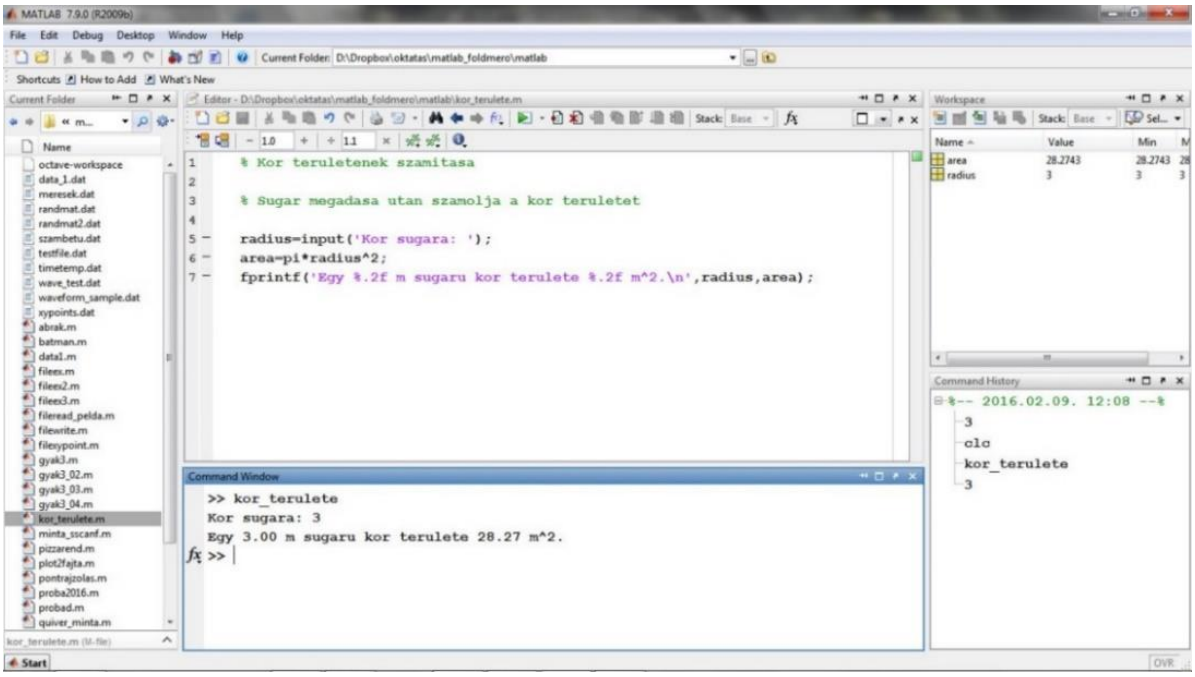
1. MATLAB/OCTAVE ALAPOZÓ[1]

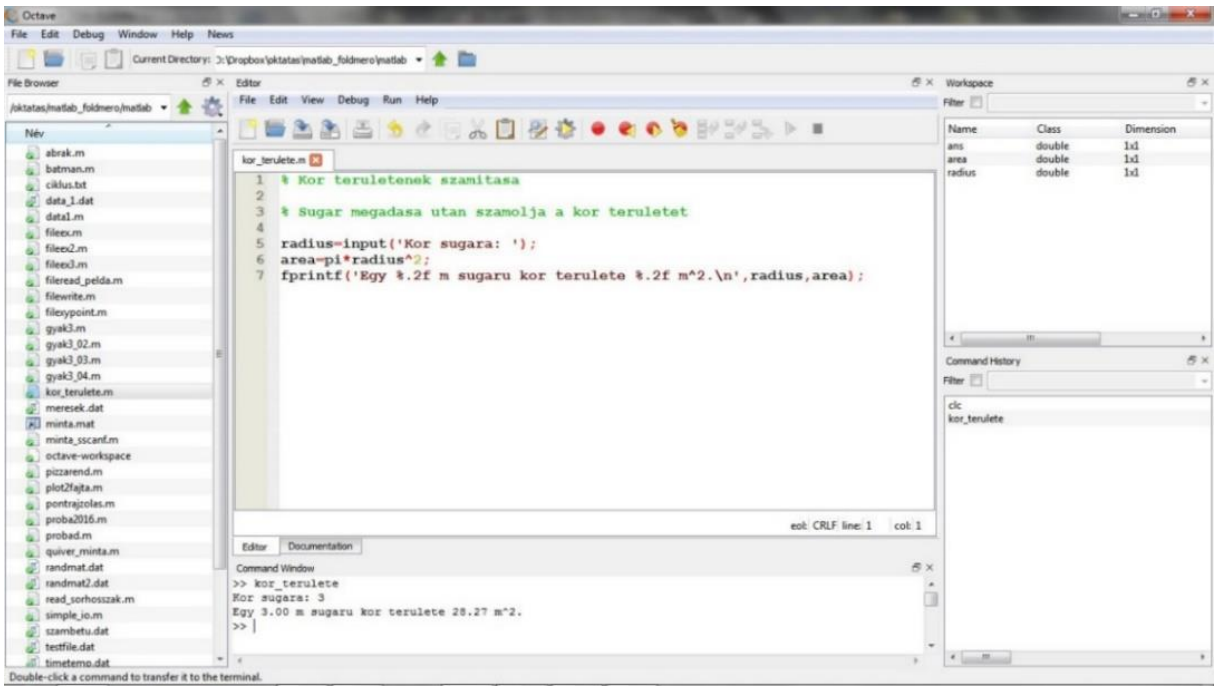
A numerikus módszerek gyakorlatok során Matlab matematikai környezetben mutatjuk meg a mérnöki problémák megoldásához használható különböző numerikus eljárásokat. Otthoni gyakorláshoz használható, 2017 márciusától ingyenesen, a Matlab szoftver a BME egyetemi licenccel (http://www.bme.hu/hirek/20170330/Matematikai_es_muszaki_megoldasok_egyszerubben_es_gyorsabban). Jó alternatív megoldás lehet az Octave matematikai környezet is, ami egy ingyenes, nyílt forráskódú program, ahol lényegében ugyanazokat a parancsokat használhatjuk, mivel az Octave készítői törekednek a Matlab kompatibilitásra. Még a grafikus felület is nagyon hasonló a két programban, tetszés szerint átrendezhető ablakokkal (lásd 1., 2. ábra). Az Octave a https://www.gnu.org/software/octave/ oldalról tölthető le, jelenleg a 5.2.0 változat, ami 2020. január 31-én jött ki.

