MATLAB
ver. 5

Petr Hora
CDM, ÚT AV ČR
Veleslavínova 11
301 14 Plzeň
hora@cdm.it.cas.cz

Abstrakt

Tyto fólie sloužily k výuce MATLABu v rámci předmětu *Matematické výpočetní systémy* přednášeného na ZČU Plzeň.

M A T L A B (MATrix LABoratory)

- je výkonné, interaktivní prostředí pro vědecké a inženýrské výpočty a vizualizaci dat.

- integruje numerickou analýzu, maticové výpočty a grafiku do uživatelsky příjemného prostředí, ve kterém se problémy a řešení zapisují stejně jako v matematice - bez obtíží tradičního programování.

M A T L A B (MATrix LABoratory)

- je výkonné, interaktivní prostředí pro vědecké a inženýrské výpočty a vizualizaci dat.

- integruje numerickou analýzu, maticové výpočty a grafiku do uživatelsky příjemného prostředí, ve kterém se problémy a řešení zapisují stejně jako v matematice - bez obtíží tradičního programování.

Historie (LINPACK, EISPACK, Cleve Moler)

MathWorks (licenční politika), Humusoft (zastoupení pro ČR)

M A T L A B (MATrix LABoratory)

- je výkonné, interaktivní prostředí pro vědecké a inženýrské výpočty a vizualizaci dat.

- integruje numerickou analýzu, maticové výpočty a grafiku do uživatelsky příjemného prostředí, ve kterém se problémy a řešení zapisují stejně jako v matematice - bez obtíží tradičního programování.

Historie (LINPACK, EISPACK, Cleve Moler)

MathWorks (licenční politika), Humusoft (zastoupení pro ČR)

Interpretační prostředí

Programovací jazyk

Snadná rozšiřitelnost systému (M-soubory), TOOLBOXY

Implementace na různých OS (PC, MAC, SGI, SUN, DEC, HP)

Instalace

Struktura adresářů

Popis menu a panelu nástrojů

Nápověda

Základní rysy

Jednoduché výpočty - kalkulačka

Operace	Symbol	Příklad
sčítání	+	3+22
odčítání	-	90-54
násobení	*	3.14*0.85
dělení	/ nebo \	56/8=8\56
umocňování	^	2^8

Výrazy se vyhodnocují zleva doprava s následující prioritou:

umocňování,
násobení a dělení,
sčítání a odčítání.

Prioritu lze změnit použitím závorek.

Pracovní prostor MATLABu

historie příkazů (šipky)

mazání příkazu (ESCAPE)

O proměnných

Pravidla o pojmenování proměnných	**Komentáře/Příklady**
Záleží na velikosti písmen	**Cost**, **cost**, **CoSt**, **COST** jsou rozdílné proměnné
Jméno proměnné může obsahovat až 31 znaků. Další znaky jsou ignorovány.	**rychlostauta**
Jméno proměnné musí začínat písmenem, dále se může vyskytovat libovolný počet písmen, číslic nebo podtržítek.	**rychlost_auta** **X51483**

Speciální proměnné	**Popis**
ans	Implicitní jméno proměnné užívané pro výsledky
pi	Ludolfovo číslo
eps	Přesnost na daném počítači
flops	Počet operací v plovoucí čárce
inf	Nekonečno (např. $1/0$)
NaN	Not-a-Number (např. $0/0$)
i nebo **j**	i=j=$\sqrt{-1}$, imaginární jednotka
nargin	Počet vstupních argumentů funkce
nargout	Počet výstupních argumentů funkce
realmin	Nejmenší možné kladné reálné číslo.
realmax	Největší možné kladné reálné číslo.

Komentáře, interpunkce a přerušení výpočtu

procento (**%**)

středník (**;**)

příkaz pokračování (**...**)

Crtl-C

Komplexní čísla

Znak pro komplexní jednotku: **i** nebo **j**

abs, **angle**, **real**, **imag**, **conj**

Matematické funkce

Trigonometrické funkce	
acos	Inverzní kosinus
acosh	Inverzní hyperbolický kosinus
acot	Inverzní kotangents
acoth	Inverzní hyperbolický kotangents
acsc	Inverzní kosecant
acsch	Inverzní hyperbolický kosecant
asec	Inverzní sekant
asech	Inverzní hyperbolický sekant
asin	Inverzní sinus
asinh	Inverzní hyperbolický sinus
atan	Inverzní hyperbolický tangents
atan2	Inverzní tangents
atanh	Inverzní hyperbolický tangents
cos	Kosinus
cosh	Hyperbolický kosinus
cot	Kotangents
coth	Hyperbolický kotangents

csc	Kosekant
csch	Hyperbolický kosekant
sec	Sekant
sech	Hyperbolický sekant
sin	Sinus
sinh	Hyperbolický sinus
tan	Tangents
tanh	Hyperbolický tangents

csc	Kosekant
csch	Hyperbolický kosekant
sec	Sekant
sech	Hyperbolický sekant
sin	Sinus
sinh	Hyperbolický sinus
tan	Tangents
tanh	Hyperbolický tangents

Exponenciální funkce	
exp	Exponenciála
log	Přirozený logaritmus
log10	Dekadický logaritmus
log2	Logaritmus při základu 2
pow2	Mocnina při základu 2
sqrt	Druhá odmocnina
nextpow2	Nejbližší vyšší mocnina při základu 2

Komplexní funkce	
abs	Absolutní hodnota nebo modul
angle	Fázový úhel
conj	Komplexně sdružená hodnota
imag	Imaginární část
real	Reálná část
unwrap	'Rozbalení' fázového úhlu
isreal	Test pro reálná pole
cplxpair	Setřídění komplexně sdružených párů

Komplexní funkce	
abs	Absolutní hodnota nebo modul
angle	Fázový úhel
conj	Komplexně sdružená hodnota
imag	Imaginární část
real	Reálná část
unwrap	'Rozbalení' fázového úhlu
isreal	Test pro reálná pole
cplxpair	Setřídění komplexně sdružených párů

Zaokrouhlovací funkce	
fix	Zaokrouhlování k nule
floor	Zaokrouhlování k $-\infty$
ceil	Zaokrouhlování k $+\infty$
round	Zaokrouhlování k nejbližšímu celému číslu
mod	Modulo
rem	Zbytek po dělení
sign	Signum

Transformace souřadnic	
cart2sph	Transformace kartézských souřadnic na sférické
cart2pol	Transformace kartézských souřadnic na polární
pol2cart	Transformace polárních souřadnic na kartézské
sph2cart	Transformace sférických souřadnic na kartézské

Transformace souřadnic	
cart2sph	Transformace kartézských souřadnic na sférické
cart2pol	Transformace kartézských souřadnic na polární
pol2cart	Transformace polárních souřadnic na kartézské
sph2cart	Transformace sférických souřadnic na kartézské

Funkce z teorie čísel	
factor	Rozklad na prvočísla
isprime	Testování prvočísel
primes	Generování prvočísel
gcd	Největší společný dělitel
lcm	Nejmenší společný násobek
rat	Racionální aproximace
rats	Výstup racionálních čísel
perms	Všechny možné permutace
nchoosek	Všechny kombinace N nad K

Speciální matematické funkce	
airy	Airyho funkce
besselj	Besselova funkce prvního druhu
bessely	Besselova funkce druhého druhu
besselh	Besselova funkce třetího druhu (Hankelova funkce)
besseli	Modifikovaná Besselova funkce prvního druhu
besselk	Modifikovaná Besselova funkce druhého druhu
beta	Funkce beta
betainc	Neúplná funkce beta
betaln	Logaritmus funkce beta
ellipj	Jacobiho eliptické funkce
ellipke	Úplný eliptický integrál
erf	Chybová funkce
erfc	Doplňková chybová funkce
erfcx	Měřítkovaná doplňková chybová funkce
erfinv	Inverzní chybová funkce
expint	Funkce exponenciálního integrálu
gamma	Funkce gama
gammainc	Neúplná funkce gama
gammaln	Logaritmus funkce gama

legendre	Legendreova funkce
cross	Vektorový součin
dot	Skalární součin

Příkazové okno

Práce s pracovním prostorem

who, **whos** (File-ShowWorkspace, **Workspace Browser**)

clear

memory, **pack**

Formáty zobrazení čísel

File-Preferences

Příkaz	π	**Komentář**
format	3.1416	5 číslic
format short	3.1416	5 číslic
format long	3.14159265358979	15 číslic
format short e	3.1416e+00	5 číslic s exponentem
format long e	3.14159265358793e+00	16 číslic s exponentem
format short g	3.1416	zvolí short nebo short e
format long g	3.14159265358979	zvolí long nebo long e
format hex	400921fb54442d18	hexadecimální formát
format bank	3.14	číslo zobrazí na dvě desetinná místa
format +	+	znaménko + pro kladná, - pro záporná a mezeru pro nulová čísla
format rat	355/113	racionální aproximace

Řízení příkazového okna

Příkaz	Popis
clc	Maže příkazové okno a přemístí kurzor do levého horního rohu.
home	Přemístí kurzor do levého horního rohu.
more	Stránkuje příkazové okno.

Informace o systému

computer

isieee

version, **ver**

Skripty

Funkce	**Popis**
disp(proměnná)	Zobrazí výsledek s potlačením jména proměnné
echo	Řídí zobrazování příkazů ve spuštěném skriptu
input	Uživatelský vstup
keyboard	Dočasné předání řízení klávesnici. Příkazem **return** se vrátí řízení skriptu.
pause	Přeruší skript a čeká na stisk libovolné klávesy.
pause(n)	Přeruší skript na **n** sekund.
waitforbuttonpress	Přeruší skript a čeká na stisk libovolné klávesy nebo tlačítka myši.

File-New-M-File, File-Save, File-Run Script

Práce se soubory a adresáři

Pracovní prostor MATLABu

diary

File-Print, File-Print Selection

File-Save Workspace as, File-Load Workspace

Ukládání, načítání a rušení proměnných

save, **load**

delete

Speciální funkce pro souborový vstup/výstup

dlmread, **dlmwrite**

wk1read, **wk1write**

Nízkoúrovňový souborový vstup/výstup

Kategorie	Funkce	Popis/Příklad syntaxe
Otevření a zavření	**fopen**	Otevření souboru **fid=fopen('filename', 'permission')**
	fclose	Zavření souboru **status=fclose(fid)**
Binární v/v	**fread**	Čtení z binárního souboru **A=fread(fid, num, precision)**
	fwrite	Zápis pole do binárního souboru **count=fwrite(fid, array, precision)**
Formátovaný v/v	**fscanf**	Čtení formátovaných dat ze souboru **A=fscanf(fid, format, num)**
	fprintf	Zápis formátovaných dat do souboru **count=fprintf(fid, format, A)**
	fgetl	Čtení řádky ze souboru, ruší znak konce řádky **line=fgetl(fid)**
	fgets	Čtení řádky ze souboru, zachová znak konce řádky **line=fgets(fid)**

Konverze řetězců	**sscanf**	Čtení formátovaných dat z řetězce **A=sscanf(string, format, num)**
	sprintf	Zápis formátovaných dat do řetězce **S=sprintf(format, A)**
Pohyb v souboru	**ferror**	Žádost o stav souborového v/v **message=ferror(fid)**
	feof	Test konce souboru **TF=feof(fid)**
	fseek	Nastavení pozice v souboru **status=fseek(fid, offset, origin)**
	ftell	Získání pozice v souboru **position=ftell(fid)**
	frewind	Přesunutí na začátek souboru **frewind(fid)**
Dočasné soubory	**tempdir**	Generování jména dočasného adresáře
	tempname	Generování jména dočasného souboru

Práce se soubory

Příkaz	**Popis**
cd nebo **pwd**	Zobrazí aktuální adresář.
p=cd	Vrací aktuální adresář v řetězci **p**.
delete *filename.m*	Maže M-soubor *filename.m*.
dir nebo **ls**	Zobrazí soubory v aktuálním adresáři.
d=dir	Vrací soubory v aktuálním adresáři v poli struktur **d**.
exist('*cow***', 'file')**	Kontrola existence M-souboru *cow.m*.
exist('*dname***', 'dir')**	Kontrola existence adresáře *dname*.
p=matlabroot	Vrací cestu k MATLABu v řetězci **p**.
type *cow*	Vypíše obsah *cow.m* v příkazovém okně.
what	Zobrazí setříděný výpis všech souborů MATLABu v aktuálním adresáři.
which *cow*	Zobrazí cestu k souboru *cow.m*.

Práce s cestou k souboru

Funkce	Popis
filesep	Oddělovač souborů pro danou platformu (\ na PC).
fullfile	Sestaví úplnou cestu z jednotlivých částí.
pathsep	Oddělovač cest pro danou platformu (; na PC).

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**

editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**

editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**
editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,

2. vestavěná funkce,

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**
editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,
2. vestavěná funkce,
3. lokální funkce,

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**
editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,
2. vestavěná funkce,
3. lokální funkce,
4. privátní funkce,

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**
editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,
2. vestavěná funkce,
3. lokální funkce,
4. privátní funkce,
5. funkce MEX, DLL, P nebo M. v aktuálním adresáři,

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**
editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,
2. vestavěná funkce,
3. lokální funkce,
4. privátní funkce,
5. funkce MEX, DLL, P nebo M. v aktuálním adresáři,
6. funkce MEX, DLL, P nebo M ve vyhledávací cestě MATLABu.

Vyhledávací cesta MATLABu

path, **matlabpath**, **addpath**, **rmpath**
editpath (File-SetPath, **Path Browser**)

Kroky k určení způsobu zpracování textového řetězce:

1. proměnná,
2. vestavěná funkce,
3. lokální funkce,
4. privátní funkce,
5. funkce MEX, DLL, P nebo M. v aktuálním adresáři,
6. funkce MEX, DLL, P nebo M ve vyhledávací cestě MATLABu.

Co se děje po spuštění MATLABu

matlabrc.m

startup.m

Pole a operace s nimi

- Jazyk MATLABu neobsahuje žádný příkaz pro nastavení dimenze nebo typu matice.

- Jazyk MATLABu neobsahuje žádný příkaz pro nastavení dimenze nebo typu matice.

- Potřebnou paměť alokuje MATLAB automaticky až do velikosti využitelné na konkrétním počítači.

Jednoduchá pole

- Jednoduchou matici lze zadat výčtem prvků, který je vložen do hranatých závorek.

Jednoduchá pole

- Jednoduchou matici lze zadat výčtem prvků, který je vložen do hranatých závorek.

- Prvky matice mohou být libovolné výrazy MATLABu.

Jednoduchá pole

- Jednoduchou matici lze zadat výčtem prvků, který je vložen do hranatých závorek.
- Prvky matice mohou být libovolné výrazy MATLABu.
- Jednotlivé prvky matice v řádce se oddělují mezerou nebo čárkou.

