



# Matlab 4. előadás

Elemi függvények és saját függvények

Dr. Szörényi Miklós,  
Dr. Kallós Gábor

2017–2018





## Tartalom

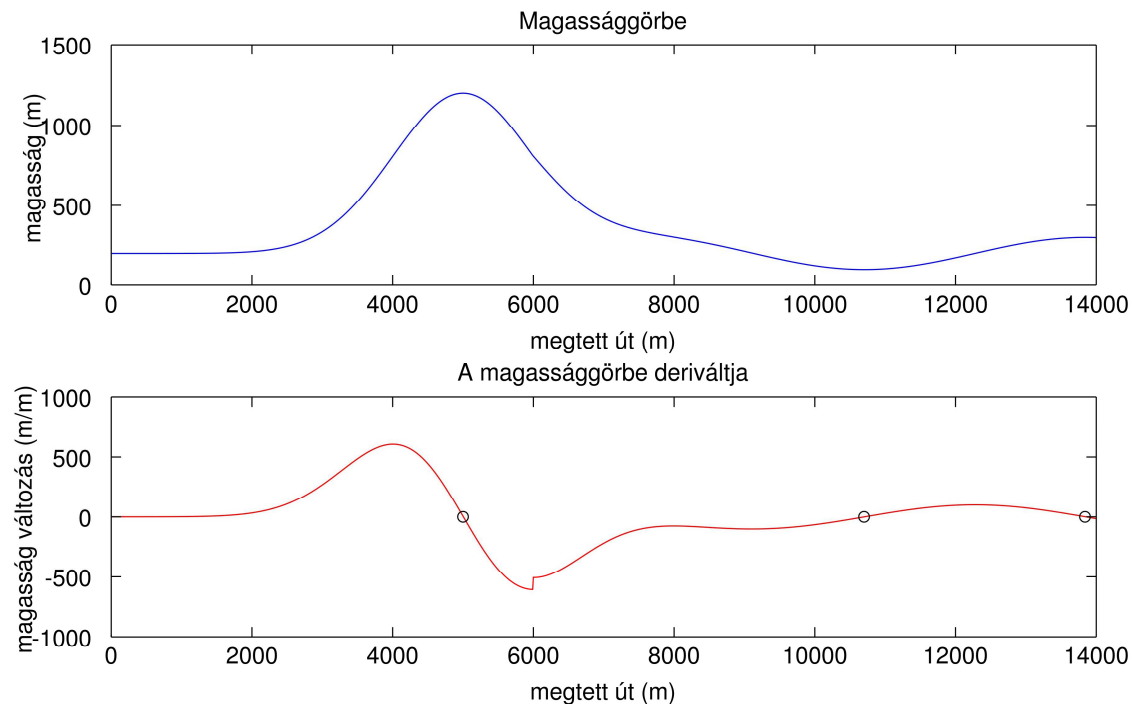
- Bevezetés, motiváció
- Elemi függvények
  - Trigonometrikus és exponenciális csoport
    - Maximális/minimális érték
  - Diszkrét matematikai csoport
  - Polinomfüggvények
- Optimalizációs feladatok
- Saját függvények definiálása és hívása
  - M-fájlos definíció
    - Editor
    - Hibák és warning-ok
  - Parancssori definíció





