

Kratak kurs MatLab-a

1. Uvodne osnove.....	3
2. Rad sa osnovnim funkcijama MatLab-a	4
2.1. Matematičke funkcije u MatLabu.....	5
2.2. Rad sa matricama	6
2.2.1. Rad sa elementima matrice.....	8
2.2.2. Osnovne operacije sa matricama	10
2.2.3. Operacije sa elementima matrice ili vektora	14
2.2.4. Logički I relacioni operatori u MatLabu	15
3. Funkcije za crtanje u MatLabu	17
4. Korisničke funkcije i rad sa njima.....	21
5. Simboličke promenljive u MatLab-u	25
6. Pregled Nekih MatLab definisanih funkcija i specijalnih vrednosti	27

1. UVODNE OSNOVE

MatLab je proizvod kompanije MathWorks Inc. I prevažodno je namenjen kao program za pomoć inženjerima pri njihovom radu.

Samo ime MatLab izvedeno je iz Matrix Laboratory. Odavde nam proizvođač softvera odmah sugeriše da ovaj program prevažodno radi sa matricama. Čak i skalar u MatLabu je matrica dimenzije 1x1.

Startovanjem MatLab programa pokreće se radno okruženje. Radno okruženje svaki korisnik može podesiti prema svojim potrebama, ali podrazumevano radno okruženje sastoji se od komandnog prozora koji zauzima najveći deo samog prostora. Zatim, obično, sa leve strane se u okviru istog prozora nalaze prikaz radnog direktorijuma i radni prostor koji prikazuje sve definisane promenljive u MatLabu.

Radni direktorijum prikazuje sadržaj direktorijuma koji je po osnovnoj postavici:

```
C:\Documents and Settings\<<windows user>\My Documents\MATLAB
```

Radni direktorijum predstavlja putanju koju MatLab koristi za traženje funkcija koje je definisao korisnik prilikom njihovog izvršavanja. Više reči o ovome biće u odeljku posvećenom korisnički definisanim funkcijama.

Radni prostor je obično po samom pokretanju prazan. On se nalazi u okviru istog prozora kao i radni direktorijum, a klikom na određeni tab prelazimo sa prikaza jednog na prikaz drugog. Uloga radnog prostora jeste da prikaže definisane promenljive. Prikaz definisanih promenljivih možemo dobiti i kucanjem u komandnom prozoru naredbe:

```
>> who
```

ili

```
>> whos (daje detaljniji prikaz od naredbe who)
```

Ispod radnog prostora nalazi se istorija. U njoj možemo videti komande koje su prethodno bile korišćene u radu. Ova istorija se čuva i ukoliko je MatLab posle instalacije bio korišćen, a istorija nije bila brisana, ovaj prozor će nam prikazati koje su to komande bile izvršavane.

MatLab je pre svega jedan interpreter. To podrazumeva da komande koje se unesu nije potrebno pre izvršavanja kompajlirati. Isto se odnosi i na korisničke funkcije koje je takođe moguće definisati i koristiti u MatLab-u.

Komande koje želimo da izvršimo unosimo u komandni prozor i pritiskom na taster Enter dajemo znak za njihovo izvršavanje.

Komandni prozor prilikom pokretanja MatLab programa sadrži samo odzivni znaka: >>

MatLab poseduje veoma dobro napisan i organizovan help. Poziv help-a možemo izvršiti na dva načina.

1. >> help

kao rezultat dobijamo osnovne teme helpa o tekstualnom obliku u okviru komandnog prozora

2. Odabirom opcije Help > MATLAB Help (ili pritiskom tastera F1) iz menija

Na ovaj način otvara se novi prozor u kome su prikazane teme koje možemo pregledati i pretraživati

2. RAD SA OSNOVNIM FUNKCIJAMA MATLAB-A

Pre nego što počnemo da unosimo bilo kakvu komandu u MatLab prvo treba nešto da kažemo o samom formatu zapisa brojeva u istom.

Kao što je ranije rečeno osnovni tip podataka u MatLab-u su matrice. Za predstavljanje elmenata matrice uvek se koristi realni zapis brojeva. Prilikom rada u programu moguće je izabrati odgovarajući tip zapisa u zavisnosti od toga koliko tačnost zapisa želimo, kao i koliko memorije imamo na raspolaganju. Korišćenjem zapisa sa većim brojem sigurnih cifara dobićemo bolju tačnost, ali sa druge strane utrošićemo više RAM memorije. Doduše ovo poslednje nas zanima samo ukoliko rešavamo dosta složen problem velikih dimenzija promenljivih, a pošto u toku

ovog kursa nećemo imati prilike da radimo sa toliko složenim problemima onda i nećemo imati potrebu da se dublje bavimo ovom problematikom.