Jednoduchá pole

- Jednoduchou matici lze zadat výčtem prvků, který je vložen do hranatých závorek.
- Prvky matice mohou být libovolné výrazy MATLABu.
- Jednotlivé prvky matice v řádce se oddělují mezerou nebo čárkou.
- Jednotlivé řádky matice se oddělují středníkem nebo znakem konce řádky.

Indexace polí

Jednotlivé prvky matice mohou být zpřístupněny:

1. indexy uvnitř kulatých závorek,

2. pomocí logických polí.

Operace s dvojtečkou (**:**)

end

Sestavování polí

Velké matice můžete vytvářet pomocí malých matic, na které pohlížíte jako na prvky.

Technika sestavení pole	**Popis**
x=[2 2*pi sqrt(2) 2-3j]	Vytvoří řádkový vektor **x** ze zvolených prvků.
x=first:last	Vytvoří řádkový vektor **x** od **first** do **last** s krokem **jedna**.
x=first:increment:last	Vytvoří řádkový vektor **x** od **first** do **last** s krokem **increment**.
x=linspace(first, last, n)	Vytvoří řádkový vektor **x** od **first** do **last** mající **n** prvků.
x=logspace(first, last, n)	Vytvoří řádkový vektor **x** s logaritmickým rozložením od **10^first** do **10^last** mající **n** prvků.

Orientace pole

transpozice (’ a .’)

Matematika polí

Operace po prvcích	**Popis** a=[a1 a2 ... an], b=[b1 b2 ... bn], c = skalár
Skalární sčítání	a+c = [a1+c a2+c . . . an+c]
Skalární násobení	a*c = [a1*c a2*+c . . . an*c]
Sčítání polí	a+b = [a1+b1 a2+b2 . . . an+bn]
Násobení polí	a.*b = [a1*b1 a2*b2 . . . an*bn]
Pravostranné dělení polí	a./b = [a1/b1 a2/b2 . . . an/bn]
Levostranné dělení polí	a.\b = [a1\b1 a2\b2 . . . an\bn]
Umocňování polí	a.^c = [a1^c a2^c . . . an^c] c.^a = [c^a1 c^a2 . . . c^an] a.^b = [a1^b1 a2^b2 . . . an^bn]

Standardní pole

ones, zeros, eye

rand, randn

diag

Manipulace s poli

Adresování polí	Popis
A(r,c)	Adresace podpole pole **A** definovaná vektorem řádkových indexů **r** a vektorem sloupcových indexů **c**.
A(r,:)	Adresace podpole pole **A** definovaná vektorem řádkových indexů **r** a všemi sloupci.
A(:,c)	Adresace podpole pole **A** definovaná vektorem sloupcových indexů **c** a všemi řádky.
A(:)	Adresace všech prvků pole **A** jako sloupcově vytvořeného sloupcového vektoru. Pokud se **A(:)** objeví na levé straně přiřazovacího znaménka, znamená to vyplnění pole **A** prvky zpravé strany přiřazovacího znaménka bez změny jeho tvaru.
A(i)	Adresace podpole pole **A** definovaná jednoduchým indexovým vektorem **i**, jako kdyby **A** bylo sloupcovým vektorem **A(:)**.
A(x)	Adresace podpole pole **A** definovaná logickým polem **x**, **x** musí mít stejnou velikost jako pole **A**.

Skalární expanze

sub2ind, **ind2sub**

logická pole

Vyhledávání podpolí

Hledání pole	Popis
i=find(x)	Vrací indexy pole **x**, jehož prvky jsou nenulové.
[r,c]=find(x)	Vrací řádkové a sloupcové indexy pole **x**, jehož prvky jsou nenulové.

Funkce pro manipulaci s poli

flipud, **fliplr**, **rot90**

diag, **triu**, **tril**

reshape, **kron**, **repmat**

Velikost polí

Velikost pole	**Popis**
s=size(A)	Vrací řádkový vektor **s**, jehož první prvek je počet řádek pole **A** a jehož druhý prvek je počet sloupců pole **A**.
[r,c]=size(A)	Vrací dva skaláry **r** a **c**, které obsahují počet řádek a počet sloupců pole **A**.
r=size(A,1)	Vrací počet řádek pole **A** v proměnné **r**.
c=size(A,2)	Vrací počet sloupců pole **A** v proměnné **c**.
n=length(A)	Vrací **max(size(A))** v proměnné **n**.

Vícerozměrná pole

Vytváření vícerozměrných polí

zeros, **ones**, **rand**, **randn**

repmat

cat

Matematika a manipulace s vícerozměrnými poli

squeeze (singleton dimension)

reshape

sub2ind, **ind2sub**

flipdim (ekvivalent **flipud** a **fliplr** pro práci s vícerozměrnými poli)

shiftdim, **permute**, **ipermute** (jiný pohled na data)

Velikost vícerozměrných polí

size

ndims

Relační a logické operace

Relační operátory

Relační operátor	Popis
<	menší než
<=	menší než nebo rovno
>	větší než
>=	větší než nebo rovno
==	rovná se
~=	nerovná se

Lze porovnávat dvě pole stejné velikosti nebo pole a skalár.

Výsledkem je logické pole o velikosti porovnávaného pole.

Pozor = a ==

Logické pole lze použít při matematických operacích.

Logické operátory

Logický operátor	Popis
&	logický součin (AND)
|	logický součet (OR)
~	negace (NOT)

Lze porovnávat dvě pole stejné velikosti nebo pole a skalár.

Výsledkem je logické pole o velikosti porovnávaného pole.

Relační a logické funkce

Funkce	Popis
xor(x,y)	Výlučné OR. Vrací hodnotu True (1) pro každý prvek, kde buď **x** nebo **y** jsou nenulové (True). Vrací hodnotu False (0), kde jak **x** tak i **y** jsou nulové (False), nebo jsou oba nenulové (True).
any(x)	Vrací hodnotu True (1), jestliže alespoň jeden prvek vektoru **x** je nenulový. Vrací hodnotu True (1) pro každý sloupec matice **x**, který obsahuje nenulový prvek.
all(x)	Vrací hodnotu True (1), jestliže všechny prvky vektoru **x** jsou nenulové. Vrací hodnotu True (1) pro každý sloupec matice **x**, který má všechny prvky nenulové.

is – testovací funkce

NaNs (Not-a-Number) a prázdná pole ([ ])

isnan, **isempty**

Funkce pro práci s množinami, bity a bázemi

Práce s množinami

Na pole lze pohlížet jako na množiny.

isequal, **ismember**

unique, **union**, **intersect**, **setdiff**, **setxor**

Práce s bity

Logické operace na úrovni jednotlivých bitů.

bitand, **bitor**, **bitxor**

bitcmp, **bitshift**

bitget, **bitset**

bitmax

Převody číselných soustav

MATLAB poskytuje funkce, které umožňují převádět desítkovou číselnou soustavu na řadu jiných a naopak.

dec2bin, **bin2dec**

dec2hex, **hex2dec**

dec2base, **base2dec**

Maximální základ číselné soustavy je 36 (0-9 a A-Z).

Řetězce znaků

Vytvoření textového řetězce

Znakové řetězce v MATLABu jsou speciální numerická pole ASCII hodnot, která se zobrazují jako jejich znaková reprezentace.

Znakové řetězce představují text uzavřený v apostrofech, např. **'text'**.

Každý znak v řetězci je jedním prvkem v poli a vyžaduje pro sebe 2 byty (UNICODE).

Znakové řetězce, které mají více než jeden řádek, musí mít stejný počet sloupců!!!

double, **char**

char, **str2mat**, **strvcat** (ignoruje prázdné řetězce), **strcat**

Konverze čísla na řetězec a naopak

int2str, **num2str**, **mat2str**

sprintf, **fprintf** (bez fid)

str2num

sscanf

Příkaz	**Výsledek**
sprintf('%.0e', pi)	3e+000
sprintf('%.1e', pi)	3.1e+000
sprintf('%.0f', pi)	3
sprintf('%.1f', pi)	3.1
sprintf('%.0g', pi)	3
sprintf('%.5g', pi)	3.1416
sprintf('%8.5g', pi)	3.1416
sprintf('%3d', 8)	8
sprintf('%.3d', 8)	008

Řetězcové funkce

Funkce	Popis
deblank(S)	Odstraní koncové mezery z **S**.
ischar(S)	Vrací True (1), je-li **S** textový řetězec.
isletter(S)	Vrací True (1), je-li **S** písmeno.
isspace(S)	Vrací True (1), je-li **S** *bílá* mezera.
findstr(S1,S2)	Vrací místa, kde se kratší řetězec objevuje v delším.
strcmp(S1,S2)	Porovnává dva řetězce.
strncmp(S1,S2,n)	Porovnává prvních **n** znaků dvou řetězců.
strjust(S)	Zarovnává znakové pole **S**.
strmatch(S1,S2)	Vrací indexy řádků **S2**, které začínají **S1**.
strrep(S1,S2,S3)	Nahrazuje všechny výskyty **S2** v **S1** řetězcem **S3**.
strtok(S1,D)	Vrací z **S1** znaky, dokud nenajde oddělovač **D**.
lower(S)	Převádí **S** na malá písmena.
upper(S)	Převádí **S** na velká písmena.

Buňková pole řetězců

Řetězce v jednotlivých buňkách mohou mít různou délku.

Většina řetězcových funkcí pracuje jak s řetězcovými poli, tak s buňkovými poli řetězců.

cellstr, **iscellstr**

char

deal (vyjmutí více než jedné buňky)

Časové funkce

MATLAB interně ukládá datum a čas jako číslo v přesnosti double, které udává počet dní od začátku roku nula.

Tento formát je vhodný pro výpočty.

Pro zobrazení data a času nabízí MATLAB mnoho konverzních funkcí.

Aktuální datum a čas

now (interní formát)

date, **clock**

Konverze datumových formátů

MATLAB podporuje tři formáty:

1. číslo v přesnosti double (interní formát),
2. znakový řetězec v různých stylech,
3. číselný vektor.

datestr, **datenum**, **datevec**

Datumové funkce

MATLAB počítá dni od neděle.

weekday (pořadové číslo dne v týdnu, 1=neděle)

eomday (poslední den v měsíci)

calendar

Časové funkce

tic, toc

cputime

etime

Popisy časových grafů

datetick

Buňková pole a struktury

Buňková pole v MATLABu jsou pole, jejichž prvky jsou buňky.

Buňky v buňkovém poli mohou obsahovat libovolný datový typ MATLABu, např. číselné pole, text, objekt, jiné buňkové pole, strukturu.

Každá buňka může obsahovat jiný datový typ.

Buňkové pole může mít libovolný počet dimenzí.

Vytváření a zobrazení buňkových polí

Buňkové pole může být vytvořeno přiřazovacím příkazem nebo alokováním pole funkcí **cell**.

Adresace:

1. buňková indexace, např. **A(1,1)={12:-1:3}**

2. obsahová indexace, např. **A{1,1}=12:-1:3**

celldisp, **cellplot**

cell

Skládání a změna tvaru buňkových polí

K vytváření větších buňkových polí slouží hranaté závorky, **[ ]**, se kterými se zachází stejně jako při tvorbě číselných nebo znakových polí.

U buňkových polí lze používat všechny konvenční adresovací techniky, které se používají u číselných nebo znakových polí.

reshape

Zpřístupnění obsahu buňkového pole

{ } – odkaz na obsah,

( ) – odkaz na buňku.

Seznamy oddělené čárkami

deal (vyjmutí více než jedné buňky)

help lists

Buňková pole znakových řetězců

cellstr, char

iscellstr, ischar

Vytváření a zobrazení struktur

Struktury jsou podobné buňkovým polím, neboť také dokáží seskupit data různého typu do jediné proměnné.

Místo adresování prvků číslem jsou však prvky struktury adresovány jménem (položkou).

Struktury mohou mít libovolný počet dimenzí.

Struktura může být vytvořena přiřazovacím příkazem nebo funkcí **struct**.

Zpřístupnění obsahu položky struktury

Jednotlivé položky struktury se zpřístupňují buď přímou adresací (přes tečku) nebo pomocí funkce **getfield**.

fieldnames

getfield, **setfield**

rmfield

Konverzní a testovací funkce

struct2cell, cell2struct

num2cell

cellstr, char

Programování

Cyklus FOR

Cyklus **for** slouží pro předem daný počet opakování skupiny příkazů.

Cyklus FOR

Cyklus **for** slouží pro předem daný počet opakování skupiny příkazů.

Obecný tvar cyklu **for** je

```
for v=výraz
  příkazy
end
```

Výraz je ve skutečnosti matice (může být i vícerozměrná).

Sloupce této matice jsou postupně přiřazovány proměnné **v** a následně jsou provedeny *příkazy*.

Cyklus FOR

Cyklus **for** slouží pro předem daný počet opakování skupiny příkazů.

Obecný tvar cyklu **for** je

```
for v=výraz
  příkazy
end
```

Výraz je ve skutečnosti matice (může být i vícerozměrná).

Sloupce této matice jsou postupně přiřazovány proměnné **v** a následně jsou provedeny *příkazy*.

Názorněji lze celou záležitost vyjádřit jako

```
E=výraz;
[m,n]=size(E);
for j=1:n
  v=E(:,j);
  příkazy
end
```

Obvykle je výraz ve tvaru **m:n** nebo **m:i:n**, což je matice s jednou řádkou, takže sloupce jsou skaláry. V tomto speciálním případě se chová cyklus **for** MATLABu jako cykly FOR a DO v jiných jazycích.

Cyklus WHILE

Cyklus **while** umožňuje opakovat příkaz nebo skupinu příkazů v závislosti na logické podmínce.

Cyklus WHILE

Cyklus **while** umožňuje opakovat příkaz nebo skupinu příkazů v závislosti na logické podmínce.

Obecný tvar cyklu **while** je

```
while výraz
  příkazy
end
```

Příkazy se opakují tak dlouho, dokud jsou všechny prvky ve *výrazu* (výraz je matice, může být i vícerozměrná) nenulové.

Výraz je téměř vždy skalárním relačním výrazem, takže nenulové hodnoty odpovídají logické hodnotě TRUE.

Pokud výraz není skalár, můžete ho redukovat funkcí **any** nebo **all**.

Konstrukce IF-ELSE-END

Příkaz **if** slouží k větvení algoritmu.

Konstrukce IF-ELSE-END

Příkaz **if** slouží k větvení algoritmu.

Obecný tvar příkazu **if** je

```
if výraz
  příkazy
[elseif výraz
  příkazy]
[else
  příkazy]
end
```

Výrazy jsou ve skutečnosti matice (mohou být i vícerozměrné).

Příkazy mezi **if** a **end** se vyhodnotí, pokud všechny prvky ve *výrazu* jsou nenulové (True).

Pokud *výraz* obsahuje několik logických podvýrazů, provádí se zkrácené vyhodnocení *výrazu*.