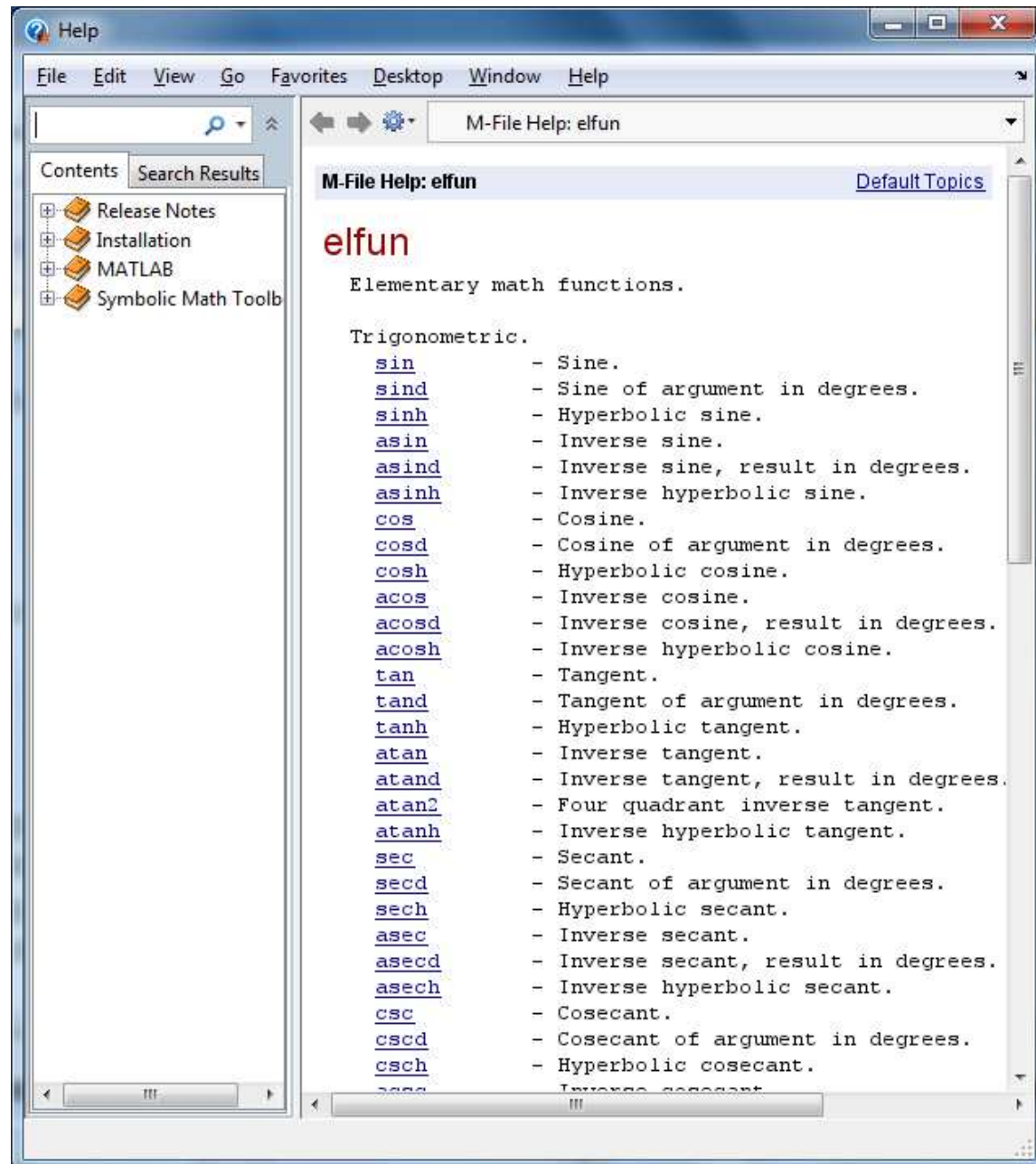
## Bevezetés, motiváció

- Célunk a továbbiakban különböző függvények definiálása ...
  - Pl. fizikai jelenségek modellezésére / mérési adatok közelítésére
- ... vizsgálata ...
  - Pl. nevezetes pontok keresése
- és kirajzolása
  - Igényes ábrán: tengelyek, feliratok, elkülönülő vonalak, jelmagyarázat stb.
- Ehhez most áttekintjük az elemi matematikai függvények csoportját, majd megismerkedünk a function m-fájlokkal



## Elemi függvények

- Elfun csoport
  - A Mathematics kategórián belül
- Alcsoportok
  - Trigonometrikus ...
  - Exponenciális ...
  - Komplex ...
    - Már néztük
  - Kerekítő ...
    - Már néztük
  - Diszkrét matematikai ...
  - Polinomfüggvények





## Elemi függvények

### Trigonometrikus függvények

- Radiános és fokok megadás (d-s változat, már tudjuk)

- Példa

```
>> x = 0:0.1:1.6;  
>> y = sin(x) % szinusz vektor  
>> yd = sind(x)
```

