Sprawozdanie z przedmiotu Modelowanie systemów dynamicznych.

Temat laboratoriów: Identyfikacja obiektu regulacji

Filip Pasternak, Grupa lab. 7, środa godz 13:45.

Identyfikacja modelu A metodą wyznaczania parametrów na podstawie odpowiedzi skokowej.

$$G(s) = \frac{ke^{-s\theta}}{Ts+1}$$ | obiekt inercyjny I rzędu z opóźnieniem

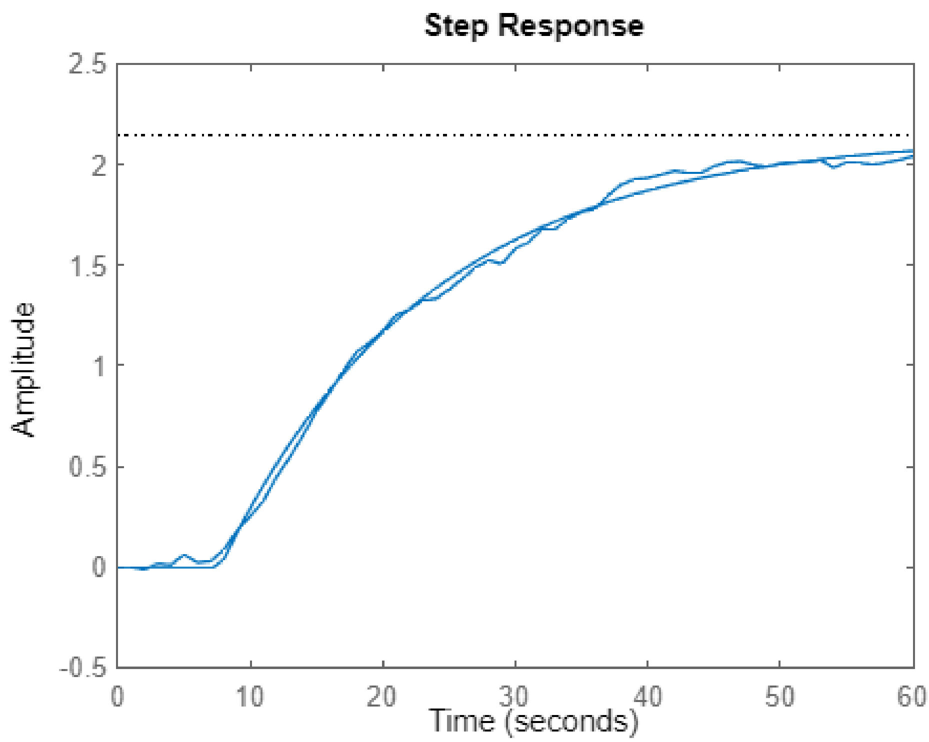
Transmitancja modelu A

Metoda ta polega na manualnym dobraniu parametrów obiektu tak, aby błąd wynikający z porównania tych dwóch wykresów był jak najmniejszy.

```
t = 1:1:60;

% A
% 2.1
k = 2.145;
T = 15.7;
theta = 7.7;

obiektA = tf([0 k], [T 1])
set(obiektA, 'outputDelay', theta)

figure
hold on
step(obiektA)
plot(t, y)
xlim([0 60])

% blad aproksymacji:
e = y - step(obiektA, t);
RMS = sum(e.^2)/length(e)
```

```
RMS = 0.0016
```

Najmniejszy błąd jaki udało mi się osiągnąć wynosił 0.0016, co jest nie najgorszym wynikiem, zwłaszcza widząc jak oba wykresy na siebie się nakładają.



Identyfikacja modelu A metodą optymalizacji numerycznej.

