

MATLAB – Kratko uputstvo za korišćenje

Matlab je programski paket koji je počeo da se razvija još 80-tih godina pa su se do danas na tržištu pojavile brojne verzije ovog programa. One se uglavnom razlikuju po broju raspoloživih funkcija, sistemskim ograničenjima, grafičkom interfejsu kao i po operativnom sistemu pod kojim su implementirane. S obzirom na veliki broj raspoloživih funkcija, one su grupisane u posebne pakete (takozvane Toolbox-ove), koji se posebno nabavljaju uz osnovnu verziju programa. Tako postoje Signal Processing Toolbox, Control System Toolbox i mnogi drugi.

Osnovni elementi svih verzija Matlab-a su isti, tako da kada se jednom nauči način na koji se u Matlabu predstavljaju podaci, sprovede osnovne računске operacije i programiraju sopstvene funkcije, prelazak sa jedne verzije programa na drugu ne predstavlja nikakav problem. Zbog toga će se u ovom tekstu izložiti osnovi korišćenja Matlab-a i predstavice se najvažnije osnovne komande i funkcije, nakon čega je lako realizovati neke nove i složenije operacije.

1. Osnovni elementi Matlab-a

Po startovanju programa otvara se komandni prozor i pojavljuje se simbol `>>`, koji se naziva “prompt”, a koji označava da je Matlab spreman da prihvati podatke ili naredbe. Tekst u komandnom prozoru se briše naredbom `clc`.

Osnovni objekt u Matlabu je matrica dimenzija $m \times n$, a sve elementarne operacije su tako definisane da podržavaju rad sa matricama. Time je omogućeno da se sve matematičke i logičke operacije i iskazi u Matlabu definišu i izvode na isti način kao što bi ih pisali na papiru. Vektori su posebni slučajevi matrica $1 \times n$ ili $n \times 1$, dok su skalarne veličine poseban slučaj matrice 1×1 .

1.1 Unošenje podataka pomoću tastature

Postoji više načina za unošenje podataka koje treba obraditi: pomoću tastature, generisanjem pomoću posebnih funkcija, učitavanjem iz fajla, ili kao rezultat neke prethodne operacije.

Pretpostavimo da u Matlab treba uneti vrednost skalarne promenljive $x = 5$. To se može uraditi na sledeći način. Kada se pojavi prompt `>>`, ukucavanjem:

```
>> x=5;
```

vrednost promenljive x se smešta u radnu memoriju Matlab-a i može se pozivati i koristiti sve dok se: a) ne izađe iz programa, b) ne predefiniše unošenjem neke druge vrednosti ili izvršavanjem određene operacije, c) ne izbriše iz radne memorije komandom `clear x`, d) komandom `clear` ne izbriše cela radna memorija. Ukoliko se prilikom bilo kakvog dodeljivanja vrednosti promenljivoj izostavi znak `;` ta promenljiva će se još i štampati na ekranu, tako da naredba:

```
>> x=5
```

daje kao rezultat na ekranu:

```
x =  
    5
```

Pretpostavimo sada da u Matlab treba uneti vrednost vektora \mathbf{b} , koji ima sledeće elemente `[1 4 5 6 7 12.3]`. To se može uraditi na sledeći način:

```
>> b=[1 4 5 6 7 12.3];
```

Matrica **A** dimenzija 4×3, zadata kao:

$$\mathbf{A} = \begin{bmatrix} 1 & 7 & 8 \\ 9 & 2 & 11 \\ 0 & 3 & 2.1 \\ 2.4 & 8 & 100 \end{bmatrix}$$

može se uneti u memoriju na više načina. Jedan od njih je:

```
>> A=[1 7 8; 9 2 11; 0 3 2.1; 2.4 8 100];
```

Matrica se može zadati i unošenjem svakog elementa posebno:

```
>> A(1,1)=1;A(1,2)=7;A(1,3)=8;A(2,1)=9;A(2,2)=2;A(2,3)=11;
>> A(3,1)=0;A(3,2)=3;A(3,3)=2.1;A(4,1)=2.4;A(4,2)=8;A(4,3)=100;
```

Takođe, matrica se može uneti zadavanjem vrsta ili kolona

```
>> A(1,:)= [1 7 8];A(2,:)= [9 2 11];A(3,:)= [0 3 2.1];A(4,:)= [2.4 8 100];
>> A(:,1)= [1 9 0 2.4];A(:,2)= [7 2 3 8];A(:,3)= [8 11 2.1 100];
```

1.2 Osnovne operacije u Matlabu

Matrice se mogu generisati i kao rezultat osnovnih aritmetičkih ili logičkih operacija. Na primer, ako se pođe od već zadate matrice **A**, kucanjem:

```
>> C=A' ;
```

dobija se transponovana matrica **C**, pošto je **'** operator za transpoziciju.

U sledećoj tabeli su prikazane elementarne matematičke operacije nad matricama u Matlabu:

Operacija	Simbol	Primer
sabiranje	+	>> B = A+C;
oduzimanje	-	>> B = A-C;
sabiranje	+	>> B = A+C;
deljenje	/	>> Y = 1/x;
stepenovanje y^x	^	>> z = y^x;
množenje matrica	*	>> D = D*E; po pravilima matričnog množenja

Ukoliko su matrice neodgovarajućih dimenzija, program prijavljuje grešku. Na primer:

```
??? Error using ==> *
Inner matrix dimensions must agree.
```

U opštem slučaju, ako se operator **.** stavi ispred simbola neke matematičke operacije, time se označava da će se ta operacija sprovoditi između odgovarajućih elemenata matrica. Tako je:

```
>> D = D.*E množenje matrica element po element
>> F = D.^E stepenovanje svakog elemenata matrice D odgovarajućim elementom matrice E
>> F = D./E deljenje svakog elemenata matrice D odgovarajućim elementom matrice E
```

dok je:

```
>> F=D\E deljenje matrica D i E sleva, odn.  $F = D \cdot E^{-1}$ 
>> F=D/E deljenje matrica D i E sdesna, odn.  $F = D^{-1} \cdot E$ 
```

1.3 Osnovne matematičke funkcije u Matlabu

U Matlabu je implementiran izvestan broj elementarnih matematičkih funkcija, među kojima su najvažnije prikazane u Tabeli 1. Ove funkcije daju skalarne vrednosti ako im je argument

skalarna veličina, ako je argument vektor ili matrica, operacija se primenjuje na svaki lement vektora ili matrice. Detaljno objašnjenje na koji način se svaka funkcija koristi može se dobiti naredbom `help ime_funkcije`, na primer, `help abs`.

Tabela 1: Elementarne matematičke funkcije.

<code>Sin (cos)</code>	sinus (kosinus)	<code>sqrt</code>	kvadratni koren
<code>tan</code>	tangens	<code>abs</code>	apsolutna vrednost
<code>Asin (acos)</code>	inverzni sinus (kosinus)	<code>Real (imag)</code>	realni (imaginarni) deo
<code>atan</code>	inverzni tangens	<code>conj</code>	konjugovani broj
<code>Sinh (cosh)</code>	inverzni hiperbolički sinus (kosinus)	<code>fix</code>	zaokruživanje ka nuli
<code>tanh</code>	hiperbolički tangens	<code>floor</code>	zaokruživanje prema $-\infty$
<code>exp</code>	e^x	<code>ceil</code>	zaokruživanje prema $+\infty$
<code>log</code>	prirodni logaritam	<code>sign</code>	signum funkcija
<code>log10</code>	logaritam sa osnovom 10	<code>rem</code>	ostatak pri deljenju

1.4 Kompleksni brojevi u Matlabu

Jedna od značajnih osobina Matlaba je što ovaj programski paket radi i sa kompleksnim brojevima. Imaginarna jedinica $j = \sqrt{-1}$ je u Matlabu definisana kao permanentna veličina, a zbog fleksibilnosti u radu dodeljena su joj dva simbola, `i` i `j`, tako da je

