

Transformada Discreta de Fourier (DFT)

A **Transformada Discreta de Fourier (DFT)** é uma ferramenta essencial no processamento de sinais digitais, permitindo a análise de sinais no domínio da frequência. Diferente da **Transformada de Fourier Contínua**, a DFT opera sobre sinais discretos e finitos.

Definição da DFT

A DFT de uma sequência $x[n]$ de comprimento N é definida como:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j 2\pi k n / N}, \quad k = 0, 1, \dots, N-1$$





onde:

- $X[k]$ representa os coeficientes espectrais do sinal.
- N é o número de pontos da DFT.
- $e^{-j 2\pi k n / N}$ é o fator de base exponencial complexo.

A **Transformada Inversa de Fourier** permite recuperar o sinal original:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j 2\pi k n / N}$$

Propriedades da DFT

A DFT possui diversas propriedades úteis:  **Linearidade**: A soma de dois sinais resulta na soma de suas DFTs.  **Periodicidade**: A DFT é periódica com período N .  **Simetria**: Para sinais reais, a DFT exibe simetria conjugada.  **Convolução**: A convolução no domínio do tempo equivale à multiplicação no domínio da frequência.

Aplicações da DFT

A DFT é amplamente utilizada em:  **Análise espectral** de sinais digitais.  **Compressão de dados** e processamento de áudio.  **Filtragem digital** e remoção de ruídos.  **Modulação e demodulação** em sistemas de comunicação.

Simulação Computacional

Podemos implementar a DFT em **Python** usando a biblioteca **NumPy**:

python

```
import numpy as np
import matplotlib.pyplot as plt

# Sinal de exemplo
```

```
N = 64
n = np.arange(N)
x = np.sin(2 * np.pi * 10 * n / N) # Sinal senoidal

# Cálculo da DFT
X = np.fft.fft(x)

# Plotando o espectro de magnitude
plt.plot(np.abs(X))
plt.title("Espectro de Magnitude da DFT")
plt.xlabel("Frequência")
plt.ylabel("Magnitude")
plt.show()
```

Esse código calcula a DFT de um sinal senoidal e exibe seu espectro de magnitude.