Soros kompenzátor tervezés Matlab segédlet

FIGYELEM: Az elektronikus labor 20 kérdésből álló (feleletválasztós) beugró teszttel indul (min. 60% kell a sikeres teljesítéshez), melynek anyaga a 2., 3.a., 3.b. EA-k, ill. az addig megtartott GYAK órák anyaga!
TIPP: Az E-labor elkezdése előtt indítsuk el a Matlabot (vagy akár az Online Matlabot: https://matlab.mathworks.com), mert annak indulása több percig eltarthat!

1 Bevezetés

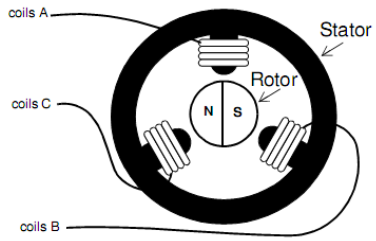
A Moodle felületen végzendő elektronikus labor során egy kefe nélküli egyenáramú villanymotor (BLDC) motor modellje alapján különböző soros kompenzátorok tervezésére kerül sor. Első lépésként a segédletben bemutatásra kerül a BLDC motor matematikai modellje, majd annak linearizálása. A feladat a mérés során előre megadott P, I és PI szabályozók mellett a zárt rendszer tulajdonságainak vizsgálata. Végül a Matlab beépített PID Tuner moduljával előre megadott minőségi jellemzők alapján egy PI szabályozó megtervezése.

2 BLDC motor

A fejezet a BLDC motor felépítését, alapvető tulajdonságait és működési elvét mutatja be.

2.1 BLDC motor jellemzői

A BLDC (Brushless DC: kefenélküli egyenáramú) motorok az egyenáramú (DC) motorok legújabb típusai. Egyszerűbb felépítésűek, mint a hagyományos DC motorok, mivel nem tartalmaznak kefét és így sokkal kisebb a mechanikai kopás, valamint jóval kevesebb karbantartást igényelnek. Az egyszerűbb felépítésüknek köszönhetően jobbak a működési paramétereik, ezért egyre szélesebb körben alkalmazzák ezeket az eszközöket. Mivel nem lép fel a kefeszikrázás jelensége, ezért tűz- és robbanásveszélyes környezetben is kiválthatja a hagyományos egyenáramú motorokat. Továbbá a méret-teljesítmény arányuk is jóval kedvezőbb, mint a hagyományos egyenáramú motoroknak.
Bár egyenáramú motorokról van szó, mégis a BLDC motorok szerkezeti felépítése (1. ábra) a szinkron AC motorokéval egyezik meg. A motor állórészből (sztator) és forgórészből (rotor) áll. Az állórészen vannak a tekercsek („coil”), míg a forgórészen az állandó mágnesek helyezkednek el. A motorok fontos jellemzője a fázisainak a száma. A folytatásban háromfázisú BLDC motorral foglalkozunk.



1. ábra BLDC motor felépítése

Egy fázishoz tartozó tekercs fizikailag általában több, egymással galvanikusan összekapcsolt tekercsből áll (elosztott tekercselés). Ekkor, pl. ha egy három fázisú motor fázisonként két tekercset tartalmaz, akkor az egyes fázisok 60 fokkal, az egy fázishoz tartozó tekercsek pedig 180 fokkal elforgatva követik egymást. A tekercsekhez hasonlóan az állandó mágneses póluspárok száma is lehet egynél több. Ez határozza meg a mechanikai és az elektromos fordulatszám közötti váltószámot:

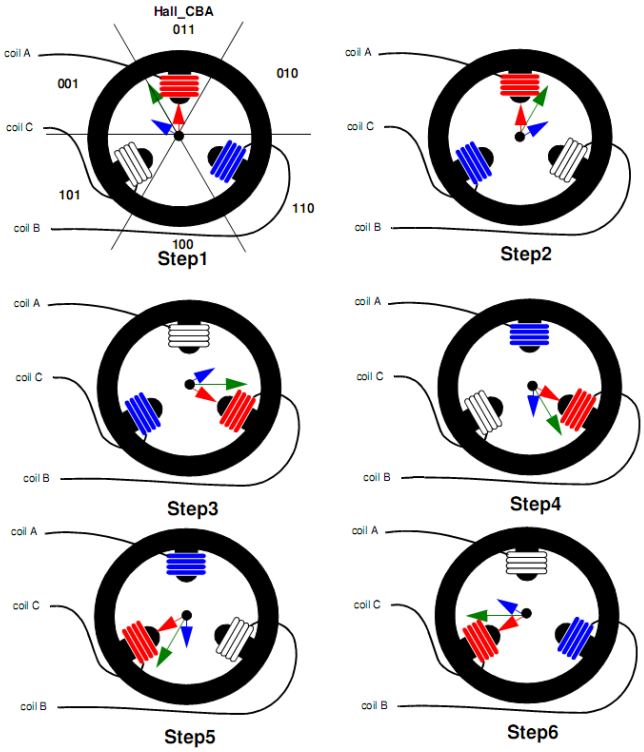
$$r_{el} = N_p \cdot r_{mech}$$

ahol r_{el} az elektromos, r_{mech} a mechanikai fordulatszám, N_p pedig a póluspárok száma. A tekercsek és a mágnesek számának növelésével a motor által leadott nyomaték időben egyenletesebbé válik, ezért javulnak a motor egyéb paraméterei. A hagyományos DC motorokkal ellentétben a BLDC motorokban nincsenek kefék és hiányzik a kommutátor is. Ezért a kommutációt

mindig valamilyen külső egységnek kell vezérelnie. Emiatt a BLDC motorokat szokás elektromosan vezérelt motoroknak is nevezni. Ezen motorok legnagyobb előnye az, hogy fordulatszám-nyomaték görbéjük közel lineáris és, hogy alacsony fordulatszámon is képesek nagy nyomatékot létrehozni. A fordulatszámuk csak a motorra kapcsolt feszültség nagyságától függ és azzal nagy működési tartományon egyenesen arányos. A BLDC motorok fordulatszám-szabályozása a feszültség amplitúdó változtatásával történik. A gyakorlatban az ún. chopperes vagy impulzus szélesség moduláció megoldást alkalmazzák: adott frekvenciájú négyszögjelet kapcsolnak a motorra és a négyszögjel kitöltési tényezőjét változtatják. Így a motor vagy tápfeszültséget kapva be van kapcsolva, vagy nem kap feszültséget és ekkor ki van kapcsolva. A kitöltési tényező változtatásával a kikapcsolt/bekapcsolt idők arányát lehet változtatni. Ugyanakkor a kapcsolások olyan gyorsan történnek, hogy azt a motor nem tudja követni és így a kitöltési tényezőnek megfelelően egy átlagos feszültségszint alakul ki a motor kapcsain.

