Exercícios para fazer com o Scilab/Matlab

1 O objetivo deste exercício é que você se familiarize com alguns comandos básicos de manipulação de matrizes no scilab e no matlab. São essencialmente os mesmos, com pequenas diferenças. Use e abuse dos helps do scilab (matlab). No scilab, com as funções em linear álgebra. No caso do matlab principalmente com as funções em matlab\elmat e matlab\matfun). Eles funcionam razoavelmente bem. *A idéia, neste primeiro exercício é que todos os itens estão numa sequência e o que é armazenado num item continua na memória do seu PC, na hora que você está adiante.*

Registramos em negrito-itálico comandos escritos na linguagem do Scilab (Matlab).

i – Armazene $A = \begin{pmatrix} 1.0 & 2.0 & 3.0 & 4.0 \\ 4.0 & 3.0 & 2.0 & 1.0 \\ -1.0 & 2.0 & 3.0 & 4.0 \\ 0.0 & 1.0 & 0.0 & 1.0 \end{pmatrix}$, $b = \begin{pmatrix} 1.0 \\ 0.0 \\ 1.0 \\ 1.0 \end{pmatrix}$ e faça ***c =A*b***

ii – Teste para diferentes valores de m e n, os comandos ***zeros(m,n)***, ***zeros(n,n)***, ***eye(n,n)***, ***rand(m,n)*** e armazene em Z uma matriz 4x4 toda formada com zeros, em I4 a matriz identidade 4x4, em B e em C duas matrizes aleatórias 4x4.

iii –Mande seu computador calcular ***B*C*** e ***C*B***. Apostamos que você obteve resultados diferentes.

iv - Renove as matrizes aleatórias B e C do item iv e peça ao computador para calcular ***B*C*** e ***C*B*** novamente. Apostamos que continua dando diferentes. Se tiver paciência, repita a operação umas 6.243 vezes. Continuamos apostando que, em nenhuma vez você obteve B*C = C*B.
v - Compute (B - C)(B + C) e $B^2 - C^2$. Não era para dar o mesmo resultado?

vi – Armazene a entrada A_{23} de A em A23, armazene a primeira linha de A na matriz A1 1x4 (***A1=A(1,:)***), a primeira coluna de A em X1, 4x1 (***X1 = A(:,1)***). Crie em X23 uma matriz 2x2, coincidente com o bloco formado pelas entradas de A nas linhas 2 e 3, bem como nas colunas 2 e 3: (***X23=A(2:3,2:3)***) .

vii - Armazene em LIX a matriz formada a partir de A, substituindo A_{23} e A_{32} por zero.

viii - Armazene em LIX a matriz formada a partir de A, substituindo o bloco formado pelas entradas de A nas linhas 2 ou 3 e nas colunas 2 ou 3, pela matriz identidade 2x2. Faça-o em bloco, ou seja sem trocar entrada por entrada: (***LIX = A ; LIX(2:3,2:3) = eye(2,2)***)

ix - Armazene em AA a matriz formada a partir de A, substituindo a sua primeira linha pela soma das duas últimas. Também aqui, nenhuma entrada de AA deve ser obtida isoladamente, muito menos somando-se na mão as entradas correspondentes de A.

x - Teste para ver que inv(A) calcula a inversa de A (digite ***A*inv(A)*** e ***inv(A)*A***. Deu o que você esperava? Cheque para ver se ***inv(B)*inv(A)*** é, de fato, a inversa de ***A*B***. Compute também ***inv(A)*inv(B)*** e verifique que ela não é uma inversa de ***A*B***.

xi - Faça ***x = invA*b*** e calcule ***A*x – b***. Não deveria dar 0? Interprete o resultado que você obteve.

xii -. Peça ao seu computador para armazenar em invAA a matriz inversa de AA que você obteve no item ix. Preste bem atenção na resposta que ele vai lhe dar. Caso ele lhe dê uma matriz invAA, leia atentamente a advertência que ele lhe estará fazendo, tente calcular ***AA*invAA*** e ***invAA*AA*** e analise o resultado.

