

# Školení MATLAB I

## SPRÁVA DAT, VÝPOČETNÍ FUNKCE, TVORBA GRAFŮ A ZÁKLADY PROGRAMOVÁNÍ

### Obsah

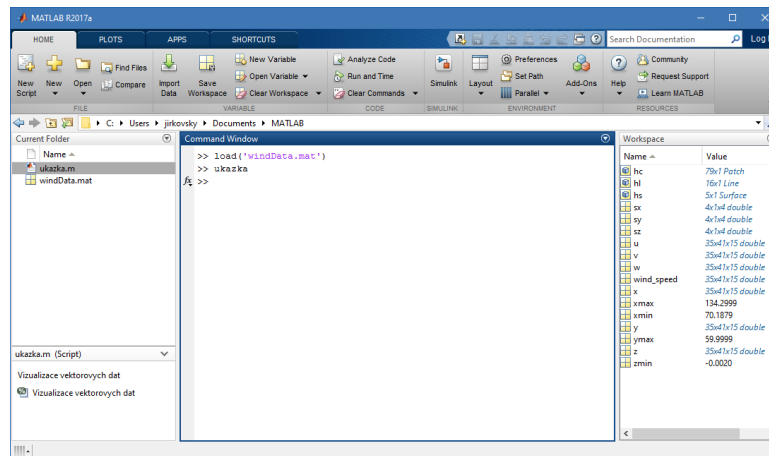
Úvod do prostředí MATLAB.....	2
Základní příkazy v MATLABu .....	3
Vektory, matice a pole .....	4
Výpočetní operace.....	8
Výpočetní funkce .....	9
Text.....	10
Nápověda/dokumentace a správa proměnných.....	11
Ukládání / Načítání dat.....	12
Grafika v MATLABu.....	14
Čárový 2-D graf.....	14
Skripty v MATLABu .....	16
Další typy 2-D grafů .....	18
3-D grafy .....	19
Live Editor a Live Script.....	20
Základní datové typy používané v MATLABu .....	23
Číselné datové typy .....	23
Datový typ logical a logické operace .....	23
Indexování maticí logických hodnot.....	24
Přehled dalších (zajímavých) funkcí .....	26
Řízení toku programu - cyklus a podmínka .....	27
Podmínkový příkaz v MATLABu.....	27
Cyklus For .....	27
Cyklus While .....	28
Přerušení jednoho průběhu cyklu, ukončení celého cyklu.....	28
Rozdělovač.....	28
Funkce v MATLABu.....	29
Publikování skriptů a funkcí.....	30
Přizpůsobení prostředí MATLAB – <i>Preferences</i> .....	31
Kde najít informace o MATLABu .....	31

## Úvod do prostředí MATLAB

- **Co je MATLAB?**
  - Programovací jazyk 4. generace a interaktivní vývojové prostředí pro vědecko-technické výpočty
    - Numerické výpočty
    - Analýza dat a vizualizace
    - Vývoj a programování algoritmů
    - Vývoj aplikací a jejich nasazení
  - Rozšiřující knihovny pro široké spektrum aplikací - toolboxy (např. zpracování signálu a komunikace, zpracování obrazu a videa, řídicí systémy, testy a měření dat)
  - MATLAB je plný programovací jazyk, včetně objektově-orientovaného programování
- **Toolboxy**
  - doplňující knihovny funkcí
- **Verze MATLABu**
  - vychází 2x ročně
  - aktuální R2022a
- **Dnes se zaměříme na tyto kroky při práci v prostředí MATLAB**
  - vytváření dat a manipulace s daty
    - `>> x = 0:0.1:2*pi`
  - výpočty a analýza dat
    - `>> y = sin(x)`
  - vizualizace výsledků
    - `>> plot(x,y)`

# Základní příkazy v MATLABu

- **MATLAB Desktop**
  - **Uživatelské rozhraní**, přes které ovládáme MATLAB



- **Command Window**
  - `>> příkaz`
  - `>> 3+6`
  - zadávání příkazů, které se ihned provedou
- **Přiřazovací příkaz**
  - *proměnná = hodnota*
    - `>> a = 3.8`
    - `>> a = 3.8;` ... znak ";" za příkazem potlačí výpis výsledku
    - `>> c = -2.3e-16` ...  $-2.3 \times 10^{-16}$ , exponenciální tvar
- **Workspace**
  - seznam proměnných, které jsou aktuálně v paměti
    - *ans* ... proměnná, výsledek poslední operace bez přímého přiřazení výstupu
- **Názvy proměnných**
  - používat jen znaky anglické abecedy, číslice a podtržítka
  - musí začínat písmenem
  - MATLAB je **case sensitive**
- **Výrazy a základní matematika**
  - výraz může obsahovat
    - konstanty, proměnné
    - operátory: `+`, `-`, `*`, `/`, `^`
    - závorky `()`
    - funkce: zadané jménem, argumenty do oblých závorek
  - priorita – jako v matematice
    - `>> q1 = 2*(3+1)`
  - příklady vestavěných funkcí:
    - *sin, cos, sind, cosd, exp, log, log10, abs, sqrt*
    - `>> Q1 = log(q1)`
  - příklad vestavěných konstant:
    - *pi* ...  $\pi$

- **Formát zobrazení čísel** ve výpisu do Command Window lze nastavit příkazem `format`
  - `>> format long` ... výpis na více cifer
  - `>> Q1`
  - `>> pi`
  - `>> format short` ... standardní výpis
  - `>> pi`
- **Command History**
  - prohlížeč historie příkazů spuštěných v *Command Window*
  - vyvoláme kurzorovými šipkami  $\uparrow\downarrow$  (+ zúžení zadáním počátečních písmen)
  - lze znovu spouštět příkazy (dvojklik), nebo i celé bloky příkazů (CTRL, SHIFT)

## Vektory, matice a pole

- proměnné mohou obsahovat více prvků, obecně **pole** (*array*) o libovolných rozměrech
- speciální případy pole:
  - dvourozměrné numerické pole ( $M \times N$  prvků) ... **matice**
  - jednorozměrné numerické pole ( $N \times 1$  /  $1 \times N$  prvků) ... **vektor**
  - jednoprvkové numerické pole ( $1 \times 1$ ) ... **skalár**
  - ve školeních MATLAB 1, 2 a 3 se seznámíme i s řadou nenumерických polí, např. pro práci s textem uvidíme pole znaků (*character array*)
- **Vektor**
  - jednorozměrné numerické pole
  - matice s jedním řádkem nebo sloupcem
  - zadání výčtem prvků
    - `>> v = [1 2 3]` ... řádkový
    - `>> v = [5;6;7]` ... sloupcový
  - výběr *n-tého* prvku ...  $v(n)$ 
    - `>> v(3)`
  - přiřazení nové hodnoty
    - `>> v(3) = 5`
    - `>> v(5) = 6` ... zvětší vektor
  - vytvoření posloupnosti s krokem 1 - znak ":"
    - `>> x = 1:5` ... vektor čísel od 1 do 5 s krokem 1
    -
  - transpozice
    - `>> x'` ... *ans* je sloupcový vektor
  - funkce lze volat na celé vektory
    - `>> y = log(x)`
    - `>> plot(x,y)` ... vykreslení grafu
  - `length(vektor)` ... délka vektoru
    - `>> length(y)`
  - Výpočetní operace s vektory
    - lze provádět s celými vektory
    - operace s vektorem a skalárem
      - `>> x+2` ... ke každému prvku vektoru se přičte 2
      - `>> 2*x` ... všechny prvky vektoru se vynásobí 2
    - **maticové operace** ... tj. operace z lineární algebry
      - přirozená syntaxe, neboť **MATLAB = MATrix LABoratory**

