

4. GYAKORLAT

MÁTRIXFÜGGVÉNYEK, SAJÁT FÜGGVÉNYEK, GRAFIKA

SÁVMÁTRIXOK, ALSÓ- ÉS FELSŐHÁROMSZÖG MÁTRIXOK

A `diag` parancs felhasználásával kiemelhetjük egy mátrix főátlóját vagy valamelyik mellékátlóját, ill. vektor felhasználásával diagonális mátrixot készíthetünk (az adott vektor a főátlóba vagy valamelyik mellékátlóba elhelyezhető).

Feladat

Legyen $D = [1:4; 11:14; 21:24; 31:34]$.

Vegyük ki D főátlóját, majd készítsük el azt a mátrixot, amely csak D főátlójának elemeit tartalmazza, és a többi eleme 0.

Mit eredményez a `diag(D, -1)` és a `diag(diag(D, -1), -1)` kifejezés?

Az előző mintára állítsuk elő azt a kifejezést, amelyik olyan mátrixot generál, amelyben csak a D mátrix felső mellékátlójának elemei nem 0 értékűek!

Feladat

Ellenőrizzük, hogy a `diag(diag(D, -1), -1) + diag(diag(D, 1), 1) + diag(diag(D))` kifejezés a D mátrix olyan sávmátrixú részét emeli ki, amelyben csak a főátló és a két első mellékátló elemei szerepelnek.

Mit állít elő a `diag(diag(D, -1), 1) + diag(diag(D, 1), -1) + diag(diag(D))` kifejezés?

A `triu` és a `tril` parancs felső- ill. alsóháromszög mátrixot készít az alapmátrixból.

Feladat

Állítsuk elő D alsó- és felsőháromszög mátrixát!

Mit eredményez a `triu(D, 1)` és a `tril(D, -1)` kifejezés?

Ez alapján állítsuk elő D -t oly módon, hogy összerakjuk (most: mátrixösszeadással!) a főátlójából, az alsó- és a felsőháromszög mátrixából!

A PLOT UTASÍTÁS

A `plot` utasítás a legegyszerűbb esetben (x, y) pontpárok összekötött megjelenítésére szolgál (a pontok koordinátáit vektorok tartalmazzák). A szintaktika: `plot(x, y)`.

Feladat

Ábrázoljuk a $[0, 0]$ és $[1, 1]$ pontok által meghatározott szakaszt!

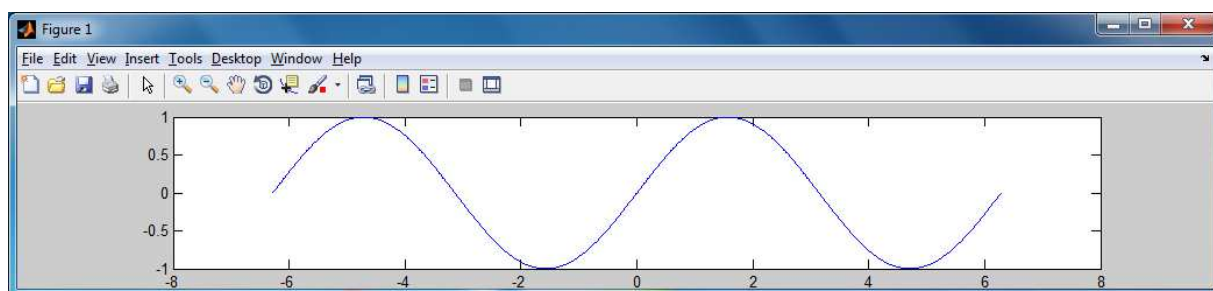
Először az alapértelmezett színt használjuk, utána legyen zöld, majd fekete a vonal.

Függvényábrák készítésénél úgy indulunk el, hogy egy vektorba legyártjuk az alappontokat (linspace parancs vagy : operátor), majd erre húzzuk rá a függvényt.

Példa

Rajzoljuk ki a $\sin(x)$ függvény grafikonjának pontjait a $[-2\pi, 2\pi]$ intervallumban 1001 pont segítségével!

