Vytváranie aplikácií v Matlabe

1. M-súbory

M-súbory slúžia na ukladanie postupností príkazov (skripty) alebo na ukladanie užívateľských funkcií (funkcie). Tieto nové súbory majú príponu *.m*.

Poznámka: m-súbory sú obyčajné textové súbory (ASCII súbory), a preto je možné ich písať aj v ľubovoľnom textovom editore. Z dôvodov vyššieho komfortu (zvýraznenie syntaxe, možnosť krokovania) je súčasťou MATLABu tiež M-editor/Debugger, ktorý sa otvára v samostatnom okne po otvorení alebo vytvorení m-súboru (skriptu či funkcie).

1.1. M-editor

Súčasťou MATLABu je aj vlastný editor, ktorý zároveň slúži ako debugger. Ide o jednoduchý editor, ktorý má niektoré užitočné vlastnosti:

- zvýrazňuje kľúčové slová MATLABu
- pri dlhšom umiestnení kurzora myši nad premennou sa zobrazí okno s jej hodnotou
- označuje čísla riadkov atď.

M-editor/Debugger je spustený v samostatnom okne po otvorení (*File » Open*) alebo vytvorení (*File » New » M-file*) nového m-súboru. Slúži k pohodlnej editácii m-súboru. Navyše umožňuje krokovať obsah m-súboru (t.j. kontrolovať vykonávanie jeho jednotlivých príkazov).

2. Skripty

Skript (z angl. script) je postupnosť príkazov uložených do súboru. Každý skript pracuje s premennými pracovného prostredia, takže môže vytvárať nové premenné alebo mazať či meniť vybrané. Výsledky skriptu teda zostávajú v pracovnom prostredí aj po jeho skončení. Skripty samozrejme môžu volať iné skripty alebo funkcie, vytvárať grafické okná, vypisovať do Command Window, ...

2.1. Vytvorenie skriptu

1. Najskôr musí byť v MATLABe nastavený pracovný adresár (napr. príkazom *cd*).
2. **Vytvorenie nového alebo otvorenie existujúceho skriptu**
   M-súbor, ktorý bude obsahovať skript, vytvoríme napríklad pomocou menu *File » New » M-file*, čím sa tiež otvorí okno M-editora/Debuggera. Pokiaľ je potrebné opraviť už existujúci skript, musíme ho otvoriť (napríklad pomocou menu *File » Open*).
3. **Zápis skriptu**
   Do prázdneho m-súboru sa zapisujú všetky príkazy, ktoré má skript vykonať -

príkazy sa píšu rovnako ako v Command Window, iba s tým rozdielom, že sa po napísaní nevykonávajú. Takto sa vytvára kód skriptu (postupnosť príkazov).

4. **Uloženie skriptu**
   Ak je napísaný kód skriptu, musí sa celý m-súbor uložiť pod nejakým menom na disk (do pracovného adresára). Uloženie sa vykonáva pomocou menu *File » Save*. Meno skriptu musí spĺňať rovnaké pravidlá ako názov premennej. Je to z toho dôvodu, aby MATLAB mohol skript spustiť. Meno m-súboru so skriptom teda môže obsahovať iba písmená anglickej abecedy, podtrhovník a čísla (číslom nesmie začínať).

2.2. Spustenie skriptu

Spustenie skriptu dáva MATLABu pokyn k vykonaniu jeho príkazov. Pred spustením musí byť skript uložený! Skript môže byť spustený buď

- v Command Window - stačí zadať meno m-súboru bez prípony, alebo
- v M-editore/Debuggeri pomocou menu *Debug » Run* (klávesa F5).

Príklad 4.1: Skript parabola.m, ktorý kreslí graf funkcie x^2

```
% GRAF - vykreslenie paraboly
x = -3:0.1:3; % vektor hodnôt z intervalu [-3;3] s krokom 0,1
y = x.^2; % závislá premenná
plot(x,y) % graf (parabola)
```

Príklad 4.2: Spustenie skriptu parabola.m

```
>> parabola
```

Pokiaľ je všetko v poriadku, objaví sa okno s grafom. V opačnom prípade musí byť nájdená a opravená chyba, uložený súbor a jeho opätovné spustenie.

3. Užívateľské funkcie

Ak sa nejaká postupnosť príkazov (algoritmus) opakuje vo viacerých situáciách (napr. pre rôzne hodnoty premenných), nie je praktické používať skripty, pretože vždy sa musia upraviť hodnoty premenných v skripte, uložiť príslušný m-súbor a skript spustiť. Riešenie ponúkajú funkcie.

Funkcie sú m-súbory, ktoré majú presne definovanú štruktúru (viď. Vytvorenie funkcie). Funkcie akceptujú vstupné parametre, ktoré môžu mať pri každom spustení inú hodnotu.

Každá funkcia má svoje vlastné pracovné prostredie, ktoré je oddelené od pracovného prostredia Command Window. Všetky premenné vo funkcii sú lokálne (existujú iba vnútri funkcie). To znamená, že:

- sa nemôžu použiť žiadne iné premenné než tie, ktoré sú funkcie predávané (pomocou vstupných parametrov) alebo tie, ktoré si funkcia sama vytvorí,

- všetky premenné (aj tie, ktoré obsahujú vypočítané výsledky) po skončení funkcie zaniknú.

