Orvosbiológiai számítógépes gyakorlatok

(BMEVITMM203)

Mérési jegyzőkönyv

Orvosbiológiai rendszerek identifikációja

Készítették:

Jánosa Dávid Péter (FDSA7Y)

Mokánszki Béla (FA8YEZ)

Veres Dániel Sándor (GLZPT9)

2014. február 27.

Tartalom

Bevezető....2

A mérés célja 2

A méréshez felhasznált eszközök 3

1. mérés: Rendszer identifikáció....4

   A rendszer leírása 4

   Az identifikáció menete....8

   ARX identifikáció....9

   IV4, avagy ARX identifikáció segédváltozóval....11

   ARMAX identifikáció....13

   OE identifikáció 16

   Box-Jenkins modell....17

   PEM identifikáció 19

   Az eredmények értékelése 21

2. mérés: Kolloidos májfunkció mérésből modellezése....23

   A mérés leírása 23

   Az eredmények értékelése 25

A mérés értékelése....26

Függelék – MATLAB kódok....27

   Rendszer leírása 27

   ARX Identifikáció....28

   Box-Jenkins identifikáció 29

   Clearance vizsgálat....31

Bevezető

Identifikáció alatt azonosítást értünk. Ez egy adott, ismeretlen rendszer modelljének felépítését, meghatározását jelenti rendszertechnikai értelemben. Ez a rendszer sokféle lehet, akár egy járó inga, vagy egy vadászrepülő is. Az identifikáció biológiai rendszerek esetében is megkerülhetetlen bizonyos folyamatok, összetevők, szervek modelljének felállításához. A cél olyan modell alkotása, amely leírja rendszer időbeli viselkedését.
Identifikáció esetén két alapesetet különítünk el: black box identifikáció esetén semmilyen előzetes ismeretünk nem áll rendelkezésre az azonosítandó rendszerről, míg gray box identifikáció esetén viszont vannak szerkezeti ismereteink, és az identifikáció során ennek az alapmodellnek a paramétereit próbáljuk meghatározni.
A gyakorlatban sokszor keveredik a kettő, ilyenkor egy ismeretlen rendszer meghatározásakor több eltérő gray box modellt parametrizálunk, majd megnézzük, ezek közül valamelyik képes-e kellő pontossággal leírni a rendszerünket.

A mérés célja

A mérés célja, hogy megismertessen az identifikáció alapvető formáival, sajátosságaival, és egy konkrét orvosbiológiai alkalmazáson keresztül szemléltesse az egyes módszereket. Az 1. feladatban egy ismeretlen rendszer vegyes identifikációját kell elvégezni, azaz a black box rendszerünket több, eltérő modell parametrizálásval próbáljuk leírni. E során össze kell hasonlítani a különböző modellek által szolgáltatott eredményeket, valamint meg kell vizsgálni az egyes modellekben a modellt leíró egyenletek fokszám változtatásának hatásait. Ezek után a 2. feladatban egy konkrét ^{198}Au kolloidos májfunkció mérésből származó adatok alapján történő paraméterbecslést kell végrehajtani és a mérés során el kell végezni a görbék identifikációját LS kritérium szerinti illesztés alapján, majd a kapott eredményeket értékelni kell.

A méréshez felhasznált eszközök

A mérés során az identifikáció elvégzéséhez MATLAB 2006b programcsomagot használtunk, amely a Campus licenszprogram keretében elérhető az egyetemen.
A Clearence vizsgálathoz az alábbi címen elérhető adatsort használtuk:
https://www.iit.bme.hu/sites/default/files/Meres2_adatsor.zip

1. mérés: Rendszer identifikáció

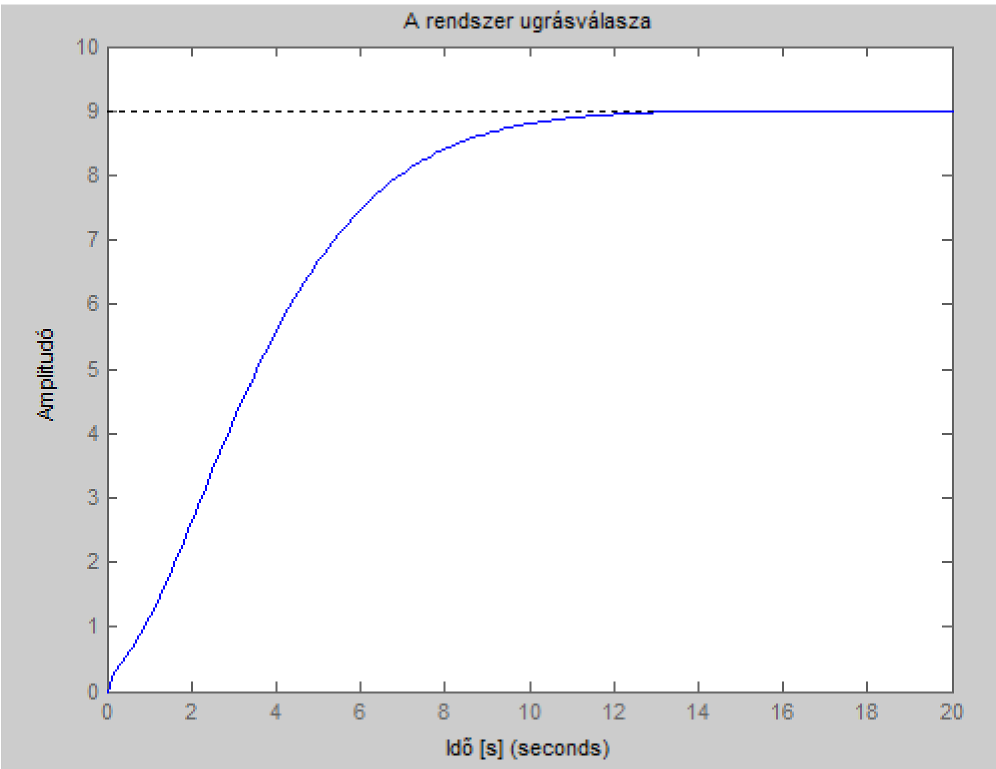
A rendszer leírása

A mérés során egy ismeretlen rendszer modelljét kíséreljük meg felépíteni, illetve meghatározni. Először létrehoztuk az alábbi rendszert:

$$W(s) = \frac{B(s)}{A(s)} = \frac{2s^2 + 3s + 9}{1s^3 + 5s^2 + 4s + 1}$$

A MATLAB konvenciója szerint a polinomokat együtthatóikkal kell megadni z hatványainak csökkenő sorrendjében, így esetünkben: m=[2 3 9]; n=[1 5 4 1].

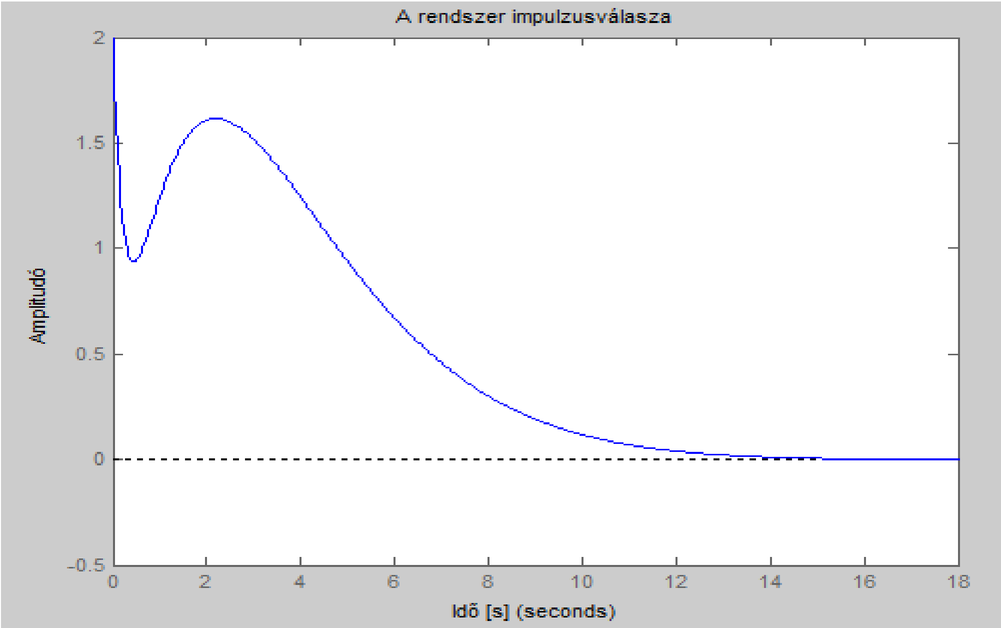
Először a step függvény használatával megvizsgáltuk rendszerünk ugrásválaszát, ezt az 1. ábra mutatja:



1. ábra: A rendszer ugrásválasza

Ez alapján a rendszerünk stabil, DC erősítése pedig 9.

Ezután megvizsgáltuk a rendszerünk impulzusválaszát is, az impulse függvény segítségével.



