Modelowanie systemów dynamicznych

Laboratorium 10

Identyfikacja obiektów dynamicznych

Łukasz Janusz
Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

14 stycznia 2025

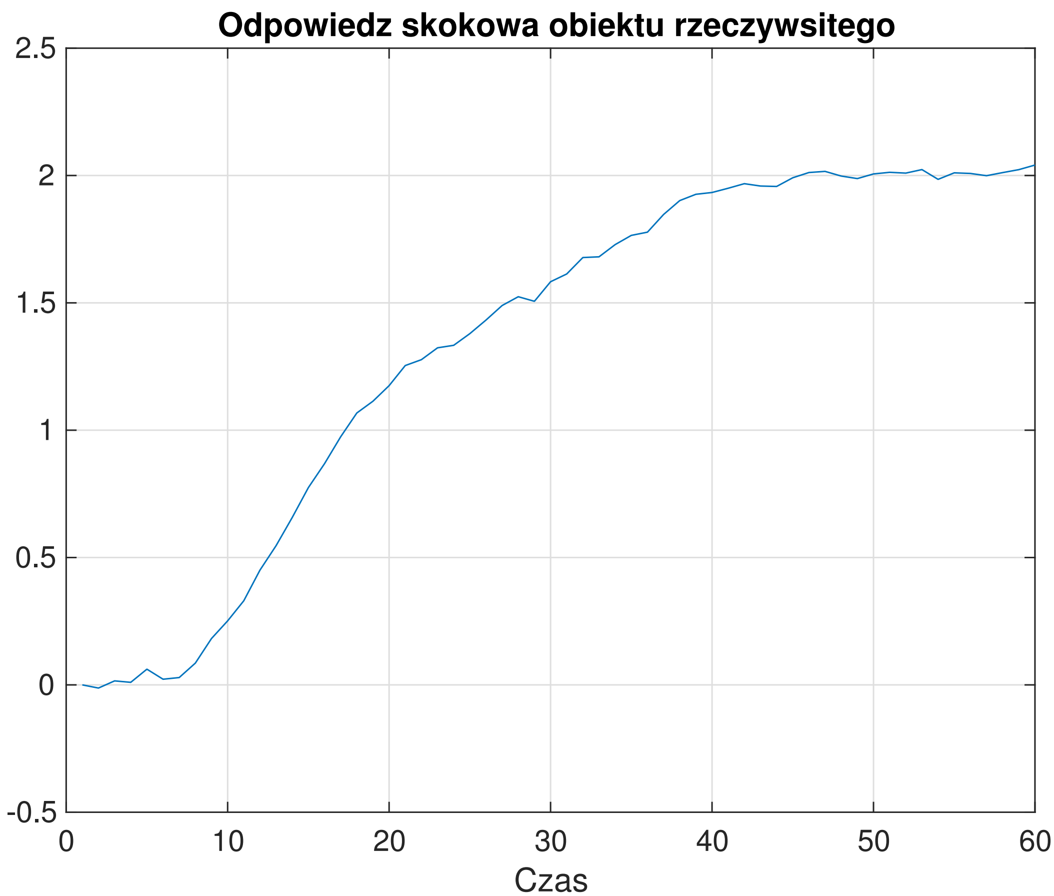
Wstęp

Celem laboratorium jest identyfikacja podanego obiektu rzeczywistego przy pomocy modeli inercyjnych I i II rzędu oraz wieloinercyjnego. W tym celu wykorzystane zostaną metody optymalizacji oraz funkcje matlaba do identyfikacji modeli.