Našťastie existuje spôsob, ktorým funkcia môže svoje výsledky predať "von": **výstupné premenné**. Počet vstupných aj výstupných parametrov funkcie sa určuje pri jej vytváraní. Pokiaľ funkcia nemá žiadne vstupné parametre, môžu sa jej príkazy napísať tiež ako skript.

Príklad 4.3: Súbor (funkcia) priemer.m

```
function y = priemer(x)
% PRIEMER stredna hodnota vektora.
% PRIEMER(X), kde X je vektor vrati strednu hodnotu elementov vektora
% Ak vstup nie je vektor, vrati chybu.
[m,n] = size(x);
if (~((m == 1) | (n == 1)) | (m == 1 & n == 1))
error('Vstup musi byt vektor')
end
y = sum(x)/length(x); % Vypočet priemeru
```

3.1. Vytvorenie funkcie

1. Najskôr musí byť v MATLABe nastavený pracovný adresár (napr. príkazom *cd*).
2. **Otvorenie m-súboru**
   Funkcia je m-súbor, a preto sa najskôr musí otvoriť nový súbor: napríklad pomocou menu *File » New » M-file*. Tým sa otvorí okno M-editora/Debuggeru s prázdnym súborom.
   Poznámka: pokiaľ chceme nejakú existujúcu funkciu opraviť, použijeme menu *File » Open*.
3. **Zápis funkcie** (štruktúra m-súboru obsahujúceho funkciu)
   Prvý riadok obsahuje definíciu funkcie, po ňom môžu nasledovať riadky s nápovedou k funkcii (komentáre) a zvyšok súboru tvoria príkazy (kód funkcie, algoritmus) potrebné na výpočet výstupu funkcie za použitia jej vstupov.
   - **Definícia funkcie**
     tvorí prvý riadok m-súboru. Má tvar:

     function [výstupy]= názov_funkcie (vstupy)

     kľúčové slovo výstupy funkcie názov funkcie vstupné parametre funkcie

   - **výstupy:**
     - ak ich je viac, oddeľujú sa čiarkou
     - ak je len jeden, zátvorky nie sú nutné
     - funkcie nemusia mať žiadny výstup

     názov_funkcie:

     - názov funkcie by mal vystihovať jej činnosť

- musí splniť pravidlá pre názov premennej, inak sa funkciu nepodarí spustiť!
- názov funkcie sa nesmie zhodovať so žiadnym názvom jej premennej!

vstupy:

- ak je ich viac, oddeľujú sa čiarkou
- funkcia nemusí mať žiadny vstup (potom sa podobá skriptu, ale má svoje lokálne workspace!)

Príklad 4.4: Príklady definovania funkcií

```
function [s]=sucet(a,b)
% funkcia s jedným výstupom a dvoma vstupmi

function [podiel,zvysok] = delenie(delenec,delitel)
% funkcia s dvoma vstupmi a výstupmi

function f = faktorial(n)
% funkcia s jedným výstupom a vstupom
```

1.
   - ```
     function graf(x,y)
     % funkcia bez výstupu a dvoma vstupmi
     ```
   -
   - **Nápoveda k funkcii**
     nie je povinnou súčasťou funkcie, ale mala by byť vytvorená, pretože uľahčuje používanie danej funkcie. Nápoveda k funkcii začína druhým riadkom, pokiaľ je tento riadok komentárom. Nápoveda k funkcii končí akýmkoľvek riadkom, ktorý už nie je komentárom (t.j. aj treba prázdnym riadkom).
     Pokiaľ je v m-súbore funkcie uvedená nápoveda, zobrazíme ju príkazom *help názov_funkcie*.

     Prvý riadok nápovedy by mal obsahovať názov funkcie a vystihovať jej činnosť, pretože je vypisovaný príkazom *lookfor slovo* (slúži na výpis všetkých funkcií obsahujúcich dané *slovo*) alebo pri výpise nápovedy k funkciám adresára *help adresár* (napríklad *help C:\temp\matlab*).
     Ďalšie riadky nápovedy by mali obsahovať opis vstupu a výstupu funkcie a tiež príklad jej použitia.

   - **Kód funkcie**
     (algoritmus; príkazy) začína hneď za nápovedou a obsahuje postupnosť príkazov, pomocou nich funkcia vypočíta svoje výstupy. Všetky výstupy funkcie musia byť jej kódom vytvorené, inak je po spustení funkcie ohlásená chyba! Všetky priraďovacie príkazy vnútri funkcie by mali byť
     ```

ukončené bodkočiarkou (aby funkcia neobťažovala okolie výpisom pomocných premenných).

Poznámka: na výpis chyby a ukončenie funkcie možno použiť funkciu *error*, jej parametrom je text chybového hlásenia (napr. *error ('Chyba: veľa vstupných parametrov!')*). Okrem vytvorenia nápovedy k funkcii je vhodné používať tiež komentáre v kóde funkcie (pri jednotlivých príkazoch).

Príklad 4.5: Funkcia, ktorá vypočíta súčet dvoch čísiel