Konstrukce SWITCH-CASE

Příkaz **switch** slouží také k větvení algoritmu.

Konstrukce SWITCH-CASE

Příkaz **switch** slouží také k větvení algoritmu.

Obecný tvar příkazu **switch** je

```
switch výraz
case test_expression1
  příkazy1
case (test_expression2, test_expression3, test_expression4)
  příkazy2
otherwise
  příkazy3
end
```

Výrazy jsou buď skaláry nebo textové řetězce.

Funkce (M-soubory)

Pravidla pro tvorbu M-souborů

M-soubory musí splňovat řadu kritérií a navíc je rozumné dodržovat některá doporučení:

Pravidla pro tvorbu M-souborů

M-soubory musí splňovat řadu kritérií a navíc je rozumné dodržovat některá doporučení:

1. Jméno M-souboru a jméno funkce, které se objevuje v prvním řádku souboru by mělo být stejné. Ve skutečnosti MATLAB jméno funkce v prvním řádku ignoruje a spouští funkci podle jména souboru.

Pravidla pro tvorbu M-souborů

M-soubory musí splňovat řadu kritérií a navíc je rozumné dodržovat některá doporučení:

1. Jméno M-souboru a jméno funkce, které se objevuje v prvním řádku souboru by mělo být stejné. Ve skutečnosti MATLAB jméno funkce v prvním řádku ignoruje a spouští funkci podle jména souboru.

2. Jméno M-souboru může mít až 31 znaků. Toto maximum může být omezeno systémem. Znaky nad tento limit MATLAB ignoruje.

Pravidla pro tvorbu M-souborů

M-soubory musí splňovat řadu kritérií a navíc je rozumné dodržovat některá doporučení:

1. Jméno M-souboru a jméno funkce, které se objevuje v prvním řádku souboru by mělo být stejné. Ve skutečnosti MATLAB jméno funkce v prvním řádku ignoruje a spouští funkci podle jména souboru.

2. Jméno M-souboru může mít až 31 znaků. Toto maximum může být omezeno systémem. Znaky nad tento limit MATLAB ignoruje.

3. Jméno funkce musí začínat písmenem, za kterým se pak dále mohou vyskytovat písmena, číslice a podtržítka. Stejné pravidlo platí i pro jména proměnných.

Pravidla pro tvorbu M-souborů

M-soubory musí splňovat řadu kritérií a navíc je rozumné dodržovat některá doporučení:

1. Jméno M-souboru a jméno funkce, které se objevuje v prvním řádku souboru by mělo být stejné. Ve skutečnosti MATLAB jméno funkce v prvním řádku ignoruje a spouští funkci podle jména souboru.

2. Jméno M-souboru může mít až 31 znaků. Toto maximum může být omezeno systémem. Znaky nad tento limit MATLAB ignoruje.

3. Jméno funkce musí začínat písmenem, za kterým se pak dále mohou vyskytovat písmena, číslice a podtržítka. Stejné pravidlo platí i pro jména proměnných.

4. První řádka M-souboru se nazývá řádka deklarace funkce a musí obsahovat slovo **function** následované volací syntaxí funkce v jejím nejobecnějším tvaru. Vstupní a výstupní proměnné určené v prvním řádku jsou proměnné lokální k funkci. Vstupní proměnné obsahují data předaná funkci a výstupní proměnné obsahují data funkcí vrácená. Není možné získávat data přes vstupní proměnné.

Pravidla pro tvorbu M-souborů

M-soubory musí splňovat řadu kritérií a navíc je rozumné dodržovat některá doporučení:

1. Jméno M-souboru a jméno funkce, které se objevuje v prvním řádku souboru by mělo být stejné. Ve skutečnosti MATLAB jméno funkce v prvním řádku ignoruje a spouští funkci podle jména souboru.

2. Jméno M-souboru může mít až 31 znaků. Toto maximum může být omezeno systémem. Znaky nad tento limit MATLAB ignoruje.

3. Jméno funkce musí začínat písmenem, za kterým se pak dále mohou vyskytovat písmena, číslice a podtržítka. Stejné pravidlo platí i pro jména proměnných.

4. První řádka M-souboru se nazývá řádka deklarace funkce a musí obsahovat slovo **function** následované volací syntaxí funkce v jejím nejobecnějším tvaru. Vstupní a výstupní proměnné určené v prvním řádku jsou proměnné lokální k funkci. Vstupní proměnné obsahují data předaná funkci a výstupní proměnné obsahují data funkcí vrácená. Není možné získávat data přes vstupní proměnné.

5. První množina souvislých komentářových řádek po řádce deklarace funkce slouží jako **nápovědný text pro funkci**. První komentářová řádka se nazývá řádka **H1** a je to řádka , kterou vyhledává příkaz **lookfor**. Řádka **H1** většinou obsahuje jméno funkce psané velkými písmeny a stručný popis účelu funkce. Následující komentářové řádky popisují volající syntaxi, použitý algoritmus a jednoduché příklady.

5. První množina souvislých komentářových řádek po řádce deklarace funkce slouží jako **nápovědný text pro funkci**. První komentářová řádka se nazývá řádka **H1** a je to řádka , kterou vyhledává příkaz **lookfor**. Řádka **H1** většinou obsahuje jméno funkce psané velkými písmeny a stručný popis účelu funkce. Následující komentářové řádky popisují volající syntaxi, použitý algoritmus a jednoduché příklady.

6. Všechny příkazy po první množina souvislých komentářových řádek tvoří **tělo funkce**. Tělo funkce obsahuje příkazy MATLABu, které zpracovávají vstupní argumenty a ukládají výsledky do výstupních argumentů.

5. První množina souvislých komentářových řádek po řádce deklarace funkce slouží jako **nápovědný text pro funkci**. První komentářová řádka se nazývá řádka **H1** a je to řádka , kterou vyhledává příkaz **lookfor**. Řádka **H1** většinou obsahuje jméno funkce psané velkými písmeny a stručný popis účelu funkce. Následující komentářové řádky popisují volající syntaxi, použitý algoritmus a jednoduché příklady.

6. Všechny příkazy po první množina souvislých komentářových řádek tvoří **tělo funkce**. Tělo funkce obsahuje příkazy MATLABu, které zpracovávají vstupní argumenty a ukládají výsledky do výstupních argumentů.

7. Zpracování M-souboru končí poté, co je zpracována poslední řádka souboru nebo se při zpracování narazí na příkaz **return**.

5. První množina souvislých komentářových řádek po řádce deklarace funkce slouží jako **nápovědný text pro funkci**. První komentářová řádka se nazývá řádka **H1** a je to řádka , kterou vyhledává příkaz **lookfor**. Řádka **H1** většinou obsahuje jméno funkce psané velkými písmeny a stručný popis účelu funkce. Následující komentářové řádky popisují volající syntaxi, použitý algoritmus a jednoduché příklady.

6. Všechny příkazy po první množina souvislých komentářových řádek tvoří **tělo funkce**. Tělo funkce obsahuje příkazy MATLABu, které zpracovávají vstupní argumenty a ukládají výsledky do výstupních argumentů.

7. Zpracování M-souboru končí poté, co je zpracována poslední řádka souboru nebo se při zpracování narazí na příkaz **return**.

8. Funkce se může přerušit a vrátit tak řízení příkazovému oknu voláním funkce **error**. Tato funkce je užitečná pro signalizování nesprávného užití funkce, např.:

```
iflength(val)>1
  error('VALmustbeascalar.')
end
```

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

9. Funkce může podat varovné hlášení a potom dále pokračovat ve zpracovávání příkazů voláním funkce **warning**. Tato funkce je užitečná pro podávání zpráv o výjimkách a jiných zvláštnostech. **warning('text')** jednoduše zobrazí textový řetězec v příkazovém okně. Rozdíl mezi funkcí **warning** a funkcí **disp** je v tom, že varovné hlášení (warning) může být zapnuto resp. vypnuto globálně příkazem **warning on** resp. **warning off**.

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

9. Funkce může podat varovné hlášení a potom dále pokračovat ve zpracovávání příkazů voláním funkce **warning**. Tato funkce je užitečná pro podávání zpráv o výjimkách a jiných zvláštnostech. **warning('text')** jednoduše zobrazí textový řetězec v příkazovém okně. Rozdíl mezi funkcí **warning** a funkcí **disp** je v tom, že varovné hlášení (warning) může být zapnuto resp. vypnuto globálně příkazem **warning on** resp. **warning off**.

10. Z M-souboru **lze volat skripty**. Skript se v tomto případě vyhodnocuje v pracovním prostoru funkce, ne v pracovním prostředí příkazového okna.

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

9. Funkce může podat varovné hlášení a potom dále pokračovat ve zpracovávání příkazů voláním funkce **warning**. Tato funkce je užitečná pro podávání zpráv o výjimkách a jiných zvláštnostech. **warning('text')** jednoduše zobrazí textový řetězec v příkazovém okně. Rozdíl mezi funkcí **warning** a funkcí **disp** je v tom, že varovné hlášení (warning) může být zapnuto resp. vypnuto globálně příkazem **warning on** resp. **warning off**.

10. Z M-souboru **lze volat skripty**. Skript se v tomto případě vyhodnocuje v pracovním prostoru funkce, ne v pracovním prostředí příkazového okna.

11. V jednom M-souboru se může vyskytovat více funkcí. První funkce je tzv. primární funkce, ostatní jsou tzv. podfunkce neboli **lokální funkce**.

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

9. Funkce může podat varovné hlášení a potom dále pokračovat ve zpracovávání příkazů voláním funkce **warning**. Tato funkce je užitečná pro podávání zpráv o výjimkách a jiných zvláštnostech. **warning('text')** jednoduše zobrazí textový řetězec v příkazovém okně. Rozdíl mezi funkcí **warning** a funkcí **disp** je v tom, že varovné hlášení (warning) může být zapnuto resp. vypnuto globálně příkazem **warning on** resp. **warning off**.

10. Z M-souboru **lze volat skripty**. Skript se v tomto případě vyhodnocuje v pracovním prostoru funkce, ne v pracovním prostředí příkazového okna.

11. V jednom M-souboru se může vyskytovat více funkcí. První funkce je tzv. primární funkce, ostatní jsou tzv. podfunkce neboli **lokální funkce**.

12. Lokální funkce mohou být volány primární funkcí M-souboru nebo jinými lokálními funkcemi ve stejném M-souboru. Lokální funkce mají svůj vlastní pracovní prostor.

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

9. Funkce může podat varovné hlášení a potom dále pokračovat ve zpracovávání příkazů voláním funkce **warning**. Tato funkce je užitečná pro podávání zpráv o výjimkách a jiných zvláštnostech. **warning('text')** jednoduše zobrazí textový řetězec v příkazovém okně. Rozdíl mezi funkcí **warning** a funkcí **disp** je v tom, že varovné hlášení (warning) může být zapnuto resp. vypnuto globálně příkazem **warning on** resp. **warning off**.

10. Z M-souboru **lze volat skripty**. Skript se v tomto případě vyhodnocuje v pracovním prostoru funkce, ne v pracovním prostředí příkazového okna.

11. V jednom M-souboru se může vyskytovat více funkcí. První funkce je tzv. primární funkce, ostatní jsou tzv. podfunkce neboli **lokální funkce**.

12. Lokální funkce mohou být volány primární funkcí M-souboru nebo jinými lokálními funkcemi ve stejném M-souboru. Lokální funkce mají svůj vlastní pracovní prostor.

13. Lokální funkce se mohou objevit po primární funkci v libovolném pořadí. Nápovědný text pro lokální funkce není přes příkaz **help** dostupný.

Pokud je funkce **error** spuštěna s prázdným řetězcem, žádná akce se nevykoná.

9. Funkce může podat varovné hlášení a potom dále pokračovat ve zpracovávání příkazů voláním funkce **warning**. Tato funkce je užitečná pro podávání zpráv o výjimkách a jiných zvláštnostech. **warning('text')** jednoduše zobrazí textový řetězec v příkazovém okně. Rozdíl mezi funkcí **warning** a funkcí **disp** je v tom, že varovné hlášení (warning) může být zapnuto resp. vypnuto globálně příkazem **warning on** resp. **warning off**.

10. Z M-souboru **lze volat skripty**. Skript se v tomto případě vyhodnocuje v pracovním prostoru funkce, ne v pracovním prostředí příkazového okna.

11. V jednom M-souboru se může vyskytovat více funkcí. První funkce je tzv. primární funkce, ostatní jsou tzv. podfunkce neboli **lokální funkce**.

12. Lokální funkce mohou být volány primární funkcí M-souboru nebo jinými lokálními funkcemi ve stejném M-souboru. Lokální funkce mají svůj vlastní pracovní prostor.

13. Lokální funkce se mohou objevit po primární funkci v libovolném pořadí. Nápovědný text pro lokální funkce není přes příkaz **help** dostupný.

14. Doporučuje se uvodit jméno lokální funkce slovem **local**, např. **local_myfun**, neboť to zlepšuje čitelnost primární funkce. Jméno každé lokální funkce může mít až 31 znaků.

14. Doporučuje se uvodit jméno lokální funkce slovem **local**, např. **local_myfun**, neboť to zlepšuje čitelnost primární funkce. Jméno každé lokální funkce může mít až 31 znaků.

15. Z M-souboru lze též volat tzv. privátní M-soubory, což jsou standardní M-soubory, které se nalézají v podadresáři volající funkce. Tento podadresář má název **private**. K privátním M-souborům mají přístup pouze M-soubory z nejbližšího nadřazeného adresáře.

14. Doporučuje se uvodit jméno lokální funkce slovem **local**, např. **local_myfun**, neboť to zlepšuje čitelnost primární funkce. Jméno každé lokální funkce může mít až 31 znaků.

15. Z M-souboru lze též volat tzv. privátní M-soubory, což jsou standardní M-soubory, které se nalézají v podadresáři volající funkce. Tento podadresář má název **private**. K privátním M-souborům mají přístup pouze M-soubory z nejbližšího nadřazeného adresáře.

16. Doporučuje se jméno privátní funkce uvodit slovem **private**, např. **private_myfun**, neboť to zlepšuje čitelnost primární funkce. Jméno každé privátní funkce může mít až 31 znaků.

Vstupní a výstupní argumenty

Funkce MATLABu mohou mít libovolný počet vstupních a výstupních argumentů. Vlastnosti těchto argumentů jsou následující:

Vstupní a výstupní argumenty

Funkce MATLABu mohou mít libovolný počet vstupních a výstupních argumentů. Vlastnosti těchto argumentů jsou následující:

1. M-soubor nemusí mít žádný vstupní nebo výstupní argument.

Vstupní a výstupní argumenty

Funkce MATLABu mohou mít libovolný počet vstupních a výstupních argumentů. Vlastnosti těchto argumentů jsou následující:

1. M-soubor nemusí mít žádný vstupní nebo výstupní argument.

2. Funkce lze volat s menším počtem vstupních nebo výstupních argumentů než je specifikováno v řádce deklarace funkce. Funkce nemohou být volány s více vstupními nebo výstupními argumenty.

