Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Sztuczna Inteligencja w Automatyce

Sprawozdanie z projektu numer 1, model numer 5

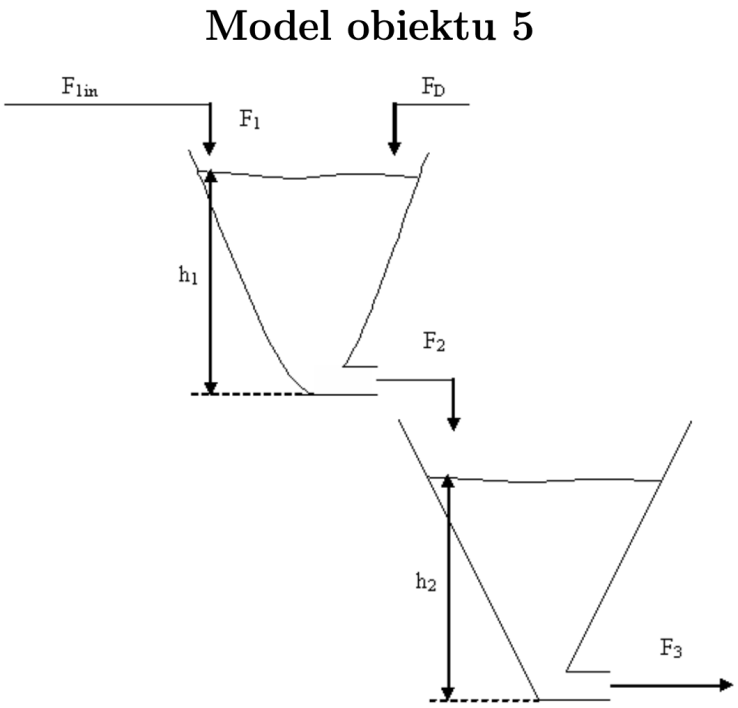
Górski Kacper 314061
Antoni Marczuk 304436

Warszawa, 2025

Spis treści

1 Rozpatrywany model

Rozważany model odpowiada układowi dwóch połączonych zbiorników, w których zachodzi przepływ cieczy. Dopływ sterujący $F_1(t)$ oraz dopływ zakłócający $F_D(t)$ wpływają do pierwszego zbiornika, natomiast odpływ ze zbiornika pierwszego stanowi dopływ do drugiego zbiornika. Z drugiego zbiornika ciecz wypływa do otoczenia. Wielkością regulowaną jest poziom cieczy $h_2(t)$ w drugim zbiorniku.



Rys. 1: Schemat modelu obiektu (Model 5)

1.1 Równania modelu

Zależności opisujące zmiany objętości cieczy w czasie przedstawiono poniżej:

$$\begin{cases} \dfrac{dV_1}{dt} = F_1 + F_D - F_2(h_1), \\ \dfrac{dV_2}{dt} = F_2(h_1) - F_3(h_2), \end{cases} \tag{1}$$

gdzie:

$$F_1(t) = F_{in}(t-\tau) \qquad F_2(h_1) = \alpha_1\sqrt{h_1}, \qquad F_3(h_2) = \alpha_2\sqrt{h_2},$$
$$V_1(h_1) = C_1 h_1^2, \qquad V_2(h_2) = C_2 h_2^3.$$

W celu uzyskania równań stanu obiektu w formie pochodnych zmiennych h_1 i h_2 obliczamy pochodne po czasie danych funkcji objętości i podstawiamy je do wzorów.

$$\frac{dV_1}{dt} = 2C_1 h_1 \frac{dh_1}{dt}, \quad \frac{dV_2}{dt} = 3C_2 (h_2)^2 \frac{dh_2}{dt}$$

Po podstawieniu zależności otrzymujemy nieliniowy układ równań różniczkowych opisujących dynamikę poziomów cieczy:

$$\begin{cases} \dfrac{dh_1}{dt} = \dfrac{F_{in}(t-\tau) + F_D - \alpha_1\sqrt{h_1}}{2C_1h_1}, \\ \dfrac{dh_2}{dt} = \dfrac{\alpha_1\sqrt{h_1} - \alpha_2\sqrt{h_2}}{3C_2h_2^2}. \end{cases} \tag{2}$$

1.2 Parametry i stałe modelu

Stałe fizyczne przyjęte dla modelu obiektu przedstawiono w tabeli 1.

Tab. 1: Parametry modelu obiektu

Symbol	Wartość	Opis
C_1	0,35	Stała geometryczna zbiornika 1
C_2	0,45	Stała geometryczna zbiornika 2
α_1	23	Współczynnik przepływu między zbiornikami
α_2	30	Współczynnik odpływu ze zbiornika 2
τ	125 s	Opóźnienie dopływu sterującego

Ostatecznie model wygląda następująco:

$$\begin{cases} \dfrac{dh_1}{dt} = \dfrac{F_1 + Fd - 23\sqrt{h_1}}{0.7h_1}, \\ \dfrac{dh_2}{dt} = \dfrac{23\sqrt{h_1} - 30\sqrt{h_2}}{1.35h_2^2}, \end{cases} \tag{3}$$

gdzie $F1(t) = F_{in}(t - 125)$.

2 Zadanie 1

2.1 Linearyzacja

Dla wskazanego punktu pracy:

$$\overline{F_1} = 200 \text{ cm}^3/\text{s}, \quad \overline{F_D} = 100 \text{ cm}^3/\text{s},$$

układ osiąga stan równowagi spełniający warunki:

$$\frac{dh_1}{dt} = 0, \quad \frac{dh_2}{dt} = 0.$$

Na tej podstawie wyznaczono wartości poziomów cieczy:

$$\overline{h_1} = \left(\frac{200+100}{23}\right)^2 \approx 170{,}3 \text{ cm}, \qquad \overline{h_2} = \left(\frac{200+100}{30}\right)^2 = 100 \text{ cm}.$$

Dalej stosujemy rozwinięcie Taylora do wyrazów pierwszego rzędu. W tym celu liczymy pochodne:

$$\frac{df_1}{dh_1} = -\frac{1.42857142857143\left(F_1 + Fd - 23\sqrt{h_1}\right)}{h_1^2} - \frac{16.4285714285714}{h_1^{\frac{3}{2}}}$$

$$\frac{df_1}{dh_2} = 0, \quad \frac{df_1}{dF_1} = \frac{1.42857142857143}{h_1}, \quad \frac{df_1}{dF_d} = \frac{1.42857142857143}{h_1}$$

$$\frac{df_2}{dh_2} = -\frac{1.48148148148148\left(23\sqrt{h_1} - 30\sqrt{h_2}\right)}{h_2^3} - \frac{11.1111111111111}{h_2^{\frac{5}{2}}}$$

$$\frac{df_2}{dh_1} = \frac{8.51851851851852}{\sqrt{h_1}h_2^2}, \quad \frac{df_2}{dF_1} = 0, \quad \frac{df_2}{dF_d} = 0$$

Podstawiając wartości punktu pracy dostajemy model zlinearyzowany:

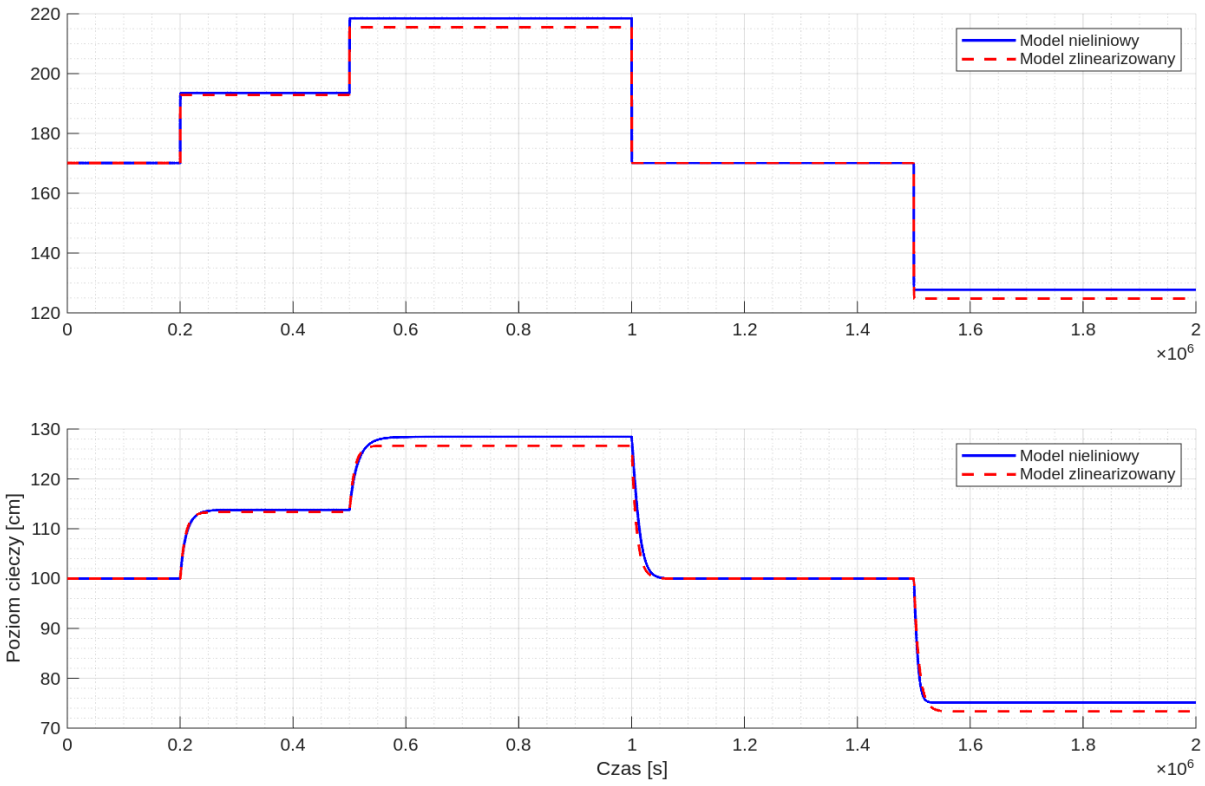
$$\frac{dh}{dt} = A(h - \overline{h}) + B(u - \overline{u}),$$

gdzie $u_1 = F_1$ i $u_2 = F_d$.

