D Függelék

Bevezetés a MATLAB használatába

A MATLAB egy numerikus programkönyvtár, amely elsősorban mátrixműveletek hatékony alkalmazásŕa készült (innen a neve is). Mindazok, akiknek van tapasztalata magasszintű programozási nyelvekkel, és dolgoztak már ciklusokkal, feltételes utasításokkal, szubrutinhívással, és logikai relációkkal, azok ezt a tudást közvetlenül használhatják a MATLAB alkalmazása során.

A programcsomag számos numerikus eljárást tartalmaz, könnyen használható két- és háromdimenziós megjelenítést, és magas szintű programozhatóságot. A MATLAB elsősorban azért alkalmas a közelítő számítások oktatására, mert könnyen lehet a segítségével ilyen programokat írni és módosítani.

A következő rövid bevezetés a MATLAB használatába inkább csak támogatást nyújt, de a programcsomag teljes körű megismeréséhez valamely felhasználói kézikönyvet érdemes elolvasni (pl. [5]).

A MATLAB programot a Linux és a Windows operációs rendszerekben a szokásos módon, a megfelelő ikonra való dupla kattintással lehet elindítani. Az ezután kapott párbeszédes ablakban a felhasználó által begépelt utasításokat a >> prompt után lehet megadni. A programból való kilépéshez egyszerűen írjuk be a quit vagy az exit utasítást a prompt után.

A tárgyalandó további nagyobb témák:

- Programozás m-fájlokkal: szkriptek
- Adattípusok (osztályok) a MATLAB-ban
- Hogyan lehet hatékony programokat írni MATLAB-ban?
- Adatállományok olvasása és írása
- Ritka mátrixok kezelése
- A MATLAB súgó rendszere
- Polinomok a MATLAB-ban

Aritmetikai műveletek és függvények

Itt és a továbbiakban is a MATLAB utasításokat typewriter betűtípussal írjuk. Az alapvető műveletek írásmódja nem nagyon meglepő:

+	összeadás
-	kivonás
*	szorzás
/	osztás
^	hatványozás
i, pi	a megfelelő konstansok
NaN	Not-a-Number, nem ábrázolható szám
Inf	végtelen

A programozási nyelvekben megszokott standard függvények itt is elérhetők, és nevük is közel van a szokásoshoz, pl.:

abs(#) cos(#) exp(#) log(#) log10(#) cosh(#) sin(#) tan(#) sqrt(#) floor(#) acos(#) tanh(#)

A # persze az adott függvény argumentumát jelöli, és további információt más függvényekről a MATLAB online súgója ad.

PÉLDA. Tekintsük a következő egyszerű képletet, amely a π egy közelítésének pontos decimális jegyei számát adja meg:

$$-\log_{10}\left(\frac{3.141626-\pi}{\pi}\right).$$

Ennek MATLAB kódja, amit tehát a >> jelű prompt után kell begépelni:

>> -log10 ((3.141626 - pi) / pi)

Az ENTER megnyomása után a következő választ (answer) kapjuk:

```
ans = 4.9741
```

Ennek eléréséhez tehát nem kellett semmit se írni a sorvégére. Ha visszaírt válasz nélkül kérjük, akkor pontosvesszőt kell írni a sor végére, mint hasonló rendszerekben.

Alapértelmezésben tehát 5 jegyet kapunk. Ha pontosabb megjelenítésre van szükség, akkor a format long utasítás kb. 15 decimális jegyet eredményez:

A felhasználók saját függvényeiket ún. M-file-okban adhatják meg. Ezek az adatállományok a MATLAB saját formátumát követik, .m kiterjesztésűek, és a MATLAB támogatja a szerkesztésüket. A fentiek szerint definiált új függvényeket ugyanúgy lehet a MATLAB-on belül használni, mint a rendszer saját függvényeit. Ha elsőre nem találja a frissen írt .m állományt, akkor ellenőrizzük a Set Path utasítást a File menüsorban, és ha kell, adjuk az új könyvtárat az eddigiekhez. PÉLDA. Definiáljuk a fun $(x) = 1 + x - x^2/4$ függvényt a MATLAB Editor/Debugger ablakában, és mentsük el a fun.m állományba. Ehhez a következő formátumot kell követni:

```
function y=fun(x)
y=1+x-x.^{2}/4;
```

