Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Projektowanie układów sterowania (projekt grupowy)

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

Autorzy:
Michał Kędzierski, Jakub Niewiński, Mariusz Słapek

Prowadzący:
mgr inż. Andrzej Wojtulewicz

Warszawa, 2019

Spis treści

I. Projekt

II. Laboratoria

Część I

Projekt

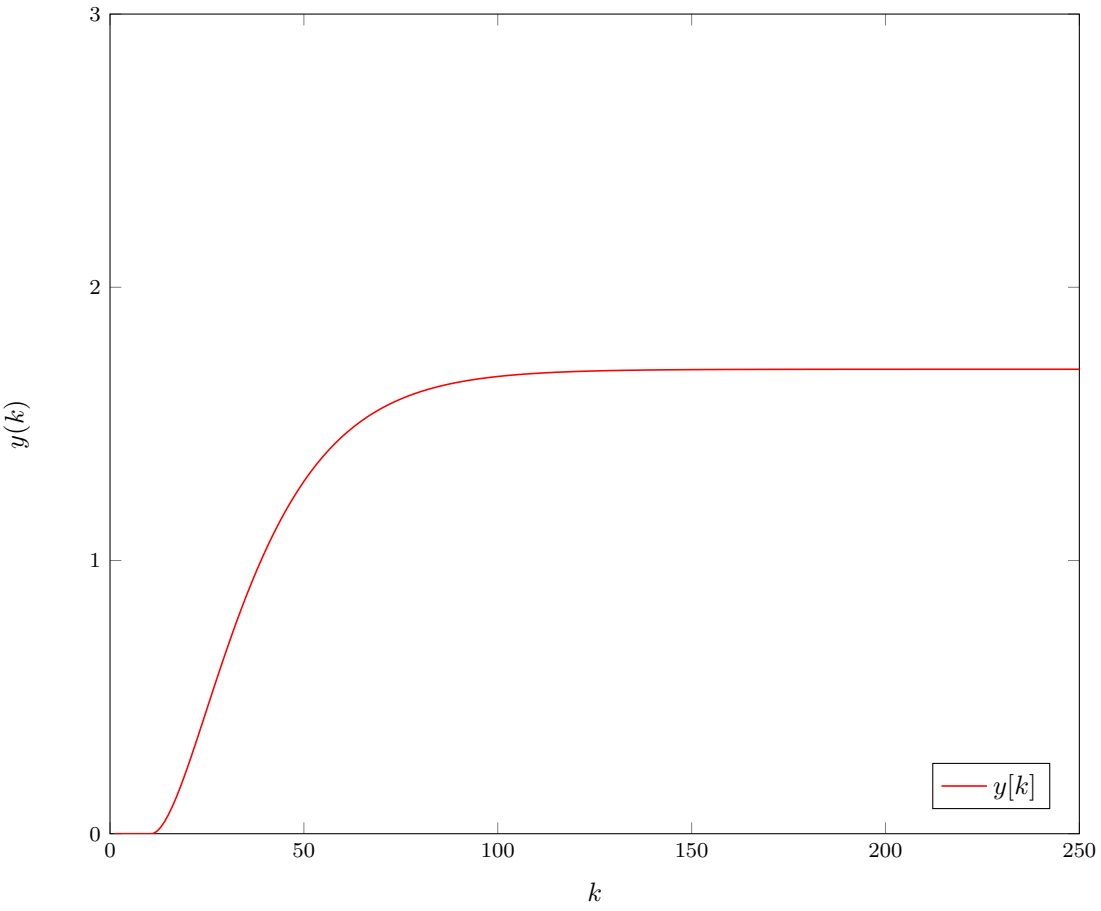
1. Sprawdzenie poprawność wartości U_{pp}, Y_{pp}

1.1. Opis

W celu sprawdzenia poprawności wartości sygnałów U_{pp}, Y_{pp} pobudziliśmy wejście obiektu sygnałem o stałej wartości - U_{pp}. Po przeprowadzenie eksperymentu sprawdzaliśmy czy sygnał wyjściowy ustabilizuje się na wartości Y_{pp}. Symulacja została przeprowadzona za pomocą dostarczonej funkcji `symulacja_obiektu8Y`. W naszym eksperymencie dla $U_{\text{pp}} = 1{,}0$ powinniśmy otrzymać wartość $Y_{\text{pp}} = 1{,}7$.

1.2. Eksperyment

W celu przeprowadzenia eksperymentu stworzyliśmy następujący skrypt `zad1P.m`. Odpowiedź skokowa, którą zebraliśmy została przedstawiona na rysunku 1.2. Jak widzimy obiekt przy stałym pobudzeniu $U_{\text{pp}} = 1{,}0$ po ustabilizowaniu ma wartość $Y_{\text{pp}} = 1{,}7$.

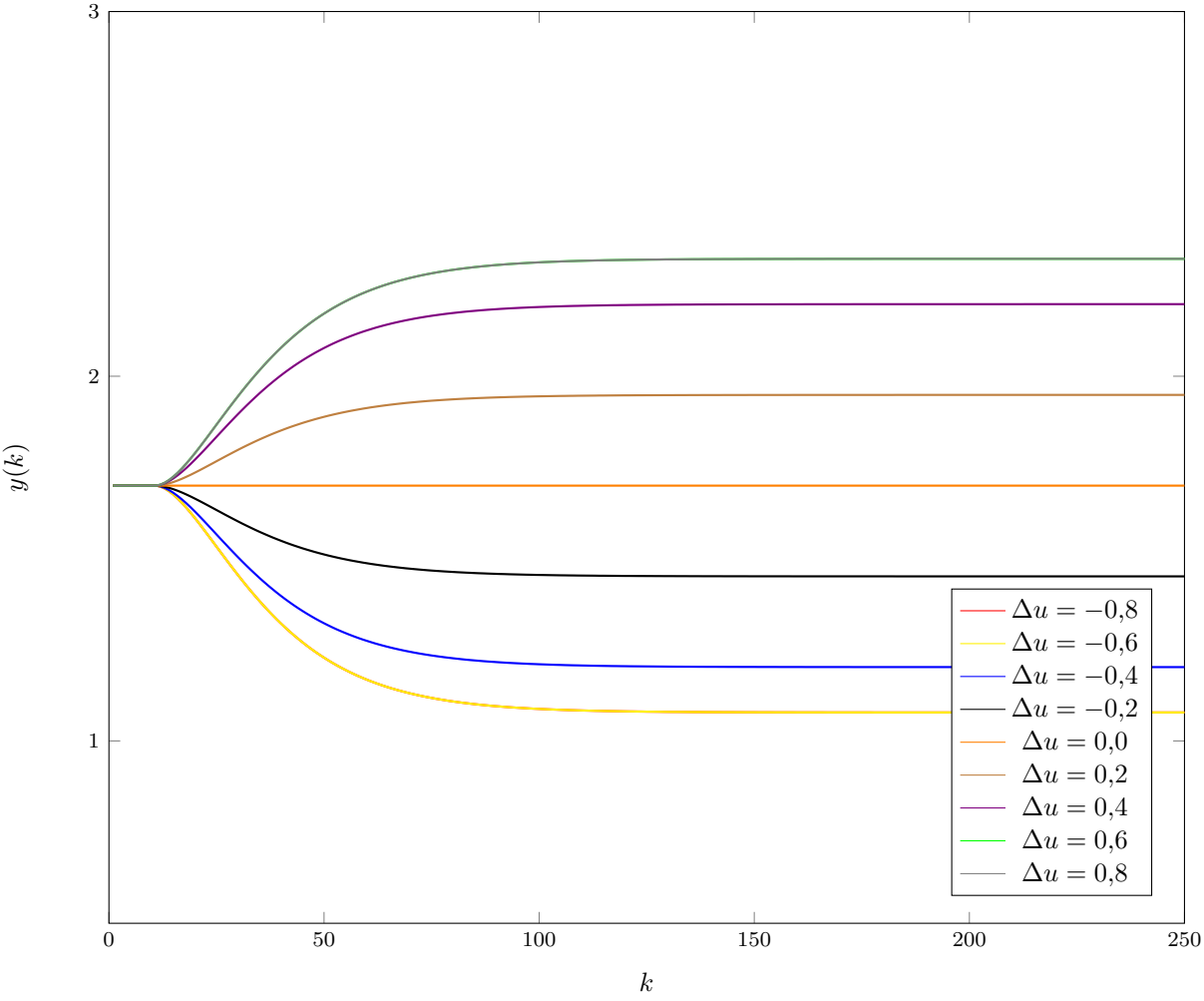


Rys. 1.1. Odpowiedź obiektu na stałe wejściowe o wartości $U_{\text{pp}} = 1{,}0$

2. Wyznaczenie symulacyjne odpowiedzi skokowej

2.1. Odpowiedzi skokowe procesu

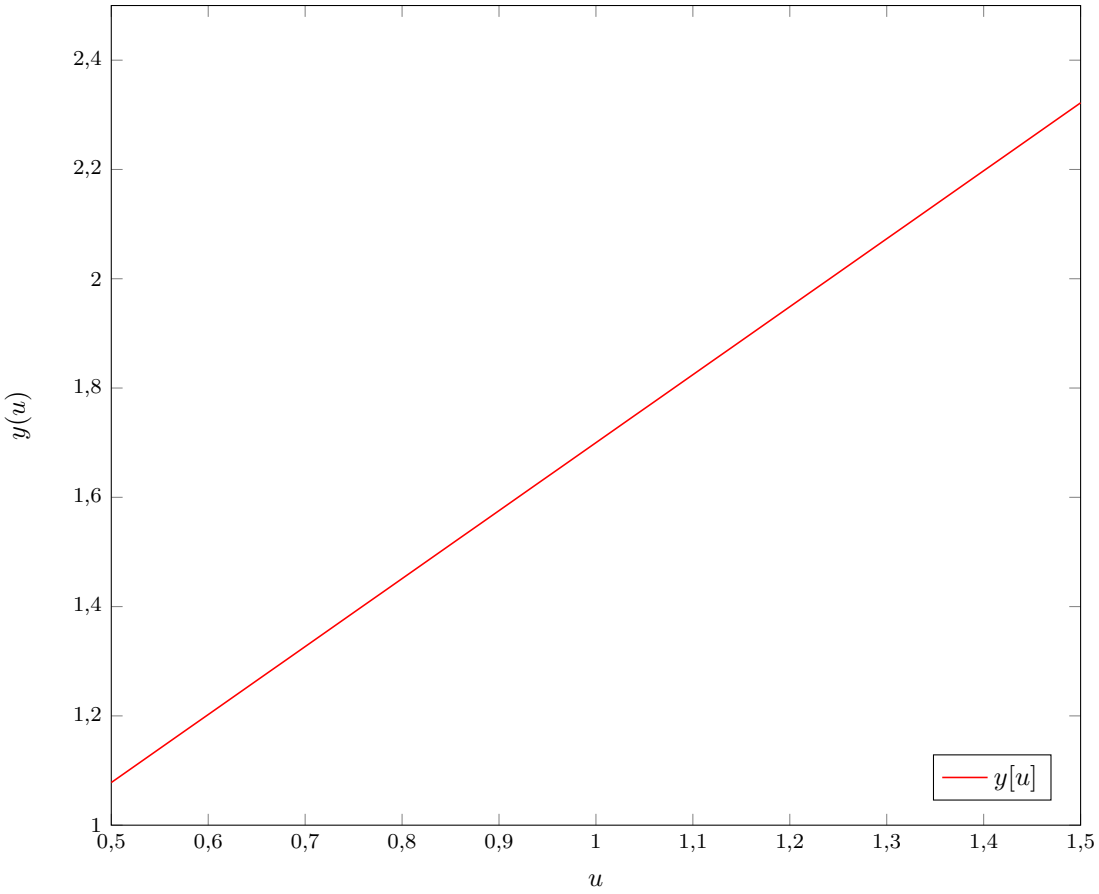
Z wykresu 2.1 wnioskujemy, iż wzrost (spadek) wartości sterującej powoduje wzrost (spadek) wyjścia procesu. Na podstawie tego wykresu można zatem zaobserwować, iż obiekt nasz jest prawdopodobnie liniowy (sprawdzimy to w następnym punkcie dokładniej). Przy rozwiązaniu tego zadania wykorzystałem skrypt `zad2P_a.m`.



Rys. 2.1. Odpowiedź obiektu na zmianę sygnału sterującego (wejścia obiektu)

2.2. Charakterystyka statyczna procesu

Aby określić czy obiekt jest linowy musieliśmy przeprowadzić symulację za pomocą skryptu `zad2P_b.m` oraz na tej podstawie stworzyć charakterystykę statyczną procesu. Zadając różne wartości sygnału wejścia (sygnału sterującego) otrzymywaliśmy różne wartości wyjściowe. Na tej podstawie otrzymaliśmy rysunek 2.2.



Rys. 2.2. Charakterystyka statyczna obiektu

2.3. Wzmocnienie statyczne procesu

Wzmocnienie statyczne procesu możemy obliczyć korzystając ze wzoru:

$$K_{\text{stat}} = \lim_{t \to \infty} \frac{y(t) - Y_{\text{pp}}}{u - U_{\text{pp}}} \tag{2.1}$$

Jak widzimy ze wzoru parametr ten jest granicą ilorazu wyjścia obiektu względem punktu pracy oraz wejścia obiektu względem punktu pracy.

Zatem mając charakterystykę statyczną procesu wystarczy wyznaczyć kąt nachylenia wykresu. Tangens tego kąta będzie naszym wzmocnieniem statycznym.

$$K_{\text{stat}} = \operatorname{tg} \alpha \tag{2.2}$$

W naszym przypadku: $\operatorname{tg}(\alpha) = 1{,}2$, zatem wzmocnienie statyczne wynosi: $K_{\text{stat}} = 1{,}2$.

3. Przekształcenie odpowiedzi skokowej

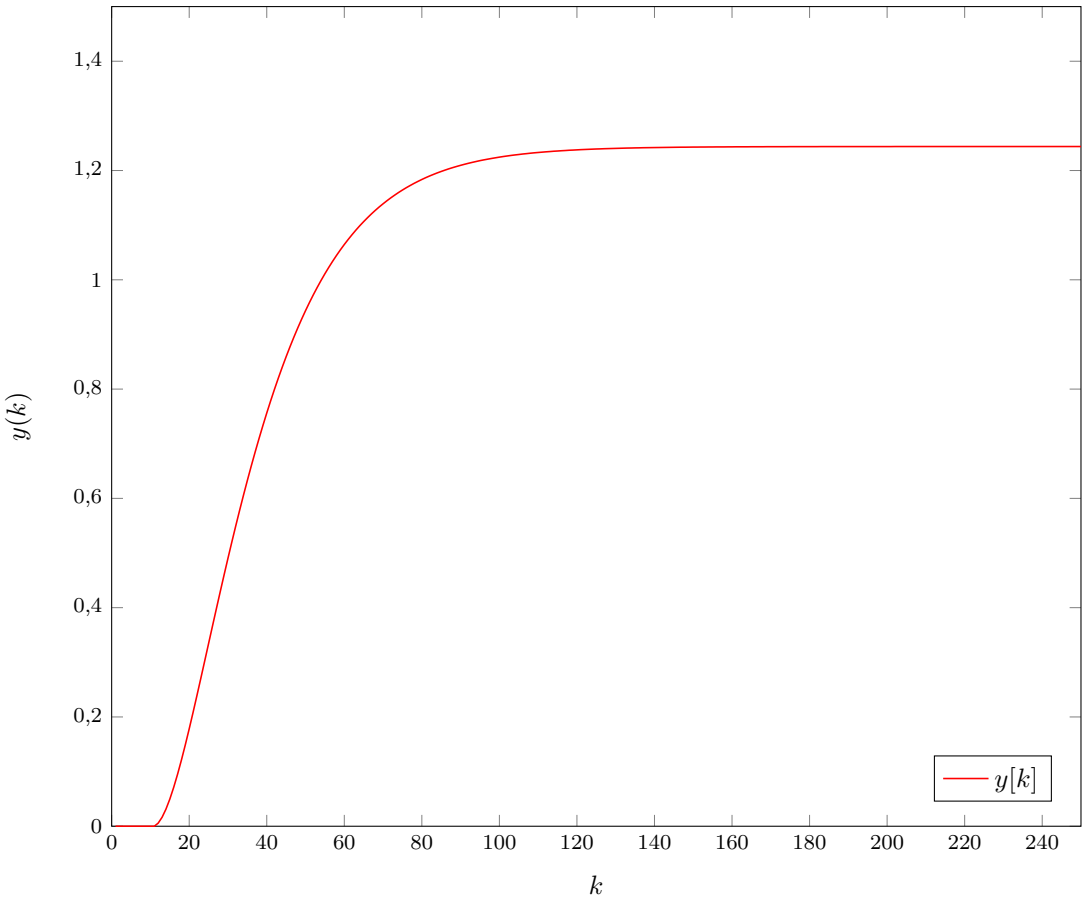
3.1. Normalizacja odpowiedzi skokowej

Aby znormalizować odpowiedź skokową posłużyliśmy się wzorem przedstawionym na ćwiczeniach:

$$s_i = \frac{s_i^0 - Y_{\text{pp}}}{\Delta U}, \text{ dla } i = 1, \ldots \tag{3.1}$$

Aby wyznaczyć znormalizowaną odpowiedź skokową wykorzystaliśmy stworzony przez nas skrypt `zad3P.m`. W naszej symulacji wykonaliśmy skok sterowania o wartość $\Delta U = 0{,}5$.

Otrzymana znormalizowana odpowiedź skokowa przedstawiona jest na rysunku 3.1.

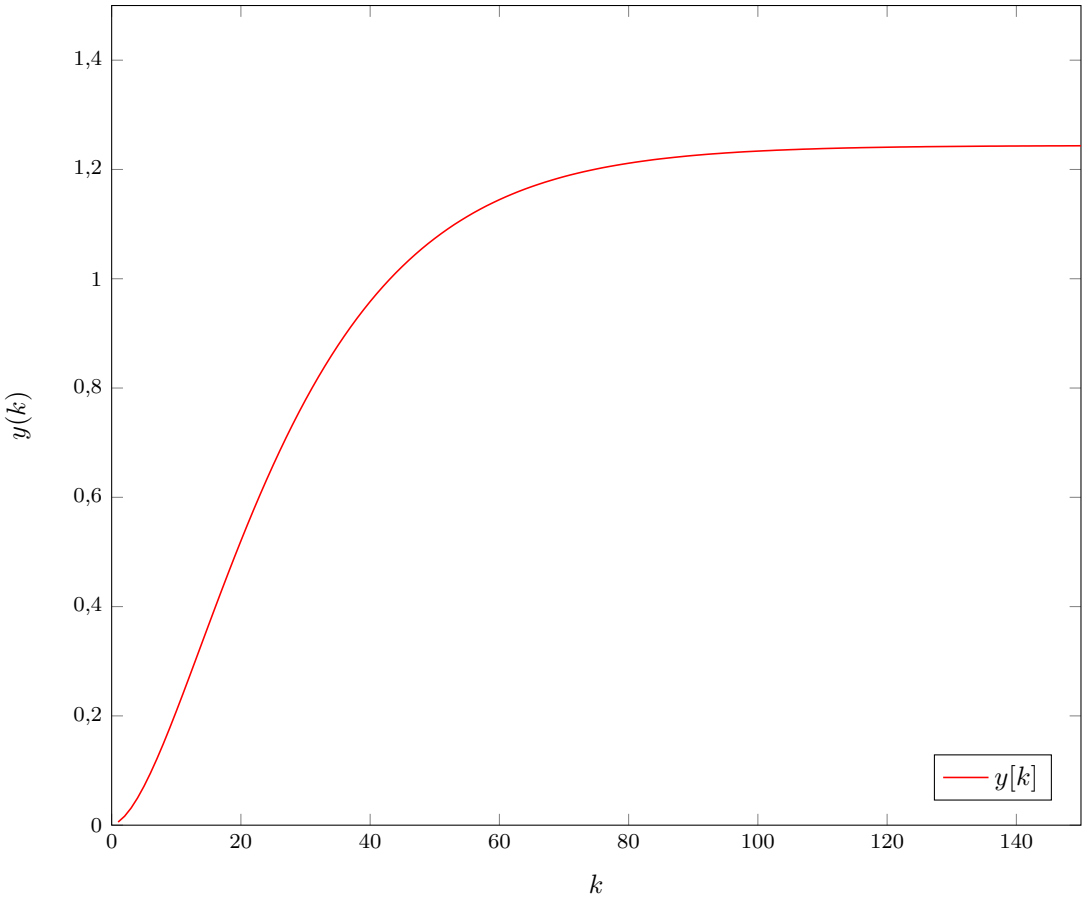


Rys. 3.1. Znormalizowana odpowiedź skokowa

3.2. Wybór horyzontu dynamiki

Analizując współczynniki odpowiedzi skokowej doszliśmy do wniosku, iż wybierzemy 150 współczynników. Wynika to z faktu, iż następne współczynniki nie poprawiają nam jakości regulacji.

Ostateczna nasza odpowiedź skokowa jest przedstawiona na rysunku 3.2.



Rys. 3.2. Znormalizowana odpowiedź skokowa obcięta do horyzontu dynamiki

4. Algorytm PID

4.1. Ogólny zarys algorytmu

Jest to regulator proporcjonalno-całkująco-różniczkujący. Regulator PID pracuje w pętli sprzężenia zwrotnego, oblicza wartość uchybu jako różnicę pomiędzy pożądaną wartością zadaną i zmierzoną wartością zmiennej procesu i działa w taki sposób, by zredukować uchyb poprzez odpowiednie dostosowanie sygnału podawanego na wejście regulowanego obiektu. Algorytm obliczeń regulatora PID zawiera trzy oddzielne stałe parametry.
Poglądowo działanie tych członów w odniesieniu do czasu można zinterpretować następująco:

— Działanie członu P kompensuje uchyb bieżący,
— Człon I kompensuje akumulację uchybów z przeszłości,
— Człon D kompensuje przewidywane uchyby w przyszłości.

W dziedzinie dyskretnej wyraża się wzorem:

$$u(t) = r_2 * e(k-2) + r_1 * e(k-1) + r_0 * e(k) + u(k-1) \tag{4.1}$$

gdzie:

$$r_0 = K(1 + \frac{T}{2T_\mathrm{i}} + \frac{T_\mathrm{d}}{T}) \tag{4.2}$$

$$r_1 = K(\frac{T}{2T_\mathrm{i}} - \frac{2T_\mathrm{d}}{T} - 1) \tag{4.3}$$

$$r_2 = \frac{KT_\mathrm{d}}{T} \tag{4.4}$$

4.2. Implementacja algorytmu PID

Naszą implementację algorytmu PID można podzielić na dwa etapy. Pierwszym etapem była deklaracja wszystkich potrzebnych zmiennych wykorzystywanych podczas eksperymentu. Drugim było opisanie głównej pętli symulacyjnej cyfrowego algorytmu PID. W naszym zespole powstały dwie koncepcje - dwie różne implementacje algorytmu PID. Poniżej zostanie przedstawiona jedna z nich.

4.2.1. Deklaracja zmiennych

W naszym skrypcie musieliśmy zdefiniować stałe - podane w poleceniu wartości w punkcie pracy sygnału wyjściowego i sterującego (U_pp, Y_pp). Następnie zdefiniowaliśmy jak długo nasza symulacja powinna przebiegać oraz od jakiego momentu powinna się zaczynać. Następnie zdefiniowaliśmy maksymalne wartości sygnału sterującego oraz masksymalnych zmian sygnału sterującego.

Listing 4.1. Deklaracja stałych

```
%deklaracja punktu pracy
Ypp = 36.06;
Upp = 33;

% deklaracja czasu symluacji procesu
simulationTime = 500;
% deklaracja czasu dyskretnego rozpoczecia symluacji
start = 2;

% deklaracja maksymalnej wartosci zmian sygnalu sterujacego
dU = 0.1;
% deklaracja maksymalnej wartosci sygnalu sterujacego
uMin = 0.4;
uMax = 1.0;
```

Musieliśmy zdefiniować ponadto wszystkie zmienne występujące w symulacji. Były to parametry do regulatora PID. Wzory na współczynniki regulatora PID, deklaracje błędów oraz deklaracja wartości zadanej. Wektory Y oraz U przechowują wartości sygnałów wejściowych oraz wyjściowych.

Listing 4.2. Deklaracja zmiennych

```
% deklaracja parametrow do regulatora PID
K = 12;
Ti = 15;
Td = 15;
T = 1;

% wzory na wspolczynniki regulatora PID
r0 = K * (1 + T/(2*Ti) + Td/T);
r1 = K * (T/(2*Ti) - 2*Td/T - 1);
r2 = K * Td/T;

% deklaracja bledow
errorR0 = 0;
errorR1 = 0;
errorR2 = 0;

% deklaracja macierzy wyjscia oraz wejscia
Y = ones(simulationTime,1)*Ypp;
y = zeros(simulationTime,1);
U = ones(simulationTime,1)*Upp;
u = zeros(simulationTime,1);

% deklaracja wartosci zadanej

yZad(1:250) = Ypp;
yZad(251:simulationTime)= Ypp + dy;
```

4.2.2. Główna pętla symulacyjna

Ostatnim etapem implementacji było utworzenie pętli, w której odbywa się symulacja. Nasza pętla rozpoczynała się od wyliczenia wyjścia obiektu. Następnie musieliśmy przeskalować wyjście obiektu oraz wartości zadanej. Następnie wyliczany był wykorzystywany przy wyliczaniu sterowania. Wyliczana wartość była przeskalowana. Następnie po sprawdzeniu wszystkich ograniczeń jedna iteracja pętli kończyła się.

Listing 4.3. Algorytm PID zaimplementowany w programie MATLAB

```
for k = start : 1 : simulationTime

        % symulacja
        Y(k) = symulacja_obiektu8Y(U(k-10),U(k-11),Y(k-1),Y(k-2));

        % skalowanie wyjsca
        y(k-1) = Y(k-1) - Ypp;

        % skalowanie wartosci zadanej
        yzad(k-1) = yZad(k-1)- Ypp;

        % obliczenie bledu
        errorR0 = yzad(k) - y(k);

        % PID rownanie
        u(k) = u(k-1) + r0 * errorR0 + r1 * errorR1 + r2 * errorR2;

        % ograniczenia
        U(k)=u(k)+Upp;

        du = U(k) - U(k-1);
        if du>= dU
            du=dU;
        end
        if du<=-dU
            du=-dU;
        end

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

        UMax = 2;
        UMin = 1;

        if U(k)> uMax
            U(k) = uMax;
        end
        if U(k)< uMin
            U(k) = uMin;
        end

        % wskaznik jakosci regulacji
        E = (yZad - Y).^2;
        e = sum(E(:, 1));
```

```
        % aktualizacja bledow
        errorR2 = errorR1;
        errorR1 = errorR0;
end
```

5. Algorytm DMC

5.1. Ogólny zarys algorytmu

Jest to regulator predykcyjny, czyli reaguje nie tylko na sprzężenie zwrotne, ale także dostosowuje swoje działanie z wyprzedzeniem, zanim nastąpią zmiany wielkości wyjściowych układu. Opisany jest równaniem:

$$\Delta U(k) = K * (Y^{\mathrm{zad}}(k) - Y(k) - M^p * \Delta u(k)) \tag{5.1}$$

Gdzie K jest macierzą utworzoną poprzez następujące operacje na macierzach:

$$K = (M^{\mathrm{T}} * M + \lambda * I) * M^{\mathrm{T}}$$

Parametr λ jest "karą" za zmianę sterowania, zwiększenie tego parametru zmniejsza *du* wyliczane w kolejnych chwilach, wygładzając charakterystykę jednak też spowalniając sterowanie.

$$M = \begin{bmatrix} s_1 & 0 & 0 & \dots & 0 \\ s_2 & s_1 & 0 & \dots & 0 \\ s_3 & s_2 & s_1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{\mathrm{N}} & s_{\mathrm{N}-1} & s_{\mathrm{N}-2} & \dots & s_{\mathrm{N}-\mathrm{N_u}+1} \end{bmatrix} \tag{5.2}$$

$$M^{\mathrm{P}} = \begin{bmatrix} s_2 - s_1 & s_3 - s_2 & \dots & s_{\mathrm{D}} - s_{\mathrm{D}-1} \\ s_3 - s_1 & s_4 - s_2 & \dots & s_{\mathrm{D}+1} - s_{\mathrm{D}-1} \\ \vdots & \vdots & \ddots & \vdots \\ s_{\mathrm{N}+1} - s_1 & s_{\mathrm{N}+2} - s_2 & \dots & s_{\mathrm{N}+\mathrm{D}-1} - s_{\mathrm{D}-1} \end{bmatrix} \tag{5.3}$$

$$\Delta U^{\mathrm{p}}(k) = \begin{bmatrix} u(k-1) - u(k-2) \\ u(k-2) - u(k-3) \\ \vdots \\ u(k-D+1) - u(k-D) \end{bmatrix} \tag{5.4}$$

5.2. Implementacja algorytmu DMC

Wykorzystanie algorytmu DMC wymaga pobrania modelu obiektu w postaci odpowiedzi skokowej, która została wykonana w wcześniejszym zadaniu. Początkowe wartości horyzontów zostały wyznaczone na podstawie wizualnej oceny wykresu rzędnych odpowiedzi skokowej na 150, a parametr λ roboczo został ustawiony na 1.

5.2.1. Deklaracja zmiennych

Implementację algorytmu DMC należy zacząć od zadeklarowania wartości horyzontów. Należy pamiętać, by nie przekraczać nimi długości pobranych rzędnych odpowiedzi skokowej - takie postępowanie nie polepszy jakości sterowania, jednak zwiększy czas potrzebny na obliczenie każdego sterowania.

Listing 5.1. Deklaracja horyzontów oraz obliczenie struktury DMCStruct

```
% Wybieramy regultor DMC
Regulator = 'DMC';

if strcmp(Regulator,'DMC')
    D = 100;
    N = 50;
    Nu =60;
    lambda = 30;
    duPop = zeros(D-1,1)';
    % generacja macierzy Mp i K o powyzszych parametrach
    DMCStruct = zad4P_dmcGeneration(s,D,Nu,N,lambda);
end
```

5.2.2. Wyznaczenie odpowiednich macierzy

Funkcja `dmcGeneration` zwracająca strukturę, posiadającą wszystkie dane na temat regulatora DMC, który tworzy.

Listing 5.2. Zawartość powyższej funkcji

```
function [DMCMacierze] = zad4P_dmcGeneration(s,D,Nu,N,lambda)

    % zwracana jest struktura DMCMacierze z nastepujacymi polami:
    % .M -> macierz M
    % .Mp -> macierz Mp
    % .K -> macierz K
    % .N -> wartosc N dla ktorej byly generowane
    % .Nu -> wartosc Nu dla ktorej byly generowane
    % .D -> wartosc D dla ktorej byly generowane

    M=zeros(N,Nu);
    Mp=zeros(N,D-1);

    for i=1:N
        for j=1:Nu
            if i>=j
                %oblicznie M
                M(i,j)=s(i-j+1);
            end
        end
    end

    for i=1:N
        for j=1:D-1
            if i+j<D
                %oblicznie Mp
                Mp(i,j)=s(i+j)-s(j);
            else
                Mp(i,j)=s(D)-s(j);
            end
        end
```

```
    end

    K = ((M'*M + lambda*eye(Nu))^(-1))*M';
    DMCMacierze.Mp = Mp;
    DMCMacierze.K = K;
    DMCMacierze.N = N;
    DMCMacierze.M = M;
    DMCMacierze.Nu = Nu;
    DMCMacierze.D = D;
end
```

5.2.3. Główna pętla symulacji

Poniższy listing przedstaiwa główną pętle symulacji. Komentarze są zawarte w kodzie.

Listing 5.3. Zawartość głównej pętli sterowania odpowiedzialna za symulację działania regulatora DMC

```
for i = delay:lenght
    y(i) = symulacja_obiektu8Y(u(i-10)+Upp...
        ,u(i-11)+Upp,y(i-1)+Ypp,y(i-2)+Ypp)-Ypp;

    if strcmp(Regulator,'DMC')
        du(i) = zad4P_dmc(DMCStruct,y(i),yZad(i),duPop);
        % uzupelnienie macierzy duPop o najnowsza wartosc oraz
        % usuniecie najstarszej
        duPop = zad4P_duNew(duPop,du(i));
    end

    if du(i) > duMax
        du(i) = duMax;
    end
    if du(i) < -duMax
        du(i) = -duMax;
    end
    u(i) = u(i-1) + du(i);
    if u(i) > uMax
        u(i) = uMax;
    end
    if u(i) < uMin
        u(i) = uMin;
    end
end
```

Funkcja `zad4P_dmc` wykonuje obliczenia analitycznej wersji regulatora DMC, zwracając tylko $d_{\mathrm{u}}(k|k)$.

Listing 5.4. Kod funkcji zwracającej

```
function [du] = zad4P_dmc(dmcMacierze,yAktualne,yZadane, duPop)
    % obliczenie zmiany sterowania du w danej chwili.
    % Metoda analityczna,
    % zwraca tylko du(k|k)
```

```
    Mp = dmcMacierze.Mp;
    N = dmcMacierze.N;
    K = dmcMacierze.K;
    yk = ones(N,1) * yAktualne;
    y0 = yk + Mp * duPop';
    du = K * ( yZadane - y0);
    du = du(1,1);
end
```

6. Strojenie regulatora PID

6.1. Metody doboru nastawów regulatora PID

Skrypty wykorzystywane do zrealizowania tego zadania: `zad4P_pid.m`, `zad5P_PID.m`, `zad4P_script.m`.

