Universidade de São Paulo
Instituto de Física de São Carlos
Física Estatística Computacional

Projeto 1

Rafael Fernando Gigante - **NºUSP** 12610500

25 de março de 2023

INTRODUÇÃO

A transformada de Fourier discreta (DFT) é uma ferramenta matemática utilizada para transformar uma sequência de dados no domínio do tempo em uma representação equivalente no domínio das frequências.

A DFT é uma forma de analisar a frequência contida em um sinal que é discretamente medido. Ela é baseada na transformada de Fourier, que pode ser usada para decompor qualquer função periódica em uma simples soma de ondas senoidais e cossenoidais com diferentes amplitudes e frequências. A DFT utiliza essa ideia e aplica em sinais digitais, os quais são uma sequência finita de dados tomados em intervalos discretos de tempo. Podemos definir a DFT como

$$y_m = \frac{1}{N} \sum_{n=0}^{N-1} Y_n \exp\left(-2\pi imn/N\right) \tag{1}$$

$$Y_n = \sum_{n=0}^{N-1} y_m \exp\left(2\pi imn/N\right) \tag{2}$$

onde o índice m em y corresponde à tempos discretos $t_m = m\Delta t$ e o índice n em Y corresponde à frequências discretas $f_n = \frac{n}{N\Delta t}$.

1 TAREFA 1

Nesta tarefa foi feito um programa o qual realiza as transformadas de Fourier discreta conforme as equações (1) e (2). Neste caso, como recebemos um sinal no domínio do tempo, caracterizamos a equação (2) como a transformada direta e a (1) como a inversa. O sinal será recebido em um arquivo ”data.in”, o qual nós sabemos o número N de dados obtidos e o seu intervalo de tempo Δt, a série resultante após a primeira transformada é armazenada em um arquivo ”data.out”

Os códigos realizados para a geração do sinal e para o cálculo da transformada de Fourier podem ser encontrados na seção **Apêndice**.

2 TAREFA 2

Testaremos agora o programa realizado na **Tarefa 1** para diferentes séries da forma

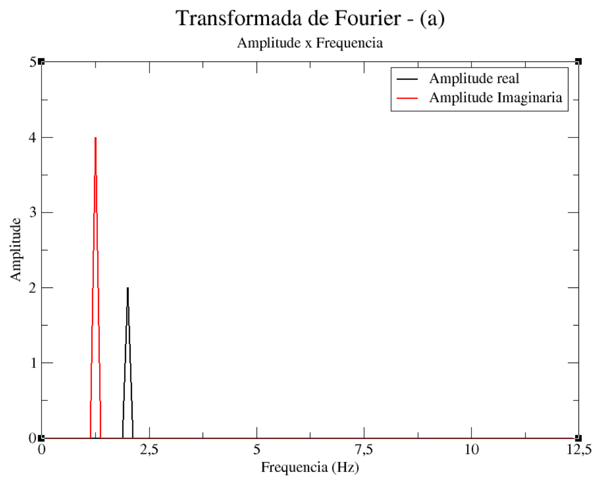
$$y_i = a_1 \cos\left(\omega_1 t_i\right) + a_2 \sin\left(\omega_2 t_i\right), t_i = i\Delta t, i = 1, ..., N \tag{3}$$

(a) $N = 200$, $\Delta t = 0.04$, $a_1 = 2$, $a_2 = 4$, $\omega_1 = 4\pi Hz$, $\omega_2 = 2.5\pi Hz$

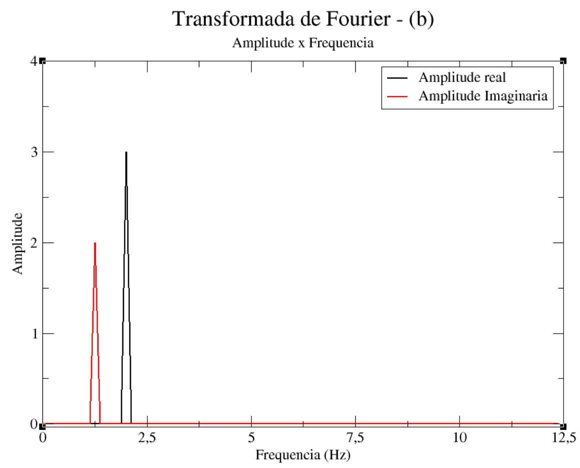
(b) $N = 200$, $\Delta t = 0.04$, $a_1 = 3$, $a_2 = 2$, $\omega_1 = 4\pi Hz$, $\omega_2 = 2.5\pi Hz$