Jest to metoda która jest swoich działaniem bardzo zbliżona do metody użytej poprzednio, jednak w tym przypadku parametry są dobrane automatycznie przy użyciu wbudowanej w środowisko Matlab funkcji *fminsearch()*. Funkcja ta zwraca parametry obiektu dla których błąd jest najmniejszy.
Do wykonania tego typu identyfikacji została zdefiniowana nowa funkcja obliczająca błąd dla modelu A. Została ona opisana w oddzielnym skrzypcie *ident.*

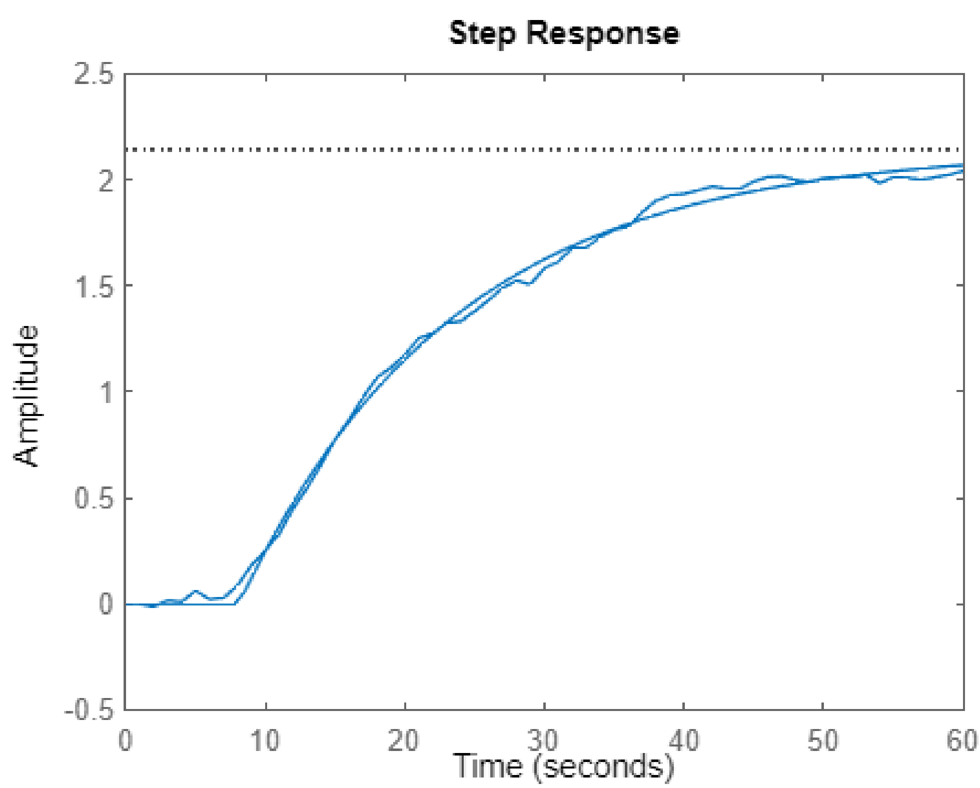
```
function blad = ident(X0)
load obiekt.mat
k = X0(1);
T = X0(2);
theta = X0(3);

t = 1:1:60;

obiektA = tf([0 k], [T 1]);
set(obiektA, 'outputDelay', theta);
y_sym = step(obiektA, t);

e = y - y_sym;
blad = sum(e.^2) / length(e);
```

```
% 2.4
[parametry, blad] = fminsearch('ident',[1,1,1])
```

```
parametry = 1×3
     2.1422    15.4254     8.0971
blad = 0.0015
```

```
obiektA = tf([0 parametry(1)], [parametry(2) 1])
set(obiektA, 'outputDelay', parametry(3))

figure
hold on
step(obiektA)
plot(t, y)
xlim([0 60])
```

Błąd tym razem wyniósł 0.0015, co jest lepszym wynikiem w porównaniu z manualnym dobieraniem parametrów.



Identyfikacja modelu B za pomocą metody optymalizacji numerycznej.