6.1.1. Metoda Zieglera-Nicholsa

Metoda oparta jest o pomiar parametrów oscylacji. Wzmocnienie **K** zwiększa się do czasu, aż osiągnie się ostatecznie wzmocnienie K_{kryt}, przy którym sygnał wyjściowy pętli zacznie oscylować ze stałą amplitudą. K_{kryt} i okres oscylacji T wykorzystuje się następnie, by ustawić wzmocnienia zgodnie z poniższą tabelą 6.1:

Typ regulacji	K_{p}	K_{i}	K_{d}
P	$0{,}50 * K_{\text{kryt}}$	—	—
PI	$0{,}45 * K_{\text{kryt}}$	$T_{\text{k}}/1{,}2$	—
PID	$0{,}60 * K_{\text{kryt}}$	$0{,}5 * T_{\text{k}}$	$0{,}125 * T_{\text{k}}$

Tab. 6.1. Tabela parametrów do metody Zieglera-Nicholsa

Wyznaczenie K_{kryt} oraz okresu oscylacji

Na podstawie wykresów może stwierdzić, iż oscylację występują dla $K_{\text{kryt}} = 3{,}0125$.

6.1.2. Metoda inżynierska

Dobór współczynnika K_{p} odbywa się identycznie jak w wyżej wymienionej metodzie Zieglera-Nicholsa. Natomiast współczynniki T_{i} i T_{d} dobierane są eksperymentalnie. Najpierw metodą prób i błędów dobieramy współczynnik T_{i}, aż otrzymamy zadowalający nas regulator PI, później w identyczny sposób dobieramy T_{d}.

6.2. Dobór parametrów regulatora PID

Dobory te miały miejsce poprzez manipulację mnożnikami odpowiednich parametrów. Wyjściowymi mnożnikami były nastawy Zieglera-Nicholsa.

6.2.1. Dobór parametru K

Rysunki 6.2 i 6.3 prezentują nastawy zgodne z Zieglerem-Nicholsem. Trajektoria jest ładna, bez uchybu ustalonego.

Rysunki 6.4 i 6.5 przedstawiają regulator po zwiększeniu mnożnika przy parametrze K do 0,8. Pomimo względnego pogorszenia trajektorii poprawie uległ wskaźnik jakości E.

Rysunki 6.6 i 6.7 to dalsze zwiększanie mnożnika przy K, do wartości 1,0. Mimo pojawienia się drgań w trajektorii wskaźnik jakości zmniejszył się.

Rysunki 6.8 i 6.9. Wskaźnik jakości E pogorszył się. Powrót do poprzedniego mnożnika przy K.

Jako optymalna wartość mnożnika przy parametrze K została wybrana wartość 1,0

6.2.2. Dobór parametru T_i

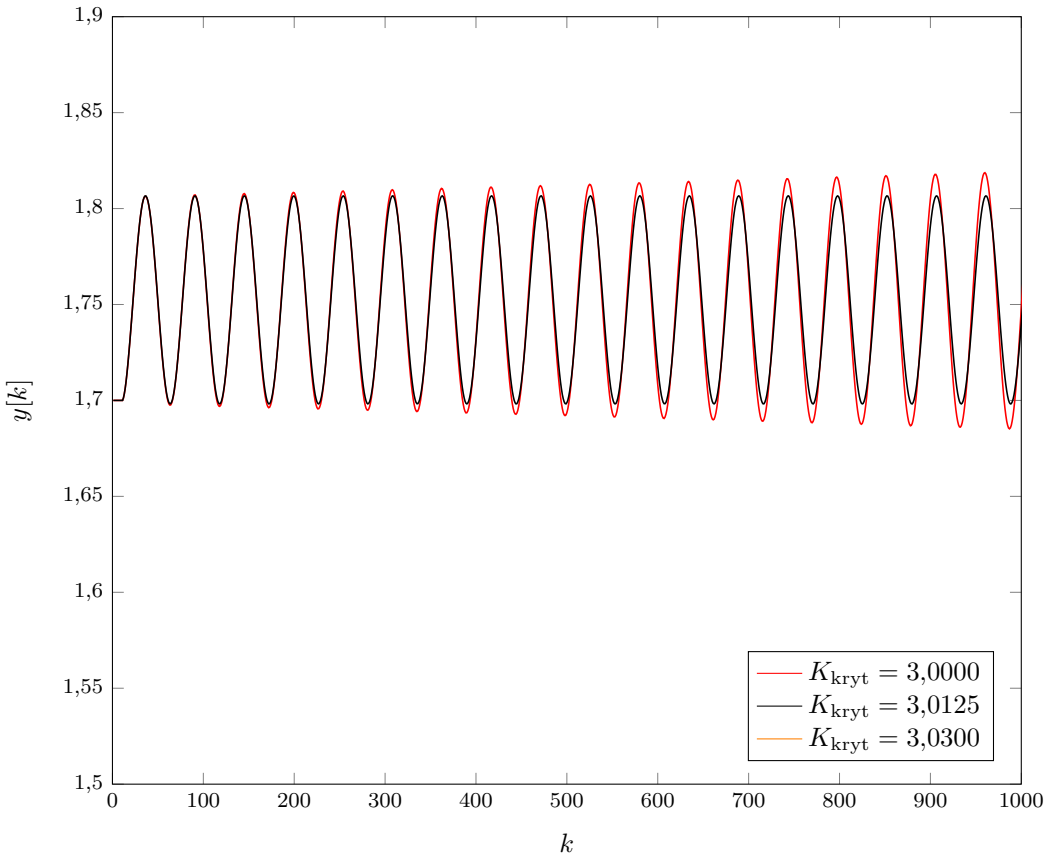
Zmniejszenie wpływu całkowania poprzez większenie mnożnika przy T_i na rysunkach 6.10 i 6.11 zwiększyło błąd E.

Zmniejszenie mnożnika przy T_i czyli zwiększenie wpływu całkowania na rysunkach 6.12 i 6.13 zmniejszyło błąd E, jednak zwiększeniu uległy oscylacje wyjścia. Po trajektorii sterowania widać, że te oscylacje gasną.

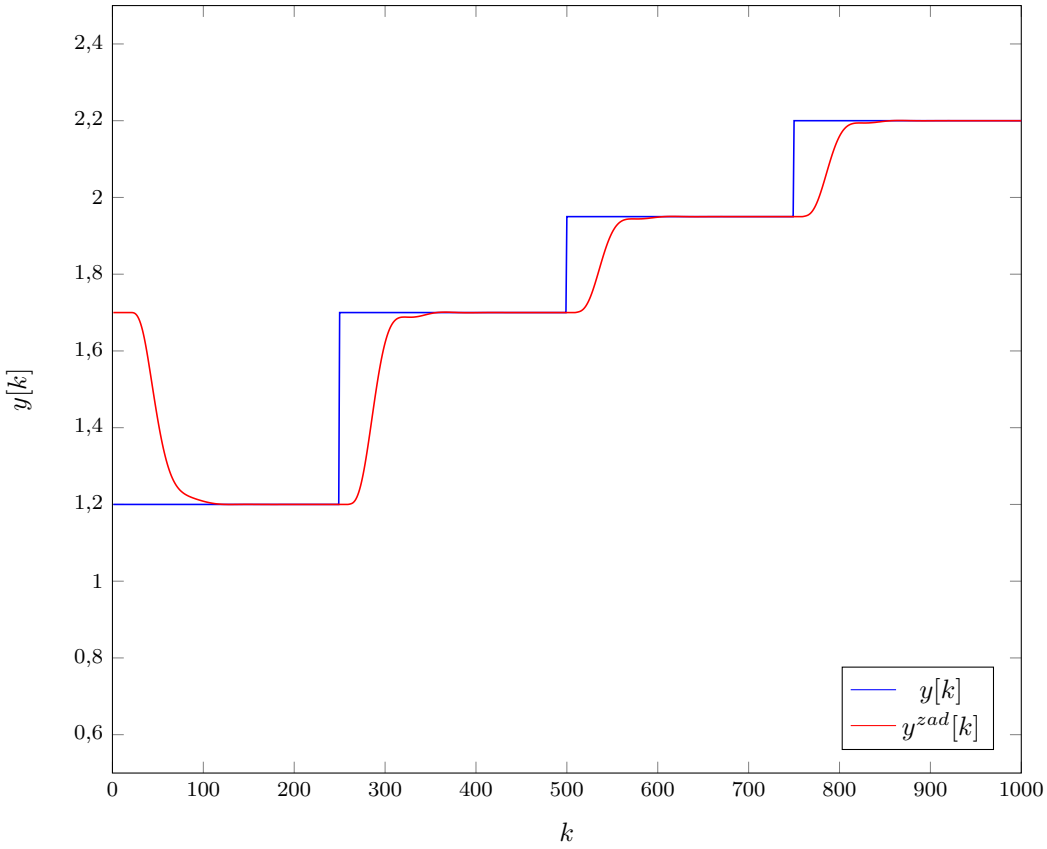
Zmniejszenie mnożnika przy T_i czyli dalsze zwiększenie wpływu całkowania na rysunkach 6.14 i 6.15 zmniejszyło błąd E. Oscylacje minimalnie się zmieniły - pierwsze przesterowanie jest silniejsze.

Zmniejszenie mnożnika przy T_i czyli dalsze zwiększenie wpływu całkowania na rysunkach 6.16 i 6.17 zwiększyło błąd E i wprowadziło obiekt w oscylacje niegasnące. Powrót do porzedniej wartości parametru T_i.

Optymalna wartość mnożnika przy parametrze T_i została ustalona na 0,3. Dla niej błąd E był najniższy.



Rys. 6.1. Wyjścia obiektu w zależności od parametru K



Rys. 6.2. Trajektoria wyjścia dla $K = 1{,}8075$, $T_\mathrm{i} = 13$, $T_\mathrm{d} = 3{,}25$, $E = 2{,}100\,90 \cdot 10^1$

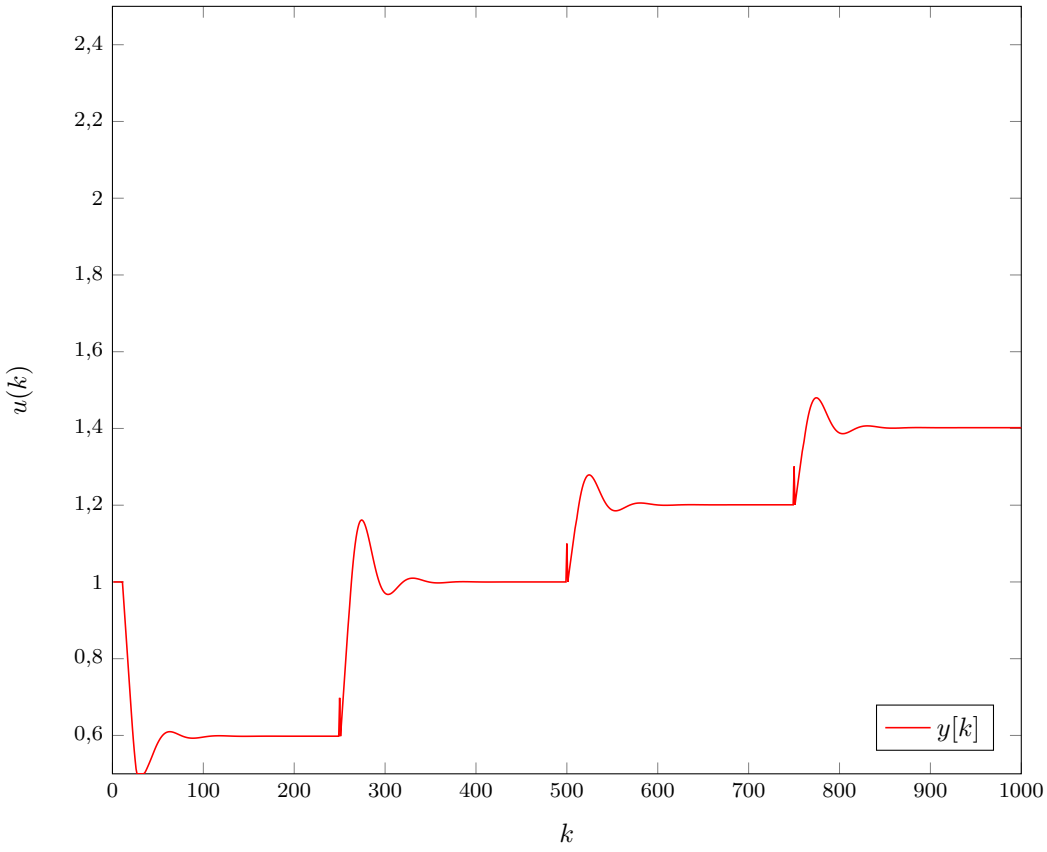
6.2.3. Dobór parametru T_d

Zmniejszenie wpływu różniczkowania poprzez zmniejszenie mnożnika przy T_d na 6.18 i 6.19 zwiększyło błąd E i wprowadziło obiekt w powoli gasnące (stwierdzenie spowodowane wyglądem trajektorii sterowania) drgania.

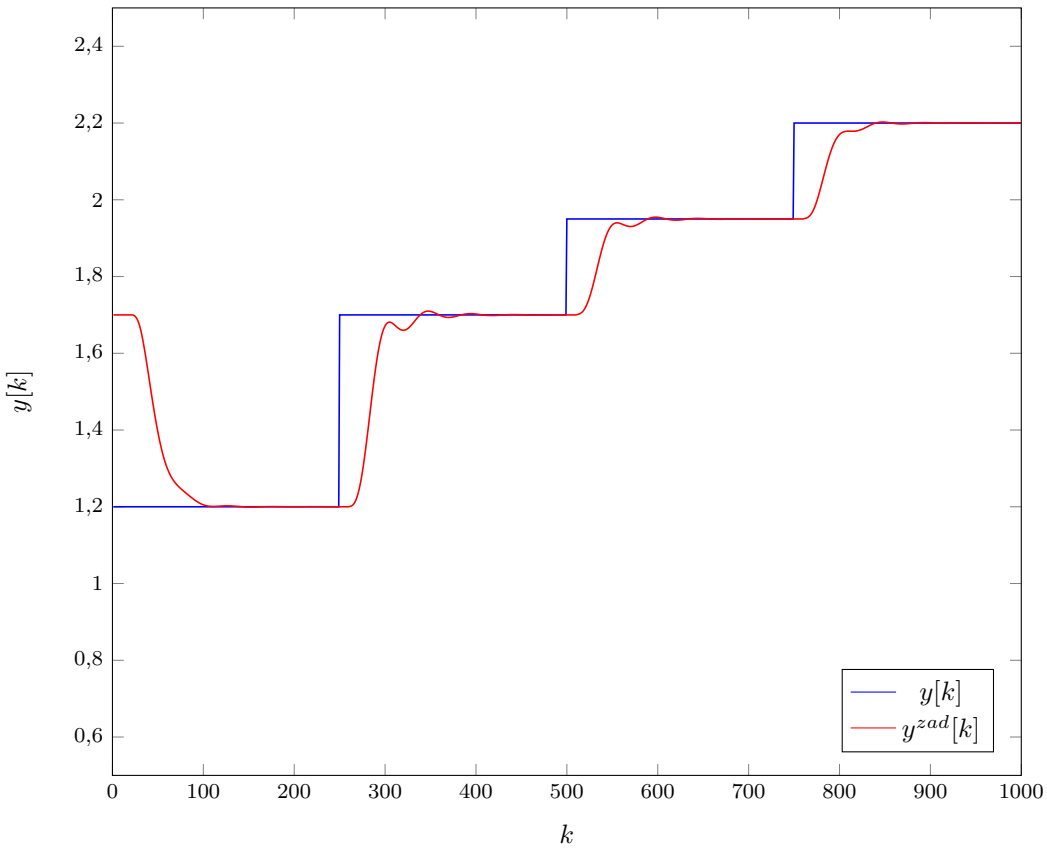
Zwiększenie wpływu różniczkowania poprzez zwiększenie mnożnika przy T_d na 6.20 i 6.21 zmniejszyło błąd E oraz prawie usunęło oscylacje.

Dalsze zwiększanie mnożnika przy T_d przywróciło poprzednią skalę oscylacji na rysunkach 6.22 i 6.23.

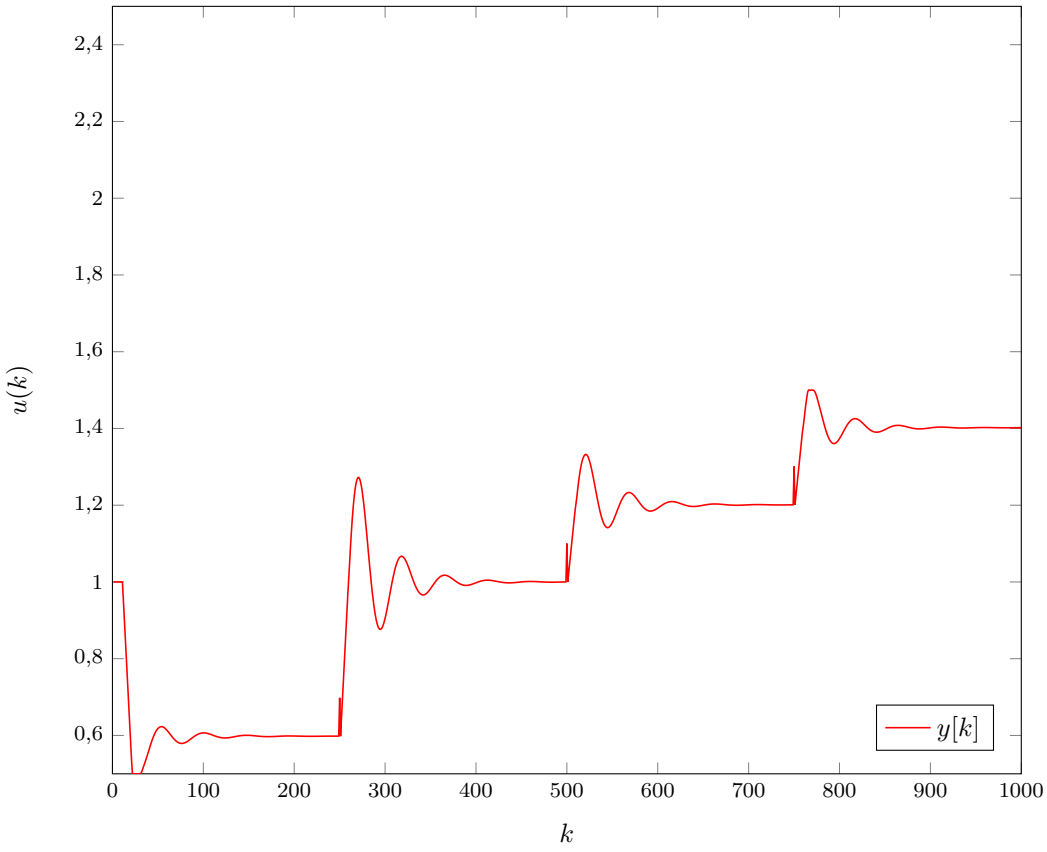
Najmniejszą wartość błędu E osiągnięto dla nastaw $K = 3{,}0125$, $T_\mathrm{i} = 7{,}8$, $T_\mathrm{d} = 5{,}2$, co odpowiada mnożnikom $K = 1{,}0$, $T_\mathrm{i} = 0{,}3$, $T_\mathrm{d} = 0{,}15$.



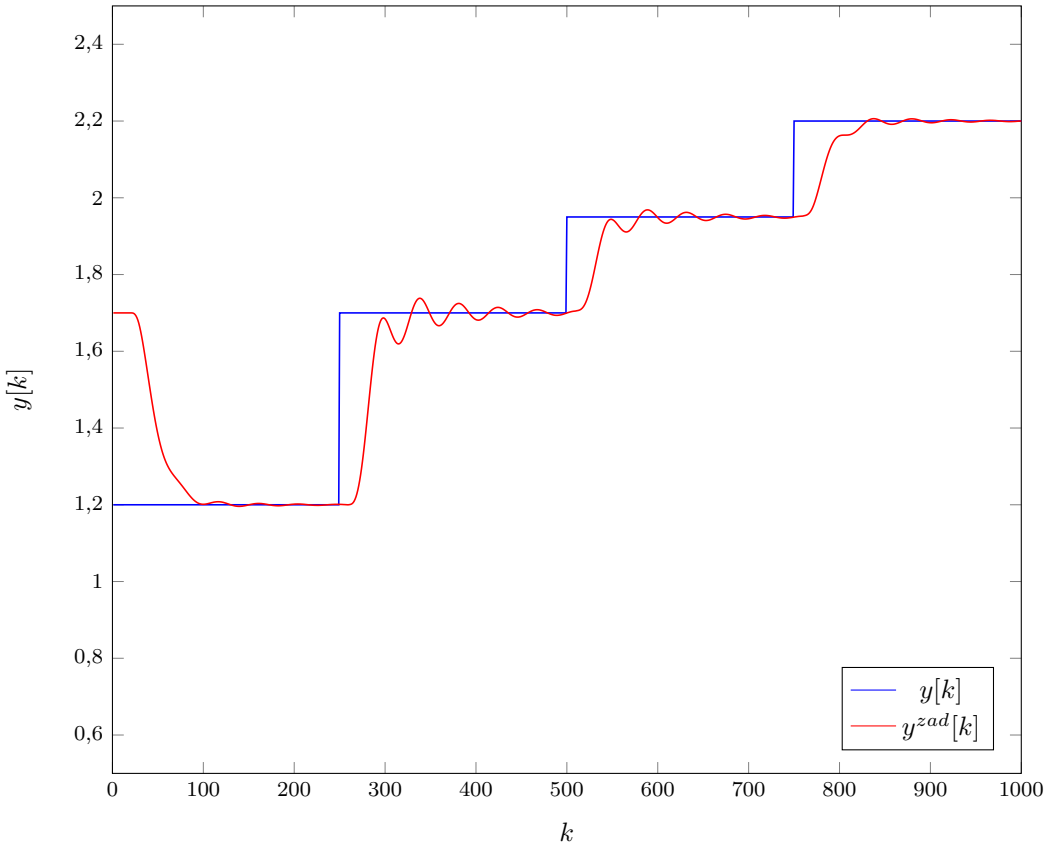
Rys. 6.3. Trajektoria sterowania dla $K = 1{,}8075$ $T_\mathrm{i} = 13$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}100\,90 \cdot 10^1$



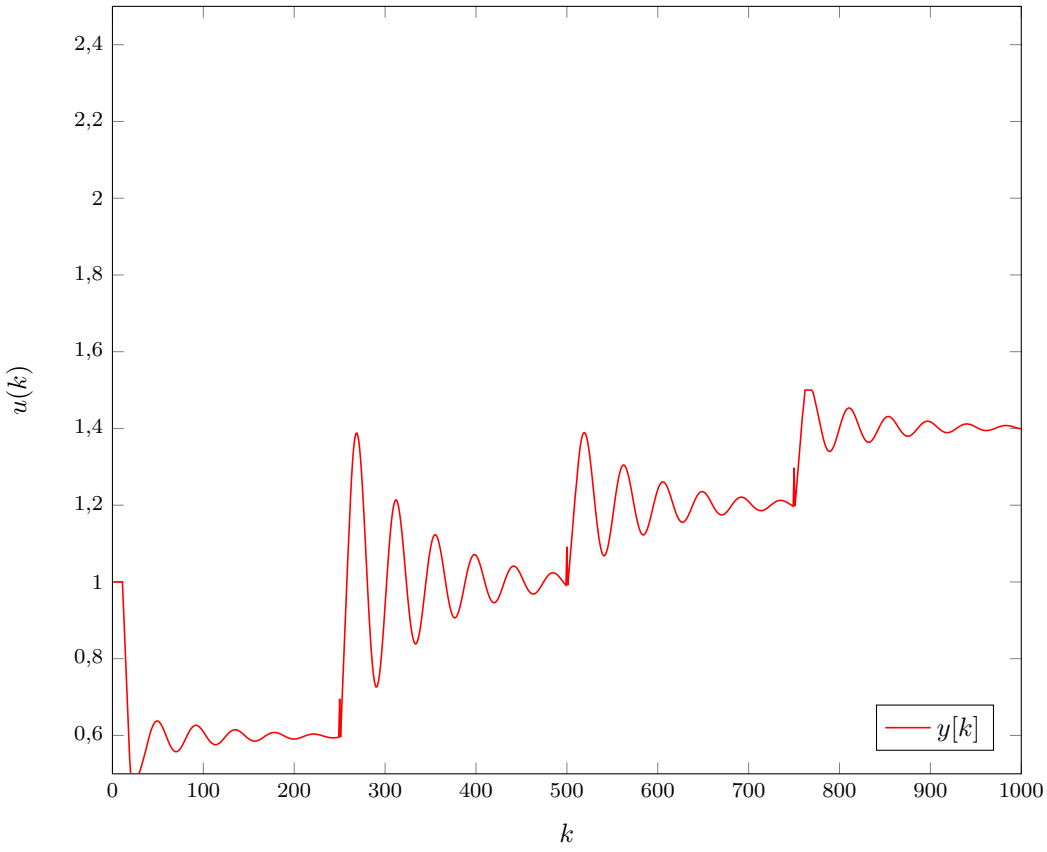
Rys. 6.4. Trajektoria wyjścia dla $K = 2{,}41$ $T_\mathrm{i} = 13$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}029\,04 \cdot 10^1$



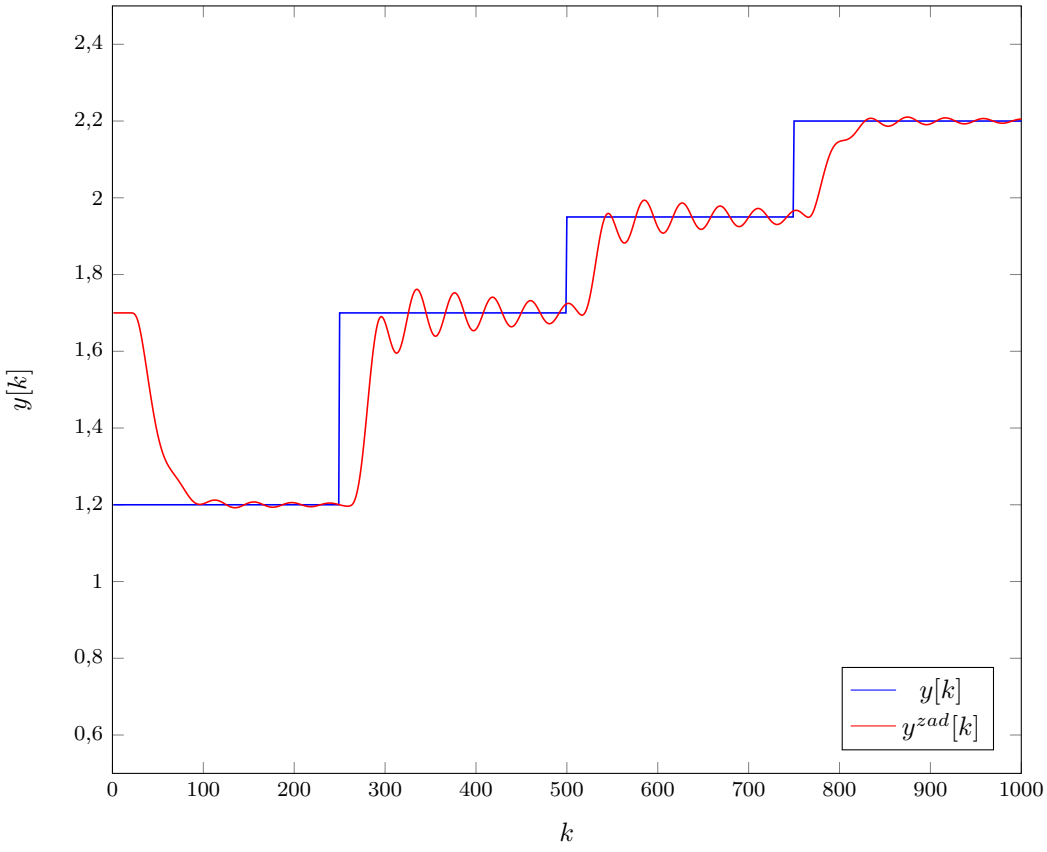
Rys. 6.5. Trajektoria sterowania dla $K = 2{,}41$, $T_\mathrm{i} = 13$, $T_\mathrm{d} = 3{,}25$, $E = 2{,}1009 \cdot 10^1$



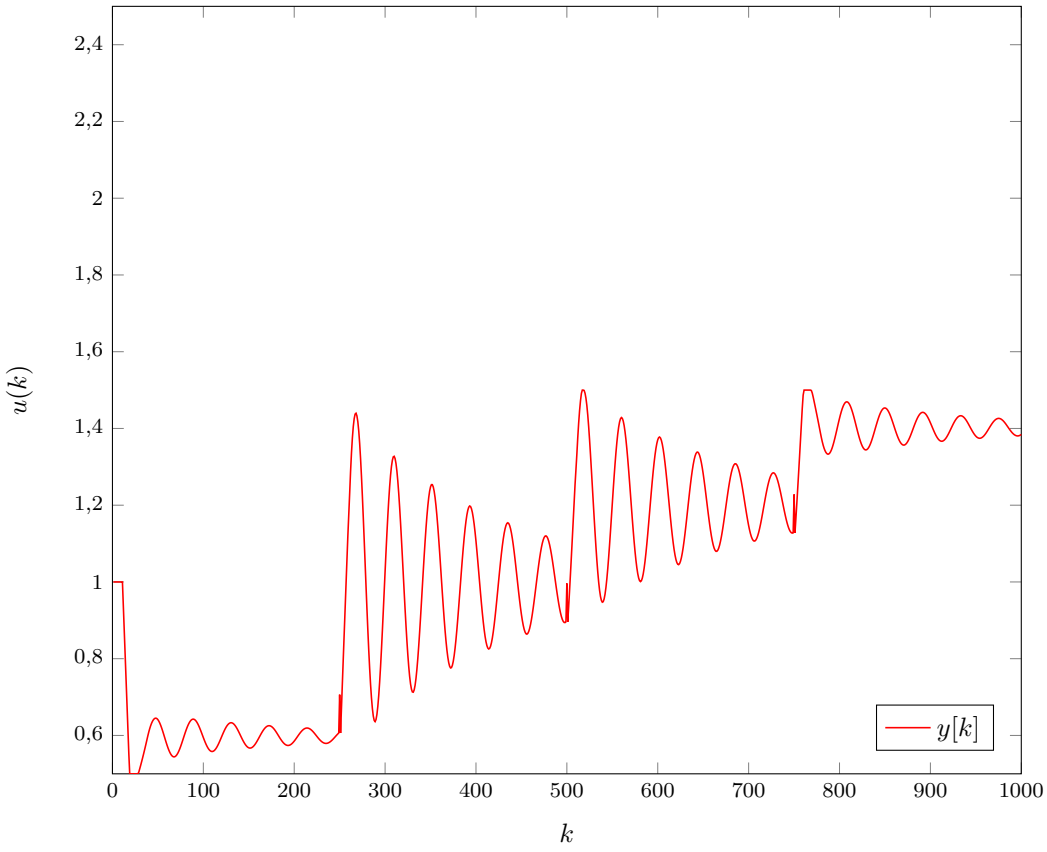
Rys. 6.6. Trajektoria wyjścia dla $K = 3{,}0125$ $T_\mathrm{i} = 13$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}029\,04 \cdot 10^1$



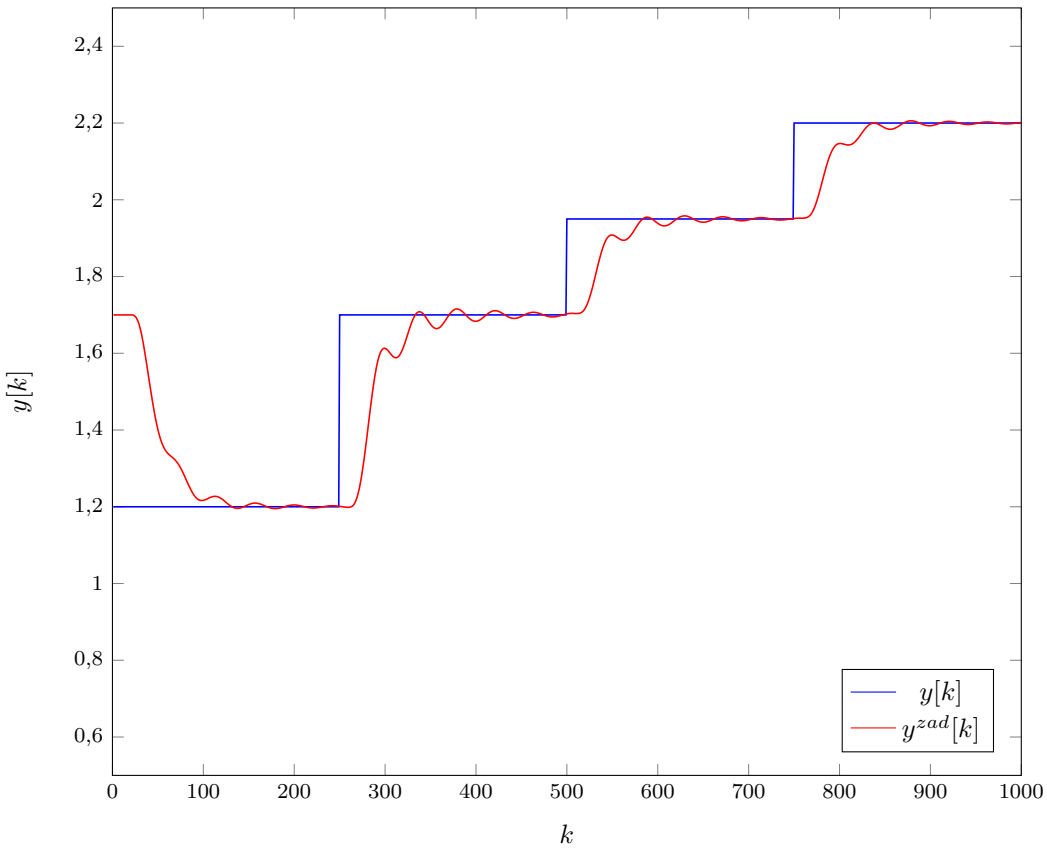
Rys. 6.7. Trajektoria sterowania dla $K = 3{,}0125$ $T_\mathrm{i} = 13$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}029\,04 \cdot 10^1$