```
function [s] = sucet(a,b)
% SUCET - sucet dvoch cisiel
% s=sucet(a,b)
% a,b ... scitance
% s ... vysledok (sucet)
% priklad volania: s=sucet(10,-2.5)
s = a+b;
```

- nápoveda k funkcii obsahuje všetko potrebné pre správne použitie funkcie
- kód funkcie zaberie síce iba jediný riadok, ale to k vypočítaniu výstupu *s* stačí
- priraďovací príkaz vnútri funkcie je ukončený bodkočiarkou, aby funkcia nevypisovala pomocné výpočty

Príklad 4.6: Funkcia, ktorá vypočíta dĺžku prepony pravouhlého trojuholníka

```
function [prep] = prepona(odvesna1,odvesna2)
% PREPONA - vypočet prepony pravouhleho trojuholnika
prep = (odvesna1^2 + odvesna2^2)^(1/2); % pouzitie
Pythagorovej vety
```

2. **Uloženie funkcie**
   Napísaná funkcia sa musí uložiť na disk ako m-súbor (napr. pomocou menu *File » Save*). Pri ukladaní je potrebné:
   - skontrolovať pracovný adresár
   - ako názov súboru sa zadáva názov funkcie, pretože názov m-súboru sa musí zhodovať s názvom danej funkcie, inak by funkcia nešla spustiť!
   - po uložení m-súboru zmizne hviezdička za jej názvom v titulkovom pruhu okna M-editora/Debuggeru.

3.2. Spustenie funkcie

Funkcie (aj vlastné) môžeme spustiť z Command Window (alebo z iných funkcií či skriptov).
Príkaz na spustenie funkcie vyzerá vo všeobecnosti takto:

```
>> [vystupy] = nazov_funkcie(vstupy)
% pokiaľ je potrebné uložiť výsledky do premenných

>> nazov_funkcie(vstupy)
% pokiaľ sa výsledky neukladajú do premenných
```

Poznámky:

- počet vstupných argumentov sa musí zhodovať s jej definíciou (výnimkou sú funkcie obsahujúce *nargin*)
- poradie vstupných argumentov pri volaní je zaväzujúce - sú spracovávané v poradí danom definíciou funkcie
- výstupy z funkcie nie je povinné priraďovať do premenných (pokiaľ nie je napísaná za volaním funkcie bodkočiarka, prvý z výstupov sa vypíše pomocou *ans*)
- pokiaľ sú výstupy funkcie priraďované do premenných, mal by byť dodržaný ich počet (funkcie obsahujúce *nargout* môžu vracať rôzne výsledky v závislosti na počte práve priraďovaných výstupov)
- ak nie je známy počet vstupov ani výstupov, potom sa doporučuje použiť nápovedu k funkcii
- pokiaľ je po spustení funkcie hlásená chyba, potom je doporučený nasledujúci postup:
    - je správne zadaný názov funkcie?
    - súhlasí počet vstupných parametrov?
    - je funkcia uložená? (pokiaľ sme ju upravovali)
    - nachádza sa funkcia v pracovnom adresári?
- pokiaľ nie je po spustení funkcie hlásená chyba, ale funkcia nevracia správne výsledky, je potrebné funkciu krokovať

Príklad 4.7:

```
>> s = sucet(7,14);

>> vysledok = sucet(7,14)
vysledok =
     21

>> x=7; y=14; sucet(x,y)
ans =
     21
```

Grafika v Matlabe

MATLAB umožňuje zobrazovať údaje v 2 a 3-rozmerných grafoch.

1. Grafické prostredie v MATLABe

Grafické funkcie automaticky otvoria nové grafické okno, pokiaľ už predtým nejaké neexistuje. V prípade, že je už nejaké grafické okno otvorené, je použité na vykresľovanie to, ktoré "je aktívne" (jedná sa o posledné okno, ktoré bolo otvorené alebo na ktoré bolo kliknuté).

Funkcia	Opis
clf	zmazanie aktuálneho obrazca
close	uzatvorenie okna grafického obrázku
figure	otvorenie okna grafického obrázku
gcf	číslo aktuálneho okna grafického obrázku

Príkaz *figure* poskytuje nasledovné možnosti zápisu:

- pre vytvorenie nového okna: *figure*
- pre vytvorenie okna s číslom *h*: *figure(h)*
- pre vytvorenie okna s vrátením jeho čísla do premennej *k*: *k=figure*

Príklad 11.1:

```
% otvorenie okna grafického obrázku č.3
>> figure(3)                          % obrázok

% otvorenie okna grafického obrázku č.2
>> figure(2)                          % obrázok

