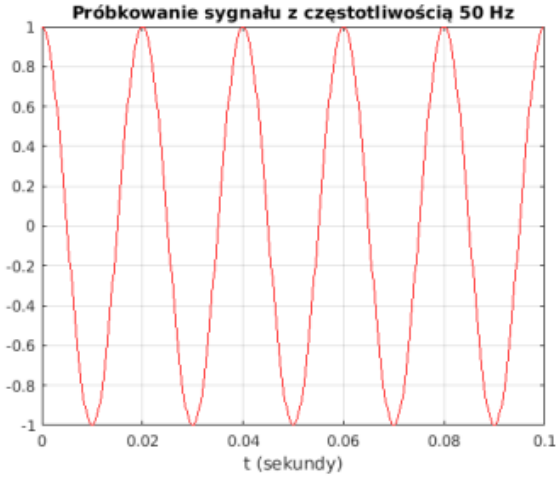
Labolatorium PTI – ćwiczenie 1, przykład 4

Pomiar sygnałów

Akwizycja i poprawa jakości sygnałów

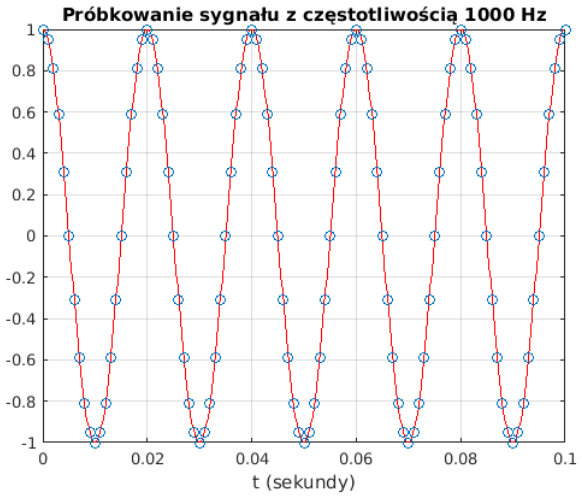
II. BADANIE ZJAWISKA ALIASINGU

Naszą pracę rozpoczniemy od symulowanego wygenerowania sinuisoidalnego sygnału o częstotliwości 50 Hz.



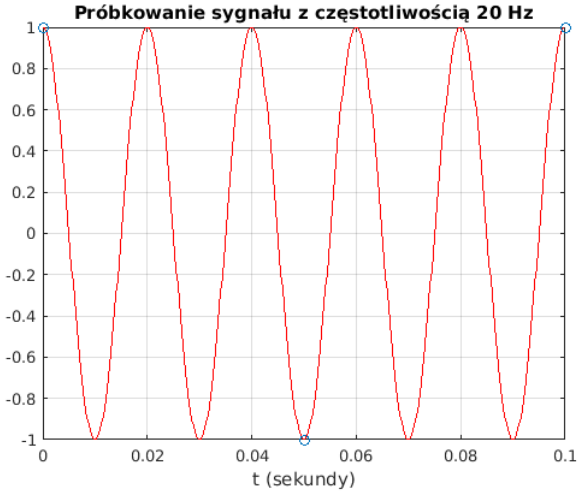
Rysunek 1: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz.

Mając nasz sygnał testowy spróbujmy zpróbkować go z częstotliwością 1000 Hz:



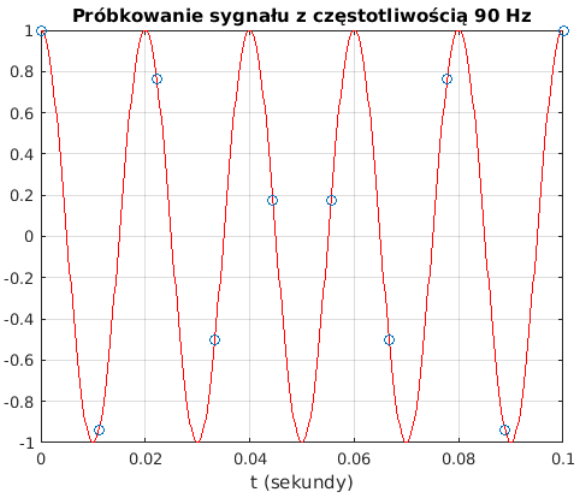
Rysunek 2: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz. Częstotliwość próbkowania 1000 Hz.