Vstupní a výstupní argumenty

Funkce MATLABu mohou mít libovolný počet vstupních a výstupních argumentů. Vlastnosti těchto argumentů jsou následující:

1. M-soubor nemusí mít žádný vstupní nebo výstupní argument.

2. Funkce lze volat s menším počtem vstupních nebo výstupních argumentů než je specifikováno v řádce deklarace funkce. Funkce nemohou být volány s více vstupními nebo výstupními argumenty.

3. Počet vstupních resp. výstupních argumentů použitých při volání funkce lze zjistit funkcí **nargin** resp. **nargout**.

Vstupní a výstupní argumenty

Funkce MATLABu mohou mít libovolný počet vstupních a výstupních argumentů. Vlastnosti těchto argumentů jsou následující:

1. M-soubor nemusí mít žádný vstupní nebo výstupní argument.

2. Funkce lze volat s menším počtem vstupních nebo výstupních argumentů než je specifikováno v řádce deklarace funkce. Funkce nemohou být volány s více vstupními nebo výstupními argumenty.

3. Počet vstupních resp. výstupních argumentů použitých při volání funkce lze zjistit funkcí **nargin** resp. **nargout**.

4. Když je funkce volána, vstupní proměnné nejsou kopírovány do pracovního prostoru funkce, pouze jsou jejich hodnoty funkci zpřístupněny. Pokud je ale libovolná vstupní proměnná změněna, dojde k jejímu zkopírování do pracovního prostoru funkce. Z důvodu ušetření paměti a urychlení výpočtů je proto výhodnější z velkého pole nejprve vyjmout příslušné prvky a ty modifikovat, než přímo modifikovat velké pole a tím vyvolat jeho kopírování do pracovního

prostoru funkce. Pokud se použije stejná proměnná pro vstup i výstup, dojde k okamžitému kopírování této proměnné do pracovního prostoru funkce.

prostoru funkce. Pokud se použije stejná proměnná pro vstup i výstup, dojde k okamžitému kopírování této proměnné do pracovního prostoru funkce.

5. Pokud funkce deklaruje jeden nebo více výstupních argumentů, ale při volání funkce není žádán žádný výstup, jednoduše se hodnoty výstupním proměnným nepřiřadí.

prostoru funkce. Pokud se použije stejná proměnná pro vstup i výstup, dojde k okamžitému kopírování této proměnné do pracovního prostoru funkce.

5. Pokud funkce deklaruje jeden nebo více výstupních argumentů, ale při volání funkce není žádán žádný výstup, jednoduše se hodnoty výstupním proměnným nepřiřadí.

6. Funkce může akceptovat proměnný a neomezený počet vstupních argumentů, pokud se uvede jako poslední vstupní argument v řádce deklarace funkce slovo **varargin**. **varargin** je předdefinované buňkové pole, jehož i-tá buňka je i-tý argument od místa výskytu **varargin**. Např.

   ```
   function a=myfunction(a,b,varargin)
   ```

   pokud je volána jako **a=myfunction(r,s,x,y,z)**, pak uvnitř funkce **varargin{1}** obsahuje pole **x**, **varargin{2}** obsahuje pole **y** a **varargin{3}** obsahuje pole **z**. Funkce **nargin** vrací skutečný počet vstupních argumentů.

prostoru funkce. Pokud se použije stejná proměnná pro vstup i výstup, dojde k okamžitému kopírování této proměnné do pracovního prostoru funkce.

5. Pokud funkce deklaruje jeden nebo více výstupních argumentů, ale při volání funkce není žádán žádný výstup, jednoduše se hodnoty výstupním proměnným nepřiřadí.

6. Funkce může akceptovat proměnný a neomezený počet vstupních argumentů, pokud se uvede jako poslední vstupní argument v řádce deklarace funkce slovo **varargin**. **varargin** je předdefinované buňkové pole, jehož i-tá buňka je i-tý argument od místa výskytu **varargin**. Např.

   ```
   function a=myfunction(a,b,varargin)
   ```

   pokud je volána jako **a=myfunction(r,s,x,y,z)**, pak uvnitř funkce **varargin{1}** obsahuje pole **x**, **varargin{2}** obsahuje pole **y** a **varargin{3}** obsahuje pole **z**. Funkce **nargin** vrací skutečný počet vstupních argumentů.

7. Funkce může akceptovat proměnný a neomezený počet výstupních argumentů, pokud se uvede jako poslední výstupní argument v řádce deklarace funkce slovo

varargout. **varargout** je předdefinované buňkové pole, jehož i-tá buňka je i-tý argument od místa výskytu **varargout**. Např.

```
function varargout=myfunction(x)
```

pokud je volána jako **[r,s]=myfunction(a)**, pak uvnitř funkce obsah **varargout{1}** bude přiřazen poli **r** a obsah **varargin{2}** poli **s**. Funkce **nargout** vrací skutečný počet výstupních argumentů.

Pracovní prostor funkcí

Každá funkce má svůj vlastní dočasný pracovní prostor, který je vytvořen při každém volání funkce a smazán při ukončení funkce. Funkce mohou být v MATLABu volány rekurzivně a každé volání má svůj oddělený pracovní prostor.

Kromě vstupních a výstupních argumentů poskytuje MATLAB několik dalších technik pro komunikaci mezi pracovním prostorem funkcí a základním pracovním prostorem:

Pracovní prostor funkcí

Každá funkce má svůj vlastní dočasný pracovní prostor, který je vytvořen při každém volání funkce a smazán při ukončení funkce. Funkce mohou být v MATLABu volány rekurzivně a každé volání má svůj oddělený pracovní prostor.

Kromě vstupních a výstupních argumentů poskytuje MATLAB několik dalších technik pro komunikaci mezi pracovním prostorem funkcí a základním pracovním prostorem:

1. Funkce mohou sdílet proměnné s jinými funkcemi, základním pracovním prostorem a rekurzivně volanými funkcemi přes proměnné, které jsou deklarovány jako **global**. Proměnná musí být deklarována jako globální v každém požadovaném pracovním prostoru.
   Z programátorského hlediska není použití globálních proměnných zrovna nejelegantnější. Pokud se globální proměnné přesto používají, doporučuje se volit jejich název co nejdelší a složený z velkých písmen.

Pracovní prostor funkcí

Každá funkce má svůj vlastní dočasný pracovní prostor, který je vytvořen při každém volání funkce a smazán při ukončení funkce. Funkce mohou být v MATLABu volány rekurzivně a každé volání má svůj oddělený pracovní prostor.

Kromě vstupních a výstupních argumentů poskytuje MATLAB několik dalších technik pro komunikaci mezi pracovním prostorem funkcí a základním pracovním prostorem:

1. Funkce mohou sdílet proměnné s jinými funkcemi, základním pracovním prostorem a rekurzivně volanými funkcemi přes proměnné, které jsou deklarovány jako **global**. Proměnná musí být deklarována jako globální v každém požadovaném pracovním prostoru.
   Z programátorského hlediska není použití globálních proměnných zrovna nejelegantnější. Pokud se globální proměnné přesto používají, doporučuje se volit jejich název co nejdelší a složený z velkých písmen.

2. Kromě sdílení dat přes globální proměnné, poskytuje MATLAB funkci **evalin**, která umožňuje přepnout se do jiného pracovního prostoru, tam vyhodnotit výraz a vrátit výsledek do aktuálního pracovního prostoru.

Funkce **evalin** je podobná funkci **eval**, až na to že je textový řetězec vyhodnocen buď ve *volacím* anebo v*základním* pracovním prostoru. *Volací* pracovní prostor je pracovní prostor, ze kterého byla volána současná funkce. *Základní* pracovní prostor je pracovní prostor příkazovéhooknaMATLABu. Např.

```
A=evalin('caller','výraz')
```

vyhodnotí '*výraz*' ve volacím pracovním prostoru a vrátí výsledek do proměnné **A** v současném pracovním prostoru,

```
A=evalin('base','výraz')
```

vyhodnotí '*výraz*' v základním pracovním prostoru a vrátí výsledek do proměnné **A** v současném pracovním prostoru.

Funkce **evalin** je podobná funkci **eval**, až na to že je textový řetězec vyhodnocen buď ve *volacím* anebo v*základním* pracovním prostoru. *Volací* pracovní prostor je pracovní prostor, ze kterého byla volána současná funkce. *Základní* pracovní prostor je pracovní prostor příkazovéhooknaMATLABu. Např.

```
A=evalin('caller','výraz')
```

vyhodnotí '*výraz*' ve volacím pracovním prostoru a vrátí výsledek do proměnné **A** v současném pracovním prostoru,

```
A=evalin('base','výraz')
```

vyhodnotí '*výraz*' v základním pracovním prostoru a vrátí výsledek do proměnné **A** v současném pracovním prostoru.

3. Poněvadž existuje možnost vyhodnotit výraz v jiném pracovním prostoru, má smysl umožnit přiřazení výsledku nějakého výrazu v současném pracovním prostoru do proměnné v jiném pracovním prostoru. Tuto možnost poskytuje funkce **assignin**.

```
assignin('pracovní prostor', 'jméno proměnné',X),
```

kde pracovní prostor je buď **'caller'** anebo **'base'**, přiřadí obsah proměnné X v současném pracovním prostoru do proměnné '$jméno\ proměnné$' v pracovním prostoru **'caller'** nebo **'base'**.

kde pracovní prostor je buď **'caller'** anebo **'base'**, přiřadí obsah proměnné X v současném pracovním prostoru do proměnné '$jméno\ proměnné$' v pracovním prostoru **'caller'** nebo **'base'**.

4. Funkce **inputname** umožňuje určit jména proměnných použitých při volání funkce. Např.

```
y=myfunction(xdot,time,sqrt(2))
```

   Vyvoláním **inputname(1)** uvnitř **myfunction** dostaneme textový řetězec **'xdot'**, **inputname(2)** vrátí **'time'** a **inputname(3)** vrátí prázdné pole, neboť **sqrt(2)** není proměnná.

kde pracovní prostor je buď **'caller'** anebo **'base'**, přiřadí obsah proměnné X v současném pracovním prostoru do proměnné '$jméno\ proměnné$' v pracovním prostoru **'caller'** nebo **'base'**.

4. Funkce **inputname** umožňuje určit jména proměnných použitých při volání funkce. Např.

   ```
   y=myfunction(xdot,time,sqrt(2))
   ```

   Vyvoláním **inputname(1)** uvnitř **myfunction** dostaneme textový řetězec **'xdot'**, **inputname(2)** vrátí **'time'** a **inputname(3)** vrátí prázdné pole, neboť **sqrt(2)** není proměnná.

5. Jméno M-souboru, který je právě spuštěn, lze uvnitř funkce zjistit proměnnou **mfilename**. Např. jeli spuštěn M-soubor **myfunction**, obsahuje pracovní prostor funkce proměnnou **mfilename**, což je textový řetězec **'myfunction'**. Tato proměnná existuje také u skriptových souborů.

M-soubory a vyhledávací cesta MATLABu

Techniky, které MATLAB používá pro co největší urychlení práce s M-soubory (nalezení, otevření a spuštění), jsou následující:

M-soubory a vyhledávací cesta MATLABu

Techniky, které MATLAB používá pro co největší urychlení práce s M-soubory (nalezení, otevření a spuštění), jsou následující:

1. Když MATLAB spouští M-soubor poprvé, tak otevře odpovídající textový soubor a **zkompiluje příkazy ve tvaru vnitřního pseudokódu** do paměti. Tento postup urychlí následná spouštění tohoto M-souboru. Pokud funkce obsahuje odkazy na jiné M-soubory, jsou tyto taktéž zkompilovány do paměti.

M-soubory a vyhledávací cesta MATLABu

Techniky, které MATLAB používá pro co největší urychlení práce s M-soubory (nalezení, otevření a spuštění), jsou následující:

1. Když MATLAB spouští M-soubor poprvé, tak otevře odpovídající textový soubor a **zkompiluje příkazy ve tvaru vnitřního pseudokódu** do paměti. Tento postup urychlí následná spouštění tohoto M-souboru. Pokud funkce obsahuje odkazy na jiné M-soubory, jsou tyto taktéž zkompilovány do paměti.

2. Funkce **inmem** vrací buňkové pole řetězců, které obsahuje seznam funkcí a skriptů zkompilovaných v paměti.

M-soubory a vyhledávací cesta MATLABu

Techniky, které MATLAB používá pro co největší urychlení práce s M-soubory (nalezení, otevření a spuštění), jsou následující:

1. Když MATLAB spouští M-soubor poprvé, tak otevře odpovídající textový soubor a **zkompiluje příkazy ve tvaru vnitřního pseudokódu** do paměti. Tento postup urychlí následná spouštění tohoto M-souboru. Pokud funkce obsahuje odkazy na jiné M-soubory, jsou tyto taktéž zkompilovány do paměti.

2. Funkce **inmem** vrací buňkové pole řetězců, které obsahuje seznam funkcí a skriptů zkompilovaných v paměti.

3. Zkompilovanou verzi funkce lze uložit na disk příkazem **pcode**. MATLAB dává P-souborům přednost před M-soubory.

M-soubory a vyhledávací cesta MATLABu

Techniky, které MATLAB používá pro co největší urychlení práce s M-soubory (nalezení, otevření a spuštění), jsou následující:

1. Když MATLAB spouští M-soubor poprvé, tak otevře odpovídající textový soubor a **zkompiluje příkazy ve tvaru vnitřního pseudokódu** do paměti. Tento postup urychlí následná spouštění tohoto M-souboru. Pokud funkce obsahuje odkazy na jiné M-soubory, jsou tyto taktéž zkompilovány do paměti.

2. Funkce **inmem** vrací buňkové pole řetězců, které obsahuje seznam funkcí a skriptů zkompilovaných v paměti.

3. Zkompilovanou verzi funkce lze uložit na disk příkazem **pcode**. MATLAB dává P-souborům přednost před M-soubory.

4. Kroky používané při hledání funkce:

   - Je to proměnná?
   - Je to vestavěná funkce?

- Je to lokální funkce?
- Je to privátní funkce?
- Je to funkce MEX, DLL, P nebo M v aktuálním adresáři?
- Je to funkce MEX, DLL, P nebo M ve vyhledávací cestě?

- Je to lokální funkce?
- Je to privátní funkce?
- Je to funkce MEX, DLL, P nebo M v aktuálním adresáři?
- Je to funkce MEX, DLL, P nebo M ve vyhledávací cestě?

5. Když je MATLAB spuštěn, schová si jména a umístění všech M-souborů nacházejících se v podadresáři toolbox a všech jeho podadresářích. Toto slouží k rychlému nalezení a spuštění M-souborů a urychluje to také příkaz **lookfor**.

