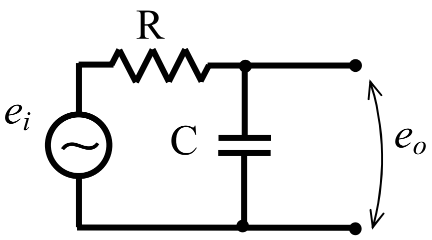
Resposta em Freqüência

Introdução:

Uma das entradas de teste para o estudo de sistemas dinâmicos são as funções senoidais. Em particular, os métodos de resposta em freqüência foram desenvolvidos inicialmente para o estudo e projeto de amplificadores de sinal em sistemas de comunicação, onde ondas senoidais aparecem naturalmente. Entretanto o mesmo método pode ser estendido para o estudo de quaisquer sistemas dinâmicos. Neste texto aplicaremos o método apenas em sistemas lineares invariantes no tempo.

Exemplo:

Considere o seguinte circuito elétrico:



Equação diferencial:

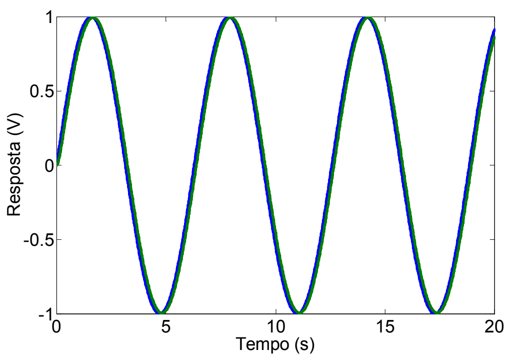
$RC\dot{e}_o + e_o = e_i$

Função de transferência:

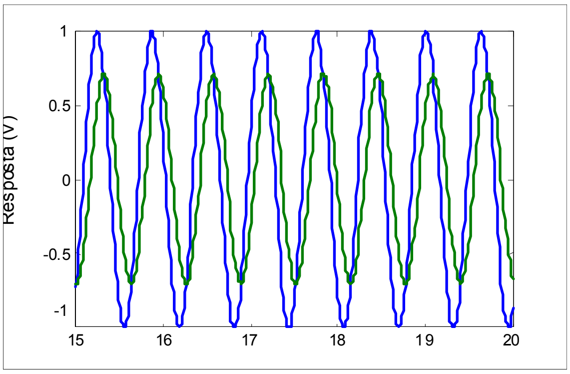
$$G(s) = \frac{1}{(RCs+1)}$$

Simulando este sistema obtemos as respostas mostradas na figura abaixo. As condições iniciais foram nulas, e as entradas foram:

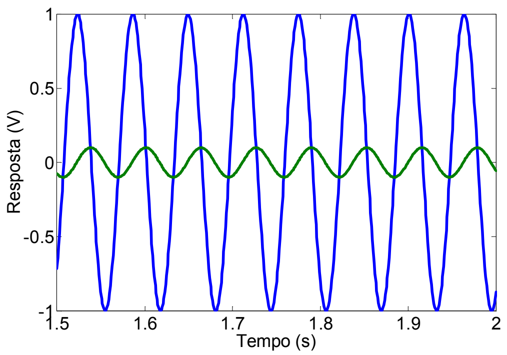
Fig. a:	e_i(t) = sen(t)	freqüência de 1 rad/s
Fig. b:	e_i(t) = sen(10*t)	freqüência de 10 rad/s
Fig. c:	e_i(t) = sen(100*t)	freqüência de 100 rad/s
Fig. d:	e_i(t) = sen(1000*t)	freqüência de 1000 rad/s



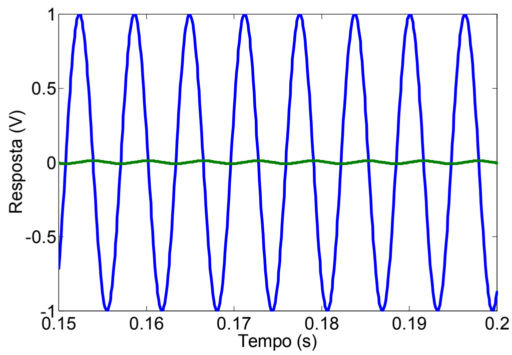
(a)



