3 Riešenie úloh regresnej analýzy v prostredí MATLAB s využitím dátových súborov

3.1 Práca s binárnymi a textovými súbormi

MATLAB umožňuje spracovanie súborov rôznych formátov. Nemusíme mať dáta uložené priamo v prostredí MATLAB. V tomto programovom prostredí dokážeme exportovať a importovať dáta vo formáte, aký si zvolíme. V tejto kapitole sa budeme venovať súborom typu mat, textovým súborom a xls súborom a prácou s nimi.

- **Ukladanie a načítavanie dát do/z *mat* súborov**

⇨ **Ukladanie dát**

Na ukladanie dát do súboru bola v Matlabe vytvorená funkcia ***save***. Táto funkcia ukladá premenné do binárneho súboru. Meno súboru do ktorého sa dáta uložia vieme ovplyvniť tým, že za názvom funkcie dopíšeme nami zvolený názov súbor. V prípade, že názov nezadáme, Matlab nám premenné uloží do súboru *subor.mat.* Keď chceme do súboru uložiť len niektoré premenné, je potrebné ich špecifikovať za názvom súboru, v opačnom prípade Matlab uloží do súboru všetky premenné Workspace.

Uloženie do súboru	subor.mat	nazov.mat	nazov.mat
Uloženie premenných	všetkých	všetkých	b, c, d
Príkaz	`a=1; b=4; c=3.8; d=b*c;` `save`	`a=1; b=4; c=3.8; d=b*c;` `save nazov.mat`	`a=1; b=4; c=3.8; d=b*c;` `save nazov.mat b c d`

Pri zadávaní premenných, ktoré chceme uložiť je možné použiť aj tzv. žolíky označované znakom hviezdičky (*). Premenné, ktoré majú časť svojho názvu rovnakú nemusíme vypisovať jednotlivo, ale využijeme spomínaného žolíka.

```
a=1; prem1=4; prem2=2.1; b=6; prem3=0.3; prem4=6;
save nazovsuboru.mat prem*
```

Takto zadaný príkaz nám uloží do súboru ***nazovsuboru.mat*** len premenné *prem1, prem2, prem3, prem4*.

MATLAB nám takto zadaným príkazom uloží dáta v **binárnom formáte**. Ďalšími príkazmi vieme zmeniť aj formát a to nasledovne :

```
save nazovsuboru.mat % štandardný binárny formát
save nazovsuboru.mat -ascii % 8-bitový textový výstup
save nazovsuboru.mat -ascii -tabs % 8-bitový textový výstup oddelený tabulátormi
save nazovsuboru.mat -ascii -double % 16-bitový textový výstup
save nazovsuboru.mat -ascii -double -tabs % 16-bitový textový výstup oddelený tablátormi
```

⇨ **Načítavanie dát**

Import dát z Mat súborov je veľmi jednoduchý pomocou príkazu ***load***. Tak ako pri ukladaní aj pri načítavaní môžeme načítať celý súbor, len určité premenné alebo využiť žolíka.

```
load nazovsuboru.mat
load nazovsuboru.mat prem1 prem2
load nazovsuboru.mat prem*
```

Predtým ako si dáta zo súboru načítame, môžeme sa pozrieť aké dáta v danom súbore sú uložené. Príkaz ***whos*** s prepínačom *file* a názvom súboru nám zobrazí názov premenných, veľkosť, počet bytov a triedu.

```
>> whos -file nazovsuboru.mat
  Name        Size              Bytes  Class     Attributes

  prem1       1x1                   8  double
  prem2       1x1                   8  double
  prem3       1x1                   8  double
  prem4       1x1                   8  double
```

- **Ukladanie a načítavanie dát do/z textových súborov**

⇨ **Ukladanie dát**

MATLAB má niekoľko vstavaných funkcií pre ukladanie textových súborov. Ich použitie závisí od množstva ukladaných dát a od formátu súboru, do ktorého chceme dáta uložiť. Pri binárnych súboroch MAT sme využívali najjednoduchšiu funkciu ***save***. Táto funkcia taktiež umožňuje ukladanie do ASCII súborov. Pri takomto uložení dát sa ako oddeľovač používa medzera.

```
a=2; b=3.1; C=[1,2,3;6,8,0];
save nazovsuboru.out -ASCII
```

Funkcia ***dlmwrite*** tiež slúži na ukladanie dát do textového súboru. Na rozdiel do funkcie ***save*** si pri tejto funkcii môžeme sami zvoliť znak slúžiaci na oddeľovanie dát. Oddeľovač sa do príkazu pridáva v úvodzovkách.

```
A=[1 2 8; 6 3 7];
dlmwrite('data.out',A, ';')
```

Ďalšou funkciou slúžiacou na ukladanie textových súborov je funkcia ***csvwrite***, ktorá je primárne určená pre dáta využívajúce tabuľkový procesor. Pri tejto funkcii sa oddeľovač nešpecifikuje, je to vždy čiarka.

```
A=[1 2 8; 6 3 7];
csvwrite('csvdata.out',A)
```

Poslednou funkciou na ukladanie dát do textového súboru je ***diary***. Ukladá výstup príkazového riadku MATLABus.

```
diary data.out
A=[1 5 8 ; 2 4 7; 2 0 1];
A
diary off
type data.out
```

Príkazom uvedeným v poslednom riadku vypíšeme obsah súboru *data.out* do príkazového riadku.

⇨ **Načítavanie dát**

Na načítavanie dát taktiež existuje niekoľko vstavaných funkcií. Výber funkcie, ktorá sa na import dát použije závisí od naformátovanie dát v súbore. Textové dáta musia byť naformátované do rovnakého počtu stĺpcov v jednotlivých riadkoch. Ako oddeľovač dát slúži tzv. oddeľovací znak.

Najjednoduchší spôsob načítania dát je pomocou funkcie ***load***. Táto funkcia sa využíva v prípade keď sú dáta uložené do obdĺžnika (v každom riadku je rovnaký počet stĺpcov). Dáta budú vložené do ***Workspace*** s menom totožným ako je názov súboru bez koncovky. Dáta je možné uložiť aj s názvom, ktorý si užívateľ zvolí sám (prípad v 2. riadku – dáta sa uložia do premennej A)

```
load data.txt
A=load('data.txt')
```

V prípade, že máme ako oddeľovač použitý iný znak ako je medzera, môžeme na načítanie dát použiť funkciu ***dlmread***. Táto funkcia ignoruje medzery medzi dátami. Ako druhý parameter je potrebné zadať znak oddeľovača. Nepovinnými parametrami je posun, od ktorého chceme začať načítavanie. Indexovať sa začína od 0,0. V druhom riadku príkladu je uvedený príkaz na načítanie dát od prvku nachádzajúceho sa v 3. riadoku 2. stĺpci.

```
A = dlmread('data.txt', ';');
A = dlmread('data.txt', ';', 3, 2);
```

Čítanie dát z textových súborov uložených v ***csv*** súboroch sa realizuje funkciou ***csvred***. V takýchto súboroch ako oddeľovač slúži čiarka, preto sa v príkaze parameter oddeľovača nezadáva. Zvyšné parametre zostávajú také isté ako pri funkcii ***dlmread***.

```
A=csvread('data.txt', 2, 3);
```

V prípade, že textový súbor obsahuje hlavičky k dátam, je potrebné použiť funkciu ***textscan***, v tejto funkcii sa môže využiť parameter ***headerlines,*** ktorý umožňuje ignorovanie zadaného počtu riadkov od začiatku súboru.

Majme súbor *data.txt*, v ktorom sú nasledovné dáta :

```
Stĺpec1     Stĺpec2     Stĺpec3     Stĺpec4     Stĺpec5
2.154       1.256       3.785       24.125      2.536
3.256       1.401       2.869       36.254      1.258
3.987       1.569       3.028       12.548      3.685
```

V takomto prípade je potrebné najskôr otvoriť súbor na čítanie. Identifikátor súboru si vložíme do premennej ***idenif***. Funkciou ***textscan*** načítame 3 riadky, znakom ***%f*** udávame, že chceme aby sa údaje uložili v desatinnom formáte. Ďalej využijeme parameter ***headerlines*** a za ním udáme číslo, ktoré nám označuje počet riadkov, ktoré sú v súbore uvedené ako hlavičky. Súbor je potrebné zavrieť a to príkazom ***fclose***.

```
identif = fopen('data.txt','r');
data = textscan(identif, '%f %f %f %f %f',3,'headerlines',1);
fclose(identif);
```

Nie všetky textové súbory musia obsahovať výhradne číselné premenné. Uvedieme si príklad súboru v ktorom sa nachádzajú premenné rôzneho dátového typu. Majme súbor ***data.txt*** s nasledujúcim obsahom:

```
Vysoký      elektrikár    512.65    2    12    Áno
Chladná     lekárka       1254.35   1    10    Áno
Hravý       účtovník      845.23    2    15    Nie
Nováková    učiteľka      658.12    3    20    Áno
```

Takto usporiadané údaje je možné načítať pomocou funkcie ***textread*** takto:

```
>> [meno, profesia, mzda, deti, odpracovane_roky, odpoved]=...
textread('data.txt','%s %s %f %d %d %s', 4)
```

Údaje zo súboru sa rozdelia do jednotlivých premenných – meno, profesia, mzda.... Typy týchto premenných určujú prepínače (*%s* – reťazec znakov, *%f* – reálne číslo, *%d* – celé číslo). Posledný parameter udáva počet riadkov, ktoré sa majú spracovať.

- **Ukladanie a načítavanie dát do/z xls súborov**

⇨ **Ukladanie dát**

Funkcia ***xlswrite*** slúži na ukladanie dát vo formáte *xls*. Pri použití funkcie je potrebné zadať názov súboru, do ktorého sa budú dáta ukladať a maticu ***m x n*** dát, ktorá sa bude ukladať. Matlab má obmedzenie pre veľkosť takejto matice. Rozmer *m* nesmie prekročiť hodnotu 65536 a rozmer *n*

nesmie prekročiť hodnotu 256. Matica môže obsahovať číselne a reťazcové hodnoty. Údaje sa zapíšu do prvého hárka na pozíciu A1. Pridaním parametrov vieme ovplyvniť názov hárka, do ktorého sa údaje uložia a pozíciu buniek od ktorej po ktorú sa údaje budú zapisovať (prvá bunka označuje ľavý horný roh, druhá označuje pravý dolný roh). Obidva spomínané parametre zadávame ako reťazec.

```
M={'júl','teplota vzduchu','teplota vody';...
1,31,23;2,29,22;3,33,24;4,31,25;5,30,21;6,34,24};
xlswrite('Zosit.xls',M);
xlswrite('Zosit.xls',M,'júl');
xlswrite('Zosit.xls',M,'júl','B3:D8');
```

⇨ **Načítavanie dát**

Na načítavanie dát zo súboru xls bola vytvorená funkcia ***xlsread***. Táto funkcia je určená na čítanie číselných údajov. Funkcia vracia maticu čísel dátového typu double.

```
A=xlsread('Názov_súboru.xls');
% načítajú sa údaje zo súboru Názov_súboru
% z prvého hárku od bunky A1
B=xlsread('Názov_súboru.xls','názov_hárku');
% načítajú sa údaje zo súboru Názov_súboru
%z hárka názov_hárku  od bunky A1
C=xlsread('Názov_súboru.xls','názov_hárku','B3:D8');
% načítajú sa údaje zo súboru Názov_súboru
%z hárka názov_hárku  od bunky B3 po bunku D8
```

3.2 Príklady práce so súbormi v prostredí MATLAB

- **Načítavanie výstupných dát z programového prostredia MATLAB**

a) Export premenných z pracovného priestoru do binárneho súboru.

