Cálculo Numérico – SME0104 – ICMC-USP

Lista 1: MATLAB e Ponto Flutuante

Parte 1: Conceitos Básicos de MATLAB

Lembrete: dúvidas sobre as funções do MATLAB use o comando `help`.

---

1. Qual a resposta da sequência de comandos:
   ```
   >> u = [1, 2, 3];
   >> v = [4; 2; 1];
   >> u'*v'
   ```
2. Dada sequência de comandos:
   ```
   >> A = [1.1 2.2 3.3; 4.4 5.5 6.6];
   >> b = [10, 20];
   >> B = ones(2);
   ```
   Diga se é verdadeiro ou falso:

   (a) `>> c = [A,b]` não gera erro.

   (b) `>> c = [A',b]` não gera erro.

   (c) `>> size([A;b'])` gera erro.

   (d) `>> size([A,b'])` gera erro.

   (e) `>> size([A,b'])` fornece `ans = 2 4`.

   (f) `>> size([A,b'])` não gera erro.

   (g) `>> c = A*B` não gera erro.

   (h) `>> c = A*B - B*A` fornece `ans = 0`.

   (i) `>> B*A` fornece `ans = 5.5 7.7 9.9`.

   (j) `>> max(A)` fornece `ans = 4.4 5.5 6.6`.

   (k) `>> max(max(A))` fornece `ans = 2 2 2`.

   (l) `>> c = A(:,1)*b` não gera erro.

   (m) `>> c = A(3,:)*b` não gera erro.

3. Execute os seguintes comandos:

    (a) Declare a matriz `A = [2 10 7 6; 3 12 25 9]`.

    (b) Altere o elemento `A(2,1)` para 18.

    (c) Acrescente uma terceira linha a matriz com os elementos 30 21 19 1.

    (d) Defina o elemento `A(2,3)` como -16.

    (e) Defina uma matriz `B` que contenha as duas primeira linhas da matriz `A` e as colunas de 2 a 4.

4. Criar um vetor com componentes ímpares entre 31 e 75.

5. Seja `x = [2 5 1 6]`.

    (a) Some 16 a cada elemento.

    (b) Some 3 apenas para as componentes com índice ímpar.

    (c) Calcule a raiz quadrada de cada elemento.

    (d) Calcule o quadrado de cada elemento.

6. Seja `x = [3 2 6 8]'` e `y = [4 1 3 5]'` (vetores colunas).

    (a) Some `x` e `y`.

    (b) Eleve cada elemento de `x` a uma potência dada pelo correspondente elemento de `y`.

    (c) Divida cada elemento de `y` pelo correspondente elemento de `x`.

    (d) Multiplique cada elemento de `x` pelo correspondente elemento de `y`, chamando o resultado de `z`.

    (e) Some todos os elemento de `z` e guarde-o em uma variável `w`.

    (f) Calcule `x'*y - w` e interprete o resultado.

7. Crie um vetor `x=1:10` e calcule a soma de todos os elementos.

8. Crie uma matriz de dimensão 4×4 e a chame de `A`. (Sugestão: crie uma matriz com números aleatórios.)

    (a) apague a segunda linha de `A`;

    (b) apague a terceira linha de `A`;

9. Construir uma matriz 3×5, em que a lei de formação da primeira linha é seno, a segunda é cosseno e a terceira é raiz quadrada. Utilize para cada linha o vetor `v = 0:0.25:1`.

10. Qual o resultados dos seguintes comandos:

    (a) `abs(-3.51)`;

(b) `sign(-3.51);`

(c) `ceil(8.73);`

(d) `floor(8.73);`

(e) `round(8.73);`

(f) `lcm(12,8);`

(g) `gcd(12,8);`

(h) `rem(11,6);`

Parte 2: Scripts e Funções no MATLAB

1. Dada a sequência de comandos abaixo, diga qual a saída:

```
n = 3; i = n; j = 1;
A = (n-1)*ones(n) - eye(n);
while true
    if i == 0
        break;
    end
    if i == j
        A(i,j) = A(i,j) - i;
    else
        A(i,j) = i + j;
    end
    i = i - 1;
    j = n - i;
end
A
```

