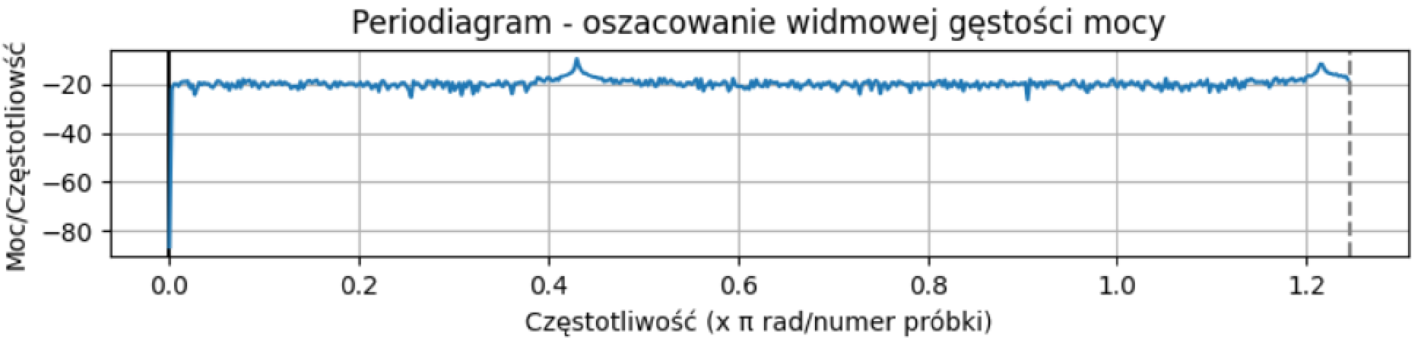
```python
welchfsin1,welchpsin1 = welch(sinthree,fs,window='hanning',nperseg=256,noverlap=128)
welchfsin2,welchpsin2 = welch(sinthree,fs,window='hanning',nperseg=128,noverlap=64)
welchfsin3,welchpsin3 = welch(sinthree,fs,window='hanning',nperseg=64,noverlap=32)
```

```python
plt.subplot(3,1,2)
plt.title('Szum sinusoidalny - Periodiagram Welscha')
plt.xlabel('Częstotliwość')
plt.ylabel('PSD (db/Hz)')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=4000,color='grey',linestyle='--')
plt.yticks(np.arange(-40,-5,5))
plt.plot(welchfsin1,np.log(welchpsin1/N),label = 'NFFT=256')
plt.plot(welchfsin2,np.log(welchpsin2/N),label = 'NFFT=128')
plt.plot(welchfsin3,np.log(welchpsin3/N),label = 'NFFT=64')
plt.legend(loc='upper right')

welchfy1,welchpy1 = welch(y,fs,window='hanning',nperseg=256,noverlap=128)
welchfy2,welchpy2 = welch(y,fs,window='hanning',nperseg=128,noverlap=64)
welchfy3,welchpy3 = welch(y,fs,window='hanning',nperseg=64,noverlap=32)

plt.subplot(3,1,3)
plt.title('Sygnał w szumie - Periodiagram Welscha')
plt.xlabel('Częstotliwość')
plt.ylabel('PSD (db/Hz)')
plt.grid(True,which='both')
plt.axvline(x=0,color='k')
plt.axvline(x=4000,color='grey',linestyle='--')
plt.yticks(np.arange(-25,-5,2))
plt.plot(welchfy1,np.log(welchpy1/N),label = 'NFFT=256')
plt.plot(welchfy2,np.log(welchpy2/N),label = 'NFFT=128')
plt.plot(welchfy3,np.log(welchpy3/N),label = 'NFFT=64')
plt.legend(loc='upper right')
```

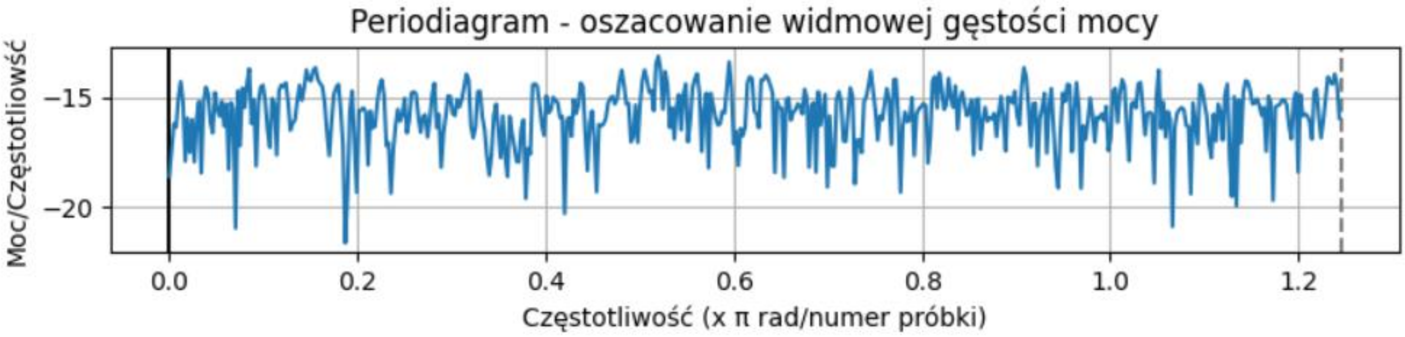
Periodogram:

