Processamento de Sinais Digitais

Aplicações a EEG

Conceitos Básicos

Senoides são funcoes f do tipo:

$f: D \rightarrow Im$

· D indica o domínio

Apesar de teoricamente $D = \mathbb{R}$, na prática:

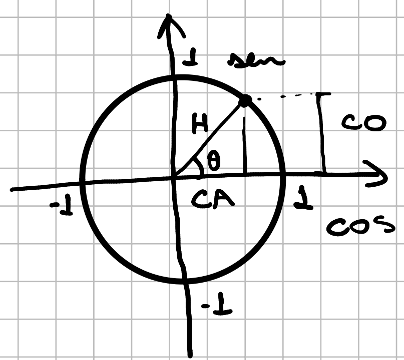
Se f é sen ou cos:

$$\cos(2\pi + s) = \cos(s)$$
$$\text{sen}(2\pi + s) = \text{sen}(s)$$

Então podemos dizer que $D = [0, 2\pi]$.

$Im = [0, 1]$

Lembremos tambem do ciclo trigonométrico e do que ele pode nos ensinar:

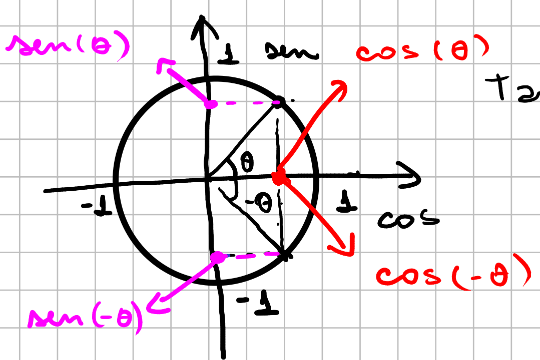


H: hipotenusa

$$\cos(\theta) = \frac{CA}{H}$$

$$\text{sen}(\theta) = \frac{CO}{H}$$

CA = cateto adjascente
CO = ——— oposto
H = hipotenusa



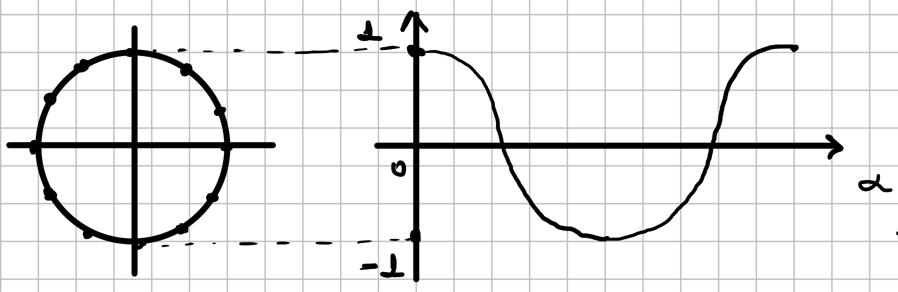
Também que:

$$\cos(\theta) = \cos(-\theta)$$

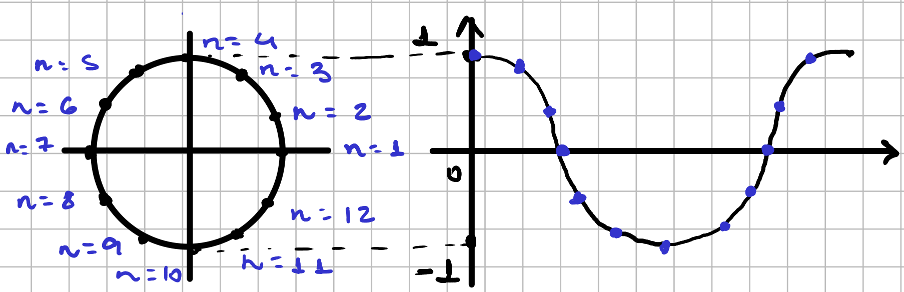
$$\text{sen}(\theta) = -\text{sen}(-\theta)$$

De modo geral, o ciclo trigonométrico é muito útil em Processamento de sinais.

