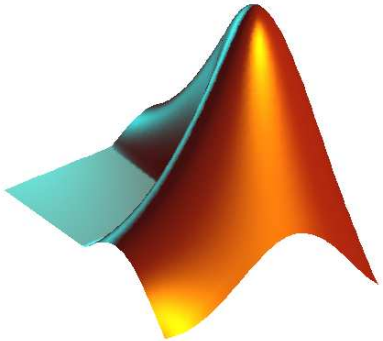
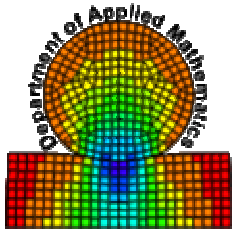


Úvod do Matlabu



Vít Vondrák

Katedra aplikované matematiky

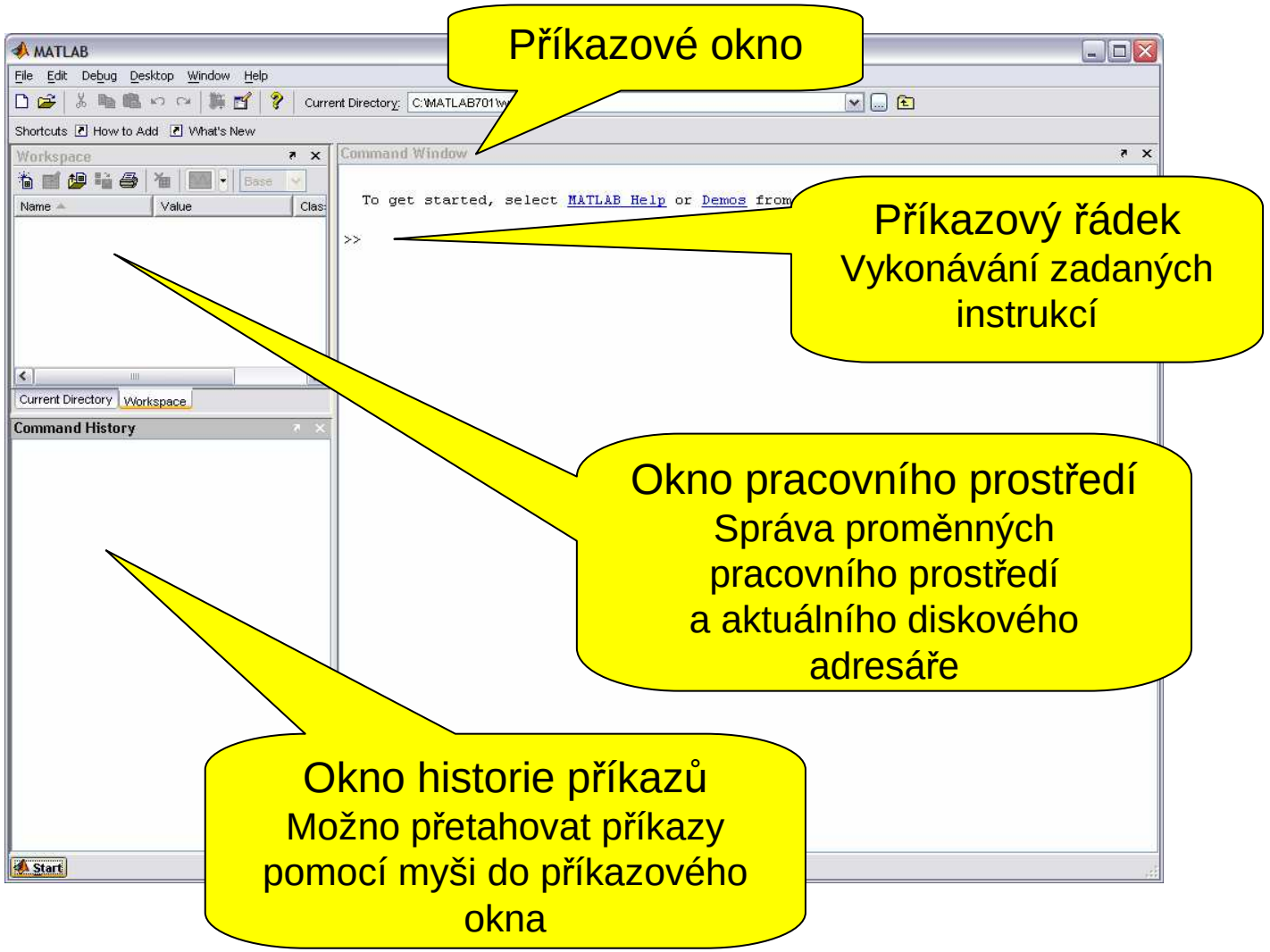
FEI, VŠB-TU Ostrava

Co je Matlab?

- Interaktivní softwarový balík MathWorks Inc.
- Matlab=MATrix LABoratory
  - Základním typem proměnné je matice
  - Číslo je maticí typu 1x1
- Obsahuje řadu příkazů pro správu proměnných, vstupně-výstupní operace, základní operace algebry vektorů matic a příkazy vyšší matematiky.
- Výkonné nástroje pro zpracování grafických výstupů.
- http://www.mathworks.com

- Interaktivní vykonávání příkazů
- Vykonávání skriptů tj. posloupností matlabovských příkazů tzv. m-souborů
- Programování vlastních matlabovských funkcí tzv. m-funkcí
- Programování funkcí v jazyce C nebo Fortran
- Možno doplňovat o toolboxy obsahující nástroje pro řešení některých praktických matematických, inženýrských či statistických problémů.

Matlab – uživatelské rozhraní



Definice proměnných

Definice proměnné probíhá automaticky přiřazením její hodnoty pomocí znaménka rovnosti:

```
>>a=3
```

K definici vektoru používáme hranaté závorky. Mezery nebo čárky oddělují prvky v řádku.

```
>>v=[3 5 7]
>>w=[1,5,1+2*i]
```

Imaginární jednotka se zapisuje pomocí `i`

Definice matice je stejná jako definice vektoru s tím, že řádky se oddělují pomocí středníku nebo klávesou **<Enter>**.

```
>>A=[1 2 3; 4 5 6]
```

nebo

```