- např. skalární součin vektorů  $x$  a  $y$ :
  - `>> x*y'`
- **operace po prvcích**
  - např. násobení po prvcích:
    - `>> x.*y`
  - první prvek vektoru  $x$  vynásobíme s prvním prvkem vektoru  $y$ , druhý s druhým, ...
  - vektory  $x$  a  $y$  musí mít stejný počet prvků
    - *např. dva sloupce v tabulce*
  - výstup má stejný počet prvků jako  $x$  a  $y$ 
    - *sloupec s výpočtem*
  - používat místo cyklu – **vektORIZACE**
    - přehlednější kód a rychlejší výpočet
- při vytvoření posloupnosti lze zvolit krok
  - `>> x = 1:2:10` ... vektor čísel od 1 do 10 s krokem 2
  - může být libovolný: `0:0.1:5`, `10:-2:0`
  - `>> x = 1:0.1:5;`
  - `>> y = log(x)`
  - `>> plot(x,y)`
- vytvoření posloupnosti s daným počtem prvků
  - `linspace(první prvek, poslední prvek, počet prvků)`
  - `>> v = linspace(0,1,5)`
- zvětšení vektoru o prvek, spojování vektorů
  - `>> v = [v 10]`
  - `>> v = [v v]`
- **Variable Editor**
  - poklepnání na proměnnou ve *Workspace* - otevře se v tabulkovém zobrazení
- **Matice**
  - zadání výčtem prvků
    - prvky zadáváme po řádcích
    - oddělovač prvků v řádku – mezera " " nebo čárka ","
    - oddělovač řádků – středník ";"
    - `>> M = [1 2 3; 4 5 6; 7 8 9; 10 11 12]` ... matice 4x3
    - `>> M = [1 2 3; 4 5 6; 7 8 9; 10 11 a]`
      - hodnota z proměnné  $a$  se předá do matice  $M$
    - `>> M = [1 2 3; 4 5 6; 7 8 9; 10 11 sqrt(a)]`
      - hodnota odmocniny z  $a$  se předá do matice  $M$
    - `>> surf(M)` ... zobrazíme prvky matice vzhledem k jejich indexům
  - **Funkce pro tvorbu speciálních matic a vektorů**
    - `zeros(m,n)` ... matice nul ( $zeros(n) \sim n \times n$  – platí i pro další)
    - `ones(m,n)` ... matice jedniček
    - `eye(m,n)` ... jednotková matice
    - `magic(n)` ... matice přirozených čísel, stejné součty řádků a sloupců
    - `rand(m,n)` ... matice náhodných čísel s rovnoměrným rozdělením
    - `randn(m,n)` ... matice náhodných čísel s normálním rozdělením
  - **Indexování ... M(řádek, sloupec)**
    - výběr prvku
      - `>> M(2,1)` ... v 2. řádku 1. sloupce

- přiřazení nové hodnoty
  - `>> M(4,3) = 12`
- výběr více prvků
  - indexovat lze i přímo vektorem
  - `>> M([1 4],1)`
  - `>> M([1 4],[1 3])` ... výběr rohových prvků matice 4x3

`M([1 4],1)`      `M([1 4],[1 3])`

	1	2	3
1	0	.	.
2	.	.	.
3	.	.	.
4	0	.	.

	1	2	3
1	0	.	0
2	.	.	.
3	.	.	.
4	0	.	0

- znak ":" lze použít pro indexování více prvků
  - `>> M(1:3,2)`
- **výběr všech prvků v dané dimenzi** - využíváme samotný znak ":"
  - `>> M(:,2)` ... všechny řádky, druhý sloupec
- Poslední index - "end"
  - `>> M(2:end,end)` ... vybere všechny řádky od druhého, poslední sloupec
- předposlední index: *end-1*
- **Dotaz na velikost matice**
  - `size(proměnná)` ... vrátí velikost
    - `>> size(a)` ... 1 1
    - `>> size(M)` ... vrátí vektor dvou čísel – počet řádků, sloupců
    - výběr dimenze: `size(x,dim)` vrací pouze rozměr dané dimenze
    - u vektoru problém s funkcí `size` – je řádkový nebo sloupcový?
      - např.: 1x5 nebo 5x1
      - o kterou dimenzi si požádat?
    - řeší `length` - vrací delší rozměr = jediné číslo
- **Spojování matic**
  - `>> K = [M;M]` ... pod sebe
  - `>> K = [M M]` ... vedle sebe
  - funkce `repmat(matice,m-krát,n-krát)`
    - pro velké množství opakování dané matice
    - `>> repmat(M,3,2)`
  - funkce `cat(dim,A,B)`
- **Vícerozměrné pole**
  - vytvoření
    - `zeros`, `ones`, `rand` a `randn` lze použít i pro tvorbu vícerozměrných polí
    - funkce `cat(dim,A,B)`
      - `>> K = cat(3,M,M)` ... vytvoření 3-D pole
  - indexování
    - `K(i1,i2,i3,...)`
- **Dynamická změna velikosti vektorů, matic a vícerozměrných polí**
  - matici není nutné definovat naráz

- postupné přidávání prvků pomocí indexování
  - `>> u = 1` ... vytvoří skalár
  - `>> u(2) = 2` ... přidá další prvek
  - `>> u(5) = 3` ... přidá 5-tý prvek a mezi prvky s hodnotou 0
  - `>> u(2,1) = 4` ... přidá celý nový řádek
- jednoduché, pohodlné, ale může být pomalé
  - $\Rightarrow$  vhodné zejména ve fázi vývoje algoritmu
- na počátku můžeme "startovat" i z prázdné matice (`u = []`) nebo rovnou od prvního prvku (`u(1) = 1`)
- vymazání prvku – přiřazením prázdné hodnoty
  - `>> v(3) = []`
  - u matic lze smazat jen celý řádek, nebo celý sloupec (`M(2,:) = []`)
- **Lineární indexování**
  - indexováno postupně po sloupcích shora-dolů, zleva-doprava
    - `>> M(3)`
    - `>> M([1 3 6])` ... prvky jsou vráceny ve formě vektoru

(1,1) <sup>(1)</sup>	(1,2) <sup>(5)</sup>	(1,3) <sup>(9)</sup>
(2,1) <sup>(2)</sup>	(2,2) <sup>(6)</sup>	(2,3) <sup>(10)</sup>
(3,1) <sup>(3)</sup>	(3,2) <sup>(7)</sup>	(3,3) <sup>(11)</sup>
(4,1) <sup>(4)</sup>	(4,2) <sup>(8)</sup>	(4,3) <sup>(12)</sup>

indexování prvků (řádek, sloupec) vs. lineární index

- převod matice na vektor
  - `M(:)` ... celá matice v lineárním indexování

### Pozn.:

Indexováním (řádek, sloupec) lze vybrat pouze jeden prvek, nebo obdélníkovou submatici. Pro výběr více konkrétních prvků, které netvoří obdélníkovou oblast, lze použít indexování lineární.

- **Přetvarování vektorů, matic a vícerozměrných polí**
  - funkce *reshape*
- **Řazení prvků ve vektorech, maticích a vícerozměrných polích**
  - *sort* ... seřadí nezávisle jednotlivé sloupce
    - `>> M = magic(3)`
    - `>> sort(M)`
    - lze řadit vzestupně i sestupně
  - *sortrows* ... seřadí matici podle daného sloupce
    - `>> sortrows(M)`
    - lze volat na vektor nebo matici
    - lze volit sloupec, podle kterého řadíme, primární i další klíče, ...

