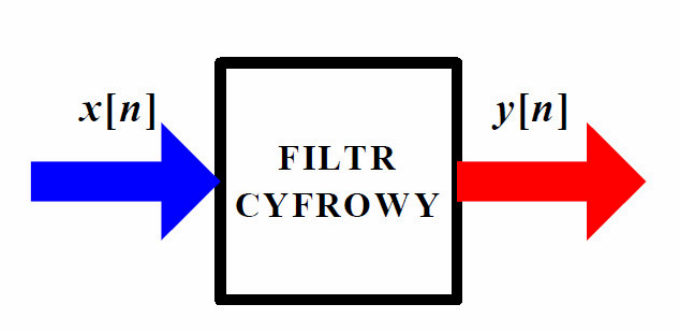
Laboratorium 5

1. Filtracja sygnałów, Filtry FIR

Filtracja jest procesem przetwarzania sygnału w dziedzinie czasu. Polega na redukowaniu (odfiltrowaniu) niepożądanych składowych zawartych w sygnale wejściowym. Filtracja to usuwanie szumu z sygnału/obrazu. Intuicyjnie szum jest stochastycznym odchyleniem sygnału od jego wartości rzeczywistej. Zadaniem filtracji jest eliminacja tego szumu głównie za pomocą metod przetwarzania sygnału/obrazu. W procesie przetwarzania filtry wykorzystywane są między innymi do:

- poprawy złej jakości technicznej sygnału,
- korekcji określonych wad sygnału,
- wzmocnienia w sygnale pewnych elementów zgodnych z posiadanym wzorcem,
- stłumienia w sygnale niepożądanego szumu,
- rekonstrukcji poszczególnych fragmentów sygnału, które uległy częściowemu uszkodzeniu.



Ze względu na typ przetwarzanych sygnałów filtrację dzieli się na:
-Analogową
(filtr analogowy działa na sygnale ciągłym, filtr = układ elektroniczny, np. RLC)
-Cyfrową
(przetwarzanie ciągu wartości próbek, filtr = program komputerowy)

Ze względu na sposób przetwarzania sygnału, filtry cyfrowe dzieli się na:
-**Filtry FIR** (Nierekursywne, o skończonej odpowiedzi impulsowej SOI, ang. Finite Impulse Response - FIR)
-**Filtry IIR** (Rekursywne, o nieskończonej odpowiedzi impulsowej NOI, ang. Infinite Impulse Response - IIR)

Charakterystyki widmowe filtrów cyfrowych:
Transmitancję filtru cyfrowego definiuje się jako stosunek transformaty zet sygnału wyjściowego do transformaty zet sygnału wejściowego.

$$H(z) = \frac{Y(z)}{X(z)} \tag{1}$$

2. Na czym polega proces filtracji?

Filtracja jest to proces przetwarzania dokonywany na sygnale w dziedzinie czasu. Proces ten powoduje zmiany w widmie sygnału oryginalnego. Zmiana polega na odfiltrowaniu pewnych niepożądanych składowych sygnału wejściowego. Można zatem powiedzieć, że filtr przepuszcza pewne częstotliwości a inne tłumi.

Sygnał ciągły jest filtrowany przez filtr analogowy. Natomiast sygnał cyfrowy jest filtrowany przez filtr cyfrowy. Filtr cyfrowy może być na przykład układem scalonym, programowalnym procesorem lub programem komputerowym. Tradycyjne liniowe filtry cyfrowe występują jako jeden z dwóch typów: filtry o skończonej odpowiedzi impulsowej SOI (ang. Finite Impulse Response - FIR) i filtry o nieskończonej odpowiedzi impulsowej NOI (ang. Infinite Impulse Response - IIR).

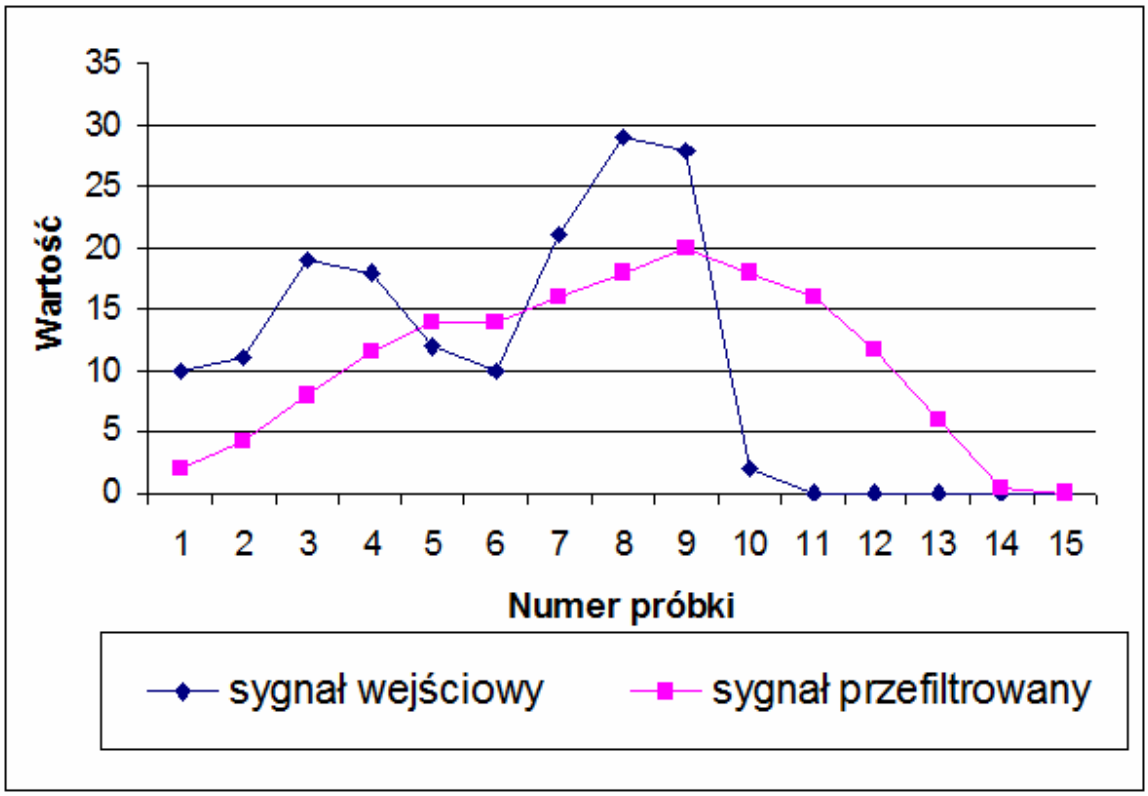
3. Filtry o skończonej odpowiedzi impulsowej (FIR)

Zasadniczą cechą charakteryzującą ten rodzaj filtrów jest to, że do uzyskania bieżącej próbki sygnału na wyjściu filtru wykorzystują one próbkę bieżącą i próbki przeszłe sygnału wejściowego, nie korzystając z żadnych przeszłych próbek sygnału wyjściowego. Z tego powodu nazywa się je czasem filtrami nierekursywnymi. Nazwa ich wzięła się stąd, że filtry te, dysponując skończoną liczbą różnych od zera próbek sygnału wejściowego, na wyjściu zawsze mają skończoną liczbę próbek sygnału wyjściowego.

Może inaczej. Jeśli na wejściu filtru FIR pojawi się nagle ciąg próbek o zerowej wartości, na wyjściu również otrzymamy ciąg, którego wartości będą równe zero. Może to wydaje się oczywiste, ale, jak się przekonamy później, wcale takie nie musi być (filtry IIR). Jednakże w przypadku KAŻDEGO filtru typu FIR powyższy warunek jest ZAWSZE spełniony.