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

odnosno

```
>> j = sqrt(-1);
```

Isto tako, korisnik može vrednost `sqrt(-1)` da dodeli proizvoljnoj promenljivoj `i` da nju u daljem radu koristi kao imaginarnu jedinicu, npr:

```
>> nova_im_jed = sqrt(-1);
```

Kompleksni brojevi se u Matlabu zadaju isto kao što bi se napisali na papiru:

```
>> z = 2+3*j;
>> q = 7-2.1*j;
```

i nad njima se po pravilima kompleksnog računa mogu izvoditi sve do sada pomenute operacije:

```
>> x = z+q;
>> x1 = z-q;
>> x2 = x*x1;
>> x_realno = real(x);      realni deo kompleksnog broja
>> x_imag = imag(x);       imaginarni deo kompleksnog broja
```

Kompleksna promenljiva može da se zada i preko modula i argumenta u sledećem obliku:

```
>> z = r*exp(j*teta);
```

tako da se unošenjem:

```
>> r = 2;
>> teta = 0.5;
>> z = r*exp(j*teta)
```

dobija rezultat:

```
z =
    1.7552 + 0.9589i
```

Kompleksne matrice se unose na isti način kao i matrice sa realnim elementima, ili unošenjem realnog i imaginarnog dela posebno (zadavanjem dve matrice).

1.5 Automatsko generisanje nekih specijalnih matrica u Matlabu

Za formiranje matrica specijalne strukture, u Matlabu postoje posebne naredbe: `ones`, `zeros`, `eye`. Tako se npr. naredbom:

```
>> A = zeros(2,3);
```

generiše matrica **A**, koja ima dve vrste i tri kolone, a čiji su svi elementi jednaki nuli. Isto tako:

```
>> B = ones(4,2);
```

generiše matricu **B**, dimenzija 4×2 , kod koje su svi elementi jednaki jedinici, dok se naredbom:

```
>> I = eye(7);
```

dobija jedinična matrica **I**, dimenzija 7×7 . Navedene naredbe se često primenjuju u radu, jer omogućavaju lako generisanje osnovnih signala koji se koriste u analizi diskretnih sistema. Tako se, na primer, pravougaoni impuls trajanja 10 odbiraka može generisati naredbom:

```
>> prav_impuls = [zeros(1,10) ones(1,10) zeros(1,10)];
```

a povorka od 5 takvih pravougaonih impulsa, amplitude 2, generiše se naredbama:

```
>> impuls = 2*[ones(1,5) zeros(1,10)];
```

```
>> povorka = [impuls impuls impuls impuls impuls];
```

Za automatsko generisanje vektora veoma često se koristi operator `:`. Tako se kucanjem:

```
>> x = 3:8
```

dobija:

```
x =  
    3    4    5    6    7    8
```

odnosno, generiše se vektor-vrsta koji sadrži elemente od 3 do 8. Opštiji oblik ove naredbe je `x = xmin:skorak:xmax`. Na taj način se generiše vektor **x** koji kao elemente ima brojeve od *xmin* do *xmax* sa korakom *skorak*. Ako je *skorak* negativan broj, a *xmin* veće od *xmax* dobija se vektor sa elementima u opadajućem redosledu.

1.6 Ugrađene vrednosti u Matlabu

Osim imaginarne jedinice, u Matlabu postoje još neke ugrađene vrednosti. Na primer, broj π je definisan kao promenljiva `pi` koja se u radu koristi kao i ostale promenljive. Tako se kucanjem:

```
>> pi
```

dobija na ekranu:

```
ans =  
    3.14159265358979
```

ili:

```
>> y = cos(pi/3)
```

daje kao rezultat:

```
y =  
    0.500000000000000
```

Posebna pogodnost Matlabu su i veličine `Inf` i `NaN`. Veličina `Inf` je skraćenica od engleske reči infinity (beskonačno) i može se generisati nekom od naredbi tipa:

```
>> x = 1/0
```

nakon čega se na ekranu dobija sledeća poruka:

```
Warning: Divide by zero  
x =  
    Inf
```

Ova i slične naredbe ne izazivaju prekid u radu (osim što se na ekranu dobija poruka upozorenja), a veličina `Inf` se u daljem računanju tretira kao i sve druge vrednosti. Veličina `NaN` potiče od engleskog izraza “not a number” i može se generisati operacijama `Inf/Inf` ili `0/0`. Ona se korisno može upotrebiti za određivanje vrednosti funkcija čiji je argument izvan opsega dozvoljenih vrednosti, kao što je, na primer, vrednost diskretnog signala za necelobrojnu vrednost indeksa.

1.7 Funkcije za pomoć u Matlabu

Verovatno najčešće korišćena funkcija u Matlabu je `help`. Kako je broj funkcija u Matlabu izuzetno veliki, a mnoge imaju i više varijanti, nemoguće je znati sintaksu i namenu svake od njih, pa naredbu `help` često upotrebljavaju i iskusni programeri. Na primer, ako je potrebno upoznati se sa naredbom `max` (koja određuje maksimalnu vrednost elementa nekog vektora), kucanje:

```
>> help max
```

daje sledeću poruku na ekranu:

```
MAX      Largest component.
For vectors, MAX(X) is the largest element in X. For matrices,
MAX(X) is a row vector containing the maximum element from each
column. For N-D arrays, MAX(X) operates along the first
non-singleton dimension.

[Y,I] = MAX(X) returns the indices of the maximum values in vector I.
If the values along the first non-singleton dimension contain more
than one maximal element, the index of the first one is returned.

MAX(X,Y) returns an array the same size as X and Y with the
largest elements taken from X or Y. Either one can be a scalar.

[Y,I] = MAX(X,[],DIM) operates along the dimension DIM.

When complex, the magnitude MAX(ABS(X)) is used, and the angle
ANGLE(X) is ignored. NaN's are ignored when computing the maximum.

Example: If X = [2 8 4   then max(X,[],1) is [7 8 9],
                7 3 9]

                max(X,[],2) is [8   and max(X,5) is [5 8 5
                9],                        7 5 9].

See also min, median, mean, sort.
```

Još jedna dobra strana naredbe `help` je što daje i spisak srodnih funkcija, što olakšava rad.

Za informaciju o zauzeću radne memorije Matlaba i promenljivima koje postoje, koriste se naredbe `who` i `whos`. Naredba `who` daje spisak promenljivih koje su u radnoj memoriji, dok `whos` daje još i podatak o količini memorije koju zauzima svaka promenljiva i veličini preostale slobodne radne memorije. Ove naredbe se koriste u slučaju rada sa velikim blokovima podataka (dugačkim vektorima ili matricama velikih dimenzija), kada zauzeće memorije postaje kritičan faktor.

Za ilustraciju različitih mogućnosti paketa Matlab, postoji naredba `demo`. Kucanjem ove naredbe na ekranu će se pojaviti meni sa izborom različitih demonstracionih programa.

Za određivanje dimenzija pojedinih promenljivih postoje naredbe `length` i `size`. Naredba `length` daje broj elemenata određenog vektora. Tako na primer, kucanjem:

```
>> x = 0:0.1:1; nx=length(x)
```

dobijamo na ekranu vrednost promenljive `nx` koja predstavlja broj elemenata vektora `x`, kao:

```
nx =
    11
```

Naredba `size` služi za određivanje dimenzija matrice. Ona kao izlaz daje vektor sa dva

elementa. Prvi element vektora je broj vrsta u matrici a drugi je broj kolona. Tako na primer, ako u radnoj memoriji postoji matrica **A** veličine 2×3, kucanjem:

```
>> dim = size(A);
```

dobijamo vektor **dim** koji je jednak [2 3]. Ili kucanjem:

```
>> [x,y] = size(A)
```

promenljivima **x** i **y** dodeljujemo vrednosti 2 i 3.