Kucanjem komande `get(0,'Format')` iza odzivnog znaka dobijamo informaciju o tome koji je trenutno aktivni format zapisa. Podrazumevani format je format short.

Format zapisa možemo promeniti kucanjem neke od sledećih komandi:

format short	zapis sa fiksnom tačkom na 4 decimale, za brojeve od 0.001 do 1000
format long	zapis sa fiksnom tačkom na 14 decimala, za brojeve od 0.001 do 100
format short e	zapis sa pokretnom tačkom na 4 decimale
format long e	zapis sa pokretnom tačkom na 15 decimala
format short g	zapis broja sa 5 cifara u fiksnom ili pokretnom zarezu u zavisnosti koj je prikaz bolji
format long g	zapis broja sa 15 cifara u fiksnom ili pokretnom zarezu u zavisnosti koj je prikaz bolji
format blank	zapis broja sa fiksnom tačkom na 2 decimale

Sada smo spremni za primere.

```
>> format short
```

```
>> 8+9-4
```

```
>>21/sqrt(9)
```

```
>>sqrt(22+9*3)
```

2.1. MATEMATIČKE FUNKCIJE U MATLABU

`sqrt(x)` - kvadratni koren

`exp(x)` - eksponencijalna funkcija sa osnovom e

`abs(x)` - apsolutna vrednost

`log(x)` - logaritamska funkcija sa osnovom e

`log10(x)` – logaritamska funkcija sa osnovom 10

`factorial(x)` - faktorijel od x

`sin(x), cos(x), tan(x), cot(x), asin(x), acos(x), atan(x), acot(x), sinh(x), cosh(x), tanh(x), coth(x)`

2.2. RAD SA MATRICAMA

Osnovni element kojim se operiše MATLAB je matrica kompleksnih brojeva. Specijalni slučajevi su matrica dimenzije 1 x 1 koja predstavlja skalar i matrica sa samo jednom vrstom ili kolonom koja predstavlja vektor vrstu odnosno vektor kolonu. Sve operacije koje se vrše nad takvim elementima su matrice. Promenljive se definišu tako što im se dodeli vrednost. Naziv promenljive je niz slova i cifara koji mora početi slovom, pri čemu se velika i mala slova razlikuju.

Sledi primer definicije matrice dimenzije 3 x 3:

```
>> A = [1 2 3; 4 5 6; 7 8 9]
```

A =

1 2 3

4 5 6

7 8 9

Matricu A je moguće definisati i na drugi način:

```
>> A = [1 2 3
```

4 5 6

```
7 8 9]
```

Vektor-vrsta se definiše kao u sledećem primeru:

```
>> x = [-1.3 sqrt(3.1) (1+2)/4*5]
```

x =

```
-1.300 1.7607 3.7500
```

Slično kao u prethodnom primeru sa matricom definiše se i vektor-kolona. Elemente matrice je neophodno odvojiti bar jednim *white space* karakterom. Prilikom dodeljivanja vrednosti promenljivim MATLAB će ispisati upravo dodeljenu vrednost. Stavljanje znaka tačka-zarez (;) na kraj reda prilikom dodele vrednosti promenljivoj će sprečiti ispis vrednosti na ekran. Sledi primer:

```
>> y = [1.25e-2 sin(5)];
```

Imaginarna jedinica *i* može se eksplicitno definisati na sledeći na način:

```
>> i = sqrt(-1)
```

i =

```
0+1.0000i
```

Ovako definisan skalar *i* potom se može koristiti u formiranju MATLAB izraza, kao u sledećem primeru:

```
>> C = [1+5*i 2+6*i; 3+7*i 8];
```

Prilikom definisanja vrednosti promenljivih posebno je koristan operator dvotačka (:). Ovaj operator je namenjen za definisanje vektora čiji se elementi dobijaju inkrementiranjem početne vrednosti za konstantan korak. Sledi primer:

```
>> x = 1:4
```

x =

```
1 2 3 4
```

Moguće je navesti i korak različit od 1, kao u sledećem primeru:

```
>> y = 1:0.5:3
```

y =

```
1.0000 1.5000 2.0000 2.5000 3.0000
```

Vektor sa konstantnim korakom između elemenata može se generisati zadavanjem prvog i poslednjeg elementa, kao i broja elemenata, na sledeći način:

```
>> v=linspace(0,8,6)
```

v =

```
0 1.6000 3.2000 4.8000 6.4000 8.0000
```

Operator dvotačka se može koristiti i za dodelu vrednosti matrici, ali samo u slučaju da je matrica bila prethodno definisana (tj. poznate joj dimenzije). Sledeći primer ilustruje ovu mogućnost.

