EA614U - Análise de Sinais - 2s2020

Exercício de fixação de conceitos (EFC) 2 - Série de Fourier

Aluno: Vinícius de Lima Quadrado - 225357

(a) No período -2t a 2t a função $x(t)$ é dada por:

$$x(t) = -t + 1, -2 < t < 2$$

Os coeficientes, pela definição, são dados por:

$$a_k = \frac{1}{T} \int_{\frac{-T}{2}}^{\frac{-T}{2}} x(t) e^{-jk\omega_0 t} dt$$

logo temos:

$$a_k = \frac{1}{4} \int_{-2}^{2} (-t+1) e^{-jk\omega_0 t} dt = -\frac{1}{4} \int_{-2}^{2} (t-1) e^{-jk\omega_0 t} dt$$

$$\text{resolvendo } \int (t-1) e^{-jk\omega_0 t} dt \text{ por substituição: } \int fg' = fg - \int f'g$$

$$f = t - 1,\ f' = 1 \qquad g' = e^{-jk\omega_0 t},\ g = \frac{je^{-jk\omega_0 t}}{k\omega_0}$$

$$\begin{aligned}
\int (t-1) e^{-jk\omega_0 t} dt &= \frac{j(t-1)e^{-jk\omega_0 t}}{k\omega_0} - \int \frac{je^{-jk\omega_0 t}}{k\omega_0} dt \\
&= \frac{j(t-1)e^{-jk\omega_0 t}}{k\omega_0} - \frac{j}{k\omega_0} \int e^{-jk\omega_0 t} dt \\
&= \frac{j(t-1)e^{-jk\omega_0 t}}{k\omega_0} + \frac{e^{-jk\omega_0 t}}{k^2\omega_0^2}
\end{aligned}$$

Substituindo este resultado na equação de a_k

$$a_k = -\frac{1}{4}\int (t-1)e^{-jk\omega_0 t}dt = -\frac{j(t-1)e^{-jk\omega_0 t}}{4k\omega_0} - \frac{e^{-jk\omega_0 t}}{4k^2\omega_0^2} = -\frac{(jk\omega_0(t-1)+1)e^{-jk\omega_0 t}}{4k^2\omega_0^2} + C$$

$$\text{simplificando: } a_k = -\frac{j(k\omega_0 t - k\omega_0 - j)e^{-jk\omega_0 t}}{4k^2\omega_0^2}$$

Avaliando o resultado para o intervalo de $-2 < t < 2$

$$a_k = \left(-\frac{j(k\omega_0 t - k\omega_0 - j)e^{-jk\omega_0 t}}{4k^2\omega_0^2}\right)\Bigg|_{-2}^{2}$$

$$\begin{aligned}
a_k &= \left(-\frac{j(k\omega_0 t - k\omega_0 - j)e^{-jk\omega_0 t}}{4k^2\omega_0^2}\right)\Bigg|_{-2}^{2} = \left(\frac{j(-k\omega_0 t + k\omega_0 + j)e^{-jk\omega_0 t}}{4k^2\omega_0^2}\right)\Bigg|_{-2}^{2} \\
&= \left(\frac{j(-k\omega_0 + j)e^{-2jk\omega_0}}{4k^2\omega_0^2}\right) - \left(\frac{j(3k\omega_0 + j)e^{2jk\omega_0}}{4k^2\omega_0^2}\right), cos(x) + jsen(x) = e^{jx} \\
&= \left(\frac{j(-k\omega_0 + j)(cos(2k\omega_0) - jsen(2k\omega_0))}{4k^2\omega_0^2}\right) - \left(\frac{j(3k\omega_0 + j)(cos(2k\omega_0) + jsen(2k\omega_0))}{4k^2\omega_0^2}\right) \\
&= \left(\frac{(-k\omega_0 + j)(jcos(2k\omega_0) + sen(2k\omega_0))}{4k^2\omega_0^2}\right) - \left(\frac{(3k\omega_0 + j)(jcos(2k\omega_0) - sen(2k\omega_0))}{4k^2\omega_0^2}\right) \\
&= \left(\frac{-k\omega_0 jcos(2k\omega_0) - cos(2k\omega_0) - k\omega_0 sen(2k\omega_0) + jsen(2k\omega_0)}{4k^2\omega_0^2}\right) - \\
&\qquad \left(\frac{3k\omega_0 jcos(2k\omega_0) - cos(2k\omega_0) - 3k\omega_0 sen(2k\omega_0) - jsen(2k\omega_0)}{4k^2\omega_0^2}\right) \\
&= \left(\frac{-4k\omega_0 jcos(2k\omega_0) + 2k\omega_0 sen(2k\omega_0) + 2jsen(2k\omega_0)}{4k^2\omega_0^2}\right) \\
a_k &= \frac{sen(2k\omega_0)(k\omega_0 + j) - 2k\omega_0 jcos(2k\omega_0)}{2k^2\omega_0^2}
\end{aligned}$$

O primeiro termo a_0 é dado calculando o limite da expressão de a_k quando k tender a zero

$$\begin{aligned}
a_0 &= \lim_{k\to 0} a_k = \lim_{k\to 0} \frac{sen(2k\omega_0)(k\omega_0 + j) - 2k\omega_0 jcos(2k\omega_0)}{2k^2\omega_0^2} \\
a_0 &= 1
\end{aligned}$$