xiii - Caso o seu computador tenha lhe devolvido uma matriz invAA no item anterior, faça ***x = invAA*b*** . Calcule ***AA*x – b*** e explique porquê não deu zero.

xiv – Se certifique que b e c ainda são matrizes 4x1, como no item i, compute os produtos b^Tc, c^Tb, bem como bc^T e cb^T e compare os resultados obtidos: ***b’*c , c’*b, b*c’ ,c*b’***

xv - Faça $F = B + B^T$ e $G = C + C^T$. Verifique que F e G resultaram simétricas, mas F*G não é simétrica. Calcule F^T*G^T e G^T*F^T. Qual delas resulta ser $(F*G)^T$?

2 - Armazene A = rand(4,4), b = [1 0 0 0]

i - Ache a solução de Ax = b: ***(b= A\x)***
ii – Calcule o posto de A. (***rank(A)***)
iii – Armazene em invA uma inversa de A . (***invA = inv(A)***).
iv – Se certifique que inv(A)*b é solução de Ax = b
iii - Achar uma matriz na forma escada reduzida linha equivalente a A (***rref(A)***)
iv - Escrever uma fatoração lu de A, ou seja, $A = P^T LU$ (***[L, U,P] =lu(A)***)
v - Explore no help do scilab, em Linear álgebra, e elementary functions, algumas das funções colocadas à sua disposição. Sugiro que você cheque algumas de Linear Álgebra que corresponde a assuntos de algebra linear que já vimos ou que você já tenha visto no segundo grau, como por exemplo, det, ,inv, lu, lusolve, rank, rref, range, trace, spaninter, spantwo,spanplus, kernel, fullrf, exp, expm. Também em elementary functions tem funções muito úteis para manipular matrizes como rand, size, triu, tril, zeros, ones, rand, eye, sparse, spones, spget, etc... Explore no "help" de Matlab, as correspondentes funções em elfun e em matfun, para reconhecer comandos com matrizes que o Matlab lhe coloca à disposição.

3 – Ainda com a matriz A do exercício 2, escreva a matriz AI = (A I), ampliada a partir da A com a identidade $I = I_{4x4}$. Peça ao Scilab que calcule, com rref, uma matriz na forma escada reduzida e linha-equivalente a AI. Verifique que nas três últimas colunas de tal matriz você obteve a inversa de A

4 - Com a mesma A do exercício 2, e considerando a fatoração $A = P^T LU$, obtida no item iv, faça:

i – bb = Pb
ii – Encontre a solução de Ly = bb (***y = L\bb***)
iii – Encontre a solução de Ux = y e cheque para ver, se de fato, Ax = b

5 - Armazene em B uma matriz de entradas aleatórias 4x4. Compute e/ou verifique:
i - A solução de $Bx = I^{(1)}$, onde I é a matriz identidade 4x4.
ii - Que o *resíduo* $BX^{(1)} – I^{(1)}$ é desprezível.
iii - Que a primeira coluna de ***inv(B)*** é a solução da equação obtida em i.
iv - Uma matriz na forma escada reduzida linha-equivalente a B.
iv - Uma fatoração lu de B.

6 - Substitua a última linha de B pela soma das três primeiras.
i - Compute o posto de B (***rank(B)***)
ii – Peça ao Scilab para calcular, invB = ***inv(B)***. Cheque para ver se $invB*B = B*invB = I_{4x4}$ e explique o que acontece
iii - Some (10^(-12) , 0, 0, 0) à última linha de B e certifique-se que agora que sua nova B tem posto 4 novamente. Peça a seu software que obtenha $X^{(1)}$ resolvendo $Bx = I^{(1)}$ e verifique que a solução obtida tem um erro significativo, calculando o *resíduo* $BX^{(1)} – I^{(1)}$

7 - O objetivo deste exercício é que você compare os tempos que seu computador leva para processar as diferentes operações. Uma maneira é digitando tic antes dos comandos cujos tempos de uso da CPU você quer medir e toc após estes comandos. Por exemplo, para calcular o tempo de CPU necessário a multiplicar matrizes A e B, digita-se ***tic; A*B; toc*** Comece com n = 100.

