

**Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska**

**Projektowanie układów w sterowania
(projekt grupowy)**

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

Hubert Kozubek, Przemysław Michałczewski

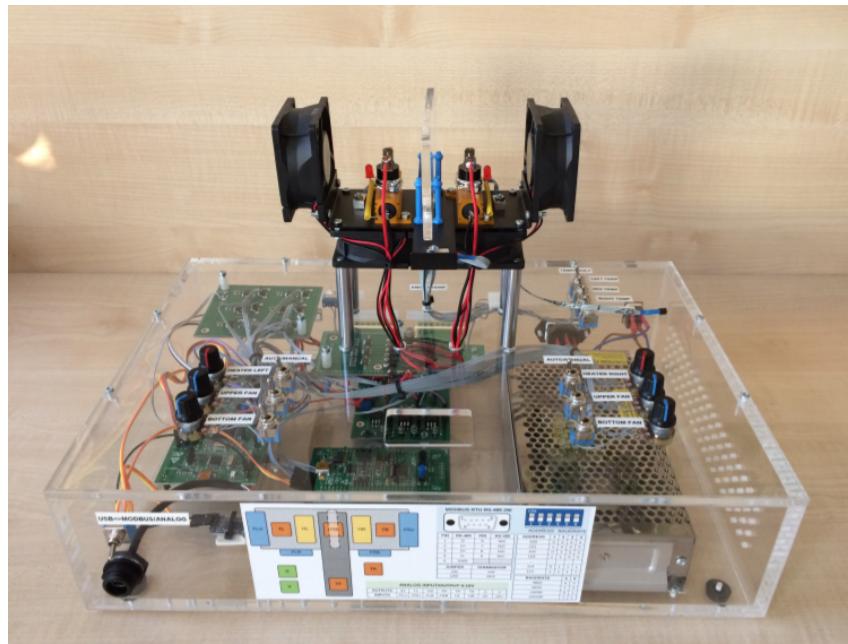
Warszawa, 2021

Spis treści

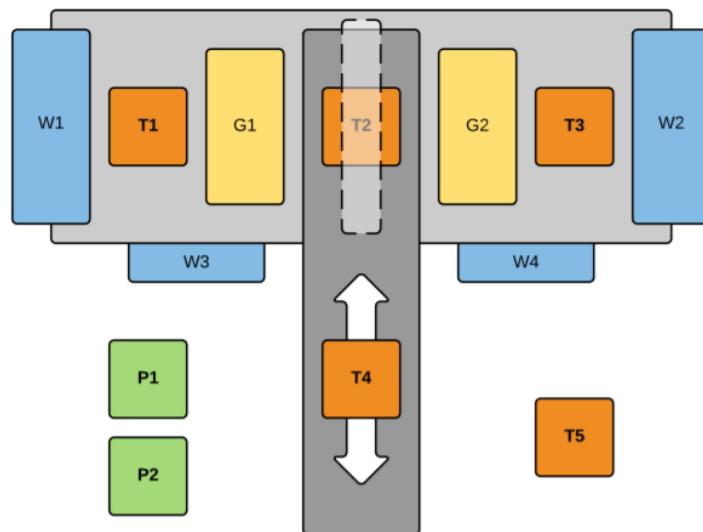
1.	Cele projektu i laboratoriów	1
2.	Przebieg laboratorium	2
3.	Punkt pracy stanowiska	3
4.	Odpowiedzi skokowe procesu	3
5.	Odpowiedź skokowa w algorytmie DMC	4
6.	Algorytm PID i DMC	6
6.1.	Regulator PID	6
6.2.	Regulator DMC	8
7.	Dostrajanie Regulatorów	9
7.1.	Strojenie PID	9
8.	Projekt	12
9.	Sprawdzanie poprawność wartości U_{pp}, Y_{pp}	12
10.	Odpowiedzi skokowe procesu	13
11.	Przekształcenie odpowiedzi skokowej	13
12.	Implementacja PID i DMC	13
12.1.	PID	13
12.2.	DMC	15
13.	Dobór nastawów PID i DMC metodą eksperymentalną	18
13.1.	Nastawy PID	18
13.2.	Nastawy DMC	18
14.	Dobór parametrów PID i DMC automatycznie	18

1. Cele projektu i laboratoriów

Celem niniejszego laboratorium **oraz projektu** było zaprojektowanie, implementacja, weryfikacja poprawności działania oraz dobór parametrów algorytmów regulacji jednowymiarowego procesu na **grzewczym** stanowisku laboratoryjnym przedstawionym na rys 1.



Rys. 1. Stanowisko grzejaco-chłodzące, używane w trakcie laboratoriów



Rys. 2. Schemat stanowiska grzejaco-chłodzącego

2. Przebieg laboratorium

Rozpoczynając pracę na stanowisku laboratoryjnym należało ustawić moc wentylatora W1 na 50%. Wentylator ten był traktowany jako cecha otoczenia. Dodatkowo sprawiał on, że temperatura grzałki opadała szybciej, co było szczególnie przydatne pomiędzy doświadczeniami.

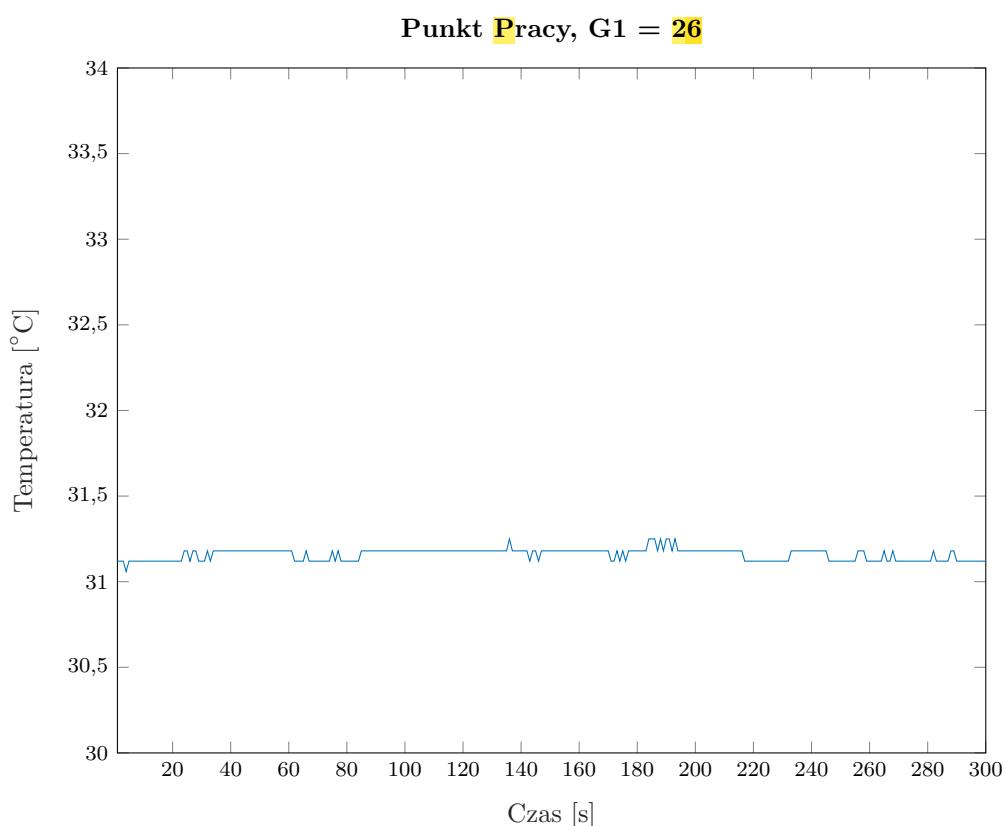
W ramach laboratorium należało wykonać 5 zadań.

1. Odczytać wartość pomiaru termometru T1 dla mocy 26 grzałki G1%.
2. Wyznaczyć odpowiedź skokową procesu dla 3 różnych wartości G1%.
3. Wybrać jedną z dopowiedzi skokowych, przekształcić ją i wykorzystać w algorytmie DMC.

4. Zaimplementować algorytm PID i DMC, od regulacji procesu stanowiska, w języku MATLAB.
5. Dobrać nastawy algorytmu PID oraz parametry algorytmu DMC metodą eksperymentalną.