2.2 BLDC motor vezérlése

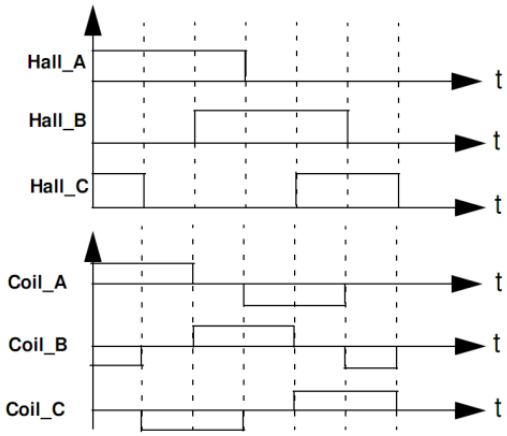
A motor működésének alapja az, hogy a forgórészen lévő mágnesek által létrehozott mágneses tér igyekszik az állórészben lévő tekercsek által gerjesztett mágneses tér irányának megfelelően beállni. Ha a két mágneses tér iránya között eltérés van, akkor a motor tengelyén az eltérés nagyságának függvényében forgatónyomaték ébred, ami ezt az eltérést csökkenteni igyekszik. A forgórész akkor forog, ha minden pillanatban valamekkora szögeltérés van a két mágneses tér iránya között. Ezért az állórészben lévő tekercsekben gerjesztett mágneses tér irányát folyamatosan változtatni kell. Mivel a tekercsek a helyükről nem mozdulnak el ezért egyetlen lehetőség az, hogy a tekercseken átfolyó áram irányát változtatjuk meg. A motor tengelyére ható forgatónyomaték többek között a sztator és a rotor mágneses mezőinek irányai közötti szög szinuszával arányos, így akkor maximális, amikor ez a szögeltérés 90 fok. A 2. ábrán látható módon egy háromfázisú motorban – mindig csak két tekercset gerjesztve - 6 eltérő irányú mágneses teret lehet létrehozni.



2. ábra BLDC motor kommutációs lépései

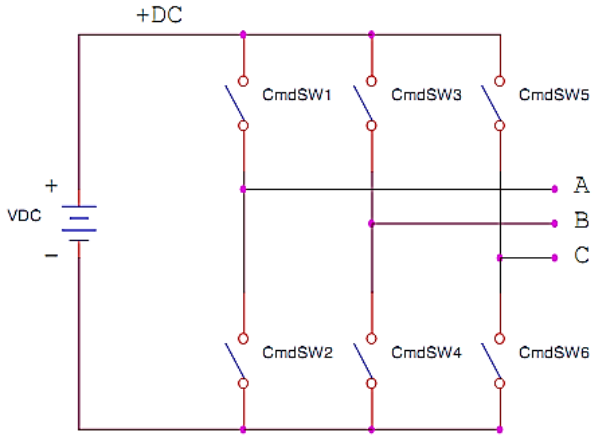
A kefés DC motorokban a tekercsek megfelelő gerjesztését a kommutátor és a kefék végzik, a forgórész lemaradását pedig a szerkezeti kialakítás határozza meg. Mivel a BLDC motorokban nincsenek kefék, se kommutátor, ezért a tekercsek kapcsolását külső hardver elemek segítségével kell

megoldani. A vezérlést ebben az esetben is a rotor és a sztator egymáshoz viszonyított helyzete alapján kell végezni. A forgórész lemaradását vagy külső érzékelő, (Hall-szenzor) segítségével vagy érzékelő nélkül a tekercsben a forgórész mágnese által indukált feszültségek mérése alapján lehet meghatározni. A forgórész lemaradásának ismeretében meg lehet valósítani a kommutációt, ami során a megfelelő fázisok tekercseit adott potenciálra kell kapcsolni. Mind a hat mágneses irányhoz egy-egy kapcsolási konfiguráció tartozik, melyek mindegyikében az egyik fázis a tápfeszültségre, a másik fázis pedig a földre kapcsolódik. A harmadik fázis nincs egyik potenciálra sem kötve, ez lebeg (lásd 3. ábra).



3. ábra BLDC motor vezérlési diagramja

A fázisok kapcsolását úgynevezett H-híd segítségével lehet megoldani (lásd 4. ábra). A vezérlő elektronika feladata a rotor lemaradásának figyelése és a megfelelő pillanatban a kommutáció elvégzése, azaz adott ütem szerint a forgásiránynak megfelelően a következő kapcsolási konfiguráció beállítása a motor fázisain. Bár adott fordulatszámon és adott terheléssel a kommutációs ütem állandónak tekinthető, azonban a pillanatértékeket folyamatosan korrigálni kell.



4. ábra H-híd

Mivel a forgatónyomaték akkor a legnagyobb, ha a sztator és a rotor mágneses tereinek iránya közötti eltérés 90 fok, ezért a vezérlés célja, hogy megfelelő ütemezés beállításával ezt a szöget 90 fok körül tartsa. Ha a kommutáció ütemezése pontos, akkor a BLDC motor ugyanúgy viselkedik, mint egy hagyományos kefés DC motor és így a szabályozásnál alkalmazhatóak az egyenáramú motorra felírt egyenletek is, ami a BLDC motorok szabályozását nagymértékben leegyszerűsíti.
A forgórész lemaradásának mérésére az egyik megoldás a Hall-szenzor alkalmazása. A Hall-szenzor akkor változtatja a kimenetének logikai szintjét, ha az érzékelő előtt a mágneses tér iránya megváltozik. Vezérlés során mindig a lebegő fázishoz tartozó Hall-szenzor kimenetét vizsgáljuk. Ez akkor vált értéket, amikor a forgórész éppen 90 fokkal van lemaradva az állórészhez képest. Vezérlés szempontjából ez az előző és a következő kommutáció közti idő felénél következik be, tehát az előző kommutáció és a Hall-szenzor kimenetének megváltozása közötti idő alapján lehet meghatározni a

következő kommutáció időpontját. Ez a tulajdonság bármilyen fordulatszámon megbízható vezérlést biztosít a motor számára.

Hall-szenzorok értékei (CBA)	Fázis	Kapcsolók zárva
101	A-B	SW1, SW4
001	A-C	SW1, SW6
011	B-C	SW3, SW6
010	B-A	SW3, SW2
110	C-A	SW5, SW2
100	C-B	SW1, SW4

A táblázat a kommutáció vezérlési tábláját mutatja be óramutató járásával megegyező forgásirány esetén. A vezérlőnek ezt a kapcsolási sorrendet kell megvalósítani a Hall-szenzorok állapotváltozásainak megfelelően.