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

```
>> A(:) = 21:29
```

A =

```
21 24 27
```

```
22 25 28
```

```
23 26 29
```

MATLAB automatski dodeljuje vrednost poslednje izračunatog izraza promenljivoj *ans*. Tako se može dobiti i vrednost izraza koji nisu dodeljeni kao vrednost nijednoj promenljivoj. Sledi primer:

```
>> sqrt(2)
```

ans =

```
1.4142
```

2.2.1. RAD SA ELEMENTIMA MATRICE

Elementima matrice pristupa se preko njihovog indeksa. Osim pojedinačnih elemenata matrice, moguće je izdvojiti vrstu, kolonu ili bilo koju podmatricu. Za matricu definisanu izrazom:

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

element A(2,2) bi bio:


```
>> A(2,2)
```

```
ans =
```

```
5
```

Izdvajanje treće kolone matrice A vrši se izrazom:

```
>> A(:, 3)
```

```
ans =
```

```
3
```

```
6
```

```
9
```

Podmatrica koja predstavlja presek prve i treće vrste i druge i treće kolone matrice A dobija se izrazom:

```
>> A([1 3], [2 3])
```

```
ans =
```

```
2 3
```

```
8 9
```

Prethodni rezultat se može dobiti i izrazom:

```
>> A(1:2:3, 2:3)
```

U izrazima za selekciju elemenata matrice moguće je koristiti i prethodno definisane vektore – promenljive. U MATLAB-u je moguće promeniti i dimenzije matrice. Na primer, izraz

```
>> A = [A; 10 11 12]
```

```
A =
```

```
1 2 3
```

```
4 5 6
```

```
7 8 9
```

10 11 12

će proširiti matricu A novom vrstom. Slično tome, sledeća komanda će ukloniti drugu kolonu iz matrice A.

```
>> A = A(:, [1 3])
```

A =

1 3

4 6

7 9

10 11

Ukoliko se matrica proširuje novim elementima, a ne definišu se vrednosti svih novih elemenata, za njihove vrednosti se uzima nula. Na primer:

```
>> x = 1 : 4;
```

```
>> x(6) = 7
```

x =

1 2 3 4 0 7

2.2.2. OSNOVNE OPERACIJE SA MATRICAMA

Transponovanje

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

```
>> B = A'
```

B =

1 4 7

2 5 8

3 6 9

Sabiranje

```
>> C = A + B
```

```
C =
```

```
2 6 10
```

```
6 10 14
```

```
10 14 18
```

Sabiranje matrice i skalara

Prilikom sabiranja matrice sa skalarom, svaki element matrice se sabira sa datim skalarom. Sabiranje matrice sa skalarom je komutativno.

```
>> C = C - 2
```

```
C =
```

```
0 4 8
```

```
4 8 12
```

```
8 12 16
```

Množenje matrica

Operator * obuhvata i množenje matrice i vektora, množenje dva vektora i množenje matrice ili vektora skalarom. Pri tome treba voditi računa o dimenzijama matrica koje se množe. Dakle ukoliko želimo da pomnožimo matrice A i B (slično je i za vektor, jer ne treba zaboraviti da su vektori matrice čija je dimenzija jedne vrste ili kolone jednaka 1) onda mora da važi da je broj kolona matrice A jednak broju vrsta matrice B. Npr. ako matrica A ima dimenziju $n \times m$ onda B mora imati dimenziju $m \times r$ da bi operacija množenja prošla bez greške.

Naredni primeri ilustruju ove mogućnosti MATLAB-a.

```
>> D = A * B
```

```
D =
```

```
14 32 50
```

```
32 77 122
```

```
50 122 194
```

```
>> x = [1 2 3]';
```

Primetimo ovde da je dimenzija vektora 3×1 , a dimenzija matrice D 3×3

```
>> E = D*x
```

Stoga je rezultat množenja vektor dimenzije 3×1

```
E =
```

```
228
```

```
552
```

```
876
```

```
>> y = [4 5 6];
```

Primetimo ovde da je dimenzija vektora y 1×3 , a vektora x 3×1

```
>> x*y
```

Stoga je rezultat množenja $x*y$ matrica dimenzije 3×3

```
ans =
```

```
4 5 6
```

```
8 10 12
```

```
12 15 18
```

```
>> y*x
```

U ovom slučaju je rezultata množenja matrica 1×1 odnosno skalar

```
ans =
```

```
32
```

```
>> y*2
```

```
ans =
```

```
8 10 12
```

Pored ovoga, MATLAB poseduje operatore / i \ koji su namenjeni za množenje matrice inverzijom druge matrice. Pri tome izraz $A \setminus B$ predstavlja množenje $A^{-1} * B$, a izraz A / B predstavlja množenje $A * B^{-1}$. Iz ovoga sledi da se sistem linearnih jednačina $Ax = b$ u MATLAB-u može relativno jednostavno rešiti, kao u sledećem primeru:

```
>> A = [10 2 6; 1 10 9; 2 -7 -10];
```

```
>> b = [28 7 -17]';
```

