Gráficos bidimensionais

Introdução

- Já vimos vários gráficos em exemplos anteriores. Veremos agora, com em mais detalhes, as principais ferramentas que o **MATLAB** disponibiliza para manipulação de gráficos.
- Vários dos recursos que serão apresentados estão disponíveis através da janela *Figure*,
  - como itens de menu no topo da janela *Figure*;
  - como botões na barra de ferramentas da janela *Figure*;
  - como botões na barra de ferramentas da janela *Camera*, acessível pelo menu **View** da janela *Figure*.
- Veja *Help –> Matlab Help –> Matlab –> Graphics*
- Acompanhe as transparências deste capítulo observando e executando o arquivo mm25demo.m.

Introdução

- Vc pode usar a barra de ferramentas e os recursos do menu da janela *Figure* se pretende ajustar uma única figura, ou utilizar as funções na janela de comandos para automatizar o processo de ajuste de gráficos. Daremos exemplos de ambos aqui.
- A seção 25.13 do livro traz uma tabela resumo com as principais funções do **MATLAB** para gráficos bidimensionais. Você encontra essas mesmas funções em *Help –> Matlab Help –> Matlab –> Functions - by Category –> Graphics*.

Comando $plot$

- Comando mais comum para a elaboração de gráficos bidimensionais;
- Cria gráficos a partir de vetores de dados em eixos adequados, conectando os pontos com linhas retas.
- Considere nosso primeiro exemplo:

```
:::::::::::::::::
:::::::::::::::::
mm2501.m
:::::::::::::::::
:::::::::::::::::
x = linspace(0,2*pi,30);
y = sin(x);
plot(x,y), title('Figure 25.1: Sine Wave')
```

- O comando $plot$ abre a janela *Figure*, ajusta os eixos para acomodar os dados, marca os pontos e os conecta com linhas retas.
- Se já houver uma janela *Figure* aberta, o gráfico existente é removido e o novo gráfico é traçado.

Comando $plot$

- O comando $plot$ permite que vários gráficos diferentes sejam desenhados na mesma janela *Figure* (gráficos simultâneos). Para isso basta fornecer à função outro par de argumentos. O exemplo abaixo traça o seno e o coseno simultaneamente.

```
:::::::::::::::::
mm2502.m
:::::::::::::::::
x = linspace(0,2*pi,30);
y = sin(x);
z = cos(x);
plot(x,y,x,z)
title('Figure 25.2: Sine and Cosine')
```

- É possível adicionar vários conjuntos de argumentos desta forma.

Comando $plot$

- Se um dos argumentos for uma matriz e o outro um vetor, o comando $plot$ representa cada coluna *versus* o vetor. Assim as linhas abaixo reproduzem o gráfico anterior.

```
>> W = [y;z];
>> plot(x,W)
```

- Alteração na ordem dos argumentos: rotação de 90 graus.

```
::::::::::::::::::
mm2503.m
::::::::::::::::::
x = linspace(0,2*pi,30);
y = sin(x);
z = cos(x);
W=[y;z];
plot(W,x) % plot x vs. the columns of W
title('Figure 25.3: Change Argument Order')
```

Comando $plot$ com argumento único

- Depende do tipo dos argumentos. Considere $plot(y)$. Se y contiver
  - valores complexos $\Rightarrow plot(real(y), imag(y))$.

```
>> y = linspace(1,10,30)+ logspace(1,2,30)*i;
>> plot(y)
```

  - apenas valores reais $\Rightarrow plot(1 : length(y), y)$.

```
>> y = y - logspace(1,2,30)*i ;
>> plot(y)
```

  - Estas regras valem quando y é matriz, coluna a coluna. Explique este exemplo.

```
x = linspace(0,2*pi,30); y = sin(x); z = cos(x); W=[y;z];
plot(W) % plot columns of W
```

