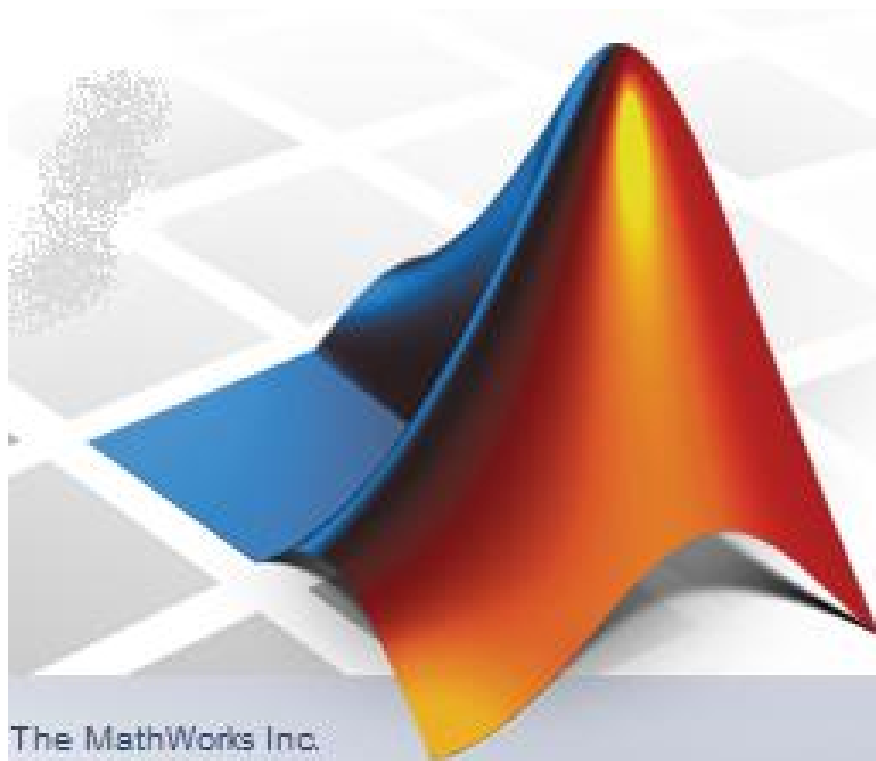


# MATLAB (1) - úvod do programovania vedeckých problémov



## Program predmetu:

1. týždeň: úvod, základné info o Matlabe, pracovné prostredie Matlabu, interaktívny režim, prvé info o písaní skriptov
2. týždeň: základné operácie s maticami, import a export dát, základné grafické zobrazovanie (grafy a mapy)
3. týždeň: práca s reťazcami, práca so súbormi
4. týždeň: pokročilejšia grafika - popis grafov a máp, 2D grafy
5. týždeň: príkazy, stavba programov, M-súborov
6. týždeň: funkcie – zabudované v Matlabe, tvorba vlastných funkcií
7. týždeň: príklady programovania úloh z oblasti prírodných vied
8. týždeň: príklady programovania úloh z oblasti prírodných vied
9. týždeň: tvorba vlastných aplikácií, práca s GUI (Graphical User Interface)
10. týždeň: tvorba vlastných aplikácií, nástroj GUIDE

*pozn.: zmeny vyhradené*



## Obsah (5. prednáška):

- základy tvorby M-súborov
- základné príkazy pre cykly a podmienky
- dôležité príkazy - polynomicke fitovanie funkcií
- dôležité príkazy - interpolácia hodnôt (1D prípad)
- trošičku z grafiky (komunikácia cez okná)
- príklady tvorby skriptov (s použitím cyklov)
- príklady základnej práce so štatistikou
- zadanie č.4