Metoda jest ta sama co w poprzednim przykładzie, jednak kod wymagał lekkich modyfikacji ze względu na prace z obiektem innego typu:

$$G(s) = \frac{ke^{-s\theta}}{(T_1 s + 1)(T_2 s + 1)}$$

obiekt inercyjny II rzędu z opóźnieniem (aproksymacja Kupfmuellera)

Transmitancja modelu B

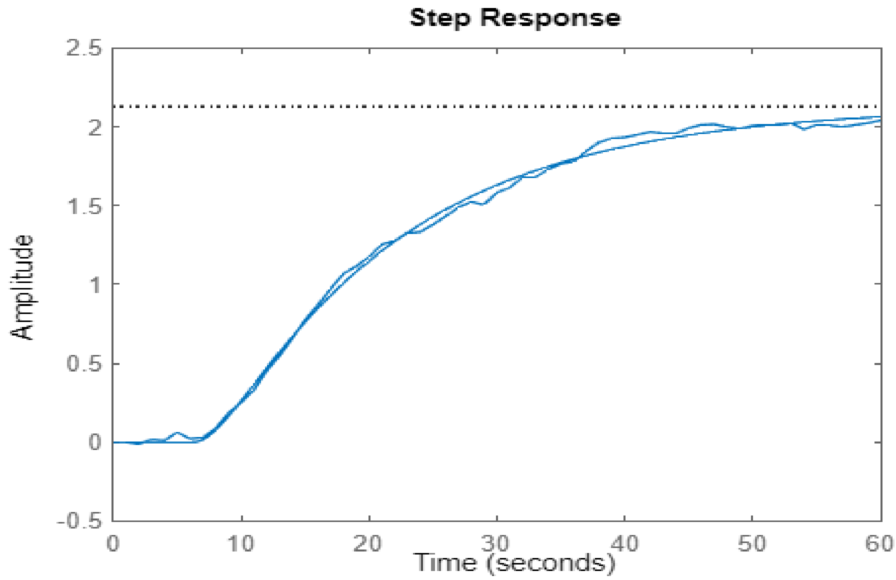
Podobnie jak poprzednio, do użycia funkcji *fminsearch()* został wykorzystany oddzielny skrypt obliczający błąd dla modelu II rzędu *ident2*

```
function blad = ident2(X0)
load obiekt.mat
k = X0(1);
T1 = X0(2);
T2 = X0(3);
theta = X0(4);

t = 1:1:60;

obiektB = tf([0 0 k], conv([T1 1], [T2 1]));
set(obiektB, 'outputDelay', theta);

y_sym = step(obiektB, t);
e = y - y_sym;
blad = sum(e.^2) / length(e);
```

```
% B
% 2.4
[parametry_b, blad_b] = fminsearch('ident2',[1,1,1,1])
```

```
parametry_b = 1×4
    2.1292    2.0541   14.8405    6.2836
blad_b = 0.0013
```

```
obiektB = tf([0 0 parametry_b(1)], conv([parametry_b(2) 1], [parametry_b(3)
1]));
set(obiektB, 'outputDelay', parametry_b(4))

figure
hold on
step(obiektB)
plot(t, y)
xlim([0 60])
```



Identyfikacja modelu C za pomocą metody optymalizacji numerycznej.

Tak samo jak w poprzednim przykładzie kod wymagał pewnych (w tym przypadku znaczniejszych) modyfikacji, ponieważ pracujemy w tym przypadku z obiektem o nieokreślonym rzędzie.
Zamiast jednego dodatkowego skryptu napisałem ich 4 dla każdego badanego rzędu: *ident3a, ident3b, ident3c* oraz *ident4d.*

```
function blad = ident3a(X0)
load obiekt.mat
k = X0(1);
T = X0(2);

global n;

t = 1:1:60;

obiektC = zpk([], [-T, -T],k);
y_sym = step(obiektC, t);

e = y - y_sym;
blad = sum(e.^2) / length(e);
```

Dla kolejnych rzędów jedyna zmianą jest zwiększenie ilości biegunów -T.

```
% C
[parametry_c_rzad2, blad_c_rzad2] = fminsearch('ident3a', [1 1])
[parametry_c_rzad3, blad_c_rzad3] = fminsearch('ident3b', [1 1])
```

```
parametry_c_rzad3 = 1×2
    0.0058    0.1411
blad_c_rzad3 = 0.0027
```