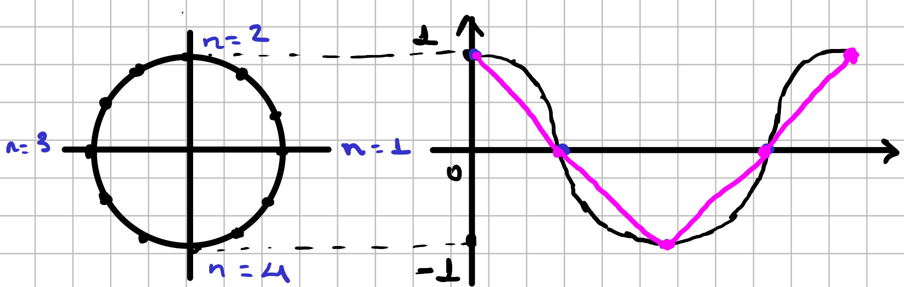
Representação gráfica de senoides



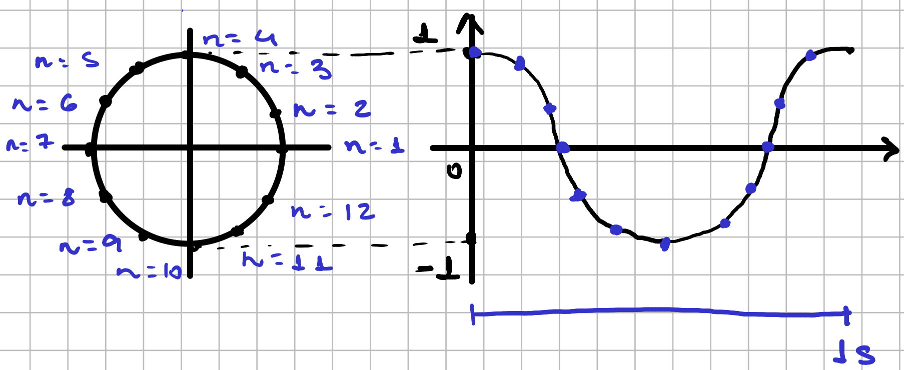
- Essa é uma representação de $\cos(\alpha)$ no intervalo $\alpha \in [0, 2\pi]$.
- É um sinal contínuo ou analógico, mas no mundo real não temos memória para guardar infinitos pontos.



- Os pontos em azul representam uma discretização.
- O computador liga os pontos mais próximos por linhas e assim obtemos uma representação do sinal.



- Em geral quanto mais pontos para um mesmo intervalo de tempo, melhor.
- A linha magenta mostra o quanto perdemos apenas com quatro pontos.



Frequência de amostragem

Se contarmos o número de pontos representativos do nosso sinal em 1 segundo temos a **freq. de amostragem**, denotada de f_s.

$$f_s = X\ Hz = X \cdot \frac{1}{s}$$

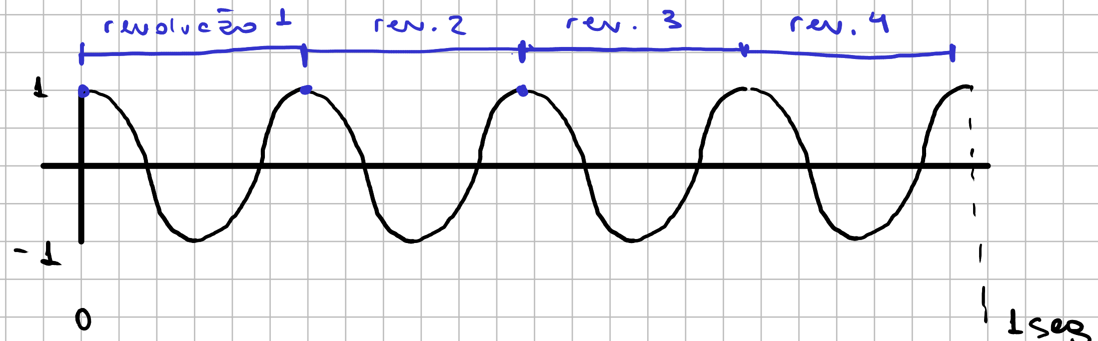
onde $X \in \mathbb{N}$.