3. Punkt pracy stanowiska

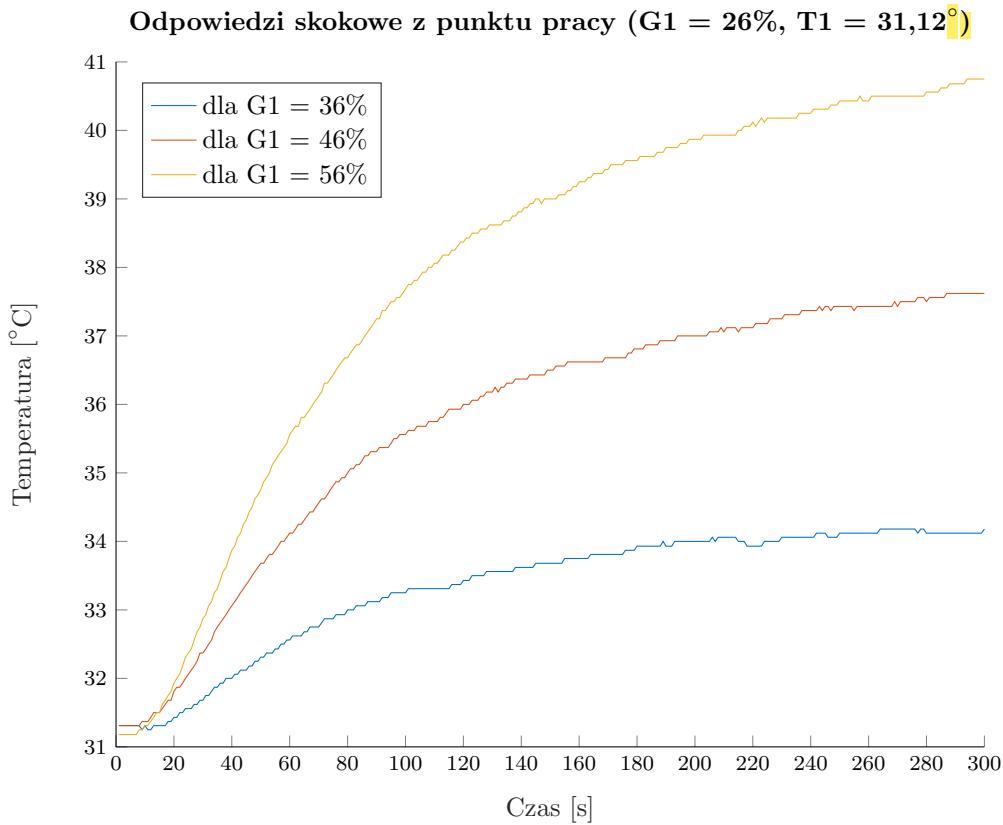
W pierwszej kolejności należało sprawdzić możliwość sterowania i pomiaru w komunikacji ze stanowiskiem. Następnie odczytać wartość temperatury termometru T1 w wyznaczonym punkcie pracy $G1=26\%$. Po ustawieniu mocy grzałki i oczekaniu, aż temperatura T1 ustabilizuje się, odczytana wartość termometru T1 wynosiła $31,12\text{ }^{\circ}\text{C}$. Wykres temperatury na termometrze T1 został przedstawiony na rys. 3



Rys. 3. Ustalanie się temperatury dla punktu pracy

4. Odpowiedzi skokowe procesu

W tej części laboratorium należało przeprowadzić eksperyment dla 3 różnych wartości mocy grzałki $G1$. Rozpoczynając eksperyment z punktu pracy $G1=26\%$, wyznaczono odpowiedzi skokowe procesu. Eksperyment był wykonyany dla trzech różnych zmian sygnału sterującego, $G1=36\%$, $G1=46\%$ oraz $G1=56\%$. Wykresy przedstawiające zmiany temperatury przedstawiono na rys. 4



Rys. 4. Odpowiedź skokowa procesu

5. Odpowiedź skokowa w algorytmie DMC

Wykonanie tego zadania polegało na przekształceniu jednej z odpowiedzi skokowych, tak aby otrzymać odpowiedź skokową używaną w algorytmie DMC. W tym celu wybrano drugą odpowiedź skokową, tj. skok G_1 z mocy 26% do mocy 46%. Do przekształcenia zebranej odpowiedzi skokowej, na taką nadającą się do algorytmu DMC wykorzystano program TODO: "SkokDMC.m". Program ten wylicza potrzebną odpowiedź skokową przy użyciu prostego wzoru.

$$S(i) = \frac{Y(i) - Y_{pp}}{U_{skok} - U_{pp}} \quad (1)$$

gdzie:

- $S(i)$ - odpowiedź skokowa potrzebna do algorytmu DMC,
- $Y(i)$ - odpowiedź skokowa przed przekształceniem,
- Y_{pp} - wartość wyjścia w chwili $k=0$ (tutaj $Y_{pp} = 31,12$),
- U_{skok} - wartość sterowania w chwili $k=0$ i później (tutaj $U_{skok} = 46$),
- U_{pp} - wartość sterowania przed chwilą $k=0$ (tutaj $U_{pp} = 26$)

W ten sposób przekształcona odpowiedź skokowa została zapisana do pliku TODO: "dane1.mat" i wykorzystana w dalszych częściach laboratorium.

Poza przekształceniem odpowiedzi skokowej należało ją jeszcze przybliżyć używając w tym celu członu inercyjnego drugiego rzędu z opóźnieniem.

$$G(s) = \frac{K}{(sT_1 + 1)(sT_2 + 1)} e^{-T_d s} \quad (2)$$

Po dyskretyzacji danej transmitancji otrzymujemy

$$G(z) = \frac{b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} z^{-T_d} \quad (3)$$

gdzie

$$\begin{aligned} a_1 &= -\alpha_1 - \alpha_2 \\ a_2 &= \alpha_1 \alpha_2 \\ \alpha_1 &= e^{-\frac{1}{T_1}} \\ \alpha_2 &= e^{-\frac{1}{T_2}} \\ b_1 &= \frac{K}{T_1 - T_2} [T_1(1 - \alpha_1) - T_2(1 - \alpha_2)] \\ b_2 &= \frac{K}{T_1 - T_2} [\alpha_1 T_2 (1 - \alpha_2) - \alpha_2 T_1 (1 - \alpha_1)] \end{aligned} \quad (4)$$

Z wykresu odpowiedzi skokowej procesu zostało odczytane opóźnienie. W naszym przypadku $T_d = 9$. Aby wyznaczyć wartości pozostałych współczynników użyto dostępnej w matlabie funkcji `ga`, która minimalizuje wartość zadanej funkcji z wykorzystaniem algorytmu genetycznego. Funkcja minimalizowana, to funkcja wyliczająca sumę kwadratów błędów pomiędzy odpowiedzią skokową, a transmitancją przybliżającą.

```
% aproksymacja odpowiedzi skokowej

function ERR = AproksSkokDMC(X)

data = load('dane1.mat');
S = data.S;
time = data.time;

T1 = X(1);
T2 = X(2);
K = X(3);
Td = 9;
y(1:time) = 0;

alpha1 = exp(-1/T1);
alpha2 = exp(-1/T2);
a1 = -alpha1-alpha2;
a2 = alpha1*alpha2;
b1 = K*(T1*(1-alpha1)-T2*(1-alpha2))/(T1-T2);
b2 = K*(alpha1*T2*(1-alpha2)-alpha2*T1*(1-alpha1))/(T1-T2);

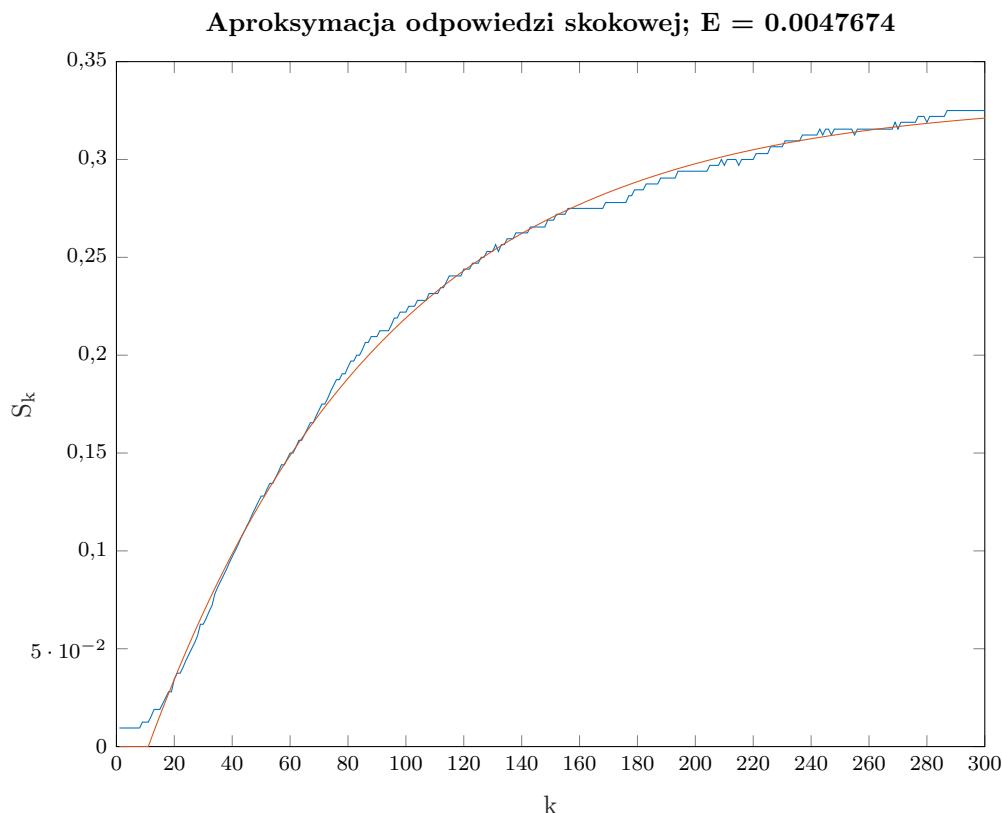
for k = Td+3:time
    y(k) = b1 + b2 - a1*y(k-1) - a2*y(k-2);
end

e = S - y';

ERR = (norm(e))^2;
end
```