## Tvorba M-súborov alebo tzv. skriptov (programov) v prostredí Matlab:

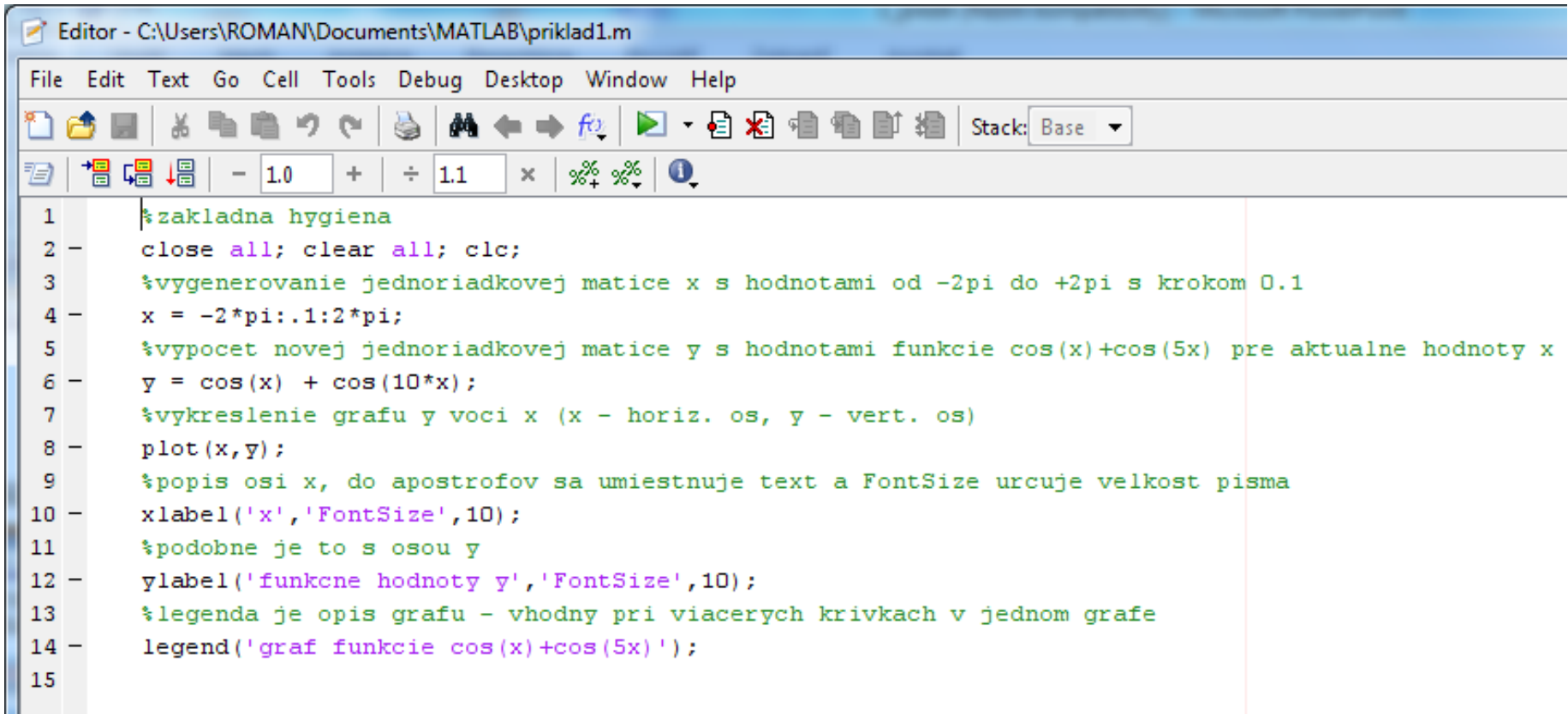
Doteraz sme pracovali najmä v tzv. interaktívnom režime, t.j. písaním alebo prekopírovaním príkazov do Command Window. Tento spôsob je však ťažkopádny, keď ide o dlhšie programy a ich odladovanie. Preto je možné pracovať s textovými súbormi (s príponou .M), ktoré obsahujú postupnosť matlabovských príkazov a ktoré sú realizované pri spustení skriptu.

M-súbory je najlepšie tvoriť a upravovať v Editore Matlabu, je však možné tvoriť (nie spúšťať) v akomkoľvek textovom editore.

M-súbory delíme na:

- **skripty**: neprímajú vstupné a nevracajú výstupné argumenty, pracujú s údajmi premenných vo workspace,
- **funkcie**: prijímajú vstupné údaje a vracajú výstupné údaje, sú väčšinou „volané“ nejakým iným skriptom (ďalšia prednáška).