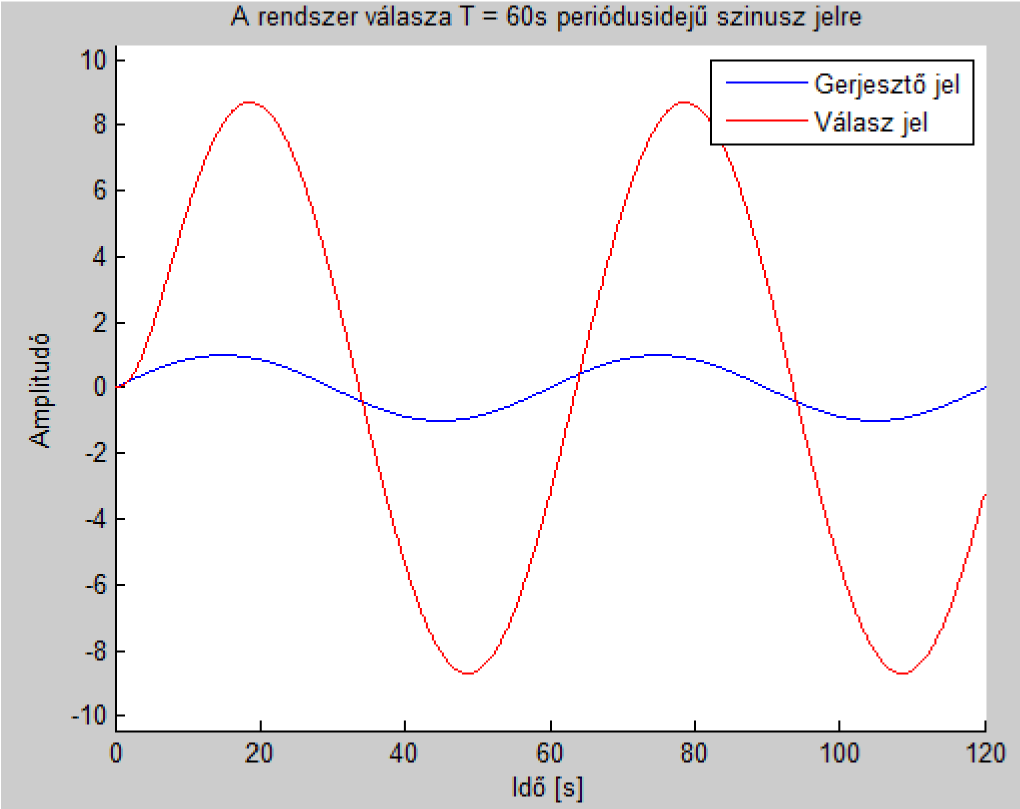
2. ábra: A rendszer impulzusválasza

A következő lépésben rendszerünket szinuszos és négyszög gerjesztő jellel vizsgáltuk. Ehhez először létre kellett hozni a vizsgáló jeleket. Fontos volt a megfelelő periódus idő (esetünkben T=60 sec, ami soknak tűnhet, de ekkor a válasz fél perióduson belül stabilan beáll, ami majd könnyebbé teszi az identifikációt) illetve a megfelelő mintavételezési idő (esetünkben Ts=0,1 sec).

MATLAB kódok:

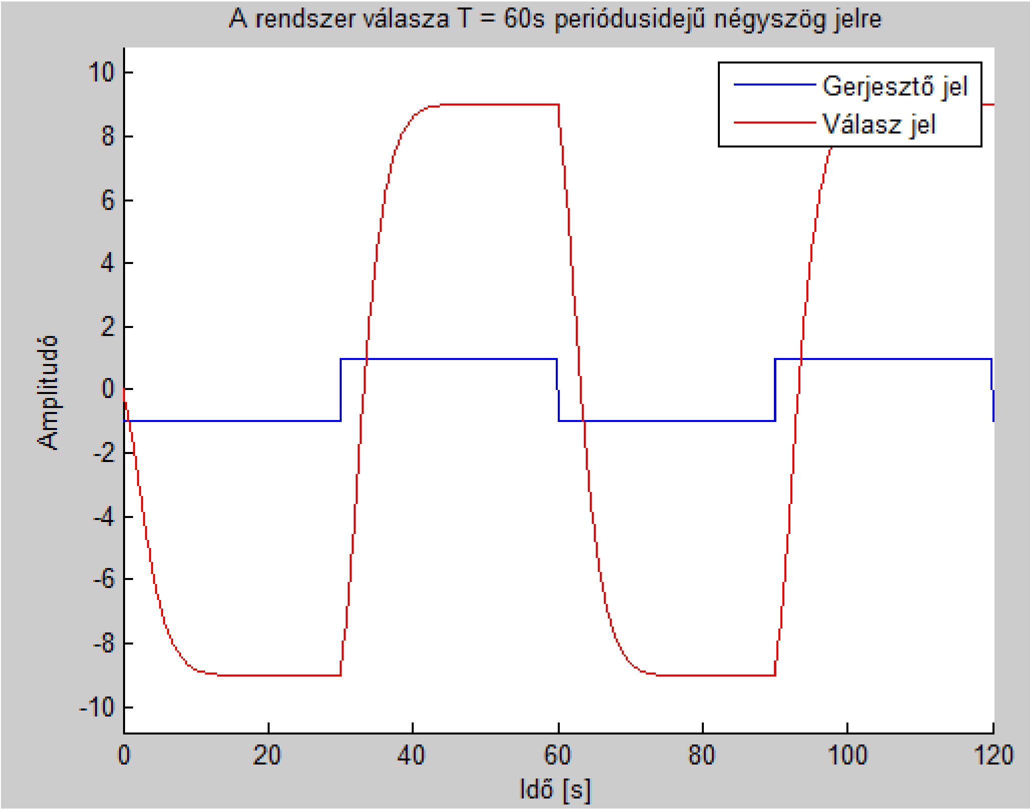
```
T = 60;              % Periódus idő [s].
Tmax = 120;          % Hossz [s] %
Ts = 0.1;            % Mintavételezési idő [s] %
[sin_input,t] = gensig('sin',T,Tmax,Ts);        % Szinuszos bemeneti jel %
[square_input] = gensig('square',T,Tmax,Ts);    % Bemeneti négyszögjel %
% Hogy 0.5 és -0.5 között legyen a jel, különben DC terhelt %
square_input = square_input-0.5;
% Hogy 1 és -1 között legyen, mert ez a standard %
square_input = square_input*2;
```

A válaszok szimulálásához a MATLAB lsim függvényét használtuk, mely LTI modellek válaszát képes szimulálni. A gerjesztő jelet és a választ a 3. ábrán láthatjuk:



3. ábra A rendszer válasza szinuszos gerjesztésre

Ezek után négyszögjeles gerjesztésre vizsgáltuk a rendszerünket. Az lsim szimuláció alapján a gerjesztést és a választ a 4. ábra mutatja:

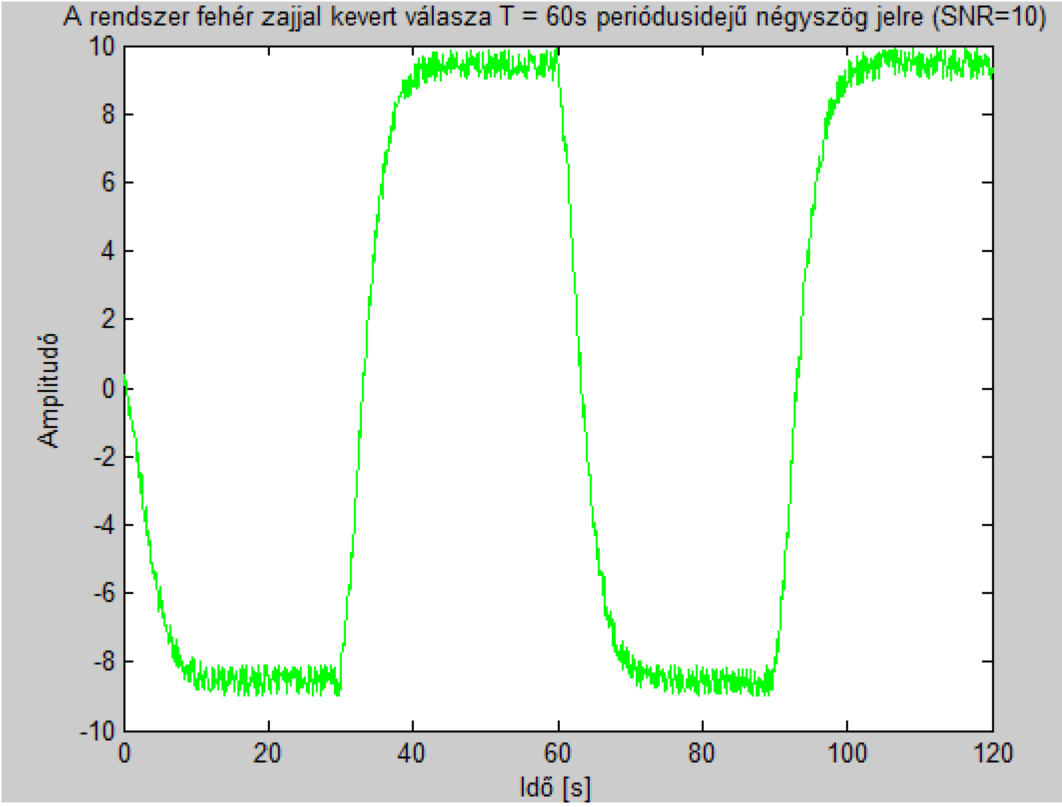


4. ábra: A rendszer válasza négyszög jelre

<u>MATLAB kódok:</u>

```
szinusz_valasz = lsim(sys1,sin_input,t);         % A szinusz jelre adott válasz szimulálása %
negyszog_valasz = lsim(sys1,square_input,t);     % A négyszög jelre adott válasz szimulálása %
```

A következő lépésben ehhez a négyszög jelhez egyenletes eloszlású, fehér zajt adtunk, melynek amplitúdója 0,9, így a jel és a zaj amplitúdó aránya nagyjából 10 dB. A későbbiekben ezen, zajosított válaszból és a tiszta négyszög gerjesztőjelből fogjuk újra identifikálni a rendszert. Az 5. ábra mutatja a rendszer fehér zajjal kevert válaszát T=60 s periódus idejű négyszög jelre:



5. ábra: A rendszer fehér zajjal kevert válasza

Az identifikáció menete

Ezek után a MATLAB által biztosított identifikációs függvényeket használva végeztük el az azonosítást.

