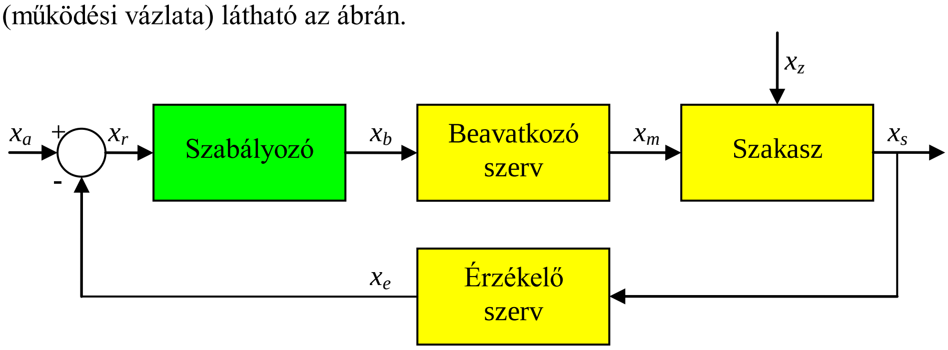
1. Gyakorlat

1. Tantermi gyakorlat – Jelek és rendszerek leírása

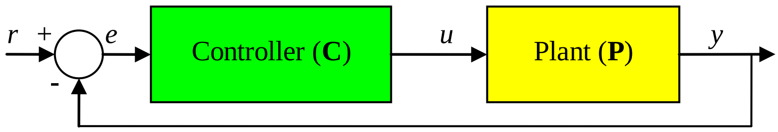
Az első tantermi gyakorlat célja, hogy a hallgatók átismételjék a *Jelek és Rendszerek* tárgyból a szabályozástechnikában a félév során közvetlenül szükséges ismereteket. A tantermi gyakorlat alkalmával tárgyalt feladatok lehetőséget adnak a Matlab használatának bemutatására. A Matlab és egyes kitüntetett szolgáltatásainak részletes ismertetését az *1. Számítógéptermi gyakorlat* tartalmazza.

A szabályozási kör

Egy egyhurkos (egy visszacsatolást tartalmazó) szabályozási kör elvi felépítése (működési vázlata) látható az ábrán.



A szabályozási kör elemei (a működési vázlat blokkjai) a bemenő jeleiken átalakítást végeznek, melynek eredménye az elem kimenő jele. A *Beavatkozó szerv*, a *Szakasz* és az *Érzékelő szerv* által definiált átalakításokat adottnak tekintjük, a *Szabályozó* által definiált átalakítást a szabályozási kör szintézise során határozzuk meg az elvárt működésre vonatkozó előírások alapján. A szabályozási kör működési vázlatát tömörebb alakra hozhatjuk, ha összevonjuk a *Beavatkozó szerv*, a *Szakasz* és az *Érzékelő szerv* által definiált átalakítást.



Bár a hatásvázlattal történő leírás lehetővé teszi tetszőleges átviteli tulajdonságokkal rendelkező tagok használatát, a félév során lineáris és időinvariáns rendszerekkel foglalkozunk. A *Jelek és rendszerek* tárgy részletesen ismertette az ilyen rendszerek leírásának módszereit folytonos és diszkrét időben egyaránt.

Laplace-transzformáció

Adott egy $f(t)$ függvény, ahol t az időt jelöli. Feltesszük, hogy $f(t)=0$, ha t negatív. Feltesszük továbbá, hogy létezik olyan σ valós szám, hogy $\int_0^{\infty} |f(t)| e^{-\sigma t} dt$ véges. Ekkor $f(t)$ Laplace-transzformáltja létezik, azt $F(s)$-sel jelöljük és az

$$F(s) = \int_0^{\infty} f(t) e^{-st} dt$$

képlettel definiáljuk. A Laplace-transzformáció lineáris operátor. Ha $f(t)$-nek létezik Fourier transzformáltja, akkor az a Laplace-transzformáltból az $s = j\omega$ helyettesítéssel adódik.

A Laplace transzformáció inverze az

$$f(t) = \frac{1}{2\pi j} \int_{c-j\omega}^{c+j\omega} F(s) e^{st} ds$$

képlettel számolható. Az alábbi felsorolás néhány hasznos összefüggést mutat be.

$$\mathcal{L}\{1(t)\} = \frac{1}{s} \qquad \mathcal{L}\{\delta(t)\} = 1 \qquad \mathcal{L}\{e^{\alpha t}\} = \frac{1}{s-\alpha}$$

$$\mathcal{L}\{1 - e^{-t/T}\} = \frac{1}{s(1+sT)} \qquad \mathcal{L}\{\sin \omega_0 t\} = \frac{\omega_0}{s^2+\omega_0^2} \qquad \mathcal{L}\{\cos \omega_0 t\} = \frac{s}{s^2+\omega_0^2}$$

$$\mathcal{L}\{t^n\} = \frac{n!}{s^{n+1}} \qquad \mathcal{L}\left\{\frac{df(t)}{dt}\right\} = sF(s) - f(0) \qquad \mathcal{L}\left\{\int_0^t f(\tau) d\tau\right\} = \frac{F(s)}{s}$$

$$\mathcal{L}\left\{\frac{d^n f(t)}{dt^n}\right\} = s^n F(s) - s^{n-1} f(0) - s^{n-2} f'(0) - \ldots - f^{(n-1)}(0)$$

$$\mathcal{L}\left\{\int_0^t f_1(\tau) f_2(t-\tau) d\tau\right\} = F_1(s) F_2(s) \qquad \lim_{t\to 0} f(t) = \lim_{s\to\infty} sF(s)$$

$$\lim_{t\to\infty} f(t) = \lim_{s\to 0} sF(s)$$

A határérték tételek közül a végérték tétel csak akkor használható, ha az $F(s)$ transzformált reguláris az s sík lezárt jobb félsíkján.

Az inverz transzformáció könnyen elvégezhető részlettörtekre bontással is, amennyiben a transzformált racionális törtfüggvény (számlálója és nevezője valós együtthatójú polinom s-ben). Tekintsük az

$$F(s) = \frac{4s+3}{s^2(s+1)(4s+1)(10s+1)}$$

Laplace-transzformáltat. A részlettörtekre bontás algoritmusát a Matlab tartalmazza. Ehhez meg kell adnunk a számláló és nevező polinomjait.

```
>> [r,p,k]=residue([4 3],conv(conv(conv([4 1],[10 1]),[1 1]),[1 0 0]) )

r =
   -0.0370
   -7.1111
   48.1481
  -41.0000
    3.0000

p =
   -1.0000
   -0.2500
   -0.1000
         0
         0

k =
     []
```

A `conv` függvény két polinom összeszorzására szolgál, amelyeket sorvektorokként kell megadni, hogy a vektor első eleme a polinom legnagyobb hatványához tartozó együttható legyen. Az eredmény értelmezése

$$F(s)=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}+\ldots+\frac{r_n}{s-p_n}+k(s)\,,$$

ha minden pólus egyszeres multiplicitású (azaz $p_i \neq p_j,\ j \neq i$). Ha egy pólus többszörös (például m) multiplicitású, azaz például $p_j = p_{j+1} = \ldots = p_{j+m-1}$, akkor ezekhez a pólusokhoz az

$$\frac{r_j}{s-p_j}+\frac{r_{j+1}}{(s-p_j)^2}+\ldots+\frac{r_{j+m-1}}{(s-p_j)^m}$$

részlettörtek tartoznak. Visszatérve a példához látható, hogy három egyszeres $\left(-\frac{1}{4},-\frac{1}{10},-1\right)$ és egy kétszeres multiplicitású pólusunk van (ez utóbbi a komplex számsík origójában), tehát a részlettörtekre bontás eredménye

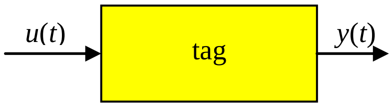
$$F(s)=\frac{4s+3}{s^2(s+1)(4s+1)(10s+1)}=-\frac{0.037}{s+1}-\frac{7.1111}{s+0.25}+\frac{48.1481}{s+0,1}-\frac{41}{s}+\frac{3}{s^2}$$

A részlettörteket a korábbi táblázat segítségével alakítjuk vissza az időtartományba.

$$f(t) = 3t - 41 - 0{,}037e^{-t} - 7{,}1111e^{-0{,}25t} + 48{,}1481e^{-0{,}1t}, \text{ ha } t \geq 0$$

Tagok leírása és jellemzőik

A folytonosidejű, lineáris és időinvariáns tagok leírását tárgyaljuk. A tag időben invariáns volta alatt itt a paraméterek időtől való függetlenségét értjük. A tag bemenetére adott időfüggvényt $u(t)$-vel, a kimenetén a bemenet hatására megjelenő időfüggvényt $y(t)$-vel jelöljük, Laplace-transzformáltjuk rendre $U(s)$ és $Y(s)$.



A tag leírható átviteli függvényével (*transfer function*)

$$W(s) = \frac{Y(s)}{U(s)} = \frac{b_0 s^m + b_1 s^{m-1} + \ldots + b_{m-1}s + b_m}{s^n + a_1 s^{n-1} + \ldots + a_{n-1}s + a_n}, \qquad n > m$$

A számláló gyökei az átviteli függvény (a tag) zérusai. A nevező gyökei az átviteli függvény (a tag) pólusai. A tag $w(t)$-vel jelölt impulzusválaszának (más néven súlyfüggvényének) Laplace-transzformáltja a tag $W(s)$ átviteli függvénye, hiszen $\mathscr{L}\{\delta(t)\} = 1$ és

$$Y(s) = W(s)U(s).$$

A tag $v(t)$-vel jelölt ugrásválasza (más néven átmeneti függvénye) az impulzusválasz integrálja, hiszen $\mathrm{L}\{1(t)\} = \frac{1}{s}$ és

$$V(s) = \frac{W(s)}{s}.$$

Az átviteli függvény az

$$y^{(n)} + a_1 y^{(n-1)} + \ldots + a_{n-1}\dot{y} + a_n y = b_0 u^{(m)} + b_1 u^{(m-1)} + \ldots + b_{m-1}\dot{u} + b_m u$$

lineáris differenciálegyenlet[1] Laplace-transzformáltja, ha a kezdeti feltételek nullák:

$$y^{(n-1)}(0) = y^{(n-2)}(0) = \ldots = \dot{y}(0) = y(0) = 0.$$

Egy tagot állapotegyenletével is jellemezhetünk, ez az ún. állapotteres (state-space) leírás:

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

[1] Egy függvény idő szerinti differenciálját ponttal is szokás jelölni.

Az állapotteres leírás „gazdagabb”, mint a tag átviteli függvénnyel történő leírása, mivel az ún. nem irányítható és nem megfigyelhető alrendszereket is tartalmazhatja. Ezek a fogalmak a félév során később kerülnek ismertetésre. Legyen $x \in R^n$ és a kezdeti állapot $x(0) = 0$. Ekkor az állapotegyenlet Laplace-transzformáltja

$$
\begin{aligned}
sX(s) &= AX(s) + BU(s) \\
X(s) &= (sI - A)^{-1} BU(s) \\
Y(s) &= [C(sI - A)^{-1} B + D]U(s) \\
W(s) &= C(sI - A)^{-1} B + D
\end{aligned}
$$

ahol I az n sorból és n oszlopból álló egységmátrix. Látható, hogy az átviteli függvény nevezője $\det(sI - A)$, amely pontosan az A mátrix karakterisztikus egyenlete, tehát az átviteli függvény pólusai az A mátrix sajátértékei.

Az átviteli függvény számlálója és nevezője valós együtthatójú polinom, melyek gyökei valós számok, illetve komplex konjugált párok lehetnek. A számlálót és a nevezőt szorzattá alakítva a

$$
W(s) = k \frac{\prod_{i=1}^{m} (s - z_i)}{\prod_{j=1}^{n} (s - p_j)}
$$

gyöktényezős alakot kapjuk.

A Matlab Control Systems Toolbox segítségével a lineáris tagok mindhárom alakban (átviteli függvény: transfer function (`tf`), állapotegyenlet: state-space (`ss`), gyöktényezős alak: zero-pole-gain (`zpk`)) vizsgálhatóak. A Control System Toolbox mindhárom reprezentációhoz egy-egy adatstruktúrát rendel, melyek létrehozása rendre a `tf`, `ss` és `zpk` utasításokkal lehetséges.

Az alábbi táblázat mutatja az adatstruktúrák mezőit. A teljesség kedvéért a diszkrét idejű reprezentációk mezőit is felsoroltuk.

Időtartomány típusa	**Reprezentáció**	**Tárolt struktúra mezői**
Folytonos	Átviteli függvény	`num` – számláló(k) cellatömbje `den` – nevező(k) cellatömbje
Folytonos	Zérus – Pólus – Erősítés	`z` – zérusok cellatömbje `p` – pólusok cellatömbje `k` – erősítések mátrixa
Folytonos	Állapotegyenlet	`a` – állapotmátrix `b` – bemeneti erősítés mátrixa `c` – kimeneti erősítés mátrixa `d` – közvetlen erősítések mátrixa
Diszkrét	Átviteli függvény	`num` – számláló(k) cellatömbje

		`den` – nevező(k) cellatömbje `Ts` – mintavételi periódusidő
Diszkrét	Zérus – Pólus – Erősítés	`z` – zérusok cellatömbje `p` – pólusok cellatömbje `k` – erősítések mátrixa `Ts` – mintavételi periódusidő
Diszkrét	Állapotegyenlet	`a` – állapotmátrix `b` – bemeneti erősítés mátrixa `c` – kimeneti erősítés mátrixa `d` – közvetlen erősítések mátrixa `Ts` – mintavételi periódusidő

A vizsgálat frekvencia és időtartományban is lehetséges.

Tartomány	Módszer	Matlab utasítás	Magyarázat
Frekvencia	Bode-diagram	`bode`	A pozitív képzetes féltengely képe az átviteli függvény alatt. A diagram egy amplitúdó-jelleggörbéből és egy fázis-jelleggörbéből áll.
Frekvencia	Nyquist-diagram	`nyquist`	A képzetes tengely képe az átviteli függvény alatt a komplex számsíkon. A kép mindig tengelyszimmetrikus a számsík valós tengelyére.
Frekvencia	pólus-zérus eloszlás	`pzmap`	Az átviteli függvény pólusai és zérusai a komplex számsíkon
Idő	impulzusválasz	`impulse`	A tag kimenetének időfüggvénye ha bemenetére a $\delta(t)$ időfüggvény kerül (nulla kezdeti feltételek mellett).
Idő	Ugrásválasz	`step`	A tag kimenetének időfüggvénye ha bemenetére az $1(t)$ időfüggvény kerül (nulla kezdeti feltételek mellett).