Następnie używając skryptu TODO: "Optymalizacja.m" zostały wyznaczone pozostałe parametry transmitancji przybliżającej odpowiedź skokową. Ostateczne wartości parametrów to

$$\begin{aligned}
 K &= 0.330938 \\
 T_1 &= 0.000907 \\
 T_2 &= 82.104622 \\
 T_d &= 9
 \end{aligned} \tag{5}$$



Rys. 5. Odpowiedź skokowa procesu $\boxed{1}$ oraz transmitancja ją aproksymująca

Wykres zarówno odpowiedzi skokowej, jak i transmitancji ją przybliżającej został zamieszczony na rys. **5**

6. Algorytm PID i DMC

Kolejnym podpunktem zadań laboratoryjnych było zaimplementowanie algorytmu regulacji PID oraz DMC w języku MATLAB.

6.1. Regulator PID

```
% implementacja PID
function U = PID(e)

    persistent Upop
    persistent e0
    persistent e1
    persistent e2
```

```
persistent K
persistent Ti
persistent Td
persistent Tp
persistent r2
persistent r1
persistent r0

% Ograniczenia sterowania
Gmax = 100;
Gmin = 0;

%      Upp = 26;
%      Ypp = 31.12;

if isempty(e0)
    Upop = 0;           % sterowanie w punkcie pracy
    e0=0;
    e1=0;
    e2=0;

% Nastawy regulatora
K = 0.5 * 43 * 1.5;      %Kk = 43, Tk = 36
Ti = 0.5 * 36*2;         % * 4; %inf; 10
Td = 0.125 * 36;         % * 0.6; % 0.4
Tp = 1;

r2 = K*Td/Tp;
r1 = K*(Tp/(2*Ti)-2*Td/Tp - 1);
r0 = K*(1+Tp/(2*Ti) + Td/Tp);

end

% przesuniecie uchybow
e2 = e1;
e1 = e0;
e0 = e;

U = r2*e2 + r1*e1 + r0*e0 + Upop;

if U > Gmax
    U = Gmax;
end

if U < Gmin
    U = Gmin;
end

Upop = U;
```

6.2. Regulator DMC

```
%implementacja DMC
function U = DMC(yzad, y, D, N, Nu, lambda)

persistent init
persistent S
persistent M
persistent Mp
persistent K
persistent dUP
persistent Upop

if isempty(init)

    % Wczytanie macierzy S z pliku dane1.mat
    data = load('dane1.mat');
    S = data.S;

    % przedluzenie wektora S
    for i = D+1:D+N
        S(i) = S(D);
    end

    % Inicjalizacja macierzy
    M = zeros(N, Nu);
    for i = 1:Nu
        M(i:N, i)=S(1:N-i+1);
    end

    Mp = zeros(N, D-1);
    for i = 1:(D-1)
        Mp(1:N, i) = S(i+1:N+i) - S(i);
    end

    I = eye(Nu);

    K = ((M'*M + lambda*I)^(-1))*M';
    dUP = zeros(D-1,1);
    Upop = 26;
    init = 1;
end

% Ograniczenia sterowania
Gmax = 100;
Gmin = 0;

Y0 = zeros(N,1);
dU = zeros(Nu,1);

% liczone online
Yzad = yzad*ones(N,1);
```

```

Y = y*ones(N,1) ;

Y0 = Y + Mp*dUP;
dU = K*(Yzad - Y0);
du = dU(1);

for n=D-1:-1:2
    dUP(n) = dUP(n-1);
end
dUP(1) = du;

U = Upop + du;

if U > Gmax
    U = Gmax;
end

if U < Gmin
    U = Gmin;
end

Upop = U;
end

```

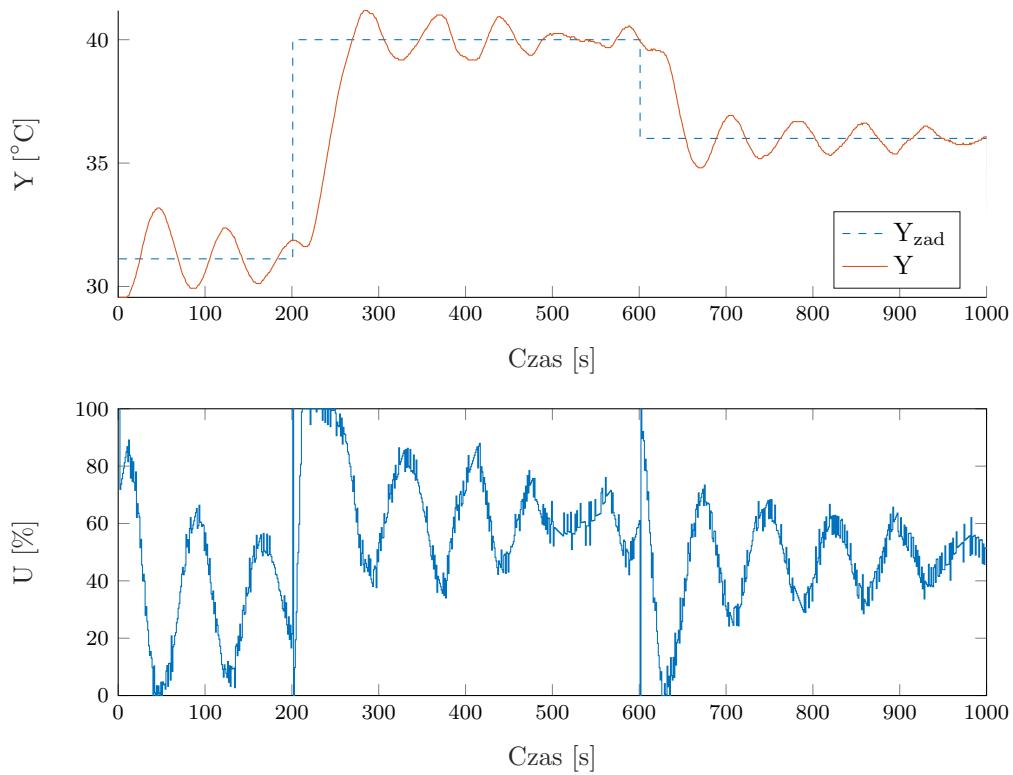
7. Dostrajanie Regulatorów

Ostatnim zadaniem był dobór nastawów obu algorytmów regulacji. W tym celu skozystano z wcześniejszej uzyskanej transmitancji aproksymującej skok procesu, aby dobrać parametry obu regulatorów. W ten sposób można było przeprowadzić więcej eksperymentów w krótszym czasie. Uzyskane w ten sposób parametry zostały przetestowane na realnym procesie.