## Výpočetní operace

- **Operace s maticí a skalárem**
  - $+$ ,  $-$ ,  $*$ ,  $/$
  - ke všem prvkům matice se přičte nebo odečte skalární hodnota
  - všechny prvky matice se vynásobí nebo vydělí skalární hodnotou
  - `>> K = M + 2`
  - `>> K = 2*M`
  - místo matice může být libovolné numerické pole
- **Maticové operace**
  - $+$ ,  $-$ ,  $*$ ,  $/$ ,  $\backslash$ ,  $^$
  - pro matice záleží na velikosti dle pravidel
    - sčítat a odčítat stejně velké matice
      - sečtou se (odečtou) odpovídající prvky
      - `>> M - M`
      - `>> M + K`
      - M a K mají stejné rozměry => mohou je přičíst
    - násobit matice s odpovídajícími vnitřními rozměry
    - $/ = * \text{inv}(A)$ ,  $\backslash = \text{inv}(A) *$ 
      - `>> v = 1:5` ... řádkový vektor (matice 1x5)
      - `>> v'` ... sloupcový vektor (matice 5x1)
      - `>> v*v'` ... skalár
      - `>> v'*v` ... matice 5x5
  - lze provádět s vektory a maticemi

⇒ Úloha:

Máme dvě přímky reprezentované rovnicemi:

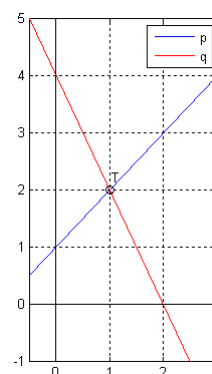
- p:  $y = x + 1$
- q:  $y = -2x + 4$

Najděte bod T, kde se přímky protínají.

⇒ Řešení:

Soustava lineárních rovnic:

- p:  $y_T = x_T + 1$  ...  $-x_T + y_T = 1$
- q:  $y_T = -2x_T + 4$  ...  $2x_T + y_T = 4$



V maticové podobě:  $\begin{bmatrix} -1 & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_T \\ y_T \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$  ...  $A\vec{x} = b \Rightarrow \vec{x} = A^{-1}b$

$$A = \begin{bmatrix} -1 & 1 \\ 2 & 1 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$$

Výpočet v příkazové řádce MATLABu:

```
>> A = [-1 1; 2 1]
>> b = [1; 4]
>> T = inv(A)*b
```

Alternativa:

```
>> T = A\b
```



- **Operace po prvcích**

- `.*` , `./` , `.^`
- pro každý prvek zvlášť
- obě pole musí mít stejné rozměry
- tzv. vektorizace – namísto cyklů používat vektory
  - `>> v.*v`
- vykreslení 3d grafu plochy zadané analyticky
  - $z = \sin(r)/r$ , kde  $r = \sqrt{x^2 + y^2}$  pro  $x, y \in (-2\pi, 2\pi)$
  - `>> x = -2*pi:pi/10:2*pi;`
  - `>> y = -2*pi:pi/5:2*pi;`
  - `>> [X,Y] = meshgrid(x,y);`
    - funkce *meshgrid* vytvoří rastr
    - matice *X* obsahuje v každém řádku vektor *x*.
    - matice *Y* obsahuje v každém sloupci vektor *y*'
    - *X* a *Y* mají stejné rozměry
  - `>> R = sqrt(X.^2+Y.^2)`
  - `>> Z = sin(R)./R;`
  - `>> surf(X,Y,Z)` ... graf plochy na níž leží body  $[X(i,j), Y(i,j), Z(i,j)]$
- lze provádět i s vícerozměrnými numerickými poli

## Výpočetní funkce

- **Volání jménem funkce, vstupní argumenty do kulatých závorek**
  - `out = fun(arg1, arg2)`
    - výstup se přiřadí do výstupní proměnné
  - `fun(arg1, arg2)`
    - jako součást složitějšího výrazu
    - samostatně => výsledek se uloží do *ans*
- **Skalární funkce**
  - funkce se vypočítá pro jednotlivé prvky pole
  - *sin, cos, sind, cosd, exp, log, log10, abs, sqrt*
    - `>> v = 0:5`
    - `>> log(v)`
- **Vektorové funkce**
  - pro celý vektor, u matic počítá zvlášť pro jednotlivé sloupce
  - *min, max, sum, mean*
    - `>> min(M)`
  - vždy popsáno v dokumentaci dané funkce
  - pokud chci aplikovat na všechny prvky matice – převedu matici na vektor
    - `>> min(M(:))`
- **Maticové funkce**
  - funkce určené pro celé matice
  - *det, inv*
    - `>> A = [1 2;3 4]`
    - `>> det(A)`
- **Seznam funkcí MATLABu**
  - Otevřít *Help > MATLAB > Functions*
    - seznam všech funkcí základního MATLABu

- seřazeno do skupin podle funkčnosti
- **Funkce s více výstupy**
  - funkce při volání primárně vrací první (hlavní) výstup
  - některé funkce mohou "na požádání" vracet i více výstupů
    - o další výstupy se požádá tím, že výstup funkce přiřadím více proměnným zadaným v hranatých závorkách
    - `[out1, out2] = fun(arg1, arg2)`
  - počet a význam výstupů je vždy popsán v dokumentaci
    - `>> doc min`
    - `>> [m,k] = min(M(:))` ... chci získat oba dva výstupy

### Pozn.:

Chování funkce se může lišit, vyžádá-li uživatel různý počet výstupů. Popis vždy v dokumentaci, např.:






```
>> doc find
```

## Text

- V MATLABu jsou dva datové typy pro práci s textem
- Oba jsou pole prvků stejně jako numerické datové typy, prvkem pole je buď:
  - jeden znak ... **pole znaků**
  - libovolný text ... **pole řetězců**
- **Pole znaků** (*character array, character vector*)
  - vytvoření: znak „'“ nebo funkce *char*
    - `>> s = 'Hello'`
  - prvkem pole je jeden znak
  - spojování polí znaků do delšího textu
    - `>> s1 = ['Hello', ' ', 'World!']`
    - platí pravidla pro spojování polí – nelze spojit pod sebe dvě pole s různým počtem prvků, tj. znaků
- **Řetězce a pole řetězců** (*string array*)
  - od R2016b
  - vytvoření: znak „"“ nebo funkce *string*
    - `>> r = "Hello World!"`
  - prvkem pole je jeden textový řetězec
  - spojování řetězců do víceprvkového pole
    - `>> r2 = ["Hello", " ", "World!"]`
    - `>> r2 = ["Hello"; "World!"]`
  - spojování řetězců do delšího textu
    - `>> r3 = "Hello" + "World!"`
    - při spojování "řetězec + číslo" se číslo automaticky převede na řetězec
      - `>> "a" + 1`
  - převod pole znaků na řetězec - funkce *string*
  - vhodné pro pokročilejší práci s textovými daty, např. textovou analytiku
- V případě zadávání textu jako vstupního parametru vestavěných funkcí, jsou oba datové typy zaměnitelné
  - `>> min(M, [], 'all');`
  - `>> min(M, [], "all");`
- zobrazení textů v příkazovém okně
  - `disp(text)` ... zobrazí text

- `>> disp("Hodnota " + sqrt(2) + " je odmocnina ze dvou.")`
- konverze: `co2naco`
  - `num2str(8)` ... číslo na pole znaků
  - `str2num('8')` ... text (řetězec/pole znaků) na číslo