Widzimy, że ta częstotliwość jest o wiele za wysoka jak dla naszych potrzeb. Dlatego dla porównania przyjrzyjmy się temu samemu sygnałowi zpróbkowanemu z częstotliwością 20 Hz:



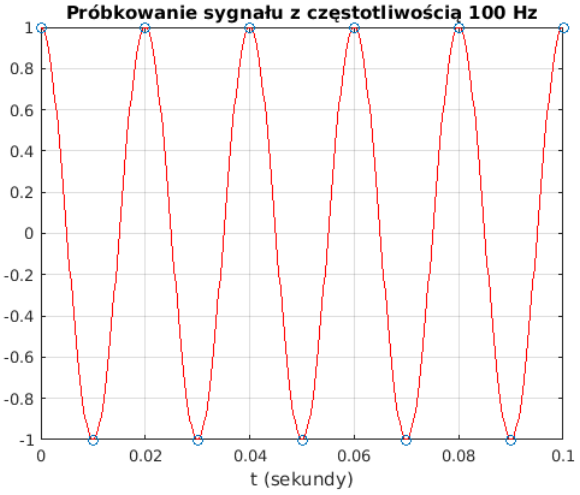
Rysunek 3: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz. Częstotliwość próbkowania 20 Hz.

Częstotliwość jest znacznie poniżej częstotliwości Nyquista tak samo jak i dla 90 Hz:



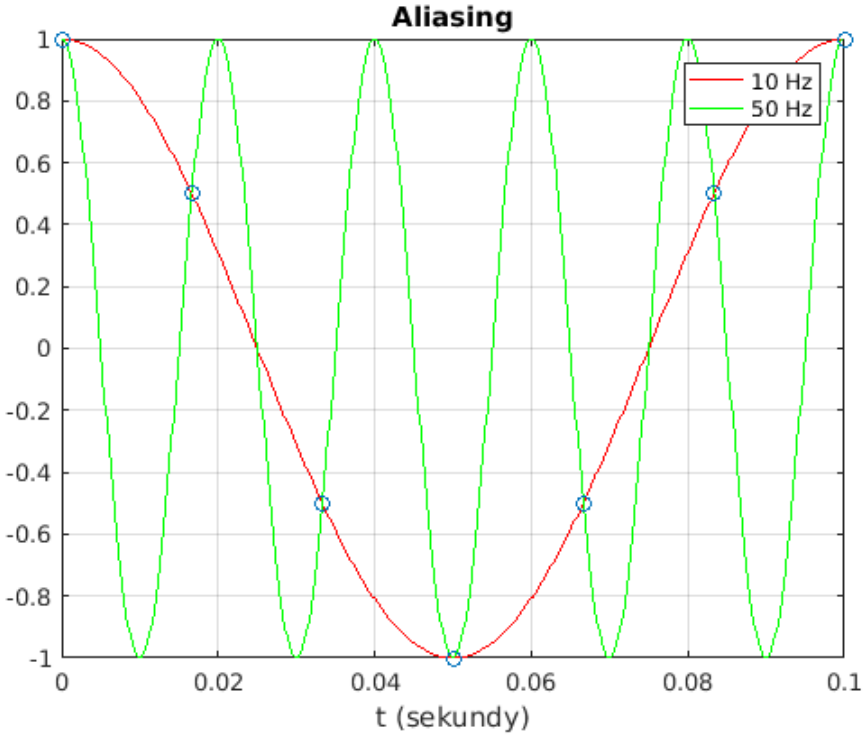
Rysunek 4: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz. Częstotliwość próbkowania 90 Hz.

Dopiero 100 H, czyli podwójna częstotliwość względem próbkowanego sygnału, pozwala na dostateczne odwzorowanie badanego sygnału:



Rysunek 5: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz.

Na poniższym rysunku widzimy problem aliasingu w pełnej krasie. Dla częstotliwości próbkowania 60 Hz sygnał 50 Hz wygląda tak samo jak dla częstotliwości 10 Hz.



Rysunek 6: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz.

III. WNIOSKI

W przypadku zamiany sygnału analogowego na cyfrowy podstawową rolę odgrywa twierdzenie Nyquista i wynikająca z niego minimalna częstotliwość próbkowania. Innymi słowy, do prawidłowego odtworzenia sygnału możliwe jest jedynie, gdy poszczególne składowe widmowe sygnału nie przeplatają się tj. $f_p > 2f_{max}$. Dzieje się tak ponieważ składowe widmowe wyższe od częstotliwości Nyquista ulegają nałożeniu na składowe o innych częstotliwościach. W wyniku tego zjawiska, które nazywamy aliasingiem, nie jesteśmy wstanie poprawnie odtworzyć pierwotnego sygnału. Zatem częstotliwość sygnału musi być równa co najmniej połowie częstotliwości próbkowania.

