Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania (projekt grupowy)

Sprawozdanie z projektu i ćwiczenia laboratoryjnego nr 1, zadanie nr 12

Paulina Dąbrowska, Miłosz Kowalewski,
Adam Rybojad, Mikołaj Wewiór

Warszawa, 2023

Spis treści

1. CZĘŚĆ PROJEKTOWA

1.1. Sprawdzenie poprawności punktów pracy

1.2. Wyznaczenie symulacyjne odpowiedzi skokowej procesu

1.2.1. Odpowiedzi skokowe

1.2.2. Charakterystyka statyczna y(u)

1.2.3. Wzmocnienie statyczne

1.3. Przekształcenie otrzymanej odpowiedzi na odpowiedź skokową dla algorytmu DMC

1.4. Program do symulacji cyfrowego algorytmu PID oraz algorytmu DMC

1.4.1. PID

1.4.2. DMC

1.5. Dobór nastaw algorytmów regulacji metodą eksperymentalną

1.5.1. PID

1.5.2. DMC

1.6. Dobór nastaw algorytmów regulacji w wyniku optymalizacji

1.6.1. PID

1.6.2. DMC

2. CZĘŚĆ LABORATORYJNA

2.1. Sprawdzenie możliwości sterowania i pomiaru stanowiska

Sterowanie stanowiska laboratoryjnego odbywało się przy pomocy funkcji w MATLABIE `MinimalWorkingExample()`

Sygnał sterujący wentylatorem W1 oraz grzałką G1 ustawiano jako argument funkcji `sendControls()`, której pierwszym argumentem była tablica z ID elementów sterowalnych, a drugim tablica z wartościami sygnału sterującego. ID wentylatora W1 to 1, a grzałki G1 5.

```
function MinimalWorkingExample()
    addpath('D:\SerialCommunication');
    initSerialControl COM5 % initialise com port
    while(1)

        %% obtaining measurements
        measurements1 = readMeasurements(1);
        measurements3 = readMeasurements(3);

        %% sending new values of control signals
        sendControls([ 1,5], ... send for these elements
                     [ 50,27]); % new values

         measurement = readMeasurements(1:1)
        %% synchronising with the control process
        waitForNewIteration();
    end
end
```