7.1. Strojenie PID

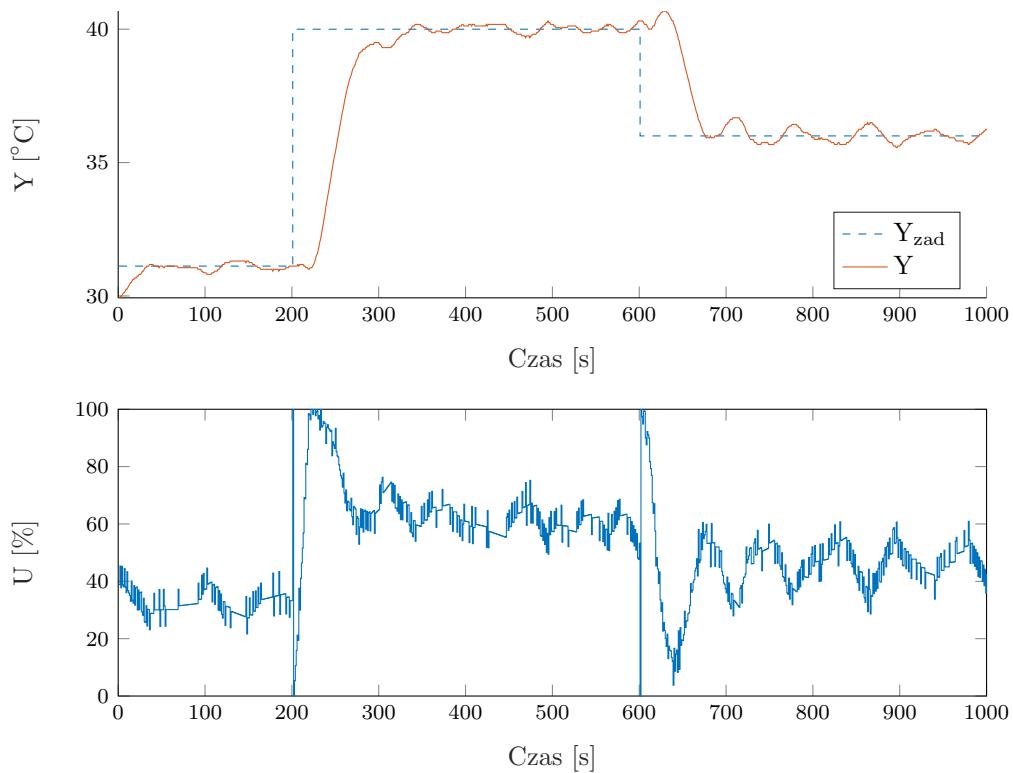
Do strojenia regulatora PID została wykorzystana metoda Zieglera-Nicholsa. Parametry do niej potrzebne wyznaczono na procesie symulowanym. Oscylacje niegasnące otrzymano dla parametrów $K_k = 43$ oraz $T_k = 36$. Na tej podstawie dobrano następujące parametry regulatora PID $K = 21,5; T_i = 18; T_d = 4,5$. Oczywiście do regulacji zostały wykorzystane parametry PID-a dyskretnego określone wzorami ($T_p = 1$)

$$\begin{aligned}
r_2 &= K \frac{T_d}{T_p} \\
r_1 &= K \left(\frac{T_p}{2T_i} - 2 \frac{T_d}{T_p} - 1 \right) \\
r_0 &= K \left(1 + \frac{T_p}{2T_i} + \frac{T_d}{T_p} \right)
\end{aligned} \tag{6}$$

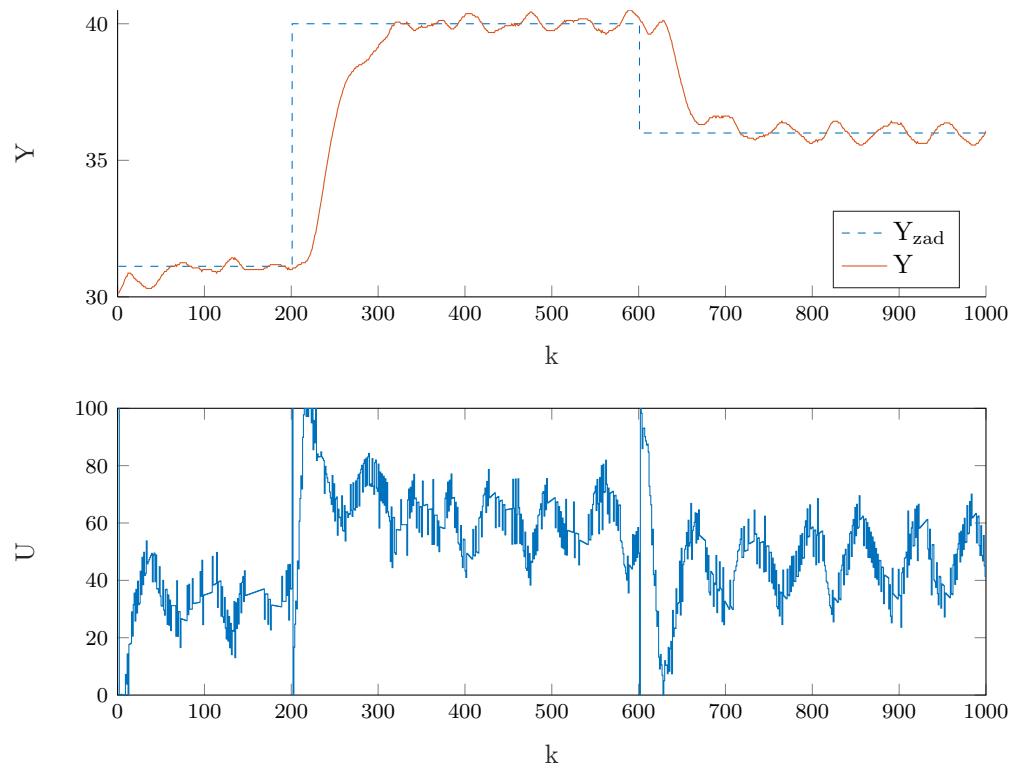


Rys. 6. Regulacja PID, eksperyment 1

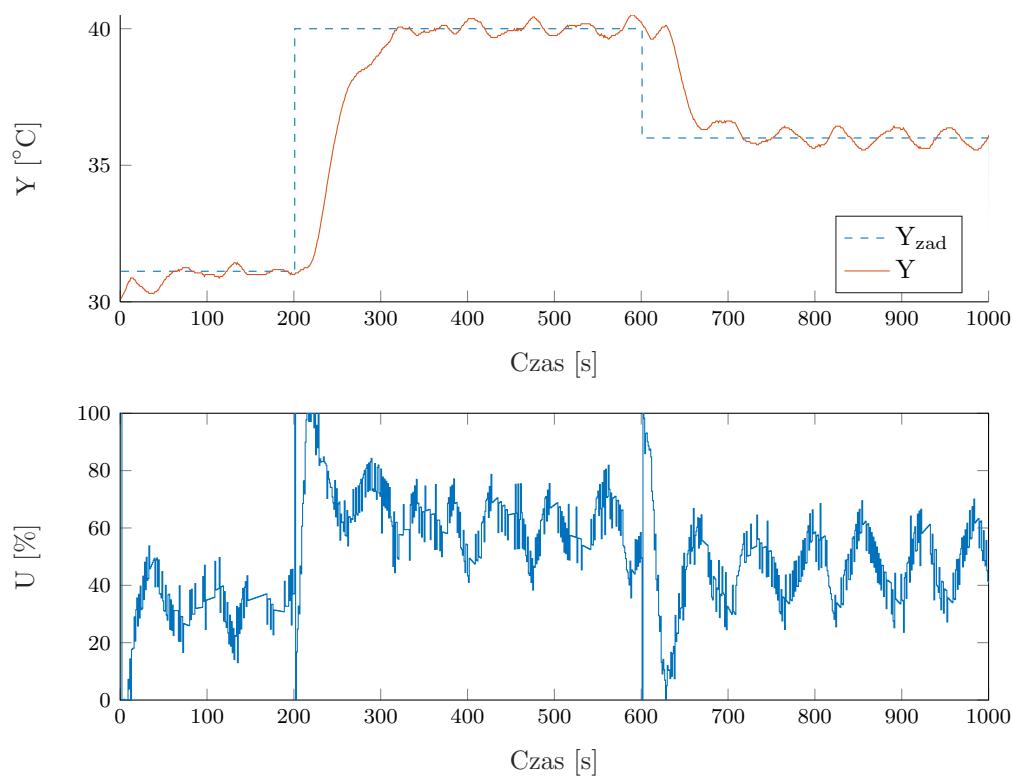
Wyniki regulacji PID dla tych parametrów przedstawiono na rys. 6. W dalszej kolejności sprawdzane były jeszcze inne wartości parametrów, w celu znalezienia jak najlepszych nastawów.