A Matlab oldalán létre lehet hozni egy **MathWorks account**-ot (célszerű BME-s email címet használni, ha az Online Matlab-hoz is szeretnénk hozzáférni). A MathWorks account-tal lehetőség van megoldani a **Matlab Onramp**-ben található alap gyakorló feladatokat, ezt mindenkinek ajánlott végigcsinálni (kb. 2-3 óra). További jó, online gyakorlási lehetőségek vannak a **Matlab Cody** oldalán.



1MATLAB GRAFIKUS KÖRNYEZETE

[1] A segédlet készítése során felhasználva: Todd Young and Martin J. Mohlenkamp: Introduction to Numerical Methods and Matlab Programming for Engineers, Department of Mathematics, Ohio University, July 24, 2018, http://www.ohiouniversityfaculty.com/youngt/IntNumMeth/book.pdf



2OCTAVE GARFIKUS KÖRNYEZETE

MATLAB MUNKAKÖRNYEZET, ALAPOK

A grafikus felület fontosabb részei: az éppen aktuális könyvtár, ahová mindent ment a program (**current folder**), a parancssor (**command window**), a munkakörnyezet (**workspace**) az éppen használt változókkal, az eddig futtatott parancsok (**command history**) és a szerkesztő (**editor**). Az adott panel nevére kattintva, lenyomva tartott bal egér gombbal áthúzhatóak a panelek, és a mozgatás során a kékre színezett területre helyezhetőek. Célszerű lehet rögzíteni a szerkesztőt (dock editor), ezt az Editor ablak jobb felső részén lévő nyílra kattintva tehetjük meg.

A későbbiekben amire mindig figyeljünk, hogy a Matlabban használt fájl nevek nem kezdődhetnek számmal és ne legyenek benne ékezetes karakterek, szóközök se! A program mindig azokat a fájlokat, függvényeket tudja éppen használni, amik a beállított aktuális könyvtárban vannak.

SEGÍTSÉG (HELP, DOCUMENTATION)

Nagyon sokat tanulhatunk a dokumentáció helyes használatából is, persze kellő angoltudás függvényében.

Ha csak egyszerűen begépeljük a **help** parancsot, akkor a Matlab kilistázza a különböző beépített függvények kategóriáit. Itt vagy duplán rákattintunk a kategória nevére, vagy begépelhetjük pl. (> jel jelzi a Matlab-ba begépelendő parancsokat, ezt a jelet nem kell begépelni!)

```
> help elfun
```

Ami ki fogja írni az ’Elementary math functions’-t, vagyis az alap matematikai függvényeket, pl trigonometriai, exponenciális, komplex és kerekítő függvények listája.

Ha tudjuk egy adott függvény nevét, akkor használhatjuk következőt:

```
> help 'parancsnév'
```

Ez megadja az adott parancs leírását, használatának módját. Pl.:

```
> help rand
```

Megadja, hogy a **rand** parancs 0-1 közötti egyenletes eloszlású véletlen számot generál, megadja a használatának módját, hogy lehet egy vagy több bemenettel is hívni és megadja a kapcsolódó parancsokat is (pl. **randn**, ami standard normális eloszlású véletlen számokat generál 0 várható értékkel és 1 szórással).

A **doc** parancs a **help**-hez hasonlóan működik, csak egy jóval részletesebb leírás az adott parancsról, sok példával. Pl.

```
> doc randn
```

Próbáljuk ki a **pwd** parancsot! Kérdezzük le **help**-pel ez mit is csinál!

Másik hasznos parancs a **lookfor** utasítás. Ezzel parancs részletekre is rákereshetünk, vagy bármilyen szóra, ami a parancs leírásban szerepel. Próbáljuk ki a következőt:

```
> lookfor rand
```

Ez minden parancsot kilistáz, aminek a nevében, vagy a rövid leírásában szerepel a rand szó. Ha túl sokáig tartana a keresés, akkor megszakíthatjuk a parancsot a **CTRL + c** billentyű kombinációval.

