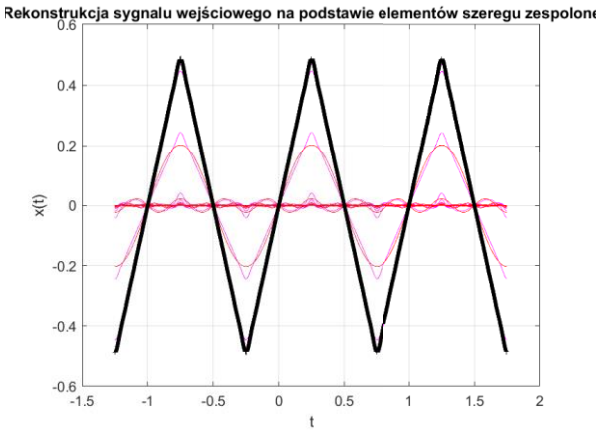
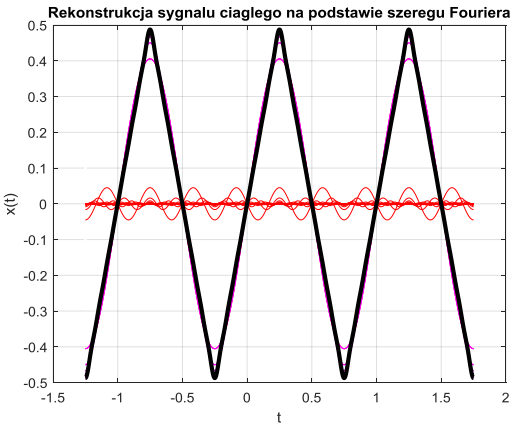
Lab. 4 Analiza harmoniczna - szeregi Fouriera cz.II

Nazwisko, Imię	Data wykonania ćw.	Planowy dzień zajęć	Planowa godz. zajęć
Dąbrowski, Mikołaj	2019.03.20	środa	08:00

Zad.5: Rekonstrukcja 3 okresów przebiegu wejściowego na podstawie elementów szeregu zespolonego



Rys1. Wykres z ćw.5



Rys2. Wykres z ćw.4

Jak widzimy, rekonstrukcje z ćw. 4 oraz 5 są niemal identyczne, osiągają te same wartości maksymalne, przecinają oś OX w tych samych punktach, są zgodne
z wykresem przebiegu trójkątnego z ćw.1.

Błąd aproksymacji przebiegu wyniósł 0.0126, został wyliczony według wzoru: $error = \max(|F(x) - f(x)|)$, gdzie $F(x)$ −funkcja oryginalna, $f(x)$ −funkcja aproksymująca

Ograniczenie liczby elementów szeregu wpływa negatywnie na aproksymacja, sprawia, że przybliżenie staje się mniej dokładne.

Poniżej umieszczony został kod programu realizujący polecenie:

```
step = (BND(2) - BND(1))/1000;
tt = [BND(1)-T0 : step: BND(2) + T0];
xx = zeros(1,length(tt));


figure
plot(tt,xx,'m'); grid on, hold on;
plot([tt(1),tt(2)],[max(xx)-0.1,min(xx)+0.1],'w.')

pause(0.5)
xlabel('t'); ylabel('x(t)');

for  n =  -NT : NT
    xx_n = X(n+NT+1)*exp(1j*w0*n*tt);
    xx = xx + xx_n;
    plot(tt,xx_n,'r');
    plot(tt,xx,'m');
    plot([tt(1),tt(2)],[max(xx)-0.1,min(xx)+0.1],'w.')
    title(sprintf('n = %d',n+1));
    pause(0.1)
end

plot(tt,xx,'k','LineWidth',3); grid on; hold on
plot([tt(1),tt(2)],[max(xx)-0.1,min(xx)+0.1],'w.')
title('Rekonstrukcja sygnalu wejściowego na podstawie elementów szeregu zespolonego')

tri = @(i) 0.5-abs(2*i-0.5);
time = -0.25:0.001:0.75;

jeden_okres = xx(1:1001);
pulse = tri(time);
diff = abs(real(pulse-jeden_okres));
error = max(diff);
```

Zad.6: Wyliczenie wartości skutecznych sygnałów

a) Dla przebiegu sinusoidalnego:

Kod realizujący dane ćwiczenie oraz otrzymany wynik:

```
T = 2*pi;
bound = [0 T];

syms fi

integral_a = int (sin(fi).^2, fi, 0, 2*pi);

s = sqrt((1/T)*integral_a);

skd = double(s);
```

```
skd =

    0.7071
```