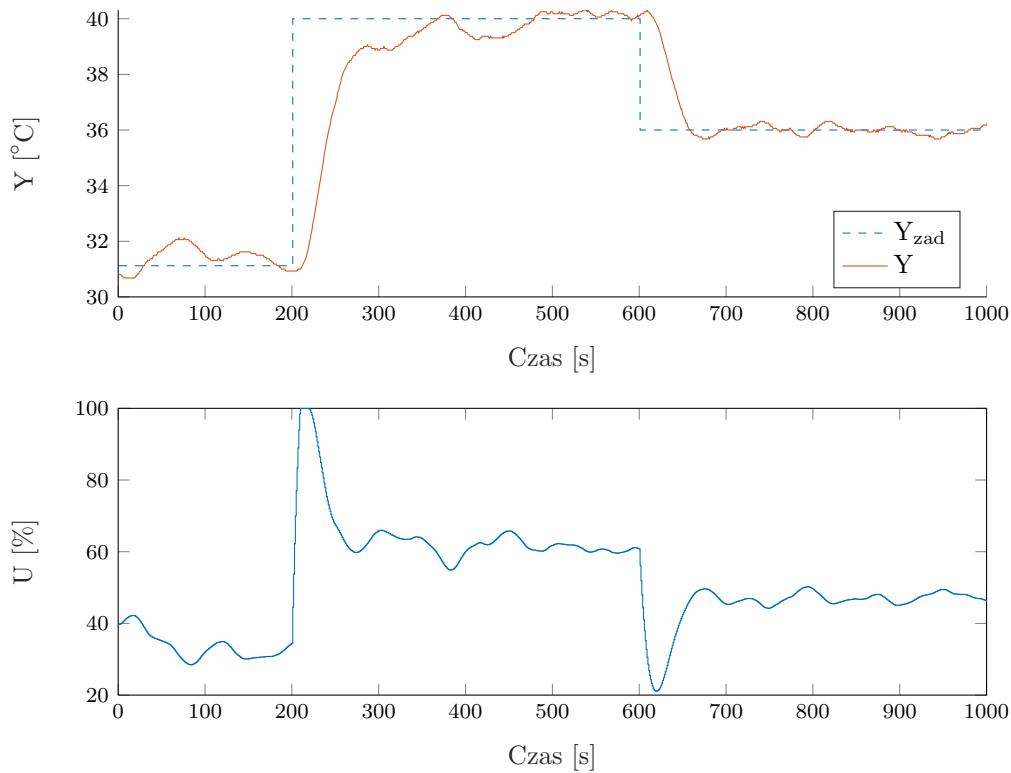
Rys. 7. Regulacja PID, eksperyment 2



Rys. 8. Regulacja PID, eksperyment 3



Rys. 9. Regulacja PID, eksperyment 4



Rys. 10. Regulacja DMC, eksperyment 1

Z wykresów można wyczytać, że zmiana parametrów PDI poprawiła jakość regulacji. Co do regulatora DMC natomiast, to działa on całkiem dobrze, jednak zarówno PDI jak i DMC są narażone na zakłócenia, które można zauważać na wykresach.

8. Projekt

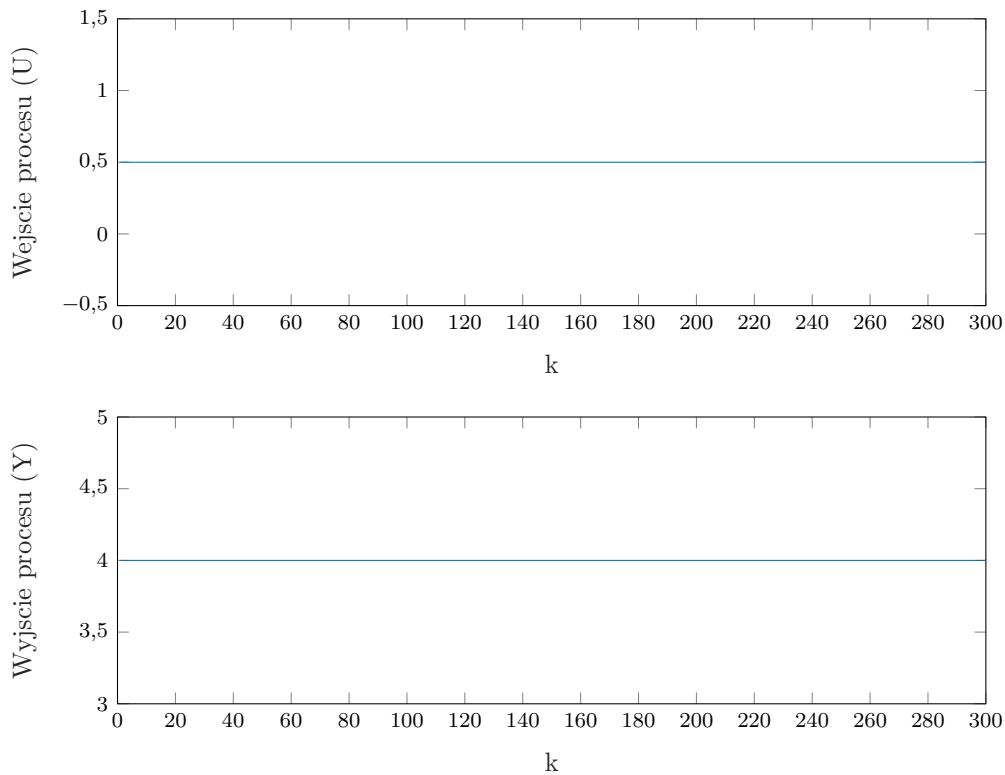
Zadanie projektowe wykorzystywało symulowany obiekt regulacji. Wyjście obiektu można wyznaczyć przy pomocy polecenia

$$Y(k) = \text{symulacja obiektu} \text{y p1}(U(k-10), U(k-11), Y(k-1), Y(k-2))$$

Wartości sygnałów w wejścia i wyjścia procesu w punkcie pracy wynoszą $U_{pp} = 0,5$, $Y_{pp} = 4$, natomiast ograniczenia sterowania wynoszą $U^{\min} = 0,3$, $U^{\max} = 0,7$, a okres próbkowania wynosi 0,5.

9. Sprawdzanie poprawność wartości U_{pp} , Y_{pp}

W celu sprawdzenia poprawności punktu pracy została przeprowadzona symulacja, gdzie na wejście podano U_{pp} , jako poprzednie wartości y podano Y_{pp} i sprawdzono wartość wyjścia procesu w następnych chwilach. Wyniki symulacji przedstawiono na rys. 11



Rys. 11. Sprawdzanie poprawości punktu pracy

10. Odpowiedzi skokowe procesu

W tej części wyznaczone zostały odpowiedzi skokowe procesu, dla kilku zmian sygnału sterującego, gdzie wartość początkowa sterowania wynosiła U_{pp} . Wyniki symulacji przedstawia rys. 12

W celu ustalenia, czy proces jest liniowy, został narysowany wykres zależności $Y(U)$, dla dozwolonych wartości sterowania. Wykres ten został przedstawiony na rys. 13. Z wykresu wynika, że zależność jest liniowa, zatem wzmacnienie statyczne można wyznaczyć jako tg prostej. W naszym przypadku $K_{stat} = 2$

11. Przekształcenie odpowiedzi skokowej

Do przekształcenia została wybrana odpowiedź skokowa dla sterowania $U_{skok} = 0,65$ W pierwszej kolejności zebrane dane, zostały przesunięte w czasie, tak aby skok sterowania następował dla $k = 0$. Następnie wartości skoku zostały przeskalowane i przesunięte w odpowiedni sposób, tak aby odpowiedź dało się wykorzystać w algorytmie DMC. Wynikowa odpowiedź skokowa została przedstawiona na rys. 14