## Editor (skriptov/programov):



The screenshot shows the MATLAB Editor interface. The title bar reads "Editor - C:\Users\ROMAN\Documents\MATLAB\prik1ad1.m". The menu bar includes File, Edit, Text, Go, Cell, Tools, Debug, Desktop, Window, and Help. The toolbar contains various icons for file operations, editing, and execution. Below the toolbar is a numeric keypad with buttons for digits 0-9, mathematical operators, and a stack display showing "Base". The main editing area contains a script with 15 lines of code. Comments are in green, and code is in black. The script initializes the environment, generates a vector x, calculates a cosine function, and plots it with labels and a legend.

```
1      %zakladna hygiena
2 -    close all; clear all; clc;
3      %vygenerovanie jednoriadkovej matice x s hodnotami od -2pi do +2pi s krokom 0.1
4 -    x = -2*pi:.1:2*pi;
5      %vypocet novej jednoriadkovej matice y s hodnotami funkcie cos(x)+cos(5x) pre aktualne hodnoty x
6 -    y = cos(x) + cos(10*x);
7      %vykreslenie grafu y voci x (x - horiz. os, y - vert. os)
8 -    plot(x,y);
9      %popis osi x, do apostrofov sa umiestnuje text a FontSize urcuje velkost pisma
10 -   xlabel('x','FontSize',10);
11     %podobne je to s osou y
12 -   ylabel('funkcne hodnoty y','FontSize',10);
13     %legenda je opis grafu - vhodny pri viacerych krivkach v jednom grafe
14 -   legend('graf funkcie cos(x)+cos(5x)');
15
```

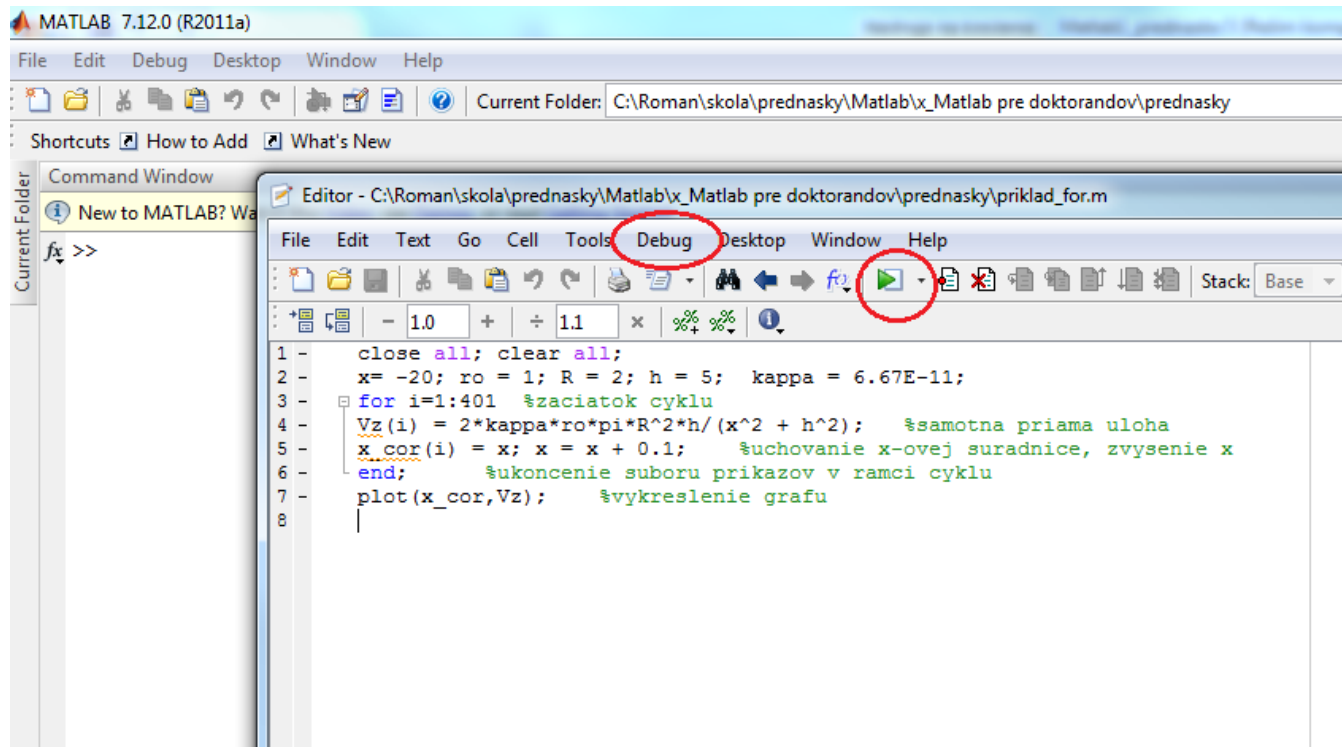
- poznámky sú zelenou, reťazce fialovou,...
- pri písaní matematických vzťahov so zátvorkami editor automaticky upozorňuje na uzavretie párov zátvoriek (tým, že na chvíľku zabliká predchádzajúca zátvorka ku danej aktuálne napísanej)