*T*ento úkon je možný použitím funkcie ***save.*** MATLAB ukladá premenné do špeciálneho binárneho súboru s príponou .mat.

```
save('názov súboru','premenná1 ','premenná2 ',...,'premennáN ');
save  názov_súboru premenná1  premenná2 ...premennáN;
```

Ak nie je špecifikovaná cesta, potom je súbor uložený do aktuálneho adresára.

Príklad 1
Vytvorte 3 premenné, prvá bude skalár, druhá vektor a tretia matica. Tieto premenné uložte do súboru s názvom „premenne.mat“.

Riešenie v programovom prostredí MATLAB:

```
>> x=1;
>> y=5:2:25;
>> A=[1 2 3; 4 5 6; 7 8 9];
>> save('premenne','x','y','A')
```

b) Export dát do textového súboru v ASCII formáte

S týmito súbormi je možné neskôr pracovať v akomkoľvek textovom editore.
Funkcia ***save*** ukladá vybrané premenné do textového súboru

```
save('názov súboru','premenná1 ','premenná2 ',...,'premennáN ','-ascii' );
save  názov súboru premenná1  premenná2 ...premennáN  '-ascii';
```

Príklad 2
Premenné vytvorené z príkladu 1 uložte do súboru s názvom „ascii“, vo formate ascii kódu.

Riešenie v programovom prostredí MATLAB:

```
>> save('ascii','x','y','A','-ascii')
```

Funkcia ***dlmwrite –*** ukladanie numerický dát do textového súboru, pričom je možné špecifikovať typ oddeľovača medzi jednotlivými prvkami.

```
dlmwrite ('názov súboru',X,'Oddeľovač')
```

Názov súboru je zadávaný spolu s príponou (napríklad „.txt“, „.dat“). Oddeľovačom môže byť akýkoľvek znak(obvykle medzera, tabulátor a pod.)

Príklad 3
Vektor z príkladu 1 uložte do textového súboru s názvom *pokus.txt*, pričom ako oddeľovače použite tabulátor:

Riešenie v programovom prostredí MATLAB:

```
>> dlmwrite('pokus.txt', y, '\t')
```

Funkcia ***fprintf*** **–** ukladanie formátovaných dát do textových súborov
Funkcia ***fopen, fclose*** **–** otvorenie / zatvorenie súboru

```
F = fopen(`nazov suboru`,`w`);  % otvorenie súboru funkciou
fprintf(F, format, X);          % uloženie dáta z matice
fclose(F);                      % zatvorenie súboru
```

Názov súboru je zadávaný aj s príponou napríklad „.txt“ , „.dat“.
Parameter formát je reťazec znakov, ktorý charakterizuje formátovanie dát. Reťazec musí začínať znakom **%**, ďalšie znaky určujú počet zobrazených číslic resp. desatinných miest, formát zápisu a pod.

formát zápisu	
znak	*popis*
%d	*desatinné číslo*
%e	*exponenciálny tvar*
%f	*pevný počet miest*
%s	*reťazec znakov*
špeciálne znaky	
\n	*posun na nový riadok*
\r	*znak nového riadku*
\t	*tabulátor*
%%	*percento*

Príklad 4
Otvorte súbor **pokus.txt,** ktorý sme si vytvorili v príklade 3, pre zápis, vložte do súboru reťazec „toto je skalár: x“, kde x bude predstavovať skalár vytvorený v príklade 1. Nakoniec súbor zatvorte.

Riešenie v programovom prostredí MATLAB:

```
>> f=fopen('pokus.txt','w');
>> fprintf(f,'toto je skalár: %d', x);
>> fclose(f);
```

c) *Export dát do tabuliek*

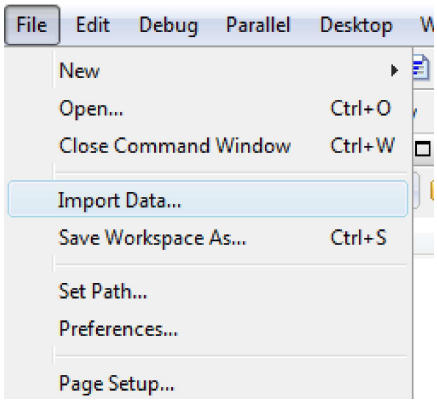
Funkcia ***xlswrite*** vykonáva export dát do tabuliek Excelu

$$xlswrite('nazov\ suboru', X, list, 'pozicia');$$

Príkaz exportuje je dáta v matici ***X*** do tabuľky v Exceli do súboru so zadaným názvom (***názov súboru***), do zadaného ***listu*** a na zadanú ***pozíciu***. Pokiaľ súbor neexistuje, tak je najprv vytvorený.

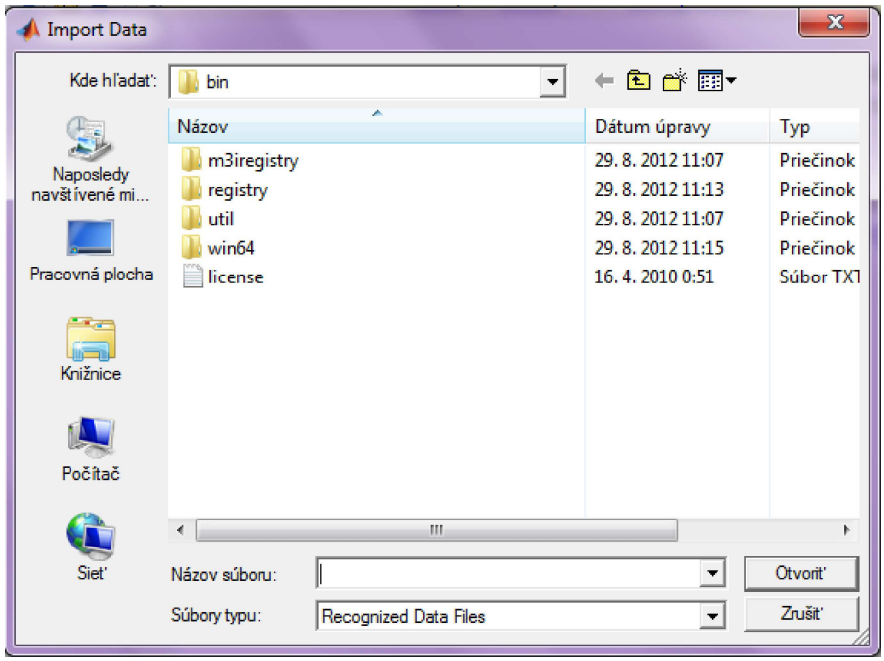
- **Načítavanie vstupných dát do programového prostredia MATLAB**

Jednou z najjednoduchších možností ako načítať vstupné dáta je použiť ***automatický nástroj pre import dát***. Tento nástroj je možné spustiť ***File → Import Data.***



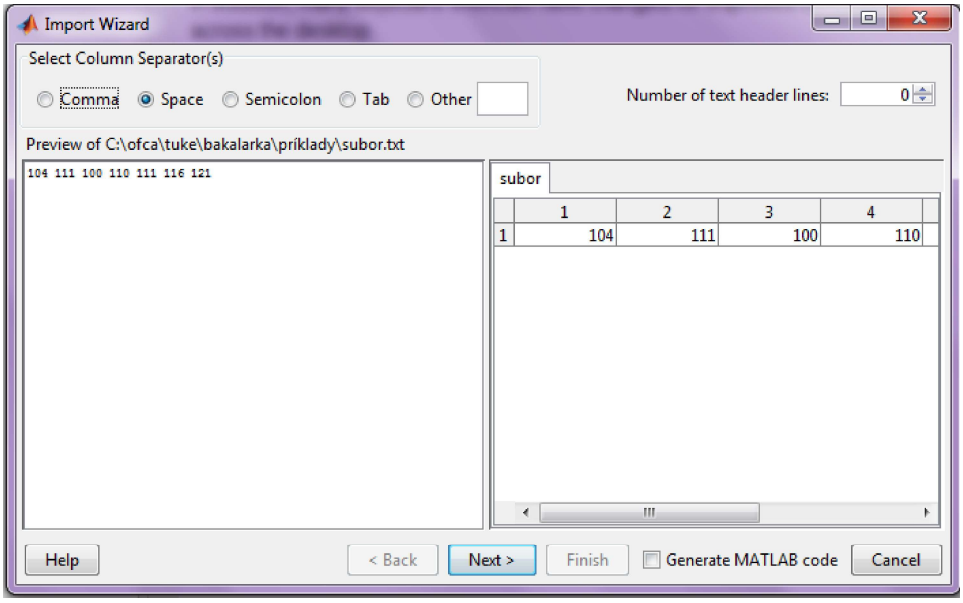
Obrázok 3-1 Spustenie nástroja pre načítanie vstupných dát do MATLABu : File -> Import Data

1) Zobrazí sa dialógové okno pre výber súboru, ktorý chceme importovať.



Obrázok 3-2 Načítanie vstupných dát do MATLABu

2) Po výbere importovaného súboru sa MATLAB automaticky pokúsi rozoznať formát súboru a dáta previesť do odpovedajúcich typov premenných.
3) V prípade textových súborov sa objaví ďalšie okno, kde je možné špecifikovať oddeľovače medzi dátami.
4)



Obrázok 3-3 Špecifikácia oddeľovačov dát

Funkcia pre načítanie dát zo súboru : ***importdata('nazov suboru')***

⇨ automatické načítavanie dát zo súboru, kde parameter ***názov súboru*** musí obsahovať aj príponu,

⇨ pokiaľ sa automaticky na základe typu prípony nepodarí rozpoznať typ dát, potom sú dáta načítane ako text s oddeľovačmi.

*a) Import dát z **mat** súborov*

$$load\,nazov\_suboru;$$

Funkcia ***load*** načíta premenné uložené v danom súbore do pracovného priestoru programového prostredia MATLAB.

b) Import dát z textových súborov

$$X = csvread(\text{´}nazov\ suboru\text{´})$$
$$X = dlmread(\text{´}nazov\ suboru\text{´}, D)$$
$$X = load(\text{´}-ascii\text{´}, \text{´}nazov\ suboru\text{´})$$

c) *Import dát z tabuľkových súborov realizuje funkcia **: xlsread***

$$[X, T] = xlsread(\text{´}nazov\ suboru\text{´}, list, \text{´}pozicia\text{´});$$

⇨ funkcia ***xlsread*** načíta dáta zo zadanej ***pozície a listu*** **zo** zadaného **súboru**, pričom numerické dáta budú načítané do **matice** *X* a textové dáta do ***poľa buniek*** *T*.

Príklad 6
Načítajte data uložené do súboru ***premenne.mat*** do programového prostredia MATLAB.
Riešenie v programovom prostredí MATLAB*:*

```
>> load premenne.mat        % pred použitím funkcie load súbor musí obsahovať data
```

3.3 Operácie s polynómami s využitím funkcií jazyka MATLAB

Polynóm je výraz (súčet alebo rozdiel jednočlenov) v tvare

$$P(x) = \sum_{i=0}^{n} a_i x^{n-i} = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots + a_n x^0 ,$$

kde $a_n \neq 0$ a $a_0, a_1, \ldots, a_n$ sa nazývajú koeficienty polynómu

- **Základné vlastnosti polynómov**

a. dva polynómy sa rovnajú, ak sú rovnakého stupňa a majú rovnaké koeficienty pri každej mocnine
b. dva polynómy sa rovnajú, ak nadobúdajú rovnaké hodnoty pre každé x,
c. nech polynómy $P_n(x)$a $Q_m(x)$; kde $Q_m(x) = b_0 x^n + b_1 x^{n-1} + \cdots + b_{n-1} x^1 + b_n$, kde $n \geq m$ a nech $P_n(x)$a $Q_m(x)$ sú rôzneho stupňa, potom existujú dva jednoznačne určené polynómy $R(x)$ a $Z(x)$, pre ktoré platí $P_n(x) = Q_m(x) * R(x) + Z(x)$, ppolynóm $R(x)$ sa nazýva čiastočný podiel a polynóm $Z(x)$ sa nazýva zvyšok po delení. Stupeň polynómu $Z(x)$ je vždy menší ako stupeň polynómu $R(x)$

V programovom prostredí MATLAB je polynóm zadávaný v podobe vektora koeficientov počnúc koeficientom pri najvyššej mocnine.