A metódusunk lényege, hogy identifikáljuk a rendszer ARX, IV4, ARMAX, OE, BJ és PEM modell alapján is, és keressük azt, melynek válasza és az identifikációs bemenetként szolgáló, eredeti, zajosított válasz közti négyzetes eltérés a legkisebb.
Ezen modellek azonban önmagukban is elég eltérőek a szerint, hogy az őket leíró polinomok fokszáma és a modell késleltetése milyen. Hogy megtaláljuk az ideális fokszámokat és késleltetést, minden modell esetén leszimuláljuk egy választott fokszám és késleltetés halmaz minden kombinációját. A kapott modellek közül azt fogadjuk el legjobbnak, melynek válasza négyzetes eltérés alapján a leginkább hasonlít a kiindulásként felhasznált, zajosított válaszjelhez. A költségfüggvény minimumát, azaz a legkisebb négyzetes eltérést a továbbiakban **LS_min**-nel jelöljük.

Ez tehát azt jelenti, hogy egy adott paraméterhalmazt az algoritmus úgy tapogat le, hogy annak minden elemét leszimulálja és elvégzi a négyzetes költségfüggvény számítást. Ez meglehetősen számítás és időigényes, feltehetően létezne ennél jobb megoldás is. Azonban a gyakorlat keretein belül fontosabb, hogy gyorsan használható kódot fejlesszünk, így bár jelen megoldásnak éles határt szab a gép számítási teljesítménye, összességében elfogadható idő alatt pontos végeredményt ad.

ARX identifikáció

Az ARX egy olyan modell, ami külső bemenő jelet tartalmaz és autoregresszív, ami annyit jelent, hogy az aktuális kimenet a korábbi kimenetektől is függ.

$$\boldsymbol{A(z^{-1})y(t) = B(z^{-1})u(t-n) + e(t)}$$

A MATLAB az ARX identifikációt az arx fügvénnyel végzi, ennek pareméterei:

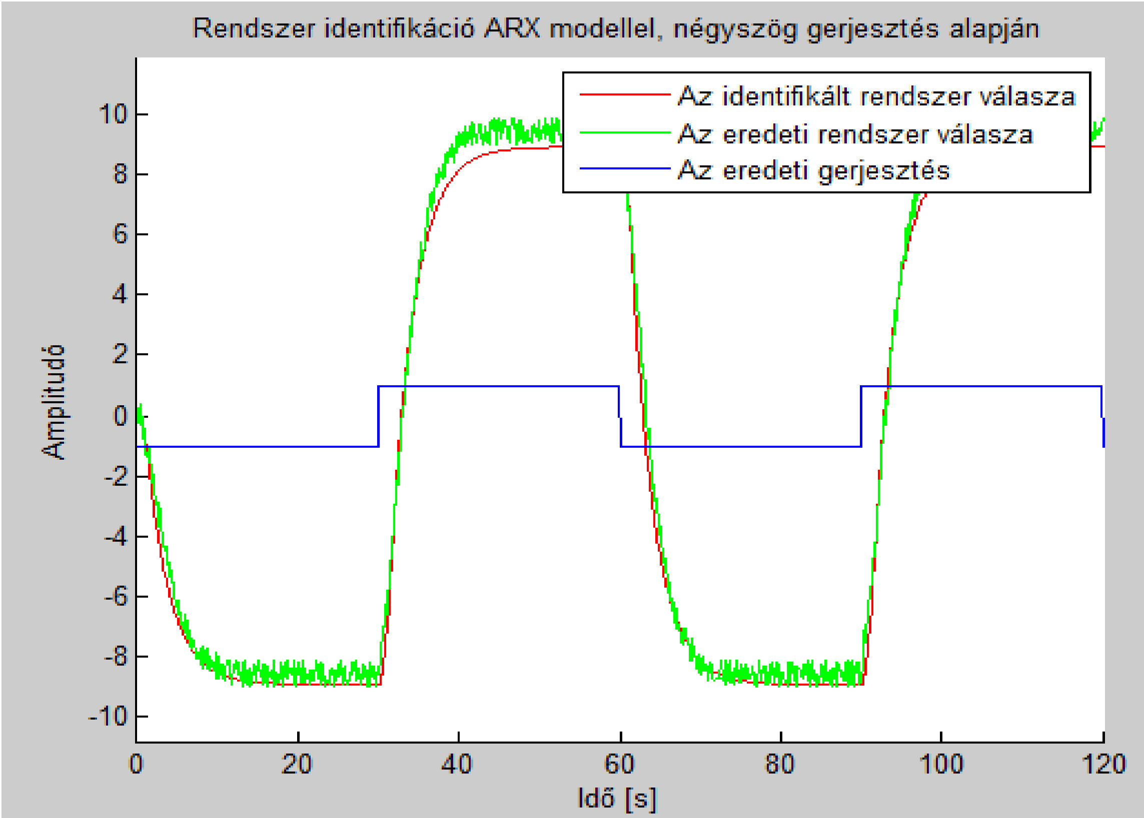
- $\mathbf{n_a}$, mely az A polinom maximális fokszáma. A letapogatás során ez 0-tól 9-ig változott.
- $\mathbf{n_b}$, mely a B polinom maximális fokszáma. A letapogatás során ez 1-től 10-ig változott.
- $\mathbf{n_k}$, mely a késleltetés paramétere. A letapogatás során ez 0-tól 9-ig változott.

Az idetartozó MATLAB kódokat a függelékbe helyeztük, annak terjedelmes mivoltából fakadólag. Ez a kód egy egyszerűbb, könnyebben érthető változata a későbbi általános paraméter letapogató kódoknak, melyekből a legkomplexebb (Box-Jenkins letapogatás) szintén a melléklet része.

Az ARX identifikáció során először is kiszámoltuk az identifikált rendszerek válaszának az eredeti, zajos válaszhoz képest vett négyzetes eltérését, ami a költségfüggvényünk. Miután ezt minden egyes modell paraméterkombinációra megtettük megkerestük költségfüggvény minimumát, ezt követően a minimum helye alapján visszakerestük az identifikáció paramétereit.

A paraméterekkel ezután létrehoztuk az optimális ARX rendszert és ábrázoltuk annak tulajdonságait.

A rendszer válasza összevetve az identifikáció eredeti válasszal látható a 6. ábrán:



6. ábra: ARX modellel történő identifikáció

Látható, hogy az identifikált rendszerünk válasza jól közelíti eredeti rendszerünket. Az ARX modell alapján történt identifikáció paraméterei:

LS_min = 452.7412
n_a = 1
n_b = 10
n_k = 4

Az ARX modell alapján történt identifikáció eredménye:

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 0.9665 q^-1

B(q) = 0.1617 q^-4 - 0.1607 q^-5 + 0.1221 q^-6 + 0.1949 q^-7 - 0.3486 q^-8
+ 0.3201 q^-9 - 0.1745 q^-10 - 0.0567 q^-11 + 0.03366 q^-12 + 0.207 q^-13

Estimated using ARX from data set to_ident
```

```
Loss function 0.118989 and FPE 0.121169
Sampling interval: 0.1
```

IV4, avagy ARX identifikáció segédváltozóval

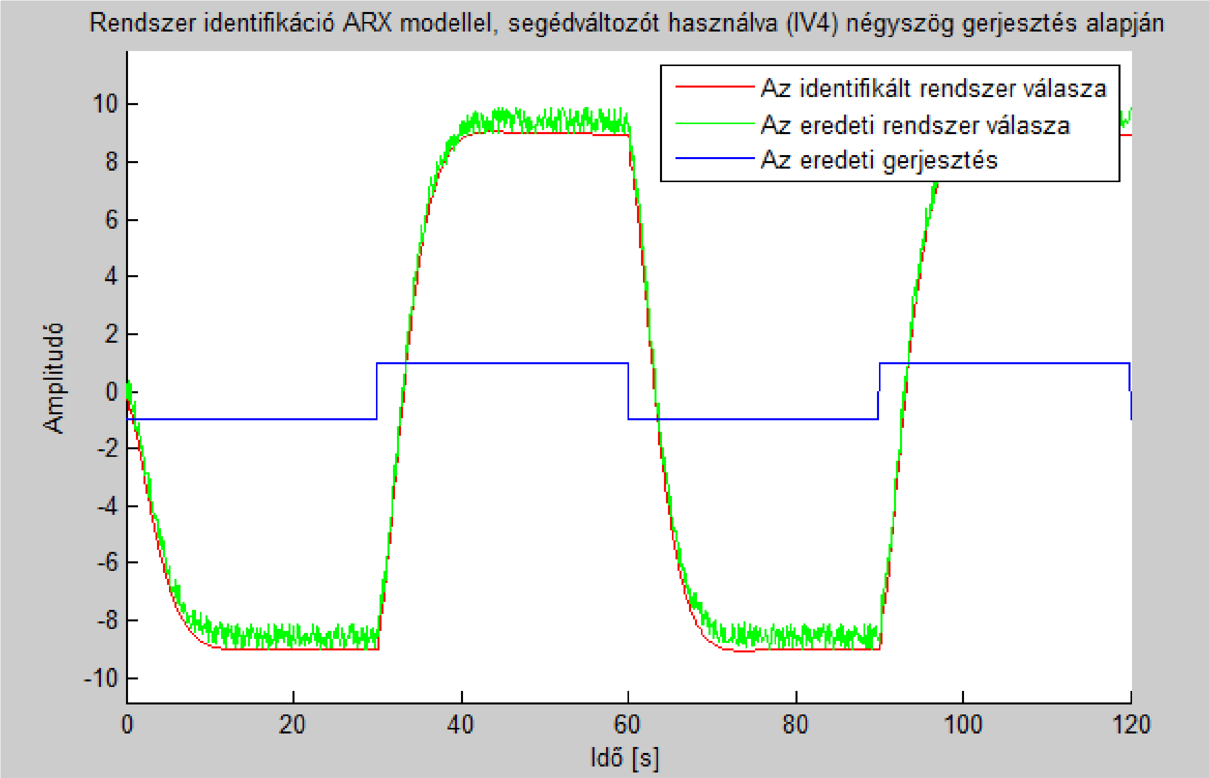
A segédváltozók módszere (IV4-Instrumentam variables) tekinthető az LS módszer általánosításának. Ebben az esetben eltérő módon definiáljuk a költségfüggvényt:

$$\boldsymbol{J} = \sum_{\boldsymbol{k}=\mathbf{1}}^{\boldsymbol{K}} \boldsymbol{z}\,[\mathbf{k}]\varepsilon[\mathbf{k}] = \mathbf{0}$$

$\mathrm{z}[\mathbf{k}]\varepsilon[\mathbf{k}]$ megfelelő megválasztásával az ARX modell segédváltozós meghatározását kapjuk. Ennek elméleti hátterét a https://www.iit.bme.hu/sites/default/files/Meres2_identifikacio.pdf címen található tanszéki jegyzet részletezi. Gyakorlati szempontból csupán a MATLAB iv4 függvényére van szükségünk. A paraméterek letapogatása és az optimális paraméterek meghatározása az ARX modellhez hasonlóan történik. A paraméterek és a keresési tartomány:

- $\mathbf{n_a}$, mely az A polinom maximális fokszáma. A letapogatás során ez 0-tól 9-ig változott.
- $\mathbf{n_b}$, mely a B polinom maximális fokszáma. A letapogatás során ez 1-től 10-ig változott.
- $\mathbf{n_k}$, mely a késleltetés paramétere. A letapogatás során ez 0-tól 9-ig változott.

Az optimális IV4 rendszer válasza látható az eredeti válasszak összevetve a 7. ábrán:



7. ábra: A rendszer identifikálása IV4 módszerrel

Az IV4 esetén is jól látható, hogy az identifikált rendszerünk válasza, követi az eredeti rendszer gerjesztés utáni válaszát.

Az IV4 modell alapján történt identifikáció paraméterei:

LS_min = 313.3601
n_a = 7
n_b = 6
n_k = 2

Az IV4 modell alapján történt identifikáció eredménye:

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + e(t)
A(q) = 1 - 0.2344 q^-1 - 1.005 q^-2 - 0.3286 q^-3 + 0.2436 q^-4 + 0.2826
q^-5 - 0.1233 q^-6 + 0.1753 q^-

B(q) = 0.1212 q^-2 + 0.3416 q^-3 - 0.06557 q^-4 - 0.4709 q^-5  - 0.2262 q^-
6 + 0.3889 q^-7

Estimated using IV4 from data set to_ident
Loss function 0.144774 and FPE 0.147908
Sampling interval: 0.1
```