2. Considere as seguintes funções:

```
function M = f(A, a, b, k)
M = A; % M recebe uma cópia de A
if k
    M(a,:)  = A(b,:);
    M(b,:)  = A(a,:);
else
    M(:,a) = A(:,b);
    M(:,b) = A(:,a);
```

```
end

end % fim da função

function v = g(B)

[m n] = size(B);
v = zeros(1,n);
for i=1:n

    v(i) = sum(B(:, i));

end

end % fim da função

function M = h(A)

M = A;

v = g(A);

n = length(v);

    for i=1:n-1

        for j=1:n-i
            if v(j) > v(j+1)
              t = v(j);
              v(j) = v(j+1);
              v(j+1) = t;
              M = f(M,j,j+1,0);
            end
        end

    end

end % fim da função
```

Diga se as afirmações abaixo são verdadeiras ou falsas:

(a) A função `f`, se `k = 0`, devolve uma matriz `M` igual a `A` só que com as colunas `a` e `b`, se existirem, trocadas.

(b) A função `g` retorna o valor da soma de todos os elementos da matriz `B`.

(c)
```
>> T = [1 2 3; 4 5 6; 7 8 9];
>> f(T,2,1,1)
>> ans =

    4 5 6
    1 2 3
    7 8 9
```

(d)
```
>> T = [9 3 7; 1 0 2; 6 4 8];
>> g(A)
>> ans =
    19 3 18
```
(e)
```
>> A = [0 2 5; 4 1 2; 7 3 1];
>> B= f(A,2,3,1);
>> g(B)
>> ans =
    11 8 6
```
(f)
```
>> G = [2 3 7; 4 6 2; 1 0 8];
>> G = f(G,1,3,0);
>> f(G,3,2,1)
>> ans =
    7 3 2
    8 0 1
    2 6 4
```
(g)
```
>> A=[9 3 7; 1 0 2; 6 4 8];
>> h(A')
>> ans =
    1 6 9
    2 8 7
    0 4 3
```
(h)
```
>> A=[184;510;726];
>> B = h(A);
>> g(B);
>> ans =
    10 11 13
```
(i) A função `h` permuta as colunas da matriz de tal maneira que fiquem ordenadas com soma (da coluna) crescente.

3. Dada a *função* que calcula a soma de dois vetores de mesma dimensão, identifique e corrija os erros.

```
function z = Soma
z = ones(n);
for i=1:n
    z(i) = x(i) + y;
end
end
```

4. Indique qual será a saída para `as chamadas f4(1,-5,6)` e `m4([1:3:100])` das funções:

(a)
```
function r1=f4(a,b,c)
r1 = [-b+sqrt(b2-4*a*c)/(2*a) -b-sqrt(b2-4*a*c)/(2*a)];
end
```

(b)
```
function y = m4(x)
y = sum(x)/length(x);
end
```

5. Faça uma *função* que receba uma matriz A de ordem n e retorne a submatriz formada pelas primeira e penúltina linhas e segunda e última colunas.

6. Faça um *script* em MATLAB que leia uma matriz do $\mathbb{R}^n$ e em seguida verifique se a primeira e a última linha são ortogonais. A saída deve ser uma mensagem `ORTOGONAIS` em caso afirmativo e `NAO ORTOGONAIS`, caso contrário.

7. Implemente em MATLAB um programa que faça a ordenação dos elementos de um vetor na ordem crescente. E em seguida:

- Compare o resultado obtido com a função `sort(v)`.
- Usando os comandos `tic` e `toc` para calcular o tempo de execução, monte um gráfico (no. de elementos$\times$tempo), usando a função `sort(v)` e seu programa.
- Justifique se seu programa é mais eficiente do que o comando `sort(v)`. Use vetores com $10, 50, 100, 150, 200$ elementos.

8. Faça um *script* que leia as medidas dos três lados de um triângulo e o classifique quanto aos lados (equilátero, isósceles ou escaleno) e quanto aos ângulos (acutângulo, rectângulo ou obtusângulo). Verifique previamente se é possível formar um triângulo com as medidas dadas.