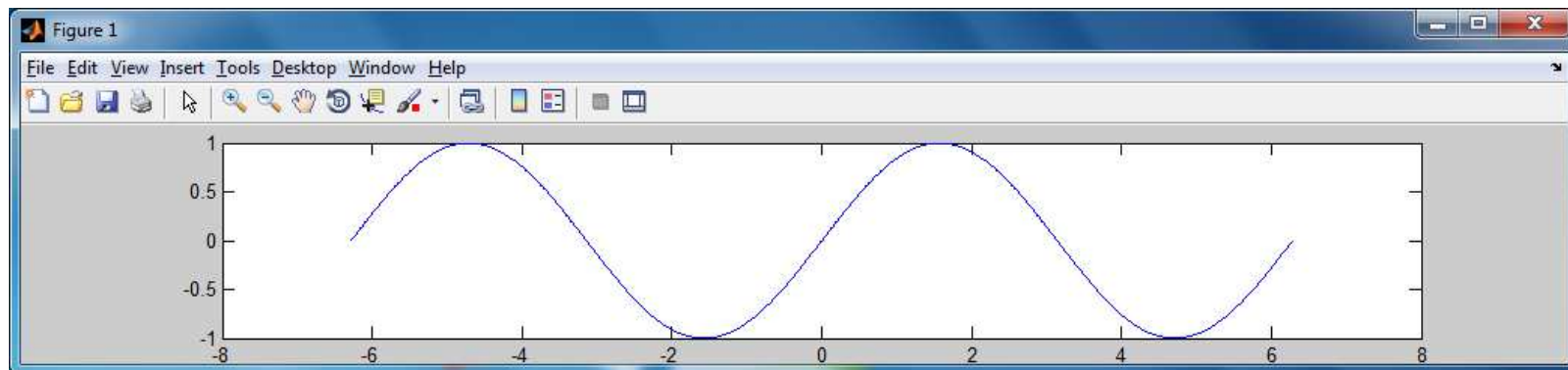
- Függvényrajz (a plot utasítást részletezően később nézzük meg)

- Példa

```
>> x = linspace(-2*pi, 2*pi, 1001); plot(x,sin(x))
```

- Hf.:

- Készítsünk igényes, szép rajzot a tangens függvényről  $-\pi/2$ -től  $+\pi/2$ -ig!





## Elemi függvények

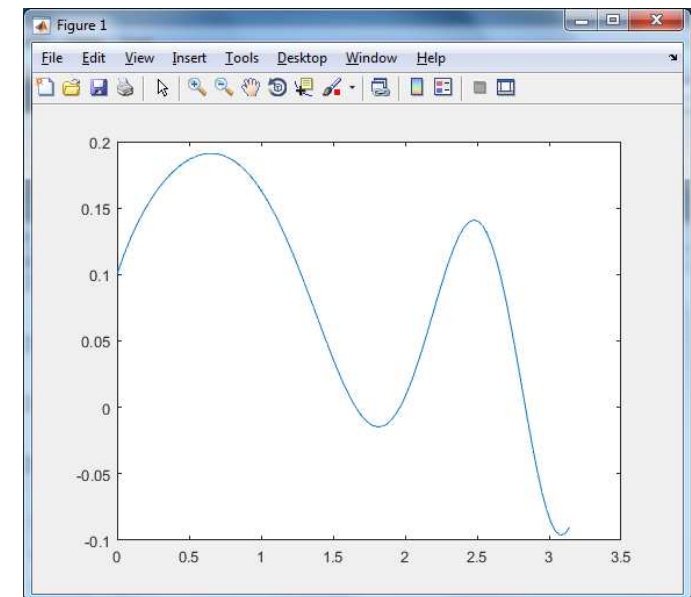
### Trigonometrikus függvények (folyt.)

- Feladat (M. D.): Hozzunk létre egy  $t$  idővektort, amely  $\epsilon$  és  $\pi$  között 100 elemet tartalmaz! Ezen a vektoron értékeljük ki a  $f_v = \sin(t)/(4,7t + 3) + 0,1\cos(t^2)$  kifejezést, majd határozzuk meg, hogy hol maximális  $f_v$  értéke!

- Megoldás

```
>> t = linspace(eps,pi,100); % idővektor
>> fv = sin(t)./(4.7*t+3) + 0.1*cos(t.^2);
% trig. kif., kell a pontozott műv.
>> plot(t, fv) % próbarajz
>> [max_ertek max_index] = max(fv)
max_ertek = 0.1911
% max. érték és index megkeresése
% szintaktika: [M, I] = max(kif)
% lásd súgó
>> max_t = t(max_index)
% indexhez tartozó t érték
```

- Ugyanezen ötlettel megoldhatunk optimalizációs feladatokat



## Optimalizációs feladatok

- Feladat (M. D.): Egy henger felszíne 50 egység. Mekkora-nak válasszuk az alapkör sugarát és a magasságot, hogy a térfogata max. legyen?