Érdekes, hogy bár az IV4 is ARX modellt keres, de az eltérő módszer miatt a kapott rendszer szerkezete lényegesen eltér az ARX eredményétől, négyzetes hibája pedig annak nagyjából 75%-a.

ARMAX identifikáció

Külső bemenőjelet tartalmazó autoregresszív modell (a zajra vonatkozó) mozgóátlagolással:

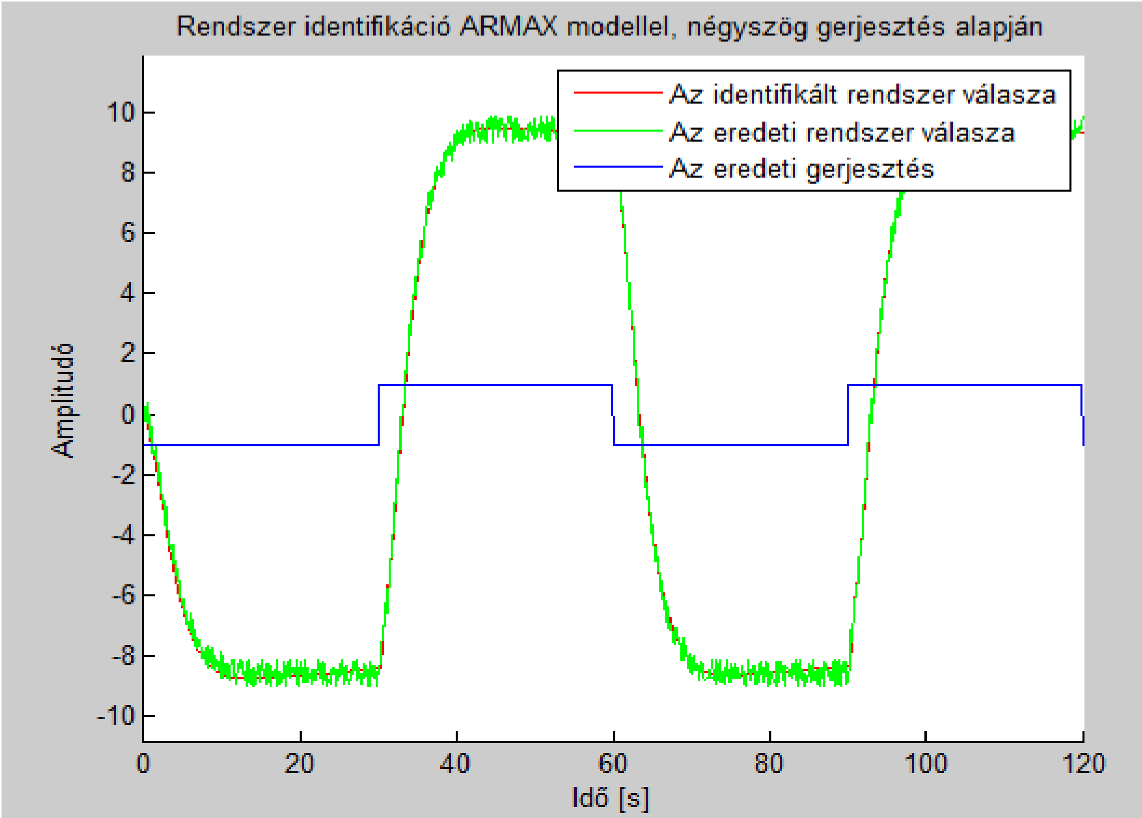
$$\boldsymbol{A}(\boldsymbol{z}^{-1})\boldsymbol{y}(\boldsymbol{t}) = \boldsymbol{B}(\boldsymbol{z}^{-1})\boldsymbol{u}(\boldsymbol{t}-\boldsymbol{n}) + \boldsymbol{C}(\boldsymbol{z}^{-1})\boldsymbol{e}(\boldsymbol{t})$$

A paramétertér letapogatásának elve az ARX modellhez hasonló, azonban a több paraméter és az ARX modell identifikálásának időigényesebb volta miatt a letapogatott paramétertér kisebb. Azért, hogy a paraméterteret empirikusan tudjuk a lehető legtágabbra húzni, a MATLAB kódot általánosítottuk, így kézzel beállítható, mely paramétert meddig szeretnénk futtatni. A jegyzőkönyv végi függelékben található BJ identifikáló kód ugyanezt az általánosított elvet használja.

A paraméterek és a keresési tartomány:

- **n_a**, mely az A polinom maximális fokszáma. A letapogatás során ez 0-tól 6-ig változott.
- **n_b**, mely a B polinom maximális fokszáma. A letapogatás során ez 1-től 6-ig változott.
- **n_c**, mely a C polinom maximális fokszáma. A letapogatás során ez 0-tól 6-ig változott.
- **n_k**, mely a késleltetés paramétere. A letapogatás során ez 0-tól 6-ig változott.

Ezt követően elvégezzük ismét az identifikációt a legjobb paraméterekkel, így a 8. ábrán látható eredményt kaptuk:



8. ábra: A rendszer identifikálása ARMAX modellel

A zajjal terhelt eredeti rendszerünk válaszához jól illeszkedik az identifikált rendszerünk válasza.

Az ARMAX modell alapján történt identifikáció paraméterei:

LS_min = 94.3805

n_a = 5

n_b = 6

n_c = 6

n_k = 5

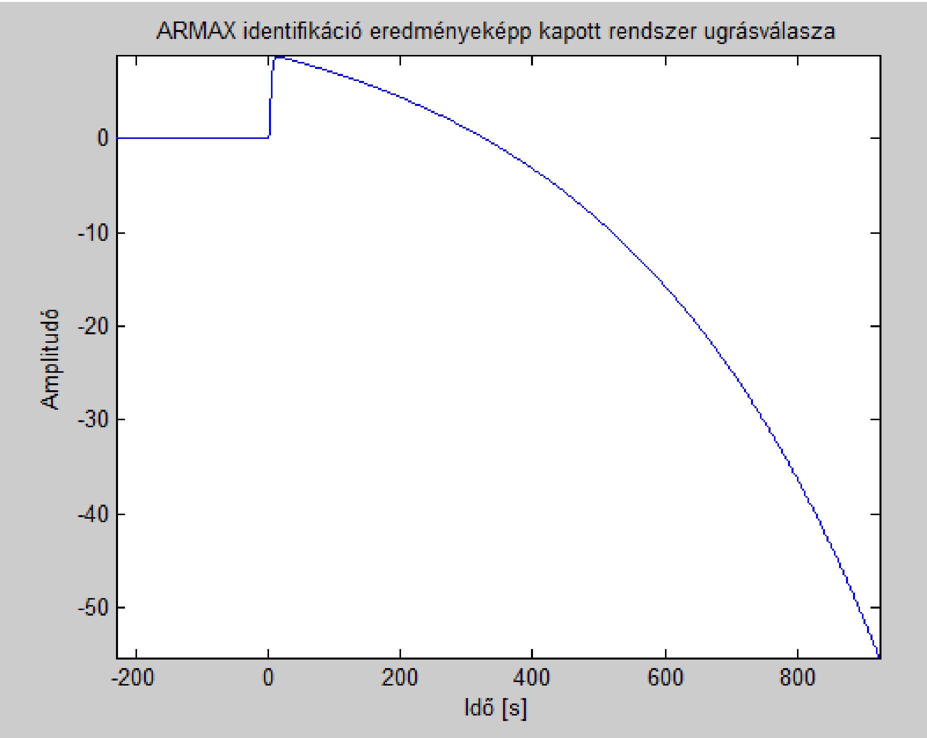
Az ARMAX modell alapján történt identifikáció eredménye:

```
Discrete-time IDPOLY model: A(q)y(t) = B(q)u(t) + C(q)e(t)
A(q) = 1 - 2.192 q^-1 + 0.6063 q^-2 + 1.496 q^-3 - 1.039 q^-4+ 0.1286 q^-5

B(q) = 0.4146 q^-5 - 0.7595 q^-6 + 0.2175 q^-7 - 0.003915 q^-8 + 0.3563 q^-
9 - 0.225 q^-10

C(q) = 1 - 2.092 q^-1 + 0.4797 q^-2 + 1.372 q^-3 - 0.8372 q^-4 + 0.1168 q^-
5 - 0.03904 q^-6
```

```
Estimated using ARMAX from data set z
Loss function 0.069178 and FPE 0.0711713
Sampling interval: 0.1

Warning: Model is unstable.
> In warning at 26
  In idmodel.sim at 359
  In create_system at 324
```