- **Operácie s polynómami**

Pri násobení polynómov P, Q platí pravidlo vynásobenia každého s každým prvkom polynómu

$$P_n(x) * Q_m(x) =$$
$$= (a_0 x^n + a_1 x^{n-1} + \cdots + a_{n-1} x^1 + a_n) * (b_0 x^m + b_1 x^{m-1} + \cdots + b_{m-1} x^1 + b_m) =$$
$$= a_0 x^n * b_0 x^m + a_0 x^n * b_1 x^{m-1} + \cdots + a_0 x^n * b_m + a_1 x^{n-1} * b_0 x^m + \cdots$$

Funkcia ***conv*** slúži na vynásobenie dvoch polynómov **P a Q** (zadaných ako vektory):

conv(P,Q)

Príklad 1
Vynásobte dva polynómy, ktoré su dané nasledovne::

$$P(x) = x^3 + 4x^2 + 3,\ Q(x) = 8x^2 - 2$$

Numerický výpočet:

$$P(x) * Q(x) = (x^3 + 4x^2 + 3) * (8x^2 - 2)$$
$$P(x) * Q(x) = 8x^5 + 31x^4 - 4x^3 + 24x^2 - 3x$$

Riešenie v programovom prostredí MATLAB :

```
>> P=[1 4 0 3];
>> Q=[8 -1 0];
>> vysledok = conv(P,Q)

vysledok =

   8   31   -4   24   -3    0
```

Funkcia ***deconv*** slúži na delenie polynómu iným polynómom

[*vysledok, zvysok] = deconv(P,Q)*

pričom polynómy ***P,Q*** musia byť dopredu inicializované. Výstupom sú polynómy vo forme vektorov prísl

Príklad 2
Vydeľte polynómy ***P,Q*** a overte výsledok výpočtu v programovom prostredí MATLAB.

Numerický výpočet :

$$(x^4 - 3x^3 + 6x^2 - 3x^1 + 5x^0) : (x^2 - 3x^1 + 5x^0) = x^2 + x^0$$

$$-(x^4 - 3x^3 + 5x^2)$$

$$0x^4 + 0x^3 + x^2 - 3x^1 + 5x^0$$

$$-(x^2 - 3x^1 + 5x^0)$$

Riešenie v programovom prostredí MATLAB:

```
>> P=[1 -3 6 -3 5];
>> Q=[1 -3 5];
>> [vysledok,zvysok] = deconv(P,Q)

vysledok =

   1   0   1


zvysok =

   0   0   0   0   0
```

Príklad 3
Numericky vypočítajte a následne overte v programovom prostredí MATLAB **delenie dvoch polynómov *P,Q*** so zvyškom.

Numerický výpočet :

$$(x^3 - 2x^2 + x - 1) : (x^2 - 3x + 2) = x + 1 + \frac{2x - 3}{x^2 - 3x + 2}$$

$$-(x^3 - 3x^2 + 2x)$$

$$x^2 - x - 1$$

$$-(x^2 - 3x + 2)$$

$$2x - 3$$

Riešenie v programovom prostredí MATLAB:

```
>> P=[1 -2 1 -1];
>> Q=[1 -3 2];
>> [vysledok, zvysok] = deconv(P,Q)

vysledok =

    1    1


zvysok =

    0    0    2   -3
```

Funkcia ***roots*** sa používa na výpočet koreňov polynómu,

roots(polynom)

Príklad 4
Majme zadaný polynóm $P(x) = x^3 + 4x^2 + x - 6$, nájdite jeho korene pomocou funkcie ***roots***.
Po inicializácii polynómu P= [1 4 1 -6] ako vektora, môžeme volať funkciu ***roots*** na výpočet koreňov

```
>> koren = roots(P)

koren =

   -3.0000
   -2.0000
    1.0000
```

Funkcia ***poly*** vytvára polynóm z zadaných koreňov (vypočíta koeficienty polynómu v klesajúcich mocninách ***x)***

polynom = poly(A)

pričom ***A*** je stĺpcový vektor obsahujúci korene polynómu

Príklad 5
Nájdite koeficienty polynómu ktorého korene sú: $x_1 = -3$, $x_2 = 1$, $x_3 = -2$

Numericky postupujeme tak, že vykonáme súčin koreňových činiteľov:

$$A(x) = (x+3) * (x-1) * (x+2) = x^3 + 4x^2 + x - 6$$

Riešenie v programovom prostredí MATLAB:

```
>> A=[-3;-2;1];
>> polynom = poly(A)

polynom =

    1    4    1   -6
```

Funkcia **polyval -** vypočíta hodnotu polynómu daného stupňa pre dané *x*

polyval(polynom, x);

pričom prvky vektora *polynom* musia byť známe a tiež musíme špecifikovať hodnotu *x*.

Príklad 6
Vypočítajte a následne overte v jazyku MATLAB hodnotu zadaného polynómu ***P(x)*** pre *x* = 2

$$P(x) = x^3 + 4x^2 + x - 6\,, \quad pre \quad x=2$$

$$P(x=2) = 2^3 + 4 * 2^2 + 2 - 6 = 20$$

Riešenie v programovom prostredí MATLAB:

```
>> P=[1 4 1 -6];
>> vysledok = polyval(P,2)

vysledok =

   20
```

- Funkcia ***polyder*** **-** derivácia polynómu
  a. ***polyder****(P)* derivácia polynómu ***P***
  b. ***polyder****(A,B)* alebo ***polyder(conv****(A,B))* derivácia súčinu polynómov ***A*B***
  c. ***polyder****(B,A)* derivácia podielu polynómov ***B/A***

Príklad 7
Vypočítajte deriváciu zadefinovaného polynómu ***P(x)*** . Výsledok overte v jazyku MATLAB s využitím funkcie ***polyder.***

$$P(x) = 9x^5 - 6x^3 + x^2 - 18$$

$$P(x)' = (9x^5 - 6x^3 + x^2 - 18)' = 45x^4 - 18x^2 + 2x$$

Riešenie v programovom prostredí MATLAB:

```
>> P=[9  0 -6 1 0 -18];
>> vysledok = polyder(P)

vysledok =

   45    0  -18    2    0
```

Príklad 8
Vypočítajte deriváciu súčinu polynómov ***A(x)*** a ***B(x)*** pričom sú zadefinované nasledovne :

$$A(x) = x^3 + 4x^2 + 3$$
$$B(x) = 8x^2 - 2$$

Numericky vypočítame súčin polynómov:

$$A(x) * B(x) = (x^3 + 4x^2 + 3) * (8x^2 - 2) = 8x^5 + 31x^4 - 4x^3 + 24x^2 - 3x$$

Výsledok súčinu polynómov ***A(x), B(x)*** následne zderivujeme:

$$\big(A(x) * B(x)\big)' = (\,8x^5 + 31x^4 - 4x^3 + 24x^2 - 3x)' =$$
$$= 40x^4 + 124x^3 - 12x^2 + 48x - 3$$

Riešenie v programovom prostredí MATLAB:

```
>> A = [1 4 0 3];
>> B = [8 0 -2];
>> vysledok = polyder(A,B);

vysledok =

  40   124   -12   48   -3
```

Funkcia ***residue*** – umožňuje rozklad racionálne lomenej funkcie na parciálne zlomky, pričom parametre ***b,a*** majú význam vektorov v ktorých sú uložené koeficienty polynómov čitateľa a menovateľa racionálne lomenej funkcie v klesajúcich mocninách.

[R,P,K] = residue(b,a)

kde :

R – zosilnenia výsledných parciálnych zlomkov
P – stĺpcový vektor koreňov
K – zosilnenie

Príklad 9

Rozložte funkciu $G(x) = \frac{-x+10}{x^2-4}$ na parciálne zlomky, riešte samostatne numericky a výsledok overte v jazyku MATLAB.

$$G(x) = \frac{b(x)}{a(x)} = \frac{-x+10}{(x-2)(x+2)} = \frac{r_1}{x-p_1} + \frac{r_2}{x+p_2} = \frac{r_1}{x-2} + \frac{r_2}{x+2}$$

Riešenie v programovom prostredí MATLAB:

```
>> b=[-1 10];
>> a=[1 0 -4];
>> [R,P,K]=residue(b,a)

R =

     2
    -3


P =

   2.0000
  -2.0000


K =

     []
```

3.4 Príklady na riešenie

1. Naplňte premenné *prem1, prem2, prem3, prem4, prem5, a, b, c, d* číselnými hodnotami. Uložte vytvorené premenné *prem1, prem2, prem3, prem4, prem5, b, d* do súboru *uloha1.mat*. Pri zadávaní premenných využite žolíka.
2. Vynulujte Workspace. Zo súboru *úloha1.mat* načítajte všetky premenné.
3. Vynulujte Workspace. Zo súboru *úloha1.mat* načítajte nasledovné premenné – a,c,d,prem1.
4. Vytvorte maticu 4x2. Uložte ju do textového súboru , v ktorom ako oddeľovač bude slúžiť čiarka.
5. Vynulujte Workspace. Do premennej A načítajte obsah súboru, ktorý ste vytvorili v predošlej úlohe.
6. Vytvorte si textový súbor, ktorý bude obsahovať „tabuľku“ údajov o spolužiakoch, a to : meno, evidenčné číslo, počet zapísaných predmetov v minulom semestri, počet úspešne zvládnutých predmetov v minulom semestri, odpoveď na otázku, či sú spokojný so svojím výberom vysokej školy. Načítajte tieto údaje do Workspace.
7. Vytvorte súbor v programe Microsoft Excel, do ktorého vložíte informácie o meniacom sa kurze americký dolár vzhľadom na eura v čase od 1.2.2013-15.2.2013 (1.stĺpec – poradové číslo dňa v mesiaci február, 2.stĺpec – predaj, 3.stĺpec – nákup). Načítajte tieto údaje do Workspace.
8. Pomocou ľubovoľného príkazu na zobrazovanie grafov zobrazte funkciu $z=\sqrt{(2x^2-y^3)}$, pričom x=3:20 a y=-3:-20. Graf popíšte.
9. POTREBNÉ PRE ĎALŠIE PRÍKLADY
   A) Vytvorte súbor *data1*, kde uložíte premenné:
   x =[-5, -4.5, -3, -2, 0.5, 2.5, 6, 10, 12.5, 16, 20.5, 22]
   y =[0.257, 0.968, 1.254, 1.685, 2.002, 2.685, 2.986, 3.574, 3.867, 4.012, 3.986, 4.876].
   B) Vytvorte súbor *data2*, kde uložíte premenné:
   x=[-4, -2, 0, 2, 4, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 21]
   y=[-0.112, -0.058, 0.218, 0.294, 0.351, 0.486, 0.687, 0.788, 1.007, 1.315, 1.419, 1.225, 1.254, 1.321, 1.457]
10. Aproximujte údaje zo súboru *data1* polynómami 2., 3. a 4. stupňa. Vykreslite do grafu príslušné priebehy. Vypočítajte sumu štvorcov odchyliek pre každú aproximáciu.
11. Aproximujte údaje zo súboru *data1* polynómom *n*- tého stupňa, pričom *n* sa zadá z klávesnice. Aproximované údaje uložte do súboru. Vykreslite do grafu priebeh aproximácie.
12. Interpolujte namerané dáta zo súboru *data2*. Použite tretinový krok vzorkovania a na interpolovanie použite metódu ‘**cubic‘** a **‘linear‘**. Výsledky zobrazte v dvoch grafoch ktoré budú vedľa seba.
13. Načítajte dáta z súboru *data1*. Tieto hodnoty interpolujte zo štvrtinovým krokom vzorkovania a použite na to funkciu interp. Následne interpolujte dáta pomocou metódy ‘spline‘ a zobrazte výsledok v grafe.

