Matlab - Skripta

1 Uvod

Matlab je programski paket koji predstavlja interaktivno okruženje za razvoj algoritama, vizuelizaciju podataka, analizu podataka i numeričke proračune. Koristeći Matlab, numeričke probleme, za koje je prvenstveno namenjen, možemo rešiti brže nego sa tradicionalnim programskim jezicima kao što su C i C++, jer ne moramo deklarisati promenljive, određivati tipove promenljivih, rezervisati memoriju i sl., a imamo igrađene sve pogodnosti tradicionalnih programskih jezika - kontrola toka, strukture podataka, objektno orijentisano programiranje itd.
U Matlabu komande se izvršavaju bez kompajliranja. Unosimo ih u prozoru Command Window. Osnovni tip podataka sa kojima Matlab radi su matrice, odnosno vektori. Matricu možemo definisati sa naredbom:

```
>> a=[1,2,3; 4 5 6]
```

gde znakovi , i blanck space imaju isto značenje - sledi novi element matrice, a znak ; se interpretira kao znak za novu vrstu matrice. Nakon pritiska tastera Enter dobijamo odgovor

```
 a=
1 2 3
4 5 6
```

Matlab komande interpetira liniju po liniju. Prethodno korišćene naredbe ostaju sačuvane u istoriji i mogu se "vratiti" pritiskom na taster strelica na gore (↑). Ukoliko ne želimo da nam se ispiše rezultat naredbe, potrebno je dodati na kraju naredbe znak ";".

```
>>a=[1,2,3; 4 5 6];
```

Dimenziju matrice (u formi vrsta kolona) dobijamo naredbom

```
 >>size(a)
ans=
2 3
```

Ovde se pojavljuje nova promenljiva `ans`. Ona se automatski generiše kada vrednost nije dodeljena nijednoj promenljivoj (ans skraćeno od answer).
Desno od komandnog prozora, nalazi se prozor Workspace. U njemu možemo videti sve promenljive koje smo inicijalizovali. Inicijalizovane promenljive možemo videti i kucanjem naredbe `who` ili `whos`:

```
 >>whos
 Name  Size  Bytes   Class   Attributes
   a    2x3    48    double
 ans    1x2    16    double
```

Na osovu ispisanog rezultata vidimo da je tip podataka koji Matlab koristi double, dok rezultat ispisuje sa četiri cifre iza pokretnog zareza.

```
>>sqrt(2)
ans=
1.4142
```

Ovo možemo promeniti naredbom

```
>> format long
>>sqrt(2)
ans=
1.414213562373095
```

ili vratiti na staro sa `format short`.

Vektore možemo definisati isto kao i matrice. Osim toga, ako nam je potreban vektor sa jednako udaljenim članovima, za koji znamo prvi i poslednji član i korak između savaka dva člana, tada možemo koristiti naredbu (npr. vektor sa članovima od 1 do 2 i korakom h=0.5):

```
 >> v=1:0.5:2
v =
1.0000 1.5000 2.0000
```

Ukoliko je korak h=1, ne moramo ga unositi već naredba može da ima oblik

`v=1:5`. Ovde treba biti pažljiv ako je h iracionalan broj, jer se zbog zaokruživanja može desiti da ne dobijemo ekvidistantne vrednosti vektora.
Još jedna naredba za definisanje vektora sa ekvidistantnim vrednostima je `linspace(x1,x2,n)`.
Vrednosti x1 i x2 su početak i kraj vektora, a n predstavlja broj tačaka. Ukoliko ne unesemo n, linspace će generisati 100 tačaka.

```
>> w=linspace(pi/2,pi,5)
```

Određeni element vektora dobijamo pomoću malih zagrada.
Napomena: Indeksiranje vektora počinje od 1.

```
>>v(1)
ans=
1
```

Ukoliko želimo da dobijemo poslednji element vektora, umesto v(5), možemo kucati i v(end). Slično je i sa matricama:

```
>>a(1,1)
ans=
1
>> a(:,1)%ispisi prvu kolonu;naredba je ista kao a(1:end,1)
ans=
1
4
```

