

- Para escrever um código, é exactamente igual escrevê-lo na linha de comandos ou num script. A vantagem de usar um script, é que o código fica guardado num ficheiro, e permite-vos consultá-lo mais tarde para estudar para os testes. Os testes serão em scripts.
- Para executar um script já feito, a maneira mais simples é ter o script guardado na pasta de trabalho.
- Os scripts têm a forma xxxx.m como se vê na figura.

- 1) Criem no ARCA a vossa pasta ACE onde irão guardar os vossos scripts das aulas.
- 2) Através do site <https://arcaweb.ua.pt/>, encontrem a pasta criada. Desta forma saberão aceder aos vossos scripts a partir de casa.
- 3) Na janela de comandos do MatLab, testem os seguintes comandos:
 - help
 - help sin
 - helpbrowser (é exactamente a mesma coisa que clicar no ponto de interrogação)
 - helpbrowser sin
 - helpwin sin
 - doc sin

Nos testes, mesmo não saindo nenhuma pergunta sobre estes comandos de help, se os souberem usar, poderão ser úteis para vos ajudar com alguma dúvida que surja sobre comandos que venham a ter que usar.

4) Na janela de comandos do MatLab, testem as seguintes operações:

- | | | |
|--|--|--|
| <ul style="list-style-type: none"> ▪ $2 + 10$ ▪ $2 - 1.5$ ▪ $2 * 10$ | <ul style="list-style-type: none"> ▪ $2 / 10$ ▪ 3^2 ▪ $(2+3) * 10 - 5$ | <ul style="list-style-type: none"> ▪ $(3i + 1) * (2i - 5)$ ▪ $\sin(0)$ ▪ $\cos(\pi/3)$ |
|--|--|--|

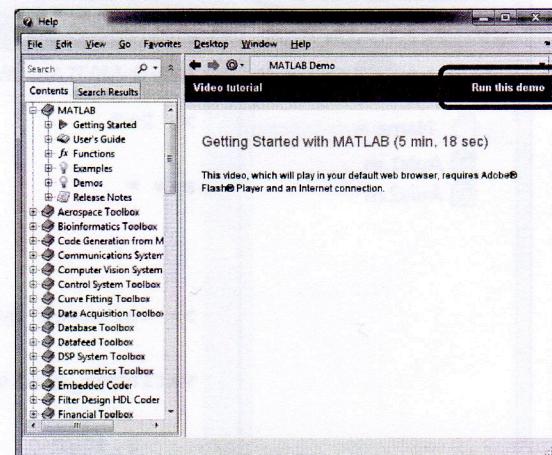
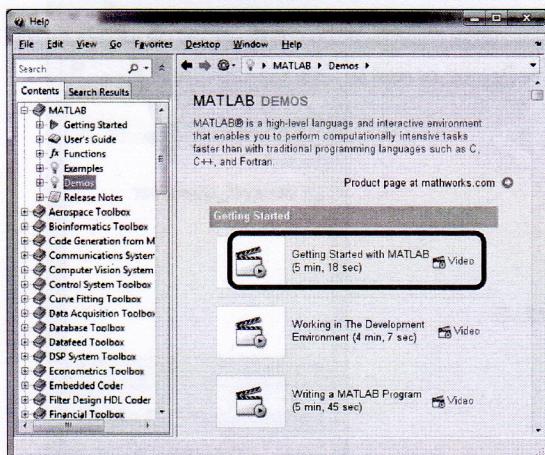
5) Experimentem utilizar variáveis, como na sequência de comandos:

- $a = 2;$
- $b = 10;$
- $nota = a + b$

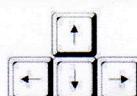
A diferença de utilizar o ponto e vírgula ou não, é que utilizando o ponto e vírgula, o resultado da operação não aparece imprimido na janela de comandos.

6) Explorem o seguinte comando:

- demo



7) Estando na janela de comandos, testem as setas de subida/descida do vosso teclado.



8) Criem o seguinte script, onde:

- Se clicarem em Run ou em F5, todo o código do script é executado
- Ou se seleccionarem uma parte do código, clicando em F9, só essa parte de código é executada.

```

Editor - C:\Users\releme\Desktop\ACE\Aula1.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Run

1 % Nome: xxxx
2 % N° Mec.: xxxx
3 % Turma: Pxx
4
5 close all, clear all,clc
6
7 %=====
8 % a)
9 teste1 = 18.5;
10 teste2 = 14.7;
11 teste3 = 17.3;
12 nota_final = 0.2*teste1 + 0.3*teste2 + 0.5*teste3
13
14 %=====
15 % b)
16 nota_em_percentagem = nota_final/20*100
17
18 %II
19 %=====
20 % a)
21 x = [-2 -1 0 1 2];
22 y = 2*x-1;
23 plot(x,y)

```

O símbolo de percentagem (%) permite fazer comentários. Ou seja, o texto escrito a seguir ao % não é executado. Isto é utilizado para facilitar a leitura do código e o que se pretende dele.

close all: fecha todas as figuras
clear all: apaga todas as variáveis
clc: limpa a janela de comandos

O plot é para fazer um gráfico. Mais tarde aprenderão mais sobre isto.
Para já é só para observar as funcionalidades do MatLab.