## Nápověda/dokumentace a správa proměnných

- jednoduchá textová nápověda
  - `help fcn`
  - ke každé funkci (příkazu)
    - `>> help plot`
  - na konci helpu nápověda souvisejících funkcí
- **hlavní HTML dokumentace**
  - `doc fcn`
    - html dokumentace k dané funkci (příkazu)
    - `>> doc plot`
  - umístit kurzor do příkazu + F1
    - zobrazí html nápovědu v malém okně
    - lze přepnout do velkého – [Open Help Browser](#)
  - otevření okna s dokumentací 
    - procházení podle obsahu
    - full-text vyhledávání zadaného výrazu v html dokumentaci
  -  Search Documentation
    - rychlý přístup k hledání v html dokumentaci
    - $\Rightarrow$  *chcete řešit lineární rovnice*  $\Rightarrow$  vyhledat: *linear equation*
- rychlé vyhledávání funkcí 
- syntaxe – automatické zobrazení syntaxe
  - `fcn( ... počkat`
  - `>> plot(`
- dokončování – automatické dokončování
  - klávesa Tab
  - `>> plo + Tab`
- **Příkazy pro správu proměnných**
  - `who ...` seznam aktuálních proměnných
    - `>> who`
    - `whos ...` seznam proměnných s dalšími informacemi
  - `clear proměnná1 proměnná2 ...` smazání proměnných
    - `>> clear c`
    - lze i graficky – označit proměnné v okně *Workspace* > klávesa Delete  $\Rightarrow$  (smazat „i“ v okně workspace)
  - `clear ...` smazání všech proměnných ve workspace 
    - `>> clear`
  - `clc ...` smazání obrazovky 
    - `>> clc`
- **Toolstrip s ikonami**
  - procházení pomocí záložek
  - funkce pro správu, vybrané příkazy

smazat data z Workspace  a vyčistit obrazovku 

### Pozn.:

Syntaxe příkazů v MATLABu:

A) funkční: `fun(arg1, arg2)`






podporují všechny příkazy

umožní předání výstupních parametrů `[out1, out2] = fun(arg1, arg2)`

B) příkazová: `fun arg1 arg2`

podporují jen příkazy, které mají vstup textový řetězec, většinou příkazy pro správu souboru, proměnných, ...

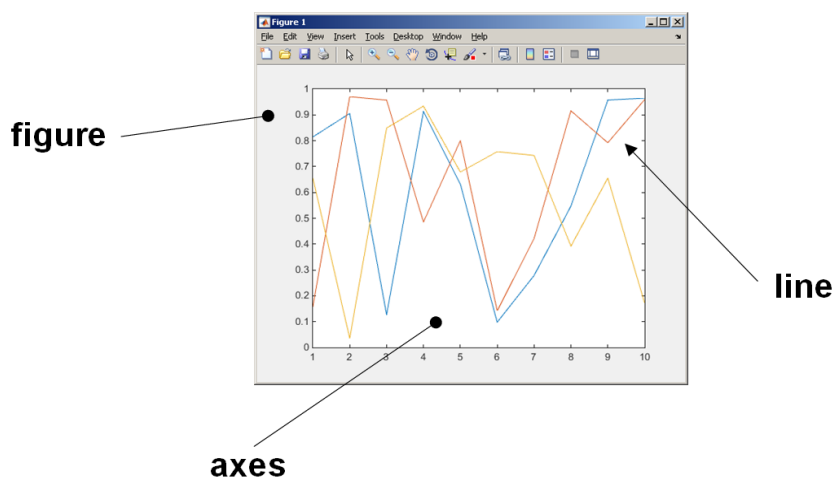
## **Ukládání / Načítání dat**

- **Current Folder**
  - aktuální složka – MATLAB vidí její obsah
  - cesta k aktuální složce napsána v řádku pod toolstripem
    -  C: > Users > studnicka > Documents > MATLAB
- Umístění oken v uživatelském rozhraní
  - okna lze maximalizovat, udělat plovoucí, zavřít, dokovat – volby v menu 
    - zadokovat Command History
  - zavřená okna a jejich rozložení lze vyvolat pomocí ikony *Layout*  (základní nastavení *Layout > Default*)
- **Uložení dat do souboru .mat**
  - GUI
    - *save workspace* 
    - označit proměnné v okně *Workspace* > pravý klik > kontextové menu > *Save As ...*
  - příkazy
    - *save jmeno\_souboru proměnná1 proměnná2*
    - `>> a = 1, b = 2`
    - `>> save mojeData a b`
- **Načtení souboru typu .mat**
  - GUI
    - Otevřít ,
    - poklepání na soubor v okně *Current Folder*
  - příkazy
    - *load jmeno\_souboru*
    - `>> load mojeData`
- **Import dat v jiných formátech**
  - GUI
    - Import 
    - poklepání na soubor v okně *Current Folder* spustí akci dle typu souboru
      - poklepat na soubor *NejvetsiMestaCR.xlsx*

- otevře se *import wizard* > *Text Options* = *Cell Array of Character Vectors* > *Import*
- dvojklikem v okně *Workspace* otevřít proměnnou *NejvetsiMestaCR* ve *Variable Editoru*
- označit sloupce *PocetObyvatel* a *Mesto* > *PLOTS* > *pie*
- *import wizard* > *Output Type* = *Column vectors* > *Import*
- lze měnit způsob importu, názvy proměnných, filtrovat data, ...
- příkazy
  - *readtable*
    - `>> Data = readtable('NejvetsiMestaCR.xlsx')`
    - umí importovat formáty MS Excel, textové soubory i formáty s fixní šířkou
    - výstupem proměnná datového typu *table*
      - podrobně ve školení MATLAB 2
      - datový typ pro snadnou manipulaci se sloupcově orientovanými nebo tabulkovými daty
      - typickým použitím je držení dat, kde sloupce představují různé měřené proměnné a řádky jednotlivá pozorování
      - vhodné pro data načtená z tabulek ve formátu Microsoft Excel, textových souborů a databází
      - `>> doc table`
      - různé sloupce mohou obsahovat různé datové typy
      - proměnné (= sloupce) jsou pojmenovány
      - přístup k datům v sloupcích tečkovou notací: *data = T.JmenoSloupce*
      - řádkům lze přiřadit jména volitelně
  - *writetable*
  - funkce pro jiné datové typy od R2019a:
    - *readtimetable* / *readmatrix* / *readcell* / *readvars*
    - *writetimetable* / *writematrix* / *writecell*
  - funkce pro jiné datové typy existující i před R2019a:
    - *xlsread* / *csvread*
    - *xlswrite* / *csvwrite*








## Grafika v MATLABu

- Grafické objekty
  - *Figure* → *Axes* → plot, image, ...




## Čárový 2-D graf

- Příkaz
  - *plot* (*dataX*, *dataY*, '*3znaky pro formátování*')
    - toto se může opakovat vícekrát
    - `>> doc plot ...` význam 3 znaků pro formátování
    - `>> x = 0:pi/20:2*pi;`
    - `>> y = sin(x);`
    - `>> plot(x,y,'r*:')`
- Úprava os a popisky grafu
  - Popisky lze vkládat pomocí **příkazů** nebo **interaktivně** (menu *Insert*)
    - lze použít syntaxi TeX
  - Příkazy
    - `>> title('Graf')` ... nadpis
    - `>> xlabel('osa x')` ... popisek osy x
    - `>> ylabel('osa y')` ... popisek osy y
    - `>> axis tight` ... nastaví osy těsně ke grafu
      - standardně nechá mezery aby osy končily celým číslem
      - číselné nastavení zobrazení oblasti
      - `axis([xmin xmax ymin ymax])`
      - `v = axis` – aktuální limity os
      - `xlim([xmin xmax])`, `ylim([ymin ymax])`, `zlim([zmin zmax])`
    - `>> grid on` ... mřížka
    - `>> legend('sin(\alpha)')` ... legenda
      - \ ... syntaxe TeX
  - GUI – galerie grafů

- záložka PLOTS
- označit proměnné v okně workspace (CTRL)
- vybrat graf > stisknutím obrázku se vykreslí
  - + vypíše do příkazové řádky příslušný příkaz
- ⇒ označit x a y > PLOTS > plot
- Toolbar
  - **Figure Toolbar**

  - *New Figure, Open File, Save, Print, Link Plot, Insert Colorbar, Legend*
  - **Axes Toolbar**