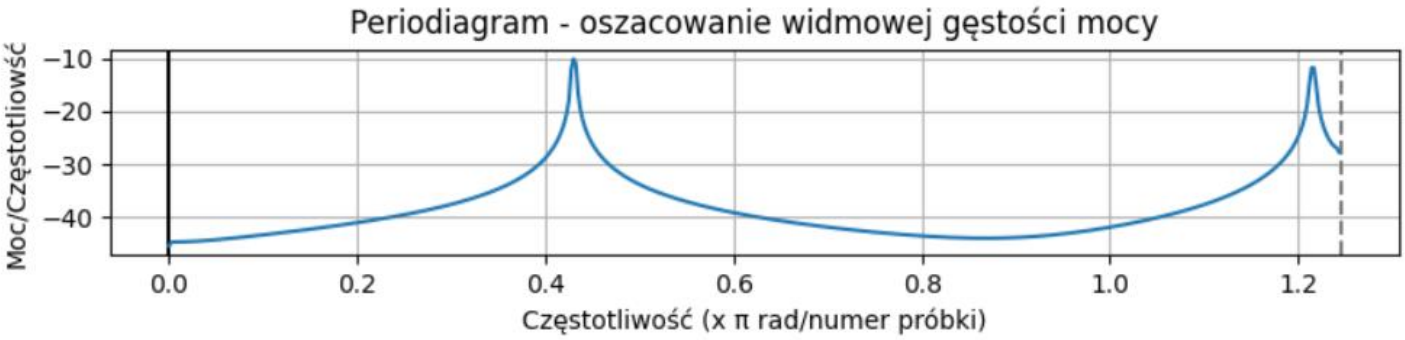




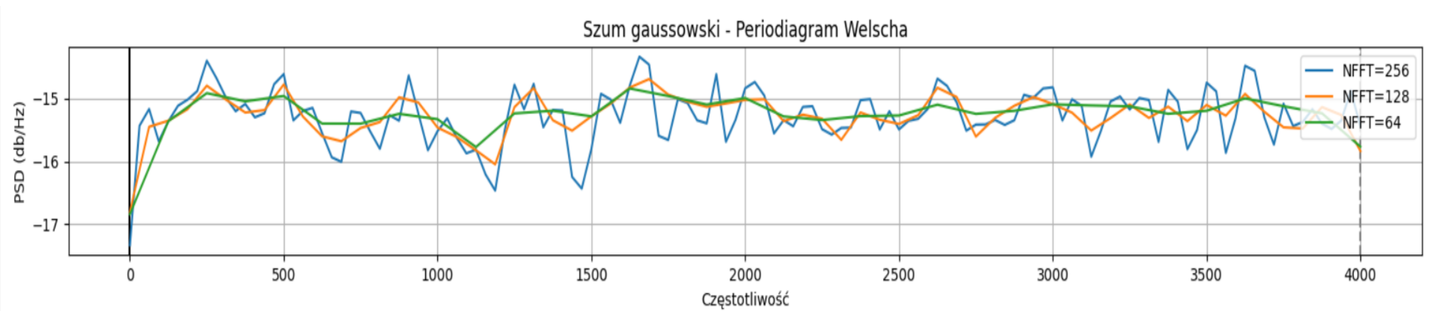


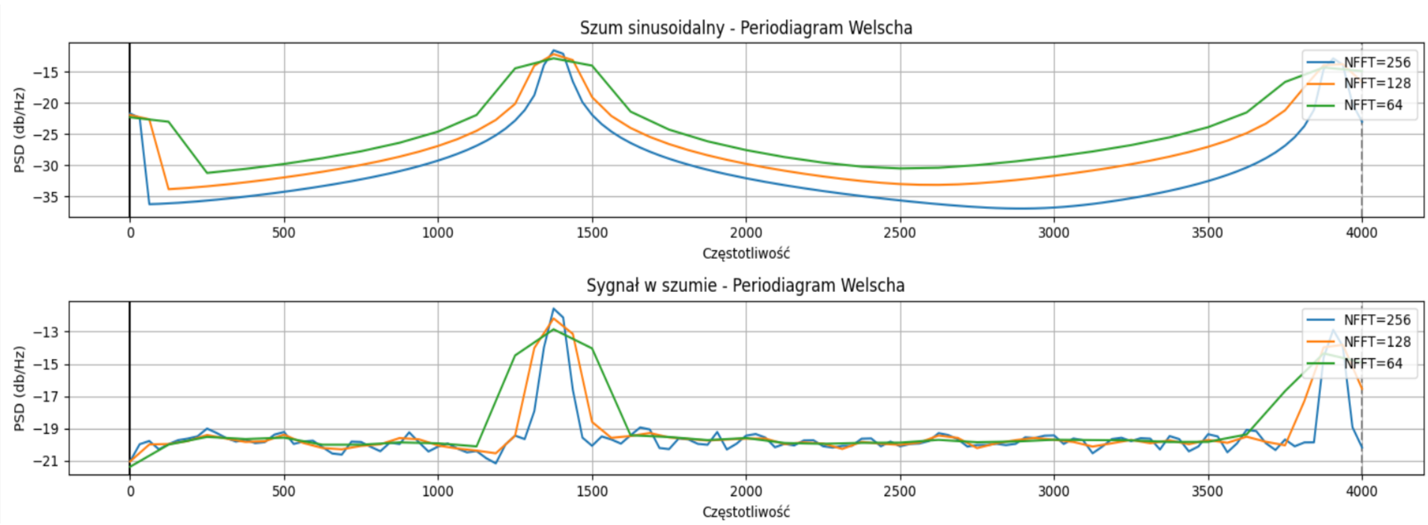
Zmodyfikowany Periodogram (dla okna Hanna):





Periodogram Welsha:





Zadanie 3

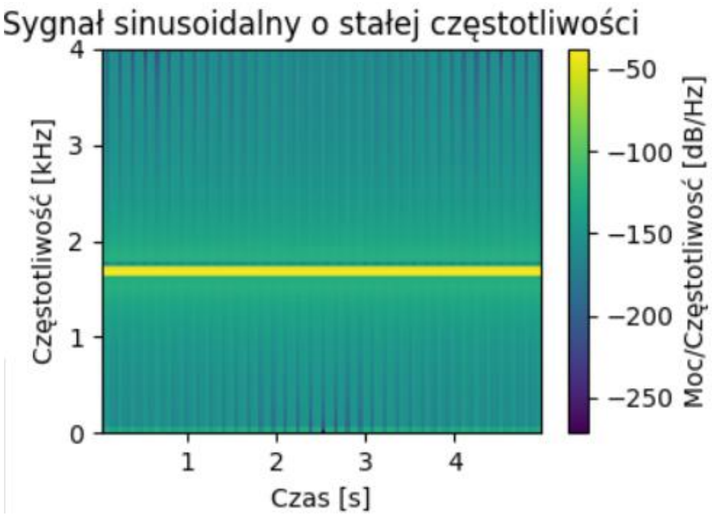
Treść zadania:
Wykorzystując funkcję spectrogram sporządzić spektrogramy dla sygnałów z zadania 4.1. Założyć, że oknem analizy jest okno Hamminga długości 256 próbek. Zwróć uwagę na to, by osie wykresu były opisane przy użyciu jednostek fizycznych (a nie znormalizowanych). Co możesz powiedzieć o rozkładzie energii w czasie i częstotliwości analizowanych sygnałów? Jaka jest relacja pomiędzy modułem widma krótkookresowego a widmową gęstością mocy w skali decybelowej?

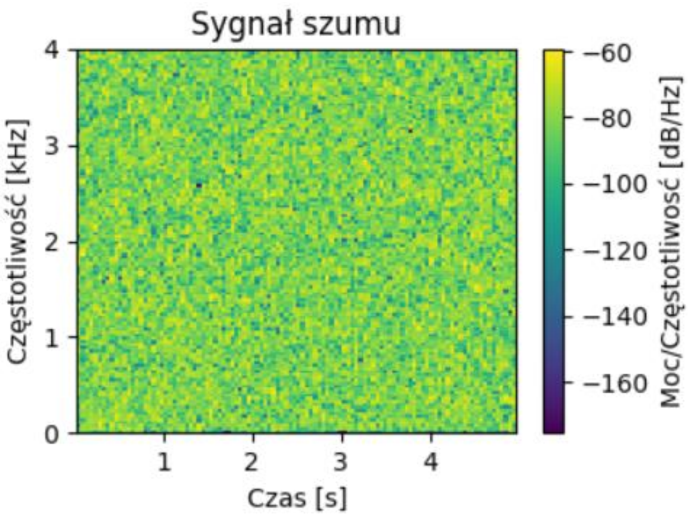
Realizacja w kodzie

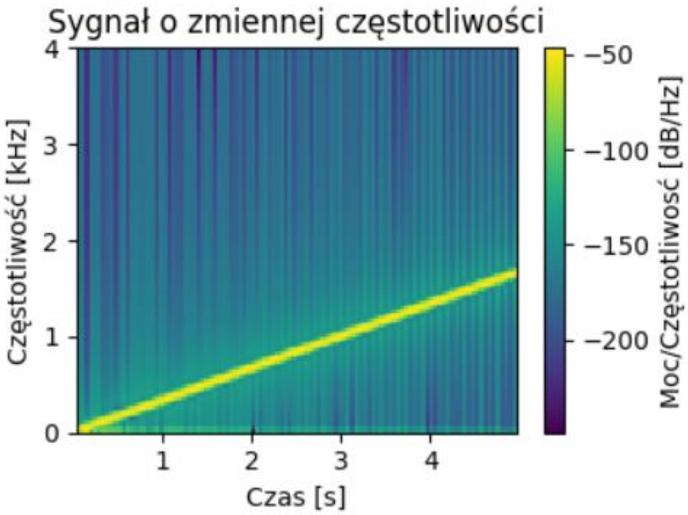
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import chirp as chirp
from scipy.signal import spectrogram as spectrogram
from scipy.io import wavfile

f = 1000
fs = 24000
c = 5
t = np.linspace(0,c,fs)
sin = 2*np.sin(2*np.pi*f*t)
f,t,Sxx = spectrogram(sin, fs, 'hamming', 256)
Sxx = 20*np.log10(Sxx)
rate,sound = wavfile.read('sound.wav')