Z trywialnych przyczyn na początku należy załadować odpowiedź skokową obiektu rzeczywistego.

```
load("obiekt.mat")\break
```

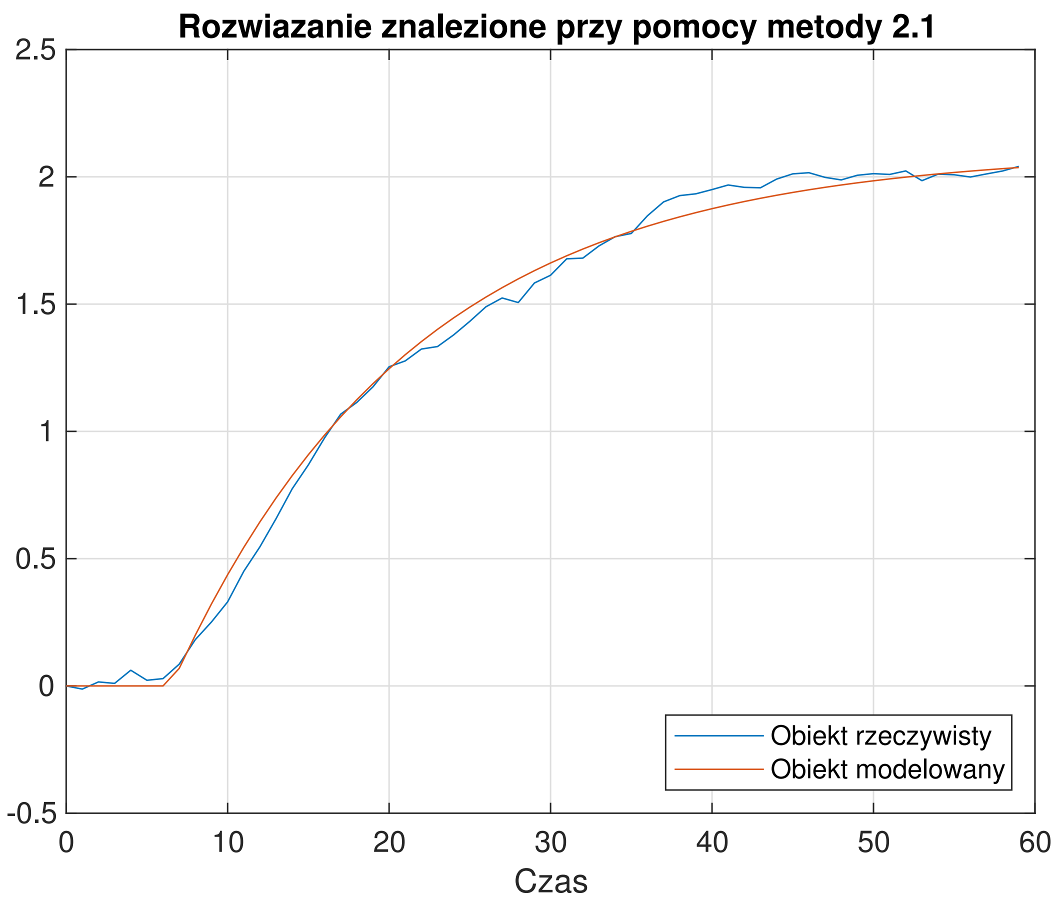
```
plot(y)
title("Odpowiedz skokowa obiektu rzeczywsitego")
xlabel("Czas")
grid on
```



Model A - obiekt inercyjny I rzędu z opóźnieniem

W pierwszym kroku należy zidentyfikować parametry modelu inercyjnego I rzędu z opóźnieniem. W tym celu najpierw użyłem mojej wiedzy o parametrach takowego obiektu, które w przybliżeniu można odczytać z wykresu. Następnie wykorzystałem funkcję *fminsearch*, w której wywołałem napisaną przeze mnie funkcję *ident.m*, która oblicza błąd średniokwadratowy między odpowiedzią rzeczywistą a odpowiedzią modelowaną. Dzięki temu uzyskałem numeryczne rozwiazanie problemu. Na końcu tej części porównałem wyniki obu metod używając jako kryterium wartości błędu średniokwadratowego.