>> gcf          % číslo aktuálneho okna grafického obrázku
ans =
     2

>> close        % uzatvorenie aktívneho okna grafického obrázku
>> close(3)     % uzatvorenie okna grafického obrázku č.3
>> close all    % uzatvorenie všetkých okien grafických obrázkov
```

top

2. Vytváranie dvojrozmerných grafov

2.1. Základné grafické funkcie zobrazenia

Funkcia	Opis
fill	vyplnenie dvojrozmerného mnohouholníka
loglog	zobrazenie, ak osi majú logaritmické mierky

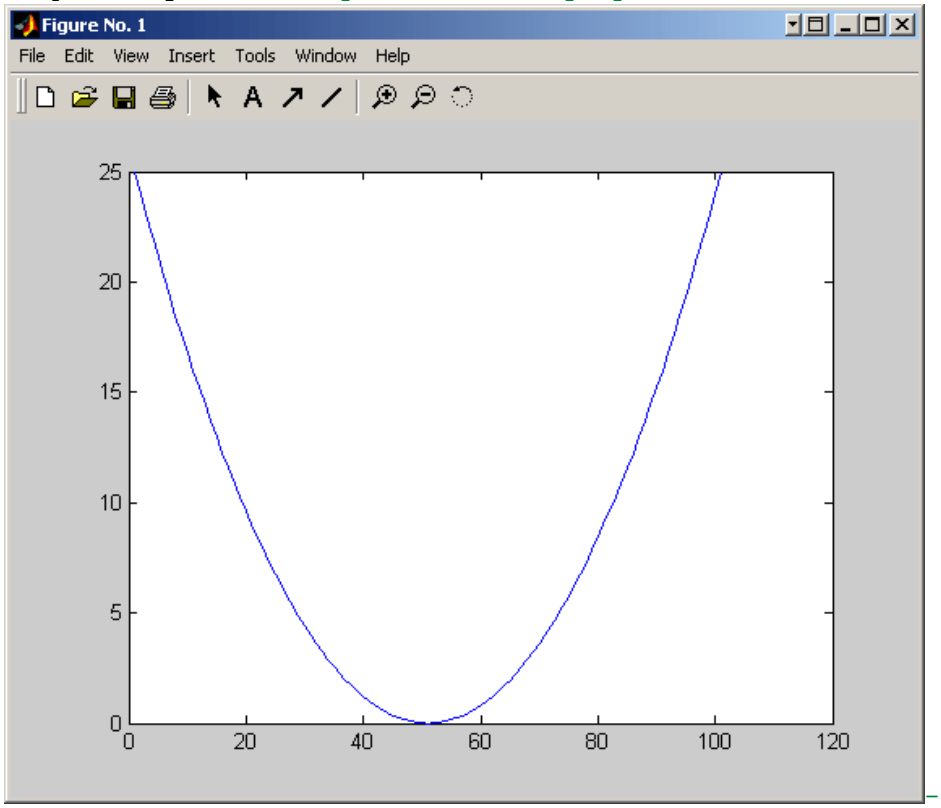
Funkcia	Opis
plot	lineárne zobrazenie
plotyy	lineárne zobrazenie s mierkami na ľavej a pravej strane
semilogx	zobrazenie, ak os x je v logaritmickej mierke
semilogy	zobrazenie, ak os y je v logaritmickej mierke

2.2. Lineárne zobrazenie

Lineárne zobrazenie bodov, priebehov veličín je možné uskutočniť funkciou *plot(x1,y1,x2,y2, ..., xn,yn)*, kde *x1, y1, x2, y2, ..., xn, yn* sú vektory s hodnotami súradníc bodov.

Príklad 11.2: Lineárne zobrazenie paraboly

```
>> x = [-5:0.1:5];
>> y = x.^2;
>> plot(y)          % vykreslí hodnoty 'y' v závislosti na ich poradí -
>> plot(x,y)        % vykreslí hodnoty 'y' v závislosti na hodnotách 'x'
```



Farba, typ a znaky čiary

Pri zobrazovaní pomocou funkcie plot je možné zadefinovať farbu, typ a znaky vykreslovanej čiary nasledovným spôsobom: *plot(x,y,'farba_typ_znak')*, kde *farba_typ_znak* je reťazec zložený zo symbolov vyjadrujúcich farbu, typ a znak čiary.

Symbol	Farba (RGB)	Symbol	Typ čiary
c	cyan (0 1 1)	**-**	plná čiara (auto)
m	magenta (1 0 1)	**--**	čiarkovaná
y	žltá (1 1 0)	**-.**	bodkočiarkovaná
r	červená (1 0 0)	**:**	bodková
g	zelená (0 1 0)	**none**	žiadna
b	modrá (0 0 1)		
w	biela (1 1 1)		
k	čierna (0 0 0)		

Znak	Opis znaku
+	plus
o	kruh
*	hviezdička
.	bodka
x	krížik
square	Štvorec
diamond	kosoštvorec
V	dole orientovaný trojuholník
<	vľavo orientovaný trojuholník
>	Vpravo orientovaný trojuholník
pentagram	päťcípa hviezda
hexagram	šesťcípa hviezda
none	žiadny (auto)

Príklad 11.3: Farba, typ a znaky čiary

```
>> x = [-5:5];
>> y = x.^2;
>> plot(x,y,'r-*',x,y/2,'m-.')          % obrázok
```

2.3. Zobrazenie v logaritmických súradniciach

Pri funkciách *semilogx*, *semilogy* a *loglog* ide o podobné zobrazenie ako v prípade lineárneho zobrazenia, len niektorá z osí alebo obe sú v logaritmickej mierke.

Príklad 11.4: Definovanie premennych pre rôzne zobrazenia

```
>> x = 0:0.01:4;
>> y = sin(0:0.01:4);
>> z = x.^2;
```

Príklad 11.5: Zobrazenie, ak osi majú logaritmické mierky

```
>> figure
>> loglog(x,y)                    % obrázok
```

2.4. Vyplnenie dvojrozmerného mnohouholníka

Funkciou *fill(x,y,c)* je možné zobraziť farebný mnohouholník, kde *x,y* sú vektory súradníc bodov mnohouholníka a *c* je premenná, ktorá definuje farbu.

Príklad 11.6: Vyplnenie trojuholníka červenou farbou