Sabiranje vektora:

v+v %vektori moraju biti iste duzine ili matrice iste dimenzije
v+2*v
v+2

Operator * je rezervisan za matrično množenje, pa će nam zato naredba
v*w
javiti grešku ukoliko je v/w vektor ili matrica.

v*w' % v' transponovan vektor vektora v

Ukoliko želimo pomnožiti odgovarajuće elemente jednog vektora sa odgovarajućim elementima drugog vektora (množenje v(i)*w(i)) za to u Matlabu imamo operator .* :

v.*w % v(i)*w(i), isto za matrice

Slično je i
v.^ 3 % stepenovanje svakog elementa vektora/matrice
v./w %

U sledećoj tabeli su date neke od naredbi u Matlabu:

Opšte naredbe

clc	obriši tekst iz komandnog prozora
clear all	obriši iz memorije sve inicijalizovane promenljive i funkcije
clear imepf	obriši iz memorije promenljivu ili funkciju imepf
help imepf	ispiši informacije o imepf
Ctrl+C	prekid komande koja se trenutno izvršava

Rad sa matricama (help elmat)

size(A)	dimenzija matrice A
length(v)	dužina vektora v ili broj kolona matrice v
A'	transponovana matrica matrice A
inv(A)	inverzna matrica matrice A
zeros(n,m)	formira nula matricu dimenzije $n \times m$
ones(n,m)	formira matricu popunjenu jedinicama dimenzije $n \times n$
eye(n)	formira jediničnu matricu domenzije $n \times n$

Ugrađene promenljive (help elmat)

ans	promenljiva koja se automatski generiše ako vredn
pi	3.14...
Inf	$+\infty = \frac{1}{0}$
NaN	Not-a-Number rezultat operacije $\frac{0}{0}$ ili $\infty - \infty$
eps	najmanja razlika dva broja koju Matlab može da p

Matlab ima veliki broj ugrađenih funkcija. Neke od njih navodimo u sledećoj tabeli. Svaka funkcija kao argument može da primi matricu.

Ugrađene funkcije (help elfun)

sin(x)	sin(x)
exp(x)	e^x
log(x)	ln(x) prirodni logaritam
log10(x)	logaritam sa bazom 10
abs(x)	apsolutna vrednost
sqrt(x)	kvadratni koren
round(x)	zaokruži x na najbliži ceo broj
ceil(x)	zaokruživanje ka većem celom broju
floor(x)	zaokruživanje ka manjem celom broju
mod(x,y)	ostatak pri deljenju broja x sa y

2 Komandni fajlovi - M fajlovi

Do sada smo unosili naredbe u Matlabu direktno u komandni prozor. Ukoliko želimo da određen skup naredbi izvršavamo više puta, rešenje je da ih smestimo u poseban komandni fajl sa ekstenzijom .m. Ovaj tip komandnog fajla naziva se skript fajl.
Komandne fajlove kreiramo u Matlab-ovom editoru, koji možemo otvoriti iz komandnog prozora kucanjem naredbe `edit` ili preko menija File->New->Script (Ctrl+N). Na primer, kreirajmo skript fajl koji sadrži sledeće naredbe:

```
%test.m
a=1.2;
b=3.7;
n=6;
X=linspace(a,b,n);
```

i sačuvajmo ga pod nazivom `test`.
Skript fajl, iz komandnog prozora, jednostavno pozivamo kucanjem njegovog imena

```
>>test
```

nakon čega će se sve promenljive inicijalizovane u `test.m` učitati u radni prostor, pa možemo dalje manipulisati sa njima.

Drugi tip komandnog fajla je funkcijski m-fajl. Ima istu ekstenziju (.m) kao i skript fajl, a definiše se unošenjem ključne reči `function` u prvoj liniji .m fajla. Funkcijski m-fajlovi su slični funkcijama kod programskih jezika.