Obliczenia analityczne:

$$S_k = \sqrt{\frac{1}{2\pi}\int_0^{2\pi} \sin^2 t\, dt} = \sqrt{\frac{1}{2\pi}\cdot\frac{1}{2}[t - \sin t \cos t]_0^{2\pi}} = \sqrt{\frac{1}{2\pi}\cdot\pi} = \sqrt{\frac{1}{2}} \approx 0.7071$$

Jak widzimy, wynik otrzymany z programu MATLAB jest zgodny z obliczeniami analitycznymi.

b) Wartość skuteczna przebiegu trójkątnego z Ćw.1:

Kod realizujący dane ćwiczenie oraz otrzymany wynik:

```
integral_b = int(x*x, BND);

s_b = sqrt((1/T0)*integral_b);

skd_tri = double(s_b);
```

$\Longrightarrow$

```
skd_tri =

    0.2887
```

c) Wartość skuteczna sygnału ciągłego z Ćw.4:

Do realizacji tego polecenia wykorzystano *tw. Parsevala*
Kod realizujący dane ćwiczenie oraz otrzymany wynik:

```
parseval = 0;

for n = -15:15
    parseval = parseval + abs(X(n+NT+1)).^2;
end

skd_c = sqrt(parseval);
```

$\Longrightarrow$

```
skd_c =

    0.2887
```

Zad.7: Wyznaczenie współczynnika zniekształceń harmonicznych THD

Poniższe wartości współczynników (idąc od góry: odpowiednio dla $n = 5,\ 10$ oraz 15) zostały wyznaczone za pomocą zamieszczonego kodu:

```
ans =

    0.1181

ans =

    0.1205

ans =

    0.1210
```

```
syms Sn t

thd_init = 0.0;
limit = 15;
for  n = 2 : limit
    Sn = 2*(a(n)*cos(w0*n*t) + b(n)*sin(w0*n*t));
    thd_init = thd_init + sqrt((1/T0*int(Sn*Sn,t,BND)))^2;
end

S_1 = 2*(a(1)*cos(w0*t)+b(1)*sin(w0*t));
THD = sqrt(thd_init)/sqrt(1/T0*int(S_1*S_1,t,BND));
double(THD)
```

Dla $n = 10$ oraz 15, kod wygląda identycznie, za wyjątkiem zmiennej limit, której wartość jest ustawiana: odpowiednio na 10 oraz 5

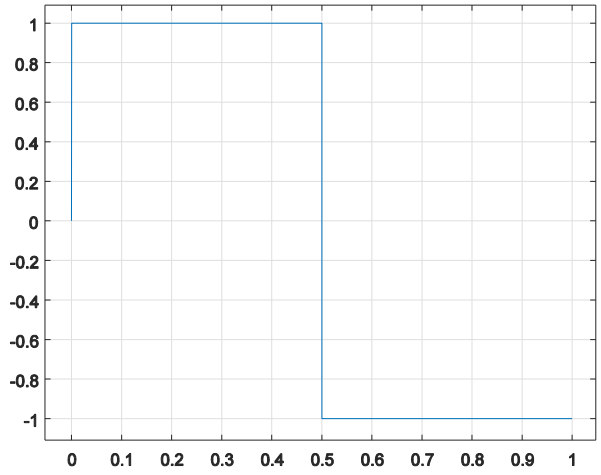
Możemy dostrzec zależność, iż wraz ze wzrostem n, wzrasta wartość współczynnika zniekształceń harmonicznych.

Wartość THD wyznaczonego analitycznie wynosi $\sqrt{\frac{\pi^4}{4} - 96} \approx 0.121$
Jak widzimy, współczynnik wyliczony w programie MATLAB, dla $n = 15$ praktycznie w idealny sposób przybliża wynik teoretyczny.

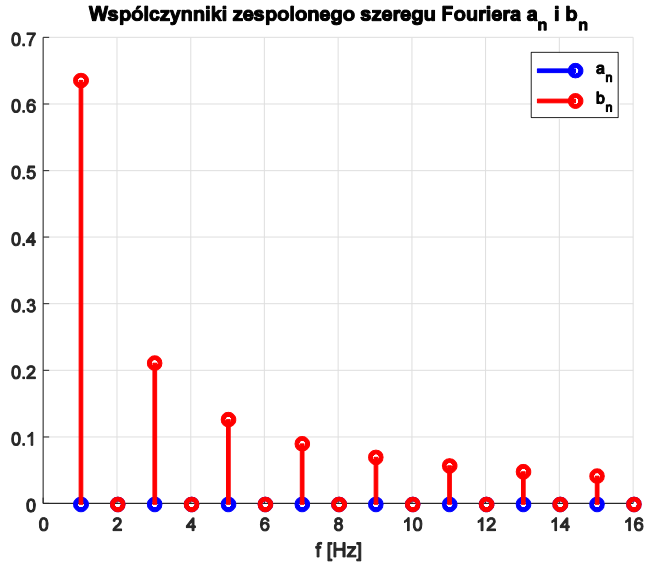
Zad.8: Odtworzenie 3 okresów przebiegu wejściowego w reprezentacji czasowej