Należało również określić wartość pomiaru temperatury w punkcie pracy

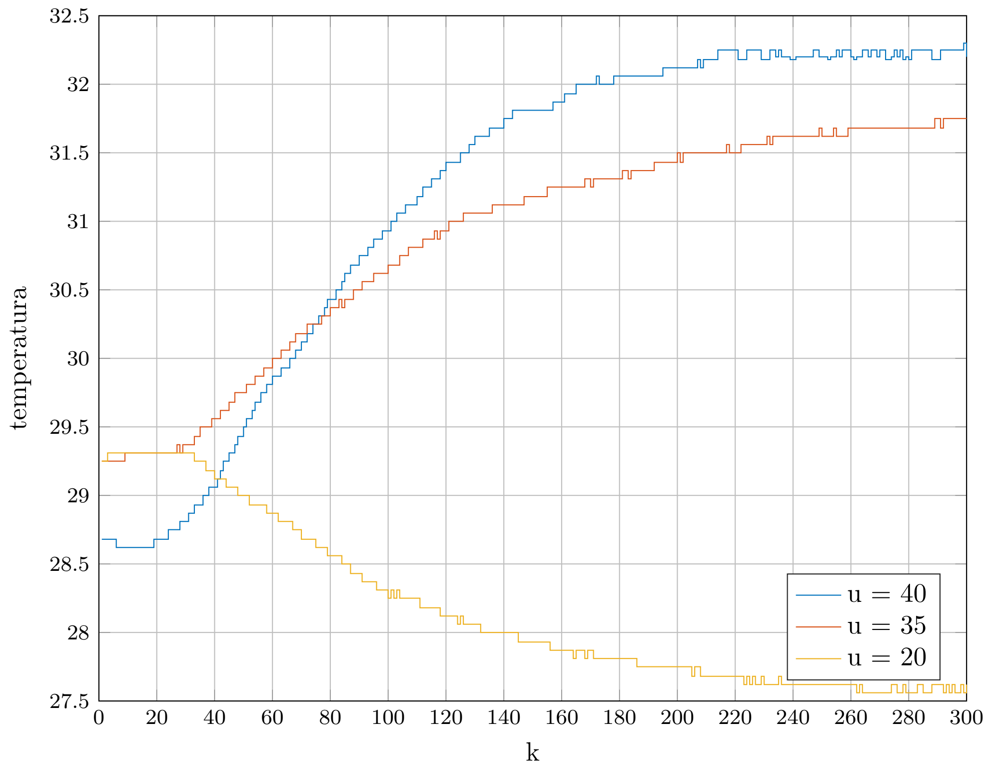
$$G1 = 27$$

Na pierwszym laboratorium temperatura w punkcie pracy $u = 27$ wyniosła 29.12 °C

2.2. Wyznaczenie odpowiedzi skokowej procesu

2.2.1. Odpowiedzi skokowe

Odpowiedzi skokowe procesu wyznaczono poprzez zmianę sygnału sterującego u z punktu pracy i obserwacji wyjścia.



Rys. 1. Odpowiedzi skokowe dla zmian sygnału sterującego

Przy jednym pomiarze przez nieuwagę zmieniono sygnał sterujący u zanim obiekt osiągnął wartość temperatury w punkcie pracy.

2.2.2. Wzmocnienie statyczne

Dla pomiarów rozpoczynających się z punktu pracy: $u = 27$, 29.12 °C otrzymujemy wzmocnienia równe: $K = 0.320$, dla skoku sterowania $u = 8$, $K = 0.328$, dla skoku sterowania $u = -7$, $K = 0.282$, dla skoku sterowania $u = 13$. Ze względu na na inny punkt pracy widać małą różnicę dla zmiany sterowania u = 13, lecz można przyjąć, że wzmocnienie statyczne to średnia wzmocnień dla dwóch pierwszych zmian sterowania, która wynosi: $K = 0.324$.

2.3. Przekształcenie odpowiedzi skokowej

2.3.1. Odpowiedź skokowe wykorzystywana w algorytmie DMC

Należał uzyskać odpowiedź skokową wykorzystywaną w algorytmie DMC (zestaw liczb s_1, s_2, ...), a więc taką, która odpowiadałaby skokowi jednostkowemu sygnału sterującego w chwili $k = 0$ na wartość $u = 1$. Przypomnienie wzoru wykorzystywanego w projekcie:

$$s_k = \frac{y(k) - Ypp}{|Uk - Upp|}$$

gdzie $k = 1, ..., D$ oraz

$$u(k) = \begin{cases} Upp & \text{dla } k < 0 \\ Uk & \text{dla } k >= 0 \end{cases}$$

Dla pomiarów z laboratoriów, powyższe równanie zaimplementowano:

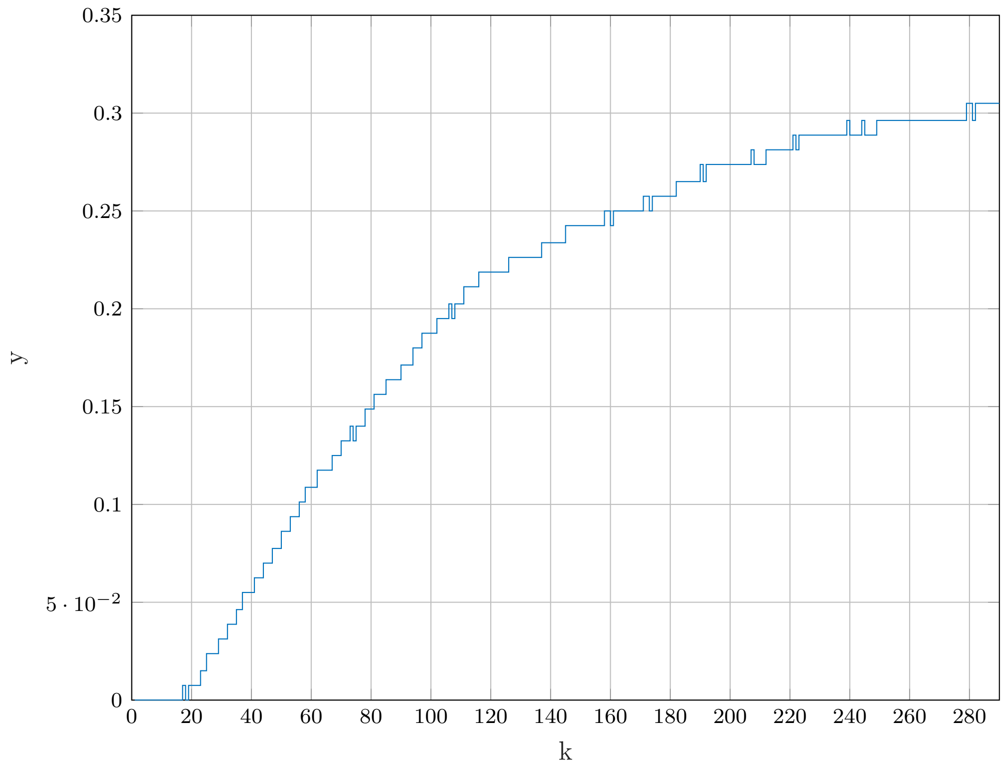
```
for k = 1:300
    if k > 10

        sp(k-10) = (z2_27_35(k)-z2_27_35(10))/8;

    end
end
```

