Politechnika Śląska

Wydział Automatyki, Elektroniki i Informatyki

Kierunek Automatyka i Robotyka

Metody optymalizacji

Ćwiczenie 4 - Problem liniowo-kwadratowy

Autorzy:

Piotr Gołuch

Mateusz Wiśniewski

Gliwice, 17 kwietnia 2020

1 Wstęp

Celem ćwiczenia było opanowanie metody projektowania układów sterowania optymalnego w wersji dyskretnej, opartych na regulatorze liniowo-kwadratowym.

Dla zadanego układu należało sprawdzić założenia problemu liniowo-kwadratowego, określić czy układ jest sterowalny oraz wyznaczyć wartości wektora stanu, wektora sterowań oraz wartość wskaźnika jakości. W tym celu napisano program rozwiązujący problem liniowo-kwadratowy oraz zbadano wpływ warunków początkowych oraz wartości macierzy $\boldsymbol{R}$ na otrzymane przebiegi czasowe (macierz $\boldsymbol{R}$ zawiera współczynniki określające „udział" wartości w wektorze sterowania na wartość wskaźnika jakości) .

1.1 Zadany układ

Dane - sekcja 8:

- wskaźnik jakości:

$$J = 0.5 \sum_{i=0}^{N-1} (13x_{1,i}^2 + 9x_{2,i}^2 - 8x_{1,i}x_{2,i} + 4u_i^2)$$

- równania stanu:

$$x_{1,i+1} = x_{1,i} + x_{2,i} + 2u_i$$

$$x_{2,i+1} = x_{1,i} + 2x_{2,i} + 4u_i$$

- liczba iteracji: 20

- wartości początkowe: $x_{1_0} = 10$, $x_{2_0} = 15$

1.2 Wyznaczone macierze

Zgodnie z oznaczeniami:

$$J = \frac{1}{2} \sum_{i=0}^{N-1} \left(\boldsymbol{x}_i^T \boldsymbol{Q} \boldsymbol{x}_i + \boldsymbol{u}_i^T \boldsymbol{R} \boldsymbol{u}_i \right) + \frac{1}{2} \boldsymbol{x}_N^T \boldsymbol{F} \boldsymbol{x}_N$$

$$\boldsymbol{x}_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i \qquad i = 0, 1, ..., N-1 \qquad \boldsymbol{x}_i = \begin{bmatrix} x_{1,i} \\ x_{2,i} \end{bmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \qquad \boldsymbol{B} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \qquad \boldsymbol{Q} = \begin{bmatrix} 13 & -4 \\ -4 & 9 \end{bmatrix} \qquad \boldsymbol{R} = \begin{bmatrix} 4 \end{bmatrix} \qquad \boldsymbol{F} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

1.3 Sprawdzenie założeń

Macierze $\boldsymbol{Q}$, $\boldsymbol{R}$ i $\boldsymbol{F}$ są symetryczne. $\boldsymbol{R}$ jest macierzą składającą się z jednego, dodatniego elementu, co oznacza, że jest dodatnio określona. Macierz $\boldsymbol{F}$ jest macierzą zerową, więc wszystkie jej wiodące minory główne są równe 0, co oznacza, że jest ona dodatnio półokreślona. Należy jeszcze sprawdzić dodatnią półokreśloność macierzy $\boldsymbol{Q}$. Jej wiodące minory główne to $\begin{vmatrix}13\end{vmatrix} = 13$ i $\begin{vmatrix}13 & -4\\ -4 & 9\end{vmatrix} = 13 \cdot 9 - (-4) \cdot (-4) = 101$. Oba te minory są dodatnie, więc macierz $\boldsymbol{Q}$ jest dodatnio określona (czyli także dodatnio półokreślona).

Ostatnim założeniem, które musi zostać spełnione, jest sterowalność układu. Sterowalność bada się tworząc macierz $\boldsymbol{H} = [\boldsymbol{B}, \boldsymbol{AB}, \boldsymbol{A}^2\boldsymbol{B}, ..., \boldsymbol{A}^{n-1}\boldsymbol{B}]$, gdzie n jest rzędem układu (wymiarem macierzy $\boldsymbol{A}$). Jeżeli rząd tak utworzonej macierzy jest równy n, to układ jest sterowalny. W tym przypadku $n = 2$, czyli $\boldsymbol{H} = [\boldsymbol{B}, \boldsymbol{AB}]$.

$$\boldsymbol{AB} = \begin{bmatrix}1 & 1\\ 1 & 2\end{bmatrix}\begin{bmatrix}2\\ 4\end{bmatrix} = \begin{bmatrix}6\\ 10\end{bmatrix}$$

$$\boldsymbol{H} = \begin{bmatrix}2 & 6\\ 4 & 10\end{bmatrix}$$

Macierz kwadratowa o wymiarze n posiada rząd równy n wtedy, gdy jej wyznacznik jest różny od 0.

$$\det \boldsymbol{H} = 2 \cdot 10 - 6 \cdot 4 = -4 \neq 0$$

Rząd macierzy $\boldsymbol{H}$ jest równy n, zatem układ jest sterowalny.

