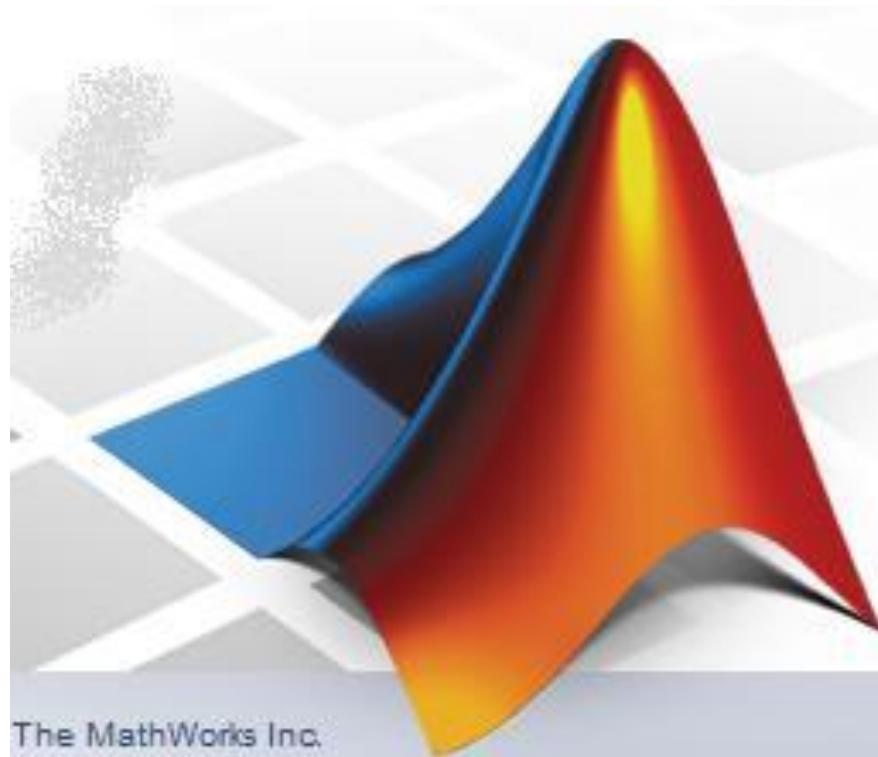


Programovanie geofyzikálnych úloh v prostredí MATLAB

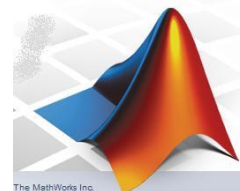


Programovanie geofyzikálnych úloh v prostredí MATLAB

Program predmetu:

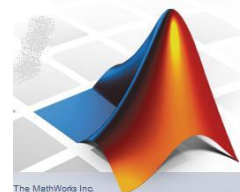
1. týždeň: úvod, základné info o Matlabe, pracovné prostredie Matlab, interaktívny režim, prvé info o písaní skriptov
2. týždeň: základné operácie s maticami, import a export dát, základné grafické zobrazovanie (grafy a mapy)
3. týždeň: príkazy, stavba programov, M-súborov
4. týždeň: práca s reťazcami, práca so súbormi
5. týždeň: pokročilejšia grafika - popis grafov a máp, 2D grafy
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 (3. prednáška):

- základy tvorby M-súborov
- základné príkazy pre cykly a podmienky
- dôležité príkazy - polynomicke fitovanie funkcií
- dôležité príkazy - interpolácia hodnôt (1D prípad)
- delenie skriptu do sekcií (úloha %%)
- trošičku z grafiky (komunikácia cez okná)
- príklady tvorby skriptov (s použitím cyklov)
- príklady základnej práce so štatistikou
- zadanie č.3