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

plt.subplot(2,2,1)
plt.title('Sygnał sinusoidalny o stałej częstotliwości')
plt.ylabel('Częstotliwość [kHz]')
plt.xlabel('Czas [s]')
plt.pcolormesh(t,f/3000,Sxx)
bar = plt.colorbar()
plt.xticks([0.2,0.4,0.6,0.8],[1,2,3,4])
bar.ax.set_ylabel('Moc/Częstotliwość [dB/Hz]')
```

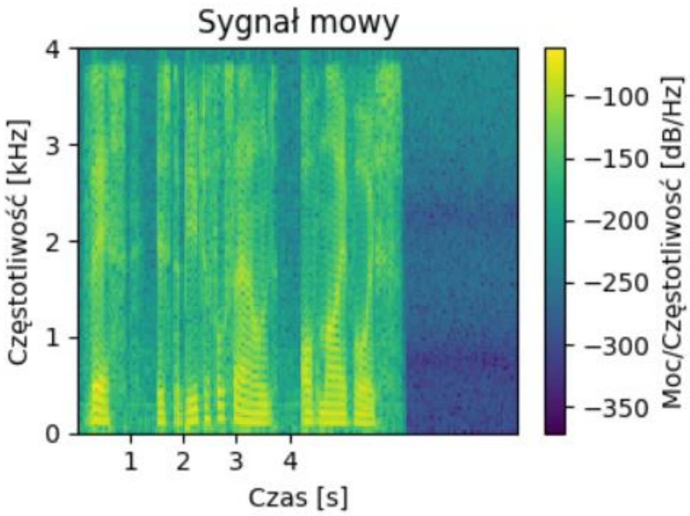
```python
y = np.random.normal(0,1.2,fs)
x =np.arange(0,fs,1)
f,t,Sxx = spectrogram(y,fs,'hamming',256)
Sxx = 20*np.log10(Sxx)

plt.title('Sygnał szumu')
plt.ylabel('Częstotliwość [kHz]')
plt.xlabel('Czas [s]')
plt.pcolormesh(t,f/3000,Sxx)
bar = plt.colorbar()
plt.xticks([0.2,0.4,0.6,0.8],[1,2,3,4])
bar.ax.set_ylabel('Moc/Częstotliwość [dB/Hz]')

plt.subplot(2,2,3)

f = 1000
t = np.linspace(0,c,fs)
ch = chirp(t,0,c,f)
f,t,Sxx = spectrogram(ch,fs,'hamming',256)
Sxx = 20*np.log10(Sxx)

plt.title('Sygnał o zmiennej częstotliwości')
plt.ylabel('Częstotliwość [kHz]')
plt.xlabel('Czas [s]')
plt.pcolormesh(t,f/3000,Sxx)
bar = plt.colorbar()
plt.xticks([0.2,0.4,0.6,0.8],[1,2,3,4])
bar.ax.set_ylabel('Moc/Częstotliwość [dB/Hz]')
f = 1000
t = np.linspace(0,c,fs)
f,t,Sxx = spectrogram(sound/(rate*np.pi),fs,'hamming',256)
Sxx = 20*np.log10(Sxx)

plt.title('Sygnał mowy')
plt.ylabel('Częstotliwość [kHz]')
plt.xlabel('Czas [s]')
plt.pcolormesh(t,f/3000,Sxx)
bar = plt.colorbar()
plt.xticks([0.2,0.4,0.6,0.8],[1,2,3,4])
bar.ax.set_ylabel('Moc/Częstotliwość [dB/Hz]')

plt.show()
```

Wyniki:









Zadanie 4

Treść zadania:

Napisać własną wersję funkcji spectrogram, wykorzystującą jako narzędzie analizy widmowej metodę uśrednianych periodogramów (ang. smoothed periodograms) i rysującą wykres czasowo częstotliwościowy w postaci siatki 3D (funkcja mesh). Przyjąć, że oś X jest osią czasu, oś Y - osią częstotliwości oraz oś Z - osią widmowej gęstości mocy (w skali decybelowej). Podpowiedź: w celu implementacji metody uśrednianych periodogramów wykorzystać wzór z zadania 4.1 zastępując moc chwilową w czasie x[n] 2 wartościami krótkookresowego widma mocy |X(k, l)| 2 , gdzie X(k, l) - k-ty prążek widma zespolonego DFT oraz l-indeks ramki sygnału. Sporządzić wykresy analogicznie jak w zadaniu 4.3, porównać wyniki.

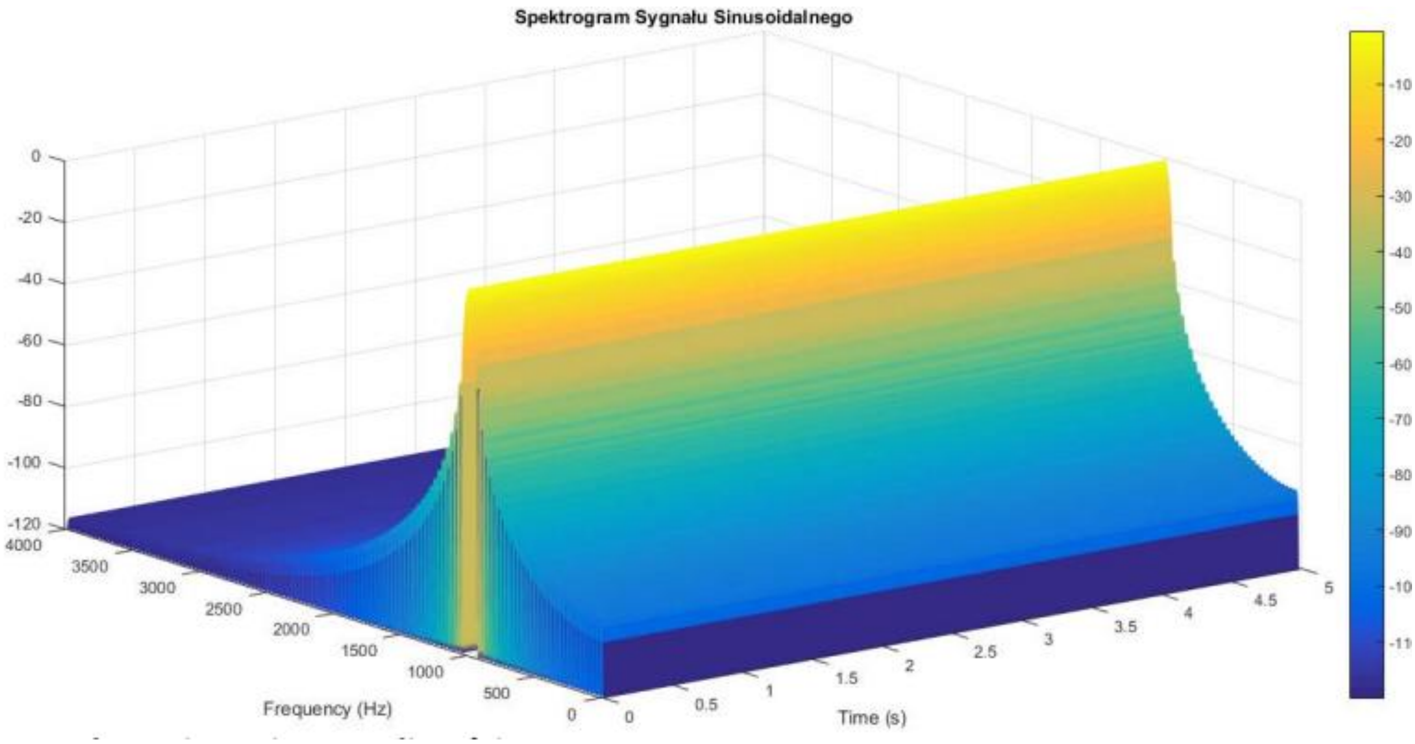
Realizacja w kodzie:

```
function [stft, f, t] = stft(x, nfft, fs)
step = 16;
if size(x, 2) > 1
x = x';
end
maxX = max(abs(x));
x = x/maxX;
signalL = length(x);
win = hann(256, 'periodic');
rows = ceil((1+nfft)/2);
columns = 1+fix((signalL-256)/step);
stft = zeros(rows, columns);
indeks = 0;
kolumna = 1;
while indeks + 256 <= signalL
xw = x(indeks+1:indeks+256).*win;
X = fft(xw, nfft);
stft(:, kolumna) = X(1:rows);
indeks = indeks + step;
kolumna = kolumna + 1;
end
t = (256/2:step:256/2+(columns-1)*step)/fs;
f = (0:rows-1)*fs/nfft;
K = sum(hamming(256, 'periodic'))/256;
stft = abs(stft)/256/K;
if rem(nfft, 2)
   st(2:end, :) = stft(2:end, :).*2;
else
stft(2:end-1, :) = stft(2:end-1, :).*2;
end
stft = 20*log10(stft + 1e-6);
end
```

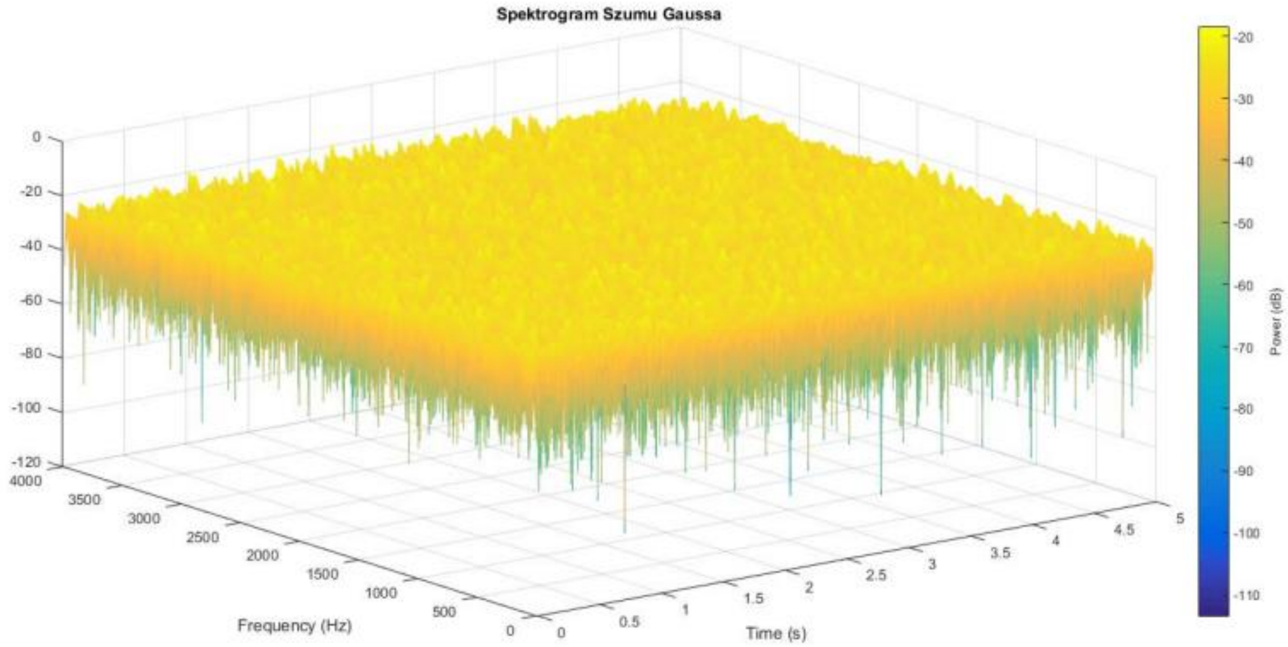
1) Szum gaussowski
figure; fs = 8000;
czas = 5;
t = [0:1/fs:czas-1/fs];
y1 = randn(czas*fs,1); %5 sekund.
y1 = transpose(y1);
[s, f, t] = stft(y1, 4096, fs); %% 4096 ->najlepiej potęga 2
mesh(t, f, s);
xlabel('Czas (s)');
ylabel('Częstotliwość (Hz)');
kolory = colorbar;
ylabel(kolory, 'Moc dB');
	3) Zmienna częstoliwość
	figure; fs = 8000;
	czas = 5;
	t = [0:1/fs:czas-1/fs];
	y3 =chirp(t,0,5,1000,'quadratic',[],'concave');
	[s, f, t] = stft(y3, 2048, fs);
	mesh(t, f, s);
	xlabel('Czas (s)');
	ylabel('Częstotliwość (Hz)');

2) Sygnał sinusoidalny
figure; fs = 8000;
czas = 5;
t = [0:1/fs:czas-1/fs];
y2 = sin(2000*pi*t);
[s, f, t] = stft(y2, 2048, fs);
mesh(t, f, s);
xlabel('Czas (s)');
ylabel('Częstotliwość (Hz)');
title('SPEKTROGRAM');
kolory = colorbar;
ylabel(kolory, 'Moc dB');
	4) Sygnał mowy
	figure; fs = 8000;
	czas = 5;
	t = [0:1/fs:czas-1/fs];
	[y4, fstest] = wavread('outputfile.wav');
	y4 = removerows(y4, 1);
	y4 = transpose(y4);
	y4 = removerows(y4, 1);
	[s, f, t] = stft(y4, 2048, fs);
	mesh(t, f, s);
	xlabel('Czas (s)');