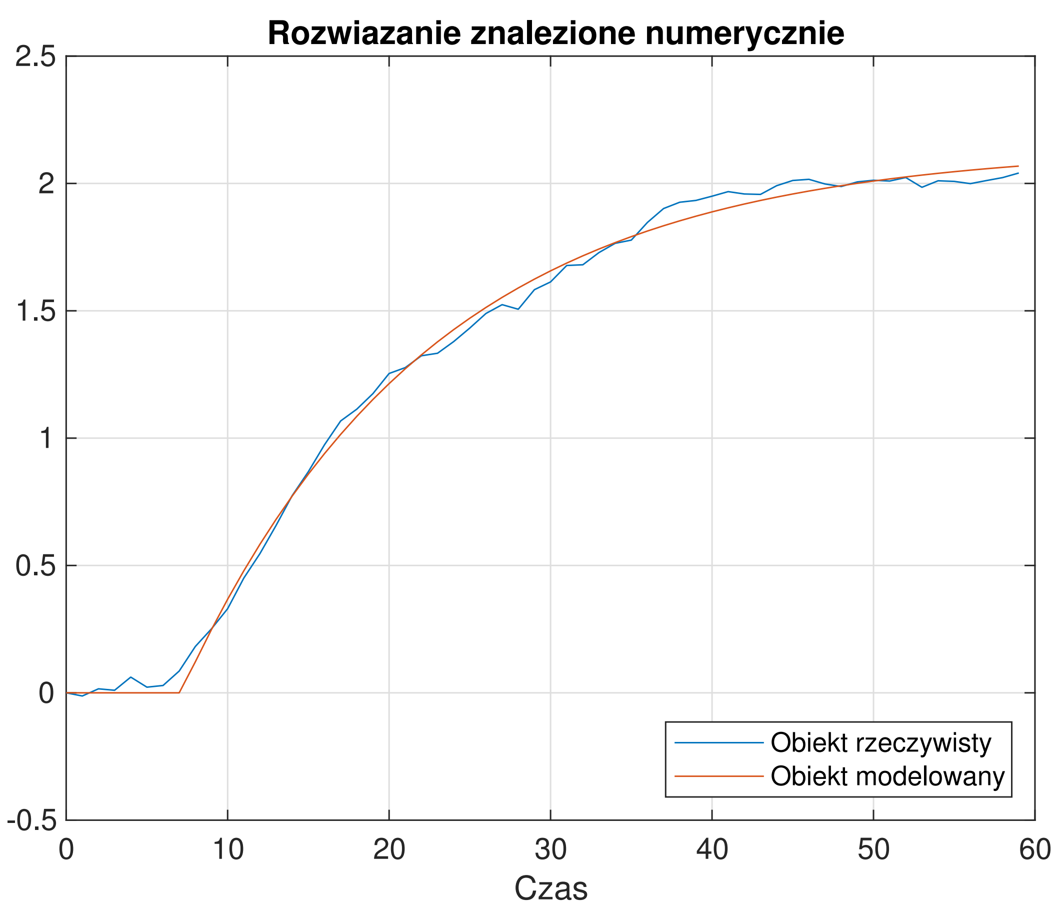
```
%Potrzebne współczynniki
%k
%T
%Theta
t = 0:length(y)-1;
T = 15;
k = 2.1;
Theta = 6.5;
obiekt = tf(k, [T,1],'OutputDelay', Theta);
res = step(obiekt, t);
plot(t,y,t,res)
grid on
legend("Obiekt rzeczywisty", "Obiekt modelowany", Location="southeast")
xlabel("Czas")
title('Rozwiazanie znalezione przy pomocy metody 2.1')
```



```
e = y - res;
blad = sum(e.^2)/length(e);
fprintf('MSE: %.4f\n', blad);
```

```
MSE: 0.0023
```

```
[parametry, blad] = fminsearch('ident',[k,T,Theta]);
k_n = parametry(1);      % 2.1422
T_n = parametry(2);      % 15.4255
Theta_n = parametry(3);  % 7.0971
obiekt = tf(k_n, [T_n,1],'OutputDelay', Theta_n);
res = step(obiekt, t);
plot(t,y,t,res)
grid on
legend("Obiekt rzeczywisty", "Obiekt modelowany", Location="southeast")
title('Rozwiazanie znalezione numerycznie')
xlabel("Czas")
```




```
fprintf('MSE: %.4f\n', blad);
```

```
MSE: 0.0015
```

Jak widać, wyniki obu metod są bardzo zbliżone, jednak metoda numeryczna daje nieco lepsze wyniki, czego można się było spodziewać.