ONLINE DOKUMENTÁCIÓ

Octave help-je: https://www.gnu.org/software/octave/doc/interpreter/

Matlab help-je: http://www.mathworks.com/help/matlab/

Sok hasznos segítség, letölthető matlab fájl található a matlab centralon is: http://www.mathworks.com/matlabcentral/

Online oktató anyag: https://matlabacademy.mathworks.com/

Egyéb gyakorló feladatok: https://www.mathworks.com/matlabcentral/cody/

NÉHÁNY HASZNOS PARANCS

`clc`	– kitörli a command window ablak tartalmát
`clear`	– kitörli a változókat (lásd workspace)
`close`	– bezárja az aktuális ábrát, vagy az összeset (close all)
`CTRL+C`	– félbeszakítja az adott parancsot (kilépés pl. végtelen ciklusból)
%	– megjegyzés (a program figyelmen kívül hagyja ami ez után van a sorban)
%%	– új szekció kezdhető vele a script-ben
;	– parancs végén a ; hatására nem jelenik meg az eredmény

'TAB' GOMB ÉS A NYILAK HASZNÁLATA A PARANCSSORBAN

Nagyon hasznos a 'tab' használata. Ha nem tudjuk pontosan egy adott parancs nevét, csak az elejét, és elkezdjük begépelni a parancssorba pl, hogy pref, majd utána nyomunk egy **tab**-t ha csak egyféle pref kezdetű parancs van, akkor kiegészíti, ha több, akkor megadja a lehetséges parancsokat. Itt több parancs is van ezzel a kezdettel pl. **prefdir** (annak a könyvtárnak a neve, ahol a beállítások, history stb. található) vagy a **preferences**, ami megnyitja a beállítások ablakot.

Szintén nagyon hasznos a nyilak használata a parancssorban, amivel korábbi parancsokat lehet újra előhozni, lefuttatni, módosítani. A korábbi parancsokat újra le lehet futtatni a **command history**-t használva is, dupla kattintással az adott parancson.

TAB	- befejezhető vele az elkezdett függvény vagy változó neve
↑↓ billentyűk	- görgetni lehet velük a korábban kiadott parancsok között
CTRL + ENTER	- lefuttatja az aktuális szekciót a scriptben
F9	- lefuttatja a kijelölt részt a scriptben
F5	- elmenti és lefuttatja az egész fájlt

ÉRTÉKADÁS, VÁLTOZÓ TÍPUSOK, FÜGGVÉNY HASZNÁLAT

```
> %% Értékadás, változótípusok
> % Egyszerű értékadás (0.01 háromféleképpen)
> a = 0.01
> b = 1e-2
> c = 1d-2;
> a+c
> clear a % kitörli az 'a' változót
> clear % vagy 'clear all' kitörli az összes változó értekét
> pi % beépített érték (3.14)
> e = exp(1) % e^1 = e = 2.71
> a = e^10 % hatványozás
> b = e^-10 % hatványozás
```

Néhány megjelenítési, formázási lehetőség:

```
> format long % több tizedes jegy megjelenítése
> e, a, b % 2.718281828459046, 2.202646579480674e+04,
  4.539992976248480e-05
> format short % rövidebb megjelenítés
> e, a, b % 2.7183, 2.2026e+04, 4.5400e-05
> format shortG % különböző nagyságrendű számok kompakt megjelenítése
> e, a, b % 2.7183, 22026, 4.54e-05
> format longG % különböző nagyságrendű számok hosszú megjelenítése
> e, a, b % 2.71828182845905, 22026.4657948067, 4.53999297624848e-05
```

A Matlab alap objektumai a mátrixok. A vektorok speciális mátrixok, pl. 1xn-es sorvektorok, vagy mx1-es oszlopvektorok. Mátrix/vektor definiálásához szögletes zárójelet használunk. Az elemeket egy soron belül vesszővel, vagy szóközzel választjuk el, sorok között az elválasztó a pontosvessző.

```
> z = [1 3 45 33 78] % sorvektor
> z = [1,3,45,33,78] % sorvektor másképp megadva
```

```
> t = [2; 4; 22; 66; 21] % oszlopvektor
> M = [1,2,3; 4,5,6] % 2x3-as méretű mátrix/tömb
```

Egy vektor megjelenítésekor problémát okozhat ha abban egyszerre vannak nagyon kicsi és nagyon nagy számok. Nézzük például a következő vektort:

```
> x = [25 56.31156 255.52675 9876899999];
> format short
> x
```

Az eredmény nehezen olvasható:

```
x = 1.0e+09 *
    0.0000    0.0000    0.0000    9.8769
```

Ebben az esetben jobb, ha más formázást használunk, például a **shortG** vagy **longG** formátumot, ami az eltérő nagyságrendű számokat eltérően jeleníti meg, minél kompaktabb formában (a **longG** több értékes jegyre).

```
> format shortG
> x
> format longG
> x
```

Eredményük:

```
x = 25       56.312       255.53          9.8769e+09 % format shortG
x = 25       56.31156     255.52675       9876899999 % format longG
```

Le lehet kérdezni egy vektor tetszőleges elemét, kerek zárójelbe téve az elem sorszámát:

```
> t(2) % eredménye: 4
> M(2,3) % eredménye: 6
> z(end) % z utolsó eleme: 78
```

Vagy felül lehet írni bármelyik elem értékét:

```
> t(2)=47
> p = [] % üres vektor
> z(3)=[]; % kitörli a 3. elemet, utána z = 1 3 33 78
```