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

```
x =
```

```
1.0000
```

```
-3.0000
```

```
4.0000
```

U slučaju da je matrica sistema singularna, ili bliska singularnoj (dakle $\det(A)=0$ ili je bliska nuli), MATLAB će ispisati upozorenje:

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

```
>> b = [10 4 2]';
```

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

```
Warning: Matrix is close to singular or badly scaled.
```

```
Results may be inaccurate. RCOND = 2.937385e-018
```

```
x =
```

```
1.0e+016 *
```

```
1.2609
```

```
-2.5218
```

```
1.2609
```

2.2.3. OPERACIJE SA ELEMENTIMA MATRICE ILI VEKTORA

Elementi dve matrice ili vektora se mogu pomnožiti po principu “element sa elementom”, pomoću operatora `.*`. Pri tome dimenzije matrica ili vektora koji se množe moraju biti jednake. Sledi primer:

```
>> x = [1 2 3];
```

```
>> y = [4 5 6];
```

```
>> x .* y
```

```
ans =
```

```
4 10 18
```

Analogno prethodnom postoje i operatori `./` i `.\` koji predstavljaju deljenje sa desna odnosno sa leva. Sledeći primer ilustruje upotrebu ovih operatora:

```
>> x ./ y
```

```
ans =
```

```
0.2500 0.4000 0.5000
```

```
>> x .\ y
```

```
ans =
```

```
4.0000 2.5000 2.0000
```

Stepenovanje

Ako je A kvadratna matrica i x ceo broj, tada izraz A^x predstavlja x -ti matrični stepen od A . Ukoliko je x realan broj, tada je $A^x = V * D.^x / V$ gde operator `.^` predstavlja stepenovanje matrica element po element, V je matrica desnih sopstvenih vektora, a D dijagonalna matrica sopstvenih vrednosti. Sledi primer:

```
>> A = [1 2 3; 4 5 6; 7 8 9]
```

```
>> A ^ 2
```

```
ans =
```

```
30 36 42
```

```
66 81 96
```

```
102 126 150
```

```
>> A .^ 2
```

```
ans =
```

```
1 4 9
```

```
16 25 36
```

```
49 64 81
```

2.2.4. LOGIČKI I RELACIONI OPERATORI U MATLABU

Relacioni operatori u MATLAB-u su sledeći simboli:

< (manje)

> (veće)

<= (manje ili jednako)

>= (veće ili jednako)

== (jednako)

~= (nije jednako)

Logički operatori u MATLAB-u su sledeći simboli:

~ (negacija)
| (disjunkcija)
& (konjunkcija)

Logički i relacioni operatori su namenjeni za formiranje logičkih izraza. MATLAB ne poseduje logički tip podataka kao poseban tip, već se kao vrednost logičkih izraza uzima 1 (ako je izraz tačan), odnosno 0 (ako je izraz netačan). Prilikom izračunavanja vrednosti logičkih izraza vrednost različite od nule se tretiraju kao tačni izrazi, a nula kao netačan izraz. Slede primeri korišćenja logičkih i relacionih operatora:

```
>> 2+2 == 4
```

```
ans =
```

```
1
```

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

```
>> B = A<5
```

```
B =
```

```
1 1 1
```

```
1 0 0
```

```
0 0 0
```

```
>> ~B
```

```
ans =
```

```
0 0 0
```

```
0 1 1
```

```
1 1 1
```

```
>> X = ((B(:, 1) | ~B(:, 2)) & B(:, 3))'
```

```
X =
```

```
0 0 1
```


3. FUNKCIJE ZA CRTANJE U MATLABU

MatLab sadrži bogat izbor funkcija koje služe za crtanje neke funkcije koja je zadatak eksplicitno, ali i implicitno. To se odnosi na funkcije kako od jedne tako i od dve promenljive.

U prvom koraku treba definisati podatke

```
>>t=0:0.1:10
```

```
>>x=sin(2*pi*t)
```

```
>>y=cos(2*pi*t)
```

Osnovni oblik naredbe "plot": na ordinati su indeksi elemenata vektora, a na apscisi sami elementi

```
>>plot(x)
```

Grafik vektora x u odnosu na vreme t (vektori moraju biti iste dužine).

```
>>plot(t,x)
```

Više linija na istom grafiku:

```
>>plot(t,x,t,y)
```

Grafik vektora y u odnosu na vektor x

```
>>plot(x,y)
```

Podešavanje opsega osa grafika pomoću naredbe axis([xmin xmax, ymin,ymax])

```
>>axis([0 2 -2 2])
```

```
>>v=[-2 2 0 1]
```

```
>>axis(v)
```

Probajte i

```
>>axis('equal')
```

```
>>axis('normal')
```

Sve opcije je moguće dobiti sa "help axis"

Naziv grafika i osa:

```
>>title('A nice sine-wave')
```

```
>>xlabel('Magnitude')
```

```
>>ylabel('time')
```

```
>>text(20,0.5,'A spot')
```

Mreža pomoćnih linija

```
>>grid
```

Naredba hold služi za nastavak crtanja na istom grafiku

```
>>hold on
```

```
>>plot(t,y)
```

```
>>hold off
```

Menjanje boja linija i označavanje tačaka.

```
>>plot(t,x,'r',t,y,'g')
```

Vektor x je u crvenoj, a y u zelenoj boji.