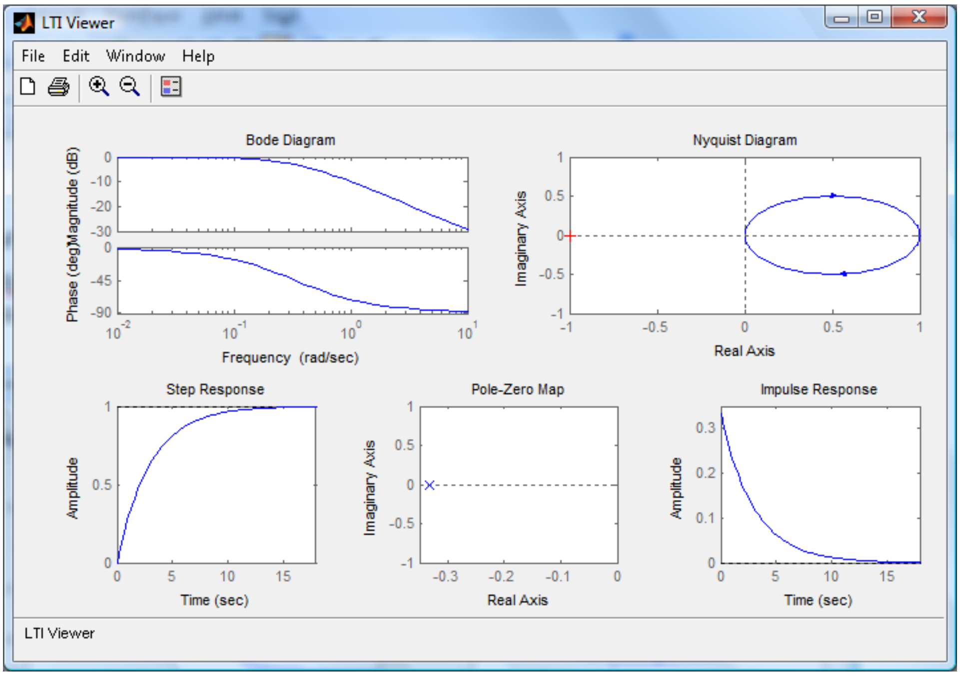
Bármely átviteli függvény előállítható egytárlós és kéttárolós tagok párhuzamos vagy soros kapcsolásával.

Az egytárolós tag

Az egytárolós tag átviteli függvénye:

$$W(s) = \frac{1}{1+Ts}$$

A tag egyetlen pozitív értékkészletű paramétere a T időállandó. A tagnak nincs zérusa, és egyetlen pólusa $-\frac{1}{T}$. Az egytárolós tag Bode-diagramját, Nyquist-görbéjét, ugrásválaszát, pólus-zérus képét és impulzusválaszát mutatja az alábbi ábra $T = 3\text{sec}$ választás mellett.



Az egytárolós tag $v(t)$ ugrásválaszát és annak végértékét, illetve $w(t)$ impulzusválaszát egyszerűen számíthatjuk a korábbi táblázat alapján:

$$v(t) = \mathcal{L}^{-1}\left\{\frac{1}{s(Ts+1)}\right\} = 1 - e^{-t/T}, \quad t \geq 0$$

$$v(\infty) = 1$$

$$w(t) = \frac{1}{T}e^{-t/T}, \quad t \geq 0$$

A Bode-diagram amplitúdó- és fázis-jelleggörbéje szintén egyszerűen meghatározható.

$$a_{dB}(\omega) = 20\lg|W(j\omega)| = 20\lg\frac{1}{\sqrt{1+(\omega T)^2}} = -20\lg\sqrt{1+(\omega T)^2}$$

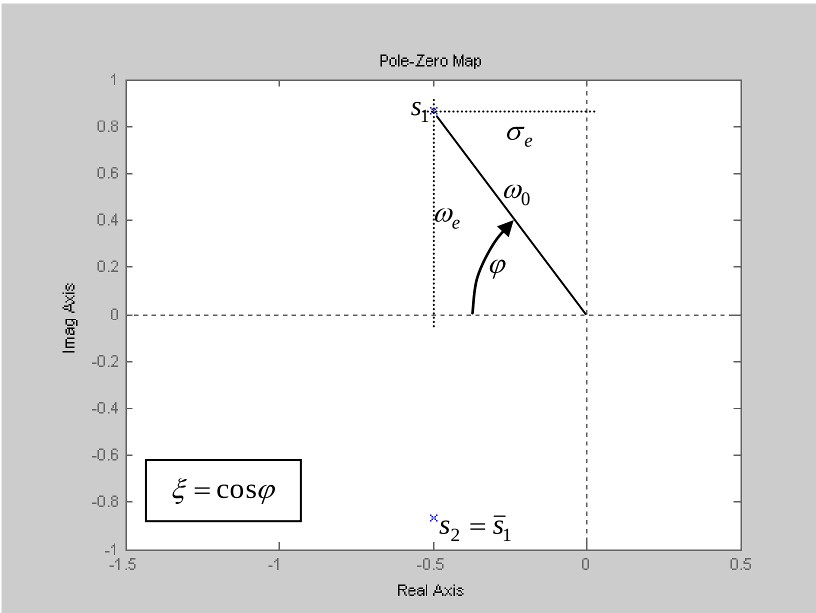
$$\varphi(\omega) = \arg W(j\omega) = -\arctan(\omega T)$$

A kéttárolós lengőtag

A kéttárolós lengőtag átviteli függvénye:

$$W(s) = \frac{1}{1 + 2\xi T s + T^2 s^2} = \frac{\omega_0^2}{\omega_0^2 + 2\xi\omega_0 s + s^2} \qquad \omega_0 = \frac{1}{T}$$

A tag két pozitív értékkészletű paramétere a T időállandó (ω_0 csillapítatlan sajátfrekvencia) és a ξ csillapítási tényező. Az időállandó és a csillapítási tényező értékkészlete garantálja, hogy a nevező gyökei nem esnek a komplex számsík jobb félsíkjára. A karakterisztikus egyenlet diszkriminánsa negatív, ha $0 \le \xi < 1$, ekkor a karakterisztikus egyenletnek a gyökei egymás komplex konjugált párjai: $s_{1,2} = -\omega_0\xi \pm j\omega_0\sqrt{1-\xi^2}$ ($0 \le \xi < 1$). Bevezethető $\sigma_e := \omega_0\xi$ és $\omega_e := \omega_0\sqrt{1-\xi^2}$, így a pólusok kifejezhetők $s_{1,2} = -\sigma_e \pm j\omega_e$ alakban is. A konjugált póluspár komplex síkbeli elhelyezkedését és a paraméterek viszonyát mutatja az ábra.

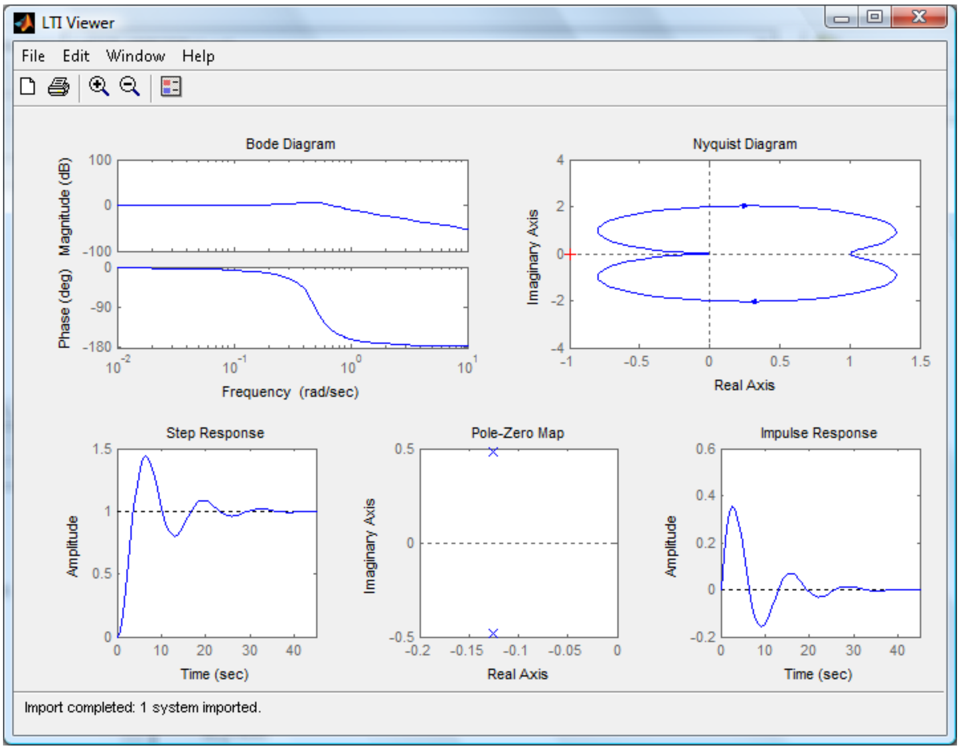


Az ugrásválasz és az impulzusválasz időfüggvényei az alábbiak

$$v(t) = 1 - \frac{e^{-\xi\omega_0 t}}{\sqrt{1-\xi^2}} \sin(\omega_0\sqrt{1-\xi^2}\,t + \arccos\xi), \quad t \ge 0$$

$$w(t) = -\frac{\omega_0}{\omega_e} e^{-\sigma_e t}(-\sigma_e \sin(\omega_e t + \varphi) + \omega_e \cos(\omega_e t + \varphi)), \quad t \ge 0$$

Az kéttárolós tag Bode-diagramját, Nyquist-görbéjét, $v(t)$ ugrásválaszát, pólus-zérus képét és $w(t)$ impulzusválaszát mutatja a következő ábra $\xi = 0.25$ és $T = 2\sec$ választás mellett.



A Bode-diagram amplitúdó- és fázis-jelleggörbéje:

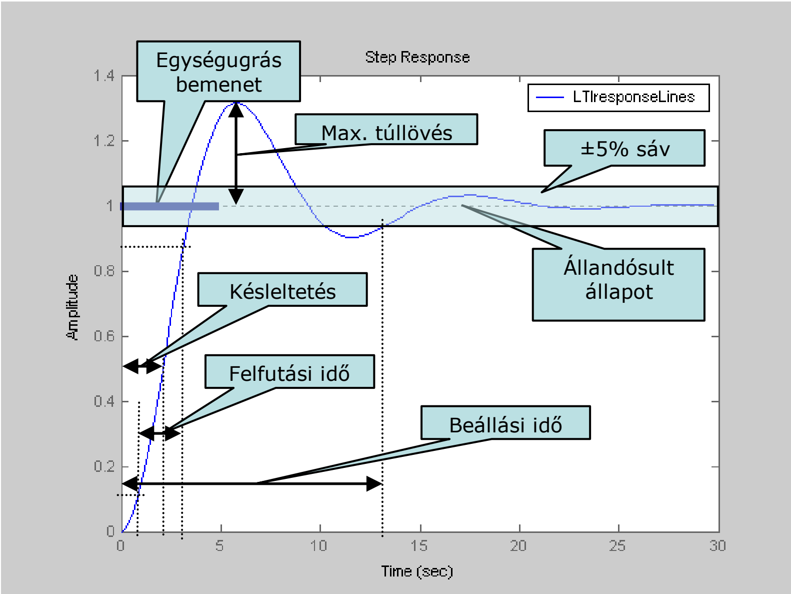
$$a_{dB}(\omega) = -20\lg\sqrt{(1-\omega^2T^2)^2 + 4\xi^2T^2\omega^2}$$

$$\varphi(\omega) = -\arctan\frac{2\xi T\omega}{1-\omega^2T^2}$$

A kéttárolós tag $v(t)$ ugrásválaszát külön is megvizsgáljuk, hogy szemléltessünk néhány időtartománybeli jellemzőt. Az ugrásválasz első maximumához tartozó T_m idő meghatározásához kiszámítjuk annak idő szinti deriváltját és annak első zérushelyét.

$$\frac{d}{dt}v(t) = w(t) = 0 \Rightarrow \frac{\omega_0}{\sqrt{1-\xi^2}}e^{-\zeta\omega_0 t}\sin\omega_0\sqrt{1-\xi^2}t = 0$$

$$T_k = \frac{k\pi}{\omega_0\sqrt{1-\xi^2}} \Rightarrow T_m = \frac{\pi}{\omega_0\sqrt{1-\xi^2}}$$



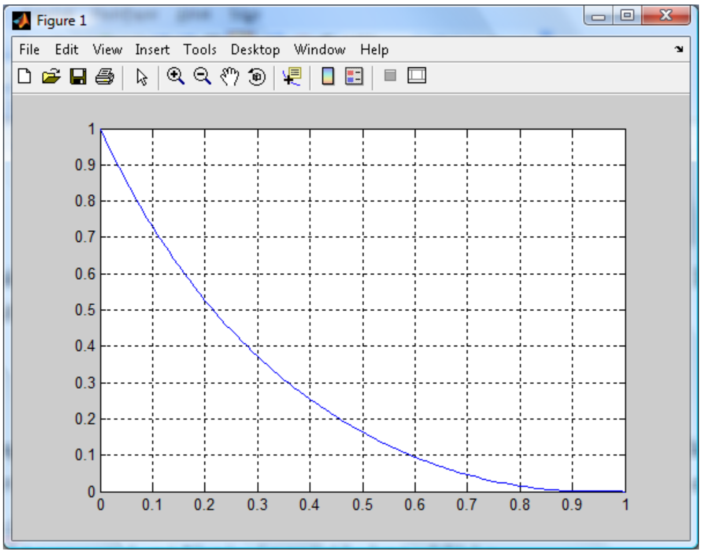
Tehát $n=1$ esetén T_m az első maximum eléréséhez szükséges idő. Innen kiszámítható a maximális túllövés értéke:

$$\Delta v = v(T_m) - 1 = \exp\left(-\frac{\pi\xi}{\sqrt{1-\xi^2}}\right)$$

A túllövés értéke a [0,1] tartományba esik, százalékos megadása is szokásos. Vizsgáljuk meg, hogy a túllövés hogyan függ a ξ csillapítási tényezőtől. Ehhez a $\xi \to \Delta v(\xi)$ hozzárendelést kell ábrázolni. Ezt a Matlab segítségével végezzük el.

```
>> ksi = 0:0.01:1;
>> Deltav = exp(-pi*ksi./sqrt(1-ksi.^2));
Warning: Divide by zero.
>> plot(ksi,Deltav); grid on;
```

A művelet közben a Matlab figyelmeztet, hogy nullával történő osztás történt, ami a $\xi = 1$ értéknek felel meg. Ebben az esetben az átviteli függvénynek egy kétszeres multiplicitású valós pólusa van, tehát nincsen túllövés (aperiodikus határeset). A `Deltav` vektor számítására szolgáló utasításban a pont (`.`) operátorok lehetővé teszik a műveletek elemenként történő elvégzését.



Leolvasható, hogy a legnagyobb, 100%-os túllövés a $\xi = 0$ érték esetében következik be, ez csillapítatlanul lengő mozgásnak felel meg (két tisztán képzetes pólus), majd a túllövés monoton módon csökken.