(b)



(c)



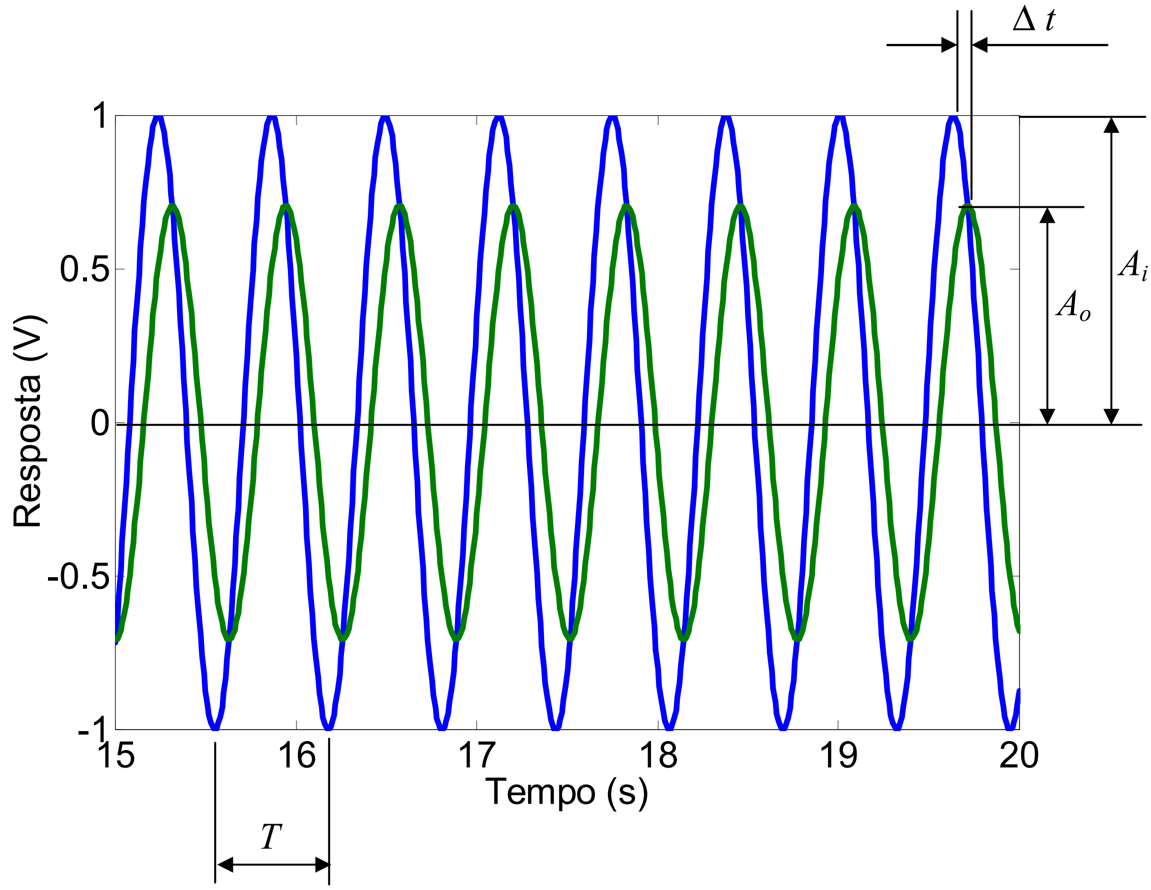
(d)

Nestas figuras temos em azul a entrada $e_i(t)$ e em verde a saída $e_o(t)$, considerando ainda $RC = 0{,}1$

Um sistema dinâmico não responde de forma instantânea, ele tem um atraso. Quando a entrada varia muito lentamente, como, por exemplo, para e_i(t) = sen(t) , o sistema possui velocidade de resposta suficiente para acompanhar a excitação externa (Fig. a). Entretanto, à medida que a entrada varia mais rapidamente (a freqüência de oscilação aumenta), a velocidade de resposta do sistema não é suficiente, o sistema não consegue acompanhar a entrada e se pode começar a notar que a amplitude da resposta diminui, e o atraso aumenta (figuras seguintes).