Le lehet kérdezni egy részét a vektornak, mátrixnak:

```
> t(2:4) % eredménye az előző parancs után: 47 22 66
> t(1:29 % elgépelés esetén hibaüzenet
  t(39
     ?
 Error: Expression or statement is incorrect--possibly unbalanced (, {, or
[.
> t(1:29)
Index exceeds matrix dimensions.

> M(1:2,2:3) % eredménye: [2,3; 5,6]
```

Vektor, mátrix transzponáltja:

```
> tt = t' % t transzponáltja, sorvektor
> Mt = M' % eredménye: [1,4; 2,5; 3,6]
```

Vannak hasznos parancsok, amelyekkel egyszerűen lehet vektorokat előállítani:

```
> x1 = 1:10 % sorvektor 1-10-ig egész számok
> x2 = 1:0.3:10 % sorvektor 1-tol 10-ig, 0.3 osztásközzel
> x3= (1:0.3:10)' % oszlopvektor 1-tol 10-ig, 0.3 osztásközzel
```

```
> x4 = linspace(1,10,4) % 1 és 10 között 4 pontot vesz fel
```

Könnyű összefűzni vízszintesen, függőlegesen azonos sor/oszlop számú vektort/mátrixot.

```
> X = rand(2,3) % 2x3-as [0,1] közti véletlen számokból álló mátrix
> Y = ones(2,4) % 2x4-es egyesekből álló mátrix
> Z = eye(3) % 3x3-as egységmátrix
> W = zeros(2,4) % 2x4-es nullákból álló mátrix
> XY = [X, Y] % 2x7 elemű mátrix, vízszintesen összefűzve X és Y
> XZ = [X; Z] % 5x3-as mátrix, függőlegesen összefűzve X, Z
> XY2 = [X; Y] % hibaüzenet
Error using vertcat
Dimensions of matrices being concatenated are not consistent.
> XZ2 = [X,Z] % hibaüzenet
Error using horzcat
Dimensions of matrices being concatenated are not consistent.
```

Sorok oszlopok leválogatása

```
> XY(1,:) % XY első sora (: az adott sorban az összes elem)
> XY(end,:) % XY utolsó sora (: az adott sorban az összes elem)
> XY(:,1) % XY első oszlopa (: az adott oszlopban az összes elem)
> XY(:,end-1) % XZ utolsó előtti oszlopa (: az adott oszlop összes
  eleme)
```

Szövegek, mint karakterekből álló vektorok

```
> s = 'p' % szöveges/karakter (string) típusú változó 1x1 méretű
> me = 'Műszaki Egyetem' % string típusu változo 1x15 méretű
> bme = ['Budapesti ',me] % Budapest Műszaki Egyetem
> bme(13:17)
```

LEGGYAKORIBB VÁLTOZÓ TÍPUSOK

- Double: dupla lebegőpontos szám, a leggyakrabban használt (alapértelmezett) típus a valós számok tárolására.
- Integer: egész típusú szám
- Vector/matrix: Több azonos típusú szám tárolására szolgál sorokban, oszlopokban.
- Character array: karakter vektor, szöveg tárolására, egyszeres idézőjelek között, pl. 'Budapest'
- Cell array: Cella tömb, több dimenziós tömb, különféle típusú elemek is tárolhatóak benne.
- Structure: struktúra típus, elnevezett mezőkkel, különféle típusú elemek is tárolhatóak benne.
- Table: táblázat, táblázatos formában tárolt értékek, elnevezett oszlopokkal, különféle típusú elemek is tárolhatóak benne..

SCRIPT ÍRÁSA

Eddig parancssorból dolgoztunk, de egy bonyolultabb számítást, programot már nehéz parancssorban megírni, szükség esetén javítani. Célszerűbb lehet egy fájlba összegyűjteni a használt parancsokat, ez a script fájl. Itt is használhatunk minden korábbi Matlab függvényt, de egyszerűbb a javítás, futtatás. A Matlab alapértelmezett

fájl típusa a *.m fájl, ez egy egyszerű szövegfájl, amit bármilyen szövegszerkesztővel szerkeszthetünk. Ezen kívül az újabb Matlab verziókban már használhatjuk a livescript fájltípust is (*.mlx), ez utóbbiban vegyesen használhatunk Matlab utasításokat és formázott szövegeket, képeket, illetve láthatjuk rögtön az eredményeket is. Ez viszont a Matlab saját formátuma, amit csak Matlab programon belül tudunk megnyitni.

A script fájlokat futtathatjuk a nevük begépelésével, de egyszerűbb az **F5** megnyomásával, ez rögtön menti és futtatja a programot. Figyeljünk, hogy a fájlnév ne kezdődjön számmal és ne legyenek benne szóközök, ékezetek! A fájlnevek csak betűvel kezdődhetnek, és csak az angol abc betűi, illetve számok és alulvonás szerepelhet bennük.

Amennyiben nem akarjuk az egész programot lefuttatni betehetünk egy **return** parancsot valahová, és akkor csak addig fog lefutni a program. Illetve az **F9** lenyomásával csak a kijelölt rész fog lefutni. Másik megoldás, ha szekciókra osztjuk a fájlt dupla **%** jelek használatával (**%%**) és utána egy szekció cím megadásával. Egy bizonyos szekcióban lévő parancsokat a **CTRL+Enter** billentyűk megnyomásával tudunk futtatni. Kezdjünk egy új script fájlt a bal felső sarokban a plusz jelre kattintva (new), és ezzel megnyílik az Editorban egy üres lap. Mentsük el az aktuális könyvtárunkba gyak1.m fájl néven! A továbbiakban ebbe fogunk dolgozni.

