EET-01
Laboratório 1 - Sinais

16 de março de 2020

1 Leitura e Notação Usada

Para os seguintes exercícios, recomendamos ler a Seção 2.0 e a Seção 2.1 em [1]. A notação que é usada a seguir é a mesma usada no curso EET-01 e [1].

2 Introdução

Os sinais básicos usados com freqüência no processamento digital de sinais são o sinal de impulso unitário $\delta[n]$, exponenciais da forma $a^n u[n]$ ($u[n]$ é a função degrau unitário), ondas senoidais e sua generalização para exponenciais complexos. Os seguintes exercícios são direcionados à geração e representação destes sinais no MATLAB. Como o único tipo de dados numéricos no MATLAB é a matriz $M \times N$, os sinais devem ser representados como vetores: $M \times 1$ matrices se vetores de colunas, ou $1 \times N$ matrizes se vetores de linhas. No MATLAB todos os sinais devem ser finitos em comprimento. Isso contrasta agudamente com a resolução de problemas analíticos, onde uma fórmula matemática pode ser usada para representar um sinal de comprimento infinito (por exemplo, um exponencial em decomposição, $a^n u[n]$).

Um segundo problema é o domínio de indexação associado a um vetor de sinal. MATLAB assume por padrão que um vetor é indexado de 1 a N, a duração do vetor. Em contraste, um sinal vector é muitas vezes o resultado da amostragem de um sinal em algum domínio onde a indexação vai de 0 a $N-1$; ou, talvez, a amostragem comece em algum índice arbitrário que seja negativo, por exemplo, em $-N$. As informações sobre o domínio de amostragem não podem ser anexadas ao vetor de sinal contendo os valores do sinal. Em vez disso, o usuário é forçado a acompanhar esta informação separadamente. Geralmente, isso não é um problema até chegar a hora de plotar de um sinal, em cujo caso o eixo horizontal deve ser rotulado apropriadamente.

Um último ponto é o uso da notação vetorial do MATLAB para gerar sinais. Um significativo poder do ambiente MATLAB é sua notação de alto nível para manipulação de vetores; para os laços são quase sempre desnecessários. Ao criar sinais como uma onda senoidal, é melhor para aplicar a função `sin` a um argumento vetorial, consistindo de todas as amostras de tempo. No projetos seguintes, nós tratamos os sinais comuns encontrados no processamento digital de sinais: impulsos, trens de impulso, exponenciais e sinusóides.

3 Exercícios

Estes exercícios de laboratório concentram-se nas questões envolvidas com a geração de sinais básicos em tempo discreto no MATLAB. Muitos dos trabalhos centram-se na utilização de rotinas de vetores internos do MATLAB para geração de sinal.

A plotagem de sinais de tempo discreto é feita com a função `stem` no MATLAB. O seguinte código de MATLAB criará 31 pontos de um sinusóide em tempo discreto:

```
nn = 0 :  30 ; %- vector of time indices
sinus= sin(nn/2+1) ;
```

Observe que o índice $n = 0$ deve ser referido como `nn(1)`, devido ao esquema de indexação do MATLAB; da mesma forma, `sinus(1)` é o primeiro valor no sinusoide. Ao traçar a onda senoidal, usaríamos a função `stem`, que produz o gráfico de sinal em tempo discreto comumente visto nos livros de texto (segue Fig. 1):

```
stem(nn,sinus) ;
```

O primeiro argumento vetorial deve ser dado para obter o eixo n correto. Para comparação, tente `stem(sinus)` para ver a rotulagem padrão.