2. Grafički prikaz podataka u Matlabu

Odlične grafičke mogućnosti su jedna od najboljih osobina Matlab, pošto postoji veliki broj naredbi koje omogućavaju crtanje podataka u dve ili tri dimenzije. Osim toga, Matlab omogućava čuvanje dobijenih grafika u mnogobrojnim standardnim formatima. Na taj način se slike napravljene u Matlabu mogu koristiti u drugim programskim paketima.

2.1 Grafičke komande

Prozor za crtanje grafika se otvara naredbom `figure`, a zatvara (briše) naredbom `close`. Naredba `close all` zatvara sve otvorene prozore.

Grafik se briše bez zatvaranja prozora naredbom `clf`. Zadavanjem bilo koje od naredbi za crtanje (`plot`, `stem`, i `sl`), prethodni grafik se briše i u tekućem prozoru se crta novi grafik. Ukoliko želimo da se stari grafik zadrži, a da se preko njega crtaju novi, koristi se naredba `hold on`. Ova naredba se poništava sa `hold off`.

2.2 Naredba `plot`

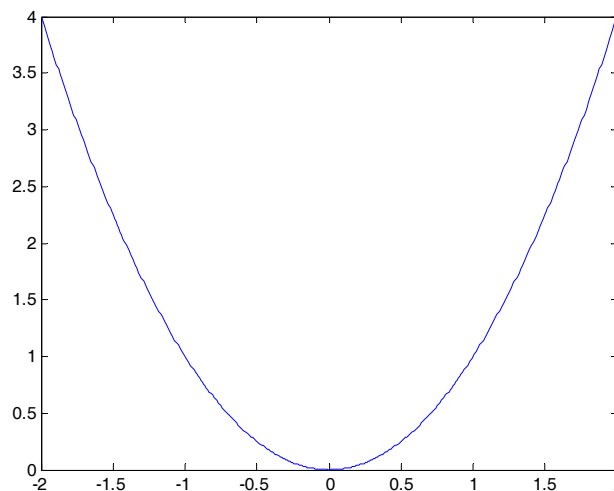
Najčešće korišćena naredba za prikaz podataka je `plot`. Naredba `plot` ima veliki broj opcija te će se u daljem izlaganju njena upotreba ilustrovati primerima, a opširniji prikaz se može dobiti kucanjem `help plot`.

Primer 1.

Nacrtati funkciju $y = x^2$ na intervalu -2 do 2.

Crtanje odgovarajućeg grafika izvodi se sa sledeće tri naredbe:

```
>> x = -2:0.01:2;
>> y = x.^2;
>> plot(x,y)
```



Slika 1: Grafički prikaz funkcije $y = x^2$ pomoću naredbe `plot`.

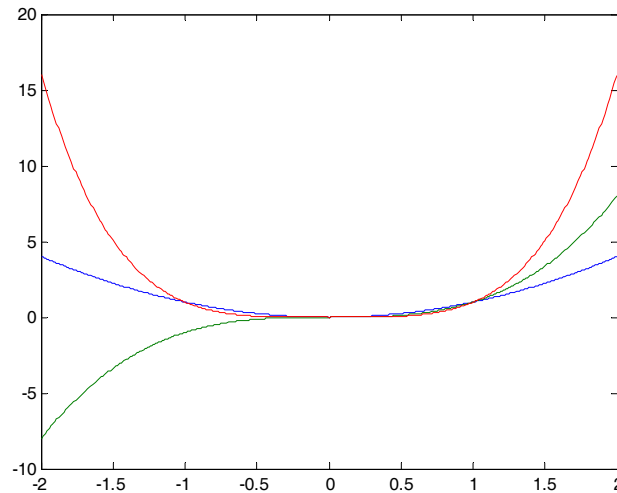
Kao što se vidi, na grafiku je izvršen automatski izbor opsega po obe ose.

Primer 2.

Nacrtati funkcije $y = x^2$, $z = x^{3/2}$ i $w = x^{5/2}$ na intervalu -2 do 2 .

Ukoliko je potrebno, na istoj slici se može nacrtati više dijagrama. Na primer, tri tražene krive mogu se generisati i predstaviti na istoj slici sledećim skupom naredbi:

```
>> x = -2:0.01:2;
>> y = x.^2;
>> z = x.^3;
>> w = x.^4;
>> plot(x,y,x,z,x,w)
```



Slika 2: Grafički prikaz funkcija $y = x^2$, $z = x^{3/2}$ i $w = x^{5/2}$ pomoću naredbe `plot`.

Na prethodnoj slici su tri krive nacrtane različitim bojama. Boja krive, simboli kojima su označene izračunate tačke (markeri), kao i tip linije koja spaja tačke, mogu se izabrati tako što se u naredbi `plot` pored nezavisne i zavisne promenljive za svaku krivu zadaju još najviše tri simbola 'mnp' s sledećim značenjem:

m – boja linije

b – plava

g – zelena

r – crvena

c – cijan (plavo-zelena)

m – magenta (ljubičasta)

y – žuta

b – crna

n – marker tačaka

. – tačka

o – kružić

x – znak x

+

* – zvezdica

s – kvadratić

d – dijamant

v – trougao na dole

^ – trougao na gore

< – trougao na levo

> – trougao na desno

p – petougao

h – šestougao

p – vrsta linije

– – puna linija

: – tačkasta linija

-. – tačka-crtica

-- – crtice

ništa – nema linije

Na primer, naredba `plot(x,y,'c+:')` nacrtaće krivu y cijan bojom, tačke će biti označene plusevima, a biće spojene tačkastom linijom. Prilikom crtanja se koristi linearna interpolacija između postojećih tačaka.

Ako boja krive nije eksplicitno specificirana, onda se boja krive određuje po redosledu crtanja kao: plava, zelena, crvena, cijan, magenta, žuta, crna. Ako nije specificiran marker, onda se tačke ne označavaju. Ako nije specificirana vrsta linije, onda se crta puna linija.

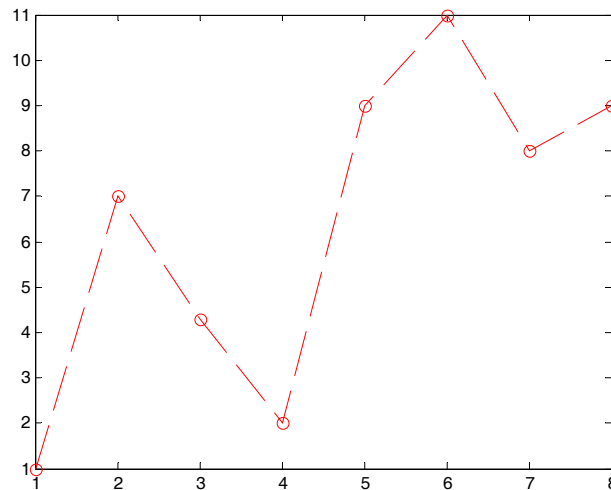
Ako u naredbi `plot` postoji samo jedan argument, to znači da će se vrednosti tog vektora nanositi na ordinatu. U tom slučaju se na apcisu nanose indeksi elemenata u vektoru.

Primer 3.

Nacrtati vektor podataka $x = [1 \ 7 \ 4.3 \ 2 \ 9 \ 11 \ 8 \ 9]$. Vrednosti podataka označiti kružićima, a tačke spojiti isprekidanom crvenom linijom.