Ganho e fase:

Estímulo: azul
Resposta: verde



Após passar o transitório inicial, a saída também é uma senóide de mesma freqüência que a senóide de entrada, e se pode determinar o ganho e a fase:

Ganho $g = \frac{A_o}{A_i}$ Fase $\phi = -\frac{360.\Delta t}{T}$

Este comportamento é característico de cada sistema, portanto, pode ser usado para comparar sistemas, quanto à velocidade de resposta, desempenho, etc. Tanto o ganho como a fase dependem da freqüência do sinal de entrada. Assim, cada sistema terá uma curva específica de ganho pela freqüência e de fase pela freqüência. Em engenharia é comum usar os diagramas de Bode, onde as escalas são logarítmicas. É possível provar ainda que o ganho e a fase estão relacionadas com a função de transferência $G(s)$ do sistema da seguinte forma:

$$g(\omega) = |G(j\omega)|$$

$$\phi(\omega) = \angle G(j\omega)$$

onde ω é a freqüência do sinal de entrada.
É importante notar que estas curvas de resposta em freqüência também podem ser obtidas experimentalmente.

Diagramas de Bode

As curvas de ganho e fase podem ser mostradas de diversas formas, como em gráficos polares (diagramas de Nyquist), gráficos de fase por ganho em db (carta de Nichols) e diagramas logarítmicos (Diagramas de Bode).

Nos diagramas de Bode as escalas logarítmicas são usadas devido à conveniência das propriedades do logaritmo. O eixo horizontal mostra a freqüência ω em rad/s, em escala logarítmica. No gráfico de ganho, o eixo horizontal mostra o ganho em db (ver definição abaixo), e no gráfico de fase o eixo vertical mostra a fase em graus em um escala linear.

Determinação de ganho e fase:

$$g(\omega) = |G(j\omega)| \qquad g_{db}(\omega) = 20.\log_{10}(g) \quad \text{(em db)}$$

$$\phi(\omega) = \angle G(j\omega) \quad \text{(em graus)}$$

No Scilab existem funções prontas (`bode e nyquist`), cujo uso veremos a seguir:

Estude o programa abaixo:

```
// Resposta em frequencia
// Sistema dinâmico (eh o exemplo de suspensão da lista E):
m=1;
b=10;
k=1000;

// Definindo os polinomios:
n=(-m)*poly(0,'s','roots');
d=poly([k b m],'s','coef');

// Montando a funcao de transferencia
G=syslin('c',n/d);

// Diagrama de Bode da funcao G (observe as escalas dos graficos):
bode(G)

// A escala horizontal eh logaritmica, e mostra a frequencia em Hz.
// A escala vertical do grafico superior eh "linear", e mostra o ganho em db.
// A escala vertical do grafico inferior eh linear e mostra a fase em graus.
// Verifique que podemos usar o comando bode tambem com o modelo no espaco de
// estados, porem se definirmos apenas uma entrada (sistemas SISO e SIMO).



////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
```