A ". " használatát hamarosan megmagyarázzuk. A változók és a függvények nevében használhatunk kis és nagybetűket, a lényeg az, hogy a hivatkozások során következetesen járjunk el. Ezután a MATLAB parancsablakában (Command Window) a következő módon használhatjuk a definiált függvényt:

A függvény kiértékelésére használhatjuk a feval utasítást is:

```
>> feval('fun',4)
ans=
1
```

Vegyük észre, hogy ebben az esetben a függvény nevét karaktersorozatként kell megadni.

Műveletek mátrixokkal

A mátrixok kezelése érthető módon az erőssége a MATLAB-nak. Lényegében minden változót mátrixként kezel:

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

Ahogy látható, a mátrixok definiálásában a sorokat pontosvesszővel választjuk el, az egyes mátrixelemeket pedig szóközzel. A mátrixokat soronként begépelve is be lehet vinni:

A mátrixokat beépített függvények segítségével is generálhatjuk:

```
>>Z=zeros(3,5); egy 3-szor 5-ös, csupa nullából álló mátrixot ad,
>>X=ones(3,5); egy 3-szor 5-ös, csupa egyesből álló mátrixot kapunk,
>>Y=0:0.5:2 egy 1-szer 5-ös mátrixot generál:
Y=
0 0.5000 1.0000 1.5000 2.0000
```

A definiált mátrixokon elemenként függvényeket lehet végrehajtani:

>>cos(Y)		egy megfelelő 1×5 -ös mátrixot ac			mátrixot ad:
ans=					
	1.0000	0.8776	0.5403	0.0707	-0.4161

A mátrixok komponenseit ügyesen lehet kezelni a MATLAB-ban, tekintsük például a következő utasításokat:

>>A(2,3)	az A mátrix egy elemét választja ki,
ans=	
6	
A(1:2,2:3)	az A mátrix egy részmátrixát adja,
ans=	
2 3	
56	
A([1 3],[1 3])	az A mátrix egy részmátrixa kiválasztásának egy másik módja:
ans=	
1 3	
79	
>>A(1,1)=sin(3.14);	értékadás egy mátrixelemnek.

A mátrixokra a következő műveleteket alkalmazhatjuk:

+	összeadás,
-	kivonás,
*	szorzás,
^	hatványozás, és
,	konjugált transzponálás.

PÉLDA. A mátrixműveletek illusztrálásaként tekintsük a következő egyszerű utasítássort:

>>B=[1 2;3 4]; >>C=B' C= 1 3 2 4 >>3*(B*C)^3 ans= 13080 29568 29568 66840

Itt tehát C a B transzponáltja, és az utolsó mátrix a $3(B * C)^3$.

A felsoroltakon túl természetesen számos egyéb utasítás érhető el mátrixok manipulálására. Ezekkel kapcsolatban érdemes az online súgót, a felhasználói kézikönyvet, vagy más leírást (pl. [5]) tanulmányozni.