Itt a MATLAB figyelmeztet minket, hogy a **rendszerünk nem stabil.**

Ezt mutatja az ugrásválasz is, mely elszáll a mínusz végtelenbe:



9. ábra: A rendszer identifikálása ARMAX modellel

OE identifikáció

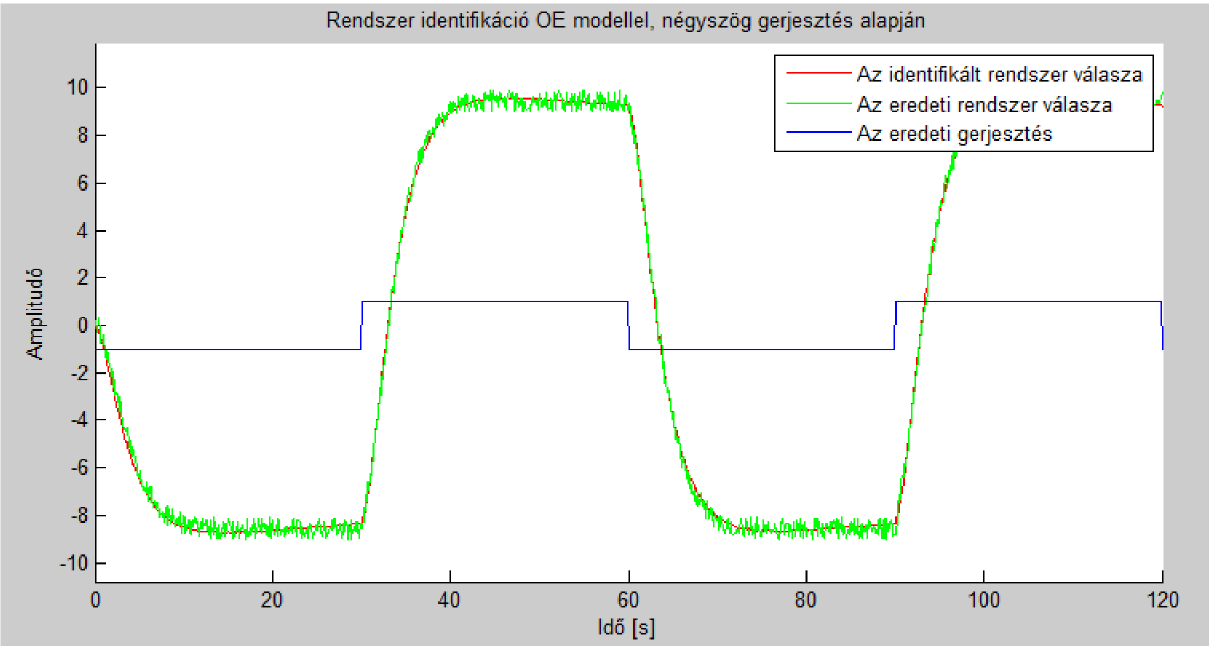
Kimenőjelre redukált, additív zajt tartamazó modell:

$$y(t) = \frac{B(z^{-1})}{F(z^{-1})} u(t-n) + e(t)$$

Az identifikációs paraméterek és azok letapogatási tartománya:

- n_b, mely a B polinom maximális fokszáma. A letapogatás során ez 1-től 6-ig változott.
- n_f, mely a F polinom maximális fokszáma. A letapogatás során ez 0-tól 6-ig változott.
- n_k, mely a késleltetés paramétere. A letapogatás során ez 0-tól 6-ig változott.

A kapott, legjobb négyzetes költséggel rendelkező kimenetű rendszer válaszát mutatja a 10. ábra:



10. ábra: A rendszer identifikálása OE modellel

Az identifikált rendszer válasza jól illeszkedik ebben az esetben is az eredeti rendszer zajjal terhelt válaszához.

Az OE modell alapján történt identifikáció paraméterei:

LS_min = 100.0996

n_b = 6

n_f = 6

n_k = 1

Az OE modell alapján történt identifikáció eredménye:

```
Discrete-time IDPOLY model: y(t) = [B(q)/F(q)]u(t) + e(t)
B(q) = 0.1441 q^-1 - 0.2077 q^-2 + 0.0563 q^-3 + 0.0102 q^-4  + 0.07049 q^-
5 - 0.0734 q^-6
F(q) = 1 - 0.7208 q^-1 - 1.343 q^-2 - 0.09543 q^-3 + 1.258 q^-4 + 0.763 q^-
5 - 0.8615 q^-6

Estimated using OE from data set to_ident
Loss function 0.0836169 and FPE 0.0853104
Sampling interval: 0.1
```

Box-Jenkins modell

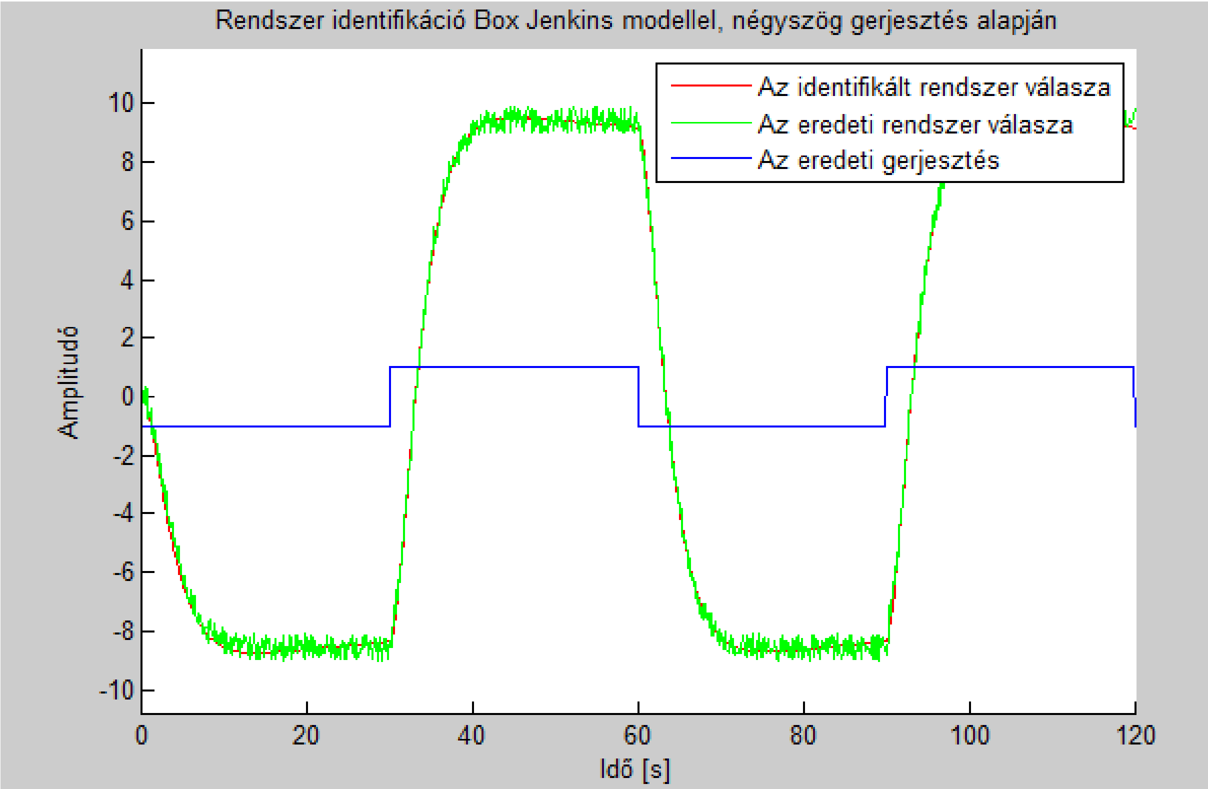
$$y(t) = \frac{B(z^{-1})}{F(z^{-1})}u(t-n) + \frac{C(z^{-1})}{D(z^{-1})}e(t)$$

Az eddigieknek megfelelően ismét kerestük a legkisebb négyzetes eltérést és ezen keresztül a legjobb Box-Jenkins (BJ) modellt. A paramétereink és a paramétertér az alábbi:

- **n_b**, mely a B polinom maximális fokszáma. A letapogatás során ez 1-től 6-ig változott.
- **n_c**, mely a C polinom maximális fokszáma. A letapogatás során ez 0-tól 4-ig változott.
- **n_d**, mely az A polinom maximális fokszáma. A letapogatás során ez 0-tól 4-ig változott.
- **n_f**, mely az A polinom maximális fokszáma. A letapogatás során ez 0-tól 6-ig változott.
- **n_k**, mely a késleltetés paramétere. A letapogatás során ez 0-tól 5-ig változott.