# MATLAB (1) - úvod do programovania vedeckých problémov, LS 2017

## Tvorba M-súborov alebo tzv. skriptov (programov) v prostredí Matlab:

### Spúšťanie M-súborov:

- stlačením klávesy **F5**,
- napísaním príkazu `run` `nazov_suboru` do Command Window,
- vyvolaním položky Debug/Run v menu Editor,
- stlačením zelenej šípky v paneli nástrojov Editor.



## Tvorba M-súborov alebo tzv. skriptov (programov) v prostredí Matlab:

Často sa stane, že sa v M-súbore pomýlime pri zadávaní príkazov.

V M-súboroch sa môžu vyskytovať dva hlavné typy chýb:

- syntaktické chyby: použitie zlého mena príkazu, zabudnuté zátvorky a pod.
- chyby vzniknuté behom programu: väčšinou vyskytnutá chyba v algoritme, neuskutočniteľný výpočet a pod.

V Command window sa užívateľovi objavuje chybové hlásenie (**červenou farbou**), opisujúce druhy chyby a číslo riadku M-súboru

Command Window

```
??? Undefined function or method 'numelx' for input arguments of type 'double'.
```

```
Error in ==> slapy at 10
```

```
m = numelx(cas);
```

```
>> |
```

**programowanie - cykly a podmienky**



## niektoré dôležité príkazy:

cykly: (*for*, *while*)

```
for variable = initval:endval
    statement
    ...
    statement
end
```

**opakuje súbor príkazov,  
kým sa hodnota *variable*  
nezmení z *initval* na *endval***

príklad:

```
for i=1:10 % i sa mení od 1 po 10
disp(i);   %vypise hodnotu i
end;       %koniec suboru prikazov
```

```
while condition
    statement
    ...
    statement
end
```

**opakuje súbor príkazov,  
kým platí podmienka  
condition**

príklad:

```
my = 1; % pociatocna hodnota my
while (1+my) > 1 % zaciatok cyklu
    my = my/2; % my sa mení
disp(my);     %vypise hodnotu my
end;          %koniec cyklu
```

Cykly je možné do seba vnárať (v rámci jedného cyklu sa nachádza ďalší...).

**V Matlabe je však možné veľmi efektívnym spôsobom využiť bodkovú syntax a „zhutniť“ mnohé cykly do jedného riadku. Príklady – neskôr.**

## príklad cyklov for, while:

### (cyklus for)

```
%vypocet Gaussovej funkcie
close all; clear all; clc;
x = -20; sigma = 3.5; my = 2;
for i=1:401 %zaciatok cyklu
    Gauss(i) = (1/(sigma*sqrt(2*pi)))*exp(-(x-my)^2/(2*sigma^2));
    x_cor(i) = x; x = x + 0.1; %uchovanie x-ovej suradnice, zvyssenie x
end; %ukoncenie suboru prikazov v ramci cyklu
plot(x_cor,Gauss); %vykreslenie grafu
```