```
>> figure
>> fill([0,1,0.5],[0,0,1],'r')    % obrázok
```

2.5. Ďalšie zobrazenia

Príklad 11.7: Lineárne zobrazenie s mierkami na ľavej a pravej strane

```
>> figure
>> plotyy(x,y,x,z)                % obrázok
```

Príklad 11.8: Bodeho diagram - definovanie premenných

```
>> [amp,faza,w] = bode([1],[1 2 1])
amp =
    0.9999
    0.9996
       ...
    0.0004
    0.0001

faza =
   -1.1459
   -2.2915
    ...
 -177.7085
 -178.8541

w =
    0.0100
    0.0200
       ...
   50.0000
  100.0000

>> amp_db = 20*log10(amp)
amp_db =
   -0.0009
   -0.0035
    ...
  -67.9623
  -80.0009
```

Príklad 11.9: Lineárne zobrazenie Bodeho diagramu

```
>> plot(w,amp_db)            % obrázok
```

Príklad 11.10: Zobrazenie, ak os x je v logaritmickej mierke

```
>> semilogx(w,amp_db)        % obrázok
```

top

3. Základné riadiace funkcie pre zobrazenie

Funkcia	Opis
axes	vytvorenie osí na ľubovoľnú pozíciu
axis	nastavenie parametrov osí a vzhľadu
caxis	nastavenie rozsahu farieb osí
cla	zmazanie osí
box	zobrazenie rámika osí
gca	vracia číslo aktuálnych osí
get	zistenie nastavenia grafického objektu
hold	zachovanie aktuálneho grafu
set	nastavenie vlastností grafického objektu
subplot	rozdelenie grafického okna na subokná

3.1. Nastavenie osí

Funkciou *axes* je možné vytvoriť osi a umiestniť ich na ľubovoľné miesto v grafickom okne

Príklad 11.11:

```
% nastavenie parametrov osí a vzhľadu
% axes('position',[left, bottom, width, height]),
% kde left, bottom sú body poč. súradnicového systému
% a width, height sú veľkosti x-ovej a y-ovej osi (rozmery 1x1)
>> axes('position',[0.2, 0.2, 0.7, 0.7])      % obrázok
```

Príklad 11.12:

```
% nastavenie rozmerov osí x, y
>> axis([0,100,0,100])                % obrázok

% nastavenie rozmerov osí x, y a z
>> axis([0,10,0,10,1,5])                      % obrázok

% funkcia vracia vektor nastavenia osí
>> axis
ans =
     0    10     0    10     1     5

% rozmery osí sa nastavia automaticky
>> axis auto                          % obrázok

% vypnutie zobrazenia osí
>> plot([0:0.01:pi],sin([0:0.01:pi]))
>> axis off                           % obrázok

% zapnutie zobrazenia osí
>> axis on
```

3.2. Zobrazenie viacerých grafov

Ak je požadované pridať do jedného grafu viac priebehov alebo v ňom vykonať ďalšie grafické operácie, potom je potrebné použiť príkaz *hold on*.

Príklad 11.13: Zachovanie aktuálneho grafu

```
>> plot(x,y,'r')                                % obrázok
>> hold on
>> plot(x,z,'b')                                % obrázok
```

Príklad 11.14: Zrušenie zachovania aktuálneho grafu

```
>> hold off
```

Príkaz *subplot(m,n,p)* rozdelí graf na subokná, kde *m,n,p* vyjadrujú počet riadkov, stĺpcov a poradie okna

Príklad 11.15: Ukážka rozdelenia grafického okna na subokná - 3 riadky a 4 stĺpce

```
>> subplot(3,4,7)                               % obrázok
```

1 2 3 4

5 6 7 8

9 10 11 12

Príklad 11.16: Rozdelenie grafického okna na subokná

```
>> subplot(1,2,1)
>> plot(x,y,'r')
>> subplot(1,2,2)
>> plot(x,z,'b')                                % obrázok
```

3.3. Nastavenie parametrov grafických objektov

Pri vytvorení grafického objektu mu Matlab pridelí identifikačné číslo. Pomocou tohto čísla je možné potom pristupovať k jednotlivým objektom. Číslo objektu vracia vždy funkcia, ktorá ho vytvorila.

Nastavenie objektu je možné získať príkazom *get(h)*, kde *h* je čislo objektu. Nastaviť určitú položku vlastnosti je možné príkazom *set(h,'položka',hodnota)*.

Príklad 11.17: Získanie čísla objektu

```
>> h = plot(w,amp_db);
```

Príklad 11.18: Výpis položiek vlastností

```
>> get(h)
        Color = [0 0 1]
        EraseMode = normal
        LineStyle = -
        LineWidth = [0.5]
        Marker = none
```

```
        MarkerSize = [6]
        MarkerEdgeColor = auto
        MarkerFaceColor = none
        XData = [ (1 by 45) double array]
        YData = [ (1 by 45) double array]
        ZData = []

        BeingDeleted = off
        ButtonDownFcn =
        Children = []
        Clipping = on
        CreateFcn =
        DeleteFcn =
        BusyAction = queue
        HandleVisibility = on
        HitTest = on
        Interruptible = on
        Parent = [102.017]
        Selected = off
        SelectionHighlight = on
        Tag =
        Type = line
        UIContextMenu = []
        UserData = []
        Visible = on
```

Príklad 11.19: Zmena typu čiary