Przykład 1
Mamy pewne wyniki obserwacji, przedstawiające się następująco:
10, 11, 19, 18, 12, 10, 21, 29, 28, 2, 0, 0, 0, 0, 0
Jeśli teraz będziemy chcieli dokonać uśrednienia tych wyników w przedziałach po 5, to pierwszy wynik pojawi nam się po odczytaniu pięciu pierwszych wyników (tyle jest nam potrzebne, aby obliczyć wartość średnią z pięciu wyników). Średnią liczymy według wzoru znanego nam (mam nadzieję) od podstawówki, czyli suma pięciu kolejnych składników podzielona przez 5. Aby policzyć kolejną średnią dodajemy pięć kolejnych składników i dzielimy je przez 5, itd. Liczby nam może niewiele mówią, ale spójrzmy na rysunek 1, na którym wykreślono zarówno poszczególne składniki, jak i średnie, poczynając od piątego składnika.

10	11	19	18	12	10	21	29	28	2	0	0	0	0	0
2	4,2	8	11,6	14	14	16	18	20	18	16	11,8	6	0,4	0



Rys.1. Sygnał wejściowy i przefiltrowany

Pierwsza rzecz to fakt, iż wykres uśrednionych wyników jest "gładszy", niż ten, który odwzorowuje przebieg poszczególnych składników "wejściowych". Na tym wszak polega idea uśredniania i nie powinno nas to dziwić. Jeśli teraz potraktujemy oba wykresy jak przebiegi czasowe pewnych sygnałów akustycznych, to należy odczytać je i wysunąć następujące wnioski:
- sygnał wejściowy jest przebiegiem zawierającym składowe o dużych częstotliwościach (wynika to z gwałtownych zmian w ciągu czasowym sygnału)
- z sygnału uśrednionego wyeliminowano te gwałtowne zmiany (został wygładzony), a to oznacza, że część składowych o dużych częstotliwościach została odfiltrowana
- dodatkowo możemy zauważyć, że jeśli sygnał wejściowy zmaleje do zera, to sygnał na wyjściu będzie dążył do zera i po 5 próbkach (tyle ile wynosi przedział uśredniania) również spadnie do zera.

Tak więc do obliczenia wartości wyjściowych pobraliśmy tylko wartości wejściowe, nie używaliśmy żadnej z przeszłych wartości wyjściowych układu uśredniającego, a poza tym wartość sygnału na wyjściu po tym, jak sygnał wejściowy będzie miał wartość zero, również osiągnie wartość zero. Można więc śmiało stwierdzić, że układ nasz jest filtrem FIR, który jest filtrem 5 rzędu i którego wagi (współczynniki) będą wszystkie takie same i wynosiły 1/5. Skąd te dwa ostatnie wnioski?

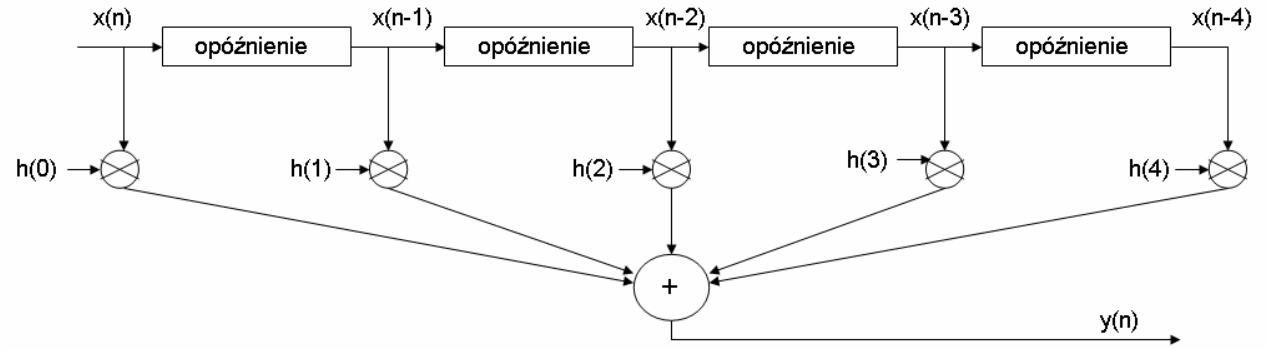
4. Struktura filtru FIR

Przy uśrednianiu dodajemy pięć wartości i dzielimy sumę przez pięć, uzyskując wynik. Równie dobrze możemy pomnożyć każdą z wartości wejściowych przez 1/5 i następnie dokonać sumowania - obie metody są równoważne - wtedy wzór na obliczenie próbek na wyjściu układu będzie wyglądał:

$$y(n) = \sum_{k=n-4}^{n} \frac{1}{5} x(k) \tag{2}$$

x(k) to kolejne próbki wejściowe sygnału, natomiast nasz współczynnik ma stałą wartość dla wszystkich próbek i wynosi 1/5

W ogólnym przypadku filtrów wcale nie musi być (i przeważnie nie jest) tak, że wszystkie współczynniki mają tę samą wartość. Struktura filtru FIR 5. rzędu przedstawiona jest na rysunku 2. Kolejne próbki wejściowe oznaczono jako x, poczynając od próbki pierwszej oznaczonej $x(0)$, następna $x(1)$ i tak dalej do $x(n)$. Współczynniki filtru oznaczamy podobnie, literką h, poczynając od pierwszego ($h(0)$) i dalej do $h(n)$.



Rys.2. 5-ogniwowy dolnoprzepustowy filtr FIR

Zauważmy, że współczynnik pierwszy filtru ($h(0)$) mnożymy przez próbkę $x(n)$ (jeśli n=4 to przez $x(4)$), następny współczynnik $h(1)$ przez $x(n\text{-}1)$, czyli $x(3)$, aż w końcu ostatni współczynnik $h(n)$, czyli w naszym przypadku $h(4)$, mnożymy przez próbkę $x(n\text{-}4)$, czyli $x(0)$. Uporządkowanie próbek wejściowych względem czasu zostało odwrócone, co ma swoje odzwierciedlenie w ogólnym wzorze dla filtrów FIR, który wygląda następująco:

$$y(n) = \sum_{k=0}^{M} h(k)x(n-k) \tag{3}$$

Równanie to jest równaniem splotu, które symbolicznie oznaczane jest gwiazdką między czynnikami, czyli

$$y(n) = h(n) * x(n) \tag{4}$$

Wbrew pozorom nie jest to trudna operacja. Odwracamy porządek czasowy ciągu próbek wejściowych i rozpoczynamy przemnażanie odwróconego ciągu przez współczynniki filtru.