Como $a_0 = 1$, o nível DC também é igual a 1

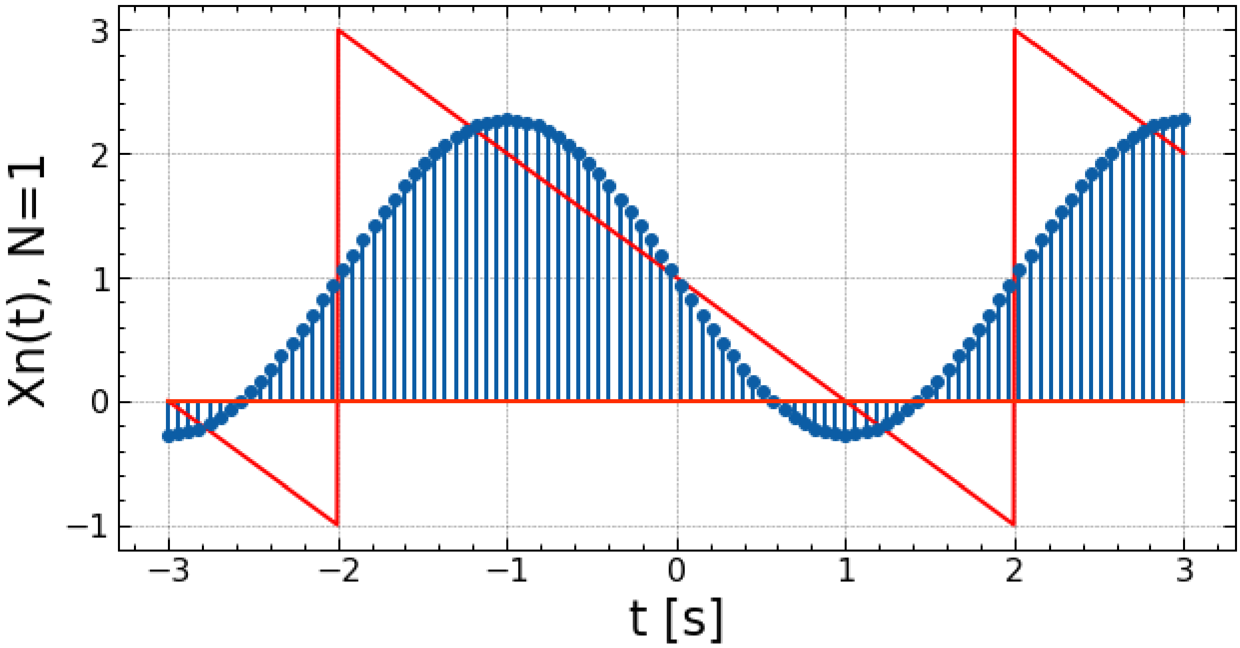
(b) Exiba, em gráficos diferentes, a onda "dente de serra" original junto com sua aproximação dada pela série de Fourier com os valores N = 1, 10, 20, 50, para um período do sinal.

Implementar computacionalmente a expressão abaixo:

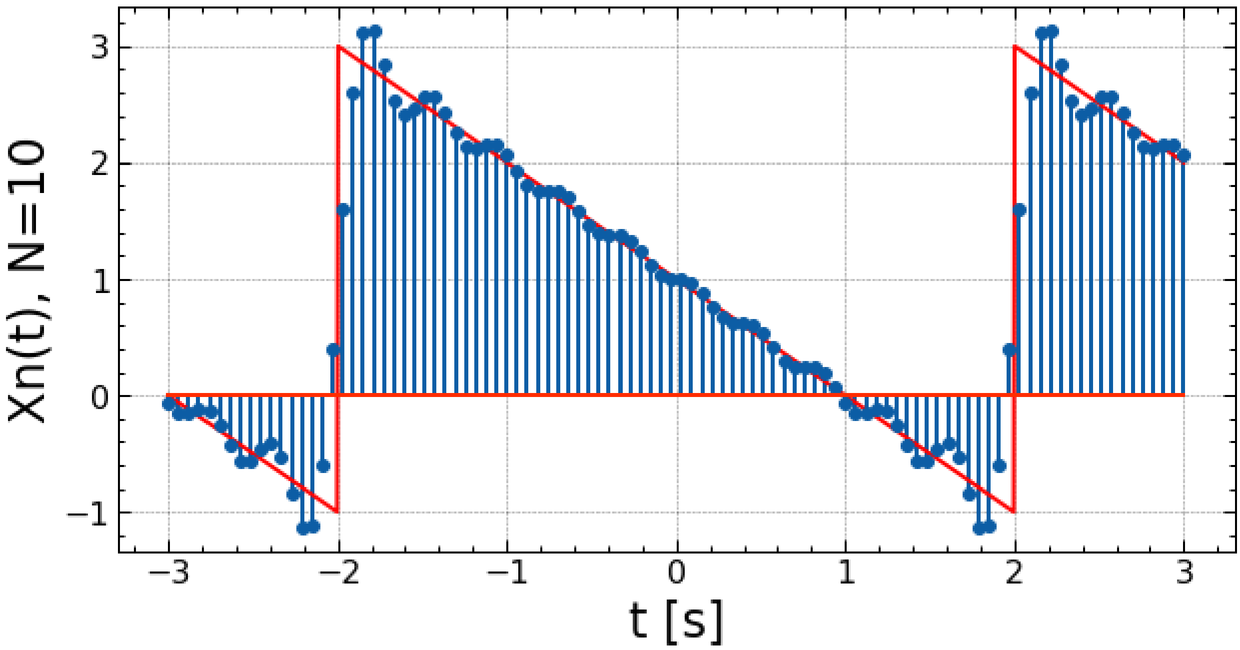
$$\widetilde{x}_N = \sum_{k=-N}^{N} a_k e^{jk\omega_0 t}$$