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

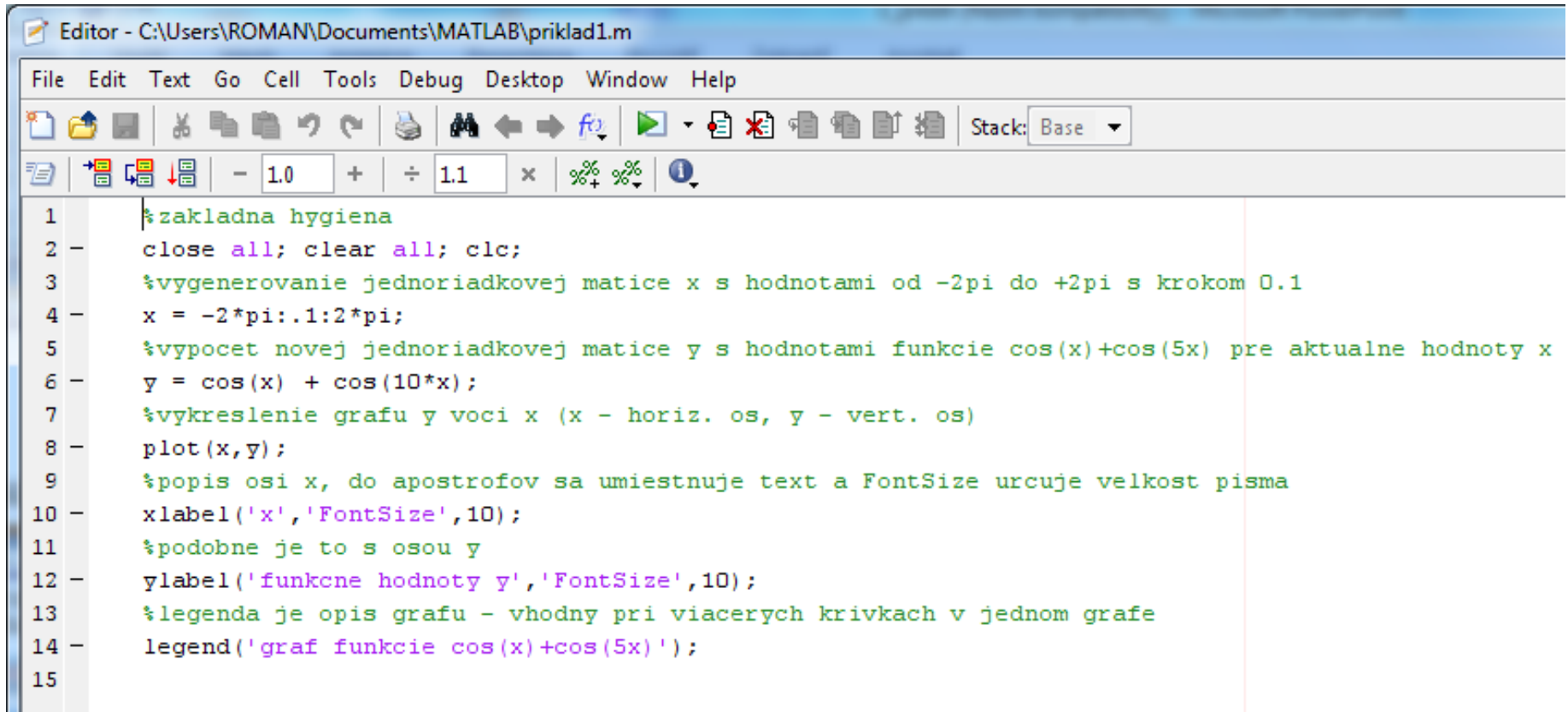
Doteraz sme pracovali často v tzv. interaktívnom režime, t.j. písaním alebo prekopírovaním príkazov do Command Window. Tento spôsob je však ťažkopádny, keď ide o dlhšie programy a ich odladovanie. Preto je možné pracovať s textovými súborami (s príponou .M), ktoré obsahujú postupnosť matlabovských príkazov 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**: hlavné súbory, neprímajú vstupné a nevracajú výstupné argumenty, pracujú s údajmi premenných vo workspace,
- **funkcie**: pomocné súbory, prijímajú vstupné údaje a vracajú výstupné údaje, sú väčšinou „volané“ nejakým iným skriptom (o funkciách budeme hovoriť v ďalšej prednáške).