12. Implementacja PID i DMC

12.1. PID

```
function E = PID_funkcja(X)
    K = X(1);
```

```
Ti = X(2);  
Td = X(3);  
  
% Punkt pracy  
Upp = 0.5;  
Ypp = 4;  
  
% Ograniczenia wartosci sygnalu sterujÄ...cego  
Umin = 0.3;  
Umax = 0.7;  
du_max = 0.05;  
  
% Czas symulacji  
time = 1500;  
  
% Deklaracja wektora sterowan i wartosci zadanych  
U(1:time) = Upp;  
Y(1:time) = Ypp;  
  
Yzad(1:50) = Ypp;  
Yzad(51:200) = 4.1;  
Yzad(201:500) = 3.85;  
Yzad(501:800) = 4.05;  
Yzad(801:1200) = 4.15;  
Yzad(1201:time) = 3.95;  
  
y_zad = Yzad - Ypp;  
u = U - Upp;  
  
% Inicjalizacja wektorÄłw  
e(1:time) = 0;  
y(1:time) = 0;  
  
u_max = Umax - Upp;  
u_min = Umin - Upp;  
  
% Wyznaczone eksperimentalnie  
Tp = 0.5;  
  
r2 = K*Td/Tp;  
r1 = K*(Tp/(2*Ti)-2*Td/Tp - 1);  
r0 = K*(1+Tp/(2*Ti) + Td/Tp);  
  
for k = 12:time  
    Y(k) = symulacja_objektu1Y_p1(U(k-10), U(k-11), Y(k-1), Y(k-2));  
    y(k) = Y(k) - Ypp;  
    e(k) = y_zad(k) - y(k);  
  
    du = r2*e(k-2) + r1*e(k-1) + r0*e(k);  
  
    if du > du_max
```

```

        du = du_max;
    end

    if du < - du_max
        du = - du_max;
    end

    u(k) = u(k-1) + du;

    if u(k) > u_max
        u(k) = u_max;
    end

    if u(k) < u_min
        u(k) = u_min;
    end

    U(k) = u(k) + Upp;
end

E = 0;
for k = 12:time
    E = E + e(k)^2;
end
end

```

12.2. DMC

```

function E = DMC_funkcja(X)

N = X(1);
Nu = X(2);
lambda = X(3);

dataS = load('S.mat');
S = dataS.S;

dataD = load('D.mat');
D = dataD.D;

% Punkt pracy
Upp = 0.5;
Ypp = 4;

% Ograniczenia wartosci sygnalu sterujÄ...cego
Umin = 0.3;
Umax = 0.7;
du_max = 0.05;

% Czas symulacji
time = 1500;

```

```
Yzad(time,1) = 0;
Yzad(1:50) = Ypp;
Yzad(51:200) = 4.1;
Yzad(201:500) = 3.85;
Yzad(501:800) = 4.05;
Yzad(801:1200) = 4.15;
Yzad(1201:time) = 3.95;

U(1:time) = Upp;
Y(1:time) = Ypp;
e(1:time) = 0;

% Obliczenia offline
S = [S; zeros(N,1)];
for i = D+1:D+N
    S(i) = S(D);
end

M = zeros(N, Nu);
for i = 1:Nu
    M(i:N, i)=S(1:N-i+1);
end

Mp = zeros(N, D-1);
for i = 1:(D-1)
    Mp(1:N, i) = S(i+1:N+i) - S(i);
end

I = eye(Nu);
K = ((M'*M + lambda*I)^(-1))*M';

% inicjalizacja
dUP = zeros(D-1,1);
Y0 = zeros(N,1);
dU = zeros(Nu,1);
Yzad_DMC = zeros(N,1);
Y_DMC = zeros(N,1);

u = U - Upp;
yzad = Yzad - Ypp;

y(1:time) = 0;
u(1:time) = 0;

u_max = Umax - Upp;
u_min = Umin - Upp;

% liczone online
```

```
for k = 12:time
    Y(k) = symulacja_objektu1Y_p1(U(k-10), U(k-11), Y(k-1), Y(k-2));
    y(k) = Y(k) - Ypp;
    e(k) = (yzad(k) - y(k))^2;

    Yzad_DMC = yzad(k)*ones(N,1);
    YDMC = y(k)*ones(N,1);

    Y0 = YDMC + Mp*dUP;
    dU = K*(Yzad_DMC - Y0);
    du = dU(1);

    if du > du_max
        du = du_max;
    end

    if du < - du_max
        du = - du_max;
    end

    for n=D-1:-1:2
        dUP(n,1) = dUP(n-1,1);
    end
    dUP(1) = du;

    u(k) = u(k-1) + du;

    if u(k) > u_max
        u(k) = u_max;
        dUp(1) = u(k) - u(k-1);
    end

    if u(k) < u_min
        u(k) = u_min;
        dUp(1) = u(k) - u(k-1);
    end

    U(k) = u(k) + Upp;
end

E = 0;
for k = 12:time
    E = E + e(k);
end
end
```

13. Dobór nastawów PID i DMC metodą eksperymentalną

13.1. Nastawy PID

W celu doboru nastawów regulatora PID skorzystano z metody Zieglera-Nicholsa. Oscylacje niegaszące otrzymano dla $K_{kryt} = 1,115$, z okresem $T_{kryt} = 37$. Wyliczając wartości parametrów jako $K = 0,6K_{kryt}$, $T_i = 0,5T_{kryt}$, $T_d = 0,12T_k$, a następnie przeliczając je na nastawy PID-u dyskretnego, rozpoczęta została regulacja, przedstawiona na rys. 15.

Modyfikując parametry regulatora przeprowadzone zostały jeszcze 3 eksperymenty przedstawione na rys. 16, rys. 17 i rys. 18.

Zmieniając wartości parametrów udało się poprawić regulację, zarówno zmniejszyła się wartość błędu, jak i wykresy wyglądają lepiej. Na tescie 3 i 4 PID działa zdecydowanie szybciej, dodatkowo oscylacje w teście 4 są zdecydowanie mniejsze.

13.2. Nastawy DMC

Dobór parametrów dla regulatora DMC odbywał się w podobny sposób. W pierwszej kolejności zostały wybrane parametry w sposób przypadkowy, następnie parametry były lekko zmieniane. Patrząc na wyniki symulacji można było wybrać najlepszy zestaw. Eksperymenty DMC zostały przedstawione na rys. ?? - ??.

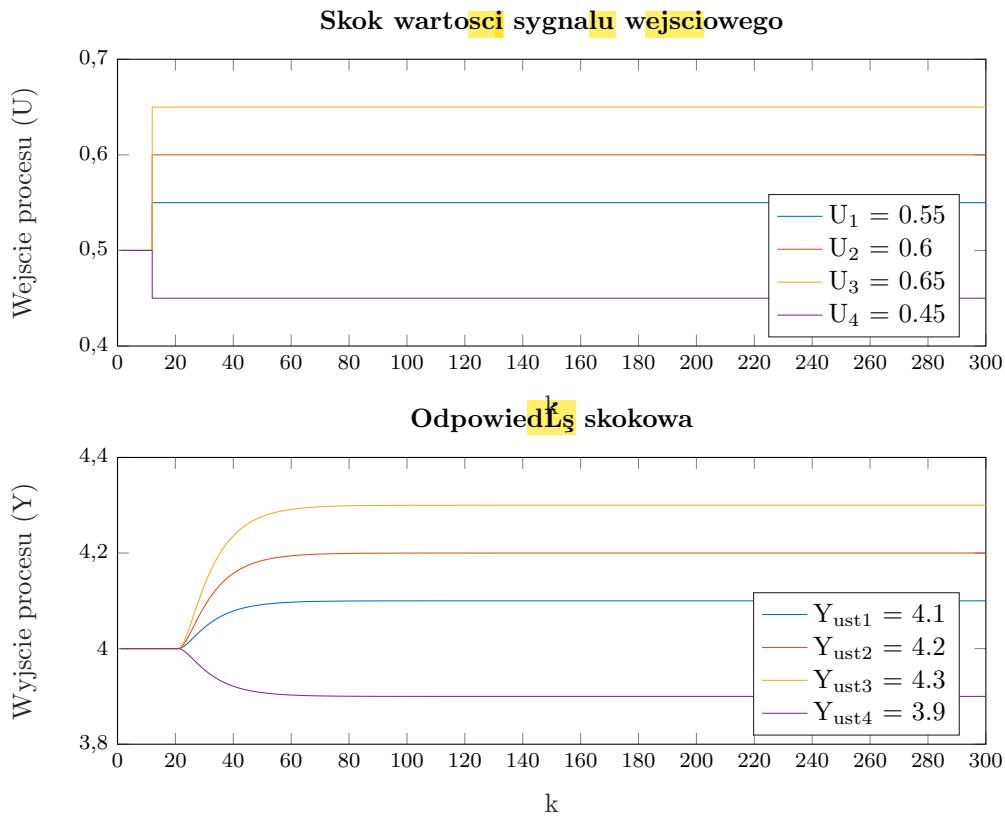
14. Dobór parametrów PID i DMC automatycznie

W celu optymalizacji wskaźnika błędu w zależności od parametrów obu regulatorów został użyty algorytm generyczny, którego zadaniem było znalezienie minimum funkcji błędów.

