Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania (projekt grupowy)

Sprawozdanie z projektu i ćwiczenia laboratoryjnego 3, zadanie nr 7

Kacper Marchlewicz, Michał Kwarciński, Adam Wróblewski
grupa nr 7

Warszawa, 2022

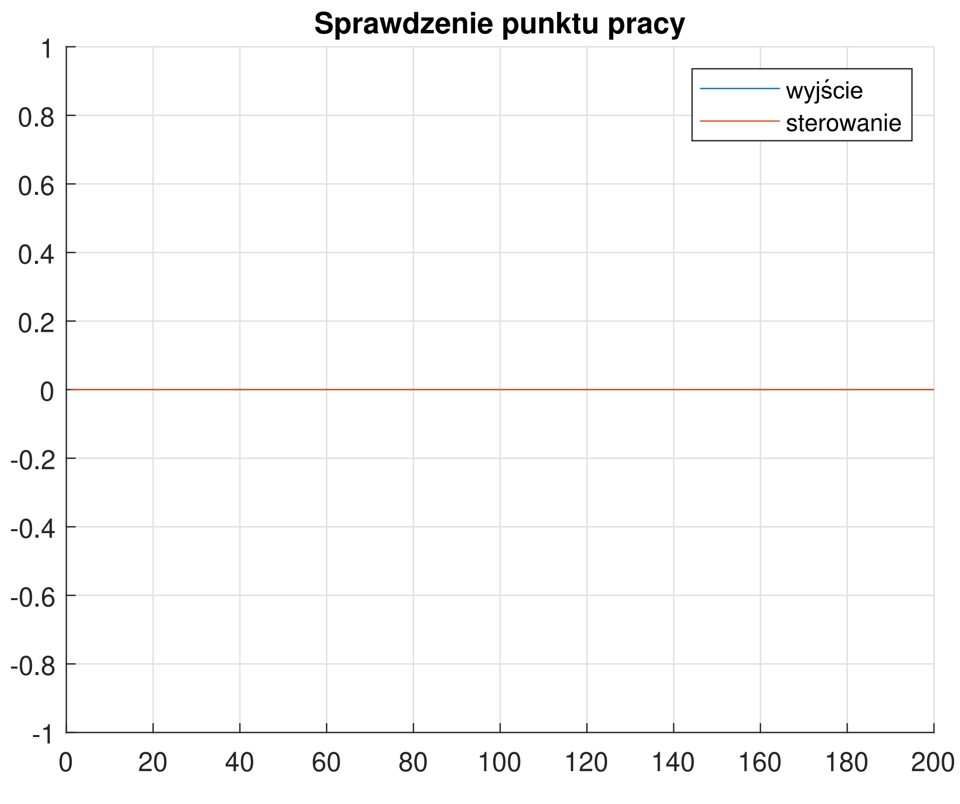
Spis treści

1. Część projektowa

1.1. Sprawdzenie poprawności podanego punktu pracy

W celu sprawdzenia poprawności punktu pracy, sygnał sterujący oraz zakłócenie ustawiamy na wartość 0 i obserwujemy wyjście obiektu.

Wartość wyjścia ustaliła się na wartości Y_{pp}=0. Rysunek 2.1



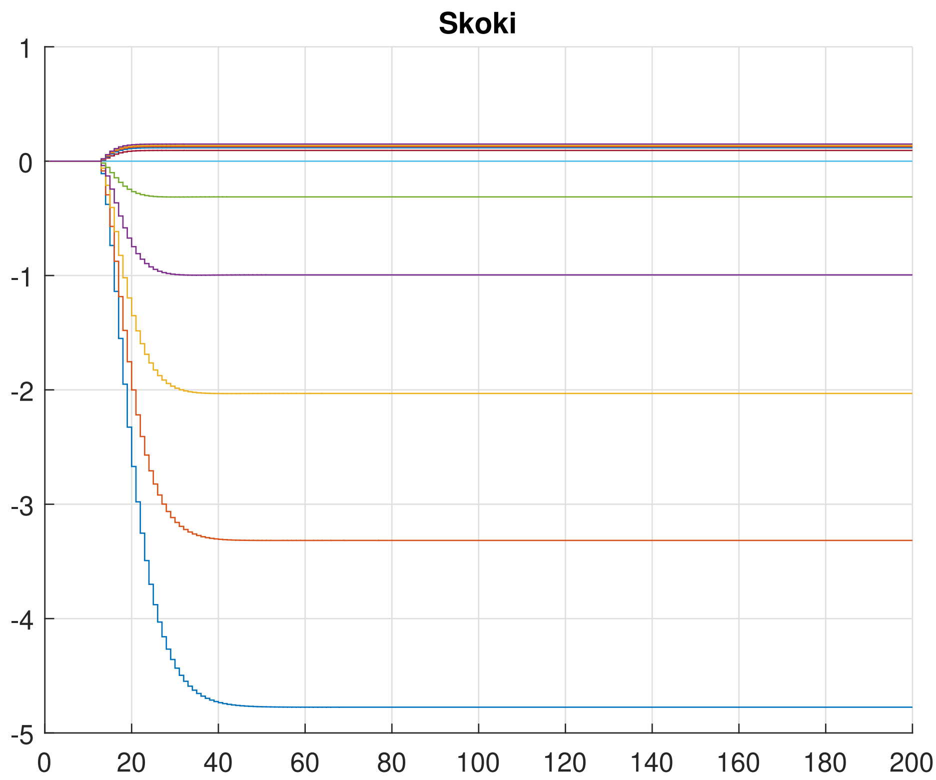
Rys. 1.1. Sprawdzenie punktu pracy

1.2. Symulacyjne wyznaczenie odpowiedzi skokowych procesu

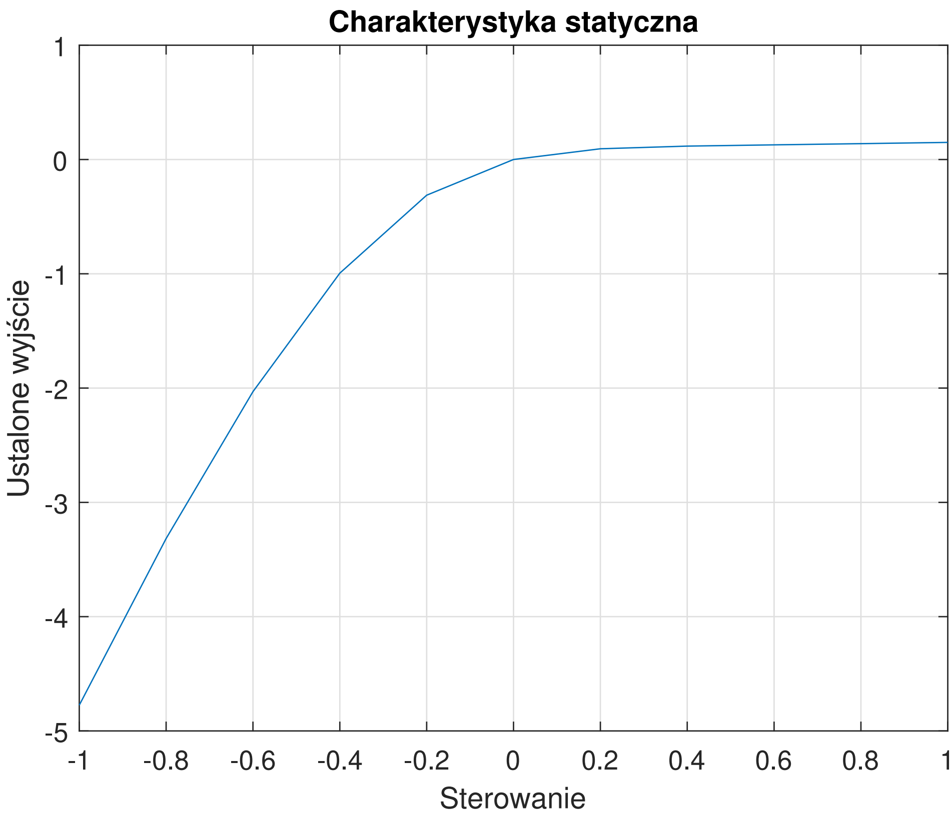
Wykonaliśmy szereg pomiarów odpowiedzi skokowych (rysunek 1.2) dla zmian wartości sygnału sterującego U = -1; -0,8; -0,6; -0,4; -0,2; 0; 0,2; 0,4; 0,6; 0,8; 1.

Następnie symulując odpowiedź układu dla powyższych wartości sygnału sterującego otrzymaliśmy charakterystykę statyczną.

Na podstawie otrzymanego wykresu (rysunek 1.3), oraz poprzednich wykresów można stwierdzić, że właściwości statyczne i dynamiczne procesu nie są liniowe.



Rys. 1.2. Odpowiedzi skokowe sterowania



Rys. 1.3. Charakterystyka statyczna sterowania

1.3. Symulacja cyfrowego algorytmu PID oraz algorytmu DMC

Napisaliśmy program do symulacji cyfrowego algorytmu PID oraz algorytmu DMC z uwzględnieniem ograniczeń wartości sygnału sterującego oraz ograniczeń szybkości jego zmian:

$$0,1 \leqslant U(k) \leqslant 1,5 \tag{1.1}$$

$$-0,2 \leqslant \triangle U(k) \leqslant 0,2 \tag{1.2}$$

Prawo regulacji cyfrowego regulatora PID:

$$u(k) = u(k-1) + r_0 e(k) + r_1 e(k-1) + r_2 e(k-2) \tag{1.3}$$

$$r_0 = K(1 + \frac{T}{2T_i} + \frac{T_d}{T}) \tag{1.4}$$

$$r_1 = K(\frac{T}{2T_i} - \frac{2T_d}{T} - 1) \tag{1.5}$$

$$r_2 = K\frac{T_d}{T} \tag{1.6}$$

Wartości K - wzmocnienie, T_d - stała różniczkowania, T_i - stała całkowania dobieramy tak samo jak w ciągłym regulatorze PID, natomiast T - okres próbkowania zależy od konkretnego zadania, w naszym przypadku wynosi 0,5. W każdej chwili dyskretnej wyznaczane jest wyjście obiektu, oraz uchyb i na tej podstawie zgodnie ze wzorem 1.3 wyliczane jest sterowanie

Fragment głównej pętli algorytmu PID zaimplementowanej w Matlabie:

```
for k=12:Ts
    %symulacja obiektu
    Y(k)=symulacja_obiektu7Y_p1(U(k-10),U(k-11),Y(k-1),Y(k-2));

    %uchyb regulacji
    e(k)=yzad(k)-Y(k);

    %sygnał sterujący regulatora PID
    U(k)=r2*e(k-2)+r1*e(k-1)+r0*e(k)+U(k-1);
```

Dla PID przed główną pętlą dodaliśmy obliczenia parametrów regulatora i zmiany wartości zadanejw czasie trwania symulacji. W głównej pętli po otrzymaniu najnowszego wyjścia y procesu liczymy w następującej kolejności: uchyb, sterowanie, a także uwzględniamy ograniczenia sterowania.

Algorytm DMC jest bardziej skomplikowanym algorytmem w porównaniu do dyskretnego PID, jego celem jest minimalizacja wskaźnika jakości J.

$$J(k) = \sum_{p=1}^{N} (y^{zad}(k+p|k) - \overline{y}(k+p|k))^2 + \lambda \sum_{p=0}^{N_k=1} (\Delta U(k+p|))^2 \tag{1.7}$$

Poniżej przedstawiamy fragment głównej płętli algorytmu DMC:

```
    for k=start:Ts
        %symulacja obiektu
        y(k)=symulacja_obiektu7Y_p1(u(k-10),u(k-11),y(k-1),y(k-2));

        %Obliczenie DU_p
        for d=1:(D-1)
            DU_p(d) = u(k-d) - u(k-d-1);
        end

        %Pomiar wyjścia
        Y = ones(N, 1) * y(k);

        %Obliczenie Y_0
        yo = M_p * DU_p + Y;

        Y_zad = ones(N, 1) * yzad(k);

        %Obliczenie sterowania
        DU = K * (Y_zad - yo);
        u(k) = u(k-1) + DU(1);
```

Obliczamy macierz K (licząc macierz M), macierz M_p i inicjalizujemy macierz DU_p. W pętli głównej po pomiarze aktualnego wyjścia obliczamy macierz DU_p, uzupełniamy macierze Y (pomiar wyjścia), Y_o, Y_{zad} (wartości zadane). W kolejnych linijkach obliczamy sterowanie, uwzględniamy ograniczenia i tak jak dla PIDa publikujemy obliczoną wartość sterowania.