Editor (skriptov/programov):



The screenshot shows a MATLAB script editor window titled "Editor - C:\Users\ROMAN\Documents\MATLAB\priklad1.m". The window contains a menu bar (File, Edit, Text, Go, Cell, Tools, Debug, Desktop, Window, Help) and a toolbar with various icons. Below the toolbar is a numeric keypad with buttons for digits 1.0, 1.1, and mathematical symbols like +, -, *, /, and %. The main area of the window displays a MATLAB script with 15 lines of code. The code includes comments in green, function calls in blue, and mathematical expressions in black. The script is as follows:

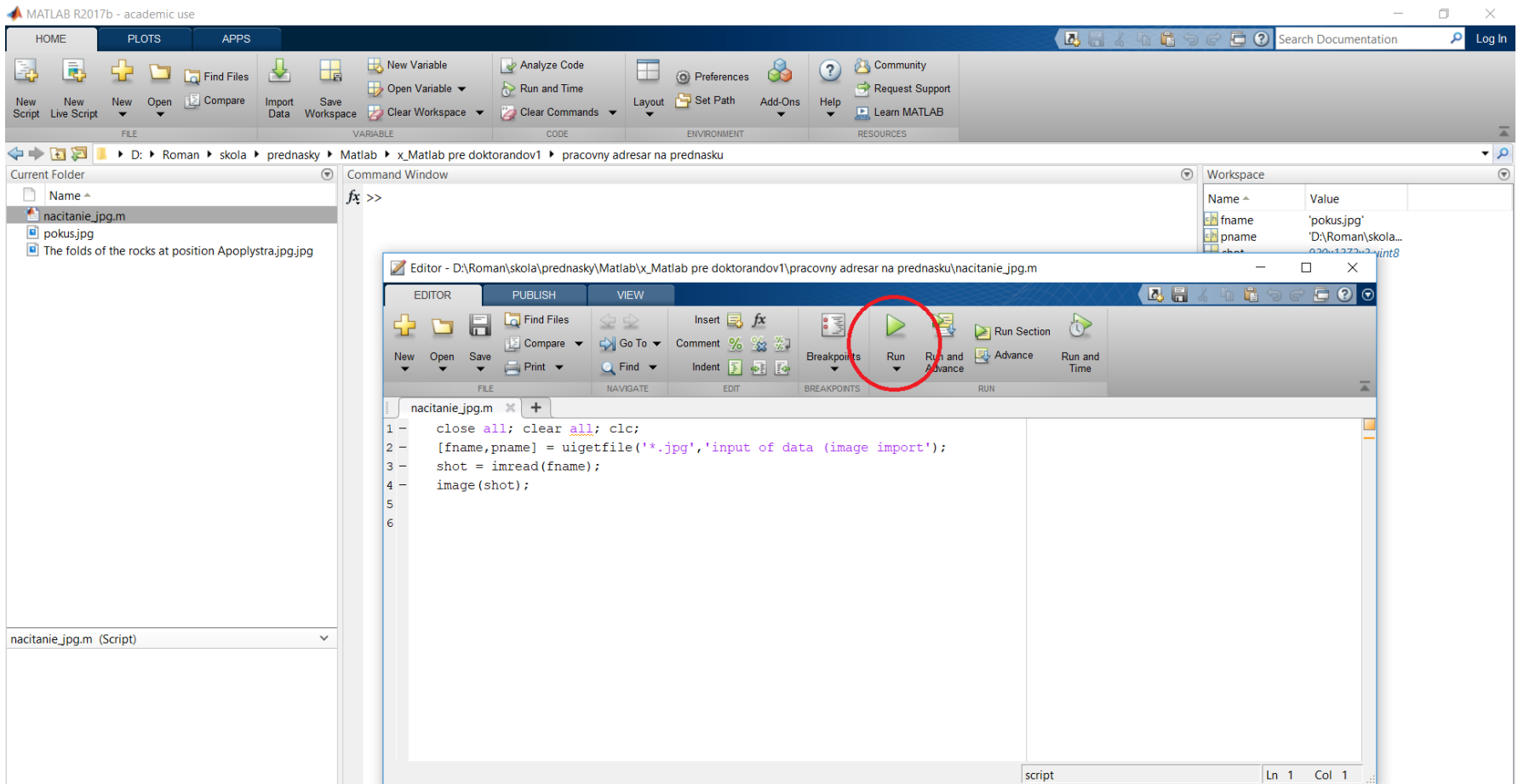
```
1 %zakladna hygiena
2 - close all; clear all; clc;
3 %vygenerovanie jednoriadkovej matice x s hodnotami od -2pi do +2pi s krokom 0.1
4 - x = -2*pi:.1:2*pi;
5 %vypocet novej jednoriadkovej matice y s hodnotami funkcie cos(x)+cos(5x) pre aktualne hodnoty x
6 - y = cos(x) + cos(10*x);
7 %vykreslenie 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, kľúčové slová modrou...
- pri písaní matematických vzťahov so zátvorkami editor automaticky upozorňuje na uzavretie párov zátvoriek (tým, že na chvíľku zabliká predchádzajúca zátvorka ku danej aktuálne napísanej)
- automaticky sa označuje začiatok a koniec cyklu (podmienky nie)
- použitie %% na vyčlenenie častí (sekcí) skriptu – neskôr v tejto prednáške

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

Spúšťanie M-súborov:

- stlačením klávesy **F5**,
- stlačením **zelenej šípky** (Run) v paneli nástrojov okna Editor.
- napísaním príkazu `run` `nazov_saboru` do Command Window,



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

Chybové hlásenia:

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 meni 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 meni
    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 „zhrnúť“ mnohé cykly do jedného riadku. Príklady – neskôr.

príklad cyklov for, while:

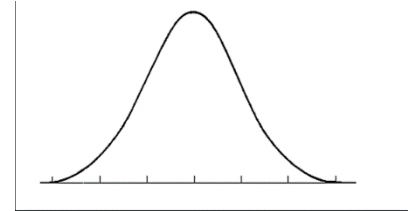
(cyklus for)