- Tudjuk, hogy  $A = 2\pi r(r + h)$  és  $V = r^2\pi h$

- Ötlet és megoldás Matlabbal

- $A = 2\pi r(r + h) \Rightarrow h = 50/(2\pi r) - r$

- $V = r^2\pi h \Rightarrow V = r^2\pi(50/(2\pi r) - r)$

- Megnézzük  $V$  értékeit a lehetséges  $r$ -ekre, és maximumot keresünk

- ```
>> r = linspace(0,3,50);
```

- ```
% elég 3-ig menni (próbálkozás)
```

- ```
>> v = r.*r*pi.*(50./(2*pi*r)-r);
```

- ```
% kell a pontozott műv.
```

- ```
>> plot(r, v)
```

- ```
>> [max_ertek max_index] = max(v) % max. keresés
```

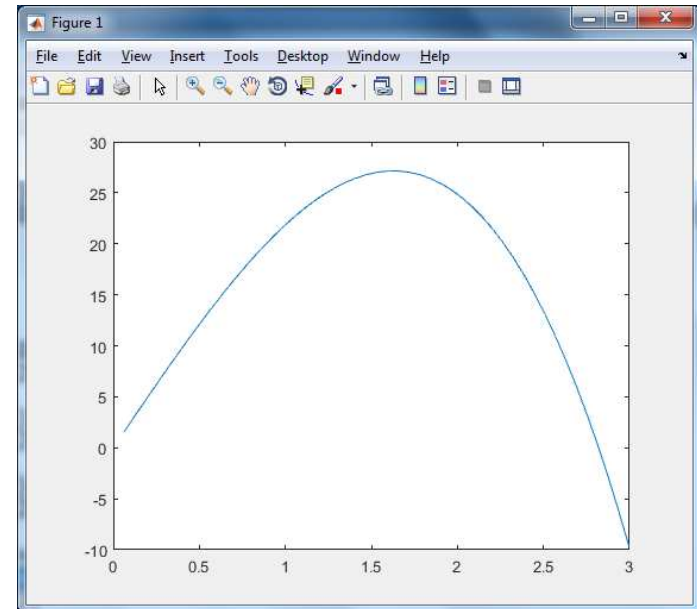
- ```
max_ertek = 27.1354
```

- ```
>> max_r = r(max_index)
```

- ```
max_r = 1.6531
```

- ```
>> h_max = 50/(2*pi*max_r) - max_r
```

- ```
>> v_max = max_r*max_r*pi*(50/(2*pi*max_r)-max_r) % ellenőrz.
```