ACE

Aula 2

- Criar um vetor

Exemplos

Vetor Linha <i>escrevendo elemento a elemento</i>	Colocar entre [] os números separados por espaços ou vírgulas.	$v = [7, 8, 5, 9]$ } <i>resultado</i> $\longrightarrow [7 8 5 9]$ $v = [7 8 5 9]$
Vetor Coluna <i>escrevendo elemento a elemento</i>	Colocar entre [] os números separados por ponto-e-vírgula que faz mudar de linha, ou usar a transposta que pode ser obtida usando um ponto-plica. Esta permite transformar linhas em colunas e colunas em linhas.	$v = [7; 8; 5; 9]$ } <i>resultado</i> $\longrightarrow \begin{bmatrix} 7 \\ 8 \\ 5 \\ 9 \end{bmatrix}$ $v = [7 8 5 9].'$
Vetor de valores com espaçamento constante <i>onde se conhece o passo</i>	Usa-se dois pontos, segundo a estrutura: $v = \text{inicio} : \text{passo} : \text{fim}$ (se não for indicado o passo, considera-se 1)	$v = 0 : 3 : 9$ <i>resultado</i> $\longrightarrow [0 3 6 9]$ $v = 1 : 4$ <i>resultado</i> $\longrightarrow [1 2 3 4]$ $v = 9 : -3 : 0$ <i>resultado</i> $\longrightarrow [9 6 3 0]$
Vetor de valores com espaçamento constante <i>onde se conhece o número de pontos</i>	Usa-se a função linspace, , segundo a estrutura: $v = \text{linspace}(\text{inicio}, \text{fim}, \text{nr_pontos})$	$v = \text{linspace}(1, 7, 4)$ <i>resultado</i> $\longrightarrow [1 3 5 7]$ $v = \text{linspace}(0, 1, 3)$ <i>resultado</i> $\longrightarrow [0 0.5 1]$

- Ler/Escrever conteúdo de um vetor ou matriz

Exemplos

Vetor	Escrever o nome do vetor, e dentro de parenteses curvos escrever outro vetor com as posições/índices pretendidos. Usa-se <u>end</u> para representar o índice do último elemento.	Seja: $v = [7 8 5 9]$ índices: 1 2 3 4	$v(1)$ é 7 $v([2,4])$ é [8 9] $v([2,4]).'$ é $\begin{bmatrix} 8 \\ 9 \end{bmatrix}$ $v(1:3)$ é [7 8 5] $v(end)$ é 9 Se fizer: $v(2) = 1$, o novo v é [7 1 5 9]
Matriz	É igual à situação anterior, mas a duas dimensões. Dentro de parenteses curvos indica-se os índices para as linhas, vírgula, índices para as colunas. Utiliza-se <u>dois pontos</u> para obter uma linha/coluna completa. $v(\text{linhas}, \text{colunas})$	Seja: $M = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 7 & -2 \\ 9 & -5 & 6 \\ -8 & -7 & 4 \end{bmatrix}$	$M(1,1)$ é 3 $M([2,3], 1)$ é $\begin{bmatrix} 9 \\ -8 \end{bmatrix}$ $M(:,1)$ é $\begin{bmatrix} 3 \\ 9 \\ -8 \end{bmatrix}$ $M(2, [1 2])$ é [9 -5] $M(2, :)$ é [9 -5 6] $M(1, end)$ é -2

- Concatenar/Juntar vetores ou matrizes num só

Faz-se: <ul style="list-style-type: none"> • $[v1 \ v2]$ ou $[v1; \ v2]$ Para colocar os vetores/matrizes lado a lado • $[v1; \ v2]$ Para colocar $v2$ em baixo de $v1$. • As dimensões devem ser consistentes. 	Seja: $M = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $v1 = \begin{bmatrix} -5 \\ -1 \end{bmatrix}$ $v2 = [7 \ 7]$	$C1 = [M \ v1]$ <i>resultado</i> $\longrightarrow \begin{bmatrix} 1 & 2 & -5 \\ 3 & 4 & -1 \end{bmatrix}$ $C2 = [M; v2]$ <i>resultado</i> $\longrightarrow \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 7 & 7 \end{bmatrix}$	$C3 = [v1 \ v2.]'$ <i>resultado</i> $\longrightarrow \begin{bmatrix} -5 & 7 \\ -1 & 7 \end{bmatrix}$ $C4 = [v1 \ v2]$ Erro !!!
---	--	--	---

ACE
Aula 3

• Criação de algumas matrizes (com L linhas e C colunas)

Zeros zeros

Matriz de zeros	$M = \text{zeros}(L, C)$	$M = \text{zeros}(2, 3) \xrightarrow{\text{resultado}} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ $M = \text{zeros}(2) \xrightarrow{\text{resultado}} \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ $M = \text{zeros}(2, 3) + 5 \xrightarrow{\text{resultado}} \begin{bmatrix} 5 & 5 & 5 \\ 5 & 5 & 5 \end{bmatrix}$
Matriz de uns	$M = \text{ones}(L, C)$	$M = \text{ones}(2, 1) \xrightarrow{\text{resultado}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $M = \text{ones}(3) \xrightarrow{\text{resultado}} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ $M = \text{ones}(2, 3) * 3i \xrightarrow{\text{resultado}} \begin{bmatrix} 3i & 3i & 3i \\ 3i & 3i & 3i \end{bmatrix}$
Matriz identidade	$M = \text{eye}(L)$	$M = \text{eye}(3) \xrightarrow{\text{resultado}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
Matriz que é repetições de outra matriz	$M = \text{repmat}(v, y, x)$ ▪ v: matriz a ser repetida ▪ y: repetições na vertical ▪ x: repetições na horizontal	$v = [1 \ 2]$ $M = \text{repmat}(v, 2, 3) \xrightarrow{\text{resultado}} \begin{bmatrix} 1 & 2 & 1 & 2 & 1 & 2 \\ 1 & 2 & 1 & 2 & 1 & 2 \end{bmatrix}$