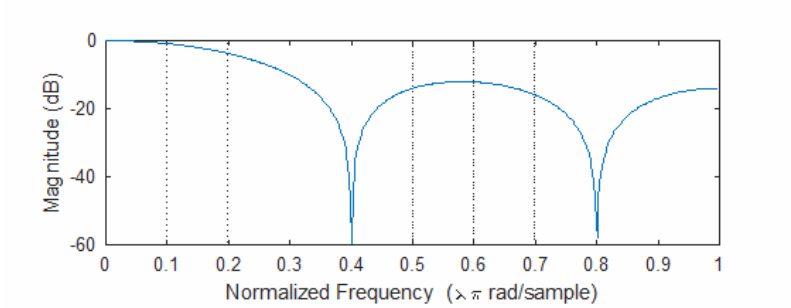
Odpowiedź impulsowa filtru FIR jest identyczna, jak wartość współczynników filtru (jeśli w naszym przypadku przemnożymy kolejne próbki przez współczynniki naszego filtru wynoszące 1/5, to otrzymamy pięć kolejno po sobie następujących próbek o amplitudzie 1/5. Dlatego pojęcia "współczynniki filtru FIR" i "odpowiedź impulsowa" są synonimami.

```
X=[10 11 19 18 12 10 21 29 28 2 0 0 0 0 0];
b = 1;
a = [0.2, 0.2, 0.2, 0.2, 0.2];
y = filter(a,b,X);
%wynik
y=  2.0000    4.2000    8.0000   11.6000   14.0000
   14.0000   16.0000   18.0000   20.0000   18.0000
   16.0000   11.8000    6.0000    0.4000         0
```

5. Charakterystyka częstotliwościowa filtru FIR

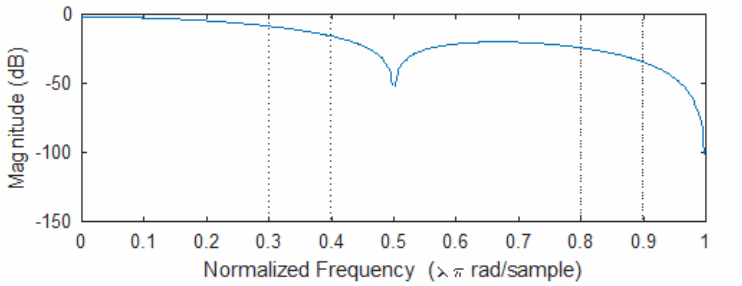
Aby otrzymać taką charakterystykę wystarczy wyznaczyć kształt odpowiedzi impulsowej w dziedzinie częstotliwości. A tego z kolei dokonamy, jeśli obliczymy DFT (a w zasadzie FFT) odpowiedzi impulsowej, czyli de facto DFT współczynników filtru. W przypadku filtru o pięciu współczynnikach równych 1/5 (czyli 0,2) kształt charakterystyki będzie wyglądał tak, jak na rysunku 3.

```
a = [0.2, 0.2, 0.2, 0.2, 0.2];
freqz(a,1)
```



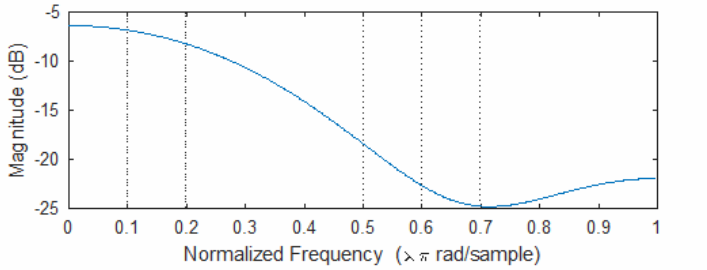
Rys.3. Widmo częstotliwości filtru o współczynnikach `a=[0.2, 0.2, 0.2, 0.2, 0.2]`

```
a = [0.1, 0.2, 0.2, 0.2, 0.1];
freqz(a,1)
```



Rys.4. Widmo częstotliwości filtru o współczynnikach `a=[0.1, 0.2, 0.2, 0.2, 0.1]`

```
a = [0.04, 0.1, 0.2, 0.1, 0.04];
freqz(a,1)
```



Rys.5. Widmo częstotliwości filtru o współczynnikach `a=[0.04, 0.1, 0.2, 0.1,0.04]`

Jak widać, nagła zmiana wartości współczynników z 0 do 0,2 powoduje powstawanie listków bocznych (te inne "krągłości" poza listkiem głównym). Sprawdźmy jak zachowywać się będzie filtr FIR o współczynnikach jak na rysunku 4 (0.1, 0.2, 0.2, 0.2, 0.1). Zauważamy spadek zafalowań okupionych jednakże poszerzeniem listka głównego, co oznacza poszerzenie pasma przejściowego, a więc zmniejszenie nachylenia. W trzecim przypadku (rysunek 5 - współczynniki 0.04, 0.1, 0.2, 0.1, 0.04) pozbyliśmy się prawie całkowicie zafalowań kosztem jeszcze większego poszerzenia listka głównego.

Mając bowiem charakterystykę częstotliwościową filtru w prosty sposób możemy wyznaczyć jego odpowiedź impulsową, czyli współczynniki filtru. A o to właśnie nam chodzi, bo przecież filtr cyfrowy FIR to nic innego, jak pewna ilość współczynników (ilość zależy od rzędu filtru), przez które będziemy przemnażać kolejne (pamiętamy, że w odwróconej kolejności) próbki, aby na wyjściu otrzymać cyfrowy sygnał odfiltrowany. Może to nie wydaje się na pierwszy rzut oka oczywiste, ale trzeba pamiętać, że mamy do czynienia z sygnałami cyfrowymi, czyli ciągami próbek, a więc ciągami liczb.

Charakterystyka częstotliwościowa filtru powstaje poprzez poddanie odpowiedzi impulsowej filtru (współczynników) DFT, aby więc z charakterystyki częstotliwościowej otrzymać odpowiedź impulsową, a więc i współczynniki, musimy obliczyć odwrotne dyskretne przekształcenie Fouriera (IDFT) z charakterystyki częstotliwościowej.

Problem w tym, że jest on nieprzyczynowy (ma wartości ujemne, a więc sygnał był na wyjściu zanim pojawiło się pobudzenie na wejściu - jest to praktycznie nie do zrealizowania), a poza tym jest nieskończony.

Z nieprzyczynowością możemy sobie dość prosto poradzić, przesuwając przebieg w stronę dodatnich wartości; to jedna z własności DFT - nie zmienia to w żaden sposób amplitudowej charakterystyki częstotliwościowej naszego filtru FIR, a jedynie dokonuje liniowego przesunięcia fazy, ale to nie jest tak istotne. Gorzej z ilością współczynników - nie możemy wziąć ich nieskończonej liczby, z oczywistych względów. Możemy więc spróbować okroić nieco ich liczbę, i to dość znacznie, bo przy bardzo dużej ich liczbie będziemy potrzebowali bardzo szybkiego procesora, aby dokonywał tysięcy mnożeń i dodawań.

Jeśli weźmiemy nieco ich więcej, 19, zbocza filtru stają się bardziej strome, jednak pojawiają się zafalowania w strefie przenoszenia.

Zobaczmy jeszcze, jak będzie wyglądał nasz filtr w przypadku, kiedy zastosujemy 31 współczynników. Jest znacznie lepiej ze stromością, ale zafalowania pozostały, i to na takim samym poziomie, niestety. Widzimy więc, że możemy wpływać na kształt filtru, a dokładnie na stromość jego charakterystyki przejściowej, zwiększając liczbę ogniw (współczynników filtru), co oczywiście znacząco spowolni nam prace takiego filtru. Niestety, nawet zwiększanie liczby ogniw nie uwolni nas od zafalowań.

6. Okna

Co prawda nie pozbędziemy się zafalowań całkowicie, jak i nigdy nie osiągniemy filtrów o zboczach idealnie pionowych, ale możemy je minimalizować. Pamiętcie może, jak poradziliśmy sobie ze stratą zafalowań w DFT? Oczywiście **okna**.

