Az .m kiterjesztésű állományok

Ha sok számítást kell egymás után elvégeznünk, és nem vagyunk kíváncsiak a részeredményekre, akkor nem szükséges a parancsablakban minden egyes utasítást egymás után begépelni, hanem érdemes ún. m-fájlt készíteni. Ezek szöveges állományok, amelyek Matlab utasításokat tartalmaznak. Adott állomány nevét (kiterjesztés nélkül) a parancsablakban utasításként kiadva a Matlab a megfelelő szöveges állomány (amelynek kiterjesztése .m kell, hogy legyen - innen az m-fájl elnevezés) utasításait egymás után értelmezi és végrehajtja. Az m-fájlokban lehetőség van ciklusok szervezésére, utasítások feltételes végrehajtására és egyéb programozási trükkök bevetésére is, lásd a következő fejezetet. Az m-fájlokon belül is igaz az, hogy ha egy utasítás után nem írunk pontosvesszőt, akkor a Matlab a parancsablakban megjeleníti az adott utasítás eredményét.

Az .m kiterjesztésű állományokat bármely, szöveges (txt) fájl létrehozását támogató szövegszerkesztő (pl. Notepad) segítségével létrehozhatjuk, de használhatjuk a Matlab beépített szerkesztőjét is. (Ez utóbbi a Notepad némileg módosított változata.) A Matlab beépített szerkesztőjéhez hozzaférhetünk a parancsablak File menüjében. Új m-fájl létrehozásához válasszuk a New almenü M-file opcióját, már létező m-fájl megnyitásához pedig az Open M-file parancsot. Nem mindegy, hogy hova mentjük el m-állományainkat. A Matlab csak azokat az állományokat képes megtalálni, amelyek vagy az éppen aktuális munkakönyvtárban, vagy a matlabpath változóban felsorolt könyvtárak egyikében van. A parancsablakban a `cd` paranccsal változtathatjuk meg az aktuális könyvtárat. Mindig abba a könyvtárba lépjünk bele, amelyikben a futtatandó m-fájl található.) Az aktuális könyvtár tartalmának kilistázására a `dir` parancs szolgál. FIGYELEM!!! A Matlab 4.2-es verziója nem kezeli a hosszú fájlneveket. Ezek használatát kerüljük. Gyakorlásképpen hozzunk létre egy saját alkönyvtárat a Matlab-fájlok tárolására. Ezután a szövegszerkesztőben készítsünk egy új fájlt a következő tartalommal:

```
A=[1 3;6 4]
B=[3 5;7 1]
szorzat=A*B
```

Mentsük el ezt pl. `matrixszorzat.m` néven az előbb létrehozott alkönyvtárba. A parancsablakban lépjünk be ebbe az alkönyvtárba a `cd` parancs segítségével, majd írjuk be a fájl nevét:

```
» matrixszorzat
```

Láthatjuk, hogy a Matlab egymás után végrehajtja a `matrixszorzat.m` fájlban felsorolt utasításokat. Ha A és B megadása után pontosvesszőt is írunk, akkor a parancsablakban csak a végeredményt írja ki a Matlab.

Az előbbi m-fájl példa úgynevezett scriptre. A script olyan m-fájl, amely utasítások sorozatából áll. A script lefuttatásakor a benne definiált változók a munkaterülethez adódnak (mintha csak a parancsablakban írtuk volna be ezeket az utasításokat egymás után). Ennek nemkívánatos mellékhatásai lehetnek: i) felülírhatunk már használatban lévő változókat, ii) a script végrehajtása során galibát okozhat, ha a benne használt változónevek korábban már értéket kaptak. Mindezek elkerülhetők, ha script helyett függvényt használunk. Az ii) probléma nem lép fel, ha a scriptet a `clear all` sorral kezdjük.

Matlab-függvények

A függvény olyan m-fájl, amelynek bemenő adata(i) ill. kimenő adata(i) is lehet(nek), és a Matlab külön munkaterületen tárolja a benne létrehozott változókat.

