Projeto 1 - Física Estatística Computacional

Edmur C. Neto - nº12558492

1 Introdução

Neste projeto buscamos aplicar o método discretizado da Transformada de Fourier, para o estudo das frequências embutidas em um sinal qualquer.

A transformada de Fourier é calculada da seguinte forma no contínuo:

$$Y(f) = \int_{-\infty}^{\infty} y(t)e^{i2\pi t}dt = \int_{-\infty}^{\infty} y(t)e^{iwt}dt \tag{1}$$

Porém no código teremos de discretizar a equação em um somatório da seguinte forma:

$$Y_n = \sum_{n=0}^{N-1} y_m e^{2\pi imn/N} \tag{2}$$

Onde o índice m corresponde à discretização do tempo em $t_m = m\Delta t$ e o índice n corresponde à discretização das frequências relacionadas à transformada $f_n = \frac{n}{(N\Delta t)}$.

Além disso, podemos calcular a transformada inversa de Fourier com o intuito de voltar ao sinal utilizado, com a utilização do seguinte somatório:

$$y_m = \frac{1}{N}\sum_{n=0}^{N-1} Y_n e^{-2\pi imn/N} \tag{3}$$

Com os mesmos índices do somatório anterior.

2 Desenvolvimento e Resultados

2.1 Tarefa - I

Na primeira tarefa basicamente criamos um programa no qual recebe um banco de dados contendo N pares de informações de um sinal, sendo estas organizadas em (tempo (s), amplitude (y_m)), das quais só utilizaremos a amplitude para realizar a transformada do sinal. A recepção do sinal contido no arquivo "data.in" foi feita da seguinte forma:

```
!Dados do sinal

        do l = 0,1000
                read(1,*, end=1) a, y_sinal(l)
        enddo
1       continue
        N = l
```

Dessa forma, os dados da amplitude serão salvos no vetor "ysinal", o qual passa a ter N casas. A próxima parte calcula, com a aplicação da função **Do**, o somatório para cada ponto do sinal, ou seja, um ponto da amplitude da transformada é o resultado do somatório do produto de todos os pontos do sinal com a exponencial da equação (2). O código para esse cálculo é tal:

```
!Transformada de Fourier do sinal

do k = 0, ((N/2)-1)
        f(k) = k/(N*dt)
        y_t = (0,0)
        do j = 0, N-1
                y_t = y_t + y_sinal(j)*exp(2.d0*pi*i*k*j/N)
        enddo
        y_transf(k) = y_t*2.d0/N
        write(3,*) f(k),  real(y_t*2.d0/N)
        write(4,*) f(k),  aimag(y_t*2.d0/N)
enddo
```

Nesse código também será calculado a frequência para cada ponto da transformada, por meio da equação $f(k) = k/N * \Delta t$, alocando o valor no vetor f(k), desse modo, construímos um espaço de frequência Y_n x f_n. Porém, como a exponencial é complexa a amplitude desse espaço também será, portanto, utilizamos uma iteração até (N/2) para gerar (N/2) pares de números, então dividimos a amplitude entre Real e Imaginário. Desse modo, destinguimos a frequência correspondida à função cosseno e seno do sinal. Salvamos os dados da frequência e sua correspondente amplitude em um arquivo Real e Imag..

Além disso, neste código realizamos a transformada inversa do sinal, a qual consiste em calcular o somatório da equação (3) afim de comparar com o sinal oferecido e validar a transformada. O código para isto está disposto abaixo:

```
!Transformada Inversa de Fourier do sinal

do k = 0, N-1
        y_i = (0,0)
        do j = 0, (N/2-1)
                y_i = y_i + y_transf(j)*exp(-2.d0*pi*i*k*j/N)
        enddo
        write(2,*) (k*dt) , real(y_i)
enddo
```

O somatório é realizado com base nas amplitude encontradas no código anterior com o intuito de voltar ao sinal oferecido. Os dados das amplitudes calculadas e os tempos respectivos são alocados em um arquivo "data.out".

2.2 Tarefa-II

Nessa tarefa iremos utilizar o programa anterior para calcular a transformada de Fourier de alguns sinais dados, os sinais seguem a seguinte equação:

$$y_i = a_1 \cos(w_1 t_i) + a_2 \sin(w_2 t_i), t_i = i\Delta t, i = 1, ..., N \tag{4}$$

Então foi preciso fazer um programa, cujo está denominado "sinal.f", para gerar os pontos a serem lidos. O código utilizado foi:

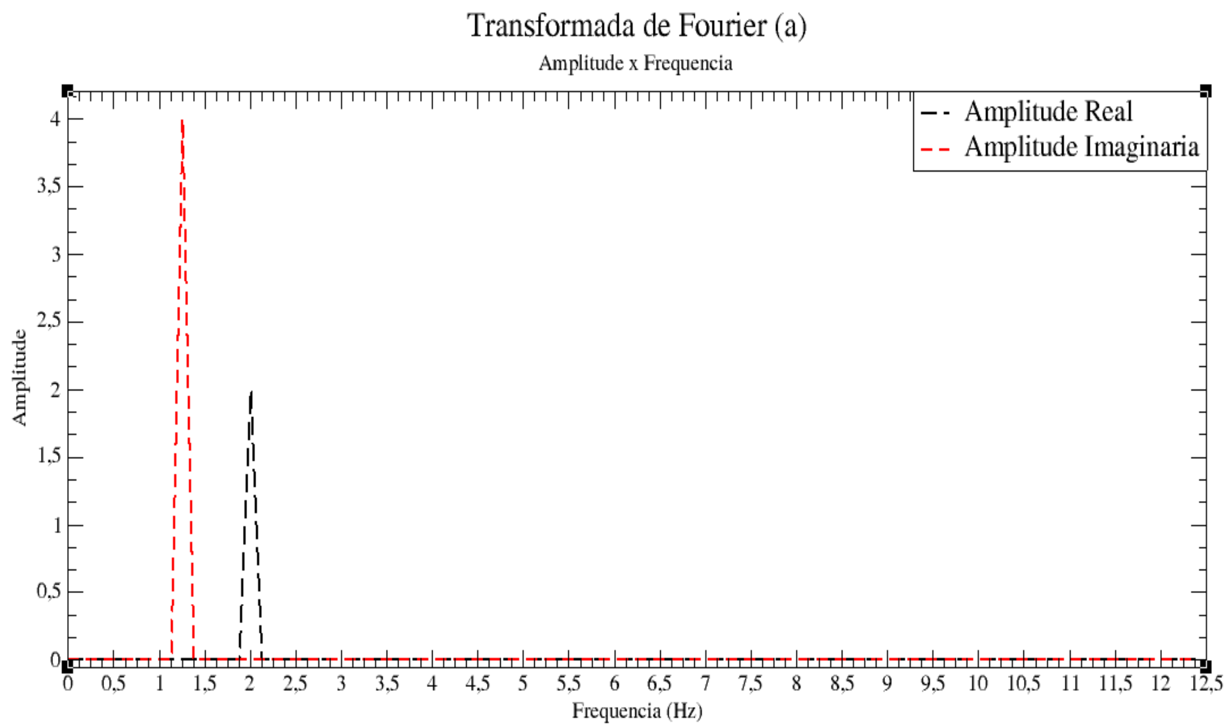
```
Program sinal.f

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

     w1 = w1*pi
     w2 = w2*pi
     t = 0.d0

     do i = 0, (N-1)
             t = i*dt
             y1 = a1*dcos(w1*t) + a2*dsin(w2*t)
             write(1,*) t,y1
     enddo
     close(1)
```

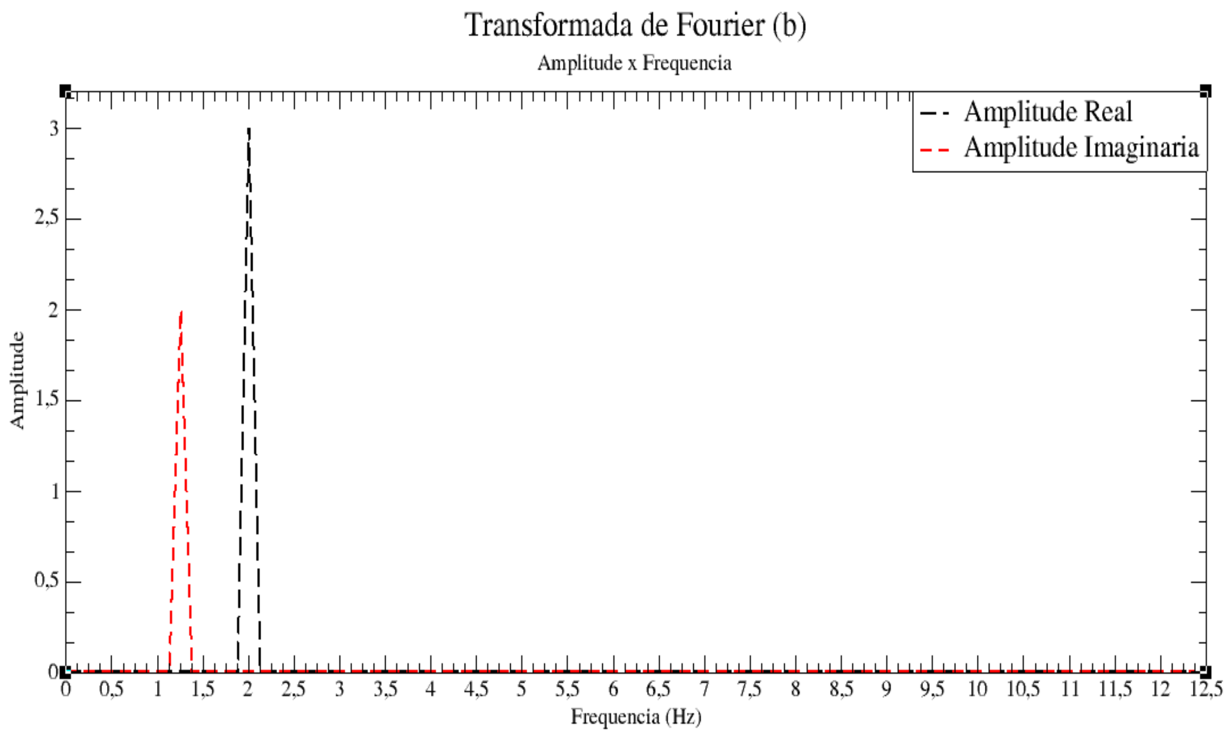
O programa recebe os parâmetros da equação (4) pelo terminal e realiza um loop calculando os valores das amplitudes de acordo com o tempo, construindo um espaço de amplitude e tempo (y_m x t) e deixando os valores no arquivo "data.in".