3.5 Tvorba grafických objektov v jazyku MATLAB s využitím základných funkcií

Grafické funkcie v programovom prostredí MATLAB automaticky otvoria nové grafické okno, v prípade ak už je nejaké grafické okno otvorené, prekresľujú pôvodné grafické zobrazenie.

- Funkcia ***figure*** sa používa na otvorenie nového okna grafického obrázku. Ak za funkciu ***figure*** zapíšeme do zátvoriek číslo - vytvorí sa grafické okno s číslom, ktoré sme zadali,

  figure;
  figure(x);

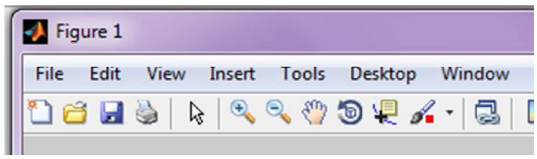
  Syntax

  x = figure;

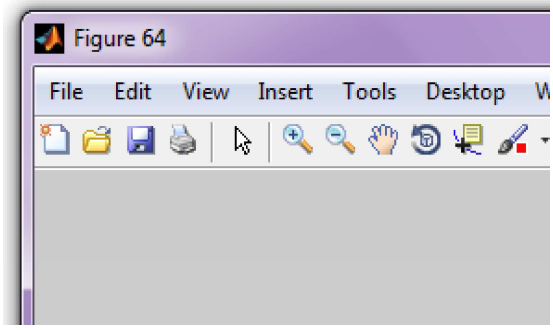
  umožní užívateľovi po otvorení okna uložiť jeho číslo

Funkcionalita funkcie ***figure*** v prostredí MATLAB je znázornená na grafických oknách

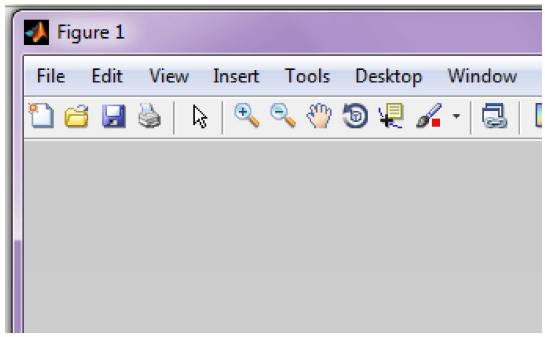
```
>> figure
```




```
>> figure(64)
```




```
>> x =figure

x =

    1
```



- Funkcia ***close*** sa používa uzatváranie grafických okien. Ak použijeme funkciu close bez parametrov, uzatvára sa momentálne aktívne okno. Ak zatvoríme okno, aktuálnym sa stane okno vytvorené pred uzatvorením okna. Ak otvoríme nové okno, považujeme ho automaticky za aktívne.

  close(x), close(´nazov´),close all

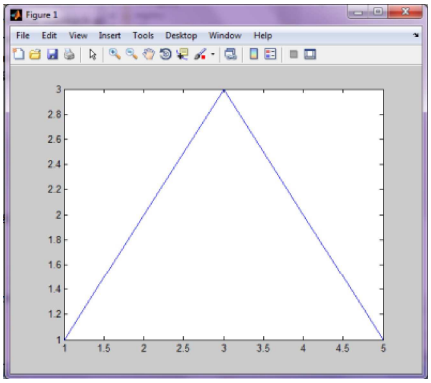
  pričom :

x **-** číslo aktualného grafického okna v ktorom pracujeme,
nazov **-** názov aktuálneho grafického okna v ktorom pracujeme,
all – funkcia ***close all*** uzatvára všetky grafické okna v ktorých sme pracovali.

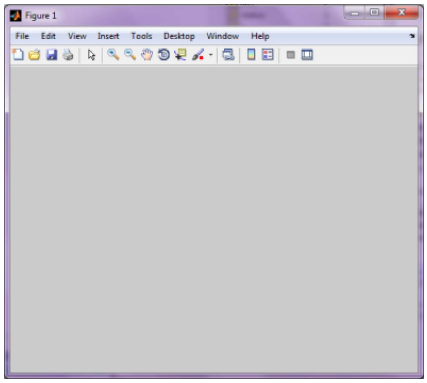
- Funkcia ***gcf*** (*Get Handle to Current Figure*) sa používa na zistenie aktuálneho okna (ak aktuálne nie je otvorené žiadne grafické okno, funkcia otvorí grafické okno)

```
>> x=gcf

x =

     1
```

- Funkcia ***clf*** **(Clear Current Figure)** sa používa na zmazanie aktuálneho obrázku z aktuálneho grafického okna.



```
>> clf
```



- Funkcia ***plot*** – vykreslí graf, ak hodnoty zavislej a nezavislej premennej boli vopred zadané vo vektoroch. Funkcia otvorí nové grafické okno, za predpokladu, že doposiaľ nebolo otvorené a ak bolo, pracuje s aktívnym oknom a vykreslí graf.

  plot(x,y) %funkcia ***plot*** vykreslí funkčnú závislosť *y = f(x)*

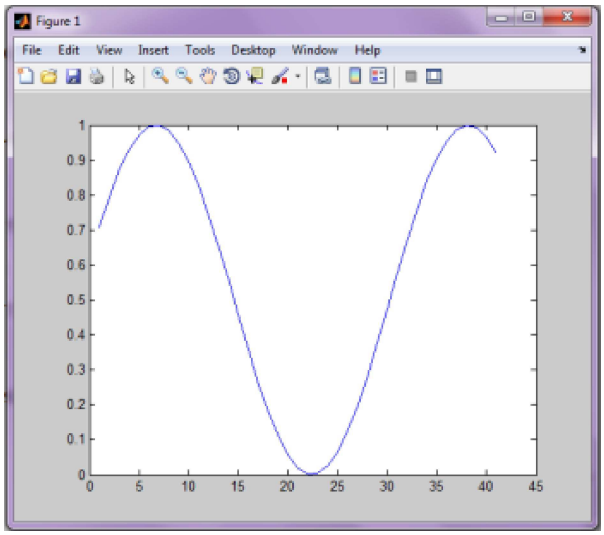
  pričom vektory hodnôt ***x,y*** musia byť dopredu zadefinované

Príklad 1

Je daná funkcia $y = (sin(x))^2$ Znázornite graficky danú funkciu pre hodnoty vektora ***x*** z intervalu : $x \in \langle 1,5 \rangle$ s krokom 0,1

Riešenie v jazyku MATLAB :

```
>> x=[1:0.1:5];
>> y=sin(x).^2;
>> plot(x,y)
```



Funkcia ***plot*** volaná nasledovne

plot (x,y, ´vlastnosti´)

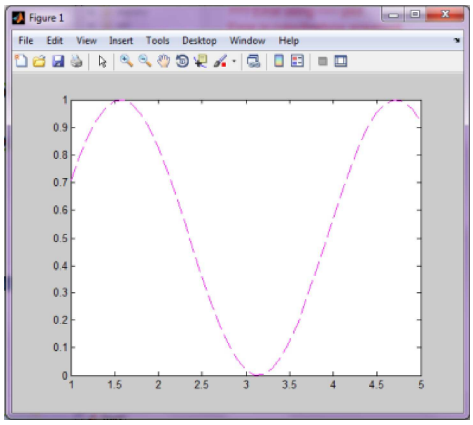
vykreslí hodnoty ***y*** v závislosti na hodnotách ***x*** a s atribútmi , ktoré sú nastavené textovým reťazcom uvedeným v apostrofoch. Môžeme použiť nasledujúce atribúty:

farby		**body**		**čiary**	
r	červená	**.**	bod	**-**	plná (default)
y	žltá	**o**	kruh	**:**	bodkovaná
k	čierna	**x**	krížik	**-.**	bodkočiarkovaná
b	modrá	**+**	plus	**--**	čiarkovaná
g	zelená	*****	hviezda	**none**	bez čiary
c	azúrová	**s**	štvorček		
m	purpurová	**d**	diamant	**LineWidth**	šírka čiary
w	biela	**v**	trojuholník (dolu)	**Color**	farba popísaná v RGB
		^	trojuholník (hore)	**MarkerEdgeColor**	farba obrysu ukazovateľa
		<	trojuholník (vľavo)	**MarkerFaceColor**	farba vypne ukazovateľa
		>	trojuholník (vpravo)	**MarkerSize**	veľkosť ukazovateľa
		p	pentagram		
		h	hexagram		

Príklad 2

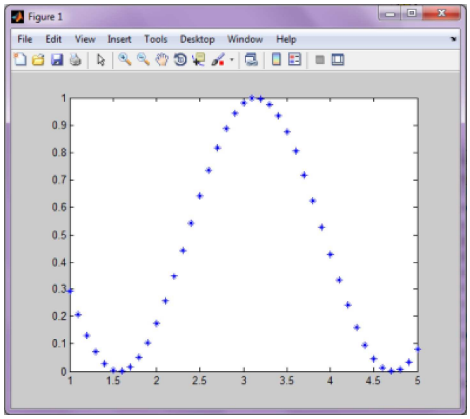
Vykreslite funkciu $y = sin(x^2)$ ***y = sin(x²)*** na intervale x ∈ <1,5> s krokom 0,1 ,čiarkovanou purpurovou čiarou

```
>> x=[1:0.1:5];
>> y=sin(x).^2;
>> plot(x,y,'m--')
```



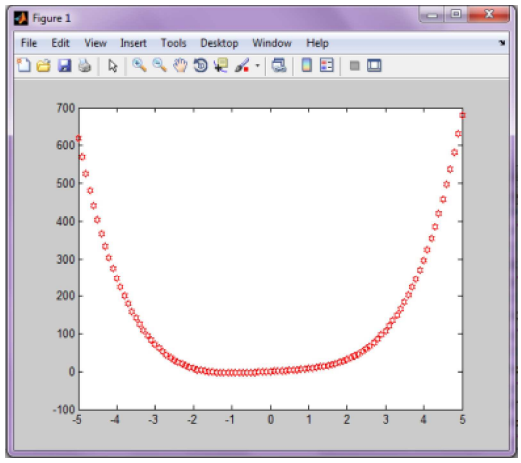
- Vykreslite funkciu ***y = (cos(x))²*** na intervale x ∈ <1,5> s krokom 0,1 , pričom pixle sú označené modrými hviezdičkami

```
>> x=[1:0.1:5];
>> y=cos(x).^2;
>> plot(x,y,'b*')
```



- Vykreslite funkčnú závislosť $y = x^4 + x^2 + 6 * x$ ***y = x⁴ +x² +6.x*** na intervale *x* ∈ <-5,5> s krokom 0,1 , použite červený šesťuholník.

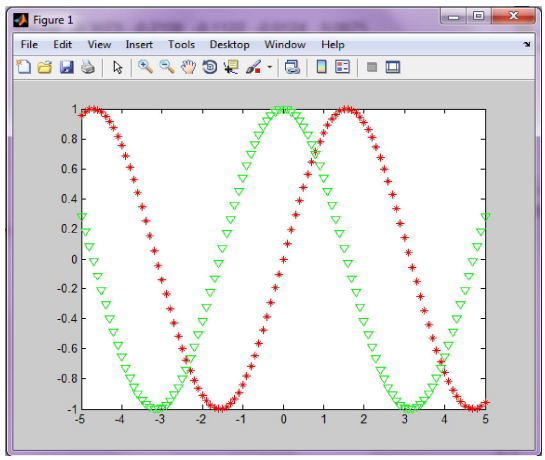
```
>> x=[-5:0.1:5];
>> y=x.^4+x.^2+6*x;
>> plot(x,y,'rh')
```



Pozn. Viacparametrovú funkciu ***plot(x,y₁,x,y₂,....)*** používame na vykreslenie viacerých funkčných závislostí vzhľadom na jediný vektor ***x*** nezávislej premennej.