A Matlab-függvények felépítését a következő példán tanulmányozhatjuk. Írjuk be a következő sorokat a szövegszerkesztőbe, majd mentsük el `faktorialis.m` néven

```
function f = faktorialis(n)
f = prod(1:n);
```

A Matlab-függvény első sora mindig a `function` szóval kezdődik. Utána a kimenő változó (ill. több változó esetén azok vektora szögletes zárójelben) szerepel, majd egyenlőségjel után a függvény neve, és utána kerek zárójelben a bemenő változó vagy változók. (A Matlab-függvényt minden esetben a saját nevén mentsük el, tehát az első sorban az egyenlőségjel után álló szó legyen magának az m-fájlnak is a neve.) A függvény neve csak betűvel kezdődhet, és nem lehet hosszabb 63 karakternél!) A példában a `faktorialis` nevű függvénynek egy bemenő adata van (egy egész szám, amelynek a faktoriálisára kíváncsiak vagyunk), és egy kimenő adata (a faktoriális értéke).

Lássuk, hogyan használhatjuk ezt a függényt egy egész szám faktoriálisának, pl. 5!-nak a kiszámítására. A parancsablakba írjuk be:

```
» f = faktorialis(5)
```

Az eredmény:

```
f =
```

```
120
```

Az történt, hogy az 5-ös szám, amelyet a függvény hívásakor megadtunk a függvény neve után zárójelben, bekerült a függvény lokális n változójába, ezen végrehajtódott a függvény kódjában megadott utasítás, és az 5 faktoriálisa bekerült a kimenő változóba. (A `prod` beépített függvény, ezt használtuk most az első n egész szám összeszorzására.) Megjegyezzük, hogy a függvény hívásakor nem szükséges a kimenő adatot ugyanúgy nevezni, amilyen néven a függvényben szerepel. Lehetett volna f helyett más betűt is használni, pl.

```
» z = faktorialis(5)
z =
120
```

Olyan változónevet használjunk, ahogyan az eredményt hívni szeretnénk. Úgy is meghívható a függvény, hogy nem adjuk meg a kimenő változó nevét:

```
» faktorialis(5)
```

Ekkor az eredmény az `ans` nevű változóba kerül.
A függvényt kiegészíthetjük további sorokkal, hogy megkönnyítsük a használatát mások és saját magunk számára:

```
function f = faktorialis(n)
% FAKTORIALIS(N) az N szam faktorialisat adja vissza
f = prod(1:n);
```

A második sorban a % jel utáni szöveg kiíródik a képernyőre, ha a parancsablakba beírjuk a `help faktorialis` utasítást. További % kezdetű sorok is lehetnek (bárhol az m-fájlban beszúrhatók ilyenek), amelyek részletesebb magyarázatot tartalmaznak. Ilyen sorok nemcsak függvényben, hanem scriptben is készíthetők.

Nézzünk példát most többváltozós függvényre is. Például egy A és egy B mátrix kommutátora definíció szerint az $AB - BA$ kifejezés. Ha elkészítjük a következő m-fájlt komm.m néven:

```
function X = komm(A,B)
X = A*B-B*A;
```

akkor egy adott E és F mátrix kommutátorára a `komm(E,F)` karaktersorral kérdezhetünk rá. Egy függvénynek lehet több kimenő adata is. Pl. készítsünk olyan függvényt `norma.m` néven amely megadja egy mátrix 1-es, 2-es és végtelen-normáját!

```
function [x y z] = norma(A)
x = norm(A,1)
y = norm(A)
z = norm(A,inf)
```

Próbáljuk ki pl. az `E = rand(5)` mátrixon! Ehhez a parancsablakba írjuk a következőket:

```
»E = rand(5)
»[n1 n2 nmax] = norma(E)
```

Ezzel az E mátrix 1-es, 2-es ill. végtelen-normája bekerült az n1, n2 és nmax nevű változókba. Ha a függvényt a `norma(E)` paranccsal hívjuk meg, akkor mindössze a legelső kimenő adatot, vagyis az 1-es normát adja vissza, ill. `[n1 n2]=norma(E)` begépelésére az első két kimenő változót, tehát az 1-es és a 2-es normát.