1.4. Dobór nastaw regulatora PID i parametrów algorytmu DMC

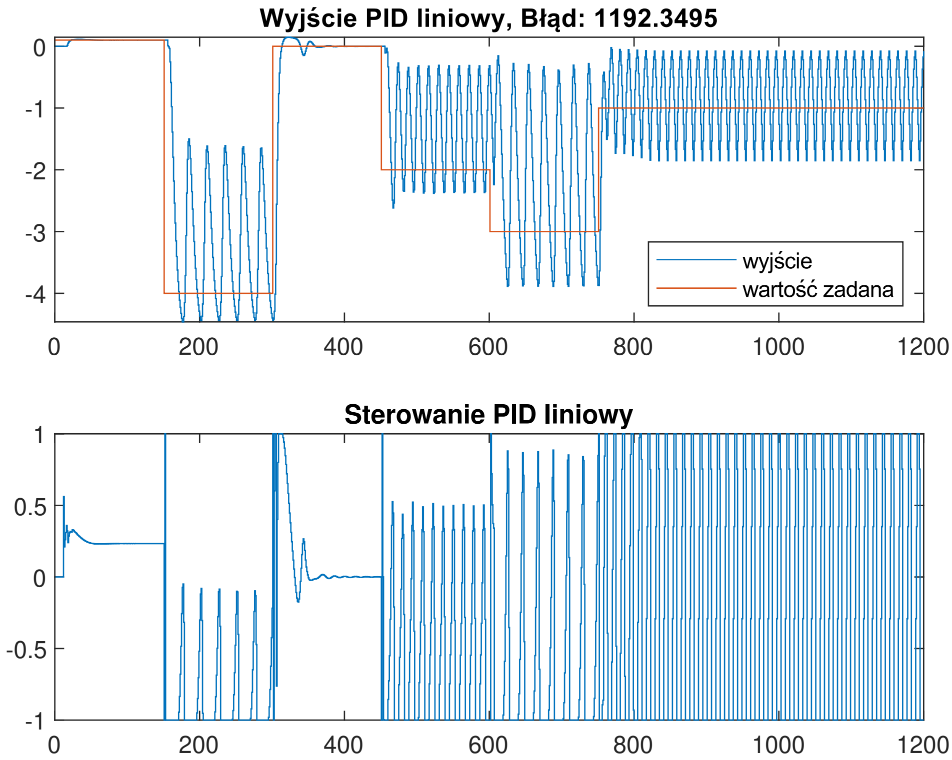
Optymalizację wskaźnika jakości dokonaliśmy dla regulatora za pomocą algorytmu genetycznego *ga*. Dla optymalizacji *ga* przyjęliśmy ograniczenia dolne: N_{min}=0,1, N_{umin}=0,1, T_{dmin}=1. I następujące ograniczenia górne: N_{max}=200, N_{umax}=200, T_{dmax}=200. Algorytm zwrócił nam następujące wartości parametrów: K=1,356; T_i=1,357; T_d= 1,438. Działanie regulatora PID z tymi nastawami przedstawia rysunek 1.4.

Wskaźnik jakości regulacji dla parametrów regulatora PID, otrzymanych wskutek optymalizacji wyniósł E=1192,3. Można zauważyć, że przy znalezionych nastawach regulatora PID, regulator nie działa dla wszystkich skoków prócz w bardzo bliskiej okolicy punktu pracy. Wynika to z faktu, że regulator PID działa tylko na liniowych obiektach, a testowany obiekt jest nieliniowy.

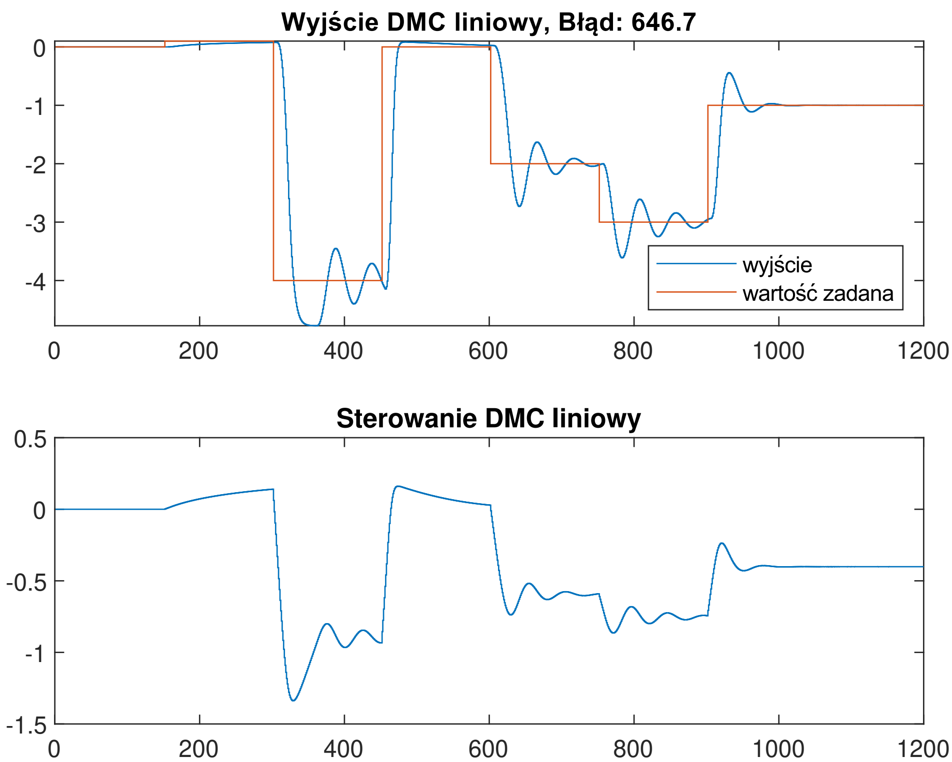
Minimalizacja błędu dla regulatora DMC wykonaliśmy również przy użyciu algorytmu genetycznego *ga*. Dla optymalizacji *ga* przyjęliśmy ograniczenia dolne: N_{min}=0,1, N_{umin}=0,1, λ=1. I następujące ograniczenia górne: N_{max}=200, N_{umax}=200, λ=1000.
Funkcja zwróciła nam następujące wartości parametrów: N=132, N_u=5, λ= 107,36. Działanie regulatora z tymi parametrami jest przedstawione na rysunku 1.5

Zminimalizowany błąd wyniósł E=646,7.

DMC poradził sobie znacznie lepiej niż PID, również widać, że im dalej od punktu pracy tym gorsza regulacja..



Rys. 1.4. Przebieg wyjścia i sterowania regulatora PID przy optymalizacji wskaźnika



Rys. 1.5. Przebieg wyjścia i sterowania regulatora DMC przy optymalizacji wskaźnika

1.5. Implementacja rozmytego algorytmu PID i DMC

Implementacje regulatora PID rozpoczęliśmy od od utworzenia bazy reguł zgodnie ze wzorem:

$$Reg.\gamma : dla, z(k) \in Z^{\gamma} to, u^{\gamma}(k) = r_0^{\gamma} e(k) + r_1^{\gamma} e(k-1) + u(k-1) \tag{1.8}$$

Następnie sterowanie było przydzielane zgodnie z funkcją przynależności. Fragment głównej pętli algorytmu PID zaimplementowanej w Matlabie:

```
funkcje_przynaleznosci =
    FunkcjePrzynaleznosci(ilosc_regulatorow, "gaus", -4.78, 0.15);
for k=12:Ts %główna pętla symulacyjna
        %symulacja obiektu
        Y(k)=symulacja_obiektu7y_p3(U(k-5),U(k-6),Y(k-1),Y(k-2));
        %uchyb regulacji
        e(k)=yzad(k)-Y(k);
        %sygnał sterujący regulatora PID
        u = cell(1, ilosc_regulatorow);
        for i=1:ilosc_regulatorow
           u{i} = r2{i}*e(k-2)+r1{i}*e(k-1)+r0{i}*e(k);
        end

        sum = 0;
        for i=1:ilosc_regulatorow
            sum = sum + funkcje_przynaleznosci{i}(U(k-1));
        end
        U(k) = 0;
        for i=1:ilosc_regulatorow
            U(k) = U(k) + u{i}*funkcje_przynaleznosci{i}(U(k-1))/sum;
        end
        U(k) = U(k) + U(k-1);
```

Implementacje regulatora DMC rozpoczęliśmy od od utworzenia bazy reguł zgodnie ze wzorem:

$$Reg.\gamma : dla, z(k) \in Z^{\gamma} to, \Delta U^{\gamma}(k) = K^{\gamma}(Y^{zad}(k) - Y(k) - M^{P\gamma} \Delta U^{P}(k)) \tag{1.9}$$

Następnie sterowanie było przydzielane zgodnie z funkcją przynależności (rysunek ref). Fragment głównej pętli algorytmu PID zaimplementowanej w Matlabie:

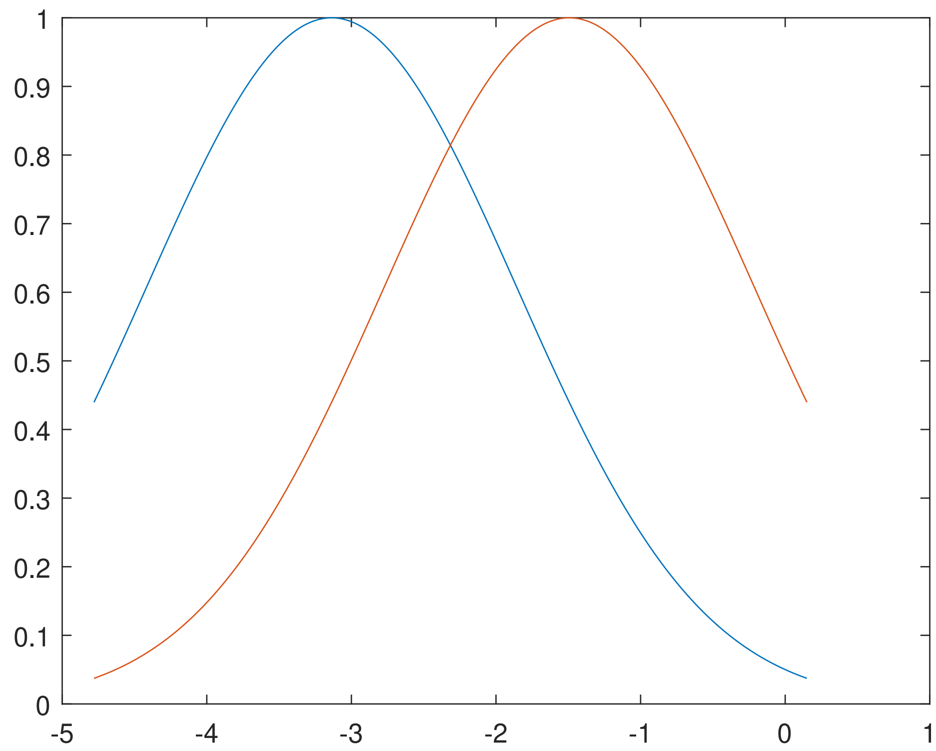
```
functions =
    FunkcjePrzynaleznosci(liczba_regulatorow, typ_funkcji, y_min, y_max);
for k=start:endt
    %symulacja obiektu
    y(k) = symulacja_obiektu7y_p3(u(k-5),u(k-6),y(k-1),y(k-2));

    %Obliczenie DU_p
    for d=1:(D-1)
        DU_p(d) = u(k-d) - u(k-d-1);
    end

    %Obliczenie sterowania
    sum_mi = 0;
    for i=1:liczba_regulatorow
        sum_mi = sum_mi + functions{i}(U(k-1));
    end

    DU = cell(1,liczba_regulatorow);
    for i=1:liczba_regulatorow
         %Pomiar wyjścia
```

```
        Y = ones(N(i), 1) * y(k);

        Y_zad = ones(N(i), 1) * yzad(k);

        DU{i} = (K{i} * (Y_zad -  M_p{i} * DU_p - Y));
    end

    U=0;
    for i =1:liczba_regulatorow
        U = U + (functions{i}(U(k-1))/sum_mi)*DU{i}(1);
    end

    u(k) = u(k-1) + U;
```