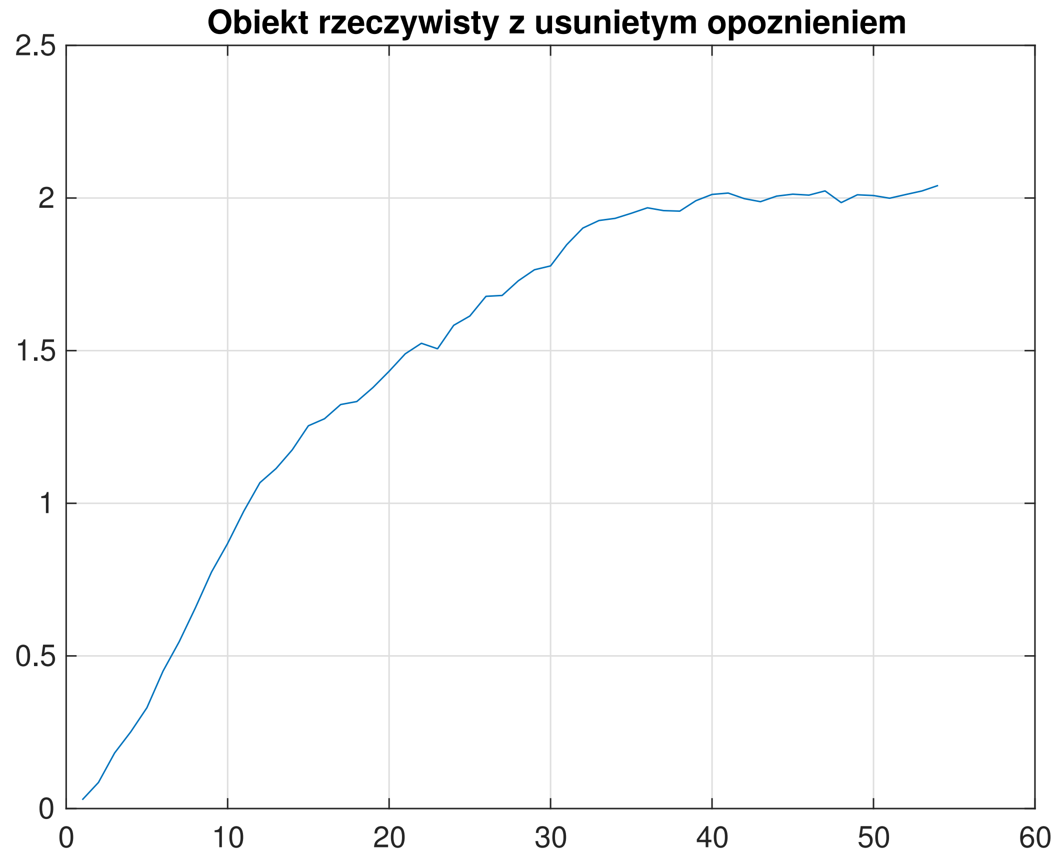
Model B - obiekt inercyjny II rzędu z opóźnieniem

Drugim modelem, przy pomocy którego zostanie zamodelowany obiekt jest model inercyjny II rzędu z opóźnieniem. W tym celu postępuję analogicznie jak w poprzednim przypadku, jednak tym razem muszę na początku znaleźć dodatkowy parametr - czas opóźnienia. W tym celu wykorzystam wiedzę z poprzedniego zadania, gdzie opóźnienie wynosiło około 7 sekund. Następnie, zgodnie z instrukcją odczytałem z wykresu wartość $y = 0,714k$, która wystąpiła dla $t_a = 22$. Na tej podstawie obliczyłem wartość $t_b = \frac{t_a}{4}$ oraz $y_{t_b} = \frac{y(t_b-0.5)+y(t_b+0.5)}{2}$. Następnym krokiem było wyznaczenie współczynnika $\frac{y_{t_b}}{k}$ i na podstawie jego wartości znalezienie w podanej tabelce wartości T_2/T_1. Ostatecznie, na podstawie tych danych oraz wzorów z instrukcji, znalazłem optymalne wartości parametrów modelu.

```
%Z poprzedniego zadania wiem, że opóźnienie wynosi ok 7 sekund
Delay = 7
```

```
Delay = 7
```

```
t_new = Delay:60;
y_new = y(t_new);
t_new = t_new - (Delay-1);
plot(t_new,y_new)
grid on
title("Obiekt rzeczywisty z usunietym opoznieniem")
```