• Outros operadores/funções úteis

Raiz quadrada	$\text{sqrt}(x)$	$\text{sqrt}(9) \xrightarrow{\text{resultado}} 3$
Somatório	$\text{sum}(x)$	$x = [1 \ 2 \ 1 \ 6]$ $\text{sum}(x) \xrightarrow{\text{resultado}} 10$
Produtório	$\text{prod}(x)$	$x = [2 \ 2 \ 3]$ $\text{prod}(x) \xrightarrow{\text{resultado}} 12$
Operações elemento a elemento (divisão, multiplicação, exponenciação)	Coloca-se um ponto antes do operador: $.$ / $.$ *	$\text{Seja: } A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $A^2 \xrightarrow{\text{resultado}} \begin{bmatrix} 7 & 10 \\ 15 & 22 \end{bmatrix}$ $A.^2 \xrightarrow{\text{resultado}} \begin{bmatrix} 1 & 4 \\ 9 & 16 \end{bmatrix}$ Enquanto que sem ponto, o resultado é a matriz A ao quadrado, com ponto faz-se o quadrado de cada um dos elementos da matriz A.

ACE

Aula 4

- **Condições lógicas** (efetuar comparações)

Igual?	Diferente?	Menor?	Maior?	Menor ou igual?	Maior ou igual?	AND	OR	NOT
<code>==</code>	<code>~=</code>	<code><</code>	<code>></code>	<code><=</code>	<code>>=</code>	<code>&</code>	<code> </code>	<code>~</code>

Exemplos (para $x = [1 \ 2 \ 5 \ 2 \ 0]$)

$c = x == 2 \xrightarrow{\text{resultado}} [0 \ 1 \ 0 \ 1 \ 0]$	$c = x < 2 \xrightarrow{\text{resultado}} [1 \ 0 \ 0 \ 0 \ 1]$
$c = x == 2 x < 2 \xrightarrow{\text{resultado}} [1 \ 1 \ 0 \ 1 \ 1]$	$c = \sim(x == 2) \xrightarrow{\text{resultado}} [1 \ 0 \ 1 \ 0 \ 1]$

O resultado de uma condição lógica é um vetor com 1, se a condição for verdadeira, e 0 se a condição for falsa. Os primeiros operadores servem para comparar variáveis. Daqui podemos ter diferentes condições. Se queremos que o resultado final dê Verdadeiro quando uma condição é verdadeira e a outra também, usa-se um AND, se queremos que dê Verdadeira quando uma condição ou outra seja Verdadeira, usa-se um OR, se quisermos trocar/negar e passar Falso a Verdadeiro e Verdadeiro a Falso, usa-se um NOT.

- **Funções úteis**

y = reshape(M,L,C)	Coloca a matriz M com dimensões LxC	$M = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $y = reshape(M, 1, 4) \xrightarrow{\text{resul}} [1 \ 3 \ 2 \ 4]$
y = round(x)	Arredonda para o inteiro mais próximo	$round([1.5 \ 4.1 \ 2.9]) \xrightarrow{\text{resul}} [2 \ 4 \ 3]$
r = rem(x,n)	Resto da divisão inteira de x por n	$rem([1 \ 2 \ 3 \ 4], 3) \xrightarrow{\text{resul}} [1 \ 2 \ 0 \ 1]$
indices = find(cond_logica)	Retorna os índices da matriz onde a condição lógica é verdadeira	$x = [1 \ 2 \ 5 \ 2 \ 0];$ $ind = find(x == 2) \xrightarrow{\text{resul}} [2 \ 4]$
y = log(x)	Faz o logaritmo neperiano	$\log(2.7) \xrightarrow{\text{resul}} 0.9933$
y = log10(x)	Faz o logaritmo de base 10	$\log_{10}(100) \xrightarrow{\text{resul}} 2$
y = abs(x)	Cálcula o modulo (para reais e complexos)	$abs([-5 \ 3 + 4i]) \xrightarrow{\text{resul}} [5 \ 5]$
n = length(x)	Retorna a dimensão da matriz x (Se a matriz é retangular, dá o número de linhas)	$v = [1 \ 2 \ 2 \ 7] \quad M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ $length(v) \xrightarrow{\text{resul}} 4$ $length(M) \xrightarrow{\text{resul}} 2$
n = size(x,N)	Retorna a dimensão da matriz x (N = 1 dá as linhas, N = 2 dá as colunas)	$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$ $size(M, 1) \xrightarrow{\text{resul}} 2$ $size(M, 2) \xrightarrow{\text{resul}} 3$