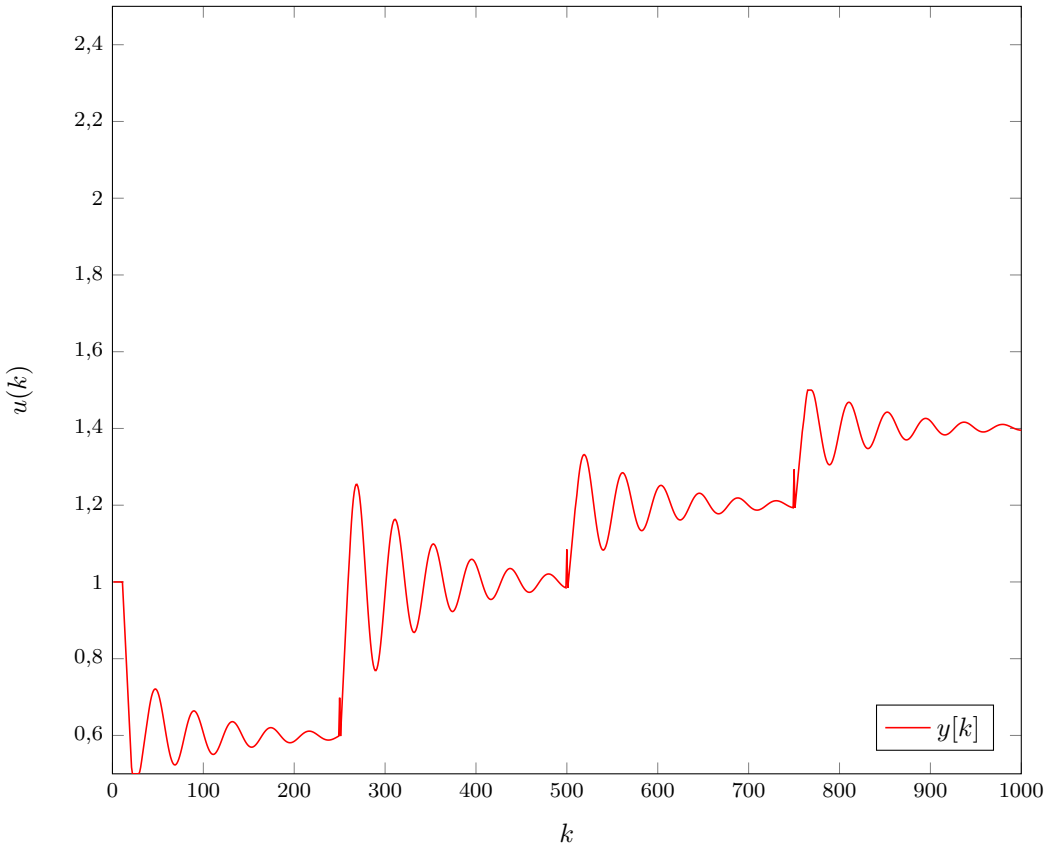
Rys. 6.8. Trajektoria wyjścia dla $K = 3{,}3138$ $T_\mathrm{i} = 13$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}050\,32 \cdot 10^1$



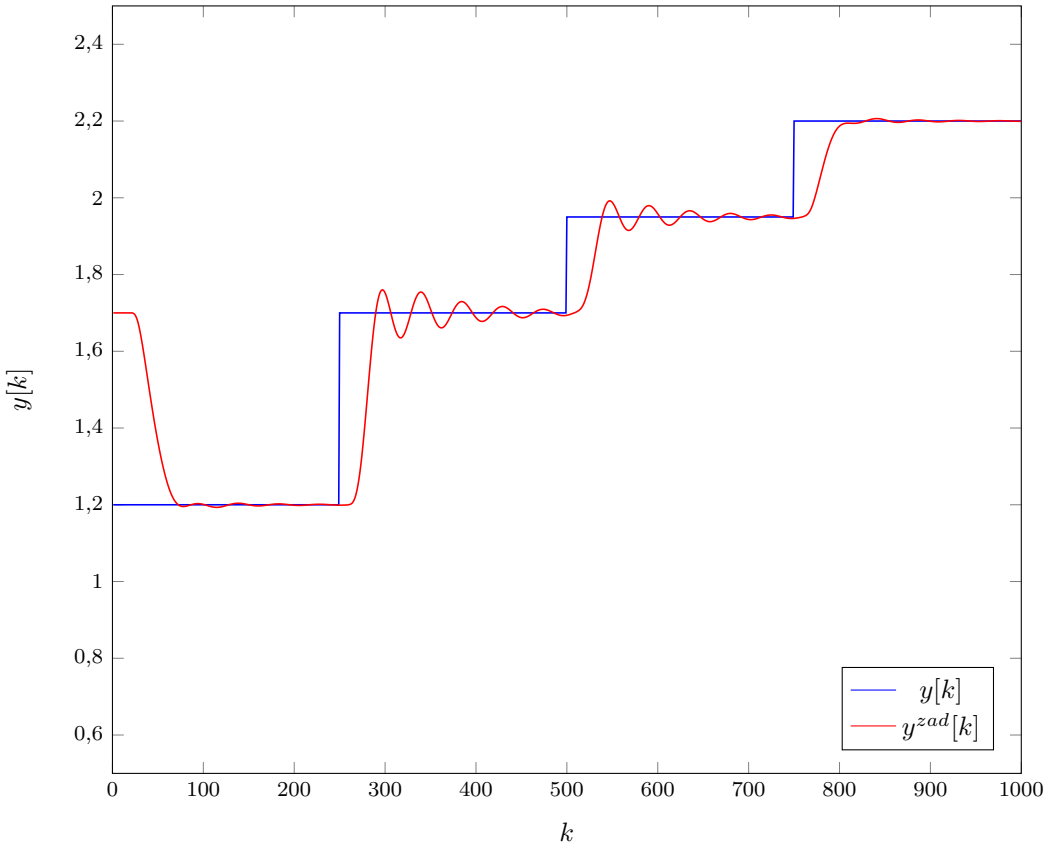
Rys. 6.9. Trajektoria wyjscia $K = 3{,}3138$ $T_\mathrm{i} = 13$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}050\,32 \cdot 10^1$



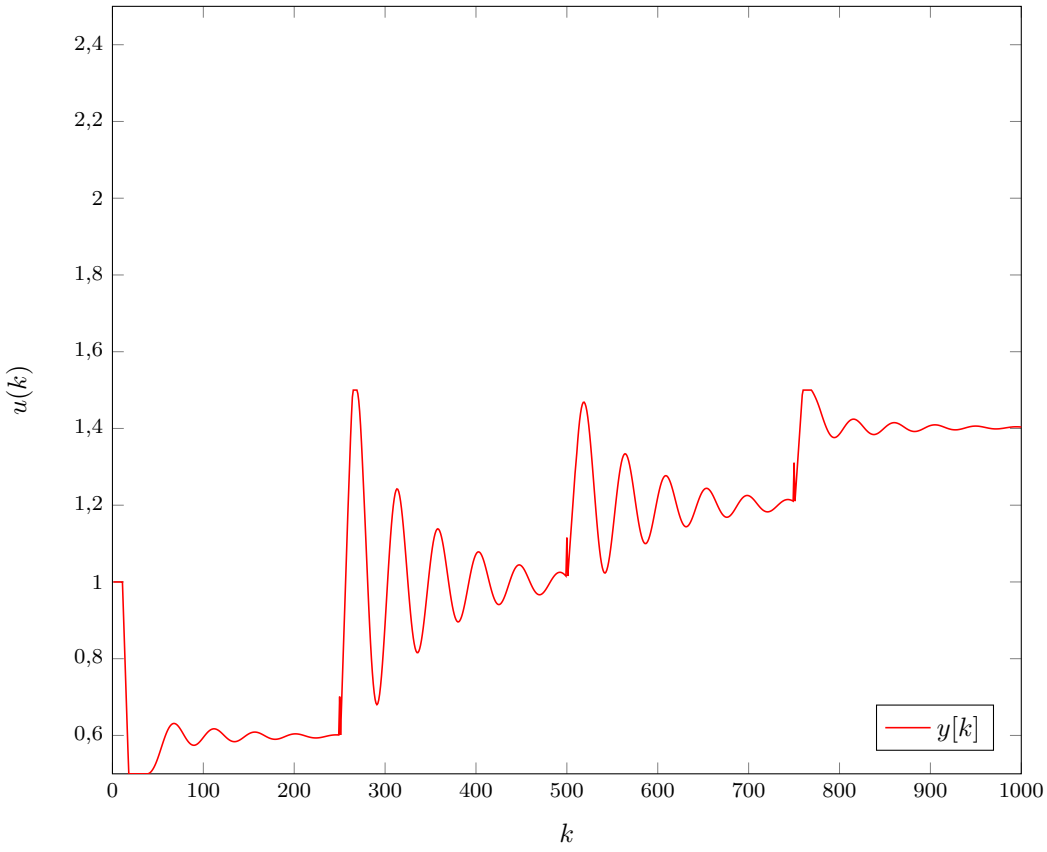
Rys. 6.10. Trajektoria wyjścia dla $K = 3{,}0125$, $T_\mathrm{i} = 15{,}6$, $T_\mathrm{d} = 3{,}25$, $E = 2{,}165\,01 \cdot 10^1$



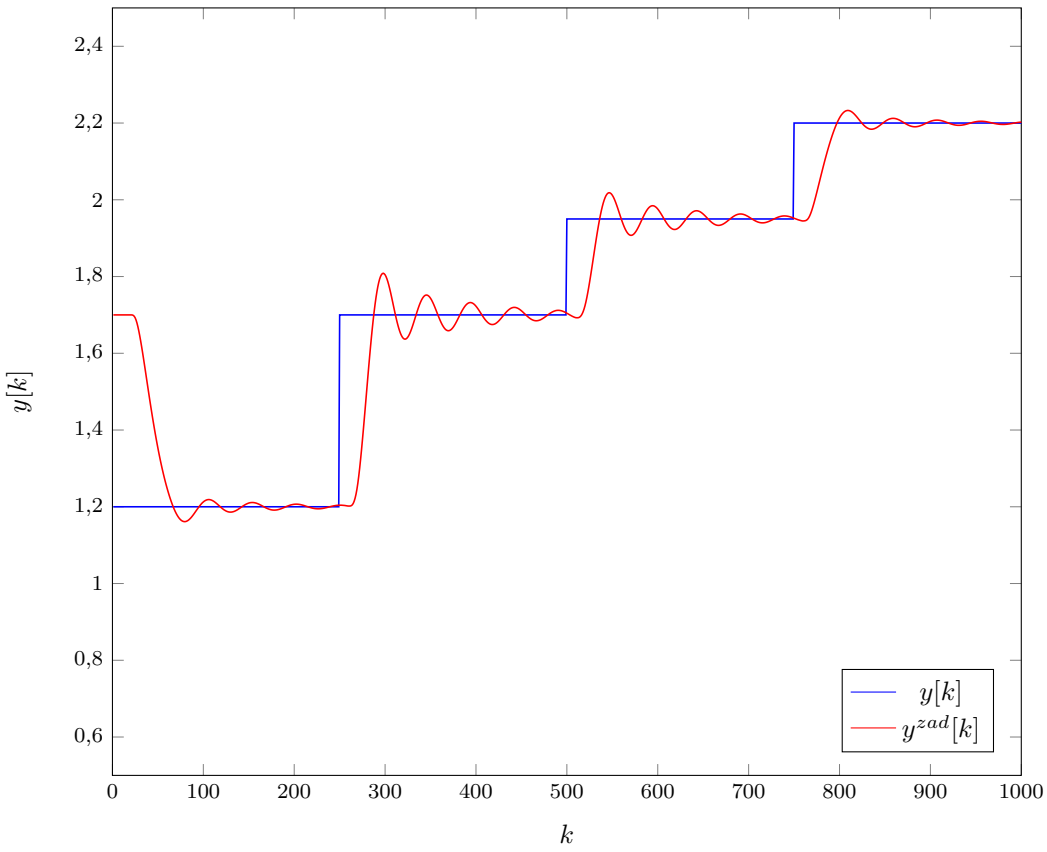
Rys. 6.11. Trajektoria sterowania dla $K = 3.0125$ $T_\mathrm{i} = 15.6$ $T_\mathrm{d} = 3.25$ $E = 2{,}165\,01 \cdot 10^1$



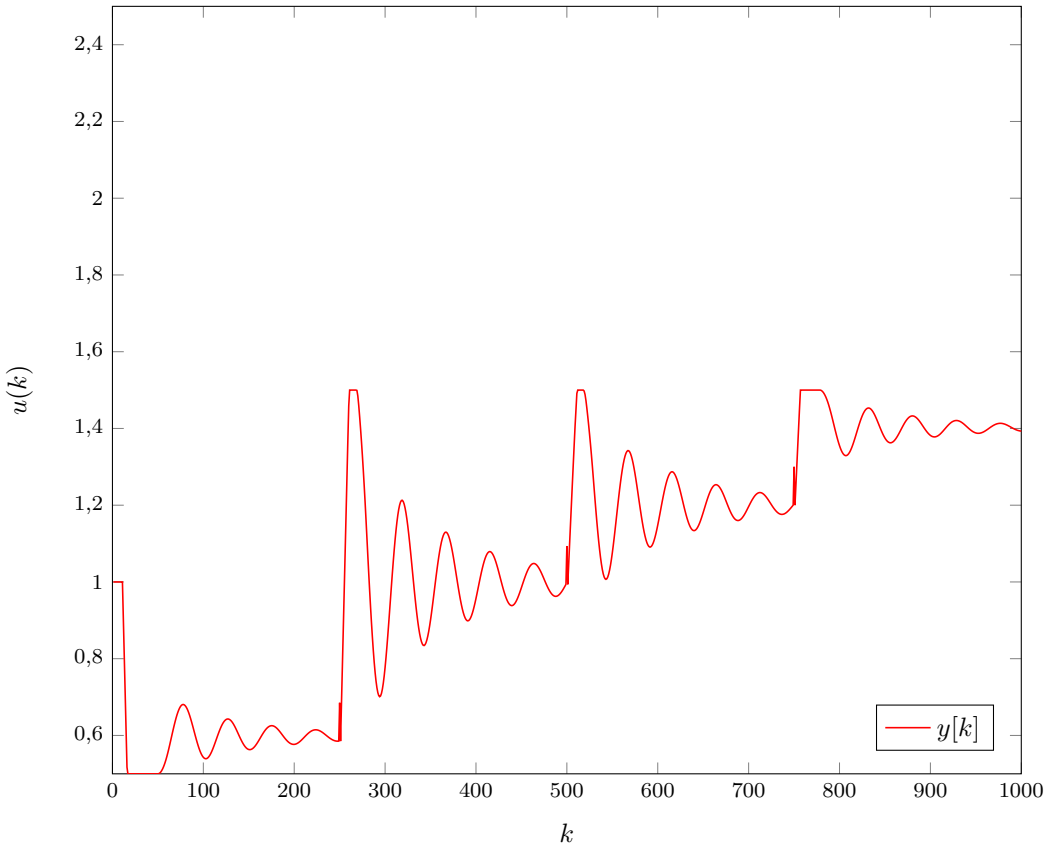
Rys. 6.12. Trajektoria wyjscia $K = 3{,}0125$ $T_\mathrm{i} = 10{,}4$ $T_\mathrm{d} = 3{,}25$ $E = 1{,}935\,33 \cdot 10^1$



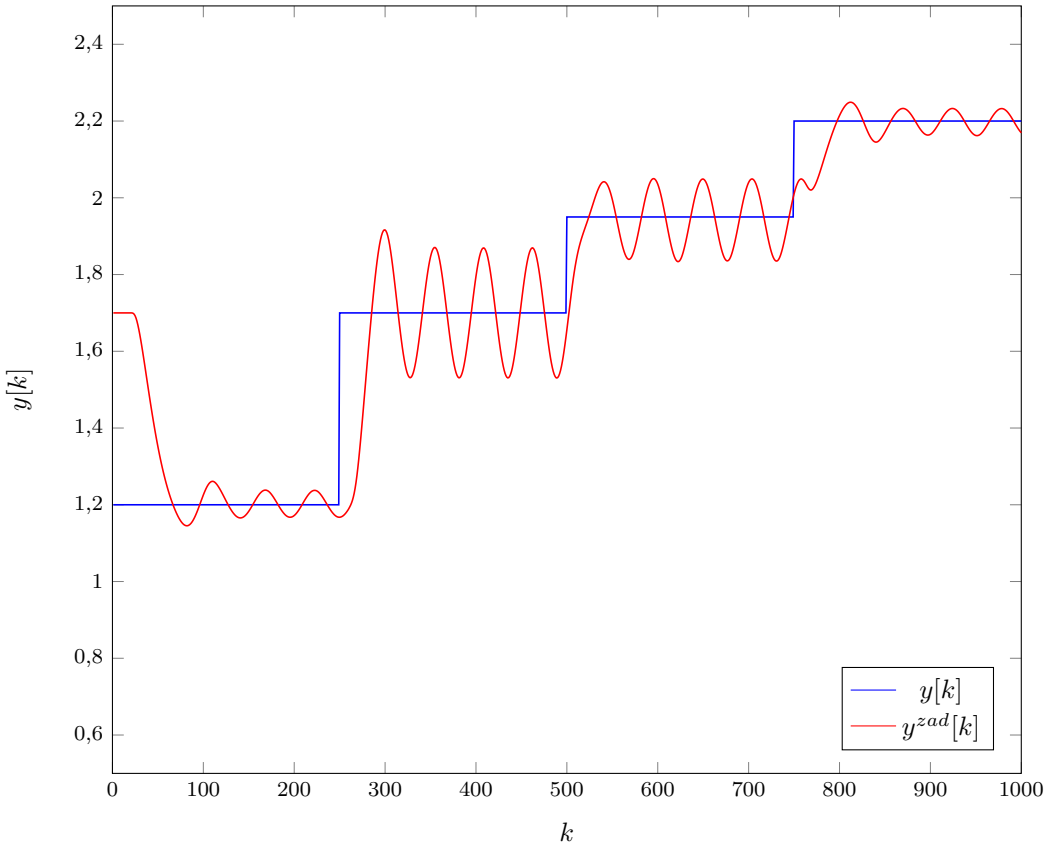
Rys. 6.13. Trajektoria sterowania $K = 3{,}0125$ $T_\mathrm{i} = 10{,}4$ $T_\mathrm{d} = 3{,}25$ $E = 1{,}935\,33 \cdot 10^1$



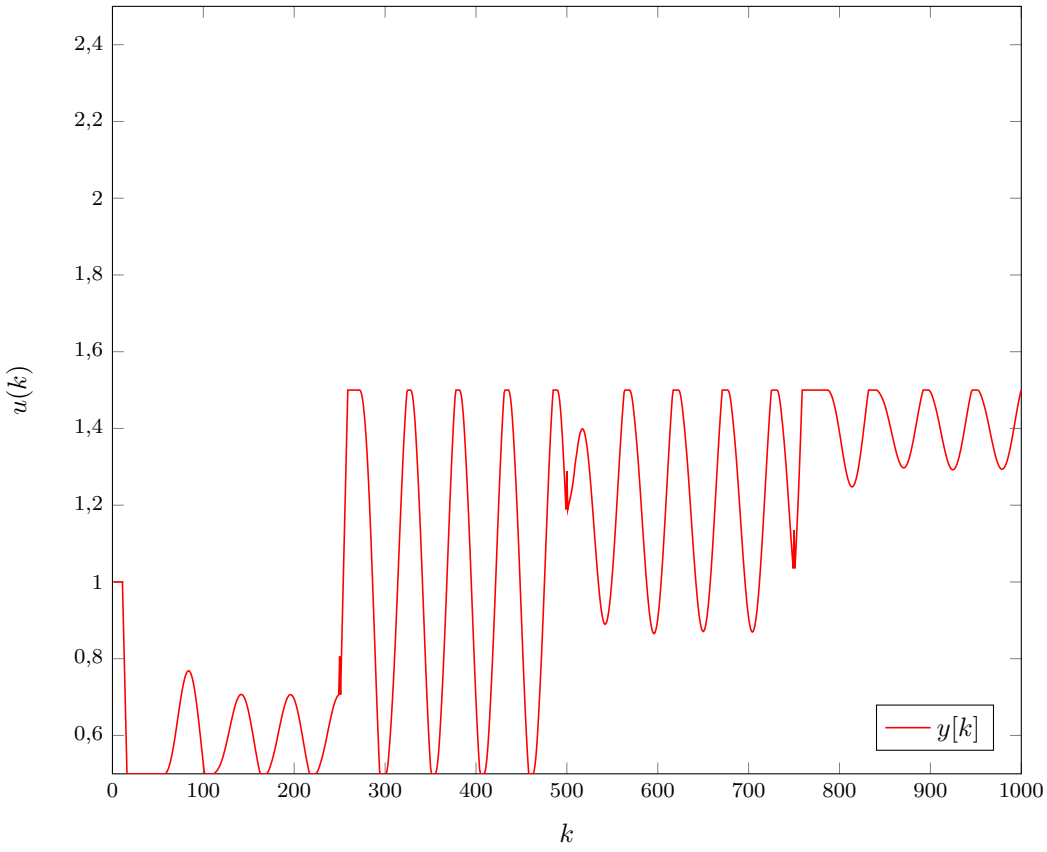
Rys. 6.14. Trajektoria wyjścia $K = 3{,}0125$ $T_\mathrm{i} = 7{,}8$ $T_\mathrm{d} = 3{,}25$ $E = 1{,}908\,32 \cdot 10^1$



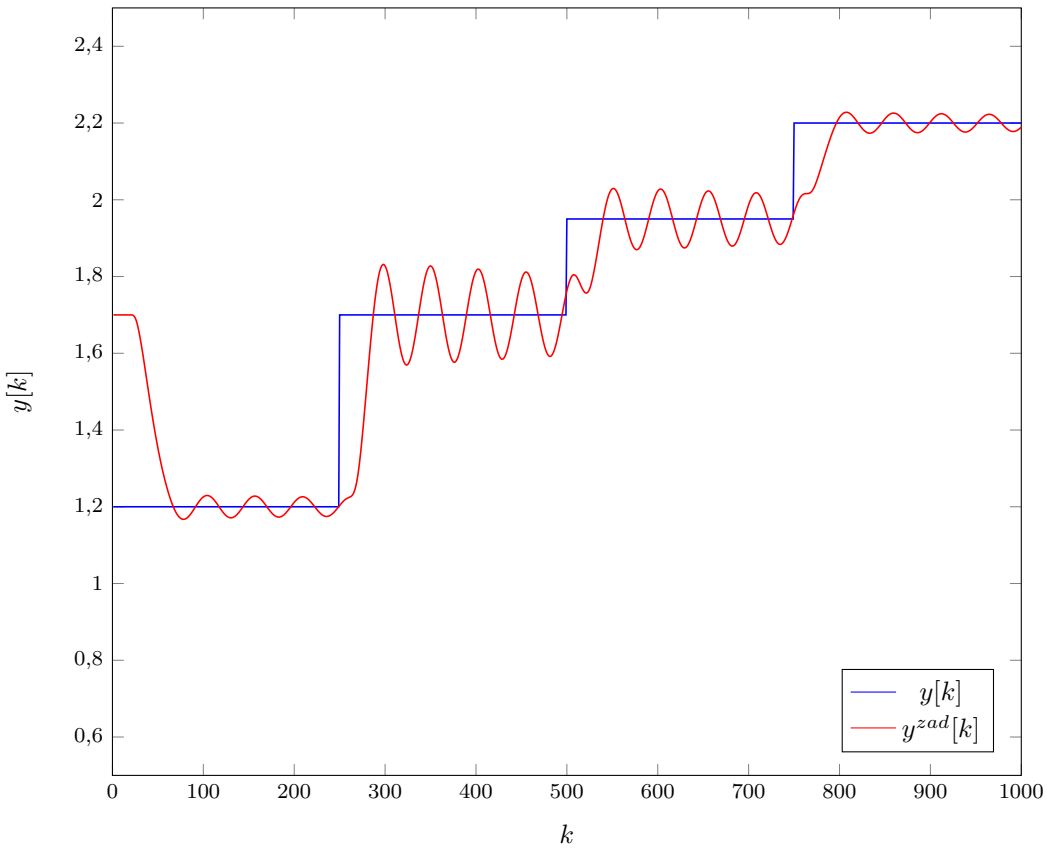
Rys. 6.15. Trajektoria sterowania $K = 3{,}0125$ $T_\mathrm{i} = 7.8$ $T_\mathrm{d} = 3.25$ $E = 1{,}908\,32 \cdot 10^1$



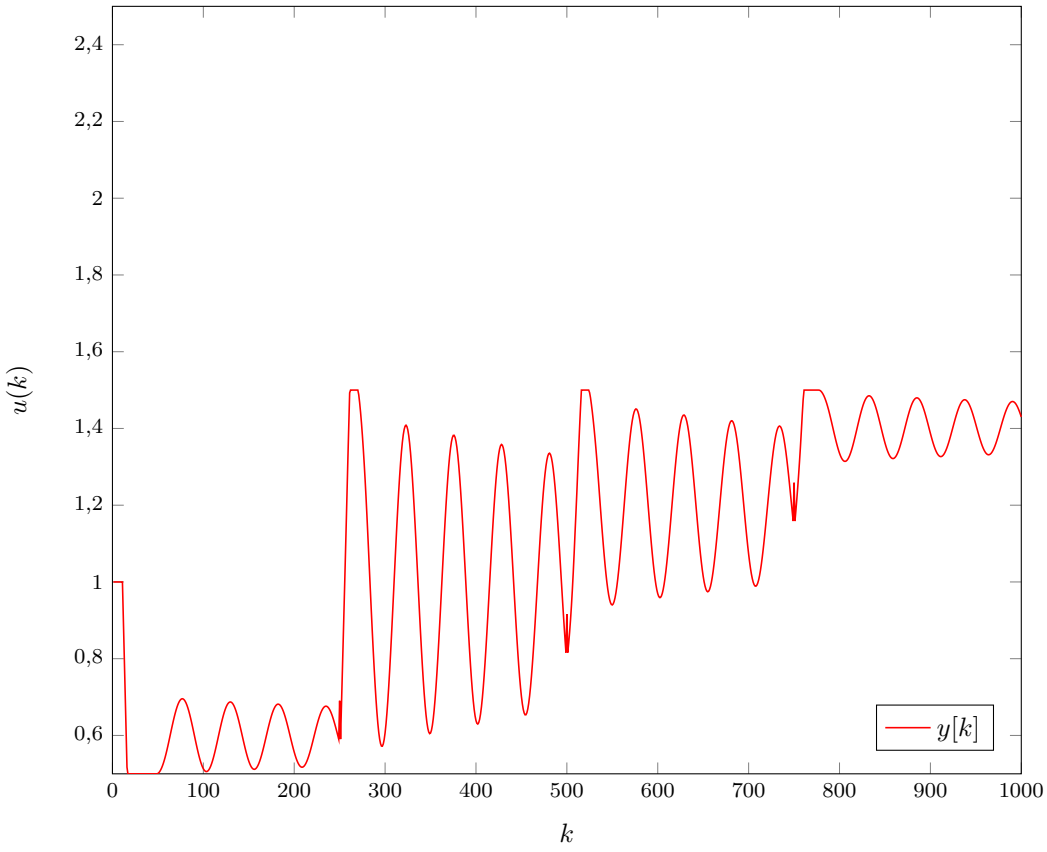
Rys. 6.16. Trajektoria wyjścia $K = 3{,}0125$ $T_\mathrm{i} = 5{,}2$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}157\,71 \cdot 10^1$



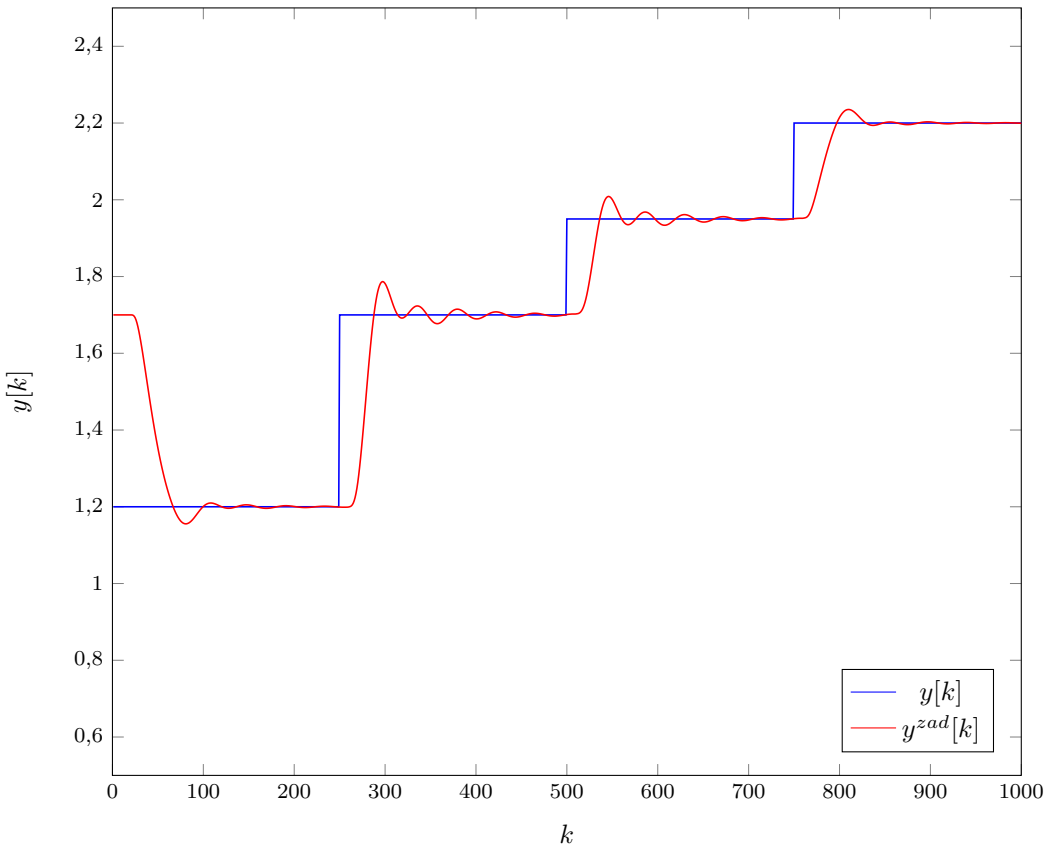
Rys. 6.17. Trajektoria sterowania $K = 3{,}0125$ $T_\mathrm{i} = 5{,}2$ $T_\mathrm{d} = 3{,}25$ $E = 2{,}157\,71 \cdot 10^1$