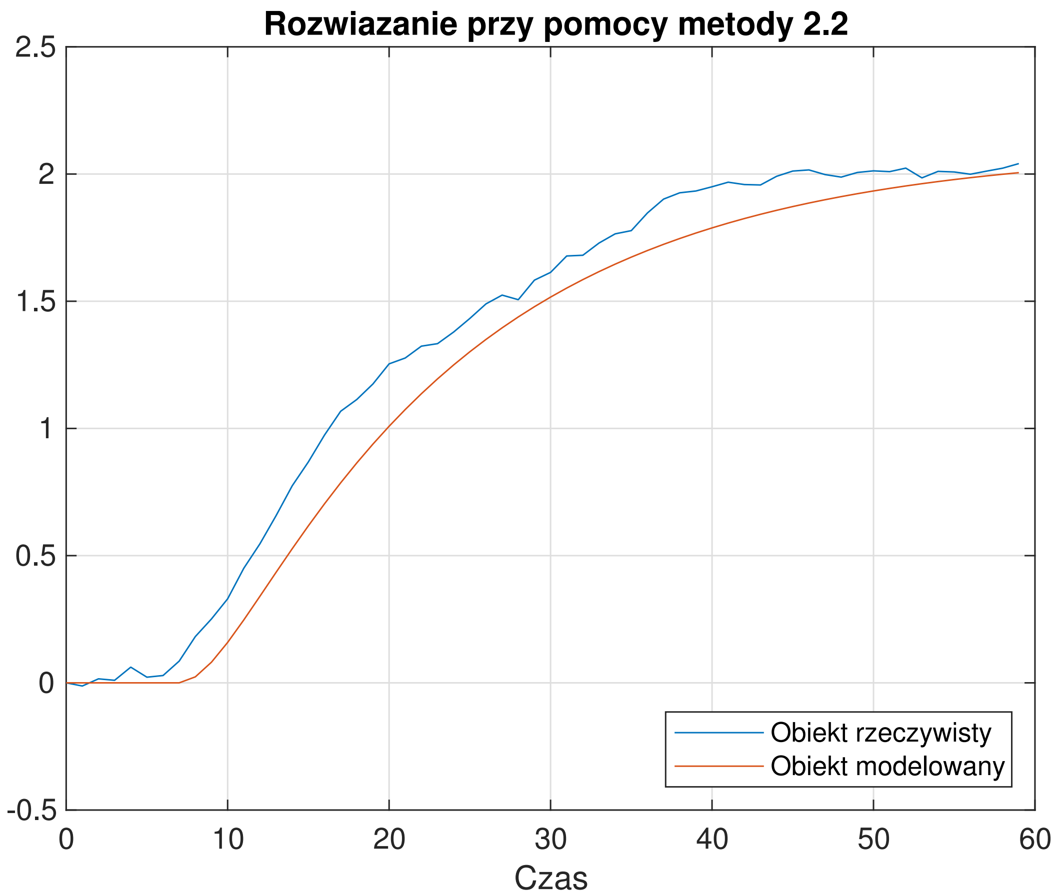



```
ta = 22;
tb = ta*0.25;
y_tb = y_new(tb-0.5)/2 + y_new(tb+0.5)/2;
wspolczynnik = y_tb/k
```

```
wspolczynnik = 0.1859
```

```
%ponieważ uzyskana wartość jest mniej więcej w połowie między danymi z
%tabelki użyję T2/T1 = 0.15
T1 = ta/(1.2*(1+0.15));
T2 = T1 * 0.15;
licz = k;
mian = [T1*T2 T1+T2 1];
obiekt2 = tf(licz,mian);
set(obiekt2, 'outputdelay', Delay)
res = step(obiekt2, t);
plot(t,y,t,res)
grid on
legend("Obiekt rzeczywisty", "Obiekt modelowany", Location="southeast")
title('Rozwiazanie przy pomocy metody 2.2')
```

```
xlabel("Czas")
```