Estilos de linha, marcadores e cores

- Default:
  - Estilo de linha: contínuo;
  - Cores: A primeira cor utilizada é o azul, seguindo ciclicamente, para os gráficos simultâneos, a ordem na tabela em *help plot* (e seção 25.2 do livro).
  - Marcador: Não há marcador padrão.
- Para alterar o padrão:
  - utiliza-se um terceiro argumento em *plot*; esse argumento é uma cadeia de um ou mais caracteres da tabela citada, inserida após cada par de argumentos a serem plotados.
- Uma especificação de cor aplica-se tanto ao estilo da linha quanto ao marcador. Para que tenham cores diferentes é necessário fazer dois gráficos diferentes com os mesmos dados, modificando as cores. Veja o exemplo.

$$plot(x, y, 'b*', x, y, 'm-')$$

Comandos úteis

- $Grid\ \ on/off$: Adidiciona e remove o quadriculado de fundo do gráfico. Quando emitido sem on/off alterna entre os dois. Default é $grid\ \ off$.
- $Box\ \ on/off$: O gráfico normalmente está inserindo em uma caixa, cujas arestas inferior e lateral esquerda representam os eixos ($box\ \ on$). Remover esta caixa significa remover as arestas superior e lateral direita. *Default:* $Box\ \ on$.
- $xlabel('nomeX')$ e $ylabel('nomeY')$: atribui nomes aos eixos x e y respectivamente.
- $title('FigureXYZ')$: adiciona um título à figura. O título será exibido no topo do gráfico.

Comandos úteis

- $text(x, y, 'Texto\ \ desejado')$: insere 'Texto desejado' na coordenada (x, y), justificado à esquerda. Veja o efeito de $text(3, 0.4, 'seno(x)')$.
- $gtext('Texto\ \ desejado')$: insere 'Texto desejado', no local especificado por um clique do mouse, cujo cursor será exibido na figura por um grande $+$. Veja o efeito de $gtext('seno(x)')$.
- Veja funcionalidade similar a *gtext* na janela da figura, no menu *Insert –> Text* ou clicando no botão 'A'.

Personalização de eixos: $axis$

- O comando *axis* permite um ajuste fino das proporções e aparências dos eixos horizontal e vertical. Deve ser usado após o comando *plot*.
- Possui vários tipos de argumentos e formas de uso. As mais comuns são
  - *axis([xmin xmax ymin ymax])* que estabelece limites para para os valores dos eixos x e y;
  - *axis 'opções'*, onde 'opções' podem ser *auto* (escalonamento automático), *on/off* (ativa ou desativa rotulação, demarcação, etc, dos eixos) e outras, encontradas em *help axis*, ou na seção 25.4 do livro.
- O comando $axis$ altera todos os eixos simultaneamente. Se o que se quer é alterar apenas um dos eixos, é melhor usar os comandos $xlim$, $ylim$, $zlim$.
- Veja exemplos de uso no arquivo 'mm25demo.m'.

Gráficos múltiplos

Suponha que haja um gráfico ativo e se queira inserir um novo gráfico na mesma janela, sem que o anterior seja removido. Para isso usa-se o comando $hold$.

- $hold\ \ on/off$:
  - Quando ativado não remove os eixos existentes quando um novo gráfico é introduzido. Entretanto pode ajustar as escalas dos eixos quando houver necessidade considerando o novo conjunto de dados.
  - Quando desativado libera a janela para novos gráficos.
  - O comando $hold$ sem argumentos alterna o valor atribuído a $hold$.
  - *Default: hold off*.

hold - exemplo

```
:::::::::::::::::
:::::::::::::::::
mm2508.m
:::::::::::::::::
:::::::::::::::::
x = linspace(0,2*pi,30);
x = linspace(0,2*pi,30);
y = sin(x);
z = cos(x);
plot(x,y)
hold on
ishold  % return 1 (True) if hold is ON
plot(x,z,'m')
hold off
ishold  % hold is no longer ON
title 'Figure 25.8: Use of hold command'
```

Observações

- Especificação da segunda cor.
- Título sem parêntesis (comando, não função: dualidade comando/função).

