

EA614 - Análise de Sinais

Exercício de Fixação de Conceitos (EFC) 4 – Amostragem e DFT

Turma U – 2º semestre de 2023

Prof: Levy Boccato Email: lboccato@dca.fee.unicamp.br

1 Introdução

Conforme visto em sala de aula, o teorema da amostragem de Nyquist-Shannon define a taxa mínima com que um sinal de banda limitada deve ser amostrado de modo a permitir sua reconstrução. Nos casos em que taxas inferiores são utilizadas, surge o fenômeno denominado *aliasing*, comprometendo a reconstrução do sinal original. Neste exercício, estudaremos este fenômeno no contexto de um sinal de áudio, assim como uma maneira de atenuá-lo.

Além disso, vimos que a transformada discreta de Fourier (DFT, do inglês *discrete Fourier transform*) realiza uma transformação que permite analisar o conteúdo espectral de uma sequência de amostras. Na segunda parte deste exercício, iremos utilizar a DFT para identificar a frequência analógica de um sinal que foi digitalizado.

2 Atividades

2.1 Amostragem e *Aliasing*

Neste exercício, vamos trabalhar com um trecho da famosa música *Numb*, da banda Linkin Park, que marcou época desde os anos 2000. Este foi um dos *singles* do álbum *Meteora*, lançado em 2003.

Para carregar um arquivo de áudio no Matlab basta usar o comando `audioread`:

```
[y, Fs]=audioread('linkin-park-numb.wav');
```

Este comando retorna o sinal de áudio, `y`, e a frequência de amostragem, `Fs`. Note que `y` corresponde a uma matriz com `num_amostras` linhas e duas colunas, uma para cada canal de áudio. Apenas por simplicidade, vamos tomar a média dos dois canais para realizar o experimento:

```
y=(y(:,1)+y(:,2))/2;
```

Em Python existem diversas bibliotecas que permitem a leitura e o processamento de arquivos de áudio. Dois exemplos populares e poderosos são `librosa` e `soundfile`. Outra possibilidade é usar o `SciPy`, que tem grandes chances de já fazer parte de sua instalação de Python. Para isso, use os comandos

```
import scipy.io as sio
Fs, y = sio.wavfile.read('linkin-park-numb.wav')
```

Talvez você receba um *Warning* ao fazer a leitura aqui. Isso está relacionado à forma como o arquivo foi gerado, e não tem impacto sobre o restante do experimento. Como em Matlab, `y` é um sinal estéreo, com dois canais. Para transformar em um sinal mono, como exigido para o experimento, execute o comando

```
y=(y[:,0]+y[:,1])/2
```

Obs.: A taxa de amostragem tipicamente empregada em sinais de áudio (e.g., música) corresponde a 44,1 kHz.

- Utilizando a rotina `espectro(y)` fornecida, mostre o espectro de frequências do sinal de áudio e discuta seu conteúdo espectral. Nesta discussão, relacione as frequências digitais (Ω [rad]) observadas no espectro com os respectivos valores de frequências analógicas (f [Hz]).
- Reduza a taxa de amostragem por um fator de $M = 6$. Para isto, a cada bloco de M amostras, basta reter uma amostra de `y` e descartar as $M - 1$ amostras seguintes. Matematicamente, a nova sequência gerada se relaciona com `y` da seguinte forma:

$$y_{\text{dec}}(n) = y(Mn). \quad (1)$$

Apresente o espectro do sinal subamostrado ($y_{\text{dec}}(n)$) e discuta as mudanças em relação ao espectro do sinal original.

Curiosidade: este procedimento de redução da taxa de amostragem via processamento digital é conhecido como **decimação**.

- (c) Ouça, então, o sinal de áudio original e o subamostrado e comente as diferenças.

Para isto, utilize o comando `soundsc` do Matlab:

```
soundsc(z,Fs),
```

onde F_s denota a taxa de amostragem associada ao sinal z .