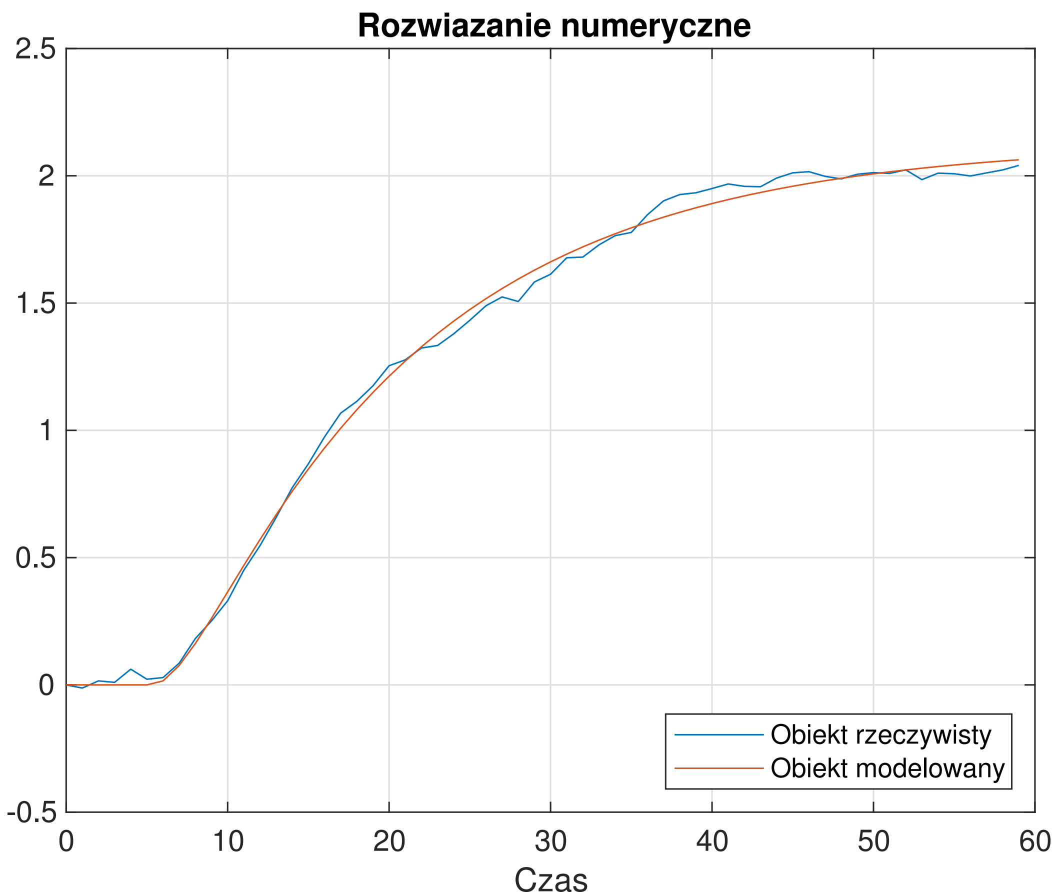

```
e = y - res;
err = sum(e.^2)/length(e);
fprintf('MSE: %.4f\n', err);
```

```
MSE: 0.0205
```

Drugim sposobem na znalezienie optymalnych parametrów modelu inercyjnego II rzędu z opóźnieniem jest wykorzystanie funkcji *fminsearch* oraz napisanej przeze mnie funkcji *ident2.m* - analogicznej jak w poprzednim zadaniu, z tą różnicą, że ma parametry przystosowane do modelu inercyjnego II rzędu. Po porównaniu wyników błędów średniokwadratowych obu metod, można zauważyć, że metoda numeryczna daje lepsze wyniki.

```
[parametry, blad] = fminsearch('ident2',[k,T1,T2,Theta]);
k = parametry(1);
T1 = parametry(2);
T2 = parametry(3);
Theta = parametry(4);
licz = k;
mian = [T1*T2 T1+T2 1];
obiekt2_n = tf(licz,mian);
set(obiekt2_n, 'outputdelay', Theta);
```

```
result = step(obiekt2_n, t);
plot(t,y,t,result)
grid on
legend("Obiekt rzeczywisty", "Obiekt modelowany", Location="southeast")
title('Rozwiazanie numeryczne')
xlabel("Czas")
```




```
fprintf('MSE: %.4f\n', blad);
```

```
MSE: 0.0013
```