Obs: código anexado

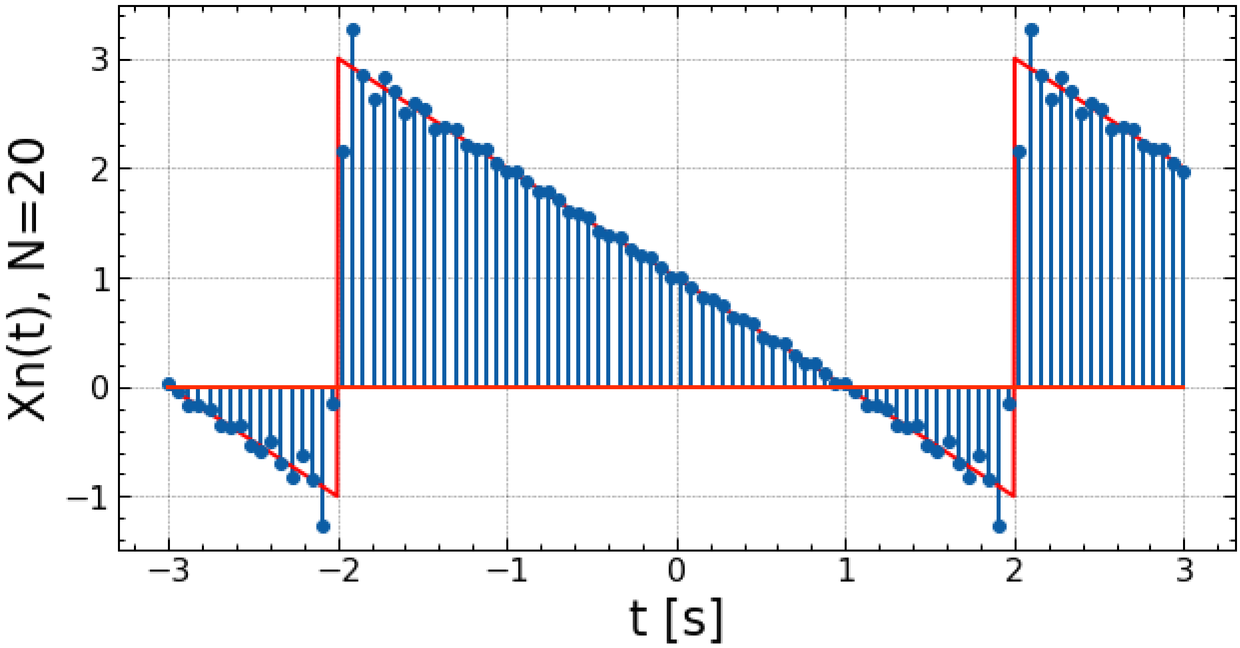
N=1



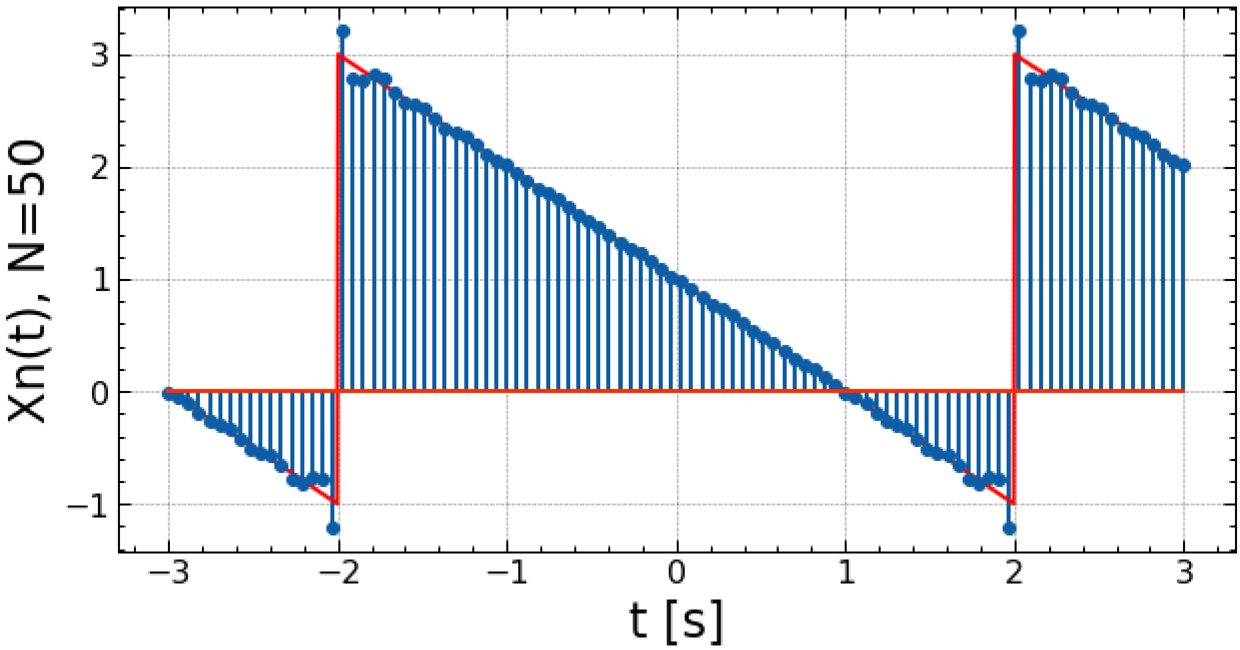
N=10



N=20



N=50



(c) Calcule a potência média do erro P_N para todos os valores de N

```
def xN(N, w, t):
    buffer = []
    for k in range(-N, N+1, 1):
        product = a(k, w)*np.exp(1j*k*w*t)
        buffer.append(np.real(product))

    return np.sum(buffer)

def pN(xt, xn, amostras):
    return (np.sum((xt-xn)**2))/amostras