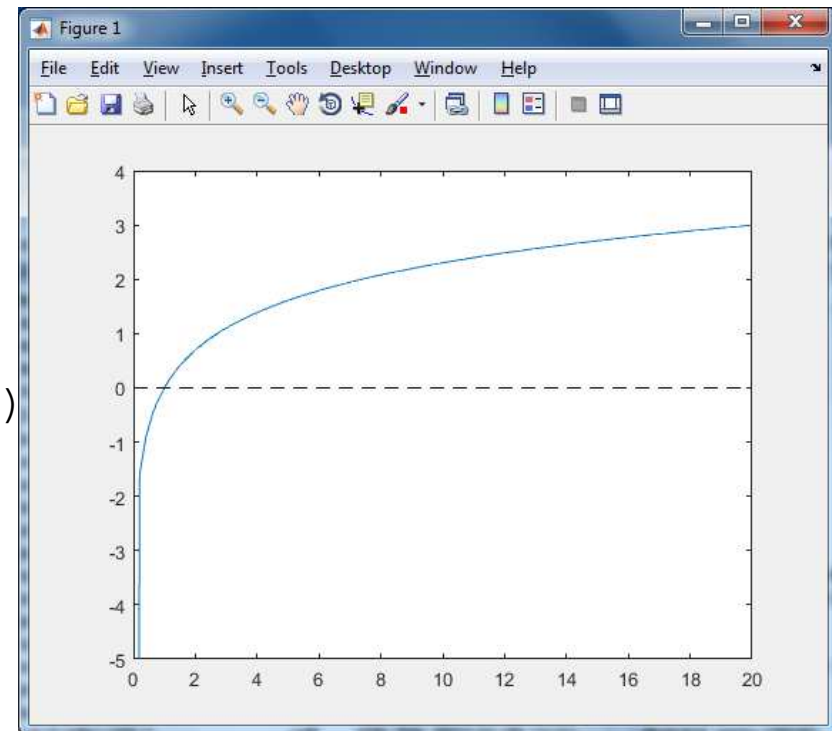
## Elemi függvények

### Exponenciális függvények

- Fontosabbak:  $\exp$ ,  $\log$ ,  $\log_{10}$ ,  $\log_2$
- Feladat: „szép” logaritmus
- Megoldás

```
>> x = linspace(eps, 20, 100);  
>> y = log(x);  
>> plot(x, y)  
% még nem szép rajz  
>> axis([0 20 -5 4])  
% tengely skálázás  
% lásd később is  
>> hold on  
>> plot([0 20], [0 0], 'k--')  
% fekete szaggatott vonal  
% lásd később is
```

- A komplex és kerekítő fv-eket nem nézzük meg most külön





## Elemi függvények

Diszkrét matematikai (számelméleti) függvények

■ Fontosabbak: *gcd*, *lcm*, *factor*, *isprime*, *primes*

■ Példák

```
>> primes(31)
>> isprime(1001)
>> factor(1001)
>> 7*11*13 % ell.
```

■ Euklideszi algoritmus megvalósítása\*

**gcd**  
Greatest common divisor

**Syntax**

```
G = gcd(A,B)
[G,C,D] = gcd(A,B)
```

**Description**

`G = gcd(A,B)` returns an array containing the greatest common divisors of the corresponding elements of integer arrays `A` and `B`. By convention, `gcd(0,0)` returns a value of 0; all other inputs return positive integers for `G`.

`[G,C,D] = gcd(A,B)` returns both the greatest common divisor array `G`, and the arrays `C` and `D`, which satisfy the equation:  $A(i) \cdot C(i) + B(i) \cdot D(i) = G(i)$ . These are useful for solving Diophantine equations and computing elementary Hermite transformations.

[Open Help Browser](#)

F1 to toggle focus; Escape to close

The screenshot shows the MATLAB 7.9.0 (R2009b) interface. The Command Window contains the following code:

```
>> % Beépített függvény
>> [lnko, x, y] = gcd(460071, 142569), 460071*x + 142569*y
lnko =
    279
x =
    163
y =
   -526
ans =
    279

>> % Mi írjuk meg az eljárást
>> p = 460071, q = 142569
p =
    460071
q =
    142569
>> while q > 0 r = mod(p,q); p = q; q = r; end; lnko = p
lnko =
    279
fx >>
```

The Workspace window on the right shows the following variables and their values:

| Name | Value |
|------|-------|
| ans  | 279   |
| lnko | 279   |
| p    | 279   |
| q    | 0     |
| r    | 0     |
| x    | 163   |
| y    | -526  |

The Command History window shows the following commands:

```
p = 460071, q = 142569
while q > 0
p = 460071, q = 142569
% Beépített függvény
[lnko, x, y] = gcd(460071, 142569)
% Mi írjuk meg az eljárást
p = 460071, q = 142569
while q > 0
```



## Elemi függvények

### Polinomokat kezelő függvények

- A polinomok gyakran szerepelnek mat. függvények megadásakor, de sokszor valós fizikai folyamatoknál is (pl. közelítés)
  - Eml.: polinom meghatározása, műveletek (deriválás, integrálás is)
- Egy polinom a Matlabban egy sorvektorral reprezentálható
  - Példa:  $P(x) = 3x^4 - 15x^3 - 10x + 2 \Rightarrow P = [3, -15, 0, -10, 2]$
- A Matlab beépített függvényei a sorvektorokat polinomokként értelmezik, és végrehajtják a megfelelő műveletet (pl. *polyval*, *conv*)
- Példák
  - ```
>> roots(P) % gyökök meghatározása (eml.: gyök)
```
  - ```
>> y0 = polyval(P,x0) % kiértékelés (egy adott pontban)
```
  - ```
>> x = -1:0.1:1; polyval(P,x) % kiértékelés (sok pontban)
```
  - ```
>> r=[0 1 2 3], p1 = poly(r) % pol. létrehozása gyökeiből
```
  - ```
>> pd1 = polyder(p1) % derivált polinom (együtthatók)
```
  - ```
>> q = polyint(pd1) % integrálás
```
  - ```
>> p2 = polyfit([0 1 2 3], [1 1 0 1], 3) % pol. illesztés
```
- Megj.: Polinomokra vonatkozó feladatok a Matlabban sokszor pol.függvények nélkül is megoldhatók (pl. fv. ábrázolás, zérushelyek, szélsőért.)