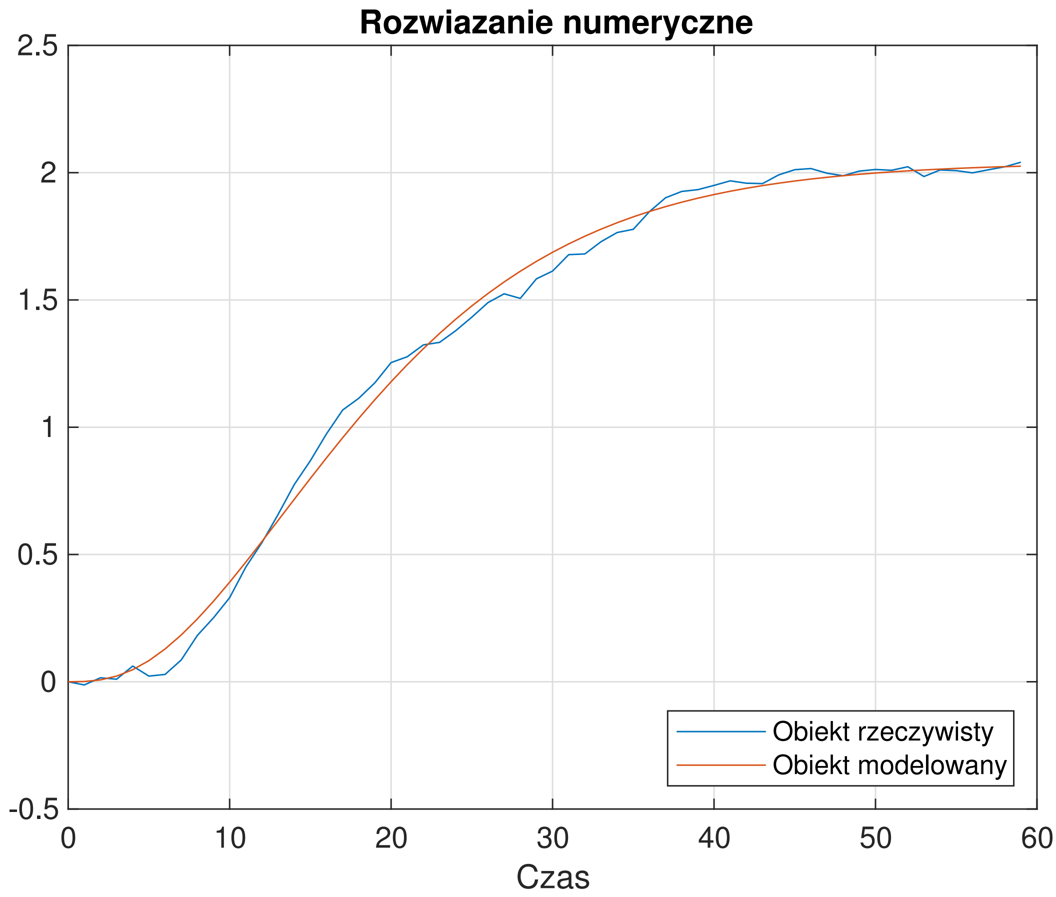
Model C - obiekt wieloinercyjny bez opóźnienia