X = 2
amostras = 1000*X
timePoints = np.linspace(-X, X, amostras)

w = np.pi/2
N = [1, 10, 20, 50]
```

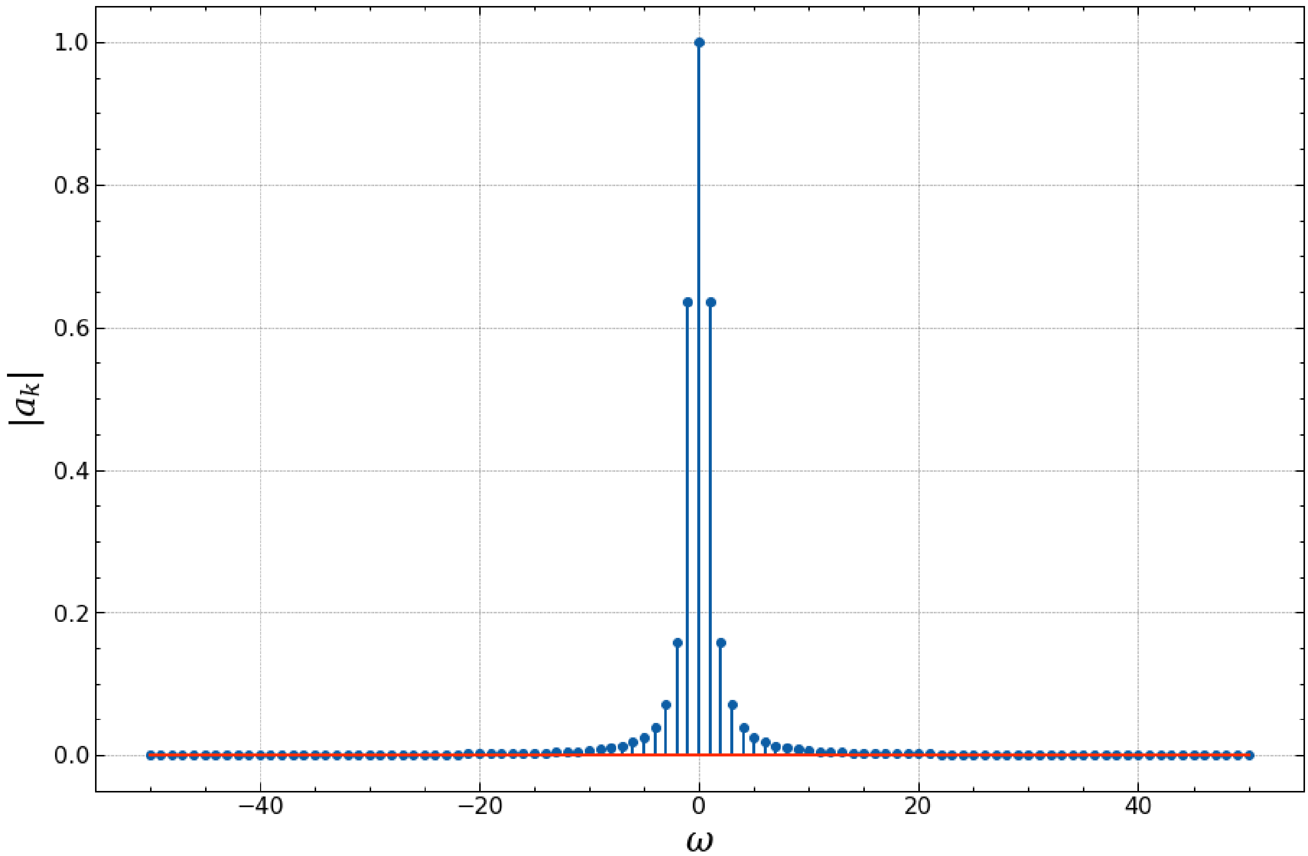
```
for n in N:
    x_N = []
    x_t = []
    for i in timePoints:
        x_N.append(xN(n, w, i))

    x_N = np.array(x_N)
    x_t = 2*signal.sawtooth(((2 * np.pi * 0.25 * timePoints)-np.pi), width=0)+1

    P = pN(xt=x_t, xn=x_N, amostras=amostras)
    print(f'Para N = {n}, Pn = {P}')
```