Para ouvir o áudio em Python, use os comandos

```
import IPython.display as ipd
ipd.Audio(z,rate=Fs)
```

Obs.: Lembre-se que, após a decimação, a taxa de amostragem foi reduzida para F_s/M .

- (d) Uma maneira de minimizar o *aliasing* produzido pela subamostragem consiste em aplicar um filtro passa-baixas (FPB) sobre o sinal original antes da decimação. Um FPB próximo ao ideal pode ser construído com o auxílio do método da janela de Kaiser. Para este exercício, a rotina `kaiser` é fornecida, a qual recebe como parâmetros a frequência de passagem (Ω_p) e a frequência de rejeição (Ω_r), ambas em *rad* e retorna a resposta ao impulso do filtro (h). Apresente e discuta a resposta em frequência do filtro (utilizando a rotina `espectro(h)`) para os seguintes casos:
- $\Omega_p = 0,45$ [rad], $\Omega_r = 2$ [rad];
 - $\Omega_p = 0,45$ [rad], $\Omega_r = 0,5$ [rad];
 - $\Omega_p = 1,5$ [rad], $\Omega_r = 2$ [rad].
- (e) Utilizando $\Omega_p = 0,45$ [rad] e $\Omega_r = 0,5$ [rad], filtre (através da convolução) o sinal original. Apresente e discuta o espectro do sinal filtrado. Escute o sinal filtrado e analise os efeitos.
- (f) Subamostre o sinal obtido no item (e) (ou seja, o sinal pré-filtrado pelo FPB de Kaiser) por um fator $M = 6$. Compare o espectro obtido com aquele associado ao sinal original subamostrado (item (b)). Escute os sinais e discuta as diferenças. Lembre-se de fazer a correção na frequência de amostragem ao utilizar o comando `soundsc`.

2.2 DFT e Identificação de Frequências

- (g) Abra o arquivo ‘EEG.txt’, o qual contém um registro de eletroencefalografia (EEG) coletado por um eletrodo posicionado no centro da região occipital (parte posterior) do crânio a uma taxa de 250 amostras por segundo, enquanto o indivíduo olhava fixamente para um estímulo visual que piscava com frequência f_e (Hz). É esperado, neste caso, que a atividade elétrica dos neurônios apresente um sincronismo com a frequência do estímulo, de modo que é possível notar uma concentração da energia do EEG em torno de f_e .

Mostre, então, o espectro de magnitude do EEG em termos de frequências analógicas (no intervalo de 0 a $f_s/2$ Hz) e identifique a frequência f_e do estímulo.

Dica: A expectativa que temos é a de que exista um pico de significativa magnitude no espectro do EEG exatamente na frequência f_e relacionada ao estímulo visual para o qual a pessoa está olhando. Por isso, a identificação de f_e passa pela análise do primeiro pico mais destacado do espectro. A fim de determinar o índice k do maior módulo dos coeficientes de $X[k]$, procure usar o comando `np.argmax()` em Python, ou `max()` em MATLAB. Dada a simetria que o espectro possui, estamos interessados apenas na primeira ocorrência da máxima amplitude (ou seja, no primeiro pico mais destacado do espectro).

Atenção: a DFT (FFT) fornece uma representação discreta em frequência, como pode ser visto pelo índice k , no eixo x do gráfico. O que queremos nesse item é associar uma frequência analógica, medida em Hz, a um determinado valor de k .

Obs.: por eficiência computacional, os pacotes de análises numéricas costumam ter um algoritmo particular da DFT, a chamada FFT, *Fast Fourier Transform*. Trata-se da mesma transformada, mas calculada de um modo bastante eficiente. Em MATLAB, podemos usar o seguinte comando para obter a FFT de uma sequência:

```
Y = fft(y);
```

Já em Python, podemos usar a biblioteca NumPy da seguinte forma:

```
Y = np.fft.fft(y);
```

Lembre-se de, em ambos os casos, tomar apenas o módulo da transformada, já que a FFT retorna uma sequência complexa.