P1 – fala prostokątna:

- przebieg jednego okresu sygnału:



- wykres współczynników:



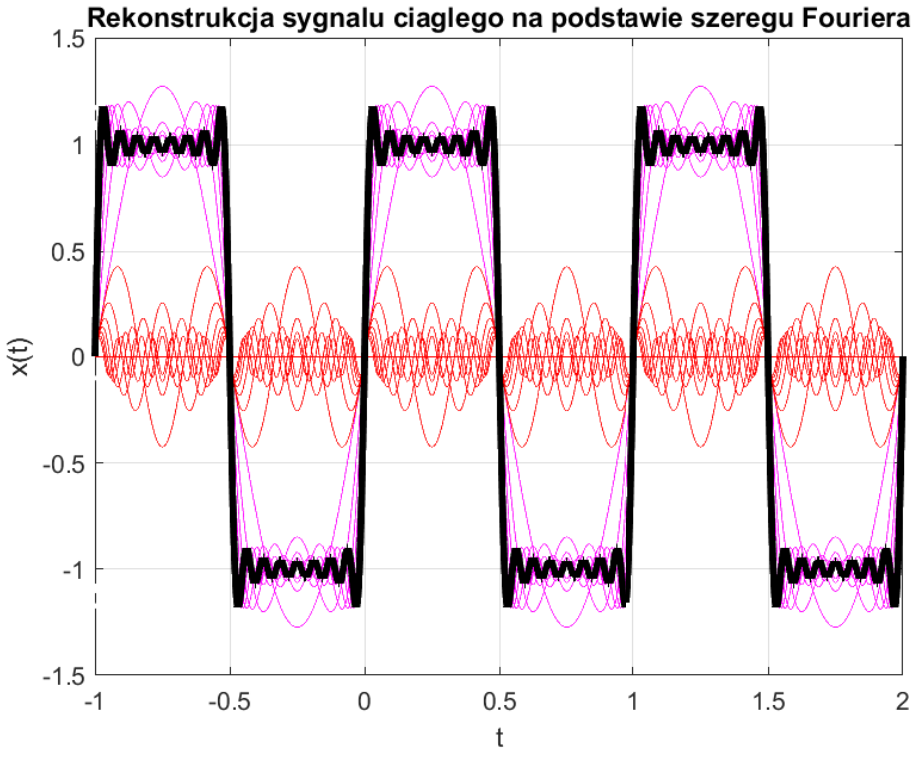
- wartość skuteczna:

```
sk_p =

    1
```

Kod wykorzystany do wyliczenia wartości skutecznej:

```
integral_p = int (x.^2, t, BND_p);

s = sqrt((1/T_p)*integral_p);

sk_p = double(s);
```

- aproksymacja za pomocą 16-tu funkcji bazowych:



- współczynnik zawartości harmonicznych THD:

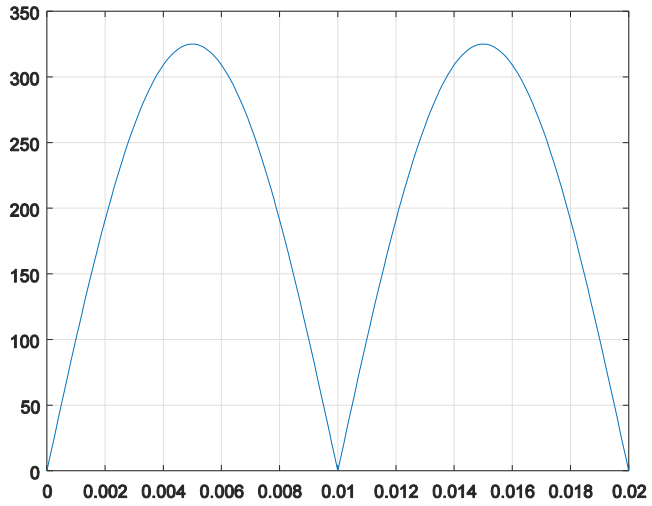
```
thd_init = 0.0;
limit = 15;
for  n = 2 : limit
    Sn_p = 2*(a_p(n)*cos(w_p*n*t) + b_p(n)*sin(w_p*n*t));
    thd_init = thd_init + sqrt((1/T_p*int(Sn_p*Sn_p,t,BND_p)))^2;
end

S_1 = 2*(a_p(1)*cos(w_p*t)+b_p(1)*sin(w_p*t));
THD = sqrt(thd_init)/sqrt(1/T_p*int(S_1*S_1,t,BND_p));
double(THD)
```