A. Materiały

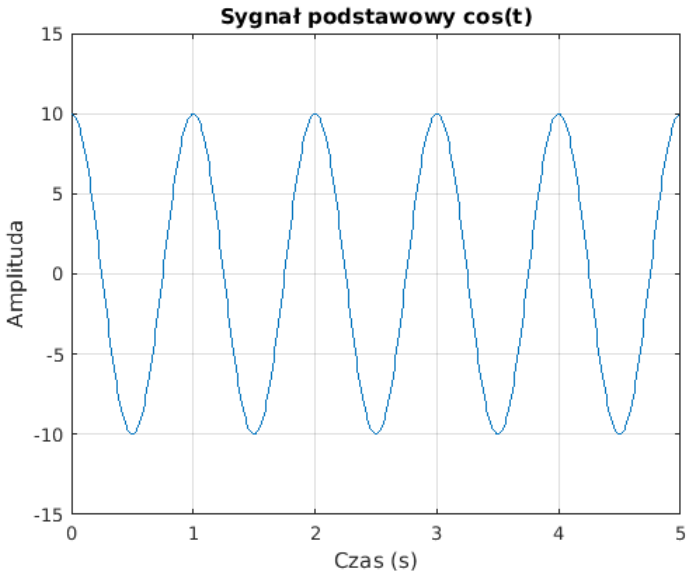
Przy przygotowaniu laboratorium skorzystałem z:
http://blogs.mathworks.com/steve/2010/03/03/aliasing-and-a-sampled-cosine-signal/

II. PRZYKŁAD FILTRU DOLNOPRZEPUSTOWEGO

Naszą pracę rozpoczniemy od symulowanego wygenerowania cosinusoidalnego sygnału o częstotliwości 50 Hz wygenerowanego przy pomocy poniższego kodu:

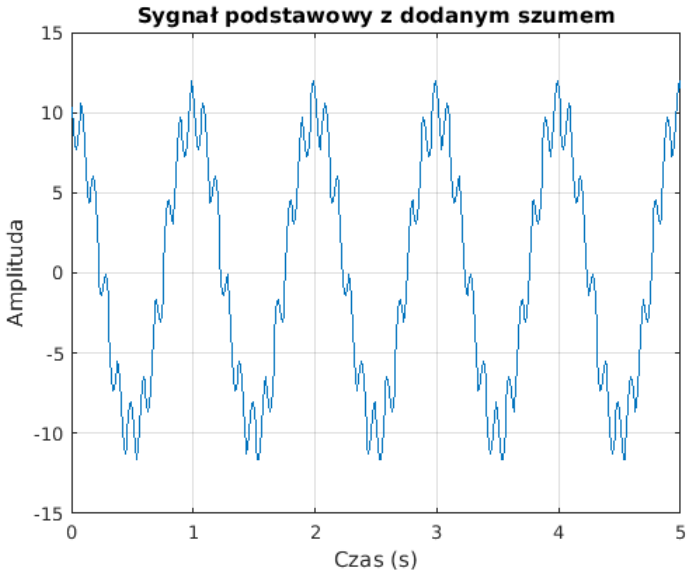
```
Fs = 100;
tmax = 5;
Nsamps = tmax*Fs;

t = 1/Fs:1/Fs:tmax;
%Sygnal podstawowy
s1 = 10*cos(2*pi*t);
%Sygnal wysokiej czestotliwosci
s2 = 2*cos(20*pi*t + pi/4);
%Sygnal podstawowy z nalozonym sygnalem wysokiej czestotliwosci
s3 = s1 + s2;
```



Rysunek 1: Wykres sygnału $s1$ o kształcie cosinusoidy dla częstotliwości 50 Hz.

Mając nasz sygnał testowy $s1$ dodamy do niego zakłócenie wysokiej częstotliwości $s2$:



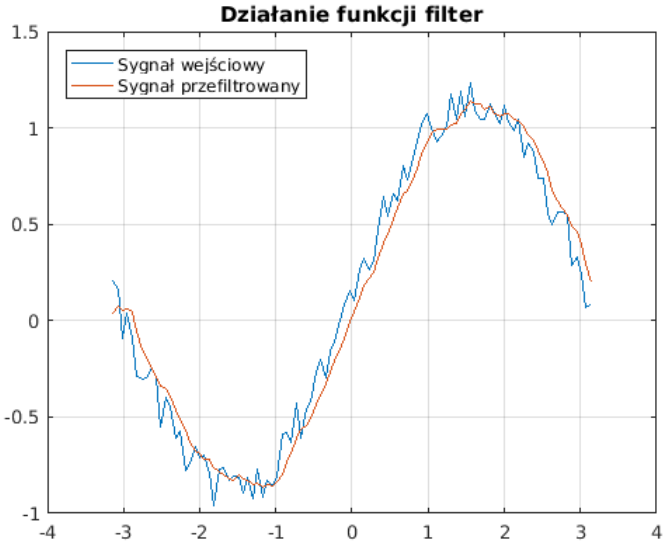
Rysunek 2: Wykres sygnału $s3$ o kształcie cosinusoidy dla częstotliwości 50 Hz z dodatnimi zakłóceniami wysokiej częstotliwości.