2.3 A BLDC motor matematikai modellje

A BLDC motor modellje nagyban hasonlít a hagyományos egyenáramú motor modelljére. Amennyiben a vezérlő pontosan valósítja meg a kommutációt, úgy az egyenáramú motorra érvényes törvényszerűségeket használhatjuk. A különbség a mechanikai és elektromos időállandók számításánál fog adódni, amelyre a megfelelő helyeken kitérünk.
A Kirchhoff huroktörvény alapján az egyenáramú motorra az alábbi egyenlet írható fel:

$$U_k = I_a R_a + L\frac{di}{dt} + U_i$$

ahol:

- U_k : kapocsfeszültség
- I_a : armatúra áram
- R_a : armatúra ellenállás
- L : a tekercs induktivitása
- U_i : indukált feszültség

Az indukált feszültséget kifejezve az alábbi egyenletet kapjuk:

$$U_i = -I_a R_a - L\frac{di}{dt} + U_k$$

A motor mechanikai viselkedését Newton II. törvénye alapján írhatjuk fel:

$$M = k_s \omega + J\frac{d\omega}{dt} + M_t$$

ahol:

- M: a forgórész nyomatéka
- k_s : súrlódási együttható
- J : a forgórész tehetetlensége
- M_t : a terhelő nyomaték

A kapcsolat az elektromos és a mechanikai rész között az indukált feszültségen keresztül jön létre. Az indukált feszültség és a forgórész nyomatéka is arányos a szögsebességgel:

$$U_i = k_i \omega$$
$$M = k_m \omega$$

ahol k_i és k_m a gépre jellemző konstansok.
A fentiek alapján a rendszert leíró differenciálegyenletek:

$$\frac{di}{dt} = -I_a\frac{R_a}{L} - \frac{k_i}{L}\omega + \frac{1}{L}U_k$$

$$\frac{d\omega}{dt} = -I_a \frac{k_m}{J} - \frac{k_s}{J}\omega + \frac{1}{J} M_t$$

A Laplace-transzformáció elvégzése után (minden kezdeti feltételt zérusnak feltételezve) az alábbi egyenleteket kapjuk:

$$s I_a = -I_a \frac{R_a}{L} - \frac{k_i}{L}\omega + \frac{1}{L} U_k$$

$$s\omega = -I_a \frac{k_m}{J} - \frac{k_s}{J}\omega + \frac{1}{J} M_t$$

A terhelő nyomatékot 0-nak tekintve:

$$s\omega = -I_a \frac{k_m}{J} - \frac{k_s}{J}\omega$$

Az armatúraáramot kifejezve az alábbi egyenleteket kapjuk:

$$I_a = \frac{s\omega + \frac{k_s}{J}\omega}{\frac{k_m}{J}}$$

$$\frac{s\omega + \frac{k_s}{J}\omega}{\frac{k_m}{J}} \left(s + \frac{R_a}{L} \right) = \frac{-k_i}{L}\omega + \frac{1}{L} U_k$$

Átrendezve:

$$U_k = \left(\frac{s^2 JL + s k_s L + s R_a J + k_s R_a + k_i k_m}{k_m} \right) \omega$$

Az átviteli függvény így a következő lesz:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{\omega}{U_k} = \frac{k_m}{s^2 JL + s k_s L + s R_a J + k_s R_a + k_i k_m}$$

A továbbiakban a következő feltételezésekkel élünk:

- A súrlódási együttható nagyon kicsi, tehát k_s közel 0.
- $R_a J \gg k_s L$
- $k_i k_m \gg k_s R_a$

Ezek alapján az átviteli függvény az alábbi lesz:

$$G(s) = \frac{k_m}{s^2 JL + s R_a J + k_i k_m}$$

Az átviteli függvény számlálóját és nevezőjét beszorozva az $\frac{R_a}{k_i k_m} \cdot \frac{1}{R_a}$ kifejezéssel majd átrendezve a következő átviteli függvényt kapjuk:

$$G(s) = \frac{1/k_i}{\frac{R_a J}{k_i k_m}\frac{L}{R_a} s^2 + \frac{R_a J}{k_i k_m} s + 1}$$

Ebből a mechanikai és elektromos időállandók:

$$\tau_m = \frac{R_a J}{k_i k_m} \quad \text{és} \quad \tau_e = \frac{L}{R_a}$$

Összevonva az egyenleteket az átviteli függvény az alábbi lesz:

$$G(s)=\frac{1/k_i}{\tau_m \tau_e s^2+\tau_m s+1}$$

Az egyenáramú motorokhoz képest, a BLDC motorok esetén a legfőbb eltérés a háromfázisú tekercselés, amely általában szimmetrikus elrendezésű. Ez a különbség a mechanikai és elektromos időállandóra van hatással. A BLDC motoroknál figyelembe kell venni a fázisok egymásra hatását is, azaz a kölcsönös indukciót. Mivel az elrendezés szimmetrikus és a fázisok száma 3, ezért az időállandók a következőképpen alakulnak: $\tau_m=\frac{3RJ}{k_i k_m}$, ill. $\tau_e=\frac{L}{3R}$.

A mérésben szereplő motor paraméterei:

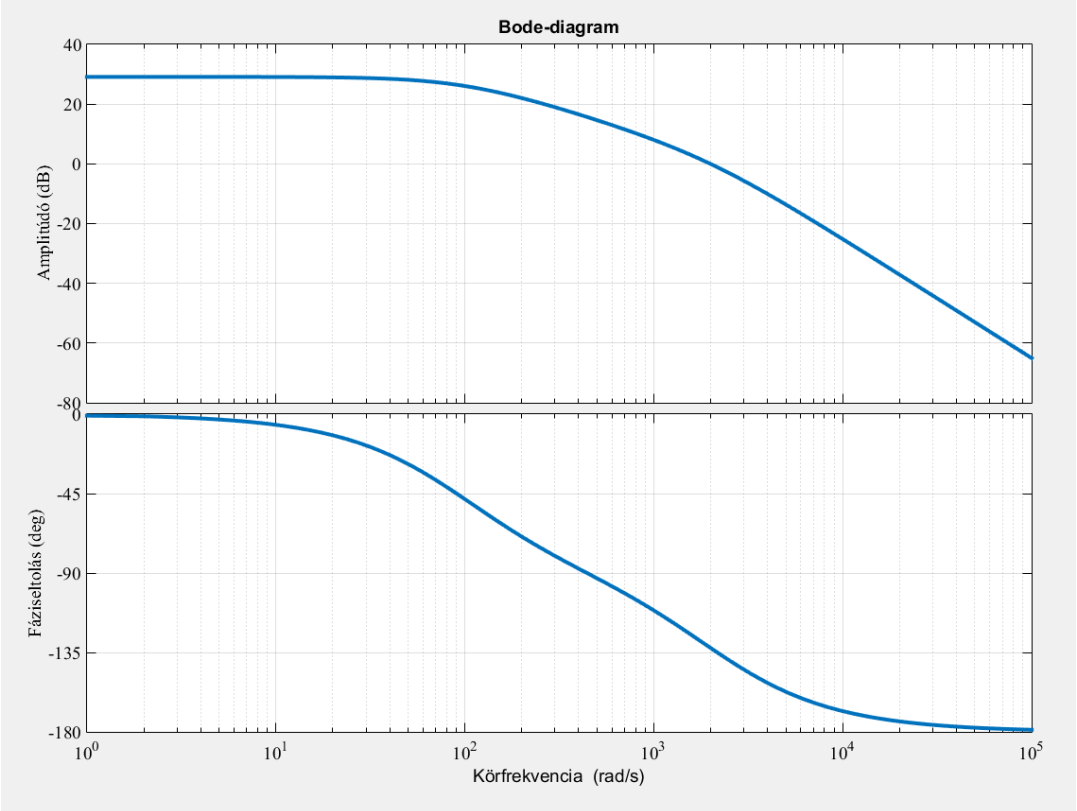
- $R=1{,}8\,\Omega$
- $L=2{,}6\,mH$
- $k_m=0{,}035\,\frac{Nm}{A}$
- $k_i=0{,}03497\,\frac{V\cdot s}{rad}$
- $J=2{,}4\cdot 10^{-6}\,kg\,m^2$