O intervalo de tempo entre duas amostras é chamado período de amostragem, denotado de T_s

$$T_s = \frac{1}{f_s}$$

ex.: Se $f_s = 4\,Hz$ $\quad T_s = \frac{1}{4\,\frac{1}{s}} = \frac{0.25}{\frac{1}{s}} = \underline{\underline{0.25\ s}}$

frequência do sinal



Consideremos o sinal acima, como construí-lo?

- Um cosseno completa uma revolução completa quando retorna ao valor 1.
- O número de revoluções da senoide em um intervalo de tempo nos dá a frequência da senóide, denotaremos de $\underline{\underline{\omega}}$.

No ex. acima $\omega = 4\,Hz$

Como nosso sinal tem valor máximo 1, também podemos dizer que

$$\omega = \underbrace{2\pi \cdot \times 1}_{} \times 4$$

perímetro da circunferência $(2\pi . r)$ ↳ onde r é o raio.

$$= 2 \times 3{,}14 \times 4 = 25{,}12\ rad/s$$

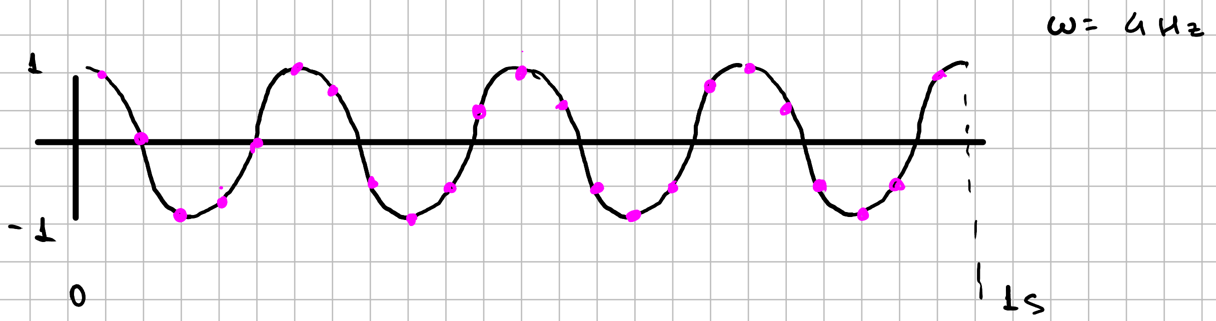
$$\begin{array}{r} {}^{1}\ \cancel{1} \\ 6{,}28 \\ \times\ \ 4 \\ \hline 25{,}12 \end{array}$$

Lembre que π é a razão entre o perímetro e o diâmetro da circunferência $\quad \pi = \frac{P}{D} = \frac{P}{2r} \div \underline{\underline{2\pi . r}}$

D = diâmetro
r = raio.

Como construir um sinal digital de uma dada frequência?

Resgatemos o sinal anterior:



Precisamos definir uma frequência de amostragem: 23 Hz

Nosso sinal terá $N = 23$ pontos indicados por $n = 1, 2, \dots, N$

Quando chegar a N deve ter realizado as 4 voltas em 2π, ou seja:

$$\cos(N x) = \cos(4 \cdot 2\pi)$$

Solucionando para x

$$N x = 4 \cdot 2\pi \Rightarrow x = \frac{4 \cdot 2\pi}{N}$$

Portanto,

$$\cos\left(n \frac{4 \cdot 2\pi}{N}\right)$$

Neste exemplo, como $N = fs$

$$\cos\left(n \frac{4\pi}{N}\right) \equiv \cos\left(n \cdot \frac{4\pi}{fs}\right)$$