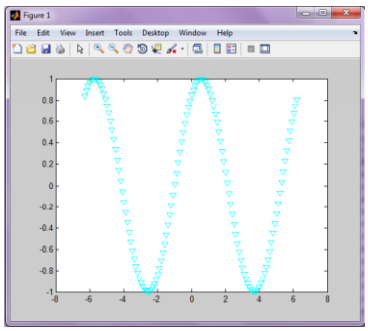
Príklad 3
Vykresli funkcie $y_1 = sin(x)$ a $y_2 = cos(x)$ na intervale <-5,5> s krokom 0,1 , pričom ***y₁*** je vykreslená červenými hviezdičkami a ***y₂*** je vykreslené zelenými trojuholníkmi.

```
>> x= [-5:0.1:5];
>> y1=sin(x);
>> y2=cos(x);
>> plot(x,y1,'r*',x,y2,'gv')
```

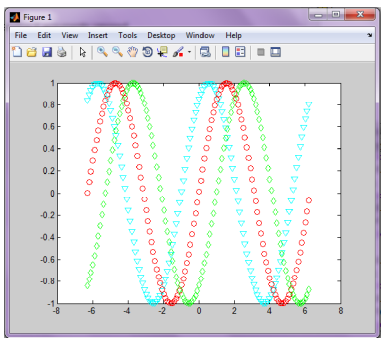


- funkcia ***hold*** – umožňuje ponechať otvorené aktuálne okno grafického zobrazenia a zakresľovať ďalšie grafické zobrazenia (***hold on/off***)

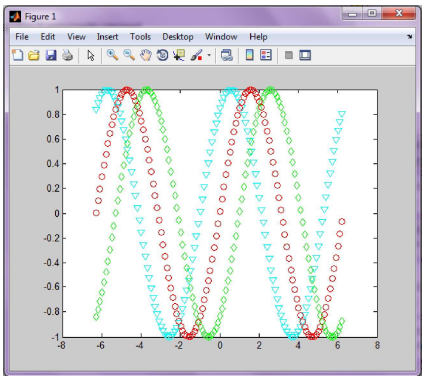
```
>> x=[-2*pi:0.1:2*pi];
>> y=sin(x+1);
>> plot(x,y,'cv')
```



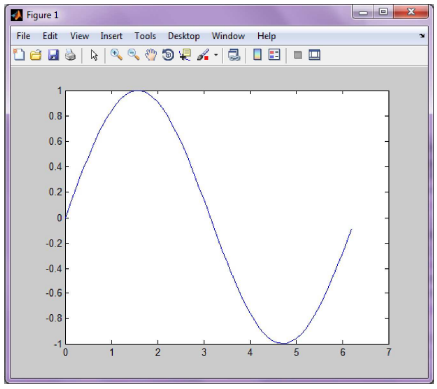

```
>> hold on
>> x=[-2*pi:0.1:2*pi];
>> y=sin(x);
>> plot(x,y,'ro')
>> x=[-2*pi:0.1:2*pi];
>> y=sin(x-1);
>> plot(x,y,'gd')
```



Príkaz ***hold off*** prepína späť do implicitného stavu, v ktorom nové príkazy grafiky vždy prepíšu starý graf a pred kreslením resetujú všetky vlastnosti objektu.



```
>> hold off
>> x=[0:0.1:2*pi];
>> y=sin(x);
>> plot(x,y)
```



- Funkcia ***subplot*** – umožňuje rozdelenie grafického okna na podokná. Syntax v tvare

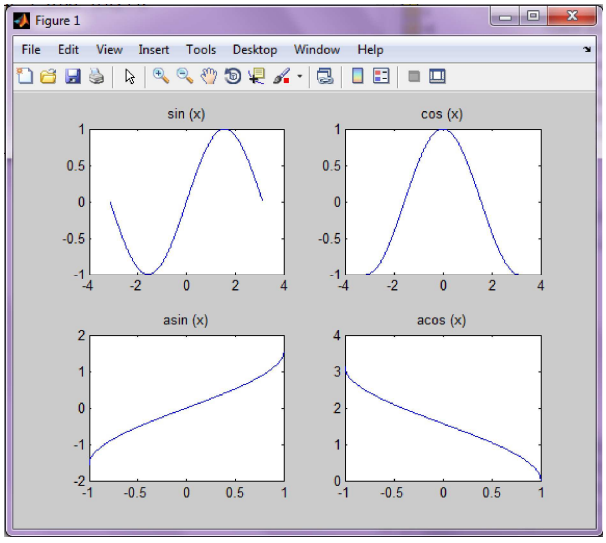
  subplot(m,n,p)

  vytvorí grafické okno v podobe matice $m \times n$ ***m x n*** a aktivuje práve jedno podokno ***p***, takže ju musíme volať pre každé podokno samostatne**,**

Príklad 4
Použitím funkcie ***subplot*** vytvorte grafické okno v ktorom budú 4 podokná a v nich budú zobrazené funkcie ***sin(x)*** a ***cos(x)*** pre $x \in \langle -\pi, \pi \rangle$ a ***acrsin(x)*** a ***arccos(x)*** v intervale $x \in \langle -1,1 \rangle$.

Riešenie v jazyku MATLAB:

```
>> x1=-pi:0.01:pi;
>> x2=-1:0.01:1;
>> y1=sin(x1);
>> y2=cos(x1);
>> y3=asin(x2);
>> y4=acos(x2);
>> subplot(2,2,1); plot(x1,y1); title('sin (x)')
>> subplot(2,2,2); plot(x1,y2); title('cos (x)')
>> subplot(2,2,3); plot(x2,y3); title('asin (x)')
>> subplot(2,2,4); plot(x2,y4); title('acos (x)')
```



- Funkcia ***title*** – umožňuje pomenovať graf (vypíše názov grafu daný textovým reťazcom na stred horného okraja grafu)

  title('nazov') ***title(´nazov´)***

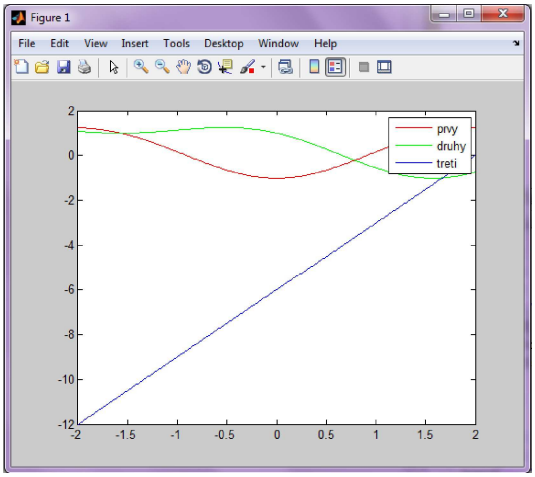
- Funkcia ***label*** – umožňuje popísať osi grafu

xlabel('popisosix')	***xlabel(´popisosix´)***	%pomenuje x-ovú os
ylabel('popisosiy')	***ylabel(´popisosiy´)***	%pomenuje y-ovú os

- Funkcia ***legend*** – umožňuje vytvorenie popisu jednotlivých objektov(grafických závislostí) – priradí k jednotlivým grafickým závislostiam ich popis, farby a prispôsobí ich poradie v akom boli zakreslené

  legend('nazov1','nazov2','nazov3', ...) ***legend(´nazov1´, ´nazov2´, ´nazov3´, ...)***

```
>> x=[-2:0.01:2];
>> y1=sin(x).^2-cos(x);
>> y2=cos(x).^2-sin(x);
>> y3=3*x-6;
>> plot(x,y1,'r')
>> hold on
>> plot(x,y2,'g')
>> plot(x,y3,'b')
>> legend('prvy', 'druhy', 'treti')
```



- Funkcia ***text*** – umožňuje priradenie textu do aktuálneho grafického okna vytvorením objektu text, čím vloží na súradnice *(x,y)* resp. *(x,y,z)* zadaný *reťazec*

  text(x,y,´retazec´), text(x,y,y, ´retazec´),

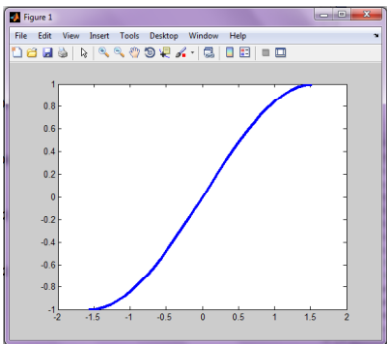
- **Funkcia *axis* -** umožňuje zmenu rozsahu a vzhľadu osi. Syntax

  axis([xmin, xmax, ymin, ymax])

  nastaví osi *x,y* podľa nami zadaných minimálnych a maximálnych hodnôt a zápis

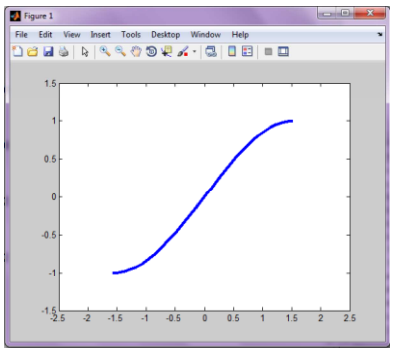
  axis(´auto´)

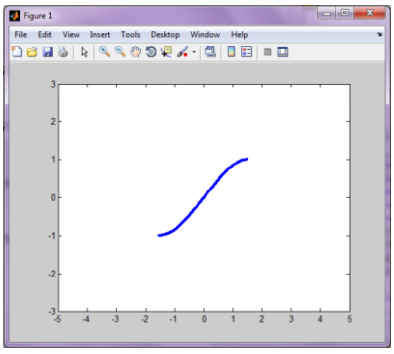
  umožní automatické nastavovanie osi.



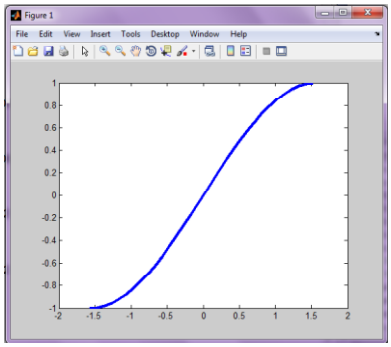

```
>> axis([-2.5, 2.5, -1.5, 1.5])
```

```
>> x=[-pi/2:0.1:pi/2];
>> y=sin(x);
>> plot(x,y,'LineWidth',3)
```





axis('auto')



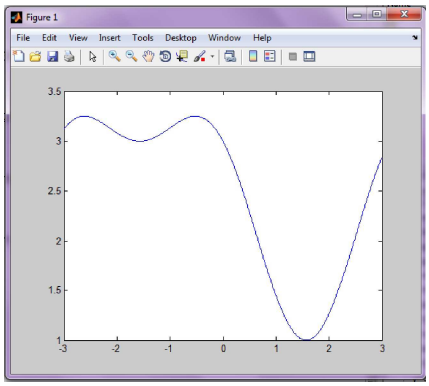
- Funkcia ***grid –*** sa používa na vytvorenie mriežky v grafickom zobrazení, pričom

  grid ongrid on

  zapína kreslenie mriežky do grafického okna a

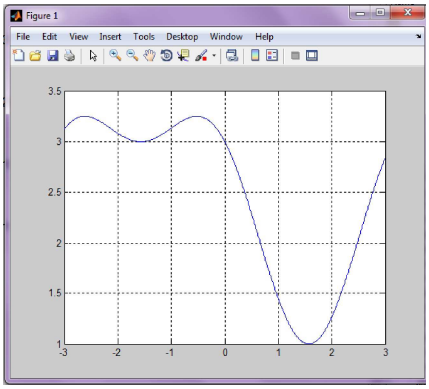
  grid offgrid off

  vypína kreslenie mriežky v grafickom okne.