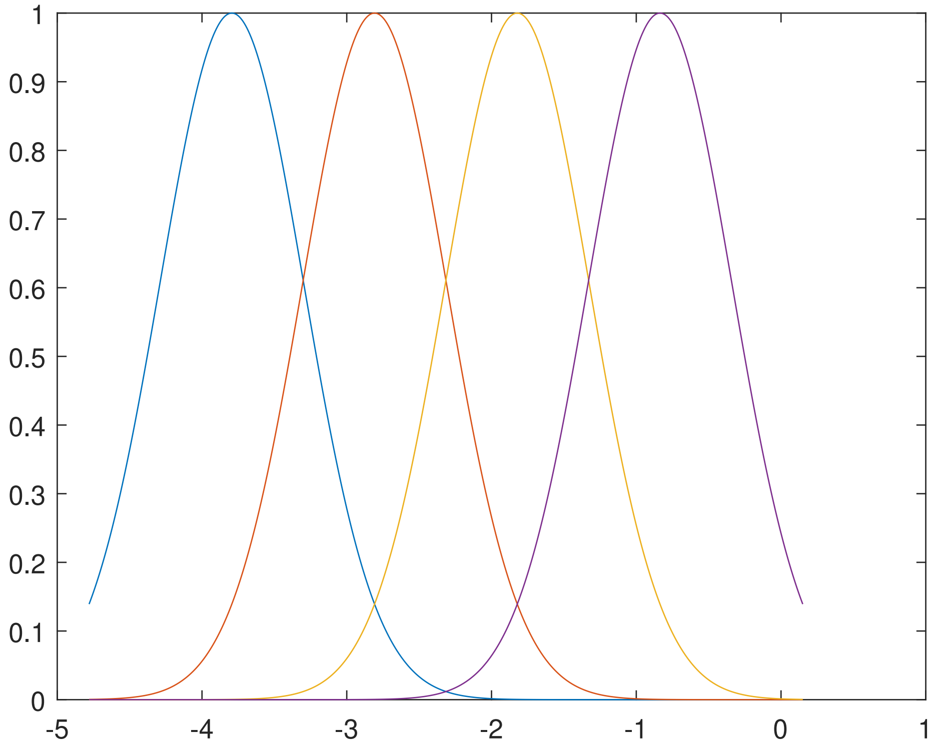
Rozmywanie dokonaliśmy poprzez rozmywanie sterowania, ponieważ bez dokładnego modelu obiektu rozmywanie wyjścia nie jest możliwe. ??????? Za funkcje przynależności wybraliśmy funkcje gaussowską, gdyż zapewnia ona bardziej delikatne przełączanie się pomiędzy lokalnymi regulatorami. Do jej utworzenia wykorzystaliśmy wzór:

```
functions{i} =...
                @(x) gaussmf(x, [sqrt((width/(liczba_regulatorow+1)))
                /(liczba_regulatorow/2), ...
                y_min+i*width/(liczba_regulatorow+1)]);
```

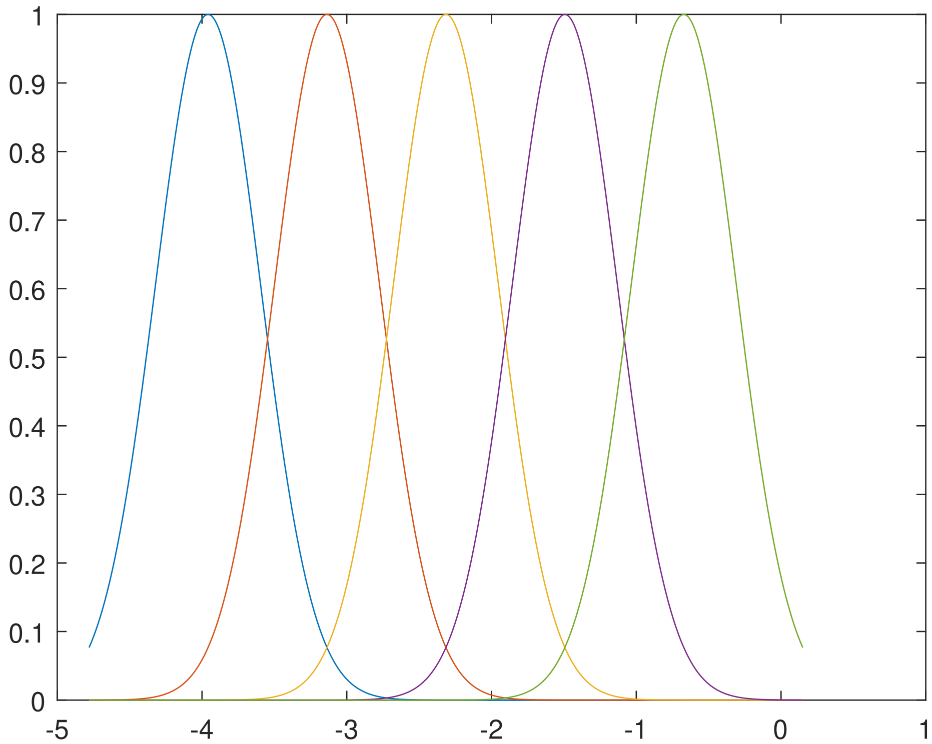
Przykładowe wykresy funkcji przynależności dla kolejno 2 (1.6), 4 (1.7), 5(



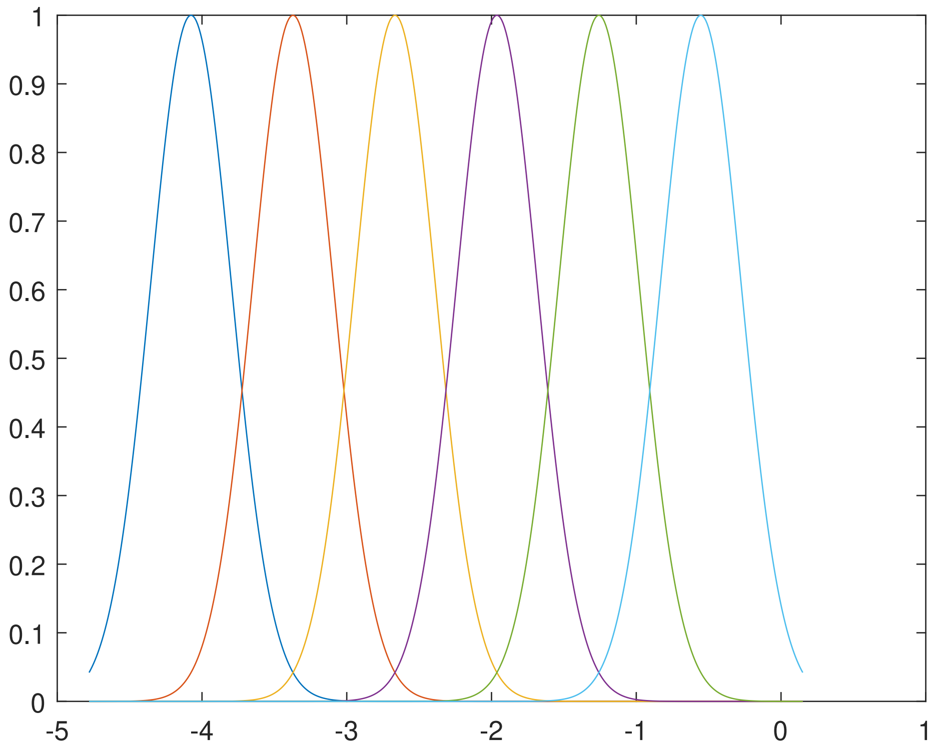
Rys. 1.6. Funkcja przynależności dla 2 regulatorów lokalnych



Rys. 1.7. Funkcja przynależności dla 4 regulatorów lokalnych



Rys. 1.8. Funkcja przynależności dla 5 regulatorów lokalnych

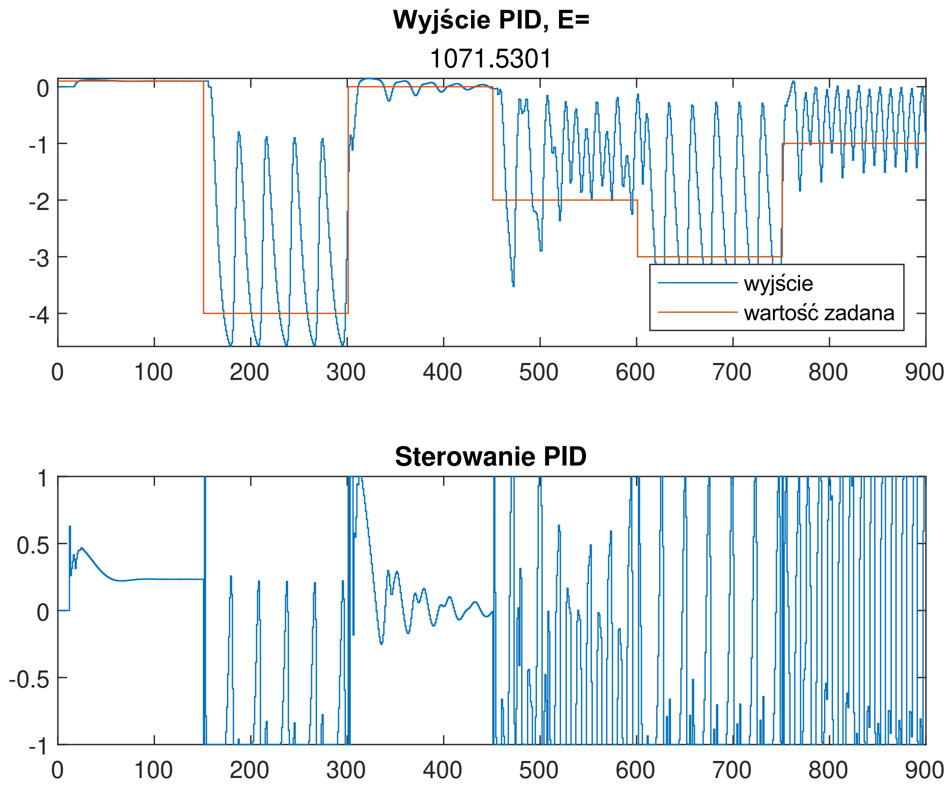


Rys. 1.9. Funkcja przynależności dla 6 regulatorów lokalnych

1.6. Dobór parametrów lokalnych regulatorów

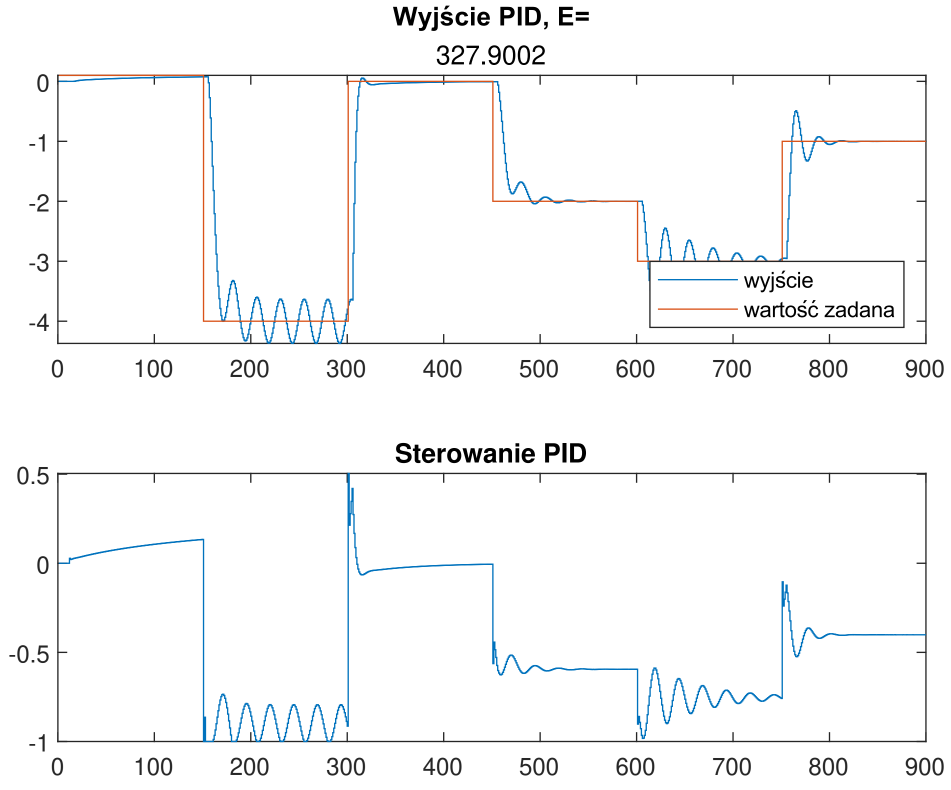
Dla każdego z regulatorów lokalnych rozmytych regulatorów PID i DMC dobraliśmy takie parametry aby osiągnąć jak najlepszą jakość regulacji. Wyniki eksperymentó przedstawione są poniżej:

PID:
Dla 2 regulatorów:
Nastawy :
Regulator 1: K=1.4942, I_i=0.2671, I_d=12.4809.
Regulator 2: K=0.3594, I_i=0.5129, I_d=3.3469.