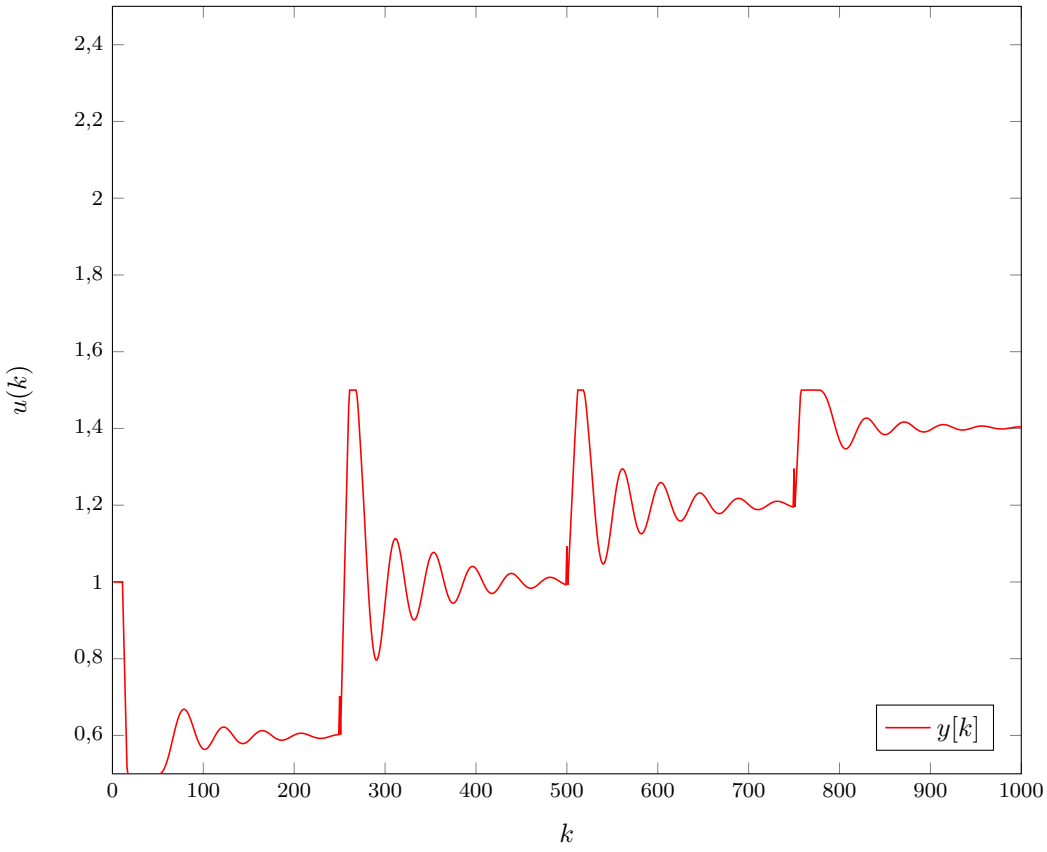
Rys. 6.18. Trajektoria wyjścia dla $K = 3{,}0125$, $T_\mathrm{i} = 7{,}8$, $T_\mathrm{d} = 2{,}6$, $E = 1{,}916\,52 \cdot 10^1$



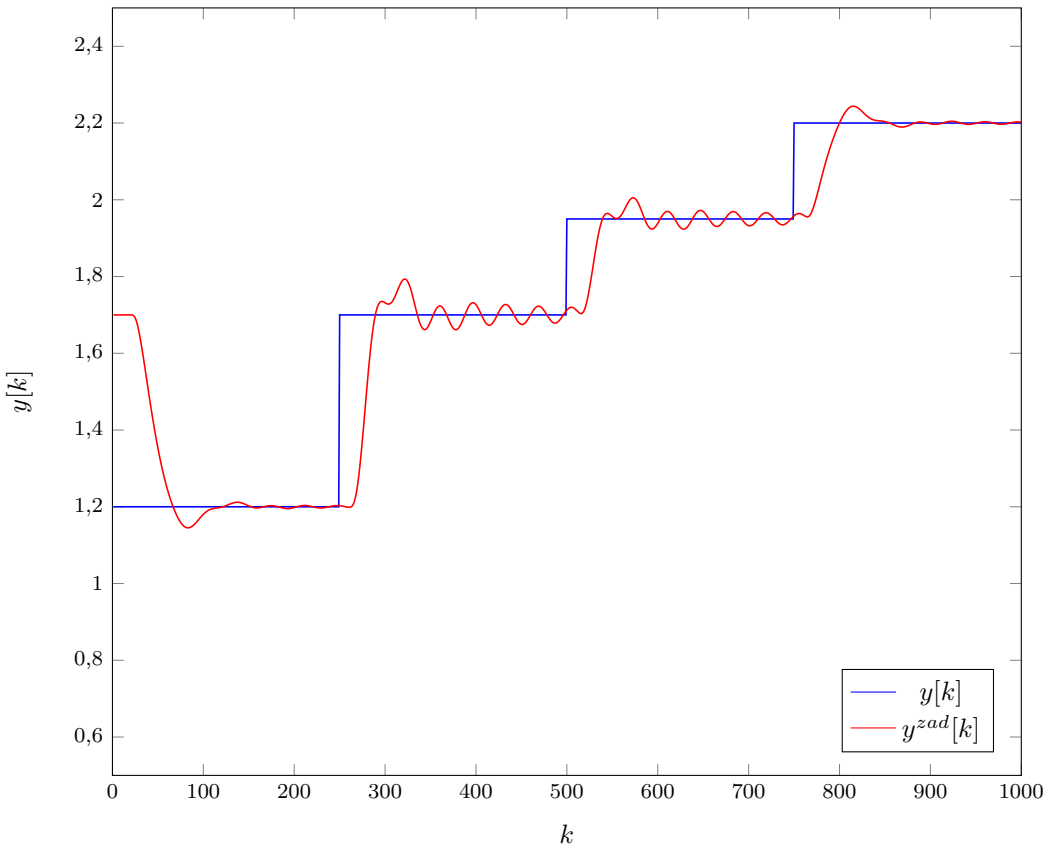
Rys. 6.19. Trajektoria sterowania dla $K = 3{,}0125$, $T_\mathrm{i} = 7{,}8$, $T_\mathrm{d} = 2{,}6$, $E = 1{,}916\,52 \cdot 10^1$



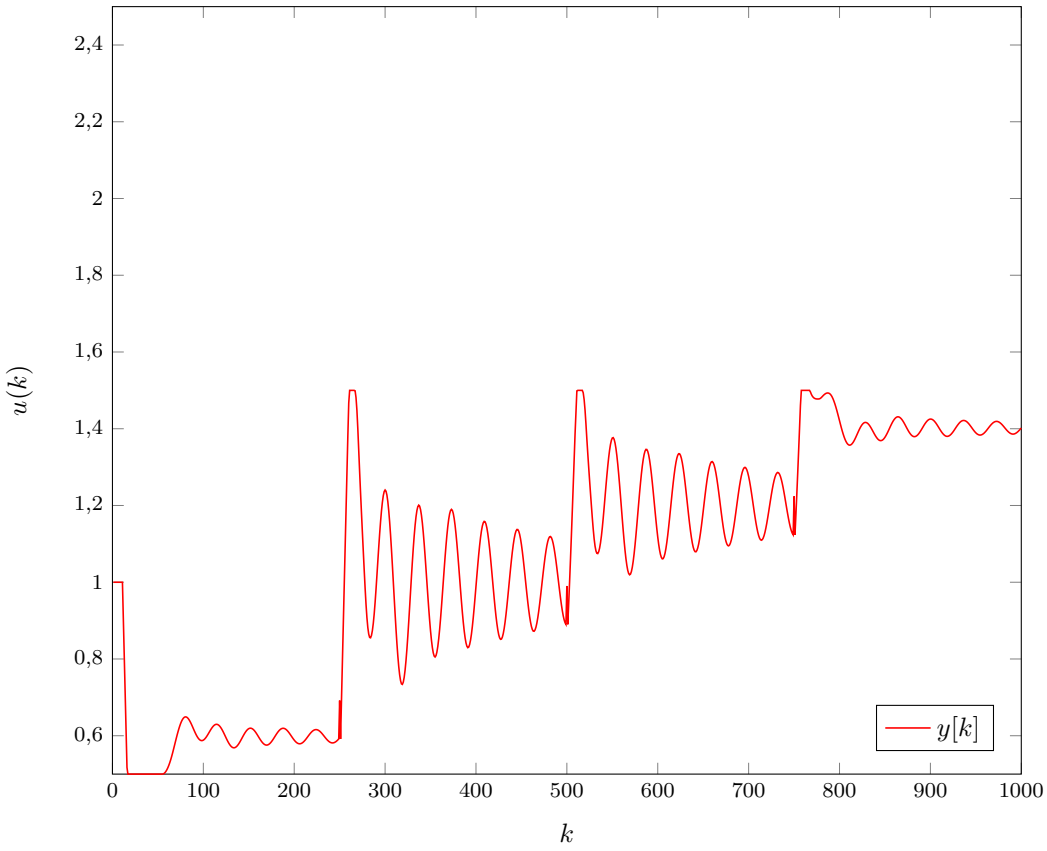
Rys. 6.20. Trajektoria wyjścia dla $K = 3{,}0125$, $T_\mathrm{i} = \mathrm{num}7.8$, $T_\mathrm{d} = 3{,}9$, $E = 1{,}870\,39 \cdot 10^1$



Rys. 6.21. Trajektoria sterowania dla $K = 3{,}0125$, $T_\mathrm{i} = 7{,}8$, $T_\mathrm{d} = 3{,}9$, $E = 1{,}870\,39 \cdot 10^1$



Rys. 6.22. Trajektoria wyjścia dla $K = 3{,}0125$, $T_\mathrm{i} = 7{,}8$, $T_\mathrm{d} = 5{,}2$, $E = 1{,}896\,43 \cdot 10^1$



Rys. 6.23. Trajektoria sterowania dla $K = 3{,}0125$, $T_\mathrm{i} = 7{,}8$, $T_\mathrm{d} = 5{,}2$, $E = 1{,}896\,43 \cdot 10^1$

7. Strojenie regulatora DMC

Skrypty wykorzystywane do zrealizowania tego zadania: `zad5P_DMC.m`, `zad4P_dmc.m`, `zad4P_dmcGeneration` `zad4P_duNew.m`, `zad4P_script.m`.

7.1. Zasady strojenia regulatora DMC

Dobór parametrów należy zacząć od przyjęcia małej wartości początkowej λ. Następnie dobiera się możliwie długie horyzonty predykcji, sterowania i dynamiki. Na początku skraca się maksymalnie horyzont dynamiki **D** do momentu, aż nie zacznie pogarszać to regulacji. Następnie dla dobranego **D** wybiera się w identyczny sposób horyzont predykcji **N**. Na końcu wybiera się horyzont sterowania N_{u}.
Dla tak ustalonych horyzontów dobiera się λ , która jest kompromisem pomiędzy szybkością regulacji i kształtem sygnału sterującego.

Dobierając odpowiednie parametry pamiętaliśmy o wartości wskaźnika jakości - E, który podany był w poleceniu.

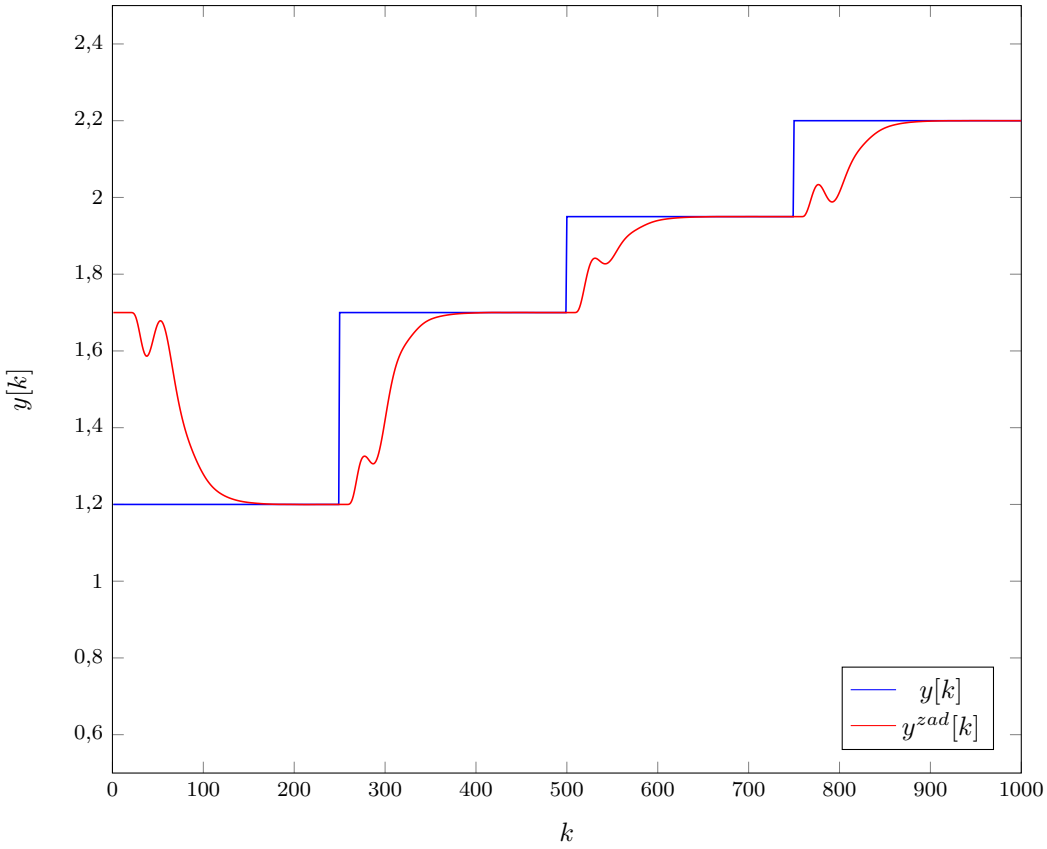
$$E = \sum_{k=1}^{K_{konc}} (y^{zad}(k) - y(k))^2 \tag{7.1}$$

Został on wpisany w opis rysunku dla danych parametrów regulatora.

7.2. Dobór parametrów regulatora DMC

7.2.1. Dobór horyzontu dynamiki D

Rysunki 7.1 i 7.2 przedstawiają horyzont dynamiki równy 150. Widać pewną "obawę" regulatora, że jego sterowanie z chwil po zmianie wartości zadanej spowoduje jej przekroczenie - co widać na nagłym spadku wartości sterowania po pierwszym skoku w trajektorii sterowania.



Rys. 7.1. Trajektoria wyjścia dla $D = 150$, $N = 150$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}949\,28 \cdot 10^{1}$

Zmniejszenie horyzontu dynamiki do 100 na rysunkach 7.3 i 7.4 niewiele zmieniło sterowanie oraz błąd.

Zmniejszenie wartości horyzontu dynamiki do 70 na rysunkach 7.5 i 7.6 zmniejszyło błąd E minimalnie, jednak pogorszyło trajektorię wyjścia. Dalsze zmniejszenie horyzontu dynamiki powiększyłoby szkodę dla trajektorii wyjścia, dlatego horyzont dynamiki został ustawiony na długość 70.

7.2.2. Dobór horyzontu predykcji N

Zmniejszenie horyzontu predykcji do 100 na rysunkach 7.7 i 7.8 niczego nie zmieniło - trajektoria wygląda tak samo, błąd E jest taki sam.

Dalsze zmniejszenie horyzontu predykcji do wartości 80 na rysunkach 7.9 i 7.10 dalej niczego nie zmieniło.

Dopiero wyraźna zmiana horyzontu predykcji na 30 na rysunkach 7.11 i 7.12 zmniejszyło błąd E. Zmiany w trajektoriach są jednak niezauważalne.

7.2.3. Dobór horyzontu sterowania N_{u}

Dla rysunków 7.13, 7.14, 7.15 i 7.16, które odpowiadają horyzontom sterowania równym 100 i 80 trajektorie wyjścia oraz błędy E są praktycznie identyczne. Powyższe doświadczenie pokazuje jak złą praktyką, przynajmniej dla tego obiektu, są horyzonty predykcji dłuższe od horyzontów dynamiki oraz horyzonty sterowania dłuższe od horyzontów predykcji.

Dopiero silna zmiana horyzontu sterowania na rysunkach 7.17 i 7.18 do wartości 10 zmniejszyło błąd E.

7.2.4. Wpływ współczynnika λ

Zwiększenie współczynnika λ do wartości 10 na rysunkach 7.19 i 7.20 zmniejszyło błąd E i przyspieszyło regulację kosztem zwiększenia przesterowania.

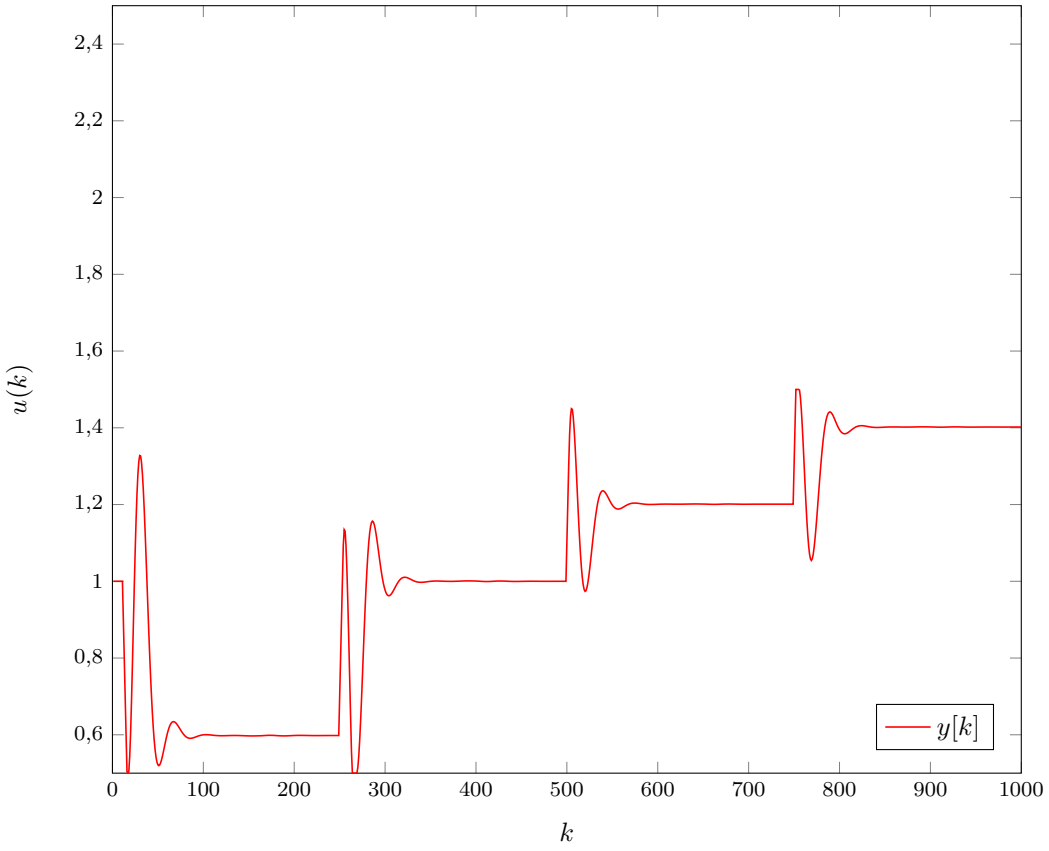
Zwiększenie współczynnika λ do wartości 20 na rysunkach 7.21 i 7.22 zmniejszyło błąd E dalej i polepszyło trajektorię wyjścia na granicach zakresu sterowania kosztem jakości wyjścia w bliskiej okolicy punktu pracy.

Zwiększenie współczynnika λ do wartości 50 na rysunkach 7.23 i 7.24 zwiększyło błąd E. Ten nastaw byłby jednak warty rozważenia przez wzgląd na trajektorię sterowania - wyglądająca znacznie płynniej niż poprzednie.

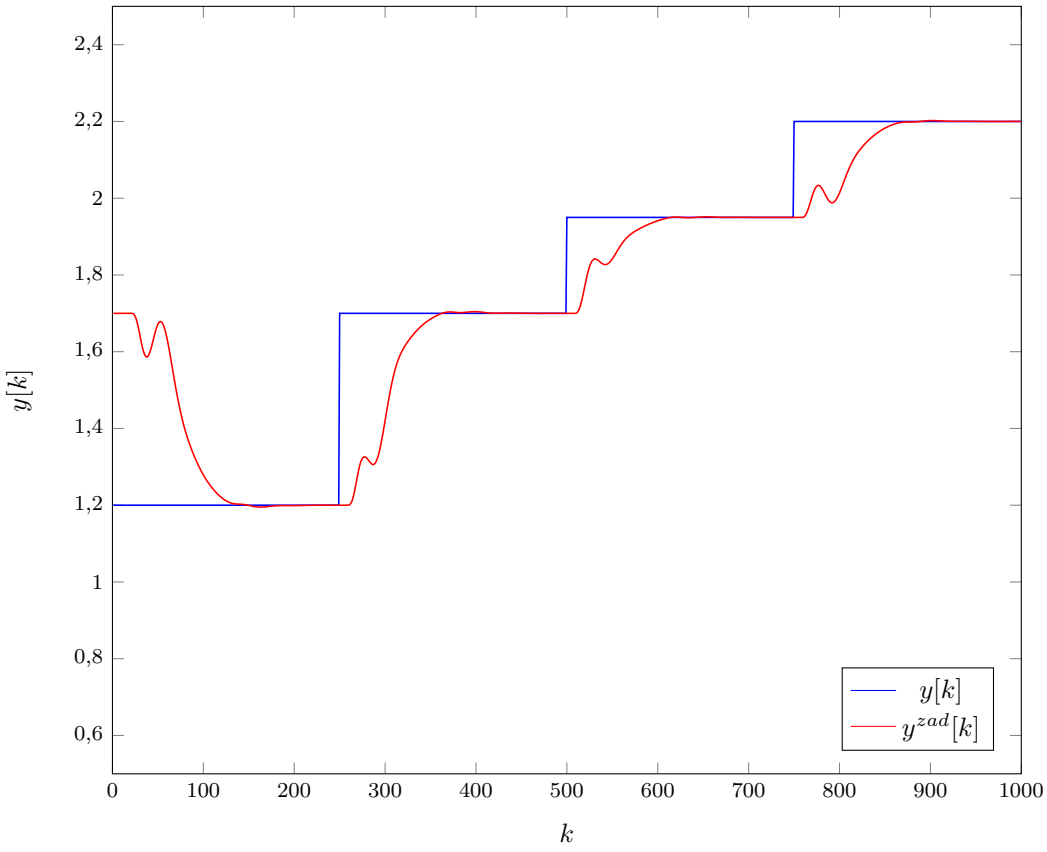
7.3. Próba wtórnej zmiany D , N i N_{u}

Regulator DMC jest zależny od 4 parametrów. Zmieniając je pojedynczo można wpaść w minimum lokalne, które znacznie odbiega od minimum globalnego. Dlatego przeprowadzona została próba podwyższenia parametrów D , N i N_{u} i ponowne porównanie wpływu współczynnika λ na charakterystyki sterowania.

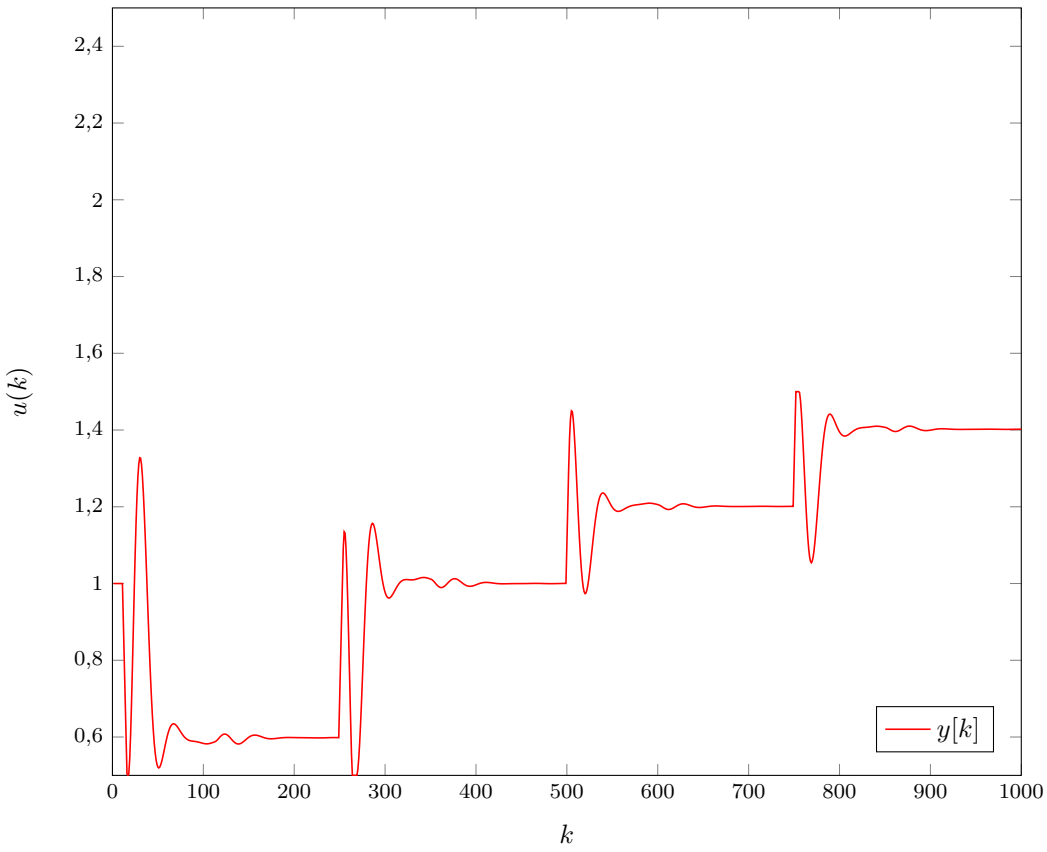
Wszystkie trzy nastawy na rysunkach 7.25, 7.26, 7.27, 7.28, 7.29 i 7.30 to optycznie jedna i ta sama regulacja, eliminująca praktycznie przesterowanie i regulująca zadowalająco szybko. Jednak dla wartości $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 20$ błąd E jest najmniejszy.



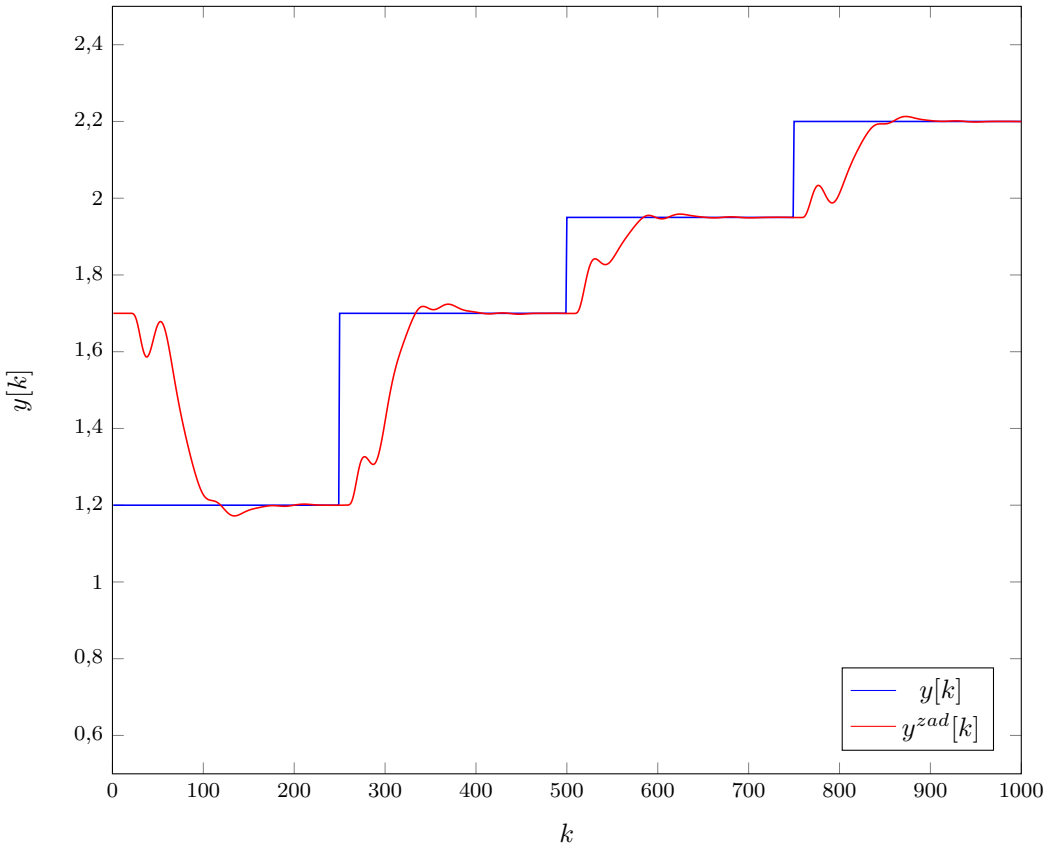
Rys. 7.2. Trajektoria sterowania dla $D = 150$, $N = 150$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}949\,28 \cdot 10^{1}$



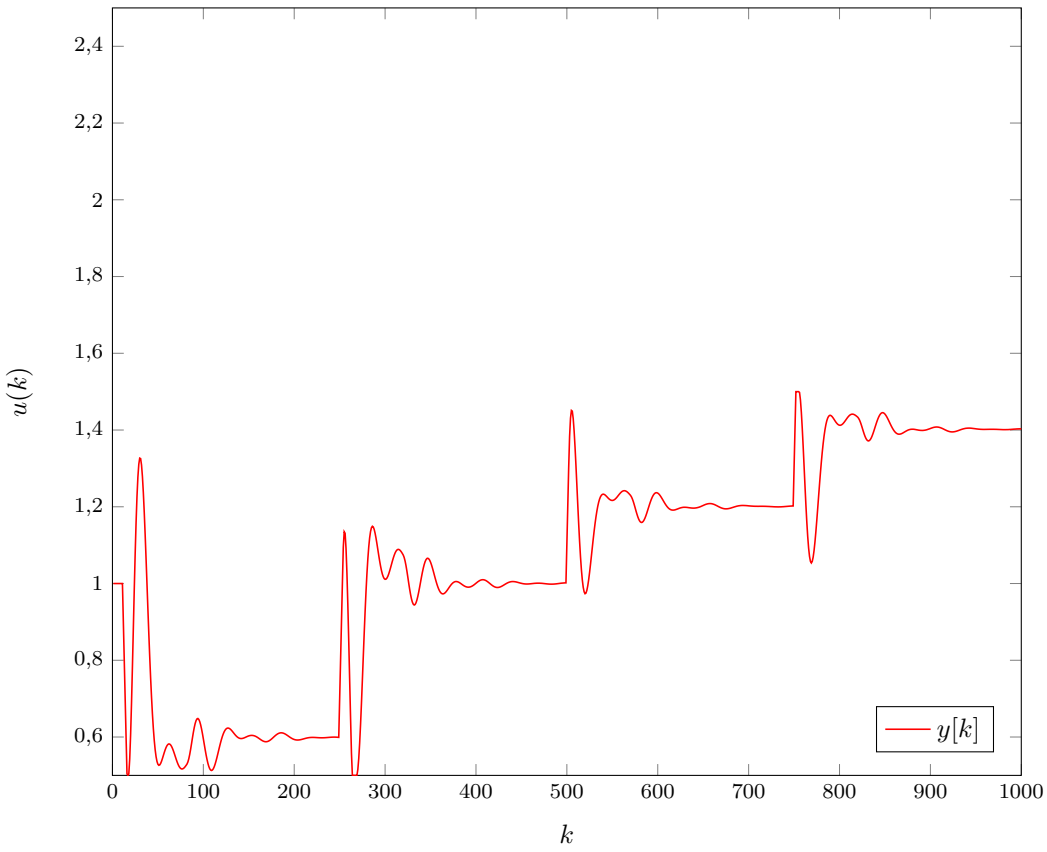
Rys. 7.3. Trajektoria wyjścia dla $D = 100$, $N = 150$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}9479 \cdot 10^1$



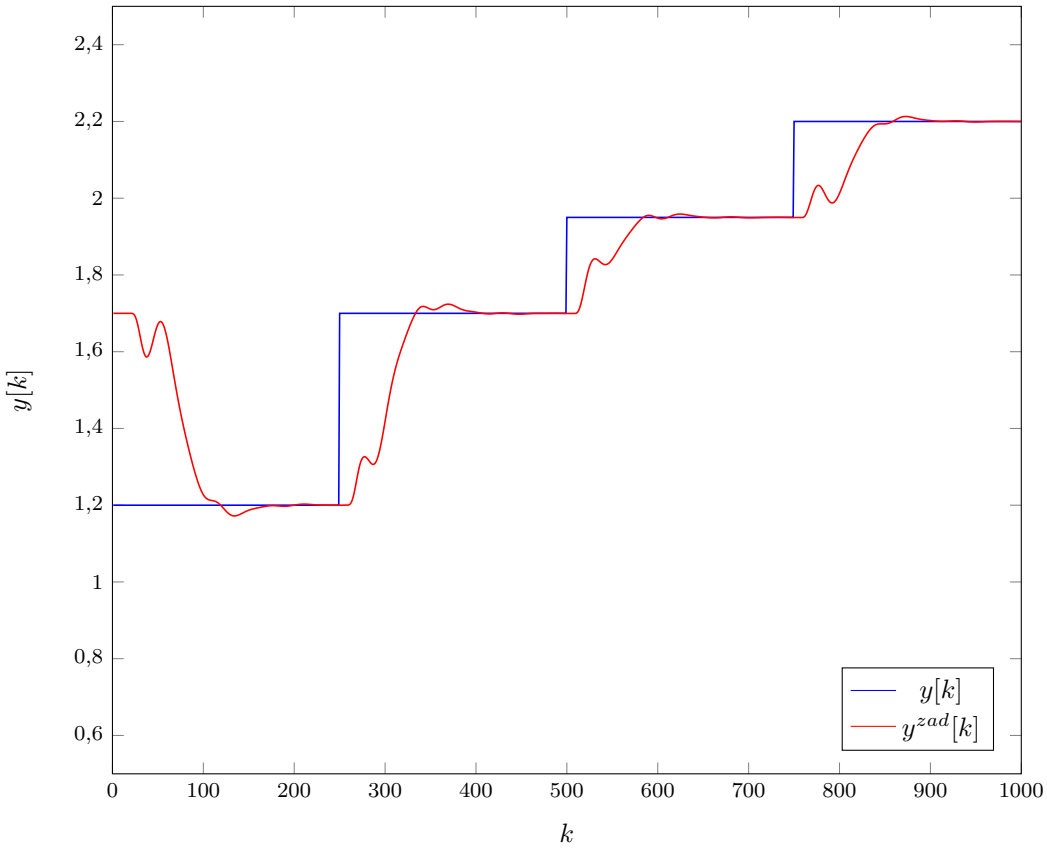
Rys. 7.4. Trajektoria sterowania dla $D = 100$, $N = 150$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}9479 \cdot 10^1$