Ezekből:

$\tau_e=4{,}815\cdot 10^{-4}$ és $\tau_m=0{,}01059$

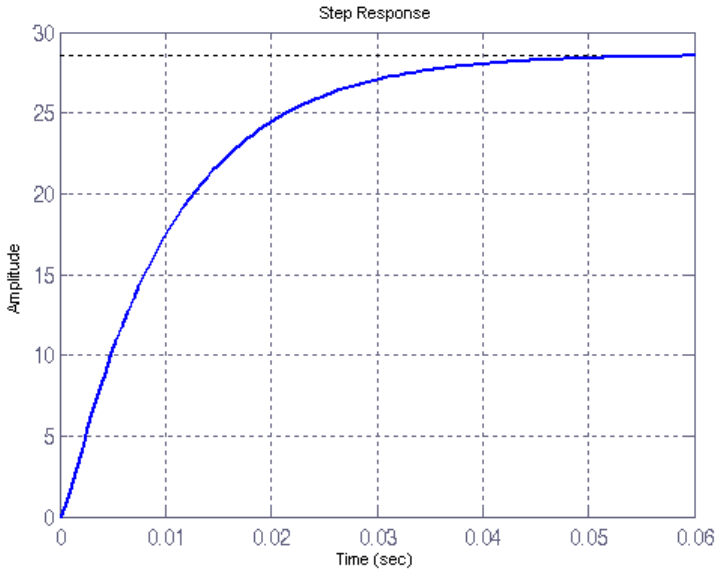
Végül megkapjuk a rendszer átviteli függvényét, amely alapján a szabályozás tervezhető (5. ábra).

$$G(s)=\frac{28{,}6}{5{,}099\cdot 10^{-6}\cdot s^2+0{,}01059\cdot s+1}$$



5. ábra BLDC motor átviteli függvényének Bode diagramja

A rendszer átmeneti függvénye a 6. ábrán látható. Az időállandókkal összhangban ebből is látszik, hogy a rendszer nagyon gyors, beállási ideje kb. 0,03 s.



6. ábra A rendszer átmeneti függvénye

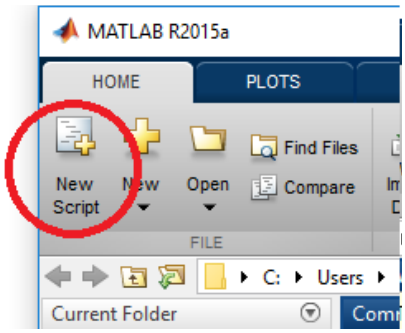
3 Soros kompenzátor tervezése

A tervezéshez a *Matlab Control Systems Toolbox*-jára lesz szükség. A feladat során *Matlab script* segítségével tervezünk PID szabályozást, majd megismerkedünk a Matlab PID Tuner moduljával is.

3.1 Matlab script

A *Matlab*-ban lehetőség van önálló programokat írni. Ez abban az esetben célszerű, ha kódunk nem csak néhány sorból áll vagy azt meg szeretnénk osztani. A mérésnek nem célja a *Matlab* programozási nyelv részletes bemutatása, itt csak a méréshez szükséges néhány parancs kerül részletezésre. Érdeklődőknek:
https://www.mathworks.com/help/matlab/learn_matlab/scripts.html

Hozzunk létre egy új script-et (.m fájl) a bal felső sarokban található ikonnal. Ekkor egy új Editor ablak jelenik meg. Ide írhatjuk a programunkat, hasonló módon, mint ahogy a Command Window-ba tennénk.



Először az alábbi 3 parancsot írjuk be a megnyíló ablakba:

```
close all %Bezárja az összes Figure ablakot
clear all %Letörli az összes Workspace változót
clc %Letörli a Command window-ba írt szöveget
```

A „%” jel utáni részek kommentek: nem futnak le, csak magyarázat készítésére szolgálnak. Itt ékezetes betűk is használhatók ellentétben a változó nevekkel.

Hozzuk létre a rendszer átviteli függvényét! Azaz a *tf()* függvény felhasználva írjuk bele a script-be az alábbit:

```
G = tf(28.6, [5.099E-6 0.01059 1]); %BLDC motor átviteli függvénye
```

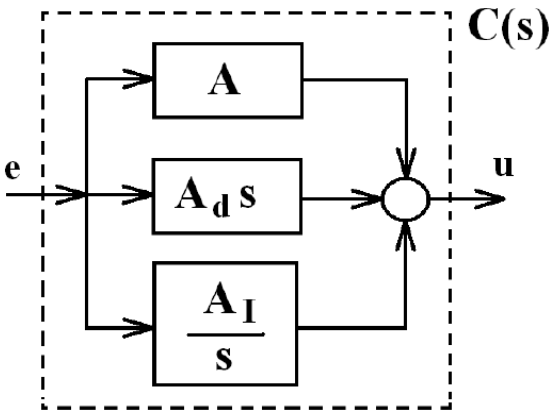
Mentsük el a script-et és fent a **Run** (zöld nyíl) gombbal vagy az **F5**-el futtassuk le. Hosszabb programok esetén ez több másodpercet vehet igénybe. Figyeljük a *Matlab* fő ablakának bal alsó

sarkában a „Busy” feliratot. Ha a program lefutott, a $G(s)$ átviteli függvény megjelenik a Workspace-ben, mint „G” változó.
A rendszer pólusai a *pole()* függvénnyel kérdezhetők le. Adjuk hozzá az eddig írt script-hez az alábbi sorokat, és futtassuk le:

```
disp('A rendszer pólusai:')
p = pole(G)
```

Itt a *disp()* paranccsal a Command Window-ba írhatunk ki szöveget. Mivel a *pole()* függvényhívás után nincs pontosvessző, ezért annak eredménye meg fog jelenni a Command Window ablakban.