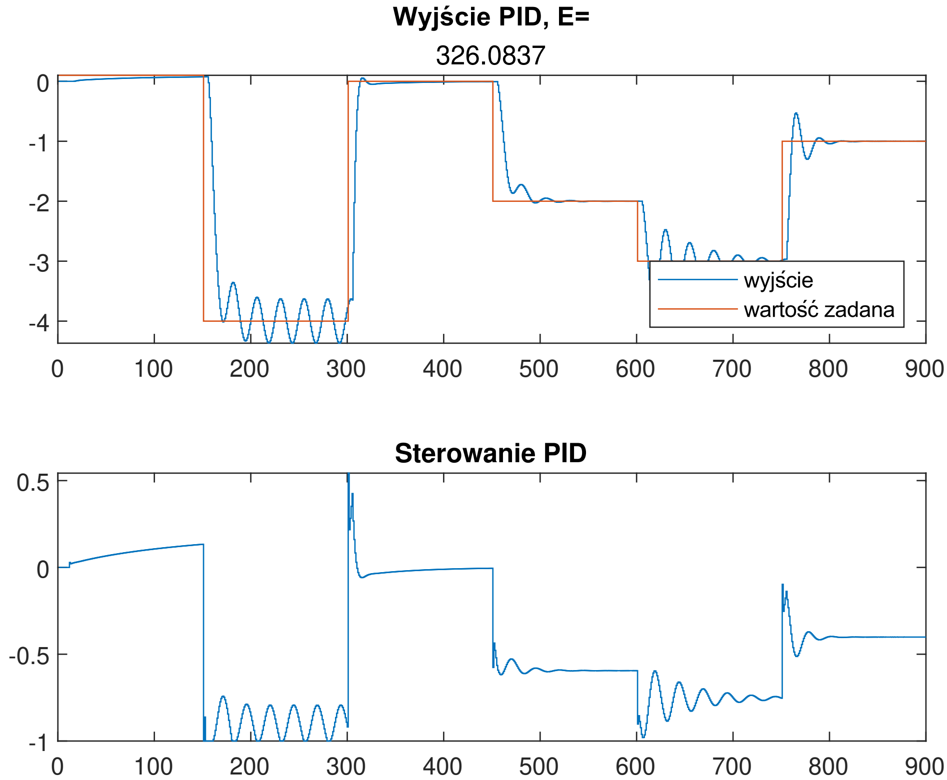
Rys. 1.10. Przebieg wyjścia i sterowania regulatora PID z 2 regulatorami lokalnymi

Dla 3 regulatorów:
Nastawy:
Regulator 1: K=19.6993, I_i=0.9230, I_d=0.0154.
Regulator 2: K=0.5908, I_i=10.3104, I_d=0.0121.
Regulator 3: K=0.1729, I_i=4.6522, I_d=0.2559.



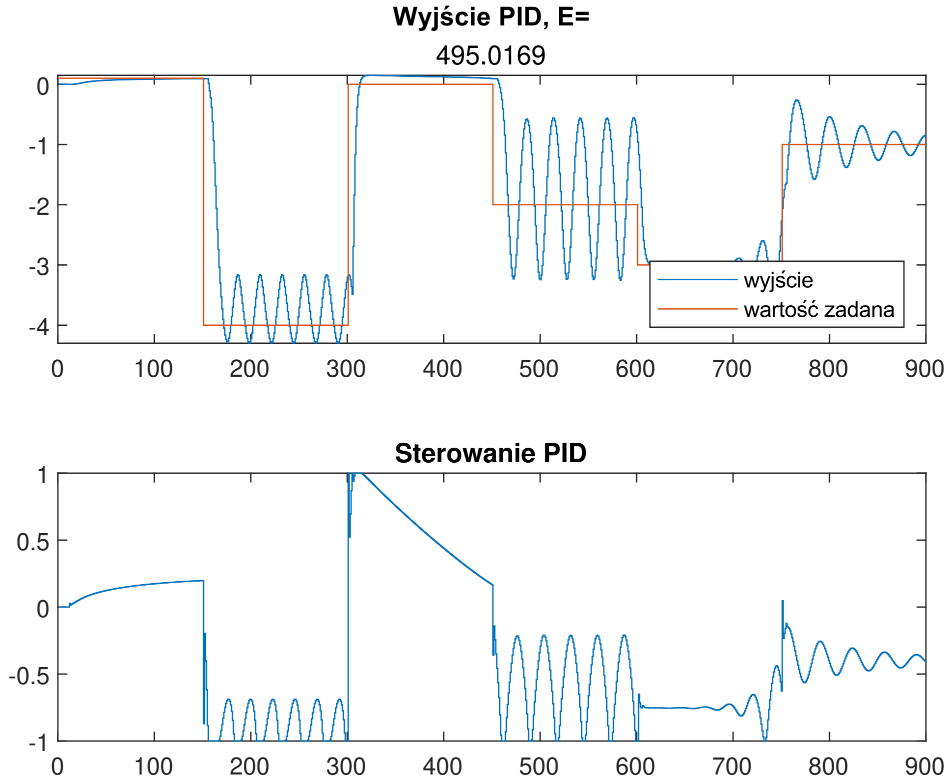
Rys. 1.11. Przebieg wyjścia i sterowania regulatora PID z 3 regulatorami lokalnymi

Dla 4 regulatorów:
Nastawy:
Regulator 1: K=11.4655, I_i=7.3778, I_d=10.2642.
Regulator 2: K=19.2346, I_i=0.9797, I_d=0.0201.
Regulator 3: K=0.6699, I_i=3.2470, I_d=0.0116.
Regulator 4: K=0.1772, I_i=4.7706, I_d=0.2735.



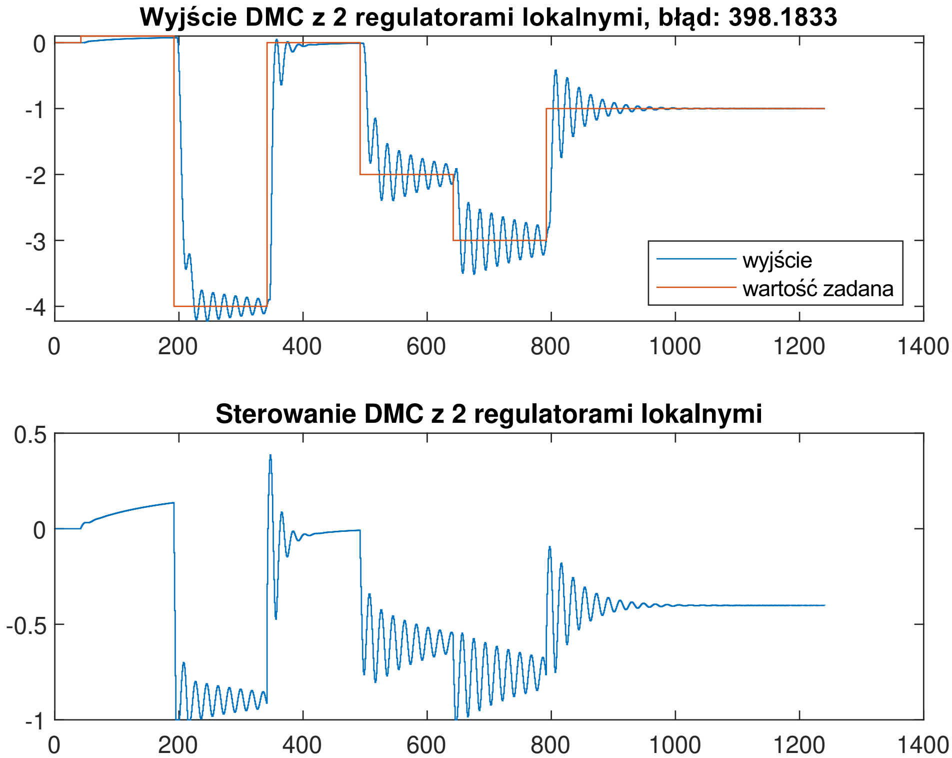
Rys. 1.12. Przebieg wyjścia i sterowania regulatora PID z 4 regulatorami lokalnymi

Dla 5 regulatorów:
Nastawy:
Regulator 1: K=18.6144, I_i=18.6144, I_d=18.6144.
Regulator 2: K=18.5952, I_i=18.5570, I_d=18.9023.
Regulator 3: K=4.7368, I_i=19.8190, I_d=19.1078.
Regulator 4: K=1.7621, I_i=2.3882, I_d=0.1272.
Regulator 5: K=0.0752, I_i=0.7733, I_d=1.0662.



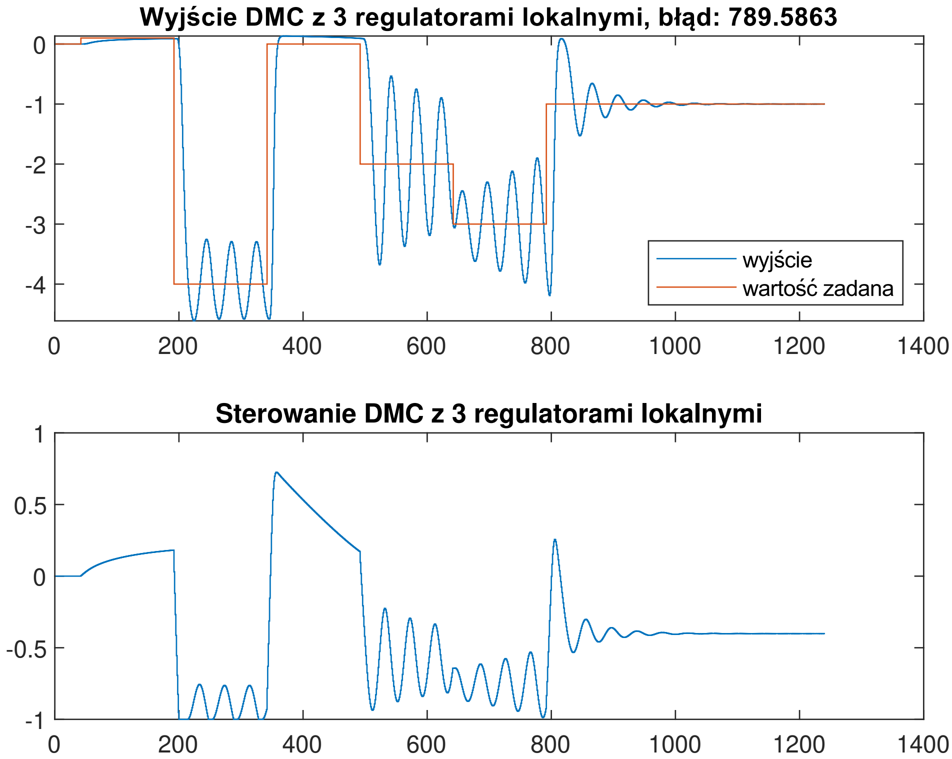
Rys. 1.13. Przebieg wyjścia i sterowania regulatora PID z 5 regulatorami lokalnymi

DMC:
Zgodnie z poleceniem parametr λ ustawiliśmy na 1 i korzystając z algorytmu *ga* stroiliśmy poszczególne wariacje regulatorów lokalnych.
Dla 2 regulatorów:
Nastawy :
Regulator 1: N=81, N_u=2.
Regulator 2: N=1, N_u=16.