- Je to lokální funkce?
- Je to privátní funkce?
- Je to funkce MEX, DLL, P nebo M v aktuálním adresáři?
- Je to funkce MEX, DLL, P nebo M ve vyhledávací cestě?

5. Když je MATLAB spuštěn, schová si jména a umístění všech M-souborů nacházejících se v podadresáři toolbox a všech jeho podadresářích. Toto slouží k rychlému nalezení a spuštění M-souborů a urychluje to také příkaz **lookfor**.

6. Když je ke schovaným jménům a umístěním přidán nový M-soubor, najde ho MATLAB pouze při znovunačtení schovaných jmen a umístění příkazem **path(path)**. Pokud je schovaný soubor modifikován, MATLAB změny rozpozná pouze v případě, že předkompilovaná verze souboru je zpaměti odstraněna příkazem clear (clear $myfunction$, clear functions).

- Je to lokální funkce?
- Je to privátní funkce?
- Je to funkce MEX, DLL, P nebo M v aktuálním adresáři?
- Je to funkce MEX, DLL, P nebo M ve vyhledávací cestě?

5. Když je MATLAB spuštěn, schová si jména a umístění všech M-souborů nacházejících se v podadresáři toolbox a všech jeho podadresářích. Toto slouží k rychlému nalezení a spuštění M-souborů a urychluje to také příkaz **lookfor**.

6. Když je ke schovaným jménům a umístěním přidán nový M-soubor, najde ho MATLAB pouze při znovunačtení schovaných jmen a umístění příkazem **path(path)**. Pokud je schovaný soubor modifikován, MATLAB změny rozpozná pouze v případě, že předkompilovaná verze souboru je zpaměti odstraněna příkazem clear (clear $myfunction$, clear functions).

7. MATLAB sleduje časy modifikací pouze u M-souborů mimo podadresář toolbox. V tomto případě se porovnává čas předkompilované verze v paměti s časem zdrojového M-souboru a v případě nutnosti je provedena nová kompilace.

Tvorba vlastních TOOLBOXů

Je zvykem dávat skupinu M-souborů do jednoho podadresáře na vyhledávací cestě MATLABu. Pokud je vývoj M-souborů ukončen, je vhodné umístit tento podadresář pod adresář **toolbox** a doplnit ho o následující dva soubory:

Tvorba vlastních TOOLBOXů

Je zvykem dávat skupinu M-souborů do jednoho podadresáře na vyhledávací cestě MATLABu. Pokud je vývoj M-souborů ukončen, je vhodné umístit tento podadresář pod adresář **toolbox** a doplnit ho o následující dva soubory:

1. Skriptový soubor **Readme.m**, který obsahuje komentářové řádky, které popisují poslední změny nebo nedokumentované vlastnosti. Příkaz **whatsnew** ***MyToolbox*** zobrazí obsah tohoto souboru.

Tvorba vlastních TOOLBOXů

Je zvykem dávat skupinu M-souborů do jednoho podadresáře na vyhledávací cestě MATLABu. Pokud je vývoj M-souborů ukončen, je vhodné umístit tento podadresář pod adresář **toolbox** a doplnit ho o následující dva soubory:

1. Skriptový soubor **Readme.m**, který obsahuje komentářové řádky, které popisují poslední změny nebo nedokumentované vlastnosti. Příkaz **whatsnew** $\boldsymbol{MyToolbox}$ zobrazí obsah tohoto souboru.

2. Skriptový soubor **Contents.m** obsahuje komentářové řádky, které vypisují všechny M-soubory v toolboxu. Příkaz **help** $\boldsymbol{MyToolbox}$ nebo **helpwin** $\boldsymbol{MyToolbox}$ zobrazí obsah tohoto souboru. První řádek souboru **Contents.m** by měl specifikovat jméno toolboxu a druhý řádek by měl uvádět verzi a datum vzniku toolboxu. Tyto dva řádky používá příkaz **ver**.

Dualita příkazů a funkcí

Příkazy jsou velice podobné funkcím.
Ve skutečnosti mezi nimi existují pouze dva rozdíly:

1. Příkazy nemají výstupní argumenty.

2. Vstupní argumenty příkazů nejsou zahrnuty v apostrofech.

Příkaz	**Funkce**
format short g	format('short','g')
save x y z	save('x','y','z')
hold on	hold('on')
axis square equal	axis('square','equal')
which $fname$	which('$fname$')

Libovolná funkce může být volána jako příkaz, pokud nemá žádný výstupní argument a požaduje pouze řetězcové vstupy.

IN-LINE funkce a funkce FEVAL

Mnoho funkcí matematické analýzy požaduje jako vstupní argument *jméno funkce*, kterou má vyhodnotit. *Jméno funkce* může představovat:

1. jméno funkčního M-souboru

   Např. a=feval('*myfunction*', x) je ekvivalentní s a=*myfunction*(x).

   feval je efektivnější než **eval**.

   eval vyvolává kompletní interpreter MATLABu a nepodporuje více vstupních a výstupních argumentů.

2. jméno řetězcové proměnné popisující celou funkci

   Např. *myfun*=`'100*(y-x^2)^2+(1-x)^2';`

   V tomto případě lze použít funkci **eval**.

   Funkci **feval** nelze použít. Tento nedostatek odstraňuje zavedení funkce **inline**.

inline, **fcnchk**, **argnames**, **formula**

Odlaďovací prostředky

Odlaďování M-souborů

Ve výrazech MATLABu můžeme mít dva druhy chyb:

1. syntaktické chyby,

2. chyby za běhu programu (run-time error).

Syntaktické chyby jsou nalezeny při vyhodnocení výrazu nebo při kompilaci funkce do paměti (výjimkou jsou ‘callback‘-řetězce v GUI).

Chyby vzniklé za běhu programu se odhalují podstatně obtížněji.

Postupy při odlaďování jednoduchých problémů:

Postupy při odlaďování jednoduchých problémů:

1. odstranit středníky,

Postupy při odlaďování jednoduchých problémů:

1. odstranit středníky,

2. přidat příkazy **disp**,

Postupy při odlaďování jednoduchých problémů:

1. odstranit středníky,
2. přidat příkazy **disp**,
3. použít příkaz **keyboard**,

Postupy při odlaďování jednoduchých problémů:

1. odstranit středníky,
2. přidat příkazy **disp**,
3. použít příkaz **keyboard**,
4. změnit funkční M-soubor na skript.

Postupy při odlaďování jednoduchých problémů:

1. odstranit středníky,

2. přidat příkazy **disp**,

3. použít příkaz **keyboard**,

4. změnit funkční M-soubor na skript.

Při složitějších problémech (velké, rekurzivní nebo hluboce vnořené M-soubory) je vhodné použít odlaďovací prostředky MATLABu:

Odlaďovací příkaz	Popis
dbstop in *mfile* **dbstop in** *mfile* **at** *lineno*	Nastav zarážku v souboru*mfile* (na řádku *lineno*)
dbstop if warning **dbstop if error** **dbstop if naninf** **dbstop if infnan**	Zastav při libovolném varování, chybě nebo výskytu NaN nebo Inf
dbclear all **dbclear all in** *filename* **dbclear in** *filename* **dbclear if warning** **dbclear if error** **dbclear if infnan** (nebo **naninf**)	Odstraň zarážku(y)
dbstatus **dbstatus** *filename*	Vypiš všechny zarážky (v souboru *filename*)
dbtype *mfile* **dbtype** *mfile m:n*	Vypiš soubor *mfile* s očíslovanými řádky (mezi řádky m a n)
dbstep **dbstep** n	Spusť jednu řádku (nebo n řádek) a zastav se.

dbcont	Pokračuj.
dbstack	Vypiš, kdo volá koho.
dbup	Přesuň se o jednu úroveň pracovního prostoru nahoru.
dbdown	Přesuň se o jednu úroveň pracovního prostoru dolů.
dbquit	Přeruš odlaďovací mód.

Grafický odlaďovač

Analýza výkonu M-souborů

profile

Analýza	**Popis**
profile *function*	Analyzuj funkci *function*.
profile report	Zobraz zprávu o analýze.
profile plot	Vykresli zprávu o analýze.
profile reset	Resetuj čas.
profile off	Znemožni analýzu.
profile on	Umožni analýzu.
profile done	Ukonči analýzu.

Grafika

Grafický systém MATLABu je vybudován na základě sady grafických objektů (**line**, **surface**, . . .), jejichž vzhled lze řídit nastavením parametrů jejich vlastností.

- funkce vyšší úrovně,
- funkce pro práci s grafickými objekty.

2-D grafika

Funkce PLOT

Kreslení matic:

Funkce PLOT

Kreslení matic:

plot(MATICE) - co sloupec, to průběh

Funkce PLOT

Kreslení matic:

plot(MATICE) - co sloupec, to průběh

plot(vektor,MATICE) - kreslí řádky nebo sloupce **MATICE** vzhledem k **vektoru** a pro každou čáru použije jinou barvu nebo jiný typ čáry. O tom, zda budou kresleny řádky nebo sloupce rozhoduje počet prvků **vektoru**. Řádková nebo sloupcová orientace je vybrána podle shody počtu prvků řádků nebo sloupců **MATICE** s počtem prvků **vektoru**. Pokud je **MATICE** čtvercová, jsou vykresleny její sloupce.

Funkce PLOT

Kreslení matic:

plot(MATICE) - co sloupec, to průběh

plot(vektor,MATICE) - kreslí řádky nebo sloupce **MATICE** vzhledem k **vektoru** a pro každou čáru použije jinou barvu nebo jiný typ čáry. O tom, zda budou kresleny řádky nebo sloupce rozhoduje počet prvků **vektoru**. Řádková nebo sloupcová orientace je vybrána podle shody počtu prvků řádků nebo sloupců **MATICE** s počtem prvků **vektoru**. Pokud je **MATICE** čtvercová, jsou vykresleny její sloupce.

plot(MATICE,vektor) - kreslí každý řádek nebo sloupec **MATICE** vzhledem k **vektoru**

Funkce PLOT

Kreslení matic:

plot(MATICE) - co sloupec, to průběh

plot(vektor,MATICE) - kreslí řádky nebo sloupce **MATICE** vzhledem k **vektoru** a pro každou čáru použije jinou barvu nebo jiný typ čáry. O tom, zda budou kresleny řádky nebo sloupce rozhoduje počet prvků **vektoru**. Řádková nebo sloupcová orientace je vybrána podle shody počtu prvků řádků nebo sloupců **MATICE** s počtem prvků **vektoru**. Pokud je **MATICE** čtvercová, jsou vykresleny její sloupce.

plot(MATICE,vektor) - kreslí každý řádek nebo sloupec **MATICE** vzhledem k **vektoru**

plot(MATICE_X,MATICE_Y) - zobrazí sloupce **MATICE_X** vůči sloupcům **MATICE_Y**

Funkce PLOT

Kreslení matic:

plot(MATICE) - co sloupec, to průběh

plot(vektor,MATICE) - kreslí řádky nebo sloupce **MATICE** vzhledem k **vektoru** a pro každou čáru použije jinou barvu nebo jiný typ čáry. O tom, zda budou kresleny řádky nebo sloupce rozhoduje počet prvků **vektoru**. Řádková nebo sloupcová orientace je vybrána podle shody počtu prvků řádků nebo sloupců **MATICE** s počtem prvků **vektoru**. Pokud je **MATICE** čtvercová, jsou vykresleny její sloupce.

plot(MATICE,vektor) - kreslí každý řádek nebo sloupec **MATICE** vzhledem k **vektoru**

plot(MATICE_X,MATICE_Y) - zobrazí sloupce **MATICE_X** vůči sloupcům **MATICE_Y**

Samozřejmě lze též použít funkci **plot** s několika dvojicemi maticových argumentů

plot(X1,Y1,X2,Y2, ...)

V každé dvojici musí být matice stejného typu, rozdílné dvojice mohou mít různou dimenzi.

Imaginární a komplexní data

Jsou-li argumenty funkce plot komplexní, tj. mají nenulové imaginární části, jsou tyto imaginární části ignorovány.

Pouze v případě, kdy argument funkce **plot** je jediný, tj. **plot(Z)**, kde **Z** je komplexní vektor nebo matice, je vykreslena závislost imaginárních částí prvků **Z** vzhledem k reálným částem.

Styly a barvy čar

symbol	barva	symbol	značka	symbol	typ čáry
b	modrá	.	bod	-	plná
g	zelená	o	kroužek	:	tečkovaná
r	červená	x	krát	-.	čerchovaná
c	tyrkysová	+	plus	--	čárkovaná
m	fialová	*	hvězdička		
y	žlutá	s	čtvereček		
k	černá	d	kosočtverec		
w	bílá	v	trojúhelník		
		^	trojúhelník		
		<	trojúhelník		
		>	trojúhelník		
		p	pentagram		
		h	hexagram		

Vykreslovací styly

Příkazem **colordef** se definuje:

- barva pozadí grafického okna,
- barva pozadí os,
- barva popisů os,
- použitá mapa barev,
- několik prvních barev pro čáry vykreslené příkazem **plot**.

colordef white
colordef black
colordef none

Mřížky a popisy

grid	čáry sítě
box	ohraničení os
title	nadpis grafu (doprostřed nad graf)
xlabel	popis x-ové osy (doprostřed pod osu)
ylabel	popis y-ové osy (doprostřed podél osy)
text	textový řetězec na určenou pozici
gtext	umístí text do grafu na místo vybrané myší

Osy

Příkaz	Popis
axis([xmin xmax ymin ymax])	Nastaví meze aktuálních os.
v = axis	Vrátí meze aktuálních os.
axis auto	Nastaví implicitní meze os.
axis manual	Zmrazí meze os.
axis xy	Používá kartézský souřadný systém, tj. počátek je vlevo dole.
axis ij	Používá maticový souřadný systém, tj. počátek je vlevo nahoře.
axis square	Nastaví čtvercové osy.
axis equal	Nastaví na obou osách stejná měřítka.
axis tight	Nastaví meze podle rozsahu dat.
axis vis3d	Zamezí změně proporcí os při změně pohledu.
axis normal	Vypne předchozí čtyři nastavení.
axis off	Vypne zobrazení os.
axis on	Zapne zobrazení os.

Vykreslování více průběhů

hold

ishold

Vícenásobná grafická okna

figure

close, close all

clf, clf reset

Příkaz SUBPLOT

subplot(m,n,p)

Interaktivní kreslicí nástroje

legend

zoom

ginput

gtext

Vždy můžete používat pouze jeden z těchto příkazů!!!