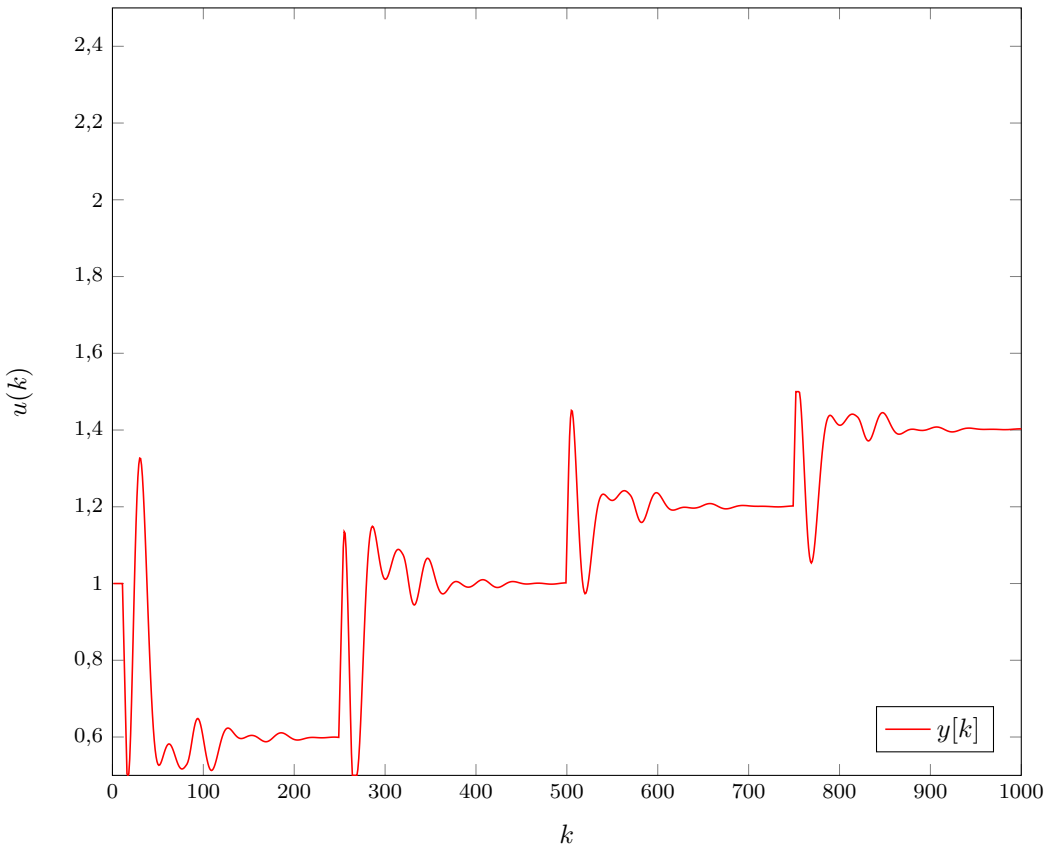
Rys. 7.5. Trajektoria wyjścia dla $D = 70$, $N = 150$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}922\,49 \cdot 10^1$



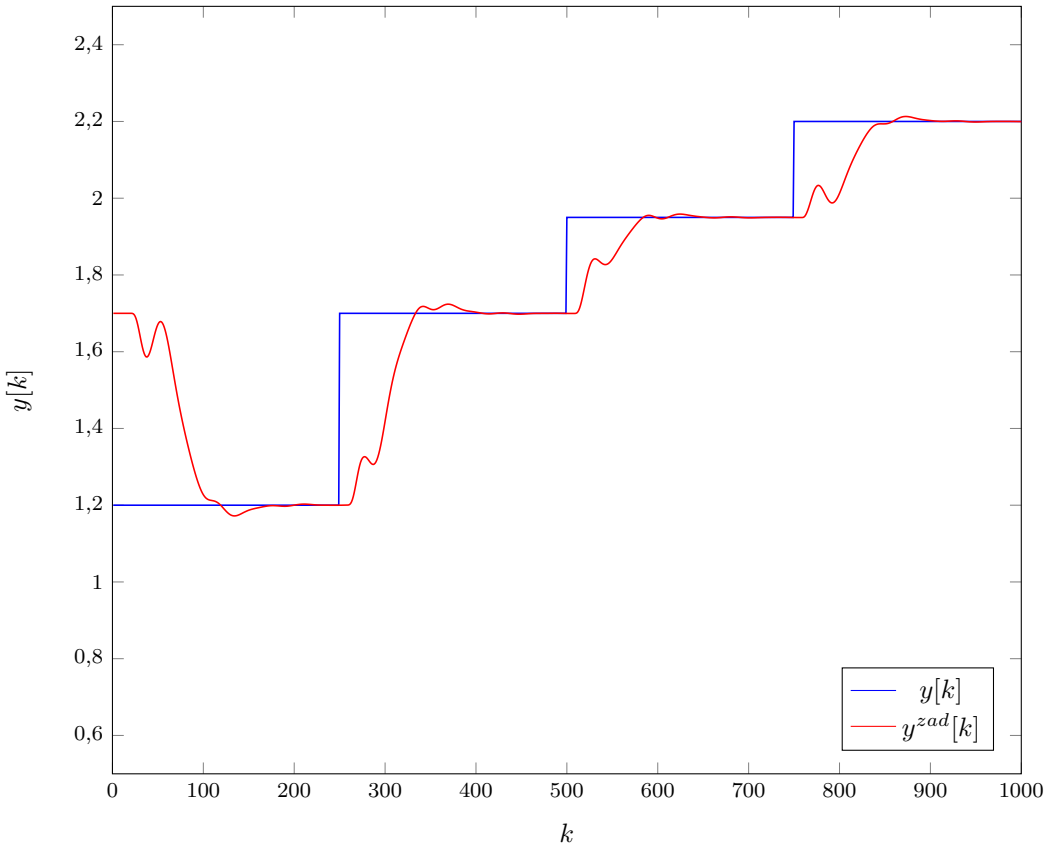
Rys. 7.6. Trajektoria sterowania dla $D = 70$, $N = 150$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}922\,49 \cdot 10^1$



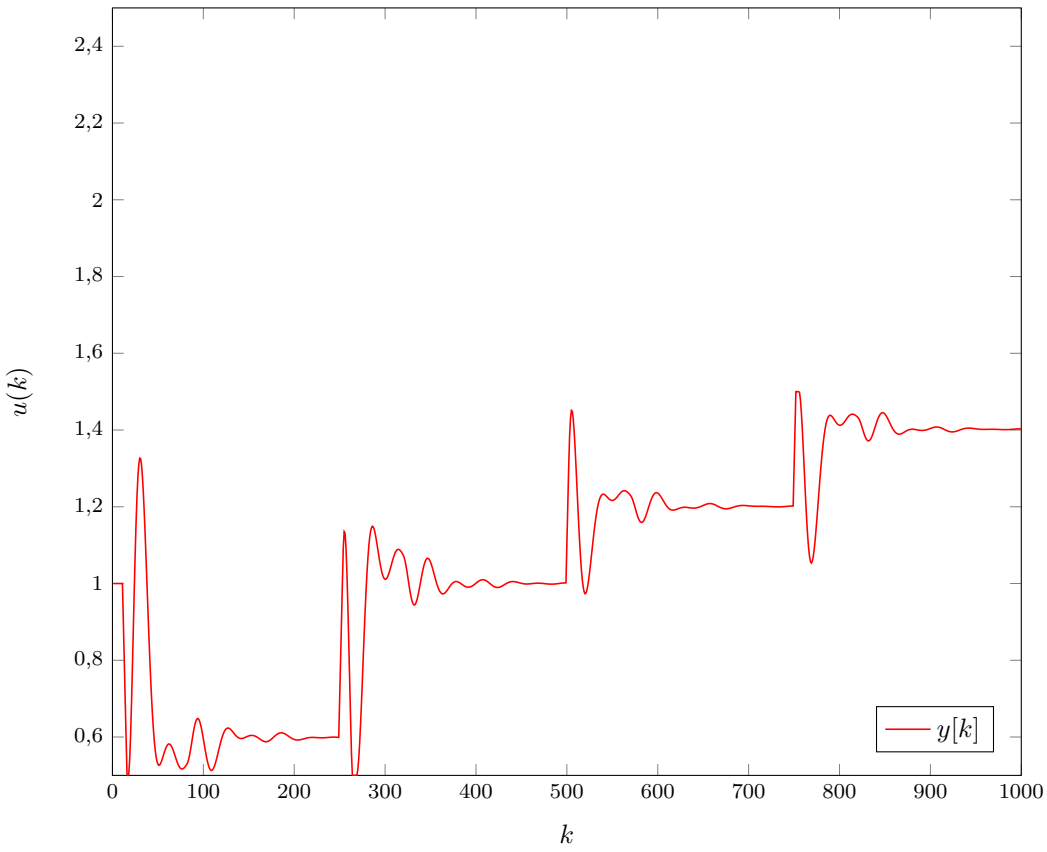
Rys. 7.7. Trajektoria wyjścia dla $D = 70$, $N = 100$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}922\,49 \cdot 10^1$



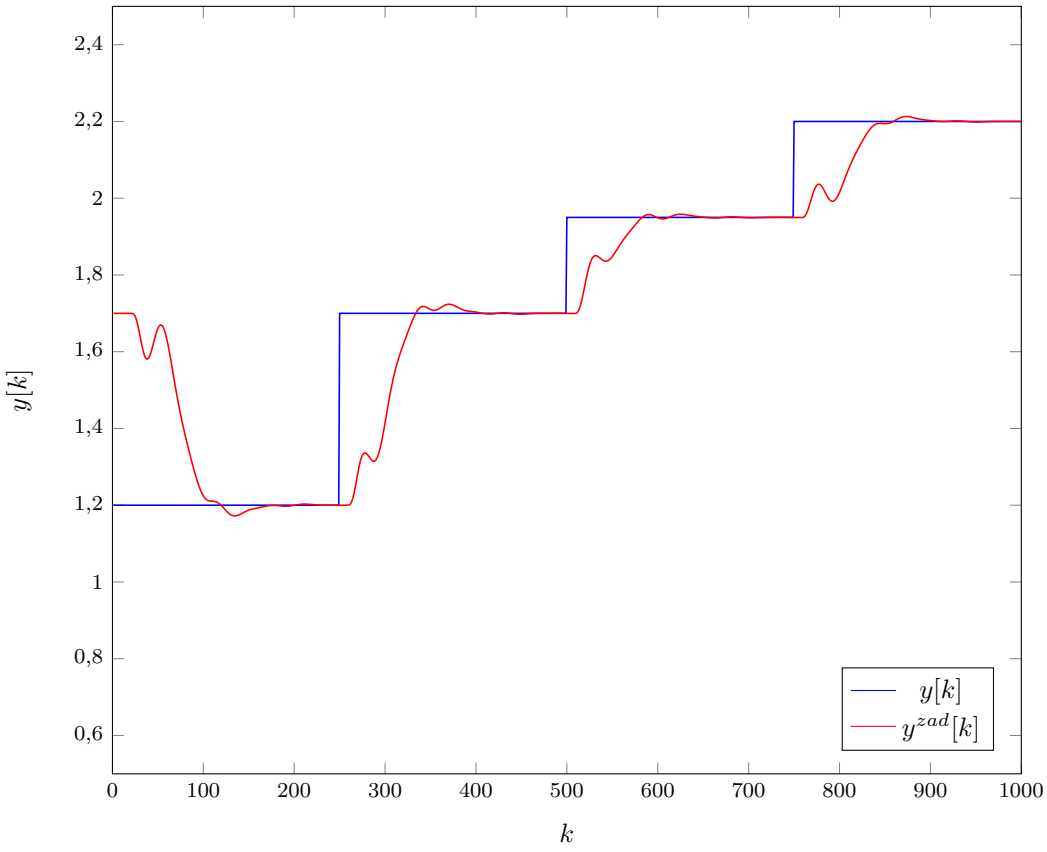
Rys. 7.8. Trajektoria sterowania dla $D = 70$, $N = 100$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}922\,49 \cdot 10^1$



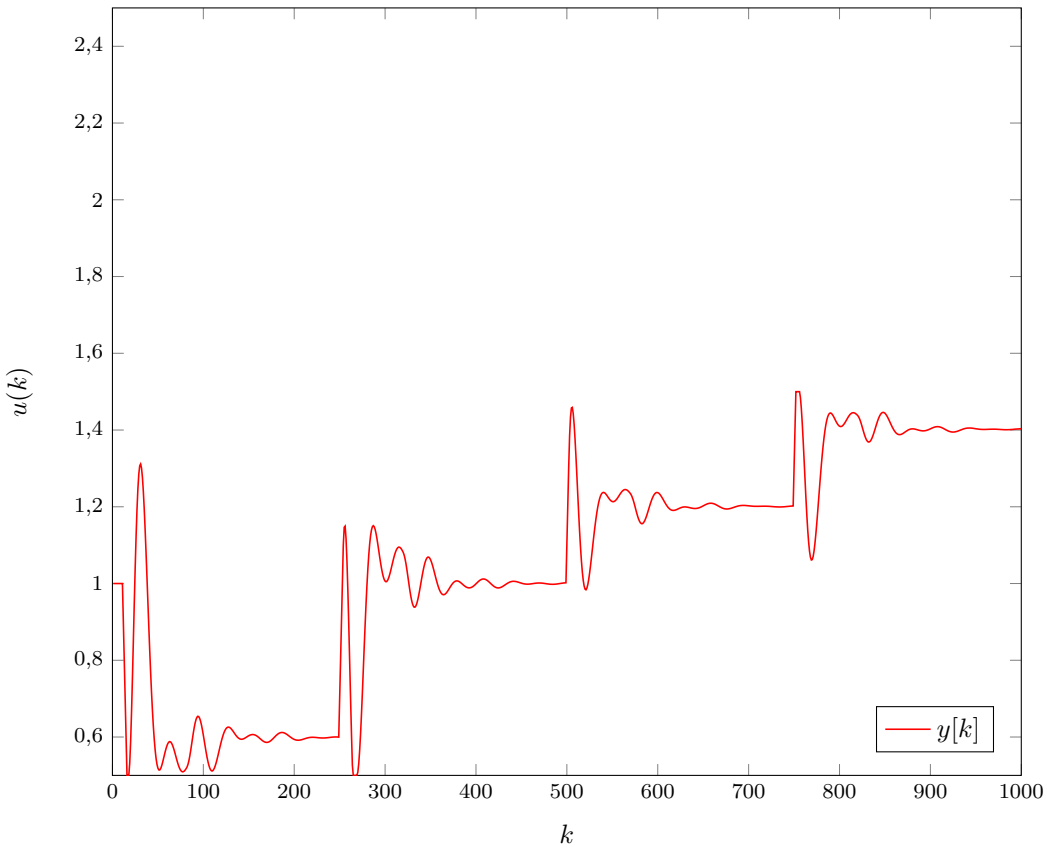
Rys. 7.9. Trajektoria wyjścia dla $D = 70$, $N = 80$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}9225 \cdot 10^1$



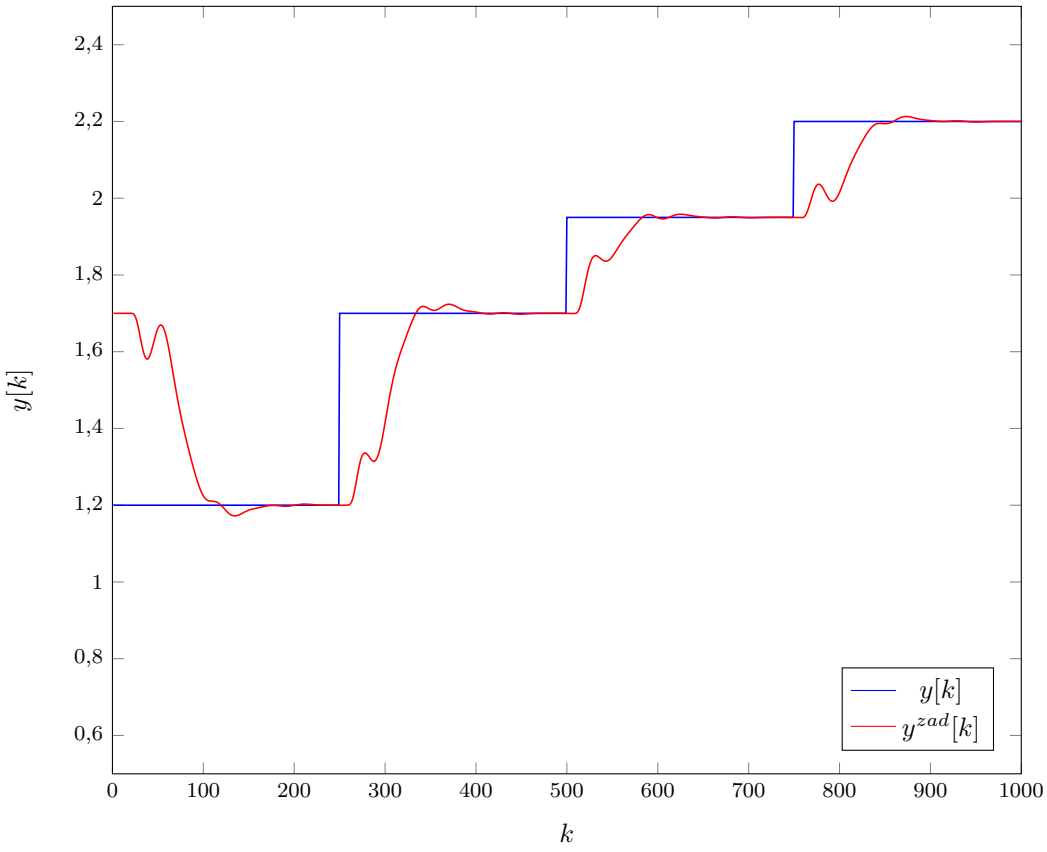
Rys. 7.10. Trajektoria sterowania dla $D = 70$, $N = 80$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}9225 \cdot 10^1$



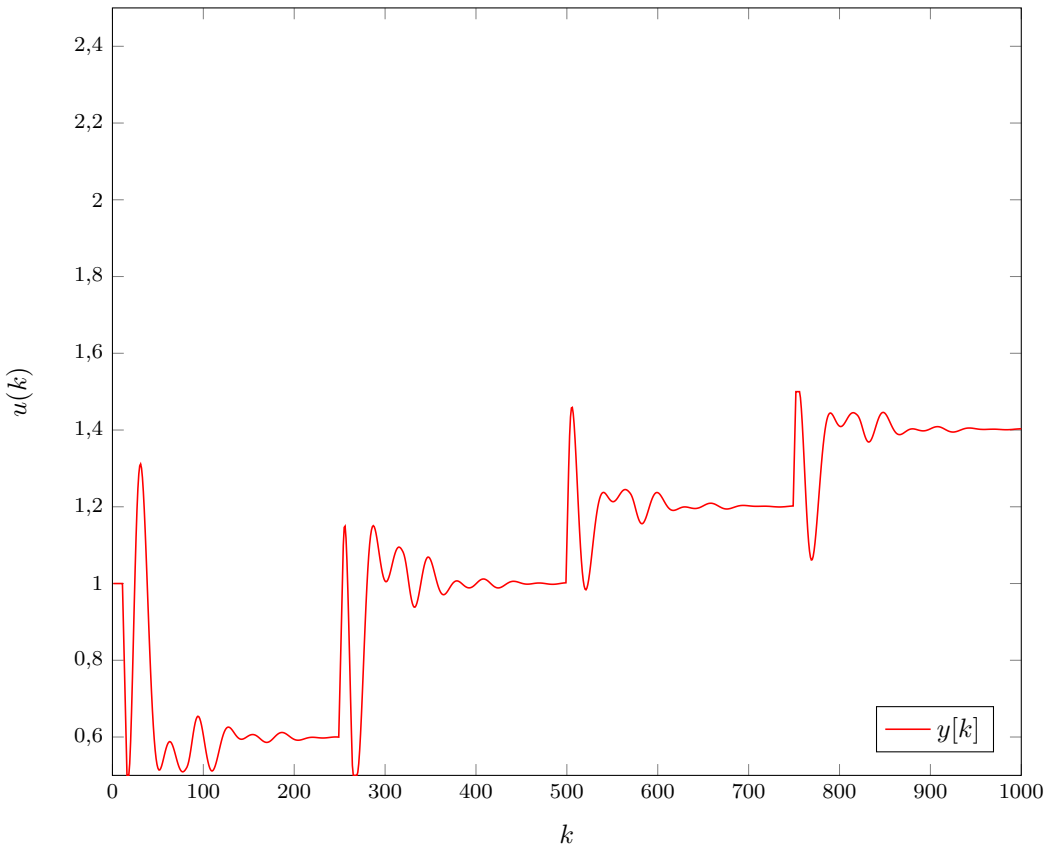
Rys. 7.11. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}874\,57 \cdot 10^1$



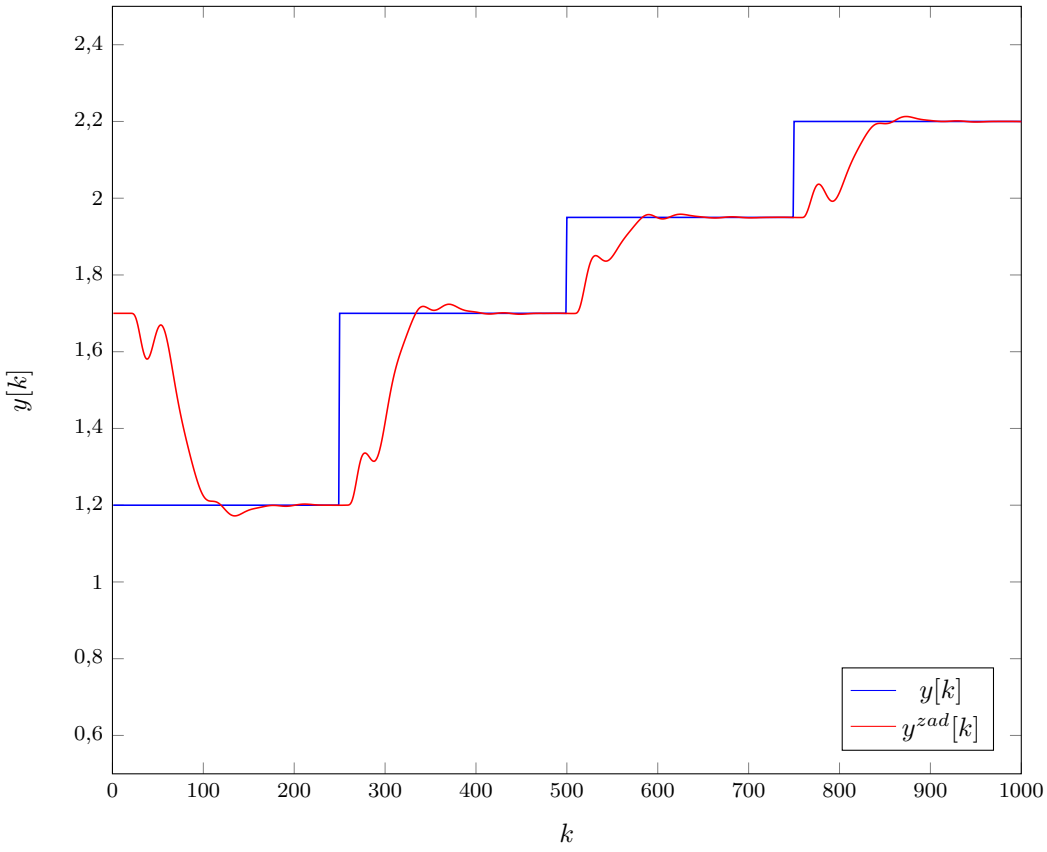
Rys. 7.12. Trajektoria sterowania dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 150$, $\lambda = 1$, $E = 2{,}874\,57 \cdot 10^1$



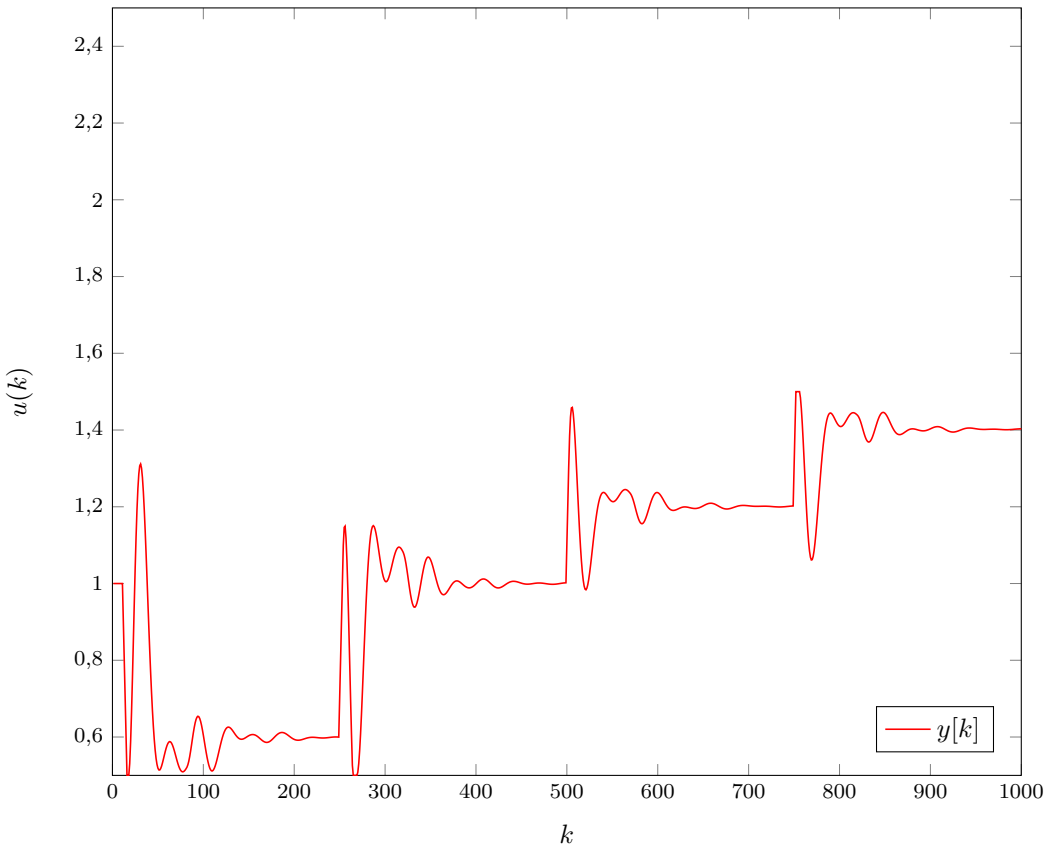
Rys. 7.13. Trajektoria wyjścia dla $D = 70$ $N = 30$, $N_{\mathrm{u}} = 100$, $\lambda = 1$, $E = 2{,}874\,57 \cdot 10^1$



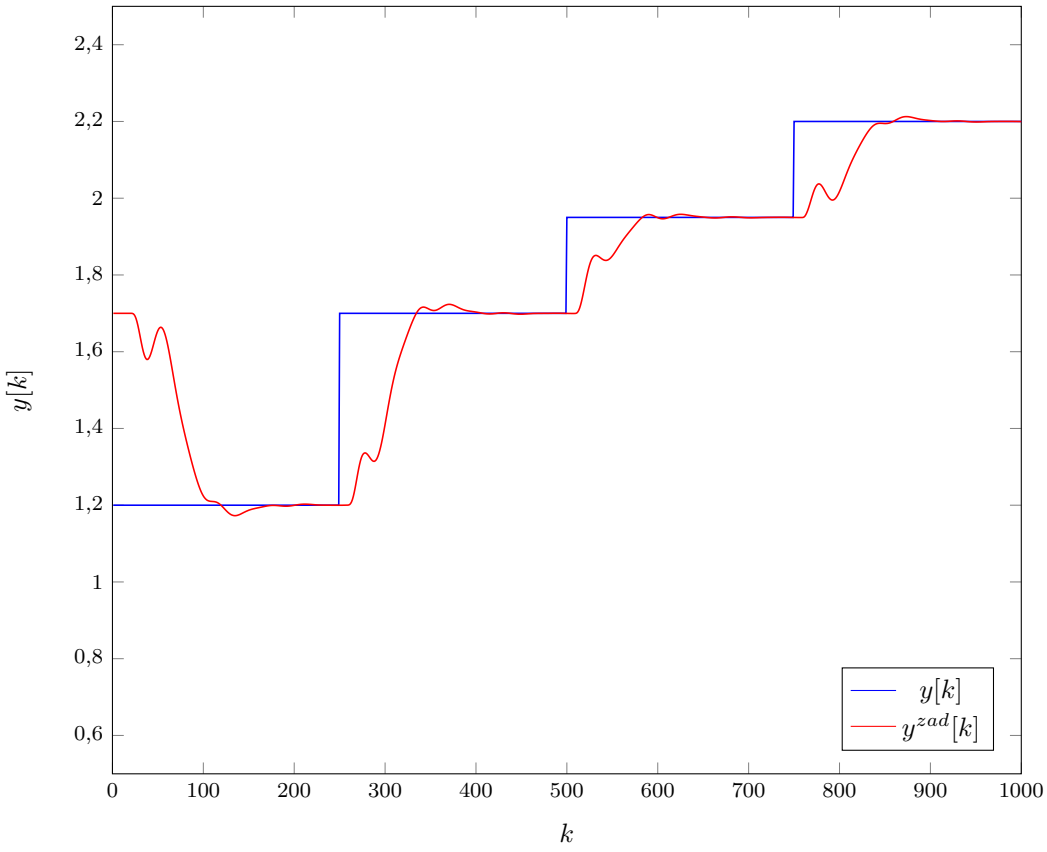
Rys. 7.14. Trajektoria sterowania dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 100$, $\lambda = 1$, $E = 2{,}874\,57 \cdot 10^1$



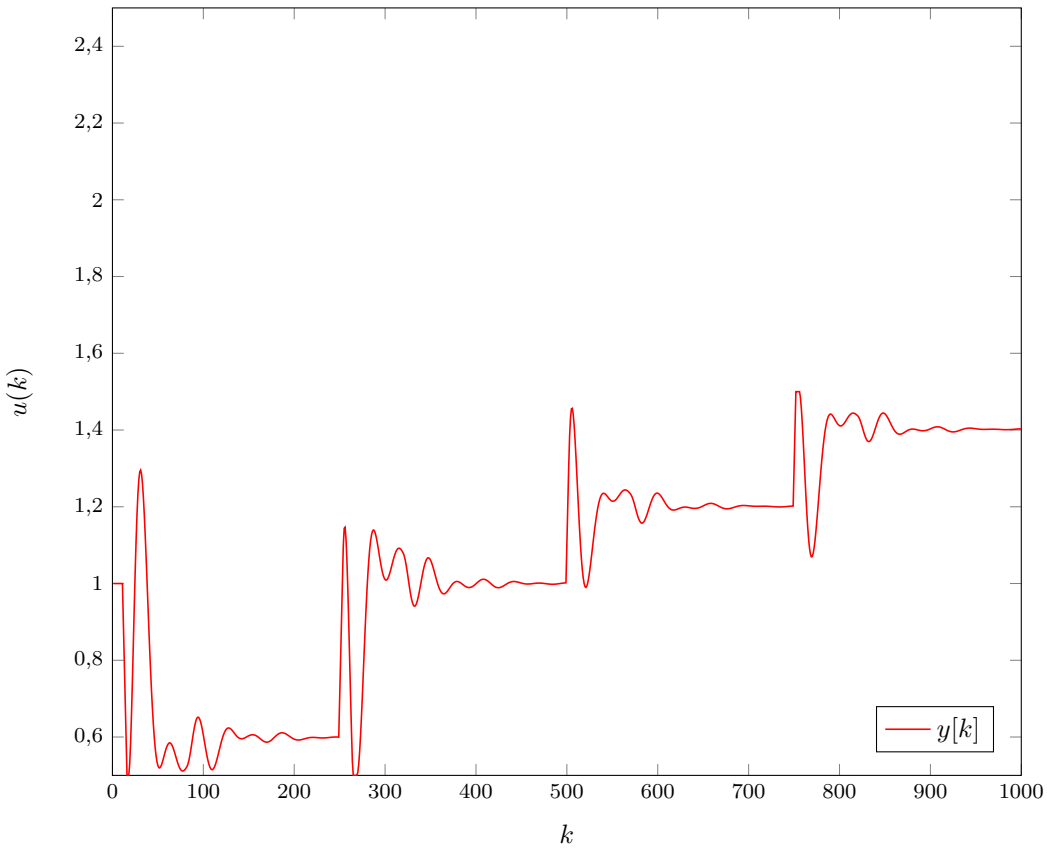
Rys. 7.15. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 80$, $\lambda = 1$, $E = 2{,}874\,57 \cdot 10^1$