• Polinómios

$$\text{Seja: } y = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = a_n(x - r_n)(x - r_{n-1}) \dots (x - r_0)$$

Considere-se o vetor de coeficientes do polinómio $a = [a_n \ a_{n-1} \ \dots \ a_1 \ a_0]$ e o vetor de raízes do polinómio $r = [r_n \ r_{n-1} \ \dots \ r_1 \ r_0]$. As funções MatLab que permitem obter uns vetores através de outros conhecidos são:

$a = \text{poly}(r)$	$y = \text{polyval}(a, x)$	$r = \text{roots}(a)$
		Ex: $y = 2x^3 + x - 1$ $x = \text{linspace}(-10, 10, 100)$ $y = \text{polyval}(a, x)$

• Gráficos 2D usando a função plot

<code>figure</code>	Abre uma nova janela de gráficos
<code>plot(x1, y1, 'cor e traço 1', x2, y2, 'cor e traço 2', ...)</code>	Cria um gráfico com as curvas $(x1, y1), (x2, y2), \dots$, com o tipo de traço e a cor indicados.
<code>hold on</code>	Para colocar várias curvas num gráfico, pode-se fazer um plot, fazer <code>hold on</code> , e as curvas dos seguintes plot's aparecerão sobrepostas.
<code>legend('texto1', 'texto2', ...)</code>	Fazer a legenda. A ordem texto1, texto2, ..., é igual à ordem com que se adicionou as curvas.
<code>axis([x_min x_max y_min y_max])</code>	Definir os eixos do gráfico.
<code>title('Texto para o título')</code>	Colocar um título no gráfico.
<code>xlabel('textoX')</code>	Legenda do eixo dos xx.
<code>ylabel('textoY')</code>	Legenda do eixo dos yy.
<code>grid on</code>	Colocar uma grelha no gráfico.
<code>[xin, yin] = ginput(N)</code>	Após apresentar o gráfico, o utilizador deverá clicar com o rato em N pontos do gráfico que serão guardados em xin e yin.

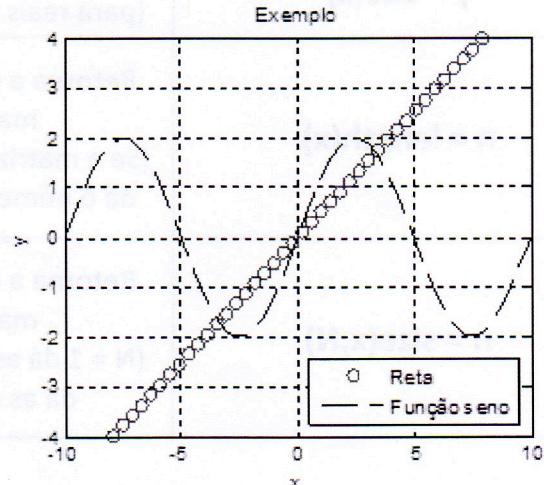
Símbolos para definir as cores e o tipo de traço das curvas:

Cor	Marcadores	Linhas	
y amarelo	.	v triângulo (cima)	
m rosa	ponto	^- triângulo (baixo)	
c azul claro	o círculo	< triângulo (esquerda)	
r encarnado	x marca x	> triângulo (direita)	
g verde	+	p pentagrama	
b azul	*	h "hexagram"	
w branco	s quadrado	- linha a cheio	
k preto	d diamante	:	ponteada
		-.	traço ponto
		--	tracejada

```

x = linspace(-10, 10, 50);
y1 = 0.5*x;
y2 = 2*sin(2*pi*0.1*x);
plot(x, y1, 'ro')
hold on
plot(x, y2, 'b--')
grid on
xlabel('x')
ylabel('y')
title('Exemplo')
legend('Reta', 'Função seno')
axis([-10 10 -4 4])

```



- Mais funções usadas em gráficos

hold off	É o oposto do hold on. Usando a seguir um plot, a nova curva já não irá sobrepor-se às anteriores.
subplot(L,C,N)	Divide uma janela gráfica em L linhas e C colunas. O plot que se fizer a seguir fará aparecer um gráfico na N ^º divisória.
axis square	Coloca os eixos com o mesmo comprimento (fica um quadrado)
semilogx(x,y,'corEtraco')	Gráfico com o eixo xx com escala logarítmica.
semilogy(x,y,'corEtraco')	Gráfico com o eixo yy com escala logarítmica.
loglog(x,y,'corEtraco')	Gráfico com ambos os eixos com escala logarítmica.

- Funções para utilizar com strings (texto)

s = strcat(string1, string2, ...)	Permite juntar diferentes strings numa só.
s = num2str(numero)	Converte um número numa string.