EGYSZERŰ PLOTTOLÁS

Nézzük az alábbi táblázat adatait egy betonacél feszültség-fajlagos alakváltozás (σ-ε) diagramjából:

ε [%]	0	0.2	2	20	25
σ [N/mm^2=Mpa]	0	300	285	450	350

1. TÁBLÁZAT BETONACÉL FESZÜLTSÉG-FAJLAGOS ALAKVÁLTOZÁS DIAGRAMJA

Írjuk be a gyak1.m script fájlba:

```
> %% Betonacel alakváltozása
> x = [0,0.2,2,20,25] % kiírja a képernyőre a tartalmát
> y = [0,300,285,450,350]; % nem írja ki a képernyőre a tartalmát
```

Futtassuk vagy az **F5** paranccsal az egész fájlt, vagy egy kijelölt részt az **F9**-cel, vagy **CTRL+Enter**-rel a szakaszt!

Ha beírjuk a script fájlba, parancssorba egy létező változó nevét, akkor kiírja az aktuális tartalmát, még akkor is, ha az előző parancsnál az y után ; szerepelt, ezért ott ugyan nem írta ki a képernyőre a változó értékét, de a workspace-be elmentette!

```
> x, y
```

Vektor formátumban lévő adatokat a plot paranccsal rajzolhatunk ki:

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

Ez egy vonallal össze fogja kötni a pontokat. Ha a pontokat szimbólumokkal szeretnénk jelölni, próbáljuk ki a következőket:

```
> plot(x,y,'x')
> plot(x,y,'o-')
```

```
> plot(x,y,'r*-')
```

Hasznos paraméterek:

Marker	Description
o	Circle
+	Plus sign
*	Asterisk
.	Point
x	Cross
s	Square
d	Diamond
^	Upward-pointing triangle
v	Downward-pointing triangle
>	Right-pointing triangle
<	Left-pointing triangle
p	Pentagram
h	Hexagram

Long Name	Short Name	RGB Triplet
'yellow'	'y'	[1 1 0]
'magenta'	'm'	[1 0 1]
'cyan'	'c'	[0 1 1]
'red'	'r'	[1 0 0]
'green'	'g'	[0 1 0]
'blue'	'b'	[0 0 1]
'white'	'w'	[1 1 1]
'black'	'k'	[0 0 0]

LineWidth	vonalvastagság
MarkerEdgeColor	pont szimbólumok határoló vonalának a színe
MarkerFaceColor	pont szimbólumok kitöltése
MarkerSize	pont szimbólumok mérete

Használata, pl.:

```
> plot(x,y,'--gs','LineWidth',2,'MarkerSize',10,...
>       'MarkerEdgeColor','b','MarkerFaceColor',[0.5,0.5,0.5])
```

Elnevezhetjük a tengelyeket, vagy jelmagyarázatot, címet is fűzhetünk hozzá:

```
> xlabel('Fajlagos alakváltozás')
> ylabel('Feszültség')
> title('Betonacél feszültség-fajlagos alakváltozás diagramja')
```

EGYÉB HASZNOS TIPPEK PLOTTOLÁSHOZ

Minden ábrát illetve a rá kirajzolt elemeket elnevezhetjük, hogy később hivatkozni tudjunk rájuk. Így bármikor módosíthatók egyenként a tulajdonságaik, vagy akár törölhetőek is. Ha ugyanabba az ábrába rajzolunk egy új elemet, akkor alapértelmezetten törli az előző ábrát és csak az új fog látszódni, ha ezt el akarjuk kerülni, akkor ki kell adjuk a **hold on** parancsot.

```
> clf         % Az ábrán lévő elemek törlése
> f1 = figure; % Egy új ábra megnyitása
> p1 = plot(x,y,'r*');
> hold on  % a hold on kiadása után nem törli a korábbi elemeket
> p2 = plot(x,y);
> p3 = plot(x,y,'bo');
> delete(p1)  % elemeket törölhetünk a nevük megadásával egyenként is
```

```
> figure(1)   % az ábra nevének megadásával újra dolgozhatunk korábbi
  ábrákon is
> close % ábra bezárása
```

FÜGGVÉNYEK

Nagyon sok matematikai és egyéb beépített függvény van, ezek megismeréséhez célszerű a **help**-et, dokumentációt böngészni. Nézzük meg néhány függvényt az alap matematikai függvények közül (Elementary math functions) – ld. **help elfun**

A függvények változói mindig kerek zárójelben vannak. A trigonometrikus függvényeknél az alapértelmezett szögegység a radián!

```
> sin(pi) % értéke 0 a számábrázolás pontosságán belül
> cos(pi) % -1
> tan(pi) % végtelen helyett egy nagy szám
> log(100) % természetes alapú logaritmus
> log10(100) % 10-es alapú logaritmus
> 3^4 % értéke: 81, ua, mint
> power(3,4) % 81
> sqrt(81) % 9
> abs(-6) % 6
> exp(0)
```

Az **exp(0)** az e^0, értéke természetesen 1.