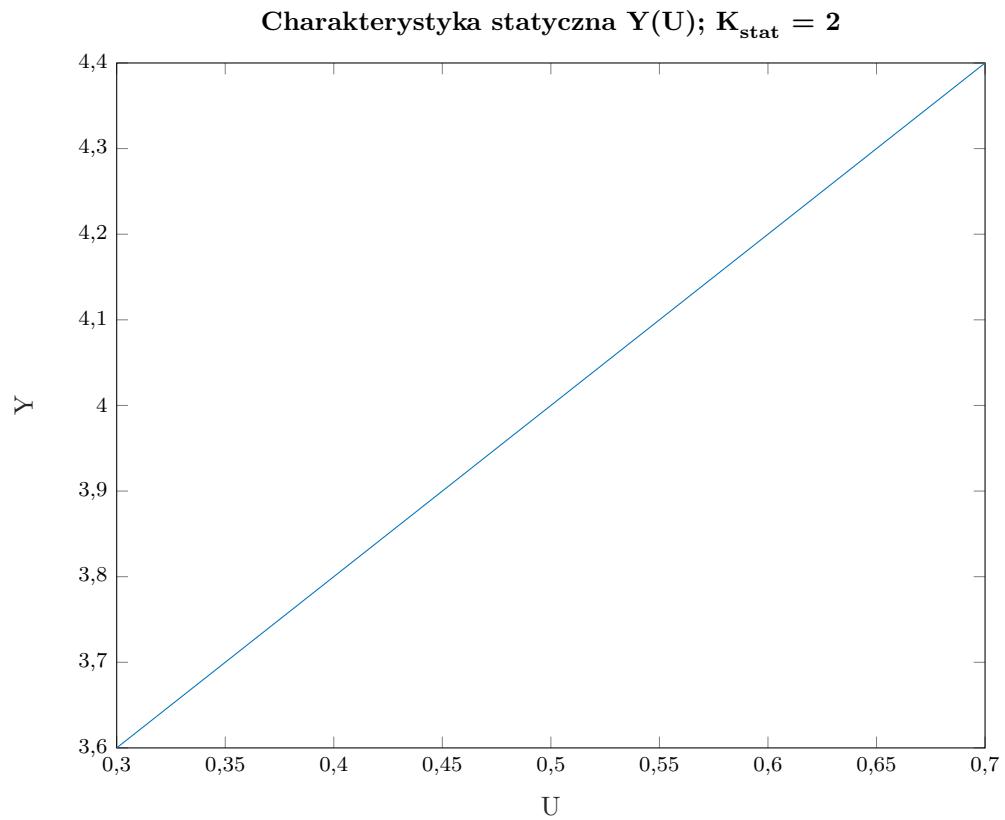
Uzyskane nastawy to:

$$\begin{aligned} \text{dla PID : } & K = 0,8706, T_i = 5,6252, T_d = 2,9545, E = 3,3250 \\ \text{dla DMC : } & N = 19, N_u = 30, \lambda = 1,9185, E = 2,4421 \end{aligned} \quad (7)$$

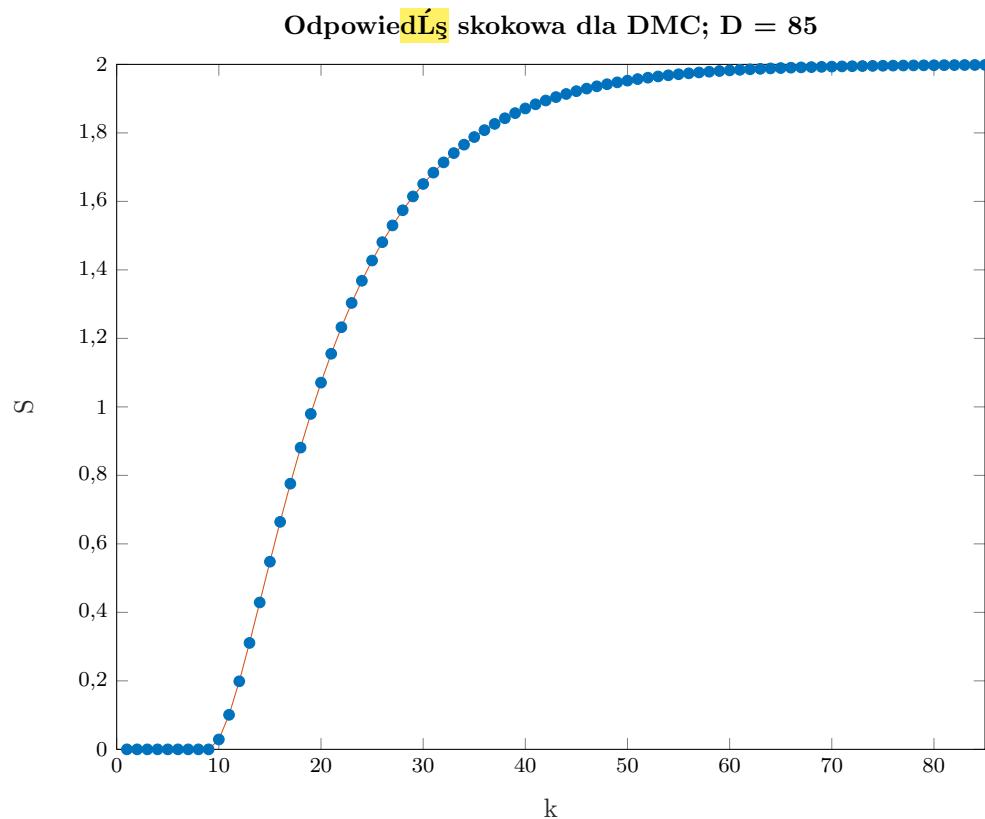
Otrzymane nastawy nie różnią się znacząco od tych uzyskanych metodą eksperymentalną, niemniej jednak błędy regulacji są mniejsze. Wykresy regulacji dla tych nastawów są przedstawione na rys. ?? i ??.