Vektor x zadaje se i crta na sledeći način:

```
>> x = [1 7 4.3 2 9 11 8 9];
>> plot(x, 'ro--')
```



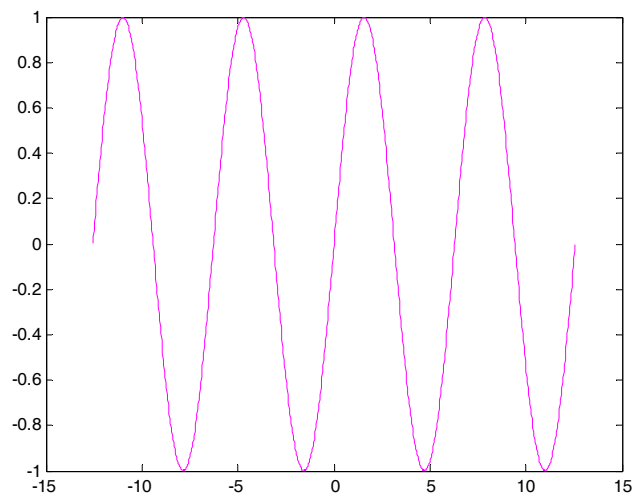
Slika 3: Grafički prikaz vektora x pomoću naredbe `plot`.

Primer 4.

Nacrtati funkciju $y = \sin(x)$ za vrednosti argumenta u opsegu $-4\pi \leq x \leq 4\pi$. Krivu nacrtati punom ljubičastom linijom, a vrednosti tačaka ne označavati.

Traženo rešenje je:

```
>> x = -4*pi:pi/100:4*pi;
>> y = sin(x);
>> plot(x,y, 'm')
```



Slika 4: Grafički prikaz funkcije $y = \sin(x)$ pomoću naredbe `plot`.

2.3 Naredba `stem`

Naredba za grafički prikaz diskretnih podataka je `stem`, koja prikazuje diskretne podatke tako da od svake tačke izvlači vertikalnu liniju do x ose. Slično naredbi `plot` i naredba `stem` ima mnogo opcija, pa će se u daljem izlaganju njena upotreba ilustrovati kroz primere, dok se detaljniji prikaz njenih mogućnosti može dobiti kucanjem `help stem`. Osnovna verzija naredbe `stem`:

```
>> stem(x)
```

crta vektor `x` u funkciji indeksa elemenata vektora. Kao marker diskretnih tačaka koristi se kružić.

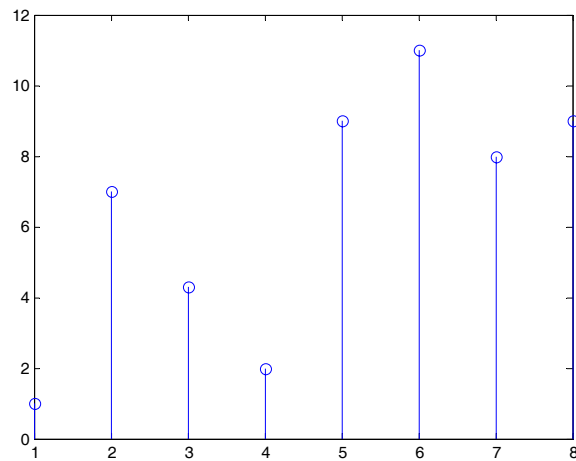
U složenijim verzijama, kao što je `stem(x,y,'mnp')` specificiraju se koordinate tačaka po obe ose, boja i tip linija, izgled markera. Značenje parametara `'mnp'` isto je kao kod naredbe `plot`.

Primer 5.

Nacrtati vektor podataka `x = [1 7 4.3 2 9 11 8 9]` pomoću naredbe `stem`.

Vektor `x` zadaje se i crta na sledeći način:

```
>> x = [1 7 4.3 2 9 11 8 9];
>> stem(x)
```



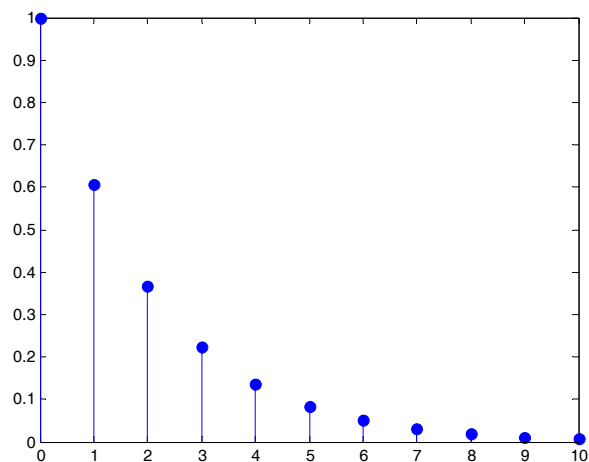
Slika 5: Grafički prikaz vektora `x` pomoću naredbe `stem`.

Primer 6.

Nacrtati diskretnu eksponencijalnu funkciju $y = e^{an}$ za $a = -0.5$ i $0 \leq n \leq 10$ pomoću naredbe `stem`.

Funkcija se crta pomoću sledećih naredbi. Opcija `'filled'` izaziva popunjavanje kružića.

```
>> n = 0:1:10;
>> y = exp(-0.5*n);
>> stem(n,y,'filled')
```



Slika 6: Grafički prikaz diskretne eksponencijalne funkcije $y = e^{an}$ pomoću naredbe `stem`.

2.4 Označavanje osa i grafika

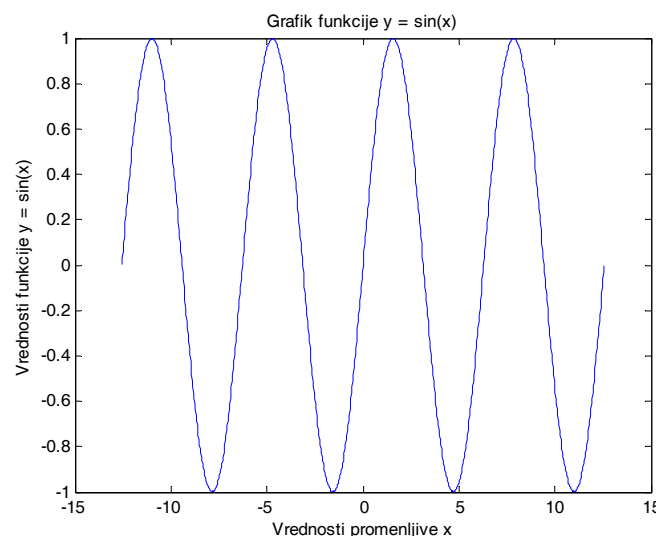
Označavanje grafika i osa, obavlja se u Matlabu naredbama `title`, `xlabel` i `ylabel`. Upotreba ovih naredbi može se ilustrovati sledećim primerom:

Primer 7.

Nacrtati funkciju $y = \sin(x)$ za vrednosti argumenta u opsegu $-4\pi \leq x \leq 4\pi$. Krivu nacrtati punom plavom linijom, a vrednosti tačaka ne označavati. Ispisati naziv grafika i označiti koordinatne ose.

Za rešavanje ovog problema mogu se iskoristiti naredbe napisane u Primeru 4, kojima treba dodati naredbe za ispisivanje naziva i označavanje koordinatnih osa:

```
>> x = -4*pi:pi/100:4*pi;
>> y = sin(x);
>> plot(x,y,'b')
>> title('Grafik funkcije y = sin(x)')
>> xlabel('Vrednosti promenljive x')
>> ylabel('Vrednosti funkcije y=sin(x)')
```



Slika 7: Grafički prikaz funkcije $y = \sin(x)$ pomoću naredbe `plot` sa označavanjem osa i naslovom.

2.5 Grafici sa logaritamskom podelom

Grafici sa logaritamskom podelom po x osi, po y osi ili sa logaritamskom podelom na obe ose, mogu se dobiti naredbama `semilogx`, `semilogy` i `loglog`. Prilikom crtanja grafika korišćenjem naredbi `semilogx` i `loglog`, potrebno je još generisati nezavisno promenljivu x u logaritamskoj razmeri, što se postiže naredbom `logspace`.

Primer 8.

Nacrtati funkciju $y = 1/(x^3 + x^2 - x + 1)$ za vrednosti argumenta u opsegu $0.1 \leq x \leq 10$ sa logaritamskom razmerom po x osi i po obe ose.