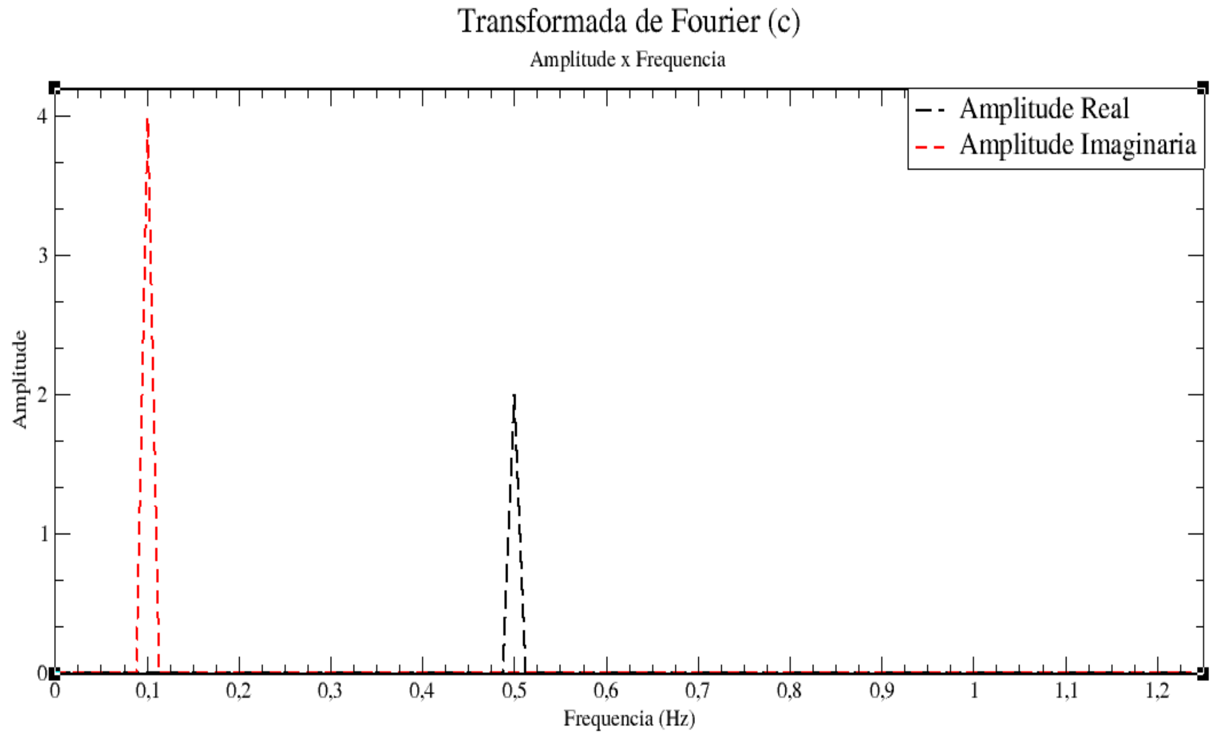
Agora serão aplicados os parâmetros dados abaixo para o cálculo da transformada.

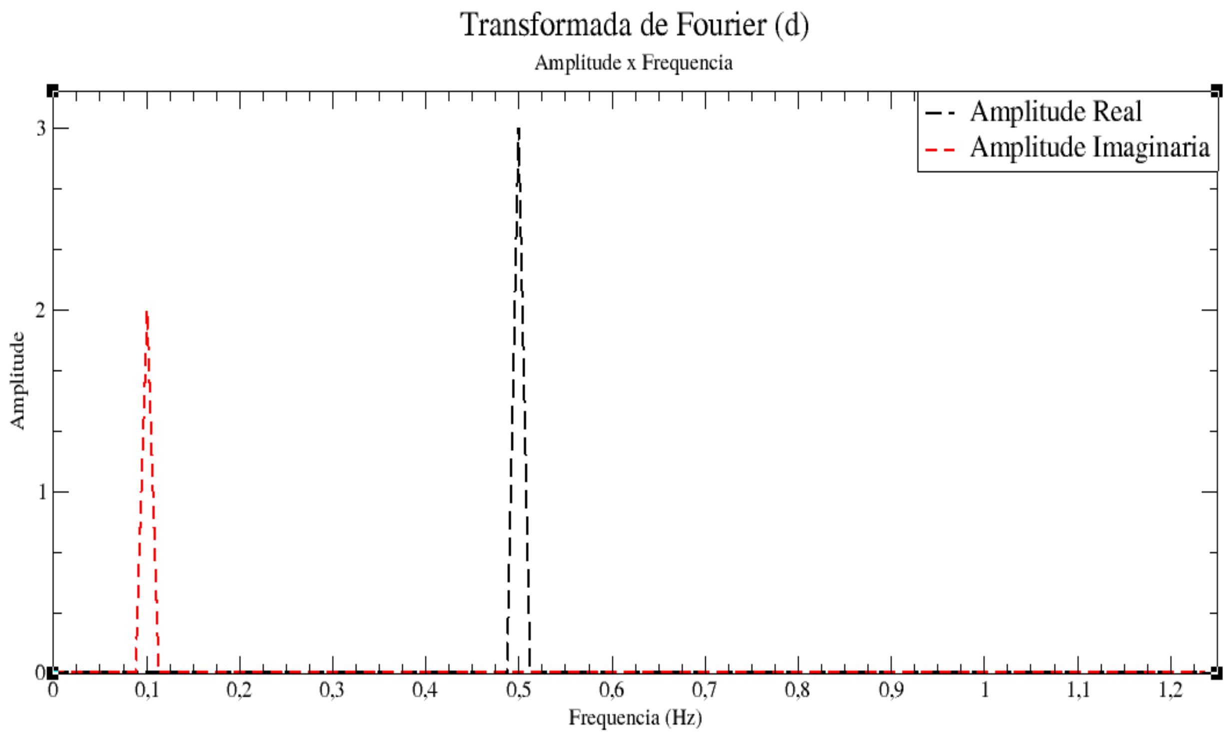
$$(a) N = 200, \Delta t = 0.04, a_1 = 2, a_2 = 4, \omega_1 = 4\pi Hz, \omega_2 = 2.5\pi Hz$$
$$(b) N = 200, \Delta t = 0.04, a_1 = 3, a_2 = 2, \omega_1 = 4\pi Hz, \omega_2 = 2.5\pi Hz$$
$$(c) N = 200, \Delta t = 0.4, a_1 = 2, a_2 = 4, \omega_1 = 4\pi Hz, \omega_2 = 0.2\pi Hz$$
$$(d) N = 200, \Delta t = 0.4, a_1 = 3, a_2 = 2, \omega_1 = 4\pi Hz, \omega_2 = 0.2\pi Hz$$