Következő lépésben a szabályozó átviteli függvényét hozzuk létre. Ez többféleképp lehetséges, legegyszerűbb esetben a *pid()* függvény meghívásával az arányos (A), integráló (A_i) és differenciáló (A_d) erősítések megadásával (ebben a sorrendben). Amennyiben pl. csak egy arányos szabályozót akarunk alkalmazni, az integráló és differenciáló erősítések értékét 0-ára állítjuk.



7. ábra PID szabályozó

```
%PID szabályozó átviteli függvényének létrehozása:
A = 2; %arányos tag
Ai = 0; %integráló tag
Ad = 0; %differenciáló tag
C = pid(A, Ai, Ad); %ebben az esetben P szabályozó csak
```

Más típusú szabályozó esetén elegendő csak az A, `Ai` és Ad változók értékét átírni.

A rendszer hurokátviteli- és átviteli függvényei a következőképp számíthatók:

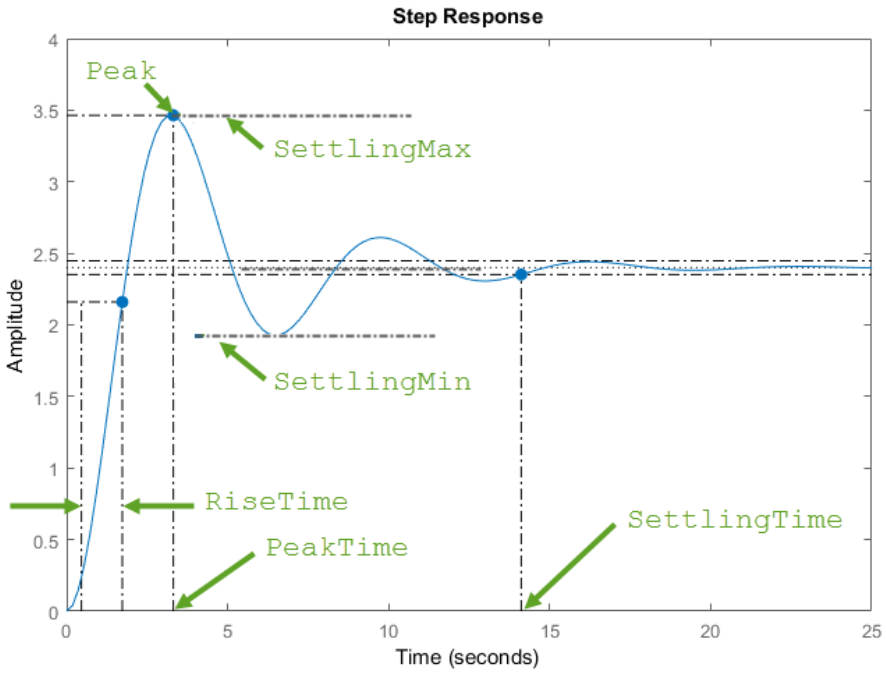
```
%Hurokátviteli függvény:
Gh = C*G;
%A zárt rendszer átviteli függvénye (nincs szenzorfüggvény a visszacsatolt ágban!):
Gz = Gh/(1+Gh);
```

A szabályozott rendszer egységugrásra adott válaszfüggvényét a *step()* parancs segítségével lehet kirajzolni. Az átmeneti függvény minőségi jellemzői pedig lekérdezhetők a *stepinfo()* paranccsal. Itt fontos beállítani a szabályozási idő számításának módját. Alapértelmezett esetben a beállási érték ±2%-ánál vett időt írná ki a program, viszont a feladatban a ±5%-os határ elérésére vagyunk kíváncsiak. Emiatt az alábbi kódot alkalmazzuk!

```
%Átmeneti függvény:
figure; %diagramot készítünk!
step(Gz);
%Időtartományi minőségi tulajdonságok
S = stepinfo(Gz,'SettlingTimeLimits', 0.05)
```

A minőségi tulajdonságok magyarázata Matlab *stepinfo()* parancs esetén:

RiseTime	Felfutási idő (beállási érték 10%-áról 90%-ára)
SettlingTime	Szabályozási idő
SettlingMin	Felfutás után a minimum érték
SettlingMax	Felfutás után a maximum érték
Overshoot	Túllövés %-ban (a túllendülés mértéke)
Undershoot	Alullövés (%-ban)
Peak	Csúcsérték
PeakTime	Túllendülési idő



8. ábra Időtartományi minőségi jellemzők (Forrás: mathworks.com)

A Bode-diagram a *bode()* paranccsal rajzolható ki. Ehhez kell egy új Figure ablak, amely a *figure()* paranccsal hozható létre:

```
%A felnyitott hurok Bode-diagramja (pl. fázistartalék ellenőrzésére)
figure; %diagramot készítünk!
bode(Gh);
```

