Analiza i synteza przebiegów odkształconych - szeregi Fouriera

Mikołaj Małecki 237339 K00-15c

25 marca 2020

1 Wprowadzenie

Szereg Fouriera jest to ciąg (dla danej chwili t) wyrażony wzorem:

Definicja 1: Szereg Fouriera

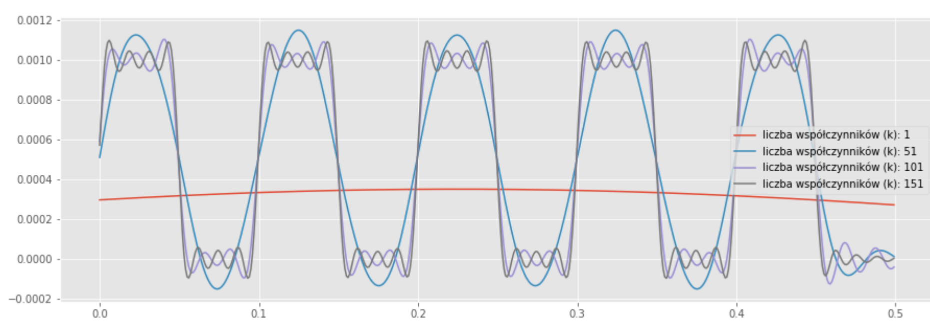
$$x(t) = a_0 + \sum_{k=1}^{\infty}(a_k \cos(k\omega_0 t) + b_k \sin(k\omega_0 t))$$

gdzie:

a_0 - wartość średnia - jego zmiana powoduje offset sugnału estymowaneg

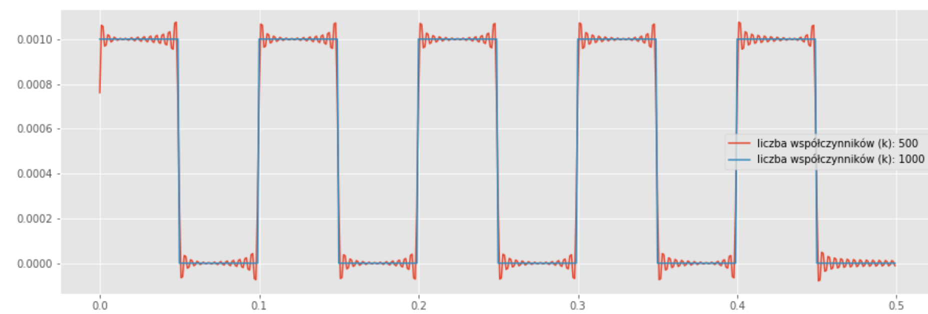
a_k, b_k - współczynniki trygonometryczne - ich odpowiednie dobranie pozwala z pewną dokładnością estymować sygnały

Dla czasu niedyskretnego operacja wyliczenia wartości z powyższej definicji oznacza nałożenie na siebie kolejnych skłądowych harmonicznych estymowanego sygnału. Dla nieskończonego ciągu sumy waryżeń szeregu - wynik jest równy estymowanemu sygnałowi. Jednakże z uwagi na niemożliwość przeprowadzenia takiej operacji numerycznie dobiera się wartość liczby wyrazów sumy optymalnie pod względem kryterium potrzebnej dokłądności do uciążliwości wykonywanych obliczeń.



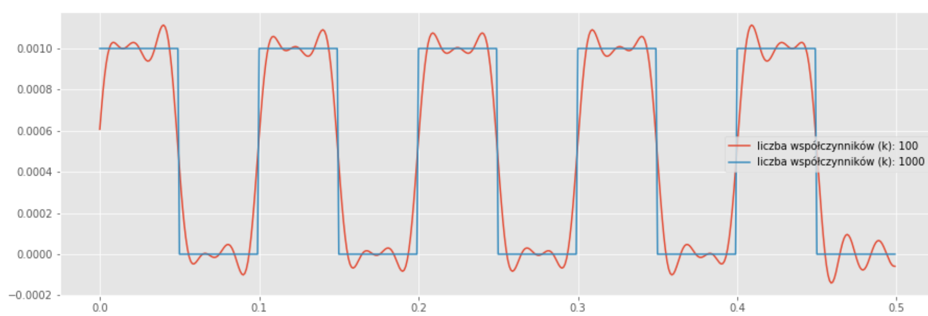
Rysunek 1: Sygnał binarny unipolarny estymowany szeregami fouriera z różną liczbą współczynników

Można zauważyć że kolejne wykresy przedstawiające bardziej dokładne estymacje coraz bardziej ukazują przebieg binarny. Co się stanie kiedy zwiększymy zmienną k ?