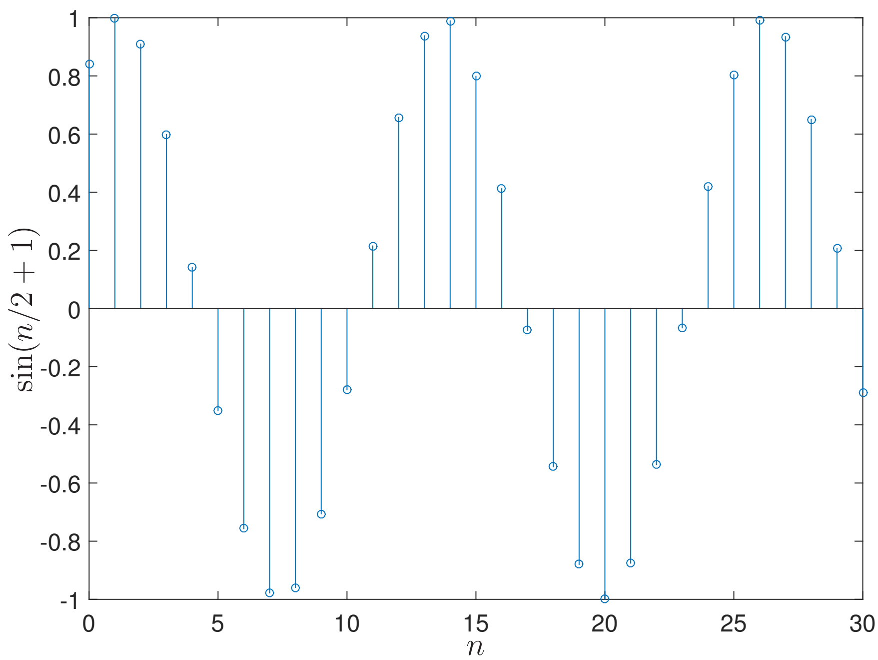

Figura 1: Gráfico de um sinal de tempo discreto.

Exercício 1.1: Impulsos

O sinal mais simples é o sinal de impulso unitário (deslocado):

$$\delta[n-n_0] = \begin{cases} 1 & n = n_0 \\ 0 & n \neq n_0. \end{cases} \tag{1}$$

Para criar um impulso no MATLAB, devemos decidir quanto do sinal é de interesse. Se o impulso $\delta[n]$ for usado para entrada de um sistema linear temporal invariante (LTI) causal, poderíamos querer ver os L pontos de $n = 0$ a $n = L-1$. Se escolhermos $L = 31$, o seguinte código de MATLAB criará um "impulso":

```
L=31;
nn= 0:(L-1);
imp= zeros(L,1);
imp(1)=1;
```

Observe que o índice $n = 0$ deve ser referido como `imp(1)`, devido ao esquema de indexação do MATLAB.

a. Gere e plote as seguintes seqüências. Em cada caso, o eixo horizontal (n) deve estender apenas no intervalo indicado e deve ser rotulado em conformidade. Cada sequência deve ser exibida como um sinal de tempo discreto usando `stem`:

$$x_1[n] = 0,9\,\delta[n-5] \qquad 1 \leq n \leq 20 \tag{2}$$
$$x_2[n] = 0,8\,\delta[n] \qquad -15 \leq n \leq 15 \tag{3}$$
$$x_3[n] = 1,5\,\delta[n-333] \qquad 300 \leq n \leq 350 \tag{4}$$
$$x_4[n] = 4,5\,\delta[n+7] \qquad -10 \leq n \leq 0. \tag{5}$$

b. Os impulsos deslocados, $\delta[n-n_0]$, podem ser usados para construir um trem de impulsos ponderado, com período P e comprimento total MP:

$$s[n] = \sum_{m=0}^{M-1} A_m \delta[n-mP]. \tag{6}$$

Os pesos são $A_m = 1$; como eles são todos iguais, o trem de impulsos é periódico com o período P. Gerar e plotar um trem de impulsos periódico cujo período seja $P = 5$ e cujo comprimento seja 50. Comece o sinal em $n = 0$. Tente usar uma ou duas operações vetoriais em vez de um laço `for` para definir os locais de impulso. Quantos impulsos estão contidos dentro do sinal de comprimento finito?

c. O seguinte código MATLAB produzirá um sinal repetitivo no vetor x:

```
x=[0;1;1;0;0;0] * ones(1,7);
x= x(:);
```

```
size(x) % <-- return the signal length
```

Plotar x para visualizar sua forma; em seguida, forneça uma fórmula matemática semelhante a (6) para descrever esse sinal.