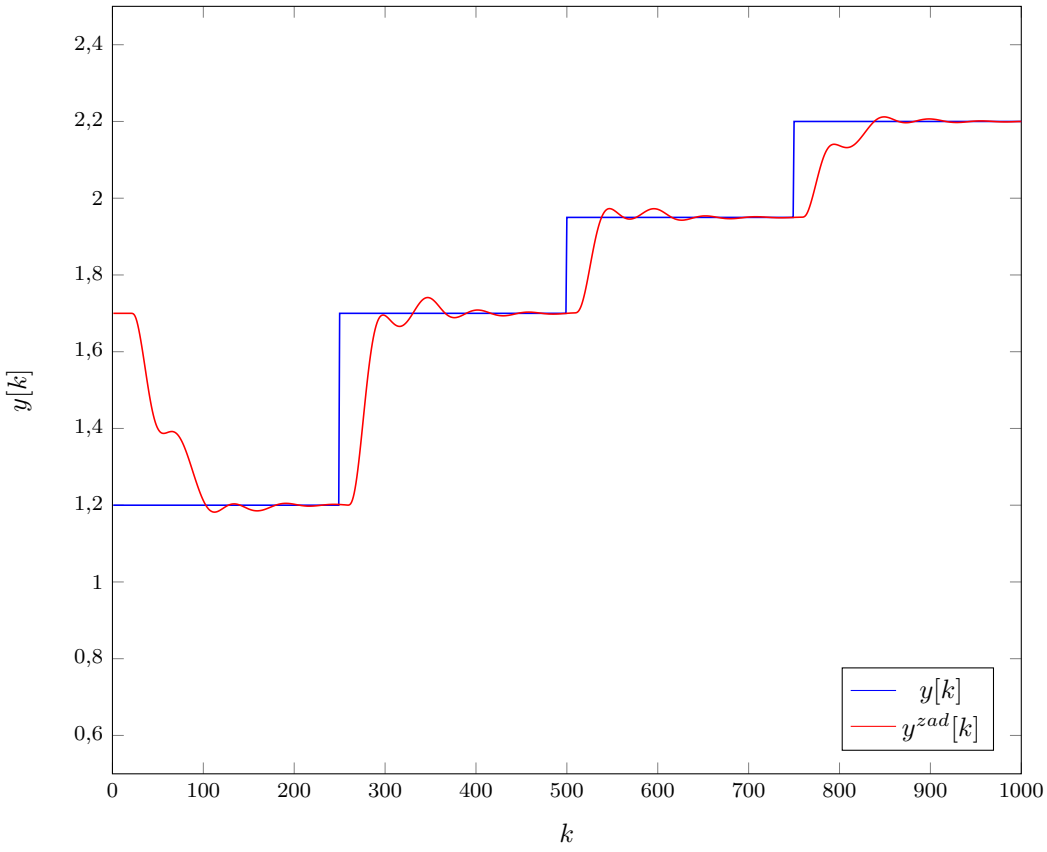
Rys. 7.16. Trajektoria sterowania dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 80$, $\lambda = 1$, $E = 2{,}874\,57 \cdot 10^1$



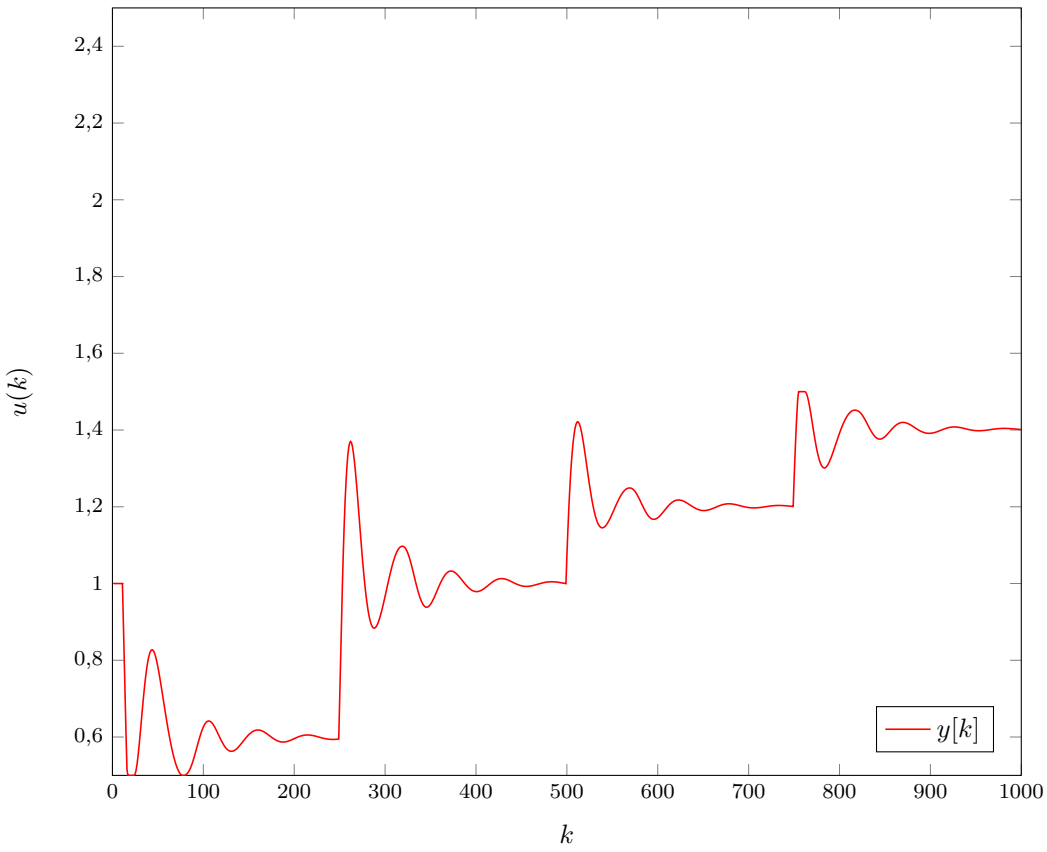
Rys. 7.17. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 1$, $E = 2{,}863\,53 \cdot 10^1$



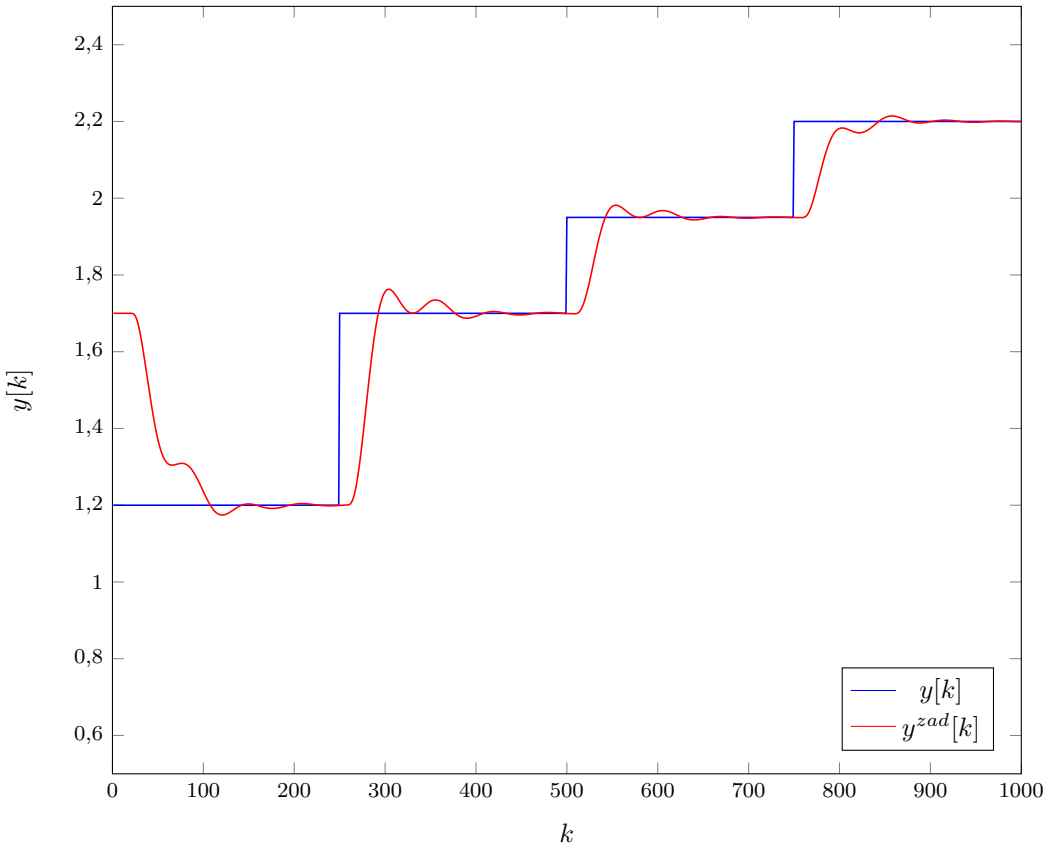
Rys. 7.18. Trajektoria sterowania dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 1$, $E = 2{,}863\,53 \cdot 10^1$



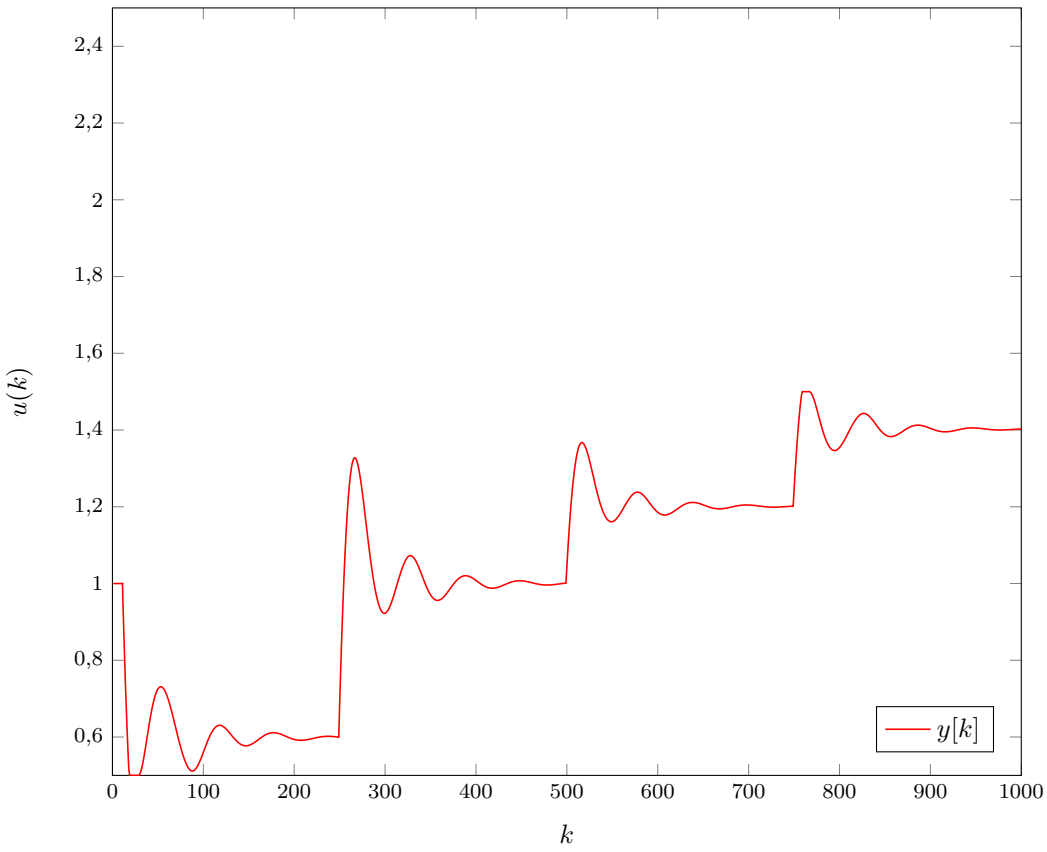
Rys. 7.19. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 10$, $E = 1{,}936\,98 \cdot 10^1$



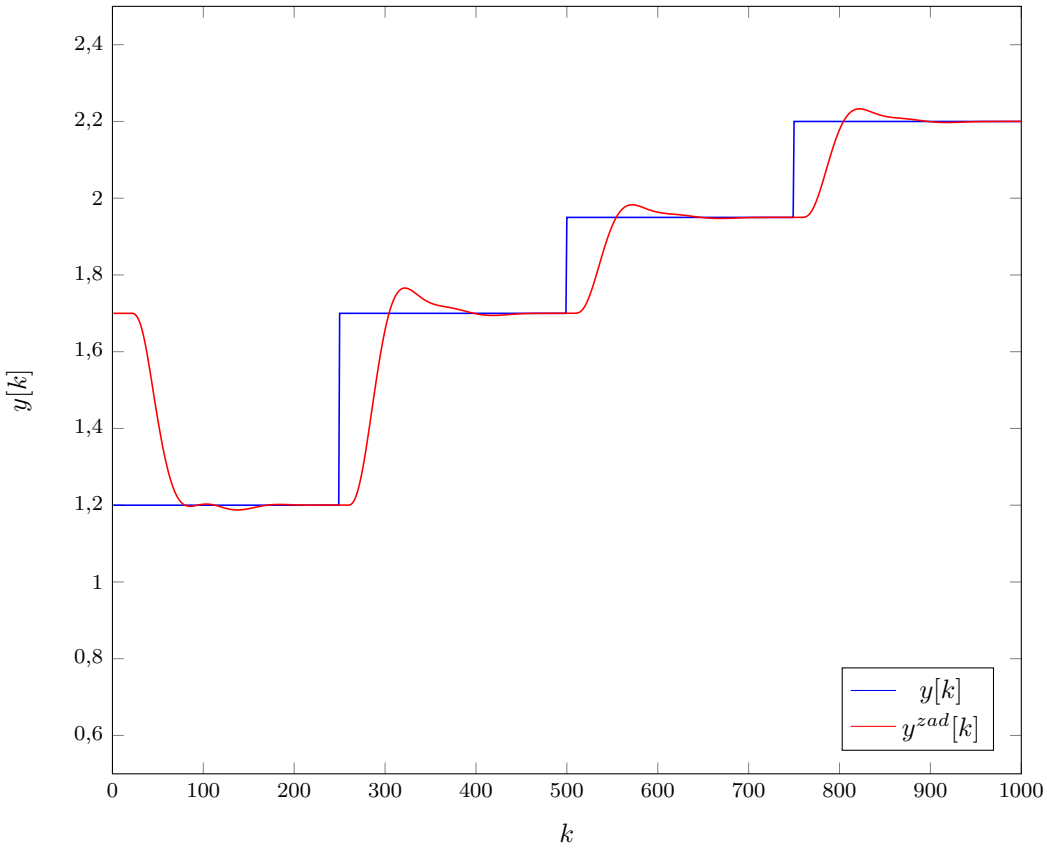
Rys. 7.20. Trajektoria sterowania dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 10$, $E = 1{,}936\,98 \cdot 10^1$



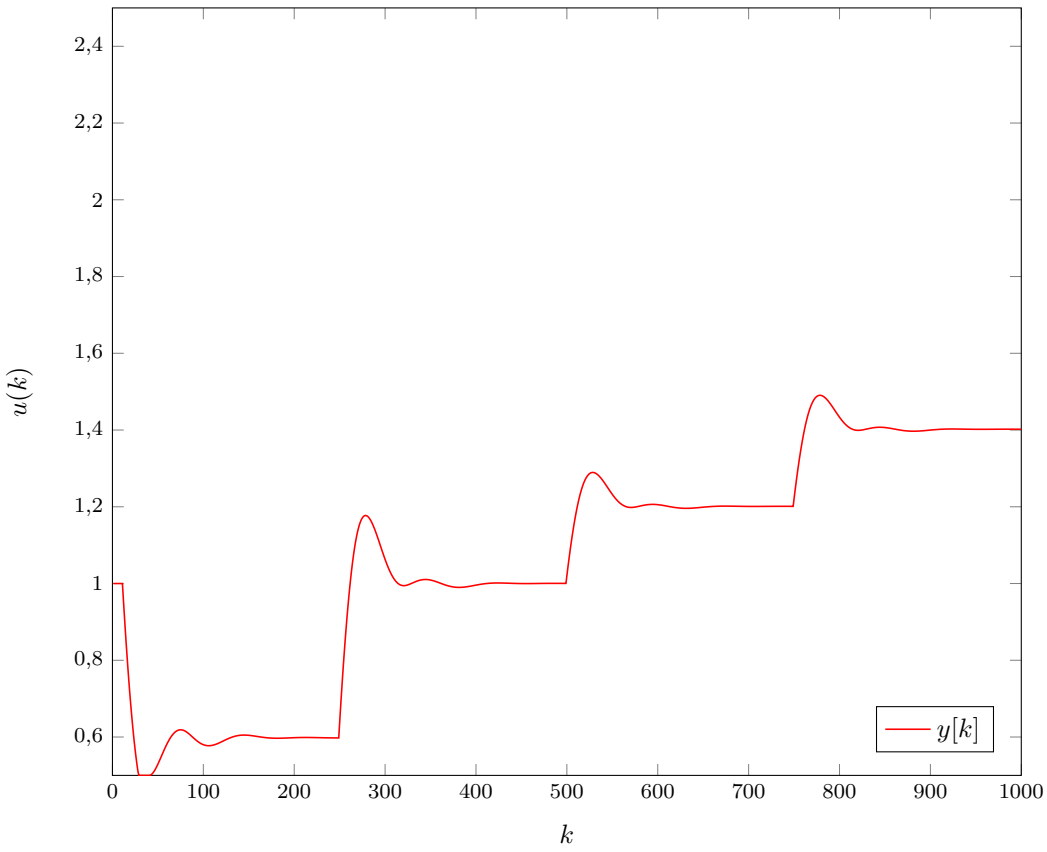
Rys. 7.21. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 20$, $E = 1{,}933\,72 \cdot 10^1$



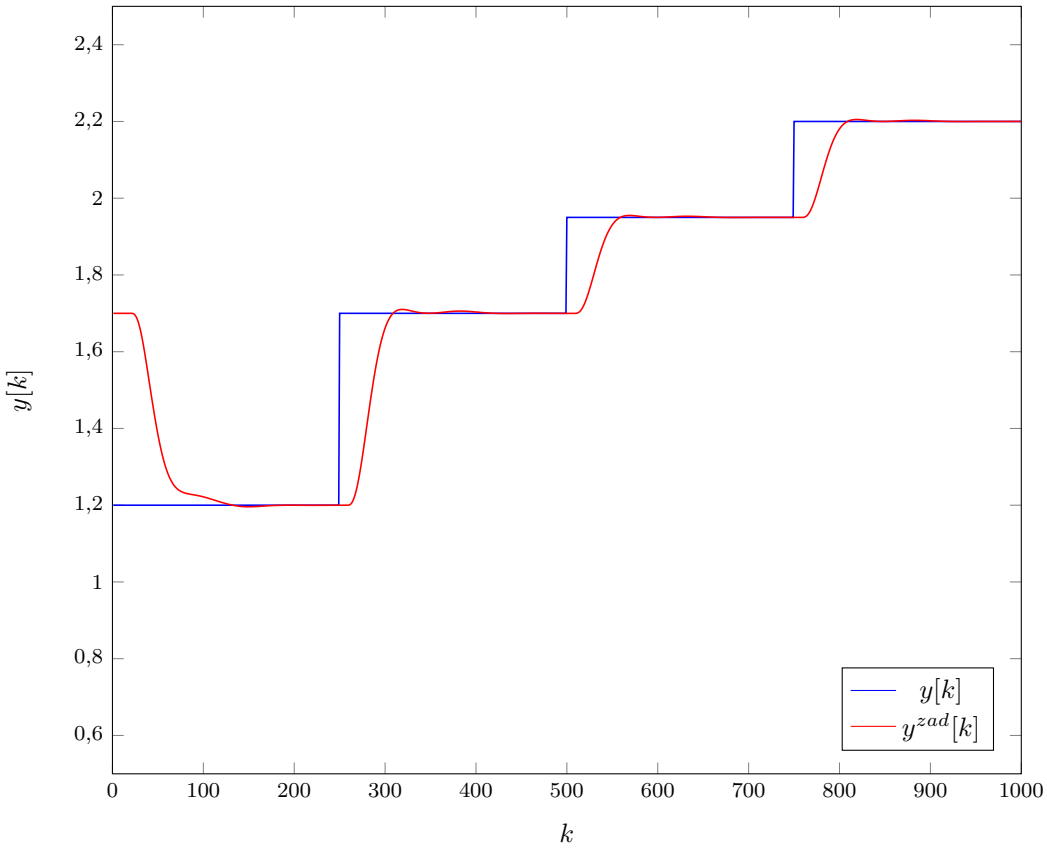
Rys. 7.22. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 20$, $E = 1{,}933\,72 \cdot 10^1$



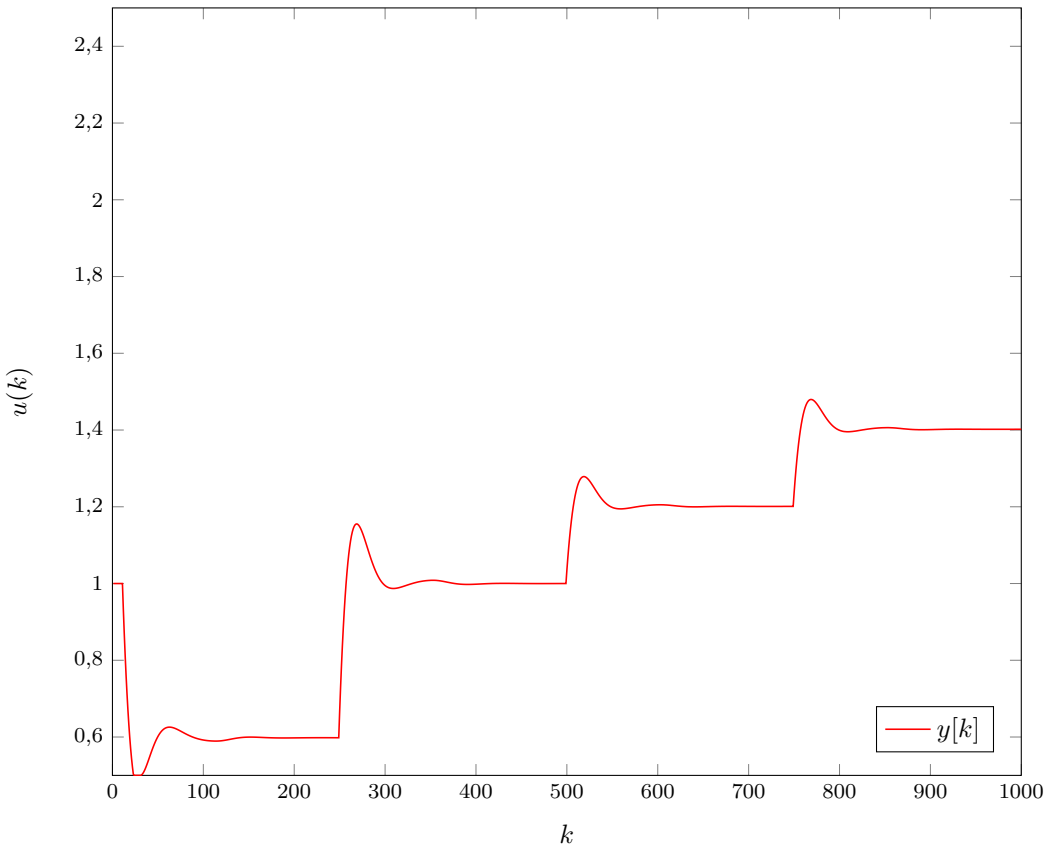
Rys. 7.23. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 50$, $E = 2{,}177\,89 \cdot 10^1$



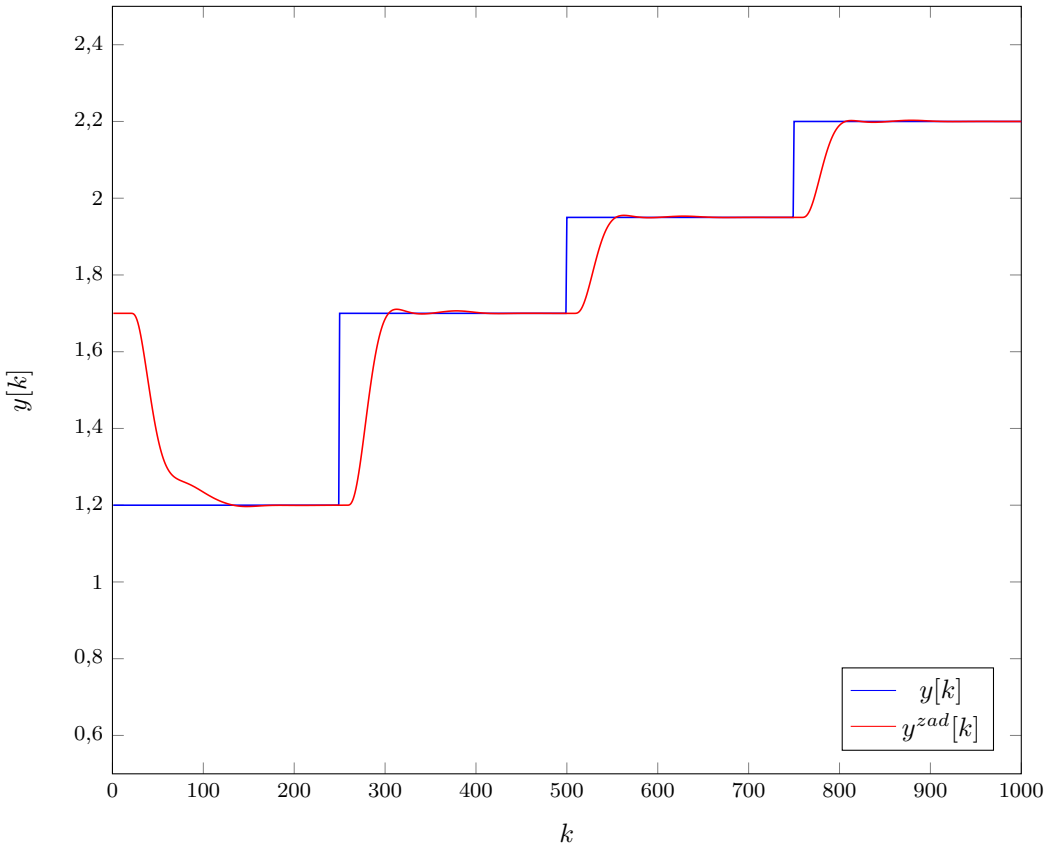
Rys. 7.24. Trajektoria wyjścia dla $D = 70$, $N = 30$, $N_{\mathrm{u}} = 10$, $\lambda = 50$, $E = 2{,}177\,89 \cdot 10^1$



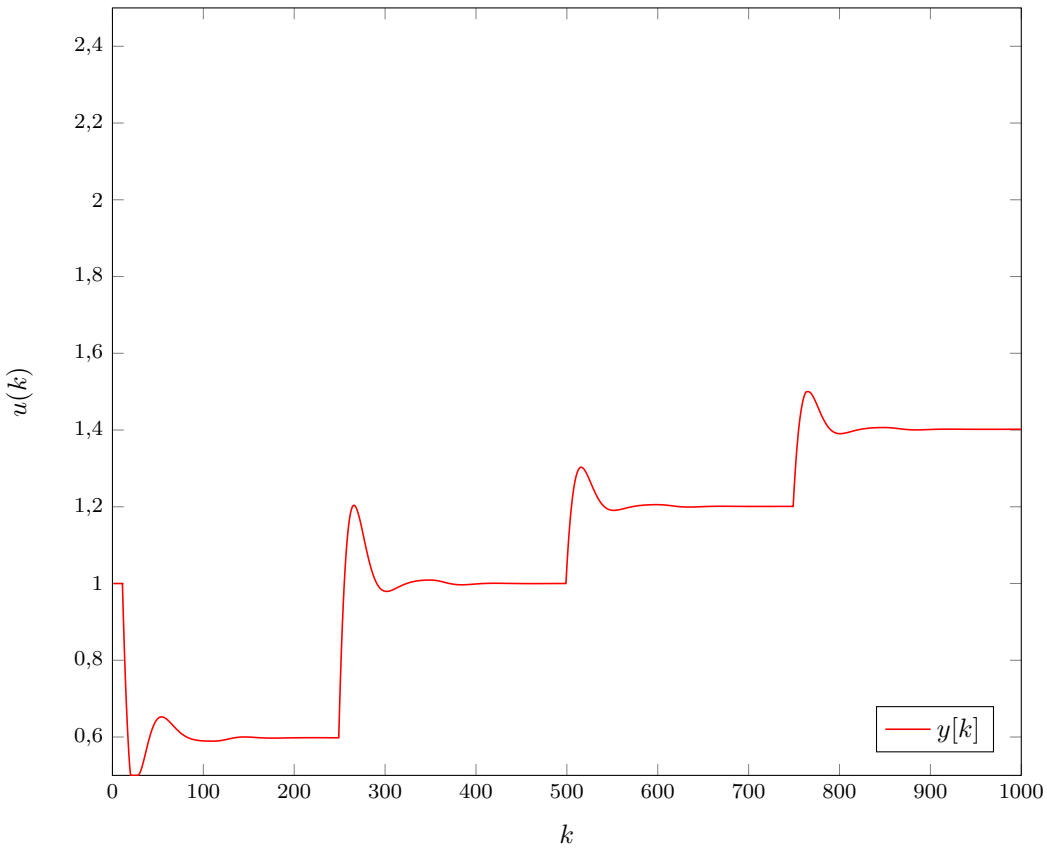
Rys. 7.25. Trajektoria wyjścia dla $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 50$, $E = 2{,}016\,62 \cdot 10^1$



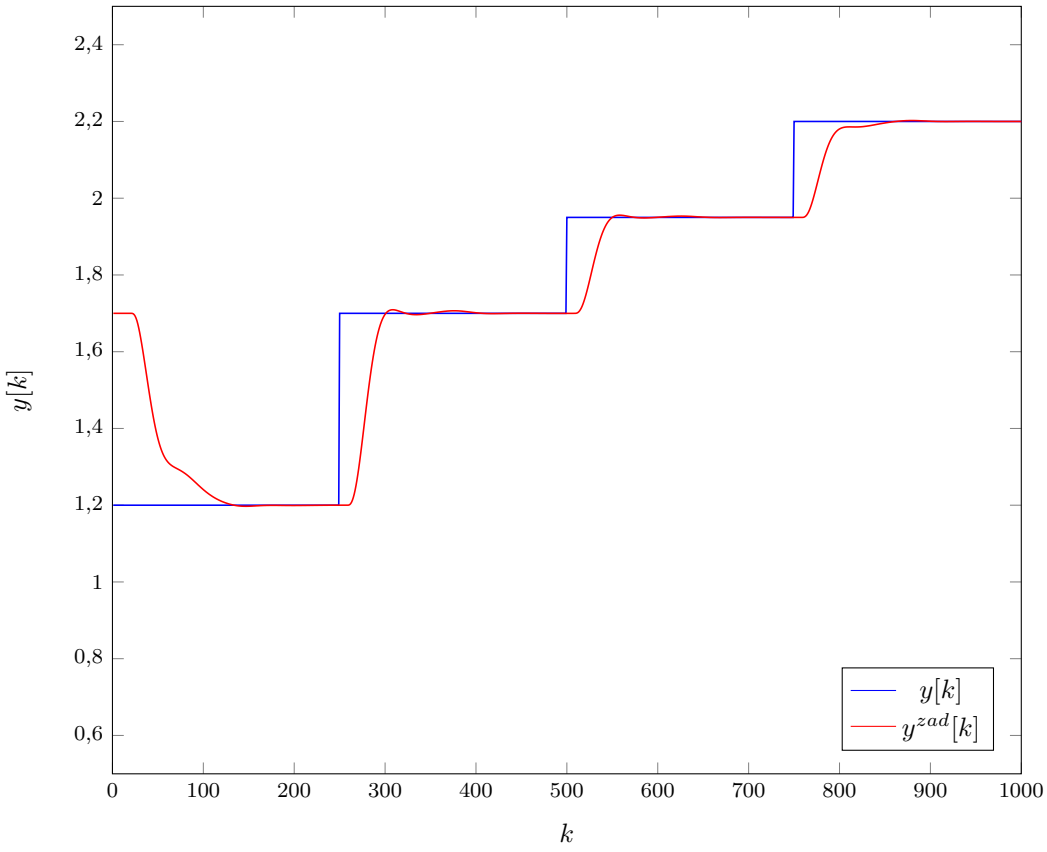
Rys. 7.26. Trajektoria sterowania dla $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 50$, $E = 2{,}016\,62 \cdot 10^1$



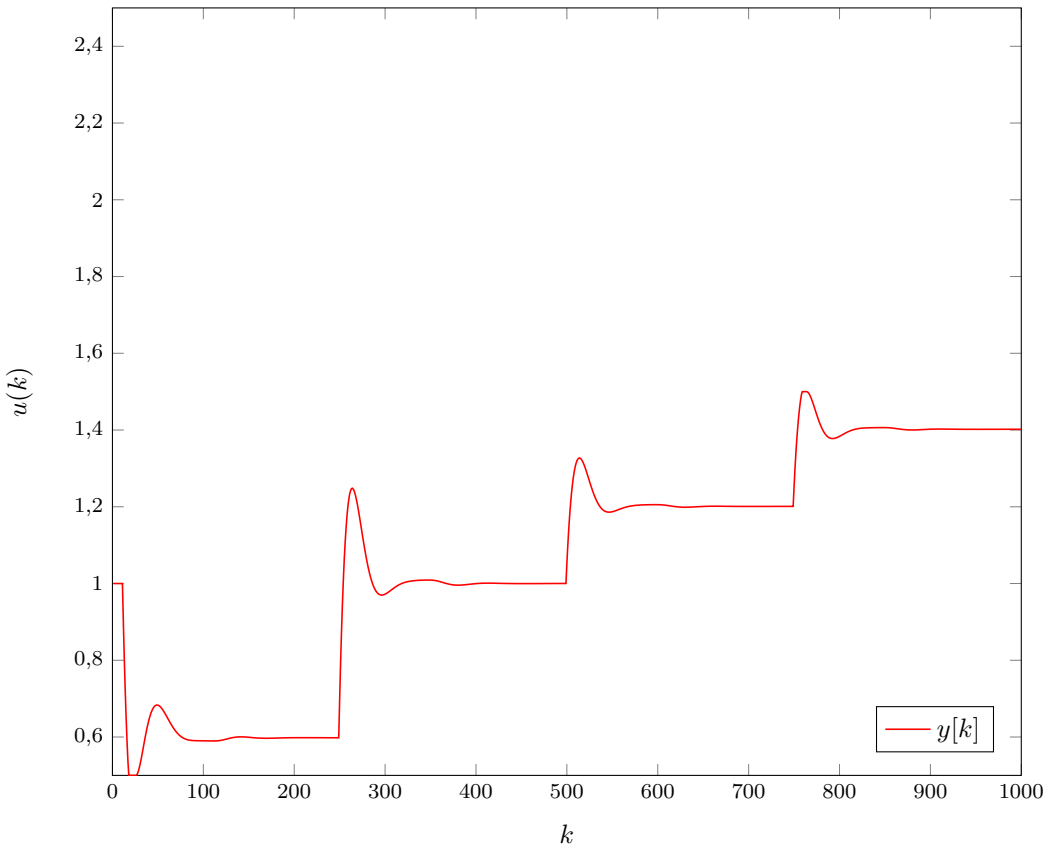
Rys. 7.27. Trajektoria wyjścia dla $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 30$, $E = 1{,}945\,22 \cdot 10^1$



Rys. 7.28. Trajektoria sterowania dla $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 30$, $E = 1{,}945\,22 \cdot 10^1$



Rys. 7.29. Trajektoria wyjścia dla $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 20$, $E = 1{,}908\,43 \cdot 10^1$



Rys. 7.30. Trajektoria sterowania dla $D = 100$, $N = 50$, $N_{\mathrm{u}} = 20$, $\lambda = 20$, $E = 1{,}908\,43 \cdot 10^1$

8. Metody optymalizacyjne regulatorów PID i DMC

8.1. Metody optymalizacyjne dostępne w programie MATLAB

Skrypty wykorzystywane do zrealizowania tego zadania: `zad6P_dmc.m`, `zad6P_dmcOptimalization.m`, `zad6P_dmcga.m`, `zad6P_pid.m`, `zad6P_pidOptimalization.m`, `zad6P_pidfmincon.m`.