```
%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, zvyshenie 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)

```
%vypocet Gaussovej funkcie
close all; clear all; clc;
x = -20; sigma = 3.5; my = 2; i = 1;
while x<=20.1 %zaciatok cyklu
    Gauss2(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, Gauss2); %vykreslenie grafu
```



alebo (Matlabovsky “hutny” zapis bez pouzitia cyklu for alebo while)

```
%vypocet Gaussovej funkcie
close all; clear all; clc;
x = -20:0.1:20; sigma = 3.5; my = 2; i = 1;
Gauss3 = (1/(sigma*sqrt(2*pi)))*exp(-(x-my).^2/(2*sigma^2));
plot(x, Gauss3); %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 naradit 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 poynómu 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 v numerickej matematike: $a_0 + a_1x + a_2x^2 + a_3x^3 + \dots$,
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');
```

Pozn.: M-skript: fit_polyn2.m je rozdelený na bloky pomocou %%.

dôležité príkazy – fitovanie iných (nelineárnych) funkcií:

Je možné v MATLABe riešiť 2 spôsobmi:

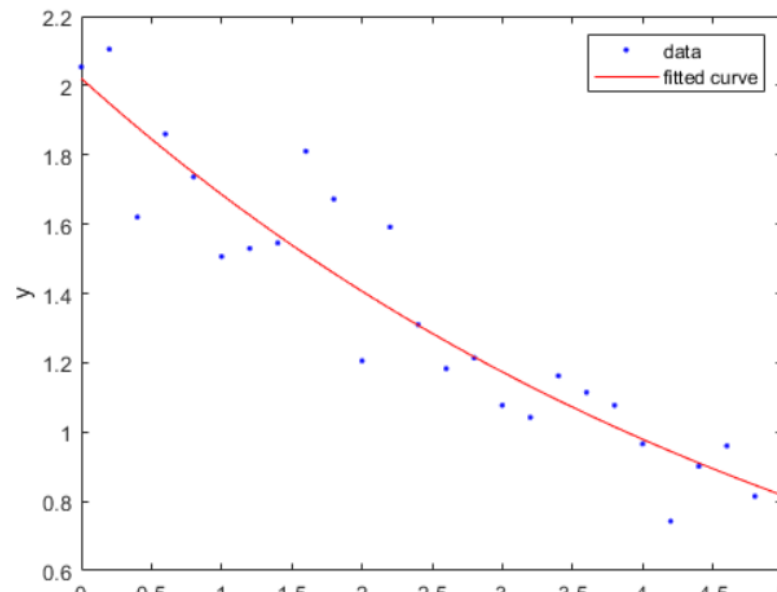
1. Naprogramovať krok po kroku pomocou metódy najmenších štvorcov (LSQ) – toto vedie ku riešeniu tzv. systému normálnych lineárnych rovníc (budúce prednášky)
2. Použiť funkciu `fit()` z Curve Fitting Toolbox