Tačke na grafiku se mogu označiti:

```
>>plot(t,x,'x')
```

Promena boja i označavanje istovremeno:

```
>>plot(t,x,'ro',t,y,'gx')
```

```
>>plot(t,x,'r-',t,y,'g-')
```

Sve oznake:

y žuta	. point
m magenta	o circle
c cyan	x x-mark
r crvena	+ plus
g zelena	- solid
b plava	* star
w bela	: dotted
k crna	-. dashdot
	-- dashed

Nacrtajte grafik date funkcije $x(t)$

```
>>t=0:0.1:10
```

```
>>x=exp(-t)+exp(-4t)
```

```
>>plot(t,x)
```

Generisanje više grafika je moguće na dva načina:

više grafika u istom grafičkom prozoru ili generisanje u posebnim prozorima.

Za više grafika u jednom grafičkom prozoru treba podeliti grafički prozor korišćenjem naredbe "subplot(mnp)", čime je jedan prozor podeljen na $m*n$ prozora, dok p označava potprozor u kome se crta grafik pomoću naredbi koje slede

```
>>t=0:0.1:10;
```

```
>>x=sin(t);
```

```
>>y=cos(t);
```

```
>>subplot(2,1,1)
```

```
>>plot(t,x); title('Sinusna funkcija');
```

```
>>subplot(2,1,2)
```

```
>>plot(t,y); title('Kosinusna funkcija');
```

Novi prozor:

```
>>figure(2)
```

```
>>plot(t,x)
```

```
>>figure(1)
```

```
>>plot(t,y)
```

Naredba za brisanje sadržaja grafičkog prozora

```
>>clg
```

2D Crtanje implicitno zadate funkcije

Za crtanje funkcija koje su implicitno zadate koristimo naredbu `ezplot`. Kompletan opis ove funkcije možemo dobiti naredbom `help ezplot`.

Osnovni oblik poziva ove naredbe je **`ezplot(fun2)`** gde se iscrtava implicitno zadata $fun2(x,y)$ sa podrazumevnim domenom $-2\pi < x < 2\pi$ i $-2\pi < y < 2\pi$.

Ostali oblici:

<code>ezplot(fun,[A,B])</code>	cрта $fun(x)$ preko $A < x < B$.
<code>ezplot(fun2,[A,B])</code>	cрта $fun(x)$ preko $A < x < B$ i $A < y < B$.
<code>ezplot(fun2, [XMIN,XMAX,YMIN,YMAX])</code>	cрта $fun(x)$ preko $XMIN < x < XMAX$ i $YMIN < y < YMAX$.

Primeri:

```
>> ezplot('x^2 - 2*x + 1')
>> ezplot('x.*y + x.^2 - y.^2 - 1')
>> ezplot('x.*y + x.^2 - y.^2 - 1',[-2,2,-4,4])
>> ezplot('x.*y + x.^2 - y.^2 - 1',[-2,2])
>> ezplot('x^2 - 2*x + 1',[-10,10]); grid on;
```

3D Crtanje funkcije dve promenljive

Ukoliko želimo da iscrtamo implicitno zadatu funkciju recimo da je to paraboloid koji ima jednačinu:

To možemo da uradimo na sledeći način:

```
>> syms x y                                definišemo simboličke promenljive, o ovome će biti više reči kasnije
>> ezmesh(x^2+y^2,[-2.5,2.5]);
>> ezmesh('sin(x)*cos(y)')
```

Argumenti funkcije `ezmesh` su slični argumentima funkcije `ezplot`. Detaljne informacije o ovoj funkciji možemo dobiti naredbom `help ezmesh`.

4. KORISNIČKE FUNKCIJE I RAD SA NJIMA

Da bismo krenuli da pišemo korisniču funkciju u MatLab-u potrebno je prvo da startujemo M-editor. Sve korisnicke funkcije se čuvaju u fajlovima koje imaju ekstenziju `m`. Kao što smo već ranije rekli ove funkcije nije potrebno kompajlirati pošto je MatLab interpreter stoga je dovoljno ispisati takvu funkciju, sačuvati je i posle je pozvati, sa odgovarajućim argumentima, iz komandnog prozora.