>>B=[1 2 3
4 5 6]
```

Definice textové proměnné se provádí pomocí apastrofů:

```
>>c='Dobre rano Matlabe!'
```

K definici matic můžeme taktéž využít některou ze standardních funkcí Matlabu, která generuje matice.

```
>>I=eye(3)
>>O=zeros(2,3)
>>e=ones(1,4);
```

Přidáme-li za příkaz středník, bude potlačen výstup.

```
>>f=ones(1,10);
```

Správa proměnných

`>>who` nebo `>>whos`

Tento příkaz vypíše všechny aktuálně definované proměnné

`>>size(name)`

Vrací rozměry proměnné name.

`>>clear name`

Vymaže proměnnou name.

`>>save file name1,name2,…`

Uloží proměnné `name1`, `name2`, … do souboru `file.mat`. Pokud není uveden `file` uloží do souboru `matlab.mat`. Pokud není uvedena žádná proměnná uloží všechny.

`>>load file`

Nahraje proměnné ze souboru file.mat.

Příkazy `load` i `save` lze použít s volbou `–ASCII`, pak jsou proměnné uloženy do textového souboru. Např.:

`>>save –ASCII A.txt A`

Data uložená v různých formátech je možné importovat do Matlabu pomocí

`>>uiimport`

Všechny zadané příkazy včetně jejich odezvy mohou být průběžne ukládány do souboru `name` pokud použijete

`>>diary name`

Přístup k prvkům matic

`A(2,1)`

Vypíše prvek z 2 řádku a 1 sloupce matice `A`.

`A(2,[1 3])`

Vypíše 1. a 3. prvek z 2. řádku.

`a:d:b`

definuje aritmetickou posloupnost s prvním prvkem rovným a a posledním rovným b. d je diference mezi sousedními prvky posloupnosti. Pokud je d=1 stačí napsat `a:b`.