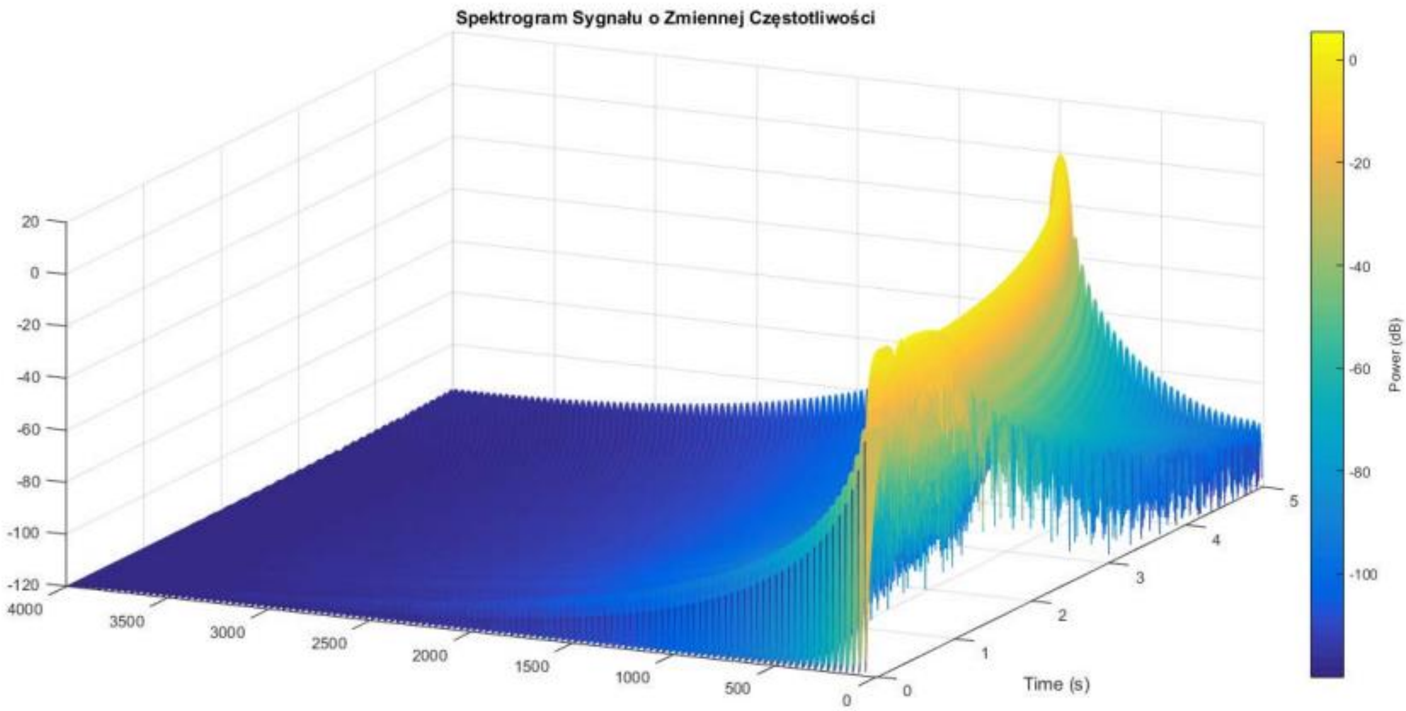
Spektrogram sygnału sinusoidalnego



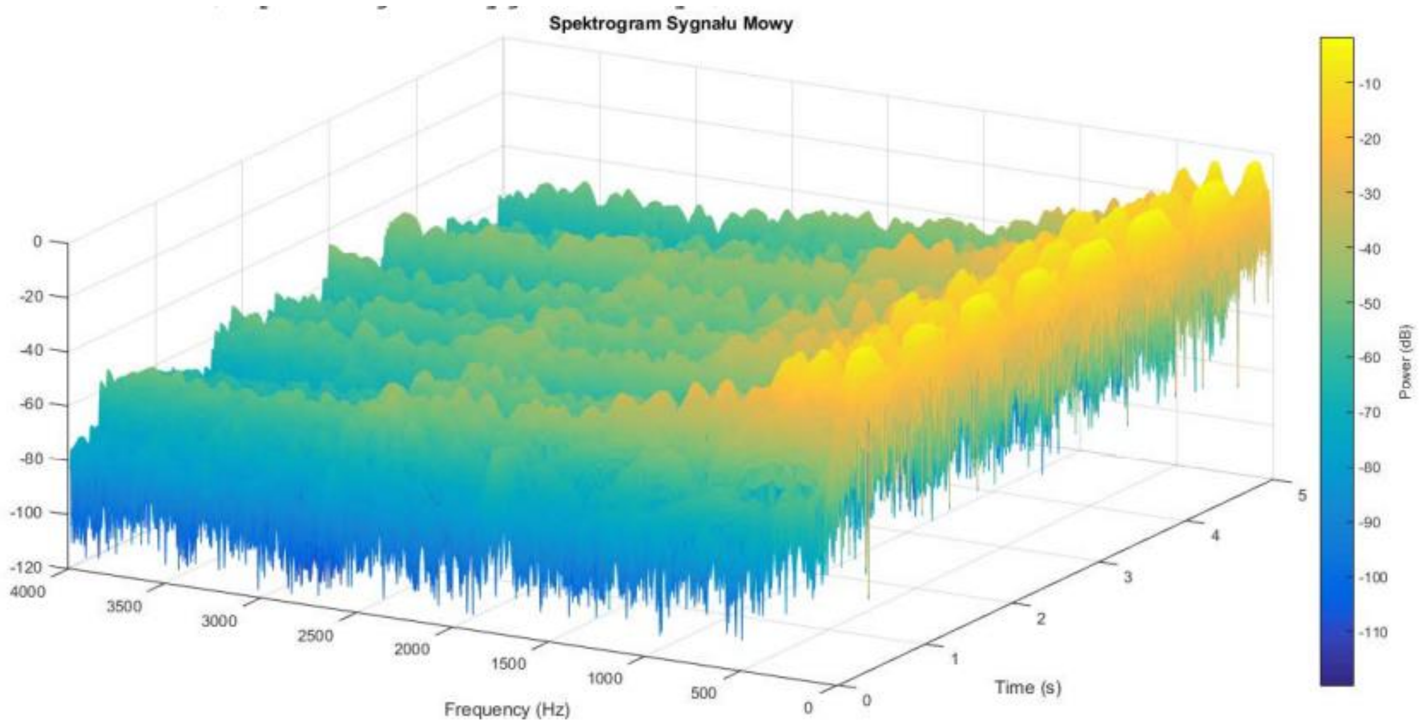
Spektrogram szumu Gaussa:



Spektrogram sygnału o zmiennej częstotliwości:



Spektrogram sygnału mowy:



Interpretacja wyników i wnioski

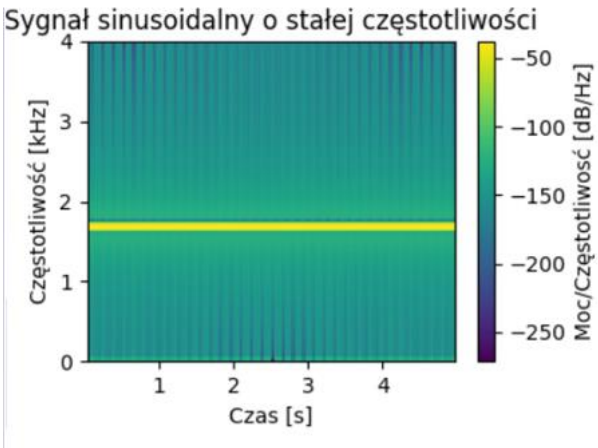
Dzięki wykonanym zadaniom zdobyto niezbędną wiedzę z zakresu analizy widmowej sygnałów. Nauczono się tworzyć własne funkcje służące do tworzenia wykresów obwiedni mocy, pogłębiono wiedzę z zakresu tworzenia wykresów, sinusoidalnych, szumu Gaussowskiego, sygnałów o zmiennej częstotliwości oraz sygnału mowy. Nauczono się korzystać z funkcji spectrogram oraz jak zaimplementować własną funkcję spectrogram. Zdobyto także wiedzę z zakresu jak poprawnie analizować uzyskany wykres.

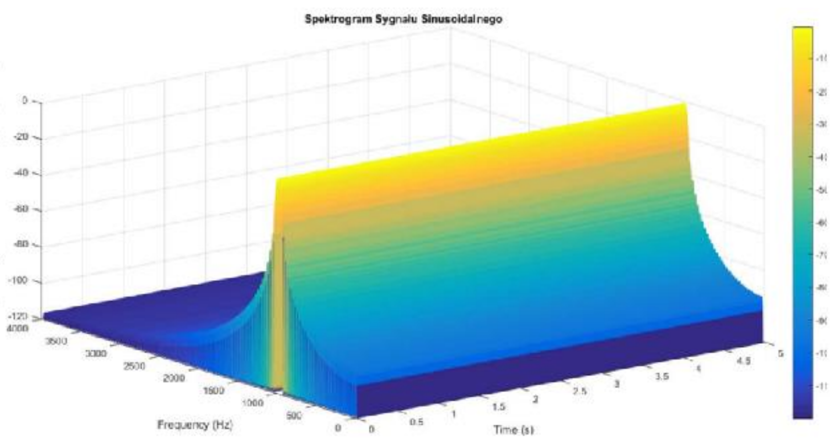
W zadaniu pierwszym dobierano parametry alfa. Zauważono, że dobór parametru alfa ma duży wpływ na wygląd wykresów obwiedni mocy. Im parametr alfy jest bardziej zbliżony do 1 tym wykresy obwiedni mocy są dokładniejsze i bardziej szczegółowe. Im bliżej do 0 tym wykresy są mniej szczegółowe a na ich podstawie ciężko odczytać obwiednię mocy dla wykresów. Na podstawie kształtu obwiedni można stwierdzić, że przy alfie a = 0.999 wykresy obwiedni mocy dla szumu Gaussowskiego, sygnału sinusoidalnego oraz sygnału o zmiennej częstotliwości wykresy są stacjonarne. W przypadku sygnału mowy nie można stwierdzić, że wykres obwiedni mocy jest stacjonarny, jest raczej słabo stacjonarny albo niestacjonarny. Na podstawie obwiedni dla alfy a = 0.001 można stwierdzić, że wykres obwiedni dla sygnału sinusoidalnego oraz sygnału o zmiennej częstotliwości są stacjonarne.