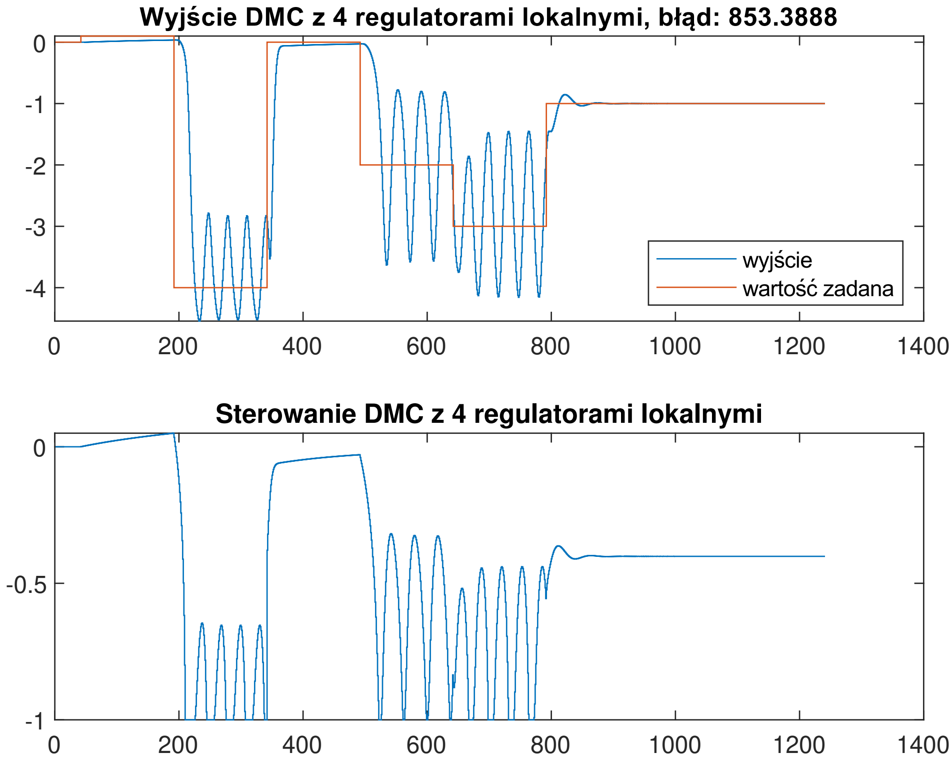
Rys. 1.14. Przebieg wyjścia i sterowania regulatora DMC z 2 regulatorami lokalnymi

Dla 3 regulatorów:
Nastawy:
Regulator 1: N=38, N_u=107.
Regulator 2: N=2, N_u=152.
Regulator 3: N=6, N_u=184.



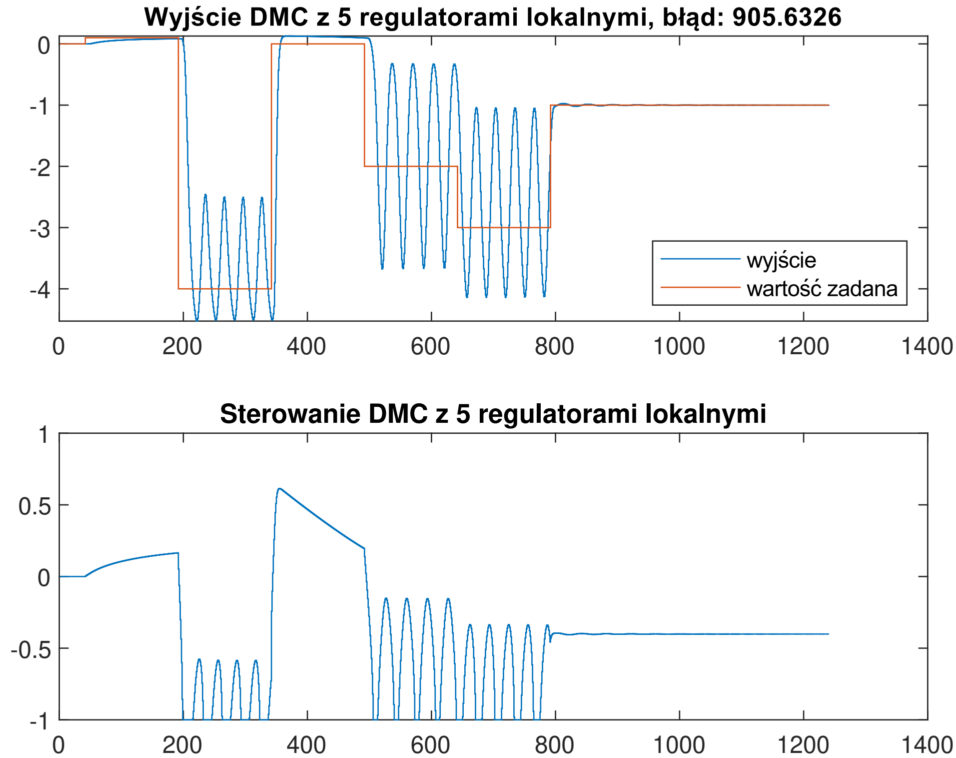
Rys. 1.15. Przebieg wyjścia i sterowania regulatora DMC z 3 regulatorami lokalnymi

Dla 4 regulatorów:
Nastawy:
Regulator 1: N=200, N_u=2.
Regulator 2: N=36, N_u=1.
Regulator 3: N=200, N_u=2.
Regulator 4: N=1, N_u=181.



Rys. 1.16. Przebieg wyjścia i sterowania regulatora DMC z 4 regulatorami lokalnymi

Dla 5 regulatorów:
Nastawy:
Regulator 1: N=9, N_u=124.
Regulator 2: N=41, N_u=56.
Regulator 3: N=173, N_u=4.
Regulator 4: N=198, N_u=4.
Regulator 5: N=6, N_u=106.



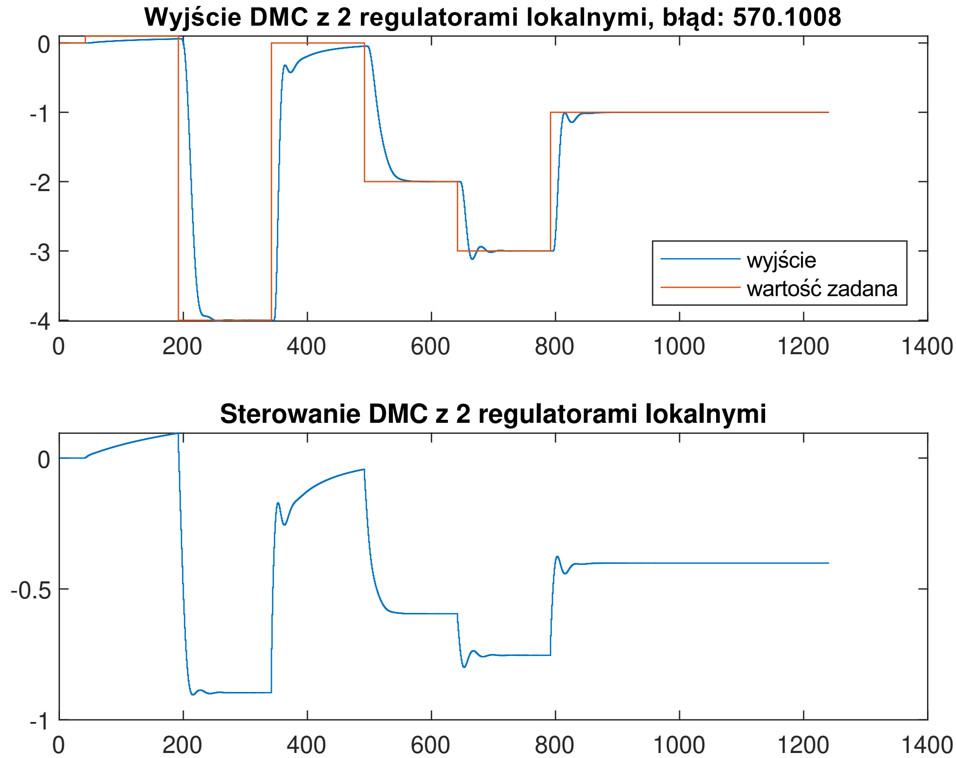
Rys. 1.17. Przebieg wyjścia i sterowania regulatora DMC z 5 regulatorami lokalnymi

1.7. Dobór parametru lambda dla lokalnych regulatorów DMC

Następnie próbowaliśmy ulepszyć otrzymane regulatory rozmyte DMC, poprzez modyfikacje parametru λ. Parametr λ jest ważnym parametrem regulatora rozmytego DMC, pozwala on na uzyskanie lepszej jakości regulacji.

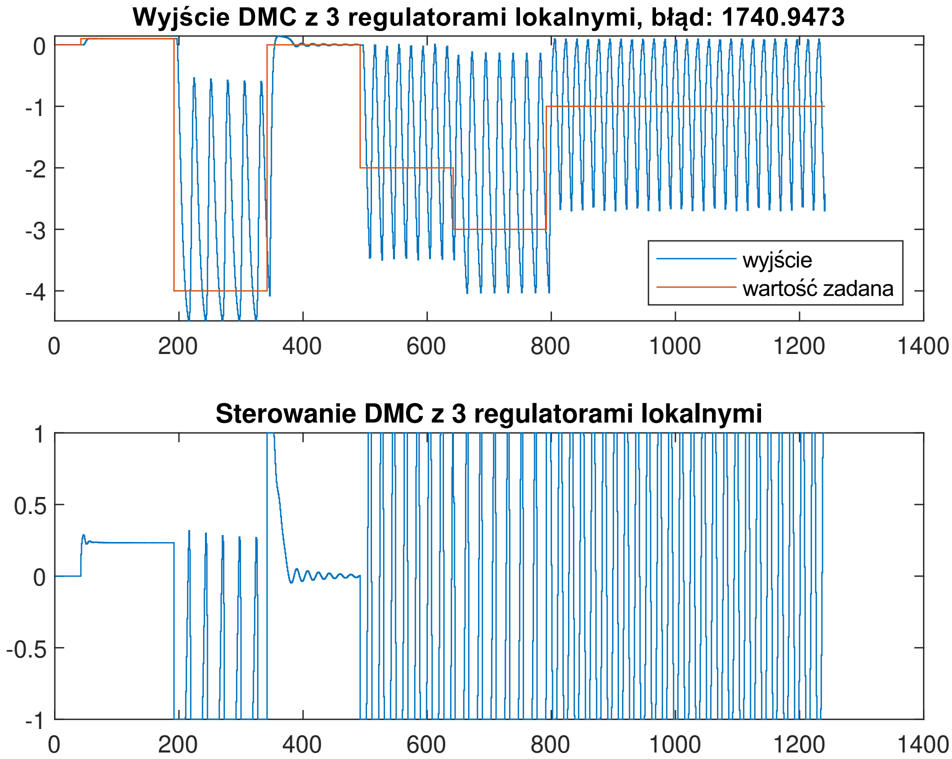
otrzymaliśmy następujące wyniki:

Dla 2 regulatorów:
Regulator 1: λ=10.
Regulator 2: λ=1.