A beépített függvények nem csak számokon, hanem vektorokon is működnek.

```
> x = linspace(0,2*pi,40)
> y = sin(x)
> figure(1); plot(x,y)
```

További beállítások lásd **help plot**!

FELHASZNÁLÓ ÁLTAL DEFINIÁLT EGYSOROS FÜGGVÉNYEK

Többféleképp lehet Matlab-ban saját függvényeket megadni, az egyszerűbb esetekben leginkább az egysoros függvényeket használjuk (ezt az angol nyelven 'anonymous function'-nek nevezik, ami nincs elmentve külön programként, csak egy változóhoz van hozzárendelve), pl definiáljuk a cos(2x) függvényt!

```
> f = @(x) cos(2*x)
```

Itt először egy @ szimbólum után meg kell adni a függvény független változóit, majd utána jön a formula. Rajzoljuk fel az előbb felrajzolt szinusz függvény mellé ezt is, most nem előre megadott pontpárokat használva, hanem szimbolikusan az **fplot** vagy **ezplot** paranccsal! (Megj.: Octave-ban és régebbi Matlab verzióba az ezplot parancs használható, újabb Matlab-ban az fplot).

```
> hold on
> fplot(f,[0,2*pi])
```

A **hold on** parancs nélkül, ha egy új rajzolási parancsot adok ki, akkor letörli az előző ábrát és úgy rajzolja fel az újat, **hold on** esetén megtartja a régit, és mellé rajzolja az újat. A **hold off** paranccsal visszaállítható az alapértelmezett mód. Az **fplot** esetén

meg lehet adni az intervallumot, ha nem az alapértelmezett tartományon szeretnénk a függvényt megjeleníteni.

Az ábra törlését, grafikus ablak bezárását a következő parancsokkal tehetjük meg:

```
> clf % Clear current figure
> close % Close figure
```

Állítsuk elő saját függvényt használva 1-10-ig a számok négyzetét!

```
> f1 = @(x) x^2
```

Ellenőrizzük le egy adott x értékére:

```
> f1(3) % értéke: 9
```

Állítsuk elő az f függvényt használva 1-10-ig a számok négyzetét!

```
> x = 1:10;
> y = f1(x)
```

Erre hibaüzenetet kapunk:

```
Error using  ^
One argument must be a square matrix and the other must be a scalar. Use
POWER (.^) for elementwise power.
Error in gyak1>@(x)x^2
Error in gyak1 (line 100)
y = f(x)
```

Miért? Azért mert az x változónk egy sorvektor, négyzetre emeléskor pedig két sorvektort próbálunk összeszorozni, ami matematikailag helytelen. Ha nem vektor/mátrix műveletet szeretnénk végrehajtani, hanem azt szeretnénk, hogy elemenként hajtódjon végre a művelet, akkor egy pontot kell tenni a műveleti jel elé.

```
> f1 = @(x) x.^2
> y = f1(x) % 1  4   9  16  25  36  49  64  81  100
```

Mivel vektorok/mátrixok esetén az összeadás, kivonás, skalárral való osztás/szorzás eleve elemenként történik, ezért ezekben az esetekben nem kell pontot tenni a műveleti jel elé, csak a szorzás, osztás és hatványozás esetén: .*,./,.^.

A Matlabnak az egyik erőssége a vektorokkal, mátrixokkal való műveletek végzése, így sok esetben elkerülhetjük a lassabb ciklus műveletek alkalmazását.

FÜGGVÉNYEK KÜLÖN FÁJLBAN

A Matlab alapértelmezett fájltípusa a *.m kiterjesztésű szöveges fájl. Ennek két fő típusa van, script fájl és függvény (function). Az előbbit már használtuk az eddigi munkánk során, nézzük meg miben különbözik ettől a függvények írása!

A függvények külön fájlba történő megírásának egyik előnye, hogy nem csak abból a script fájlból használhatjuk, ahol épp dolgozunk, hanem bármely fájlból meghívhatóak, így ha van olyan feladat, ami gyakran ismétlődik, akkor azt célszerű egy külön függvényben megírni. Az egysoros függvényekkel szemben sokkal bonyolultabb számítások, műveletek elvégzésére használhatjuk őket, könnyebben paraméterezhetjük, hogy hány ki és bemenete legyen, magyarázatokat fűzhetünk hozzá.

Írjuk meg külön függvény fájlba az előbb megadott négyzet függvényt! Kattintsunk a bal felső sarokban a plusz jelre (new), és megnyílik az Editorban egy üres lap. írjuk be a következőket, majd mentsük el negyzetfv.m néven az aktuális könyvtárba. Fontos: a függvény neve (ami vastaggal van írva a következő kódban) meg kell egyezzen a fájl nevével, különben nem lehet meghívni!

```
> function y = negyzetfv(x)
> % Kiszámolja a bemenet négyzetét
>     y = x.^2;
> end
```

A függvények néhány fontos tulajdonsága:

- function szóval kezdődik
- Van legalább egy-egy kimenet és bemenet
- A kimenet, a függvény neve és a bemenet az első sorban található, és a függvény neve meg kell egyezzen az *.m fájl nevével
- A függvény belsejében valahol értéket kell adjunk a kimenetnek
- A függvény belső változói lokális változók, nem fognak megjelenni a workspace-ben, és a függvény sem fér hozzá a workspace-ben lévő változókhoz, csak ahhoz, amit megadtunk a bemenetnél.
- A függvényt nem lehet futtatni, csak egy másik fájlból, vagy parancssorból meghívni! A meghíváshoz a függvénynek az aktuális könyvtárban kell lennie (vagy egy olyan könyvtárban, ami benne van az elérési útban (path).
- A függvény első sora után megadott kommentek tartalmát listázza ki a help parancs az adott függvényre!