Widzimy więc, że cyfrowe przetwarzanie sygnałów rządzi się pewnymi żelaznymi prawami, które obowiązują wszędzie: w DFT, w FFT i także w filtracji cyfrowej. Są to dwie generalne zasady:
- zwiększając ilość próbek (współczynników) polepszamy rozdzielczość (stromość zboczy)
- stosując okna zmniejszamy zafalowania charakterystyki kosztem rozdzielczości (zmniejszenie nachylenia).

7. Faza filtrów FIR

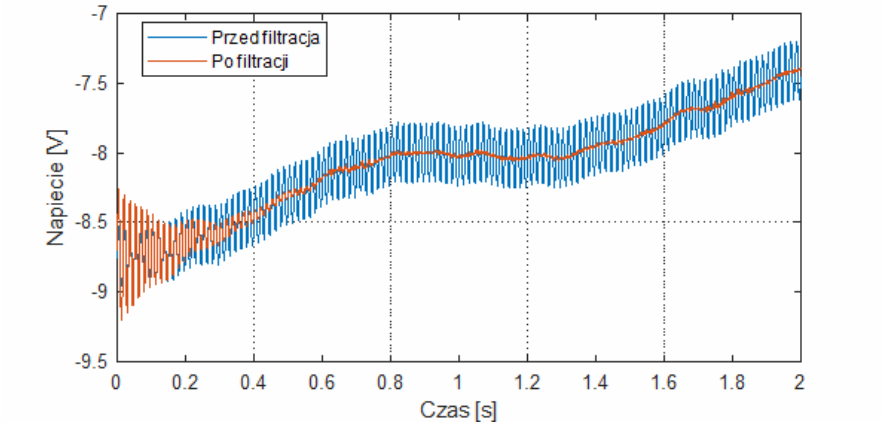
Na zakończenie tematu o filtrach FIR kilka zdań o fazie, gdyż, jak się okazuje, jest to bardzo ważna właściwość tych filtrów. A mianowicie filtry FIR, mające symetryczne współczynniki, oferują liniową charakterystykę fazy w paśmie przenoszenia, dzięki czemu to właśnie te filtry są chętniej wybierane, niż filtry IIR. Czytelnik mógłby się dziwić, cóż tak istotnego jest w tym fakcie. Ano jest, gdyż ma to wpływ na tzw. opóźnienia grupowe.

Opóźnienie grupowe w filtrach FIR jest stałe (gdyż faza jest liniowa), co oznacza, że wszystkie składowe częstotliwościowe sygnału wejściowego są jednakowo opóźniane, a to z kolei oznacza, że nie ma żadnych zniekształceń fazowych w sygnale wyjściowym. Ma to kapitalne znaczenie, gdyż w przypadku sygnałów cyfrowych zniekształcenia fazowe są o

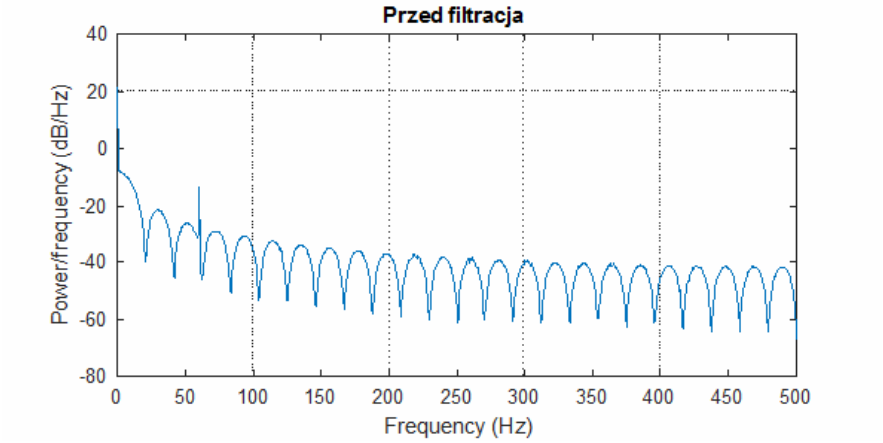
wiele bardziej "nieprzyjemne", niż zniekształcenia amplitudowe, a więc jest to bardzo istotna zaleta filtrów FIR.

Przykład 2

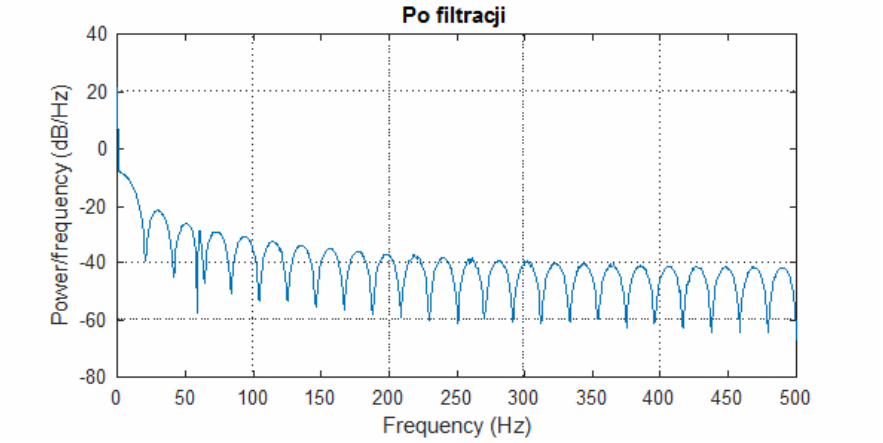
Mając sygnał napięcia (openloop60hertz), który jest zakłócony na częstotliwości 60 Hz (Rys. 6-9). Zaprojektować filtr usuwający szum. Wykonać filtrację i wyrysować sygnał wolny od szumu.



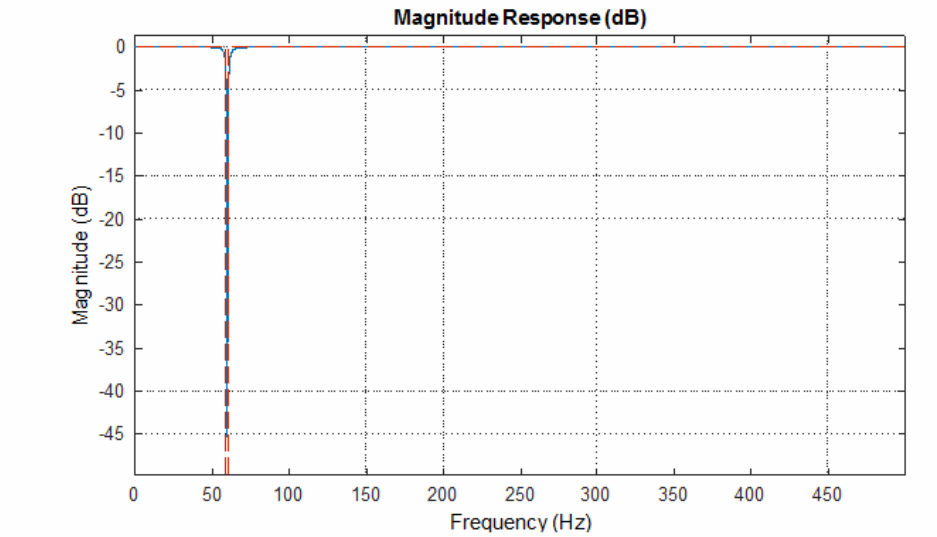
Rys. 6. Sygnał w dziedzinie czasu przed filtracją i po filtracji