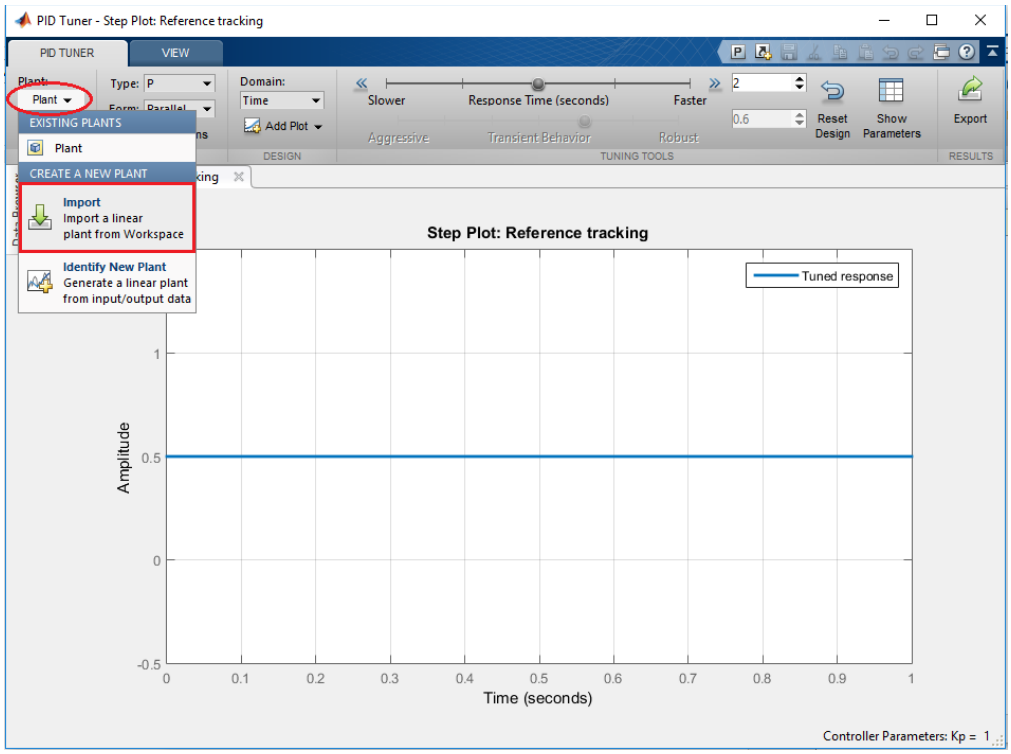
A kész program, amely menthető egy külön Matlab script (.m kiterjesztésű) fájlba):

```
close all %Bezárja az összes Figure ablakot
clear all %Letörli az összes Workspace változót
clc %Letörli a Command window-ba írt szöveget

G = tf(28.6, [5.099E-6 0.01059 1]); %BLDC motor átviteli függvénye

disp('A rendszer pólusai:')
p = pole(G)

%PID szabályozó átviteli függvényének létrehozása:
A = 2; %arányos tag
Ai = 0; %integráló tag
Ad = 0; %differenciáló tag
C = pid(A, Ai, Ad); %ebben az esetben P szabályozó

%Hurokátviteli függvény:
Gh = C*G;

%A zárt rendszer átviteli függvénye (kiegészítő érzékenységi függvény):
Gz = Gh/(1+Gh);

%Átmeneti függvény:
figure; %diagramot készítünk
step(Gz);

%Időtartományi minőségi tulajdonságok
S = stepinfo(Gz,'SettlingTimeLimits', 0.05)

%A felnyitott hurok Bode-diagramja (pl. fázistartalék ellenőrzésére)
figure; %hogy külön ablakban jelenjen meg
bode(Gh);
```

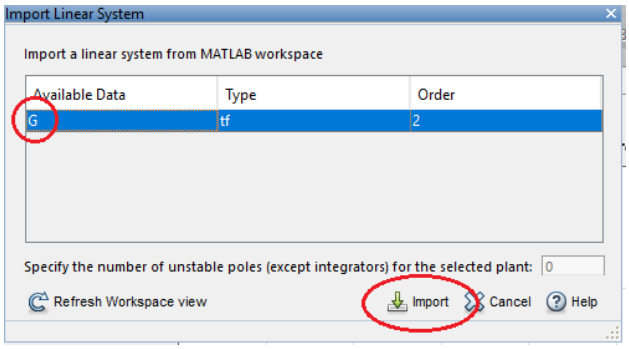
3.2 PID tuner

A *Matlab*-ban egyszerűen, intuitív módon hangolhatunk PID szabályozót lineáris rendszermodellek alapján. Ehhez a *Command window*-ból indítsuk el a *PID tuner*-t.
>>pidTuner

A megjelenő ablakban bal felső részén a „Plant” menüben válasszuk ki a Workspace-ből a BLDC motor átviteli függvényét. Ha nem létezik, hozzuk létre újra a *tf()* paranccsal!



9. ábra PID tuner kezdőablak

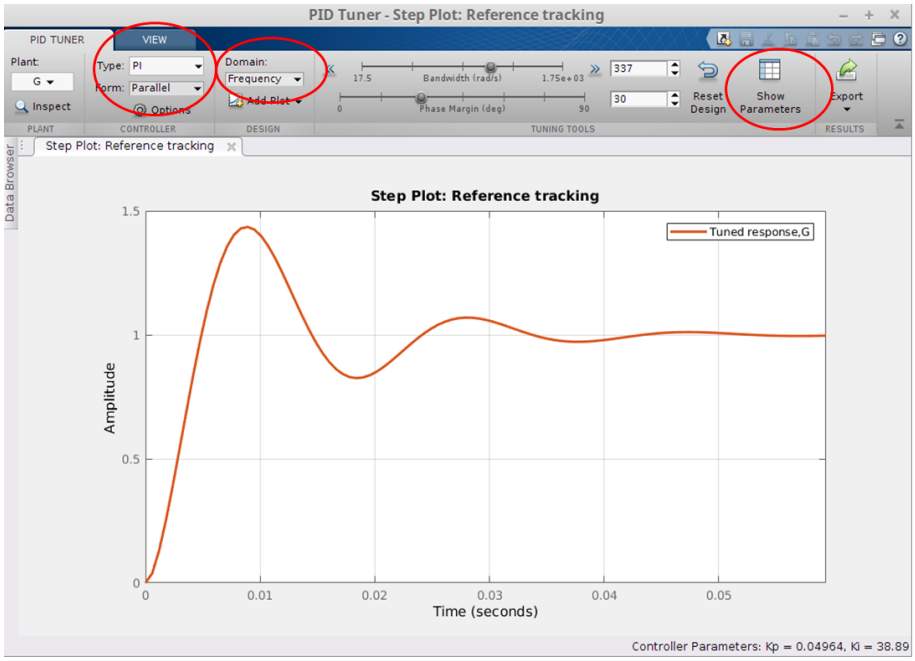


10. ábra Rendszermodell importálása

Ha sikeresen betöltöttük a szabályozni kívánt rendszer modelljét, válasszuk ki a felső sorból, hogy milyen típusú szabályozót szeretnénk. Kiválaszthatjuk továbbá, hogy idő- vagy frekvenciatartományi minőségi jellemzők alapján szeretnénk hangolni (Domain: Time/Frequency). A csúszkákkal vagy mellette a szövegdobozba írt értékekkel beállíthatók a kívánt minőségi jellemzők.

Megjegyzés: Az idő és a frekvenciatartományi jellemzők nem függetlenek egymástól!

Tipp: A tervezés során csak frekvencia-tartomány alapú hangolást alkalmazzon ("Domain: Frequency"), ahol rögzítve a megadott φ_t fázistartalékot ("Phase Margin") a sávszélességgel ("Bandwidth") tud hangolni!