Gráficos em janelas diferentes

- É possível criar várias janelas *Figure*, cada uma com seu próprio conjunto de dados.
  - Toda janela é criada na mesma posição, mova-as para vê-las simultaneamente.
- Para criar uma janela *Figure*:
  - use o comando $figure$ na janela de comandos; ou
  - selecione a opção **New Figure** do menu **File** na janela de comandos ou na janela *Figure*.
- Cada janela criada possui um número (seu *handle*), que é usado para identificar aquela janela.
- Quando uma janela é criada, esta se torna a janela ativa. A cada momento apenas uma janela pode estar ativa.

Janelas múltiplas

- Apenas a janela ativa responde aos comandos $axis$, $hold$, $ylabel$, $xlabel$, $title$, $grid$ e box.
- A seleção de uma janela pode ser feita via *mouse* ou utilizando o comando $figure(n)$, onde n é o seu *handle*.
- Assim, o comando $figure$ tem duas funções:
  - Sem argumentos: Cria uma nova janela cujo *handle* é o primeiro disponível.
  - $figure(n)$: Se o *handle* n existe, o comando torna esta janela ativa, caso contrário cria uma nova janela com este *handle*. (Note que isto permite que sejam criadas janelas com *handles* não consecutivos.)

Janelas múltiplas

- Para remover janelas de gráficos podemos:
  - Usar o *mouse* da forma padrão, janela a janela;
  - Usar o comando $close$ com as seguintes sintaxes:
    - Sem argumentos: fecha a janela ativa;
    - $close(n)$: fecha a janela cujo *handle* é n (retorna msg de erro caso o *handle* não exista.);
    - $close\ \ all$: remove todas as janelas.
- Para limpar o conteúdo de uma janela sem fechá-la utiliza-se clf. Pode ser usado sem argumentos (atua na janela ativa) ou com um handle.
- O comando $reset$ restaura as propriedades da figura (exceto a posição) para os seus valores padrão. Pode ser usado com ou sem *handle* como nos outros comandos.

Subgráficos

- O comando $subplot(m, n, p)$ subdivide a janela de gráfico ativa em $m * n$ subgráficos, dispostos como uma matriz $m \times n$. Ademais, torna o p-ésimo subgráfico ativo.
- A numeração dos subgráficos é feita da esquerda para direita na linha e partir da primeira linha.
- Quando um subgráfico está ativo, apenas este subgráfico responde aos comandos $axis$, $hold$, $ylabel$, $xlabel$, $title$, $grid$ e box.
- Um subgráfico permanece ativo até que outro subgráfico, ou figura, seja ativado.
- Quando um comando $subplot$ modifica o número de subgráficos que a janela possui, os subgráficos anteriores são removidos para ajustar o novo gráfico.
- Para voltar a usar a janela toda como um gráfico único usa-se o comando $subplot(1, 1, 1)$ (ou clf).

Subgráficos - exemplo

```
:::::::::::::::::
:::::::::::::::::
mm2509.m
:::::::::::::::::
x = linspace(0,2*pi,30);
y = sin(x);
z = cos(x);
a = 2*sin(x).*cos(x);
b = sin(x)./(cos(x)+eps);
subplot(2,2,1) % pick the upper left of a 2-by-2 grid of subplots
plot(x,y), axis([0 2*pi -1 1]), title('Figure 25.09a: sin(x)')
subplot(2,2,2) % pick the upper right of the 4 subplots
plot(x,z), axis([0 2*pi -1 1]), title('Figure 25.09b: cos(x)')
subplot(2,2,3) % pick the lower left of the 4 subplots
plot(x,a), axis([0 2*pi -1 1]), title('Figure 25.09c: 2sin(x)cos(x)')
subplot(2,2,4) % pick the lower right of the 4 subplots
plot(x,b), axis([0 2*pi -20 20]), title('Figure 25.09d: sin(x)/cos(x)')
```

Ferramentas interativas

Ferramentas para anotações em gráficos, que surgiram antes que as facilidades de barra de menus e barra de ferramentas existissem.