Encontramos os seguintes gráficos de Amplitude x Frequência a partir dos dados:


Transformada de Fourier (a)
Amplitude x Frequencia
Amplitude Real
Amplitude Imaginaria
Amplitude
Frequencia (Hz)


Transformada de Fourier (b)
Amplitude x Frequencia
Amplitude Real
Amplitude Imaginaria
Amplitude
Frequencia (Hz)


Transformada de Fourier (c)
Amplitude x Frequencia
Amplitude Real
Amplitude Imaginaria
Amplitude
Frequencia (Hz)
0
1
2
3
4
0
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
1
1,1
1,2


Transformada de Fourier (d)
Amplitude x Frequencia
Amplitude Real
Amplitude Imaginaria
Amplitude
Frequencia (Hz)
0
1
2
3
0
0,1
0,2
0,3
0,4
0,5
0,6
0,7
0,8
0,9
1
1,1
1,2

Observamos que nos sinais (a) e (b), a frequência de maior amplitude é justamente $f = \frac{\omega}{2\pi}$, no caso real $\omega = \omega_1$ e para o caso imaginário $\omega = \omega_2$.

Porém, nos sinais (c) e (d) a frequência real não correspondeu a equação, isso é explicado por meio da frequência de Nyquist, $f_N = 1/2\Delta t$, esta frequência limita o espaço Amplitude x Frequência, portanto, como ocorreu no caso de (c) e (d), não conseguimos representar frequências maiores que $f_N = 1.25Hz$.

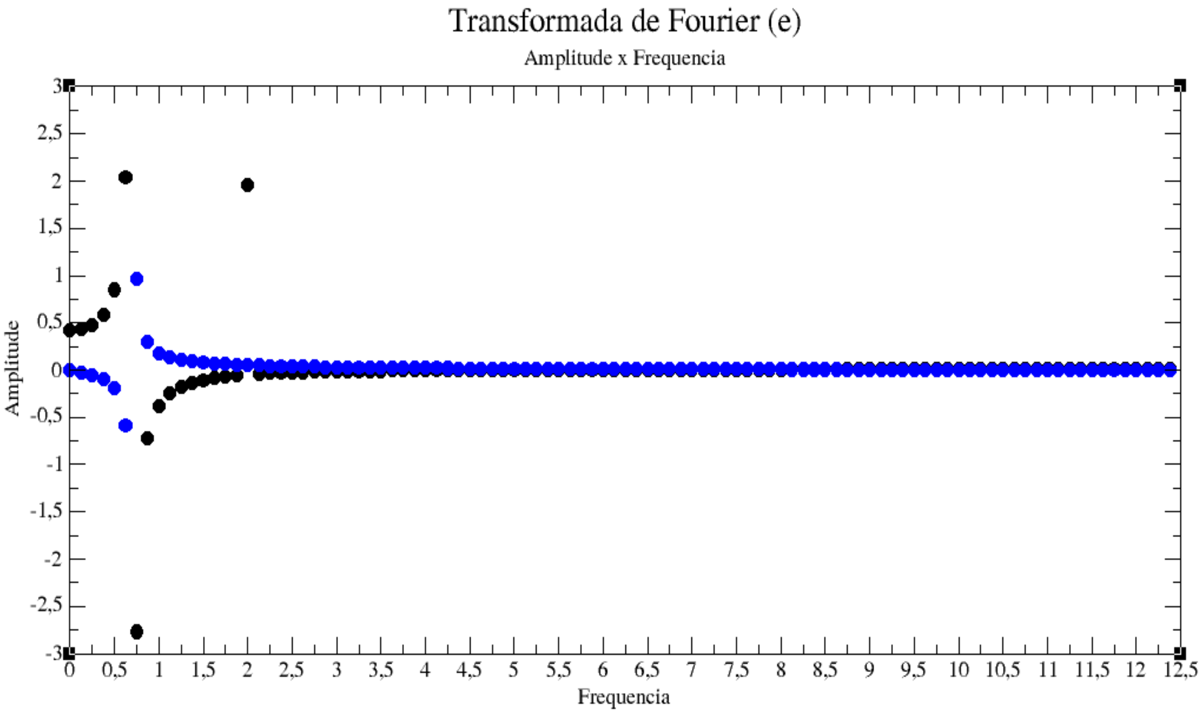
Para estimar se os dados estão corretos podemos calcular onde será o pico "falso"e comparar com o gráfico, a partir da equação $f_{verdadeiro} - f_N = f_N - f_{falso}$, sendo assim, o pico "falso"estará em f = 0.5 Hz, como vimos no gráfico, validando o código utilizado.