Rysunek 2: Sygnał binarny unipolarny estymowany szeregami fouriera z różną liczbą współczynników

Można zauważyć że przebiegi dla odpowiednio większy wartości k w widocznym przybliżeniu nie odbiegają od oryginalnego.
Jak natomiast wygląda sytuacja czasu obliczeń? Dla poniższych przebiegów:



Rysunek 3: Sygnał binarny unipolarny estymowany szeregami fouriera z różną liczbą współczynników

- Dla k = 10 $\rightarrow$ `100 loops, best of 3: 5.54 ms per loop`
- Dla k = 1000 $\rightarrow$ `10 loops, best of 3: 55.9 ms per loop`

2 Przebiegi czasowe i ich estymacje

2.1 Kod

- Dla wygenerowania żądanych przebiegów stworzony został generator sygnałów:

```
def make_func(func_L, func_R, amp, f):
    dx = 1
    time = np.arange(0, num_periods * f, dx)
    h_T = int(np.floor(len(time)/(num_periods*2)))
    func = np.zeros_like(time)

    for i in range(num_periods):
        func[i*2 * h_T : (i*2+1) * h_T] = func_L(h_T) * amp
        func[(i*2+1) * h_T : (i*2+2) * h_T] = func_R(h_T) * amp
    return Signal(time/1000, time/1000, func/1000, dx/1000)
```

- Natomiast w celu obliczenia współczynników potrzebnych do otrzymowania estymowanego sygnału utworzona została metoda:

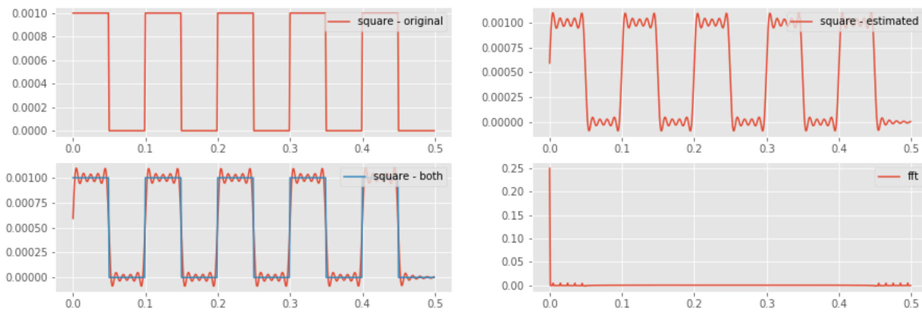
```
def m_approx(signal, n_harmonics):
    A0 = np.sum(signal.y * np.ones_like(signal.x)) * signal.dx
    fFS = A0/2

    A = np.zeros(n_harmonics)
    B = np.zeros(n_harmonics)
    for k in range(n_harmonics):
        A[k] = np.sum(signal.y * np.cos(np.pi*(k+1)*signal.time)) * signal.dx
        B[k] = np.sum(signal.y * np.sin(np.pi * (k+1)*signal.time)) * signal.dx
        fFS += A[k]*np.cos(np.pi*(k+1)*signal.time) + B[k]*np.sin(np.pi * (k+1)*signal.time)
    return Signal(signal.time, signal.x, fFS, signal.dx)
```

- Wreszcie zestawienie żądanych wykresów zostało wykonane za pomocą funkcji:

```
def make_plots(signal: Signal, name: str, coefs: int):
    fig, ax = plt.subplots(2,2)

    ax[0, 0].plot(signal.x, signal.y, label=f"{name} - original")
    ax[0, 0].legend(loc="upper right")

    sig = m_approx(signal, coefs)
    ax[0, 1].plot(sig.x, sig.y, label=f"{name} - estimated")
    ax[0, 1].legend(loc="upper right")
    ax[1, 0].plot(sig.x, sig.y)
    ax[1, 0].plot(signal.x, signal.y, label=f"{name} - both")
    ax[1, 0].legend(loc="upper right")

    ax[1, 1].plot(sig.x, sc.fft(sig.y), label=f"fft")
    ax[1, 1].legend(loc="upper right")

    plt.show(fig)
```

2.2 Prostokąt

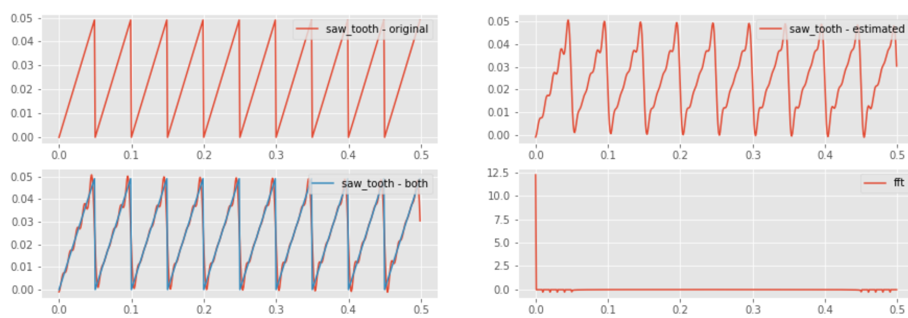
```
L = lambda h_T: np.ones(h_T)
P = lambda h_T: np.zeros(h_T)
square = make_func(L,P,1,100)
make_plots(square, "square", 200)
```