Ostatnim modelem, który zostanie użyty do zamodelowania obiektu jest model wieloinercyjny bez opóźnienia. W tym celu postępuję analogicznie jak w poprzednich zadaniach, jednak tym razem muszę znaleźć optymalną wartość parametru n. W tym celu użyję funkcji *fminsearch* oraz napisanej przeze mnie funkcji *ident3.m*, która oblicza błąd średniokwadratowy między odpowiedzią rzeczywistą a odpowiedzią modelowaną. Po znalezieniu optymalnej wartości parametru n oraz optymalnych wartości parametrów k i T wyznaczyłem błąd średniokwadratowy, w celu porównania otrzymanego wyniku z pozostałymi.

```
% parametry k, T, n
k = 2.1;
T = 30;
n_values = 2:7;
for n = n_values
    global n;
    [~, blad] = fminsearch('ident3', [k,T]);
    disp(blad)
end
```

```
    0.0052

    0.0024

    0.0040

    0.0071

    0.0106

    0.0142
```

```
%widać, że najmniejszy błąd wystąpił dla n=3 oraz daleszemu wzrostowi n
%towarzyszył wzrost błędu
n = 3;
global n;
[parametry, blad] = fminsearch('ident3', [k,T]);
k = parametry(1);
T = parametry(2);
poles = (-1/T) .* ones(1, n);
obiekt = zpk([], poles, k/(T^n));
result = step(obiekt, t);
plot(t,y,t,result)
grid on
legend("Obiekt rzeczywisty", "Obiekt modelowany", Location="southeast")
title('Rozwiazanie numeryczne')
xlabel("Czas")
```




```
fprintf('MSE: %.4f\n', blad);
```

```
MSE: 0.0024
```

Używane funkcje

W zadaniach używałem poniższych funkcji do numerycznego wyznaczania parametrów modelu. Każda z nich wyznacza błąd średniokwadratowy między odpowiedzią rzeczywistą a odpowiedzią modelowaną. Jedynymi różnicami między poniższymi miejscami są parametry, które są przystosowane do konkretnych modeli obiektu rzeczywistego.

```
function blad = ident(X0)
k = X0(1);
T = X0(2);
Theta = X0(3);
%---------------------------------------- ------%
% tutaj kod, który będzie obliczał %
% odpowiedź skokową obiektu symulowanego %
% o takiej samej długości jak odpowiedź %
% obiektu rzeczywistego %
%---------------------------------------- ------%
load("obiekt.mat", 'y');
```

```
t = 0:length(y)-1;
obiekt = tf(k, [T,1],'OutputDelay', Theta);
y_sym = step(obiekt, t);
e = y - y_sym;
blad = sum(e.^2) / length(e);
```

```
function blad = ident2(X0)
k = X0(1);
T1 = X0(2);
T2 = X0(3);
Theta = X0(4);
%-------------------------------------- ------%
% tutaj kod, który będzie obliczał %
% odpowiedź skokową obiektu symulowanego %
% o takiej samej długości jak odpowiedź %
% obiektu rzeczywistego %
%-------------------------------------- ------%
load("obiekt.mat", 'y');
t = 0:length(y)-1;
obiekt = tf(k, [T1*T2 T1+T2 1],'OutputDelay', Theta);
y_sym = step(obiekt, t);
e = y - y_sym;
blad = sum(e.^2) / length(e);
```

```
function blad = ident3(X0)
k = X0(1);
T = X0(2);
global n;
%-------------------------------------- ------%
% tutaj kod, który będzie obliczał %
% odpowiedź skokową obiektu symulowanego %
% o takiej samej długości jak odpowiedź %
% obiektu rzeczywistego %
%-------------------------------------- ------%
load("obiekt.mat", 'y');
t = 0:length(y)-1;
poles = (-1/T) .* ones(1, n);
obiekt = zpk([], poles, k/(T^n));
y_sym = step(obiekt, t);
e = y - y_sym;
blad = sum(e.^2) / length(e);
```

Podsumowanie

Wszystkie modele, które zostały użyte do zamodelowania obiektu rzeczywistego, dały zadowalające wyniki. Wszystkie błędy średniokwadratowe są na akceptowalnym poziomie, jednak najlepsze wyniki dał model inercyjny II rzędu z opóźnieniem, co jest zgodne z moją oceną na podstawie wzrokowego porównania linii wykresu odpowiedzi skokowej obiektu rzeczywistego i modelowanego. Warto zauważyć, że metoda numeryczna daje lepsze wyniki niż metoda analityczna, co nie jest niespodzianką.