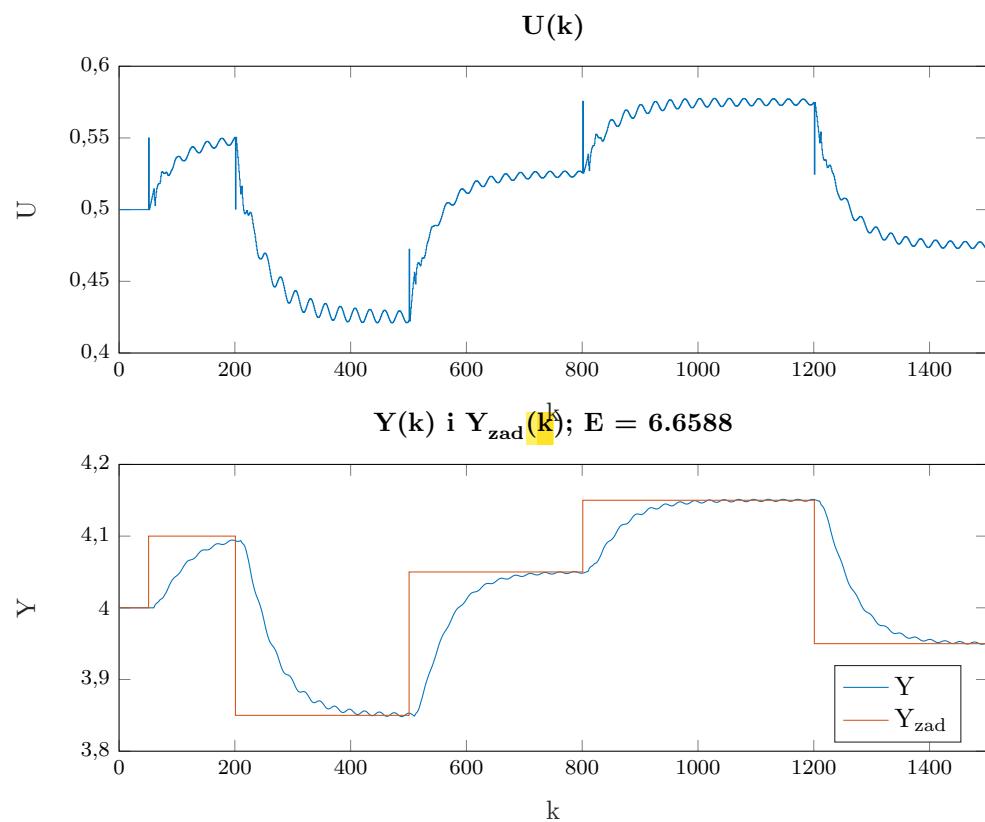
Rys. 12. Odpowiedzi skokowe procesu

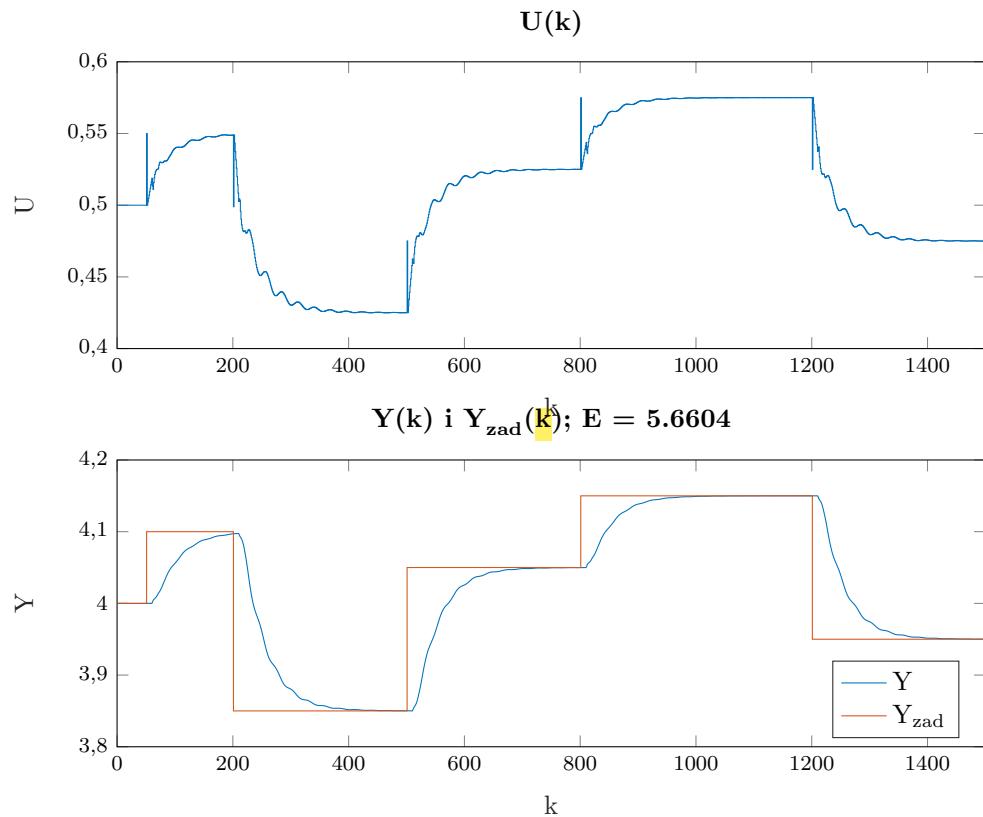
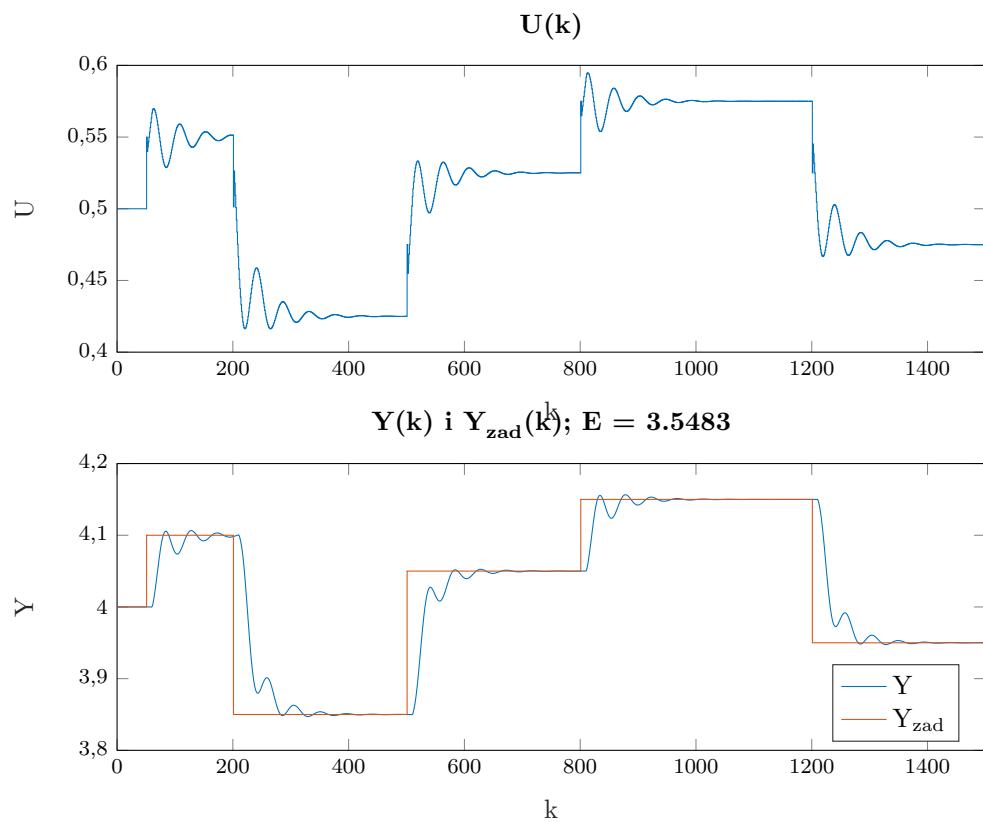


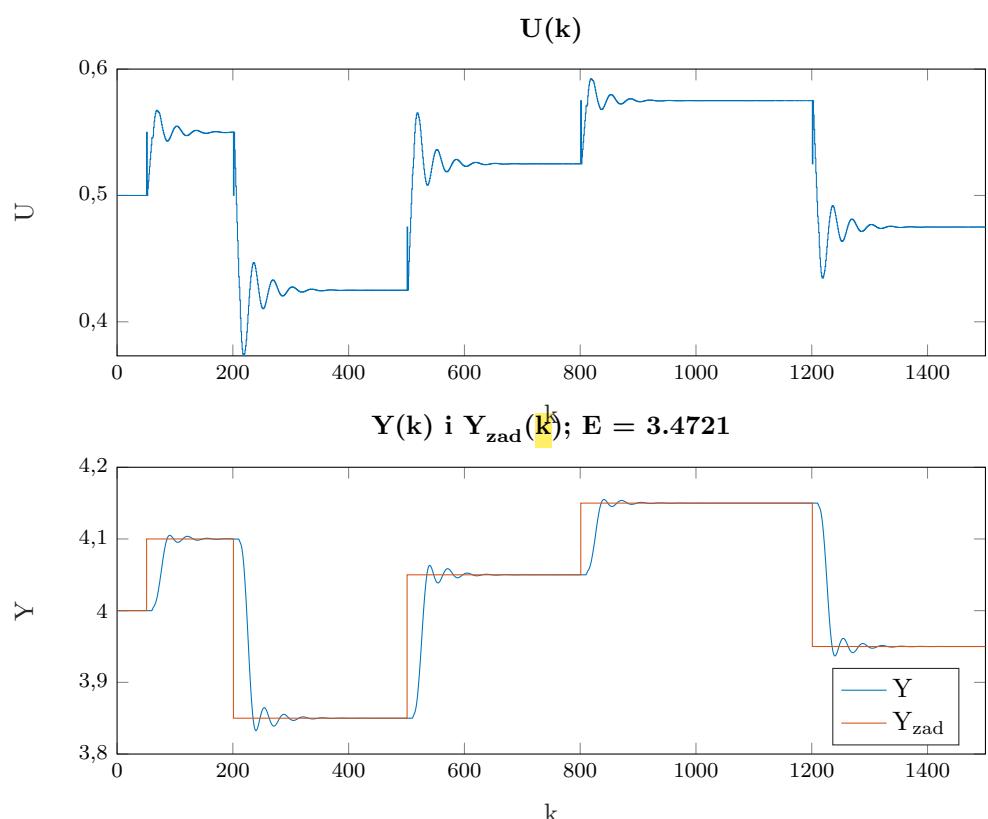
Rys. 13. Charakterystyka statyczna, wyznaczanie wzmacnienia



Rys. 14. Przekształcona odpowiedź skokowa

Rys. 15. Regulator PID - test1, $K = 0,669, T_i = 18,5, T_d = 4,44$

Rys. 16. Regulator PID - test2, $K = 0,75, T_i = 16, T_d = 3$ Rys. 17. Regulator PID - test3, $K = 0,67, T_i = 10, T_d = 0,1$

Rys. 18. Regulator PID - test4, $K = 0,8, T_i = 7, T_d = 2$