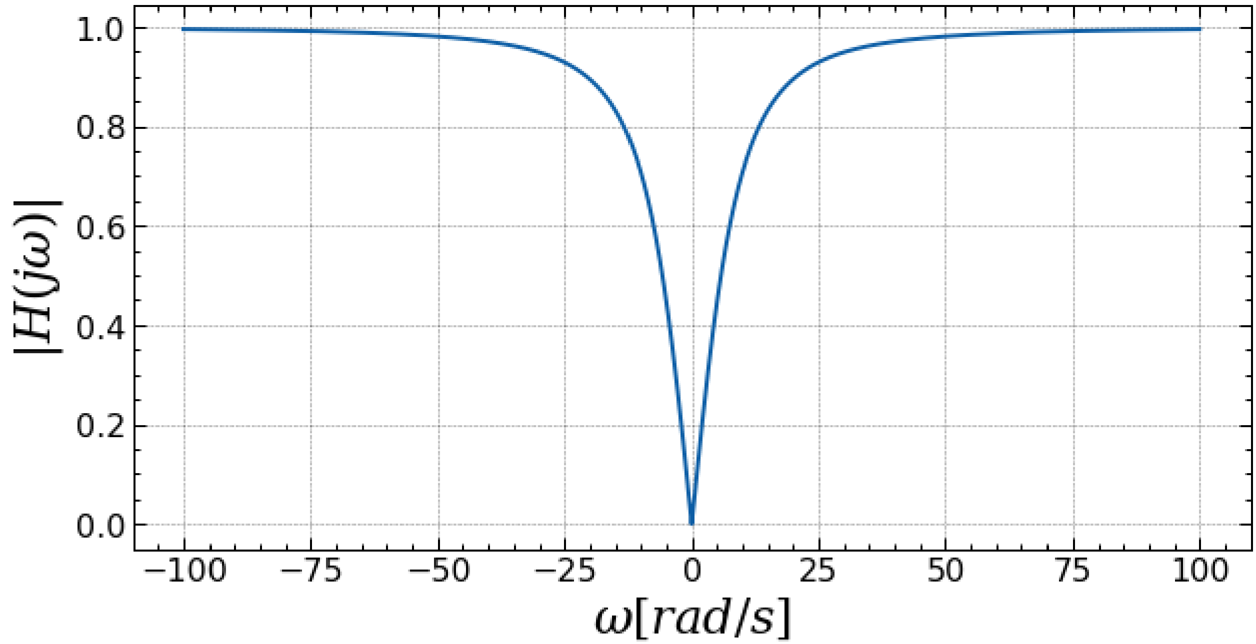
```
Para N = 1, Pn = 0.5245044832632534
Para N = 10, Pn = 0.07911436381835221
Para N = 20, Pn = 0.04153972162329742
Para N = 50, Pn = 0.018109707432599136
```

(d) Para $N = 50$, plotar os valores absolutos dos coeficientes da série a_k em função de ω

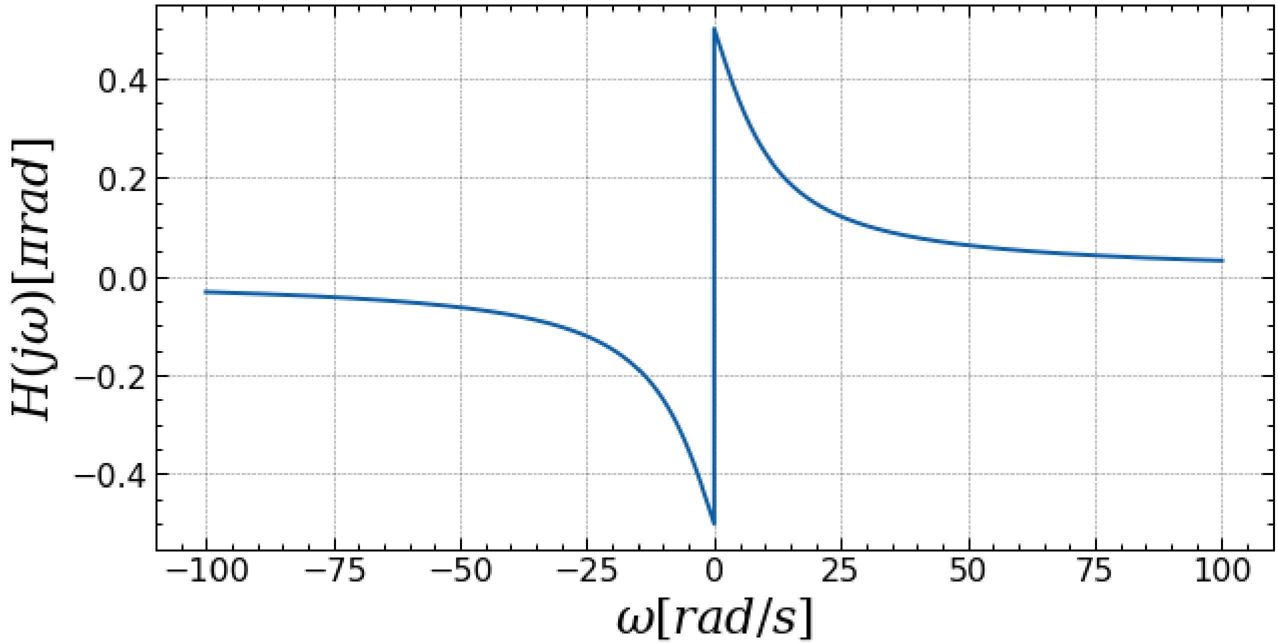


(e) Plotar o módulo e a fase da resposta em frequência de $H(jw) = \frac{1}{1-j\frac{\omega_c}{\omega}}$ e discuta a ação deste sistema como um filtro:

- Módulo de $H(jw)$



- Fase de $H(jw)$



A ação deste sistema como um filtro:

Este sistema atua como um filtro passa-alta, isto é, ele atenua frequências baixas e deixa passar frequências mais altas, observando o gráfico do módulo de $H(j\omega)$, sinais de frequências acima de $\omega = 25rad/s$ são atenuados em cerca de 10%, e sinais de frequências acima de $\omega = 75rad/s$ já não sofrem atenuação. Este filtro possui boa filtragem de frequências próximas a zero, portanto é um bom filtro de níveis DC.

(f) Determine a saída $y(t)$ quando $x(t)$ for o sinal dente de serra aproximado com $N = 50$ e a resposta em frequência do sistema seja $H(jw) = \frac{1}{1-j\frac{\omega_c}{\omega}}$

Pelas propriedades de linearidade e autofunção, a saída $y(t)$ é dada por:

$$y(t) = \sum_{k=-\infty}^{\infty} c_k H(jk\omega_0) e^{jk\omega_0 t}$$

Onde $c_k = a_k$, calculado no item (a) e $H(jkw_0)$ é:

$$H(jk\omega_0) = \frac{1}{1 - j\frac{\omega_c}{k\omega_0}}, \omega_c = \frac{1}{RC} = 10, \omega_0 = \frac{\pi}{2} \implies H(jk\omega_0) = \frac{1}{1 - j\frac{10}{k\frac{\pi}{2}}}$$

$$H(jk\omega_0) = \frac{1}{1 - j\frac{20}{k\pi}}$$

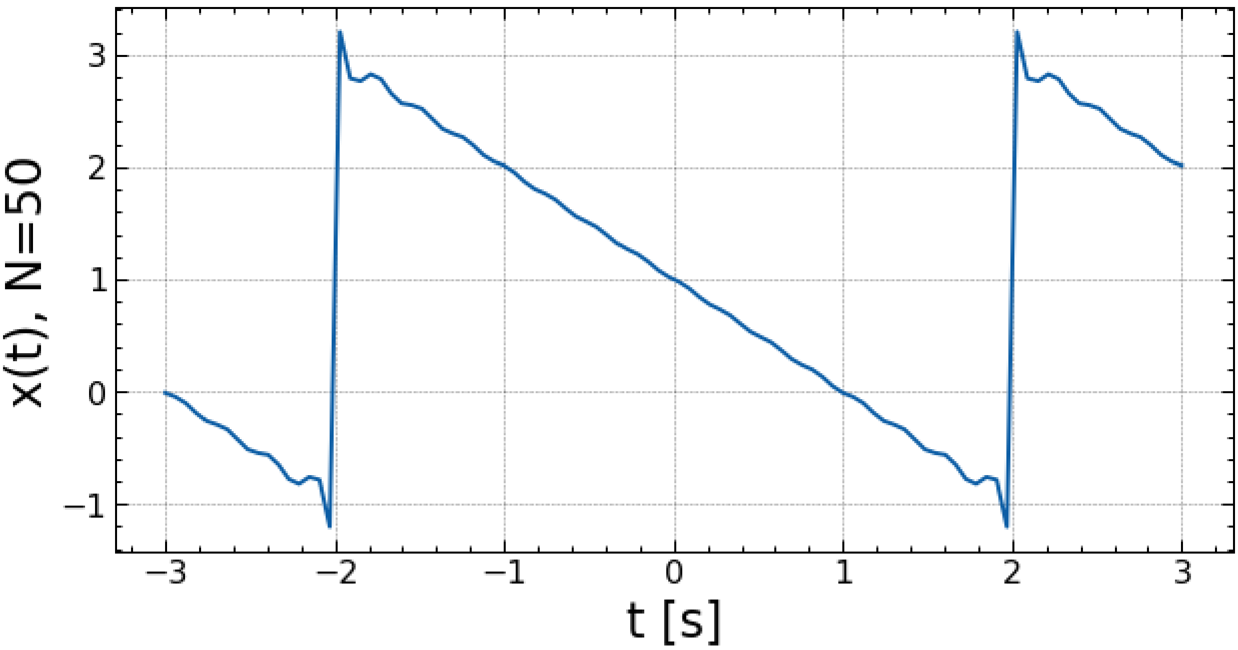
e para construção da função no código é necessário saber o limite de $H(jkw_0)$ para k tendendo a zero:

$$\lim_{k \to 0} H(jk\omega_0) = \lim_{k \to 0} \frac{1}{1 - j\frac{20}{k\pi}} = 0$$