Rys. 7. Sygnał w dziedzinie częstotliwości przed filtracją



Rys. 8. Sygnał w dziedzinie częstotliwości po filtracji



Rys. 9. Zaprojektowany filtr w dziedzinie częstotliwości

Zapoznać się z kodem poniżej, wpisać do Matlaba, zaobserwować wyniki.

```
% wczytywanie pliku

figure
load openloop60hertz, openLoop=openLoopVoltage;
Fs=1000;
t=(0:length(openLoop)-1)/Fs;
plot(t, openLoop); box on; grid on;
ylabel 'Voltage [V]', xlabel 'Time [s]'
title 'Open-Loop Voltage with Noise'

%% Power spectrum - Moc
%% na ok 60 Hz widzimy zaklocenie
```

```
figure;
periodogram(openLoop, [], [], Fs);

%% Projektujemy filtr

filtCoeff= designfilt('bandstopiir', 'FilterOrder', 2,...
'HalfPowerFrequency1', 59, 'HalfPowerFrequency2', 61, ...
'SampleRate', Fs);

%% Wykres filtru

fvtool(filtCoeff)

noiseFreeSignal= filter(filtCoeff, openLoop);

%% sprawadzamy wynik w dziedzinie czasu

close all;
figure;

plot (t, openLoop, t, noiseFreeSignal); grid on;
legend('Przed filtracja', 'Po filtracji');
ylabel 'Napiecie [V]', xlabel 'Czas [s]'

%% sprawdzamy wynik w dziedzinie częstotliwości

figure;
periodogram(noiseFreeSignal, [], [], Fs);
title('Po filtracji ');

figure;
periodogram(openLoop, [], [], Fs);
title('Przed filtracja ');
```

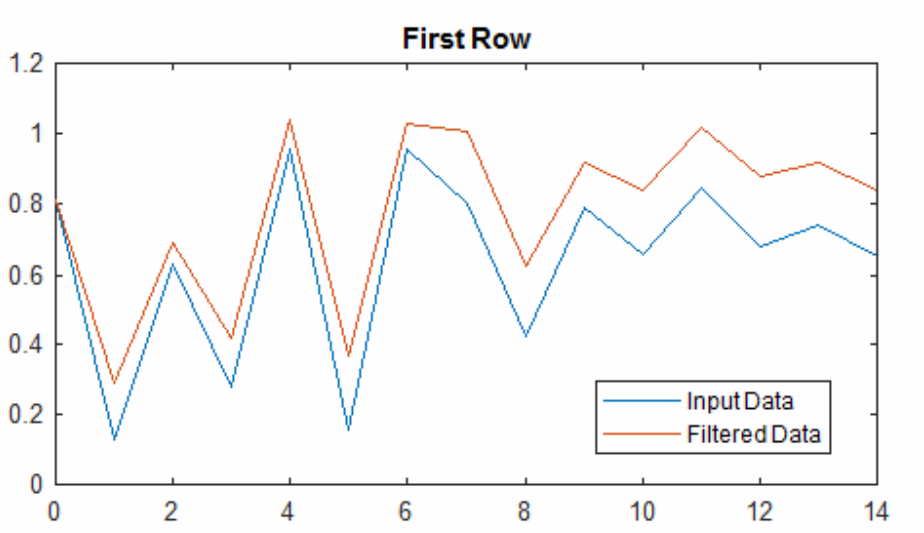
Przykład 3
Zaprojektować filtr o transmitancji

$$H(z) = \frac{b(1)}{a(1) + a(2)z^{-1}} = \frac{1}{1 - 0.2z^{-1}} \tag{5}$$

dla ciągu losowego `x = rand(2,15)`
Porównać wyniki na rysunku 10.

```
rng default  %losowe liczby
x = rand(2,15);

b = 1;
```

```
a = [1 -0.2];

y = filter(b,a,x,[],2);

t = 0:length(x)-1;  %indeks wektora

plot(t,x(1,:))
hold on
plot(t,y(1,:))
legend('Input Data','Filtered Data')
title('First Row')
```



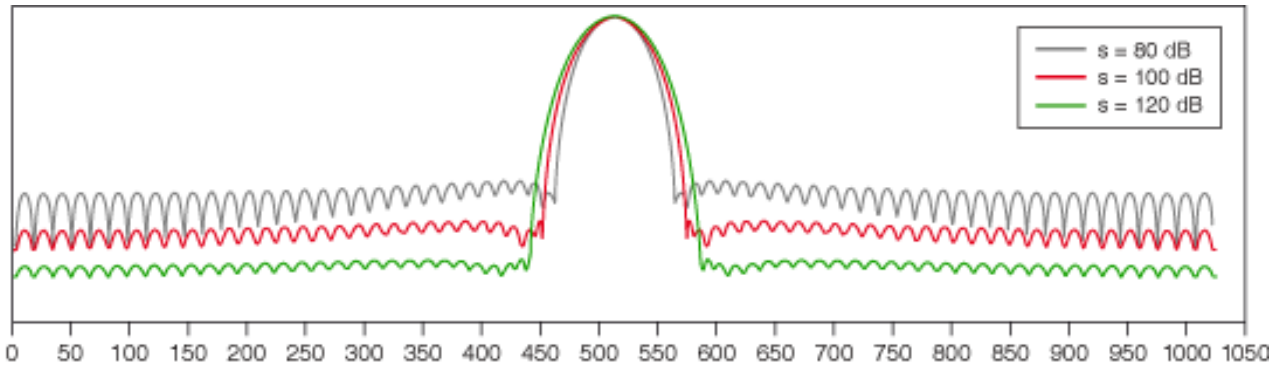
Rys. 10. Sygnał w dziedzinie czasu przed filtracją i po filtracji

Przykład 4

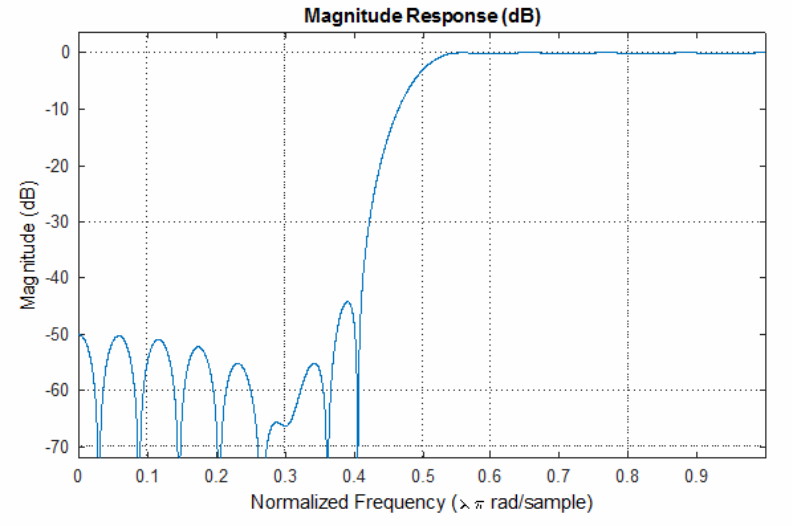
Proszę przeanalizować działanie poniższego kodu. Wczytujemy sygnał `chirp`. Następnie wykonujemy FFT aby sprawdzić jak wygląda owy sygnał w dziedzinie częstotliwości. Projektujemy filtr górnoprzepustowy. Filtr odcina niskie częstotliwości w 0,48. Natomiast okno Czebyszewa (Rys. 11, 12) ustawione jest na 30 dB.

```
bhi = fir1(34,0.48,'high',chebwin(35,30));
```

i sprawdzamy jego działanie. (parametry można sprawdzić `help fir1`)