Hívjuk meg a megírt függvényt az x vektorunkra! Ehhez váltsunk vissza a gyak1.m script fájlra!

```
> negyzetfv(11) % 121
> negyzetfv(x) % 1  4  9  16  25  36  49  64  81  100
> help negyzetfv %  Kiszámolja a bemenet négyzetét
```

Egy függvénynek lehet több bemenete is, pl módosítsuk az előző függvényünket az alábbi módon, és mentsük el hatvany.m néven!

```
> function y = hatvany(x,p)
>     y = x.^p
> end
```

Egy függvénynek lehet több kimenete is egy vektorban összegyűjtve (hatvanyok.m):

```
> function [x2 x3 x4] = hatvanyok(x)
>     x2 = x.^2;
>     x3 = x.^3;
>     x4 = x.^4;
> end
```

Hívjuk meg a fenti függvényeket is a script fájlunkból!

```
> hatvany(x,3) % 1  8  27  64  125  216  343  512  729  1000
> [a b c] = hatvanyok(x)
> % a = 1 4  9 16 25 36 49 64 81 100
```

```
> % b = 1 8 27 64 125 216 343 512 729 1000
> % c = 1 16 81 256 625 1296 2401 4096 6561 10000
```

Ábrázoljuk az eredményeket egy új 3-as számú ábrán a **figure** parancs használatával! A **plot** parancsnál egymás után több összetartozó értéket is felsorolhatunk!

```
> figure(3)
> plot(x,a,x,b,x,c)
```

Mi is adhatunk egyéni színeket hozzá, illetve jelmagyarázatot is fűzhetünk hozzá olyan sorrendben megadva a jelmagyarázat szövegét, ahogy felrajzoltuk az ábrákat:

```
> plot(x,a,'black',x,b,'blue',x,c,'green')
> plot(x,a,'k',x,b,'b',x,c,'g')
> legend('négyzet','köb','x^4','Location','North')
> legend('négyzet','köb','x^4','Location','Best')
```

PROGRAM KOMMENTEK, 'HELP' ÍRÁSA SAJÁT FÜGGVÉNYEKHEZ

A több sorból álló programok fontos részei a kommentek. Egyrészt mások is megértik a programkódunkat, másrészt mi is emlékezni fogunk rá, ha később újra használni akarjuk. Célszerű nem csak a program elején, hanem minden új szekcióhoz is kommenteket írni. A Matlab-ban % jel után írhatunk kommenteket. Egy függvény esetében a kommentben célszerű megadni, hogy mi a függvény célja, milyen bemeneti és kimeneti értékek szerepelnek benne. Függvény esetében az első sor után megadott kommentek fognak megjelenni a help utasítás hatására segítségként.

MATLAB HIBAÜZENETEK

A programok írása során számos hibaüzenettel találkozunk, eddig is láttunk már néhányat. Fontos, hogy tudjuk ezeket értelmezni, ez segíthet kijavítani a hibáinkat!

Nézzünk egy példát elírásra a clear all helyett!

```
> cler all
Undefined function or variable 'cler'.
Did you mean:
>> clear all
```

A Matlab érzékeny a kis nagy betűk változására is:

```
> X = 3/4; x
Undefined function or variable 'x'.
```

Nézzünk példát egy szintaktikailag hibás Matlab utasításra:

```
> 1 X
1 X
  ↑
Error: Unexpected MATLAB expression.
```

Túl sok bemenő paraméter:

```
> sin(pi,3)
Error using sin
Too many input arguments.
```

Nem egyezik a mátrixban lévő sor/oszlop elemeinek száma:

```
   > M = [1 2;3]
Dimensions of matrices being concatenated are not consistent.
   > [3, 4, 5] * [1; 2; 3; 4]
Error using  *
Inner matrix dimensions must agree.
   > a = 1:5, b = 1:3
   > [a;b]
Error using vertcat
Dimensions of matrices being concatenated are not consistent.
```

Könnyen elgépelhető hiba lehet, hogy zárójel helyett 8 v 9 kerül beírásra:

```
   > sin(pi9
sin(pi9
       ↑
Error: Expression or statement is incorrect--possibly unbalanced (, {, or [.
```

Lemarad egy zárójel:

```
   > abs(sin(rand(2))
abs(sin(rand(2))
                ↑
Error: Expression or statement is incorrect--possibly unbalanced (, {, or [.
```

Elemenkénti műveletet akarunk végezni vektoron, de lemarad a pont:

```
   > v =1:4;
   > 1/v
Error using  /
Matrix dimensions must agree.
```

A legrosszabb, amikor nincs hibaüzenet, az eredmény mégis hibás. Példa: számítsuk ki $\frac{1}{2\pi}$ értékét a következő utasítással! Miért hibás az eredmény?

```
   > 1 / 2*pi %  1.5708
```

KIEGÉSZÍTÉS OCTAVE HASZNÁLATÁHOZ

Ha valaki az Octave használata mellett dönt otthoni gyakorláshoz, akkor az Octave a https://www.gnu.org/software/octave/ oldalról tölthető le, jelenleg a 5.1.0 változat, ami 2019. március 1-én jött ki. Az Octave előnye az egyetemi Matlab-bal szemben, hogy mivel egy nyílt forráskódú program, nemcsak tanuláshoz, oktatási célra használható, hanem munkához is. Sok kiegészítő csomag is van hozzá, lásd https://octave.sourceforge.io/ illetve https://octave.sourceforge.io/packages.php , ezek egy jó része telepítve is van, csak be kell tölteni használatkor. Le lehet kérdezni, hogy mi van telepítve a **pkg list** paranccsal).