9. Dado um vetor t, de comprimento n, faça *funções* que calculem as expressões:

1. $ln(2+t+t^2)$;
2. $e^{t(1+\cos(3t))}$;
3. $\cos^2(t) + \text{sen}^2\,(t)$;
4. $\tan^{-1}(1)$;
5. $\sec^2(t) + \cot(t) - 1$.

10. Faça um *script* que determine o maior entre N números. A condição de parada é a entrada de um valor 0, ou seja, o algoritmo deve ficar calculando o maior até que a entrada seja igual a 0 (ZERO).

11. Faça um *função* em MATLAB que leia um número inteiro positivo e retorne 0 (falso) caso esse número não seja *perfeito* e 1 (verdadeiro) caso contrário. Um número inteiro é dito perfeito quando ele é igual à soma dos seus divisores próprios, exceto ele mesmo (isto é, o número 6 possui divisores 1, 2, 3 e 6, portanto $1+2+3=6$; 6 é um número perfeito – o número 8 possui divisores 1, 2, 4 e 8, portanto $1+2+4=7$; 8 não é um número perfeito).

12. Faça uma *função* que receba um número inteiro natural e que devolva um vetor com os divisores desse número. Se o número não for inteiro, a função deve considerar só a parte inteira desse número. Se o número for inferior a um, a função deve devolver um vetor vazio.

13. Escreva uma *função* que retorne 1 se um número for primo e 0 em caso contrário.

14. Implemente em MATLAB uma *função* que receba um inteiro n e que retorne uma *matriz de Hilbert* de ordem $n \times n$. Definição: Uma matriz de Hilbert é uma matriz cujos coeficientes são definidos por:

$$a_{ij} = \frac{1}{i+j-1}$$

Dica: Compare o resultado obtido com a função `hilb(n)`.

15. Implemente em MATLAB a fórmula de *Bailey-Borwein-Plouffe (BBP)* para computar o número π. A fórmula é dada por:

$$\pi = \sum_{k=1}^{n} \left[\frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right) \right]$$

Sua função deve receber n e retornar o valor aproximado de π (use `format long`).

16. Uma aproximação de $\cos(x)$ pode ser feita utilizando a seguinte série:

$$\cos(x) \approx \sum_{k=0}^{n} \frac{(-1)^k}{(2k)!} x^{2k}$$

Tomando $n = 10$, implemente em MATLAB uma *função* que receba de entrada um número $x \in \mathbb{R}$ em radianos e calcule o valor aproximado de $\cos(x)$ utilizando a série acima. Compare o resultado obtido com a função `cos(x)` através de um gráfico.

Parte 3: Noções Básicas de Ponto Flutuante

Lembrete (informação que vai estar disponível na prova)

Os números de $\mathbb{F}(\beta, t, m, M)$ são todos da forma:

$$\overline{x} = \pm(0.d_1 d_2 \ldots d_t)_\beta \times \beta^e$$

com $1 \leq d_1 \leq (\beta - 1)$, $0 \leq d_i \leq (\beta - 1)$ para $i = 2, \ldots, t$ e $m \leq e \leq M$. O sub-índice β indica a base na qual o número está representado.

1. Dados os números decimais: $x = 7.125$ e $y = 35.27$. Escreva-os na base 2.

2. Dados os números binários: $x = 11.0111$ e $y = 111.001$. Escreva-os na base 10.

3. Considere os números na base 4: $x = 33$, $y = 0.31$ e $z = 21.013$. Escreva-os na base 5.

4. Considere o sistema $\mathbb{F}(2, 5, -3, 2)$.

- **(a)** Quantos números podemos representar neste sistema?
- **(b)** Qual o menor e maior número em valor absoluto na base 10 que podemos representar neste sistema?
- **(c)** Qual é a precisão de máquina nesse sistema?

5. Faça uma *função* em MATLAB que faça conversão de um número real positivo decimal para binário.

6. Faça uma *função* em MATLAB que faça conversão de um número real positivo binário para decimal.