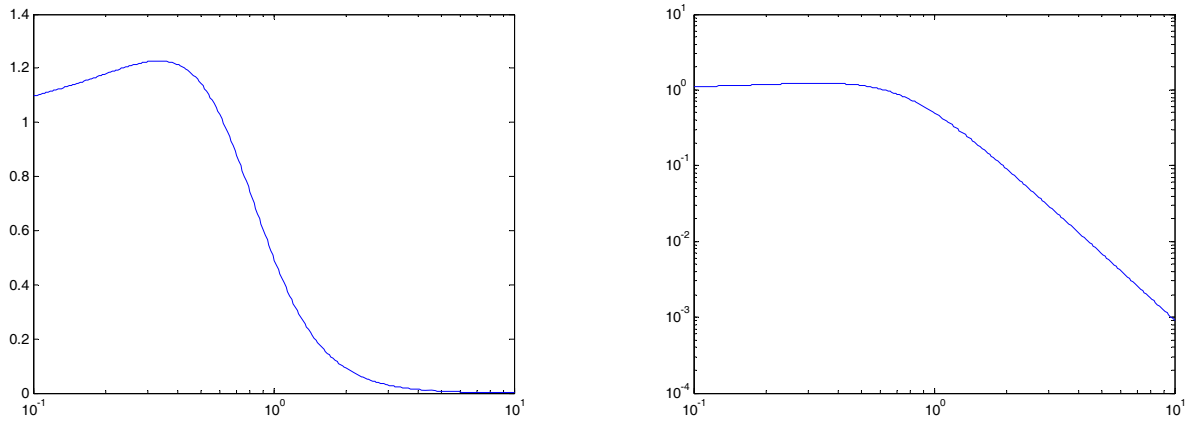
Funkcija $y(x)$ se crta u logaritamskoj razmeri (samo po x osi) pomoću naredbi:

```
>> x=logspace(-1,1,256);
>> y=1./(x.^3+x.^2-x+1);
>> semilogx(x,y)
```

dok se u slučaju crtanja u logaritamskoj razmeri po obe ose koristi se naredba `loglog(x,y)`.

```
>> loglog(x,y)
```

Traženi grafici su prikazani na slici 8.



Slika 8: Grafički prikaz funkcije $y = 1/(x^3 + x^2 - x + 1)$ u logaritamskoj razmeri po x osi i po obe ose.

2.6 Skaliranje osa

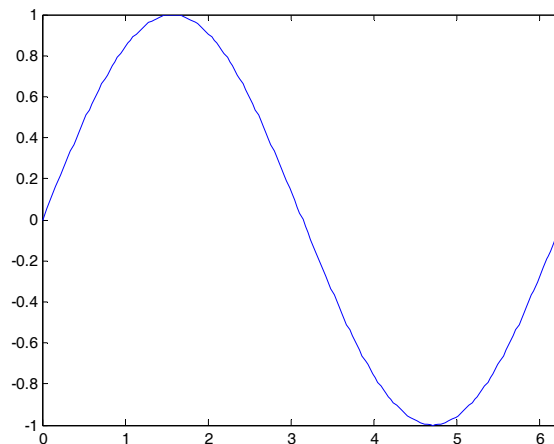
Prilikom crtanja grafika, opseg vrednosti na apcisi i ordinati određuje se automatski, na osnovu minimalne i maksimalne vrednosti promenljivih koje se zadaju u naredbi za crtanje (`plot`, `semilogx`, itd.). Ukoliko je potrebno nacrtati samo deo celokupnog opsega, koristi se naredba `axis`. Ulazni argument je vektor `[xmin xmax ymin ymax]` kojim se zadaje opseg za crtanje po x i y osi.

Primer 9.

Izračunati funkciju $y = \sin(x)$ za vrednosti argumenta u opsegu $-4\pi \leq x \leq 4\pi$, a nacrtati grafik za samo jedan period sinusoide.

Traženo rešenje je:

```
>> x = -4*pi:pi/100:4*pi;
>> y = sin(x);
>> plot(x,y)
>> axis([0 2*pi -1 1]);
```



Slika 9: Grafički prikaz jednog perioda funkcije $y = \sin(x)$ pomoću naredbe `plot`.

Od interesa su i sledeće varijante naredbe `axis`:

Naredba `axis manual` zamrzava tekuće granice `[xmin xmax ymin ymax]`. Ako je pri tome aktivirana naredba `hold on`, svi naredni grafici će koristiti iste granice. Automatsko određivanje granica za crtanje se vraća naredbom `axis auto`.

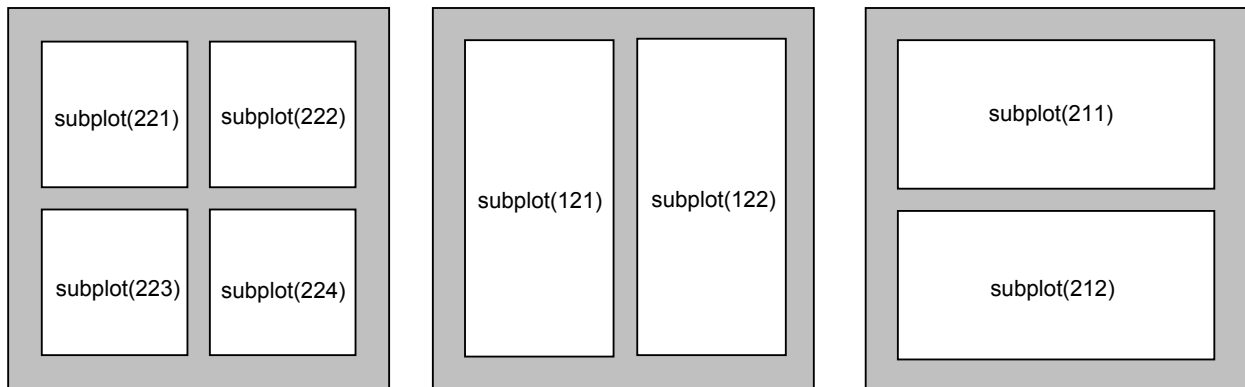
Naredba `axis off` ukida označavanje i obeležavanje po osama. `axis on` vraća ukinuto označavanje za sledeće grafike.

Naredba `v = axis` daje kao rezultat trenutno važeći vektor `[xmin xmax ymin ymax]`.

Naredba `axis equal` izaziva da inkrementi po svim osama budu jednaki, tako da krug izgleda kao krug a ne elisa, a lopta izgleda kao lopta a ne elipsoid.

2.7 Crtanje više grafika na ekranu

U dosadašnjim primerima grafici su prikazivani u celom prozoru. Međutim, u Matlabu postoji mogućnost i da se nacrtaju više različitih grafika u jednom prozoru, za šta se koristi naredba `subplot`, koja se navodi pre naredbe za crtanje. Opšti oblik ove naredbe je `subplot(m,n,p)` ili `subplot(mnp)`, što znači da će se ekran podeliti na m delova po horizontali, n delova po vertikali i tekući grafik će se nacrtati na p -tom delu. Redosled grafika se određuje sa leva na desno, odozgo na dole. Na Slici 10 je ilustrovano na koji način se koristi `subplot` da bi se ostvarile različite podele ekrana.



Slika 10: Korišćenje naredbe `subplot` za podelu ekrana prilikom crtanja grafika.

Ukoliko se ne zada naredba `subplot`, grafik se crta preko celog prozora. Isto tako, podela ekrana zadana poslednjom naredbom `subplot`, ostaje i prilikom crtanja sledećih grafika, sve dok se ne promeni sledećom `subplot` naredbom.

Primer 10.

Izračunati diskretnu funkciju $y = 5e^{(-0.1 + j5\pi/9)n}$ za vrednosti argumenta u opsegu $0 \leq n \leq 20$ i nacrtati grafike njenog realnog i imaginarnog dela u istom prozoru..