A kéttárolós rendszer esetében az ugrásválasz exponenciális burkológörbéjének segítségével számítható ki, hogy a válasz mikor kerül a végérték körüli α %-os sávba.

$$\exp(-\sigma_e T_{\alpha\%}) = \frac{\alpha}{100} \qquad T_{\alpha\%} = \frac{\ln\frac{100}{\alpha}}{\sigma_e}$$

Speciálisan

$$T_{2\%} = \ln\frac{50}{\sigma_e} \approx \frac{4}{\sigma_e} \qquad T_{5\%} = \frac{\ln 20}{\sigma_e} \approx \frac{3}{\sigma_e}$$

Nemlineáris rendszerek linearizálása

Nemlineáris rendszer szabályozása is lehetséges lineáris technikákkal, amennyiben annak viselkedése abban a tartományban, amelyben a szabályozás során működik, jól közelíthető egy lineáris rendszer viselkedésével. Nemlineáris rendszerek értéktartó szabályozását lineáris technikákkal egy munkapont környezetében végezhetjük. A szabályozási kör szintézisekor ekkor a nemlineáris rendszer munkapont körüli linearizált modelljét használhatjuk.

Adott az

$$\dot{x} = f(x,u) \qquad x \in R^n;\ u \in R^r$$
$$y = h(x,u) \qquad y \in R^m$$

nemlineáris állapotegyenlettel leírt rendszer. Tegyük fel, hogy az (x_0, u_0, y_0) a rendszer egy egyensúlyi állapota, bemenete és kimenete, azaz teljesül, hogy

$$0 = f(x_0, u_0)$$
$$y_0 = h(x_0, u_0)$$

Vezessük be a $\delta x = x - x_0$, $\delta u = u - u_0$, $\delta y = y - y_0$ munkaponthoz képesti elmozdulásokat és az *f* és *h* függvények Taylor-sorai alapján a

$$A = Df_x\big|_{x_0,u_0} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{x_0,u_0} \qquad B = Df_u\big|_{x_0,u_0} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_r} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_r} \end{bmatrix}_{x_0,u_0}$$

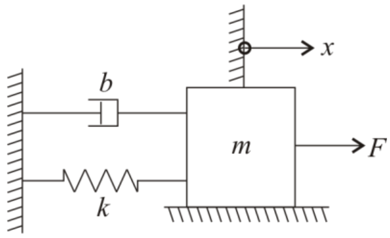
$$C = Dh_x\big|_{x_0,u_0} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial h_m}{\partial x_1} & \cdots & \frac{\partial h_m}{\partial x_n} \end{bmatrix}_{x_0,u_0} \qquad D = Dh_u\big|_{x_0,u_0} = \begin{bmatrix} \frac{\partial h_1}{\partial u_1} & \cdots & \frac{\partial h_1}{\partial u_r} \\ \vdots & & \vdots \\ \frac{\partial h_m}{\partial u_1} & \cdots & \frac{\partial h_m}{\partial u_r} \end{bmatrix}_{x_0,u_0}$$

mátrixokat, amelyek a linearizált rendszer állapotegyenletének mátrixai.

Példarendszerek

Mechanikai lengőrendszer

Egy egyszerű, egy-szabadságfokú lengőrendszert mutat az alábbi ábra.



Az $m = 2\,\text{kg}$ tömeget az F erő gyorsítja pozitív irányba. A tömeget egy $k = 0.5\,\text{N/m}$ keménységű rugóval és egy $b = 0.25\,\text{Ns/m}$ csillapítású elemmel rögzítettük a falhoz. A rugó nyugalmi helyzetében a tömeg x irányú kitérése 0. A feladat az egyszerű rendszer differenciálegyenletének, állapotegyenletének, illetve az F erő és a tömeg x kitérése közötti átviteli függvénynek felírása. Feladat továbbá a kapott átviteli függvény paramétereinek meghatározása. A rendszer differenciálegyenlete a Newton-törvényből adódik:

$$m\ddot{x} = F - kx - b\dot{x}$$

A differenciálegyenelet másodfokú, így a rendszernek két állapota van ($x_1 = x,\ x_2 = \dot{x}$), a bemenet $u = F$, a kimenet $y = x_1$.

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -\frac{k}{m}x_1 - \frac{b}{m}x_2 + \frac{1}{m}u \qquad \dot{x} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

$$y = x_1$$

Az átviteli függvény a differenciálegyenlet vagy az állapotegyenlet Lapalce-transzformálásával is előállítható.

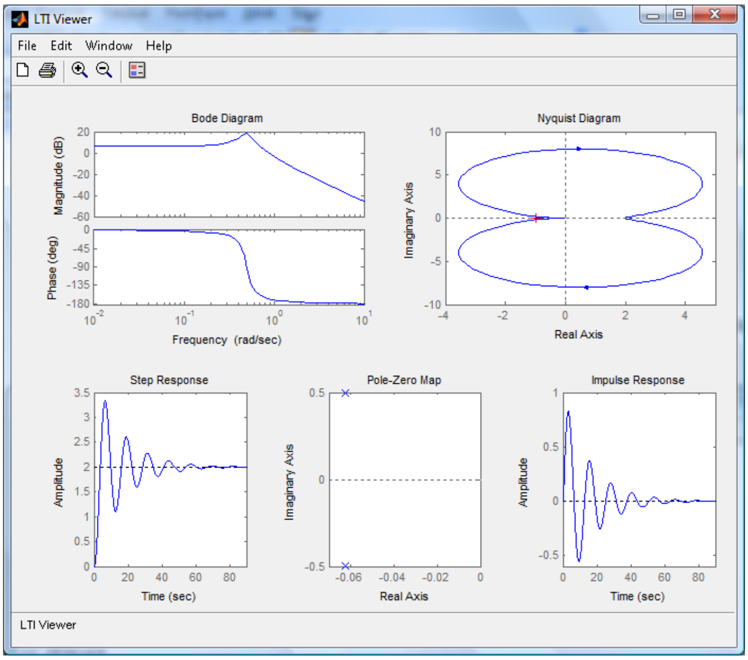
$$ms^2Y(s) = U(s) - kY(s) - bsY(s)$$

$$W(s) = \frac{Y(s)}{U(s)} = \frac{1}{ms^2 + bs + k} = \frac{1}{k} \cdot \frac{1}{1 + \frac{b}{k}s + \frac{m}{k}s^2}$$

Látható, hogy ez egy kéttárolós lengőtag. A tag erősítése $1/k$. Egyszerű összehasonlítással meghatározható a csillapítás és az időállandó.

$$T = \sqrt{\frac{m}{k}} \qquad 2\xi T = 2\xi\sqrt{\frac{m}{k}} = \frac{b}{k} \qquad \xi = \frac{b}{2\sqrt{km}}$$

Számszerűen, a példa adataival $T = 2\,\mathrm{sec}$, $\xi = 0.125$. A jellemzőket az ábra mutatja.

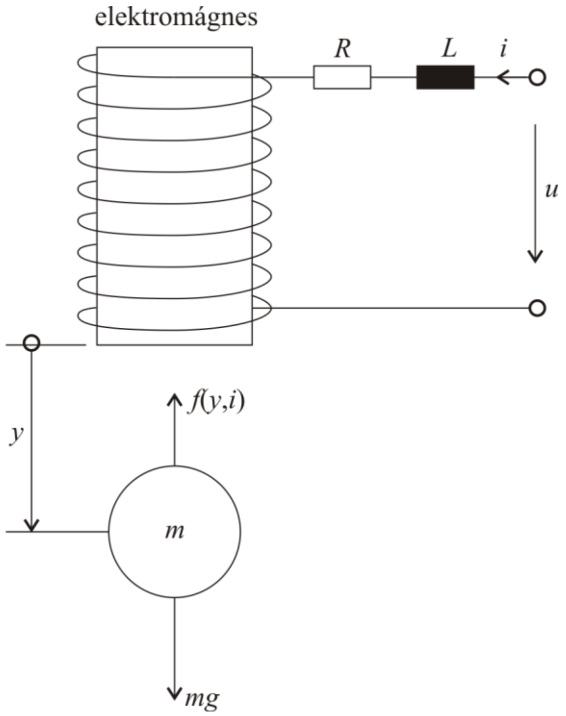


A csillapítás segítségével meghatározható a túllövés és az ugrásválasz első maximumának eléréséhez szükséges idő

$$T_m = 6.3329\text{sec} \qquad \Delta v = 0.6731.$$

Mágneses lebegtetés

Tekintsünk egy egyváltozós nemlineáris rendszert, egy mágnesesen lebegtetett golyót. Az ábrán a lebegtető rendszer sematikus szerkezeti vázlata látható.



Az itt ismertetett rendszer modellje hasonló a mágneses felfüggesztések (például mágneses úton lebegtetett vasutak) számára felállítható, persze akkor bonyolultabb különféle modellekhez.

A elektromágnes tekercsére kapcsolt feszültség a golyóra ható mágneses erőt befolyásolja, lehetővé téve annak függőleges gyorsítását. A rendszer dinamikus modelljét $L = \dfrac{Q}{y} + L_0$ esetén az

$$m\ddot{y} = mg + \frac{i^2}{2}\frac{dL}{dy} = mg + \frac{i^2}{2}\left(-\frac{Q}{y^2}\right) \Rightarrow \ddot{y} = g - \frac{Qi^2}{2my^2}$$

$$u = Ri + \frac{d}{dt}(Li) = Ri + \left\{-\frac{Q}{y^2}\frac{dy}{dt}i + \left(\frac{Q}{y} + L_0\right)\frac{di}{dt}\right\} \Rightarrow \frac{di}{dt} = \left(\frac{Q}{y} + L_0\right)^{-1}\left(u - Ri + \frac{Q\dot{y}i}{y^2}\right)$$

differenciálegyenletek definiálják. A golyó tömege $m = 0.8$ kg. A tekercs induktivitása $L_0 = 0.5$ H, ellenállása $R = 10\Omega$. A légréshez tartozó induktivtásváltozás $Q = 0.001$ Hm. A nehézségi gyorsulás $g = 9.8$ kgm/s^2.

A lebegtetett golyó függőleges pozícióját y jelöli, i az elektromágnes tekercsében folyó áram, u pedig a tekercs kapcsaira kapcsolt feszültség. A modell kimenete a golyó pozíciója.

Első feladat az állapotegyenlet felírása. A rendszert egy másodrendű és egy elsőrendű differenciálegyenlet írja le, így három állapota van. Az állapotok $x = (x_1, x_2, x_3)^T = (y, \dot{y}, i)^T$. Innen az állapotegyenlet

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = g - \frac{Qx_3^2}{2mx_1^2}$$

$$\dot{x}_3 = \left(\frac{Q}{x_1} + L_0\right)^{-1}\left(u - Rx_3 + \frac{Qx_2x_3}{x_1^2}\right)$$

$$y = x_1$$

Munkapontnak egy adott magasságban a mozdulatlan golyóhoz tartozó egyensúlyi értékeket választjuk. Egyensúly esetén az állapotok nem változnak, $x_2 = \dot{y} = 0$. Legyen az egyensúlyi magasság $x_{10} = y_0 = 0.01$m. A golyónak ebben a magasságban történő megtartásához az elektromágnes tekercsében folyó áram egyszerűen meghatározható a második egyenletből

$$x_{30} = i_0 = \sqrt{\frac{2mgy_0^2}{Q}} = 1.2522 \text{ A}$$

az ehhez tartozó feszültség pedig a harmadikból

$$u_0 = Ri_0 = 12.522\text{V}$$

A parciális deriváltakat tartalmazó mátrixok meghatározása és a munkapontnak megfelelő értékek behelyettesítése utána a következő eredményeket kapjuk:

$$A=\begin{bmatrix} 0 & 1 & 0 \\ \dfrac{Qx_3^2}{mx_1^{-3}} & 0 & -\dfrac{Qx_3}{mx_1^2} \\ a_{31} & \dfrac{Qx_1x_3}{x_1^2(Q+L_0x_1)} & \dfrac{Qx_2-Rx_1^2}{x_1(Q+L_0x_1)} \end{bmatrix}_{x_0,u_0}=\begin{bmatrix} 0 & 1 & 0 \\ 1960 & 0 & \text{-15.65} \\ 0 & 20.87 & \text{-16.66} \end{bmatrix}$$

$$B=\begin{bmatrix} 0 \\ 0 \\ \dfrac{x_1}{Q+L_0x_1} \end{bmatrix}_{x_0,u_0}=\begin{bmatrix} 0 \\ 0 \\ 1{,}6667 \end{bmatrix} \qquad C=\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

ahol

$$a_{31}=\frac{(u-Rx_3)(Q+L_0x_1)-L_0x_1(u-Rx_3)}{(Q+L_0x_1)^2}+\frac{Qx_1^2x_2x_3(Q+L_0x_1)-Q^2x_1^2x_2x_3(2+3L_0x_1)}{(Qx_1^2+L_0x_1^3)}$$

Ezzel megkaptuk a linearizált rendszer állapotegyenletét. Egy nemlineáris rendszer adott munkapont körüli linearizált állapotegyenletének meghatározását a Matlab Simulink eszköze és a `linmod` utasítás is támogatja. Az utasítás használatával az első számítógéptermi gyakorlat során ismerkedünk meg.

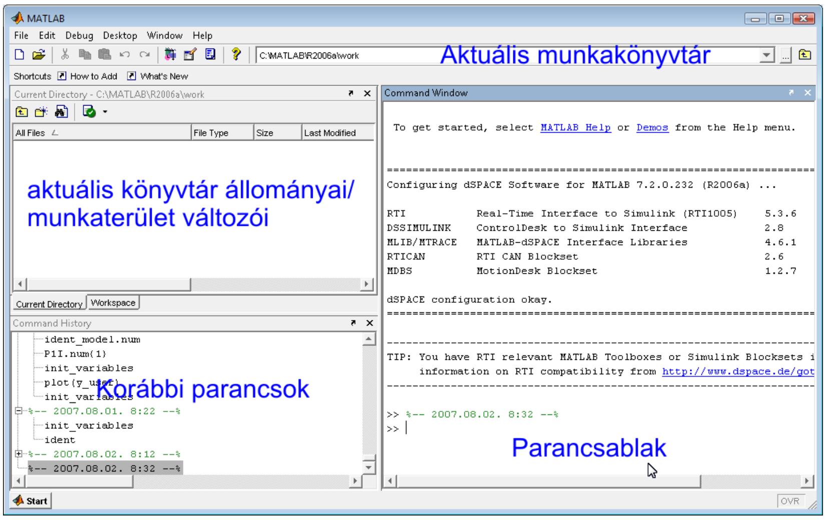
1. Számítógépes gyakorlat – A Matlab használata

Az 1. Számítógéptermi gyakorlat célja, hogy a hallgatók készségszinten elmélyítsék és tovább gyarapítsák a Matlab környezet használatához szükséges ismereteiket. A gyakorlat során elsajátított ismeretek a további (tantermi és számítógéptermi) gyakorlatok mindegyikéhez szükségesek. A Matlab programmal a hallgatók korábban már találkoztak a *Jelek és Rendszerek* tárgy keretein belül, szintén gyakorlati foglalkozásokon. Ennek dacára a gyakorlat néhány alapfogalom ismétlésével kezdődik.