Bardzo ważnym szczegółem jest przesunięcie obliczeń o 10 próbek. Odpowiedzi skokowe były mierzone od chwili $k = 0$, natomiast sama zmiana sygnału u następowała dopiero w chwili $k = 10$. Ponieważ zestaw liczb s_k zakłada zmianę sygnału sterującego w chwili $k = 0$ konieczne jest przeliczenie punktów pomiarowych od 10. próbki.

Poniżej przedstawiono wykres odpowiedzi skokowej wykorzystywanej w algorytmie DMC.



Rys. 2. Odpowiedź skokowa dla skoku jednostkowego sygnału sterującego

2.3.2. Aproksymacja odpowiedzi skokowej

Mając odpowiedź skokową, można wyznaczyć model transmitancyjny obiektu. W tym celu należy aproksymować odpowiedź jako człon inercyjny drugiego rzędu z opóźnieniem, który opisany jest następującą transmitancją:

$$G(s) = \frac{K}{(sT_1+1)(sT_2+1)} e^{-T_d T_p s}$$

która po zastosowaniu transformaty Z wygląda:

$$G(z) = \frac{b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} z^{-T_d}$$

gdzie:

$$a_1 = -\alpha_1 - \alpha_2$$

$$a_2 = \alpha_1 \alpha_2$$

$$\alpha_1 = e^{-\frac{1}{T_1}}$$

$$\alpha_2 = e^{-\frac{1}{T_2}}$$

$$b_1 = \frac{K}{T_1 - T_2}[(T_1(1-\alpha_1) - T_2(1-\alpha_2)]$$

$$b_2 = \frac{K}{T_1 - T_2}[(\alpha_1 T_2(1-\alpha_2) - \alpha_2 T_1(1-\alpha_1)]$$

co przekłada się na równanie różnicowe postaci:

$$y(k) = b_1 u(k - T_d - 1) + b_2 u(k - T_d - 2) - a_1 y(k-1) - a_2 y(k-2);$$

Poniżej przedstawiono implementację wzorów w MATLABIE:

```
% wyliczenie stałych
    alfa1 = eu^(-1/T1);
    alfa2 = eu^(-1/T2);
    a1 = -alfa1 - alfa2;
    a2 = alfa1*alfa2;
    b1 = (K/(T1-T2))*(T1*(1-alfa1)-T2*(1-alfa2));
    b2 = (K/(T1-T2))*(alfa1*T2*(1-alfa2)-alfa2*T1*(1-alfa1));

    % inicjalizacja modelu
    ymod(1:duration) = 0;
    u(1:19) = 1;
    u(20:duration) = 1;
    e(1:duration) = 0;

    % fmincon czasem wylicza niepoprawne wspolczynniki
    if(~isnan(a1) && ~isnan(a2) && ~isnan(b1) && ~isnan(b2))

        err = 0;
        for k=20:duration
            ymod(k) = b1*u(k-Td-1) + ...
                b2*u(k-Td-2) - ...
                a1*ymod(k-1) - ...
                a2*ymod(k-2);
```

```
            e(k) = sp(k) - ymod(k);
            err = err + e(k)^2;

        end
    end
```