Rys. 11. Okno Czebyszewa



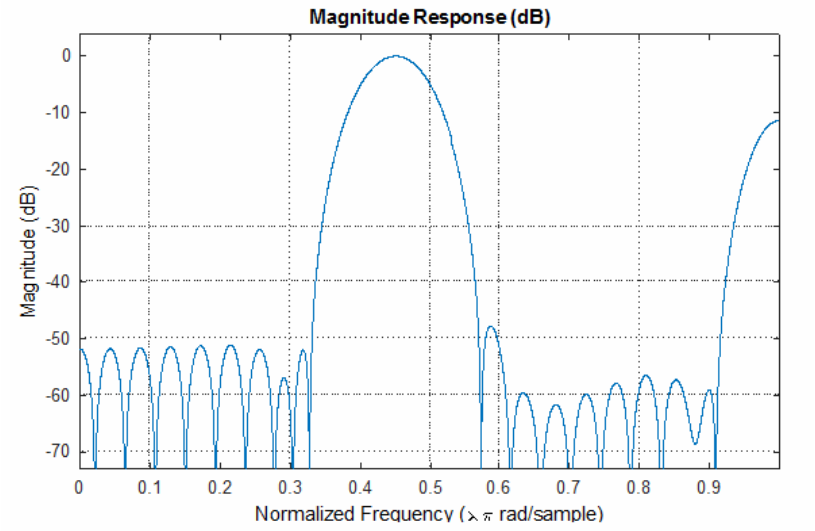
Rys. 12. Zaprojektowany filtr

```
load chirp
t = (0:length(y)-1)/Fs;   % 1.6 sekundy

xfft=abs(fft(y));
xfft=xfft/13129;
x1=1:1:6564;
bar(x1(1:6564), xfft(1:6564));
axis([0,6564, 0,0.01]) ;

bhi = fir1(34,0.48,'high',chebwin(35,30));
freqz(bhi,1)
fvtool(bhi)
outhi = filter(bhi,1,y);

xfft=abs(fft(outhi));
xfft=xfft/13129;
x1=1:1:6564;
```

```
bar(x1(1:6564), xfft(1:6564));
axis([0,6564, 0,0.01]) ;


subplot(2,1,1)
plot(t,y)
title('Original Signal')
ys = ylim;

subplot(2,1,2)
plot(t,outhi)
title('Highpass Filtered Signal')
xlabel('Time (s)')
ylim(ys)
```

Przykład 5

Proszę przeanalizować działanie poniższego kodu. Wczytujemy sygnał `chirp`. Projektujemy filtr pasmowoprzepustowy `bM = fir1(ord,[low bnd])`. Filtr odcina niskie częstotliwości w 0,4. Przepuszcza w 0,5 i 0,99 i sprawdzamy jego działanie (Rys. 13).



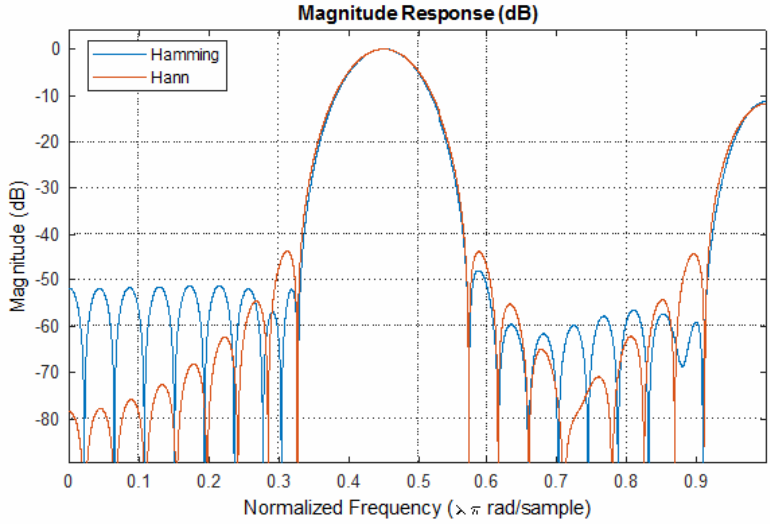
Rys. 13. Zaprojektowany filtr

```
load chirp
t = (0:length(y)-1)/Fs;   % 1.6 sekundy

ord = 46;
low = 0.4;
bnd = [0.5 0.99];
bM = fir1(ord,[low bnd]);
fvtool(bM)
```

```
outF = filter(bM,1,y);

xfft=abs(fft(outF));
xfft=xfft/13129;
x1=1:1:6564;
bar(x1(1:6564), xfft(1:6564));
axis([0,6564, 0,0.01]) ;
```

Przykład 6

Proszę przeanalizować działanie poniższego kodu. Wczytujemy sygnał chirp. Projektujemy filtr pasmowoprzepustowy z oknem Hanna – `hM = fir1(ord,[low bnd],'DC-0',hann(ord+1))`. Filtr odcina niskie częstotliwości w 0,4. Przepuszcza w 0,5 i 0,99 i sprawdzamy jego działanie (Rys. 14).



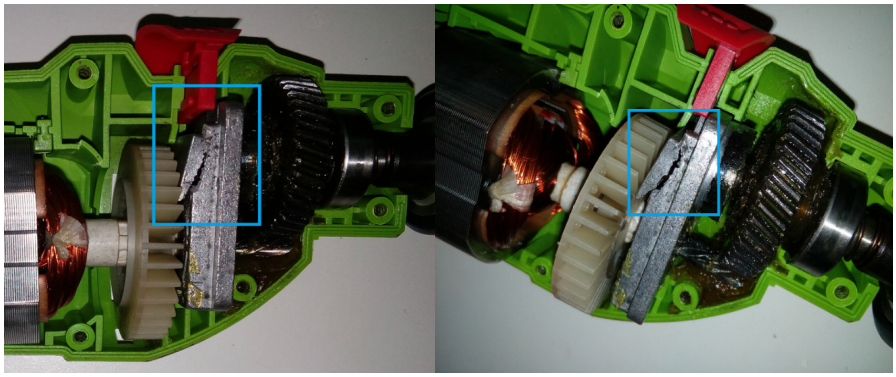
Rys. 14. Porównanie okien Hamminga i Hanna

```
load chirp
t = (0:length(y)-1)/Fs;   % 1.6 sekundy

hM = fir1(ord,[low bnd],'DC-0',hann(ord+1));

hfvt = fvtool(bM,1,hM,1);      %porownanie okien
legend(hfvt,'Hamming','Hann')

outhann = filter(hM,1,y);
```

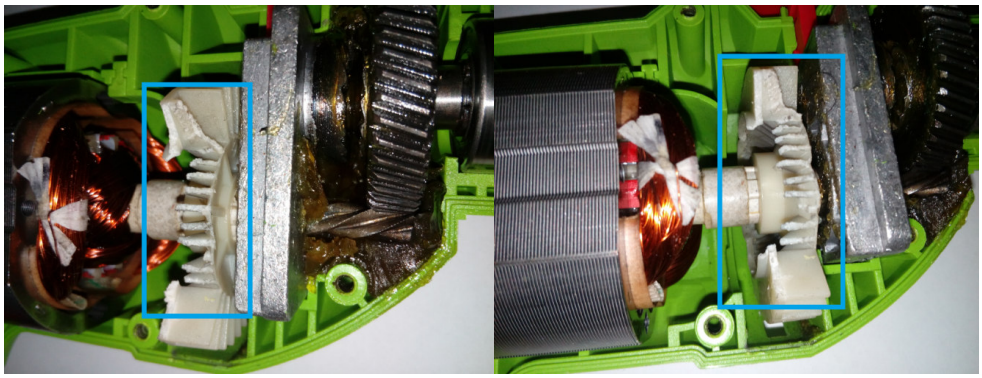
```
xfft=abs(fft(outhann));
xfft=xfft/13129;
x1=1:1:6564;
bar(x1(1:6564), xfft(1:6564));
axis([0,6564, 0,0.01]) ;
```

Przykład 7

Dane są 2 sygnały akustyczne (wiertarka z uszkodzoną przekładnią (Rys. 15) i wiertarka z uszkodzonym wentylatorem (Rys. 16). Każdy sygnał akustyczny ma 4 próbki (w sumie 8 próbek).