  - *Data Tips, Rotate 3D, Pan, Zoom, Restore View*
  - lze přidávat a odebírat nástroje (i vlastní)
  - další položky lze ovládat nebo přidávat přes menu *Tools*
- Interaktivní editace 
  - kliknout v menu na ikonu šipky
  - kliknutí na grafický prvek ... výběr prvku
  - pravý klik na grafický prvek > kontextové menu pro změnu vlastností
    - ⇒ změna barvy, tloušťky čáry, značek, ...
- Property Inspector
  - kliknout v menu na ikonu
 
  - nastavení vlastností zobrazených objektů (okna, axes, plotů, ...)
  - kliknutím na daný prvek se zobrazí jeho vlastnosti a nastavení
    - ⇒ změna detailního nastavení značek, popisků os, mřížky, ...
  - Alternativní zobrazení nástrojů
    - *View > Figure Palette, Plot Browser, Property Editor*
- **Interaktivní označování a změna vykreslených dat**
  - *Data Tips* 
    - kliknutí na data = přidá popisek
    - lze posouvat, přidávat další (přes kontextové menu), mazat (delete)
      - ⇒ označit datový bod, posunout značku, ... , vymazat
  - *Link Plot* 
    - interaktivní sepnutí grafu s proměnnou se zdrojovými daty
    - po změně dat změna zobrazení
- **Uložení grafu**
  - *Save*  ⇒ soubor.fig
    - ⇒ uložit jako mujGraf.fig
  - *File > Save As ...* další formáty (bmp, ...)
- **Tisk**
  - *File > Print Preview, File > Print*
  - *print -dps soubor*

## Skripty v MATLABu

- o namísto zápisu jednotlivých příkazů do příkazové řádky je zapíšeme do textového souboru a spustíme najednou
- o založení nového skriptu





- New Script 
- komentáře uvozujeme %
- řádky komentáře od prvního až do prvního vynechaného řádku
  - nápověda, která se zobrazí po zadání `>> doc jmeno_skriptu`

### ⇒ P01 v1

```
% Grafy funkci
%
% - sinus

% priprava dat
x = 0:pi/20:2*pi;
y = sin(x);

figure      % nove graficke okno
plot(x,y)   % vykresleni grafu
```

- o **Využití Command History**
  - příkazy z okna *Command History* lze přetahovat (=kopírovat) do Editoru
  - pro více řádků - CTRL, SHIFT
- o **Uložení**
  - tl. Save  ⇒ soubor.m
    - `Priklad1.m`
  - označení neuloženého skriptu – hvězdička u jména v záhlaví
- o **Spuštění**
  - tl. Run 
    - uloží a spustí
  - v příkazové řádce zadáním jména skriptu
    - `>> Priklad1`
  - Nápověda
    - `>> doc Priklad1`
    - zobrazí počáteční komentáře ve skriptu po první prázdný řádek
- o **Sekce**
  - skript je možné rozdělit do celků tzv. sekcí, které lze spouštět samostatně
  - sekce začíná znakem %% a zahrnuje příkazy až k dalšímu znaku %%
    - vložení: *EDITOR* > *Section* > *Section Break* nebo ručně zapsat %%
  - jednotlivé sekce lze spouštět tlačítka ze záložky *EDITOR*
    - *Run and Advance*  ... spustí sekci a přejde na další
    - *Run Section*  ... spustí sekci bez přechodu k další sekci

### ⇒ P01 v2

```
%% Grafy funkci
%
% - sinus
```



```
%% priprava dat
x = 0:pi/20:2*pi;
y = sin(x);
y = sin(x);

%%
figure      % nove graficke okno
plot(x,y)   % vykresleni grafu
```

⇒ spouštět po sekcích

### Pozn.:

Spuštění ručně vybrané části kódu v příkazové řádce ⇒ myší vybrat (označit) úsek kódu a stisknout klávesu **F9**

⇒ označit figure a plot(x,y) > stisknout F9

- **Aktuální figure**
  - když mám otevřeno více oken figure, kde se projeví editace
    - graf (plot) se vykreslí vždy v aktuálním figure = v okně figure, na které se naposledy kleplo myší, nebo bylo vyvoláno
    - vyvolání konkrétního okna
      - vytvoření: figure(1)
      - následné zavolání figure(1) udělá tento figure aktuálním
- **Více grafů v jenom okně figure**
  - do jednoho souřadného systému (axesu)
    - ve výchozím stavu dojde po novém zavolání příkazu plot ke smazání dosavadního grafu a vykreslení nového
    - pro vykreslení nového grafu přes starý
      - *hold on*
      - *hold off* ... vypnutí vrátí původní funkčnost s mazáním
    - *barvy se přiřazují automaticky nebo je zadá uživatel (plot(x,y,'r'))*

⇒ P01 v3

```
% Grafy funkcí
%
%      - sinus a cosinus

% priprava dat
x = 0:pi/20:2*pi;
y = sin(x);
y1 = cos(x);

figure % nove graficke okno

% vykresleni grafu
plot(x,y)
hold on
plot(x,y1)
```

- *Insert > Legend*, dvojklik na legendu, upravit text

- více samostatných grafů (axesů) v jednom figure
  - `subplot(a,b,c)`
    - rozdělí figure na  $a \times b$  polí a vezme aktuální  $c$ -té pole – po řádcích lineárně – do kterého vykreslí souřadný systém pro graf
    - `>> subplot(2,3,3)`

	1	2	3
2	4	5	6
		3	

- pro další subplot ve figure ~ `subplot(a,b,d)`
- pro zpětné zaktualnění  $c$ -tého pole ~ znovu `subplot(a,b,c)`


### ⇒ P01 v4

```
% Grafy funkcí
%
% - sinus a cosinus

% příprava dat
x = 0:pi/20:2*pi;
y = sin(x);
y1 = cos(x);

figure % nové grafické okno

% vykreslení grafu
subplot(2,1,1)
plot(x,y) % doplnění - specifikace cary: plot(x,y,'b*:')
hold on
subplot(2,1,2)
plot(x,y1) % doplnění - specifikace cary: plot(x,y1,'ro-')
```

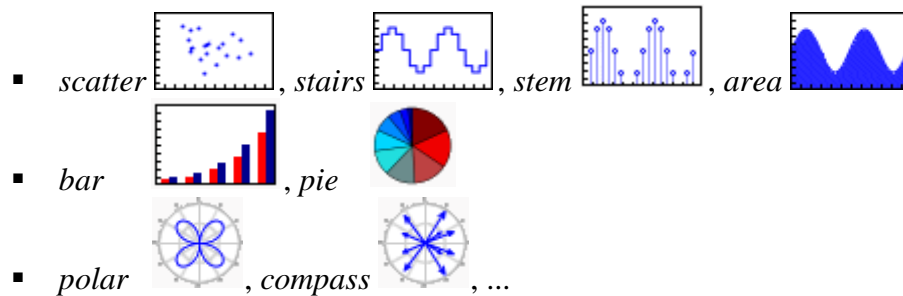
- subplot vždy vymezuje oblast ve figure "znovu"
  - oblasti nemusí být stejně velké
  - překryté grafy se odstraní
  - `>> subplot(2,2,3)`
- změna polohy a velikosti osy editační šipkou 
- Nové grafy lze přidávat ke stávajícím také interaktivně v nástrojích *Plot Browser* a *Figure Palette*
  - *View > Figure Palette, Plot Browser*

### Pozn.:

Zavření všech oken figure příkazem `close all`.