Wykonując zadanie drugie zauważono, że żaden periodogram nie jest doskonały i często zawierają one błędy. Często zdarza się, że są one naprawdę duże. Wykresy są często postrzępione, a to z powodu posiadanej bardzo dużej wariancji oraz dużego błędu estymacji. Odczytanie PSD dla sygnału mowy i sygnału o zmiennej częstotliwości nie jest możliwe z powodu dużego błędu estymacji. Gdyby sygnały były okresowe, możliwe byłoby ustalenie częstotliwości, mocy oraz ilości poszczególnych sygnałów okresowych znajdujących się w danym sygnale. Jednak tak nie jest i określenie tych parametrów jest nie możliwe. Mimo wszystko okienkowanie ma bardzo dobry wpływ na przetwarzane próbki sygnałów. Pozwala ono na szybszą likwidację skutków przecieku niż poszerzanie okna. Nie można jednak w niektórych przypadkach tej metody zastosować, szczególnie w tych sygnałach, w których występują składowe o niewielkiej różnicy częstotliwości. Okna potrafią tą różnicę "uprościć" do jednej składowej - co może mieć w niektórych przypadkach znaczenie. Do niektórych zastosowań należy niestety użyć szerszego okna kosztem zwiększonej mocy obliczeniowej. Jedną z wad periodogramu jest nakładanie się widma (przeciek). Występuje on najwyraźniej w przypadku badania krótkich sekwencji danych. Modyfikacja periodogramu polega na zastosowaniu czasowej funkcji okna (bartlett, chebwin, hamming, kaiser, hann0, tukeywin) do próbek sygnału w celu zmniejszenia niektórych niekorzystnych właściwości

Na podstawie wykonanego zadania trzeciego można stwierdzić, że spektrogram sygnału sinusoidalnego, jest w zakresie mocy 1 kHz. Na innych wartościach moc spada, a powyżej około 500 Hz praktycznie nie ma już żadnej mocy. Spektrogram szumu, wartości mocy ma różną częstotliwość. Nie ma stałej wartości mocy na wykresie, a wartości mocy maksymalnej i mocy minimalnej występują w różnych miejscach. W spektrogramie sygnału o zmiennej częstotliwości zmienia się względem czasu, cały czas zwiększa swoją częstotliwość. Wartość