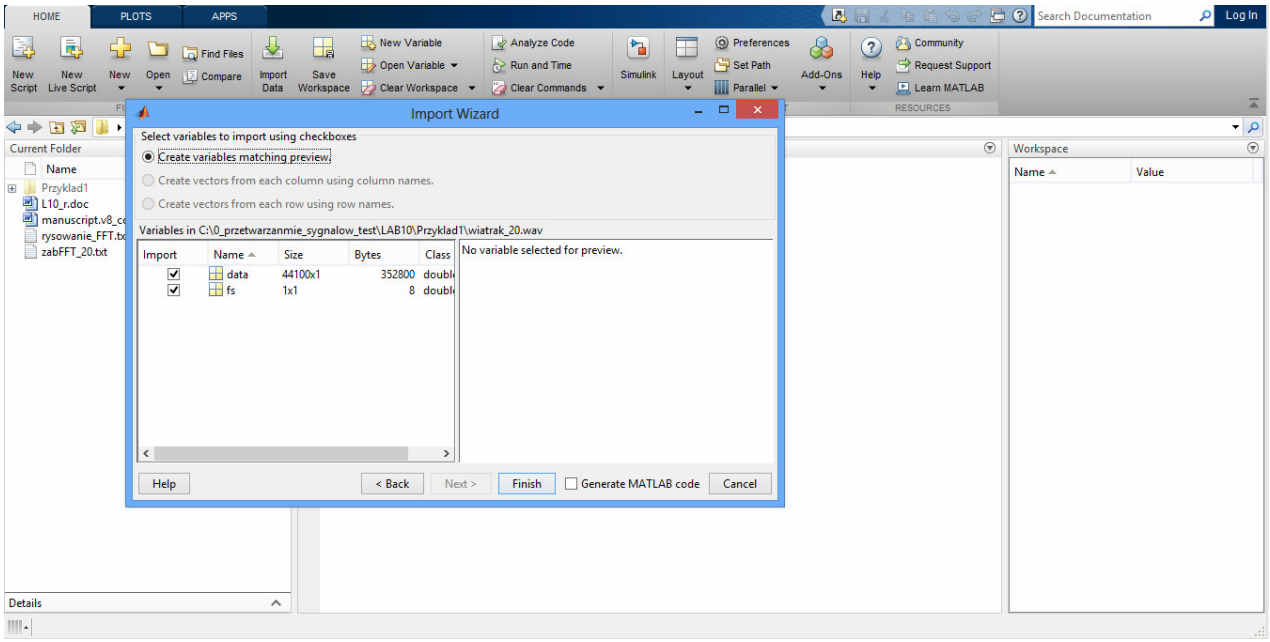


Rys. 15. Wiertarka z uszkodzoną przekładnią



Rys. 16. Wiertarka z uszkodzonym wentylatorem

Proszę wczytać plik *wiatrak_20.wav*
Proszę kliknąć przycisk *Import->wiatrak_20.wav* (Rys. 17)



Rys. 17. Import z pliku do Matlaba

Proszę wpisać w konsoli Matlaba.

```
x2=data;
% normalizacja do przedzialu [-1, 1]
% jesli sygnaly sa mierzone w roznych odlegloscіach
max_data=max(abs(data));
data=data/max_data;
xfft=abs(fft(x2));
xfft=xfft/44100;
```

Następnie wyrysowujemy wykresy FFT. Proszę wpisać.

```
x1=1:1:44100;
bar(x1(1:44100), xfft(1:44100));

% wyrysowalo 2 razy zatem wyrysujemy 22050

x1=1:1:22050;
bar(x1(1:22050), xfft(1:22050));

% dodajmy osie i etykiety osi

axis([0,1000, 0,0.02]) ;

xlabel('Skladowa czestotliwości [Hz]');
ylabel('Znormalizowania amplituda skladowej czestotliwosci');
```

Zapisujemy plik TXT. Proszę wpisać.

```
%zapisywanie do pliku FFT_wiatrak_20.txt

fid = fopen('FFT_wiatrak_20.txt','w+t','n');
fprintf(fid,'%f\n',xfft(1:22050));
fclose(fid)
```

Dla pozostałych próbek WAV wygenerować odpowiednio pliki (do zrobienia)

```
FFT_wiatrak_21.txt
FFT_wiatrak_23.txt
FFT_wiatrak_24.txt

FFT_przekladnia_20.txt
FFT_przekladnia_21.txt
FFT_przekladnia_23.txt
FFT_przekladnia_24.txt
```

Następnie przeprowadzić rozpoznawanie z zastosowaniem klasyfikatora KNN (k-najbliższych sąsiadów)
Próbki następujące będą wzorcami: `FFT_wiatrak_20.txt, FFT_wiatrak_21.txt, FFT_przekladnia_20.txt, FFT_przekladnia_21.txt`

Próbki następujące będą testowe: `FFT_wiatrak_23.txt, FFT_wiatrak_24.txt, FFT_przekladnia_23.txt, FFT_przekladnia_24.txt`

Proszę wczytać w Matlabie pliki TXT.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% wczytujemy probki

load FFT_wiatrak_20.txt
load FFT_wiatrak_21.txt
load FFT_przekladnia_20.txt
load FFT_przekladnia_21.txt


load FFT_wiatrak_23.txt
load FFT_wiatrak_24.txt
load FFT_przekladnia_23.txt
load FFT_przekladnia_24.txt
```

Klasyfikator K-NN (metryka Mahattana - wzór (5)) linijka po linijce. Proszę wpisać.

$$Odleglosc(\mathbf{a}-\mathbf{hd}) = \sum_{i=1}^{n} |(a_i - hd_i)| \quad (6)$$

gdzie, **a** i **hd** – wektory cech, *Odleglosc*(**a**-**hd**) – obliczona odległość, *n* – liczba elementów wektora.