## **Další typy 2-D grafů**

- záložka PLOTS



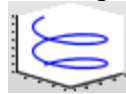
### • Doplnkové nástroje

- statistické údaje o zobrazených datech
- fitování polynomem a interpolace
- poklepat na soubor `fit_data.mat`
- `>> plot(x,y,'r+')`
- *Figure > Tools > Data Statistics*
- *Figure > Tools > Basic Fitting > rozbalit šipkou hodnoty parametrů proložení, zaškrtnout linear, quadratic, ..., Show equations, Plot residuals, ...*

## 3-D grafy

### ○ Parametricky dané křivky

- souřadnice bodů v prostoru  $x, y, z$

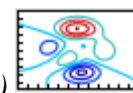
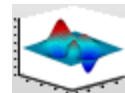
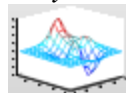


- `plot3(x,y,z)`
- př. šroubovice s proměnným poloměrem

⇒ P02

```
t = 0:pi/20:50*pi;
x = t.*cos(t);
y = t.*sin(t);
z = t;
plot3(x,y,z)      % 3D graf
grid
```

### ○ Plochy - $z$ je funkcí $x$ a $y$



- `mesh(x,y,z)` , `surf(x,y,z)` , `contour(x,y,z)`
- $x, y, z$  ... jsou matice o stejném rozměru

- souřadnice  $x_s = [1 \ 2 \ 3]$ ,  $y_s = [0.3 \ 0.4 \ 0.5]$

$$x = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{bmatrix}, y = \begin{bmatrix} 0.3 & 0.3 & 0.3 \\ 0.4 & 0.4 & 0.4 \\ 0.5 & 0.5 & 0.5 \end{bmatrix}, z = \begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix}$$

- pokud máme jen vektory  $x$  a  $y \Rightarrow$  funkce `meshgrid`

⇒ P03

```
x = -2*pi:pi/10:2*pi;
y = -2*pi:pi/5:2*pi;

[X,Y] = meshgrid(x,y);
R = sqrt(X.^2+Y.^2);
Z = sin(R)./R;

% Plot

subplot(2,2,1)
mesh(X,Y,Z);

subplot(2,2,2)
surf(X,Y,Z);

subplot(2,2,3)
contour(X,Y,Z);

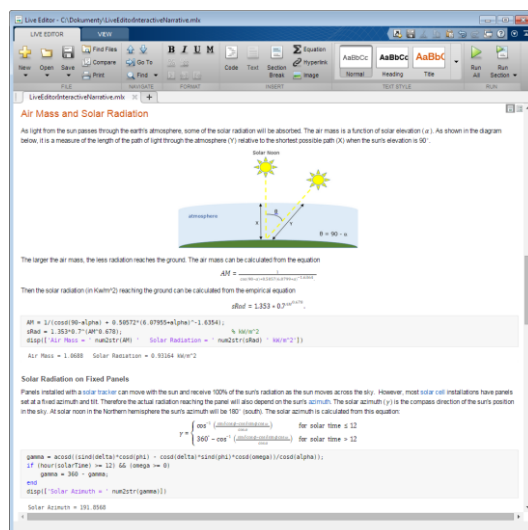
subplot(2,2,4)
contour3(X,Y,Z);
```

## ▪ Colormap

- barevná mapa = přiřazení barev hodnotám v ose z
  - výběr přednastavených barevných map
  - možnost úpravy nebo tvorba vlastních barevných map
- GUI
  - pro celé okno figure: **Property Inspector > klik na figure > Plotting > změnit colormap**
  - pro jednotlivé axes: **Property Inspector > klik na axes > Color and transparency maps > změnit colormap**
- příkaz `colormap`
- `>> colormap hot`

**Live Editor a Live Script**

- od verze R2016a
- Live Editor umožňuje tvorbu dokumentů obsahujících výsledky a grafické výstupy společně s příslušnými výpočty: *Live Script*
- dokument lze doplnit formátovanými texty, odkazy, obrázky a rovnicemi
- *založení nového dokumentu*
  - **HOME > New > Live Script**
- hotový dokument je možné využívat v MATLABu nebo jej exportovat do HTML, PDF, Word či LaTeXu
  - **LIVE EDITOR > Export > Export to PDF...(HTML.../ Word.../ LaTeX...)**
- klasický *Script* lze převést na *Live Script* a opačně
  - převod mezi standardním skriptem a Live skriptem
    - **Save > Save As > Uložit jako typ \*.m, \*.mlx**



⇒ Př.: MujLiveScript.mlx (otevřít)

HOME > New Live Script

Na Toolstripu přibudou záložky LIVE EDITOR, INSERT a VIEW. Přidáme

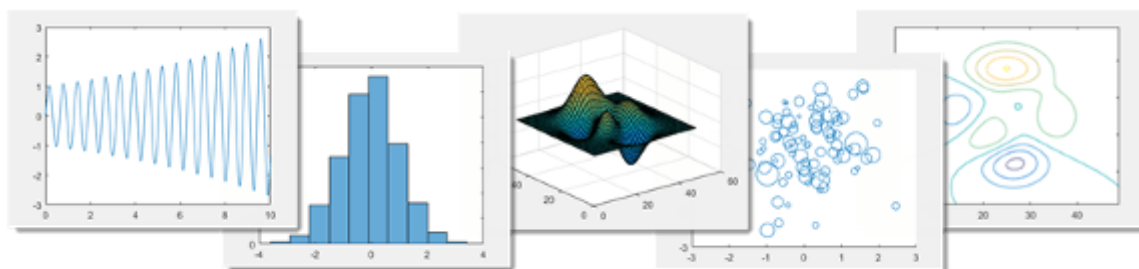
nadpis „Grafy“ a zarovnáme na střed: LIVE EDITOR > Text,  > Title, ,

obrázek a konec sekce: INSERT > , INSERT > ,

podnadpis „3D Graf“: LIVE EDITOR > Text,  > Heading 1,

kód: LIVE EDITOR > ,

## Grafy

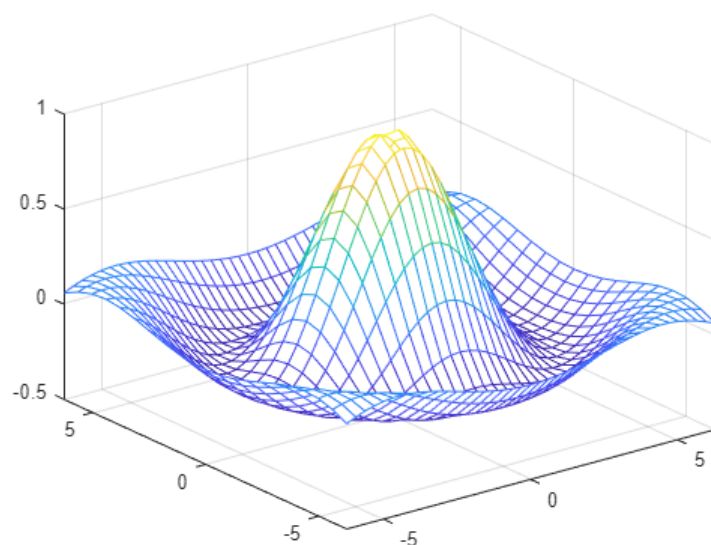


### 3D Graf

```
x = -2*pi:pi/10:2*pi;
y = -2*pi:pi/5:2*pi;

[X,Y] = meshgrid(x,y);
```

```
R = sqrt(X.^2+Y.^2); % zabrání delení nulou a vzniku NaN  
Z = sin(R)./R;  
mesh(X,Y,Z);
```