Překreslování obrazovky

Pouze následujících pět událostí přinutí MATLAB překreslit obsah grafického okna:

Překreslování obrazovky

Pouze následujících pět událostí přinutí MATLAB překreslit obsah grafického okna:

1. návrat do příkazového režimu,

Překreslování obrazovky

Pouze následujících pět událostí přinutí MATLAB překreslit obsah grafického okna:

1. návrat do příkazového režimu,

2. spuštění funkce, která dočasně přeruší běh programu, např. **pause**, **keyboard**, **input**, **waitforbuttonpress**,

Překreslování obrazovky

Pouze následujících pět událostí přinutí MATLAB překreslit obsah grafického okna:

1. návrat do příkazového režimu,

2. spuštění funkce, která dočasně přeruší běh programu, např. **pause**, **keyboard**, **input**, **waitforbuttonpress**,

3. spuštění příkazu **getframe**,

Překreslování obrazovky

Pouze následujících pět událostí přinutí MATLAB překreslit obsah grafického okna:

1. návrat do příkazového režimu,

2. spuštění funkce, která dočasně přeruší běh programu, např. **pause**, **keyboard**, **input**, **waitforbuttonpress**,

3. spuštění příkazu **getframe**,

4. spuštění příkazu **drawnow**,

Překreslování obrazovky

Pouze následujících pět událostí přinutí MATLAB překreslit obsah grafického okna:

1. návrat do příkazového režimu,
2. spuštění funkce, která dočasně přeruší běh programu, např. **pause**, **keyboard**, **input**, **waitforbuttonpress**,
3. spuštění příkazu **getframe**,
4. spuštění příkazu **drawnow**,
5. změna velikosti grafického okna.

Specializované 2-D grafy

loglog	graf s logaritmickou stupnicí na obou osách
semilogx	graf slog. stupnicí na ose x a lin. stupnicí na ose y
semilogy	graf slog. stupnicí na ose y a lin. stupnicí na ose x
comet	vytváří graf pohybujícím se bodem
area	vyplněné grafy
fill	vykreslí mnohoúhelník a vyplní jej
pie	koláčový graf
pareto	Pareto graf
plotyy	graf, který má dvě y-ové osy
bar	sloupcový graf
barh	sloupcový horizontální graf
bar3	3-D sloupcový graf
bar3h	3-D sloupcový horizontální graf
stairs	schodový graf
hist	histogram
stem	graf pro vykreslování diskrétních posloupností
errorbar	graf chyb

polar	graf v polárních souřadnicích
compass	graf vektorů ve formě šipek vycházejících zpočátku
feather	graf vektorů vycházejících z ekvidistantně rozložených bodů podél horizontální osy
plotmatrix	graf rozptylu
quiver	graf vektorového pole
rose	úhlový histogram

Vykreslování matematických funkcí

fplot

ezplot

Formátování textu

\alpha	α	\upsilon	υ	\Re	$\Re$	\neg	$\neq$
\beta	β	\omega	ω	\Im	$\Im$	\approx	$\approx$
\chi	χ	\xi	ξ	\aleph	$\aleph$	\equiv	$\equiv$
\delta	δ	\psi	ψ	\wp	$\wp$	\cong	$\cong$
\epsilon	ε	\zeta	ζ	\otimes	$\otimes$	\pm	$\pm$
\theta	ϕ	\Delta	Δ	\oplus	$\oplus$	\propto	$\propto$
\gamma	γ	\Phi	Φ	\oslash	$\emptyset$	\partial	∂
\eta	η	\Gamma	Γ	\cap	$\cap$	\bullet	$\bullet$
\iota	ι	\Lambda	Λ	\cup	$\cup$	\circ	$\circ$
\phi	φ	\Pi	Π	\supset	$\supset$	\div	$\div$
\kappa	κ	\Theta	Θ	\supseteq	$\supseteq$	\infty	∞
\lambda	λ	\Sigma	Σ	\subset	$\subset$	\leftarrow	$\leftarrow$
\mu	μ	\Upsilon	Υ	\subseteq	$\subseteq$	\rightarrow	$\rightarrow$
\nu	ν	\Omega	Ω	\ni	$\ni$	\leftrightarrow	$\leftrightarrow$
\o	o	\Xi	Ξ	\in	$\in$	\uparrow	$\uparrow$
\pi	π	\Psi	Ψ	\o	$\emptyset$	\downarrow	$\downarrow$
\theta	θ	\varsigma	ς	\int	$\int$	\clubsuit	$\clubsuit$

\rho	ρ	\vartheta	υ	\sim	$\sim$	\diamondsuit	$\diamondsuit$
\sigma	σ	\forall	$\forall$	\leq	$\leq$	\heartsuit	$\heartsuit$
\tau	τ	\exist	$\exists$	\geq	$\geq$	\spadesuit	$\spadesuit$
\prime	$\prime$	\neg	$\neg$	\cdot	$\cdot$	\varpi	ϖ
\Rightarrow	$\Rightarrow$	\Uparrow	$\Uparrow$	\surd	$\surd$	\angle	$\angle$
\Leftarrow	$\Leftarrow$	\langle	$\langle$	\rangle	$\rangle$	\ldots	$\ldots$
\Leftrightarrow	$\Leftrightarrow$	\nabla	∇	\times	$\times$	\mid	$\mid$
\Downarrow	$\Downarrow$						

Omezená podmnožina formátovacích příkazů TeXu:

`\bt` – tučné písmo,

`\it` – kurzíva,

`\sl` – šikmé písmo,

`\rm` – normální písmo,

`_` – dolní index,

`^` – horní index.

\fontsize{n} - užij písmo o velikosti n bodů

\fontname{$fontname$} - užij písmo $fontname$

Formátovací příkazy mají platnost do konce řetězce nebo se týkají obsahu definovaného uvnitř složených závorek { }.

3-D grafika

Vykreslování čar

plot3	čáry a body v prostoru
zlabel	popis z-ové osy (doprostřed podél osy)

Skalární funkce dvou proměnných

meshgrid

Zobrazování sítí

mesh	drátový model 3-D plochy
meshc	kombinace funkcí mesh a contour
meshz	drátový model 3-D plochy včetně nulové roviny
waterfall	graf tvaru vodopádu
hidden	režim zobrazování skrytých čar

Zobrazování ploch

surf	stínovaná 3-D plocha
surfc	kombinace funkcí surf a contour
surfl	stínovaná 3-D plocha s osvětlením
surfnorm	výpočet a zobrazení normál plochy
shading	nastavení vlastnosti pro stínování

Zobrazování sítí a ploch z neuspořádaných dat

trimesh	zobrazí trojúhelníky jako síť
trisurf	zobrazí trojúhelníky jako plochu
delaunay	triangulace
voronoi	Voronoiův graf

Změna bodu pohledu

Příkaz	Popis
view(az,el) view([az,el])	Nastav azimut (**az**) a elevaci (**el**) pohledu.
view([x,y,z])	Nastav pohled v kartézských souřadnicích **x**, **y** a **z**.
view(2)	Nastav implicitní 2-D pohled, **az=0**, **el=90**.
view(3)	Nastav implicitní 3-D pohled, **az=-37.5**, **el=30**.
[az,el]=view	Vrať aktuální azimut (**az**) a elevaci (**el**).
view(T)	K nastavení pohledu použij transformační matici **T**.
T=view	Vrať aktuální transformační matici.

rotate3d

Vykreslování vrstevnic

contour	vrstevnicový 2-D graf
contour3	vrstevnicový 3-D graf
contourf	vyplněný vrstevnicový 2-D graf
clabel	popis vrstevnic
pcolor	pseudo barevná plocha

Specializované 3-D grafy

ribbon	proužkový graf
quiver	2-D graf vektorového pole
quiver3	3-D graf vektorového pole
fill3	vyplněný 3-D mnohoúhelník
stem3	3-D graf pro vykreslování diskrétních posloupností
comet3	vytváří graf pohybujícím se bodem ve 3-D
slice	objemová vizualizace pomocí řezů

Použití barvy a světla

Vykreslovací styly

colordef

Mapy barev

Intenzita:			Barva
červené	zelené	modré	
1	0	0	červená
0	1	0	zelená
0	0	1	modrá
1	1	0	žlutá
1	0	1	fialová
0	1	1	azurová
0	0	0	černá
1	1	1	bílá
0	0,5	0	tmavo zelená
0,67	0	1	fialkově modrá
1	0,5	0	oranžová
0,5	0	0	tmavo červená
0,5	0,5	0,5	středně šedá

Standardní mapy barev	Popis
hsv	Mapa barev hsv (Hue-saturation-value)
jet	Varianta mapy barev hsv (jako v kartografii)
hot	Černo-červeno-žluto-bílá mapa barev
cool	Mapa barev s odstíny tyrkysové a fialové
white	Bílá mapa barev
gray	Lineární šedá mapa barev
bone	Šedá mapa barev se zabarvením do modra
pink	Mapa barev s pastelovými odstíny růžové
copper	Mapa barev s lineárními tóny mědi
prism	
flag	Mapa barev tvořená střídavě červenou, bílou, modrou a černou
lines	
colorcube	

Použití mapy barev

colormap

Zobrazení mapy barev

rgbplot

colorbar

Vytvoření a změna mapy barev

colormap

brighten

rgb2hsv, **hsv2rgb**

caxis - ovládání barevné osy, **caxis([cmin cmax])**

Funkci **caxis** můžeme použít i k dosažení následujících dvou efektů:

Vytvoření a změna mapy barev

colormap

brighten

rgb2hsv, **hsv2rgb**

caxis - ovládání barevné osy, **caxis([cmin cmax])**

Funkci **caxis** můžeme použít i k dosažení následujících dvou efektů:

- Nastavíme-li **cmin**, popř. **cmax** na hodnoty, které jsou menší než rozsah dat plochy, potom data menší než **cmin** a větší než **cmax** se budou transformovat do krajních hodnot mapy barev, tj. **cmin**, resp. **cmax**.

Vytvoření a změna mapy barev

colormap

brighten

rgb2hsv, **hsv2rgb**

caxis - ovládání barevné osy, **caxis([cmin cmax])**

Funkci **caxis** můžeme použít i k dosažení následujících dvou efektů:

- Nastavíme-li **cmin**, popř. **cmax** na hodnoty, které jsou menší než rozsah dat plochy, potom data menší než **cmin** a větší než **cmax** se budou transformovat do krajních hodnot mapy barev, tj. **cmin**, resp. **cmax**.

- Nastavíme-li **cmin**, popř. **cmax** na hodnoty, které jsou větší než rozsah dat plochy, MATLAB transformuje mapu barev do většího rozsahu, jako kdyby data byla rozprostřena od **cmin** až do **cmax**. V důsledku toho jsou aktuální data zobrazena použitím pouze části mapy barev.

Použití více než jedné mapy barev

spinmap

Použití barvy k popisu čtvrté dimenze

surf(X,Y,Z,?)

Osvětlovací modely

Odrazivost je tvořena řadou komponent:	
ambient light	příspěvek okolního světla
diffuse reflection	příspěvek rozptýleného (difúzního) odrazu
specular reflection	příspěvek zrcadlového odrazu
specular exponent	koeficient řídící velikost zrcadlové plošky
specular color reflectance	příspěvek povrchové barvy

light

lighting

material

diffuse, **specular**

surfl

Obrázky, animace a zvuk

Obrázky

Existují tři typy obrazových dat:

Obrázky

Existují tři typy obrazových dat:

1. Indexovaný obraz
   vyžaduje mapu barev a interpretuje obrazová data jako indexy do mapy barev.

Obrázky

Existují tři typy obrazových dat:

1. Indexovaný obraz
   vyžaduje mapu barev a interpretuje obrazová data jako indexy do mapy barev.

2. Intenzitní obraz
   měřítkuje obrazová data do rozsahu intenzit.

Obrázky

Existují tři typy obrazových dat:

1. Indexovaný obraz
   vyžaduje mapu barev a interpretuje obrazová data jako indexy do mapy barev.

2. Intenzitní obraz
   měřítkuje obrazová data do rozsahu intenzit.

3. RGB obraz
   je tvořen třístránkovým polem (R, G, B).

Obrázky

Existují tři typy obrazových dat:

1. Indexovaný obraz
   vyžaduje mapu barev a interpretuje obrazová data jako indexy do mapy barev.

2. Intenzitní obraz
   měřítkuje obrazová data do rozsahu intenzit.

3. RGB obraz
   je tvořen třístránkovým polem (R, G, B).

image zobrazí matici jako obraz, každý prvek matice určuje barvu políčka v obraze. Prvky matice jsou užity jako indexy aktuální mapy barev k určení barvy.

Porovnání objektů **image** a grafů, kreslených funkcí **pcolor**:

- **image** slouží pro zobrazování fotografií, obrazů apod.

- **pcolor** slouží pro zobrazování abstraktních matematických objektů

Obrazové formáty a soubory

double, **uint8**

bílá barva = [1 1 1] ve formátu double nebo [255 255 255] ve formátu uint8

Převody obrazových formátů	
Indexovaný obraz	Xdouble=double(Xuint8)+1 Xuint8=uint8(Xdouble-1)
Intenzitní nebo RGB obraz	Xdouble=double(Xuint8)/255 Xuint8=uint8(round(Xdouble*255))

MATLABem podporované průmyslové standardy:

Přípona	**Format**	**Popis**
bmp	BMP	MS Windows Bitmap Format
hdf	HDF	Hierarchical Data Format
jpg (jpeg)	JPEG	Joint Photographic Experts Group Format
pcx	PCX	PC Paintbrush Format
tif nebo **tiff**	TIFF	Tagged Image File Format
xwd	XWD	X Window Dump Format

Následující grafické formáty lze číst funkcí **imread**:

Formát	Popis
BMP	1-, 4-, 8- a 24-bitové nekomprimované obrázky a 4- a 8-bitové RLE-komprimované obrázky
HDF	8-bitové obrázky s nebo bez přidružené mapy barev a 24-bitové obrázky
JPEG	Libovolné obrázky dle standardu JPEG. Podporovaná jsou také běžná rozšíření.
PCX	1-, 8- a 24-bitové obrázky.
TIFF	Libovolné 1-, 8- a 24-bitové obrázky dle standardu TIFF s nebo bez komprese.
XWD	XY bitmapa, 1-bitová XY pixmapa a 1- a 8-bitová Zpixmapa.