Látható, hogy ez a paramétertér jóval szűkebb, mint a korábbiak. Enneok oka, hogy a BJ identifikáció használja az eddigi legtöbb paramétert, és futása önmagában is a leglassabb. Jelen paramétertér teljes letapogatása Intel Core i5 M560 @2,67 GHz processzorral, 4 GB munkamemória mellett, 64 bites MATLAB-bal nagyjából 20 perc.

A legjobb paraméterekkel identifikált rendszer válaszát mutatja a 11. ábra:



11. ábra: A rendszer identifikálása Box-Jenkins modellel

Az identifikált rendszerünk válasza jól illeszkedik az eredeti rendszer válaszához.
A Box Jenkins modell alapján történt identifikáció paraméterei:

LS_min = 96.9211

n_b = 2

n_c = 1

n_d = 0

n_f = 6

n_k = 2

A Box Jenkins modell alapján történt identifikáció eredménye:

```
Discrete-time IDPOLY model: y(t) = [B(q)/F(q)]u(t) + C(q)e(t)
B(q) = 0.06755 q^-2 - 0.06757 q^-3

C(q) = 1 + 0.2309 q^-1

F(q) = 1 - 2.261 q^-1 + 1.673 q^-2 - 0.3293 q^-3 - 1.12 q^-4 + 1.85 q^-5 -
0.8123 q^-6

Estimated using BJ from data set z
Loss function 0.0783882 and FPE 0.0795769
Sampling interval: 0.1
```

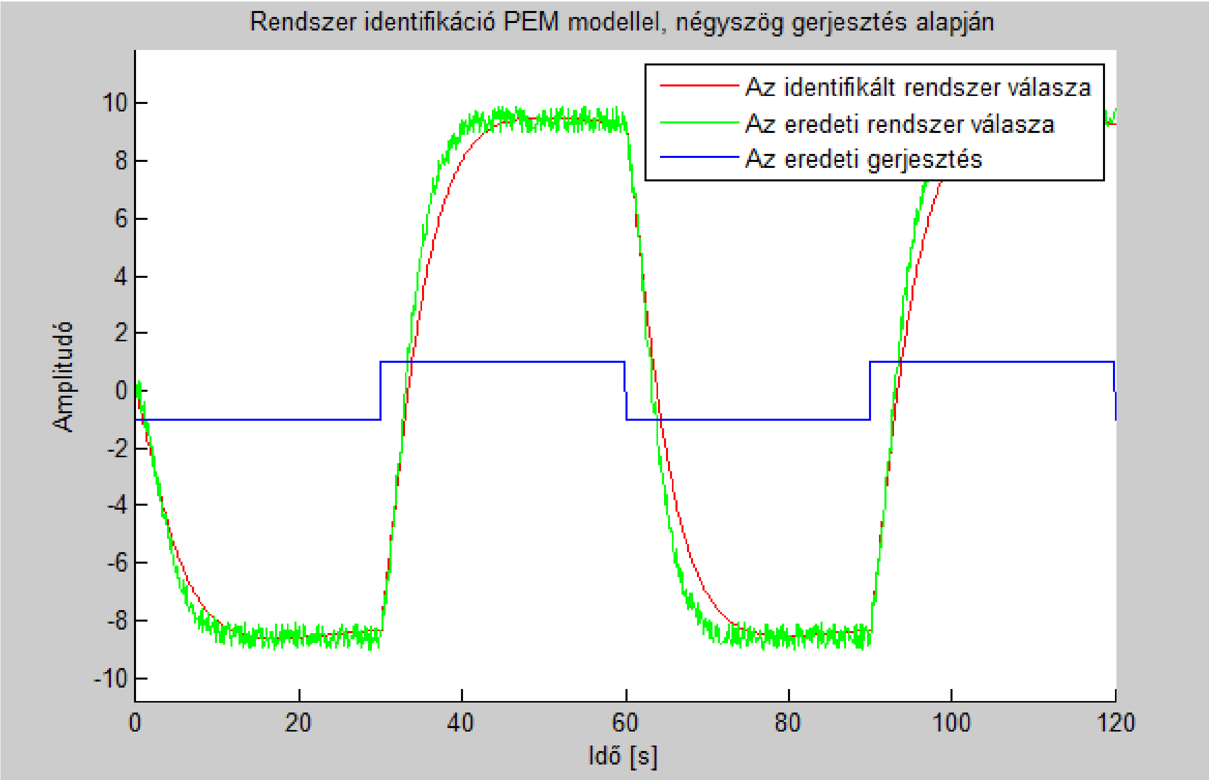
PEM identifikáció

Az általános lineáris paraméterbecslő modell:

$$A(z^{-1})y(t) = \frac{B(z^{-1})}{F(z^{-1})}u(t-n) + \frac{C(z^{-1})}{D(z^{-1})}e(t)$$

Az előzőekben használt módszerünket itt nem alkalmazhattuk letapogatásra, ugyanis ennek számítás igénye egy használható paramétertérre hatalmas, a futás rengeteg időbe telne egy hagyományos asztali PC--n. A paraméterek megválasztása így tapasztalati úton, kísérletezéssel történt.

Az identifikált rendszer szimulációja a 12. ábrán látható:



12. ábra A rendszer identifikálása PEM modellel

Látható, hogy az identifikált modellünk válasza (ha nem is úgy mint az előzőekben) de jól követi az eredeti rendszer válaszát, ez a kis eltérés betudható, annak, hogy most magunk adtuk meg a paramétereket, és nem futtatunk erre az előzőekben használt letapogató ciklusokat, így feltehetően paraméterkészletünk nem optimális.

A PEM modell alapján történt identifikáció paraméterei:

Négyzetes eltérés = 326.2641

n_a = 3

n_b = 4

n_c = 3

n_d = 4

n_f = 5

n_k = 3

```
Discrete-time IDPOLY model: A(q)y(t) = [B(q)/F(q)]u(t) + [C(q)/D(q)]e(t)
```

```
A(q) = 1 - 1.505 q^-1 + 1.408 q^-2 - 0.8625 q^-3

B(q) = 0.2873 q^-3 - 0.6818 q^-4 + 0.5184 q^-5 - 0.1239 q^-6

C(q) = 1 - 0.4166 q^-1 - 0.3863 q^-2 + 0.08528 q^-3

D(q) = 1 + 0.7338 q^-1 - 0.24 q^-2 - 0.3885 q^-3 - 0.1365 q^-4

F(q) = 1 - 1.704 q^-1 + 0.02719 q^-2 + 1.362 q^-3 - 0.9445 q^4 + 0.2592 q^-
5


Estimated using PEM using SearchMethod = Auto from data set z
Loss function 0.110529 and FPE 0.1141
Sampling interval: 0.1
```

Az eredmények értékelése

Az identifikációs modelleket átnézve láthattuk, hogy mindegyik nagyon jól becsüli az eredeti rendszert, de pontosságban eltérések vannak.

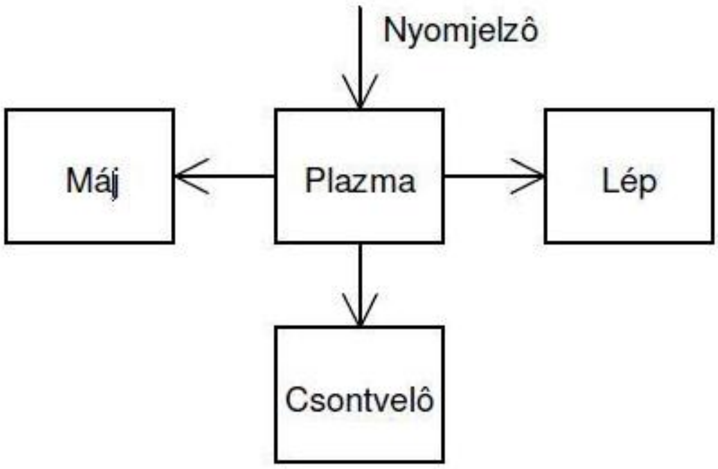
	LS_min	**Loss Function**	**FPE**
ARX	452,7412	0,118989	0,121169
ARMAX	94,3805	0,069178	0,0711713
OE	100,0996	0,0836169	0,0853104
Box-Jenkins	96,9211	0,0783822	0,147908
IV4	313,3601	0,144774	0,147908
PEM **(letapogatás nélkül)**	326,2641	0,0783882	0,0795769

Az identifikációkat összesítő táblázatunkból jól látható, hogy a legjobb becslést minden mérőszám szerint az ARMAX modell adta, annak ellenére, az IV4 robosztusabb algoritmusokat használ ugyanazon a modellen, míg a BJ és PEM modellek sokkal finomabbak és több változót tartalmaznak. Bár a letapogatási terek nem voltak egyforma nagyok minden modell esetén, az optimális ARMAX-hoz szükséges fokú polinomok minden modellben megengedettek voltak. Az eredmény azért is különösen jó, mert az ARMAX modell számítási igénye nem jelentős.

Fontos azonban megjegyezni, hogy a labormérésnek egy lényeges része, hogy az identifikációs bemenethez véletlen zajt kevertünk. Emiatt ezek az eredmények egy újabb szimulálási sor után, újból lefuttatva eltérhetnek a most kiszámolt értékektől, és lehet, hogy újabb futtatás után A Box Jenkins módszer mutatkozna legjobbnak. Ezen kívül a mérés numerikus szimulációra és identifikációra épül, itt pedig nem megkerülhető a mintavételi idő, a számítási pontosság és a platform szerepe.