Matlab alapok (ismétlés)

A Matlab egy interaktív környezet tudományos és mérnöki számítások elvégzésére valamint dinamikus rendszerek szimulációinak futtatásához. A környezet támogatja az adatsorok két és háromdimenziós megjelenítését is. A Matlab egyik kiegészítő bővítménye a Simulink, amely dinamikus rendszerek grafikus eszközökkel támogatott létrehozását és szimulációját teszi lehetővé. A Matlab moduláris felépítésének (toolboxok) köszönhetően tetszőlegesen bővíthető. A legnépszerűbb kiegészítő eszközök közé a Simulink mellett a szabályozástechnikai számításokat segítő Control System Toolbox és a szimbolikus számításokat támogató (Extended) Symbolic Toolbox csomagok tartoznak. A Matlab, a Simulink és a Handle Graphics a MathWorks Inc. cég bejegyzett védjegyei.

A Matlab a *Matrix Laboratory* elnevezésből származik, és azért született, hogy könnyű és intelligens hozzáférést biztosítson a korábban LINPACK és EISPACK néven kifejlesztett szoftverekhez, amelyek a mátrix számítási (lineáris algebrai) szoftverek legfejlettebb formáinak tekinthetők. A Matlab a 6-os verziótól kezdve alapesetben az alábbi felépítésű felülettel indul.



A Matlab számítógéptermi verziója ettől kismértékben eltérhet. A felület főbb elemei az alábbiak:

1. Parancsablak: a Matlab utasítások megadására, a Matlab szkriptek és függvények meghívására szolgál. A parancsokat a parancs prompt (>>) sorába írjuk.
2. Aktuális könyvtár: az aktuális könyvtár állományainak megjelenítése (a megfelelő állományok közvetlenül megnyithatóak szerkesztésre, elindíthatóak, törölhetőek, stb.)
3. Munkaterület változói: a munkaterületen létrehozott változók listája (típus, dimenzió). A változók értéke az erre szolgáló külön felületen szerkeszthető.
4. Korábbi parancsok: munkamenetenként csoportosítva hozzáférhetőek a korábban kiadott parancsok. Ezek között egyébként a parancsablakban a fel és le gomb segítségével is navigálhatunk.

A felület természetesen testre szabható. Ehhez a `Desktop` menü parancsai, illetve a `File` menü `Preferences` dialógusablaka használható.

A Matlab rendszer magja egy mátrix-műveletekre közvetlenül felkészített parancsértelmező. A műveletek alapeleme tehát a kétdimenziós (téglalap-szerű) mátrix. A mátrix speciális esete a skalár (egyelemű mátrix) és a sor- vagy oszlopvektor (egy sorból, vagy egy oszlopból álló mátrix). A mátrix elemei automatikusan 8 bájt széles lebegőpontos módban (*double*) ábrázolt értékek. Az elemek IEEE szabvány szerinti számábrázolást feltételeznek, ami lehetővé teszi, hogy a valós számok mellett 3 kitüntetett kód is ábrázolható legyen: *NaN*, *inf*, -*inf*. Ezek rendre a definiálatlan értéket (*NaN*, not a number), valamint a pozitív (*inf*) és negatív (-*inf*) végtelent jelölik. Ezekkel bizonyos korlátozások mellett műveletek is végezhetők. A mátrix elemeinek indexelése a soron és oszlopon belül 1-től kezdődik. Definiálható az üres mátrix is, amelynek jelölése `[]`.

A mátrixok esetében definiált műveletek, mint az összeadás, kivonás, szorzás, transzponálás (ez azonban komplex elemű mátrix esetén a konjugált komplex transzponáltja), közvetlenül elvégezhetők. Kétféle osztás áll rendelkezésre: balosztás (`X=A\B` az `AX=B` megoldása), jobbosztás (`X=B/A` az `XA=B` megoldása). Hasonlóképpen végezhető el a hatványozás (ha `p`>1 egész, akkor `A^p` ismételt szorzással képződik, különben `A^p=V*D.^p/V`, ahol `[V,D]=eig(A)`), az elemenkénti szorzás a `.` (pont) operátor segítségével (`X=A.*B`), az elemenkénti osztás (`X=A./B`) és az elemenkénti hatványozás (`X=A.^B`).

A standard függvények (`sin, cos, sqrt, exp, log, log10` stb.) és elemi matematikai függvények (`fix, floor, ceil, round, rem, real, imag, conj, abs, angle, sign` stb.) elemenként hajtódnak végre, más standard függvények a mátrix oszlopain hajtódnak végre (pl. `max` illetve `min` egy sorvektorral tér vissza, amelynek elemei az oszlopokhoz tartozó maximális illetve minimális elemet tartalmazzák). Vannak azonban mátrixfüggvények is, amelyeknek a neve '`m`' betűre végződik (`expm, logm, sqrtm`), ezek argumentuma és értéke mátrix.

Vannak speciális mátrixok, így pl. az egységmátrix, a csak nulla elemeket tartalmazó mátrix és a tiszta 1-elemű mátrix. Ezek megadására rendre az `eye(n)`, a `zero(n,m)` és a `ones(n,m)` szolgál, ahol `n` és `m` a sorok és az oszlopok száma. Generálhatók véletlenszám-elemű mátrixok egyenletes vagy normális eloszlás szerint a `rand(n,m)` és `randn(n,m)` függvényekkel. A véletlenszám generátor definit kezdőértékre állítható be, ami a programfejlesztéseknél hasznos lehet, pl. `randn('seed',0)` alaphelyzetbe állítja a normális eloszlás szerinti véletlenszám generátort.

A függvényeket újabb függvényekbe és programokba szervezhetjük. A C nyelvhez hasonlóan használhatók feltételes utasítások (`if, else`) és szervezhetők ciklusok (`for, while`). Léteznek relációs és logikai függvények (`any, all, find, exist, isnan, isfinite, isempty, isstr, strcmp`).

Utasítások <script>`.m` fájlba foglalhatók, és a <script> parancs hatására végrehajthatók. Kialakíthatók a szokásos módon függvények <function>`.m` fájlban a `function` alapszó után, és meghívhatók a <function> névvel paramétereket átadva neki. A függvények és szkriptek fejlesztése a Matlab saját szöveges szerkesztője segítségével célszerű, ahol a hibakeresést és javítást segítő szokásos eszközök rendelkezésre állnak.

A tipikus lineáris algebrai és jelfeldolgozási feladatok megoldását professzionális (azaz gyors és numerikusan robusztus) algoritmusokat használó függvények támogatják, így `inv(A)` a mátrix invertálást, a balosztás és/vagy jobbosztás a lineáris egyenletrendszer megoldását, `[V,D]=eig(A)` a sajátvektor/sajátérték számítást, `pinv(A)` a Moore-Penrose pszeudoinverz és `[U,S,V]=svd(A)` a szinguláris érték felbontás számítását végzi el.

Az parancsértelmező használata

A Matlab elindításakor a parancsértelmező ablakban az alábbi üzenetet jelenik meg.

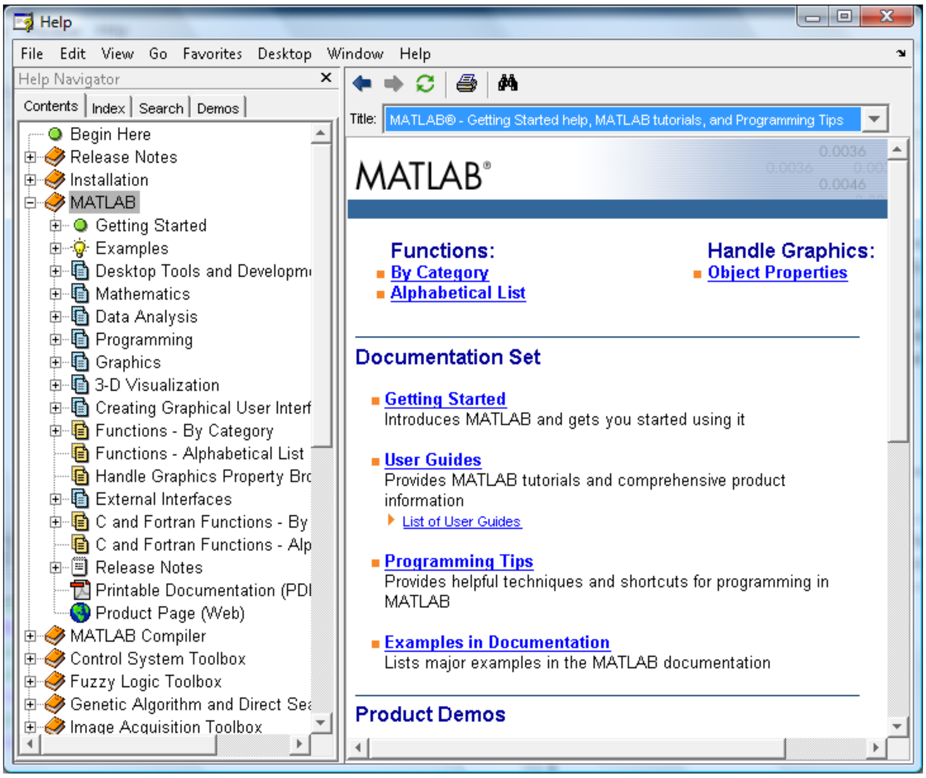
```
To get started, select MATLAB Help or Demos from the Help
menu.

>>
```

A szövegben `MATLAB Help` és a `Demos` szavak aktív linkek. Itt említjük meg, hogy a Matlab igen jól szervezett és részletes súgóval rendelkezik, amely egyszerre több lehetőséget is biztosít valamely információhoz történő hozzáféréshez. Rendelkezésre áll egy html típusú súgórendszer a hozzá tartozó böngészővel, amely a Matlab és az összes telepített kiegészítő toolbox kimerítő leírását tartalmazza. A legtöbb utasításhoz, adott típusú műveletet végző függvények csoportjához és a toolboxokhoz is interaktív bemutatók (demo-k) tartoznak. A súgó felületét az ábra mutatja.

Ugyanakkor – a Matlab korábbi verzióiból öröklött módon – rendelkezésre áll a `help` parancs. Bármely utasításról a parancsablakban szöveges információt nyerhetünk, ha begépeljük:

```
>>help utasitás_név
```

Így még a `help` parancs használatáról is kaphatunk eligazítást. A `help` parancs mellett rendelkezésre áll a `doc` parancs is, amely mögé szintén az utasítás nevét kell írnunk egy szóköz után. Ez utóbbi hatására a súgónak az adott utasításra vonatkozó fejezete nyílik meg.



A parancsablakban végrehajthatunk egyszerű matematikai műveleteket:

```
>>sqrt(4*9)

ans =

     6
```

Az `sqrt(x)` függvény `x` négyzetgyökét számítja. Az `ans` (`ans`wer) a Matlab egy speciális, beépített változója, értéke mindig a legutolsó parancs végrehajtásakor keletkezett eredmény, ha a legutolsó parancs nem tartalmazott értékadást. A Matlab egyéb, fontosabb beépített változói közül csak az alábbiakat soroljuk fel:

`pi=3.14159265358979`
`j=i=sqrt(-1)`
`eps=2.220446049250313e-16` - a numerikus felbontás finomsága 1.0 közelében.

Az `i` és `j` változók definíciójából látszik, hogy a Matlab komplex számokat is képes kezelni. Például:

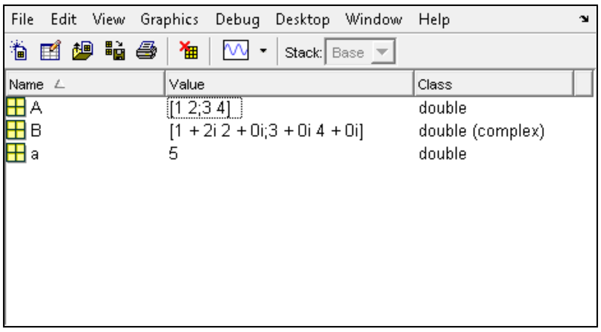
```
>>sqrt(-9)+3

ans =

   3.0000 + 3.0000i
```

Most lássunk egy értékadó utasítást! A Matlab különbséget tesz nagy és kisbetűk között.

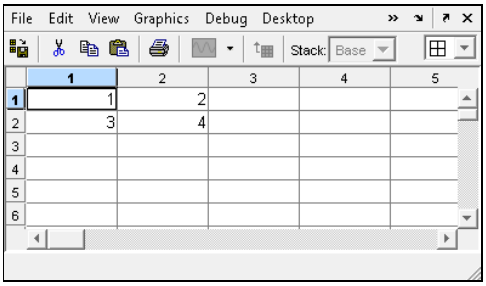
```
>>a=5

a =

     5
```

Ha nem szeretnénk, hogy a Matlab minden utasítás végrehajtása után beszámoljon az eredményről, akkor az utasítás után tegyünk `;`-t (pontosvesszőt).

Bármely változó értékét egyszerűen lekérdezhetjük, nevének begépelésével (persze üssünk `entert`). A rendelkezése álló változók a Matlab felületének Workspace részében táblázatos formában kerülnek nyilvántartásra.



A táblázat tartalmazza a változó nevét, értékét és típusát. A változók értékét új értékadással és egy szerkesztő felület segítségével is módosíthatjuk. A szerkesztőfelület eléréshez az adott változó nevére kell kattintani. A változók böngészésére szolgáló ablak közvetlen lehetőséget biztosít azok értékeinek grafikus megjelenítésére, erről később lesz szó.



Ahogy azt korábban említettük, a Matlab belső számábrázolása igazodik a megfelelő IEEE szabványhoz. Ez azt jelenti, hogy lehetőség van a mínusz és plusz végtelen (`-inf`, `inf`) használatára. Ha egy művelet a Matlab számára értelmezhetetlen, akkor az eredmény `NaN` (Not a Number).

Például

```
>>6/0

Warning: Divide by zero

ans =

     inf

>>inf/inf

ans =

   NaN
```

A Matlab széleskörű használhatóságának titka, hogy a mátrixokkal ugyanolyan könnyen végezhetünk műveleteket, mint skalárokkal. Bármely változó értéke (komplex elemű) mátrix is lehet.

A mátrixok elemeit `[]` zárójelek között sorfolytonosan adhatjuk meg. Az elemeket szóköz (space) vagy vessző (`,`) választhatja el a soron belül. A sor végén új sor (enter) vagy pontosvessző (`;`) állhat, az utolsó sor utolsó elemét közvetlenül a `]` zárójel követi. Ha a gépelés során az összetartozó szintaktikai egységeket egy sorban nem tudjuk megadni (a mátrix sorának elemeit, vagy az utasítást, vagy egy függvény paramétereit, stb.), akkor (elemhatáron, pl. valós szám vagy változó név után), három pontot (...) téve a sor végére, az értékadás vagy függvényhívás, stb. folytatható az új sorban. A mátrix egy absztrakt eleme lehet egy teljes mátrix is, így nehézség nélkül definiálhatunk és kezelhetünk blokkmátrixokat is. Világos, hogy használhatunk változókat is, amelyek szintén mátrixok. A Matlab a változók és függvények neveiben megkülönbözteti a kisbetűt a nagybetűtől.