największej mocy zachowuje się jak funkcja liniowa. Poza punktami z wartością maksymalną, moc jest mniejsza, choć zachowuje się tak, że tworzy pionowe pasy, w których jedne mają większą moc, drugie mniejszą. Ostatni spektrogram sygnału mowy nie przypominają żadnej funkcji. Moc maksymalna jest w miejscach, gdzie była słyszana rozmowa w nagraniu. W pozostałych miejscach jest mniejsza moc spowodowana tym, że nie było wtedy żadnego dźwięku. Kształt maksymalnej mocy jest podobny do kształtu wykresu dla sygnału mowy uzyskanego w zadaniu 4.1. Pomiędzy modułem widma krótkookresowego a widmową gęstością mocy w skali decybelowej następuje taka relacja, że dokładność odczytu widmowej gęstości jest bardzo zależna od długości odcinków. Wartości częstotliwości oraz długość odcinków są w relacji wzajemnej proporcjonalności, co oznacza, że można zmieniać zakres jednej wartości kosztem drugiej.

W czwartym zadaniu należało wykonać własny spektrogram 3D. Z powodu problemów z wykonaniem jego za pomocą języka Python, zdecydowaliśmy się na wykonanie jego w Matlabie. Jak widać poniżej wyniki są identyczne, zatem funkcja spełniła swoje zadanie. Porównując jednak wyniki z zadania trzeciego i czwartego, można przyznać, że spektrogram 2D jest bardziej czytelny. W momencie, gdy mamy bardzo zróżnicowane wartości mocy – te wysokie mogą zasłonić pozostałe wyniki. Dobrze widać to na poniższym przykładzie. W standardowym spektrogramie mamy dostęp do każdej możliwej wartości.





.

Źródła

[1] https://sound.eti.pg.gda.pl/~greg/dsp/01-AnalizaWidmowa.html
[2] https://pl.wikipedia.org/wiki/Analiza_częstotliwościowa
[3] https://sound.eti.pg.gda.pl/student/akmuz/02-Analiza.pdf
[4] http://zasoby.open.agh.edu.pl/~10swlabaj/fourier/okna.html
[5]http://www.prz.rzeszow.pl/kpe/materialy/astadler/DAQWWW/LabVIEW/Dataprocessing/Okno/Okno.pdf