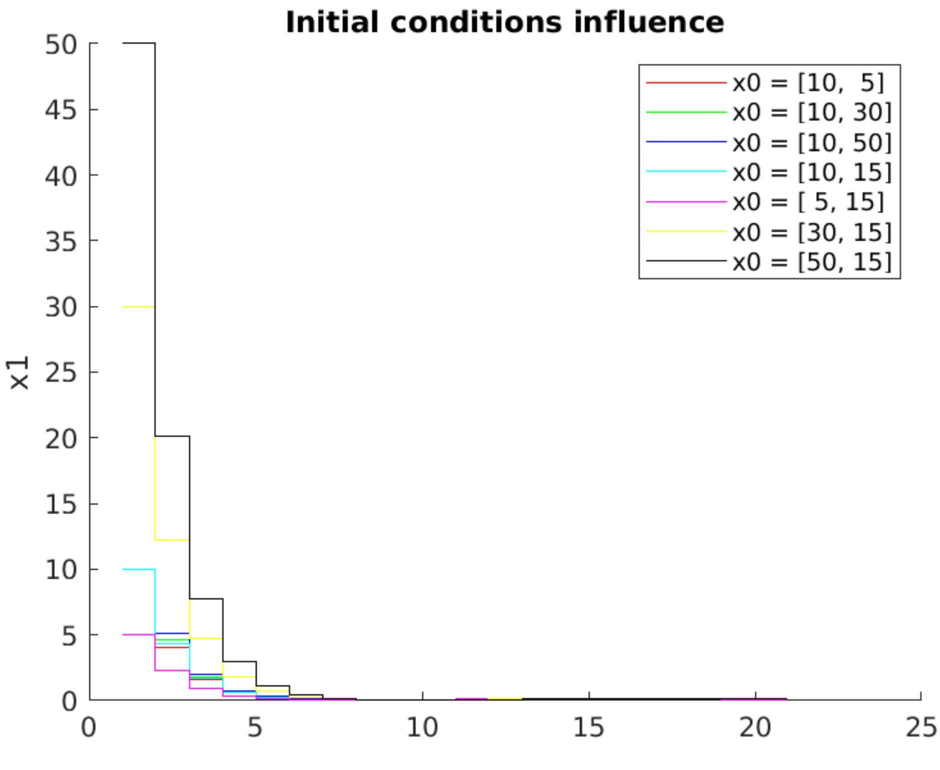
2 Wyniki

Wartości sterowania i stanu układu wyznaczane były dla dyskretnych chwil czasu. Wartość sterująca zmienia się w sposób skokowy, a stan układu może również zmieniać się skokowo (jeśli obiekt jest z natury dyskretny) lub w sposób ciągły (jeśli obiekt jest ciągły, a wszystkie obliczenia przeprowadzane były dla jego zdyskretyzowanego modelu). Wszystkie wartości jednak znane były jedynie w dyskretnych chwilach czasu, dlatego wykres ich zmian powinien być wykresem schodkowym. Wykres schodkowy jednak jest mało przejrzysty i ciężko zauważyć na nim niektóre różnice między sygnałami, dlatego w dalszej części sprawozdania zamieszczone zostały zarówno wykresy schodkowe (przedstawiające faktyczne przebiegi) jak i wykresy liniowe, które pozwalają łatwiej dostrzec charakter przebiegów (jednak nie odpowiadają ich rzeczywistemu wyglądowi).

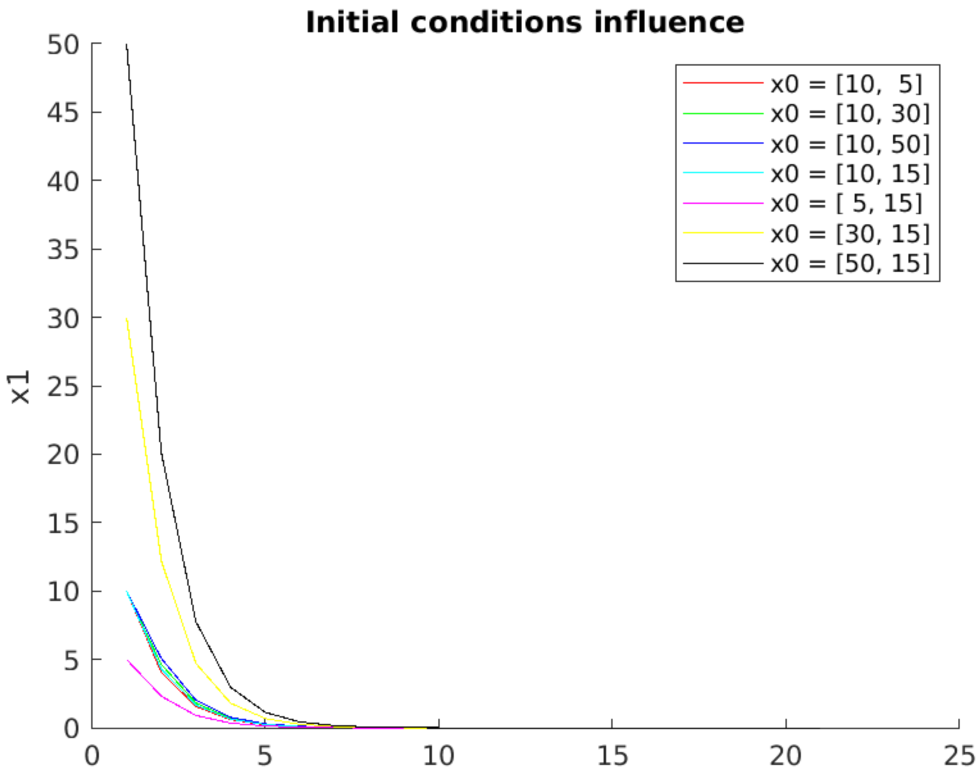
2.1 Wpływ warunków początkowych

Przebiegi czasowe zbadano dla wartości $\boldsymbol{R} = 4$. Warunki początkowe zmieniano następująco:

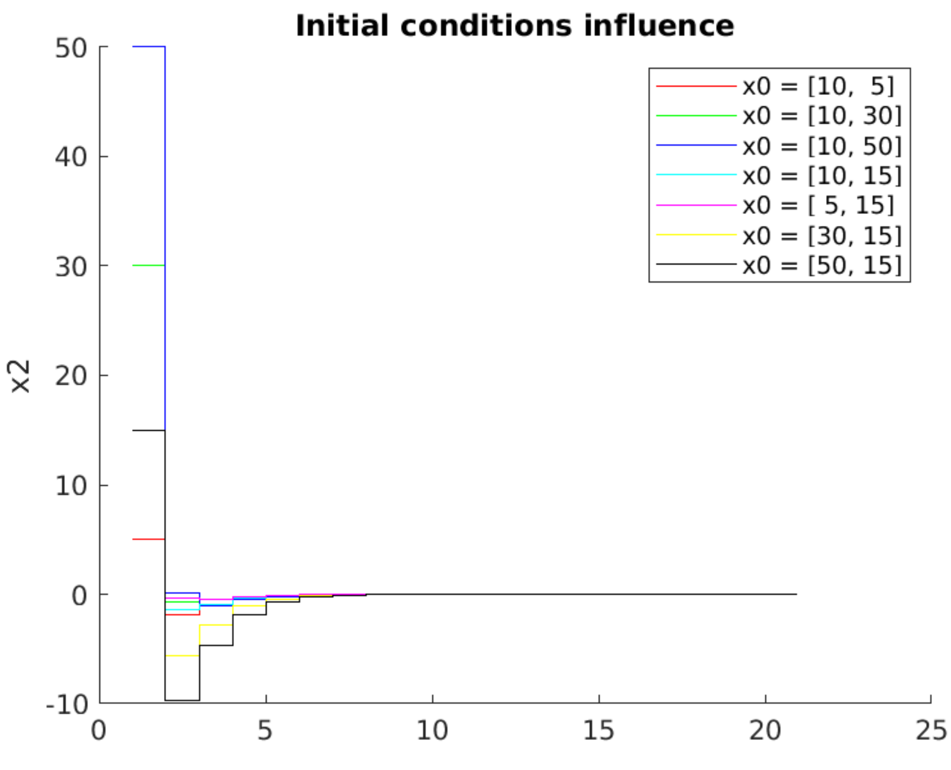
$$\boldsymbol{x}_0 = \left\{ \begin{bmatrix} 10 \\ 5 \end{bmatrix}, \begin{bmatrix} 10 \\ 30 \end{bmatrix}, \begin{bmatrix} 10 \\ 50 \end{bmatrix}, \begin{bmatrix} 10 \\ 15 \end{bmatrix}, \begin{bmatrix} 5 \\ 15 \end{bmatrix}, \begin{bmatrix} 30 \\ 15 \end{bmatrix}, \begin{bmatrix} 50 \\ 15 \end{bmatrix} \right\}$$



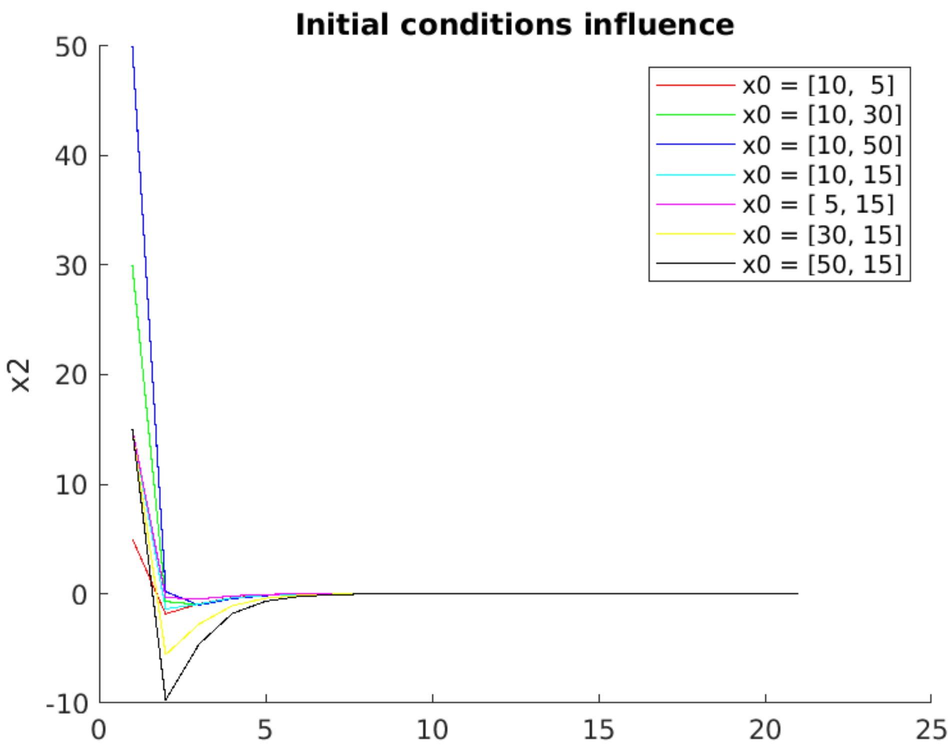
Rysunek 1: Przebiegi x_1 dla różnych warunków początkowych (wykres schodkowy).



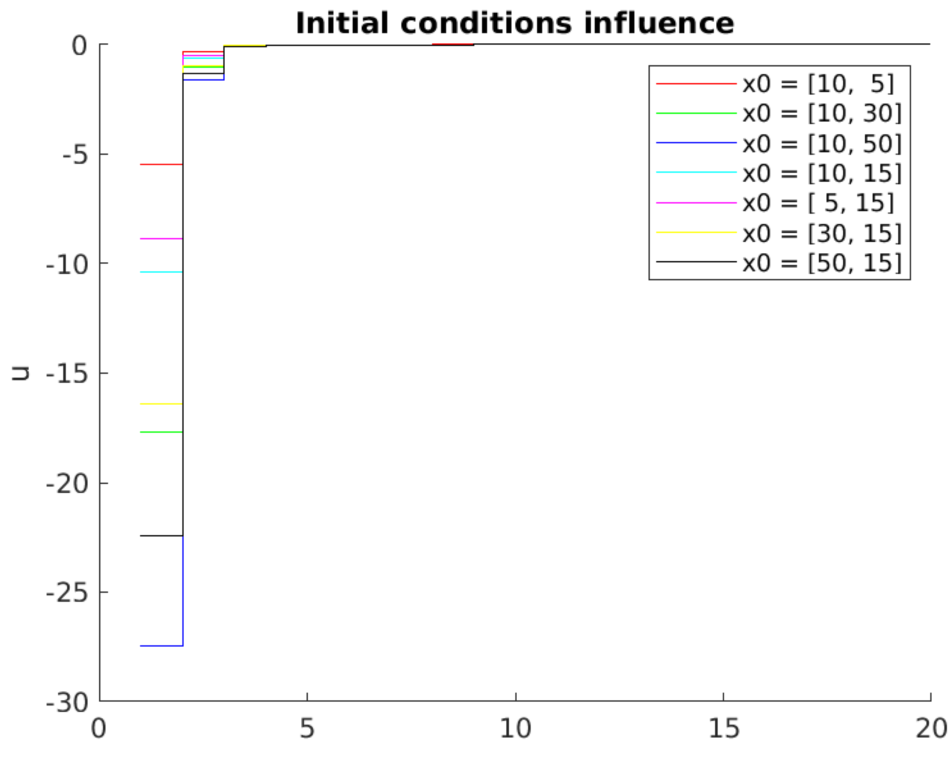
Rysunek 2: Przebiegi x_1 dla różnych warunków początkowych (wykres liniowy).