Exercício 1.2: Sinusoides

Outro sinal muito básico é a onda cosseno ou senoidal. Em geral, são necessários três parâmetros para descrever um sinal totalmente senoidal: amplitude A, frequência ω_0 e fase ϕ

$$x[n] = A\cos(\omega_0 n + \phi). \tag{7}$$

a. Gere e plote cada uma das seguintes seqüências. Use a capacidade de vetor de MATLAB para fazer isso com uma chamada de função pegando o cosseno (ou seno) de um argumento vetorial. Em cada caso, o eixo horizontal n deve se estender somente sobre o intervalo indicado e deve ser rotulado de acordo. Cada sequência deve ser exibida como uma sequência usando `stem`:

$$x_1[n] = \sin\left(\frac{\pi}{17}n\right) \qquad 0 \le n \le 25 \tag{8}$$

$$x_2[n] = \sin\left(\frac{\pi}{17}n\right) \qquad -15 \le n \le 25 \tag{9}$$

$$x_3[n] = \sin\left(3\pi n + \frac{\pi}{2}\right) \qquad -10 \le n \le 10 \tag{10}$$

$$x_4[n] = \cos\left(\frac{\pi}{\sqrt{23}}n\right) \qquad 0 \le n \le 50. \tag{11}$$

Dê uma fórmula mais simples para $x_3[n]$ que não usa funções trigonométricas. Explique porque $x_4[n]$ não é uma sequência periódica.

b. Escreva uma função MATLAB que irá gerar uma sinusóide de comprimento finito. A função será precisa de um total de cinco argumentos de entrada: três para os parâmetros e mais dois para especificar o primeiro e último índice n do sinal de comprimento finito. A função deve retornar uma vetor de coluna que contém os valores da sinusóide. Teste esta função plotando os resultados para várias opções dos parâmetros de entrada. Em particular, mostre como gerar o sinal $2\sin(\pi n/11)$ para $-20 \le n \le 20$.

c. Reescreva a função na parte b. para retornar dois argumentos: um vetor de índices sobre o intervalo de n e os valores do sinal.

Exercício 1.3: Sinusoides Amostrados

Freqüentemente, um sinal em tempo discreto é produzido pela amostragem de um sinal de tempo contínuo, como onda senoidal de freqüência constante. A relação entre a frequência de tempo contínuo e a frequência de amostragem é o ponto principal do teorema de amostragem de Nyquist-Shannon, que requer que a frequência de amostragem seja pelo menos o dobro da freqüência mais alta no sinal reconstrução perfeita.

Em geral, uma sinusóide de tempo contínuo é dada pela seguinte fórmula matemática:

$$s(t) = A\cos(2\pi f_o t) + \phi) \tag{12}$$

onde A é a amplitude, f_0 é a frequência em Hertz e ϕ é a fase inicial. Se um discreto sinal é produzido por amostragem regular de $s(t)$ a uma taxa de $f_s = 1/T$, obtemos

$$s[n] = s(t)|_{t=nT} = A\cos(2\pi f_0 T n + \phi) = A\cos\left(2\pi \frac{f_0}{f_s} n + \phi\right). \tag{13}$$

Comparação com (7) para uma sinusóide em tempo discreto. $x[n] = A\cos(\omega_0 n + \phi)$, mostra que a freqüência de radianos normalizada é agora uma versão em escala de f_0, $\omega_0 = 2\pi(f_0 T)$.

a. Escreva uma função que irá gerar amostras de $s(t)$ conforme definido em (12) para criar um sinal de tempo discreto de comprimento finito. Esta função exigirá seis entradas: três para os parâmetros de sinal, dois para os tempos de início e de parada e um para os a taxa de amostragem (em Hertz). Para fazer a função MATLAB corresponder ao tempo contínuo definição de sinal, faça as unidades dos tempos de início e fim segundos, não o número de índice. Usar esta função para gerar uma sinusóide amostrada com a seguinte definição:

```
Freqüência de sinal= 1200 Hz Frequência de amostragem= 8kHz
Fase inicial = 45 deg        Hora de início= 0 s
Amplitude= 50                Tempo final= 7 ms
```