A. Opis funkcji *filter*

W dalszej części naszego laboratorium będziemy korzystać z wbudowanej funkcji *filter*, której opis znajduje się pod adresem: http://www.mathworks.com/help/matlab/ref/filter.html

Poniżej przykładowy kod:

```
t = linspace(-pi,pi,100);
rng default
x = sin(t) + 0.25*rand(size(t));
windowSize = 5;
b = (1/windowSize)*ones(1,windowSize)
a = 1;
y = filter(b,a,x);
```



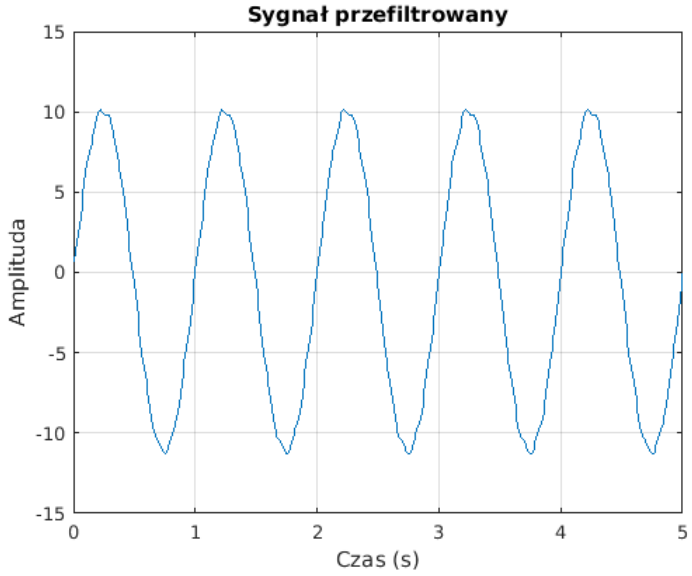
Rysunek 3: Przykładowy sygnał przed i po przeflitrowaniu.

B. Użycie funkcji *filter* do obróbki sygnału

Do zaprezentowania możliwości filtrowania użyjemy funkcji Matlaba *filter* z funkcją przejścia $H(z) = \frac{1}{1-z^{-1}}$ przy użyciu poniższego kodu:

```
b = 1;
a = [1  -1];

s3_f = filter(b,a,s3);
```



Rysunek 4: Sygnał po przefiltrowaniu przez filtr dolnoprzepustowy.

III. ZASTOSOWANIE TRANSFORMATY FOURIERA DO ANALIZY SYGNAŁU

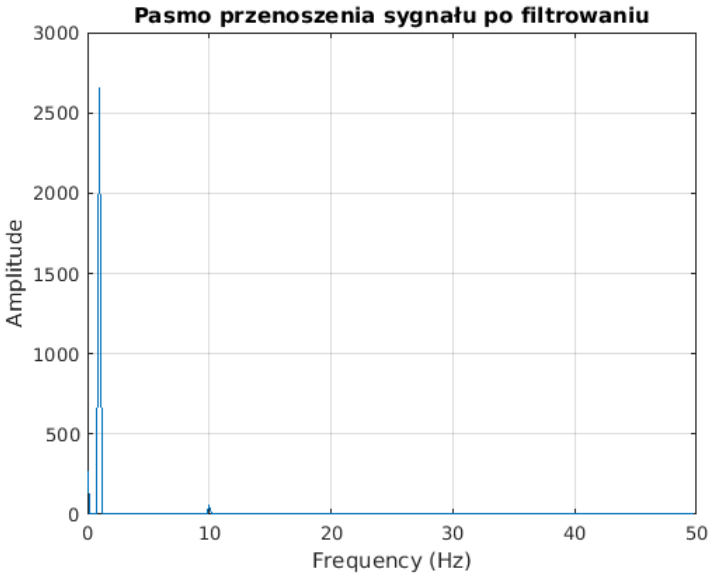
Do zobrazowania działania filtru przed i po jego użyciu, posłużymy się transformatą Fouriera przy użyciu poniższego kodu wykorzystującego Matlabową funkcję *fft*:

```
s3_fft = abs(fft(s3));
s3_fft = s3_fft(1:Nsamps/2);
```

Rysunek 5: Sygnał przed przefiltrowaniu przez filtr dolnoprzepustowy.