MATLAB udostępnia swoim użytkownikom wiele funkcji służących do optymalizacji parametrów względem danej funkcji celu. W projekcie zostały wykorzystane dwie z nich: `fmincon` do optymalizacji nastaw regulatora PID.

TODO nie zaczynamy od nazwy funkcji `ga` do optymalizacji parametrów N, N_u oraz λ regulatora DMC. Wybór tej funkcji był podyktowany faktem, iż wyznaczane parametry musiały być całkowitoliczbowe - funkcja `fmincon` nie umożliwia wymuszenia tego warunku.

8.2. Implementacja

Do optymalizacji nastaw regulatora PID został użyty `fmincon` wraz z warunkiem zapisanym macierzami A i b. Punkt początkowy jest dobierany losowo, z faworyzacją nastaw bliskich 2,5

```
x0  = [rand*rand*10 rand*rand*10 rand*rand*10];
fun = @(variables) zad6P_pidOptimalization(variables);
A = [1 0 0; 0 -1 0;0 0 -1];
b = [10 ; 0.001 ; 0 ];
x = fmincon(fun,x0,A,b);
E = zad6P_pidOptimalization(x)
```

Listing 8.1. Wyznaczanie wskaźnika jakości

```
E = 0;
for i = 1:lenght
        E = E + ( y(i) - yZad(i) )^2;
end
```

Wywołanie funkcji `fmincon` oznacza następujące zadanie programowania nieliniowego: Minimalizacja wskaźnika jakości E. Warunki zadania programowania nieliniowego są następujące: $x_1 <= 10, x_2 >= 0{,}001 i x_3 >= 0$ odpowiednio K, T_i i T_d.

Do optymalizacji parametrów regulatora DMC została użyta funkcja `ga` który umożliwia warunek ograniczenia zmiennych do zbioru liczb całkowitych.

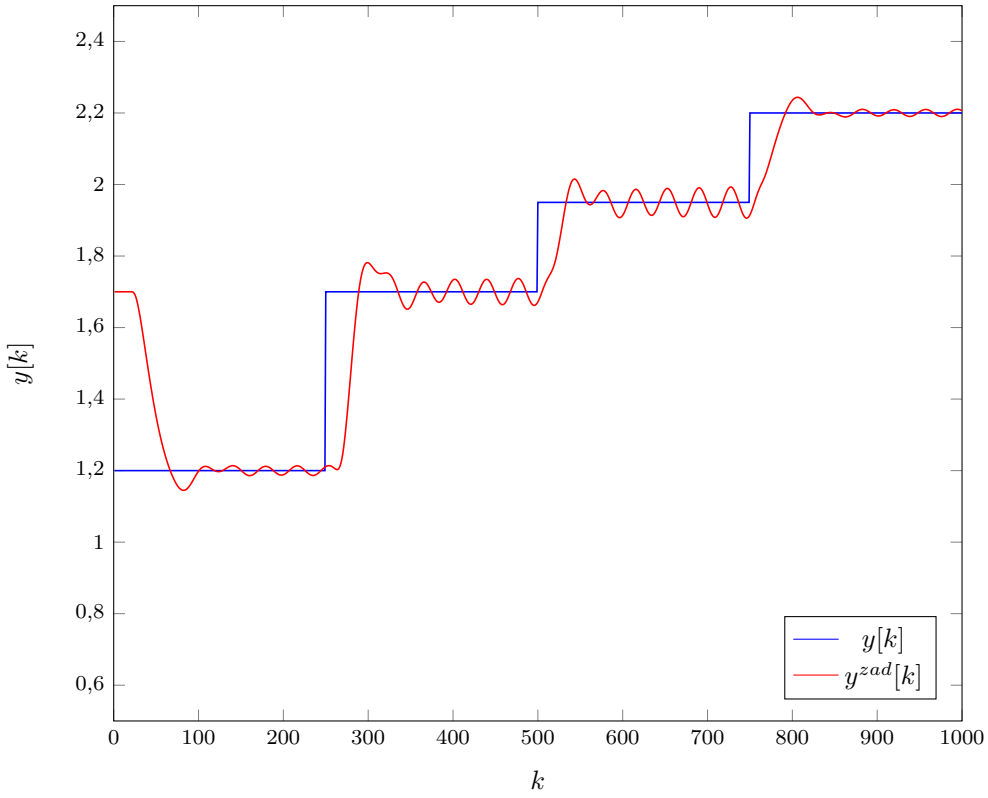
```
x0  = [2 2 0.01];
fun = @(variables) zad6P_dmcOptimalization(variables);
A = [1 0 0;0 1 0;0 0 1];
b = [100 ; 100 ; 10000];
x = ga(fun,3,A,b,[],[],[ 2 2 0.1 ], [ 150 150 10000],[],[1 2 3]);
E = zad6P_dmcOptimalization(x)
```

Wywołanie funkcji gaoznacza następujące zadanie minimalizacji wskaźnika jakości

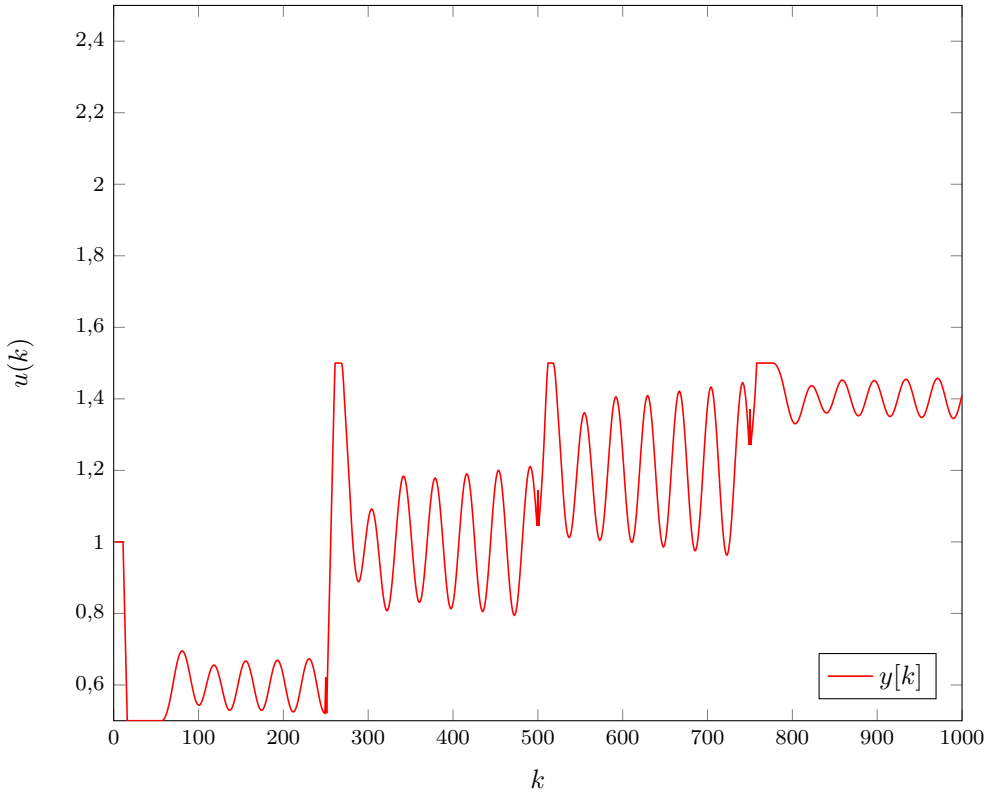
To jest minimalizacja funkcji `fun`, z 3 zmiennymi, ograniczenia liniowe: $_1 < 100, x + 2 < 100ix_3 < 10\,000$, bez ograniczeń równościowych, dolne granice to: $2, 2, 0{,}1$, natomiast górne granice to: $150, 150, 10\,000$, bez ograniczeń nieliniowych.

8.3. Wykresy

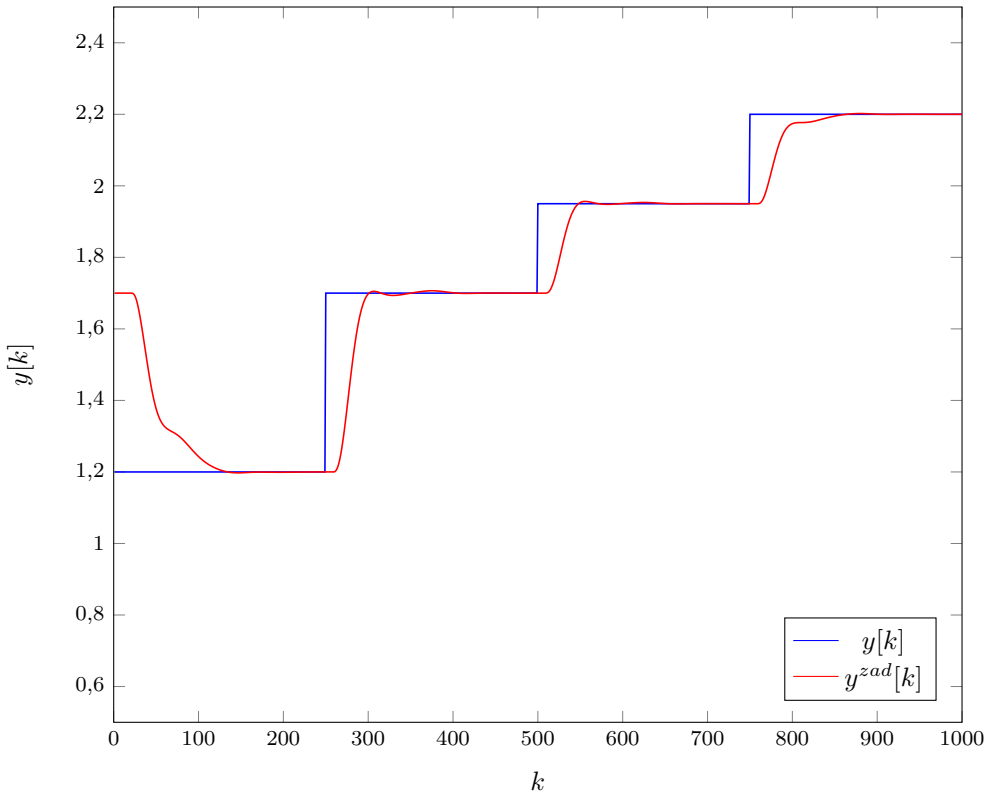
Na poniższych wykresach zostały przedstawione wyjścia obiektu, które parametry zostały wyznaczone przy pomocy metod optymalizacji.



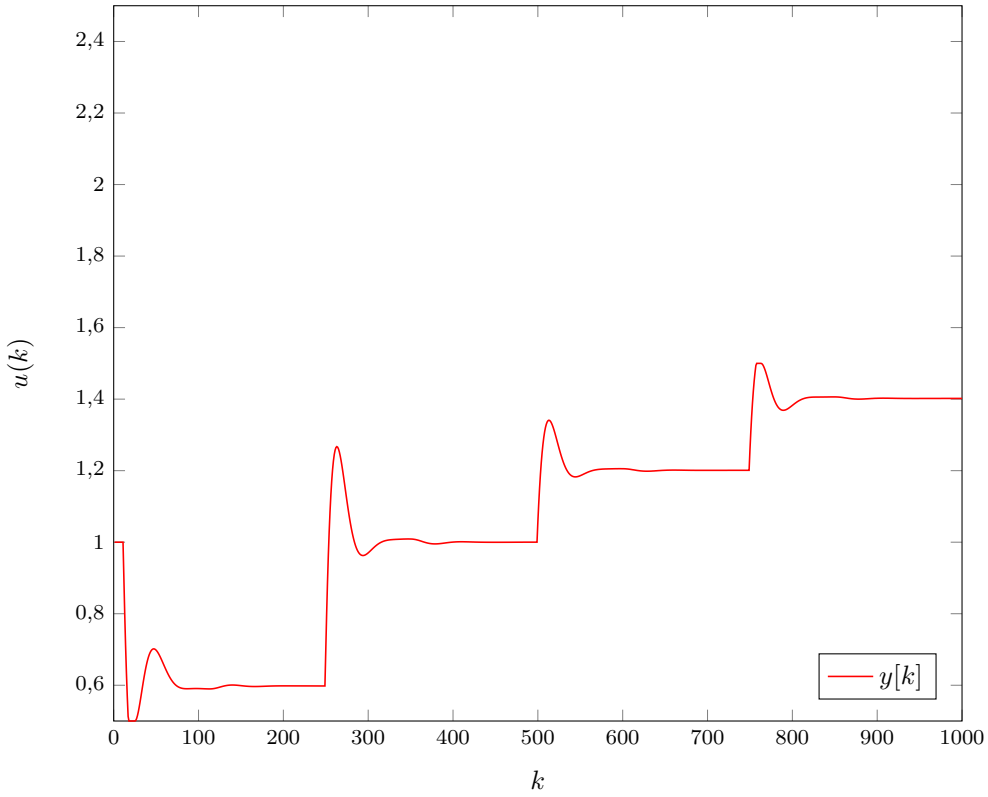
Rys. 8.1. Trajektoria wyjścia dla regulatora PID z nastawami wyznaczonymi przez funkcję *fmincon*



Rys. 8.2. Trajektoria sterowania dla regulatora PID z nastawami wyznaczonymi przez funkcję *fmincon*



Rys. 8.3. Trajektoria wyjścia dla regulatora DMC z nastawami wyznaczonymi przez funkcję *ga*



Rys. 8.4. Trajektoria sterowania dla regulatora DMC z nastawami wyznaczonymi przez funkcję *ga*

Część II

Laboratoria

9. Obiekt laboratoryjny

9.1. Test komunikacji

Laboratorium rozpoczęliśmy od napisania programu, który wysyła kilka różnych wartości sterowania dla wiatraka i grzałki. Stwierdziliśmy, że wszystko działa poprawnie i przeszliśmy do kolejnej części zadania. Do komunikacji użyliśmy funkcji opisanych w skrypcie do laboratoriów. Port, który został użyty do komunikacji z obiektem to COM19. Funkcja `sendControl` służy do wysyłania komunikatów do obiektu. Pierwsza wartość to sterowanie wiatrakiem (podawana w procentach, z warunków zadania 50 %). Drugą wartością jest aktualna wartość sterowania. Funkcja `readMeasurements` służy do odbierania komunikatów z obiektu. W naszym przypadku została użyta do odczytywania obecnego sygnału wyjściowego $Y(k)$.

Listing 9.1. Realizacja komunikacji w MATLAB

```
addpath ('F:\SerialCommunication');
% inicjalizacja portu
initSerialControl COM19;
% funkcja wysylajaca wartosci sterowania do obiektu
sendControls ([ 1, 5], [ 50, U(k)]);
% funkcja pobierajaca wartosci sygnalu wyjsciowego
Y(k) = readMeasurements (1)
% czekanie na kolejna iteracje
waitForNewIteration ();
```

9.2. Wyznaczenie punktu pracy

Według zaleceń prowadzącego U_{pp} zostało przyjęte jako 25 + numer stanowiska, w naszym przypadku U_{pp} było równe 33. Następnie dobrana została moc wiatraka jako 50% (z treści zadania). Obie wartości zostały wysłane do stanowiska grzejąco-chłodzącego. Po ustabilizowaniu się obiektu odczytaliśmy wyjście obiektu - Y_{pp} jako 36,06. Warto zauważyć, że w algorytmie regulacji punkt pracy traktujemy jako wartość zerową. Wynika z tego, że wartość jaką odczytujemy z obiektu - $Y(k)$ musimy pomniejszyć o Y_{pp}. To samo musimy zrobić z wartością zadaną. Następnie wysyłając do obiektu wartość sterowania musimy ją analogicznie zwiększyć o U_{pp}.

10. Odpowiedzi skokowe procesu dla trzech różnych zmian sygnału sterującego

10.1. Opis

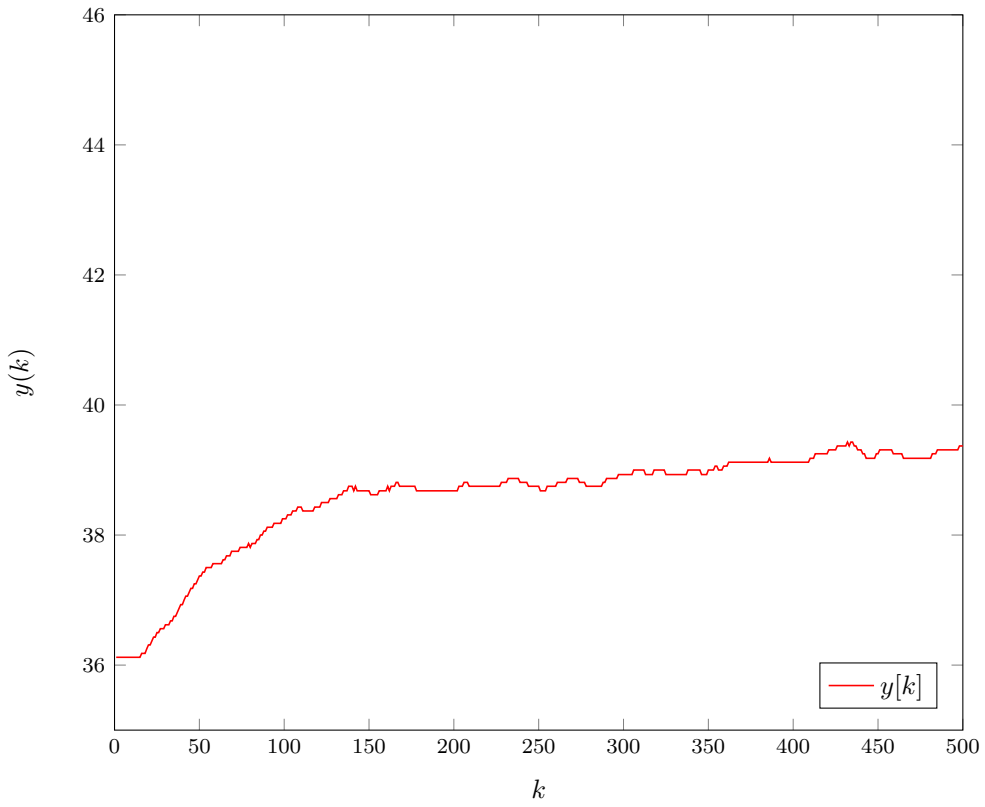
Badanie odpowiedzi skokowej rozpoczęliśmy od sprowadzenia obiektu do punktu pracy. Później zwiększyliśmy wartość sterowania kolejno o 10, 20 i 30. Symulacje przeprowadziliśmy dla 600 próbek, ponieważ po tym czasie wszystkie odpowiedzi skokowe przyjmowały stałą wartość.

10.2. Wykresy odpowiedzi skokowych dla różnych zmian sterowania

Skrypt wykorzystywany do zrealizowania tego zadania: `odpowiedzSkokowa.m`.

10.2.1. Zmiana wartości sterowania o $\Delta u = 10$

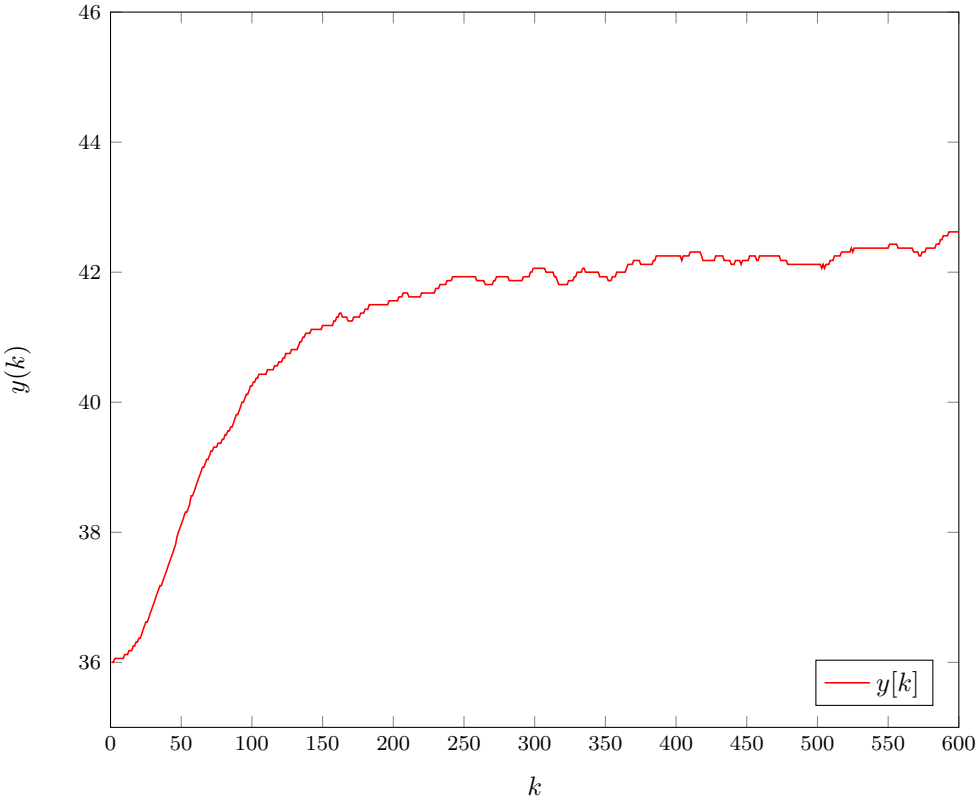
Skok wartości sterowania został wykonany o 10 z punktu pracy. Wzięliśmy tylko 500 próbek, bo po tym czasie sygnał wyjściowy ustawił się na względnie stałej wartości. Zostało to przedstawione na wykresie 10.1.



Rys. 10.1. Wykres zmiany wartości sterowania o $\Delta u = 10$

10.2.2. Zmiana wartości sterowania o $\Delta u = 20$

Skok wartości sterowania został wykonany o 20 z punktu pracy. W tym wypadku musieliśmy przedłużyć czas symulacji o 100 próbek, ponieważ wyniki tylko dla 500 próbek były niezadowalające i nie mogliśmy jednoznacznie stwierdzić czy wyjście układu się ustabilizowało. Zostało to przedstawione na wykresie 10.2.

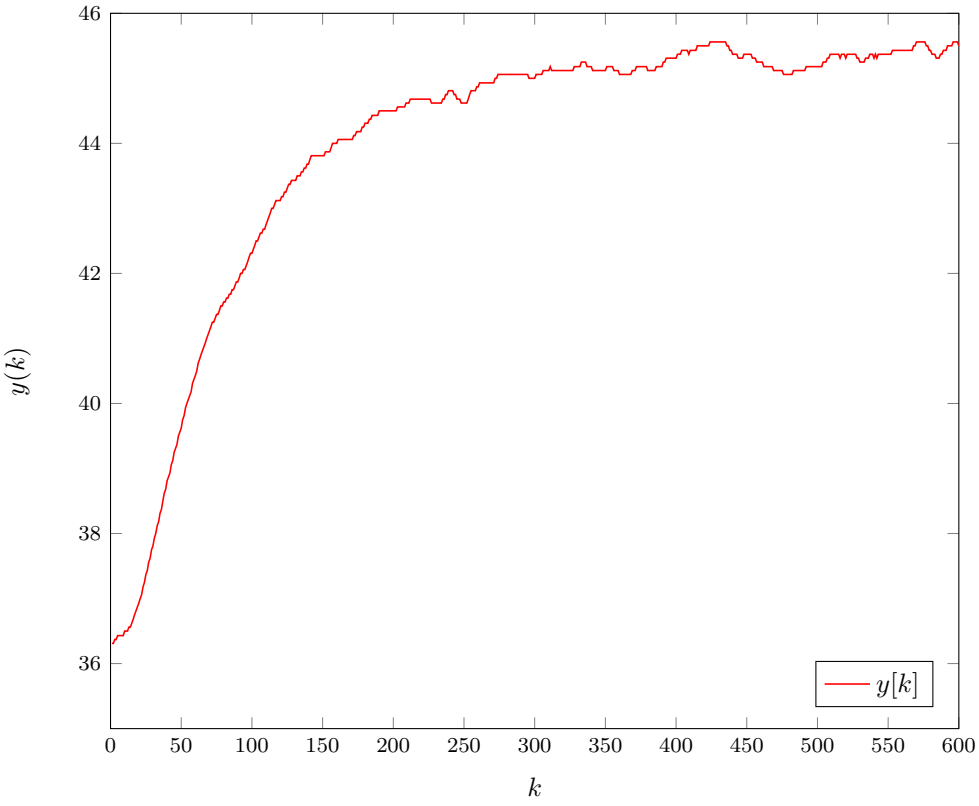


Rys. 10.2. Wykres zmiany wartości sterowania o $\Delta u = 20$

10.2.3. Zmiana wartości sterowania o $\Delta u = 30$

Skok wartości sterowania został wykonany o 30 z punktu pracy. Symulacje również przeprowadziliśmy dla 600 próbek. Zostało to przedstawione na wykresie 10.3.

Z przedstawionych trzech odpowiedzi skokowej wybraliśmy odpowiedź, gdzie zmiana wartości sterowania wyniosła $\Delta u = 20$ (według nas jest ona najdokładniejsza).



Rys. 10.3. Wykres zmiany wartości sterowania o $\Delta u = 30$

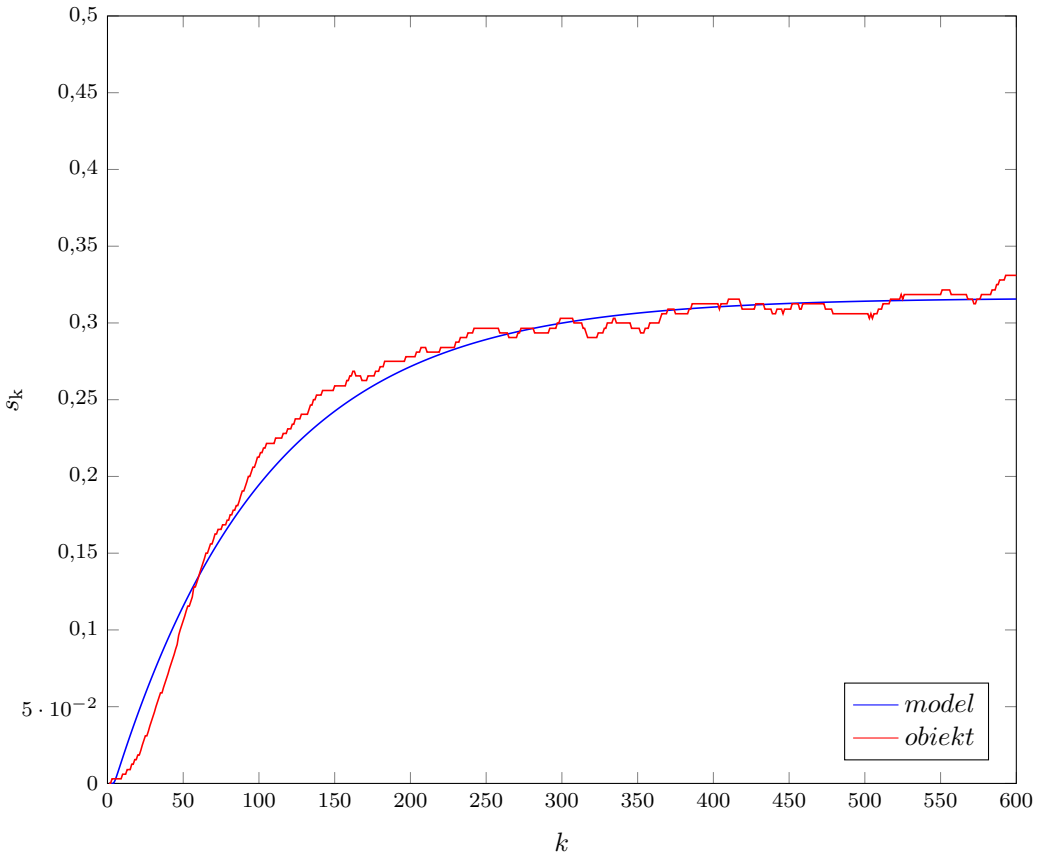
11. Odpowiedź skokowa do algorytmu DMC

11.1. Przekształcenie odpowiedzi skokowej

Skrypt wykorzystywany do zrealizowania tego zadania: `zad3L_aproksymacja.m`.

Do algorytmu DMC wykorzystaliśmy odpowiedź skokowa przy zmianie sterowania o $\Delta u = 20$. Na początku wektor odpowiedzi skokowej musiał zostać przesunięty z punktu pracy do wartości 0. Następnie cały wektor został podzielony przez wartość 20 (Δu).

11.2. Aproksymacja odpowiedzi skokowej



Rys. 11.1. Porównanie aproksymowanej odpowiedzi skokowej z rzeczywistą odpowiedzią skokową

Odpowiedź skokowa została aproksymowana jako człon inercyjny drugiego rzędu z opóźnieniem, który po przekształceniach wyraża się wzorem:

$$y(k) = b_1 * u(k - T_D - 1) + b_2 * u(k - T_D - 2) - a_1 * y(k - 1) - a_2 * y(k - 2) \tag{11.1}$$

gdzie:

$$a_1 = -\alpha_1 - \alpha_2 \tag{11.2}$$

$$a_2 = \alpha_1 * \alpha_2 \tag{11.3}$$

$$\alpha_1 = e^{-\frac{1}{T_1}} \tag{11.4}$$

$$\alpha_2 = e^{-\frac{1}{T_2}} \tag{11.5}$$

$$b_1 = \frac{K}{T_1 - T_2}[T_1 * (1 - \alpha_1) - T_2 * (1 - \alpha_2)] \tag{11.6}$$

$$b_2 = \frac{K}{T_1 - T_2}[\alpha_1 * T_2 * (1 - \alpha_2) - \alpha_2 * T_1 * (1 - \alpha_1)] \tag{11.7}$$

Parametry K, T_D, T_1 i T_2 zostały dobrane tak aby suma kwadratów błędów była jak najmniejsza.