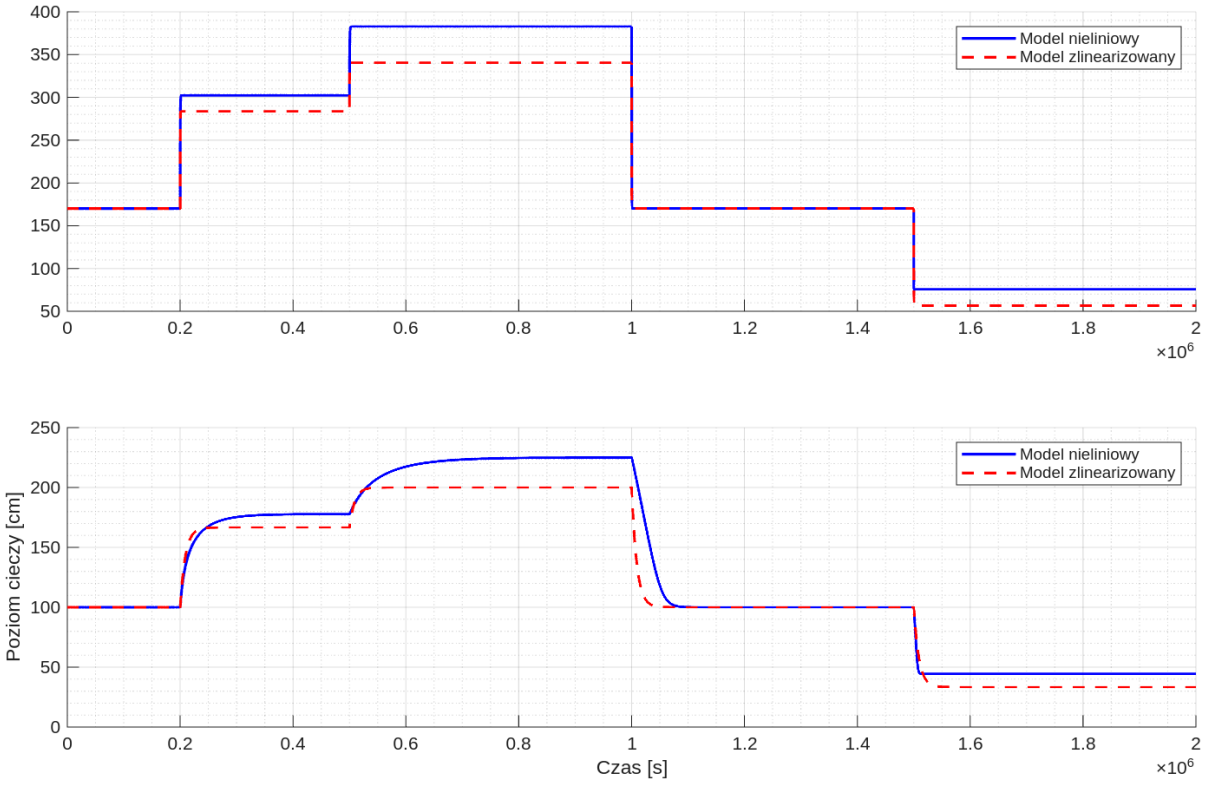
2.2 Symulacja i porównanie

Poniższe symulacje wykonaliśmy w skrypcie NonlinearVsLinear.m

Przy niewielkich odchyleniach od wyznaczonego punktu linearyzacji model liniowy dobrze przybliża model nieliniowy jak na rys. 2. Ale przy większych odchyleniach model zlinearyzowany zaczyna mocno odbiegać od rzeczywistej pracy obiektu co widzimy na rys. 3. Pierwszy wykres z pary przestawia h_1 a drugi wyjście modelu czyli h_2.


Model nieliniowy
Model zlinearizowany
220
200
180
160
140
120
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
×10^6
Model nieliniowy
Model zlinearizowany
Poziom cieczy [cm]
130
120
110
100
90
80
70
Czas [s]
×10^6

Rys. 2:


Model nieliniowy
Model zlinearizowany
400
350
300
250
200
150
100
50
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
×10^6
Model nieliniowy
Model zlinearizowany
Poziom cieczy [cm]
250
200
150
100
50
0
Czas [s]
×10^6

Rys. 3:

Można zauważyć, że macierze linearyzacji A i B zależą od sumy $\overline{F_1}+\overline{F_d}$ a nie od samych wartości z osobna. Z naszych eksperymentów wynika, że model zlinearyzowany całkiem nieźle przybliża obiekt nieliniowy o ile wartość $F_1 + F_d$ nie oddala się od punktu linearyzacji o więcej nić 20%.

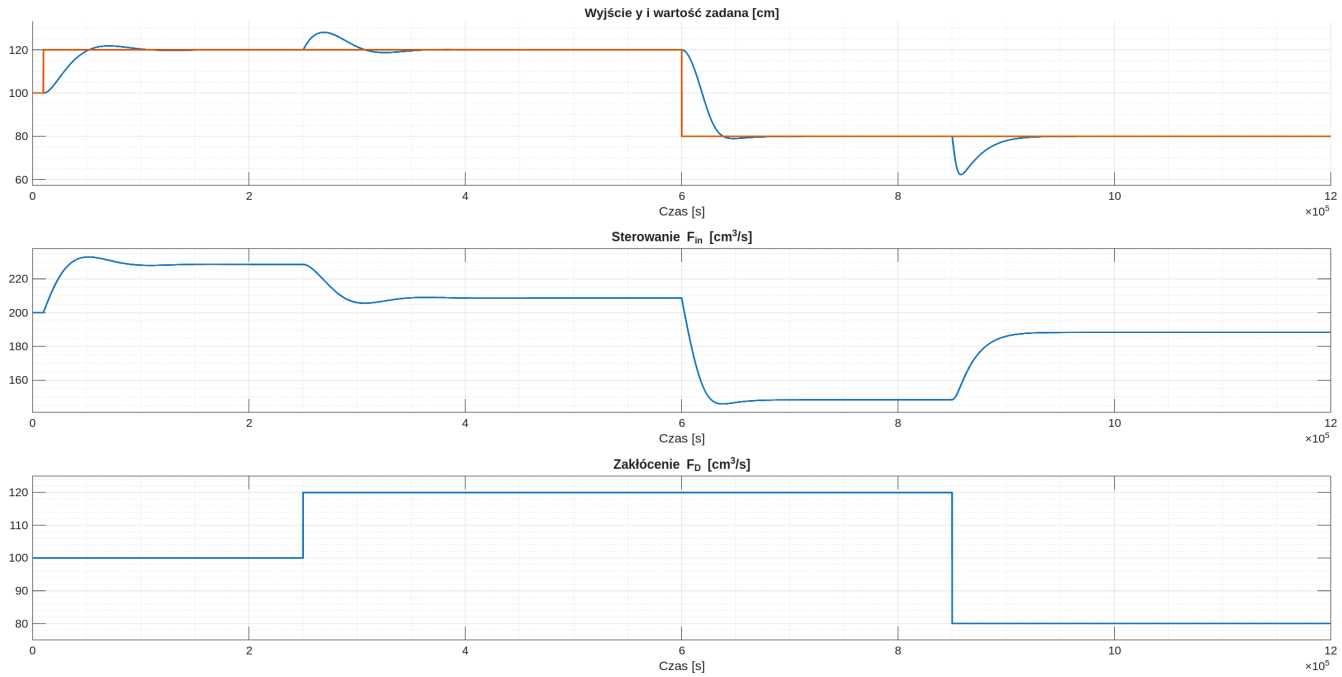
Symulacje pokazują, że sterowana wielkość h_2 reaguje znacznie wolniej na wejście niż h_1.

2.3 Liniowy regulator DMC

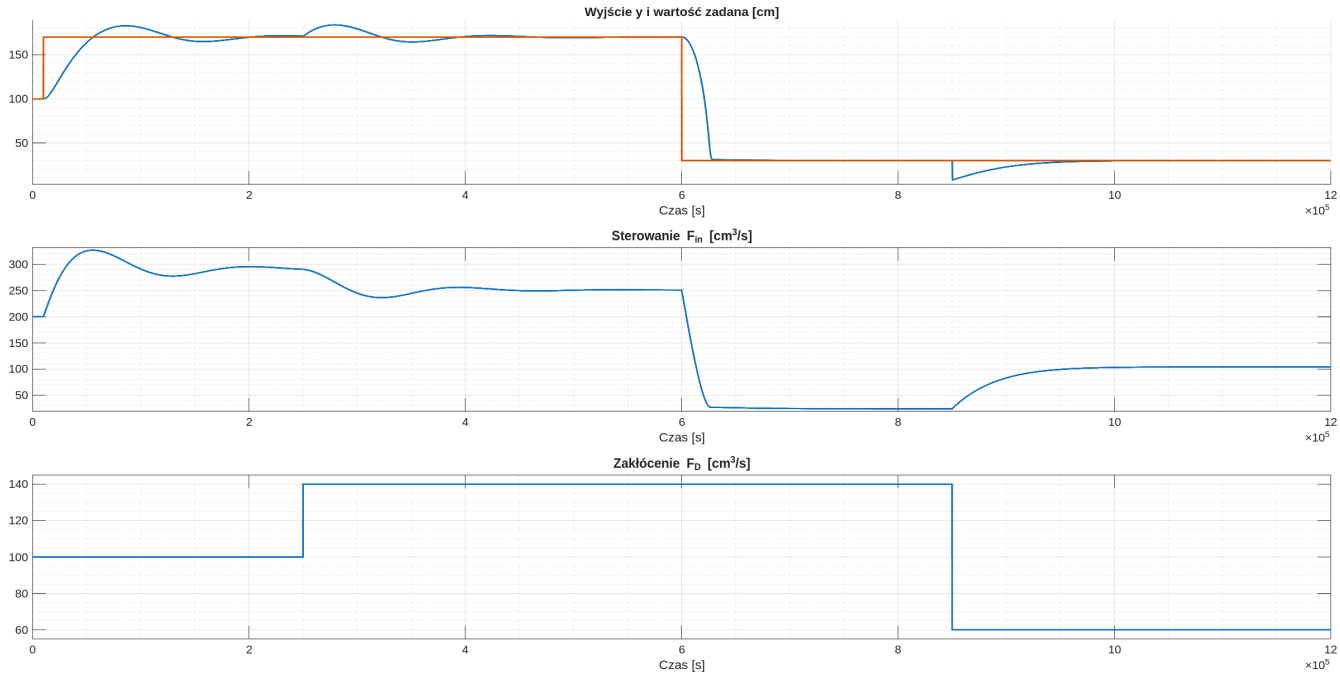
Poniższe symulacje wykonaliśmy w skrypcie DMC.m

Mając model zlinearyzowany zaprojektowaliśmy dla niego regulator DMC. Regulator ten całkiem dobrze sprawdza się dla modelu nieliniowego o ile znajdujemy się blisko punktu linearyzacji. W momencie gdy oddalamy się od tego punktu jakość regulacji niestety znacząco spada.