A mátrixok elemeire indexeikkel hivatkozunk, de skalár változó esetén nem kell indexet megadni, és vektor esetén pedig csak 1 index megadása szükséges. Az indexek helyén változók is állhatnak. A mátrix elemeinek megadásakor az elemet kijelölő indexeket gömbölyű zárójelben ‘`()`’, vesszővel elválasztva adjuk meg, pl. `A(1,2)`, vagy `v(2)`. Egyszerre egy mátrixblokkot is kijelölhetünk, amely az eredeti mátrixból úgy keletkezik, hogy megmondjuk, mely sorokból és oszlopokból kell kivenni a mátrixblokk elemeit, pl. `A([2,4],[1,3])`. Ilyenkor az elemet definiáló indexpár helyén egy-egy vektort adunk meg, ahol az első vektorban álló számok mondják meg, hogy mely sorokból szelektálunk, és a második vektorban álló számok mondják meg, hogy mely oszlopokból szelektálunk. A kijelölt sorok és oszlopok találkozási helyén álló elemek adják az eredménymátrixot. Speciálisan, ha valamelyik vektor nulla elemet is tartalmaz (mivel az indexelés 1-től indul, ezért a 0 index illegális lenne), akkor a vektor nemnulla elemeinek 1 értékűnek kell lenniük, és ekkor az 1 értékű elemek indexei jelölik ki a kiválasztandó sorokat vagy oszlopokat. Ezért egy mátrix nemnulla sorai vagy oszlopai könnyen kiszelektálhatók a Matlab szolgáltatásai segítségével, pl. `B=A(:,any(A))` kiszelektálja az `A` mátrix nemnulla oszlopait és beteszi a `B` mátrixba.

Létezik egy rövidített kijelölési lehetőség is. Ha a vektor helyére kettőspontot teszünk, akkor az első vektor helyén a mátrix minden sorát, a második vektor helyén pedig mindegyik oszlopát jelöljük ki, pl. `A(:,[1,3])`, illetve `A([2,4],:)`. Bármelyik vektor helyén megadhatunk egy intervallumot is úgy, hogy a kezdete és vége közé kettőspontot teszünk, pl. `A(1:3,2)` illetve `A(1,2:4)`. Bármelyik vektor helyén megadhatunk egy ekvidisztáns lépésközű sorozatot is úgy, hogy megadjuk a kezdő indexet, majd kettőspont után a lépésközt, majd újabb kettőspont után a felső index korlátot, pl. `A(:,4:-1:1)`.

Lehetséges a mátrixot oszlopfolytonosan egy hosszú vektorba kiteríteni, pl. `b=A(:)` révén. Ilyenkor a hosszú vektorban az eredeti mátrix első oszlopát a második oszlop, azt a harmadik oszlop stb. követi. Ha viszont az értékadó utasítás baloldalán áll `A(:)`, amit a jobboldalon egy vele összhangban lévő hosszúságú hosszú vektor követ, akkor az előbbi fordítottja játszódik le, az új mátrix mérete az `A` mátrix aktuális méretének megfelelő lesz, de elemei a `b` hosszú vektorból állnak majd elő $A(:)=b$ esetén. Világos, hogy ez a mátrixok könnyű átstrukturálását teszi lehetővé.

Abban a speciális esetben, ha az értékadó utasítás jobboldalán a gömbölyű zárójelben egyetlen mátrix szerepel, pl. `b=A(L)`, ahol `L` és `A` mérete azonos, továbbá `L` elemei csak 0 vagy 1 értékűek lehetnek, akkor rendre megtörténik `L` elemeinek vizsgálata az oszlopok sorrendjében, és az oszlopokban az 1 értékű elemek kijelölése, majd az `A` ugyanolyan helyen álló elemeinek kiemelése és elhelyezése a `b` hosszú vektorban.

Nézzünk néhány egyszerű esetet is.

```
>>A=[1 2; 3 4]

A =

     1     2
     3     4
```

Látható, hogy a mátrix létrehozásakor az oszlop elemit `szóközzel` (lehet '`,`'-vel is), a sorokat pedig '`;`'-vel választjuk el egymástól. Tehát egy sorvektort például a

```
>>b=[1 2];
```

utasítással hozhatunk létre. Természetesen hivatkozhatunk egy mátrix (vagy vektor) bármelyik elemére, sorára vagy oszlopára, illetve blokkjára. Néhány példa:

```
>>A(1,2)       % az első sor második eleme

ans =

     2

>>A(2,:)       % a második sor

ans =

     3     4

>>A(:,1)      % az első oszlop

ans =

     1
     3
```

Nagyobb mátrixnál mindezt tovább bonyolíthatjuk. A `b` és `A` vektort és mátrixot könnyedén összeszorozhatjuk.

```
>>A*b
??? Error using ==> *
Inner matrix dimensions must agree.
```

De nem így! A helyes sorrendben (mivel `b` sorvektor):

```
>>b*A
```

```
ans =

     7    10
```

vagy pedig

```
>>A*b'

ans =

     5
    11
```

ahol `'` (aposztróf) a transzponált képzésének operátora. (A fordított aposztróf ` nem használható!) Adott mátrix determinánsának, inverzének, sajátértékeinek és szinguláris értékeinek számítására rendre a `det`, `inv`, `eig`, `svd` utasítások szolgálnak. Például az `A` mátrix sajátértékei az alábbiak:

```
>>eig(A)

ans =

   -0.3723
    5.3723
```

Egy vektor vagy mátrix dimenzióinak lekérdezéséhez a `length` és a `size` utasításokat használhatjuk.

```
>>length(b')

ans =

     2

>>size(A)

ans =

     2     2
```

Természetesen egyéb függvényeknek is lehetnek mátrixok az argumentumai. Egyes mátrix- és vektortípusokra gyakran lehet szükségünk és ezek előállítása – főleg nagy sor- és oszlopszám esetén – igen fáradságos lenne a fenti módszerekkel. Ezért erre külön utasítások állnak rendelkezésre (pl. `eye`, `ones`, `zeros`, `hankel`, `toeplitz`).

Lényeges még megemlíteni, hogy a Matlab-on belül különleges szerep tulajdonítható a sorvektoroknak. Minden sorvektorhoz hozzárendelhető ugyanis egy olyan egyváltozós polinom, melynek együtthatói egy sorvektor elemei. Tehát például a `p=[2 3 1 4]` sorvektorhoz a

$$2x^3+3x^2+x+4$$

polinomot rendeljük hozzá. A fentiekből látszik, hogy a polinom legmagasabb hatványú tagjának együtthatója a sorvektor első eleme, míg a konstans tag az utolsó eleme. Egy `n`-ed fokú polinom megadásához következésképp egy `n+1` elemű sorvektor szükséges. A Matlabban a `polyval(p,t)` utasítás szolgál a `p` sorvektorral adott polinom `x=t` helyen felvett értékének számításához.

```
>>polyval(p,3)

ans =

    88
```

A `p` polinom deriváltját a `polyder(p)` utasítással számíthatjuk.

```
>>polyder(p)

ans =

     6     6     1
```

Természetesen kiszámolhatjuk két polinom szorzatát is. Erre a `conv(p,q)` utasítás szolgál. Kettőnél több polinom összeszorzásakor több `conv` utasítást kell egymásba ágyazni.

Említsünk meg még néhány utasítást, melyekre szükségünk lehet. A Matlab természetesen fejben tartja az összes létrehozott változót, de mi erre nem biztos, hogy képesek vagyunk. Adott változó törlésére a `clear változó_név`, az összes változó törlésére pedig a `clear all` utasítás szolgál.

Ha munkánkat befejeztük, az eredményt tartalmazó változó(kat) elmenthetjük későbbi felhasználás céljára. Erre a `save` utasítás szolgál. Például a

```
save result A b
```

a `result.mat` állományba menti el az `A` és `b` változókat. A `save` utasítással elmentett változókat a `load` utasítással tölthetjük be. Itt a változok neveit nem kell megadni. A

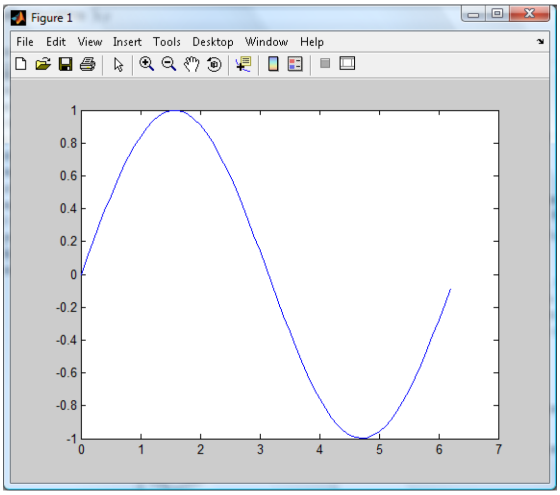
```
load result
```

a `result.mat` állományba tett változókat tölti be. Elmenthetjük a teljes munkaterület tartalmát is (összes létrehozott változónkat). Ekkor a `save` utasítás után nem írunk változónevet, csak az `állomány nevét.` Ha ez utóbbit is elhagyjuk, akkor változóink automatikusan a `matlab.mat` állományba kerülnek. (Hasonló eredményre vezet ha a parancs ablak `File` menüjének `Save Workspace As` parancsát választjuk.) Az egyes változókat a Workaspace böngésző segítségével is elmenthetjük ehhez a jobb egérgombbal kattintsunk az adott változóra és a megjelenő menüből válasszuk a `Save As`... parancsot.

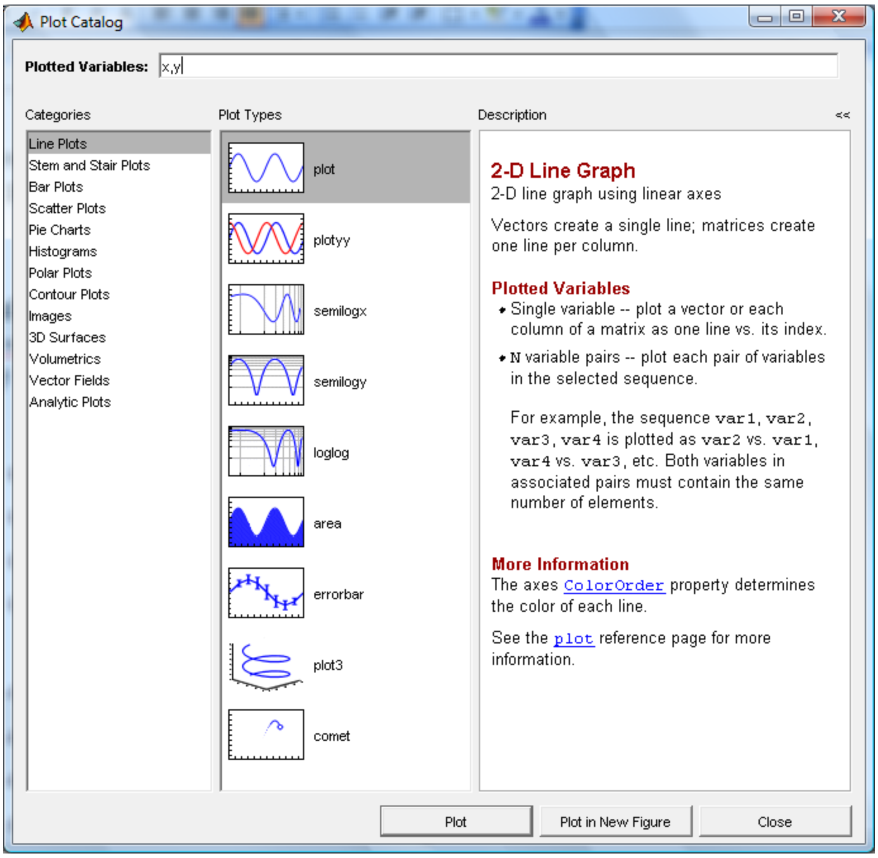
Adatok grafikus megjelenítése

A Matlab lehetőséget ad két- és háromdimenziós grafikai objektumok megjelenítésére. Mi most csak a kétdimenziós megjelenítéssel foglalkozunk. Ennek alapvető eszköze a `plot(x,y)` utasítás. Az utasítás segítségével az aktuális grafikus ablakba (ha ilyen egyáltalán nincs, akkor a `plot` utasítás létrehoz egyet) egy görbét rajzolhatunk. A görbe úgy keletkezik, hogy a síkban `[x(i); y(i)]` koordinátákkal adott pontokat a Matlab egyenes szakaszokkal összeköti.

```
>>x=0:0.1:2*pi; y=sin(x); plot(x,y);
```

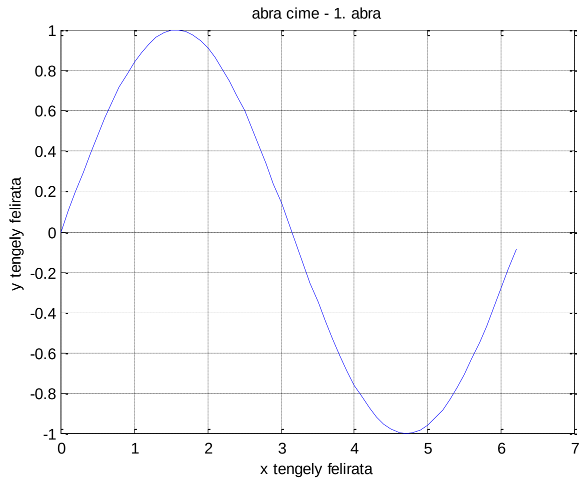


A `plot` utasítással több görbét is megjeleníthetünk, változtathatjuk a színeket és a görbék típusait (szaggatott, pontozott, folytonos, stb.). Ezekkel a lehetőségekkel kapcsolatban lásd a `plot` utasítás helpjét `(doc plot)`. A Matlab ezen kívül számos más síkbeli grafikus ábrázolási módot ismer. Ezek katalógusa a Workspace ablak menüsorából hozzáférhető (`Graphics/More plots…`).



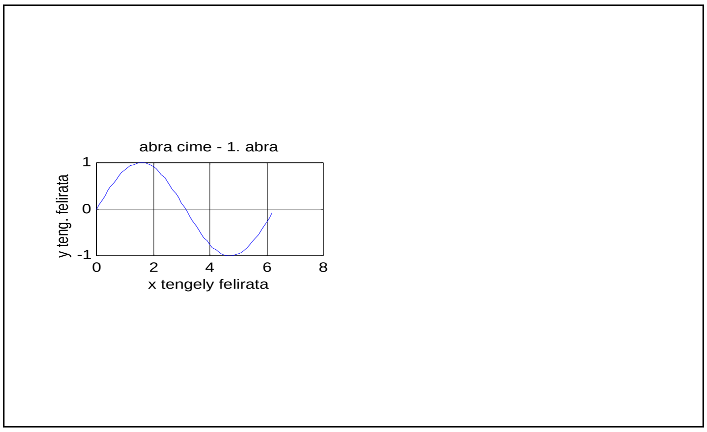
Itt valójában az történik, hogy grafikusan választható ki a megjelenítő utasítás és a `Plotted Variables` sorban kell megadni az argumentumokat. A parancs a `Plot` gomb lenyomásának eredményeképp hajtódik végre a parancsablakban.