Pokud chceme vypsat 1. řádek matice `A` pak stačí použít jeden z příkazů

`A(1,1:3)`

`A(1,1:end)`

`A(1,:)`

Obdobně můžeme získat i libovolnou submatici. Např. minor matice `A` příslušný prvku `(1,1)` získáme příkazem

`M11=A(2:end,2:end)`

Maticové operace

`+` … operátor sčítání.

`-` … operátor odečítání.

`*` … operátor násobení.

`^` … operátor mocnění.

`/` … operátor dělení zprava. (`a/b = a*b^(-1), 2/3=0.6667`)

`\` … operátor dělění zleva. (`a\b = a^(-1)*b, 2\3=1.5`)

`'` … operátor transponování

Při použití operátorů pro operace mezi dvěmi skaláry je význam operací zřejmý. Pro operace matice-matice provede Matlab tuto maticovou operaci je-li definována. Obdobně pro operace vektor-vektor popř. matice-vektor.

K provedení operací mezi jednotlivými prvky matice či vektoru musíme umístit `.` před operátor.

```
u=[1 2 3];
u*u'
u.*u
```

Je možno využívat celou řadu běžných funkcí jako `cos`, `sin`, `sqrt`, ...

Práce s polynomy

`roots(c)`

kde c je vektor koeficientů polynomu seřazených sestupně podle mocnin x,

tj. $p(x) = c_n x^n + c_{n-1} x^{n-1} + \ldots + c_1 x + c_0$

Pomocí funkce `polyval` můžeme vyhodnotit funkční hodnotu polynomu. Syntaxe je:

`polyval(c,x)`

Kde x je hodnota argumentu a c jsou koeficienty.

Např.

```
>>c=[1 1 0 -1]
>>x=roots(c)
>>polyval(c,x)
```

Soustavy lineárních rovnic

Matlab má taktéž definovánu i Gaussovu eliminaci pro řešení soustav lineárních rovnic. Např.:

```
3x + 5y = 6
4x + 8y = 1
```

definujme

```
>>A=[3 5; 4 8]
>>b=[6 1]
```

Nyní musíme transponovat vektor b abychom dostali korektní maticový zápis výše uvedené soustavy.

```
>>b=b'
```

Pak soustavu vyřešíme pomocí operátoru dělení zleva

```
>>x=A\b
```

Budeme-li chtít řešit soustavu LU rozkladem můžeme použít funkci `lu`:

```
>>[L,U]=lu(A)
>>y=L\b
>>x=U\y
```

Vlastní čísla a vektory

Pokud budeme chtít nalézt vlastní čísla a vlastní vektory matice, můžeme využít funkce `eig`.

```
>>[V,L]=eig(A);
>>diag(L)
>>V
```

kde `diag(L)` vypíše vlastní čísla, která jsou na diagonále matice `L` a matice `V` obsahuje vlastní vektory po sloupcích.

Zkouška:

```
>>A*V-V*L
```

Funkce `poly` vrací koeficienty charakteristického polynomu

```
>>c=poly(A)
>>l=roots(c)
```

Kontrola:

```
>>trace(A),sum(l)
>>det(A),l(1)*l(2)
```

2D grafika

Základním grafickým příkazem v Matlabu je `plot`. Syntaxe tohoto příkazu je

`plot(x,y,options)`

kde x jsou x-souřadnice bodů které se mají vykreslovat a y jsou jejich odpovídající y-souřadnice. `Options` definuje jakým způsobem se body budou vykreslovat (určují barvu a styl vykreslení bodů).

Barvy: `k`-černá, `b`-modrá, `r`-červená, `g`-zelená, `w`-bílá

Styl: `-` body spojí čarou,`--` přerušovanou,`*` body vykreslí jako hvězdičky, `+` křížky, o kolečka.

```
>>plot([10 15],[13 5],'r*')
```

Pro popis grafického výstupu můžete použít taktéž následující příkazy:

```
>>xlabel('Toto je osa x')
>>ylabel('Toto je osa y')
>>title('Použití funkce plot.')
```

Velmi důležitý je příkaz `hold on`, který zruší překreslování grafického okna a umožní tak výstup více grafických příkazů do jednoho okna. `hold off` vrátí vše do původního stavu.

```
>>figure
```