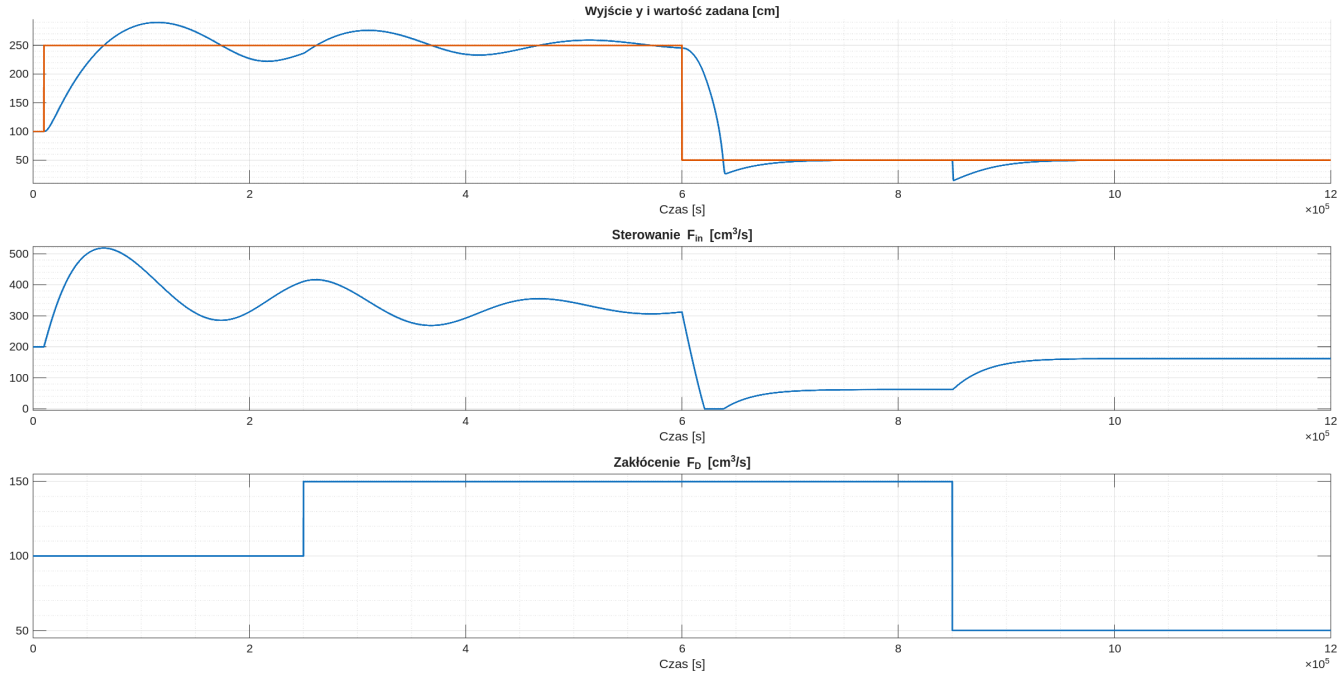
Poniżej przedstawiamy trzy przypadki co raz bardziej oddalające się od punktu pracy. Dla pierwszego mamy całkiem niezłą regulację, natomiast dla ostatniego już jest ona niezadowalająca.



Rys. 4:


Wyjście y i wartość zadana [cm]
150
100
50
0
2
4
6
8
10
12
Czas [s]
×10^5
Sterowanie F_in [cm^3/s]
300
250
200
150
100
50
0
2
4
6
8
10
12
Czas [s]
×10^5
Zakłócenie F_D [cm^3/s]
140
120
100
80
60
0
2
4
6
8
10
12
Czas [s]
×10^5

Rys. 5:


Wyjście y i wartość zadana [cm]
250
200
150
100
50
0
2
4
6
8
10
12
Czas [s]
×10^5
Sterowanie F_in [cm^3/s]
500
400
300
200
100
0
2
4
6
8
10
12
Czas [s]
×10^5
Zakłócenie F_D [cm^3/s]
150
100
50
0
2
4
6
8
10
12
Czas [s]
×10^5

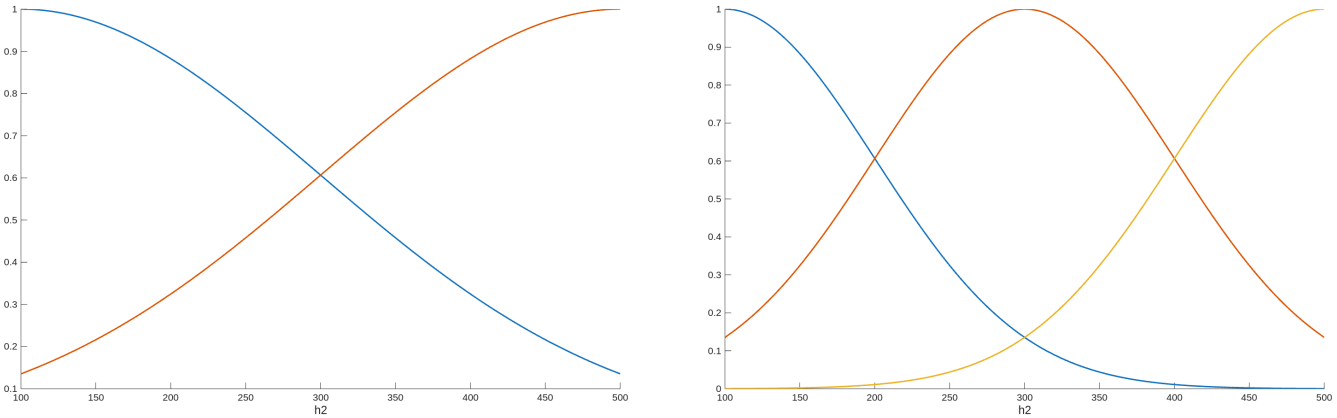
Rys. 6:

3 Zadanie 2

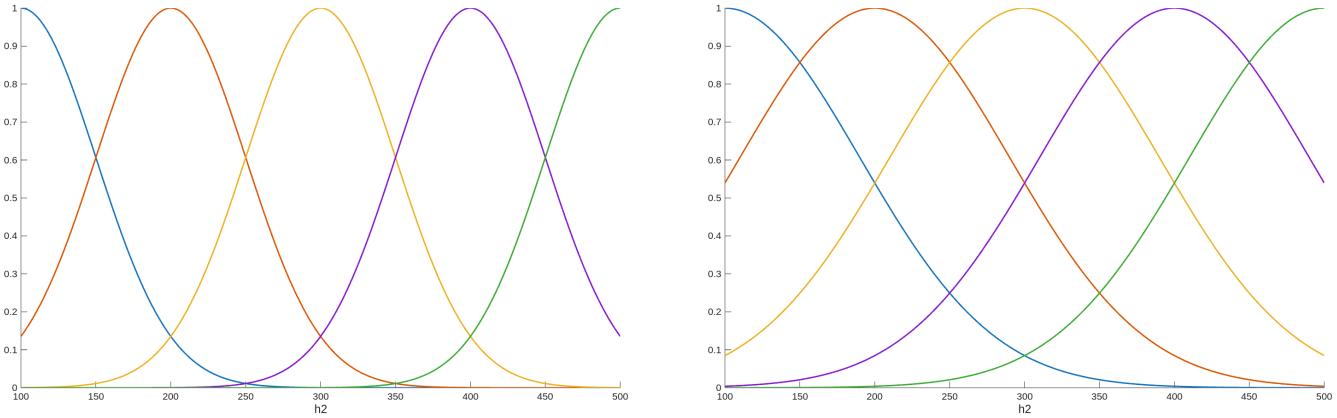
3.1 Model rozmyty

Poniższe symulacje uzyskaliśmy w skrypcie FuzzyModelh2.m

Dokonaliśmy aproksymacji modelu nieliniowego za pomocą rozmytego modelu Takagi-Sugeno gdzie lokalne modele liniowe otrzymaliśmy z linearyzacji w kilku punktach pracy. Początkowo próbowaliśmy rozmywać sumę $F1 + F_d$, później kolejno F_1, h_2 i h_1. Najlepsze dopasowanie wyjścia modelu było przy rozmywaniu h_2 czyli wyjścia. Wykorzystaliśmy równomiernie rozłożone gaussowskie funkcje przynależności i najlepsze rezultaty dało $\sigma = 0.5 \cdot (rozstaw\_punktow\_linearyzacji)$.