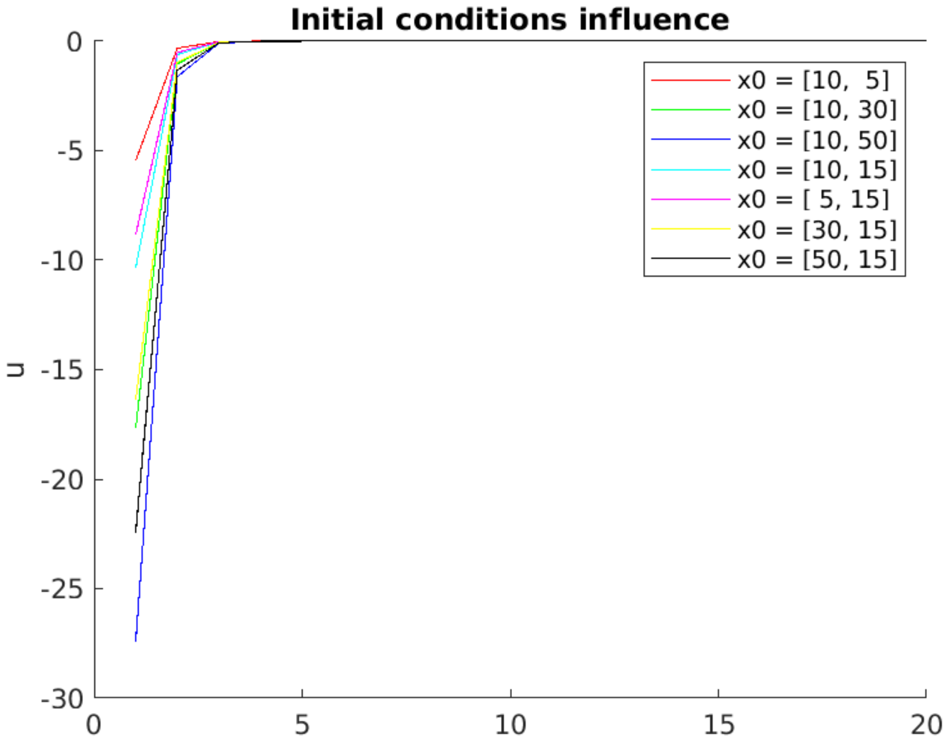
Rysunek 3: Przebiegi x_2 dla różnych warunków początkowych (wykres schodkowy).



Rysunek 4: Przebiegi x_2 dla różnych warunków początkowych (wykres liniowy).



Rysunek 5: Przebiegi u dla różnych warunków początkowych (wykres schodkowy).



Rysunek 6: Przebiegi u dla różnych warunków początkowych (wykres liniowy).

Wskaźniki jakości:

```
x0 = [10,  5]'--->Jo = 806.214612
x0 = [10, 30]'--->Jo = 4322.375166
x0 = [10, 50]'--->Jo = 11625.157515
x0 = [10, 15]'--->Jo = 1464.369850
x0 = [ 5, 15]'--->Jo = 1080.593792
x0 = [30, 15]'--->Jo = 7255.931510
x0 = [50, 15]'--->Jo = 19857.825056
```

Zgodnie z intuicją, większa wartość warunków początkowych zwiększa czas osiągania przez układ stanu ustalonego. Dla zmiennej x_1 kluczową rolę odgrywa wartość $x_{1,0}$, która określa jej wartość początkową, a tym samym „odległość” od wartości w stanie ustalonym (podobnie jest w przypadku x_2, dla którego największy wpływ ma wartość $x_{2,0}$). Na ogólny przebieg czasowy mają jednak w obu przypadkach wpływ obie wartości początkowe, co wynika bezpośrednio z postaci macierzy $\boldsymbol{A}$ (obie zmienne zależą od siebie nawzajem).