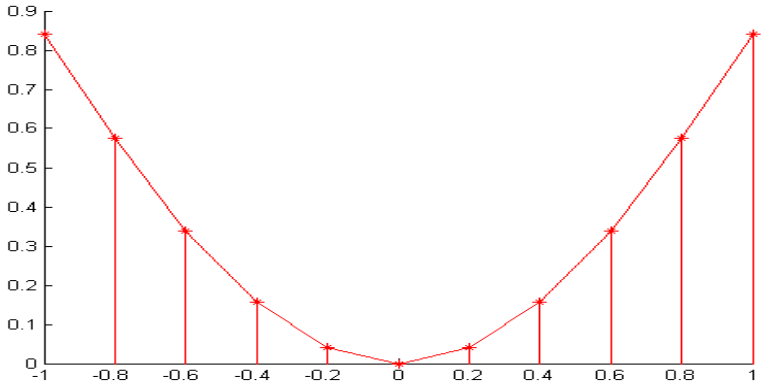
vytvoří nové grafické okno.

Další možnosti úprav grafického výstupu jsou dostupná přes menu grafického okna

Graf funkce

Chceme-li vykreslit graf funkce musíme nejprve diskretizovat osu x a v těchto bodech vypočíst funkční hodnoty. Mějme tedy funkci $y=xsin(x)$ jejíž graf chceme vykreslit na intervalu <-1,1>.

```
>>xx=-1:0.05:1;
>>yy=xx.*sin(xx);
>>plot(xx,yy)
```



Můžeme ovšem využít

i funkci ezplot

```
>>ezplot('x*sin(x)',[-1 1])
```

Graf křivky zadané parametricky

Mějme dánu křivku

$x=2\cos(t)$

$y=3\sin(t)$, $0 \leq t < 2\pi$.

Křivku vykreslíme následovně:

```
>>t=0:0.1:2*pi;
>>x=2*cos(t);
>>y=3*sin(t);
>>plot(x,y,’g--‘)
```

Takto vykreslená křivka je ovšem zobrazena deformovaně vlivem různých měřítek na jednotlivých osách. Toto můžeme napravit příkazem

```
>>axis equal
```

Pro vykreslení je možné opět využít funkci `ezplot`

```
>>ezplot('2*cos(x)','3*sin(x)',[0 2*pi])
```

3D grafika

Základním příkazem podobně jako v případě 2D grafiky je `plot3`. Jeho syntaxe je intuitivně zřejmá:

```
plot3(x,y,z,options)
```

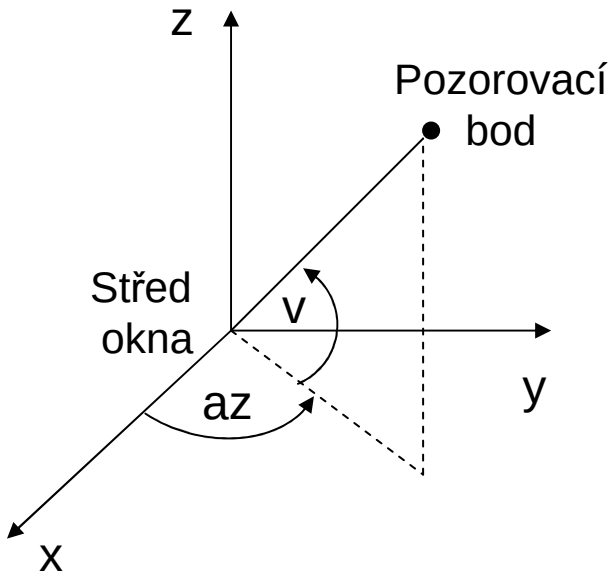
Další příkazy:

```
view(az,v)
```

Určuje pohled na 3D objekt. `view(3)` je standardní 3D pohled, `view(2)` je pohled na xy rovinu.

```
zlabel(text)
```

Popisuje osu z.