Analogicznie dla wykresu po przefiltrowaniu:

```
1 s3_f_fft = abs(fft(s3_f));
2 s3_f_fft = s3_f_fft(1:Nsamps/2);
```



Rysunek 6: Wykres sygnału o kształcie sinusoidy dla częstotliwości 50 Hz.

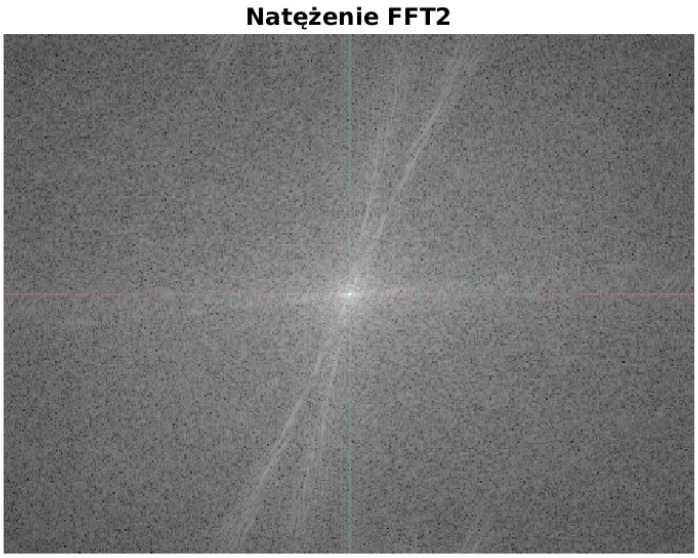
IV. ZASTOSOWANIE TRANSFORMATY FOURIERA DO OBRÓBKI ZDJĘĆ

Transformatę Fouriera można również użyć do przetwarzania obrazów. Pokażemy procedurę otrzymania wykresu widma natężenia obrazu i jego fazy.

Rysunek 7: Zdjęcia oryginalne.

Poniżej przykład zastosowania FFT do obróbki kolorowego zdjęcia w celu otrzymania obrazu widma natężenia i fazy. Do otrzymania widma natężenia zastosujemy poniższy kod:

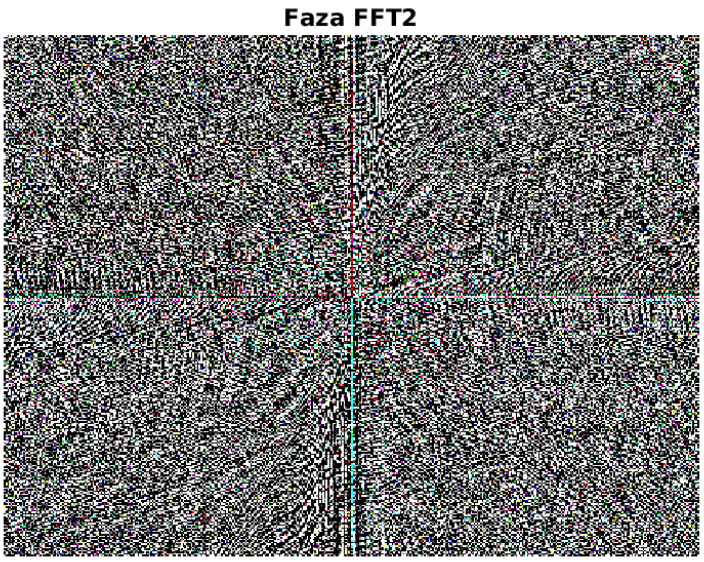
```
%Transformata Fouriera
pic = fft2(img);
%''Centrowanie'' FFT
pic = fftshift(pic);
%Pobranie wartosci natezenia
pic = abs(pic);
%Skalowanie
pic = log(pic+1);
pic = mat2gray(pic);
```



Rysunek 8: Widmo natężenia obrazu.

Poniższy kod posłużył do otrzymania fazy obrazu:

```
%Transformata Fouriera
pic = fft2(img);
%''Centrowanie'' FFT
pic = fftshift(pic);
pic = angle(pic);
```



Rysunek 9: Widmo fazy obrazu.

A. Materiały

Przy przygotowaniu laboratorium skorzystałem z:

http://matlabgeeks.com/tips-tutorials/digital-filtering-in-matlab/

http://matlabgeeks.com/tips-tutorials/how-to-do-a-2-d-fourier-transform-in-matlab/

http://www.mathworks.com/help/matlab/ref/filter.html;jsessionid=fa2737b141c6d046b82c47582a57