   Faça dois gráficos do sinal resultante: um como uma função do tempo t (em milissegundos) e o outro como uma função do índice de amostra n usado em $t_n = nT$. Determinar o comprimento do sinal discreto resultante e o número de períodos da sinusóide incluídos no vetor.

b. Mostrar por manipulação matemática que amostrar um cosseno nos tempos $t_n = nT$ e com fase $\phi = \frac{2}{3}\pi$ resultará em um sinal de tempo discreto que parece ser uma onda senoidal. Use a função da parte a. para gerar uma onda senoidal discreta, alterando o tempo do início e do fim para a amostragem.

Exercício 1.4: Exponenciais Complexos

Embora no mundo real, os sinais devem ter valores reais, geralmente é extremamente útil gerar, processar e interpretar pares de sinais de valor real como sinais de valor complexo. Este é feito combinando os sinais em um par, como as partes reais e imaginárias de um complexo número, e processando este par com outros sinais de valor complexo usando as regras de aritmética complexa. O uso de pares de sinais é uma parte importante de muitos sistemas de processamento digital de sinais, especialmente aqueles que envolvem modulação.

Exponenciais complexos são uma classe de sinais complexos que é extremamente importante porque a amplitude complexa (notação fasorial) fornece uma maneira concisa de descrever as sinais sinusoidal . A maioria dos estudantes de engenharia elétrica está familiarizada com os fasores em conexão com circuitos de corrente alternada (CA) ou sistemas de energia, mas seu uso em radar, propagação de ondas e análise de Fourier é igualmente significativo (embora o termo fasor nem sempre seja usado).

No MATLAB, as funções `real` e `imag` extrairão as partes reais e imaginárias de um número complexo. Ao traçar um vetor complexo, os padrões para `plot` e `stem` podem nos confundir. Se `z` for complexo, então `plot(z)` plotará a parte imaginária versus a real parte; e `plot(n, z)` plotará a parte real de `z` versus `n`. No entanto, `stem(z)` apenas traçar a parte real. Se você quiser visualizar gráficos simultâneos das partes real e imaginária, os comandos `subplot(211)` e `subplot(212)` antes de cada comando `stem` force as duas parcelas a serem colocadas na mesma tela, uma acima da outra. Veja Fig. 2, que foi criado usando o seguinte código:

```
nn = 0 :  25 ;
xx= exp(j *nn/3 ); %-- complex exponential
subplot (211)
stem(nn , real(xx))
title( 'Real part ' ) ,xlabel('n' )
subplot(212)
stem(nn , imag(xx))
title( 'Imaginary part' ) , xlabel( 'n' )
```

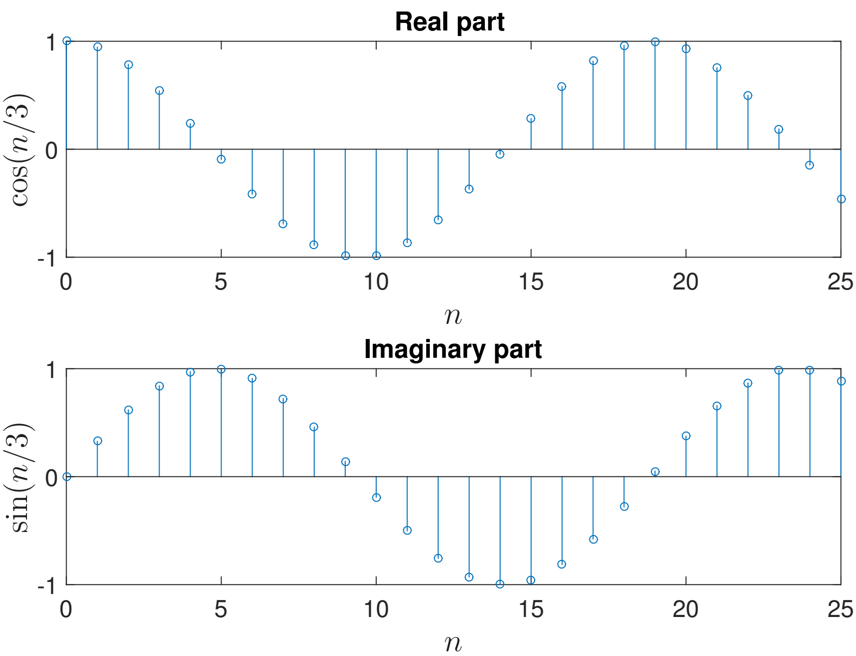

Figura 2: Gráfico de um sinal complexo de tempo discreto.