A grafikont és a tengelyeket feliratokkal láthatjuk el. Erre a `title, xlabel, ylabel` utasítások szolgálnak. Vetítővonalakat a `grid` utasítással rajzoltathatunk ki. Így feldíszítve az alábbi ábrát kapjuk:



Ha azt szeretnénk, hogy egy új `plot` utasítás ne törölje ki eddigi grafikonunkat, akkor új grafikus ablakot kell létrehoznunk. Erre a `figure` utasítás szolgál. Minden létrehozott grafikus ablakhoz tartozik egy (pozitív egész) szám. Ha több grafikus ablak közül szeretnénk kiválasztani egyet, mondjuk a következő `plot` utasításunk számára, akkor azt a `figure(n)` utasítással tehetjük, ahol `n` az aktivizálni kívánt grafikus ablak száma.

Egy adott grafikus ablakot vízszintesen és függőlegesen is több részre oszthatunk és az így keletkezett részeket külön, külön kezelhetjük. Adott grafikus ablak ilyen felosztására a `subplot(x,y,z)` utasítás használható. `x` és `y` határozza meg, hogy az ablakot hány sorra és hány oszlopra osztjuk, `z` azt mutatja meg, hogy hányadik felületelem lesz éppen az aktuális grafikus kimenet, amelyre a `plot` utasítás (vagy bármely más grafikai utasítás) vonatkozik. Ha például a 2. sor 1. oszlopában lévő felületelemre szeretnénk a grafikus kimenetet irányítani és a grafikus ablakot 2 oszlopra osztottuk fel, akkor `z` értékét 2*1+1=3-ra kell állítani.



A Windows alatt futó alkalmazások számára az ábrát a vágólapon elhelyezve tehetjük hozzáférhetővé. A grafikus ablak tartalmának vágólapra helyezésekor számos opciót állíthatunk be az ablak `Edit/Copy options` parancsa segítségével. Számos mentési formátum áll rendelkezésre a `File/Save As` parancsnál.

Az m kiterjesztésű állományok

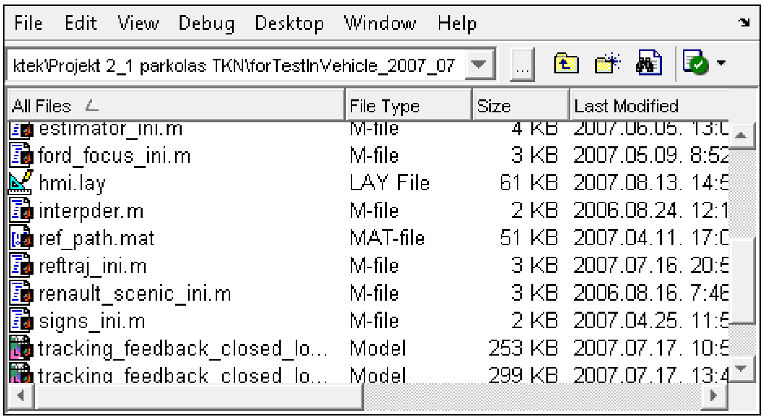
Numerikus módszerek használatakor gyakran van szükség iteratív számítások elvégzésére és nagy mennyiségű adathalmaz kezelésére. Nyilvánvaló, hogy ez nehézkes lenne a parancsablakot használva, egymás után begépelve a szükséges utasításokat. E helyett használhatjuk az ún. `M` állományokat. Ezek szöveges állományok, melyek Matlab utasításokat tartalmaznak. A szöveges állományokat nevük azonosítja. Adott állomány nevét (kiterjesztés nélkül) a parancsablakban utasításként kiadva a Matlab a megfelelő szöveges állomány (melynek kiterjesztése `m` kell, hogy legyen - innen az `m` állomány elnevezés) utasításait egymás után értelmezi és végrehajtja.

Az `m` állományokban lehetőség van ciklusok szervezésére, utasítások feltételes végrehajtására és egyéb programozási trükkök bevetésére is.

`M` kiterjesztésű állományainkat bármely, szöveges (txt) állomány létrehozását támogató szövegszerkesztő segítségével létrehozhatunk, de használhatjuk a Matlab beépített szerkesztőjét is, amely számos fejlesztést támogató funkcióval rendelkezik. A Matlab beépített szerkesztőjéhez hozzáférhetünk a parancs ablak `File` menüjében. Új `m` állomány létrehozásához válasszuk a `New` almenü `M`-file opcióját, már létező `m` állomány 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. Javasolható, hogy a saját `m` állományok (és függvények) a `..\matlab\toolbox\local` alkönyvtárban vagy egy külön erre a célra létrehozott alkönyvtárban kerüljenek elhelyezésre. Az alapértelmezett munkakönyvtár a `\matlab\work`.

Az épp aktuális munkakönyvtárat a `cd` parancs segítségével változtathatjuk meg. Az aktuális könyvtár tartalmának kilistázására a `dir` parancs szolgál. A könyvtárak között a `Current Directory` ablak parancssora segítségével is navigálhatunk.



A Matlab „programozási nyelve”

Mint fentebb említettük az `m` állományokon belül lehetőség van hurkok, elágazások stb. szervezésére. Ezek közül itt a legalapvetőbbeket ismertetjük. A `for` utasítás szolgál ciklus szervezésére. (lásd `help for`) Az `if, else, elseif` utasítások szolgálnak elágazások szervezésére (lásd `help if`).

Az `m` állományokon belül a `';'` szolgál az utasítások elválasztására (egy sorban több utasítás is szerepelhet.) Ügyeljünk a `;` kitételére még akkor is, ha az adott sorban csak egy utasítás szerepel, mert ellenkező esetben a Matlab a parancsablakban megjeleníti az adott utasítás eredményét.

Programunk könnyebb átláthatósága érdekében ne fukarkodjunk megjegyzésekkel. Ha egy sorban bárhol kitesszük a `%` jelet, akkor a sor `%`-ot követő részét a Matlab megjegyzésének tekinti és nem értelmezi.

Szükség lehet arra, hogy a program futását megszakítsuk, majd a felhasználótól valamilyen adatot kérjünk, vagy egyszerűen egy billentyű leütését bevárjuk. A felhasználótól adatot az `input` függvény segítségével kérhetünk. Például a

```
kor=input('Hany eves vagy?');
```

utasítás hátasára a felhasználó szembesül a `Hany eves vagy?` kérdéssel, majd válasza a `kor` változóba kerül. Ebben az esetben a felhasználó bármely értelmes Matlab kifejezést megadhat. A kifejezést a Matlab kiértékeli és az eredmény kerül a `kor` változóba.

A `pause` utasítás hatására a program végrehajtása megáll, gombnyomásra folytatódik.

Függvények létrehozása

A függvények speciális `m` állományok. Mindössze azért speciálisak, mert a `function` kulcsszóval kezdődnek.

```
function [eredmény1,eredmény2,...]=
függvény_név(bemeneti_valtozo1, bemeneti_valtozo2,...)
```

Mint korábban láthattuk a Matlab "beépített" függvényeiről a `help` utasítás segítségével kaphattunk információt. Saját magunk által létrehozott függvényekről is adhatunk `help`-et. Ehhez csak annyit kell tennünk, hogy a `function` kulcsszót követő sorokba megjegyzésként leírjuk azt, amit látni szeretnénk, ha kiadjuk a `help függvény_név` utasítást. A függvények még abban különböznek az `m` állományoktól, hogy minden felhasznált változó lokális. Ha egyes változókat szeretnénk hozzáférhetővé tenni más függvények, vagy a parancsablak számára, akkor azt a

```
global gloabalis_valtozo_1 globalis_valtozo_2 ...
```

utasítás segítségével tehetjük meg, de ezt csak megfontoltan használjuk. Egyébként minden másban az `m` állományoknál ismertetett szabályok maradnak érvényben. Itt is ügyeljünk arra, hogy függvényünket megfelelő alkönyvtárba helyezzük. Lássunk egy példát. Számítsuk ki rekurzív módon n!-t.

```
function [eredmeny]=fakt(n)

%FAKT(n) - n! szamitasa rekurziv modszerrel. Ha n
%nem egesz szam, akkor felfele kerekitjuk.

n=ceil(n); %kerekites
if n==0,
      eredmeny=1;
else
     eredmeny=n*fakt(n-1);
end;
```

Próbáljuk ki függvényünk működését.

```
>>help fakt

  FAKT(n) - n! számítása rekurzív módszerrel. Ha n
  nem egesz szam, akkor felfele kerekitjuk.

>>fakt(6)

ans =

   720
```

Megjegyezzük, hogy a Matlab függvények egy része is `m` állományokként került implementálásra. Egy `toolbox` pedig nem más, mint adott témakör köré csoportosított hasznosabbnál hasznosabb függvények gyűjteménye, melyek szintén `M` állományokként vannak implementálva (és/vagy sok esetben `mex` fájlokként is). Ezért javasoljuk, hogy mielőtt bárki nekiesne saját függvények írásának, vizsgálja meg előbb, hogy nem tartalmaz-e valamelyik `toolbox` céljainak megfelelő (és minden bizonnyal numerikus szempontból stabilabb és gyorsabb) függvényt. Ugyanakkor megfelelő ismeretek birtokában bárki létrehozhatja saját külön alkalmazási ill. kutatási területének megfelelő toolboxot.

A Control System Toolbox

A Control System Toolbox (CST) a szabályozástechnika általunk vizsgált területén hasznos Matlab függvények gyűjteménye. A további gyakorlatok során a feladatok megoldásához mindig szükségünk lesz erre a toolboxra.

A gyakorlatok folyamán véges dimenziójú, lineáris és időinvariáns (angolul Finite Dimensional Linear Time Invariant – FDLTI vagy röviden LTI) rendszerekkel foglalkozunk. A CST ilyen rendszerek analíziséhez nyújt segítséget.

Tekintsük először a folytonos idejű rendszereket. Adott szakasz modelljét többféleképpen megadhatjuk.

Állapottér (state space) reprezentáció. A folyamatot folytonos idejű elsőfokú lineáris differenciálegyenletekkel (állapotegyenlettel) adjuk meg. Ennek a reprezentációnak a Matlab-ban az állapotegyenlet négy mátrixa `(A,B,C,D)` felel meg.

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Átviteli függvény racionális tört alakban (transfer function). A Matlab-ban ennek a reprezentációnak két `sorvektort` feleltetünk meg, melyek az átviteli függvény számlálójának és nevezőjének polinomjait határozzák meg. Az átviteli függvény formája egybemenetű-egykimenetű rendszer esetén az alábbi:

$$\frac{b_0 s^m + b_1 s^{m-1} + \cdots + b_{m-1} s + b_m}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n}$$

Átviteli függvény gyöktényezős alakban (zero pole). Csak annyiban különbözik az előző alaktól, hogy az átviteli függvény számlálóját és nevezőjét faktorizáltuk. Ezt a fajta reprezentációt a Matlab-ban egy skalárral és két vektorral adjuk meg. A skalár határozza meg a rendszer erősítését, a két vektor elemei pedig a számláló és a nevező gyökeinek felelnek meg.

$$k\frac{\prod_{i=1}^{m}(s-z_i)}{\prod_{j=1}^{n}(s-p_j)}$$

Konverzió a leírások között. Bizonyos feltételek teljesülése esetén a fenti három reprezentáció megfeleltethető egymásnak. Ha egy szakasz az egyik reprezentáció szerint adott, kiszámolhatjuk a szakaszhoz tartozó másik két reprezentáció paramétereit is. Erre szolgálnak a Matlab CST `ss2tf, ss2zp, tf2ss, tf2zp, zp2ss, zp2tf` utasításai. Például szeretnénk kiszámolni a

$$\frac{2}{s^2+s+1}$$

átviteli függvénnyel adott szakasz állapotegyenletét. Először rögzítsük változókban az átviteli függvény számlálóját és nevezőjét!

```
>>num=2;
>>den=[1 1 1];
```

Az állapotegyenlet mátrixai ezek után:

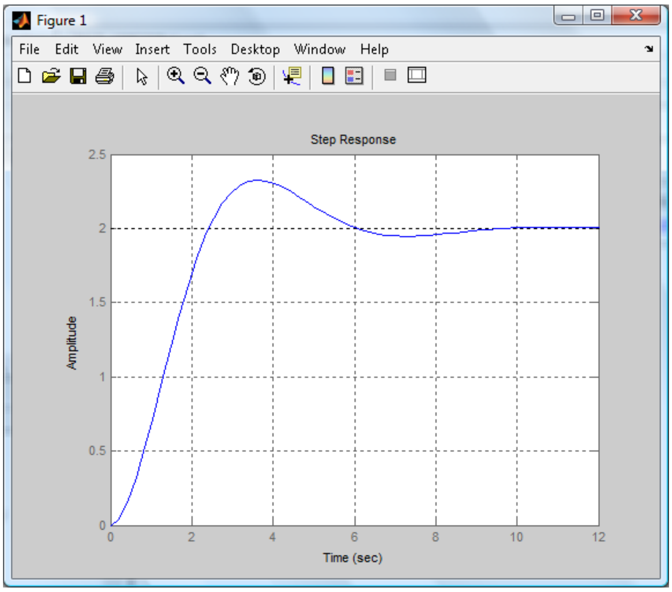
```
>>[A,B,C,D]=tf2ss(num,den)

A =

    -1    -1
     1     0


B =

     1
     0


C =

     0     2


D =

     0
```

A Control System Toolbox segítségével adott szakasz vagy szabályozási kör idő és frekvenciatartománybeli tulajdonságait is vizsgálhatjuk.