Príklad: fitovanie expon. funkcie

```
x = (0:0.2:5)';  
y = 2*exp(-0.2*x) + 0.1*randn(size(x));  
f = fit(x,y,'exp1')  
plot(f,x,y)
```

f =

```
General model Exp1:  
f(x) = a*exp(b*x)  
Coefficients (with 95% confidence bounds):  
a = 2.021 (1.89, 2.151)  
b = -0.1812 (-0.2104, -0.152)
```



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,

x_i je matica x -ových súradníc pre nové intepolované hodnoty,
výsledkom je matica nových interpolovaných hodnôt y_i .

Metóda sa volí cez parameter `method`:

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

<code>'nearest'</code>	Nearest neighbor interpolation
<code>'linear'</code>	Linear interpolation (default)
<code>'spline'</code>	Cubic spline interpolation
<code>'pchip'</code>	Piecewise cubic Hermite interpolation
<code>'cubic'</code>	(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');
```

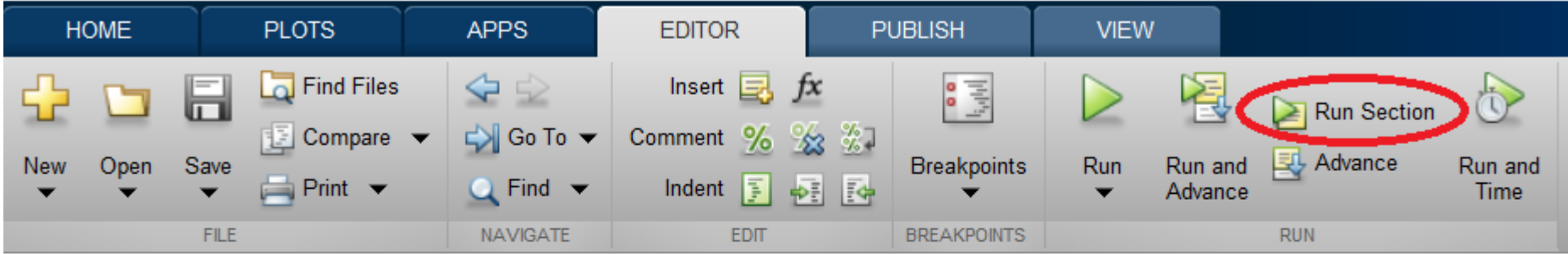
Pozn.: Vyskúšajte si podrobnejší interpolačný krok (napr.: 0.05).

Možnosť delenia skriptu do sekcií (sections, cells)

Pri použití dvoch znakov % % a medzery za tým sa script rozdelí na bloky (sekcie, bunky). Takto je možné script rozdeliť na určité celky. Tieto sa aj farebne rozlíšia v editovanom skripte.

Toto je výhodné pri odlad'ovaní zložitejších skriptov – je možné potom spúšťať samostatne tieto bloky (vyskúšajte si script `fit_poly2.m`).

MATLAB R2017b - academic use



The screenshot shows the MATLAB R2017b interface. The ribbon is set to the 'RUN' tab, and the 'Run Section' button is circled in red. The current folder is 'D:\Roman\skola\prednasky\Matlab\x_Matlab pre doktorandov1\pracovny adresar na prednasku'. The editor window shows the script 'fit_polyn2.m' with the following code:

```
1 - clear all; close all; clc;
2 - x = (0: 0.1: 2.5)'; %vygenerovanie matice
3 - y = erf(x); %chybova funkcia (error functi
4 - p = polyfit(x,y,5); %samotne nafitovanie p
5 - m = numel(x)
6 - %% 1. verzia s cyklom
```

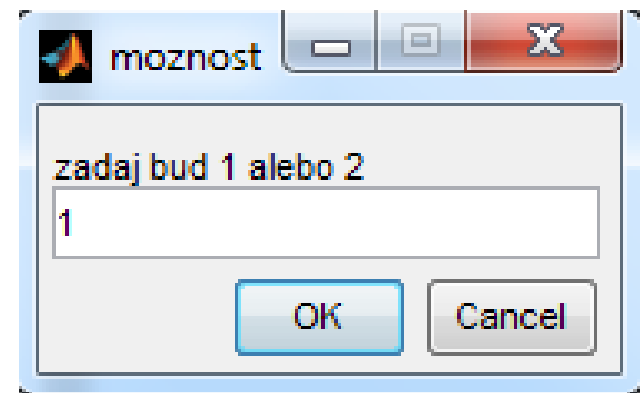
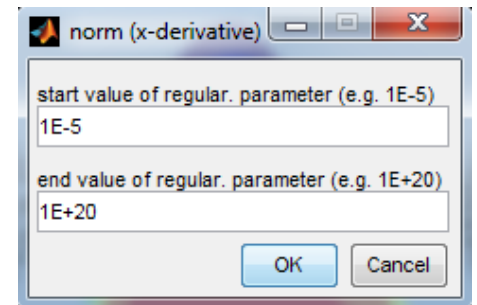
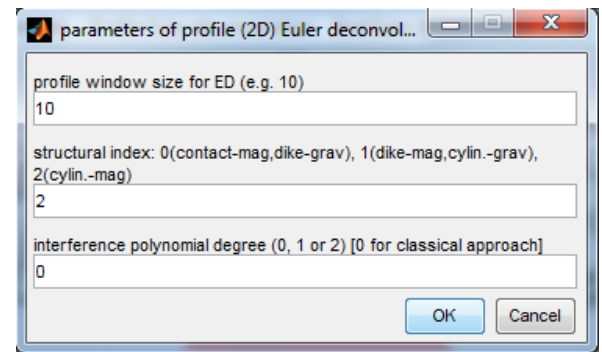
Doplnok ku komunikácii cez okná:

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`.

Využitie cyklu a podmienok – ďalší príklad:

Výpočet gravitačného účinku (pole V_z) pre 2D polonekonečnú dosku (sill) v gravimetrii

Je potrebné vypočítať pole anomálneho gravitačného zrýchlenia (jeho vertikálnej zložky) V_z pozdĺž profilu ponad 2D polonekonečnú dosku (nachádza sa v hĺbke h so začiatkom presne pod bodom $x = 0$). Ďalšie parametre pre dosku sú: plošná hustota μ , ktorá sa vypočíta ako súčin samotnej objemovej hustoty ρ (v tomto prípade negatívnej, keďže sa opisuje účinok sedimentárnej výplne) a hrúbky dosky dh .

Dĺžka profilu je daná načítanými údajmi zo súboru (`Bouguer_data.dat`), kde sú interpolované hodnoty zo slovenskej Bouguerovky pozdĺž profilu v cca Z-V smere cez okraj Malých Karpát približne na úrovni Trnavy.

Vzorec pre výpočet účinku V_z tohto telesa pozdĺž profilu nájdete v prezentáciách z predmetu Gravimetria (1):

$$V_z(x,0) = 2\kappa\mu \left[\frac{\pi}{2} + \text{artctg} \left(\frac{x}{h} \right) \right]$$

Samotný skript nájdete v súbore: `sill_gravimetry.m`

Zadanie č.3:

Upravte (prerobte) existujúci skript "sill_gravimetry.m" tak, aby bol schopný načítať údaje zo súboru "pf1-Gagarinova.dat" (skúste uhádnuť, že odkiaľ sú tieto údaje...?) a porovnať krivku ÚBA s účinkom 2D horizontálnej tyče (pozor, tento krát má vstupný súbor 4 stĺpce: x, y, h, UBA).

Na pripomenutie - priama úloha pre 2D tyč v gravimetrii:

$$V^{(2D)}_z(x,0,0) = 2\kappa\lambda \frac{h}{x^2 + h^2}$$

kde h je hĺbka stredu tyče, λ je tzv. dĺžková hustota = prierez tyče krát hustota ($\lambda = \rho\pi R^2$).