- Outras funções

exp(x)	Faz e^x
a = real(z)	Permite obter a parte real de um número complexo.
b = imag(z)	Permite obter a parte imaginária de um número complexo.
x = logspace(inicial,final,nr_pontos)	Semelhante ao linspace, mas depois faz $10^{inicial}, \dots, 10^{final}$, exemplo: $x = logspace(1,3,3) \xrightarrow{resultado} [10 \ 100 \ 1000]$

- Para pensar...

$$\begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} * [1 \ 1 \ 1 \ 1 \ 1 \ 1] = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 & 4 & 4 \end{bmatrix}$$

Exemplo de código

```

x = linspace(2,100,20);
n = [1 2 3]';

y1 = n*x;
subplot(3,2,1); plot(x,y1)

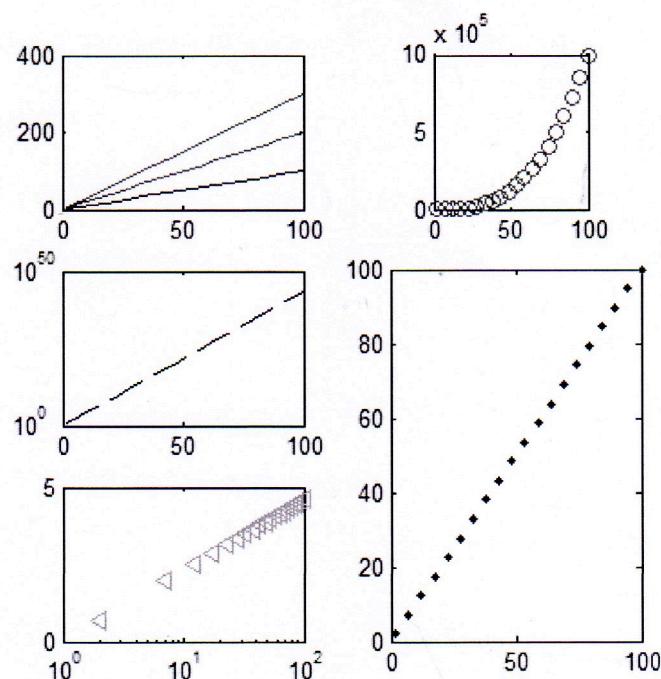
y2 = x.^3;
subplot(3,2,2); plot(x,y2, 'ro')
axis square

y3 = exp(x)
subplot(3,2,3); semilogy(x,y3, 'k--')

y5 = log(x)
subplot(3,2,5); semilogx(x,y5, 'g<')

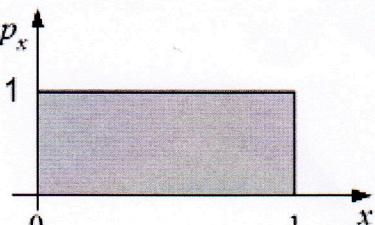
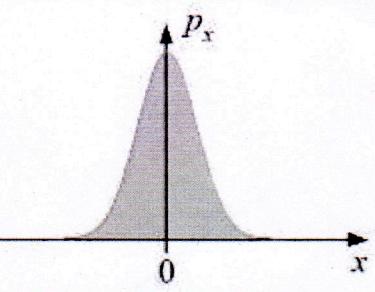
y46 = x;
subplot(3,2,[4 6]); plot(x,y46, '.')

```



$$8 * \underbrace{\text{rand}(1,1) - 3}_{\text{de } 0 \text{ a } 1} + \underbrace{3}_{\text{de } 8}$$

• Variáveis aleatórias

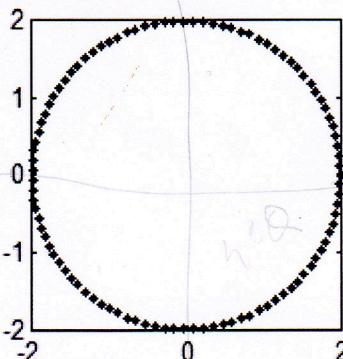
Números aleatórios com distribuição uniforme		$x = \text{rand}(L, C)$ <i>x será uma matriz de L linhas e C colunas com números entre 0 e 1, com probabilidade uniforme</i>	$x = (x_{\max} - x_{\min}) * \text{rand}(L, C) + x_{\min}$ <i>Para se generalizar para um intervalo x_{\min} a x_{\max}, pode-se utilizar a expressão indicada.</i>
Números aleatórios com distribuição normal		$x = \text{randn}(L, C)$ <i>x será uma matriz de L linhas e C colunas com distribuição normal de média $\mu=0$, desvio padrão $\sigma=1$ e variância $\sigma^2=1$</i>	$x = \sigma * \text{randn}(L, C) + \mu$ <i>Para se generalizar para uma média qualquer μ e um desvio padrão qualquer σ, pode-se utilizar a expressão indicada.</i>

Funções úteis:

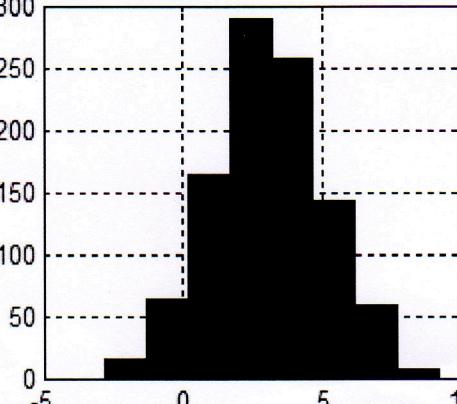
Média:	Desvio Padrão:	Variância:	Número de elementos de x
<code>mean(x)</code>	<code>std(x)</code>	<code>var(x)</code>	<code>numel(x)</code>