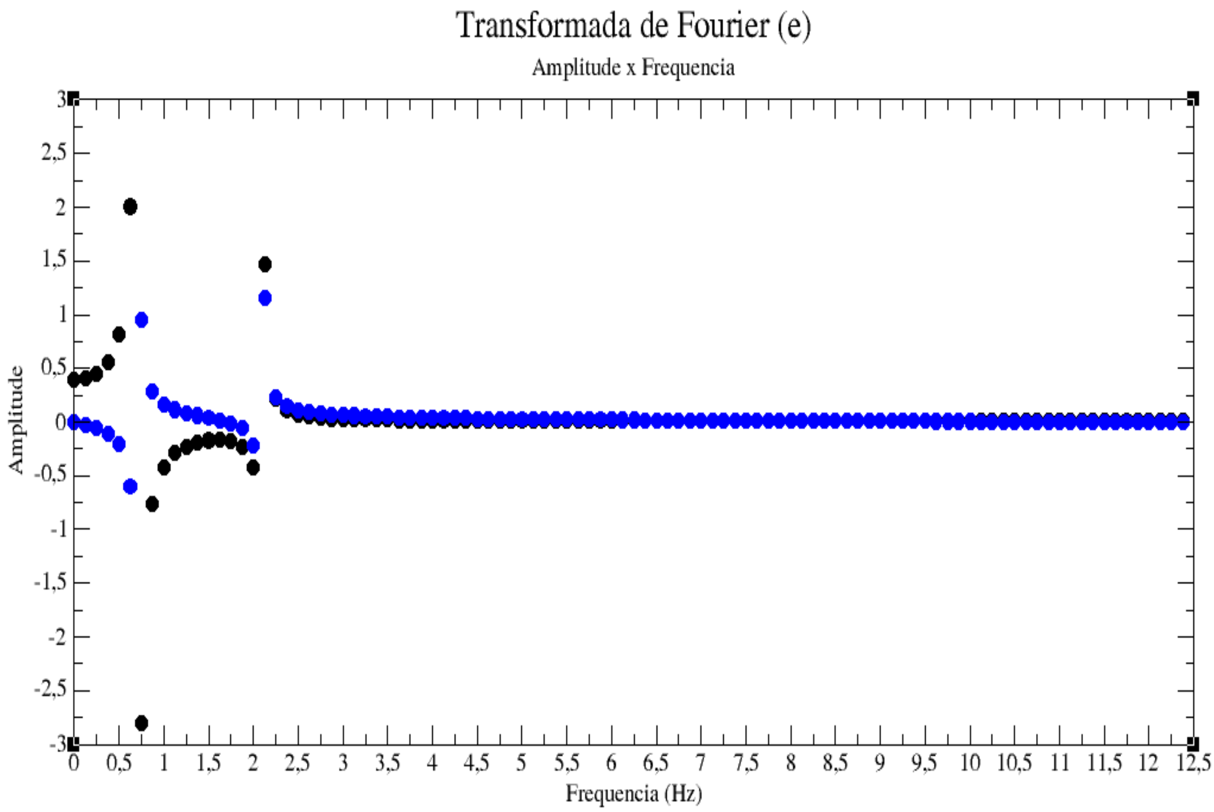
2.3 Tarefa III

Nessa tarefa aplicamos a transformada de Fourier em outros dois sinais:

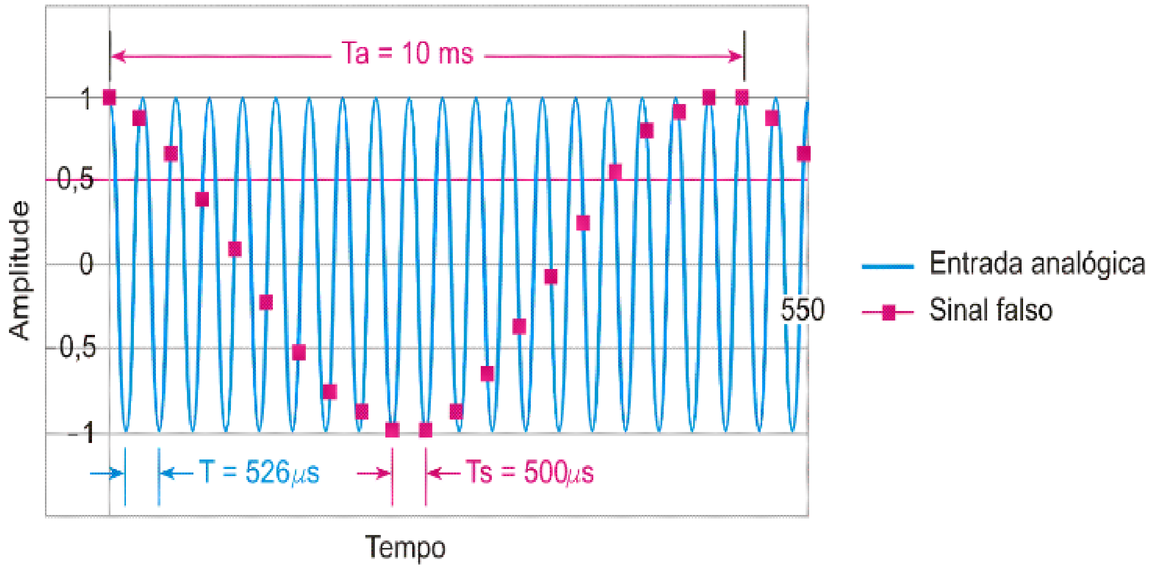
$$(e) N = 200, \Delta t = 0.04, a_1 = 2, a_2 = 4, \omega_1 = 4\pi Hz, \omega_2 = 1.4\pi Hz$$
$$(f) N = 200, \Delta t = 0.04, a_1 = 2, a_2 = 4, \omega_1 = 4.2\pi Hz, \omega_2 = 1.4\pi Hz$$

Obtemos os seguintes gráficos:





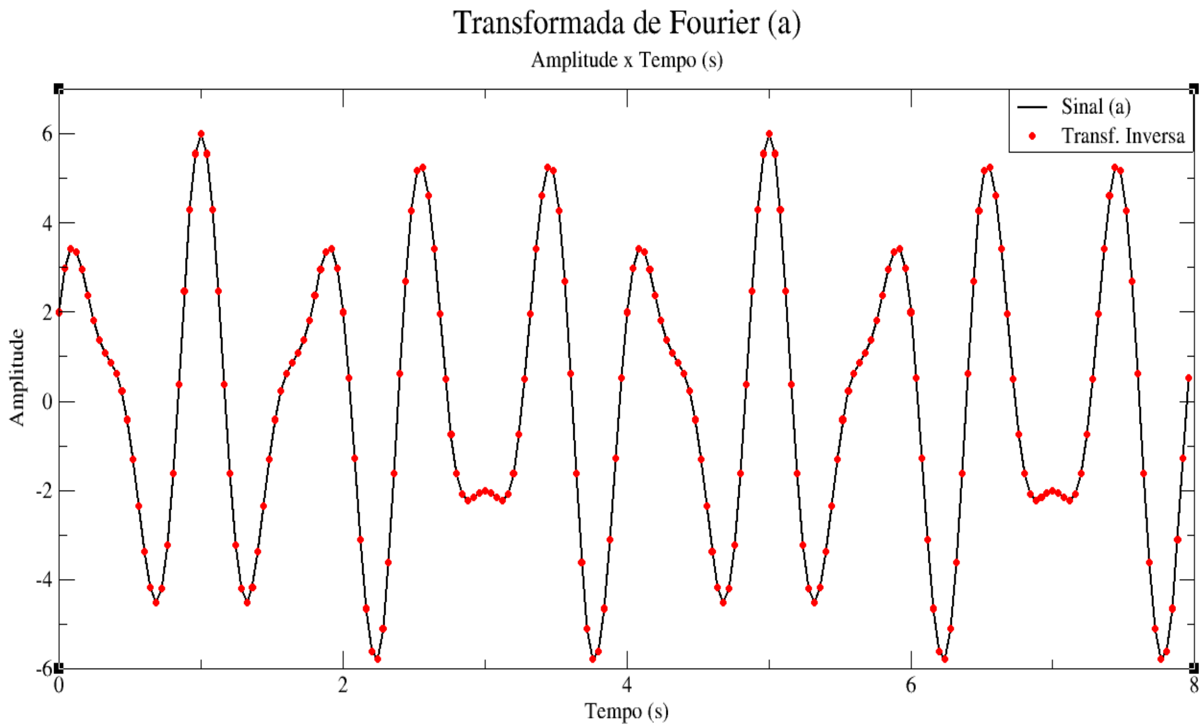
Observamos um espalhamento nos pontos da frequência do sinal diferente do que vimos nos sinais acima, isto é devido ao intervalo de tempo de recolhimento dos dados, pois se o intervalo for suficientemente grande, estaremos recolhendo sinais falsos e, com isso, teremos mais que uma frequência correta, assim como esta exemplificado na imagem abaixo:

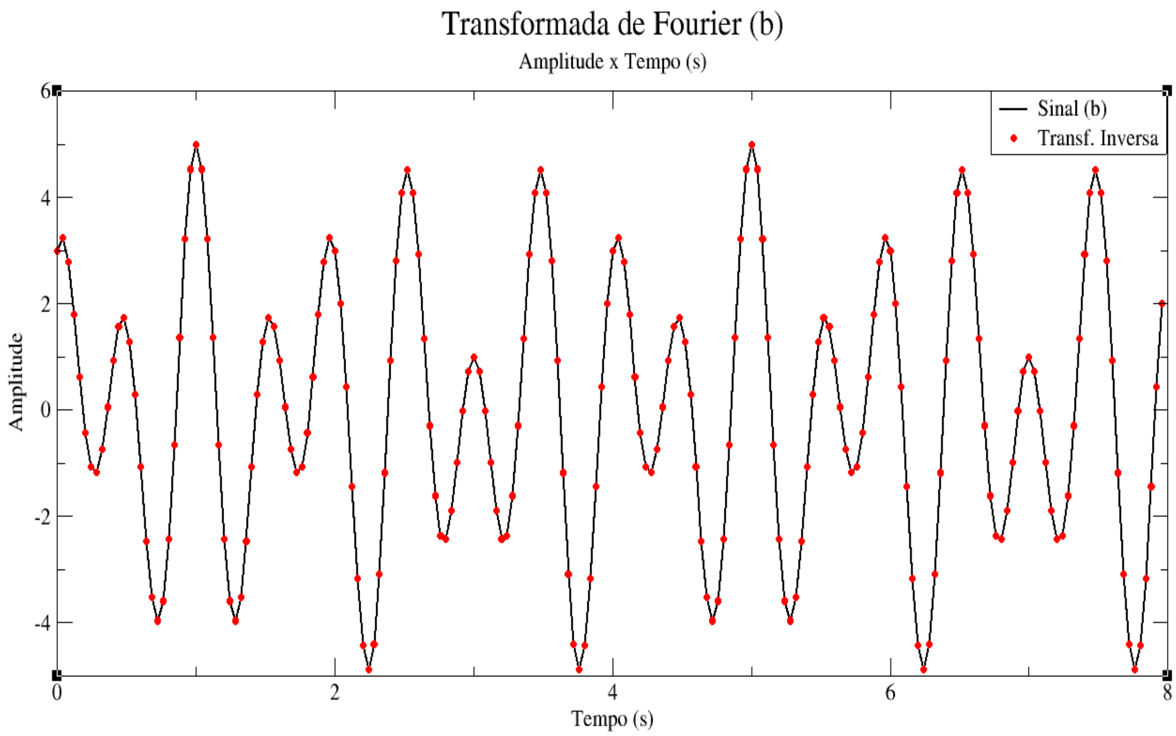


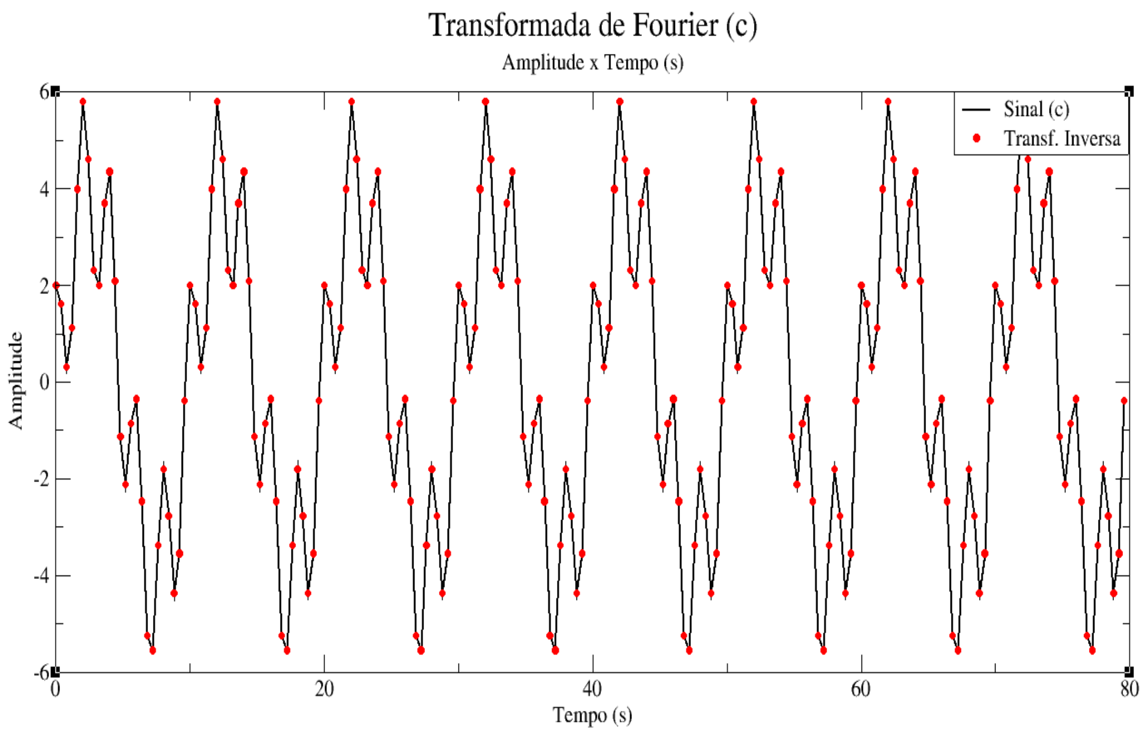
Por mais que as outras frequências estejam perto da frequência verdadeira, temos a formação de um "Aliasing", com a frequência dos outros sinais, de acordo com o gráfico de (e) e (f).

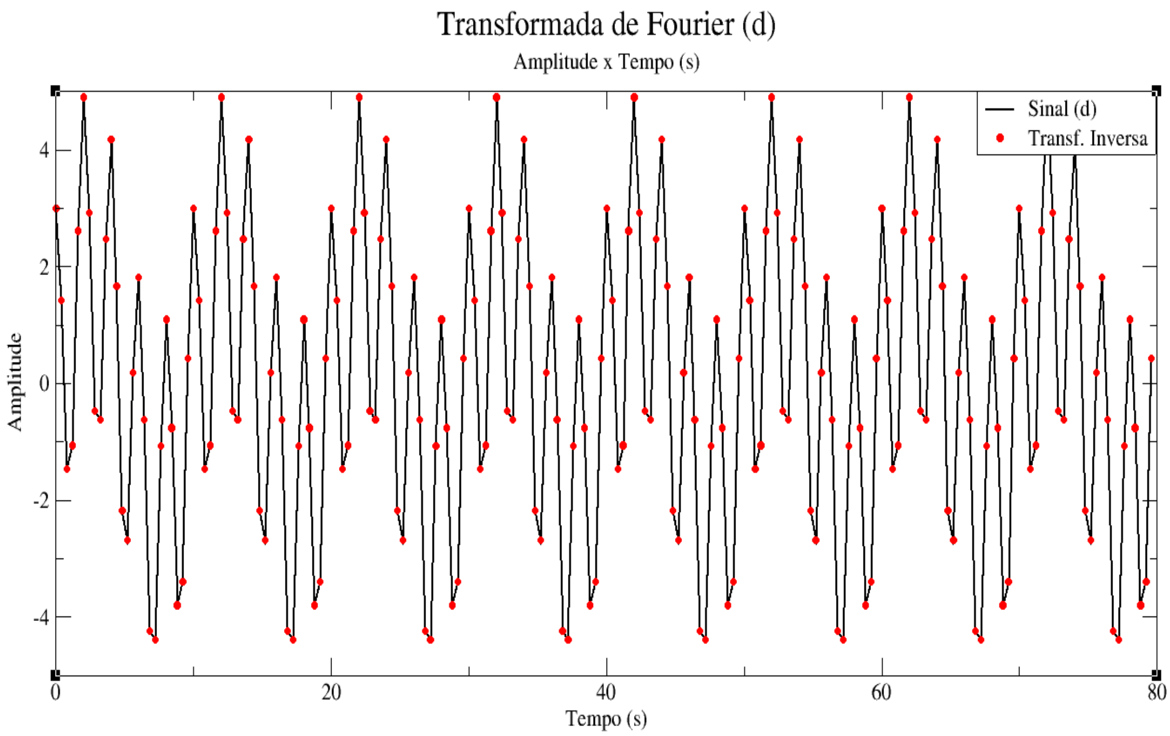
2.4 Tarefa IV

Na tarefa IV utilizamos a última parte do código da tarefa I, cáculo da transformada inversa de Fourier, para aferir se as análises estão válidas, o resultado esperado é que a transformada inversa esteja semelhante ao sinal colocado no código, obtemos os seguintes gráficos:




Transformada de Fourier (b)
Amplitude x Tempo (s)
Sinal (b)
Transf. Inversa
Amplitude
Tempo (s)
6
4
2
0
-2
-4
0
2
4
6
8


Transformada de Fourier (c)
Amplitude x Tempo (s)
Sinal (c)
Transf. Inversa
Amplitude
Tempo (s)
6
4
2
0
-2
-4
-6
0
20
40
60
80



Podemos observar que o código está válido, pois a transformada inversa está correspondendo absolutamente ao sinal de entrada.

2.5 Tarefa V

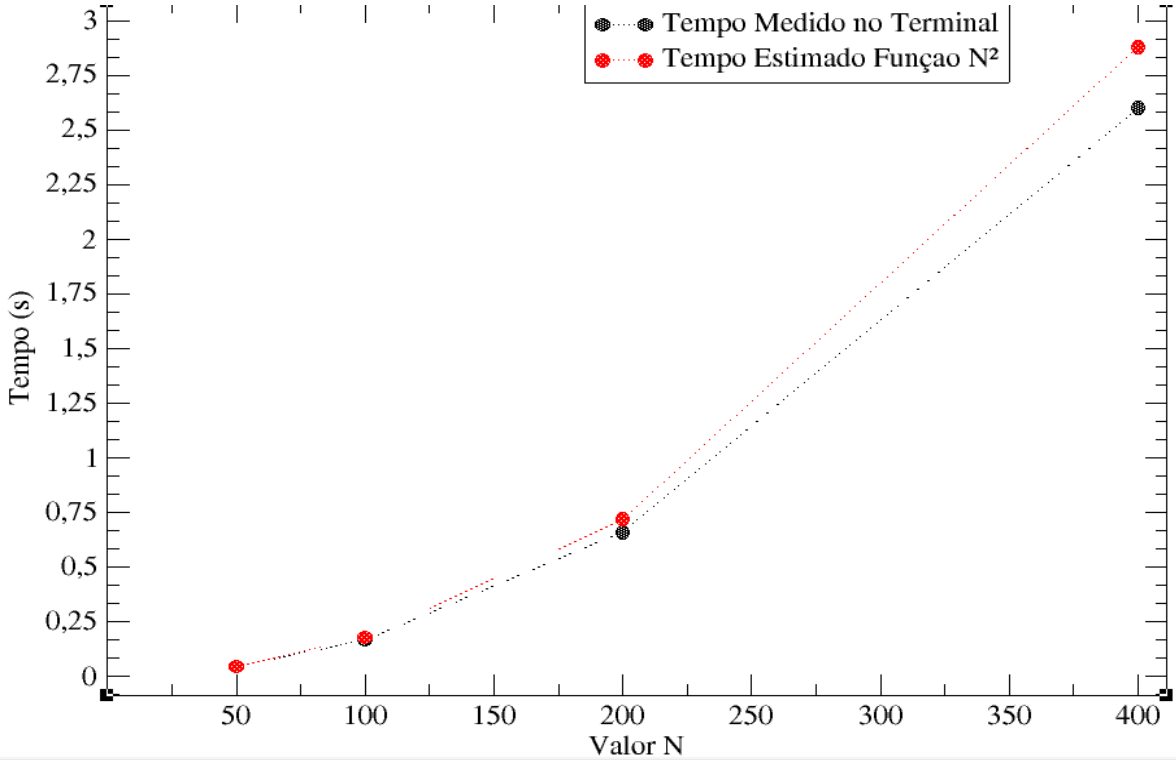
O intuito da Tarefa V é medir a capacidade computacional do método discretizado da tranformada de Fourier, e, além disso, aferir se há uma correlação entre o tempo de processamento e o número de dados do sinal.

Para medir o tempo operacional utilizamos da função intrínseca do Fortran **cpu..time**, e ainda adicionamos outro **do** ao código para aumentar o tempo de processamento, sem alterar a relação com o N, e tornar mais fácil a análise. Os tempos obtidos e seus respectivos N foram escritos no terminal:

```
N igual a   50             N igual a  100
Tempo igual a 0.45E-01s    Tempo igual a 0.17E+00s


N igual a  200             N igual a  400
Tempo igual a 0.66E+00s    Tempo igual a 0.26E+01s
```

Ainda, foi construido uma análise comparando com alguns tempos estimados pela relação N^2, disposta no gráfico abaixo:

Figura 1: Gráfico Tempo x N



No qual, conseguimos observar uma equivalência entre a distribuição e o tempo medido.

3 Referências

[1] - Nicholas J. Giordano, Computational Physics (Prentice Hall, New Jersey,1997)

[2] - Prof. Dr. Marcelo Andrade da Costa Vieira, Processamento Digital de Imagens Médicas: ppt. Disponível em: ¡https://edisciplinas.usp.br/pluginfile.php/4551919/mod-resource/content/0/Aula4-TransformadadeFourier.pdf. Acesso em: 25 mar. 2023