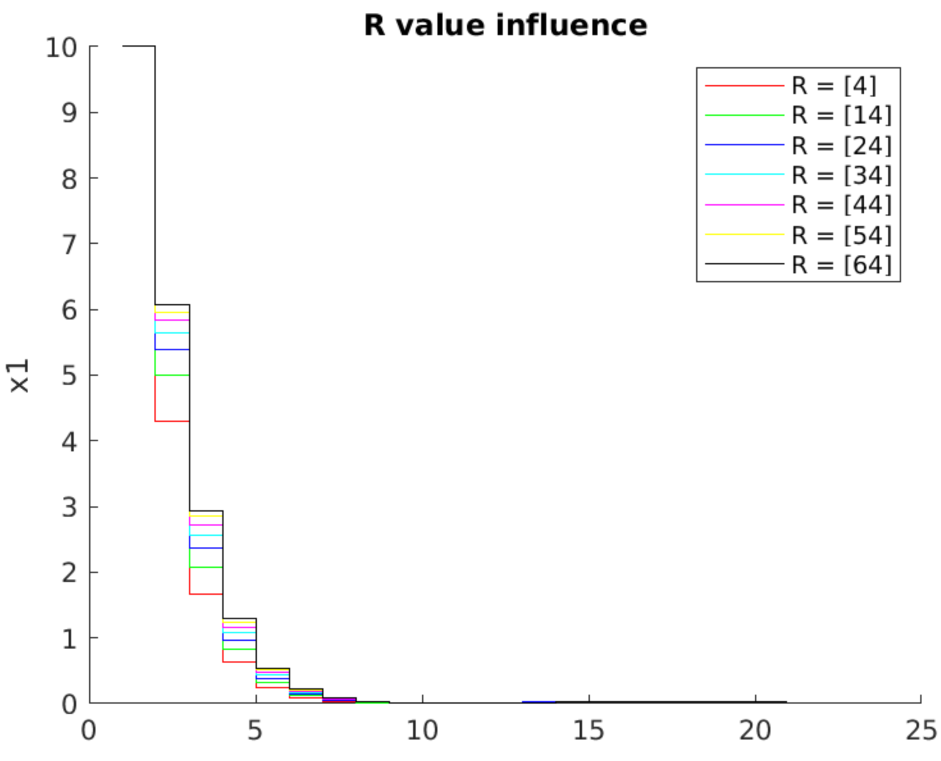
Większa wartość warunków początkowych ma również wpływ na większą wartość bezwzględną sterowania. Wynika to z dodatniej określoności macierzy $\boldsymbol{Q}$ (koszt pochodzący od stanu układu jest zawsze nieujemny). Można zauważyć również, że wartości początkowe obu zmiennych mają różny wpływ na wartość sterowania. Przykładowo sterowanie dla warunków początkowych $[10, 50]^T$ (niebieski wykres na rysunku 5) przyjmuje większe wartości niż dla warunków początkowych $[50, 15]^T$ (linia czarna), pomimo że w obu przypadkach wartości $x_{1,0}$ i $x_{2,0}$ wynoszą na przemian 50, a w drugim przypadku wartość $x_{2,0}$ jest nawet większa niż wartość $x_{1,0}$ w przypadku pierwszym.

Wartości wskaźnika jakości są tym większe, im większe są wartości warunków początkowych, co wynika z dłuższego czasu dochodzenia do stanu ustalonego (większy koszt pochodzący od stanu układu) oraz z większej wartości sterowania. Można również zauważyć, że większy wpływ na wskaźnik jakości ma zmienna x_1, co jest związane z większym współczynnikiem odpowiadającym x_1^2 w macierzy $\boldsymbol{Q}$.

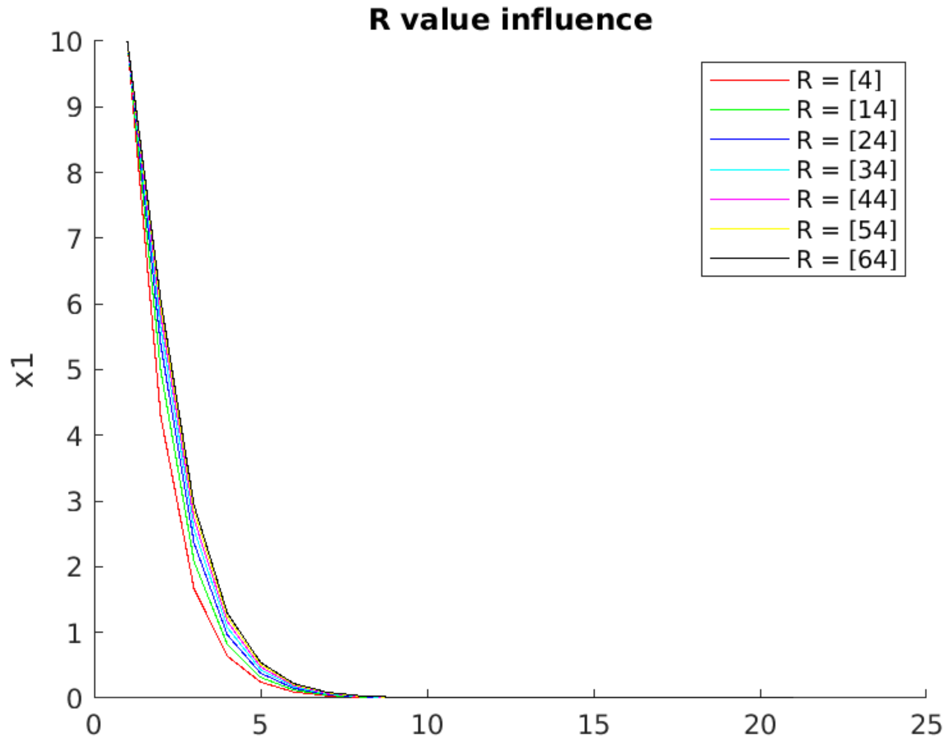
2.2 Wpływ wartości $\boldsymbol{R}$

Przebiegi zbadano dla warunków początkowych $x_0 = [10, 15]^T$. Wartość $\boldsymbol{R}$ zmieniano w zakresie:

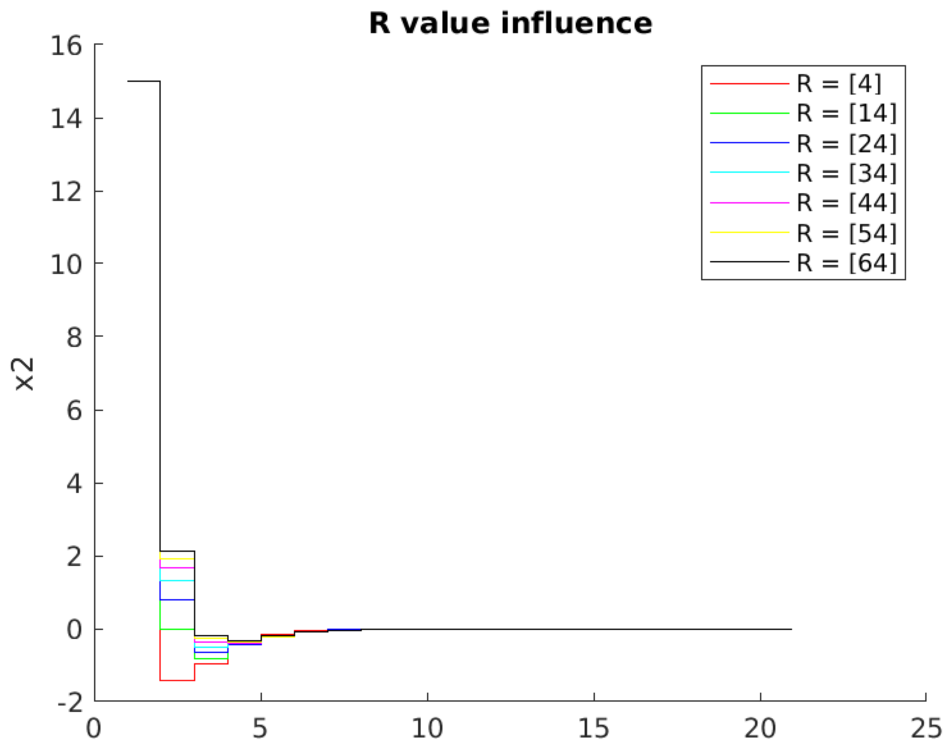
$$\boldsymbol{R} = \{4, 14, 24, 34, 44, 54, 64\}$$



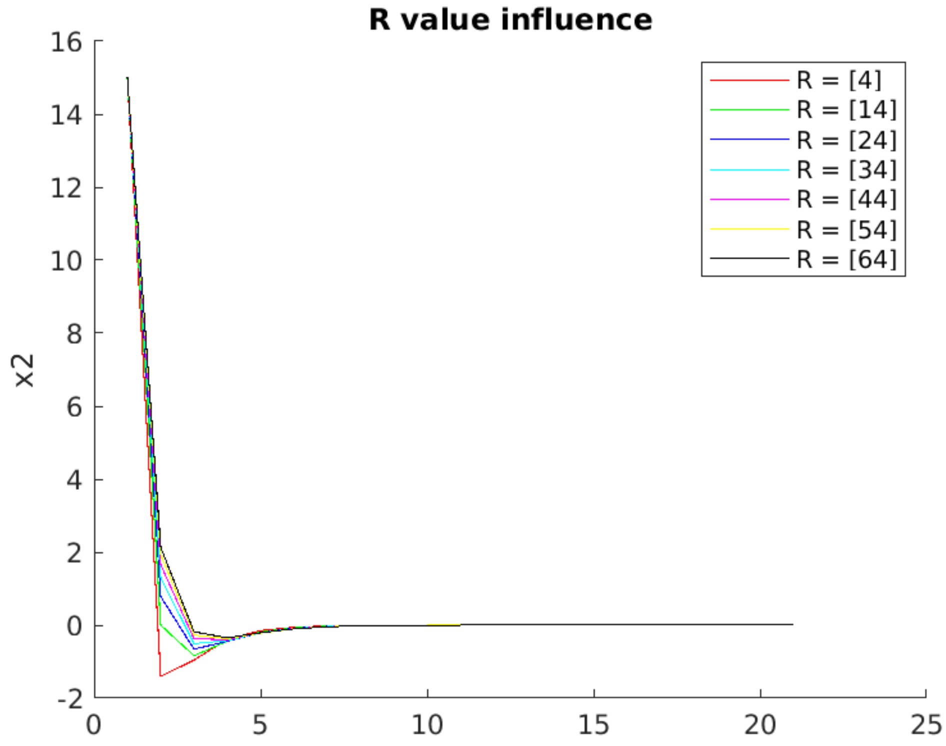
Rysunek 7: Przebiegi x_1 dla różnych wartości $\mathbf{R}$ (wykres schodkowy).



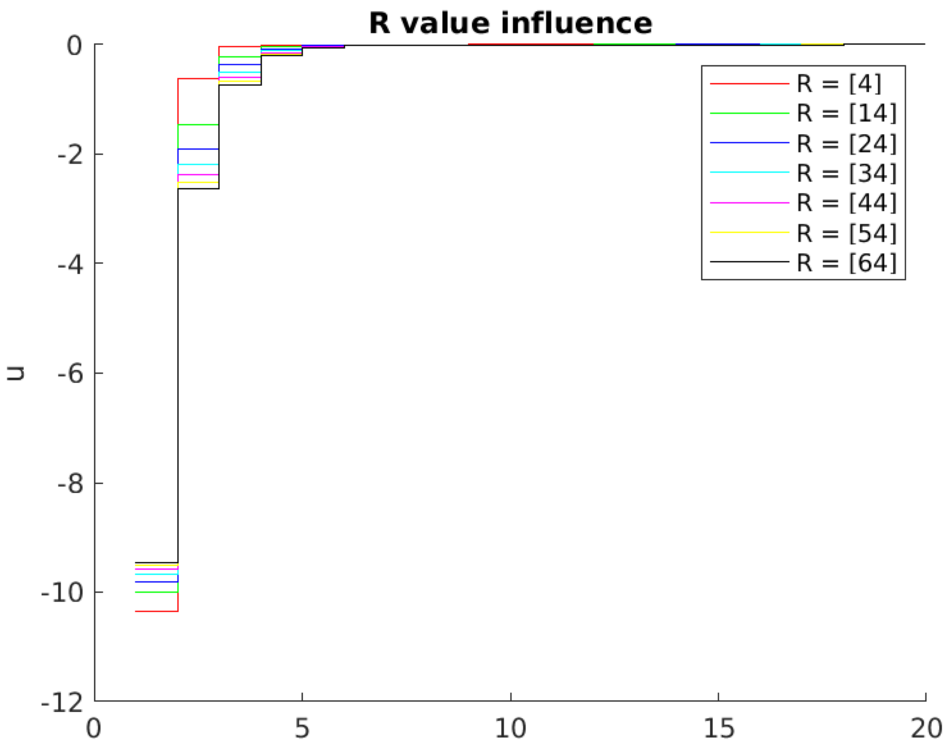
Rysunek 8: Przebiegi x_1 dla różnych wartości $\mathbf{R}$ (wykres liniowy).