(c) $N = 200$, $\Delta t = 0.4$, $a_1 = 2$, $a_2 = 4$, $\omega_1 = 4\pi Hz$, $\omega_2 = 0.2\pi Hz$

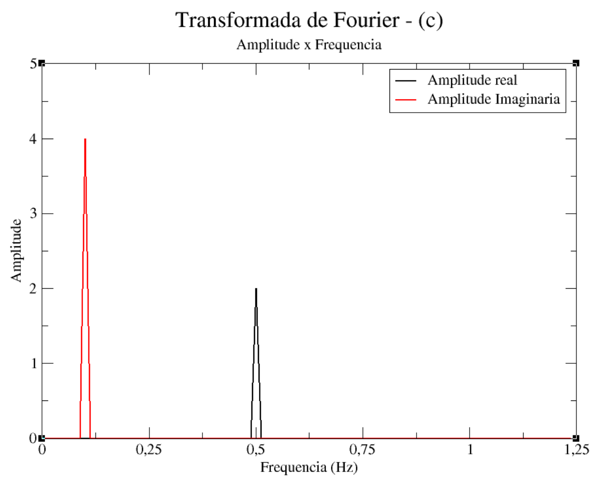
(d) $N = 200$, $\Delta t = 0.4$, $a_1 = 3$, $a_2 = 2$, $\omega_1 = 4\pi Hz$, $\omega_2 = 0.2\pi Hz$



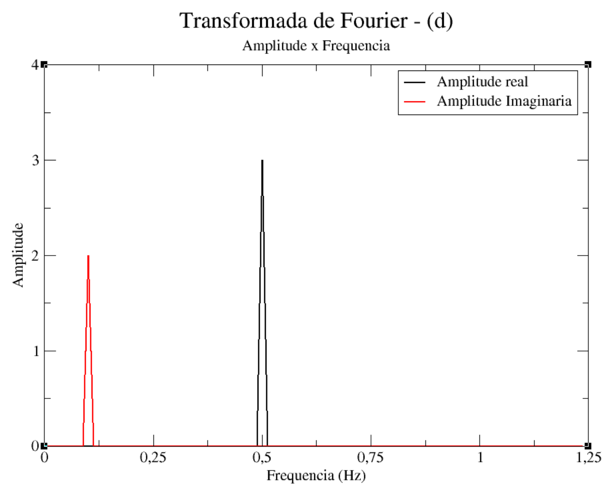
(a)



(b)



(c)



(d)

Figura 1: Transformadas de Fourier - Tarefa 2.

Ao realizarmos a Transformada de Fourier direta, o sinal real que recebemos inicialmente se transforma em um sinal complexo, portanto devemos separar a parte real da parte da imaginária. Com isso, na elaboração dos gráficos temos a amplitude real associada ao cosseno e a amplitude imaginária associada ao seno, isso pois $exp(\pm i\theta) = \cos\theta \pm i \sin(\theta)$. Ao analisar os gráficos, vemos que as amplitudes obtidas coincidem com as amplitudes a_1 e a_2 da série. Comparando diretamente os gráficos (a) e (b) que possuem as mesmas frequências ω_1 e ω_2, observa-se que as posições dos picos se mantêm a mesma, havendo apenas as diferenças de amplitude devido à diferença dos valores a_1 e a_2. O mesmo ocorre com as séries (c) e (d).