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



Az alappontok megfelelően sűrű előállítása kulcslépés, anélkül a grafikonunk nem lesz korrekt. **F:** Nézzük meg, hogy mi történik, ha az x sorozat csak 11 elemű!

Több rajz egy ábrán a hold on/off parancsokkal jeleníthető meg. A hold on kiadása után minden ábra egymásra kerül mindaddig, amíg a hold off parancsot ki nem adjuk.

Feladat

Ismételjük meg az előző két grafikon kirajzolását, de most már egy közös ábrán! A vonalak színe legyen különböző (pl. piros és kék)! (Próbáljuk ki a vonalstílus megváltoztatását is.)

Tipp: hold on és hold off között gyártunk le a megfelelő x vektorokat (pl. x1 és x2 néven), és adjuk ki a rajzoló utasításokat.

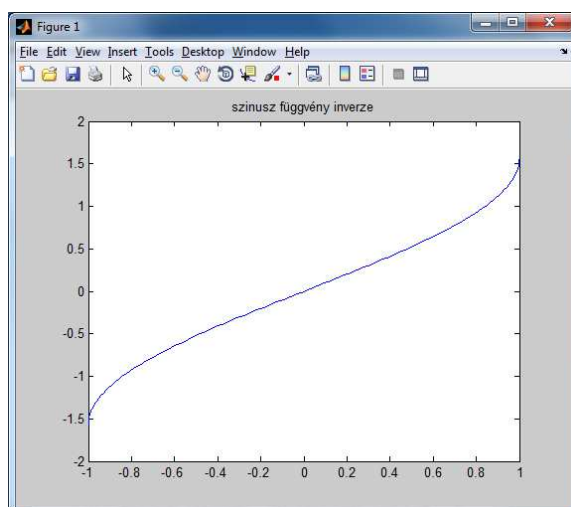
Ismételjük meg az előző kirajzoló utasításokat úgy, hogy a vonalvastagságot is változtatjuk, és a vonalszínt az RGB skálán állítjuk be.

```
>> x1 = linspace(...); plot(x1,sin(x1),  
'LineWidth',1,'Color',[1 0 0]);
```

Ha a plot parancs megadásánál az x és az y sorozatot felcseréljük, akkor így módon az inverz függvényt tudjuk direkt módon kirajzoltatni. A szintaktika ekkor tehát plot(y, x).

Feladat

Rajzoltassuk ki a $\sin(x)$ függvény inverzét a $[-1, 1]$ intervallumban!



Kétféle módon is oldjuk meg a feladatot: a `plot(y, x)` szintaktikával, és az `asin` inverz függvény felhasználásával!

Tegyük az ábrára feliratot!

Ezután oldjuk meg a feladatot úgy is, hogy egy ábrán helyezzük el a szinusz függvényt (megfelelő szakasz) és inverzét! (Készítsünk ehhez scriptet – script M-fájlt.)

Megoldás (részlet)

```
>> x = linspace(-pi/2, pi/2, 101); y = sin(x); plot(y, x);  
title('szinusz függvény inverze') % plot(y, x) szintaktika  
vagy  
>> x = linspace(-1, 1, 101); plot(x, asin(x)) % inverz függvény
```

Feladat (forgatómátrix)

Készítsük el az origó körüli α fokos forgatást megvalósító mátrixot, és a felhasználásával forgassunk el egy adott háromszöget! Legyen például $\alpha = 80^\circ$, a háromszög pontjai pedig rendre $A(1, 1)$, $B(4, 0)$ és $C(3, 4)$.

Mutassuk be megfelelő ábrán az eredeti és az elforgatott háromszöget! (Rakjunk ki feliratot is.)