>> grid on

>> grid off



Zápis ***grid*** slúži ako prepínač na zakreslenie mriežky, ak nebola zakreslená a naopak

3.6 Riešenie príkladov na regresnú analýzu v jazyku MATLAB

Budeme skúmať funkčný vzťah (priebeh závislosti), $y_i = f(x_i)$ podľa ktorého sa mení závislá premenná y pri zmenách nezávislých veličín $x_1, x_2, \ldots, x_n$. Cieľom aproximácie je nájsť vhodnú regresnú funkciu – funkčný predpis. Z nameraných hodnôt x_i / y_i
Regresná analýza pomáha :

- ⇨ riadiť a predpovedať chovanie sledovaných premenných
- ⇨ predpovedať hodnoty výstupných premenných aj tam, kde na výpočet nebolo dostatočné množstvo dát
- ⇨ zistiť body, ktoré sa výrazne odlišujú od očakávaného výsledku

Metóda najmenších štvorcov

Metóda najmenších štvorcov spočíva v nájdení funkcie $y = f(x)$, ktorá čo najlepšie aproximuje namerané hodnoty.

- **Aproximácia nameraných dát priamkou s využitím metódy najmenších štvorcov**

Majme súbor, ktorý obsahuje n nameraných hodnôt v tvare usporiadaných dvojíc $[x_i, y_i]$. Hľadáme lineárnu závislosť (priamku – polynóm 1.stupňa) v tvare

$$y(x) = a_1 x + a_0$$

s takými parametrami priamky a_0 a a_1, aby súčet druhých mocnín (štvorcov) vzdialeností jednotlivých bodov od tejto priamky bol čo najmenší. Túto podmienku je možné zapísať vzťahom:

$$\sum_{i=1}^{n} (a_1 x_i + a_0 - y_i)^2 \rightarrow min$$

Pomocou metódy najmenších štvorcov dostaneme priamku využitím nasledujúcich rovníc pre aproximáciu:

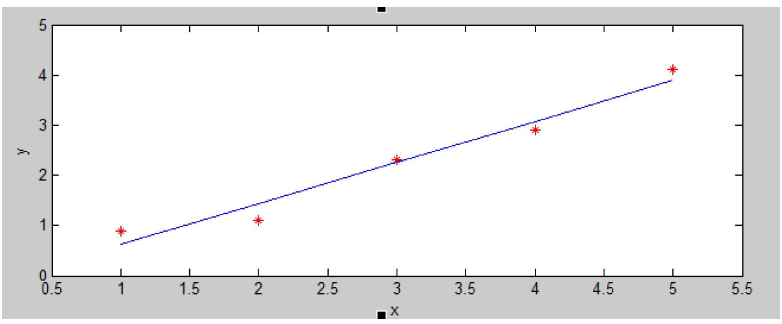
$$a_0(n+1) + a_1 \sum_{i=0}^{n} x_i = \sum_{i=0}^{n} y_i$$

$$a_0 \sum_{i=0}^{n} x_i + a_1 \sum_{i=0}^{n} x_i^2 = \sum_{i=0}^{n} x_i y_i$$

Z tejto podmienky vieme vypočítať parametre priamky a_0 , a_1 nasledovne :

$$a_1 = \frac{n \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n \sum_{i=1}^{n} x_i x_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} x_i}$$

$$a_0 = \left(\sum_{i=1}^{n} y_i - a_1 \sum_{i=1}^{n} x_i \right) / n$$



Obrázok 3-4 Názorna ukážka aproximácie nameraných údajov x_i, y_i priamkou

Korelačný koeficient určuje do akej miery lineárny vzťah $y = a_1 x + a_0$ aproximuje namerané hodnoty y_i, x_i. Korelačný koeficient je vždy číslo z intervalu <-1,1> a vypočítame ho podľa vzorca :

$$r_{x,y} = \frac{s_{xy}}{s_x s_y} = \frac{\frac{1}{n}\sum(x_i - \bar{x}).(y_i - \bar{y})}{s_x s_y}.$$

- **Aproximácia nameraných dát polynómom *n* - tého rádu v prostredí MATLAB**

Niektoré závislosti *x* a *y* nie je vhodné aproximovať priamkou, ale je potrebné ich aproximovať polynómom vyššieho rádu. Napr. pri aproximácii funkcie polynómom druhého stupňa aplikujeme metódu najmenších štvorcov na základe vzťahov

$$a_0(n+1) + a_1 \sum_{i=0}^{n} x_i + a_2 \sum_{i=0}^{n} x_i^2 = \sum_{i=0}^{n} y_i,$$

$$a_0 \sum_{i=0}^{n} x_i + a_1 \sum_{i=0}^{n} x_i^2 + a_2 \sum_{i=0}^{n} x_i^3 = \sum_{i=0}^{n} x_i y_i,$$

$$a_0 \sum_{i=0}^{n} x_i^2 + a_1 \sum_{i=0}^{n} x_i^3 + a_2 \sum_{i=0}^{n} x_i^4 = \sum_{i=0}^{n} x_i^2 y_i.$$

Vo všeobecnosti je v MATLABe je možné zadané (namerané) údaje aproximovať polynómom n-tého stupňa s využitím metódy najmenších štvorcov.

- Funkcia ***polyfit*** realizuje výpočet koeficientov aproximujúceho polynómu *n*-tého stupňa metódou najmenších štvorcov, ktoré zabezpečia funkčné hodnoty polynómu čo najbližšie k funkčným hodnotám aproximovanej funkcie.

P = polyfit(x,y,n)

kde:

x – vektor hodnôt nezávislej premennej
y – vektor hodnôt závislej premennej
n – stupeň polynómu (závisí na fyzikálnej podstate problému, napr. ak predpokladáme aproximáciu priamkou, tak n = 1, ak parabolou, tak n = 2)
P – obsahuje vypočítané koeficienty výsledného polynómu

$$P(x) = p_n x^n + p_{n-1} x^{n-1} + \cdots + p_1 x^1 + p_0,$$

- Funkcia ***polyval*** slúži na výpočet hodnôt aproximačného polynómu v zvolených bodoch:

y_aproximovane = polyval(P,x)

kde:

x je vektor hodnôt nezávislej premennej
P je vektor koeficientov aproximovaného polynómu
y_aproximovane je vektor hodnôt aproximovaného polynómu

Chyby aproximácie

Chybu aproximácie vieme vypočítať pomocou funkcie prostredia MATLAB ***sum.***

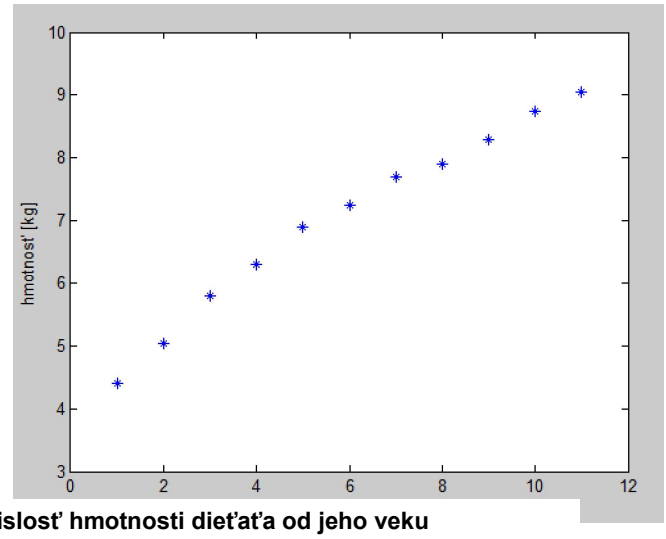
$$S = sum((y\_aproximované - y).\hat{}2)$$

pričom p_1 je približná hodnota funkcie v bode x.

Príklad 1

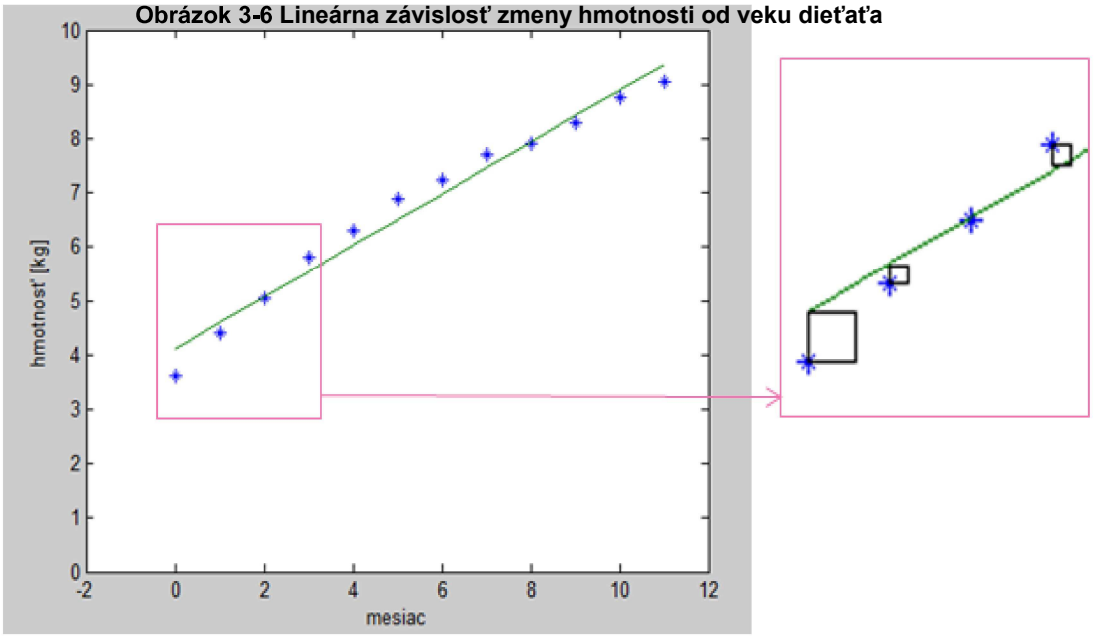
V tabuľke je uvedená meniaca sa hmotnosť dieťaťa vzhľadom na vek dieťaťa v prvých 12-tich mesiacoch. Úlohou je zistiť, aká je závislosť medzi hmotnosťou [kilogram] dieťaťa a jeho vekom [mesiac]. Tabuľkové hodnoty vykreslíme v grafe.