Agora comparando as séries (a) e (c) percebemos uma diferença nos valores de Δt e de ω_2. A diferença do valor do intervalo de tempo dos dados é muito importante, pois ela determina a frequência máxima onde podemos recuperar as componentes espectrais com a Transformada de Fourier. Tal frequência é conhecida como frequência de Nyquist e é definida como $f_{Nyquist} \equiv \frac{1}{2\Delta t}$. Para a série (a) com $\Delta t = 0.04$ temos que $f_{Nyquist} = \frac{1}{2 \cdot 0.04} = 12.5Hz$ com $\omega = 25\pi$, como as frequências de seno e cosseno estão abaixo desse valor, o sinal é completamente recuperado conforme ilustrado no gráfico. No entanto, para a série (c) temos $f_{Nyquist} = \frac{1}{2 \cdot 0.4} = 1.25Hz$, com $\omega = 2.5\pi$, apenas a parte senoidal do sinal está abaixo dessa frequência e vemos que ela é recuperada totalmente, enquanto a parte cossenoidal está incompleta e deveria coincidir com a mesma frequência que em (a). Isso ocorre pois essa onda pode ser descrita com a mesma precisão de duas formas diferentes, uma com a frequência verdadeira e outra com a frequência abaixo da frequência de Nyquist. Como a Transformada de Fourier é simétrica em relação à frequência de Nyquist, se o pico dessa onda é em $2Hz$ e ela está $0.75Hz$ acima de $f_{Nyquist}$, então também deve haver um pico refletido em $0.75Hz$ abaixo de $f_{Nyquist}$, sendo ele $0.5Hz$, conforme observado no gráfico.

De forma análoga ocorre essa diferença entre as séries (b) e (d).

3 TAREFA 3

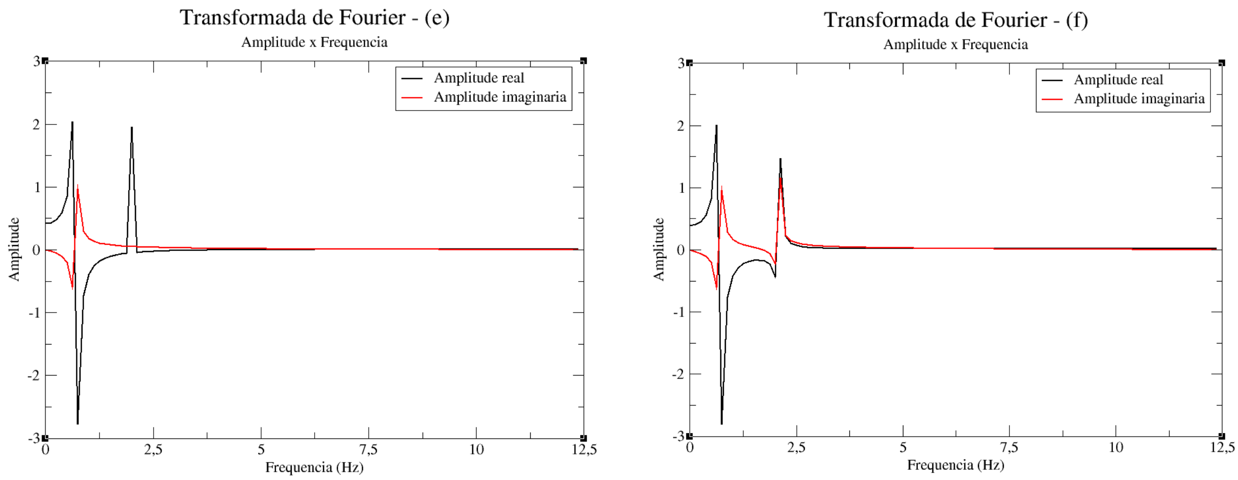
Novamente, conforme a equação (3) geramos as seguintes séries

(e) $N = 200$, $\Delta t = 0.04$, $a_1 = 2$, $a_2 = 4$, $\omega_1 = 4\pi Hz$, $\omega_2 = 1.4\pi Hz$

(f) $N = 200$, $\Delta t = 0.04$, $a_1 = 2$, $a_2 = 4$, $\omega_1 = 4.2\pi Hz$, $\omega_2 = 1.4\pi Hz$

As duas séries desta tarefa possuem ondas com frequência abaixo de $f_{Nyquist} = 25\pi$, porém ao observar o gráfico encontramos um comportamento estranho em ambas.