Szakasz tulajdonságainak vizsgálata időtartományban

Időtartományban adott szakasz lényeges tulajdonságairól úgy nyerhetünk képet, hogy speciális vizsgálójelekkel gerjesztjük. A CST egyes függvényei a szakasz válaszát számítják ki ilyen vizsgálójelek esetén. A `step` utasítással számíthatjuk ki az egységugrásra adott választ (`dstep`-et használunk, ha diszkrét idejű szakaszról van szó). Az `impulse` utasítás a szakasz válaszát számítja Dirac-delta bemenet esetén (diszkrét idejű megfelelője a `dimpulse`). Példaként számítsuk ki az `A,B,C,D` mátrixokkal adott folytonos idejű szakasz egységugrásra adott válaszát.

```
>>step(A,B,C,D); grid;
```

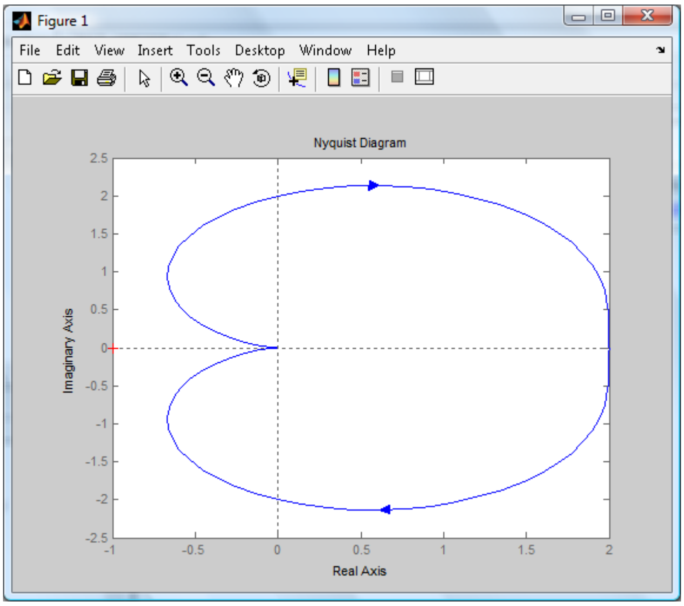


Szakasz tulajdonságainak vizsgálata frekvenciatartományban

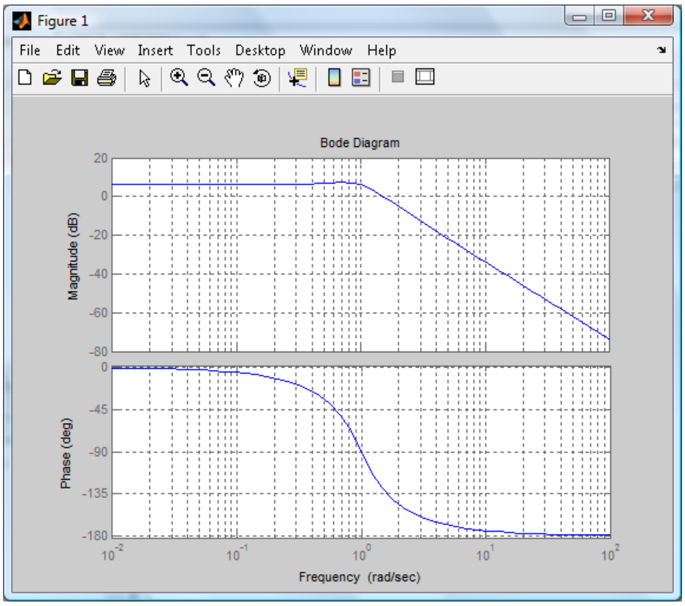
Laplace vagy frekvenciatartományban elsősorban a Nyquist- illetve Bode-diagramokra vagyunk kíváncsiak. Érdekelhet bennünket ezen kívül a pólusok és zérusok elhelyezkedése, illetve a gyökhelygörbe. A Nyquist-diagramot a `nyquist` utasítással állíthatjuk elő. A `bode`, `pzmap`, `rlocus` utasítások szolgálnak a Bode-diagram, a pólus-zérus elhelyezkedés és a gyökhelygörbe megjelenítésére. (A `nyquist` és a `bode` utasítások diszkrét idejű párjai: `dnyquist`, `dbode`).

Példaként nézzük meg hogyan alakul az `A,B,C,D` mátrixokkal adott folytonos idejű szakasz Nyquist- és Bode-diagramja, illetve a pólusok és zérusok elhelyezkedése a komplex számsíkon.

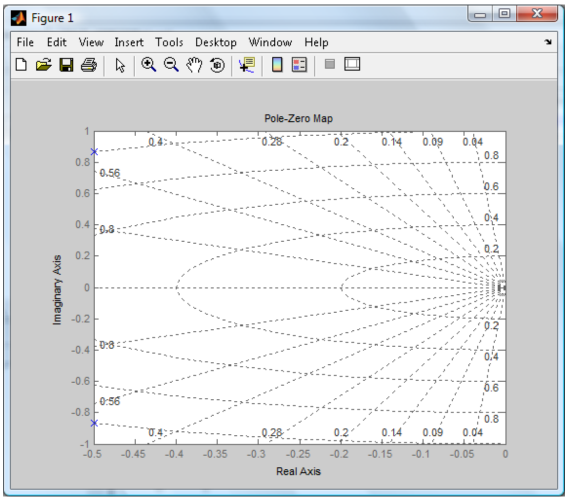
```
nyquist(A,B,C,D);
```




```
bode(A,B,C,D); grid on;
```



```
pzmap(A,B,C,D); grid on;
```



Megjegyzés: A felhasznált függvényeket az itt ismertetettől eltérően is használhatjuk. Ennek módja az adott utasítás help-jében található meg. A utasítások teljes körének és összes lehetséges felhasználásuk ismertetése meghaladja jelen leírás kereteit.

Adatstruktúrák használata a Control System Toolboxban

A Matlab lehetővé teszi a C programozási nyelvből már ismert, azonosítóval rendelkező mezőkből felépített struktúrák kezelését. A mezők bármilyen típusú adatot tartalmazhatnak (másik struktúrát, mátrixot, stb). Tekintettel arra, hogy a Matlab alaptípusa a mátrix, természetesen azonos struktúrákból is építhető mátrix. Valamely adatstruktúra mezőire a pont (`.`) operátor segítségével hivatkozhatunk. A Control System Toolbox lehetővé teszi, hogy folytonos- vagy diszkrétidejű, időinvariáns, lineáris rendszereket struktúraként kezeljünk. A toolbox az eddigiekkel összhangban három adatstruktúrát ismer: átviteli függvény (transfer function), zérus-pólus (zero-pole) és állapotegyenlet (state-space).

Egy adott modell tárolásához szükséges struktúra mezőt az határozza meg, hogy a modellt milyen reprezentáció szerint adtuk meg.

Időtartomány típusa	Reprezentáció	Tárolt struktúra mezői
Folytonos	Átviteli függvény	`num` – számláló(k) cellatömbje `den` – nevező(k) cellatömbje
Folytonos	Zérus – Pólus – Erősítés	`z` – zérusok cellatömbje `p` – pólusok cellatömbje

		`k` – erősítések mátrixa
Folytonos	Állapotegyenlet	`a` – állapotmátrix `b` – bemeneti erősítés mátrixa `c` – kimeneti erősítés mátrixa `d` – közvetlen erősítések mátrixa
Diszkrét	Átviteli függvény	`num` – számláló(k) cellatömbje `den` – nevező(k) cellatömbje `Ts` – mintavételi periódusidő
Diszkrét	Zérus – Pólus – Erősítés	`z` – zérusok cellatömbje `p` – pólusok cellatömbje `k` – erősítések mátrixa `Ts` – mintavételi periódusidő
Diszkrét	Állapotegyenlet	`a` – állapotmátrix `b` – bemeneti erősítés mátrixa `c` – kimeneti erősítés mátrixa `d` – közvetlen erősítések mátrixa `Ts` – mintavételi periódusidő

A különböző modellek reprezentációinak tárolására szolgáló struktúráknak más mezőik is vannak, amelyek jelenleg számunkra érdektelenek. Egy adott struktúra mezőiről a `ltiprops ss/tf/zpk` parancs segítségével kaphatunk tájékoztatást.

Egy modell létrehozásához valamely reprezentációjának mezőit kell megadnunk. Tekintsük például a

$$W(s) = \frac{43{,}71}{s(s+8{,}662)}$$

átviteli függvényű tagot. A modell megadható a

```
>>sys1 = zpk([],[0 -8.662],43.71)

Zero/pole/gain:
   43.71
-----------
s (s+8.662)
```

paranccsal, vagy a

```
>> sys2 = tf([43.71],[1 8.662 0])

Transfer function:
    43.71
```

```
-------------
s^2 + 8.662 s
```

utasítással, ahol a `zpk` és a `tf` argumentumai rendre az adott reprezentációnak megfelelő mezők. A reprezentációk közötti áttérésre a `zpk`, `tf` és `ss` parancsok használhatóak. Például az állapotegyenletet mátrixait

```
>> ss(sys2)

a =
          x1      x2
   x1  -8.662      0
   x2       1      0

b =
       u1
   x1   8
   x2   0

c =
         x1     x2
   y1     0  5.464

d =
       u1
   y1   0

Continuous-time model.
```

utasítással számíttathatjuk ki.

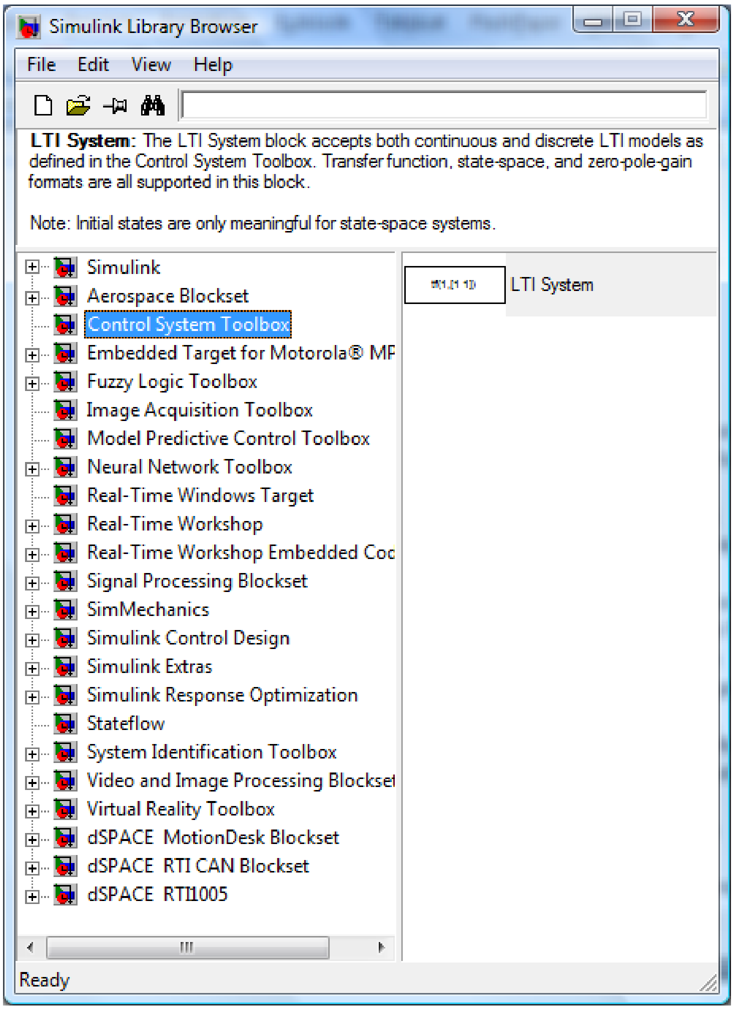
Simulink

A Simulink a Matlab numerikus számítási algoritmusaira támaszkodó grafikus felületű eszköz, melynek segítségével dinamikus rendszerek szimulációját végezhetjük. A szimulált rendszerek nem szorítkoznak az időben invariáns és lineáris rendszerek osztályára és a szorosan vett szimuláció mellett a egyes kiegészítő csomagok segítségével a Simulink más célokra is használható (például automatikus kódgenerálásra, mechatronikai rendszerek szimulációjára és grafikus megjelenítésére). A félév során a Simulink lehetőségeinek csak egy szűkebb körét ismerjük meg. A Simulink számunkra jelenleg legfontosabb előnyei:

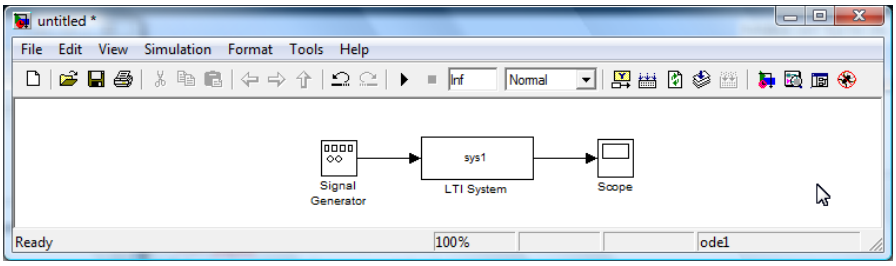
1. A szimulálni kívánt dinamikus rendszer paraméterei a Matlab munkateréből választhatóak.
2. A szimulálni kívánt dinamikus rendszer bemenő jelei a Matlab munkaterében létrehozhatók, a szimuláció során meghatározott jelek ugyanott hozzáférhetőek.
3. Összetett modellek is egyszerűen létrehozhatók.

Modell létrehozása

A Simulink-en belül a szimulálni kívánt dinamikus rendszert modellnek nevezzük, amely egy csomópontokból és irányított élekből álló, hierarchikusan felépíthető, hatásvázlathoz hasonló gráf. A csomópontokban jelformálás és jelátalakítás végezhető, a lehetséges átalakító elemek előre definiált könyvtárakban kerültek csoportosításra. A modellek `mdl` kiterjesztésű állományokban, szöveges formában kerülnek tárolásra. A modell elemeit tartalmazó könyvtár böngészőjének megnyitása a parancsablakból a `Simulink` utasítással, vagy az eszközsorban található Simulink gomb megnyomásával kezdeményezhető.



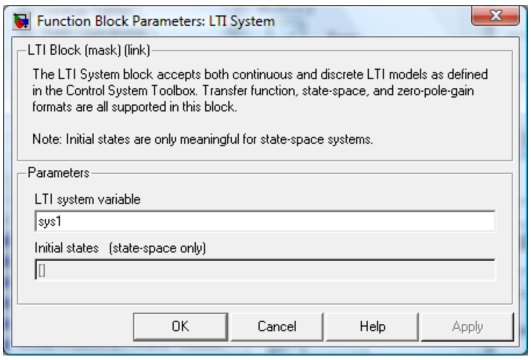
Lényegében két-három könyvtár elemeinek használata fordul elő a félév folyamán: Simulink, Control System Toolbox, Simulink Extras. A modellek létrehozása egy üres modell megnyitásával és annak benépesítésével lehetséges. Ehhez a könyvtárból a jelformáló elemeket áthúzzuk a modellbe és ott a megfelelő módon összekötjük őket. A szerkesztés módja értelemszerű. A jelátalakító elemek működését paramétereik értéke befolyásolja. A paramétereknek közvetlenül is adhatunk értéket, de szerepeltethetünk munkatérbeli változókat, illetve érvényes Matlab kifejezéseket is. Példaként hozzuk létre az alábbi egyszerű modellt!



A modell elemeit az alábbi könyvtárakban találjuk:

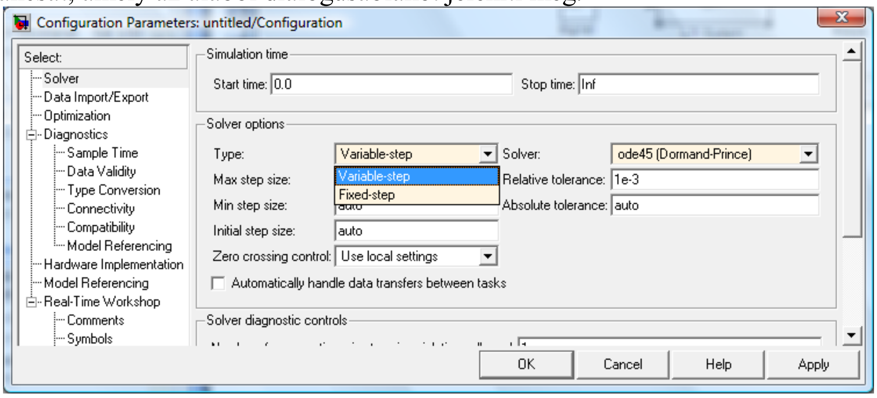
Elem neve	Könyvtár
Signal generator (jelgenerátor)	`Simulink/Sources`
LTI System (LTI rendszer)	`Control System Toolbox`
Scope (oszcilloszkóp)	`Simulink/Sinks`

Az `LTI System` tag esetében a rendszert a korábban ismertetett három struktúra egyikeként kell megadni. Alternatív módon használhatjuk a `Simulink/Continuous` könyvtár `State-Space`, `Transfer Fcn` és `Zero-Pole` elemeit is. Az elemek paramétereinek beállításához annak tulajdonságaihoz tartozó dialógusablakát kell megnyitni. Az LTI rendszerhez tartozó dialógusablak egyik mezőjében adható meg az LTI rendszert leíró modell a `zpk/ss/tf` alakok egyikében.