Promenljive koje se kreiraju u funkcijskom m-fajlu ostaju unutar fajla i nestaju kada se izvrši funkcija. Kompletna definicija funkcijskog m-fajla glasi:

```
function [y1,y1,...,yn]=imef(x1,x2,...,xm)
```

gde su y1, ..., yn izlazni argumenti, a x1, ..., xm ulazni argumenti funkcije. Ukoliko imamo samo jedan izlazni argument, ne moramo kucati uglaste zagrade. Ime funkcijskog m-fajla i ime funkcije trebaju biti isti, jer funkciju pozivamo imenom fajla.
Komentari trebaju ići odmah iza definicije funkcije. Sve lilnije komantara koje slede odmah iza definicije funkcije čine dokumentaciju te funkcije, koju možemo videti kucanjem naredbe
>> `help imef`
Na primer, definišimo sledeću funkciju:

```
function y=kvadrat(x)
%Fukcija y=kvadrat(x) kvadrira elemente date matrice
y=x.^ 2
```

Iz komandnog prozora je možemo pozvati sa (ako smo je sačuvali sa imenom kvadrat)

```
>> kvadrat(2)
ans
4 % ili
>> kvadrat([2,3])
ans
4 9
```

Definišimo još jednu funkciju:

```
function [y,z]=stepen(x,s)
%stepen(x,s) stepenuje i korenuje matricu x stepenom s
y=x.^ s;
z=x.^ (1/s);
```

Ako pozovemo:
>> `stepen(3,2)`
u promenljivu ans će biti upisana samo vrednost prvog izlaznog argumenta, tj. dobićemo

```
ans
9
```

Ako želimo da dobijemo vrednosti oba izlazna argumenta moramo izlazne

vrednosti dodeliti dvema promenljivim:

```
>> [x,y]=stepen(3,2)
x=
9
y=
1.7321
```

3 Funkcije definisane od strane korisnika

Jedan od načina na koji korisnik može definisati funkciju smo izložili u prethodnoj sekciji. Ukoliko je funkcija jednostavna (sastoji se od jedne linije), umesto definisanja funkcijskog fajla, bolje je definisati inline objekat ili anonimnu funkciju.

3.1 Inline objekat

Inline objekat se definiše pomoću naredbe:

```
f=inline ('fja', 'x1','x2',...)
```

Argument 'fja' je string koji predstavlja telo funkcije. Argumenti koji slede iza su argumenti funkcije koju želimo da definišemo. Njih ne moramo navoditi; Matlab će u opštem slučaju sam izdvojiti znakove koji su argumenti i poređati ih u alfabetskom redosledu. Na primer, možemo definisati:

```
>> f=inline('x.*sin(x)') % ili f=inline('x.*sin(x)', 'x')
```

Ipak, ako imamo više argumenata, i odlučili smo da ih navodimo, onda ih moramo navesti sve! Jer će u suprotnom, na primer

```
>> g=inline('sin(x).*y','x')
g =
Inline function:
g(x) = sin(x).*y
```

argument y ostati nedefinisan.
Vrednost inline objekta (tj. vrednost funkcije u nekoj tački) možemo dobiti direktno kucanjem naredbe:

```
>> f(pi/2)
ans =
```

```
1.5708
```
ili u starijim verzijama Matlaba pomoću naredbe feval (evaluate a function)
```
>> feval(f,pi/2)
ans =
1.5708
```

3.2 Anonimne funkcije

Anonimna funkcja se definiše pomoću naredbe

f=@(x1,x2,...) fja

gde se u malim zagradama, iza znaka @, navode argumenti anonimne funkcije, a nakon toga telo funkcije. Na primer:

```
>>fa=@(x) x.*sin(x)
```

Kod definicije anonimne funkcije mora biti naveden BAR jedan argument. Anonimna funkcija može učitati vrednost promenljive koja je prethodno definisana, npr. naredbe:
```
>> k=3;
>> fa=@(x) x.*sin(x*k)
```
definišu anonimnu funkciju fa(x)=xsin(3x). Ako promenimo kasnije vrednost promenljive k, u anonimnoj funkciji fa(x) se neće ništa promeniti!