SYMBOLIC CSOMAG TELEPÍTÉSE

A numerikus módszerek gyakorlatok során szimbolikus számításokat is végezni fogunk, ehhez szükséges az Octave-ban egy extra symbolic csomagot telepíteni (ez a matlab-ban is egy külön toolbox). Ez a csomag python-sympy alapra épül, ezért külön kell telepíteni. Windows alatti telepítés (a következő link alapján: https://github.com/cbm755/octsympy/wiki/Notes-on-Windows-installation)

1. Töltsük le a symbolic-win-py-bundle-x.y.z.zip fájlt következő oldalról : https://github.com/cbm755/octsympy/releases (itt x.y.z a verziószám, pl.

symbolic-win-py-bundle-2.8.0.tar.gz) Ez tartalmazza a Python interpretert és a SymPy-t is.

2. Indítsuk el az Octave-t, állítsuk be aktuális könyvtárnak, ahová mentettük az előbbi fájlt.
3. Gépeljük be a következő parancsot (x.y.z helyett a megfelelő verziószám):

```
> pkg install symbolic-win-py-bundle-x.y.z.tar.gz
```

4. Töltsük be Octave-ba a telepített csomagot:

```
> pkg load symbolic
```

Ellenőrizzük, hogy működik-e pl.

```
> syms x
> f = sin(cos(x));
> diff (f)
```

Eredmény ⇒ `-sin(x)·cos(cos(x))`

A symbolic csomag függvényei részletesen:
https://octave.sourceforge.io/symbolic/overview.html

HASZNÁLT MATLAB BEÉPÍTETT FÜGGVÉNYEK

help	-	matlab helpjének kategóriái, vagy segítség megadott témakörhöz, függvényhez a Command Window-ban
rand	-	Véletlen számok 0-1 között egyenletes eloszlásban
randn	-	Véletlen számok sztenderd normális eloszlásban, 0 várható értékkel és 1 szórással
doc	-	részletes dokumentáció adott függvényhez, parancshoz új ablakban
lookfor	-	keresés a help-ben adott szóra, szórészletre
clc	-	kitörli a command window ablak tartalmát
clear, clear all	-	kitörli a megadott változókat, vagy az összes változót
close, close all	-	bezárja az aktuális ábrát, vagy az összeset
CTRL+C	-	félbeszakítja az adott parancsot (kilépés pl. végtelen ciklusból)
%	-	megjegyzés (a program figyelmen kívül hagyja ami ez után van a sorban)
;	-	parancs végén a ; hatására nem jelenik meg az eredmény a Command Window-ban
Tab gomb	-	elkezdett parancsot kiegészíti
preferences	-	megnyitja a beállítások ablakot
prefdir	-	annak a könyvtárnak a neve, ahol a beállítások, history stb. található
↑↓ gombok	-	korábbi parancsokat vissza lehet hozni a Command Window-ba
pi	-	3.14…. (pi szám)

exp(1), exp(n)	-	e^1 =2.71…, e^n
^, power	-	hatványozás
format long, format short	-	több tizedes jegy (14) vagy kevesebb (4) megjelenítése
format shortG, format longG	-	különböző nagyságrendű számok kompakt formában történő megjelenítése (5 vagy 15 értékes jegyre).
[1, 2, 3; 4, 5, 6]	-	vektor, mátrix megadása
'	-	vektor, mátrix transzponáltja
[A,B] vagy [A B]	-	mátrixok összefűzése egymás mellé (sorok száma egyenlő)
[A;B]	-	mátrixok összefűzése egymás alá (oszlopok száma egyenlő)
A(1,:)	-	mátrix első sora
A(:,1), A(:,end)	-	mátrix első/utolsó oszlopa
linspace(x1,x2,n)	-	[x1,x2] intervallumban n pont felvétele egyenletesen
ones	-	egyesekből álló mátrix
zeros	-	nullákból álló mátrix
eye	-	egységmátrix
figure	-	új ábra nyitása
plot	-	összetartozó pontpárok felrajzolása
xlabel, ylabel	-	x,y tengely feliratozása
title	-	ábra címe
sin, cos, tan	-	szögfüggvények (alapértelmezett mértékegység a radián!)
log, log10	-	természetes alapú logaritmus, 10-es alapú logaritmus
sqrt	-	négyzetgyök
abs	-	abszolút érték
hold on, hold off	-	felülírja, vagy ne írja felül a meglévő ábrát az új ábrával
fplot, ezplot	-	függvények felrajzolása
.* ./ .^	-	elemenkénti szorzás, osztás, hatványozás vektoroknál
clf	-	ábra törlése (nem zárja be az ablakot)
legend	-	jelmagyarázat
return	-	Visszatérés – eddig a pontig hajtja végre az F5 a programot