Rysunek 4: Zestaw przebiegów dla syngału prostokątnego

2.3 Piła

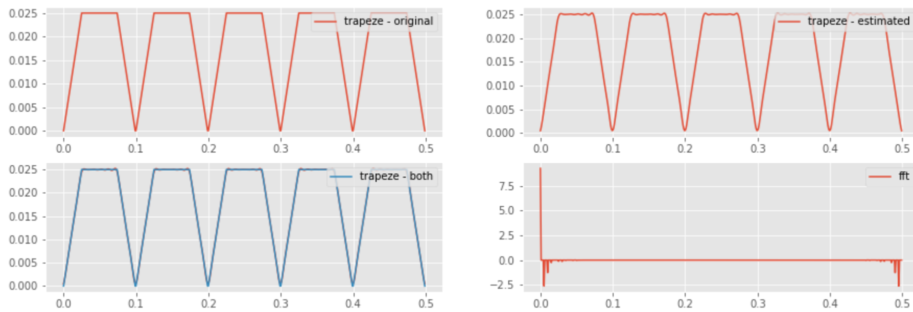
```
L = lambda h_T: np.arange(0, h_T)
P = lambda h_T: np.arange(0, h_T)
saw_tooth = make_func(L,P,1,100)
make_plots(saw_tooth, "saw_tooth", 200)
```



Rysunek 5: Zestaw przebiegów dla syngału piły

2.4 Trapez

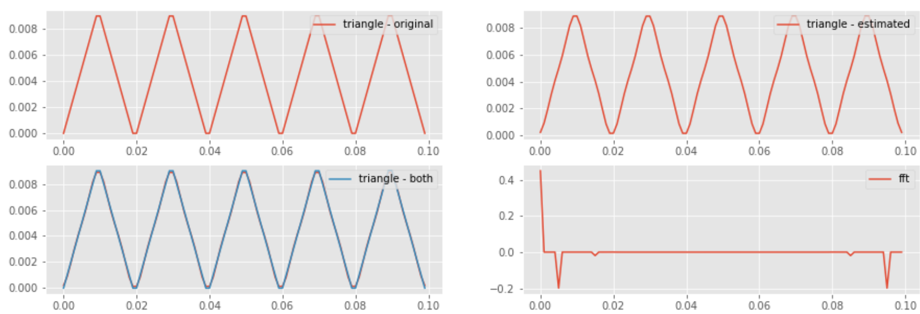
```
L = lambda h_T: np.hstack([np.arange(0, h_T/2), np.ones(int(h_T/2))*int(h_T/2)])
P = lambda h_T: np.hstack([np.ones(int(h_T/2))*int(h_T/2), np.flipud(np.arange(0, h_T/2))])
trapeze = make_func(L,P,1,100)
make_plots(trapeze, "trapeze", 200)
```



Rysunek 6: Zestaw przebiegów dla syngału trapezoidalnego

2.5 Trójkąt

```
L = lambda h_T: np.arange(0, h_T)
P = lambda h_T: np.flipud(np.arange(0,h_T))
triangle = make_func(L,P,1,20)
make_plots(triangle, "triangle", 400)
```



Rysunek 7: Zestaw przebiegów dla syngału trójkątnego

3 Obliczenia z wykorzystaniem zespolonego szeregu fouriera

Bazując na wygodzie pracowania w dyskretnym środowisku programistycznym korzyść ze zmniejszenia liczby całkowań w tej metodzie nie jest adekwatna do złożoności zaprogramowania tej metody. Operacja całkowania dla dyskretnych skończonych danych jest poprostu operacją sumowania dlatego większość obliczeń została wykonana z podstawowej definicji.

Natomiast dla sygnału piłokształtnego, obliczenia z wykorzystaniem zespolonego szeregu fouriera przedstawiają się dla sygnału → **Rys.5** *saw_tooth - original* - sygnał piły w czasie.

$$A = 0.05 \quad T = 0.05s \quad f = 20Hz \quad \omega_0 = 40\pi \frac{rad}{s}$$

$$c_k = \int_0^1 t \cdot e^{-jk\omega_0 t} dt = -\frac{e^{-2i\pi k}(-2i\pi k + e^{2i\pi k} - 1)}{4\pi^2 k^2}$$

$$c_k = \frac{j}{2k\pi}$$

$$a_k = 2Re(c_k) = 0 \quad b_k = -2Im(c_k) = \frac{-1}{k\pi} \quad a_0 = 0.05$$

Zatem sygnał można zapisać w postaci:

$$x_{ap}(t) = 0.05 + \sum_{k=1}^{k_0} (\frac{-1}{k\pi} \sin(40k\pi t))$$

4 Wnioski

- Wraz ze zwiększaniem liczby uwzględnianych harmonicznych dokładność estymacji rośnie
- Wraz ze zwiększaniem liczby uwzględnianych harmonicznych efekt gibsa maleje (niestandardowo dla ostrych krawędzi sygnału bazowego)
- Kolejne harmoniczne są widoczne jako skwantowane stałe wartości na widmie fazowym

Literatura

[1] Instrukcja do laboratorium
Szeregi Fouriera.

[2] 3Blue1Brown
Ale czym jest szereg Fouriera? Od przepływu ciepła do szkiców okręgów.
textttthttps://www.youtube.com/watch?v=r6sGWTCMz2k