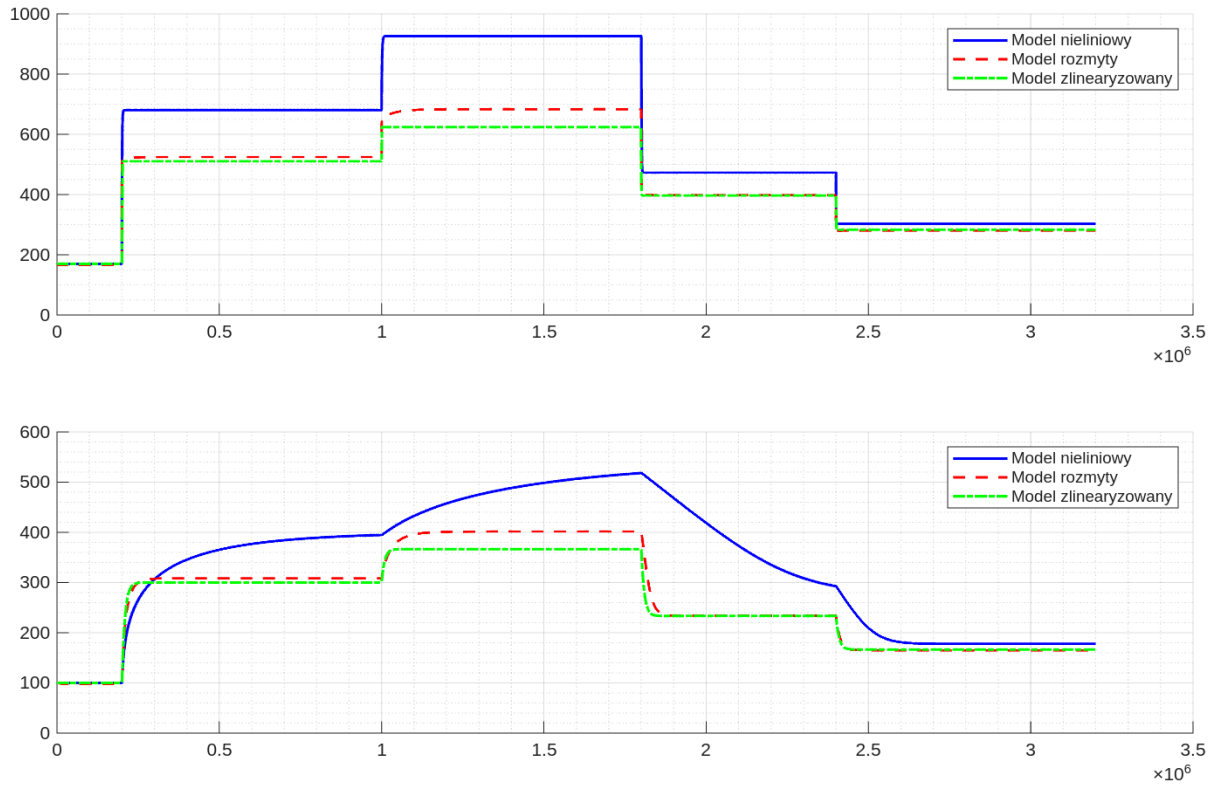
Rys. 7:



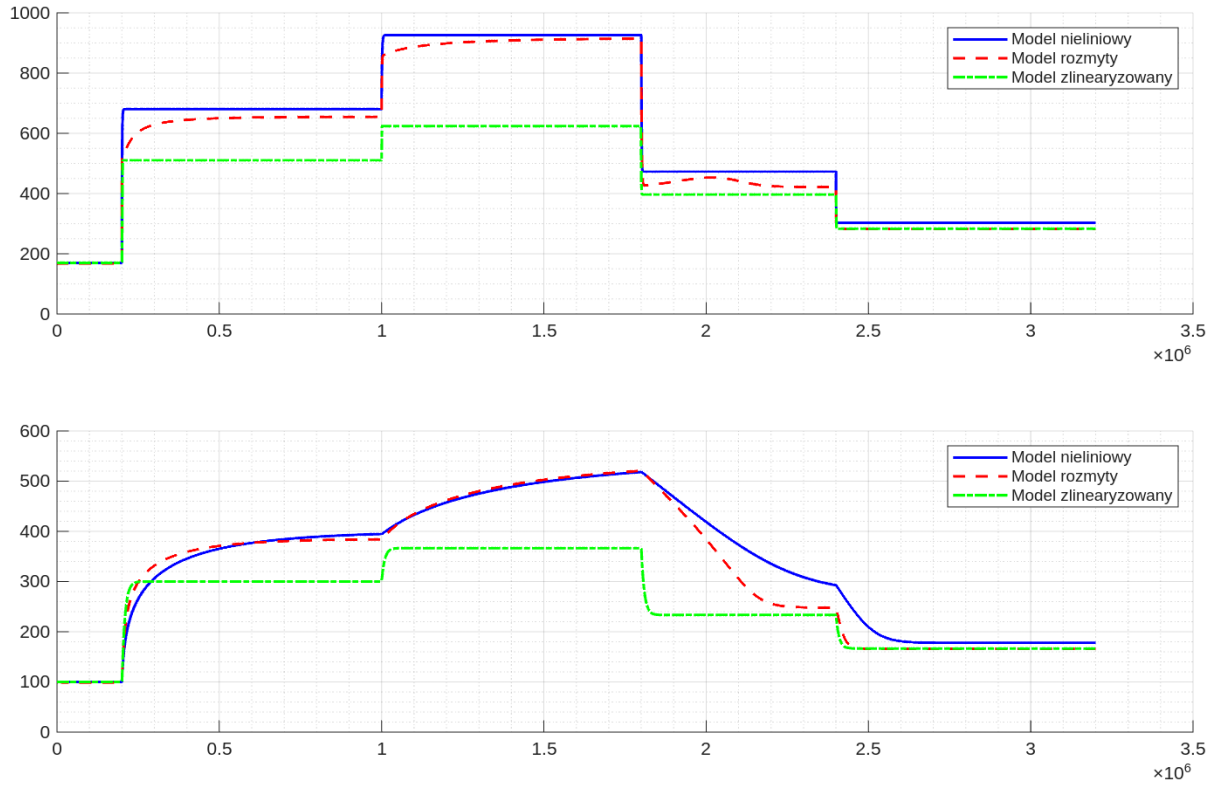
Rys. 8:

Poniżej przedstawiamy dopasowanie modelu kolejno dla 2, 3, 4 i 5 zbiorów rozmytych przy rozstawie $\sigma = 0.5$. Przedstawiamy zarówno dopasowanie h_1 jak i h_2, ale do regulacji predykcyjnej tak naprawdę interesuje nas jedynie dopasowanie wyjścia, czyli h_2.

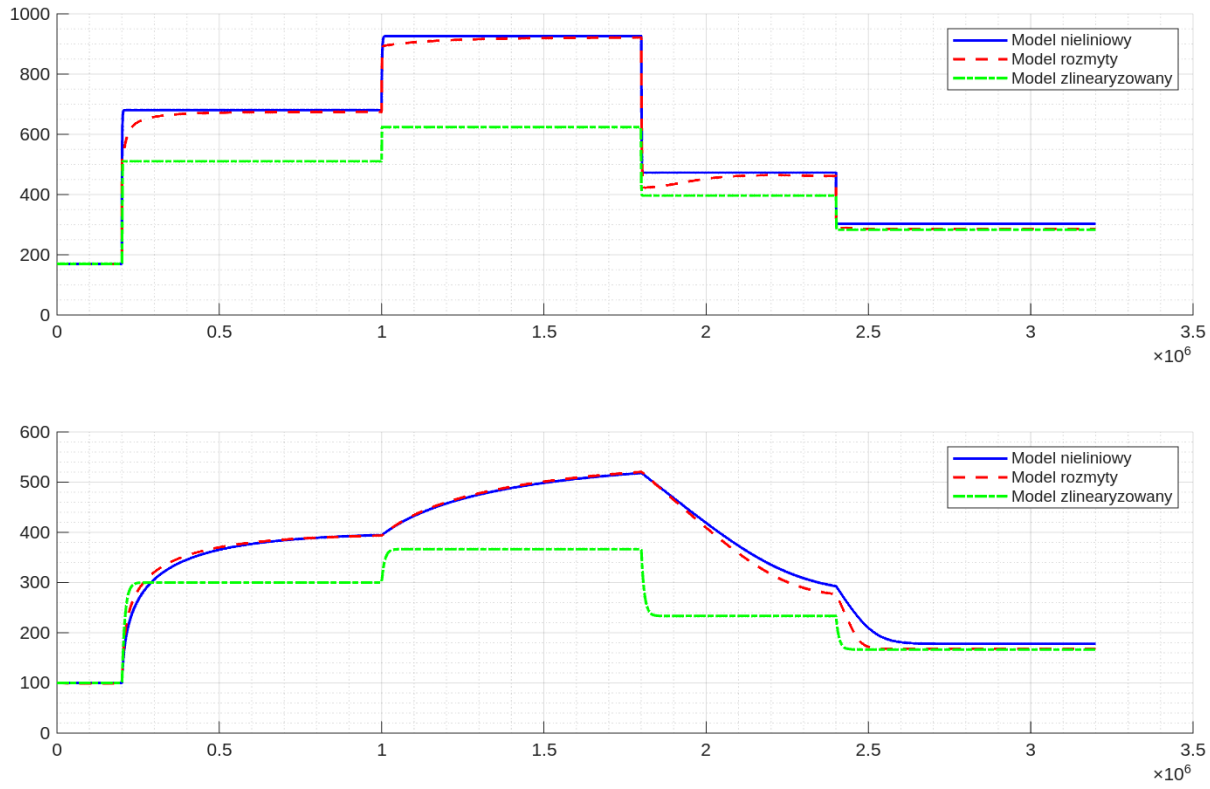
Widać, że już nawet przy czterech zbiorach rozmytych mamy dość dobre dopasowanie wyjścia modelu.


Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6
Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6

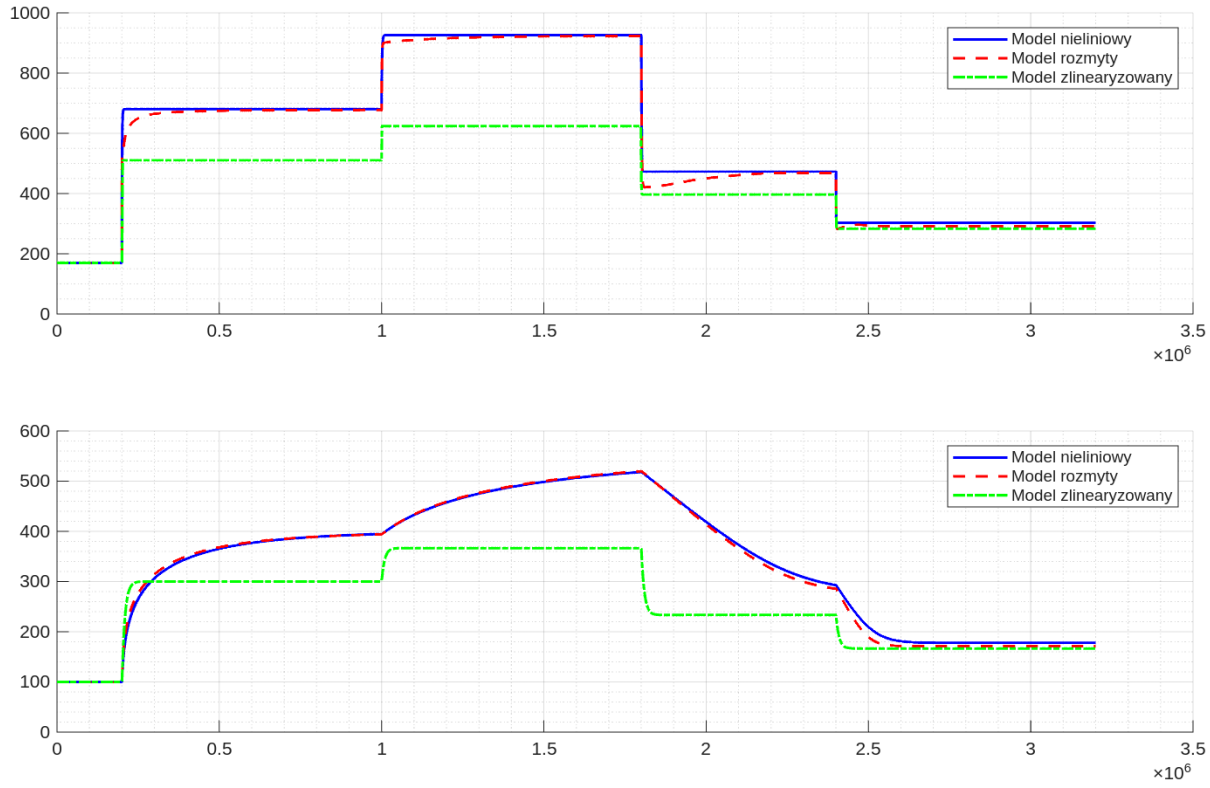
Rys. 9:


Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6
Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6

Rys. 10:


Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6
Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6

Rys. 11:


Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6
Model nieliniowy
Model rozmyty
Model zlinearyzowany
×10^6

Rys. 12:

Jak widać model rozmyty działa zdecydowanie lepiej niż model zlinearyzowany tylko w jednym punkcie. Oczywiście przy niewielkich odchyleniach wszystkie trzy modele będą się zachowywać podobnie, ale my chcemy mieć dobre dopasowanie w szerokim zakresie zmienności wyjścia.

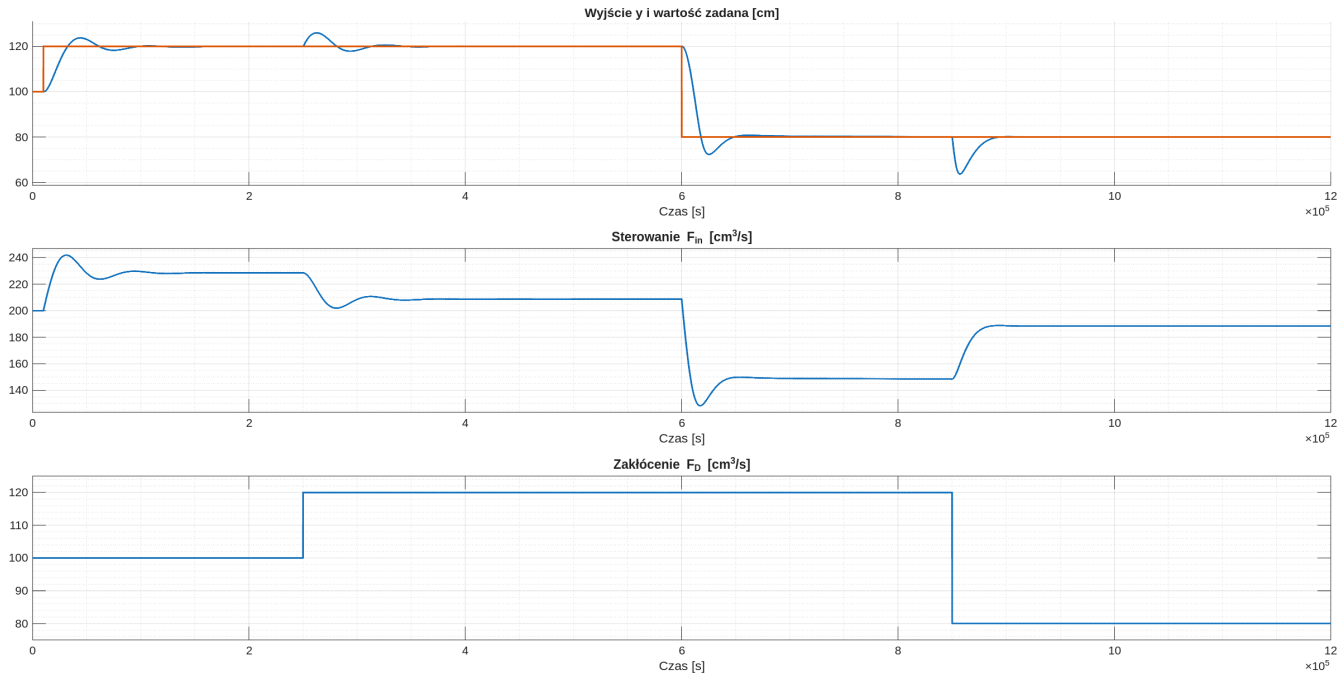
Dalej do projektowania regulatorów rozmytych stosujemy pięć gaussowskich zbiorów rozmytych.

3.2 Rozmyty regulator DMC

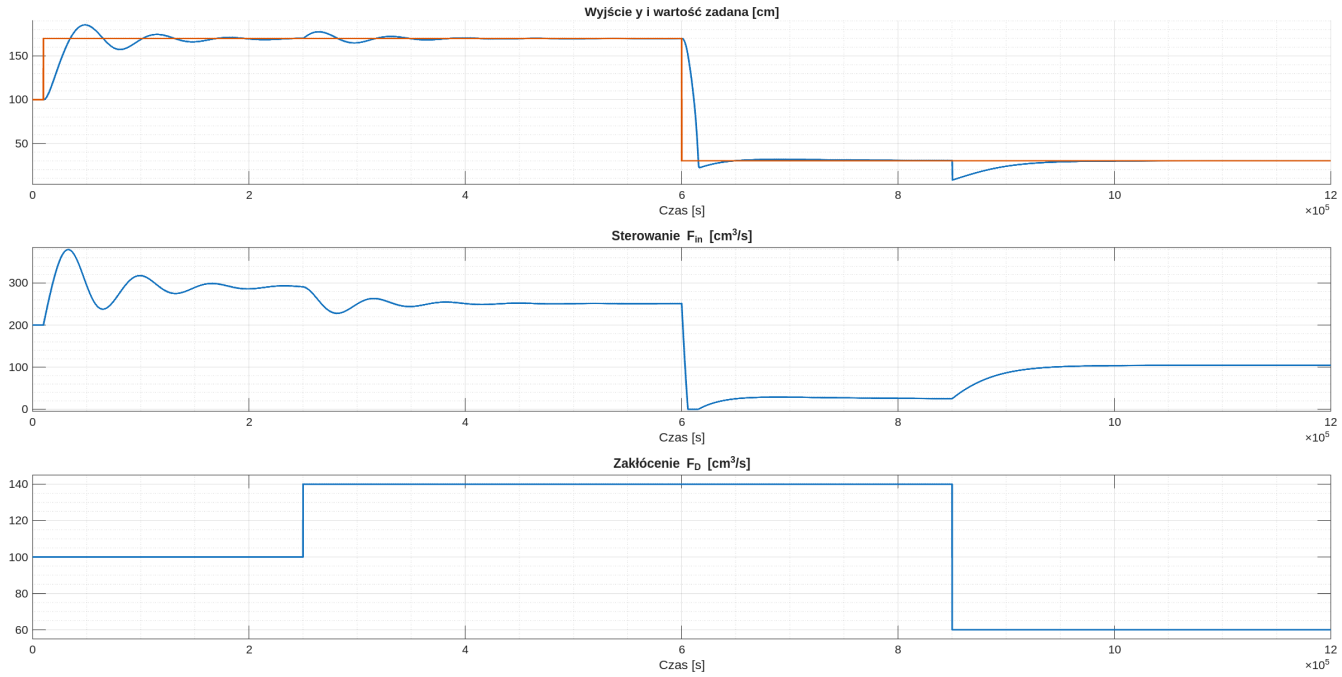
Poniższe wyniki uzyskaliśmy w skrypcie DMC_Fuzzy.m

Zrobiliśmy regulator rozmyty DMC korzystając z lokalnych modeli liniowych. Regulator ten pracuje znacznie lepiej niż zwyczajny DMC przy wartościach znacznie oddalających się od punkty pracy liniowego DMC (w pobliżu tego punktu pracują podobnie).