Funkcijska M-datoteka mora da u prvoj liniji sadrži izraz sledećeg oblika:

```
function [y1, y2, ...] = ime_funkcije (x1, x2, ...)
```

gde `y1,y2,...` predstavljaju rezultate izvršavanja funkcije, `ime_funkcije` je naziv MATLAB funkcije koja se definiše, a `x1,x2,...` su ulazni parametri funkcije. Promenljive koje su definisane pre poziva funkcije nisu vidljive unutar same funkcije. Slično tome, promenljive koje se definišu unutar funkcije nisu vidljive izvan nje. Ukoliko je potrebno da neke promenljive budu vidljive i izvan funkcije, moraju biti proglašene globalnim pomoću naredbe `global`. **Naziva funkcije mora biti jednak nazivu datoteke u kojoj je smeštena funkcija.**

Korisničke funkcije je moguće definisati i u okviru REX ili DLL datoteka (u verziji MATLAB-a za *Windows* operativne sisteme).

Dodatni skupovi funkcija organizovani u *toolbox*-ove se, nakon instaliranja, nalaze u nekom od poddirektorijuma `\matlab\toolbox` direktorijuma. Da bi stalno bile vidljive, promenljivu `matlabpath` definisanu u `matlabrc.m` datoteci treba dopuniti odgovarajućim vrednostima. Tu se nalaze primeri M, MAT, REX, i DLL funkcijskih datoteka.

MATLAB poseduje i skup komandi namenjen debugiranju komandnih i funkcijskih datoteka.

Kontrola toka MATLAB programa

MATLAB poseduje tri uobičajene strukture za kontrolu toka programa: *for*, *if* i *while*. Naredba *for* ima sledeći opšti oblik:

```
for brojac = vektor
```

```
    naredbe
```

```
    ...
```

```
end
```

pri čemu vektor može biti bilo koji vektor definisan u MATLAB-u. Promenljiva *brojac* će u svakoj iteraciji petlje kao vrednost primati odgovarajući element vektora.

Naredba *if* ima sledeći opšti oblik:

```
if izraz1
```

```
    naredbe1
```

```
[elseif izraz2
```

```
    naredbe2]...
```

```
[else
```

```
    naredben]
```

```
end
```

gde su *izraz1,izraz2,...* logički izrazi.

Naredba *while* ima sledeći opšti oblik:

```
while izraz
```

```
    naredbe
```

```
end
```

gde je *izraz* logički izraz.

sistem.m

```
A = [10 2 6; 1 10 9; 2 -7 -10];
```

```
b = [28 7 -17]';
```

```
x = A\b
```

Datoteka *sistem.m* se iz MATLAB okruženja poziva na sledeći način:

```
>> sistem
```

Tokom izvršavanja skripta će se u komandnom prozoru MATLAB-a ispisati:

```
x =
```

```
    1.0000
```

```
   -3.0000
```

```
    4.0000
```

faktorijal.m

```
%funkcija koja racuna faktorijal (n!) celog broja
```

```
function f = faktorijal(n)      % definisemo zaglavlje funkcije
```

```
if((n<0) || ~(fix(n)-n==0))    % proveravamo uslov da li je broj ceo i da li je pozitivan  
    disp('Argument faktorijala mora biti pozitivan ceo broj!');
```

```
elseif(n==0)                  % proveravamo uslov da li je broj jednak 0 jer je u tom  
    f=1;                       % slucaju faktorijal jednak 1
```

```
else                            % slucaj kada su ispunjeni svi uslovi br>0, br je ceo  
    f=1;                         % postavljamo pocetnu vrednost faktorijala na 1  
    while n>1                     % prolazimo kroz while petlju sve dok je broj veci od 1  
        n=n-1;                   % umanjujemo vrednosti broja za 1  
        f=f*n;                   % racunamo vrednost faktorijala  
    end                            % po zavrsetku f sadrzi vrednost faktorijala za zadati broj  
end                                % kraj if naredbe
```

```
end                                % kraj funkcije
```

Sledi primer funkcije za rešavanje matrične jednačine oblika $Ax=b$, pri čemu su matrice A i b ulazni argument funkcije, a x izlazni.

```
function x=mat_jna(A, b)
%resavanje maticne jednacine Ax=b
%matrice A i b se zadaju, matrica x je rezultat

v=size(A); %v je vektor ciji prvi element predstavlja broj vrsta
           %matrice A, a drugi elemnt broj kolona matrice A
w=size(b);
if (v(2)~=w(1) || v(1)~=v(2)) %proveravamo da li je matrica A kvadratna i
                               %da li su dimenzije matrica A i b
                               %odgovarajuće (da bi 2 matrice mogle da se
                               %pomnože potrebno je da broj kolona prve
                               %bude jednak broju vrsta druge matrice)
    disp('Formati nisu odgovarajuci') %ako nesto od prethodna dva
                                       %uslova nije zadovoljeno, ispisuje se
                                       %poruka
else if (det(A)==0)
    disp('Matrica je singularna')
else
    x=inv(A)*b; %ukoliko su formati u redu i postoji inverz matrice A
               %(A nije singularna) u x smestamo rezultat
               %funkcija inv vraća inverz matrice A
end
end
end
```