11.3. Wzmocnienie statyczne procesu

Zebraliśmy trzy odpowiedzi skokowe z punktu pracy. Na podstawie wykresów odpowiedzi skokowej doszliśmy do wniosku, iż obiekt można traktować jak obiekt liniowy. Korzystając ze wzoru 2.2 wyznaczyliśmy wzmocnienie. Dla znormalizowanej odpowiedzi skokowej wartość sygnału wyjściowego ustabilizowała się na 0,331, więc wzmocnienie statyczne jest równe tej wartości.

12. Regulacja cyfrowego algorytmu PID oraz algorytmu DMC

12.1. Algorytm PID

Algorytm implementowany na laboratoriach był bardzo podobny jaki implementowaliśmy na projekcie. Zmiany głównie były zauważalne w szczegółach. Z tego powodu szczegółowa implementacja jest omówiona w **??**. W tym rozdziale wskażemy różnice pomiędzy kodem laboratoryjnym oraz kodem wykorzystywanym w projekcie.

W przypadku laboratorium mieliśmy inne parametry podane. W tym celu zadeklarowaliśmy je tak jak w poniższym przykładzie.

Listing 12.1. Deklaracja stałych

```
Ypp = 36.06;
Upp = 33;

UMin = 0;
UMax = 100;
uMin = UMin-Upp;
uMax = UMax-Upp;
```

Inną różnicą był fakt, iż na laboratoriach do odczytu wartości i wysłania wartości sygnałów korzystaliśmy z gotowych funkcji.

Listing 12.2. Realizacja algorytmu PID w MATLAB

```
for k = start : 1 : simulationTime
    % odczytanie wartosci
    Y(k) = readMeasurements (1) ;
    y(k) = Y(k)- Ypp;

    % obliczenie bledu
    errorR0 = yZad(k) - y(k);

    % obliczenie sygnalu sterujacego
        u(k) = u(k-1) + r0 * errorR0 + r1 * errorR1 + r2 * errorR2;

    % ograniczenia
    du = u(k)-u(k-1);

    % prawo regulacji
    u(k) = u(k-1) + du;

    if u(k)> uMax
                u(k) = uMax;
```

```
    end
    if u(k)< uMin
        u(k) = uMin;
    end


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

    % wyslanie sygnalow
    sendControls ([ 1, 5], [ 50, U(k)]) ;

    % obliczenie bledow
    errorR2 = errorR1;
    errorR1 = errorR0;

    % czekanie na nowa iteracje
    waitForNewIteration ();
end
```

12.2. Algorytm DMC

Podobne małe różnice pomiędzy kodem laboratoryjnych a kodem z projektu wynikały z innych deklaracji stałych oraz wykorzystaniu gotowych funkcji do odczytu wartości. Ponadto na laboratorium nie ograniczaliśmy różnic wartość sygnałów sterujących, a jedynie wartość sygnału sterującego. Wszystkie funkcje w poniższym listingu zostały omówione w rozdziale 5.

Listing 12.3. Realizacja algorytmu DMC w MATLAB

```
for k = start : 1 : simulationTime
    % odczytanie wartosci
    Y(k) = readMeasurements (1);
    y(k) = Y(k)- Ypp;

        du = dmc(dmcMacierze,y(k),yZad(k),duPop);


    % Prawo regulacji
    u(k) = u(k-1) + du;

    % ograniczenia
    if u(k)> uMax
                u(k) = uMax;
    end
    if u(k)< uMin
        u(k) = uMin;
    end
    dureal = u(k) - u(k-1);
    duPop = duNew(duPop,dureal);

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

```
    sendControls ([ 1, 5], [ 50, U(k)]);
    % odczekania na nowa iteracje
    waitForNewIteration ();
end
```

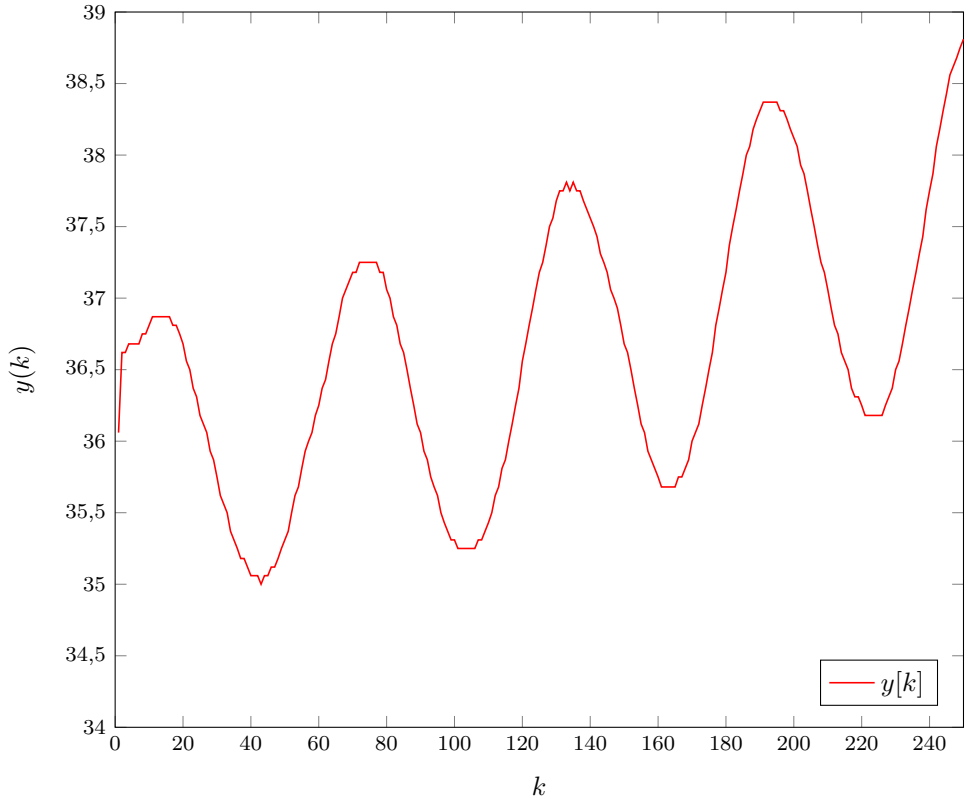
13. Dobór nastawów regulatora PID i parametrów algorytmu DMC metodą eksperymentalną

13.1. Metody doboru nastawów regulatora PID

Skrypty wykorzystywane do zrealizowania tego zadania: `pid.m`, `pidScript.m`.

13.1.1. Metoda Zieglera-Nicholsa

Metoda Zieglera-Nicholsa została opisana w 6.1.1. Uzyskane wzmocnienie krytyczne to $K_{kryt} = 40$, a okres oscylacji $T_\mathrm{k} = 60$. Przy użyciu tabeli 6.1, możemy odczytać wartości regulatora jako $K = 24$, $T_\mathrm{d} = 7{,}5$ i $T_\mathrm{i} = 30$.



Rys. 13.1. Oscylacje niegasnące

13.1.2. Dobieranie parametrów

Pomimo, że metoda Zieglera-Nicholsa (13.2) dała całkiem zadowalające wyniki spróbowaliśmy eksperymentalnie znaleźć lepsze nastawy regulatora:

Dla poniższego wykresu (13.3) dobrane współczynniki $K = 10$, $T_\mathrm{i} = 20$ i $T_\mathrm{d} = 4{,}5$. Współczynnik jakości to $E = 4{,}9777 \cdot 10^3$.

Dla poniższego wykresu (13.4) dobrane współczynniki $K = 8$, $T_\mathrm{i} = 16$ i $T_\mathrm{d} = 5$. Współczynnik jakości to $E = 5{,}1852 \cdot 10^3$.

13.1.3. Wynik

Podsumowując najlepszą regulację zapewniły nam nastawy dobrane metoda Zieglera-Nicholsa. Współczynnik jakości **E** był najmniejszy i równy $4{,}1594 \cdot 10^3$.

13.2. Dobór parametrów regulatora DMC

Skrypty wykorzystywane do zrealizowania tego zadania: `zad4L_dmc.m`, `duNew.m`. `dmcGeneration.m`, `dmc.m`.

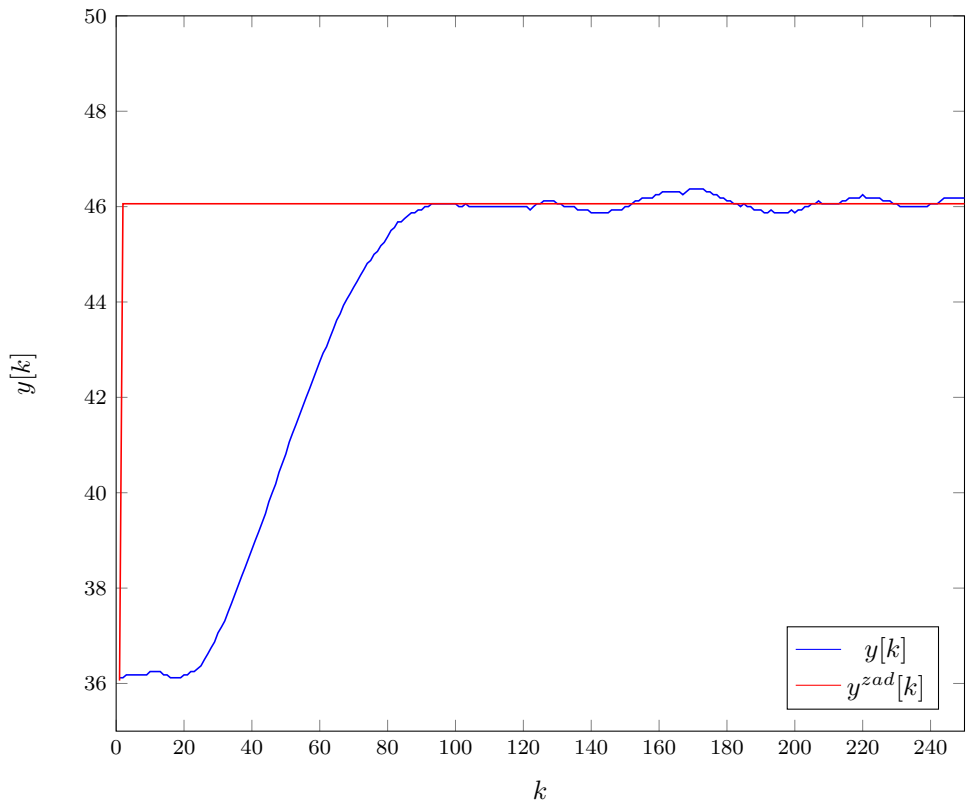
Proces strojenia regulatora DMC został opisany w 7, więc tutaj umieszczone zostaną tylko wyniki naszej pracy w laboratorium. Początkowe wartości zostały przyjęte jako $D = 300$, $N = 300$, $N_\mathrm{u} = 300$ i $\lambda = 1$. Otrzymaliśmy $E = 3{,}2109 \cdot 10^3$. Zależność ta została przedstawiona na wykresie 13.5.

13.2.1. Dobór horyzontu dynamiki D

Skracając horyzont dynamiki do 100 pogorszyła nam się jakość regulacji, więc ustawiliśmy go na 200. $E = 3{,}5613 \cdot 10^3$. Zależność ta została przedstawiona na wykresie 13.6.

13.2.2. Dobór horyzontu predykcji N

Horyzont predykcji udało nam się skrócić do 100 nie pogarszając jakości regulacji. Błąd $E = 3{,}5613 \cdot 10^3$. Zależność ta została przedstawiona na wykresie 13.7.



Rys. 13.2. Współczynniki PID: $K = 8$, $T_\mathrm{i} = 16$ i $T_\mathrm{d} = 5$, $E = 4{,}1594 \cdot 10^3$

13.2.3. Dobór horyzontu sterowania N_u

Horyzont sterowania skróciliśmy do wartości 50 minimalnie pogarszając jakość regulacji, ale znacząco oszczędzając na ilości obliczeń. $E = 3{,}5730 \cdot 10^3$. Zależność ta została przedstawiona na wykresie 13.8.

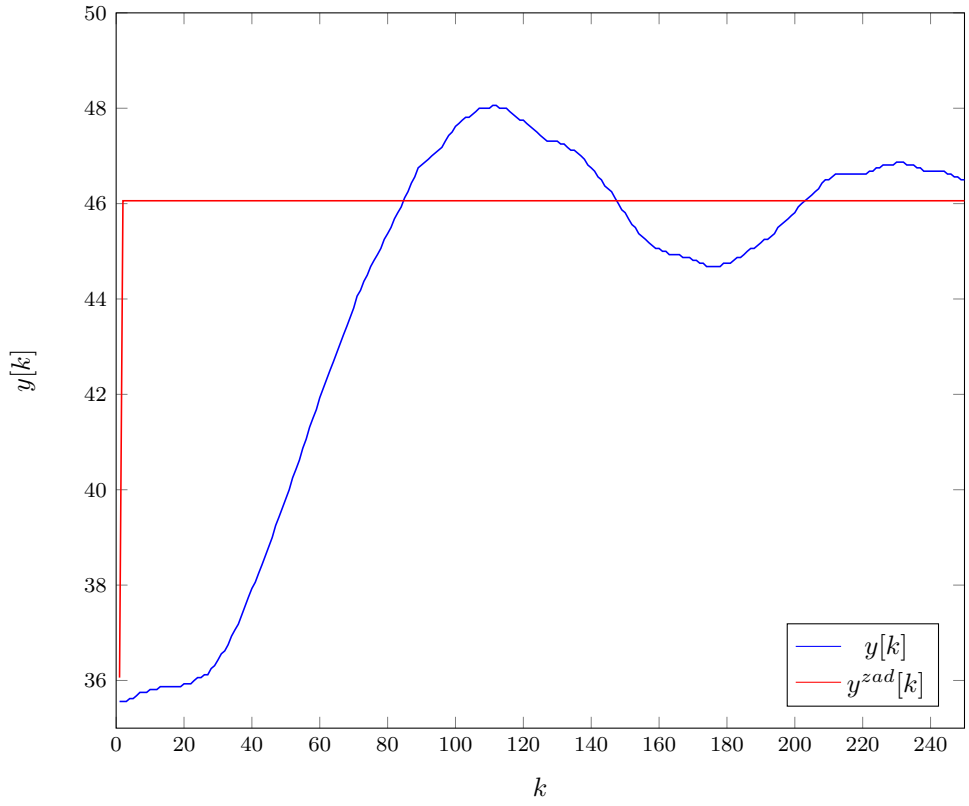
13.2.4. Wpływ współczynnika λ

Dla pokazania wpływu współczynnika λ przeprowadziliśmy dwa eksperymenty, jeden dla $\lambda = 5$, a drugi dla $\lambda = 0{,}1$.

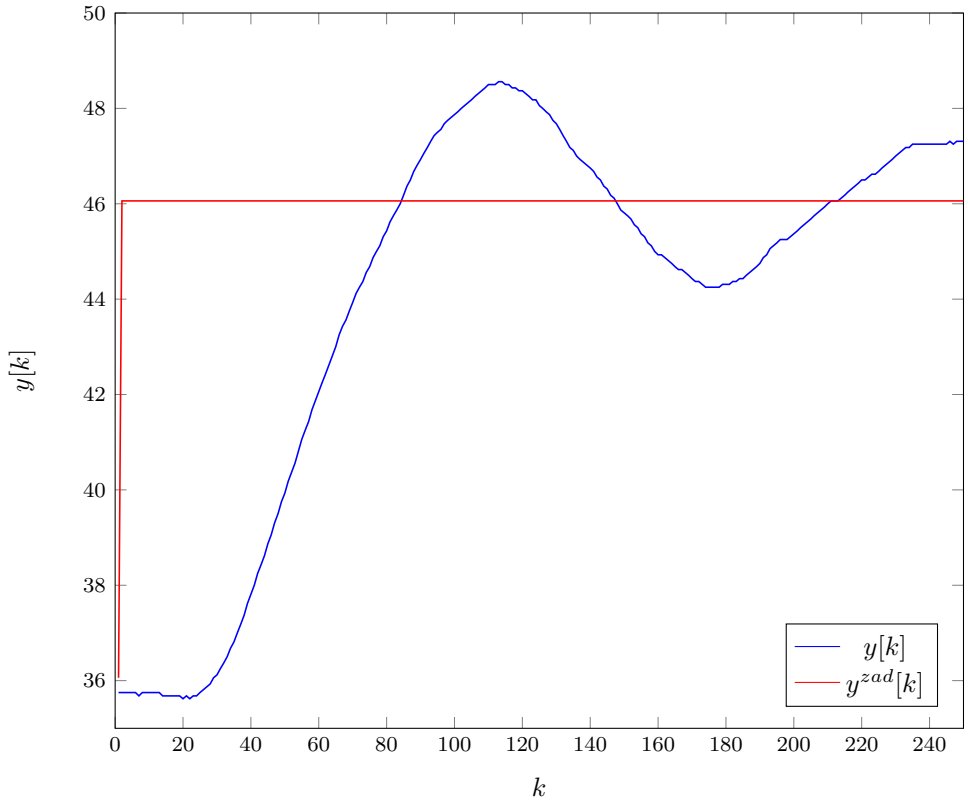
Dla $\lambda = 5$ otrzymaliśmy $E = 4{,}1577 \cdot 10^3$. Zależność ta została przedstawiona na wykresie 13.9.

Dla $\lambda = 0{,}1$ otrzymaliśmy $E = 3{,}2120 \cdot 10^3$. Zależność ta została przedstawiona na wykresie 13.10.

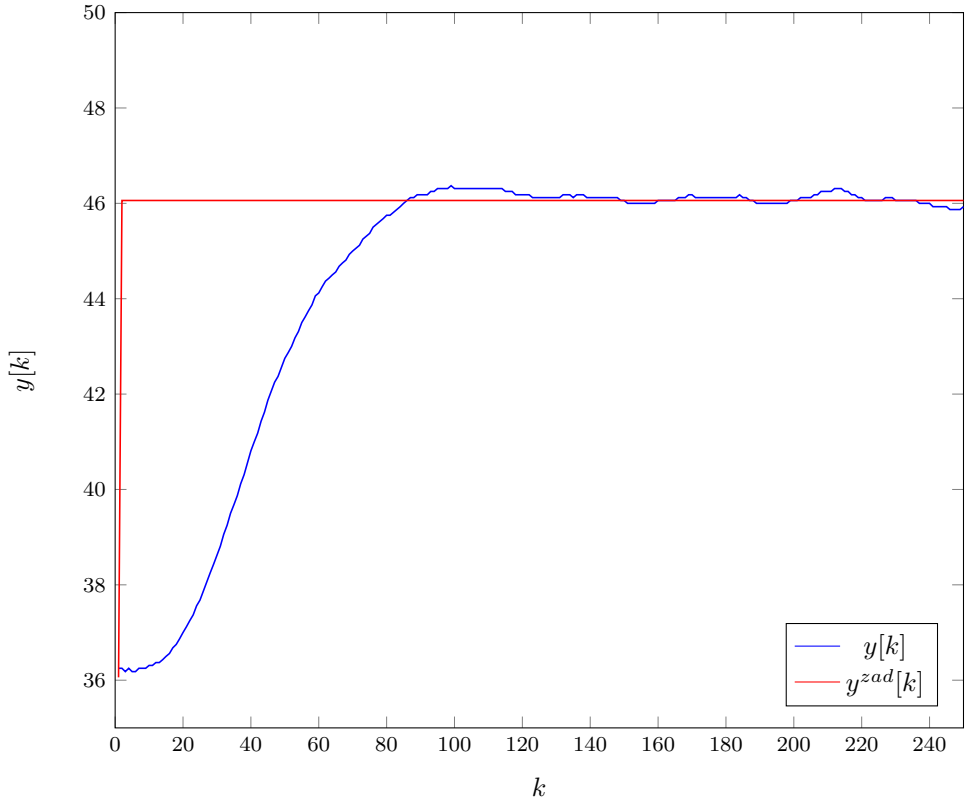
Trzeba zaznaczyć, że wybór współczynnika λ powinien być kompromisem pomiędzy szybkością regulacji, a bezpiecznym przebiegiem sygnału sterującego.



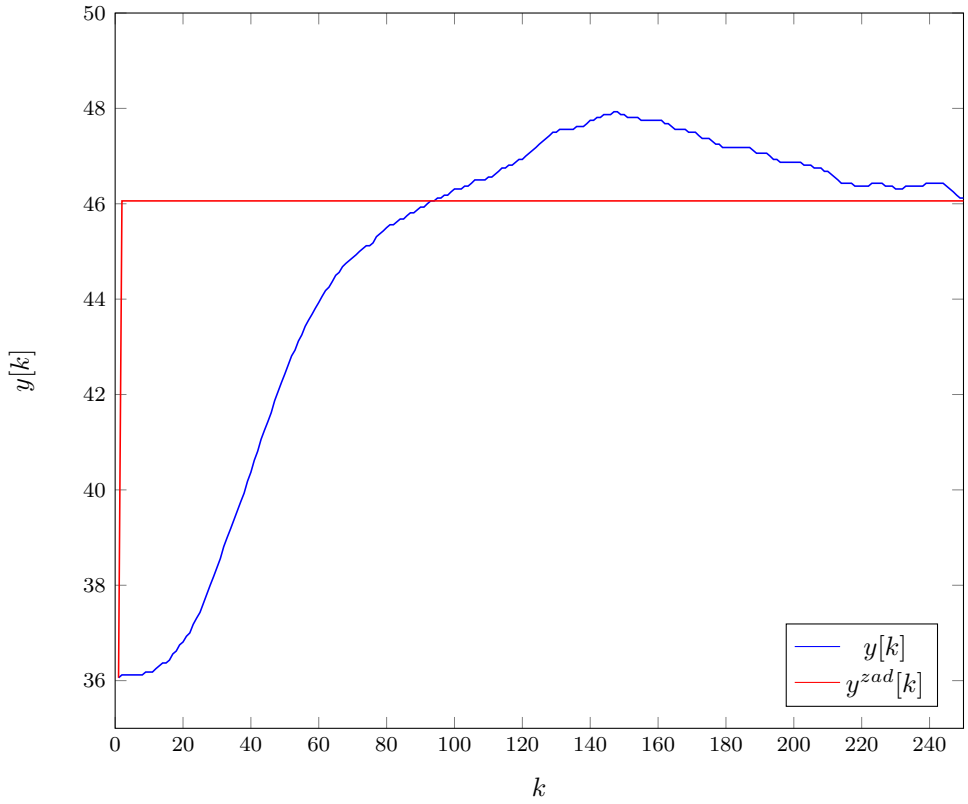
Rys. 13.3. Współczynniki PID: $K = 10$, $T_i = 20$ i $T_d = 4{,}5$, $E = 4{,}9777 \cdot 10^3$



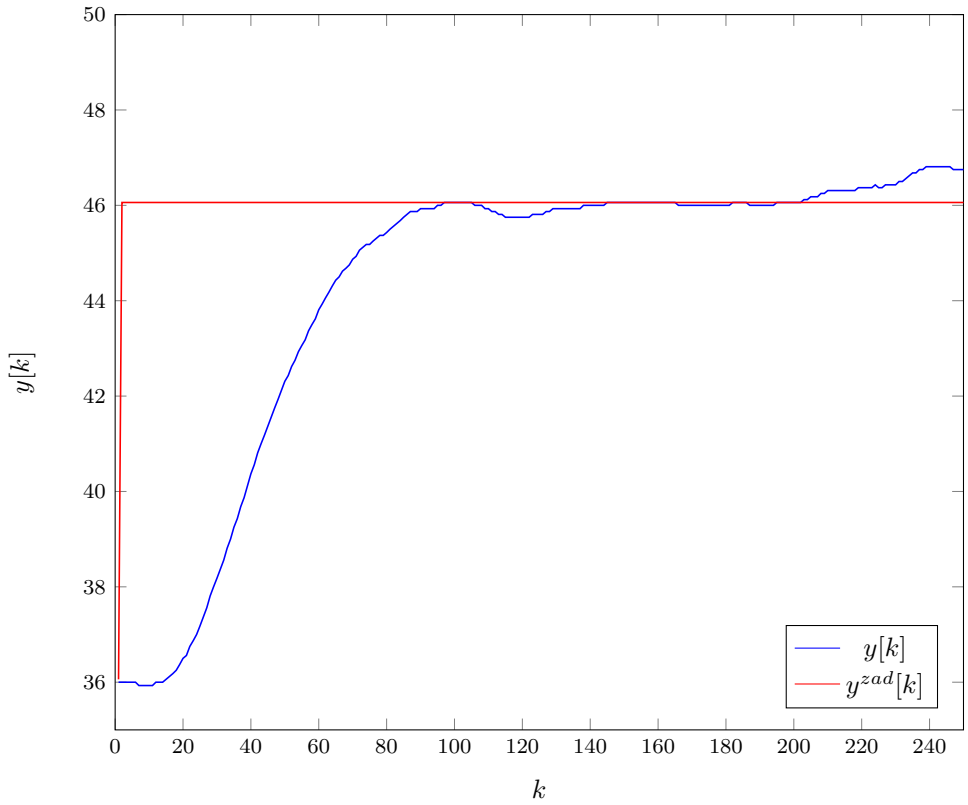
Rys. 13.4. Współczynniki PID: $K = 8$, $T_\mathrm{i} = 16$ i $T_\mathrm{d} = 5$, $E = 5{,}1852 \cdot 10^3$



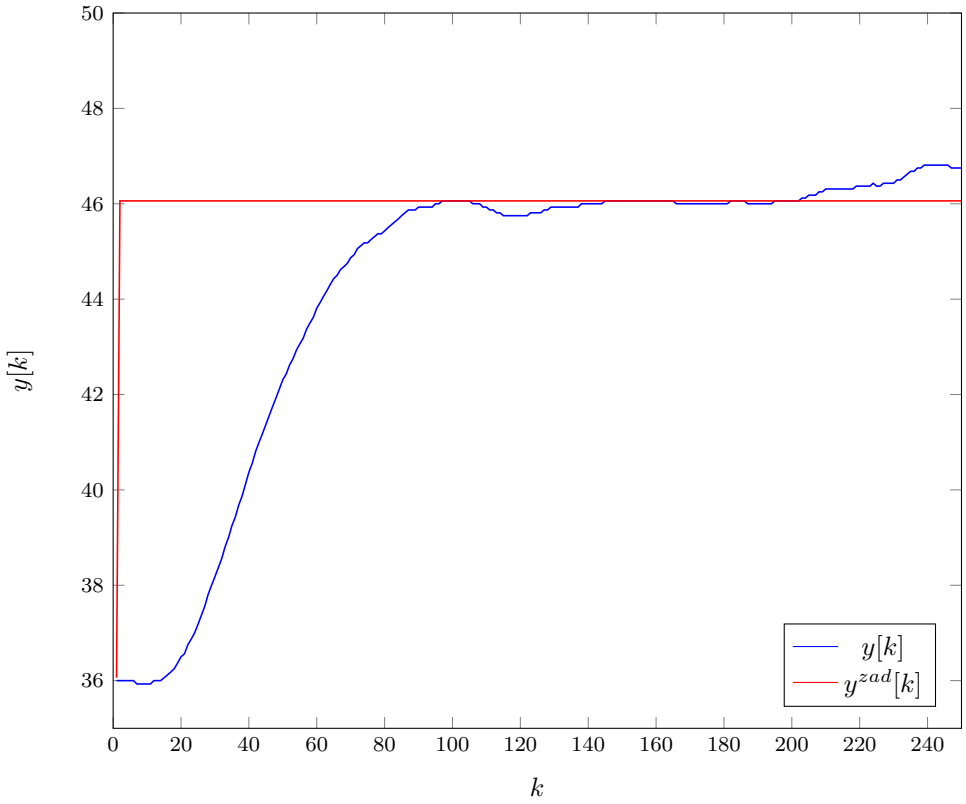
Rys. 13.5. Współczynniki DMC: $D = 300$, $N = 300$, $N_\mathrm{u} = 300$ i $\lambda = 1$



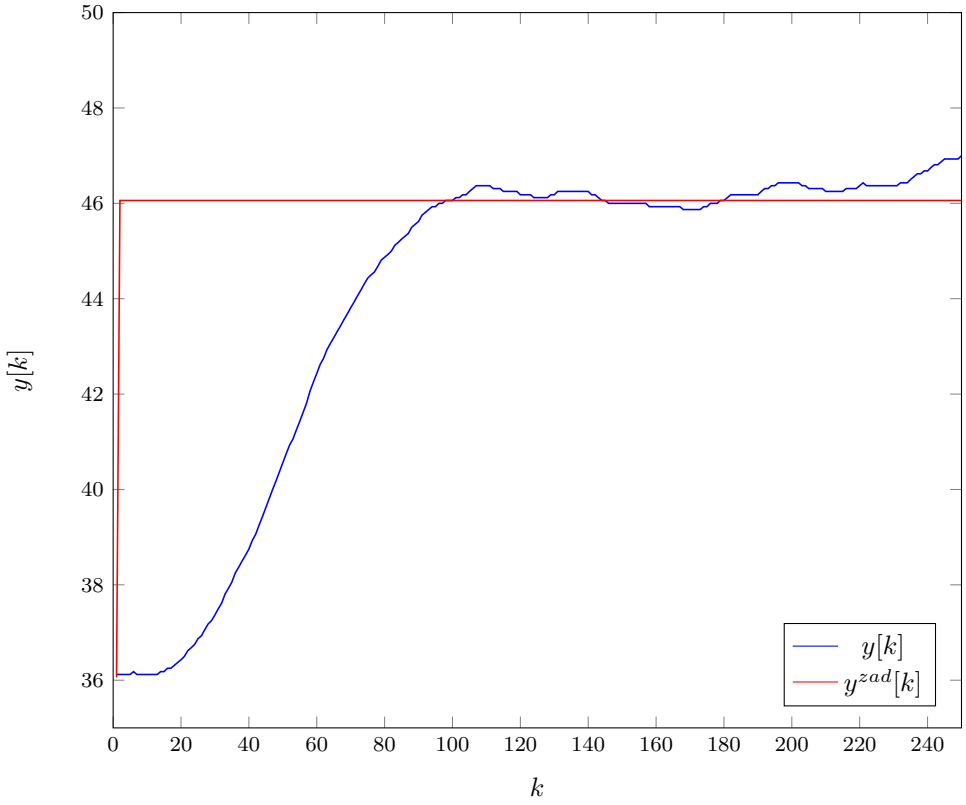
Rys. 13.6. Współczynniki DMC: $D = 100$, $N = 300$, $N_{\mathrm{u}} = 300$ i $\lambda = 1$



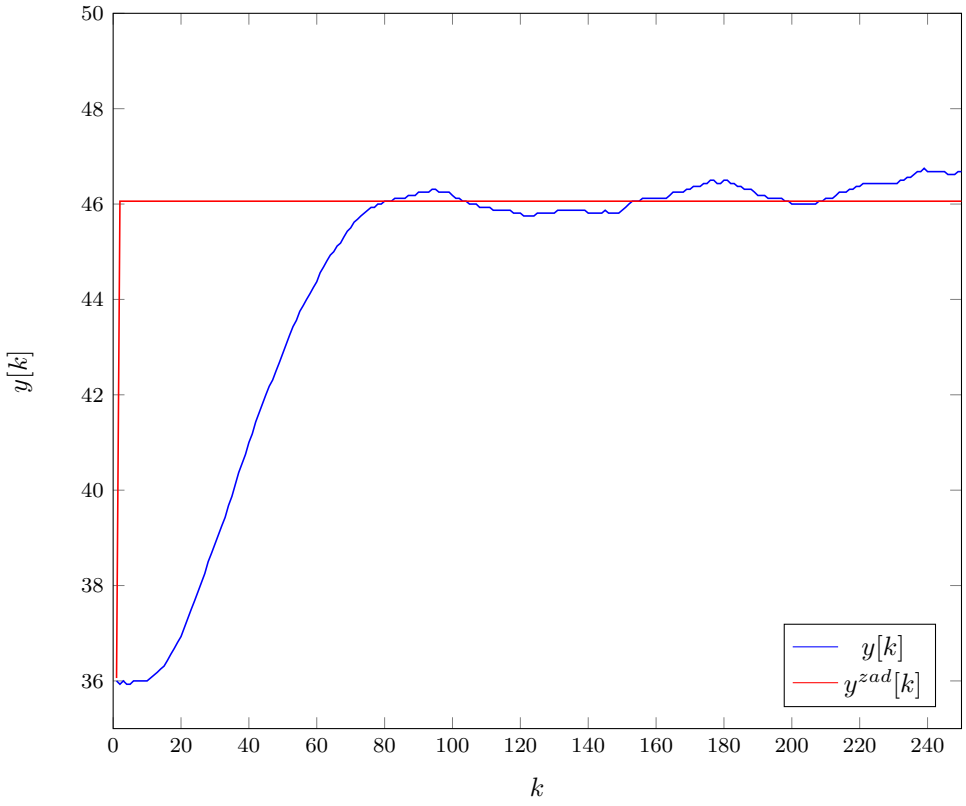
Rys. 13.7. Współczynniki DMC: $D = 200$, $N = 100$, $N_{\mathrm{u}} = 300$ i $\lambda = 1$



Rys. 13.8. Współczynniki DMC: $D = 200$, $N = 100$, $N_{\mathrm{u}} = 100$ i $\lambda = 1$



Rys. 13.9. Współczynniki DMC: $D = 200$, $N = 100$, $N_{\mathrm{u}} = 100$ i $\lambda = 5$



Rys. 13.10. Współczynniki DMC: $D = 200$, $N = 100$, $N_{\mathrm{u}} = 100$ i $\lambda = 0{,}1$