Ex: média dos 10 últimos elementos: `mean(x(end-9:end))`

• Desenhar um círculo

Um número complexo pode ser escrito na forma $z = r e^{i\theta}$. Pode-se desenhar um círculo de raio r , fazendo um varrimento de todos os ângulos entre 0 e 2π , dando assim uma volta completa. O resultado de <code>abs(z)</code> será r e o resultado de <code>angle(z)</code> será θ .	$r = 2;$ $teta = linspace(0, 2*pi, 100);$ $z = r * exp(i*teta);$ <code>plot(z, '.')</code> <code>axis square</code>	
---	---	---

• Fazer um histograma

Considerando uma variável aleatória X e um histograma com N barras, este pode ser feito usando: <code>hist(X, N)</code>	$X = 2 * \text{randn}(1, 1000) + 3;$ <code>hist(X, 8)</code> <code>grid on</code> <i>nº de barras</i>	
--	--	---

ACE

Aula 7

- Método de Monte Carlo:** Geração de uma elevada amostragem aleatória, com posterior análise estatística.

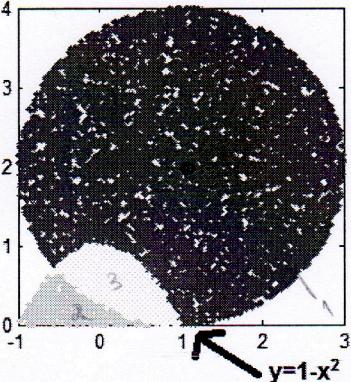
Exemplo 1: Para se estimar a probabilidade de sair cara ou coroa numa moeda, atirou-se a moeda ao ar 1000 vezes. Saiu 508 vezes cara e 492 vezes coroa, logo estima-se que $P(\text{cara}) = 50.8\%$ e $P(\text{coroa}) = 49.2\%$.

Exemplo 2: Para se determinar a área de uma região, considerou-se um rectângulo que envolve-se toda a região. Esse rectângulo tem área 10. Gerou-se 1000 pontos aleatórios pertencentes ao rectângulo. Para se ter uma perspectiva visual, imagine-se que esse rectângulo é um alvo, e os pontos são o sítio onde se acertou com a seta no jogo do tiro ao alvo, mas com alvo rectangular. Escreveu-se as condições lógicas, e determinou-se que desses 1000 pontos, 500 estavam dentro da região, e 500 estavam fora da região. Como o rectângulo tem área 10, estima-se que a área da região pretendida é 5.

Fórmulas úteis:

$A_{\text{pretendida}} = A_{\text{retângulo}} \frac{\text{Número de pontos na região pretendida}}{\text{Número total de pontos}}$
Equação de uma circunferência de centro (c_x, c_y) e raio r : $(x - c_x)^2 + (y - c_y)^2 = r^2$

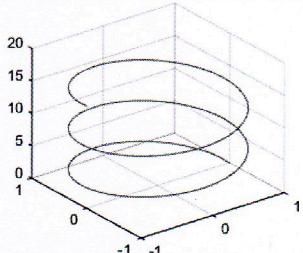
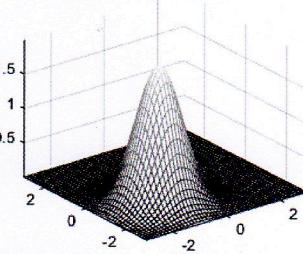
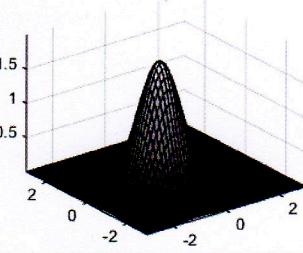
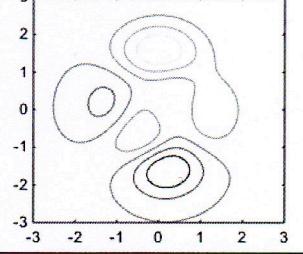
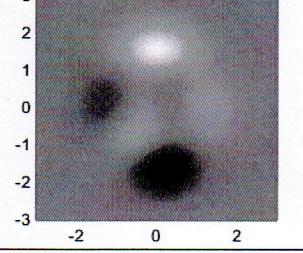
Exemplo de Código MatLab: Determine a área de região amarela

 $y = 1 - x^2$	1º) Gerar um conjunto de pontos aleatórios dentro de um rectângulo que inclua a região pretendida. <i>No exemplo o rectângulo será com x entre -1 a 3, e y entre 0 a 4.</i>	$\text{abs}(z) < \frac{\pi}{4}$ e $\text{abs}(z) > \frac{\pi}{4}$ $x^2 + y^2 < 1$ $(0,0)$ $C(1,2)$ <pre> N = 1e4; xa = 4*rand(1,N)-1; ya = 4*rand(1,N); za = xa + 1i*ya; </pre>
2º) Escrever as condições lógicas que definem cada elemento da figura.	3º) Fazer o gráfico para confirmar que está certo.	<pre> 1Cond_circulo = (xa-1).^2+(ya-2).^2 <= 2.^2; 2cond_funcao = ya <= 1 - xa.^2; 3cond_Regiao = cond_circulo & cond_funcao; plot(xa(cond_circulo),ya(cond_circulo),'r.') hold on plot(xa(cond_funcao),ya(cond_funcao), 'g.') plot(xa(cond_Regiao),ya(cond_Regiao), 'y.') </pre>
4º) Calcular a estimativa da área da região		$A_{\text{retângulo}} = 4 \times 4;$ $A = A_{\text{retângulo}} \times \text{sum}(\text{cond_Regiao}) / N$