## Saját függvény létrehozása és hívása

- A saját függvények definícióit *célszerűen* m-fájlokban helyezzük el (máshogy is lehet!)
  - Létrehozás: File/New/Function (Home menüszalag)
- Szintaktika:  
**function** kimenő paraméter(ek) = **függvéynév**(bemenő paraméterek)  
utasítások;  
**end**    % függvéynév
- A „Matlab” kottát ad a kitöltéshez
  - Felesleges sorok törölendők
  - Figyeljünk a paraméterekre! (aktuális, formális, bemenő, kimenő)
- Szabályok
  - A függvény neve *kötelezően* azonos a tartalmazó m-fájl nevével
  - Több visszatérési/kimenő paraméternél szögletes zárójelet kell használni a megadásnál
  - Paraméterek szeparálása: szóköz, ill. ‘,’

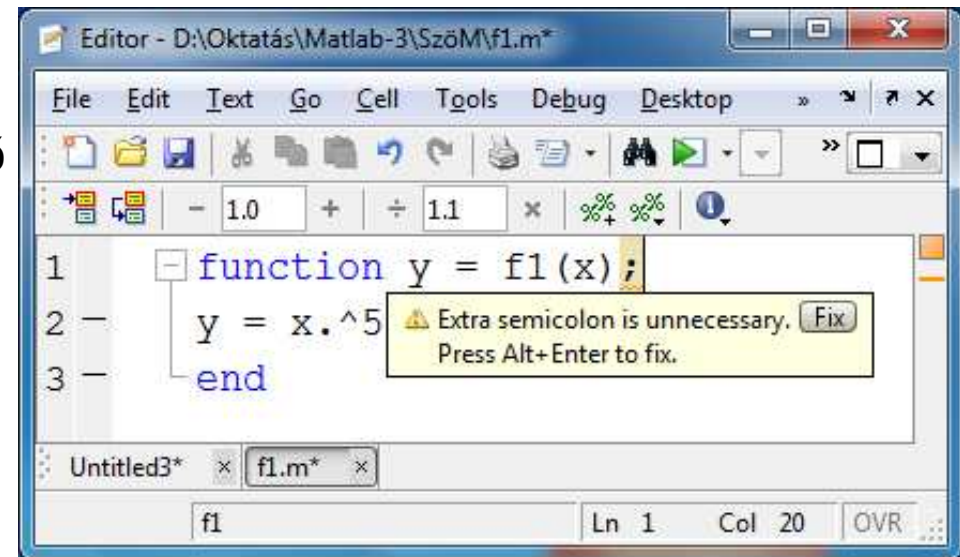
```
Editor - Untitled3*
File Edit Text Go Cell Tools Debug Desktop Window Help
1 function [ output_args ] = Untitled3( input_args )
2 %UNTITLED3 Summary of this function goes here
3 % Detailed explanation goes here
4
5 end
```



## Saját függvény létrehozása és hívása

M-fájlos definíció, folyt.

- Kezelőfelület (Editor)
  - Egyszerű, barátságos szerkesztő
  - A Matlab a beírás közben jelzi a hibákat, és segítséget ad a javításhoz
- Hibák és warning-ok
  - Lásd csúszka is
  - Előbbiek kötelezően javítandók!
- A hiba lehet: szintaktikai és szemantikai
  - Tipikus kellemetlen hiba a pontozott műveletek nem-alkalmazása (amikor szükségesek lennének)
- Használjunk kommenteket!
  - Az infó a help-nél megjelenik
  - Később (pl. 1 év múlva) számunkra is fontos lehet...



## Saját függvény létrehozása és hívása

M-fájlos definíció, folyt.

- Példafüggvény 1:  $fv(x) = 1/x + 2x^2$

- `function y = fv(x)`  
`y=1./x+2*x.^2; % pontozott műveletek, kell a pontosvessző`  
`end`

- Hívás:

- `>> x = 0.01:0.01:5; plot(x, fv(x))`

- Példa 2: hagyományos koordinátákból polárba váltó függvény

- Több bemenő és több kimenő paraméter!