2.3.3. Optymalizacja parametrów modelu

W celu optymalizacji parametrów aproksymowanego modelu, ponownie skorzystano z funkcji `fmincon`.

```
%% Optymalizacja modelu

% fmincon
objective = @(x) aproksymacja (x(1), x(2), x(3));
x_initial = [90, 100, 0.3];
lb = [10, 11, 0];
ub = [150, 151, 1];

nonlcon = [];

options = optimset('Display', 'iter');
[x_optimal, fval] = fmincon(objective, x_initial, ...
    [], [], [], [], lb, ub, nonlcon, options);

%% Funkcja błędu:

% funkcja bledu dla fmincon
function err = aproksymacja(T1, T2, K)

    % opoznienie
    Td = 17;

    % kiedy T1 == T2 wspolczynniki wychodza NaN
    if(T1 ~= T2)
        load("zad2_pomiary.mat");
        pomiary = z2_27_35;

        % odpowiedzi skokowe
        for k = 1:300
            if k > 10
                sp(k-10) = (pomiary(k)-pomiary(10))/8;
            end
        end
        %%%
        ZAIMPLEMENTOWANE PARAMETRY
        ORAZ RÓWNANIE RÓŻNICOWE
        %%%
    end
end
```

Warto zwrócić uwagę, że `fmincon` nie zawsze poprawnie wylicza współczynniki równania różnicowego, także potrzebne jest sprawdzenie, czy są one liczbami rzeczywistymi oraz, że T_1 nie jest równe T_2 (w przypadku równości dochodzi do dzielenia przez 0).

Po przeprowadzonej optymalizacji, otrzymano optymalne parametry to:

$$T_1 = 10$$

$$T_2 = 75$$

$$K = 0.307$$

Uśredniając opóźnienie ze zmierzonych odpowiedzi skokowych (zarówno przy podnoszeniu jak i obniżaniu temperatury obiektu), otrzymuje się opóźnienie:

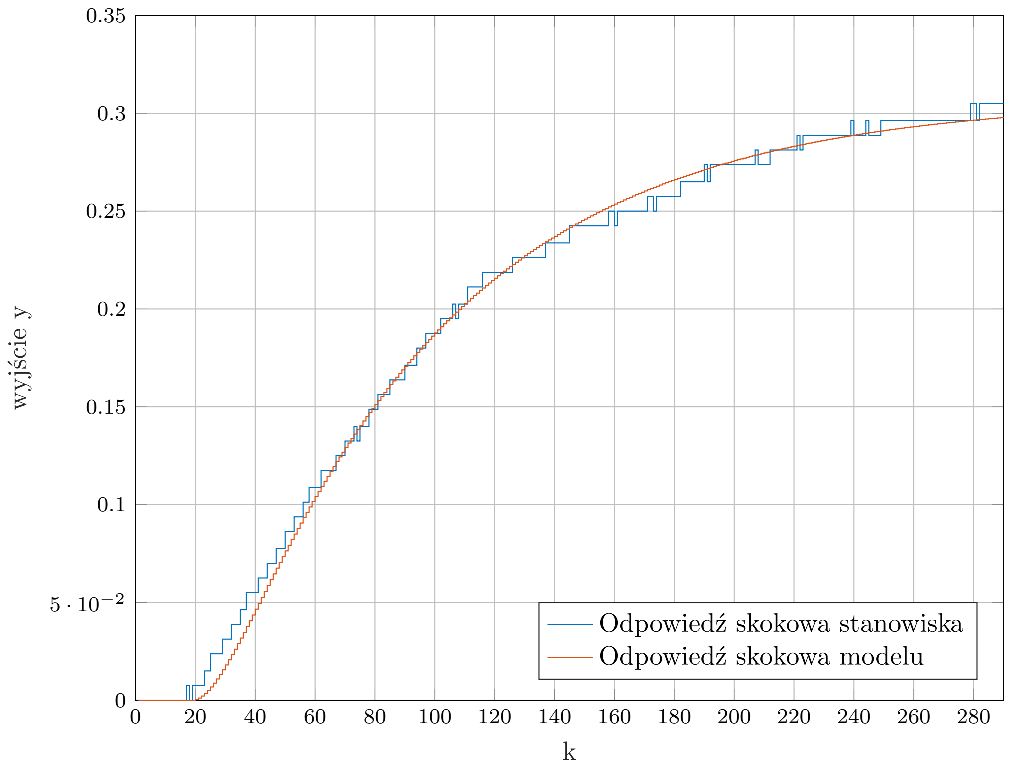
$$T_p = 17$$

A zatem transmitancja modelu będzie wyrażać się wzorem:

$$G(s) = \frac{0.307}{(10s+1)(75s+1)} e^{-17T_p s}$$

2.3.4. Porównanie aproksymowanego modelu z obiektem

Poniżej przedstawiono na jednym rysunku odpowiedź skokową modelu oraz odpowiedź skokową stanowiska grzewczo-chłodzącego.