Traženo rešenje se dobija naredbama:

```
>> n=0:1:20;
>> y = 5*exp((-0.1+j*5*pi/9)*n);
>> ry = real(y);
>> iy = imag(y);
>> subplot(2,1,1), stem(n,ry);
>> title('Realni deo');
>> subplot(2,1,2), stem(n,iy);
>> title('Imaginarni deo');
```

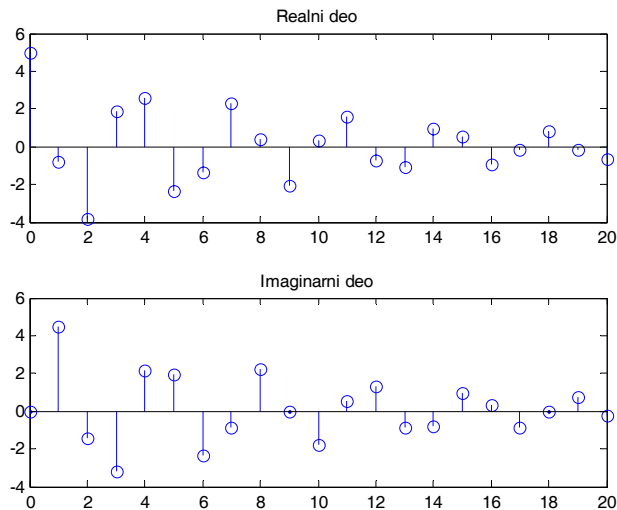
i prikazano je na Slici 11.

2.8 Ostale naredbe za crtanje

Naredba:

```
>> axes('position',RECT)
```

postavlja grafik u prozoru na poziciju određenu vektorom `RECT=[left,bottom,width,height]`. Tačka (0,0) odgovara donjem levom uglu, a tačka (1,1) odgovara gornjem desnom uglu grafika.



Slika 11: Grafički prikaz realnog i imaginarnog dela diskretne funkcije $y = 5e^{(-0.1 + j5\pi/9)n}$.

Naredba:

```
>> grid on
```

crta mrežu isprekidanih linija paralelnu koordinatnim osama. Mreža se briše naredbom:

```
>> grid off
```

Naredba:

```
>> zoom on
```

omogućava uvećanje ili smanjenje razmere crteža pritiskom na levo ili desno dugme miša. Zumiranje se ukida naredbom:

```
>> zoom off
```

Osim dosad opisanih naredbi, koje se najčešće koriste u radu, u Matlabu postoje i druge mogućnosti za prikaz podataka. Neke važnije naredbe za crtanje dvodimenzionalnih i trodimenzionalnih grafika su date u Tabeli 2. Detaljnija uputstva za njihovo korišćenje mogu se dobiti korišćenjem komande `help`.

Tabela 2: Neke od važnijih naredbi za crtanje 2D i 3D grafika.

<code>polar</code>	crtanje u polarnom koordinatnom sistemu
<code>hist</code>	crtanje histograma
<code>bar, stairs</code>	crtanje "stepeničastih" grafika
<code>mesh</code>	crtanje trodimenzionalnih grafika
<code>contour</code>	crtanje konturnih preseka
<code>plot3</code>	crtanje linija i tačaka u trodimenzionalnom prostoru

2.9 Zapisivanje grafika u fajl

Grafik koji se trenutno nalazi na ekranu može da se zapiše u fajl na više načina. Jedan od načina je da se u prozoru slike levim dugmetom miša klikne na opciju "File" a zatim na "Save as" i izabere ime fajla i format fajla u kome će slika biti zapisana. Na raspolaganju je veliki broj raspoloživih standardnih formata, treba izabrati neki format koji komprimuje sliku bez gubitaka, kao što su png i tif formati grafičkih podataka.

Drugi način zapisa je da se slika prvo prebaci u privremenu memoriju (clipboard), a iz nje u neki drugi grafički program ili u program za obradu teksta. Ova operacija se vrši tako što se u prozoru slike levim dugmetom miša klikne na opciju “Edit” a zatim na “Copy Figure”.

Treći način zapisa je korišćenjem naredbe `print` u komandnom prozoru. Jedan od oblika ove naredbe je:

```
>> print <format> <ime fajla>
```

Tako na primer, naredbom:

```
>> print -dmeta slika.met
```

tekući grafik će se zapisati u datoteku `slika.met` kao Windows meta fajl, ili sa opcijom `-dbitmap` tekući grafik će se zapisati u datoteku `slika.bmp` u bitmap formatu. Detaljno uputstvo za korišćenje ove naredbe kao i spisak svih mogućih opcija dobija se sa `help print`. Ovaj način zapisa je pogodan kada je u programu potrebno generisati veći broj fajlova i zapisati ih u odgovarajuće fajlove. Prethodna dva načina zahtevala bi suviše manuelnog rada za zapis slika.

3. Programiranje u Matlabu

U dosadašnjem izlaganju, sve operacije su se kucale u komandnoj liniji (što je označavano sa `>>`) i izvršavale posle pritiska na tipku Enter ili Return. Međutim, prilikom ozbiljnijeg korišćenja Matlabu za rešavanje složenih problema, ovakav način izvršavanja programa postaje nepraktičan i spor. U takvim slučajevima korisnik treba da napiše sopstveni program, ili da kreira sopstvene funkcije, što je takođe podržano u Matlabu.

3.1 Pisanje programa u Matlabu

Kada je potrebno više naredbi objediniti u jednu celinu koja će se često izvršavati, poželjno je napisati program (skriptu) i startovati ga iz komandne linije sa:

```
>> ime_programa
```

Program se piše u editoru koji je deo Matlab programa ili u nekom od editora teksta koji su dostupni u operativnom sistemu koji se koristi (Notepad, UltraEdit, i sl.). Ime datoteke u kojoj se nalazi program može biti proizvoljno, ali ekstenzija mora biti `.m`, dakle, kompletan naziv programa je `ime_programa.m`. Na taj način Matlab “prepoznaje” da se radi o korisničkom programu ili funkciji, zbog čega se obično te datoteke nazivaju još i *m-fajlovi*.

Prilikom pisanja programa poželjno je što češće koristiti komentare. Linija (ili deo linije) u programu, koja predstavlja komentar, počinje znakom `%`.

Postupak pisanja programa, može se jednostavno ilustrovati na sledećem primeru.

Primer 11.

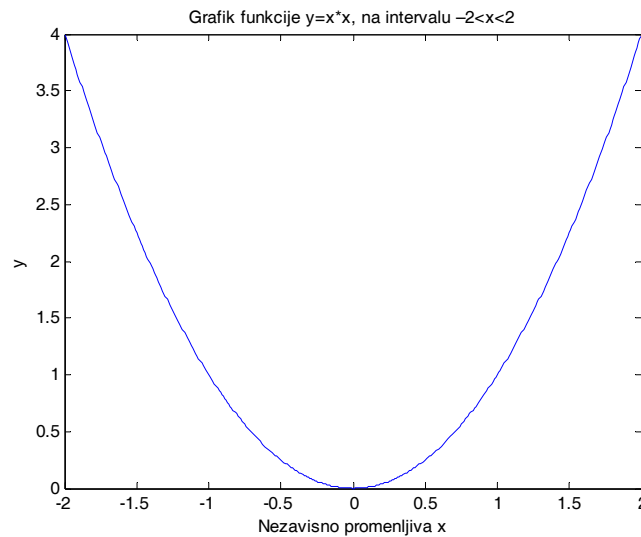
Napisati program kojim će se nacrtati funkcija $y = x^2$ na intervalu -2 do 2 i sačuvati ga u datoteci `crtez.m`.

```
% Program za crtanje funkcije y=x*x na intervalu -2<x<2
x = -2:0.01:2;           % generise se nezavisno promenljiva x
y = x.^2;               % izracunava se zavisno promenljiva y
plot(x,y)               % crta se funkcija
title('Grafik funkcije y=x*x, na intervalu -2<x<2')
xlabel('Nezavisno promenljiva x')
ylabel('y')
```

Iz komandnog prozora Matlabu, program se startuje kucanjem u komandnoj liniji:

```
>> crtez
```

čime se dobija sledeći rezultat:



Slika 12: Rezultat programa crtez.