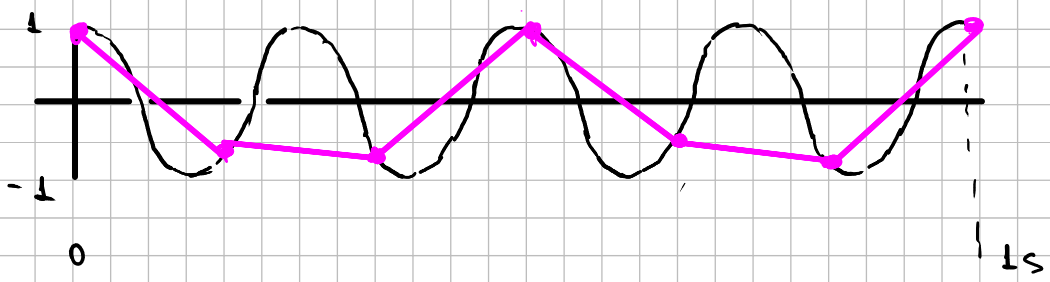
Dado essa construção, $\cos\left(n \cdot \frac{\omega}{fs} \cdot 2\pi\right)$, você pode expandir o sinal. Digamos que você queira um sinal de 60 segundos com frequência de $\omega = 4\,Hz$.

$$\cos\left(n \cdot \frac{\omega}{fs} \cdot 2\pi\right), \text{ onde } n = 1, 2, \dots, N = 60 \times fs.$$

! Use a rotina illustrative_examples para gerar sinais de diferentes frequências !

Aliasing e a Frequência de Nyquist.

Retomemos nosso sinal anterior.



$\omega = 4 Hz$

$f_s = 6 Hz$

A linha em magenta indica o que pode ocorrer quando escolhemos de maneira inadequada a frequência de amostragem dado o sinal. Neste caso, a representação indica uma senoide de frequência inferior ao sinal de interesse. Isso leva à má-interpretação da frequência do sinal de interesse e recebe o nome de aliasing.

Para evitar isso, o teorema de Nyquist determina que

$$f_s \geq 2 \cdot \max\{\omega\} + 1$$

onde $\max\{\omega\}$ é a maior frequência presente no sinal registrado.

Transformada de Fourier

Todo sinal com que iremos trabalhar pode ser representado por uma soma de senoides. A transformada discreta de Fourier (DFT) permite que saibamos quais as <u>frequências</u> dessas senoides assim como a <u>fase</u> dessas senoides.

Considere a <u>notação</u>:

$$W_N^{nk} = e^{-J.(n.k.\frac{2\pi}{N})} \quad , \text{ onde } J = \sqrt{-1}$$

A DFT em uma frequência k é dada por:

$$X[K] = \sum_{n=0}^{N-1} x[n].W_N^{nK} \quad , \text{ onde } x[n] \text{ é o seu sinal no ponto } n = 1, \dots, N$$

$$= \sum_{n=0}^{N-1} x[n].\, e^{-J(n.k.\frac{2\pi}{N})}$$

Lembre que a famosa fórmula de Euler indica que:

$$e^{J.\theta} = \cos(\theta) + J.\,\text{sen}(\theta)$$

Portanto,

$$= \sum_{n=0}^{N-1} x[n].\left(\cos(-n.k.\frac{2\pi}{N}) + J \sin(-n.k.\frac{2\pi}{N}) \right)$$

$$= \overbrace{\sum_{n=0}^{N-1} x[n].\cos \underbrace{(n.k.\frac{2\pi}{N})}_{*}}^{\dagger} - J.\overbrace{\sum x[n]\,\text{sen}\underbrace{(n.k\frac{2\pi}{N})}_{*}}^{\ddagger}$$

* Perceba a similaridade com a construção dos nossos sinais.

se $\vec{x} = [x_0, \dots, x_{N-1}]^T$ e $\vec{y}_{k,\cos} = [\cos 0.k.\frac{2\pi}{N}, \dots, \cos((N-1).k\frac{2\pi}{N})]^T$

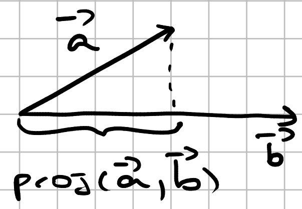
Portanto, o primeiro somatório é:

$\dagger$ $\langle \vec{x}, \vec{y}_{K,\cos} \rangle$

e o segundo

$\ddagger$ $\langle \vec{x}, \vec{y}_{K,\text{sen}} \rangle$

Lembre que a projeção de um vetor $\vec{a}$ sobre $\vec{b}$ é dado por:



$$proj(\vec{a},\vec{b}) = \frac{\langle \vec{a}, \vec{b} \rangle}{\| b \|}$$

onde $\langle \vec{a}, \vec{b} \rangle$ é o produto interno de $\vec{a}$ e $\vec{b}$ e $\|\vec{b}\|$ é a norma euclidiana do vetor $\vec{b}$

Portanto, podemos reescrever a DFT da seguinte forma:

$$= \underbrace{\sum_{n=0}^{N-1} x[n].\cos\left(n.k.\frac{2\pi}{N}\right)}_{A_1} - J.\underbrace{\sum x[n]\,sen\left(n.k\,\frac{2\pi}{N}\right)}_{A_2}$$

A_1 pode ser visto tanto como $\langle \vec{x}, \vec{y}_{k,cos} \rangle = \langle \vec{y}_{k,cos}, \vec{x} \rangle$

A_2 ______ $\langle \vec{x}, \vec{y}_{k,sin} \rangle = \langle \vec{y}_{k,sin}, \vec{x} \rangle$

Dado que o produto interno é comutativo.

Assim, reescrevendo a DFT:

$$= \|\vec{x}\|.proj(\vec{y}_{k,cos}, \vec{x}) + J\|\vec{x}\|.proj(\vec{y}_{k,sin}, \vec{x})$$

$$= \|\vec{x}\| \{ proj(\vec{y}_{k,cos}, \vec{x}) + J.proj(\vec{y}_{k,sin}, \vec{x}) \}$$

Perceba que $\|\vec{x}\| \geq 0$, já que:

$$\|\vec{x}\| = \sqrt{\sum_{n=0}^{N-1} x[n]}$$

se $\|\vec{x}\| > 0$ e $\langle \vec{y}, \vec{x} \rangle = 0 \rightarrow proj \langle \vec{y}, \vec{x} \rangle = 0$

Agora suponha que nosso sinal x seja:

$$x[n] = \cos\left(n.k.\frac{2\pi}{N}\right)$$

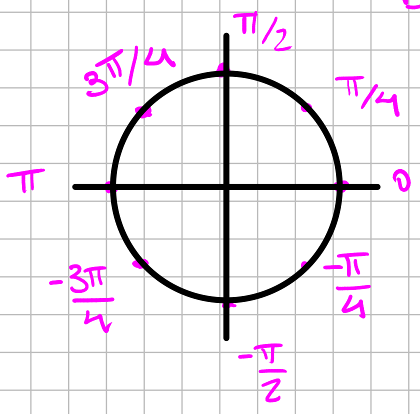
$2\cos(\alpha).\sin(\alpha) = \sin(2\alpha)$

Calculemos $proj(\vec{y}_{k,sin}, \vec{x})$:

$$= \sum_{n=0}^{N-1} \cos\left(n.k.\frac{2\pi}{N}\right).\sin\left(n.k.\frac{2\pi}{N}\right)^{*}$$

$$= \frac{1}{2} \underbrace{\sum_{n=0}^{N-1} \sin\left(2.n.k.\frac{2\pi}{N}\right)}_{B_1}$$

O resultado da expressão acima é sempre 0. Veja o exemplo abaixo:



$N = 8, \; k = 1$

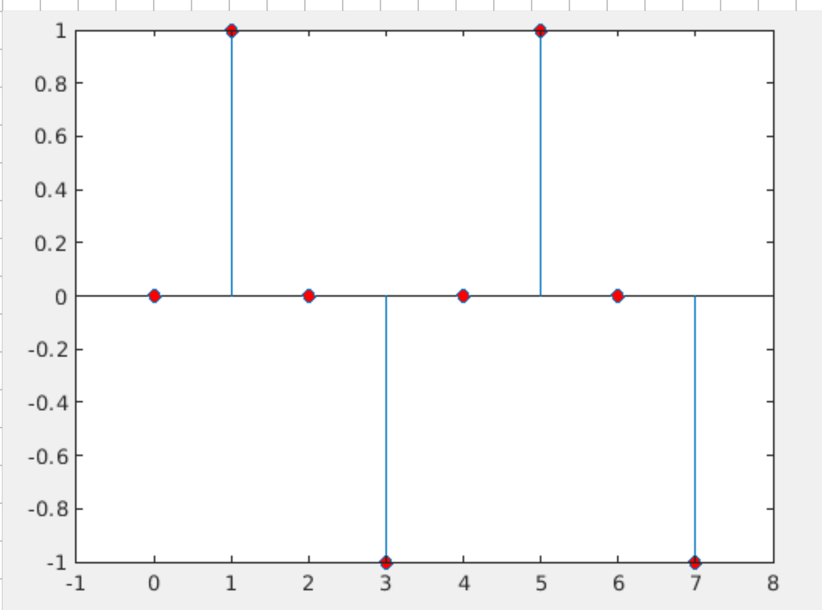
$$B_1 = \sum_{n=0}^{N/2} \sin\left(2 \cdot n \cdot \frac{2\pi}{N}\right) + \sum_{n=1}^{N/2-1} \sin\left(-2 \cdot n \cdot \frac{2\pi}{N}\right)$$

$$B_1 = \cancelto{0}{\sin(0)} + \cancelto{1}{\sin\left(\frac{1\pi}{2}\right)} + \cancelto{0}{\sin(\pi)} \dots + \cancelto{-1}{\sin\left(\frac{3\pi}{2}\right)} + \cancelto{0}{\sin(2\pi)} \dots + \cancelto{-1}{\sin\left(-\frac{\pi}{2}\right)} + \cancelto{0}{\sin(-\pi)} + \cancelto{1}{\sin\left(-\frac{3\pi}{2}\right)} \dots$$

$$= 1 - 1 - 1 + 1 = \underline{\underline{0}}$$

Esse código Matlab permite visualizar melhor o comportamento:

```
N = 8;
n = 0:N-1;
k = 1;
proj_parts = sin(2*n*k*2*pi/length(n));
proj_k = sum( sin(2*n*k*2*pi/length(n)) );
stem(n,proj_parts,'MarkerFaceColor', 'r')
xlim([-1 N])
```



→ Os valores no eixo vertical são os termos da soma.

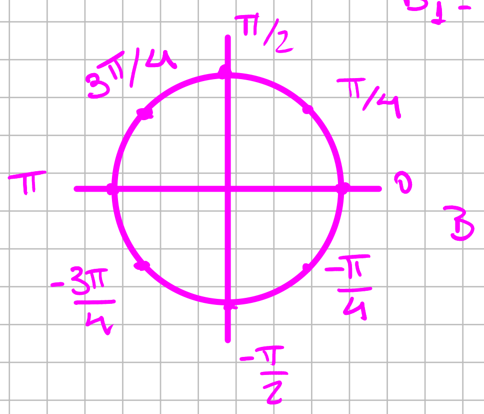
Agora, calculemos $\text{proj}(\vec{y}_{k,\cos}, \vec{x})$

$$\text{proj}(\vec{y}_{k,\cos}, \vec{x}) = \sum_{n=0}^{N-1} \cos\left(n \cdot k \cdot \frac{2\pi}{N}\right) \cdot \cos\left(n \cdot k \cdot \frac{2\pi}{N}\right)$$

$$= \sum_{n=0}^{N-1} \cos^2\left(n \cdot k \cdot \frac{2\pi}{N}\right)$$

Agora veja o resultado dessa expressão:

$N = 8$, $k = 1$



$$B_1 = \;= \sum_{n=0}^{N/2} \cos^2\left(n \cdot \frac{2\pi}{N}\right) + \sum_{n=1}^{N/2-1} \cos^2\left(-n \cdot \frac{2\pi}{N}\right)$$

$$= \cancel{\cos^2(0)}^{\,1} + \cos^2\left(\frac{\pi}{4}\right) + \cancel{\cos^2\left(\frac{\pi}{2}\right)}^{\,0} \ldots$$

$$+ \cos^2\left(\frac{3\pi}{4}\right) + \cancel{\cos^2(\pi)}^{\,1} \ldots$$

$$\cos^2\left(-\frac{\pi}{4}\right) + \cos^2\left(-\frac{\pi}{2}\right) + \cos^2\left(-\frac{3\pi}{4}\right)$$

$$= 1 + \frac{2}{4} + 0 + \frac{2}{4} + 1 \ldots$$

$$+ \frac{2}{4} + 0 + \frac{2}{4}$$

$$= 2 + 2 = 4$$

Ou seja, o resultado é $\frac{N}{2}$.

Veja com o código em matlab abaixo:

```
N = 8;
n = 0:N-1;
k = 1;
proj_parts = cos(n*k*2*pi/length(n)).^2;
proj_k = sum( cos([redacted]n*k*2*pi/length(n)).^2 );
stem(n,proj_parts,'MarkerFaceColor', 'r')
xlim([-1 N])
```

Então, podemos dizer que:

$$X[k] = \frac{N}{2} \quad p/ \quad x[n] = \cos\left(n \cdot k \cdot \frac{2\pi}{N}\right)$$

A transformada de Fourier é simétrica. Portanto, os k's tem frequências correspondentes negativas.
Somando a magnitude de k e k', sendo a última a correspondente negativa temos $\frac{N}{2} + \frac{N}{2} = \underline{\underline{N}}$.

Por isso nos gráficos representamos somente metade das transformada (correspondente às frequências positivas). Multiplicamos por 2 e dividimos por N!
Assim a potência corresponde à magnitude do sinal.
Veja o trecho do código:

```
def fft_spec(sig, fs):
    aux = sig.shape[0] % 2
    if aux != 0:
        sig = np.append(sig, [0])
    f_sig = (1 / sig.shape[0]) * np.fft.fft(sig)  # apply the normalization factor
    p_sig = np.absolute(f_sig)[0: sig.shape[0] / 2 + 1]
    l_f = p_sig.shape[0]
    p_sig[1:] = 2 * p_sig[1:]
    f_hz = (fs / sig.shape[0]) * np.array(range(l_f))
    return p_sig, f_hz
```

veja **

→ dividindo por N.

* → representando metade

→ multiplicando por 2.

A transformada de Fourier representa apenas de $[0, fs/2]$ em frequência.

Portanto, para identificar cada frequência em p_sig, podemos fazer uma regra de 3 simples

$$\frac{\frac{N}{2}}{i} = \frac{\frac{fs}{2}}{\omega_{Hz}}, \text{ onde } i = 0, 1, \ldots, \frac{N}{2}$$

$$\frac{N}{2} \cdot \omega_{Hz} = i\,\frac{fs}{2} \Rightarrow \omega_{Hz} = i\,\frac{\cancel{2}}{N} \cdot \frac{fs}{\cancel{2}} \Rightarrow \boxed{\omega_{Hz} = i\,\frac{fs}{N}}$$

* Perceba que se

$N = 2\ell + 1$, onde $\ell \in \mathbb{N}$ (ou seja, ímpar)

o que acontece quando $N = 9$, por exemplo, então

$$N = 9 \qquad \frac{N}{2} + 1 = \frac{9}{2} + 1 = 4.5 + 1$$

então o vetor irá de [0, 5.5]. O que não é possível de endereçar, portanto escolha sempre um N par. O mesmo para fs.