- $legend$: Cria uma caixa de legenda no gráfico. Recebe uma string para associar a cada um dos gráficos (simultâneos) daquela janela.

```
.....................
.....................
mm2510.m
.....................
.....................
x = linspace(0,2*pi,30);
y = sin(x);
z = cos(x);
plot(x,y,x,z)
legend('sin(x)','cos(x)')
title('Figure 25.10: Legend Example')
```

- Para mover a caixa de legendas utilize o *mouse*.

Ferramentas interativas- $zoom$

- $zoom\ \ on/off$: Ativa e desativa o modo de zoom. Uma vez ativado, cada click do mouse expande o gráfico de um fator de 2. Podemos selecionar uma área específica selecionando uma região retangular com o mouse.
  - O comando sem argumentos alterna o modo de zoom;
  - $zoom\ \ out$: Retorna ao estado inicial, mas *não* desativa o modo de zoom;
  - $zoom\ \ n$: expande por um fator de n.

Ferramentas interativas - $ginput$

- $ginput$: É utilizada para selecionar pontos (coordenadas (x, y)) específicos do gráfico.
  - $[x, y] = ginput(n)$: Seleciona n pontos do gráfico, retornando-os nos vetores x e y onde $(x(i), y(i))$ são as coordenadas do i-ésimo ponto. A seleção é feita utilizando-se o mouse.
    - A tecla **Enter** pode ser utlizada para interromper a seleção antes que todos os n pontos tenham sido selecionados.
  - $[x, y] = ginput$: seleciona um número arbitrário de pontos. A seleção termina utilizando-se a tecla **Enter**.

Ferramentas interativas - $ginput$

- Se a parte $[x, y]$ do comando for omitida o resultado é retornado na variável de saída padrão (ans), no formato de uma matrix $n \times 2$, onde n é o número de pontos efetivamente selecionados.
- Os pontos selecionados não necessariamente são pontos do gráfico, mas aqueles que o mouse selecionou explicitamente.
- Se um ponto selecionado estiver fora do eixo, o seu valor é calculado por extrapolação.
- No caso de subgráficos, os pontos selecionados são referentes ao subgráfico ativo. Isto siginifica que, dado que o i-ésimo subgrafo esteja ativo, se o click é feito no j-ésimo subgráfico, $i \neq j$, o cálculo é feito referente aos eixos do i-ésimo subgráfico. Veja o exemplo no arquivo mm25demo.m

Atualização da tela

- A atualização da tela não é sempre feita, já que é um processo demorado.
- Por exemplo:
  - se $plot(x, y)$, $axis[xmin\ xmax\ ymin\ ymax]$ e $grid$ são digitados um por linha, o **MATLAB** atualizará após cada um destes comandos.
  - Entretanto se os mesmos forem digitados na mesma linha, separados por vírgulas, o **MATLAB** atualizará a tela apenas uma vez, quando o *prompt* reaparece.
- No caso de M-files, mesmo se os comandos aparecerem em linhas separadas, a tela é atualizada apenas uma vez, quando todos os comandos forem completados e o *prompt* do **MATLAB** reaparecer.

Atualização de tela

- Existem cinco eventos principais que, em geral, causam atualização da tela:

  1. Retorno do *prompt* do **MATLAB**;
  2. Funções que temporariamente interrompem a execução tais como $pause$, $keyboard$, $input$, $waitforbuttonpress$;
  3. Execução do comando $getframe$;
  4. Execução do comando $drawnow$;
  5. Alteração no tamanho da janela *Figure*.