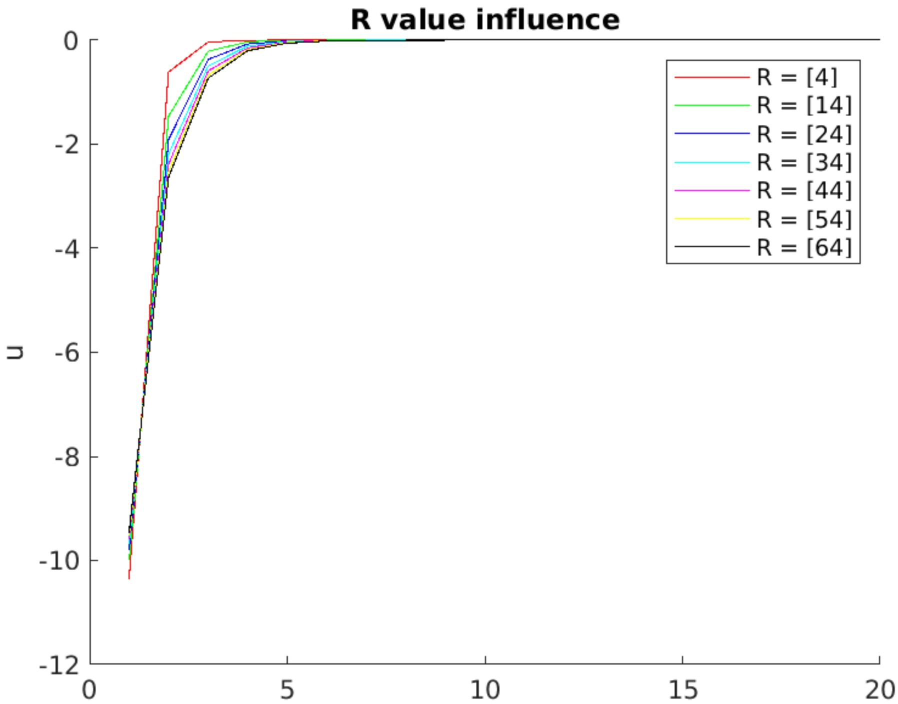
Rysunek 9: Przebiegi x_2 dla różnych wartości $\mathbf{R}$ (wykres schodkowy).



Rysunek 10: Przebiegi x_2 dla różnych wartości **R** (wykres liniowy).



Rysunek 11: Przebiegi u dla różnych wartości **R** (wykres schodkowy).



Rysunek 12: Przebiegi u dla różnych wartości **R** (wykres liniowy).

Wskaźniki jakości:

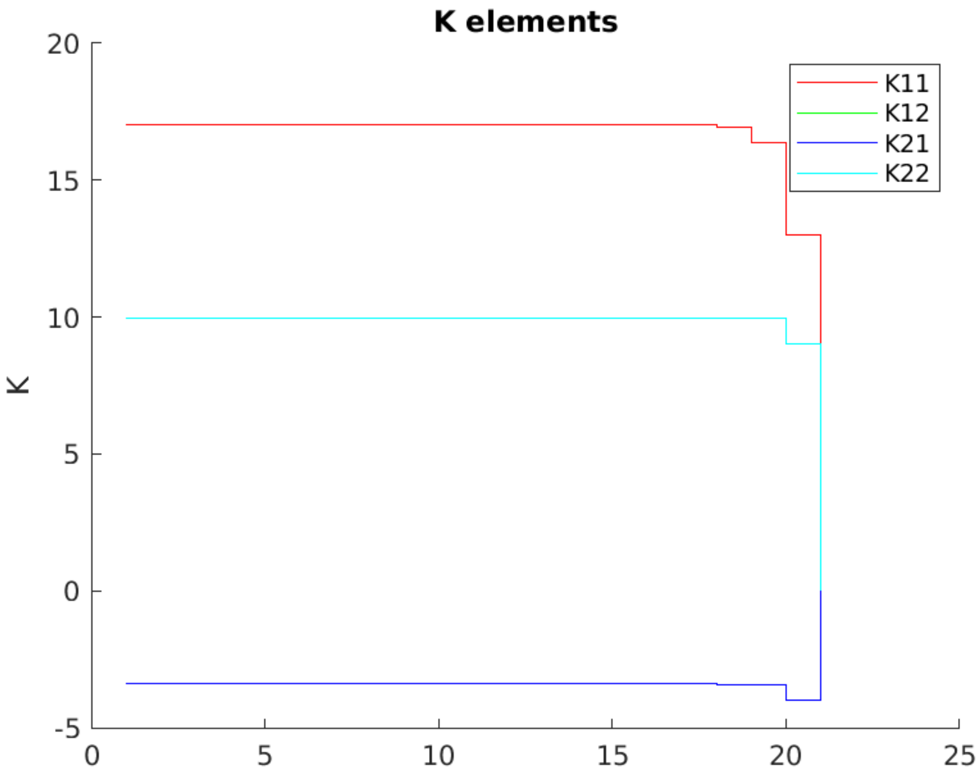
```
R =  4 --->Jo = 1464.369850
R = 14 --->Jo = 1986.435688
R = 24 --->Jo = 2490.908308
R = 34 --->Jo = 2986.972608
R = 44 --->Jo = 3478.270194
R = 54 --->Jo = 3966.577520
R = 64 --->Jo = 4452.876062
```

Wartość $\boldsymbol{R}$ określa wpływ sterowania na wartość wskaźnika jakości. W związku z tym, większy współczynnik $\boldsymbol{R}$ powoduje, że preferowane będą takie sterowania, które nie przyjmują dużych wartości, nawet kosztem dłuższego dochodzenia do stanu ustalonego. Istotnie, dla mniejszych wartości $\boldsymbol{R}$ układ szybciej osiągał stan ustalony. W przypadku x_2 wiązało się to jednak również z większym przeregulowaniem, co spowodowane było większą wartością początkową sterowania. W późniejszych chwilach sterowanie dla mniejszej wartości $\boldsymbol{R}$ jednak malało, co było spowodowane tym, że układ zdążył już mocno zbliżyć się do wartości stanu ustalonego.