Graf parametrické křivky ve 3D

Obdobně jako ve 2D můžeme i v 3D vykreslovat křivky dané parametricky.

```
>>t=0:0.1:8*pi;
>>x=cos(t);
>>y=sin(t);
>>z=t;
>>plot3(x,y,z)
>>view(2)
```

Stejný graf obdržíme použitím funkce `ezplot3`

```
>>ezplot3('cos(t)','sin(t)','t',[0 8*pi])
```

Graf funkce 2 proměnných

Budeme-li chtít zobrazit funkci

f(x,y)=exp(-x^2-y^2)

je postup následující:

Určíme intervaly a dělení os x a y.

```
>>x=-1:0.1:1;
>>y=-1:0.1:1;
```

Pak vytvoříme síť na které budeme funkci zobrazovat.

```
>>[xx,yy]=meshgrid(x,y);
```

Vypočítáme funkční hodnoty funkce nad touto sítí.

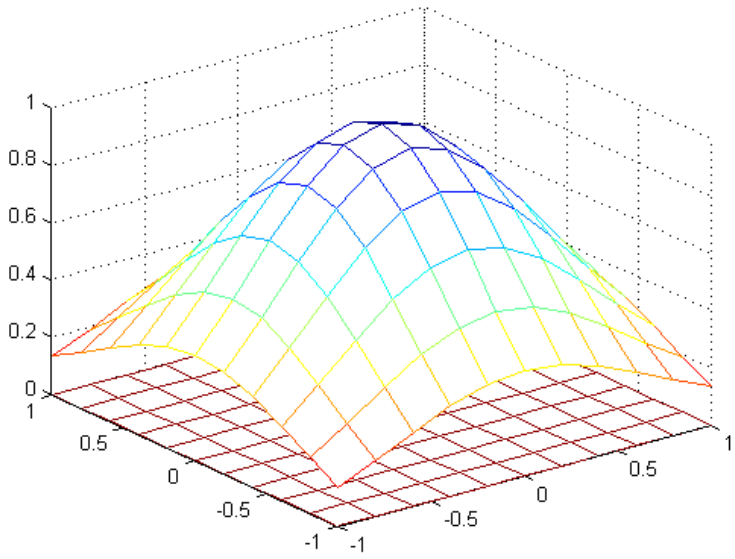
```
>>zz=exp(-xx.^2-yy.^2);
```

Vykreslíme graf pomocí

```
>>mesh(xx,yy,zz)
```

nebo plochu

```
>>surf(xx,yy,zz)
```



K vykreslení je možné taktéž použít funkce `ezmesh` či `ezsurf`

```
>> ezmesh('exp(-x.^2-y.^2)',
[-1,1,-1,1])
```

Programování v Matlabu

- Skripty
  - Samostatné textové soubory obsahující posloupnost matlabovských příkazů
  - Soubor musí mít příponu .m
  - Všechny proměnné definované ve skriptu mají platnost i v prostředí Matlabu
  - Spouští se zadáním názvu souboru (bez přípony).
  - Prohledává se aktuální adresář + adresáře uvedené v proměnné `path`

- Funkce
  - Samostatný textový soubor obsahující definici funkce ve formě

```
function [op1,...,opm]
   =name(ip1,…,ipn)
```

Pak následuje posloupnost matlabovských příkazů.

  - Všechny proměnné mají pouze lokální platnost (může být změněna pomocí deklarace `global`)
  - Funkce se volá příkazem

```
>>[o1,…,om]=name(i1,…,in)
```

Řídící příkazy

- `for` cyklus
  `for i=a:d:b,` *posloupnost příkazů*`, end`
- `while` cyklus
  `while` *podmínka*`,` *posloupnost příkazů*`, end`
- if příkaz
  `if` *podmínka*
  posloupnost příkazů
  `elseif` *podmínka*
  posloupnost příkazů
  `else`
  posloupnost příkazů
  `end`
  - *podmínka* se testuje na nenulovost (logické „ano“) či nenulovost (logické „ne“)
  - Je možné používat následující logické operátory: & (and), | (or), ~ (not)
  - Relační operátory: <, <=, >=, >=, ==, ~=