Následující grafické formáty lze ukládat funkcí **imwrite**:

Formát	Popis
BMP	8-bitové nekomprimované obrázky s přidruženou mapou barev a 24-bitové nekomprimované obrázky
HDF	8-bitové obrázky s nebo bez přidružené mapy barev a 24-bitové obrázky
JPEG	Standardní JPEG obrázky.
PCX	8-bitové obrázky.
TIFF	Libovolné obrázky dle standardu TIFF zahrnující 1-, 8- a 24-bitové obrázky s nebo bez komprese.
XWD	8-bitová Zpixmapa.

imwrite akceptuje následující volitelné parametry:

Formát	**Parametr**	**Přípustná hodnota**	**Implicitní hodnota**
JPEG	'Quality'	0 až 100	75
TIFF	'Compression'	'none', 'packbits', 'ccitt' (jen pro binární obrázky)	'ccitt' pro binární jinak 'packbits'
TIFF	'Description'	Libovolný řetězec.	Prázdný řetězec
TIFF	'Resolution'	Skalární hodnota	72
HDF	'Compression'	'none', 'rle' (jen pro šedé a indexované), 'jpeg' (jen pro šedé a RGB)	'none'
HDF	'WriteMode'	'overwrite' nebo 'append'	'overwrite'
HDF	'Quality'	0 až 100; jen pro 'jpeg' kompresi	75

Animace

moviein

getframe

movie

im2frame, frame2im

capture

Zvuk

sound, soundsc

wavwrite, wavread

Tisk a export grafiky

Tisk zmenu

File-Print Setup

File-Print, **printdlg**

Umístění a velikost grafiky

File-Page Setup, **pagedlg**

Tisk z příkazové řádky

orient

print -d*device* -*option filename*

Volba ovladače

- vestavěné ovladače (PostScript, HPGL, AI88, TIFF, JPEG, mfile),
- dodatečné ovladače (Ghostscript, -dwin, -dwinc, -dmeta, -dbitmap).

Zařízení (MATLAB)	**Popis**
-dps	Černobílý PostScript.
-dps2	Černobílý Level 2 PostScript.
-deps	Černobílý zapouzdřený PostScript.
-deps2	Černobílý zapouzdřený Level 2 PostScript.
-dpsc	Barevný PostScript.
-dpsc2	Barevný Level 2 PostScript.
-depsc	Barevný zapouzdřený PostScript.
-depsc2	Barevný zapouzdřený Level 2 PostScript.
-dhpgl	HPGL kompatibilní s HP plotrem 7475A.
-dill	Adobe Illustrator 88.

-dmfile	M-soubor a MAT-soubor obsahující příkazy k obnovení grafického okna a jeho dětí.
-djpeg[nn**]**	JPEG obraz, *nn* – kvalita.
-dtiff	TIFF obraz s kompresí.
-dtiffnocompression	TIFF obraz bez komprese.

Zařízení (GhostScript)	**Popis**
-dlaserjet	HP LaserJet
-dljetplus	HP LaserJet+
-dljet2p	HP LaserJet IIP
-dljet3	HP LaserJet III
-ddeskjet	HP DeskJet a DeskJet Plus
-ddjet500	HP Deskjet 500
-dcdjmono	HP DeskJet 500C černobíle
-dcdjcolor	HP DeskJet 500C barevně
-dcdj500	HP DeskJet 500C
-dcdj550	HP Deskjet 550C

-dpaintjet	HP PaintJet barevně
-dpjxl	HP PaintJet XL barevně
-dpjetxl	HP PaintJet XL barevně
-dpjxl300	HP PaintJet XL300 barevně
-ddnj650c	HP DesignJet 650C
-dbj10e	Canon BubbleJet BJ10e
-dbj200	Canon BubbleJet BJ200
-dbjc600	Canon Color BubbleJet BJC-600 a BJC-4000
-dln03	DEC LN03
-depson	Epson-kompatibilní jehličkové tiskárny (9 nebo 24)
-depsonc	Epson LQ-2550 a Fujitsu 3400/2400/1200
-deps9high	Epson-kompatibilní 9-jehličkové, prokládané řádky
-dibmpro	IBM 9-jehličkový Proprinter
-dbmp256	8-bitový (256-barev) BMP formát
-dbmp16m	24-bitový BMP formát
-dpcxmono	Černobílý PCX formát
-dpcx16	Starší barevný PCX formát (EGA/VGA, 16-barev)
-dpcx256	Novější barevný PCX formát (256-barev)

-dpcx24b	24-bitový barevný PCX formát, tři 8-bitové roviny
-dpbm	Portable Bitmap (plain format)
-dpbmraw	Portable Bitmap (raw format)
-dpgm	Portable Graymap (plain format)
-dpgmraw	Portable Graymap (raw format)
-dppm	Portable Pixmap (plain format)
-dppmraw	Portable Pixmap (raw format)

Zařízení (MSWindows)	**Popis**
-dwin	Aktuální tiskárna, černobílý tisk
-dwinc	Aktuální tiskárna, barevný tisk
-dmeta	schránka, Enhanced Metafile formát
-dbitmap	schránka, Bitmap formát
-dsetup	Vyvolej dialog k nastavení tiskárny.

Ostatní tiskové volby

Přepínač	Popis
-fn	Určí grafické okno k tisku.
-P$printer$	Určí tiskárnu (pouze Unix).
-append	Přidá k existujícímu souboru.
-noui	Potlačení tisku objektů uicontrol.
-painters	Použij malířův algoritmus.
-zbuffer	Použij Z-bufer.
-rdpi	Určí rozlišení v bodech na palec.
-epsi	Zahrň do EPS černobílý náhled EPSI.
-tiff	Zahrň do EPS náhled ve formátu TIFF.
-loose	Zahrň do EPS ohraničení.
-cmyk	Použij barvy CMYK místo RGB.
-adobeset	Použij implicitní kódování znakové sady.
-v	Zobraz dialog Print (normálně je potlačen).

Změna implicitního nastavení

printopt

Implicitní nastavení lze změnit editací souboru MATLABROOT\toolbox\local\printopt.m.

Exportování obrázků

capture

imwrite

schránka - Copy, Paste

Objektová grafika

Handle Graphics je jméno souboru nízkoúrovňových grafických funkcí v MATLABu.

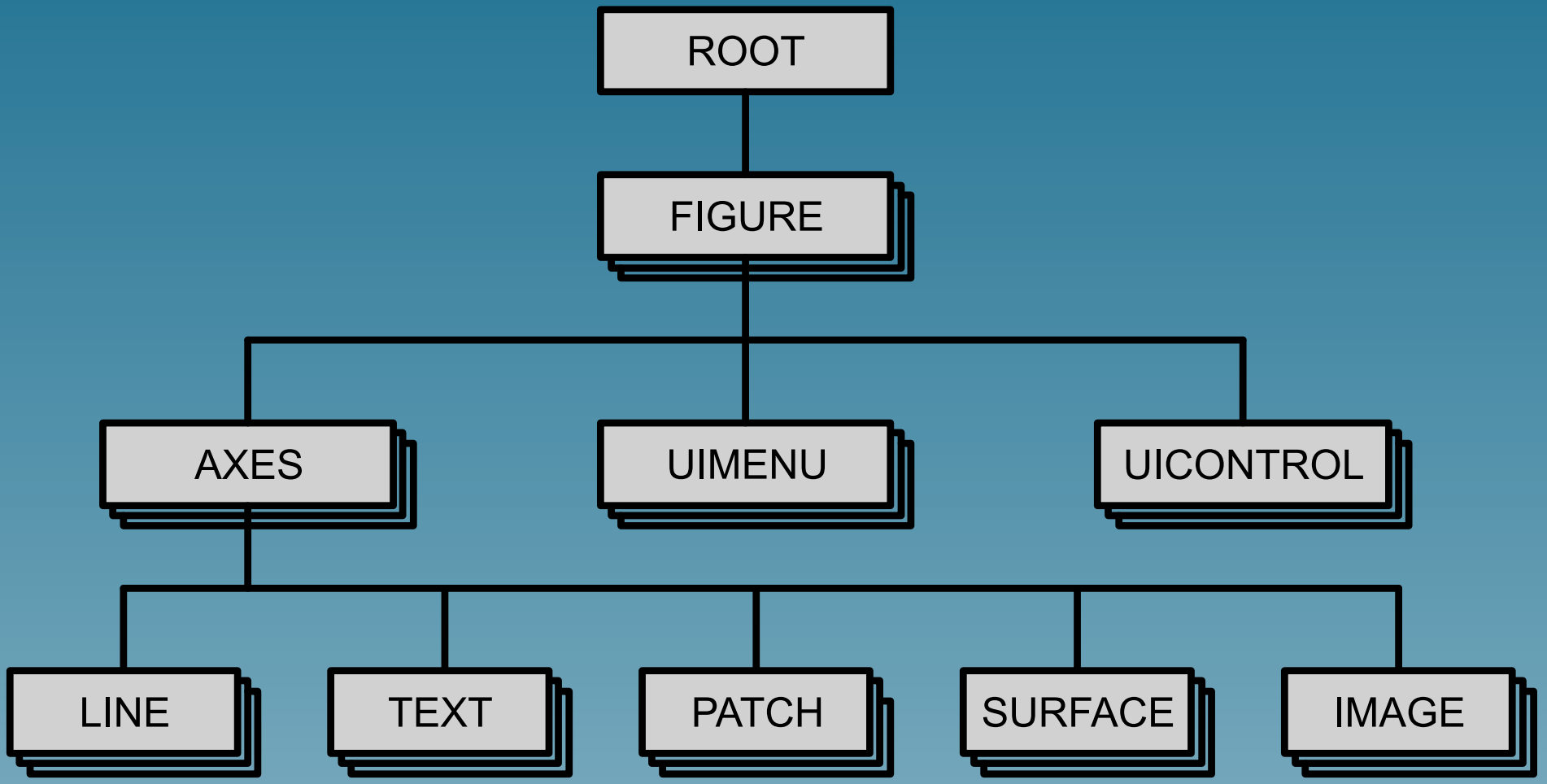
Kdo potřebuje Handle Graphics?

- Uživatel, kterému již nestačí základní grafické funkce MATLABu.

- Uživatel, který chce např.:
  - změnit font v titulku grafu,
  - změnit barvu pozadí grafického okna,
  - změnit sílu čáry u všech průběhů v grafu.

Objekty

MATLAB definuje grafické objekty jako základní grafické jednotky svého grafického systému a organizuje je do stromově strukturované hierarchie. Tyto objekty zahrnují:



Identifikátory objektů

Každý samostatný grafický objekt má svůj vlastní identifikátor, tzv. **handle**, který je tomuto objektu přiřazen při jeho vytvoření. Některé grafy, např. vrstevnice, jsou složeny z několika objektů a každý z nich má svůj vlastní identifikátor, tj. každá vrstevnice má svůj identifikátor.

Identifikátor objektu **root** je vždy nulový.

Identifikátor objektu **figure** je (obvykle) celé kladné číslo, které je implicitně zobrazeno v názvu grafického okna.

Identifikátory ostatních objektů jsou reálná čísla, která obsahují informace používané MATLABem.

Všechny funkce MATLABu, které vytvářejí objekty, vrací identifikátory (nebo vektor identifikátorů) vytvořených objektů. A to jak funkce vyšší úrovně jako **surf** (generuje jak plochu, tak čáry), tak i funkce nižší úrovně, které generují pouze jeden objekt, např. funkce **surface**.

Funkce vytvářející objekty	
root	Prapředek všech objektů, obrazovka.
figure	Okno, do kterého se vykresluje grafika.
uimenu	Programovatelná nabídka pro grafické okno.
uicontrol	Programovatelné prvky GUI, např. tlačítko, apod.
uicontextmenu	Programovatelná kontextová nabídka.
axes	Obdélníková oblast v grafickém okně (figure).
line	Čára spojující datové body.
text	Znakový řetězec.
patch	Plocha definovaná spojenými čárami.
surface	Plocha definovaná spojenými ploškami.
light	Světelný zdroj (působí na objekty patch a surface).
image	2-D obraz definovaný jednotlivými barevnými body.
rectangle	Obdélník, ovál, elipsa, kružnice.

K jednoduchému přístupu k identifikátorům objektů definuje MATLAB následující funkce :

gcf - vrací identifikátor aktuálního objektu figure,

gca - vrací identifikátor aktuálního objektu axes.

Tyto funkce můžeme použít jako vstupní argumenty pro jiné funkce, které požadují identifikátor objektů figure nebo axes.

Vlastnosti objektů

Všechny objekty mají vlastnosti, které rozhodují o tom, jak budou tyto objekty zobrazeny. Tyto vlastnosti zahrnují jak obecné informace (typ objektu, jeho rodiče a děti, zda je nebo není objekt viditelný), tak i informace jedinečné pro jednotlivý typ objektu (např. rozsah x-ové osy objektu axes nebo data, kterými je definován objekt surface).

Tvořený grafický objekt je inicializován množinou implicitních hodnot vlastností.

Aktuální hodnoty všech vlastností můžeme získat a většinu z nich specifikovat. Některé vlastnosti jsou nastaveny MATLABem a jsou určeny pouze ke čtení. Hodnoty vlastností se aplikují jednoznačně na konkrétní objekt, nastavení hodnoty pro jeden objekt neovlivňuje hodnotu u ostatních objektů téhož typu.

Poznámka o názvech vlastností

Podle zvyklostí dává MATLAB u názvů vlastností objektů vždy první písmeno každého slova velké, např. **LineStyle** nebo **XMinorTickMode**. Tento způsob je vhodný pro jejich lepší čtení. MATLAB nekontroluje v názvech vlastností velikost písmen, proto lze pro správnou identifikaci názvu vlastnosti použít písmena libovolné velikosti. Lze dokonce použít i zkrácených názvů, ale tak, aby tato zkratka jednoznačně určovala danou vlastnost.

POZOR!!! Název vlastnosti uvedený v apostrofech nesmí obsahovat žádné mezery.

Kdy nastavit vlastnosti objektu:

- Určit vlastnost objektu v době, kdy voláme funkci, která příslušný grafický objekt generuje, např. **figure('Color', 'yellow')**.

- Nastavit hodnotu vlastnosti po vytvoření objektu pomocí funkce **set**.

Univerzální funkce get a set

set - nastavuje vybrané vlastnosti zvolených objektů

get - vrací hodnoty vybraných vlastností

copyobj

delete

reset

Hledání objektů

gcf, **gca**

gco - na co naposledy klikla myš

findobj

Zvolení objektů myší

gco - vrací identifikátor objektu, na který se naposledy kliklo myší

Jaké čára se vybere, když se klikne na průsečík dvou čar?
– dáno pořadím hodnot vlastnosti 'Children'

Jak daleko od čáry můžu kliknout, aby se ještě vybrala?
– v okolí 5-ti pixlů

Tisk grafických oken

PaperPosition

PaperPositionMode

PaperUnits

PaperType

PaperSize

PaperOrientation

Implicitní vlastnosti

Všechny vlastnosti objektů mají své implicitní hodnoty vestavěné v MATLABu (factory settings). Navíc ale můžeme definovat své vlastní implicitní hodnoty v libovolném bodu hierarchie objektů.

Hledání implicitních hodnot začíná u aktuálního objektu a pokračuje přes předky do té doby, dokud není nalezena implicitní hodnota definovaná uživatelem nebo dokud není dosaženo vestavěných implicitních hodnot. Proto je hledání implicitních hodnot vždy úspěšné.

Implicitní hodnoty můžeme nastavit pomocí řetězce začínajícího slovem **Default**, za kterým následuje typ objektu a nakonec vlastnost objektu. Např. nastavení implicitní barvy čáry na bílou barvu v úrovni aktuálního objektu figure provede příkaz

```
set(gcf, 'DefaultLineColor', 'w')
```

Bod hierarchie, ve kterém definujeme implicitní hodnotu, určuje, které objekty tuto hodnotu použijí.