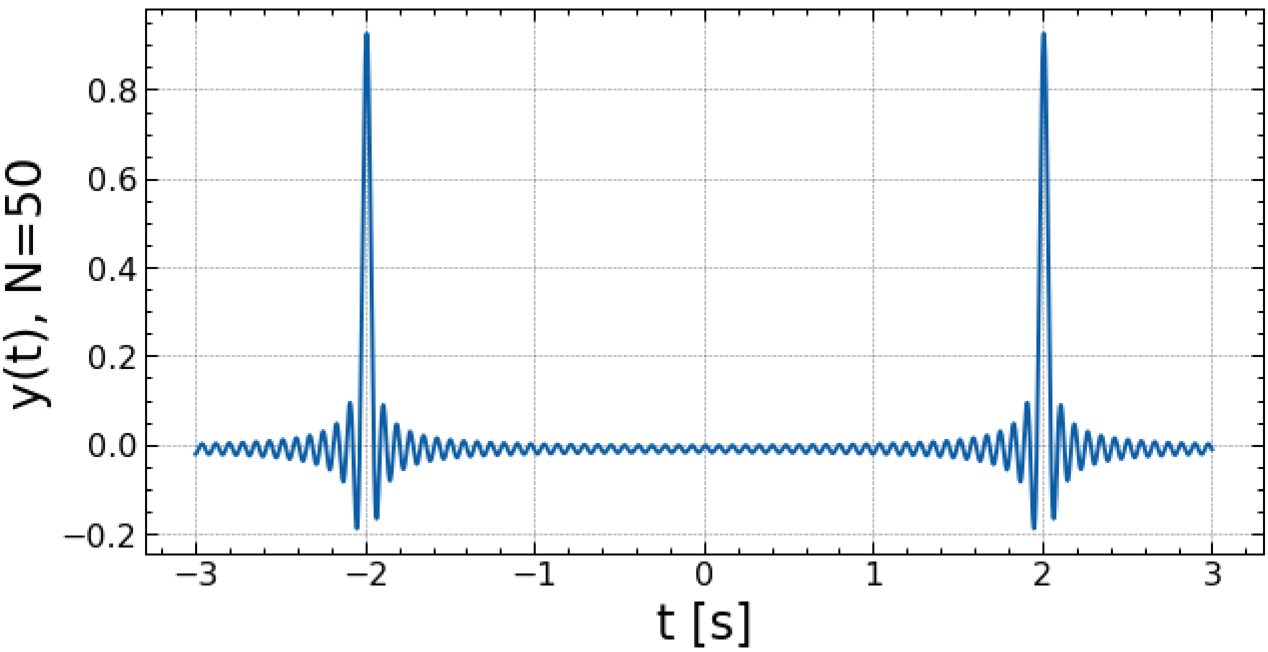
portanto, $y(t)$ equivale a:

$$y(t) = \sum_{k=-\infty}^{\infty} a_k \frac{1}{1 - j\frac{20}{k\pi}} e^{jk\omega_0 t}$$

Seja o grafico de $x(t)$ para $N = 50$



A saída $y(t)$ quando $x(t)$ for o sinal dente de serra aproximado com $N = 50$ e a resposta em frequência do sistema seja $H(jw) = \frac{1}{1 - j\frac{\omega_c}{\omega}}$ é representada pelo gráfico abaixo:



Observe que estamos usando N pequeno, então este resultado é apenas aproximado ao exato, resultado que se obtém apenas quando se soma os termos da equação que define $y(t)$, varrendo k de $-\infty$ a ∞

Observa-se em ambos os gráficos acima que $y(t)$ tem magnitude máxima no mesmo tempo t onde se observa mudanças bruscas de magnitude em $x(t)$, para valores de N tendendo ao infinito poderíamos dizer que esses picos de $y(t)$ coincidem com as descontinuidades de $x(t)$.

E sabemos que nos pontos de descontinuidades, ou no caso aproximado, nos pontos de mudança brusca de magnitude de $x(t)$, para construir este sinal é necessário mais componentes $k\omega_0$, ou seja, componentes com frequências maiores.

Como este filtro deixa frequências mais altas passarem e atenua frequências baixas, vemos na saída $y(t)$ magnitudes maiores quando $x(t)$ em um dado t, necessita de maiores frequências $k\omega_0$ para ser representado.