Ao olhar com cuidado o intervalo de tempo Δt notamos que ele não combina perfeitamente com as frequências das componentes de Fourier, isso pois $f_n = 0.7Hz = \frac{n}{200 \cdot 0.04} \rightarrow n = 5.6$. Nos gráficos pode-se observar picos nas frequências das ondas utilizadas para construir o sinal, porém também existem amplitudes pequenas de outras frequências diferentes. Se a frequência contida no sinal não corresponde com uma frequência discreta na forma $f_n = \frac{n}{N \cdot \Delta t}$, então a transformada é forçada a representar o sinal como uma soma das componentes em todo o intervalo f_n. Por isso vemos outras frequências espalhadas nos gráficos das séries (e) e (f).



(a) (b)

Figura 2: Transformadas de Fourier - Tarefa 3.

4 TAREFA 4

Nesta tarefa realizamos a transformada de Fourier inversa da série obtida na Figura1 (a). Como pode-se visualizar na Figura 3, temos o sinal original em uma linha preta e os pontos obtidos do sinal recuperado. Vemos que ambos os sinais estão superpostos, significando que a transformada inversa funcionou e conseguimos ter o nosso sinal de volta.

5 TAREFA 5

Vemos que o cálculo da DFT utilizando as equações (1) e (2) é computacionalmente caro. Cada soma para uma determinada componente da frequência envolve N termos e temos N componentes de frequência. Portanto, o número total de operações é da ordem de N^2. Utilizando a função intrínseca CPU_TIME do Fortran, o código da Tarefa 1 foi testado para diferentes valores de N recebidos do sinal. Ao plotar os dados obtidos, vemos que conseguimos fazer um ajuste parabólico, conforme a Figura 4. Portanto, podemos concluir que o tempo computacional é proporcional à N^2.

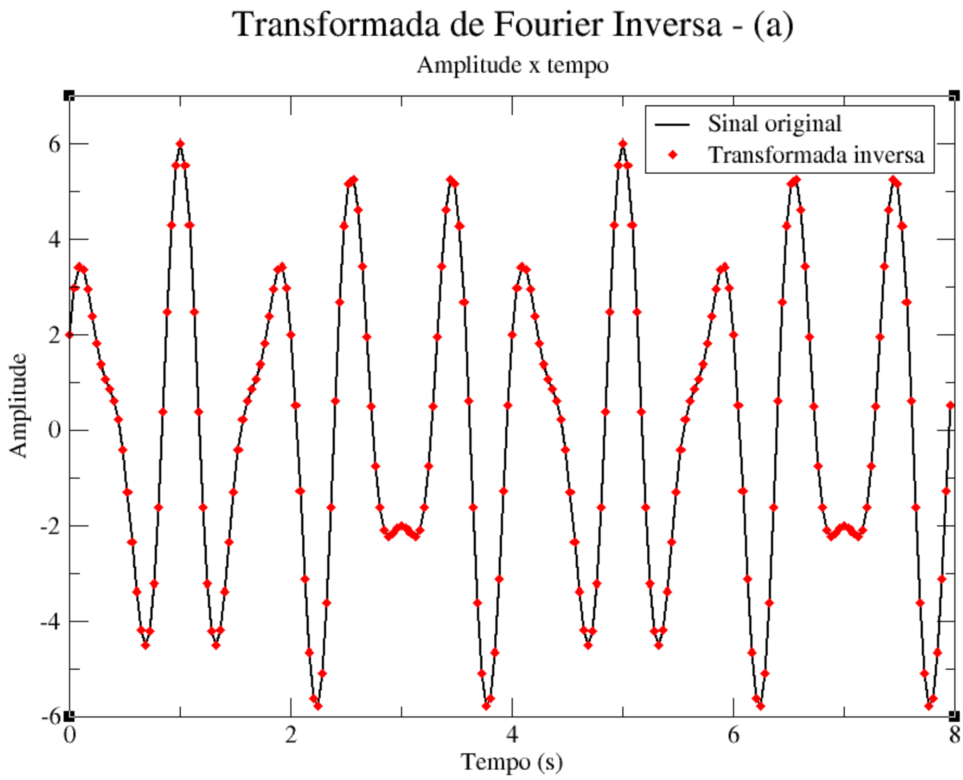


Figura 3: Transformada de Fourier Inversa - Tarefa 4.