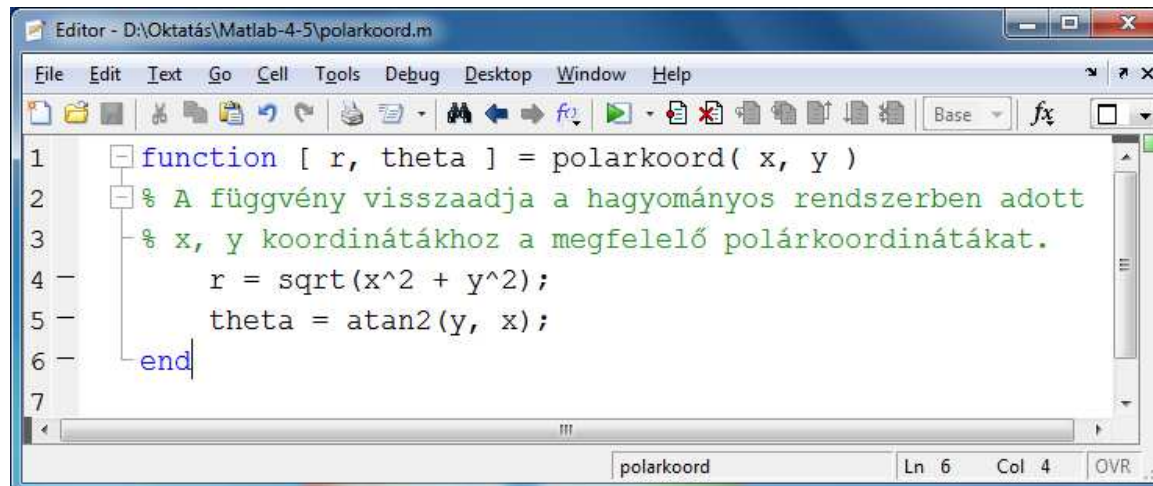
- Hívás:

- `>> polarkoord(1, 1)`

- `% csak r-et kapjuk meg`

- `>> [r, theta] = polarkoord(1, 1)`

- `% r-et és theta-t is megkapjuk, most pi/4`



```
Editor - D:\Oktatás\Matlab-4-5\polarkoord.m
File Edit Text Go Cell Tools Debug Desktop Window Help
1 function [ r, theta ] = polarkoord( x, y )
2 % A függvény visszaadja a hagyományos rendszerben adott
3 % x, y koordinátához a megfelelő polárkoordinátákat.
4     r = sqrt(x^2 + y^2);
5     theta = atan2(y, x);
6 end
7
```

## Függvény létrehozása és hívása

A Matlab további lehetőségei

- Közvetlen függvénydefiníció (parancssorban, ún. anonymous megadás)

- Létrehozás: @ jellel (function handle), nem tárolódik külön fájlban

- Példa 1.

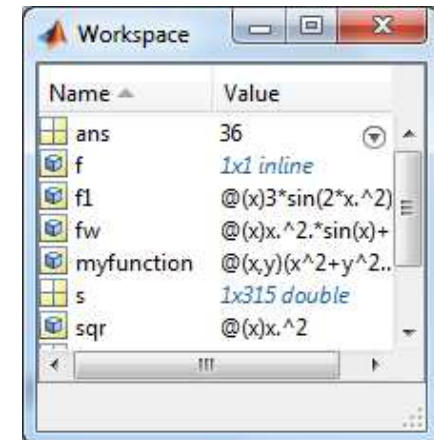
```
>> sqr = @(x) x.^2;  
% input paraméterek zárójelben  
>> sqr(6) % hívás
```

- Példa 2.

```
>> fw = @(x) x.^2.*sin(x) + 1; % saját fv.  
>> s = -3:0.1:3; plot(s, fw(s)) % hívás
```

- Példa 3.

```
>> fn = @(x,y) (x^2 + y^2 + x*y) % kétváltozós eset  
>> x = 1; y=10; z = fn(x,y) % kezdőérték, hívás
```



- Inline megadás

- A fv. egy sztringben megadott kifejezés (újabb Matlab verziókban *már kerülendő!*)

```
>> f = inline('3*sin(2*x^2)'), argnames(f), fplot(f, [0 pi])  
% az argnames itt biztonsági ellenőrzésre szolgál  
% fplot esetén a . elhagyható a megf. műveletek előtt
```