Se nao quisermos usar este comando "pronto" (bode), podemos usar os comandos a seguir:

```
// Para obtermos o diagrama de Bode tal como eh mais conhecido, devemos deslocar
// a fase de 360 graus, se a versao for a 2.6 ou anterior. Alem disso, a escala
// horizontal deve mostrar a frequencia em rad/s, e nao em Hz.

// Vamos construir um vetor w onde o logaritmo da distancia entre um valor e
// outro eh constante, comecando em 10 elevado a (-3), terminando em 10 elevado
// a 5, com 2000 pontos, e convencionando que a unidade eh rad/s:
w=logspace(-3,5,2000);

// Como o Scilab usa a frequencia em Hz nos calculos, devemos modificar nosso
// vetor de w para fr, transformando a unidade de rad/s para Hz:
fr=w/(2*%pi);

// Determinamos os vetores de fase e magnitude, calculando o valor vf da funcao
// de transferencia G em cada ponto do vetor de frequencias fr.
vf=repfreq(G,fr);

// Calculando a magnitude (em db) e a fase (em graus):
[fase, mag]=phasemag(vf);

// Se for a versao 2.6 ou anterior, devemos deslocar a fase de 360 graus:
//fase=fase-360;

// Plotando o diagrama de Bode de ganho com a unidade de frequencia em rad/s:
// Abrindo uma nova janela de grafico:
xset('window',1)

// Dividindo a janela em nxm partes, n linhas e m colunas, e usando a parte p:
// subplot(n,m,p). Em nosso exemplo usaremos 2 subjanelas, 2 linhas e 1 coluna:
subplot(2,1,1)

// Plotando o Diagrama de Bode de ganho:
// Ajustando a espessura da linha (colocando este comando aqui ajustamos a
```

```
// espessura dos eixos do grafico):
xset('thickness',1);

// Ajustando o grafico para podermos desenhar a grade antes, e já desenhando os
// eixos (os valores devem ser ajustados caso a caso):
plot2d('ln',w(1),mag(1),5,'011',rect=[10^(-3),-120,10^5,-20],nax=[2,8,2,5]);

// 'ln' significa que a escala horizontal eh logaritmica (l) e a escala vertical
// eh normal (n). O default eh 'nn', normal, normal. Se fosse um grafico loglog,
// deveriamos usar 'll'.
// Nao estamos plotando a curva ainda, por isso colocamos apenas o primeiro
// ponto da curva, ou seja, w(1) e mag(1).
// O numero 5 indica que a cor da curva eh vermelha.
// O parametro '011' indica:
// 0 – a legenda não eh mostrada.
// 1 – os limites do grafico serao dados pelo parametro 'rect'.
// 1 – os eixos sao desenhados, e o eixo vertical eh desenhado no lado esquerdo.
// O parametro 'rect' define os limites das escalas:
// O limite inferior da escala horizontal eh 10^(-3).
// O limite inferior da escala vertical eh -120.
// O limite superior da escala horizontal eh 10^5.
// O limite superior da escala vertical eh -20.
// O parametro 'nax' define as divisoes das escalas:
// Na escala horizontal sao 8 divisoes com 2 subdivisoes por divisao. Neste caso
// este comando eh ignorado por que a escala horizontal eh logaritmica.
// Na escala vertical sao 5 divisoes com 2 subdivisoes por divisao.

// Desenhando a grade antes para que a grade apareca sob a curva:
xgrid(4)

// O numero 4 indica a cor da grade, e lembrando que a espessura das linhas esta
// ajustada em 1, a grade tera espessura 1.

// Plotando a curva de ganho depois da grade, para que a curva apareca sobre a
// grade:

// Primeiro aumentando a espessura da linha (com isso apenas a curva tera
// espessura 2):
xset('thickness',2);

// Agora desenhando a curva de ganho (observe que nao abrimos uma nova janela
// grafica):
plot2d('ln',w,mag,5,'000',rect=[10^(-3),-120,10^5,-20],nax=[2,8,2,5]);

// O parametro '000' indica:
// 0 – a legenda nao eh mostrada.
// 0 – as escalas sao as mesmas do grafico anterior (por isso nao abrimos uma
// nova janela, pois queremos as mesmas escalas).
// 0 – os eixos nao sao desenhados. Os eixos já foram desenhados anteriormente
// com espessura de linha 1. Se desenhassemos os eixos novamente, teriam
// espessura 2.

// Colocando o titulo e nomeando os eixos:
xtitle("Magnitude","rad/s","db")

// Usando a segunda subjanela:
subplot(2,1,2)

// Plotando o diagrama de Bode de fase:
// Ajustando a espessura da linha (queremos eixos e grade com espessura 1):
xset('thickness',1);

// Ajustando o grafico para podermos desenhar a grade, e já desenhando os
// eixos:
```

```
plot2d('ln',w(1),fase(1),5,'011',rect=[10^(-3),-270,10^5,-90],nax=[2,8,3,4]);

// Desenhando a grade antes para que a grade apareca sob a curva:
xgrid(4)

// Plotando a curva de fase depois da grade, para que a curva apareca sobre a
// grade:
// Primeiro aumentando a espessura da linha (queremos a curva com espessura 2):
xset('thickness',2);

// Agora desenhando a curva de fase:
plot2d('ln',w,fase,5,'000',rect=[10^(-3),-270,10^5,-90],nax=[2,8,3,4]);

// Colocando o titulo e nomeando os eixos:
xtitle("Fase","rad/s","graus")



////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
```