```
% pôvodné: LineStyle = -
% zmeniť na: LineStyle = x
>> set(h,'LineStyle','x')                          % obrázok
```

Príklad 11.20: Zmena farby čiary

```
% pôvodné: Color = [0 0 1] (R,G,B)
% zmeniť na: Color = [0 1 0] (Color = green alebo Color = g)
>> set(h,'Color','green')                          % obrázok
```

top

4. Označenie a popis grafov

Funkcia	Opis
grid	čiarová sieť
gtext	text umiestnený myšou
legend	vytvorenie legendy priebehov
text	umiestnenie textu
title	názov grafu
xlabel	popis osi x
ylabel	popis osi y
zlabel	popis osi z

Príklad 11.21: Vykreslenie a popis grafov

```
>> t = 0:0.01:2*pi;                 % definovanie premennych
>> y = sin(t);

>> plot(t,y)                        % vykreslenie
```

```
>> xlabel('t = 0 \rightarrow 2pi') % popis osi x
>> ylabel('sin(t)')                % popis osi y
>> title('Priebeh funkcie sínus')  % názov grafu
```

Príklad 11.22: Zápis textu pomocou súradníc dát

```
>> text(3*pi/4,sin(3*pi/4),'\leftarrow sin(t) = 0.707')
>> text(7*pi/6,sin(7*pi/6),'sin(t) = -0.5 \rightarrow',...
'HorizontalAlignment','right')                        % obrázok
```

Príklad 11.23: Legenda

```
>> y1 = sin(t); y2 = cos(t); y3 = y1+y2;
>> plot(t,y1,t,y2,t,y3);
>> legend('y_1=sin(t)','y_2=cos(t)','y_3=sin(t)+cos(t)')     % obrázok
```

Príklad 11.24: Mriežka (hlavná, vedlajšia)

```
>> plot(t,y1,t,y2,t,y3);
>> grid on                                                   % obrázok
>> grid minor                                                % obrázok
```

Poznámka: ak chceme zmeniť orientáciu opisu vertikálnej osi (y) z vertikálnej na vodorovnú môžeme použiť príkaz *set(get(gca,'YLabel'),'Rotation',0.0)*.

4.1. Špeciálne znaky (T_EX)

Znak	Zápis	Znak	Zápis	Znak	Zápis
α	\alpha	β	\beta	γ	\gamma
δ	\delta	ε	\epsilon	ω	\omega
λ	\lambda	ξ	\xi	π	\pi
ρ	\rho	σ	\sigma	τ	\tau
σ	\sigma	Δ	\Delta	Σ	\Sigma
∇	\nabla	∂	\partial	∞	\infty
√	\surd	∫	\int	≠	\neq
∈	\in	⊂	\subset	⊆	\subseteq
≤	\leq	≥	\geq	↑	\uparrow
∧	\wedge	∨	\vee	↓	\downarrow
↔	\leftrightarrow	←	\leftarrow	→	\rightarrow
¬	\neg	∀	\forall	∃	\exists

horný index ^{horný index} dolný index _{dolný index}

5. Špeciálne grafy

Špeciálne grafy sú využívané najmä v oblastiach:

- štatistické grafy
- zobrazenie diskrétnych dát
- priame a rýchlostné vektorové grafy
- hladinové zobrazenie

5.1. Štatistické grafy

Funkcia	Opis
bar	dvojrozmerný zvislý stĺpcový graf
bar3	trojrozmerný zvislý stĺpcový graf
barh	dvojrozmerný vodorovný stĺpcový graf
bar3h	trojrozmerný vodorovný stĺpcový graf
errorbar	zobrazenie s odchýlkou
hist	zobrazenie histogramu
rose	zobrazenie uhlového histogramu
polar	zobrazenie v polárnych súradniciach
area	plošný graf
pie	kruhový graf

Príklad 11.25:

```
>> Y = [6 3 2;9 6 4;7 5 4;5 6 5;4 3 2];
>> bar3(Y)
>> xlabel('Os x'), ylabel('Os y'),
>> zlabel('Os z')                                  % obrázok
```

Príklad 11.26: Zobrazenie v polárnych súradniciach

```
>> t = 0:0.01:2*pi;
>> polar(t,abs(sin(2*t).*cos(2*t)));              % obrázok
```

5.2. Zobrazenie diskrétnych dát

Funkcia	Opis
stem	dvojrozmerné kmeňové zobrazenie
stem3	trojrozmerné kmeňové zobrazenie
stairs	schodové zobrazenie

Príklad 11.27: schodové zobrazenie

```
>> x = 0:0.25:10;
>> stairs(x,sin(x));                % obrázok
```

Príklad 11.28: Stonkové zobrazenie

```
>> x = 0:0.1:4;
>> y = sin(x.^2).*exp(-x);
>> stem(x,y);                       % obrázok
```

5.3. Priame a rýchlostné vektorové grafy

Funkcia	Opis
compass	zobrazenie vektorov v polárnych súradniciach
feather	zobrazenie vektorov pozdĺž vodorovnej čiary
quiver	zobrazenie 2-D vektorov definovaných (u,v) parametrami
quiver3	zobrazenie 3-D vektorov definovaných (u,v,w) parametrami

Príklad 11.29: Zobrazenie v polárnych súradniciach, ktoré sú zadané uhlom a veľkosťou