Rys. 3. Odpowiedź skokowa modelu oraz stanowiska grzewczo-chłodzącego

2.4. Regulacja PID oraz DMC

Regulacja PID:

```
function uk = regulacjaPID(T, k, u, ypom, yzad, K_pid, Ti, Td)

    r0 = K_pid * (1 + T/(2*Ti) + Td/T);
    r1 = K_pid * (T/(2*Ti) - 2*Td/T - 1);
    r2 = K_pid * Td/T;

    e(k) = yzad(k) - ypom(k);
    e(k-1) = yzad(k-1) - ypom(k-1);
    e(k-2) = yzad(k-2) - ypom(k-2);

    uk = r2*e(k-2) + r1*e(k-1) + r0*e(k) + u(k-1);
end
```

Regulacja DMC:

```
function du = regulacjaDMC(k, u, ypom, yzad, D, N, K, Mp)

    e(k) = yzad(k) - ypom(k);

    Ke = 0;
    for i = 1 : N
        Ke = Ke + K(1, i);
    end

    elem = 0;
    for j = 1:D-1

        ku = K(1,:) * Mp(:,j);

        if k-j<= 1
            du_kmj = 0;
        else
            du_kmj = u(k-j) - u(k-j-1);
        end

        elem = elem + ku*du_kmj;
    end

    du = Ke * e(k) - elem;

end
```

Macierz M oraz Mp liczone są tak samo jak w projkcie. Po wyznaczeniu odpowiedzi skokowej obiektu, liczone są jednorazowo dla zadanych paramterów długości horyzontów.

2.5. Dobór nastaw regulatorów PID oraz DMC metodą eksperymentalną

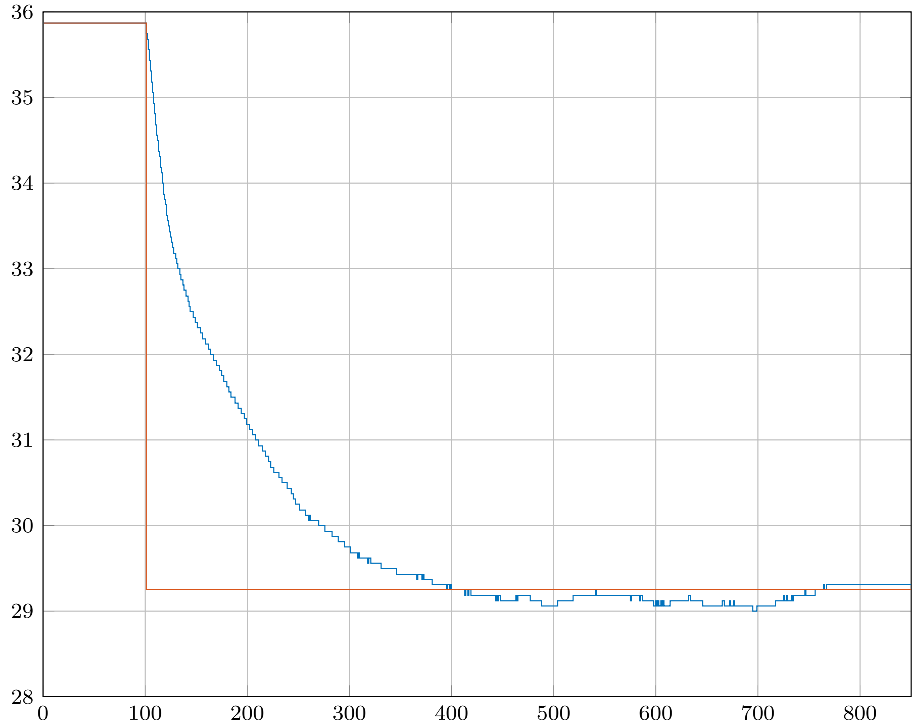
Parametry dobieraliśmy eksperymentalnie, lecz ze względu na długie stałe czasowe pozwoliliśmy sobie na zamieszczenie jedynie wykresów dla optymalnych nastaw, gdzie wyjście ustaliło się po czasie reprezentatywnym na wykresie.