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

### (cyklus while)

```
close all; clear all; clc;
%priama uloha pre 2D horizont. valec v gravimetrii
x = -20; sigma = 3.5; my = 2; i = 1;
while x<=20.1 %zaciatok cyklu
    Gauss(i) = (1/(sigma*sqrt(2*pi)))*exp(-(x-my)^2/(2*sigma^2));
    x_cor(i) = x; x = x + 0.1; i = i + 1; %uchovanie x-ovej suradnice
end; %ukoncenie suboru prikazov v ramci cyklu
plot(x_cor, Gauss); %vykreslenie grafu
```

### alebo (Matlabovsky “hutny” zapis bez pouzitia cyklu)

```
close all; clear all; clc;
x = -20:0.1:20; sigma = 3.5; my = 2; i = 1;
Gauss = (1/(sigma*sqrt(2*pi)))*exp(-(x-my).^2/(2*sigma^2));
plot(x, Gauss); %vykreslenie grafu
```

## niektoré dôležité príkazy:

podmienky: (*if*, *switch*)

```
if expression1
    statements1
elseif expression2
    statements2
else
    statements3
end
```

**vykoná príkazy** statements1,  
**keď je splnená podmienka**  
expression1;  
**keď splnená nie je, tak**  
**vykoná príkazy** statements3;  
**(ešte sa tam dá dopĺňať**  
**dodatočná podmienka s elseif)**

```
switch switch_expr
case case_expr
    statement, ..., statement
case {case_expr1, case_expr2, ...}
    statement, ..., statement
otherwise
    statement, ..., statement
end
```

**prepína medzi viacerými**  
**možnosťami výrazu** switch\_expr  
**a realizuje ten príkaz, ktorý**  
**zodpovedá danému prípadu**  
**(zohľadnenému v case);**  
**ak nenájde žiaden prípad, tak**  
**vykoná príkazy pod otherwise)**

## niektoré dôležité príkazy:

podmienky: (*if*, *switch*)

S týmito príkazmi sú veľmi úzko spojené  
**relačné** a **logické** operátory:

`A < B`  
`A > B`  
`A <= B`  
`A >= B`  
`A == B`  
`A ~= B`

| Inputs |   | and   | or    | not | xor      |
|--------|---|-------|-------|-----|----------|
| A      | B | A & B | A   B | ~A  | xor(A,B) |
| 0      | 0 | 0     | 0     | 1   | 0        |
| 0      | 1 | 0     | 1     | 1   | 1        |
| 1      | 0 | 0     | 1     | 0   | 1        |
| 1      | 1 | 1     | 1     | 0   | 0        |

## príklad použitia podmienky if: (nájdete ako skript prikklad\_if.m)

```
%vypocet Gaussovej funkcie
close all; clear all; clc;
x = -20:0.1:20; sigma = 3.5; my = 2; i = 1;
Gauss = (1/(sigma*sqrt(2*pi)))*exp(-(x-my).^2/(2*sigma^2));
plot(x, Gauss);      %vykreslenie grafu
```

```
%pokracovanie - pouzitie podmienky if ku vytriedeniu dat
%pomocna indexova premenna k
    k = 1;
for i=1:401 %zaciatok cyklu
    if Gauss(i) > 0.1
        vyssia_uroven(k) = Gauss(i);
        %kedze bola splnena podmienka, musi byt k zvyssene o 1
        k = k + 1;
    end; %ukoncenie podmienky if
end;      %ukoncenie suboru prikazov v ramci cyklu
%naprava premennej k
k = k - 1;
figure, plot(vyssia_uroven);
```

```
%v prostredi Matlab je mozne cely tento cyklus naradiť 1.riadkom!
vyssia_uroven2 = Gauss(Gauss>0.1); %v zatvorke je podmienka
figure, plot(vyssia_uroven2,'g');
```

## príklad použitia podmienky *switch*:

Príklad:

```
close all; clear all; clc;  
method = 'Bilinear';  
switch lower(method)  
    case {'linear','bilinear'}  
        disp('Method is linear')  
    case 'cubic'  
        disp('Method is cubic')  
    case 'nearest'  
        disp('Method is nearest')  
    otherwise  
        disp('Unknown method.')  
end
```

Ďalší (zložitejší) príklad bude uvedený ďalej v prednáške.