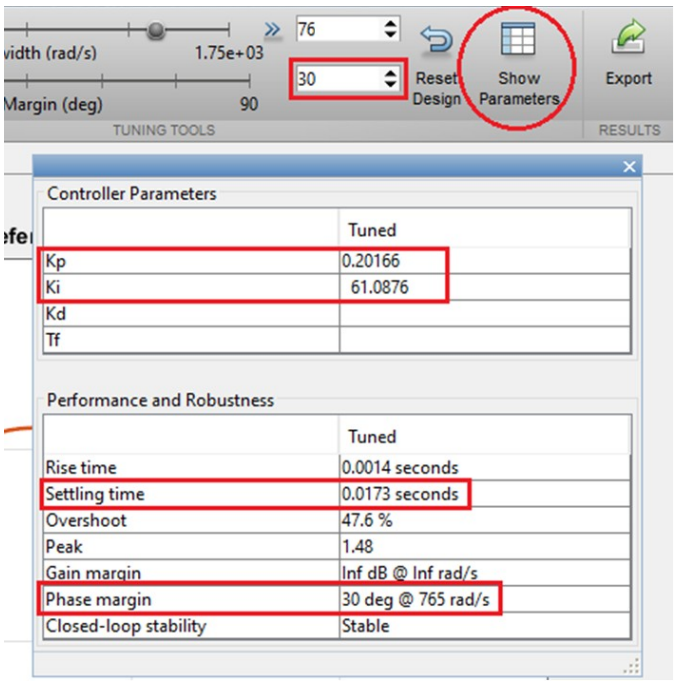


11. ábra A PID tuner javasolt beállításai

A *Show Parameters* gombra kattintva táblázatosan megjelennek a szabályozó paraméterei illetve a zárt rendszer minőségi tulajdonságai adott paraméterek mellett.

A hangolást javasolt menete:

- A *Show Parameters* gombbal lenyitott ablakot hagyjuk nyitva!
- Rögzítsük a Moodle-rendszer által előírt φ_t fázistartalékot ("Phase Margin")!
- Módosítjuk a sávszélesség ("Bandwidth") értékét (miközben a fázistartalékot ("Phase Margin") nem változtatjuk) addig, amíg az előírt szabályozási időt ("Settling Time") el nem érjük.
- A sikeres hangolást követően a „Controller Parameters" értékek - tehát a P és I szabályozó tagok erősítései - kiolvashatók (lásd alább) és beírhatók a Moodle-felületre.

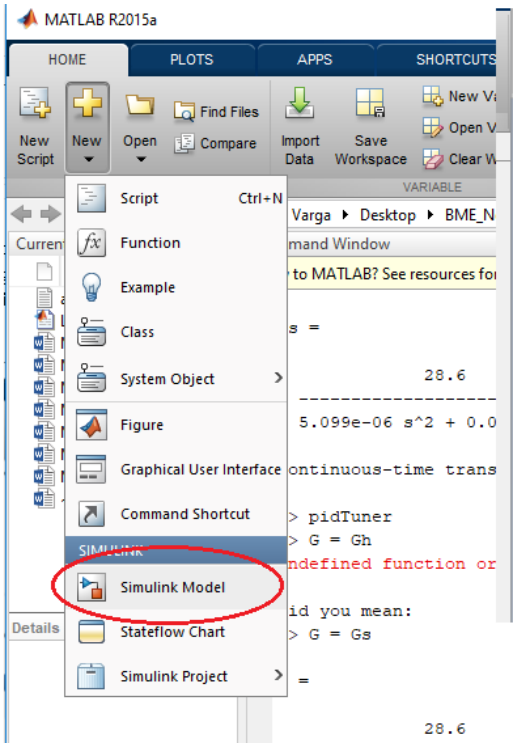


12. ábra PID hangolása

Lehetőség van további plot-ok (pl. Bode diagram) hozzáadásához is a PID Tunerben az „Add Plot” gombbal, így vizsgálható a szabályozott rendszer frekvenciatartományban is.

3.3 Matlab Simulink-ben történő modellezés – E-laboron kívüli szorgalmi feladat érdeklődőknek

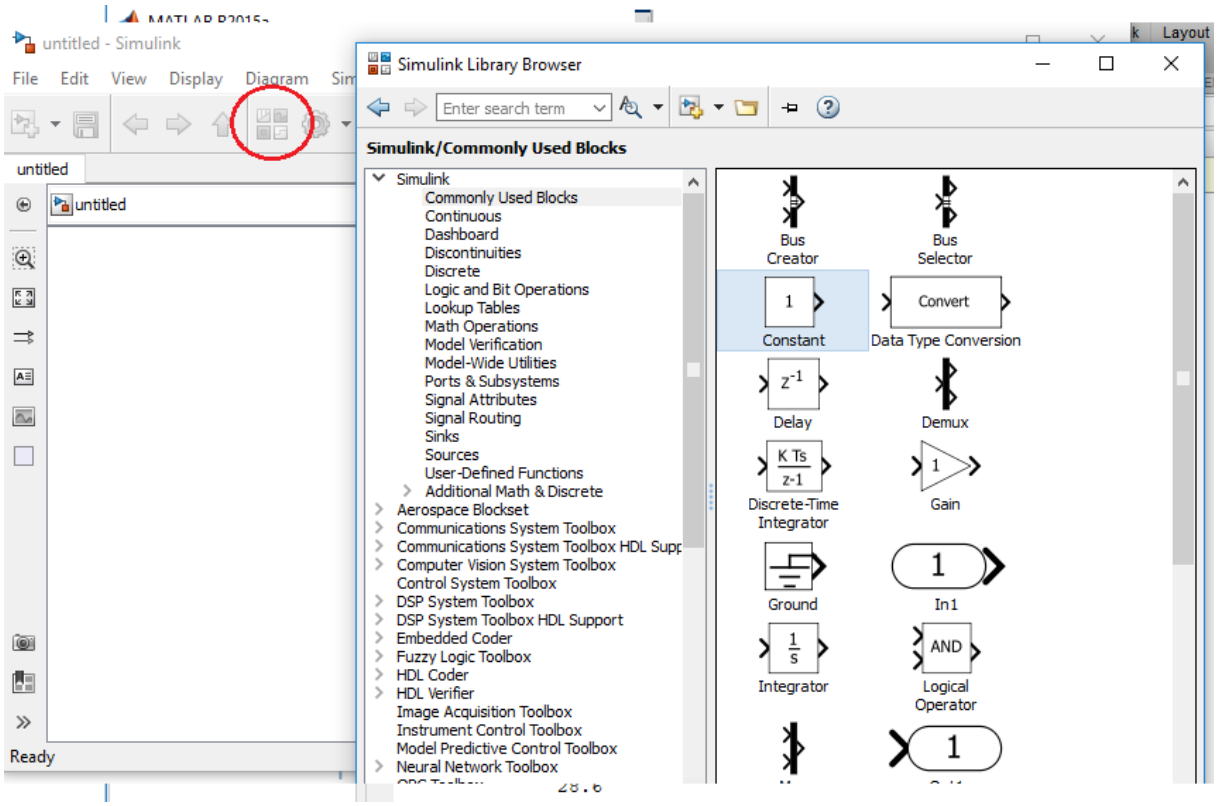
A szabályozott rendszer modellje létrehozható a Matlab *Simulink* környezetében is. A rendszer itt tetszőleges bemenő jelre vizsgálható időtartományban.
A *Simulink* megnyitása a *New* menüből a *Simulink Model* gombbal. Ez picit több időt vehet igénybe.