- Resolver sistemas de equações**

1º) Organizar as equações (manualmente) para se ter as variáveis todas pela mesma ordem.	$\begin{cases} -2x_2 - 5i + 2ix_1 = -3x_1 + 7 \\ x_1 - 10 = 0 \end{cases} \Leftrightarrow \begin{cases} (2i+3)x_1 - 2x_2 = 5i + 7 \\ x_1 + 0x_2 = 10 \end{cases}$
2º) Escrever as matrizes do sistema no MatLab.	$A = [2i+3 \ -2 \ ; \ 1 \ 0];$ $b = [5i+7; \ 10];$
3º) Resolver o sistema ($Ax = b \Leftrightarrow x = A^{-1}b$).	$x = A^{-1}b$ ou $A \backslash b$

- Tipos de gráficos que representam 3 dimensões

Gráfico obtido por simples colocação dos pontos: plot3(x,y,z)	Desenha num espaço a 3 dimensões um conjunto de pontos em que se conhecem as coordenadas (x,y,z).	
Gráfico de Superfície: mesh(x2,y2,z2)		
Gráfico de Superfície: surf(x2,y2,z2)	Representado por uma superfície num espaço tridimensional. O mesh é apenas como uma rede, enquanto no surf a superfície é totalmente colorida. Se nada disserem, sugiro que optem pelo surf.	
Curvas de Nível: contour(x2,y2,z2,N)	Curvas que unem pontos de igual valor (isolinhas). A variável N indica o número dessas curvas.	
Gráficos de pseudo-cor: pcolor(x2,y2,z2)	Gráfico cuja cor representa um dado valor de z.	

y	(x ₁ ,y ₃)	(x ₂ ,y ₃)	(x ₃ ,y ₃)
	(x ₁ ,y ₂)	(x ₂ ,y ₂)	(x ₃ ,y ₂)
	(x ₁ ,y ₁)	(x ₂ ,y ₁)	(x ₃ ,y ₁)
x	É preciso construir-se pares de valores de (x,y), e para cada par, calcular o z. Simplificadamente, antes de se utilizar as funções mesh, surf, contour ou pcolor, usar x2 e y2 obtidos pela seguinte instrução: [x2 y2] = meshgrid(x,y)		

- Opções utilizadas nos gráficos

axis([x_{min} x_{max} y_{min} y_{max} z_{min} z_{max}])	Permite definir os eixos manualmente.
axis tight	Ajusta os eixos tendo em conta as gamas de x,y e z.
axis equal	Coloca os eixos com a mesma escala.
shading interp	Sombreado interpolado, para que as cores variem suavemente.
colorbar	Coloca uma barra de cores associada aos valores que representa.

ACE

Aula 9

- Instrução condicional e ciclos de repetição

se tivermos
várias condi-
ções positivas,
apenas é
executada
a 1ª verda-
deira

Optional

<p>Instrução condicional</p> <p>Testa cada condição até achar uma verdadeira, executa o pedaço de código selecionado, e depois "salta fora". Se for tudo falso, executa o que está dentro do else.</p> <pre> if cond1 Se cond1 verdadeira, faz o código que puseres aqui. elseif cond2 Se cond1 falsa, cond2 verdadeira, faz o código que puseres aqui. elseif cond3 Se cond1, cond2 falsa, cond3 verdadeira, faz o código que puseres aqui. elseif cond4 Se cond1, cond2, cond3 falsa, cond4 verdadeira, faz o código que puseres aqui. else Se é tudo falso, faz o código que puseres aqui. end </pre>	<p>Ciclo for</p> <p>Executa o código para cada um dos valores do vector indicado.</p> <pre> for vector Faz repetidamente o código que aqui escreveres. end </pre>
	<p>Ciclo while</p> <p>Executa o código enquanto a condição for verdadeira.</p> <pre> while condicao Faz repetidamente o código que aqui escreveres. end </pre>

Exemplos:

```

>> nota = 20;

if nota >= 9.5
    disp('Positiva')
elseif nota == 20
    disp('És um rei!')
else
    disp('Negativa')
end

Positiva → P9 é a

```

1º q aparece q é verdadeira

```

>> for i=1:2:6
    rem(i,3)
end

```

vai repetir

ans =

1 1 e P15

1/3 → resto=1

ans =

2 0 resto0

2/3 → resto0

```

>> i=1;
while i<=6
    rem(i,3)
    i = i + 2;
end

```

ans =

i=1

1 1 resto1

1/3 → resto1

ans =

i=1+2=3

0 3 resto0

0/3 → resto0

ans =

i=3+2=5

2 5 resto2

2/3 → resto2

ans =

i=5+2=7

7/6 → resto3

é falsa e já não executa

- Outras funções

<code>x = input('Aparece isto escrito')</code>	Faz aparecer escrito o texto indicado e lê do teclado o que for digitado.
<code>disp('Aparece isto escrito')</code>	Faz aparecer escrito o texto indicado.
<code>y = fix(x)</code>	Arredonda para o inteiro mais próximo de 0. Ex: fix(-1.9) é -1; fix(2.1) é 2.