Nastawy regulatora PID:
$K = 5$, $T_i = 60$, $T_d = 0.3$

Nastawy regulatora DMC:
$D = 580$, $N = 580$, $N_u = 80$, $\lambda = 1$

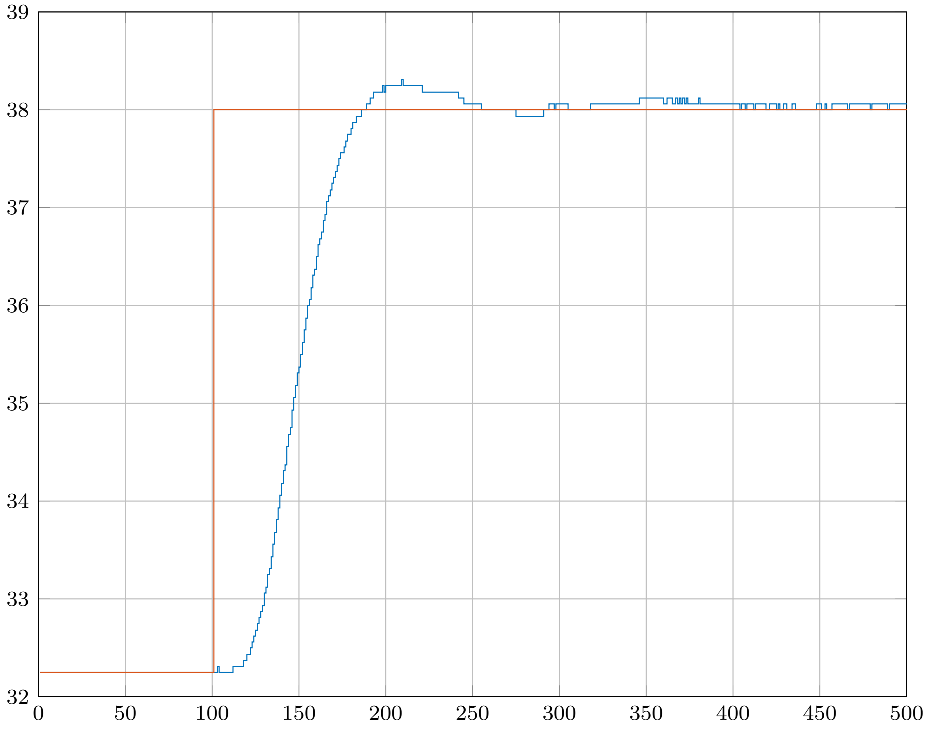
2.5.1. Omówienie wyników i ocena jakości regulacji

Poniżej przedstawione zostały przykłady chłodzenia oraz grzania dla ustalonych nastaw dla regulatora DMC oraz PID:



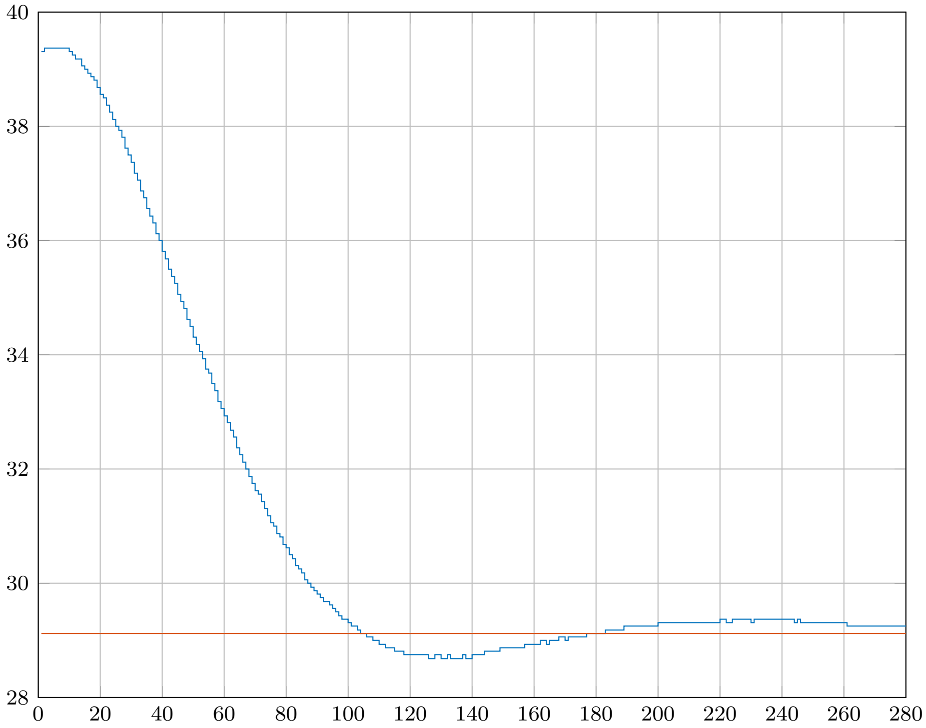
Rys. 4. DMC: Chłodzenie z 36°C do 29°C, Nu = 80, $\lambda = 1$