3.2 Unošenje promenljivih u program sa tastature

Unos vrednosti sa tastature u toku rada programa omogućen je naredbom `input`. Korišćenje ove naredbe najlakše se može objasniti na sledećem primeru.

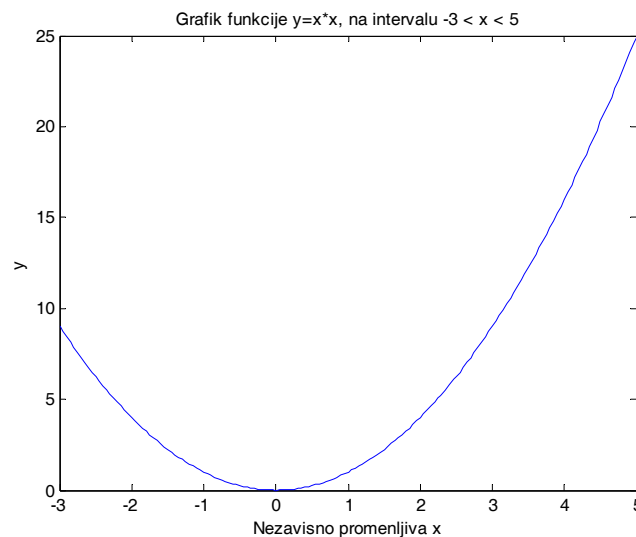
Primer 12.

Napisati program kojim će se nacrtati funkcija $y = x^2$ na intervalu a do b . Vrednosti a , b i koraka između dve susedne vrednosti nezavisne promenljive treba učitati sa tastature, a program sačuvati u datoteci `crtez1.m`.

Traženi program `crtez1.m` izgleda ovako:

```
% Program za crtanje funkcije y=x*x na proizvoljnom intervalu a<x<b
a = input('Pocetak intervala (a): ');      % unosi se a sa tastature
b = input('Kraj intervala (b): ');         % unosi se b sa tastature
korak = input('korak iznosi: ');          % unosi se korak sa tastature
x = a:korak:b;                             % generise se nezavisno promenljiva x
y = x.^2;                                  % izracunava se zavisno promenljiva y
plot(x,y)                                  % crta se funkcija
title(['Grafik funkcije y=x*x, na intervalu ' num2str(a) ' < x < ' num2str(b)]);
xlabel('Nezavisno promenljiva x'); ylabel('y')
```

Funkcija `num2str(a)` pretvara broj a u niz alfanumeričkih simbola koji se jedino mogu koristiti u naredbi `title`.



Slika 13: Rezultat programa `crtez1` za vrednosti parametara $a = -3$ i $b = 5$.

3.3 Funkcije za upravljanje tokom programa

Kao i u drugim programskim jezicima, u Matlabu postoje uobičajene funkcije za upravljanje tokom programa kao što su `for`, `if`, `while`, `return`, i sl. Njihova sintaksa je ista, ili vrlo slična, sintaksi koja se koristi u programskim jezicima C i C++.

3.3.1 FOR petlja

Opšti oblik naredbe `for` je:

```
for promenljiva = a:inkrement:b,
    {   blok naredbi
end
```

Blok naredbi unutar `for` petlje će se izvršavati dok `promenljiva` uzima vrednosti od `a` do `b` sa korakom koji je zadat vrednošću `inkrement`.

Primer 13.

Napisati program koji generiše matricu A , dimenzija $n \times m$, čiji su elementi $A(i, j) = 2^{i+j}$. Dimenzije matrice n i m treba učitati sa tastature.

Rešenje ovog primera je:

```
n = input('Broj vrsta (n) je:');
m = input('Broj kolona (m) je:');
for i=1:n,
    for j=1:m,
        A(i,j)=2^(i+j);
    end
end
```

3.3.2 IF naredba

Naredba `if` je takođe implementirana u Matlabu. Opšti oblik ove naredbe je:

```
if izraz,
    {   blok naredbi
end
```

Blok naredbi unutar `if` naredbe se izvršava ako je `izraz` različit od nule. Prilikom definisanja promenljive `izraz`, kao uslova za izvršavanje, koriste se logičke i relacione operacije definisane u Tabeli 3. Dozvoljeno je i višestruko grananje, kada se koristi kontrolna struktura `if else end`, kao u sledećem primeru.

Primer 14.

Napisati program koji ispituje parnost brojeva. Izlazna promenljiva `par` je jednaka 0 ako je broj negativan; jednaka 1, ako je broj pozitivan i neparan; i jednaka 2, ako je broj pozitivan i paran.

Rešenje ovog primera je:

```
if n < 0
    par = 0;
elseif rem(n,2) == 0
    par = 2;
else
    par = 1;
end
```

3.3.3 WHILE petlja

Opšti oblik naredbe `while` je:


```
while izraz,
    { blok naredbi
end
```

Blok naredbi unutar `while` petlje će se izvršavati dok je `izraz` različit od nule. Prilikom definisanja `izraz-a`, kao dela `while` petlje, mogu da se koriste različite logičke i relacione operacije date u Tabeli 3.

Tabela 3: Relacioni i logički operatori i funkcije.

<	manje
<=	manje ili jednako
>	veće
>=	veće ili jednako
==	jednako
~=	različito
&	logičko i
	logičko ili
~	negacija
find	u nizu (matrici) nalazi indekse koji zadovoljavaju određenu relaciju
isnan	nalazi NaN-ove u zadatom nizu ili matrici
finite	u zadatom nizu (matrici) nalazi elemente koji su Inf
isempty	nalazi prazne matrice (nizove)

Primer 15.

Napisati program koji za zadato k (učitano sa tastature) izračunava sumu: $S = \sum_{n=1}^k 1/n$.

Rešenje ovog primera je:

```
k = input('k iznosi:');
s = 0; n = 1;
while n <= k,
    s = s+1/n;
    n = n+1;
end
```

3.4 Pisanje funkcija u Matlabu

Jedna od pogodnosti Matlab-a je što korisniku pruža mogućnost kreiranja sopstvenih funkcija. One se u daljem radu tretiraju isto kao i funkcije koje su deo samog programa ili pripadaju nekom od Toolbox-ova. Fajlovi koji sadrže korisničke funkcije nazivaju se još i *funkcijski fajlovi* i pišu se na isti način kao i programi (skripte). Jedina razlika između fajla koji sadrži funkciju i fajla koji sadrži program (skriptu) jeste to što *funkcijski fajl u prvoj liniji obavezno mora da sadrži deklaraciju funkcije* koja počinje naredbom `function`. Pisanje korisničkih funkcija se može ilustrovati na sledećem primeru:

Primer 16.

Napisati funkciju `fakt`, koja računa faktorijel zadatog broja.