```
>> uhol = [45 0 90 150 230 300];
>> velkost = [5 7 10 6 12 8];
>> ruhol = uhol*pi/180;                % prepočet na radiány
>> [x,y] = pol2cart(ruhol,velkost);    % prepočet do kart. súradníc
>> compass(x,y);                       % obrázok
```

Príklad 11.30: Zobrazenie vektorov pozdĺž x-ovej osi, ktoré sú zadané uhlom od 90º do 0º a veľkosťou 5

```
>> uhol = 90:-10:0;
>> velkost = 5*ones(size(uhol));
>> [u,v] = pol2cart(uhol*pi/180,velkost);
>> feather(u,v);                       % obrázok
```

Príklad 11.31: Zobrazenie vektorov intenzity el. magnetického poľa dvoch nábojov pomocou zobrazenia zmeny súradníc

```
>> x = -2:.2:2;
>> y = -1.5:.2:1.5;
>> [xx,yy] = meshgrid(x,y);
>> zz = xx.*exp(-xx.^2-yy.^2);
>> [px,py] = gradient(zz,.2,.2);
>> quiver(x,y,px,py,2);                % obrázok
```

5.4. Hladinové zobrazenie - rezy a pohľady

Funkcia	Opis
clabel	popis hladinového zobrazenia
contour	plošné hladinové obrazenie
contour3	trojrozmené hladinové obrazenie
contourf	plošné hladinové obrazenie s farebnou výplňou
pcolor	pseudofarebné zobrazenie

Príklad 11.32: Príkladová funkcia 'peaks'

```
>> z = peaks(25);
>> pcolor(z)
>> colormap(hsv)                       % obrázok
```

Príklad 11.33: Hladinové zobrazenie obrazca s popisom jednotlivých hladín

```
>> [C,h] = contour(z,10);
>> clabel(C,h);                 % obrázok
```

Príklad 11.34: Hladinové zobrazenie obrazca s farebným rozlíšením

```
>> [C,h] = contourf(z,10);
>> caxis([-10 10]);             % obrázok
```

Vytváranie trojrozmerných grafov

6.1. Príprava údajov

Pri zobrazení trojrozmerného obrazca potrebujeme mať dve matice (napr. *X*, *Y*), ktorých prvky *X(i,j)*, *Y(i,j)* budú obsahovať súradnice bodov v rovine. Na generovanie týchto matíc slúži funkcia *meshgrid*:

[X,Y] = meshgrid(v1,v2) - vektor *v1* určuje interval na osi *x* s daným krokom a vektor *v2* určuje interval na osi *y* s daným krokom

[X,Y] = meshgrid(v) - vektor v určuje interval na osi *x* aj na osi *y*

6.2. Základné funkcie 3-D zobrazenia

V 3-D zobrazeniach sa používajú podobné funkcie *fill3* a *plot3* ako v 2-D grafike

Funkcia	Opis
fill3	Vyplnenie trojrozmerného mnohouholníka
plot3	zobrazenie čiar a bodov 3-D priestore (lineárne zobrazenie)

Príklad 11.35: Zobrazenie špirály (lineárne zobrazenie)

```
>> t = 0:pi/50:10*pi;
>> plot3(sin(t),cos(t),t);                    % obrázok
```

Príklad 11.36: Zobrazenie trojrozmerného obrazca

```
>> [x,y] = meshgrid([-2:0.1:2]);
>> z = x.*exp(-x.^2-y.^2);
>> plot3(x,y,z);                              % obrázok
```

Príklad 11.37: Zobrazenie kocky s rezom

```
>> x = [0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0];
>> y = [0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1];
>> z = [0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 1 0 1 0 1 0];
>> axis([0 1 0 1 0 1]);
>> hold on
>> for i=1:2:23
plot3([x(i) x(i+1)],[y(i) y(i+1)],[z(i) z(i+1)],'r-');
end
>> fill3([0 0.5 0],[0 0 0.5],[0.5 1 1],'g');  % obrázok
```

6.3. Zobrazenie povrchu obrazca maticovou reprezentáciou

Funkcia	Opis
mesh	trojrozmerné sieťové zobrazenie povrchu
meshc	kombinácia zobrazení mesh/contour
meshz	zobrazenie mesh s nulovou rovinou
slice	objemové predstavenie zobrazenia
surf	zobrazenie povrchu zatienením
surfc	kombinácia zobrazení surf/contour
surfl	zobrazenie surf s vysvietením
surface	vytvorenie povrchu grafického obrazca
waterfall	vlnové zobrazenie (priestorové)

6.4. Možnosti nastavenia zobrazenia

Funkcia	Opis
axis	nastavenie typu osí
caxis	nastavenie rozsahu farieb z aktuálnej farebnej palety
colormap	nastavenie farebnej palety
hidden	skrytie siete 3-D zobrazenia
shading	nastavenie módu tieňovania
brighten	vyjasňovanie alebo stmavovanie farebnej palety
lighting	nastavenie módu osvetlenia
rotate3d	zapnutie/vypnutie možnosti natáčania obrazca
view	špecifikácia bodov zobrazenia v 3-D grafe
viewmtx	transformácia matíc pre zobrazenie

Príklad 11.38: Zobrazenie povrchu obrazca tieňovaním