# Základní datové typy používané v MATLABu

## Číselné datové typy

- Defaultně jsou všechny numerické hodnoty datového typu *double*
  - číslo s pohyblivou řádovou čárkou ve dvojnásobné přesnosti (*double-precision floating-point*), definováno ve Standardu IEEE 754
  - *eps* ... relativní přesnost datového typu *double*, rozdíl 1 a nejbližšího většího čísla
- Krajní případy aritmetických operací
  - `>> 1/0` ... dělení konstanty nulou
    - *Inf* ...  $\infty$
  - `>> -1/Inf`
    - *-0*
  - `>> 0/0` ... dělení nuly nulou
    - *NaN* ... není číslo, není ani rovno samo sobě
  - konst/0 nebo 0/0 není v MATLABu error
    - $\Rightarrow$  je definováno jako *Inf* nebo *NaN*
  - s *Inf* i *NaN* lze počítat
    - lze se na ně dotazovat a takto ošetřit jejich výskyt v programu
  - `>> sqrt(-1)` ... odmocnina ze záporného čísla
- Práce s komplexními čísly:
  - *i, j* ... imaginární jednotka

### Poznámka:

Pozor!: předdefinované hodnoty (*pi*, *i*, *j*, ...) lze "přebít" zadáním proměnné stejného jména s jinou hodnotou

```
>> i = 2
>> v = 2 + 2i    ... zůstane imaginární jednotka
>> v = 2 + i     ... je 4, počítá s i = 2  $\Rightarrow$  psát v = 2 + 1i
```

- *real*, *imag*
  - *abs*, *angle* ... polární souřadnice
    - `>> abs(v)`
- číselné datové typy v MATLABu
  - *double*, *single*
  - *int8*, *int16*, *int32*, *int64*
  - *uint8*, *uint16*, *uint32*, *uint64*
  - *logical*
- nastavení datových typů:
- `>> A = [1 2; 3 4]` ... ač je A tvořeno celými čísly, datový typ A je *double*
- `>> A = int8([1 2; 3 4])` ... nyní má A datový typ *int8*
- `>> Z = zeros(3, 'logical')` ... Z je pole logických hodnot, datový typ *logical*

## Datový typ *logical* a logické operace

- Skalární datový typ *logical* může nabývat dvou hodnot:
  - *true* ... 1 (*logical*)
  - *false* ... 0 (*logical*)
- vektor, matice a vícerozměrné pole datového typu *logical* má v každém prvku jednu z těchto dvou hodnot

- Logické operátory, spojky a další funkce
  - Logické operátory
    - $<$ ,  $>$ ,  $>=$ ,  $<=$ ,  $==$ ,  $\sim$
  - Logické spojky
    - $\&$  ... logický součin (AND)
    - $|$  ... logický součet (OR)
    - $\sim$  ... negace (NOT)
  - Logické "dotazy"
    - *isnan*, *isinf*, *isempty*, *isequal*, ... kontrola různých věcí: *is\**
  - Logické operátory pro všechny prvky vektorů a matic
    - *any* ... log. 1 když je alespoň jeden prvek vektoru nenulový
    - *all* ... log. 1 když jsou všechny prvky vektoru nenulové
- Logické operátory a spojky mohou operovat nad skaláry, vektory, maticemi
  - v případě vektorů a matic se pracuje s odpovídajícími prvky
  - $==$  vs. *isequal*
    - $==$  porovnává odpovídající prvky
    - *isequal* posuzuje rovnost celého pole

## Indexování maticí logických hodnot

- indexujeme maticí logických hodnot (1 a 0) stejného rozměru, jako matice, do které indexujeme
  - $>> v = 1:5;$
  - $>> v(\text{logical}([1\ 0\ 0\ 1\ 0]))$ 
    - vypíše jen hodnoty odpovídající jedničkám
  - takto nemá smysl používat (jednodušší je standardní indexování)
- využití při výběru dle logických podmínek
  - $>> M = [1\ 2\ 3; 4\ 5\ 6; 7\ 8\ 9; 10\ 11\ 12]$  ... z *Command History*
  - $>> k = M < 5$ 
    - vrátí matici log. hodnot stejného rozměru jako matice M
      - 1 na místech, kde je podmínka splněna
      - 0 na místech, kde podmínka splněna není
    - lze využít jako logický index
      - $>> M(k)$  ... indexování logickým indexem
      - $>> M(M < 5)$  ... zkrácený zápis

matice M				indexování šedých polí					
sloupec									
lineární index		1	2	3	M([1 4], [1 2])		k:		
řádek	1	(1)	(5)	(9)	M([1 4 5 8])		1	1	0
	2	(2)	(6)	(10)	M(k)		0	0	0
	3	(3)	(7)	(11)			0	0	0
	4	(4)	(8)	(12)			1	1	0

Porovnání všech tří typů indexování



- Indexy nenulových prvků matice (= prvků splňujících podmínku) - příkaz *find*
  - `>> doc find`

### Pozn.:

Indexování pomocí logických hodnot lze kombinovat s indexováním (řádek, sloupec) nebo i s lineárním indexováním. Např.:

- `>> idx = M(:, 2) < 8;`
- `>> M(idx, :)`

### ⇒ Úloha:

- Vytvořte náhodný vektor s 10-ti prvky s hodnotou mezi 0 a 1 (funkce *rand*) a najděte prvky mezi hodnotami 0,4 a 0,8.

### ⇒ Řešení:

- `>> x = rand(10,1)`
- `>> k = x > 0.4 & x < 0.8`
- `>> x(k)`

### Význam znaků:

.	desetiny, položky struktur
..	systémové příkazy – o úroveň výše
...	pokračování na dalším řádku
a,	oddělovač: indexů, položek v řádku pole, parametrů ve funkcích
a;	oddělovač řádků v maticích, potlačení výpisu výsledku do příkaz. řádky
a', 'a'	transpozice matice, zadání řetězce
()	priorita ve výrazu, indexování matic, vstupní parametry funkcí
[]	tvorba matic, přiřazení více výstupních proměnných, definice funkcí
{ }	pole buněk
:	tvorba číselné řady
+ - * /\	aritmetické a maticové operace
^	umocnění
.* ./ .^	operace "prvek po prvků"
= < >	přiřazovací příkaz =, porovnávací operace <, >, <=, >=, ==, ~=
&   ~	AND, OR, NOT
!	systémový příkaz

%	komentář
%%	oddělení sekce ve skriptu

## Přehled dalších (zajímavých) funkcí

- **Seznam funkcí MATLABu:** *Help > MATLAB > Functions*
- **Interpolace**
  - *interp1, interp2, interp3, interpn ... lineární, 2-D, 3-D, ...*
  - *griddata ... interpolace bodových dat plochou*
- **Řešení rovnic a optimalizace**
  - *fzero ... kořeny nelineární funkce*
  - *fminsearch, fminbnd ... hledání minima funkcí*
  - *\ ... řešení lineární úlohy ve smyslu nejmenších čtverců*
- **Numerická integrace, derivace**
  - *integral ... numerická integrace*
  - *integral2, integral3 ... numerický dvojný trojný integrál*
  - *diff ... diference, použitelné pro aproximaci derivace*
- **Polynomy**
  - reprezentovány vektory  $[a_2 \ a_1 \ a_0] \approx a_2 x^2 + a_1 x + a_0$
  - funkce pro práci s polynomy: pracují s vektorem jako s polynomem
    - *polyval ... vyhodnocení polynomu v daném bodě*
    - *conv ... násobení polynomů*
    - *roots ... kořeny polynomu*
    - *polyint, polyder ... integrace a derivace polynomu*
    - *polyfit ... proložení bodů polynomem daného řádu*
- **Řešení diferenciálních rovnic**
  - *ode\* ... numerické řešení zadané soustavy obyčejných diferenciálních rovnic*
  - *ode45, ode23, ode113, ode15s, ode23t, ode23s, ...*
- **Filtrace dat**
  - *filter ... filtrace datového vektoru zadaným filtrem*
- **Fourierova transformace**
  - *fft, ifft ... rychlá fourierova transformace, zpětná f. transformace*
- **Statistické funkce**
  - *min, max, mean, median, mode, std, var, movmin, movmax, movmean, movmedian, movstd, movvar, movmad, movprod, movsum, cov, corrcoeff, cummax, cummin, mink, maxk, topkrows*
- **Předzpracování dat**
  - *ismissing, fillmissing, rmmissing, standardizeMissing, isoutlier, filloutliers, rmoutliers, ischange, islocalmin, islocalmax, smoothdata, detrend, normalize, rescale, discretize, groupsummary, grouptransform, histcounts, histcounts2, findgroups, splitapply, rowfun, varfun, accumarray*
- **Zaokrouhlování**
  - *round, fix, floor, ceil*
- **Aktuální datum a čas a související funkce**