Sledeći primer ilustruje korišćenje *for* petlje. Funkcija računa uzoračku sredinu i uzoračku disperziju. Ulazni argument funkcije je vektor koji predstavlja uzorak, dok je izlazni argument vektor dužine dva, čiji prvi element prestavlja uzoračku sredinu, a drugi uzoračku disperziju datog uzorka.

```
function [uz_sredina, uz_disp]=stat(uzorak)
%uzorak je neki vektor (x_1, ..., x_n)
us=0; %inicijalna vrednost pomocne promenljive koju cemo koristiti
      %za racunanje uzoracke sredine
ud=0; %inicijalna vrednost pomocne promenljive koju cemo koristiti
      %za racunanje uzoracke disperzije
n=length(uzorak); %velicina uzorka

for i=1:n
    us=us+uzorak(i);
    ud=ud+(uzorak(i))^2;
end

uz_sredina=us/n; %uzoracka sredina=(x_1+...+x_n)/n
uz_disp=ud/n-(uz_sredina)^2; %uzoracka disperzija=((x_1)^2+...+(x_n)^2)/n-
                             %-(uzoracka sredina)^2

end
```

Prethodnu funkciju možemo pozvati na sledeći način:

```
>> [a b]=stat(u)
```

Gde je u neki vektor koji predstavlja uzorak.

5. SIMBOLIČKE PROMENLJIVE U MATLAB-U

Pored realnog tipa podataka MatLab poseduje i simbolički tip podataka. Uvodjenjem ovog tipa promenljivih omogućeno je da se složeni izrazi ili funkcije mogu računati bez njihovog evaluiranja u svakom koraku, već da se evaluaciju uradi u trenutku kada to korisniku odgovara. Ovaj originalan pristup je moguć pošto je kao što smo već ranije rekli MatLab pre svega interpreter. Ni u jednom programskom jeziku nemamo ovakav tip podataka. To nam samo daje uvid u moć ovog programa.

Simbolička promenljiva se definiše naredbom:

>> x=sym('x') definiše jednu simboličku promenljivu x

>> syms x y definiše dve logičke promenljive x i y

Nakon čega u radnom prostoru možemo videti naše definisane logičke promenljive. Primetimo da se njihovo predstavljanje u radnom prostoru razlikuje od predstavljanja osnovnih promenljivih. Kucanjem naredbe *whos* možemo primetiti da ovaj ti promenljivih koristi više memorije za zapis od ostalih.

Recimo da hoćemo da pomnožimo polinome:

$P_1 = (4*x^5 - 3*x^4 + 6*x^3 + 2*x^2 - x)$ i $P_2 = (x^4 + 134*x^3 - 12*x^2)/(x^2 + x + 1)$

To se u MatLabu možemo predstaviti na sledeći način:

>> p = (4*x^5 - 3*x^4 + 6*x^3 + 2*x^2 - x) * ((x^4 + 134*x^3 - 12*x^2)/(x^2 + x + 1))

Ako uradimo sledeće:

>> pp = expand(p)

MatLab će izračunati proizvod ovih polinoma. A ukoliko uradimo

>> pretty(pp)

Dobijamo lep zapis dobijenog proizvoda.

Funkcija *expand* međusobno množi, primenjuje trigonometrijske formule, pravila koja se odnose na eksponencijalnu i logaritamsku funkciju na članove koji sadrže operacije sabiranja i oduzimanja:

>> expand(sin(x-y))

ans =

$\sin(x)\cos(y) - \cos(x)\sin(y)$

Ukoliko imamo polinom oblika: $g = x^3 + 6x^2 + 11x + 6$;

Naredbom:

```
>>factor(g)
```

Dobićemo ovaj polinom razložen na faktore.

Recimo da sada želimo da izračunamo vrednost gornjeg polinoma p u tački $x=3.2$. Sve što treba da uradimo jeste:

```
>>subs(p,3.2)
```

i dobićemo vrednost polinoma u željenoj tački.