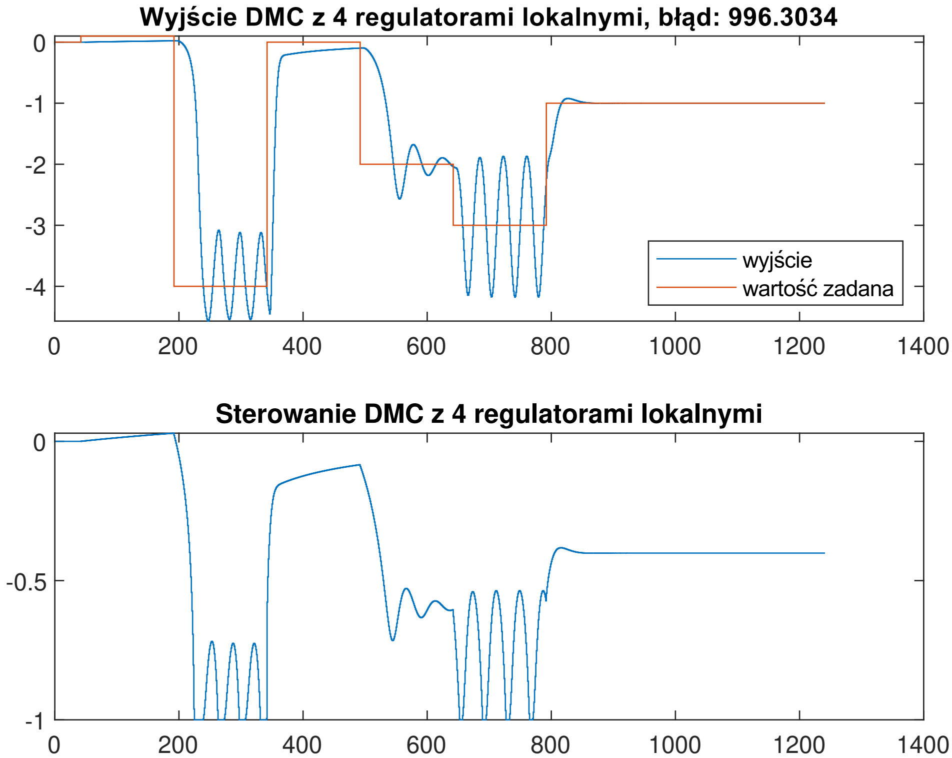
Rys. 1.18. Przebieg wyjścia i sterowania regulatora DMC przy optymalizacji wskaźnika

Dla 3 regulatorów:
Regulator 1: λ=10.
Regulator 2: λ=5.
Regulator 3: λ=3.



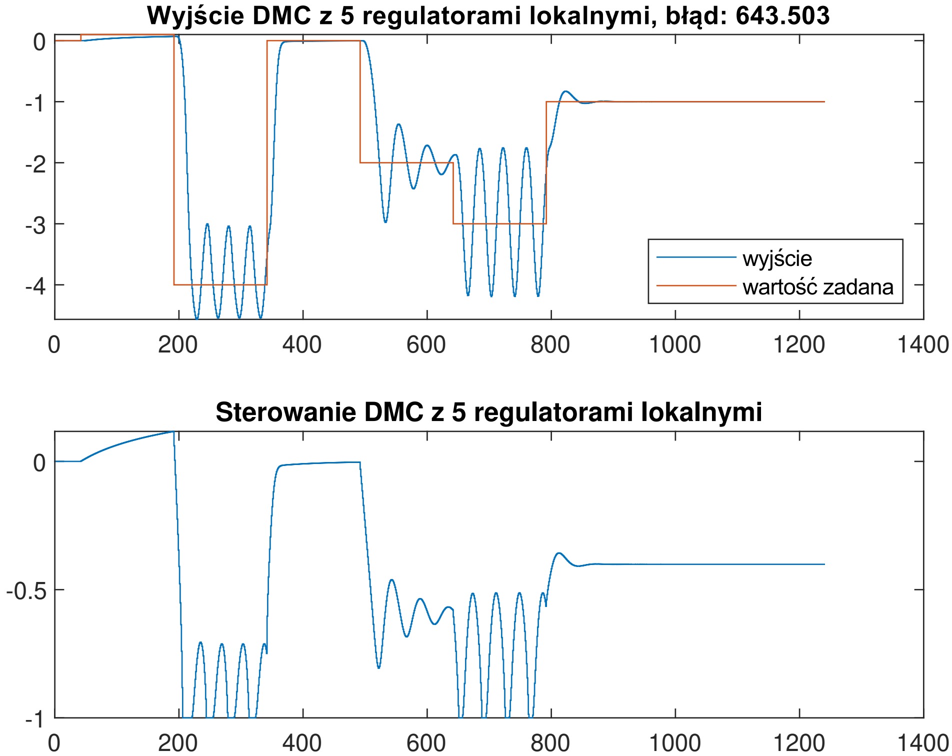
Rys. 1.19. Przebieg wyjścia i sterowania regulatora DMC przy optymalizacji wskaźnika

Dla 4 regulatorów:
Regulator 1: λ=10.
Regulator 2: λ=10.
Regulator 3: λ=50.
Regulator 4: λ=20.



Rys. 1.20. Przebieg wyjścia i sterowania regulatora PID przy optymalizacji wskaźnika

Dla 5 regulatorów:
Regulator 1: λ=18.
Regulator 2: λ=1.
Regulator 3: λ=1.
Regulator 4: λ=16.
Regulator 5: λ=2.

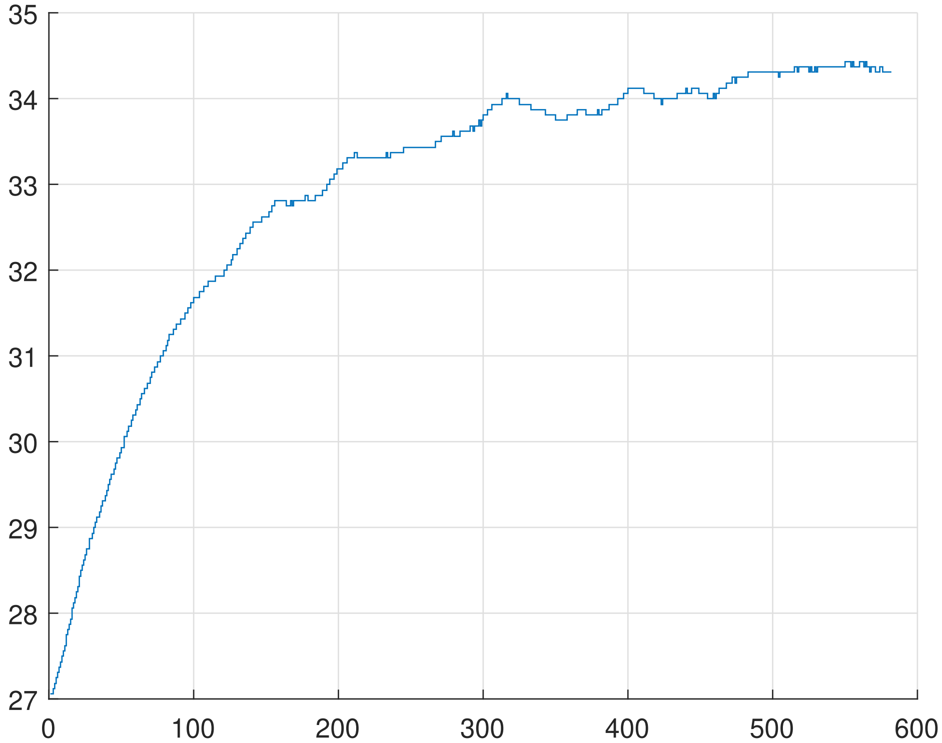


Rys. 1.21. Przebieg wyjścia i sterowania regulatora DMC przy optymalizacji wskaźnika

2. Część laboratoryjna

2.1. Sprawdzenie możliwość sterowania i pomiaru w komunikacji ze stanowiskiem i wyznaczenie punktu pracy

Uruchomiliśmy stanowisko, oraz sprawdziliśmy możliwość sterowania i komunikacji z nim. Określiliśmy wartość pomiaru temperatury w punkcie pracy $Z = 0$, $G1 = 27$, sygnał $W1$ traktujemy jako cechę otoczenia, jego wartość jest stała i równa $W1 = 50$. Wartość temperatury ustaliła się na wartości Y_{pp}=34,3 (jest to nasz punkt pracy) co przedstawia rysunek 2.1.



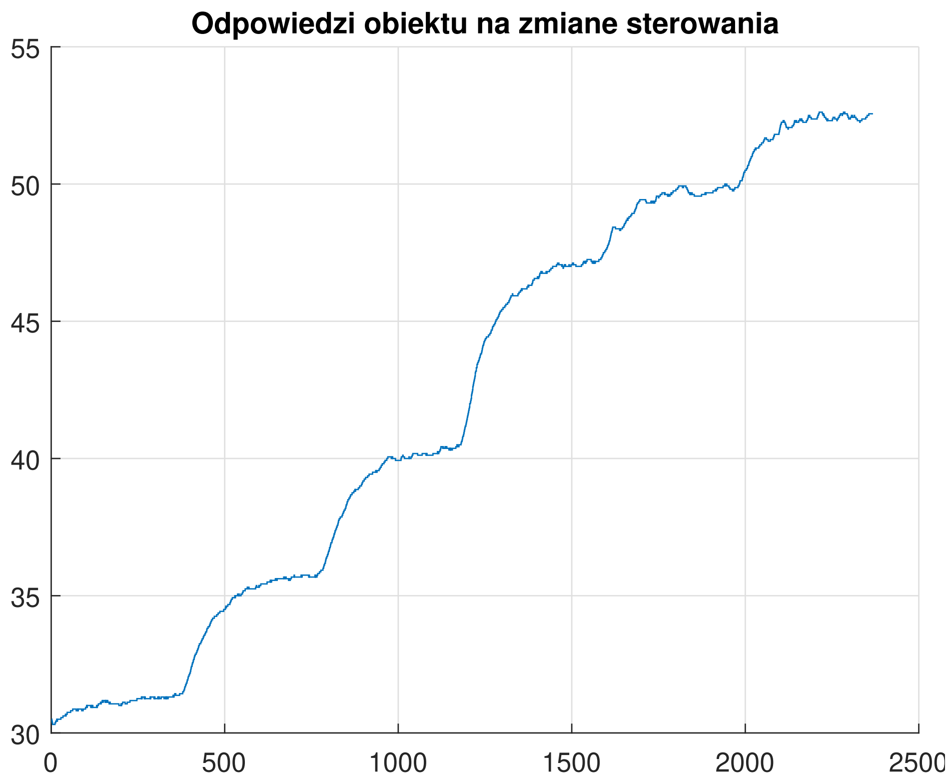
Rys. 2.1. Wyznaczenie punktu pracy.

2.2. Pozyskanie wartości ustabilizowanego sygnału wyjściowego i wyznaczenie charakterystyki statycznej obiektu.

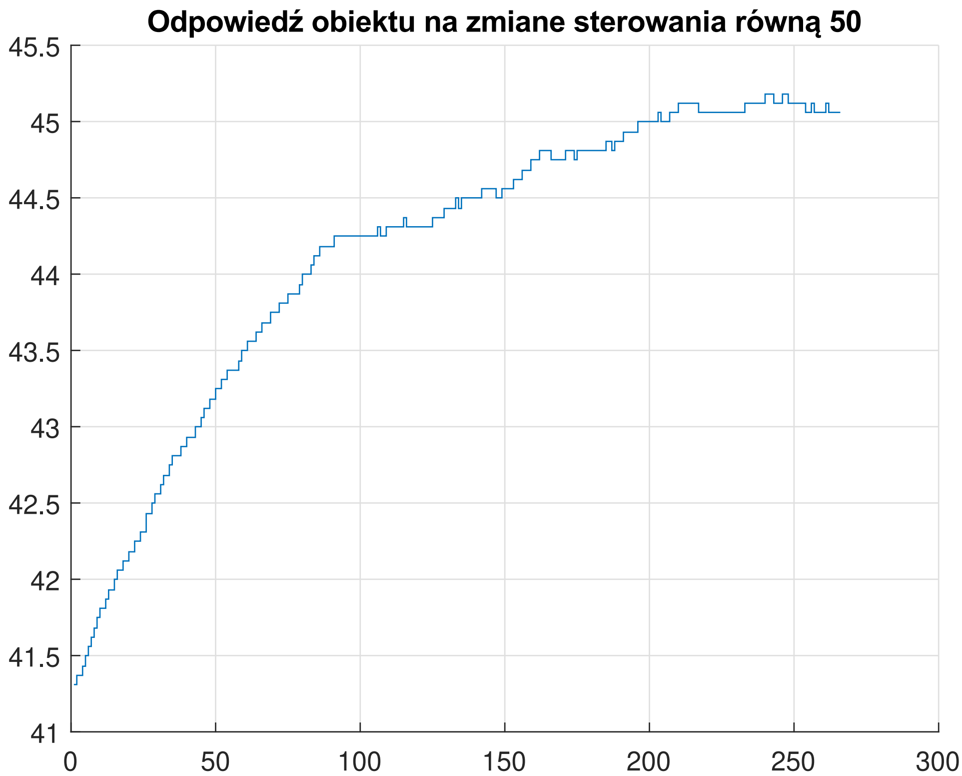
Dla kolejnych wartości sterowania (20, 30, 40, 50, 60, 70, 80) pozyskaliśmy wartości ustabilizowanego sygnału wyjściowego. Przebieg eksperymentu obrazuje wykres 2.2. Niestety w wyniku błędu w kodzie, nie został wykonany pomiar dla sterownia równego 50, zatem aby nie powtarzać całego czasochłonnego procesu, wykonaliśmy dodatkowy pomiar dla tej wartości sterowania - rysunek 2.3.