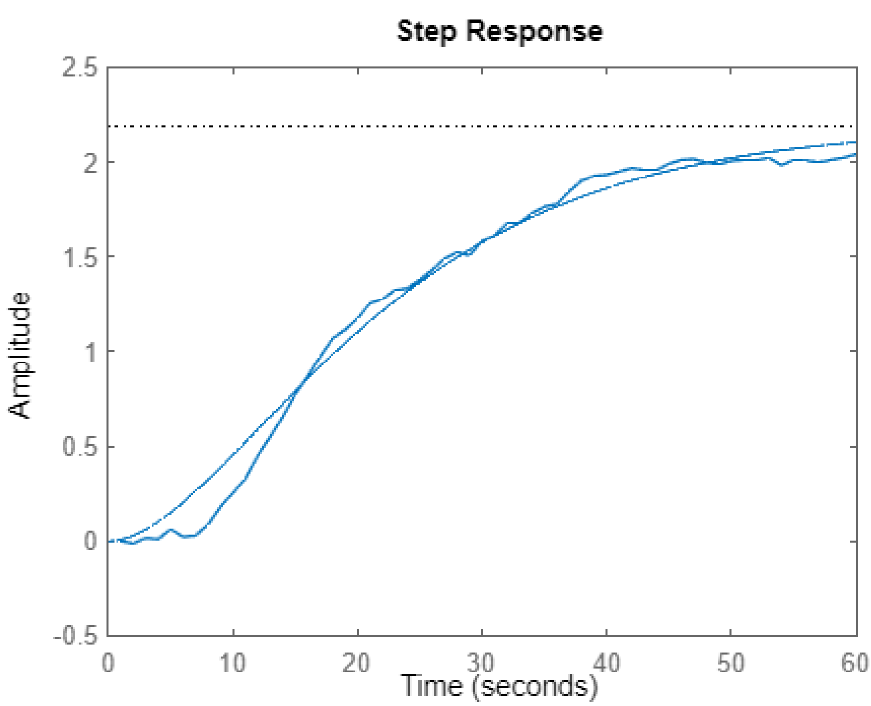
```
[parametry_c_rzad4, blad_c_rzad4] = fminsearch('ident3c', [1 1])
```

```
parametry_c_rzad4 = 1×2
    0.0030    0.1967
blad_c_rzad4 = 0.0032
```

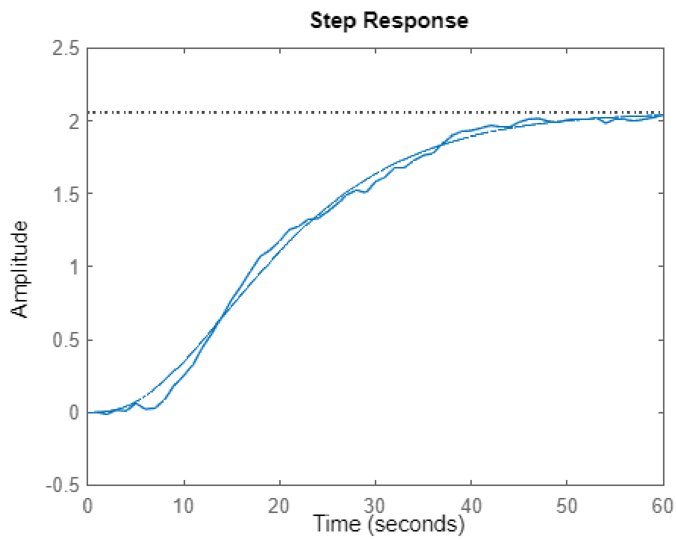
```
[parametry_c_rzad5, blad_c_rzad5] = fminsearch('ident3d', [1 1])
```

```
parametry_c_rzad5 = 1×2
    0.0020    0.2522
blad_c_rzad5 = 0.0055
```