Tipp:

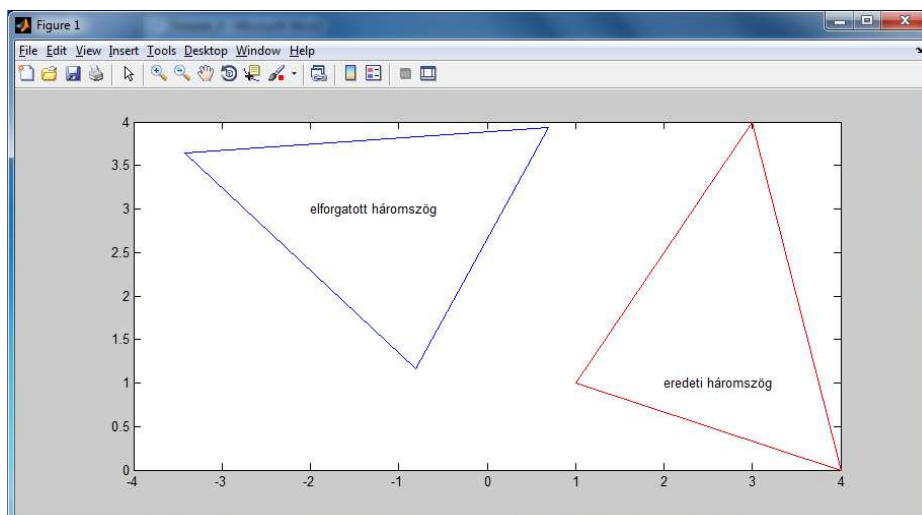
Az α fokos forgatást megvalósító mátrix alakja: $A = [\cos(\alpha) \ -\sin(\alpha); \sin(\alpha) \ \cos(\alpha)]$, ahol az α szög értéke radiánban adott.

Ennek megfelelően a Matlabos megvalósításban a `cosd` és a `sind` függvényeket használjuk.

A háromszög pontjait vegyük fel koordinátáinként egy megfelelő vektorban, majd szorozzuk össze a transzformációs mátrixot a vektorkoordinátákkal.

Az ábrázolásnál figyeljünk arra, hogy az *alakzat záródjon*, azaz a harmadik pontot is kössük össze az elsővel!

(További segítségként érdemes felidézni az előadáson szereplő `dot2dot` és `forгат` eljárást.)



SAJÁT FÜGGVÉNY LÉTREHOZÁSA, FPLOTT

Feladat

Ábrázoljuk az $f: x \rightarrow \sin(x)/(x^2 + 1)$ függvényt a $[-10, 10]$ intervallumon úgy, hogy ehhez saját függvényt (function M-fájlt) hozunk létre!

Tipp: A File/New menüponttal hozzunk létre egy új function m-fájlt, ennek neve legyen f.m. Alakítsuk ki a fejléct és függvény törzsét megfelelő módon (pl. $y = \dots$).

Vigyázzunk arra, hogy egyes helyeken pontozott műveleteket kell használni!

Mentsük el az f.m fájlt a d: meghajtó megfelelő könyvtárába.

A kirajzoláshoz hozzunk létre egy x vektort -10 -tól $+10$ -ig, $0,1$ -es lépésközzel. Az ábrázolás ezek után egyszerűen a `plot(x, f(x))` paranccsal történhet.

A saját (és a beépített) függvények ábrázolása az **fplot** utasítással is végrehajtható. Ekkor a Matlab automatikusan generál osztáspontokat, az x vektort tehát nem kell nekünk létrehozni.

Az utasítás szintaktikája:

`fplot('függvényképlet', [intervallum határok], 'megjelenés-vezérlő')`

Feladat

Ábrázoljuk az `fplot` paranccsal a szinusz függvényt a $[0, 2\pi]$ intervallumban!

Feladat

Ábrázoljuk az előző $f: x \rightarrow \sin(x)/(x^2 + 1)$ függvényt a $[-10, 10]$ intervallumon az `fplot` utasítással!

Fontos: az `fplot` parancs *elfogadja az olyan függvénydefiníciót is, ahol nem pontozott műveleteket írtunk!* (Próbáljuk ki!)

F: Az előző feladatot oldjuk meg közvetlen függvénydefinícióval (anonymus megadás) és `inline` megadással is!

Próbáljuk ki, hogy az `inline` és az anonymus megadásnál is elhagyható a pont a megfelelő műveletek előtt, ha az `fplot` utasítást használjuk.