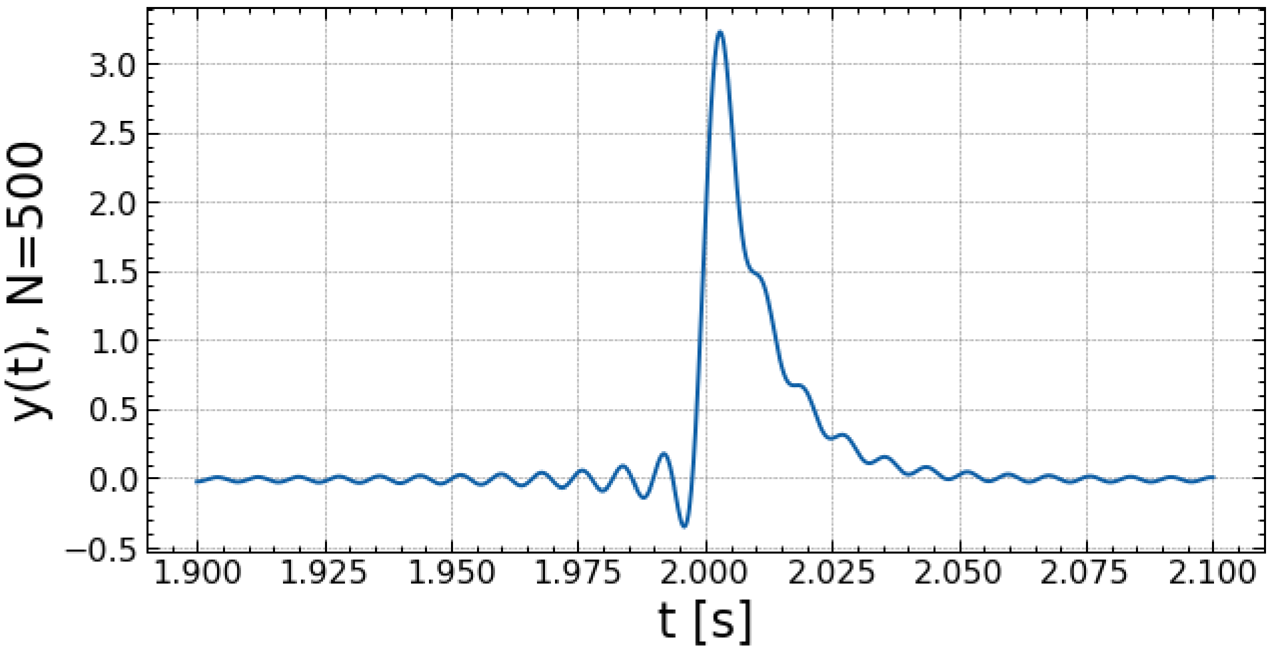
(g) Explique as diferenças entre o resultado obtido em **(f)** com a figura 3.

O resultado obtido na figura 3 é devido à onda dente de serra da figura 1 ser formada por muitas componentes de frequência $k\omega_0$, varrendo k de $-\infty$ a ∞, no caso de ambos os sinais serem sinais ideais. Ou a varredura é feita para valores elevados de k e esta é uma aproximação muito exata por série de Fourier.

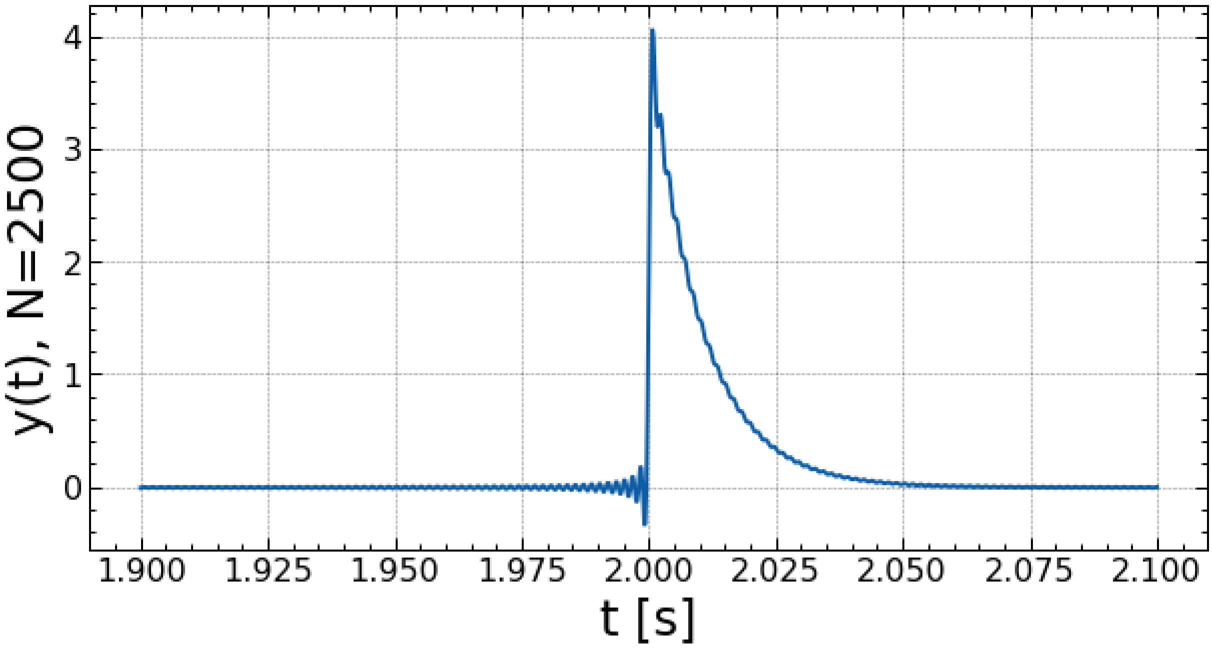
Já o resultado obtido em **(f)** possui poucas componentes, e é uma aproximação da forma de onda por serie de Fourier, calculada pela definição.

E para mostrar que, com valores de k cada vez maiores, mais nos aproximamos da forma de onda exibida na figura 3, abaixo alguns exemplos para N = 500, 2500 e 5000.

y(t), N = 500



y(t), N = 2500



y(t), N = 5000