Funções especializadas

- Em algumas situações, escalas lineares nos eixos podem não ser adequadas. O **MATLAB** oferece algumas opções para lidar com esta necessidade.
  - $semilogx$: usada para fazer gráficos com escala logarítmica (base 10) no eixo x;
  - $semilogy$: escala logarítmica (base 10) no eixo y;
  - $loglog$: escala logarítmica (base 10) em ambos os eixos;

Todos os recursos da função $plot$ discutidos anteriormente aplicam-se à estas funções.

Gráficos especializados

- $area$: Quando x, y são ambos vetores, $area(x, y)$ constrói gráficos da mesma forma que $plot$, mas preenche a área sob o gráfico com alguma cor. Útil para construir gráficos hachurados. Veja um exemplo clicando o mouse na figura corrente.
- O limite inferior para a área pode ser especificado; se omitido é assumido como 0.
- Quando y for uma matriz, a função traçada e hachurada para cada coluna i de x corresponde ao acumulado das colunas de y de 1 até i. Veja o exemplo em $mm2511.m$ clicando o mouse sobre a figura.

Gráficos especializados

- $fill(x, y,' c')$: Constrói e preenche um polígono (fechado) bidimensional definido pelos vetores coluna x e y ($(x(i), y(i))$ é o i-ésimo vértice do polígono), com a cor definida por $'c'$.

- Exemplo: $mm2512.m$

  - Este exemplo usa a função $text$ com argumentos extras que pedem ao **MATLAB** que use *Handle Graphics* para modificar o texto.

  - *Handle Graphics* é um conjunto de funções destinadas a ajustes finos em gráficos, objeto do capítulo 30 do livro.

Gráficos especializados

- $pie(a, b)$: Constrói um círculo, subdividindo-o em regiões (fatias) de acordo com os valores contidos em a.
  - O parâmetro b é opcional. É um vetor lógico que determina quais fatias serão destacadas do todo.
  - Os valores de a são normalizados via $a/sum(a)$.
  - Se $sum(a) < 1.0$, aparecerá apenas uma parte do círculo, que contém a área usada.
  - É possível determinar rótulos para cada fatia. Maiores detalhes veja a ajuda *on line*.
- Exemplo: $mm2513.m$.

Gráficos especializados

- $plotyy$: função usada para construir gráficos simultâneos com escalas diferentes no eixo y. Cria dois eixos y, um à esquerda (padrão), outro à direita, cada um com sua própria escala. Exemplo: $mm2514.m$.

- As funções a seguir constroem gráficos de barras e escada. Exemplo: $mm2515.m$.

  - bar e $bar3$: Constroem gráficos de barras verticais. A função bar exibe em visão bidimensional e a função $bar3$ em visão tridimensional.

  - $barh$ e $bar3h$: Análogas às anteriores, mas com barras horizontais.

  - $stairs$: Gráfico escada.

    - Maiores detalhes sobre opções para estas funções veja ajuda *on line*.

Gráficos especializados

- $hist(x, y)$: constrói um histograma usando os dados do vetor x. Se o argumento y:
  - for omitido, o histograma é construído com 10 subdivisões;
  - for um escalar o histograma é criado com y subdivisões;
  - for um vetor, o histograma é criado utilizando as subdivisões especificadas no vetor y.
- Exemplo: $mm2516.m$, histograma da distribuição normal.

Gráficos especializados

- $stem(z)$: constrói um gráfico com os dados do vetor z conectados ao eixo horizonal através de uma linha. Isto é, representa sequências discretas.
  - O tipo de linha pode ser especificado como um argumento opcional. Exemplo: $mm2517.m$.
  - O argumento opcional *'filled'* determina se o marcador será vazio ou preenchido.
  - A versão $stem(x, z)$ imprime os dados de z nos valores especificados pelo vetor x. Considere o exemplo:

```
% Usando z defi nido em mm2517.m
>> fi gure % nova janela
>> x = logspace(0,2,30);  stem(x,z)
>> set(gca,'YGrid','on')
```

Gráficos especializados

- $errorbar(x, y, e)$: cria um gráfico do vetor x *versus* o vetor y, com barras especificadas pelo vetor e. Todos os vetores devem ter as mesmas dimensões.
  - Para cada ponto $(x(i), y(i))$ uma barra de erro é desenhada a uma distância $e(i)$ acima e abaixo do ponto.
  - Exemplo: $mm2518.m$.