```
>> z = peaks(25);              % vytvorí maticu 25x25
>> surf(z);            % zobrazenie povrchu zatienením - obrázok
>> shading flat;               % tieňovanie 'flat' - obrázok
>> shading interp;             % tieňovanie 'interpolated' - obrázok
>> shading faceted;    % tieňovanie 'faceted' (pôvodné) - obrázok
>> colormap(copper)    % zmena farby (pink) - obrázok
>> surfc(z);           % kombinácia zobrazení surf/contour - obrázok
```

Príklad 11.39: Zobrazenie povrchu obrazca sieťou

```
>> z = peaks(25);              % vytvorí maticu 25x25
>> mesh(z);            % zobrazenie povrchu sieťou - obrázok
>> colormap(cool);             % zmena farby (cool) - obrázok
>> colormap(hot);              % zmena farby (hot) - obrázok
>> colormap(gray);             % zmena farby (gray) - obrázok
>> meshc(z);           % kombinácia zobrazení mesh/contour - obrázok
>> meshz(z);           % zobrazenie mesh s nulovou rovinou - obrázok
```

Príklad 11.40: Zobrazenie povrchu obrazca vlnami

```
>> z = peaks(25);              % vytvorí maticu 25x25
>> waterfall(z);               % zobrazenie povrchu vlnami - obrázok
>> colormap(pink);             % zmena farby (pink) - obrázok
```

top

7. Práca s farebnými obrazmi

Funkcia	Opis
image	zobrazenie obrazu (vytvorenie objektu obrazu)
imagesc	mierkovanie dát a zobrazenie obrazu
imread	načítanie obrazu z grafického súboru
imwrite	uloženie obrazu do grafického súboru
imfinfo	získanie informácie o obraze z grafického súboru
axis	nastavenie mierky a vzhľadu osí pri zobrazení

7.1. Typy obrazov

Farebný obraz je v MATLABe tvorený maticou dát a maticou palety farieb.

Zápis zobrazenia	Typ obrazu
image(X); colormap(map)	indexový
image(I,[0,1]); colormap(gray)	intenzívny
image(RGB)	trojfarebný

Príklad 11.41: Zobrazenie obrazu z dát uložených v earth.mat

```
>> load earth;  % Načítanie matice dát X,
                  matice farebnej palety map
>> image(X)     % Zobrazenie obrazu
>> colormap(map);
>> axis image;  % Nastavenie osí v pomere 1:1
                % obrázok
```

Príklad 11.42: Vygenerovanie náhodného farebného obrazca

```
>> X = magic(6);       % Vytvorenie matice X,
                  matice farebnej palety map
>> image(X)     % Zobrazenie farebného obrazu
                % obrázok
```

7.2. Načítanie a uloženie grafického súboru

MATLAB umožňuje načítanie a uloženie dát farebného obrazu v niekoľkých grafických súboroch a nasledovných formátov: BMP, HDF, JPEG(JPG), PCX, TIFF, XWD.

Príklad 11.43: Načítanie, zobrazenie a uloženie obrazu

```
[X,MAP] = imread('penguin','bmp');
image(X);               % obrázok
axis image;             % obrázok
imwrite(X,'penguin.jpg')       % Uloženie farebného obrazu
```

top

8. Úlohy

1. Zobrazte priebeh funkcie 2*cos(x) na intervale (0,4*pi) s krokom 0.01. Postupne aplikujte nasledujúce nastavenia:
   - farbu zobrazenia nastavte modrú
   - typ čiary zvoľte čiarkovaný
   - popíšte osi (x » time [s], y » cos) a nadpis zobrazenia (graf funkcie kosínus)
   - pridajte do obrázku priebeh funkcie 3*sin(2x)
   - vytvorte legendu pre jednotlivé priebehy (y_1=2*cos(x), y_2=3*sin(2x))
2. Zobrazte priebehy funkcii y=f(t) a x=f(t), kde [x,y,t] = bode([2],[1 3 3 1]):
   - s mierkami na ľavej (y=f(t)) a pravej (x=f(t)) strane
   - s nezávislou logaritmickou osou y=f(t)
   - s nezávislou logaritmickou osou x=f(t)
3. Pre funkciu z=x.*exp(-x.^2-y.^2) vytvorte zobrazenia:

- sieťové zobrazenie s farebnou mapou 'cool'
- tieňové zobrazenie s farebnou mapou 'pink' a tieňovaním 'interpolated'
- lineárne zobrazenie s farebnou mapou 'hot'

4. Vytvorte jedno grafické okno, v ktorom bude 6 grafov (rozdelenie 2 x 3):
   - na pozícii [1,1] zobrazte sin(t);
   - na pozícii [1,2] zobrazte cos(t);
   - na pozícii [2,1] zobrazte log(t);
   - na pozícii [3,3] zobrazte e^t,

   kde t=[0.01:0.01:2*pi]

5. Vykreslite pre t=[0.01:0.01:2*pi] priebeh funkcie sin(t) tak, aby bol zobrazený iba interval <0.5*pi, 1.5*pi> (- interval na nezávislej osi). Nerobte to zmenou vektora t.