13. ábra Simulink megnyitása

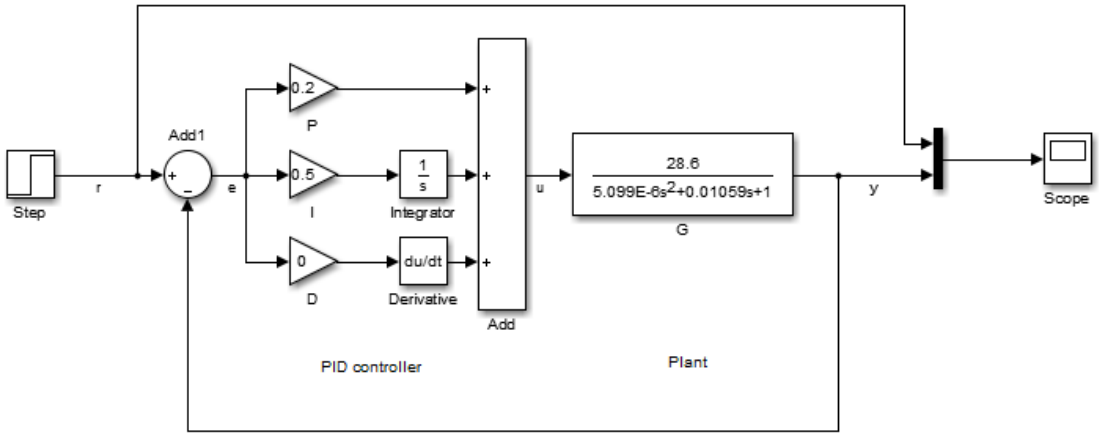
A 14. ábrán jelölt ikonra kattintva nyissuk meg a *Simulink Library Browser*-t.
A különböző blokkok (pl. összeadás, logikai műveletek, átviteli függvény stb.) innen választhatók. A bal oldali oszlopban van a Blockset-ek listája. A Blockset neve melletti + ikonra kattintva megnyílik (a Blockset neve alatt) a Blockset-en belüli könyvtárak listája (az ábrán éppen a *Simulink* alap Blockset-hez tartozó lista látható). A listában egy adott könyvtárra kattintva annak tartalma (a blokkok) megjelenik a *Library Browser* jobb oldali ablakában. A blokkok az üres modellbe drag and drop technikával vihetők át (az egérmutatót az adott blokkra állítjuk, majd a bal gombot lenyomva tartva a blokkot áthúzzuk a modellbe).

A blokkok összekötése a következő módon történik: egy blokk kimenetét a másik blokk bemenetével a kimeneten bal gombbal kattintva, majd a „vezetéket” lenyomott bal gomb mellett a másik blokk bemenetéig húzva tudjuk megtenni. Egy már meglévő vezetékről leágaztatást (úgynevezett branch összekötés) a „vezeték” adott pontjára az egér jobb gombjával kattintva és lenyomott jobb gomb mellett a „vezetéket” a cél blokk bemeneti pontjáig húzva tudunk létesíteni.



14. ábra Simulink Library Browser

Az alábbi modellt hozzuk létre:

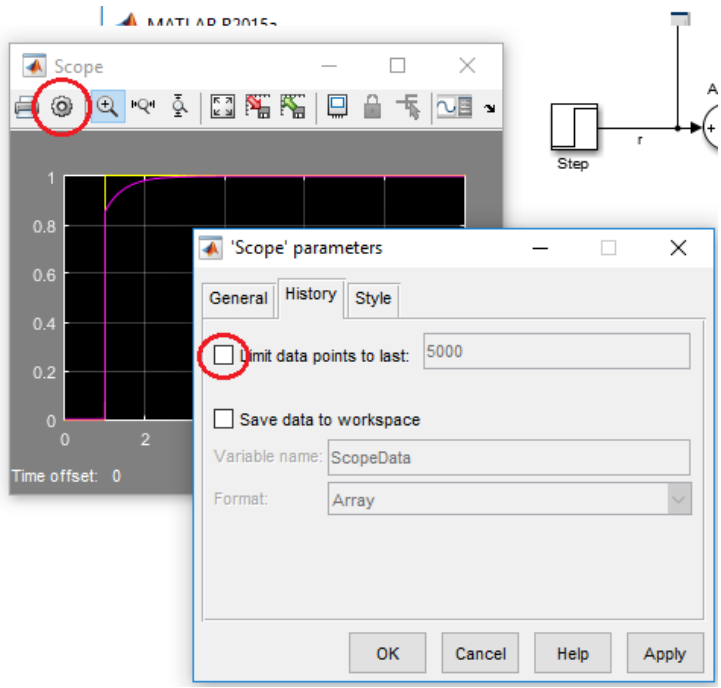


15. ábra Simulink modell

Ehhez az alábbi blokkokra van szükség:

- Sources/Step: egységugrás bemenet. Duplán rákattintva beállíthatjuk, a szimuláció hányadik másodpercében és mekkora ugrás jöjjön.
- Math Operations/Add: Beállítható a blokk alakja illetve a bemenetek száma.
- Math Operations/Gain: Konstanssal megszorozza a bemenő jelet. Ezeket használjuk a P, I és D erősítések megadására.
- Contionous/Integrator: Integráló blokk.
- Contionsous/Derivative: Deriváló blokk.
- Continous/Transfer function: Ebben az átmeneti függvényben adható meg a BLDC motor modellje [num], [den] alakban, hasonlóan, mint a *tf()* függvény esetén.

- Signal routing/Mux: Megadott számú jelet egy vektorba rendez. Azért kell, hogy a Scope-on egy diagramban lehessen látni a referenciát és a kimenetet.
- Sinks/Scope: Diagram ablak. Duplán rákattintva megnyílik a diagram. Felül a fogaskerékre kattintva érdemes kikapcsolni a Limit data points beállítást.



16. ábra Simulink Scope

Felül a **Run** gombra kattintva a szimuláció elindítható.

Feladat: megnézni a rendszer viselkedését különböző bemenő jelekre (Sources/...) a mérés során meghatározott szabályozó paraméterekkel.