Isto kao kod inline objekta, vrednost anonimne funkcije možemo dobiti direktnim pozivom `fa(pi/2)` ili pomoću `feval(fa,pi/2)`.

4 Funkcije kao argumenti funkcija- funkcija quad

Jedna od mnogih funkcija koja kao argumente prima drugu funkciju je

quad(f,a,b)

Funkcija (naredba) quad izračunava vrednost određenog integrala funkcije f na segmntu (a,b), tj. $\int_a^b f(x)dx$. Za izračunavanje određenog integrala se koristi numerička metoda - kvadraturno pravilo odakle potiče naziv naredbe.

Prvi argument funkcije, f, je pokazivač na funkciju jedne promenljive, a druga dva početak i kraj intervala integracije. Tako, na primer, možemo izračunati:

```
>> quad(f,0,1)% gde je f gore definisan inline objekat
ans=
0.3012
```

ili

```
>> quad(fa,0,1)% gde je fa gore definisana anonimna funkcija
ans=
0.3457
```

Ukoliko nam je potrebna vrednost određenog integral funkcije koja je data funkcijskim m-fajlom, moramo kreirati pokazivač na funkcijski fajl-dodavanjem znaka @ ispred naziva fajla:

```
>> quad(@kvadrat,0,1)
0.3333
```

4.1 Integral složene funkcije

Neka su definisane sledeće funkcije (mogu i inline objekti):

```
 >> f=@(x) x.^ 2;
>> g=@(x) sin(x);
```

Složenu funkciju h=f(g(x)) možemo definisati kao inline objekat:

```
>> hi=inline('f(g(x))','x','f','g')
```

gde dobijamo funkciju sa tri argumenta (inline ne može učitati u definiciju funkcije vrednosti prethodno definisanih promenljivih!). Pokazivač na funkciju sa tri argumenta ne možemo proslediti quad-u, već je moramo predefinisati tako da dobijemo pokazivač na funkciju sa jednim argumentom, tj.:

```
>>quad(@(x) hi(x,f,g), 0,1)% sada će se učitati vrednosti od f i
g
ans =
0.2727
```

Ako složenu funkciju definišemo kao anonimnu funkciju, stvar je mnogo jednostavnija

```
>> ha=@(x) f(g(x))
```

gde će se učitati vrednosti promenljivih f i g, pa dobijamo pokazivač na funkciju jedne promenljive. Naredba za određeni integral je

```
>>quad(ha, 0,1)
% ili možemo direktno quad(@(x) f(g(x)), 0,1)
```

5 Kontrola toka

Matlab ima četiri naredbe za kontrolu toka: for petlja, while petlja, if-else grananje i switch grananje. Svaka od njih se završava sa ključnom reči end. Navodimo samo opšte oblike naredbi:

```
 for brojac=a:b
narebda
naredba
end

if uslov1
naredba
elseif uslov2
naredba
else
naredba
end

while uslov
naredba
end

switch promenljiva
case vrednost1
naredba
case vrednost2
naredba
otherwise
naredba
end
```

Logički operatori :
~ negacija

`&&` logičko i
`||` logičko ili

6 Rad sa polinomima

U Matlabu se polinomi predstavljaju vektorom (vrstom) svojih koeficijenata, počevši od koeficijenta uz najviši stepen. Dakle, ako je polinom n-tog stepene, tada vektor koeficijenata sadrži n+1 element. Na primer polinom x^2-1 se u Matlabu predstavlja vektorom p=[1, 0, -1].
Funkcije za rad sa polinomima možemo videti kucanjem naredbe `help polyfun`. Izlistaće se nekoliko grupa funkcija. Nas interesuje neke funkcije koje se nalaze u grupi Polynomials:
`roots (p)` - vraća korene polinoma p
`poly(v)` - vraća koeficijente polinoma čije su nule elementi vektora v
`polyval(p,a)` - vrednost polinoma p u tački (vektoru) a
`polyder(p)` - Izvod polinoma p
`polyint(p)` - integral polinoma p
`conv(p,q)` - proizvod polinoma p i q
`deconv(p,q)` - količnik polinoma p i q