Resolução explicada dos exercícios 4, 5 e 6 da folha 6

exercício 4.a) No Matlab, comecemos por definir a matriz do sistema e o vetor dos termos independentes

```
>> A=[2 4 1; 0.5 1 1.25; 2 3 4], b=ones(3,1)

A =

    2.0000    4.0000    1.0000
    0.5000    1.0000    1.2500
    2.0000    3.0000    4.0000


b =

     1
     1
     1
```

Em seguida, usamos a função GaussElim (disponível na BB). Trata-se de uma implementação do método de eliminação de Gauss sem troca de linhas da matriz ampliada (isto é, sem troca de equações).

```
>> x=GaussElim(A,b)
ans =

   2.0000e+00   4.0000e+00   1.0000e+00   1.0000e+00
            0            0   1.0000e+00   7.5000e-01
            0  -1.0000e+00   3.0000e+00            0
```

Este é o resultado do primeiro passo de redução (em que são usadas operações de equivalência para introduzir zeros na primeira coluna da matriz A, abaixo da diagonal principal). O segundo passo de redução produz

```
ans =

   2.0000e+00   4.0000e+00   1.0000e+00   1.0000e+00
            0            0   1.0000e+00   7.5000e-01
            0          NaN          Inf          Inf
```

e o resultado final é

```
x =

   NaN
   NaN
   NaN
```

(nota: a divisão por zero produz Inf, Inf-Inf e Inf*Inf produz NaN, isto é, "Not-a-Number") Neste caso, o método falha porque no 2º passo de redução é nula a entrada na posição (2,2) o que produz o multiplicador $-1/0$.

exercício 4.b) Vamos usar o *format short e* para melhor apreciarmos o resultado obtido na posição(2,2) no final do primeiro passo de redução.

```
>>format short e;  A(2,2)=A(2,2)+2^-52; x=GaussElim(A,b)
ans =

   2.0000e+00   4.0000e+00   1.0000e+00   1.0000e+00
            0   2.2204e-16   1.0000e+00   7.5000e-01
            0  -1.0000e+00   3.0000e+00            0


ans =

   2.0000e+00   4.0000e+00   1.0000e+00   1.0000e+00
            0   2.2204e-16   1.0000e+00   7.5000e-01
            0            0   4.5036e+15   3.3777e+15


x =

  -3.8750e+00
   2.0000e+00
   7.5000e-01
```

mas esta solução tem erros grandes como se pode concluir por comparação com a solução dada por

```
>> A\b

ans =

  -4.3750e+00
   2.2500e+00
   7.5000e-01
```

Isto acontece porque a entrada $A(2,2) = 2.2204e - 16$ é muito mais pequena, em valor absoluto, do que a entrada $A(3,2) = -1.0000$ o que produz um multiplicador muito grande $m_{3,2} = -A(3,2)/A(2,2)$. É esta a causa da instabilidade numérica do método de eliminação de Gauss.

exercício 5.a) A função GaussElimPP (disponível na BB) incorpora a pivotação parcial, isto é, no início do k-ésimo passo de redução escolhe como pivot a entrada de maior valor absoluto de entre

$$A(k,k), A(k+1,k), ..., A(k,n),$$

digamos $A(p,k)$, e efetua a troca das linhas p e k da matriz ampliada. Isto evita a ocorrência de grandes multiplicadores já que todos terão valor absoluto não superior a 1.

exercício 5.b)

```
>> x=GaussElimPP(A,b)

A =

    2.0000    4.0000    1.0000
    0.5000   -1.0000    3.0000
    2.0000         0    1.0000


x =

   -4.3750
    2.2500
    0.7500
```

Esta solução coincide com a que é produzida pela função do Matlab com se pode concluir de

```
>> x-(A\b)

ans =

     0
     0
     0
```

exercício 6.a)

```
>> A=hilb(10); x=ones(10,1); b=A*x

b =

    2.9290
    2.0199
    1.6032
    1.3468
    1.1682
    1.0349
    0.9307
    0.8467
    0.7773
    0.7188
```

exercício 6.b)

```
>> xtil=A\b

xtil =

    1.0000
    1.0000
    1.0000
    1.0000
    0.9999
    1.0003
    0.9995
    1.0004
    0.9998
    1.0001
```

Como se pode ver, há erros importantes em xtil (algumas entradas não têm mais do que 4 algarimos corretos). Isto acontece porque o sistema é muito mal condicionado. Com efeito, o número de condição é

```
>> norm(A)*norm(inv(A))

ans =

   1.6024e+13
```

o que significa que erros nos dados poderão ser muito ampliados e produzir erros muito maiores nos resultados (o que de facto acontece).