```
D=sum(abs(FFT_wiatrak_23-FFT_wiatrak_20))
D=sum(abs(FFT_wiatrak_23-FFT_wiatrak_21))
D=sum(abs(FFT_wiatrak_23-FFT_przekladnia_20))
D=sum(abs(FFT_wiatrak_23-FFT_przekladnia_21))

D=sum(abs(FFT_wiatrak_24-FFT_wiatrak_20))
D=sum(abs(FFT_wiatrak_24-FFT_wiatrak_21))
D=sum(abs(FFT_wiatrak_24-FFT_przekladnia_20))
D=sum(abs(FFT_wiatrak_24-FFT_przekladnia_21))


D=sum(abs(FFT_przekladnia_23-FFT_wiatrak_20))
D=sum(abs(FFT_przekladnia_23-FFT_wiatrak_21))
D=sum(abs(FFT_przekladnia_23-FFT_przekladnia_20))
D=sum(abs(FFT_przekladnia_23-FFT_przekladnia_21))

D=sum(abs(FFT_przekladnia_24-FFT_wiatrak_20))
D=sum(abs(FFT_przekladnia_24-FFT_wiatrak_21))
D=sum(abs(FFT_przekladnia_24-FFT_przekladnia_20))
D=sum(abs(FFT_przekladnia_24-FFT_przekladnia_21))



D=sum(abs(FFT_wiatrak_23-FFT_wiatrak_20))
```

```
D=sum(abs(FFT_wiatrak_23-FFT_wiatrak_21))

D=sum(abs(FFT_wiatrak_23-FFT_przekladnia_20))

D=sum(abs(FFT_wiatrak_23-FFT_przekladnia_21))
D =

   11.3832

D =

   11.2326      %wynik rozpoznawania, najmniejsza odleglosc

D =

   12.5240

D =

   12.4945
```

Przykład 8

Korzystając z Przykładu 7, przeanalizować składowe częstotliwości 1-1000 i przeprowadzić rozpoznawanie, analogiczne do tego z Przykładu 7.

Proszę wczytać plik `wiatrak_20.wav`
Proszę kliknąć przycisk `Import->wiatrak_20.wav`

Proszę wpisać w konsoli Matlaba.

```
x2=data;
% normalizacja do przedzialu [-1, 1] jesli sygnaly sa mierzone w
roznych odlegloscіach
max_data=max(abs(data));
data=data/max_data;
xfft=abs(fft(x2));
xfft=xfft/44100;

xfft(1001:44100)=0;

%zapisywanie do pliku FFT_filtracja_wiatrak20.txt
fid = fopen('FFT_filtracja_wiatrak20.txt','w+t','n');
fprintf(fid,'%f\n',xfft(1:22050));
fclose(fid)
```

Proszę wczytać w Matlabie pliki TXT.

```
% wczytujemy probki

load FFT_filtracja_wiatrak20.txt
load FFT_filtracja_wiatrak21.txt
load FFT_filtracja_przekladnia20.txt
load FFT_filtracja_przekladnia21.txt

load FFT_filtracja_wiatrak23.txt
load FFT_filtracja_wiatrak24.txt
load FFT_filtracja_przekladnia23.txt
load FFT_filtracja_przekladnia24.txt

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_wiatrak20))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_wiatrak21))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_przekladnia20))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_przekladnia21))


D=sum(abs(FFT_filtracja_wiatrak24-FFT_filtracja_wiatrak20))

D=sum(abs(FFT_filtracja_wiatrak24-FFT_filtracja_wiatrak21))

D=sum(abs(FFT_filtracja_wiatrak24-FFT_filtracja_przekladnia20))

D=sum(abs(FFT_filtracja_wiatrak24-FFT_filtracja_przekladnia21))


D=sum(abs(FFT_filtracja_przekladnia23-FFT_filtracja_wiatrak20))

D=sum(abs(FFT_filtracja_przekladnia23-FFT_filtracja_wiatrak21))

D=sum(abs(FFT_filtracja_przekladnia23-FFT_filtracja_przekladnia20))

D=sum(abs(FFT_filtracja_przekladnia23-FFT_filtracja_przekladnia21))


D=sum(abs(FFT_filtracja_przekladnia24-FFT_filtracja_wiatrak20))

D=sum(abs(FFT_filtracja_przekladnia24-FFT_filtracja_wiatrak21))

D=sum(abs(FFT_filtracja_przekladnia24-FFT_filtracja_przekladnia20))
```

```
D=sum(abs(FFT_filtracja_przekladnia24-FFT_filtracja_przekladnia21))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_wiatrak20))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_wiatrak21))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_przekladnia20))

D=sum(abs(FFT_filtracja_wiatrak23-FFT_filtracja_przekladnia21))

D =
    0.6867            %wynik rozpoznawania, najmniejsza odleglosc

D =
    0.7195

D =
    0.7703

D =
    0.7607
```

Zadania do wykonania

W sprawozdaniu powinny znaleźć się:
1) Informacje na temat filtracji sygnałów.
2) Informacje na temat w jaki sposób projektujemy filtry FIR.
3) Wykonane zadania - skrypty w m.plikach oraz otrzymane wykresy.
4) Wnioski z przeprowadzonych zadań.

Zad. 1
Korzystając z Przykładu 2 zaprojektować filtr wycinający składowe częstotliwości od 50-70 Hz.

Zad 2
Zaprojektować filtr o transmitancji (6):

$$H(z) = \frac{b(1)}{a(1) + a(2)z^{-1}} = \frac{1}{1 - 0.5z^{-1}} \tag{7}$$

dla ciągu `x=[1, 2, 3, 4, 5, 4, 3, 4, 5, 6, 7]`

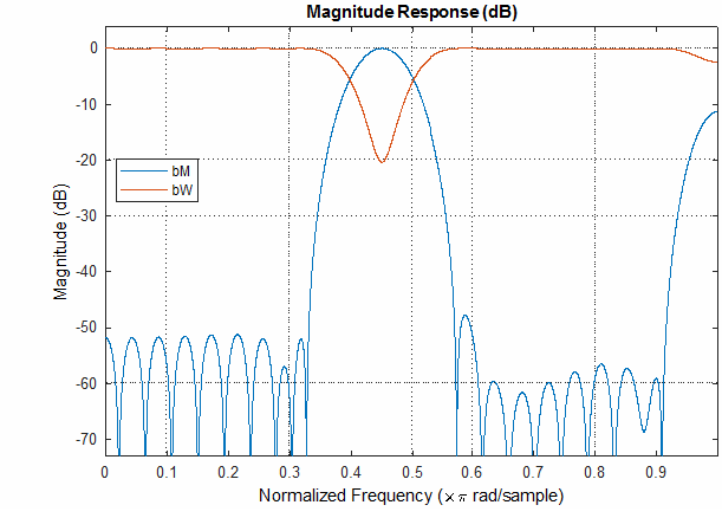
Zad 3
Zaprojektować filtr o transmitancji (7):

$$H(z) = \frac{b(1) + b(2)z^{-1}}{a(1) + a(2)z^{-1}} = \frac{2 + 3z^{-1}}{1 + 0.2z^{-1}} \tag{8}$$

dla ciągu `x=[1, 2, 3, 4, 5, 4, 3, 4, 5, 6, 7]`

Zad 4
Na podstawie przykładu 4 zaprojektować filtr dolnoprzepustowy. Wczytać sygnał chirp. Filtr odcina wysokie częstotliwości w 0,48. Pozostałe parametry jak okno Czebyszewa ustawić na 30 dB i sprawdzić jego działanie.

Zad 5
Na podstawie przykładu 5 zaprojektować filtr, który będzie przepuszczać częstotliwości tłumione. Natomiast tłumić będzie te przepuszczane (Rys. 18).



Rys 18. Porównanie filtrów

Wskazówka:
Do funkcji `fir1()`, użyć parametru `'DC-1'`.

Zad 6
Na podstawie przykładu 6 zaprojektować filtr, który używa okna Tukey (`'Tukey'`). Porównać filtry z oknem Hamminga (`'Hamming'`) i Tukey.

Zad 7
Korzystając z Przykładu 7, przeanalizować składowe częstotliwości 500-1000 i przeprowadzić rozpoznawanie, analogiczne do tego z Przykładu 8. Zaobserwować wyniki.

Pytania
1) Co to jest filtracja sygnałów i po co ją stosujemy?
2) Co to jest filtr FIR i czym się charakteryzuje?
3) W jaki sposób projektujemy filtry FIR?
4) Do czego służą okna?