Mesiac (x)	Hmotnosť dieťaťa (y)
0.	3,6
1.	4,4
2.	5,03
3.	5,8
4.	6,3
5.	6,9
6.	7,25
7.	7,7
8.	7,9
9.	8,3
10.	8,75
11.	9,05



Obrázok 3-5 Závislosť hmotnosti dieťaťa od jeho veku

Obrázok 3-6 Lineárna závislosť zmeny hmotnosti od veku dieťaťa



Riešenie

$$12a_0 + 67a_1 = 80{,}98$$
$$67a_0 + 580{,}1784a_1 = 513{,}91$$

Riešením systému rovníc dostávame hľadané hodnoty koeficientov a_0 = 4,11295 , a_1 =0,4791 a výsledná regresná priamka je teda v tvare:

$$y = 0{,}4791x + 4{,}11295$$

Overíme vypočítané parametre priamky a_0, a_1 s využitím príkazu

```
polyfit(x,y,1)
```

```
Command Window
>> x=0:11;
>> y=[3.6 4.4 5.03 5.8 6.3 6.9 7.25 7.7 7.9 8.3 8.75 9.05];
>> f=polyfit(x,y,1)

f =

    0.4792    4.1129
```

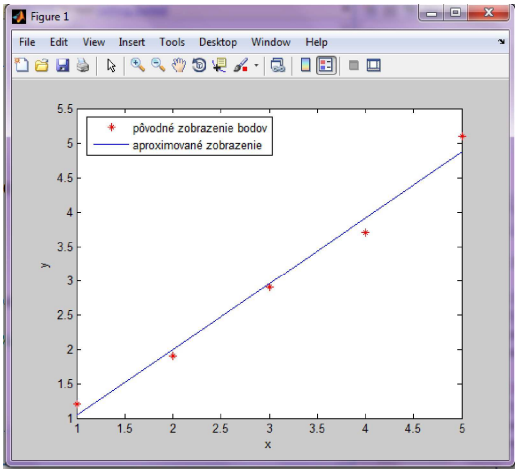
Príklad 2

Napíšte program v simulačnom jazyku MATLAB pre aproximáciu nameraných hodnôt x_i, y_i , priamkou, pričom n = 5, namerané hodnoty x_i, y_i sú uvedené v tabuľke:

x_i	1	2	3	4	5
$y_i = f(x_i)$	1.2	1.9	2.9	3.7	5.1

Riešenie v jazyku MATLAB:

```
>> x=[1:5];
>> y=[1.2,1.9,2.9,3.7,5.1];
>> P=polyfit(x,y,1)

P =

   0.9600   0.0800

>> y_aprox = polyval(P,x)

y_aprox =

   1.0400   2.0000   2.9600   3.9200   4.8800

>> S=sum((y_aprox-y).^2) %výpočet sumy štvorcov odchyliek

S =

   0.1360
```

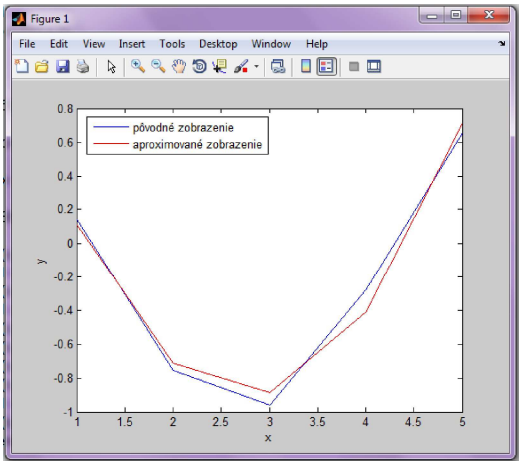



```
>> plot(x,y,'r*')
>> hold on
>> plot(x,y_aprox)
>> legend('pôvodné zobrazenie bodov', 'aproximované zobrazenie')
```

Príklad 3

Napíšte program pre aproximáciu bodov x_i, y_i polynómom 2. rádu pričom $x_i = 1,2,\ldots,5$ a hodnoty y_i vypočítate podľa predpisu $y_i = sin(x_{i+2} + 2)$. Zistite chybu aproximácie

```
>> x=[1:5];
>> y=sin(x+2);
>> P=polyfit(x,y,2)

P =

   0.3250  -1.7992   1.5830

>> y_aprox = polyval(P,x)

y_aprox =

   0.1088  -0.7153  -0.8894  -0.4135   0.7125

>> S=sum((y_aprox-y).^2) %výpočet sumy štvorcov odchyliek

S =

   0.0286
```

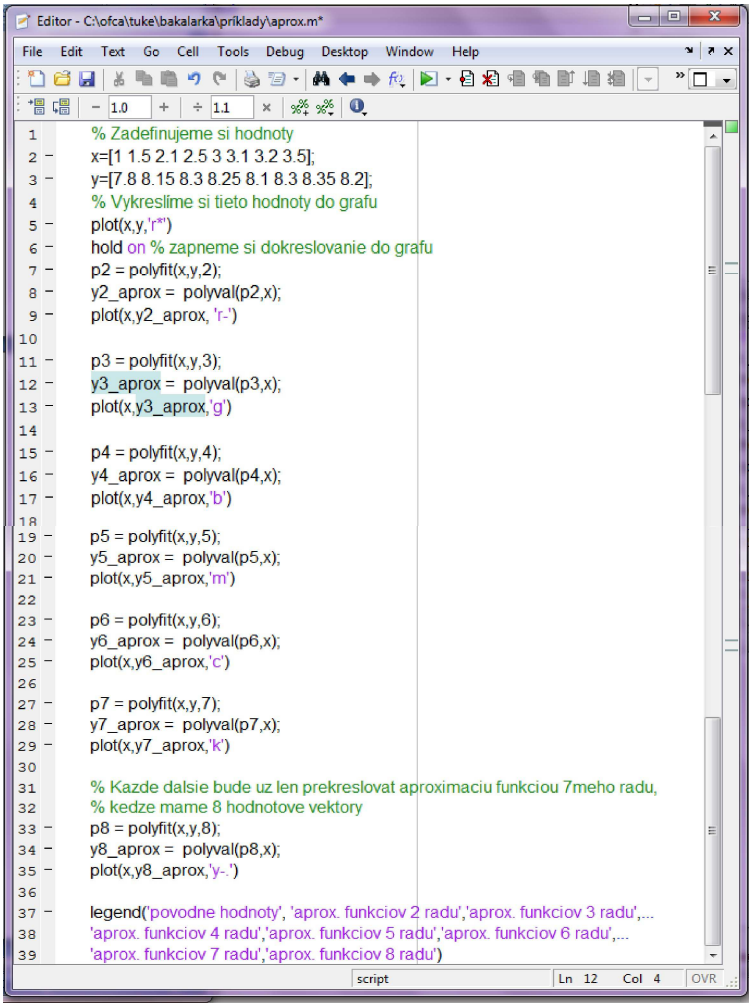



```
>> plot(x,y)
>> hold on
>> plot(x,y_aprox,'r')
>> legend('pôvodné zobrazenie', 'aproximované zobrazenie')
```

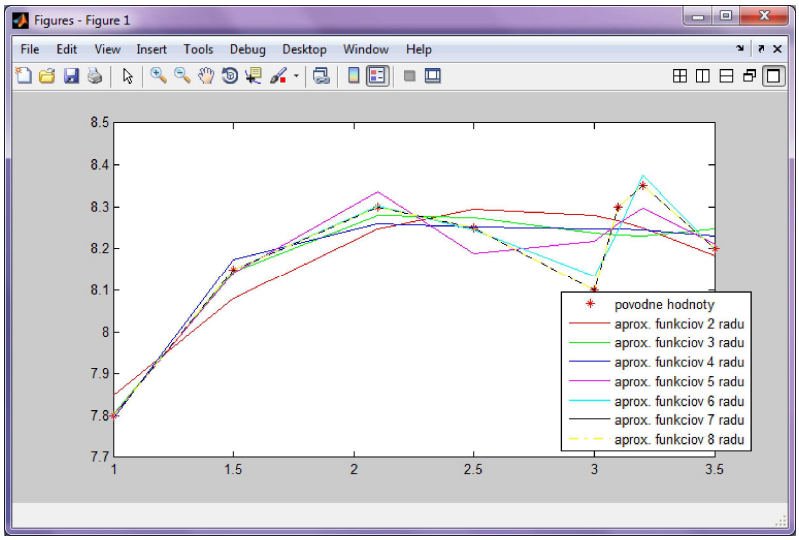
Príklad 4

Aproximujte namerané hodnoty x_i, y_i v tabuľke funkciou (polynómom) 2, 3, ... 8 rádu

x_i	1	1,5	2,1	2,5	3	3,1	3,2	3,5
y_i	7,8	8,15	8,3	8,25	8,1	8,3	8,35	8,2



Obrázok 3-7 Riešenie Príkladu 3 – aproximácia nameraných bodov x_i, y_i polynómom

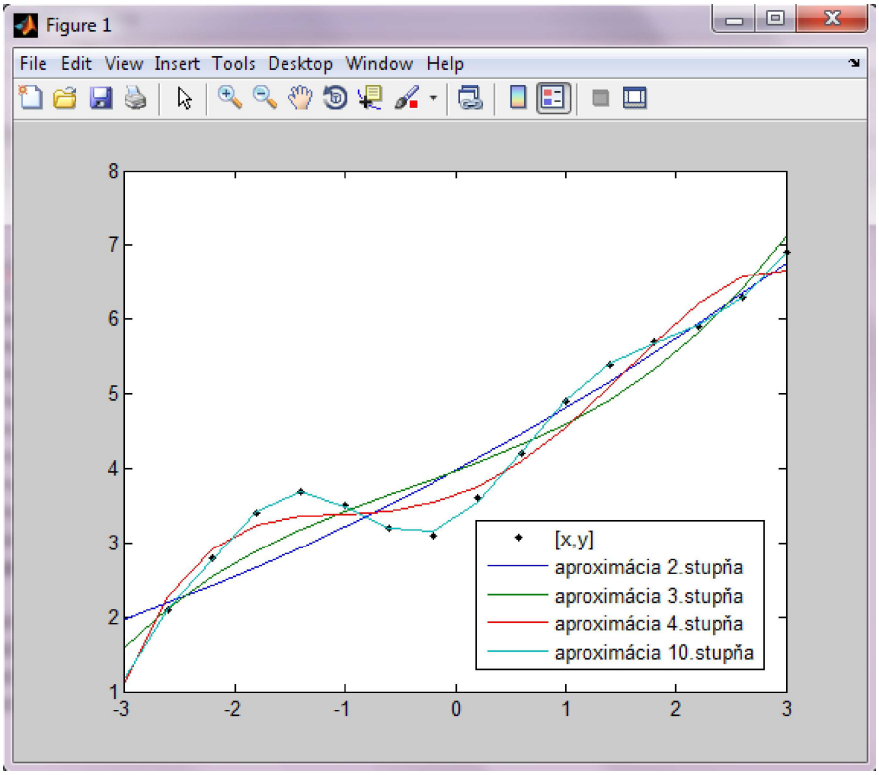


Obrázok 3-8 Grafický priebeh aproximácie funkcie zadanej tabuľkou z Príkladu 3

Príklad
Nech *x*=-3:0.4:3, y=[1.2 2.1 2.8 3.4 3.7 3.5 3.2 3.1 3.6 4.2 4.9 5.4 5.7 5.9 6.3 6.9]. Aproximujte údaje x_i a y_i polynómami *n*=2., 3., 4. a 5. stupňa. Vykreslite do grafu príslušné priebehy. Vypočítajte chybu odchyliek pre každú aproximáciu.

Riešenie v prostredí MATLAB:

```
x=-3:0.4:3;
y=[1.2 2.1 2.8 3.4 3.7 3.5 3.2 3.1 3.6 4.2 4.9 5.4 5.7 5.9 6.3 6.9];
a1=polyfit(x,y,2);
a2=polyval(a1,x);
b1=polyfit(x,y,3);
b2=polyval(b1,x);
c1=polyfit(x,y,4);
c2=polyval(c1,x);
d1=polyfit(x,y,10);
d2=polyval(d1,x);
plot(x,y,'k.',x,a2,x,b2,x,c2,x,d2)
legend('[x,y]','aproximácia 2.stupňa','aproximácia 3.stupňa',...
    'aproximácia 4.stupňa','aproximácia 10.stupňa')
odchylka2=sum(sum(y-a2).^2);
odchylka3=sum(sum(y-b2).^2);
odchylka4=sum(sum(y-c2).^2);
odchylka10=sum(sum(y-d2).^2);
```



Obrázok 3-9 Aproximácia polynómu pre n = 2, 3, 4, 5

3.7 Riešenie úlohy interpolácie z nameraných dát v programovom prostredí MATLAB

Interpolácia je postup pre odhad hodnôt bodov, ktoré neboli priamo namerané, ale ležia medzi nameranými uzlovými bodmi. Polynóm $P_n(x)$ je interpolačným polynómom množiny bodov $[x_i, y_i]$, i = 1, 2,...,k práve vtedy, ak pre jeho hodnoty v uzloch interpolácie platí $P_n(x_i) = y_i$, $i = 1, 2, \dots, k$.
Stupeň n interpolačného polynómu $P_n(x)$ je menší nanajvýš rovný $k - 1$.