$\Longrightarrow$

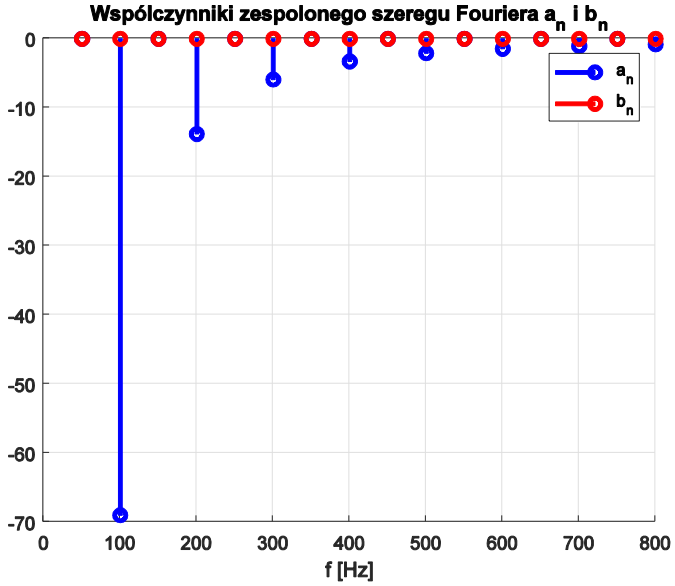
```
ans =

    0.4500
```

P2 – sinusoida wyprostowana dwupołówkowo:

- przebieg jednego okresu sygnału:



- wykres współczynników:

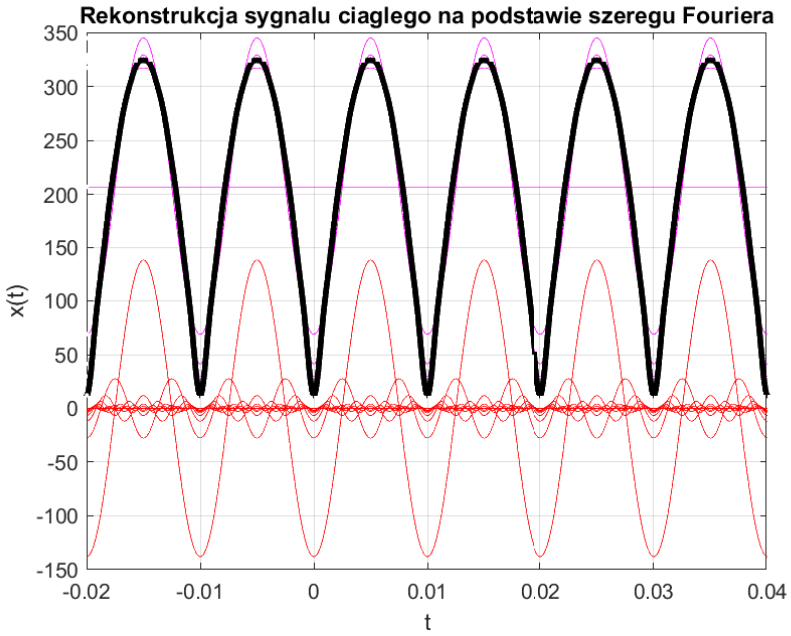


- wartość skuteczna: `sk_s =`

`229.8097`

Kod wykorzystany do wyliczenia wartości skutecznej był analogiczny, jak w przypadku przebiegu prostokątnego – zmienione zostały tylko parametry.

- aproksymacja za pomocą 16-tu funkcji bazowych:



- współczynnik zawartości harmonicznych THD:

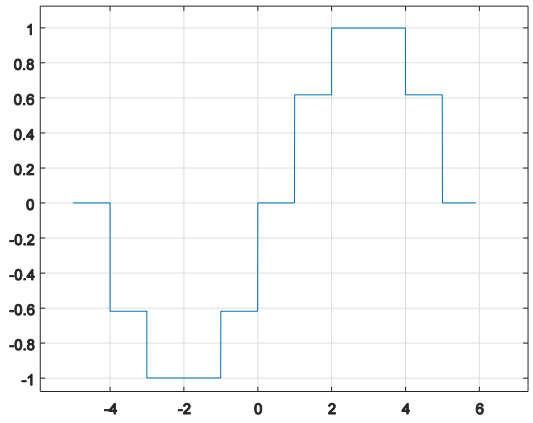
```
thd_init = 0.0;
limit = 15;
for  n = 3 : limit
    Sn_s = 2*(a_s(n)*cos(w_s*n*t) + b_s(n)*sin(w_s*n*t));
    thd_init = thd_init + sqrt((1/T_s*int(Sn_s*Sn_s,t,BND_s)))^2;
end

S_1 = 2*(a_s(2)*cos(w_s*2*t)+b_s(2)*sin(w_s*2*t));
THD = sqrt(thd_init)/sqrt(1/T_s*int(S_1*S_1,t,BND_s));
double(THD)
```