A MATLAB erőssége azon függvények széles köre, amelyek mátrixok elemein hajthatók végre. Korábban erre láttunk példát, amikor egy 1 × 5-ös mátrix elemeinek koszinuszát határoztuk meg. A mátrixokra vonatkozó összeadás, kivonás és skaláris szorzás természetesen mátrix elemenként történik, de a szorzás, osztás és a hatványozás már nem. Ahhoz, hogy ezeket a műveleteket a mátrixelemeken hajthassuk végre, a műveleti jelek elé egy pontot kell írni: .*, ./ és .^. A mátrixokra és azok elemeire vonatkozó műveleteket nem szabad összekeverni:

>>A=[1 2;3 4]	;
>>A^2	ez az AA mátrixszorzatot adja:
ans=	
7 10	
15 22	
>>A.^2	ezzel az A mátrix elemei négyzetét kapjuk:
ans=	
1 4	
9 16	
>>cos(A./2)	ezzel pedig az A mátrix elemei felének koszinuszát határozzuk meg:
ans=	
0.8776	0.5403
0.0707	-0.4161

Megjelenítés

A MATLAB görbék és felületek két-, vagy háromdimenziós ábráit tudja megjeleníteni. Az itt röviden bemutatott lehetőségeken túliakat az online súgó, szakkönyv ([5]), vagy a felhasználói leírás segítségével kereshetjük meg.

A kétdimenziós görbék megjelenítésére a plot utasítást lehet használni. A következő példa az $y = \cos(x)$ és az $y = \cos^2(x)$ függvényeket ábrázolja a $[0, \pi]$ intervallumon:

```
>>x=0:0.1:pi;
>>y=cos(x);
>>z=cos(x).^2;
```

>>plot(x,y,x,z,'o')

Az első sor adja meg a megjelenítési tartományt, 0.1 lépésközzel. A következő kettő definiálja a két függvényt. Vegyük észre, hogy az első három sor mindegyike pontosvesszővel végződik. Ez megakadályozza, hogy az ezekben definiált mátrixok megjelenjenek a párbeszédes ablakban. A negyedik sor tartalmazza a plot utasítást, és jeleníti meg a grafikont. Az első két argumentum eredményezi az $y = \cos(x)$ függvény ábrázolását, az utolsó három pedig a $z = \cos^2(x)$ megjelenítését, éspedig úgy, hogy az egyes (x_k, z_k) pontokat 'o' jelöli.

A plot utasításnak egy hasznos alternatívája az fplot. Általános alakja a következő:

```
fplot('name',[a,b],n).
```

Ennek hatására a program veszi a name.m adatállományból a függvényt, és az [a, b] intervallumból vett n darab alappontban meghatározott érték alapján elkészül az ábra. Az n alapértelmezése 25.

>>fplot('tanh', [-2,2]) a tanh(x) függvényt jeleníti meg a [-2,2] intervallumon.

A plot és a plot 3 utasítások alkalmasak paraméteres függvények két-, illetve háromdimenziós ábrázolására. Ezek különösen differenciálegyenletek megoldásának megjelenítésére alkalmasak. Például a $c(t) = (2\cos(t), 3\sin(t))$ ellipszist a $0 \le t \le 2\pi$ tartományon a következő utasítással lehet ábrázolni:

>>t=0:0.2:2*pi;
>>plot(2*cos(t),3*sin(t))

A $c(t) = (2\cos(t), t^2, 1/t)$ 3-dimenziós görbe képét a $0.1 \le t \le 4\pi$ paraméter-tartományon pedig a következő utasítással lehet ábrázolni:

>>t=0.1:0.1:4*pi;
>>plot3(2*cos(t),t.^2,1./t)

A háromdimenziós felületek ábrázolásához egy téglatestet kell megadni a felület értelmezési tartományán, amelyet a meshgrid paranccsal lehet definiálni. Ezután a mesh vagy a surf parancsokkal kapjuk az ábrát, mint a következő példában is:

```
>>x=-pi:0.1:pi;
>>y=x;
>>[x,y]=meshgrid(x,y);
>>z=sin(cos(x+y));
>>mesh(z)
```

(Vegyük észre, hogy az utolsó sorban egyik példában sincs pontosvessző a sor végén.)