- $scatter(x, y, s, c)$: constrói um gráfico de dispersão. É construído um gráfico de círculos, com tamanhos e cores específicas.
  - O centro dos círculos é determinados pelos vetores x e y, a área pelo vetor s (se omitido é usado o tamanho padrão), e as cores pelo vetor c (se omitido é usada a mesma cor para todos os círculos).
  - Exemplo: $mm2520.m$

Gráficos especializados

- $polar(t, r, S)$: Constrói o gráfico em coordenadas polares, onde t é o ângulo em radianos, r é o vetor que contém o raio e s, que é opcional, é usada para selecionar cor, marcadores e estilo de linha.
  - Exemplo: $mm2519.m$, $subplot(2, 2, 1)$

- $rose(v, x)$: histograma em ângulos. Desenha o histograma usando os ângulos do vetor v, que deverá estar em radianos. Se o argumento x:
  - for omitido, o gráfico é desenhado com 20 subdivisões, igualmente espaçadas, de 0 a 2π;
  - for um escalar, o gráfico é desenhado com x subdivisões, igualmente espaçadas de o a 2π; e
  - for um vetor, desenha o histograma usando os intervalos especificados em X.
  - Exemplo: $mm2519.m$, $subplot(2, 2, 4)$.

Gráficos especializados

- Dados complexos podem ser representados usando $compass$ and $feather$.
  - $compass(z)$: constrói um gráfico que representa o ângulo e o módulo dos dos elementos de z. Cada elemento é uma seta representada a partir da origem.

$$\boxed{compass(z) \equiv compass(real(z), imag(z))}$$

  - $feather(z)$: análogo à $compass(z)$, mas as setas partem de retas equidistantes em uma linha horizontal.

- Exemplo: $mm2519.m$, $subplot(2,2,2)$ e $subplot(2,2,3)$.

Ganhando tempo

- Para evitar especificar todos os pontos dos dados explicitamente, pode-se recorrer às funções abaixo. Estas poupam o usuário de definir os dados para a variável independente explicitamente.
  - $fplot$: Cria gráficos a partir de funções definidas em M-files ou *function handles*.
  - $ezplot$: Cria gráficos utilizando funções definidas em expressões em strings, ou objetos matemáticos simbólicos. Usa coordenadas cartesianas.
  - $ezpolar$: Análoga à anterior, mas usa coordenadas polares ao invés de cartesianas.
- Exemplo: $mm2521.m$ ($fplot$), $mm2522.m$ e $mm2523.m$ ($ezplot$). O último exemplo mostra que $ezplot$ pode ser usada para fazer gráficos de funções implícitas. A função é uma elipse centrada em $(2, -1)$.

Formatação de texto

- É possível usar textos de várias linhas em qualquer string, inclusive títulos e legendas, além das funções $text$ e $gtext$. Para isso, utiliza-se vetores de strings, ou vetores de células.
- Exemplo: *xlabel('Primeira linha', 'Segunda linha');* insere duas linhas como legenda para o eixo x. O separador de strings pode ser tanto espaço, vírgulas, ou ponto e vírgula. Todos possuem o mesmo efeito.
- O **MATLAB** disponibiliza um conjunto de símbolos, letras gregas e caracteres especiais para serem usados nas strings. Isto inclui um conjunto, limitado, de comandos TEX. A seção 25.12 exibe uma tabela contendo estes símbolos.
  - As mesmas informações podem ser encontradas pesquisando-se a propriedade $string$ do objeto *handle graphic* $text$ na ajuda *on line*.
- Exemplo: $mm2524.m$ ilustra o uso de comandos TEX.