Mintafeladat

Próbáljuk ki az $1,2x^2 \cdot e^{-0,5x}$ függvény különböző megadásait is a következők szerint.

```
>> f = @(x) 1.2*x^2*exp(-.5*x), fplot(f,[0 20])
% itt csak a függvénynév kell aposztrófok nélkül
>> fp = inline('1.2*x^2*exp(-.5*x)'); fplot(fp, [0 20])
% ekkor sem kell fp-t aposztrófok közé tenni!
>> fplot('f2', [0 20])
% a függvény máshol definiált (saját függvény), aposztrófok közé tesszük!
```

Feladat

Írjuk át az előző feladatban szereplő saját függvényt úgy, hogy az 1,2, a 2 és a -0,5 érték paraméterként legyen megadható! Ábrázoljuk így is a függvényt!

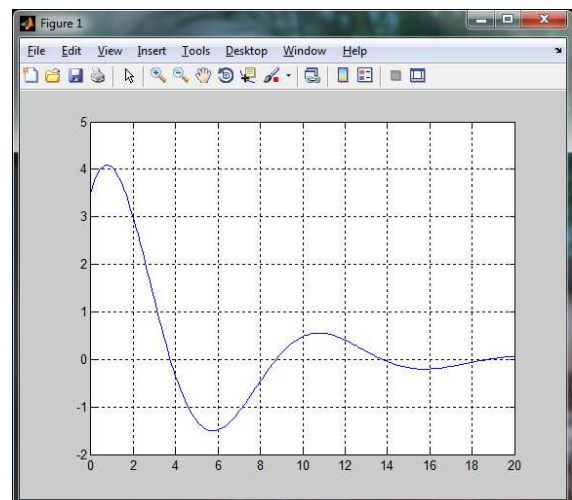
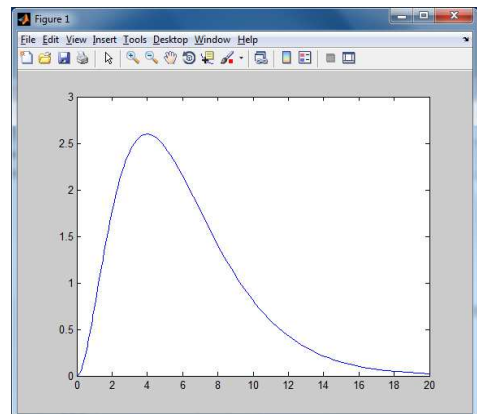
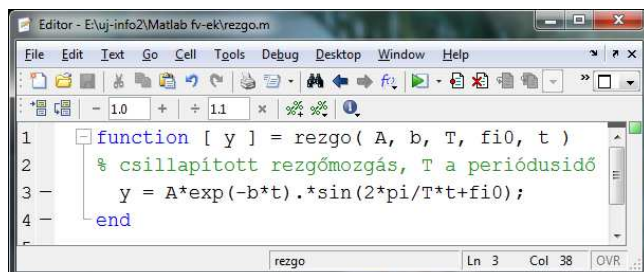
Pl. `x=0:0.1:20; plot(x, f1(1.2,2,-.5, x))`.

Feladat

Írjunk saját függvényt az
 $y = \text{amplitúdó} \cdot e^{-\text{csillapítás} \cdot t} \cdot \sin(2\pi/T \cdot t + \text{fázisszög})$
 szabállyal adott csillapított rezgőmozgást leíró
 képlet megvalósítására, majd ábrázoljuk a függvényünket!

A paramétereket a következők szerint válasszuk:
 amplitúdó – A (értéke: 5), csillapítás – b (értéke: 0,2),
 t – idő (változó vektor, előállítását lásd lent),
 T – periódusidő (értéke: 10), fázisszög – fi0 (értéke: $\pi/4$).
 A függvény és az m-fájl neve legyen „rezgo”.
 A hívás eszerint a következő:

```
>> A=5; b=0.2; T=10; fi0=pi/4;
>> t = linspace(0, 2*T, 181);
>> plot(t, rezgo(A,b,T,fi0,t));
>> grid on
```

**OTTHONI MUNKA****Feladat** (üres mátrix/vektor)

Érdekes, hogy a Matlabban többféle méretű üres vektor/mátrix is létrehozható.

Milyen méretű lesz az `A = [eye(3); 6:3]` mátrix?