Na podstawie wykonanych pomiarów wyznaczyliśmy charakterystykę statyczną obiektu - rysunek 2.4

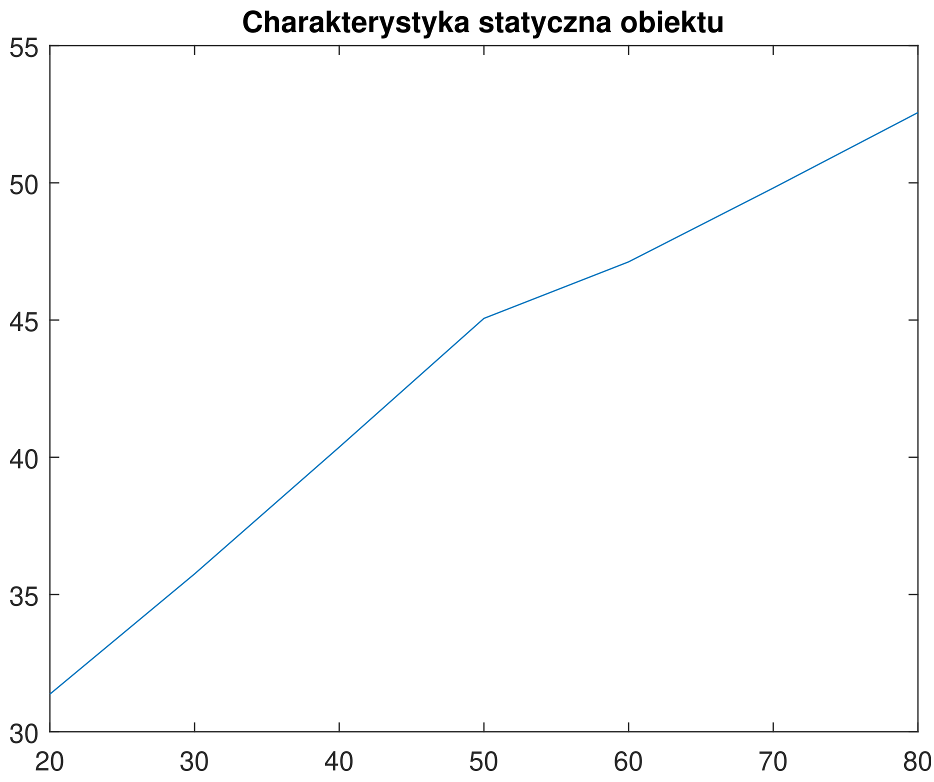
Właściwości statyczne obiektu nie można określić jako liniowe, potwierdza to wykres charakterystyki statycznej 2.4, który nie jest liniowy.



Rys. 2.2. Odpowiedzi obiektu na kolejne zmiany sterowania 20, 30 ..., 80



Rys. 2.3. Odpowiedź obiektu na sterowanie 50



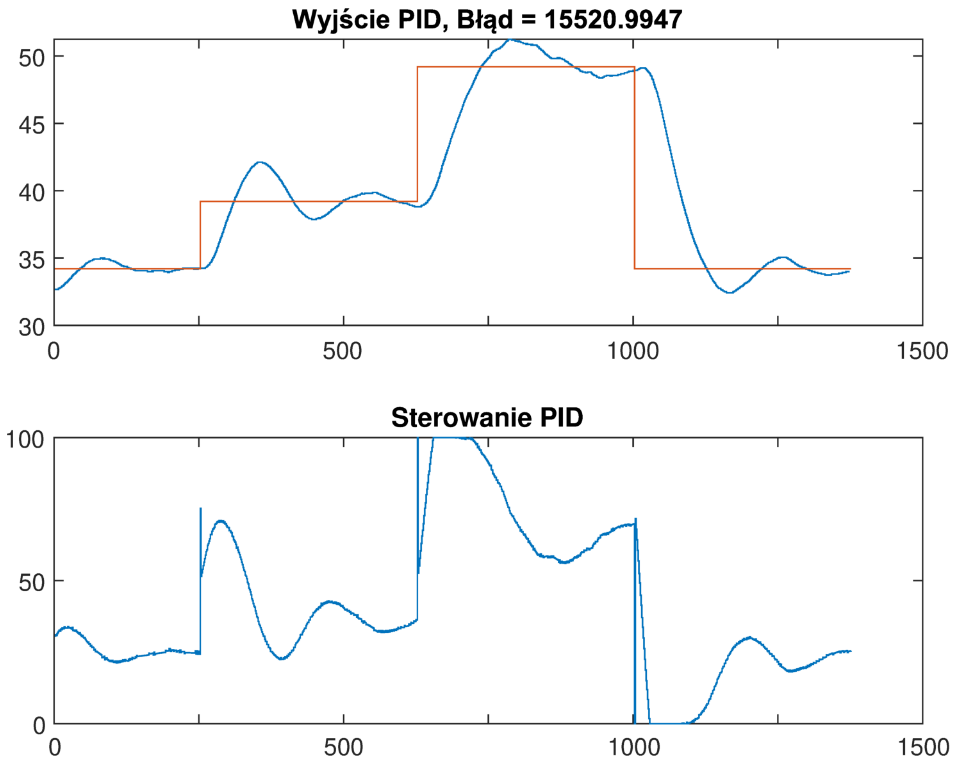
Rys. 2.4. Otrzymana charakterystyka statyczna obiektu

2.3. Przetestowanie regulatorów PID i DM z laboratorium 1.

Dla trajektorii zmian sygnału wartości zadanej: 35.1, 40.1, 50.1, 35.1 (pkt pracy nieznacznie się zmienił) przetestowaliśmy regulatory PID i DMC z laboratorium 1.

Działanie regulatora PID przedstawia wykres 2.5. W zależności od wartości zadanej obserwujemy różne zachowanie wyjścia obiektu. Dla wartości $y_{zad} = 40.1$ i $y_{zad} = 50.1$ widzimy duże przerególowanie, dość długie oscylacje i długi czas ustalania. Dla wartości $y_{zad} = 35.1$ regulator działa nieco lepiej, jednak jakość regulacji jest także niesatysfakcjonująca.

Niestety z nieznanych nam przyczyn dane z eksperymentu z regulatorem DMC nie zostały poprawnie zapisane, o czym zorientowaliśmy się dopiero po zakończeniu laboratorium. Dlatego możemy przedstawić w tym miejscu jedynie opis jego działania. Jakość regulacji regulatora DMC na obiekcie nielinowym była lepsza od PID-a, co wynika z charakterystyki obu regulatorów - na laboratorium 1 DMC oferował znacznie lepszą regulację niż PID. Jakość regulacji DMC nie była jednak w pełni satysfakcjonująca. W zależności od przedziału wartości zadanej obserwowaliśmy większe lub mniejsze przeregulowanie i oscylacje, a także czas osiągania wartości zadanej mocno się różnił.



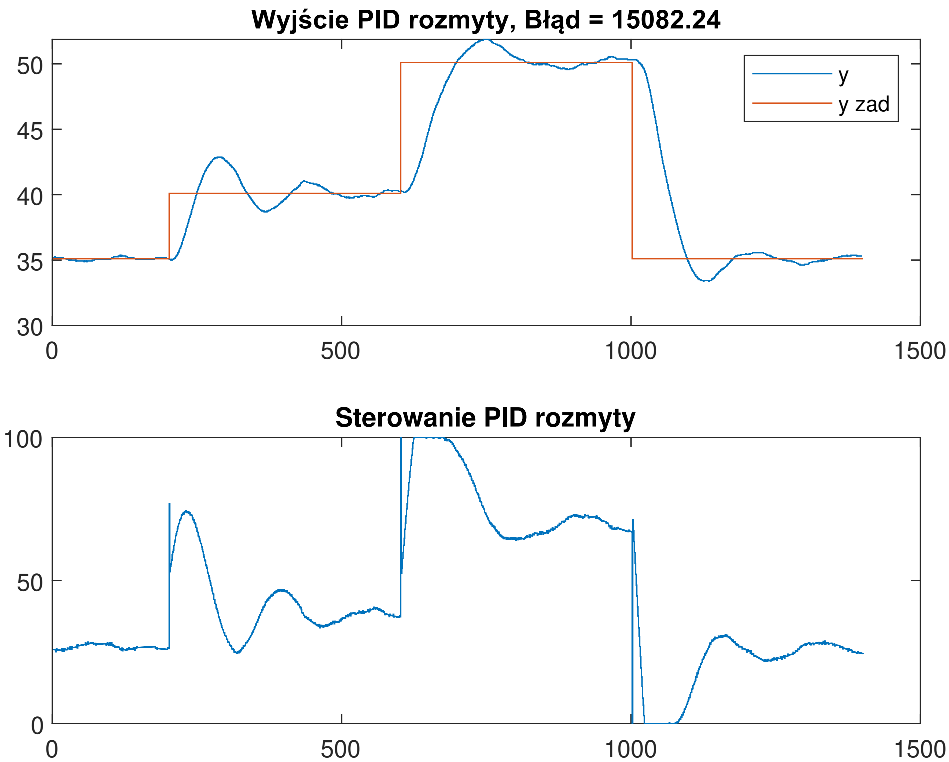
Rys. 2.5. Działanie regulatora PID na obiekcie nieliniowym

2.4. Implementacja i strojenie rozmytego regulatora PID

Zaimplementowaliśmy rozmyty PID z 3 regulatorami lokalnymi. Testy jego działania przeprowadzaliśmy dla takiej samej trajektorii zmian sygnału wartości zadanej (35.1, 40.1, 50.1, 35.1).

Działanie rozmytego regulatora PID przedstawia wykres 2.6. Każdy z lokalnych regulatorów miał takie same nastawy: $K = 5, T_i = 20, T_d = 1$. Rozmycie regulatora PID pozwoliło uzyskać lepszą jakość regulacji oraz mniejszy błąd. Wyjście obiektu sterowanego rozmytym PID-em ma mniejsze oscylacje, nieco mniejsze przeregulowanie, oraz czas ustalania jest mniejszy.

Proces strojenia regulatora rozmytego PID polega na doborze nastaw poszczególnych regulatorów. Można to wykonać przy pomocy metody inżynierskiej - zmieniamy nastawy i obserwujemy czy nastąpiła poprawa jakości sterowania.