ACE

Aula 10

- Criar e utilizar uma função nova

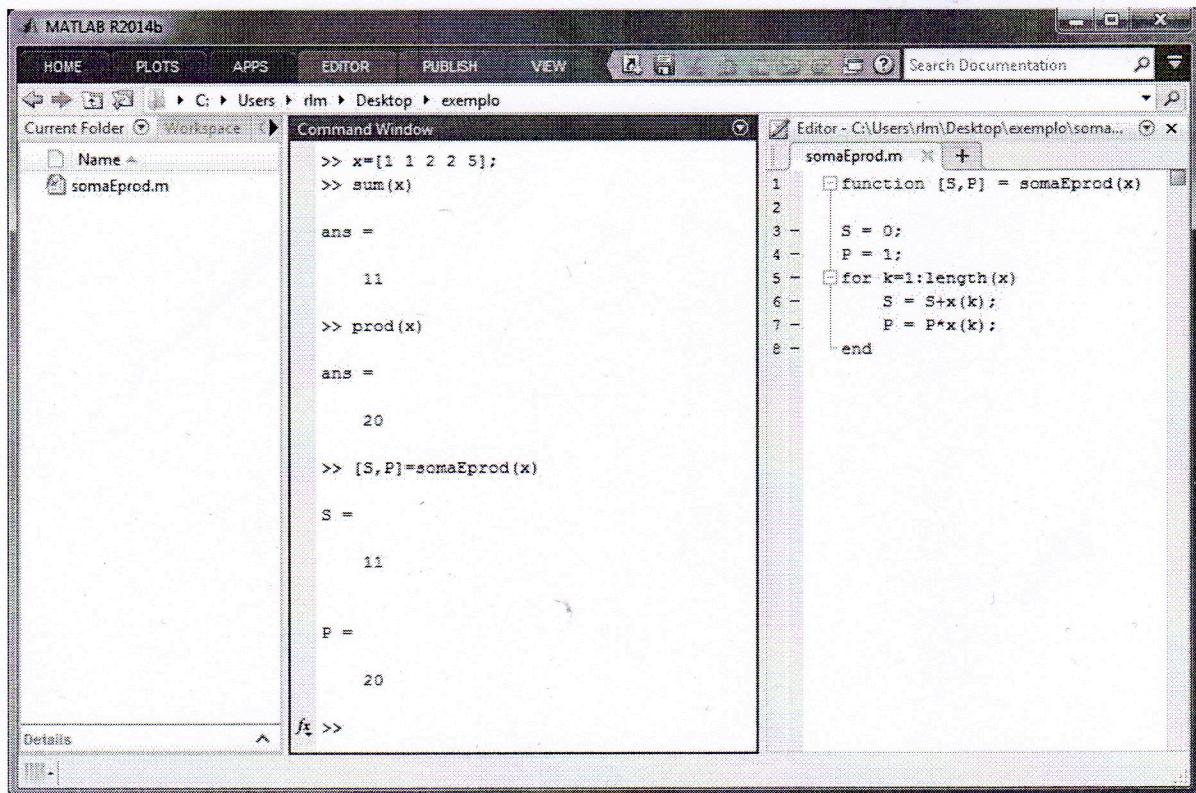
- 1) Abrir um novo script.
- 2) Escrever o código da função e guardar o ficheiro *nomeDaFuncao.m*.

```
function [y1 y2 ... yN] = nomeDaFuncao(x1, x2, ..., xK)
```

% Escrever aqui um código de MatLab da forma habitual

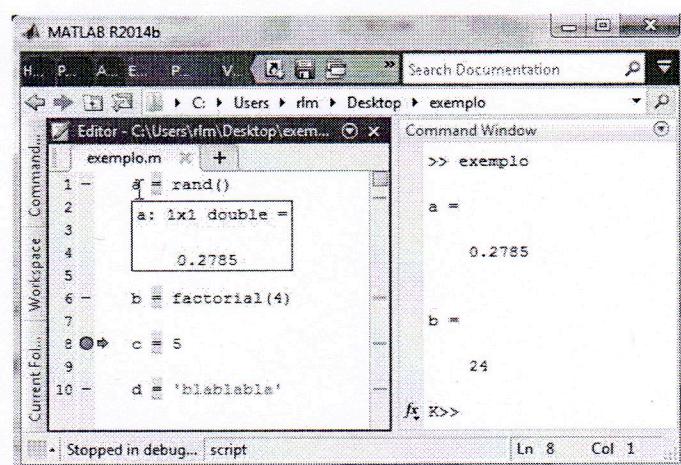
```
end
```

- 3) Utilizar a função normalmente como se fosse uma vulgar função do MatLab, no entanto o ficheiro da função deve estar guardado na pasta de trabalho.



- Usar debug (útil na procura de erros)

- 1) Clicar ao lado do número da linha (irá aparecer uma bola vermelha)
- 2) Executar o programa. Este irá parar nas bolas vermelhas.
- 3) Passar o rato por cima das variáveis para ver o seu valor, ou executar comandos normalmente para avaliar os resultados.
- 4) Carregar em F5 para continuar.



$n! \rightarrow factorial(n)$