Próbáljuk ki a következő előállításokat, és hasonlítsuk össze a létrejövő objektumok méreteit!

```
>> c = [10:0], isempty(c), c_meret = size(c), c_hossz = length(c)
>> d = [1:0]', isempty(d), d_meret = size(d), d_hossz = length(d)
>> e = [], isempty(e), e_meret = size(e), e_hossz = length(e)
% programozásnál az utóbbi fontos!
```

Feladat (sávmátrix)

Állítsunk elő a diag és a ones függvények használatával egy olyan 6×6 -os mátrixot, amelynek főátlója csupa 8-as, felső mellékátlójában minden elem 3-as és az alsó mellékátlóban 1-esek vannak! (Vigyázat: az első mellékátlókban csak 5-5 elem van!)

Feladat (sávmátrix)

Tekintsük a jegyzetben szereplő, ill. a gyakorlat könyvtárában adott tridiag.m fájlt, amely egy speciális sávmátrixot állít elő (főátló + első mellékátlók). Az .m fájl segítségével adjuk meg egyetlen Matlab-utasítással azt a 20×20 -as mátrixot, amelynek

- bal felső 10-szer 10-es blokkja olyan tridiagonális mátrix, amelynek főátlójában -2 -esek, két mellékátlójában pedig 1-esek állnak (tridiag[1, -2 , 1] típusú);
- jobb alsó 10-szer 10-es blokkja tridiag[2, -4 , 2] típusú;
- a többi eleme pedig 0.

Feladat (plot)

A hold utasítás nélkül is lehet több grafikont egy ábrára tenni. A szintaktika:
plot(x1, y1, string1, x2, y2, string2, ...), ahol a string1 pl. 'r' lehet. Próbáljuk ki!

Feladat (plot, fplot)

Próbáljuk ki, hogy a plot parancsnál megismert megjelenést módosító vezérlők (szín, vonalstílus) az fplotnál is működnek.

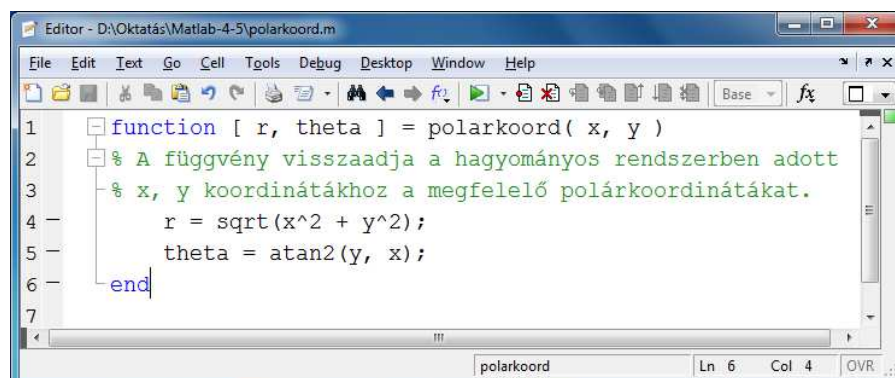
Feladat (egyszerű saját függvény, több visszatérési értékkel)

Készítsünk saját függvényt, amely paraméterként megkapja a kör sugarát, és kiírja a kerület és a terület értékét.

Módosítsuk a függvényünket úgy, hogy paraméterként egy ϕ szög is megadható legyen, és az adott szögű körcikkkel dolgozzon.

Feladat (saját függvény)

Tekintsük a következő példát.



Hívás:

```
>> [r, theta] = polarkoord(3, 4)
```

Ezen függvény mintájára készítsük el azt a hasonló függvényt, amelyik polárkoordinátákból készíti el az x és y koordinátákat.

Feladat (saját függvény, forgatómátrix)

A forgatómátrixos feladatot oldjuk meg úgy is, hogy saját függvényt készítünk a forgatás végrehajtására. Paraméterként legyen megadható a forgatandó pont x és y koordinátája, eredményként kapjuk meg az új koordinátákat! Az első sikeres megoldás után módosítsuk az eljárásunkat még azzal is, hogy az α érték is paraméterként megadható!