i - Escreva matrizes A e B nxn, com entradas aleatórias, bem como um vetor b, nx1, igualmente com entradas aleatórias. Não se esqueça de acrescentar **;** aos comandos (***A = rand(n,n); B = rand(n,n); b = rand(n,1);***)

ii – Compare os tempos que o seu computador leva para, :multiplicar A e B, inverter A, calcular a fatoração LU de A**, r**esolver Ax = b, resolver um sistema triangular inferior Ly = Pb, onde P é a matriz de permutação obtida na fatoração LU e L a triangular inferior.

iii - O que você espera que ocorra com os tempos que você obtem no seu computador em i e em ii para cada uma das operações acima, se você dobrar n.

iv – Dobre o valor de n e refaça os item i e ii. Compare como variam os tempos que você obtem no seu computador em ii para cada uma das operações acima, quando você dobra n.

8 - Abra um diretório com seu nome em C: \ ··· e grave nele o programa abaixo, um arquivo de nome triaginf.sci para resolver sistemas triangulares Lx=b, sem zeros na diagonal principal, definido por:

```
function   x=triaginf(L,b)
// Triaginf recebe uma matriz triangular inferior L, nxn, sem zeros na diagonal, um vetor
//  coluna  b   e devolve em x a solução de Lx = b
 n=size(L,1);
 x = zeros(n,1);
 x(1)=b(1)/L(1,1);
 for i=2:n
     x(i) = (b(i) - L(i,1:i-1)*x(1:i-1))/L(i,i);
end
endfunction
```

Para o Scilab, triaginf(· , ·) funciona exatamente como uma função armazenado no seu diretório. no arquivo triaginf.m. É rodado sempre que chamado, numa sessão do scilab, na forma triaginf(L,b), onde L deve ser uma matriz triangular nxn, sem zeros na diagonal e b um vetor-coluna nx1.

ii – Mande executar, na sessão do scilab o arquivo,digitando exec(c:\ (seu diretório)\triaginf.sci.
Entre com alguma matriz triangular inferior 4x4 sem zeros na diagonal, b = [1; 0; 0; 0] e digite
x = triaginf(L,b). Cheque se, com um tal x, você de fato resolveu a equação Lx=b

(No matlab, há duas pequenas diferenças. O sinal //, que representa comentários destinados a serem lidos,mas não executados, é substituído por %, em matlab. E endfunction não é necessário ou então é substituido por end. Os arquivos se guardam como M-files, ou seja, *.m. Não é necessário mandar executar previamente o arquivo. Basta colocá-lo no diretório atual, ou então informar o path do arquivo no teclado)

9 - Verifique que o programa abaixo corresponde ao algoritmo da fatoração LU e digite-o, num arquivo de nome lu001.m no seu diretório de trabalho.

```
function   [L,U] = lu006(A)
//  L e U sao matrizes triangulares inferior  e superior tais que A =LU.
//  Se na diagonal principal aparecer algum número inferior a   10^(-12)  , considera-se que o
// algoritmo fracassou

n=size(A,1);
U=A;
L = eye(n,n);
for j=1:n-1
if   abs(U(j,j)) <   10 -12 , 'Pivô nulo atrapalhou a brincadeira.',  break, end
   b = 1/U(j,j);
   for i=j+1:n
         L(i,j) = b*U(i,j)  ;
         U(i,j:n) = U(i,j:n) - L(i,j)*U(j,j:n);
   end
end
endfunction
```

ii - Para rodá-lo, entre com uma matriz quadrada no teclado, (por exemplo ***A=rand(5,5)***),
e ***[L,U]=lu006(A)***. Verifique se deu ***A = L*U*** conforme o esperado.
iii - Explique o que acontece ao tentar rodá-lo com a matriz

$$B = \begin{pmatrix} 1 & 1 & 3 & 2 & 0 \\ 2 & 1 & 3 & 1 & 2 \\ 1 & 2 & 6 & 5 & 0 \\ 0 & 1 & 4 & 0 & 1 \\ 0 & 0 & 0 & 0 & 2 \end{pmatrix}$$