Pozn.: Funkcia `lower ( )` berie všetky znaky v reťazci ako malé.

## dôležité príkazy – fitovanie polynómu:

Fitovanie polynómu cez skupinu dvojíc hodnôt (metódou najmenších štvorcov – LeasSquares, LSQ):

```
p = polyfit(x, y, n);
```

výsledkom je matica  $p$  s koeficientami polynómu stupňa  $n$

$$p_1x^n + p_2x^{n-1} + \dots + p_nx + p_{n+1}$$

matica  $p$  má  $n+1$  prvkov  
(posledný prvok je konštantný člen).

Pre  $n = 1$  ide o známu lineárnu regresiu.

Pozor! Tento prístup sa líši od často používanej aproximácie:  
 $a_0 + a_1x + a_2x^2 + a_3x^3 + \dots$ , kde prvý prvok je konštantný člen.

$$p_1x^n + p_2x^{n-1} + \dots + p_nx + p_{n+1}$$

Príklad: (nájdete ako M-skript: fit\_polyn.m)

```
clear all; close all; clc;
x = (0: 0.1: 2.5)'; %vygenerovanie matice x
y = erf(x); %chybova funkcia (error function)
p = polyfit(x,y,5); %samotne nafitovanie polynomom 5.stupna
m= numel(x)
for j = 1:m
% do matice fit(j) sa vygenerujú hodnoty polynomickej aproximácie
fit(j) = p(1)*x(j)^5+ p(2)*x(j)^4+ p(3)*x(j)^3 + p(4)*x(j)^2 + ...
        p(5)*x(j) + p(6);
end;
plot(x,y, x,fit);
xlabel('x[]','FontSize',10); ylabel('functions []','FontSize',10);
legend(' error function ' , 'polynomial fit');title('prve riesenie');

%alebo (namiesto cyklu for)
fit2 = p(1)*x.^5 + p(2)*x.^4 + p(3)*x.^3 + p(4)*x.^2 + ...
        p(5)*x.^1 + p(6)*x.^0;
figure, plot(x,y, x,fit2);
xlabel('x[]','FontSize',10); ylabel('functions []','FontSize',10);
legend(' error function ' , 'polynomial fit'); title('druhe riesenie');
```



## dôležité príkazy – interpolácia hodnôt (1 D):

Interpolácia hodnôt na základe známych dvojíc hodnôt;  
základná metóda je lineárna interpolácia (metódu je možné voliť):

```
yi = interp1(X,Y,xi)
```

kde X, Y sú matice so známymi hodnotami,  
xi je matica x-ových súradníc pre nové intepolované hodnoty,  
výsledkom je matica nových interpolovaných hodnôt yi.

Metóda sa volí cez parameter method:

```
yi = interp1(X,Y,xi,method)
```

'nearest'

Nearest neighbor interpolation

'linear'

Linear interpolation (default)

'spline'

Cubic spline interpolation

'pchip'

Piecewise cubic Hermite interpolation

'cubic'

(Same as 'pchip')