- ⇨ V praxi je metóda interpolácie používaná pokiaľ poznáme danú funkciu $f(x)$ v určitých diskrétnych bodoch x_i a požadujeme , aby s aproximovanou funkciou $\varphi(x)$ súhlasila vo všetkých bodoch x_i, t. j. platí **interpolačná podmienka**:
  - $f(x_i) = \varphi(x_i), \quad i = 0,1, \dots, n$
- ⇨ Interpolovať dáta teda znamená nájsť z danej sady takú závislosť, ktorá vyhovuje všetkým dátam.
- ⇨ Túto úlohu často potrebujeme v prípadoch, keď poznáme hodnotu funkcie v určitých diskrétnych bodoch $x_i \in (a, b)$ a chceme určiť hodnotu v iných bodoch intervalu *(a,b)*.
- ⇨ Pri interpolacií sa výsledná aproximovaná funkcia φ volí ako lineárna kombinácia čiastkových funkcií φ_i a teda je riešená úloha hľadania koeficientov c_i,*i*=1, ... , *n* tak , aby platilo:

$$\varphi(x_k) = \sum_{i=1}^{n} c_i \varphi_i(x_k) = y_k = f(x_k).$$

Funkcie φ_i sú dopredu zvolené uživateľom.

- **Interpolácia pomocou polynómov**

- ⇨ Všeobecne platí, že $(n + 1)$ bodov je možné jednoznačne preložiť polynómom n-tého stupňa. Pre danú množinu čísel x_i a pre danú množinu funkcií φ_i je možné vytvoriť množinu funkcií v_i ortogonálych na množine x_ia generujúcich rovnaký priestor ako množina φ_i. Príkladom je tzv. ***Lagrangeov interpolačný polynóm***.
- ⇨ Ak označíme l_k *k*-tým Lagrangeových polynómom na množine x_i, $k = 1, \dots, n$, potom môžeme zapísať:

$$l_k = \frac{(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)}{(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)}$$

potom Lagrangeov interpolačný polynóm $L_n(x)$ pre *i*=0, 1, ..., *n* má tvar:

$$L_n(x) = \sum_{i=0}^{n} \frac{(x - x_0) \dots (x - x_{i-1})(x - x_{i+1}) \dots (x - x_n)}{(x_i - x_0) \dots (x_i - x_{i-1})(x_i - x_{i+1}) \dots (x_i - x_n)} f(x_i)$$

Príklad
Majme zadané hodnoty funkcie *f* : <1;4> tabuľkou:

x	1	2	4
f(x)	1	4	16

Nahradením funkcie pomocou polynómu $L_2(x)$ sa vypočíta približná hodnota funkcie *f* v bode *x*=3 pomocou $L_2(3)$

$$L_2(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \cdot f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \cdot f(x_1) + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \cdot f(x_2).$$

Dosadením konkrétnych hodnôt do predchádzajúceho vzťahu dostaneme :

$$L_2(3) = \frac{(3-2)(3-4)}{(1-2)(1-4)} \cdot 1 + \frac{(3-1)(3-4)}{(2-1)(2-4)} \cdot 4 + \frac{(3-1)(3-2)}{(4-1)(4-2)} \cdot 16 = 9.$$

Pre interpoláciu funkčných hodnôt *y* v závislosti na *x (y=f(x))* existuje v prostredí MATLAB funkcia ***interp1***. Pre interpoláciu funkčných hodnôt ***z*** v závislosti od *x,y* ***(z=f(x,y))*** sa využíva funkcia ***interp2***. Pre závislosť *v=f(x,y,z)* je k dispozícii funkcia ***interp3***.

Funkcia ***interp1*** ***interp1*** rieši úlohu interpolácie pomocou vhodne zvolených aproximačných funkcií pre zadefinované body x_i. Syntax funkcie je:

```
interp1(x,y,xi,'metoda'),
```

kde :

x – vektor nameraných hodnôt nezávislej premennej
y – vektor nameraných hodnôt závislej premennej
x_i – body interpolácie – hodnoty pre ktoré nepoznáme funkčnú hodnotu y_i
„metoda“ – zvolená konkrétna aproximačná funkcia pre interpoláciu
y_i – vektor funkčných hodnôt interpolačnej krivky v bodoch x_i

Parameter **‚metóda‘** nie je povinný, predstavuje akou metódou sa má interpolácia vykonať:

‚nearest‘ – interpolácia susedných bodov
‚linear‘ – lineárna interpolácia
‚spline‘ – kubická splajnová interpolácia
‚cubic‘ – štvorcová interpolácia

Syntax funkcie ***interp2*** je:

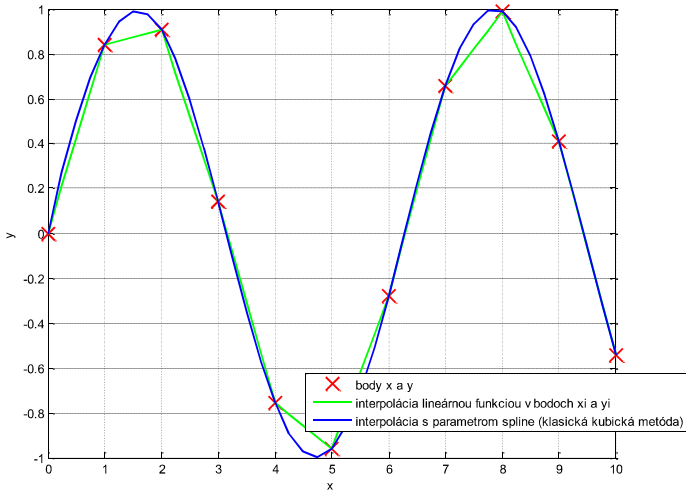
```
interp1(x,y,xi,'metóda')
```

kde *z* je funkčná hodnota pre dvojrozmernú interpoláciu, *x* a *y* sú súradnice interpolácie, *xi* a *yi* súbor bodov interpolácie. Pre parameter **‚metóda‘** platí to, čo pri funkcii *interp1.*

Budeme sa zaoberať možnosťami interpolácie v rovine (2D) a v priestore (3D).

Príklad 1
Vytvorte vektor $x \in <0,10>$ s krokom 1. Vypočítajte v programovom prostredí MATLAB hodnoty $y_i = \sin(x)$ a následne tieto dvojice x_i,y_i vykreslite. Aplikujte metódu interpolácie s využitím funkcie ***interp1*** a **lineárnej /kubickej** funkcie pre aproximáciu hodnôt $x_i \in <0,10>$ s krokom 0,25.

```
>> x=0:10;
>> y=sin(x);
>> plot(x,y,'rx');
>> xi=0:0.25:10;
>> yi=interp1(x,y,xi,'linear');
>> plot(xi,yi,'g');
>> yi=interp1(x,y,xi,'spline');
>> plot(xi,yi,'b');
>> hold on
>> grid on
>> legend('body x a y','interpolácia lineárnou funkciou v
bodoch xi a yi', 'interpolácia s parametrom spline (klasická
kubická metóda)')
```

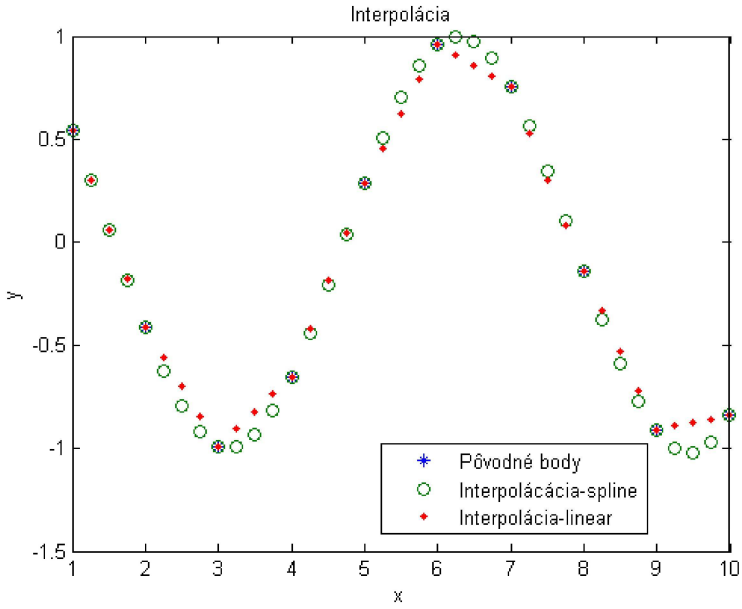


Obrázok 3-10 Interpolácia lineárnou funkciou a splinom v bodoch x_i , y_i

Príklad 2

Interpolujte dáta *x* a *y* so štvornásobnou periódou vzorkovania (so štvrtinovým krokom). Výsledok vykreslite v grafe, kde *x*=0:10, *y*=cos(x).

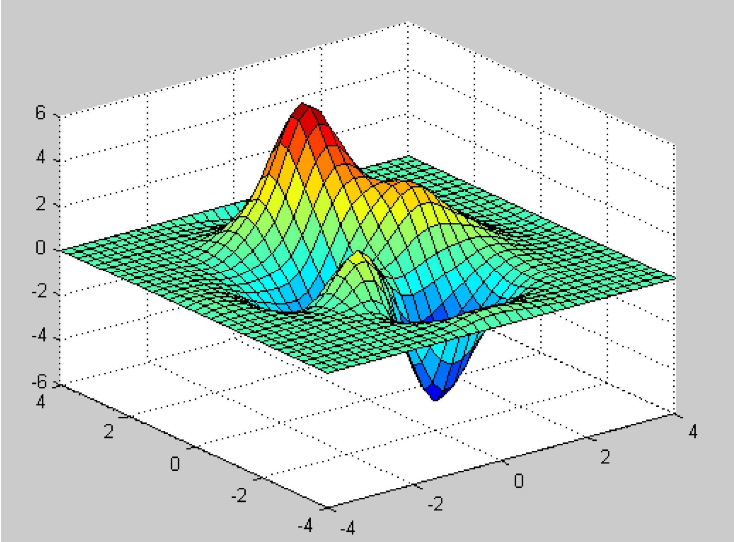
```
x=1:10;
y=cos(x);
xi=1:0.25:10;
yi1=interp1(x,y,xi,'spline');
yi2=interp1(x,y,xi,'linear');
plot (x,y,'*',xi,yi1,'o',xi,yi2,'.')
title ('Interpolácia')
xlabel ('x')
ylabel('y')
legend('Pôvodné body', 'Interpolácácia-spline','Interpolácia-linear')
```



Obrázok 3-11 Interpolácia lineárnou funkciou a splinom v bodoch x_i , y_i

Príklad 3
Vytvorte matice *X* a *Y* ktoré budú obsahovať sieť hodnôt v rozmedzí od <-3;3>. Sieť pre interpoláciu vytvorte s krokom 0,25. Dáta interpolujte druhým stupňom interpolácie. Výsledok zobrazte v grafe pomocou ***surf()***.

```
[x,y]=meshgrid(-4:1:4);
z=peaks(x,y);
% Vytvorenie siete pre interpoláciu:
[xi,yi]=meshgrid(-4:0.25:4);
% Interpolácia štvorcovou metódou
zi1=interp2(x,y,z,xi,yi,'cubic');
surf(xi,yi,zi1)
```



Obrázok 3-12 Výsledok interpolácie funkcie z = f(x,y)