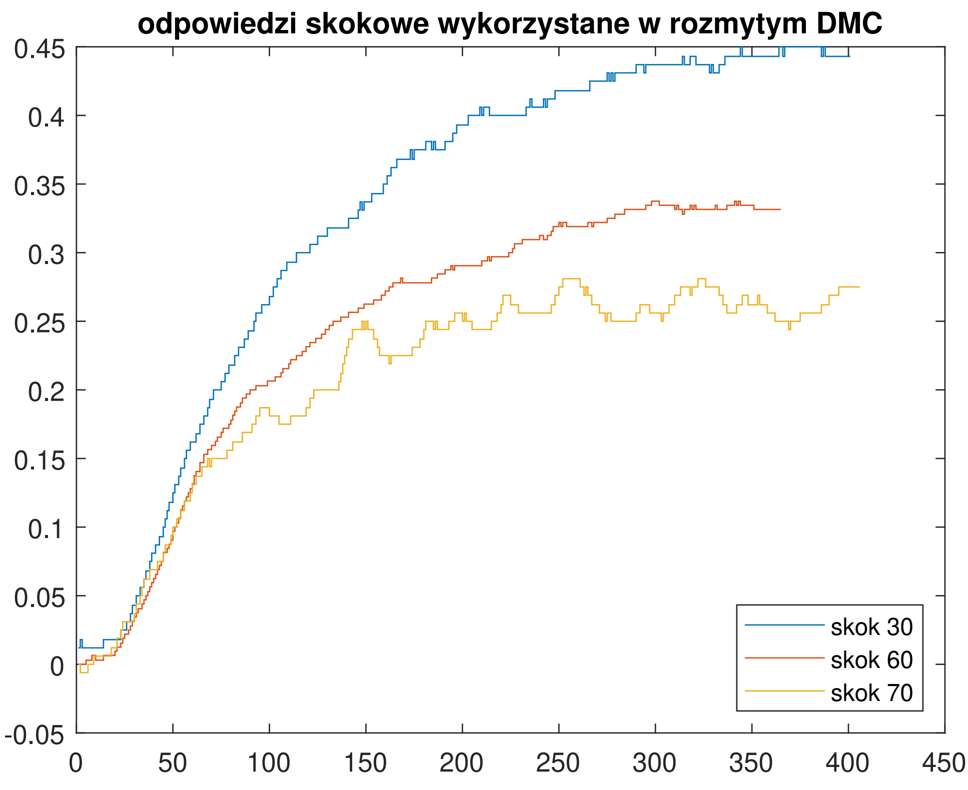


Rys. 2.6. Działanie regulatora rozmytego PID

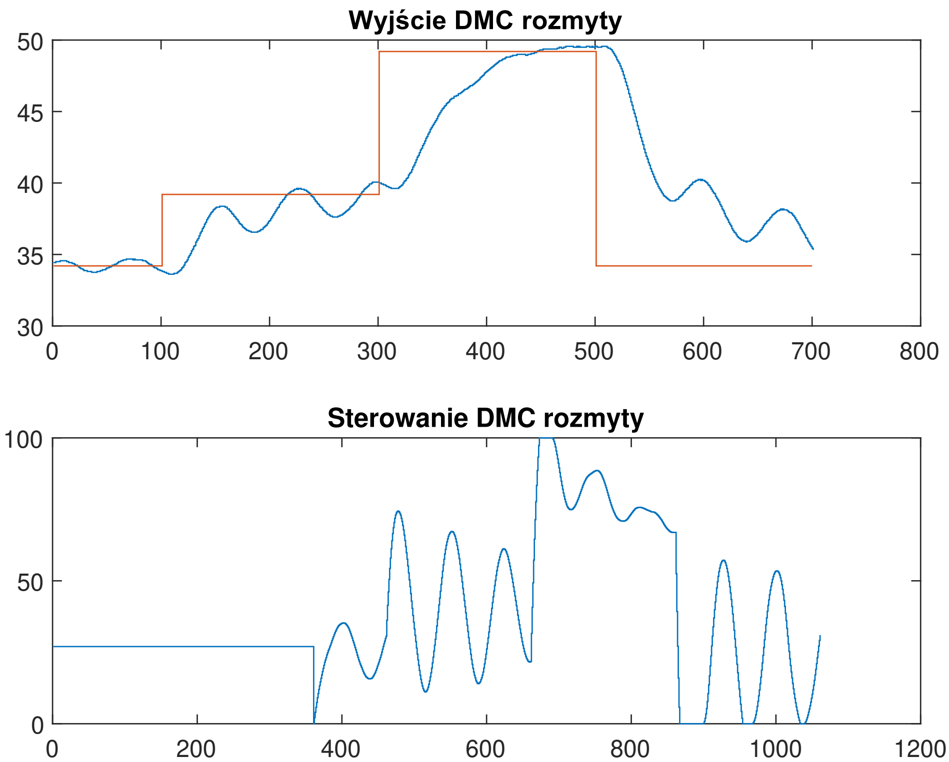
2.5. Implementacja rozmytego algorytmu DMC

W celu implementacji algorytmu DMC z 3 regulatorami lokalnymi potrzebowaliśmy odpowiedzi skokowych z 3 przedziałów. Otrzymaliśmy je z danych które przedstawia rysunek 2.2. Uzyskane odpowiedzi skokowe przedstawia rysunek 2.7

Rezultat działania regulatora rozmytego DMC o nastawach $N_u = N = D = 360$ i $\lambda = 1$ przedstawia rysunek 2.8



Rys. 2.7. odpowiedzi skokowe wykorzystane w regulatorze rozmytym DMC

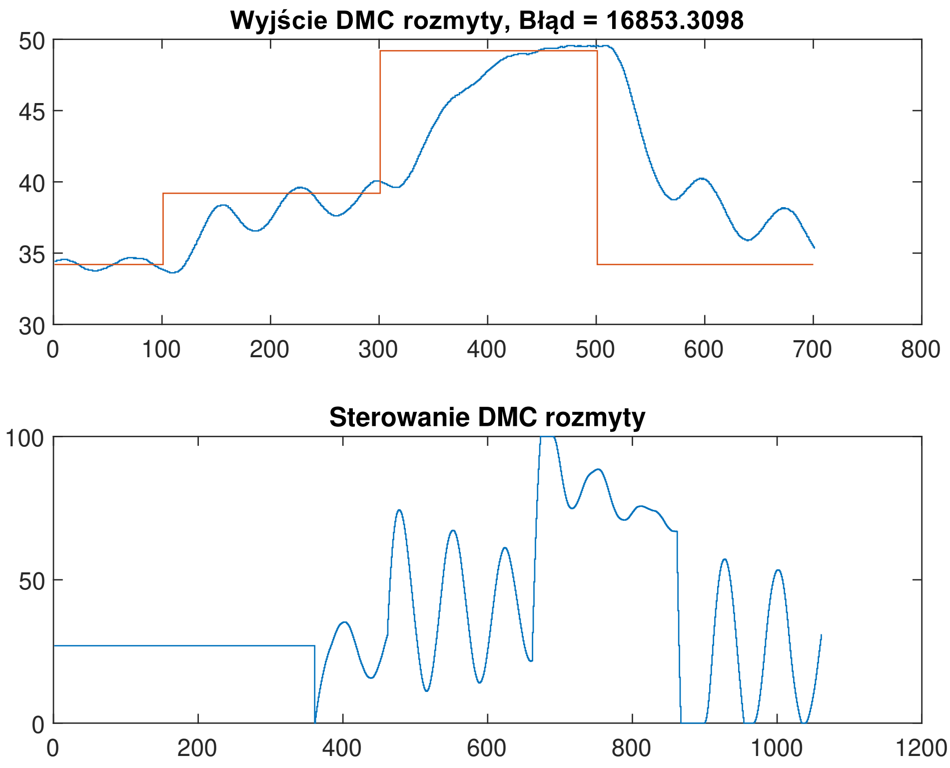


Rys. 2.8. Rozmyty DMC z 3 regulatorami lokalnymi o nastawach $N_u = N = D = 360 i \lambda = 1$

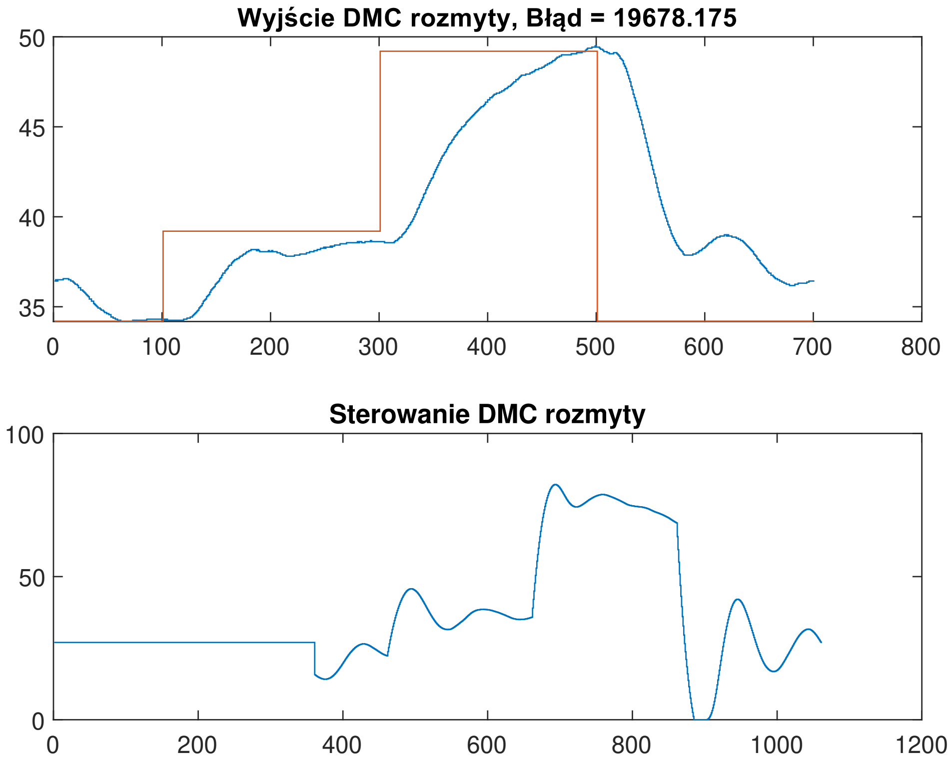
2.6. Dobór parametru λ rozmytego regulatora DMC

Spróbowaliśmy dobrać parametr $/lambda$ metodą eksperymentalną tak aby poprawić jakość regulacji. Ekspreymenty przeprowadziliśmy dla wartości $\lambda = 1$ dla wszystkich 3 regulaotrów lokalnych (rys 2.9), $\lambda = 10$ dla wszystkich 3 regulatorów lokalnych (rys 2.10) oraz $\lambda = [5, 5, 1]$ dla kolejnych 3 regulatorów lokalnych (rys 2.11).

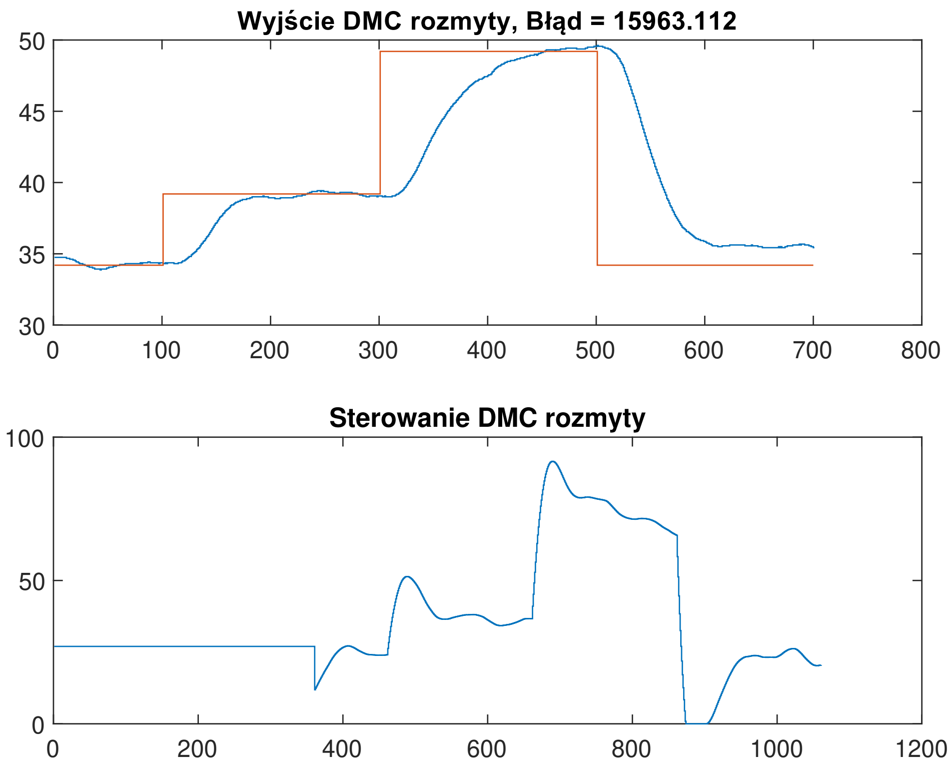
Z przeprowadzonych eksperymentów możemy stwierdzić że parametr λ znacząco wpływa na wyjście obiektu oraz na sygnał sterujący. Parametr ten najlepiej jest dobierać oddzielnie dla każdego z regulatorów lokalnych o czym świadczą przebiegi 2.11, dla $\lambda = [5, 5, 1]$ regulator rozmyty działa najlepiej - sygnał wyjściowy obiektu ma najlepszy przebieg (m.in brak oscylacji) a także ma najmniejszą wartość błędu spośród wszystkich testowanych.



Rys. 2.9. Działanie regulatora rozmytego DMC z $\lambda = 1$ dla każdego z regulatorów lokalnych.



Rys. 2.10. Działanie regulatora rozmytego DMC z $\lambda = 10$ dla każdego z regulatorów lokalnych.



Rys. 2.11. Działanie regulatora rozmytego DMC z $\lambda = [5, 5, 1]$ dla kolejnych z regulatorów lokalnych.