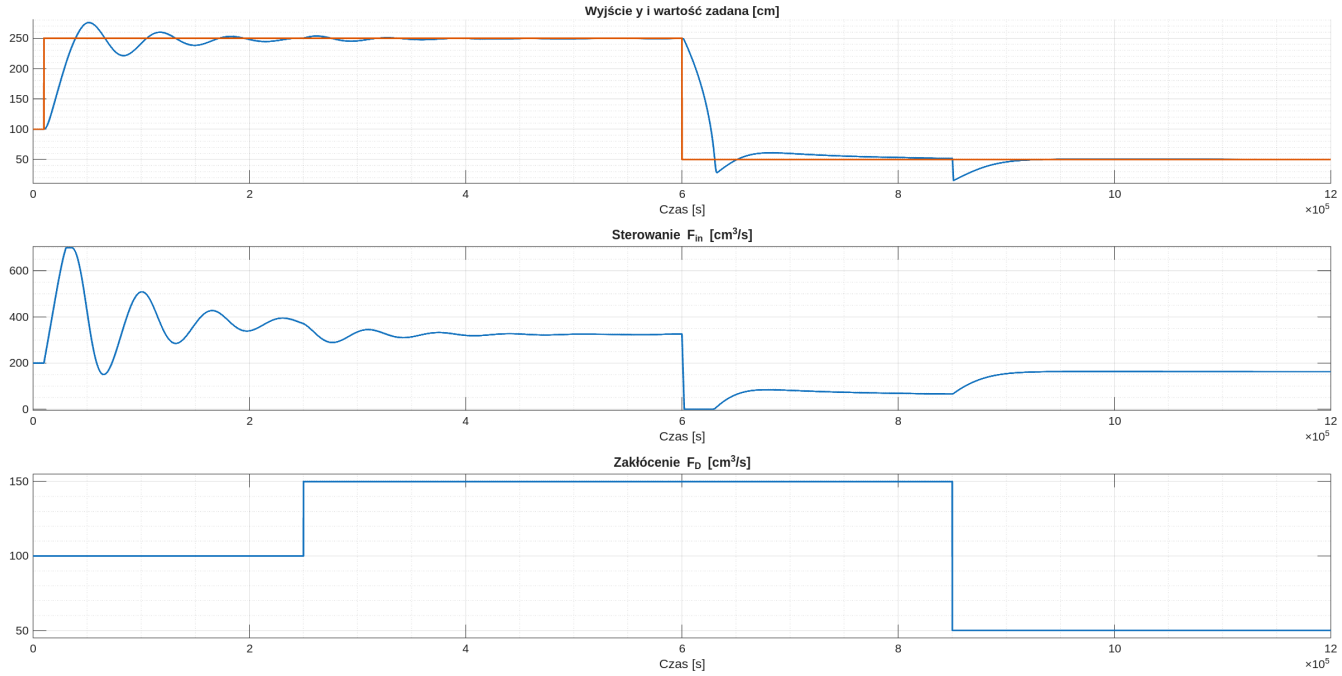
Z symulacji wynika, że rozmyty regulator ma większą tendencję do oscylacji, dlatego warto zwiększyć współczynnik kary λ.



Rys. 13:

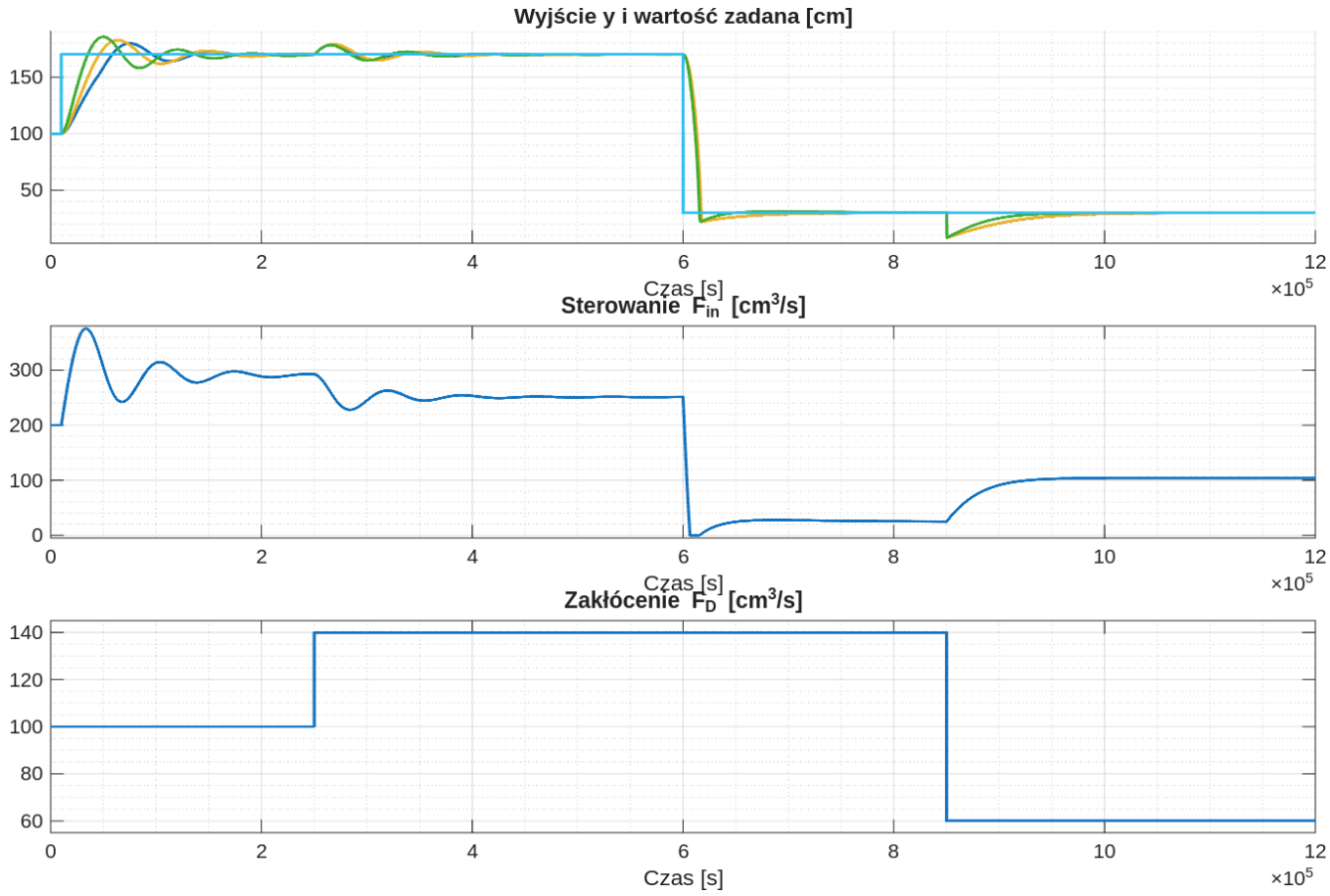


Rys. 14:



Rys. 15:

Spróbowaliśmy nieco dostroić funkcje przynależności dla regulatora, ale nie dało to jakieś super widocznej poprawy regulacji. Generalnie zmiany rozrzutu powodowały nieznaczne skrócenie lub wydłużenie regulacji oraz odpowiednio zwiększenie i zmniejszenie przeregulowania.


Wyjście y i wartość zadana [cm]
150
100
50
0
2
4
6
8
10
12
Czas [s]
×10^5
Sterowanie F_in [cm^3/s]
300
200
100
0
0
2
4
6
8
10
12
Czas [s]
×10^5
Zakłócenie F_D [cm^3/s]
140
120
100
80
60
0
2
4
6
8
10
12
Czas [s]
×10^5

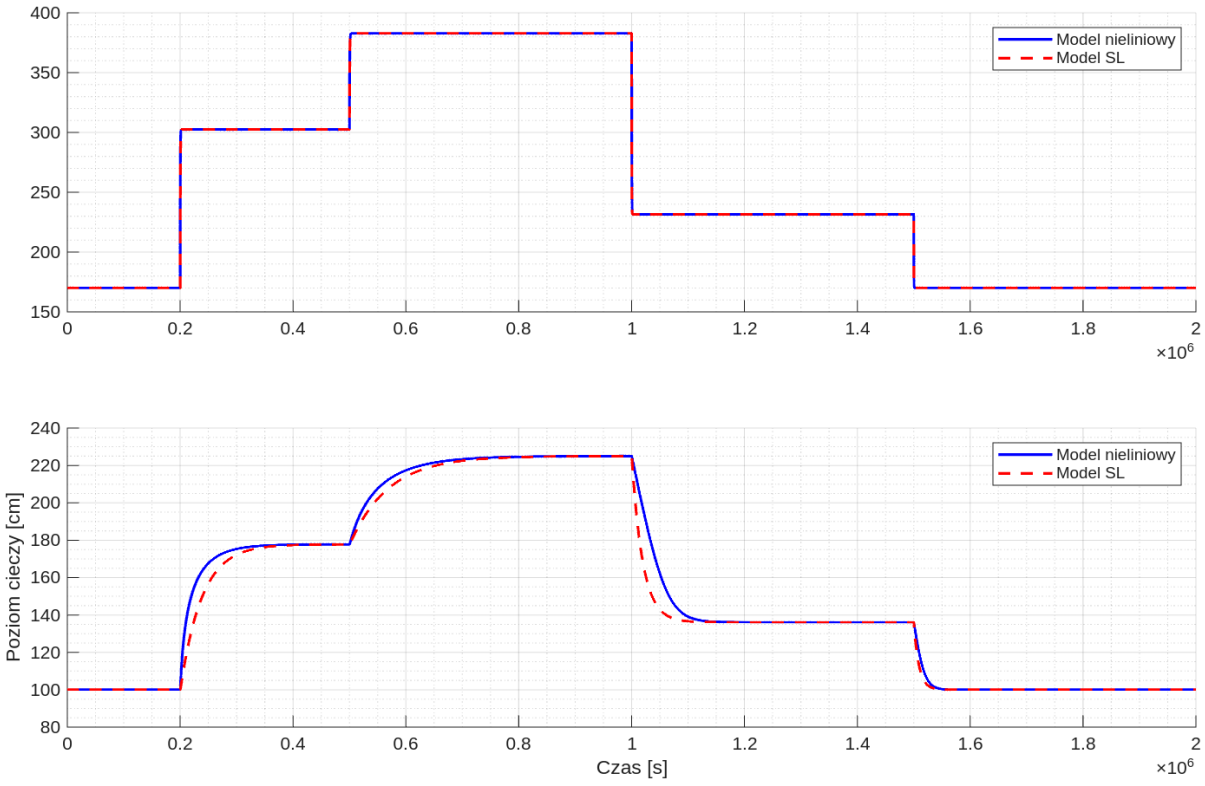
Rys. 16:

4 Zadanie 3

4.1 Model SL

Model ten uzyskaliśmy w skrypcie ModelSL.m

Zrobiliśmy model z sukcesywną linearyzacją w każdym kroku próbkowania. Model ten dobrze przybliża model nieliniowy i to w dowolnym zakresie zmienności dowolnych zmiennych. Wadą jest konieczność linearyzacji w każdym kroku, a więc koszt obliczeń.