2. mérés: Kolloidos májfunkció mérésből modellezése

A feladat megoldása során egy konkrét ^{198}Au kolloidos májfunkció mérés eredményeit használtuk fel. A vizsgálat azon alapszik, hogy bizonyos szemcsenagyságú kolloidokat a szervezetben található RES sejtek kiszűrnek. A RES sejtek legnagyobb számban a májban vannak jelen, de megtalálhatók a lépben és a csontvelőben is. A vizsgálat kezdetekor a páciensnek beadják intravénás injekció formájában a plazmába az ^{198}Au kolloidot. A jelzett anyag kis idő után felhígul, ahogy elkeveredik a szervezetben. Az elkeveredés után mérik az agy felett a plazma aktivitást. A mérési pontok elméletileg egy multiexponenciális jelleggel csökkenő görbén helyezkednek el. A vizsgálatnak megfelelően matematikai modell struktúrája négy kompartmentből álló úgynevezett mammilary rendszer, melyet a 13. ábra szemlélteti:

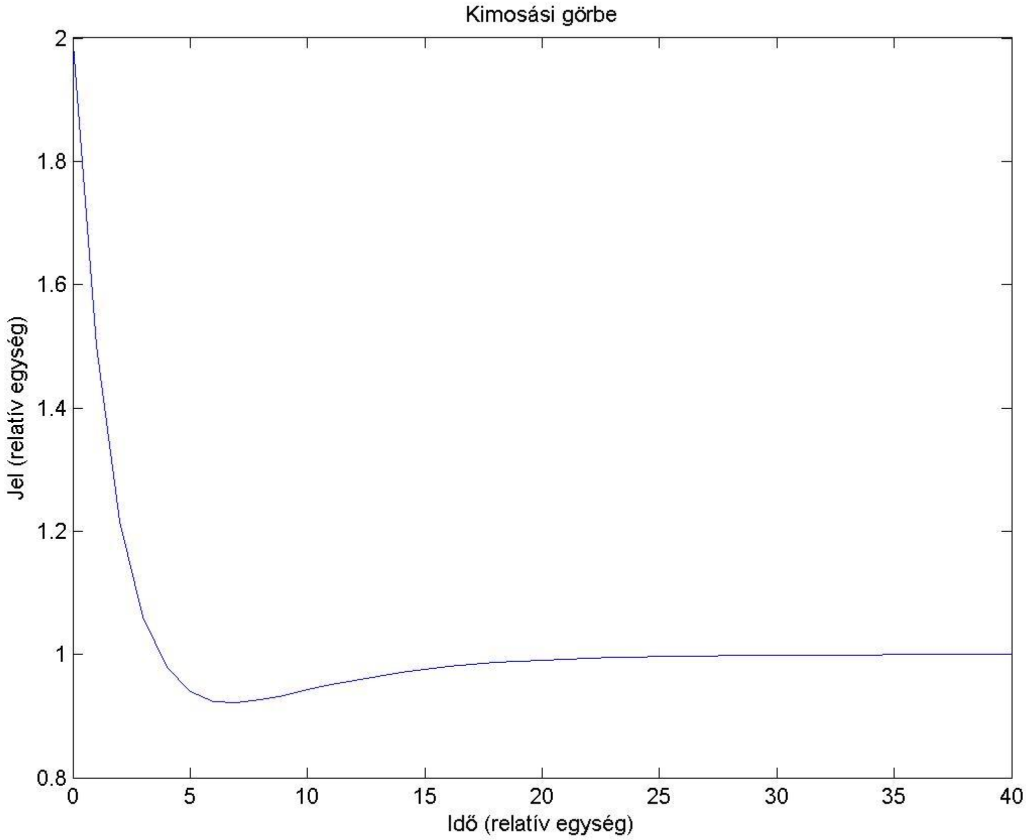


13. ábra: Máj izotópos vizsgálatának kompartment rendszere

A mérés leírása

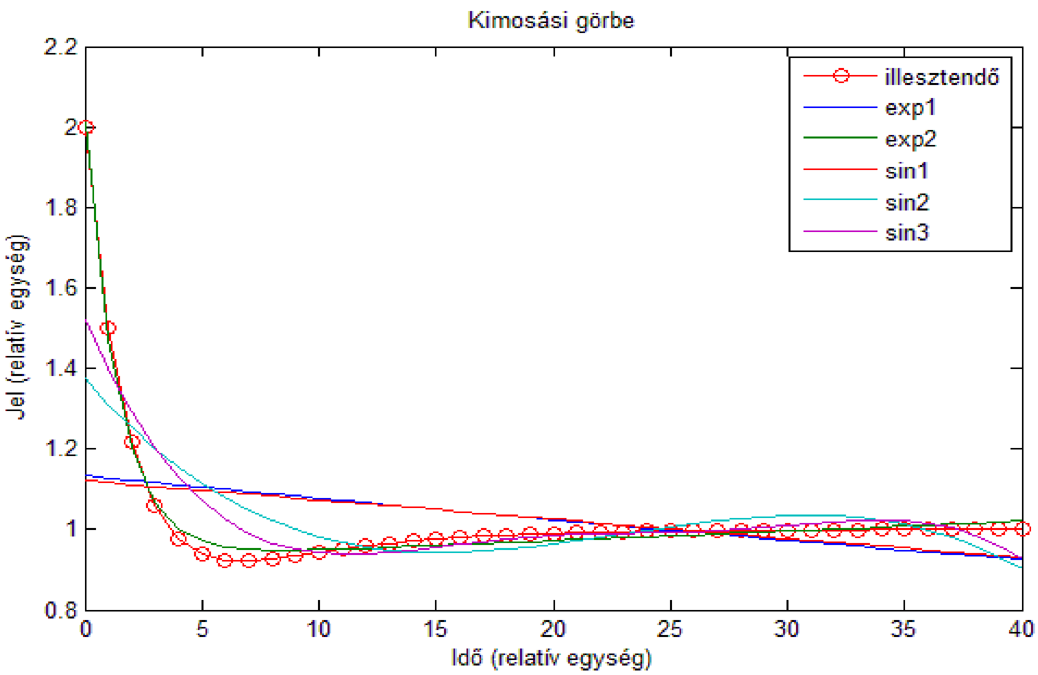
A feladat során egy konkrét ^{198}Au kolloidos májfunkció mérés eredményeit használjuk fel. A multiexponenciális görbe felfogható úgy, mint két polinom hányadosával felírható rendszer impulzusválasz függvénye, amely csak a mérés során terhelődik zajjal. Az általunk vizsgált rendszerek ebből adódóan leginkább a kimenőjelre redukált, additív zajt tartalmazó modellel jellemezhetők.

Először ábrázoltuk a kimosási görbét a *plot* függvény segítségével ezt a 14. ábra reprezentálja:



14. ábra: Au 198 kimosási görbéje

Majd erre a görbére illesztettünk egy illetve több komponensből álló függvényeket: sin1, sin2, sin3, exp1, exp2 különböző paraméterű függvényeket. Az illeszkedésüket az 15. ábra mutatja:



15. ábra Kimosási görbe és azt közelítő függvények

A görbéken látszik, hogy minél több paraméterű a függvény, annál jobban illeszkedik a kimosási görbéhez.
A függvények illeszkedésének négyzetes hibáit alábbi táblázatunkban mutatjuk:

Függvény típus	exp1	exp2	sin1	sin2	sin3
Négyzetes hiba	*1,16709*	*0,010713*	*1,178085*	*0,599324*	*0,335146*

Az eredmények értékelése

Az egy komponensből álló függvények (egy exponenciális- exp1, illetve az egy szinusz - sin1) rosszul illeszkednek, ezt mutatja a nagy négyzetes hiba és az ábra is, a több komponensű függvények jobban illeszkednek. A két exponenciális összegéből álló függvény ezek közül is kiemelkedően jól illeszkedik - ezt érdemes használnunk.

A mérés értékelése

A mérés során először létrehoztunk egy saját rendszert, ezt különböző vizsgáló jellel gerjesztettük, legszemléletesebbnek a négyszög vizsgálójel bizonyult. Majd ezt követően a megismert rendszermodellekkel identifikáltuk ismeretlen rendszerünket. Megfelelően parametrizálva mindegyik modell alkalmasnak bizonyul identifikációs feladatokra, de a mi mérésünkön legpontosabb eredményt az ARMAX adta.

A 2. feladatban egy májátáramlás vizsgálatára használ ^{198}Au izotóp kimosási görbéjét próbáltuk meg közelíteni többparaméteres függvényekkel, ahol is bebizonyosodott, hogy minél több paraméterű egy függvény, annál jobban közelíti ezt a multiexponenciális kimosási görbét.

Függelék – MATLAB kódok

Rendszer leírása

```
%% Ez lesz a rendszerünk. Átviteli függvénnyel definiáljuk %
A1 = [2,3,9];        % 1. rendszer számlálója %
B1 = [1,5,4,1];      % 1. rendszer nevezője %

sys1 = tf(A1,B1);    % Ez a rendszer, MATLAB system modelben %

%% A rendszer ugrásválasza %
figure(1);
step(sys1);

% A rendszer impulzusválasza %
figure(2);
impulse(sys1);

%% Szinuszos és négyzetes gerjesztő jel generálsása %
T = 60;       % Periódus idő [s]. Azért ilyen hosszú, hogy be tudjon állni a
válasz%
Tmax = 120;  % Hossz [s] %
Ts = 0.1;  % Mintavételezési idő [s] %

[sin_input,t] = gensig('sin',T,Tmax,Ts);        % Szinuszos bemeneti jel %
[square_input] = gensig('square',T,Tmax,Ts);    % Bemeneti négyszögjel %
% Hogy 0.5 és -0.5 között legyen a jel, különben DC terhelt %
square_input = square_input-0.5;
% Hogy 1 és -1 között legyen, különben nem lesz meg az SNR
square_input = square_input*2;
% A szinusz jelre adott válasz szimulálása %
szinusz_valasz = lsim(sys1,sin_input,t);
plot(t,szinusz_valasz,'r','LineWidth',1.4);

% A négyszög ábrázolása %
plot(t,square_input,'b','LineWidth',1.4);
% A válasz ábrázolása %
% A négyszög jelre adott válasz szimulálása %
negyszog_valasz = lsim(sys1,square_input,t);
plot(t,negyszog_valasz,'r','LineWidth',1.4);
%% Rendszer négyszögválaszának zajosítása %
% Zajosítás kb. 10-es SNR mellett %
zajos_negyszog_valasz = negyszog_valasz + 0.9*rand(size(negyszog_valasz));
figure(5);
%% A négyszög gerjesztésből fogunk identifikálni, ehhez elrakjuk a
%  választ és a gerjesztést
to_ident = iddata(zajos_negyszog_valasz, square_input, Ts);
```