Programok, ciklusok, vezérlés

A logikai és relációs jelek, műveletek a MATLAB-ban is hasonlók a magasszintű programozási nyelvekben megszokottakhoz:

Relációjelek:

==	egyenlő
~=	nem egyenlő
<	kisebb
>	nagyobb
<=	kisebb vagy egyenlő
>=	nagyobb vagy egyenlő

Logikai műveletek és konstansok:

~	negáció
&	és
	vagy
1	igaz
0	hamis

A for, if és while utasítások a MATLAB-ban is úgy működnek, mint a hasonló programozási nyelvekben. Ezeknek az alapvető formája:

```
for (ciklusváltozó = kifejezés)
    utasítások
end
if (logikai kifejezés)
    utasítások
else
    utasítások
end
while (kifejezés)
    utasítások
end
```

A következő példa azt mutatja, hogyan lehet egymásbaágyazott ciklusokkal egy mátrixot generálni. Ha a példaprogramot nest.m néven mentjük el, akkor a MATLAB promptjához nest-et írva az A mátrixot eredményezi. Vegyük észre, hogy a mátrix bal felső sarkából kiindulva a Pascal háromszöget kapjuk.

```
for i=1:5
    A(i,1)=1;A(1,i)=1;
end
for i=2:5
    for j=2:5
        A(i,j)=A(i,j-1)+A(i-1,j);
        end
end
A
```

A break parancs hatására befejeződik egy ciklus:

```
for k=1:100
    x=sqrt(k);
    if ((k>10)&(x-floor(x)==0))
        break
    end
end
k
```

A disp utasítás szöveg, vagy egy mátrix kiíratására használható:

```
n=10;
k=0;
while k<=n
    x=k/3;
    disp([x x<sup>2</sup> x<sup>3</sup>])
    k=k+1;
end
```

A MATLAB programírás hatékony módja a felhasználó által definiált függvények összeállítása. Ezek input és output paramétereket is használhatnak, és más programokból szubrutinként hívhatók. Ennek bemutatására tekintsük a következő egyszerű programot, amit a MATLAB File / New menüpontban elérhető Editor / Debugger segítségével szerkesszünk meg, és mentsük el pasc.mnéven.

```
function P=pasc(n,m)
%Input - n a sorok száma
% - m a prímszám
%Output - P a Pascal háromszög
for j=1:n
    P(j,1)=1;P(1,j)=1;
end
for k=2:n
    for j=2:n
        P(k,j)=rem(P(k,j-1),m)+rem(P(k-1,j),m);
```

end end

Ezután a MATLAB parancssorába írjuk be azt, hogy P=pasc(5,3), és láthatjuk a mod 3 Pascal háromszög első 5 sorát. A másik érdekes teszt P=pasc(175,3); (most fontos a pontosvessző), és aztán gépeljük be azt, hogy spy(P), ami ritka mátrixot generál nagy n esetén.

PÉLDA. Az alábbi rövid Matlab program megjeleníti az $y = (1 - x)^6$ függvényt, és ennek Hornerelrendezés szerint átrendezett, de ekvivalens alakját, ahol

$$z = ((((((x-6) * x + 15) * x - 20) * x + 15) * x - 6) * x + 1).$$

>> x = (9950:10050)/10000; % definialja a pontsorozatot
>> y = (1-x).^6;
>> z = ((((((x-6).*x+15).*x-20).*x+15).*x-6).*x+1);
>> plot(x,[y;z]); % egy grafikont jelenit meg
>> print -deps hornerdemo.ps % kiirja egy fajlba

A kapott ábra a két nagyon eltérő görbével (a sima az $(1 - x^6)$):



Néhány olyan Matlab utasítás, amely az adott számítógép, illetve a szoftver környezet megismerését szolgálja: az eps a gépi pontosság aktuális értékét adja,

a computer azonosítja a használt számítógép típusát,

a realmax a legnagyobb pozitív gépi szám,

a realmin a legkisebb pozitív gépi szám.

Végül, meglepetésként gépeljük be spy utasítást, és a sorvégi pontosvessző nélkül nyomjuk meg az Enter gombot.