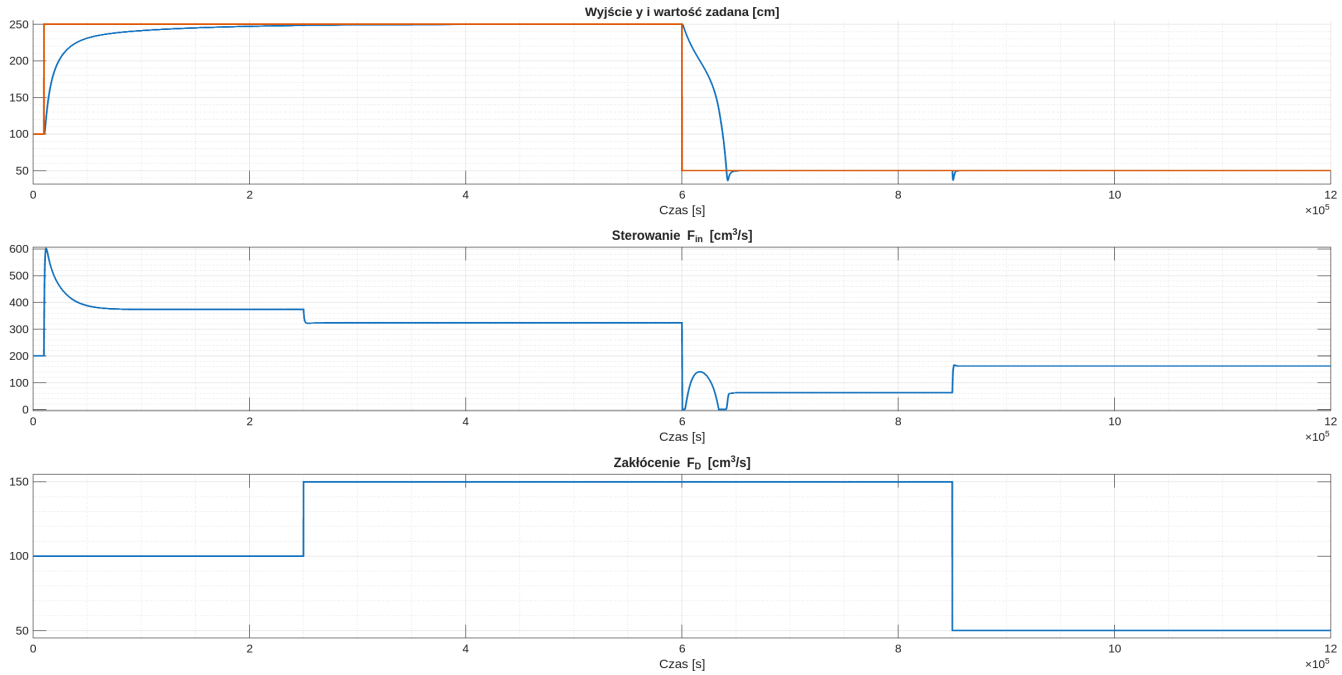
Rys. 17:

4.2 Numeryczny regulator DMC-SL

Poniższe wyniki uzyskaliśmy w skryptach DMC_SL_n.m oraz DMC_SL_a.m

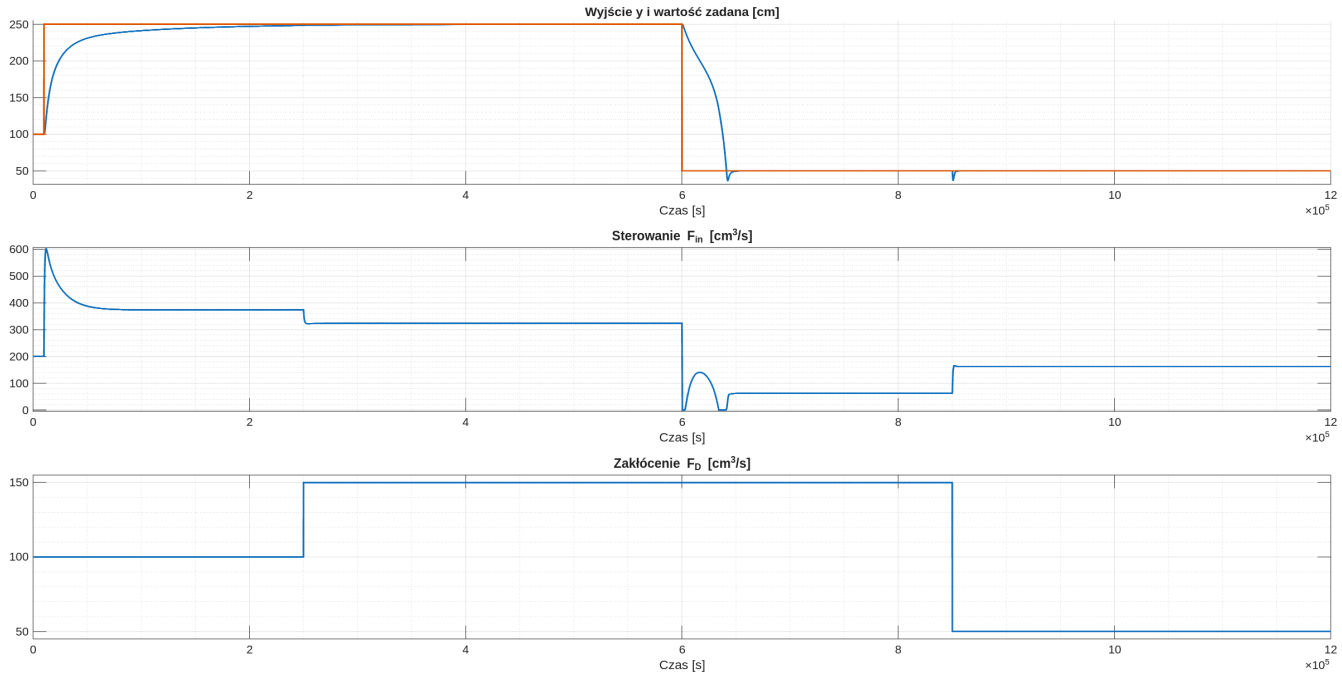
Regulator DMC-SL w każdym kroku próbkowania na nowo linearyzuje model i wyznacza nowe parametry regulatora DMC. Dzięki temu znakomicie sobie radzi w dowolnym zakresie zmian wartości, ale wadą jest koszt obliczeń. Poniżej widać, że regulacja jest lepsza od poprzednich rozważanych regulatorów. Możemy regulować jego szybkość przez nakładanie ograniczeń w zadaniu optymalizacji oraz parametr λ.

W naszym skrypcie do optymalizacji wykorzystujemy wbudowany solver matlaba do programowania kwadratowego *quadprog*.



Rys. 18:

Zaprojektowaliśmy też regulator DMC-SL analityczny z rzutowaniem ograniczeń. Wynik jego działania jest niemal identyczny.

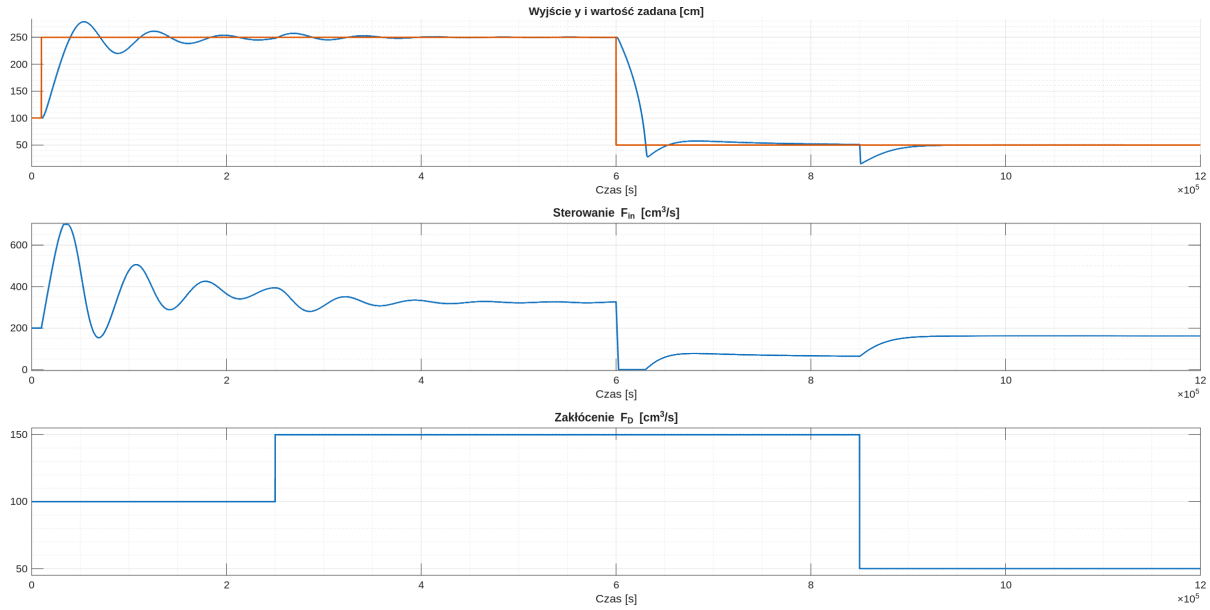


Rys. 19:

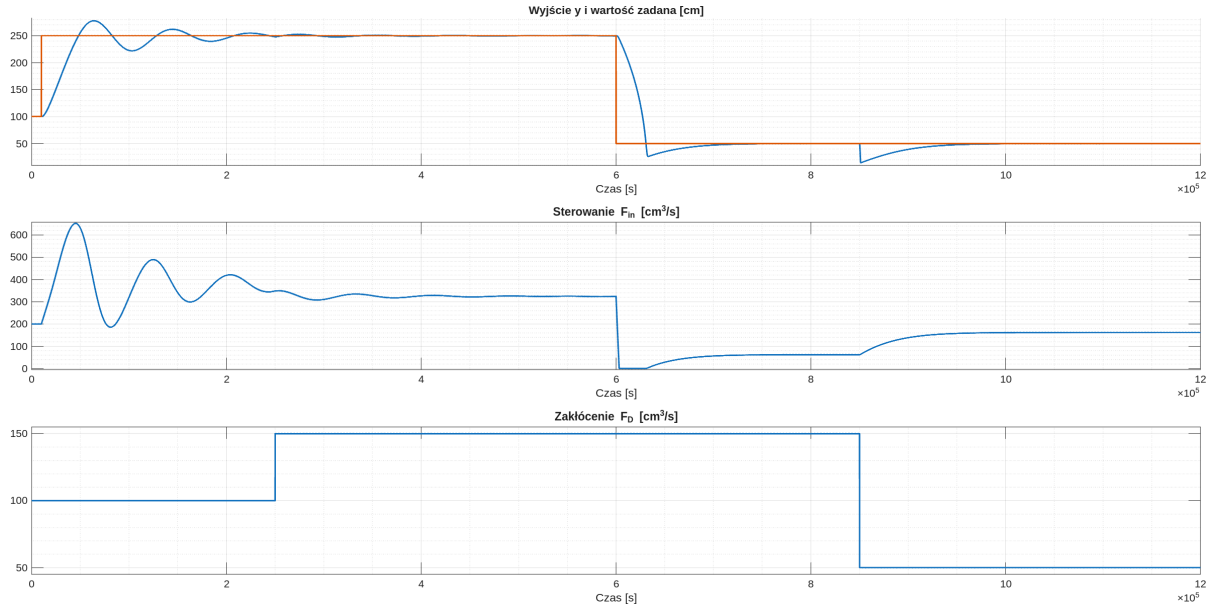
5 Zadanie 4

Skrypty wykorzystane do tego zadania to wszystkie wcześniej wymienione skrypty, gdyż od razu projektowaliśmy regulatory DMC uwzględniając historię zakłóceń. W tym zadaniu w celu porównania, wyłączyliśmy te fragmenty, przez zakomentowanie.

Poniżej widzimy najpierw działanie regulatora rozmytego DMC bez historii zakłóceń, a następnie z uwzględnieniem tej historii.

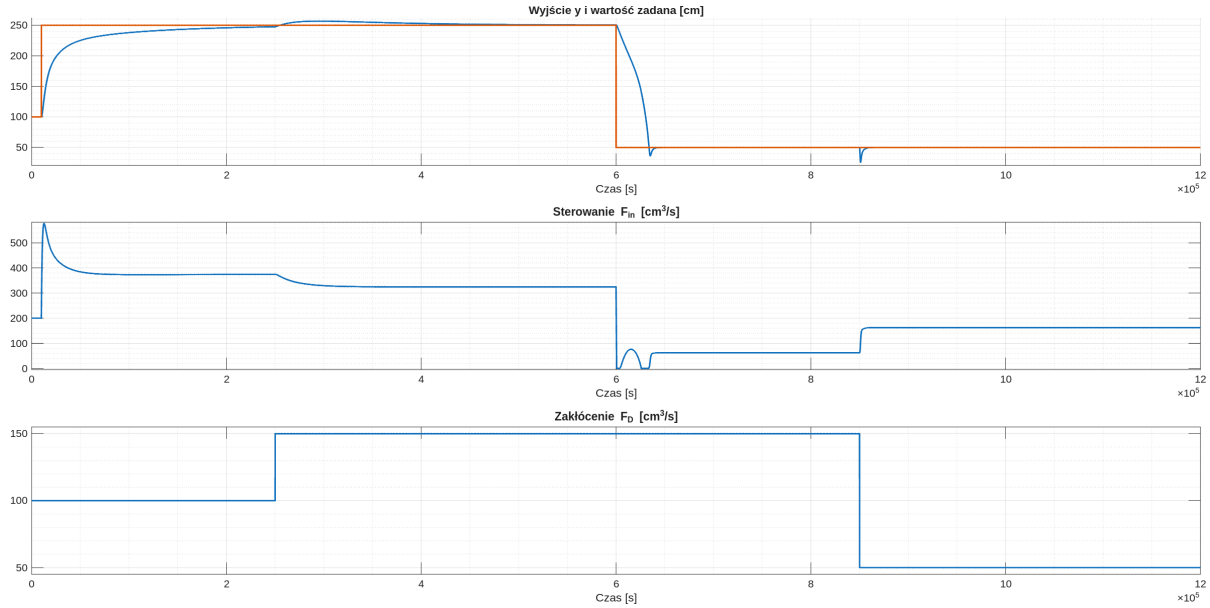


Rys. 20:

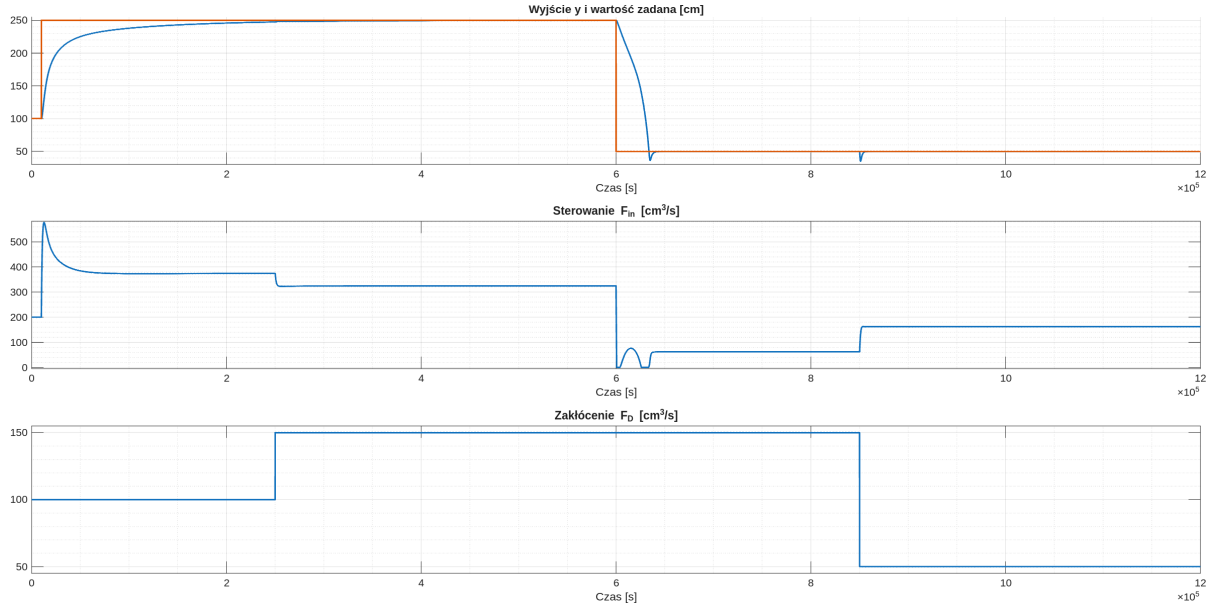


Rys. 21:

Poniżej widzimy najpierw działanie regulatora DMC-SL bez historii zakłóceń, a następnie z uwzględnieniem tej historii.



Rys. 22:



Rys. 23:

A zatem uwzględniając historię zakłóceń uzyskujemy nieco szybsze i lepsze działanie regulatora na ich zmianę. W tym modelu nie jest to może jakaś drastyczna poprawa, ale jest zauważalna.

6 Podsumowanie

Z przeprowadzonych symulacji wynika, że liniowy regulator DMC dobrze pracuje dla modelu nieliniowego o ile nie oddalamy się zbytnio od punktu pracy dla którego model był linearyzowany. Jeśli zależy nam na szerszym zakresie zmienności zmiennych to musimy zastosować regulator rozmyty DMC lub DMC-SL. Regulator DMC-SL daje nalepszą jakość regulacji, ale wymaga dużych nakładów obliczeń (nasz skrypt z symulacją po uruchomieniu liczył od kilkanastu minut w górę). W celu zmniejszenia liczby obliczeń można np. stosować linearyzacjię tylko co kilka okresów próbkowania np. co trzy okresy - przeprowadziliśmy kilka takich eksperymentów i obliczenia trochę przyspieszyły z niewielkim spadkiem jakości regulacji. Regulator rozmyty daje gorszą jakość regulacji niż SL, ale za to jest szybki, gdyż wystarczy na etapie projektowania policzyć macierze dynamiczne, a później już je tylko wykorzystujemy do obliczeń. Ponadto regulatory rozmyte zapewne można zaprojektować lepiej niż zostało to zrobione w naszym projekcie - my robiliśmy linearyzację w kilku punktach i dopasowywaliśmy równo rozstawione funkcje przynależności, a przy poważnym projektowaniu warto by wykorzystać metody optymalizacji i uczenia maszynowego do lepszego dopasowania modelu Takagi-Sugeno (w tym również funkcji przynależności).

Wszystkie obliczenia numeryczne zostały wykonane w MATLAB. Do obliczeń symbolicznych posłużył nam SymPy.