Zadáme-li hodnotu **factory**, nastaví se vlastnost na svou hodnotu vestavěnou v MATLABu.

Řetězcem **remove** můžeme odstranit implicitní hodnoty nastavené uživatelem.

Implicitní hodnoty jsou respektovány **pouze** grafickými funkcemi nejnižší úrovně (**figure**, **axes**, **line**, **text**, atd.) !!!

Poznámky:

MATLAB má vestavěn implicitní font písma Helvetica. Pokud tento font není na našem počítači, nebude otočení textu pro **ylabel** provedeno. V tomto případě je vhodné nastavit pro text na úrovni root implicitní název fontu pomocí vlastnosti **DefaultTextFontName**.

Chceme-li mít v MATLABu definované určité hodnoty vždy, je vhodné je definovat v msouboru **startup.m**.

Společné vlastnosti

Vlastnost	Popis
ButtonDownFcn	Funkce, která se spustí při stisku myši.
Children	Identifikátory všech potomků.
Clipping	Umožnění nebo zakázání ořezávání.
CreateFcn	Funkce, která se spustí při vytváření objektu.
ChangeFcn	Funkce, která se spustí po změně objektu.
DeleteFcn	Funkce, která se spustí před zrušením objektu.
BusyAction	Řídí mód přerušení callback-funkcí.
HandleVisibility	Určuje viditelnost identifikátorů objektů.
Interruptible	Určuje, zda je možno přerušit callback-funkci.
Parent	Identifikátor rodiče.
Selected	Indikuje, zda byl objekt vybrán.
SelectionHighlight	Určuje ,zda se objekt po vybrání zvýrazní.
Tag	Uživatelem specifikovaný popis objektu.
Type	Typ objektu.
UserData	Místo pro libovolnou proměnnou.
Visible	Určuje viditelnost objektu.

Poznámka:

Nedokumentovaná vlastnost objektu root **'HideUndocumented'** určuje, zda funkce **get** vrací všechny vlastnosti nebo pouze vlastnosti dokumentované.

Pro zviditelnění všech vlastností použijte příkaz

```
set(0, 'HideUndocumented', 'off');
```

Grafické uživatelské rozhraní

Uživatelské nabídky

uimenu

Umístění menu:

```
set(gcf, 'MenuBar', 'none')
set(gcf, 'MenuBar', 'figure')
```

Vytváření submenu

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox **edit**

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox **edit**
frame

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**
popupmenu	

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**
popupmenu	**pushbutton**

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**
popupmenu	**pushbutton**
radiobutton	

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**
popupmenu	**pushbutton**
radiobutton	**slider**

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**
popupmenu	**pushbutton**
radiobutton	**slider**
text	

Uživatelské řídící prvky

uicontrol

Existuje 10 různých typů řídících prvků:

checkbox	**edit**
frame	**listbox**
popupmenu	**pushbutton**
radiobutton	**slider**
text	**togglebutton**

Callback-ové funkce

Hodnota vlastnosti **'CallBack'** je řetězec MATLABu, který se předá k vyhodnocení funkci **eval**.

Vyhodnocování se provádí v základním pracovním prostoru MATLABu.

Vytváření funkcí grafického uživatelského prostředí

- Vytvořit separátní M-funkci, která bude navržena speciálně ke spouštění jednoho nebo více callbacků.

- **Vytvořit jednu M-funkci, která se bude volat rekurzivně.**

Uchovávání dat v GUI

- Globální proměnné.
- Použití vlastnosti **'UserData'**.

Pomocné funkce

selectmoveresize

```
set(gca, 'ButtonDownFcn', 'selectmoveresize')
set(gca, 'ButtonDownFcn', '')
```

setptr

rotate

refresh

clf, **cla**, **shg**

gcbo, **gcbf**

dragrect, **rbbox**

makemenu

umtoggle

hidegui

edtext

ishandle

uiwait, **uiresume**, **waitfor**

btngroup, **btnstate**, **btnpress**, **btndown**, **btnup**, **btnicon**, **iconsdisp**

setuprop, **getuprop**, **clruprop**

popupstr

textwrap

GUIDE

GUIDE je množina interaktivních nástrojů pro návrh grafického uživatelského prostředí dodávaná s MATLABem.

guide

align

menuedit

propedit

cbedit

Dialogové boxy

Dialogové boxy

Dialogové boxy v MATLABu nejsou objekty Handle Graphics.

dialog

Normální a modální dialogové boxy.

Předdefinované dialogové boxy:

helpdlg

warndlg

errordlg

questdlg

Žádosti

inputdlg

menu

listdlg

uigetfile

uiputtfilen

printdlg

pagedlg

uisetfont

uisetcolor

Pomocné funkce

waitbar

axlimdlg

tabdlg

Numerická lineární algebra

Soustavy lineárních rovnic

Symbol hvězdička ($*$) označuje násobení matic.

Operace je definována, pokud vnitřní rozměry dvou operandů jsou stejné.

Mezi nejběžnější patří vnitřní (skalární) součin.

V MATLABu existují dva symboly pro dělení matic, lomítko (/) a zpětné lomítko (\).

```
x = A\b je řešením A*x = b
x = A/b je řešením x*A = b
```

Maticové funkce

Funkce	Popis
A^n	Mocnina.
balance(A)	
cdf2rdf(A)	Převod komplexní diagonální formy na reálnou.
chol(A)	Choleskyho rozklad.
cholinc(A,DropTol)	Neúplný Choleskyho rozklad.
cond(A)	Číslo podmíněnosti matice.
condest(A)	
condeig(A)	
det(A)	Determinant.
eig(A)	Vektor vlastních čísel.
[V,D]=eig(A)	
[V,D]=eigs(A)	
expm(A)	Maticová exponenciela.
expm1(A)	
expm2(A)	
expm3(A)	

funm(A,'fun')	
hess(A)	Hessenbergova forma.
inv(A)	Inverze matice.
logm(A)	Maticový logaritmus.
lscov(A,b,V)	
lu(A)	LU rozklad.
luinc(A,DropTol)	Neúplný LU rozklad.
nnls(A,b)	
norm(A)	Norma matice a vektoru.
norm(A,1)	
norm(A,2)	
norm(A,inf)	
norm(A,p)	
norm(A,'fro')	
normest(A)	
null(A)	Nulový prostor.
orth(A)	Ortogonalizace.
pinv(A)	Pseudoinverze.

planerot(x)	
poly(A)	Charakteristický polynom.
polyeig(A0,A1,AP)	
polyvalm(A)	Vyhodnocení maticového polynomu.
qr(A)	QR rozklad.
qrdelete(Q,R,j)	Vymaž sloupec zqr rozkladu.
qrinsert(Q,R,j,x)	Vlož sloupec do qr rozkladu.
qz(A,B)	Zobecněná vlastní čísla.
rank(A)	Hodnost matice.
rref(A)	
rsf2csf(U,T)	Převod reálné Schurovy formy do komplexní.
schur(A)	Schurův rozklad.
sqrtm(A)	Maticová druhá odmocnina.
subspace(A,B)	Úhel mezi 2 podprostory.
svd(A)	Rozklad do singulárních čísel.
svds(A,k)	Najde několik singulárních čísel.
trace(A)	Stopa matice.

Speciální matice

Matice	Popis
[ ]	Prázdná matice.
compan	Matice přidružená k charakteristickému polynomu.
eye	Jednotková matice.
gallery	Testovací matice.
hadamard	Hadamardova matice.
hankel	Hankelova matice.
hilb	Hilbertova matice.
invhilb	Inverzní Hilbertova matice.
magic	Magický čtverec.
ones	Matice jedniček.
pascal	Pascalův trojúhelník.
rand	Matice čísel s rovnoměrným rozložením.
randn	Matice čísel s normálním rozložením.
rosser	Testovací matice.
toeplitz	Töplitzova matice.
vander	Vandermondeova matice.

wilkinson	Wilkinsonova testovací matice.
zeros	Matice nul.

Řídké matice

V mnoha praktických aplikacích (FEM, simulace obvodů, apod.) se vyskytují matice, které obsahují pouze několik nenulových prvků. Těmto maticím se říká **řídké**.

Úspora paměti - ukládají se pouze nenulové hodnoty a jejich indexy.

Úspora výkonu - využití speciálních algoritmů.

Funkce pro práci s řídkými maticemi

Funkce	Popis
bic	
bicstab	
cgs	
cholinc	
colmmd	
colperm	
condest	
dmperm	
eigs	
etree	
etreeplot	
find	
full	
gmres	
gplot	
issparse	

luinc	
nnz	
nonzeros	
normest	
nzmax	
qmr	
randperm	
spalloc	
sparse	
spaugment	
spconvert	
spdiags	
speye	
spfun	
spones	
spparms	
sprand	
sprandn	

sprandsym	
sprank	
spy	
svds	
symbfact	
symmd	
symrcm	
treelayout	
treeplot	

Datová analýza

Funkce	Popis
corrcoef(x)	Korelační koeficienty.
cov(x)	Kovarianční matice.
cplxpair(x)	Setřídění vektoru do komplexně sdružených párů.
cumprod(x)	Kumulativní součin.
cumsum(x)	Kumulativní součet.
cumtrapz(x,y)	Kumulativní trapézová integrace.
del2(A)	Diskrétní Laplacián.
diff(x)	Rozdíly mezi prvky.
gradient(Z,dx,dy)	Přibližný gradient.
histogram(x)	Histogram.
max(x), **max(x,y)**	Maximum.
mean(x)	Střední hodnota.
median(x)	Medián.
min(x), **min(x,y)**	Minimum.
rand(x)	Náhodná čísla s rovnoměrným rozdělením.
randn(x)	Náhodná čísla s normálním rozdělením.

sort(x)	Setřídění dat.
sortrows(A)	Setřídění řádek.
std(x)	Standardní odchylka.
subspace(A,B)	Úhel mezi 2 podprostory.
sum(x)	Součet prvků.
trapz(x,y)	Trapézová integrace.

Polynomy

V MATLABu je polynom reprezentován řádkovým vektorem, jehož prvky odpovídají koeficientům polynomu (uspořádaným sestupně).

Operace	**Funkce**
Kořeny	**roots** **poly**
Násobení	**conv**
Sčítání	+
Dělení	**deconv**
Derivace	**polyder**
Vyhodnocení	**polyval**
Racionální polynomy	**roots** **residue**
Prokládání křivek	**polyfit**

Interpolace

1-D interpolace

interp1

2-D interpolace

interp2

meshgrid

interp3, interpn

ndgrid

Triangulace

delaunay

trimesh

tsearch, dsearch

convhull

griddata

Kubické spliny

Je třeba dopsat:

- Basic Features
- Piecewise Polynomials
- Integration
- Differentiation
- Spline Interpolation on a Plane

Fourierova analýza

Je třeba dopsat:

- Discrete Fourier Transform
- Fourier Series

Optimalizace

Je třeba dopsat:

- Zero Finding
- Minimization in One Dimension
- Minimization in Multiple Dimensions
- Practical Issues

Derivování a intergrování

Je třeba dopsat:

- Integration
- Differentiation

Obyčejné diferenciální rovnice

Je třeba dopsat:

- Initial Value Problem Format
- ODE Suite Solvers
- Basic Use
- ODE File Options
- Solver Options
- Finding Events

Objektově orientované programování

Je třeba dopsat:

- Object Identification
- Creating a Class
- The Constructor
- Object Precedence
- Displaying Objects
- Overloading Functions
- Adding Stack Elements
- Communicating Between Workspaces

- Removing Stack Elements
- Examining Stack Contents
- Overloading Operators
- Converter Functions
- Inheritance

Toolbox pro symbolickou matematiku

Calculus	
diff	Differentiate
int	Integrate
jacobian	Jacobian matrix
limit	Limit of an expression
symsum	Summation of series
taylor	Taylor series expansion

Linear Algebra	
colspace	Basis for column space
det	Determinant
diag	Create or extract diagonals
eig	Eigenvalues and eigenvectors
expm	Matrix exponential
inv	Matrix inverse
jordan	Jordan canonical form
null	Basis for null space
poly	Characteristic polynomial

rank	Matrix rank
rref	Reduced row echelon form
svd	Singular value decomposition
tril	Lower triangle
triu	Upper triangle

Variable Precision Arithmetic	
digits	Set variable precision accuracy
vpa	Variable precision arithmetic

Simplification	
collect	Collect common terms
expand	Expand polynomials and elementary functions
factor	Factor
horner	Nested polynomial representation
numden	Numerator and denominator
simple	Search for shortest form
simplify	Simplification
subexpr	Rewrite in terms of subexpressions

Solution of Equations	
compose	Functional composition
dsolve	Solution of differential equations
finverse	Functional inverse
solve	Solution of algebraic equations

Arithmetic Operations	
+	Addition
-	Subtraction
*	Multiplication
.*	Array multiplication
/	Right division
./	Array right division
\	Left division
.\	Array left division
^	Matrix or scalar raised to a power
.^	Array raised to a power
'	Complex conjugate transpose
.'	Real transpose

Special Functions	
cosint	Cosine integral, Ci(x)
hypergeom	Generalized hypergeometric function
lambertw	Solution of $\lambda(x)\, e^{\lambda(x)} = x$
sinint	Sine integral, Si(x)
zeta	Riemann zeta function

Access To Maple	
maple	Access Maple kernel
mapleinit	Initialize Maple
mfun	Numeric evaluation of Maple functions
mhelp	Maple help
mfunlist	List of functions for mfun
procread	Install a Maple procedure

Pedagogical and Graphical Applications	
ezcontour	Contour plotter
ezcontourf	Filled contour plotter
ezmesh	Mesh plotter
ezmeshc	Combined mesh and contour plotter
ezplot	Function plotter
ezplot3	3-D curve plotter
ezpolar	Polar coordinate plotter
ezsurf	Surface plotter
ezsurfc	Combined surface and contour plotter
funtool	Function calculator
rsums	Riemann sums

Conversions	
char	Convert sym object to string
double	Convert symbolic matrix to double
poly2sym	Function calculator
sym2poly	Symbolic polynomial to coefficient vector

Basic Operations	
ccode	C code representation of a symbolic expression
conj	Complex conjugate
findsym	Determine symbolic variables
fortran	Fortran representation of a symbolic expression
imag	Imaginary part of a complex number
latex	LaTeX representation of a symbolic expression
pretty	Pretty print a symbolic expression
real	Real part of an imaginary number
sym	Create symbolic object
syms	Shortcut for creating multiple symbolic objects

Integral Transforms	
fourier	Fourier transform
ifourier	Inverse Fourier transform
ilaplace	Inverse Laplace transform
iztrans	Inverse z-transform
laplace	Laplace transform
ztrans	z-transform