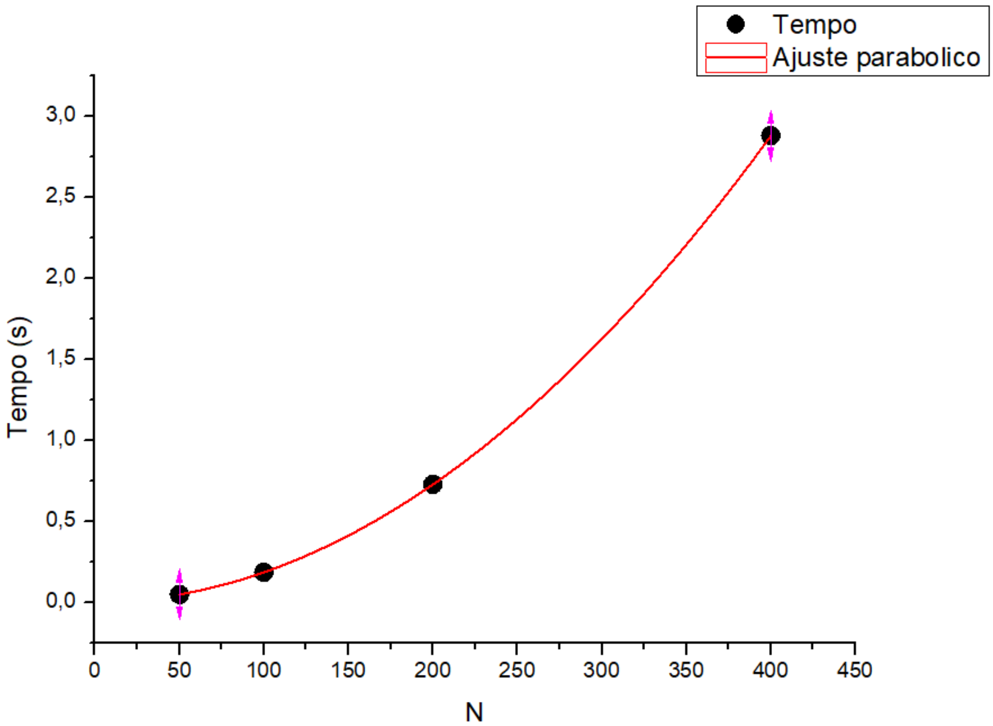


Figura 4: Tempo computacional x N - Tarefa 5

6 APÊNDICE

1. **Código gerador do sinal**

```
program signal
    implicit none
    integer :: N, a1, a2, i
    real(8), parameter :: pi=acos(-1.d0)
    real(8) :: yi, w1, w2, dt

    open(1, file='data.in')
    !open(2, file='sinal_real.in')

    read(*,*) N, dt, a1, a2, w1, w2

    do i=0, N-1
        yi = a1*cos(w1 * pi * i * dt) + a2*sin(w2 * pi* i * dt)
        write(1,*) (i * dt), complex(yi,0.0d0)
        !write(2,*) (i * dt), yi
    end do

    close(1)
end program signal
```

2. **Código da Transformada de Fourier Direta e Inversa**

```fortran
program fourier_transform
    implicit none
    integer :: N, j, k
    real(8) :: pi = acos(-1.0d0), dt, aux
    complex(8) :: i = (0.0d0, 1.0d0), yy
    complex(8), dimension(0:400) :: y_j, Y_k

    read(*,*) N, dt
    open(1, file='data.in', status='old')
    open(2, file='tarefa-2-freq-real-12610500.out')
    open(3, file='tarefa-2-freq-imag-12610500.out')
    open(4, file='data.out')
    read(1,*) (aux, y_j(j), j=0, (N-1))

    !Transformada de Fourier Direta
    do k=0, (N/2 - 1)
       yy = 0.d0
        do j=0, (N-1)
            yy = yy + y_j(j) * exp((2.0d0 * pi * i * j * k) / N)
        end do
        Y_k(k) = yy * 2.d0/N
        write(2,*) (k / (N * dt)), real(Y_k(k))
        write(3,*) (k / (N * dt)), aimag(Y_k(k))
        write(4,*) (k / (N * dt)), (Y_k(k))
    end do
    !Transformada de Fourier Inversa
    do j=0, (N - 1)
        y_j(j) = 0
        do k=0, (N/2 -1)
           y_j(j) = y_j(j) + Y_k(k) * exp((2.0d0 * pi * i * j * k) / N)
        end do
    end do
end program
```