## interpolácia hodnôt (1 D) - príklad: (nájdete ako M-skript interpol\_1D.m)

### Príklad:

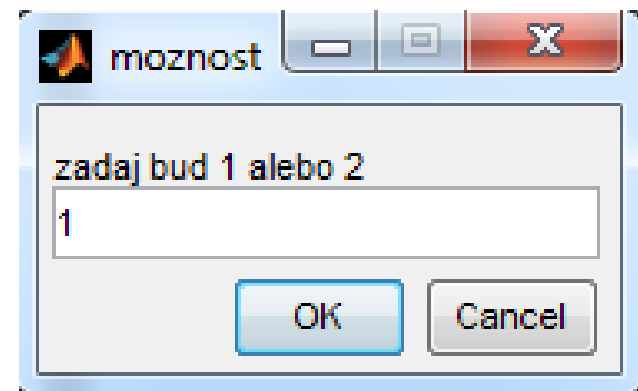
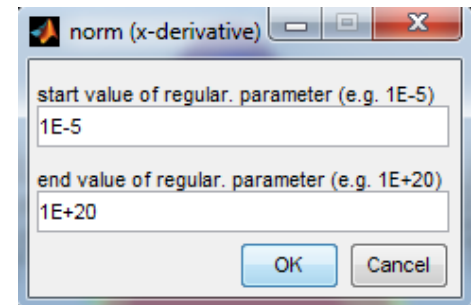
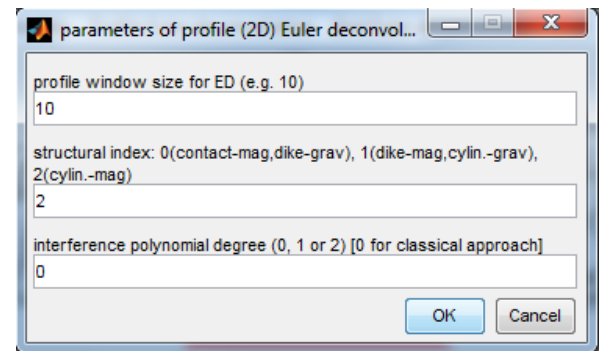
```
close all; clear all; clc;  
%generovanie matice x  
x = 0:10;  
%vypocet matice y (sinx)  
y = sin(x);  
%vygenerovanie novej matice xi, pre ktoru budu  
    interpolovane hodnoty  
xi = 0:.25:10;  
%samotna interpolacia  
yi = interp1(x,y,xi);  
%kreslenie vystupov  
plot(x,y,'o'); legend('povodne udaje');  
figure, plot(xi,yi); legend('interpolovane udaje');  
figure, plot(x,y,'o',xi,yi); title('spolu');
```

Pozn.: Vyskúšajte si podrobnejší interpolačný krok (napr.: 0.05).

## Doplnok ku komunikácii cez okná: príkaz inputdlg() Príklad:

```
close all;clear all;clc;  
%nazov okna  
titlei = 'moznost';  
%text, uvedeny v apostrofoch je zobrazeny  
%ako informacia v okne  
prompti = {'zadaj bud 1 alebo 2'};  
%pocet riadkov pre hodnoty v okne  
linesi = 1;  
%pred-definovana hodnota pre uzivatela  
defi = {'1'};  
%ziskanie odpovede uzivatela  
answer = inputdlg(prompti,titlei,linesi,defi);  
%odpoved z prveho riadku okna  
opergstr1 = answer(1);  
%prevedenie odpovede na ciselnu hodnotu  
moznost = str2double(opergstr1);  
  
disp(moznost);
```

**Pozn.: Upozornenie na nefunkčnosť klávesy Enter  
(pri potvrdení zadania čísla).**



## Využitie cyklu a podmienok – hlavný príklad:

Ako príklad využitia cyklu a podmienok si ukážeme krátky skript na analýzu údajov z meraní prejavov gravitačnej príťažlivosti Mesiaca a Slnka – tzv. tiažové slapy.

1. Údaje sú v súbore `slapy_fakulta_11012009_spikes.dat`: čas, meraná hodnota.
2. Po ich načítaní ich treba analyzovať a vylúčiť chybné („vyskočené“) body s hodnotou 0.1 – nahradiť ich s novými.
3. Nasleduje fitovanie analyzovaných hodnôt pomocou polynómu určitého stupňa (stupeň sa zadáva cez okno).

Samotný program ako M-skript nájdete v súbore `slapy.m`.

**príklady základnej práce so štatistikou**

# Základné štatistické funkcie v prostredí MATLAB:

Základné funkcie pre popisnú štatistiku:

```
hist(A) %nakresli histogram pre maticu A (deli stand. na 10 tried)
hist(A,m) %nakresli histogram pre maticu A s m triedami
[n,v]=hist(A)%vrati pocet vzoriek v intervaloch a ich stredy
    %stredne hodnoty:
mean(A) %vypocita aritmeticky priemer z hodnot prvkov matice A
median(A) %vypocita median z hodnot prvkov matice A
mode(A) %vypocita modus z hodnot prvkov matice A
    %disperzie:
std(A) %vypocita stand. odchylku z hodnot prvkov matice A
var(A) %vypocita varianciu z hodnot prvkov matice A
```