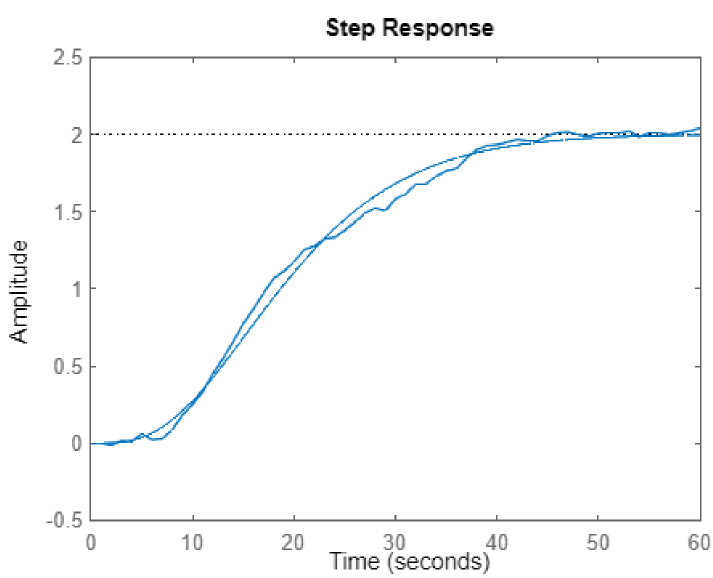
```
obiektC_rzad2 = zpk([], [-parametry_c_rzad2(2), -
parametry_c_rzad2(2)],parametry_c_rzad2(1));
obiektC_rzad3 = zpk([], [-parametry_c_rzad3(2), -parametry_c_rzad3(2), -
parametry_c_rzad3(2)],parametry_c_rzad3(1));
obiektC_rzad4 = zpk([], [-parametry_c_rzad4(2), -parametry_c_rzad4(2), -
parametry_c_rzad4(2), -parametry_c_rzad4(2)],parametry_c_rzad4(1));
obiektC_rzad5 = zpk([], [-parametry_c_rzad5(2), -parametry_c_rzad5(2), -
parametry_c_rzad5(2), -parametry_c_rzad5(2), -
parametry_c_rzad5(2)],parametry_c_rzad5(1));
% obiekt rzedu 2
figure
hold on
step(obiektC_rzad2)
plot(t, y)
xlim([0 60])
```



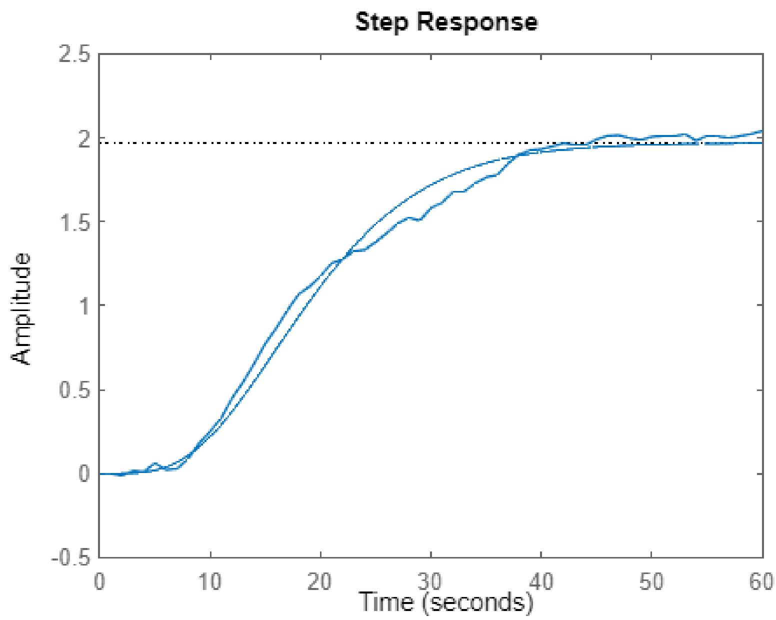
```
% obiekt rzedu 3
figure
hold on
step(obiektC_rzad3)
plot(t, y)
xlim([0 60])
```




```
% obiekt rzedu 4
figure
hold on
step(obiektC_rzad4)
plot(t, y)
xlim([0 60])
```



```
% obiekt rzedu 5
figure
hold on
step(obiektC_rzad5)
plot(t, y)
xlim([0 60])
```



Rząd	K	T	Błąd
2	0.0156	0.0844	0.0073
3	0.0058	0.1411	0.0027
4	0.0030	0.1967	0.0032
5	0.0020	0.2522	0.0055

Można wyciągnąć prosty wniosek, że dla obiektu inercyjnego rzędu III bez opóźnienia metoda optymalizacji numerycznej zwraca najdokładniejszy wynik. Wraz z dalszym wzrostem rzędów, zwiększa się także błąd jednocześnie pogarszając jakość naszej aproksymacji.