Slučaj za dve promenljive:

```
>>syms x y;
```

```
>>f = x^2*y + 5*x*sqrt(y);
```

```
>>ff=subs(f, x, 3)
```

```
>>ff =
```

```
9*y + 15*y^(1/2)
```

Pa zatim:

```
>>subs(ff, y, 2)
```

Dobili smo vrednost funkcije u tački (3,2)

Ukoliko želimo da *diferenciramo* naš polinom po x, to možemo uraditi na sledeći način:

```
>> pp= diff(p)
```

Pošto rezultat i nije baš lep možemo ga „ulepšati“

```
>> pretty(pp)
```

Slično ako želimo da izračunamo *neodređeni integral*.

```
>> p=1/x
```

```
>> int(p)
```

```
>> int(3*cos(x)-12*x)
```

```
ans =
```

```
3*sin(x)-6*x^2
```

Korišćenjem simboličkih promenljivih moguće je *rešiti algebarsku jednačinu i sistem jednačina*:

```
>> solve(exp(2*x)-5)
```

```
ans =
```

```
1/2*log(5)
```

```
>> syms x y z; [x, y, z] = solve('z = 4*x', 'x = y', 'z = x^2 + y^2')
```

Prodnosti korišćenja simboličkih promenljivih su velike. Jedina mana pri korišćenju ovih promenljivih jeste ta što njihovo korišćenje usporava rad MatLab programa. Recimo ukoliko u for petlji radimo nešto sa simboličkom promenljivom i ukoliko se izvodi veliki broj iteracija pre nego što se petlja završi, vreme izvršavanja programa će se dosta produžiti.

6. PREGLED NEKIH MATLAB DEFINISANIH FUNKCIJA I SPECIJALNIH VREDNOSTI

Neke funkcije

sin(X)

Vraća matricu čiji su elementi vrednosti sinusa odgovarajućih elemenata matrice X. Funkcija očekuje vrednosti za veličinu ugla u radijanima.

exp(X)

Vraća matricu čiji su elementi vrednosti funkcije e^x za odgovarajuće elemente matrice X.

```
>> t = (0:0.2:1)';
```

```
>> y = sin(t) .* exp(-t);
```

```
>> [t y]
```

```
ans =  
    0    0  
0.2000 0.1627  
0.4000 0.2610  
0.6000 0.3099  
0.8000 0.3223  
1.0000 0.3096
```

max(X)

Vraća vektor koji sadrži maksimalne elemente po kolonama matrice X.

mean(X)

Vraća vektor koji sadrži aritmetičke sredine elemenata u odgovarajućim kolonama.

size(X)

Vraća dimenzije argumenta u obliku vektora.

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

```
>> max(A)
```

```
ans =  
    3    6    9
```

```
>> mean(A)
```

```
ans =  
    4    5    6
```

```
>> size(X)
```

```
ans =  
    3    3
```

length(X)

Vraća veću dimenziju argumenta.

ones(m,n)

Vraća matricu dimenzije m x n čiji su elementi jedinice.

zeros(m,n)

Vraća matricu dimenzije mxn čiji su elementi nule.

eye(m,n)

Vraća jediničnu matricu dimenzije mxn.

sort(X)

Rezultat je matrica dobijena sortiranjem kolona matrice X u rastućem redosledu.

inv(X)

Vraća inverznu matricu kvadratne matrice X.

det(X)

Vraća determinantu matrice X.

rank(X)

Rezultat funkcije je rang matrice X.

cond(X)

Vraća kondicioni broj matrice X.

norm(X)

Vraća normu matrice X.

eig(X)

Izraz *eig(X)* predstavlja vektor čiji su elementi karakteristične vrednosti matrice X. Ukoliko se funkcija pozove sledećim izrazom:

```
>> [V, D] = eig(X);
```

tada je D dijagonalna matrica sa karakterističnim vrednostima matrice X, a V je matrica čije su kolone odgovarajući desni karakteristični vektori.

poly(X)

Ako je X kvadratna matrica dimenzije n, tada je *poly(X)* vektor sa n+1 elementom koji predstavljaju koeficijente karakterističnog polinoma matrice X. Ukoliko je X vektor, tada je *poly(X)* vektor čiji su elementi koeficijenti polinoma koji za svoje korene ima elemente vektora X.

roots(X)

Rezultat funkcije je vektor čiji su elementi koreni polinoma datog koeficijentima u vektoru X.

Specijalne vrednosti

pi	konstanta 3.141592653589793e+000
i,j	imaginarna jedinica ($\sqrt{-1}$)
inf	matlab oznaka za beskonačno (∞)
NaN	Not-a-Number (nije broj)
clock	vreme i datum
date	datum

ans	ako rezultatu izračunavanja nije dodeljena promenljiva, podaci se uvek smeštaju na ovu lokaciju
eps	tačnost realnih brojeva(razlika između 1.0 i prvog većeg) ($\text{eps}=2.2204\text{e-}16$)
realmin	vrednost najmanjeg realnog broja ($2.2250733858507202\text{e-}308$)
realmax	vrednost najvećeg realnog broja ($1.797697134862316\text{e+}308$)