- *datetime, clock, now, date, datestr, datenum, datevec, etime*

## Řízení toku programu - cyklus a podmínka

### Podmínkový příkaz v MATLABu

<code>if podmínka</code>	<code>if podmínka</code>
<code>    příkazy;</code>	<code>    příkazy;</code>
<code>end</code>	<code>elseif podmínka</code>
	<code>    příkazy;</code>
	<code>else</code>
<code>if podmínka</code>	<code>    příkazy;</code>
<code>    příkazy;</code>	<code>end</code>
<code>else</code>	
<code>    příkazy;</code>	
<code>end</code>	

- podmínky
  - `<, >, >=, <=, ==, ~=`
  - `&, |, ~`
  - *isnan, isinf, isempty, ...* kontrola různých věcí: `is*`
  - *any, all ...* alespoň jeden prvek/všechny prvky vektoru nenulové

⇒ Př: Příklad\_if.m

```
% ax^2 + bx + c = 0

a = input('Zadej a: '); % MATLAB ceka na vstup z prikazove radky
b = input('Zadej b: '); % MATLAB ceka na vstup z prikazove radky
c = input('Zadej c: '); % MATLAB ceka na vstup z prikazove radky

D = b^2-4*a*c;

if D == 0
    disp('Koren je dvojnásobný.')
elseif D > 0
    disp('Koreny jsou realne různé.')
else
    disp('Koreny jsou komplexní.')
end
```

### Cyklus For

- cyklus se známým počtem opakování

<pre>for <i>idx</i> = začátek:krok:konec     příkazy; end</pre>	<pre>for <i>idx</i> = vektor_hodnot     příkazy; end</pre>
---	--

- Př.  $x_1 = konst.$ ,  $x_i = 0.9x_{i-1}$ ,  $i = \langle 2;100 \rangle$

⇒ Př. Příklad\_for.m

```
x(1) = 1;

for k = 2:100
    x(k) = 0.9*x(k-1);
end

plot(x)
```

## Cyklus While

- cyklus s neznámým počtem opakování
- provádí se dokud platí podmínka

```
while podmínka
    příkazy;
end
```

⇒ Př. Příklad\_while.m

```
x = [];
a = input('Zadej a: ');

while a ~= 0
    x = [x,a];
    a = input('Zadej a: ');
end

disp(x)
```

## Přerušení jednoho průběhu cyklu, ukončení celého cyklu

- *continue*, *break*
- **Ctrl + C** ... přeruší probíhající příkaz

## Rozdělovač

- rozdělení do daného počtu větví podle hodnoty řídicí proměnné

```
switch proměnná

    case hodnota1

        příkazy;

    case hodnota2

        příkazy;

    otherwise

        příkazy;

end
```

⇒ Př: Příklad\_switch.m



```
s = input('Zadej příbuzného: ','s');

switch s

    case 'otec'
        disp('Je rodič.')
    case 'matka'
        disp('Je rodič.')
    otherwise
        disp('Není rodič.')
end
```

## Funkce v MATLABu

~ funkcím se podrobně věnuje školení MATLAB 2

- **Vytvoření**
  - ze skriptu
    - definice hlavičky v prvním řádku
    - *function [výstupní parametry] = jmeno\_funkce(vstupní\_parametry)*
      - hlavička definuje vstupní a výstupní proměnné
      - výstupní proměnné musí být ve funkci přiřazeny (vyčísleny)
  - New  >  Function
    - vytvoří prázdný soubor s předdefinovanou strukturou funkce
- **Stejně jako u skriptu:**
  - komentáře od prvního řádku až do prvního vynechaného řádku
    - nápověda, která se zobrazí po zadání `>> doc jmeno_funkce`
- **Soubor musí být uložen pod stejným jménem jako je jméno funkce**
- **Volání funkce z MATLABu**
  - stejně jako vestavěné funkce
  - `>> [out1, out2, ...] = jmeno_funkce(in1, in2, ...)`
- **Funkce má lokální workspace**
  - předané vstupní proměnné a proměnné vytvořené uvnitř funkce nejsou viditelné zvenčí a po skončení funkce zanikají

- o uvnitř funkce nejsou vidět proměnné z hlavního workspace
- o parametry dovnitř a ven z funkce se předávají pomocí hlavičky
  - hlavička definuje jednoznačné rozhraní funkce ⇒ přehlednost

⇒ Příklad:

```
function [zn,an] = signum(n)

%SIGNUM   Znamenko a absolutni hodnota.
%
% [ZN,AN] = SIGNUM(N) vraci znamenko ZN a absolutni hodnotu AN cisla
N.

zn = sign(n);
an = abs(n);


o >> signum(5)
o >> signum(-5)
o >> [z,a] = signum(-5)
o >> doc signum
```

- Řada interaktivních nástrojů umožňuje generovat funkci
  - o *Figure, Import Tool* aj.
  - o nemusíme opakovaně provádět stejné interakce
  - o Př.: Generování funkce z okna *Figure*
    - dvojklik na soubor *mujGraf.fig* v *Current Folderu* > otevře okno *Figure*
    - *File > Generate Code*
    - vytvoří funkci, pomocí které lze vytvořit stejně nastavený graf s novými daty
      - generovat kód > (změnit jméno na *mujfigure*) > uložit > zavolat:  
o >> *mujfigure(x,y.^2)*

## Publikování skriptů a funkcí

- Kód funkcí a skriptů lze automaticky dokumentovat = zachytit texty (komentáře), kód programu, výsledky výpočtů a grafické výstupy do dokumentu zvoleného formátu
- **Záložka PUBLISH**
  - o možnosti pro formátování textu dokumentu (Bold, Italic, ... , Preformatted Text, Code, Display LaTeX)
  - o nastavení – co zahrnout do dokumentu, v jaké formě, ...
    - rozbalovací nabídka Publish > Edit Publishing Options...
- **Formát výstupních souborů**
  - o html, xml, latex, doc, ppt a pdf

⇒ Př.: *fourier\_demo\_publish.m* (otevřít)

- **Run ...** zobrazí výsledky v MATLABu
- **záložka PUBLISH > Publish**  ... grafický dokument s nadpisy, výsledky, grafy, ...



## Přizpůsobení prostředí MATLAB – Preferences

- Nastavení MATLABu
  - HOME >  Preferences

### Př.:

Od verze R2021b se změnil vzhled Editoru a přibylo v něm automatické doplňování závorek a uvozovek či zobrazení nápovědy na stisknutí klávesy *Tab*. Toto doplňování jde zrušit či nastavit v *Preferences > Editor/Debugger > Automatic Completions*.

## Kde najít informace o MATLABu

- dokumentace
  - >> doc, F1, >> help, tab,  
  - pdf dokumentace – lze stáhnout z webu ([www.mathworks.com](http://www.mathworks.com))
  - nejen syntaxe funkcí, ale také teorie, odkazy na použitou literaturu, ...
- web [www.mathworks.com](http://www.mathworks.com)
- MATLAB Central – uživatelská komunita
- www semináře – webinars – natočená videa, seznam on-line webinářů do budoucna = ukázkové semináře Čj, Sj, Ang, ...
- publikace
- elektronický časopis MATLAB Digest