$\Longrightarrow$

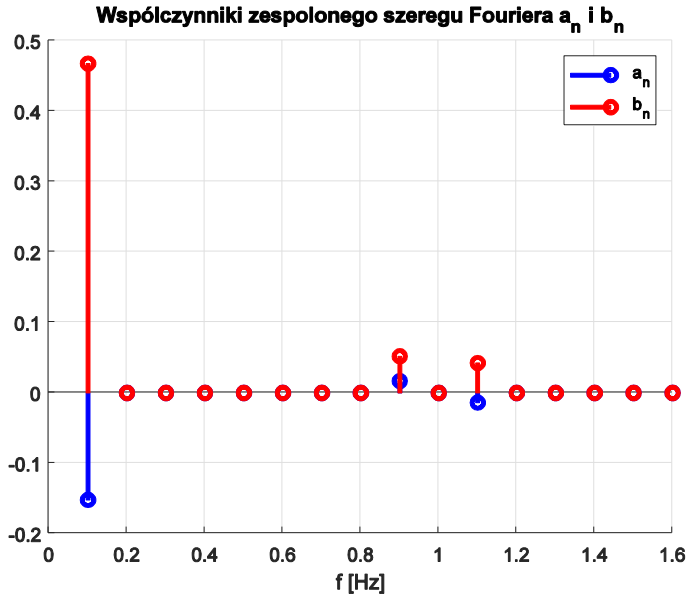
```
ans =

    0.2263
```

P3 – sinusoida modyfikowana:

- przebieg jednego okresu sygnału:



- wykres współczynników:

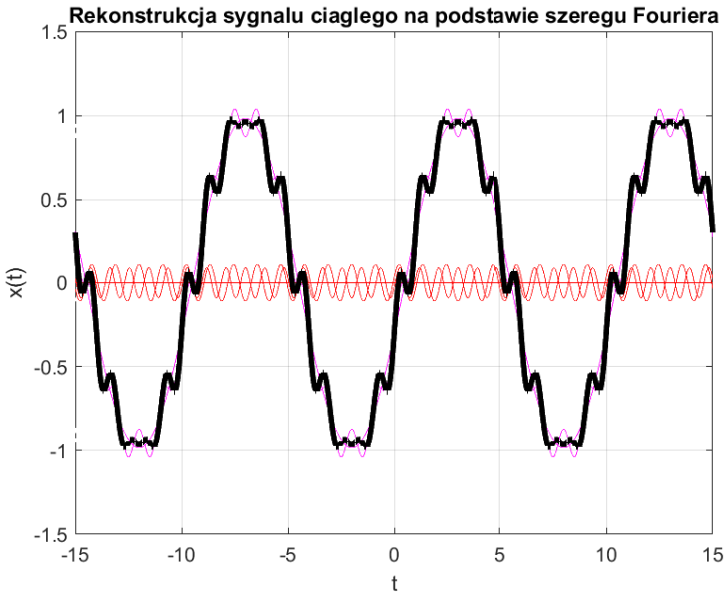


- wartość skuteczna:

```
sk_m =

    0.7071
```

Kod wykorzystany do wyliczenia wartości skutecznej był analogiczny, jak w przypadku przebiegu prostokątnego – zmienione zostały tylko parametry.

- aproksymacja za pomocą 16-tu funkcji bazowych:



- współczynnik zawartości harmonicznych THD:

```
thd_init = 0.0;
limit = 15;
for  n = 2 : limit
    Sn_m = 2*(a_m(n)*cos(w_m*n*t) + b_m(n)*sin(w_m*n*t));
    thd_init = thd_init + sqrt((1/T_m*int(Sn_m*Sn_m,t,BND_m)))^2;
end

S_1 = 2*(a_m(1)*cos(w_m*t)+b_m(1)*sin(w_m*t));
THD = sqrt(thd_init)/sqrt(1/T_m*int(S_1*S_1,t,BND_m));
double(THD)
```

$\Longrightarrow$

```
ans =

    0.1436
```