Graficos polares:

```
// Grafico polar (grafico da resposta em frequencia onde o eixo vertical eh a
// parte imaginaria da resposta e o eixo horizontal eh a parte real):
////////////////////////////////////////////////////////////////////////////////
// O grafico polar pode ser obtido usando-se o comando nyquist:
// Abrindo uma nova janela de graficos:
xset('window',2);

// Desenhando o grafico polar:
nyquist(G,fr)



///////////////////////////////////////////////////////////////////////////////
///////////////////////////////////////////////////////////////////////////////
// O grafico polar tambem pode ser obtido usando-se comandos mais basicos:

// Lembrando, a resposta em frequencia vf da funcao de transferencia G ja foi
// calculada anteriormente para o vetor de frequencias fr:
imaginario=imag(vf);
partereal=real(vf);

// Abrindo uma nova janela de graficos:
xset('window',3)

// Desenhando o grafico polar (Diagrama de Nyquist):
plot2d(partereal,imaginario,-1,'045')

// O parametro '-1' indica que a curva esta sendo desenhada com a marca '+'.
// O parametro '045' significa:
// 0 – a legenda nao eh mostrada.
// 4 – a escala eh isometrica, para que, ao desenharmos um circulo, ele nao
// fique deformado, aparentando ser uma elipse.
// 5 – os eixos sao desenhados de tal forma que passam pelo ponto 0,0.
```

Obs.: nas versões mais recentes do Scilab a obtenção dos diagramas de Bode com escalas definidas pelo usuário é facilitada por interfaces um pouco mais amigáveis.

Conversão de modelos, função de transferência

Introdução:

Sistema representado no espaço de estados. Considere as matrizes **A**, **B**, **C** e **D**.

<u>**Exemplo**</u>**:**

Considere o seguinte sistema:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \overbrace{\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}}^{\mathbf{A}} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}^{\mathbf{B}} \cdot u$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{\mathbf{C}} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \end{bmatrix}}_{\mathbf{D}} \cdot u$$

ou, usando uma notação mais compacta:

$$\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u}$$
$$\mathbf{y} = \mathbf{C} \cdot \mathbf{x} + \mathbf{D} \cdot \mathbf{u}$$

Construção do modelo no Scilab:

```
// Definicao das Matrizes do Sistema:
A=[0 1;-2 -3];
B=[0;1];
C=[1 0];
D=0;
// Construcao do sistema:
sistema=syslin('c',A,B,C,D);
```

Determine a função de transferência entre a saída y e a entrada u:

Exemplo:

```
G=ss2tf(sistema)
```

No espaço de estados, podemos escolher outro conjunto de variáveis para representar o sistema. No exemplo do massa-mola, ao invés de deslocamento e velocidade, poderíamos usar força da mola e quantidade de movimento.

Voltando ao exemplo numérico, e escolhendo as variáveis z_1 e z_2, tais que:

$$z_1 = x_1 - x_2$$
$$z_2 = x_1 + x_2$$

(ao invés de escrever as equações usando x_1 e x_2, iremos escrevê-las usando z_1 e z_2.

ou seja:

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}}_{\mathbf{T}^{-1}} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0{,}5 & 0{,}5 \\ -0{,}5 & 0{,}5 \end{bmatrix}}_{\mathbf{T}} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

, ou seja: $\mathbf{z} = \mathbf{T}^{-1} \cdot \mathbf{x} \quad \Rightarrow \quad \mathbf{x} = \mathbf{T} \cdot \mathbf{z}$

Derivando a equação temos:

$$\mathbf{z} = \mathbf{T}^{-1} \cdot \mathbf{x} \quad \Rightarrow \quad \dot{\mathbf{z}} = \mathbf{T}^{-1} \cdot \dot{\mathbf{x}}$$

mas sabemos que $\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u}$, logo:

$$\dot{\mathbf{z}} = \mathbf{T}^{-1} \cdot \dot{\mathbf{x}} \quad \Rightarrow \quad \dot{\mathbf{z}} = \mathbf{T}^{-1} \cdot (\mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u}) \quad \Rightarrow \quad \dot{\mathbf{z}} = \mathbf{T}^{-1} \cdot \mathbf{A} \cdot \mathbf{x} + \mathbf{T}^{-1} \cdot \mathbf{B} \cdot \mathbf{u}$$

como $\mathbf{x} = \mathbf{T} \cdot \mathbf{z}$:

$\dot{\mathbf{z}} = \mathbf{T}^{-1} \cdot \mathbf{A} \cdot \mathbf{T} \cdot \mathbf{z} + \mathbf{T}^{-1} \cdot \mathbf{B} \cdot \mathbf{u}$, que podemos escrever como $\dot{\mathbf{z}} = \mathbf{A_T} \cdot \mathbf{z} + \mathbf{B_T} \cdot \mathbf{u}$

Da mesma forma:

$\mathbf{y} = \mathbf{C} \cdot \mathbf{x} + \mathbf{D} \cdot \mathbf{u} \quad \Rightarrow \quad \mathbf{y} = \mathbf{C} \cdot \mathbf{T} \cdot \mathbf{z} + \mathbf{D} \cdot \mathbf{u}$, ou seja: $\mathbf{y} = \mathbf{C_T} \cdot \mathbf{z} + \mathbf{D_T} \cdot \mathbf{u}$

Resumindo:

$$\dot{\mathbf{z}} = \mathbf{A_T} \cdot \mathbf{z} + \mathbf{B_T} \cdot \mathbf{u}$$
$$\mathbf{y} = \mathbf{C_T} \cdot \mathbf{z} + \mathbf{D_T} \cdot \mathbf{u}$$

com:

$$\mathbf{A_T} = \mathbf{T}^{-1} \cdot \mathbf{A} \cdot \mathbf{T} \qquad \mathbf{B_T} = \mathbf{T}^{-1} \cdot \mathbf{B}$$
$$\mathbf{C_T} = \mathbf{C} \cdot \mathbf{T} \qquad \mathbf{D_T} = \mathbf{D}$$

Como as variáveis y e u não foram trocadas, a relação entre elas, que é a função de transferência, permanece a mesma. Verifique calculando a função de transferência usando o sistema com as variáveis de estado z_1 e z_2:

```
Ti=[1 -1;1 1];
T=inv(Ti);
AT=Ti*A*T;
BT=Ti*B;
CT=C*T;
DT=D;
// Construcao do sistema convertido:
sistemaT=syslin('c',AT,BT,CT,DT);
// Funcao de transferencia do sistema convertido:
GT=ss2tf(sistemaT)
```

Compare `sistema` com `sistemaT` e compare `G` com `GT`.

Exercícios:

1 – Considerando o sistema do exemplo:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}0 & 1\\ -2 & -3\end{bmatrix}\cdot\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}0\\ 1\end{bmatrix}\cdot u$$

$$y=\begin{bmatrix}1 & 0\end{bmatrix}\cdot\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}0\end{bmatrix}\cdot u$$

Escolhendo as variáveis z_1 e z_2, tais que:

$$z_1 = 2x_1 + x_2 \qquad z_2 = x_1 + x_2$$

ou seja: $$\begin{bmatrix}z_1\\ z_2\end{bmatrix}=\underbrace{\begin{bmatrix}2 & 1\\ 1 & 1\end{bmatrix}}_{\mathbf{T}^{-1}}\cdot\begin{bmatrix}x_1\\ x_2\end{bmatrix} \Rightarrow \begin{bmatrix}x_1\\ x_2\end{bmatrix}=\underbrace{\begin{bmatrix}1 & -1\\ -1 & 2\end{bmatrix}}_{\mathbf{T}}\cdot\begin{bmatrix}z_1\\ z_2\end{bmatrix}$$

$$\mathbf{z}=\mathbf{T}^{-1}\cdot\mathbf{x} \qquad \Rightarrow \qquad \mathbf{x}=\mathbf{T}\cdot\mathbf{z}$$

Determine o sistema convertido. Existe algo de particular neste sistema convertido?

Determine os autovetores da matriz **A**. Existe alguma relação entre a matriz **T** e os autovetores da matriz **A**?

Determine as funções de transferência do sistema e do sistema convertido, e compare.

Determine os autovalores e autovetores da matriz $\mathbf{A_T}$ e compare com os autovalores e autovetores da matriz **A**.

2 – Obtenha os diagramas de Bode do circuito elétrico (ver o início desta lista) para diferentes valores de constante de tempo. Use o comando `bode`.