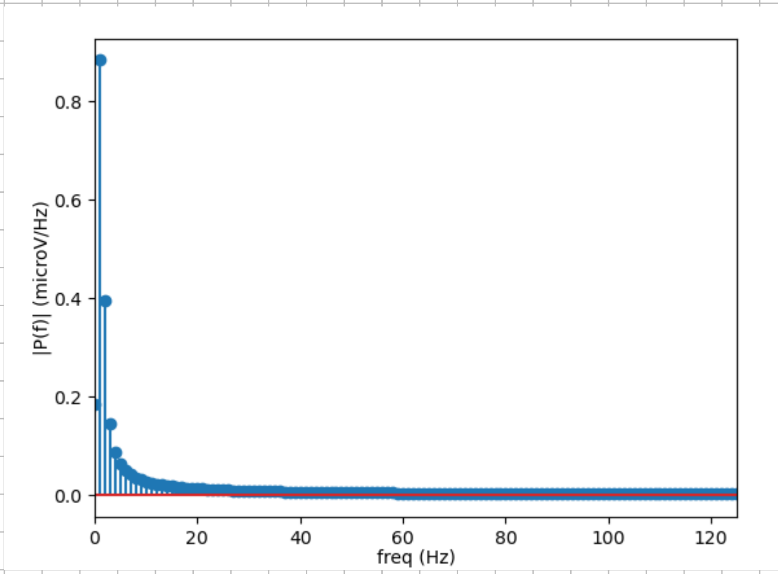
** Abrindo a função fft:

```
N = 8;
n = 0:N-1;
k = 1;
x = cos(n*k*2*pi/length(n));

X = zeros(length(n),1);

for l = 1:length(n)
    X(l) = sum(sum(x.*exp(-i*n.*(l-1)*2*pi/length(n))));
end

subplot(1,2,1)
stem(n,X)        → nossa função
xlim([0 N]); ylim([-1,5]);
subplot(1,2,2)
stem(n,fft(x)); xlim([0 N]); ylim([-1,5])
        ↳ função do software
```

Mundo Real

- Quando não sabemos a exata frequência do nosso sinal, corremos o risco de escolher fs de modo que nenhuma frequência de f_Hz corresponda à frequência do sinal. Neste caso, a energia do sinal que deveria estar concentrada acaba se espalhando:

! Em "illustrative_example.py" construa um sinal onde K = 1.35



espectro correspondente

- Obs.: No EEG isso é um grande problema. Pois dificulta o isolamento do sinal.
- Filtros podem eliminar parte de "energia" que se espalha!

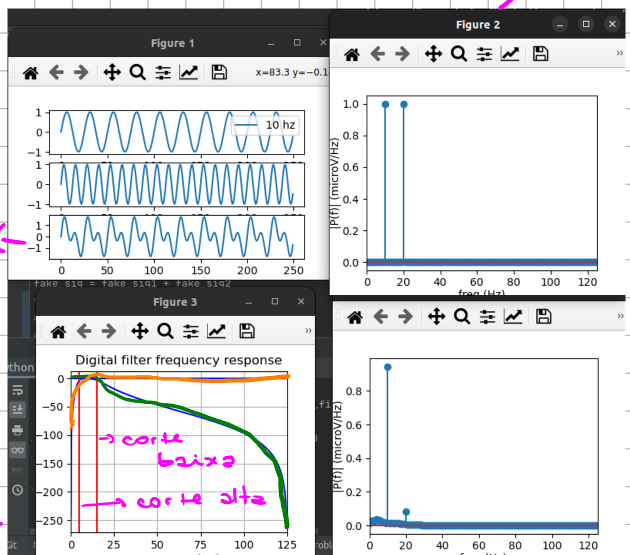
filtros

- Filtros são polinômios que atenuam ou amplificam sinais em frequências específicas.
  - Se um filtro é passa baixa, sua frequência de corte determina que frequências acima serão atenuadas
  - Se um filtro é passa alta, sua frequência de corte determina que frequências abaixo serão atenuadas.

→ espectro na entrada do filtro

sinal a ser filtrado ←

DB



frequências

espectro na saída.

- passa baixa
- passa alta

- A resposta em frequência do sinal mostra como o sinal é alterado.
- Sua escala é em DB's

$$DB = 10 \log_{10} \left(\frac{[\text{SAÍDA}]^*}{[\text{ENTRADA}]^*} \right)$$

*Amplitude do sinal na entrada e saída do filtro