---

## Príklad:

Súbor `organicmatter_one.txt` z učebnice

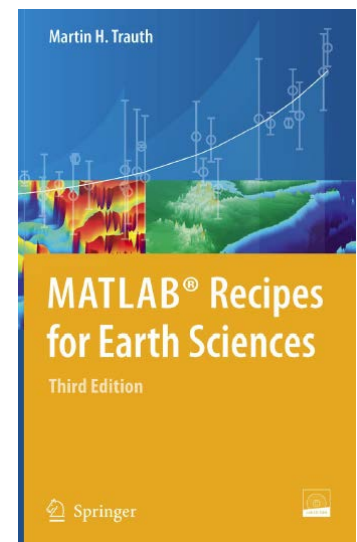
Trauth,M: MATLAB Recipes of Earth science, 2010

obsahuje **pomerné hmotnostné percentá koncentrácie organickej hmoty** v rámci 60 vzoriek odberov jazerných sedimentov (lokalita neznáma...).

Nachádza medzi pracovnými súbormi na túto prednášku.

Tieto údaje vieme načítať do premennej (matice) `corg`

a realizovať s ňou základné štatistické výpočty (ďalší snímok).



## Základné štatistické funkcie v prostredí MATLAB:

Príklad: (nájdete ako skript `zakl_stat.m`)

```
close all; clear all; clc;
corg = dlmread('organicmatter_one.txt');
plot(corg, '-ro', 'LineWidth', 2, 'MarkerFaceColor', 'b', 'MarkerSize', 4);
plot(corg, zeros(1, length(corg)), 'o'); %zaujímavé zobrazenie udajov
figure, hist(corg);
[n,v] = hist(corg);
figure, hist(corg, 8);
% vypocet zakl. stat. charakteristik
corg_mean = mean(corg)
corg_median = median(corg)
corg_mode = mode(corg)
corg_std = std(corg)
corg_var = var(corg)
```

$$s = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2}$$

$$s^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2$$

## Základné štatistické funkcie v prostredí MATLAB:

Ďalšie užitočné funkcie sa nachádzajú  
v **Statistics and Machine Learning Toolbox** – napr.:

`range()` %rozsah  
`quantile()` %vypocet kvantilov (2. kvantil = median)  
`nanmean()` %aritmet. priemer z hodnot prvkov A (bez NaN)  
`nanstd()` %stand. odchylka z hodnot prvkov A (bez NaN)  
`skewness()` %kvantifikacia nesumernosti histogramu (rozdelenia hodnot)  
`kurtosis()` %kvantifikacia tvaru histogramu (rozdelenia hodnot)  
*atd'...*

Ďalej – rôzne druhy rozdelení:

normálne (Gaussovo), Poissonovo, binomiálne (Bernoulliho),...

Prehľad funkcií a ďalších nástrojov

z **Statistics and Machine Learning Toolbox:**

<http://de.mathworks.com/help/stats/functionlist.html>

**Inštalácia na CVTI obsahuje tento toolbox!**



## Zadanie č.4:

Skúste vytvoriť Matlabovský skript, ktorý bude počítat' základné štatistické Parametre (pridajte aj *skewness* a *kurtosis*) z načítaného súboru `slapy_fakulta_11012009_spikes.dat`

Skúste tieto údaje najprv zobrazit' formou grafu, potom zobrazit' histogram ich rozdelenia a napokon do Command Window vypísať hodnoty všetkých vypočítaných štat. parametrov.

Kľudne môžete použiť štýl zobrazenia cez príkaz `subplot()`.

(popíšte všetky osi, hodnoty slapov sú v  $[10^{-9} \text{ m} \cdot \text{s}^{-2}]$ , čas je v hodinách)

$$skewness = \sum_{i=1}^N \frac{(x_i - \bar{x})^3}{s^3}$$

$$kurtosis = \sum_{i=1}^N \frac{(x_i - \bar{x})^4}{s^4}$$

Výsledky vášho výpočtu parametrov *skewness* a *kurtosis* si môžete (nemusíte) skontrolovať pomocou zabudovaných funkcií v rámci inštalácie na CVTI.