Błąd średniokwadratowy: E = 1471.5374



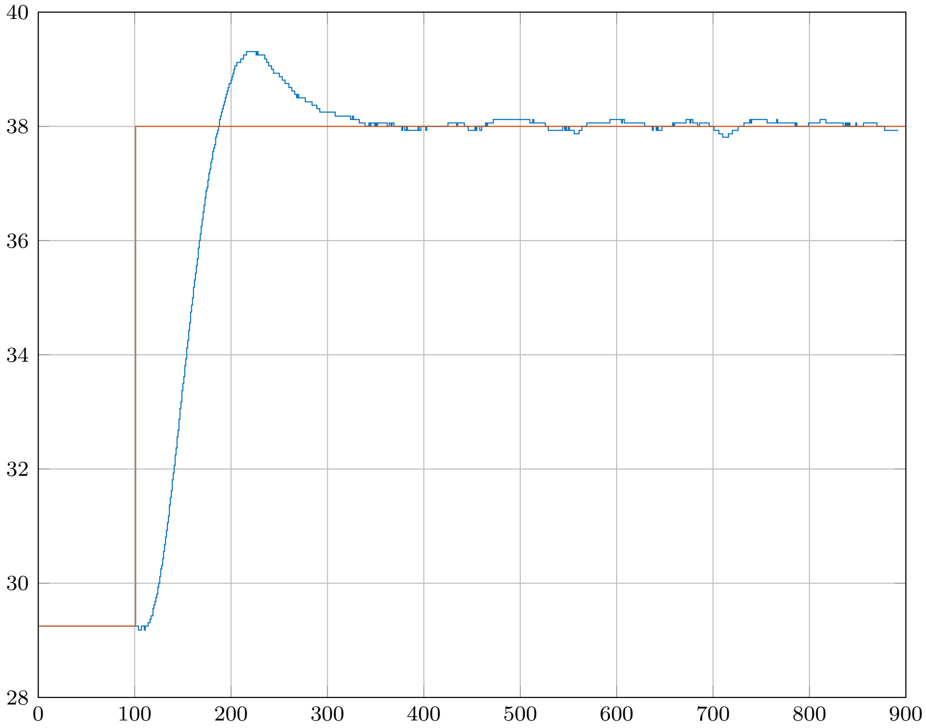
Rys. 5. DMC: Grzanie z 32°C do 38°C, Nu = 80, $\lambda = 1$

Błąd średniokwadratowy: E = 1279.4381



Rys. 6. PID: Chłodzenie z 38°C do 29°C, K=5; Ti=60; Td=0.3

Błąd średniokwadratowy: E = 4047.4239



Rys. 7. PID: Grzanie z 32°C do 38°C, K=5; Ti=60; Td=0.3

Błąd średniokwadratowy: E = 3001.1457

Analizując powyższe wykresy oraz błędy średniokwadratowe można stwierdzić, że zdecydowanie lepiej radzi sobie regulator DMC niż PID. Generuje on mniejsze przesterowanie i mniejszy błąd. Mimo tego warto zauważyć, że grzanie i chłodzenie wykonywane było dla różnych temperatur, więc trudno jednoznacznie ocenić te regulatory, w szczególności na podstawie wartości błędu średniokwadratowego. Czas dążenia do wartości zadanej jest podobny dla obydwu regulatorów, co sugerować może, że jakość działania jest relatywnie podobna z podkreśleniem przewagi DMC nad PIDem na polu przesterowania.