ARX Identifikáció

```
%% ARX identifikáció
% Keressük a legkisebb négyzetes eltérést, ez a költségünk %
J = [];
for i = 0:9                %na
    for j = 1:10           %nb  >> +1
        for k = 0:9        %nk
            % Lesz pár instabil modell, ne dobjuk rá warningot
            warning off
            % Az aktuális ARX identifikáció paraméterei %
            params = [i j k];
            % ARX identifikáció %
            ARX = arx(to_ident,params);
            % Szimuláljuk az identifikált rendszer válaszát
            % lsim helyett sim, mert ez idpoly struktúra, nem LTI %
            ARX_valasz = sim(ARX,square_input);
            % Hasonlítsuk össze az eredeti (zajos) kimenettel %
            helyi_J = sum((ARX_valasz-zajos_negyszog_valasz).^2);
            % Rögzítsük az eredményt %
            J = [J,helyi_J];
            warning on
        end
    end
end

% Keressük meg a költségfüggvény minimumát %
[LS_min,LS_pos] = min(J);

disp('Az ARX modell alapján történt identifikáció paraméterei:')
disp('');
LS_min
% Számoljuk vissza a paramétereket %
% Az elv a következő: a fenti for ciklus indexelése tulajdonképp
% tízes számrendszer szerint megy végig a pozíciókon, nullától indulva.
% j értéke kivételnek tűnhet, de ott is 10 érték van, így csak hozzá kell
% majd adni egyet a többihez képest.
% Szóval vegyük, hogy a minimum hányas pozíción van. Ebből levonunk egyet,
% mert a MATLAB 1-től indexel, mi pedig 0-tól megyünk.
% A maradékot leosztjuk helyiérték szerint, és meglesznek az indexek.
na = fix((LS_pos-1)/100)            % Mert 0-tól ment 9-ig %
% Hogy helyiérték szerint tovább bonthassunk, le kell vonni na*100-at.
nb = fix((LS_pos-1-na*100)/10) + 1  % Mert 1-től 10-ig ment %
% Hogy helyiérték szerint tovább bonthassunk, le kell vonni további (nb-
1)*10-et
nk = fix((LS_pos-1-na*100-(nb-1)*10)/1) % Mert 0-tól ment 9-ig %

% Végezzük el az identifikációt újra, a legjobb paraméterekkel
params = [na nb nk];
% ARX identifikáció elemzése %
disp('Az ARX modell alapján történt identifikáció eredménye:')
disp('');
ARX = arx(to_ident,params)
% Szimuláljuk az identifikált rendszer válaszát
ARX_valasz = sim(ARX,square_input);

% Hasonlítsuk össze grafikusan az eredeti (zajos) kimenettel %
figure(10)
```

```
hold on
% Az identifikált rendszer válasza %
plot(t,ARX_valasz,'r','LineWidth',1.4);
% Az eredeti rendszer válasza %
plot(t,zajos_negyszog_valasz,'g','LineWidth',1.4);
% Az eredeti gerjesztés %
plot(t,square_input,'b','LineWidth',1.4);

% Címkézés %
legend('Az identifikált rendszer válasza','Az eredeti rendszer válasza',...
    'Az eredeti gerjesztés');
% Tengely beállítás %
axis([min(t),max(t),1.2*min([min(ARX_valasz),min(zajos_negyszog_valasz),...
    min(square_input)]), 1.2*max([max(ARX_valasz),...
    max(zajos_negyszog_valasz),max(square_input)])]);
xlabel('Idő [s]');
ylabel('Amplitudó');
title('Rendszer identifikáció ARX modellel, négyszög gerjesztés alapján');

hold off
```

Box-Jenkins identifikáció

```
%% Box Jenkins identifikáció
% Keressük a legkisebb négyzetes eltérést, ez a költségünk %
% A vizsgálati paraméterek tartományának maximumai
max_i = 6;    %nb >> +1
max_j = 4;    %nc
max_k = 4;    %nd
max_l = 6;    %nf
max_m = 5;    %nk
% Több számolásra készülünk, ezért előre foglalunk helyet
J = zeros(1,(max_i)*(max_j+1)*(max_k+1)*(max_l+1)*(max_m+1));
index = 1;
for i = 1:max_i
    for j = 0:max_j
        for k = 0:max_k
            for l = 0:max_l
                for m = 0:max_m
                    % Lesz pár instabil modell, ne dobjuk rá warningot
                    warning off
                    % Az aktuális Box Jenkins identifikáció paraméterei %
                    params = [i j k l m];
                    % BJ identifikáció %
                    BJ_model = BJ(to_ident,params);
                    % Szimuláljuk az identifikált rendszer válaszát
                    BJ_model_valasz = sim(BJ_model,square_input);
                    % Hasonlítsuk össze az eredeti (zajos) kimenettel %
                    helyi_J = sum((BJ_model_valasz-
zajos_negyszog_valasz).^2);
                    % Rögzítsük az eredményt %
                    J(index) = helyi_J;
                    index = index + 1;
                    warning on
                end
            end
        end
    end
```

```
end

% Keressük meg a költségfüggvény minimumát %
[LS_min,LS_pos] = min(J);
% Számoljuk vissza a paramétereket %
disp('A Box Jenkins modell alapján történt identifikáció paraméterei:')
disp('');
LS_min
% Az összeset száma a kominációk szorzata, ezt használva fejtünk vissza %
% Olyan, mint az előző, csak nem tízes, hanem egy általános számrendszerben
% Figyeljünk, hogy mi megy 0-tól és mi 1-től
rb = (max_j+1)*(max_k+1)*(max_l+1)*(max_m+1);
nb = fix((LS_pos-1)/rb) + 1

rc = (max_k+1)*(max_l+1)*(max_m+1);
nc = fix((LS_pos-1-(nb-1)*rb)/rc)

rd = (max_l+1)*(max_m+1);
nd = fix((LS_pos-1-(nb-1)*rb-nc*rc)/rd)

rf = (max_m+1);
nf = fix((LS_pos-1-(nb-1)*rb-nc*rc-nd*rd)/rf)

nk = mod(LS_pos-1,(max_m+1))

% Végezzük el az identifikációt újra, a legjobb paraméterekkel
params = [nb nc nd nf nk];
% Box Jenkins identifikáció elemzése %
disp('A Box Jenkins modell alapján történt identifikáció eredménye:');
disp('');
BJ_model = BJ(to_ident,params)
% Szimuláljuk az identifikált rendszer válaszát
BJ_model_valasz = sim(BJ_model,square_input);

% Hasonlítsuk össze grafikusan az eredeti (zajos) kimenettel %
figure(14)
hold on
% Az identifikált rendszer válasza %
plot(t,BJ_model_valasz,'r','LineWidth',1.4);
% Az eredeti rendszer válasza %
plot(t,zajos_negyszog_valasz,'g','LineWidth',1.4);
% Az eredeti gerjesztés %
plot(t,square_input,'b','LineWidth',1.4);

% Címkézés %
legend('Az identifikált rendszer válasza','Az eredeti rendszer válasza',...
    'Az eredeti gerjesztés');
% Tengely beállítás %
axis([min(t),max(t),1.2*min([min(BJ_model_valasz),min(zajos_negyszog_valasz
),...
    min(square_input)]), 1.2*max([max(BJ_model_valasz),...
    max(zajos_negyszog_valasz),max(square_input)])]);
xlabel('Idő [s]');
ylabel('Amplitudó');
title('Rendszer identifikáció Box Jenkins modellel, négyszög gerjesztés
alapján');

hold off
```

Clearance vizsgálat

```
%2. feladat
%% A megadott új adatsor betöltése adott helyről
clear all;
clc;
[FileNames FilePaths] = uigetfile('*.mat','Adatsor kiválasztása');
load(strcat(FilePaths,FileNames))
% A kimosási függvény ábrázolása, mentése
plot(t, data)
title('Kimosási görbe');
xlabel('Idő (relatív egység)');
ylabel('Jel (relatív egység)');
[FileNames FilePaths] = uiputfile('*.jpg','Kimosasi gorbe mentese');
saveas(gcf,strcat(FilePaths,FileNames));
% az illesztes típusainak felsorolása
f=['exp1';'exp2';'sin1';'sin2';'sin3'];
% illesztések - illeszesztések paraméterei
ill1=fit(t,data,f(1,:));
ill2=fit(t,data,f(2,:));
ill3=fit(t,data,f(3,:));
ill4=fit(t,data,f(4,:));
ill5=fit(t,data,f(5,:));
% illesztések - illesztések értékei az időnek (t) megfelelően
illesztett1=feval(ill1,t);
illesztett2=feval(ill2,t);
illesztett3=feval(ill3,t);
illesztett4=feval(ill4,t);
illesztett5=feval(ill5,t);
% négyzetes hiba számítása
sq_diff(1)=sum((data-illesztett1).^2);
sq_diff(2)=sum((data-illesztett2).^2);
sq_diff(3)=sum((data-illesztett3).^2);
sq_diff(4)=sum((data-illesztett4).^2);
sq_diff(5)=sum((data-illesztett5).^2);
% az eredmények közlése
eredmeny(1,2:6)={'exp1';'exp2';'sin1';'sin2';'sin3'};
eredmeny(2,2:6)=num2cell(sq_diff);
eredmeny(1:2,1)={'típus','négyzetes hiba'};
eredmeny
```