3 – Considere o sistema massa-mola-amortecedor (o exercício – não o exemplo – da lista E), e obtenha os diagramas de Bode para diferentes valores de freqüência natural e diferentes valores de coeficiente de amortecimento (mude *m* e *c*, mas adote o valor fixo de *k* = 1000 N/m). Use o comando `bode`.

4 – Escolha um dos casos do exercício 3 e obtenha o diagrama de Bode para os mesmos parâmetros *m*, *c* e *k* escolhidos, mas modificando o modelo matemático de tal forma que se tenha a deformação da mola em mm, mantendo as unidades das demais variáveis. Sugestão para modificar o modelo: use a abordagem do exercício 1 acima, com:

$$z_1 = 1000\cdot x_1$$

$$z_2 = x_2$$

z_1 é a deformação da mola em mm, e x_1 é a deformação da mola em m

z_2 é a velocidade da massa em m/s, e x_2 também é a velocidade da massa em m/s

Use o comando `bode`.

Lição de casa:

1 – Você é o programador, e precisa desenvolver, para um usuário, uma função do Scilab (arquivo com extensão sci) que resolve um sistema de *n* equações diferenciais de primeira ordem, ordinárias, lineares e a parâmetros constantes, usando o método de Runge Kutta de 4ª ordem.

Sistema de equações diferenciais que a função (que você está desenvolvendo) deve resolver:

$$\dot{\mathbf{x}}=\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u}$$
$$\mathbf{y}=\mathbf{C}\mathbf{x}+\mathbf{D}\mathbf{u}$$

x: vetor de estados (tamanho n×1)
u: vetor de entradas (tamanho m×1)
y: vetor de saídas (tamanho p×1)
A: matriz n×n
B: matriz n×m
C: matriz p×n
D: matriz: p×m

Sintaxe que a função deve ter:

```
[y,x]=integrador(A,B,C,D,u,t,x0)
```

Para o Scilab:

t: vetor de instantes de tempo (tamanho 1×N) – fornecido pelo usuário da função.

x: matriz com as soluções das equações diferenciais (tamanho n×N) – saída da função.

u: matriz com o vetor de entradas em cada instante (tamanho m×N) – fornecida pelo usuário da função.

y: matriz com o vetor de saídas em cada instante (tamanho p×N) – saída da função.

x0: vetor com os estados no instante inicial (tamanho n×1) – fornecido pelo usuário da função.

A: matriz n×n – fornecida pelo usuário da função.

B: matriz n×m – fornecida pelo usuário da função.

C: matriz p×n – fornecida pelo usuário da função.

D: matriz: p×m – fornecida pelo usuário da função.

Matriz de Transição de Estados

2 – Desenvolva um algoritmo de solução numérica de um sistema de equações diferenciais ordinárias de 1ª ordem, lineares, a parâmetros constantes e homogêneas, que use o conceito de matriz de transição de estados. Implemente o algoritmo na forma de função do Scilab (arquivo do tipo sci), em que os dados fornecidos sejam a matriz A do sistema, as condições iniciais x0, e um vetor t de números correspondentes a instantes igualmente espaçados.

Sistema de equações diferenciais ordinárias de 1ª ordem, lineares, a parâmetros constantes e homogêneas:

$$\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x}$$

Sintaxe que a função deve ter:

```
[x]=integrador2(A,t,x0)
```

3 – Desenvolva um algoritmo de solução numérica de um sistema de equações diferenciais ordinárias de 1ª ordem, lineares, a parâmetros constantes, que use o conceito de matriz de transição de estados. Implemente o algoritmo na forma de função do Scilab (arquivo do tipo sci), em que os dados fornecidos sejam as matrizes A, B, C e D do sistema, as condições iniciais x0, e um vetor t de números correspondentes a instantes igualmente espaçados, e um vetor de entradas u.

Sistema de equações diferenciais ordinárias de 1ª ordem, lineares, a parâmetros constantes:

$$\dot{\mathbf{x}} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u}$$
$$\mathbf{y} = \mathbf{C} \cdot \mathbf{x} + \mathbf{D} \cdot \mathbf{u}$$

Sintaxe que a função deve ter:

```
[x]=integrador3(A,B,C,D,u,t,x0)
```

4 – Faça a Lista G.