A notação exponencial real pode ser estendida a sinais exponenciais de valor complexo que incorporam os sinais seno e cosseno. Esses sinais formam a base da transformada de Fourier.

a. No MATLAB, um sinal complexo é uma extensão natural da notação

$$z[n] = a^n u[n] \tag{14}$$

onde a função degrau unitário é definida como

$$u[n] = \begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}. \tag{15}$$

A função degrau unitário está relacionado ao impulso

$$u[n] = \sum_{k=0}^{\infty} \delta[n-k]. \tag{16}$$

Assim, o parâmetro a pode ser tomado como um número complexo para gerar esses sinais. Recorde a fórmula de Euler para o exponencial complexo (em uma forma que dê um sinal):

$$x[n] = z_0^n = e^{(\ln(z_0)+\mathrm{j}\angle z_0)n} = r^n e^{\mathrm{j}\theta n} = r^n(\cos(\theta n) + \mathrm{j}\sin(\theta n)) \tag{17}$$

onde $z_o = re^{\mathrm{j}\theta} = r\angle\theta$. Use este relacionamento para gerar um exponencial complexo com $z_0 = 0,9\angle 45°$. Plote as partes reais e imaginárias de $x[n]$ no intervalo $0 \leq n \leq 20$. Aviso prévio que o ângulo de z_0 controla a frequência dos sinusóides.

b. Para o sinal na parte a. faça um gráfico da parte imaginária versus a parte real. O resultado deve ser uma espiral. Experimente com diferentes ângulos para θ - um valor menor deve produzir uma imagem melhor de uma espiral.

c. A equação (17) não é genérica o suficiente para produzir todos os exponenciais complexos. O que está faltando é uma constante complexa para dimensionar a amplitude e a fase dos sinusóides. Este é o chamado *notação fasorial*:

$$G\, z_0^n = A\, r^n e^{\mathrm{j}(\theta n+\phi)} = Ar^n(\cos(\theta n + \phi) + \mathrm{j}\sin(\theta n + \phi)) \tag{18}$$

onde $G = A\, e^{\mathrm{j}\phi} = A\angle\phi$ é a amplitude complexa do complexo exponencial. Gerar e traçar cada uma das seguintes seqüências. Converta os sinusosides em notação complexa; então crie o vetor de sinal usando `exp`. Se o sinal é puramente real, deve ser gerado por tomando a parte real de um sinal complexo. Em cada parcela, o eixo horizontal n deve estender apenas no intervalo indicado e deve ser rotulado de acordo:

$$x_1[n] = 3\sin\left(\frac{\pi}{7}n\right) + \mathrm{j}3\cos\left(\frac{\pi}{7}n\right) \qquad 0 \leq n \leq 20 \tag{19}$$

$$x_2[n] = \sin\left(\frac{\pi}{17}n\right) \qquad -15 \leq n \leq 25 \tag{20}$$

$$x_3[n] = 1,1^n\cos\left(\frac{\pi}{11}n + \frac{\pi}{4}\right) \qquad 0 \leq n \leq 50 \tag{21}$$

$$x_4[n] = 0,9^n\cos\left(\frac{\pi}{11}n\right) \qquad -10 \leq n \leq 20 \tag{22}$$

Para cada sinal, determine os valores das constantes de amplitude e fase que devem ser usados em G; determine também o ângulo e a magnitude de z_0.

Referências

[1] A. V. Oppenheim and R. W. Schafer with J. R. Buck, *Discrete-Time Signal Processing*, Prentice-Hall Inc., 1999.