Oblik tražene funkcije je $y = \text{fakt}(n)$, gde je n ulazni argument (broj čiji se faktorijel računa), a izlaz iz funkcije je izračunata vrednost faktorijela. Program koji realizuje funkciju `fakt` izgleda ovako:

```

function y = fakt(n)
% Funkcija y = fakt(n), racuna faktorijel broja n i dodeljuje ga
% promenljivoj y. Ako je n<0 na ekranu se ispisuje poruka
% o grešci, a promenljivoj y se ne dodeljuje nikakva vrednost.
if n<0,
    disp(' Greska. Faktorijel negativnog broja nije definisan.')
    y=[];                % y je prazan vektor
    return              % izlazak iz programa
end
y = 1; ind = n;
while (ind>0),
    y = y*ind;
    ind = ind-1;
end

```

Nakon što se napiše funkcijski fajl (korišćenjem bilo kog editora), on mora da se sačuva pod imenom `ime_funkcije.m`. Dakle funkcijski fajl *mora* da ima isto ime kao i funkcija koja je u njemu deklarirana. Nakon toga, funkcija može da se upotrebljava kao i sve druge funkcije ugrađene u Matlab. Tako na primer, kucanjem u komandnoj liniji:

```
>> fakt(5)
```

na ekranu se dobija:

```
ans =
    120
```

ili sa:

```
>> z = fakt(3);
```

promenljivoj `z` se dodeljuje vrednost 6, bez ispisa na ekranu. Ukoliko se kao ulazni argument zada broj koji je manji od nule (što ispituje `if` pitalica u programu) na ekranu će se pojaviti sledeća poruka:

```
Greska. Faktorijel negativnog broja nije definisan.
```

što je ostvareno naredbom `disp(' Greska. Faktorijel negativnog broja nije definisan.')`. U opštem slučaju, naredba `disp` na ekranu štampa tekst koji se nalazi između dva znaka `'`. Ova naredba može da se iskoristi i za štampanje vrednosti neke promenljive. U tom slučaju se kao argument navodi samo ime promenljive, npr. `disp(z)`. U slučaju da se u komandnoj liniji Matlab-a otkuca `help fakt`, na ekranu će se dobiti komentar koji sledi posle deklaracije funkcije.

3.5 Efikasnost izvršavanja programa u Matlabu

Matlab pripada grupi interpreterskih programskih jezika. Kada se naredba unosi kucanjem u komandnom prozoru, ona se prvo interpretira a zatim izvršava. Ako se radi o programu, onda se naredbe redom učitavaju, interpretiraju i izvršavaju. Ukoliko u programu postoje petlje, svaka instrukcija u petlji se u svakom prolazu učitava, interpretira i izvršava. U slučaju kada se radi sa velikim količinama podataka, kakav je slučaj u obradi govora ili slike, to može ozbiljno povećati vreme izvršenja programa. Stoga je od interesa razmotriti mehanizme za procenu složenosti programa i načine za popravljjanje efikasnosti programa.

3.5.1 Merenje vremena izvršavanja programa

Merenje vremena izvršavanja programa, ili dela programa, u Matlabu je omogućeno korišćenjem naredbi `tic` i `toc`. One se primenjuju na sledeći način:

```

tic
    { blok naredbi
toc

```

Prvom naredbom `tic` se aktivira merenje vremena (štoperica). Druga naredba `toc` očitava sadržaj brojača bez resetovanja. Resetovanje brojača se vrši novom naredbom `tic`. Naredba `toc` može imati i oblik `t = toc`, čime se omogućava pamćenje prolaznih vremena tokom izvršavanja nekog složenog programa

Primer 17.

Generisati matricu slučajnih brojeva 100×100 , invertovati je i izmeriti vreme izvršenja programa.

Rešenje problema je:

```
tic
a = rand(100);
b = inv(a);
toc
```

Izvršavanjem ovog programa dobija se na ekranu rezultat:

```
Elapsed time is 0.140000 seconds.
```

3.5.2 Povećanje efikasnosti izvršavanja programa

Radi povećanja efikasnosti izvršavanja programa, što je od ogromne važnosti kod realizacije složenih algoritama, potrebno je izbegavati upotrebu programskih petlji što je više moguće. Da bi ilustrovali potrebu za izbegavanjem programskih petlji posmatrajmo generisanje odbiraka diskretne sinusoide pomoću programa:

```
for t=1:5000
    y(t) = sin(2*pi*t/10);
end
```

Merenjem vremena izvršenja ovog programa dobija se vreme od 0.18 s. Ako sada umesto 5000 odbiraka generišemo 20000 odbiraka sinusoide, novo vreme izvršenja će biti 2.1 s, dakle ne četiri nego 11.7 puta duže. Razlog za to je dinamičko povećanje dužine vektora y , za po jedan u svakom prolazu kroz petlju.

Postoji jednostavan način za ubrzanje izvršenja opisanog programa. Ako se pre ulaska u petlju rezerviše prostor u memoriji za ceo vektor (ili matricu), što se nalakše izvodi naredbom za popunjavanje nulama, program se značajno ubrzava, tako da je vreme izvršenja programa:

```
y = zeros(1,20000);
for t=1:20000
    y(t) = sin(2*pi*t/10);
end
```

samo 0.16 s, dakle čak kraće nego za početnu verziju programa sa 5000 odbiraka.

Još brže izvršenje se može postići potpunim izbegavanjem izračunavanja u petlji:

```
y = zeros(1,20000);
t = 1:20000;
y = sin(2*pi*t/10);
```

Ovaj program traje samo 0.04 s.

4. Dodatne funkcije

Osim elementarnih matematičkih funkcija, koje su prikazane u dosadašnjem izlaganju, u Matlabu je implementiran izuzetno veliki broj različitih funkcija: za analizu i obradu podataka, za rad sa polinomima, funkcije koje se koriste u obradi signala, numeričkoj analizi, linearnoj algebri, statistici, itd. Jedan broj jednostavnijih funkcija nalazi se u osnovnom Matlab modulu, dok su ostale grupisane u posebne specijalizovane module (tzv. Toolbox–ove) koji se nezavisno nabavljaju. Tako postoje *Signal Processing Toolbox* (za primenu u digitalnoj obradi signala), *Control System Toolbox* (za primenu u sistemima upravljanja), *Identification Toolbox* (za primenu u identifikaciji

sistema), *Neural Network Toolbox* (za rad sa neuralnim mrežama), *Image Processing Toolbox* (za primenu u digitalnoj obradi slike), *Optimization Toolbox* (za rešavanje različitih linearnih i nelinearnih problema), *Statistics Toolbox* (za primenu u statistici), itd. Broj funkcija koje su implementirane u Matlabu je toliko veliki da čak i iskusni korisnici ne koriste niti znaju sve postojeće funkcije. Međutim, kada se jednom prihvati način rada sa programom i usvoje elementarne operacije, sve složenije komande i funkcije se otkrivaju i usvajaju izuzetno brzo i lako (za šta se često koristi naredba `help`). Za početnike u radu sa Matlabom, u Tabelama 4, 5 i 6 su navedene neke od jednostavnijih funkcija, koje nisu pominjane u ovom tekstu.

Tabela 4: Funkcije za rad sa matricama.

<code>expm, logm i sqrtm</code>	stepenovanje, logaritam i koren matrice
<code>det</code>	determinanta matrice
<code>norm</code>	norma matrice
<code>rank</code>	rang matrice
<code>poly, eig</code>	karakteristični polinom i sopstvene vrednosti matrice
<code>inv</code>	inverzija matrice
<code>orth</code>	ortogonalizacija matrice
<code>pinv</code>	pseudonverzija matrice
<code>qr</code>	ortogonalno–trougona dekompozicija
<code>rref</code>	kanonička forma matrice
<code>svd</code>	dekompozicija preko singularnih vrednosti

Tabela 5: Neke od funkcija za analizu podataka.

<code>max</code>	maksimalna vrednost
<code>min</code>	minimalna vrednost
<code>mean</code>	srednja vrednost
<code>std</code>	standardna devijacija
<code>sum</code>	suma elemenata vektora
<code>diff</code>	aproksimacija diferencijala
<code>corrcoef</code>	koeficijent korelacije
<code>prod</code>	skalarni proizvod
<code>cov</code>	kovarijansna matrica

Tabela 6: Neke od funkcija za rad sa polinomima.

<code>roots</code>	izračunavanje korena polinoma
<code>poly</code>	izračunavanje koeficijenata polinoma iz korenova
<code>polyval</code>	izračunavanje vrednosti polinoma
<code>conv</code>	množenje polinoma
<code>deconv</code>	deljenje polinoma
<code>polyfit</code>	interpolacija polinomom