Szimuláció futtatása

A szimuláció futtatását befolyásoló paraméterek beállításához válasszuk a modellt tartalmazó ablak menüjének `Simulation/Configuration Parameters…` parancsát, amely az alábbi dialógusablakot jeleníti meg.



A számunka jelenleg jelentőséggel bíró beállítási lehetőségek az alábbiak.

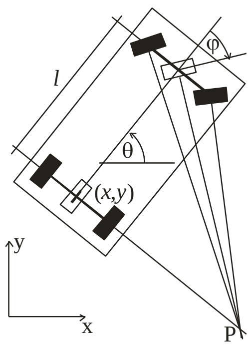
Konfigurációs paraméter	**Magyarázat**
Start time	A szimuláció kezdete
Stop time	A szimuláció leállításának ideje
Solver options / Type	1. Variable-step – változó lépésközű szimuláció. A Simulink állapítja meg a lépésközt és folyamatosan változtatja. Általában ez a gyorsabb alternatíva, de bonyolultabb modellek esetében megbízhatatlanná válhat 2. Fixed-step – állandó lépésközű szimuláció. Általában lassabb.
Solver options / Solver	Az integrálási algoritmus kiválasztása. A magasabbrendű algoritmusok integrálási hibája kisebb, de számításigényesebbek.

Simulink segítségével felépített modell linearizálása

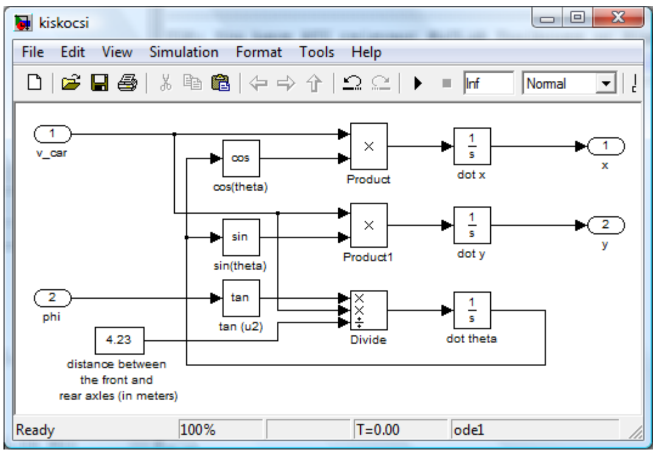
A Simulink segítségével nemlineáris dinamikával rendelkező rendszerek is felépíthetőek és szimulálhatóak, a Matlab pedig lehetővé teszi a rendszerek valamely választott munkapont körül érvényes, kis változásokra linearizált modelljének meghatározását. Erre a `linmod` (`dlinmod`, `linmod2`) utasítás ad lehetőséget. A következő példával a `linmod` utasítás használatát mutatjuk meg. Egy gépjármű egyszerű kinematikai (azaz csak a sebességeket figyelembe vevő) modellje az

$$\dot{x} = v_{car}\cos\theta$$
$$\dot{y} = v_{car}\sin\theta$$
$$\dot{\theta} = \frac{v_{car}}{l}\tan\varphi$$

egyenletekkel adott.



A jármű hátsó tengelyközéppontjának helyzetét a mozgás vízszintes síkjához rögzített koordinátarendszerben az x és y koordinátákkal jellemezzük, az orientációt a θ szög adja meg. A jármű hossztengely irányú sebességét v_{car} jelöli, az elkormányzási szöget pedig φ. Az egyenletekkel adott modell nemlineáris. Tegyük fel, hogy a járművet a hosszirányú sebesség és a φ kormányszög változtatásával irányítjuk és egy olyan lineáris modellt keresünk, amely az egyenes (kis kormányszögek), állandó sebességű (1 m/s) mozgás környezetében felel meg a nemlineáris modellnek. Az egyenleteket megadhatjuk hatásvázlat alakjában is.



A hatásvázlat határozza meg a modell bemeneteit és kimeneteit. Ehhez a `Simulink/Ports & Subsystems` könyvtár `In` és `Out` elemeit kell használni. A hatásvázlatról (és természetesen közvetlenül az egyenletekről) az is látszik, hogy a három állapot az x, az y és a θ, míg a bemenetek v_{car} és φ. A modellt a `kiskocsi.mdl` állományba mentettük el. A nemlineáris modellt az állapottér origója körül nulla kormányelfordulási szög és 1 m/s hossziranyú sebesség mellett kívánjuk linearizálni. A `linmod` alkalmazásával az alábbi rendszert kapjuk. Az utasítás első argumentuma a modellt tartalmazó Simulink állomány neve kiterjesztés nélkül, amit rendre a munkaponti állapotok és bemenet értéke követ.

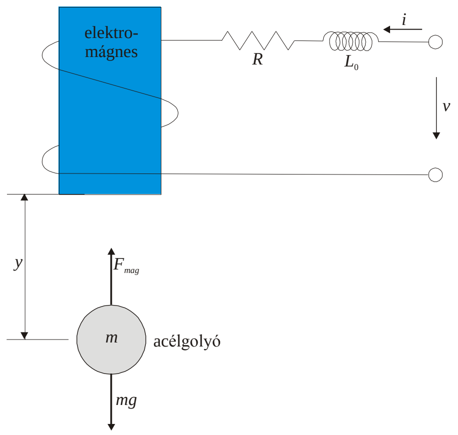
```
>>[A,B,C,D] = linmod('kiskocsi',[0 0 0],[1 0])

A =
         0         0         0
         0         0    1.0000
         0         0         0

B =
    1.0000         0
         0         0
         0    0.2364

C =
     1     0     0
     0     1     0

D =
     0     0
     0     0
```

A másik nemlineáris rendszer, amelyet szintén vizsgálunk, egy mágnesesen lebegtetett golyó. Hasonló modelleket kapunk mágneses felfüggesztések (például mágneses úton lebegtetett vasutak) vizsgálatakor. A elektromágnes tekercsére kapcsolt feszültség a golyóra ható mágneses erőt befolyásolja lehetővé téve annak függőleges gyorsítását. A rendszert $L = \frac{Q}{y} + L_0$ esetén az

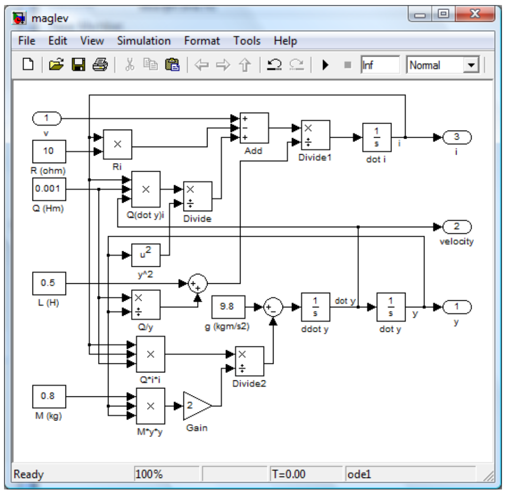
$$m\ddot{y} = mg + \frac{i^2}{2}\frac{dL}{dy} = mg + \frac{i^2}{2}\left(-\frac{Q}{y^2}\right) \Rightarrow \ddot{y} = g - \frac{Qi^2}{2my^2}$$

$$u = Ri + \frac{d}{dt}(Li) = Ri + \left\{-\frac{Q}{y^2}\frac{dy}{dt}i + \left(\frac{Q}{y} + L_0\right)\frac{di}{dt}\right\} \Rightarrow \frac{di}{dt} = \left(\frac{Q}{y} + L_0\right)^{-1}\left(u - Ri + \frac{Q\dot{y}i}{y^2}\right)$$

differenciálegyenletek írják le. Az ábrán a lebegtető rendszer sematikus működési vázlata látható. A golyó tömege $m = 0.8$ kg. A tekercs induktivitása $L_0 = 0.5$ H, ellenállása $R = 10\Omega$. A légréshez tartozó induktivtásváltozás $Q = 0.001$ Hm.



A lebegtetett golyó függőleges pozícióját *y* jelöli, *i* az elektromágnes tekercsében folyó áram, *v* pedig a tekercs kapcsaira kapcsolt feszültség. A rendszert leíró differenciálegyenleteket ismét hatásvázlat alakjában adtuk meg a Simulink segítségével.



A modellnek három kimenete van, a golyó pozíciója, sebessége és a tekercs árama, azaz mindhárom állapotváltozó. Munkapontnak egy adott magasságban mozdulatlan golyóhoz tartozó egyensúly értékeket választunk. Egyensúly esetén az állapotok nem változnak. Legyen az egyensúlyi magasság $y_0 = 0.01\text{m}$. A golyónak ebben a magasságban történő megtartásához a elektromágnes tekercsében folyó áram egyszerűen meghatározható:

$$i_0 = \sqrt{\frac{2mgy_0^2}{Q}} = 1.2522 \text{ A},$$

az ehhez tartozó feszültség pedig

$$v_0 = Ri_0 = 12.522\text{V}$$

A munkapont környezetében a rendszert kis változásokra jó közelítéssel leíró lineáris modell állapotegyenletének mátrixai itt is meghatározhatók a `linmod` utasítással.

```
>>[A,B,C,D] = linmod('maglev',[y0 0 i0],u0)

A =
  1.0e+003 *

         0    0.0010         0
    1.9600         0   -0.0157
         0    0.0209   -0.0167

B =
         0
         0
    1.6667

C =
     1     0     0
     0     1     0
     0     0     1

D =
     0
     0
     0
```

Határozzuk most meg a linearizált rendszer sajátértékeit is.

```
>>eig(A)

ans =

   41.5553
  -36.9436
  -21.2784
```

Vegyük észre, hogy az egyik sajátérték pozitív valós szám, tehát a komplex számsík jobb félsíkjára esik! Az ilyen rendszert labilis rendszernek nevezzük. Ez itt azt jelenti, hogy szabályozás nélkül a golyót nem tudjuk lebegésben tartani.

1. Ellenőrző kérdések a gyakorlathoz

1. Adja meg az $\delta(t)$, $1(t)$, e^{2t}, $t\,e^{3t}$ függvények Laplace transzformáltját!
2. Adja meg az $f(t)$ függvény *deriváltjának* és *integráljának* Laplace transzformáltját, ha az $f(t)$ függvény Laplace transzformáltja $F(s)$!
3. Adja meg a szabályozási kör hatásvázlatát, a benne szereplő blokkok és jelek elnevezését!
4. Adja meg az egytárolós tag átviteli függvényét, amplitúdó- és fázis-jelleggörbéjét és a fázis jó közelítő értékét $0.1/T, 1/T, 10/T$ körfrekvenciákon!
5. Adja meg az egytárolós tag átmeneti függvényét (ugrásválaszát)! Hol metszi a kezdeti érintő a végérték egyenesét? Hány T után állandósul gyakorlatilag az átmeneti függvény?
6. Adja meg a kéttárolós lengő tag átviteli függvényét és pólus/zérus eloszlását! Hogyan függ a pólus valós és képzetes része a csillapítástól és a csillapítatlan sajátfrekvenciától? Milyen csillapítás tartományban van rezonanciája az amplitúdó-jelleggörbének? Milyen csillapítás tartományban van túllövése az átmeneti függvénynek?
7. Adja meg egy kéttárolós lengő tag átmeneti függvényének főbb jellemzőit (túllövés, első maximumig terjedő idő) a csillapítatlan sajátfrekvenciájának és a csillapításának segítségével!
8. Adja meg a kéttárolós lengő tag átmeneti függvényének a végérték α %-os sávjába kerüléséhez szükséges $T_{\alpha\%}$ időt a domináns pólus valós részének σ_e paraméterével kifejezve!
9. Adja meg egy kéttárolós lengő tag (általános) átviteli függvényét és az abban szereplő T időállandó és a ξ csillapítás értékét, ha a lengő tag pólusai:
$$s_{1,2} = -\frac{1}{\sqrt{2}} \pm j\frac{1}{\sqrt{2}}$$
10. Adja meg egy folytonosidejű LTI rendszer állapotegyenletének általános alakját! Mi a kapcsolat az állapotegyenlet mátrixai és az átviteli függvény között? Milyen kapcsolat van a rendszer pólusai és sajátértékei között?
11. Legyen egy rendszer átviteli függvénye
$$W(s) = \frac{2}{s+10}$$
Adja meg a rendszer sajátértékét, pólusát, statikus erősítését és átmeneti függvényét! Mi a különbség a *zpk* alak *k*-ja és a statikus (dc) erősítés értéke között?

12. Legyen egy rendszer átviteli függvénye

$$W(s) = \frac{2}{s+10}$$

Adja meg a rendszer időállandóját, rajzolja fel aszimptotikus amplitúdó menetét, és adja meg a pontos fázisfüggvényt!

13. Legyen egy rendszer átviteli függvénye:

$$W(s) = \frac{s+1}{s^2 + \sqrt{2}s + 1}$$

Adja meg a rendszer sajátértékeit, pólusait, zérusait, T időállandóját (a csillapítatlan sajátfrekvencia reciprokát) és ξ csillapítását!

14. Legyen egy rendszer átviteli függvénye:

$$W(s) = \frac{1}{s^2 + \sqrt{2}s + 1}$$

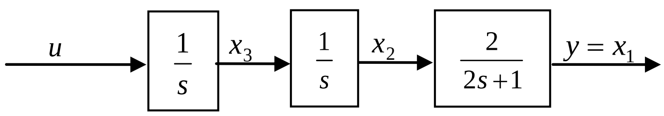
Adja meg a rendszer aszimptotikus amplitúdó menetét és a fázisfüggvény kezdeti és végértékét!

15. Adja meg az

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{bmatrix} -1 & 2 \\ 1 & 0 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u$$
$$y = x_1$$

állapotegyenlettel leírható rendszer karakterisztikus polinomját, sajátértékeit és az átviteli függvény pólusait!

16. Egy u bemenetű és y kimenetű időinvariáns SISO lineáris rendszer hatásvázlata az alábbi:



Adja meg az állapotegyenletet az ábrán feltüntetett állapotválasztás mellett! Mik az állapotmátrix sajátértékei? Adja meg az eredő átviteli függvényt és annak pólusait! Adja meg a SISO rendszerhez tartozó pólus-zérus eloszlást a komplex számsíkon!

17. Adja meg az LTI rendszer Control System Toolboxban lehetséges 3 reprezentációjának általános alakját! Adja meg a leírások közötti konverzióra szolgáló függvényeket!
18. Definiálja a

$$\dot{x} = f(x,u) \qquad x \in R^n;\ u \in R^r$$

$$y = h(x,u) \qquad y \in R^m$$

nemlineráis állapotegyenlettel adott rendszer egyensúlyi állapotát és a hozzá tartozó bemenetet és kimenetet! Adja meg egy egyensúlyi állapot környezetében érvényes lineáris állapotegyenlet mátrixainak számítására szolgáló kifejezéseket!