Wskaźnik jakości rośnie wraz ze wzrostem $\boldsymbol{R}$, co jest spowodowane tym, że zwiększa się wartość kosztu pochodzącego od sterowania, oraz dłuższym czasem dochodzenia do stanu ustalonego (większy koszt pochodzący od stanu układu).

2.3 Ustalanie się elementów macierzy $\boldsymbol{K}$

Do wyliczenia wartości przyjęto $\boldsymbol{x}_0 = [10, 15]^T$ i $\boldsymbol{R} = 4$.



Rysunek 13: Ustalanie się elementów macierzy $\boldsymbol{K}$.

Macierz $\boldsymbol{K}$ ma wymiary 2x2, składa się więc z 4 elementów. Jest ona macierzą symetryczną, więc dwa z nich (K_{12} i K_{21}) są sobie równe (na wykresie wartość K_{12} jest niewidoczna). Jak widać, wszystkie elementy macierzy przez większość kroków utrzymują praktycznie stałą wartość, dopiero pod koniec zbiegają się w punkcie 0 (macierz K_N jest równa macierzy $\boldsymbol{F}$, która w tym przypadku jest macierzą zerową). Macierz $\boldsymbol{K}$ pozwala wyliczyć wartość wskaźnika jakości począwszy od i-tego kroku. Wartość ta zależy również od stanu układu, dlatego, analizując wykres zmienności $\boldsymbol{K}$ i charakter przebiegów x_1, x_2 i u, można wywnioskować, że dla $i \geqslant 7$ wartość wskaźnika począwszy od i-tego kroku jest w przybliżeniu zerowa (stan układu jest praktycznie zerowy). Dla $i < 7$ wartość współczynników w macierzy $\boldsymbol{K}$ jest praktycznie stała, dlatego wartość współczynnika jakości zależy jedynie od obecnego stanu układu.

3 Program

Program składa się z trzech głównych części. Każda z nich odpowiada za wyniki przedstawione w jednym z podrozdziałów rozdziału 2.

Główny program `LQ.m`:

```
clear variables;
close all;

% System and quality index data
A = [ 1,  1;
      1,  2];
B = [ 2,  4]';
Q = [13, -4;
     -4,  9];
R = [ 4];
F = zeros(2, 2);
N = 20;

% Test if Q, F and R are symmetrical and definite/semidefinite
if ~(testMat(Q, GreaterThanEqual(0)) && ...
     testMat(F, GreaterThanEqual(0)) && ...
     testMat(R, GreaterThan(0)))

    error('Choose proper matrices.');
end

% Test if system is controllable
if ~isControllable(A, B)
    error('System is not controllable');
end

% ------------------------------------------------------------
% --------------- Initial conditions influence ----------------
% ------------------------------------------------------------

% Set of all initial conditions
x0set = {[10,  5]', ...
         [10, 30]', ...
```

```
            [10, 50]', ...
            [10, 15]', ...
            [ 5, 15]', ...
            [30, 15]', ...
            [50, 15]'};

% Legend array
legendArray = makeLegend(x0set, 'x0');

% plot colors
colors = ['r', 'g', 'b', 'c', 'm', 'y', 'k'];

% Labels
labels = ["x1", "x2", "u"];

% Figures
fx1 = figure;
hold on;
fx2 = figure;
hold on;
fu = figure;
hold on;
f = [fx1, fx2, fu];

% Calculate all Ki
K = calculateK(A, B, Q, R, F, N);

% For all initial conditions
for i = 1:length(x0set)
    x0 = x0set{i};
    color = colors(i);

    % The performance index
    Jo = perfIndex(x0, K{1});

    % Print results
    fprintf("x0 = [%2d, %2d]'\t--->\tJo = %f\n", x0(1), x0(2), Jo
       );

    % Calculate x and u
```

```
    [x, u] = calcXandU(x0, N, A, B, R, K);

    % Plot everything
    figure(fx1);
    stairs(x(1, :), color);
    figure(fx2);
    stairs(x(2, :), color);
    figure(fu);
    stairs(u, color);
end

% Finish figure setup
for i = 1:3
    figure(f(i));
    title('Initial conditions influence');
    ylabel(labels(i));
    legend(legendArray);
end

% ---------------------------------------------------------------
% ---------------------- R value influence ----------------------
% ---------------------------------------------------------------

% Array of R values to test
Rset = num2cell(4:10:64);

% Initial condition
x0 = [10, 15]';

% Figures
fx1 = figure;
hold on;
fx2 = figure;
hold on;
fu = figure;
hold on;
f = [fx1, fx2, fu];

% Legend array
legendArray = makeLegend(Rset, 'R');
```

```matlab

% For all R values
for i = 1:length(Rset)
    R = Rset{i};
    color = colors(i);

    % Calculate all Ki
    K = calculateK(A, B, Q, R, F, N);

    % The performance index
    Jo = perfIndex(x0, K{1});

    % Print results
    fprintf("R = %2d\t--->\tJo = %f\n", R, Jo);

    % Calculate x and u
    [x, u] = calcXandU(x0, N, A, B, R, K);

    % Plot everything
    figure(fx1);
    stairs(x(1, :), color);
    figure(fx2);
    stairs(x(2, :), color);
    figure(fu);
    stairs(u, color);
end

% Finish figure setup
for i = 1:3
    figure(f(i));
    title('R value influence');
    ylabel(labels(i));
    legend(legendArray);
end

% -----------------------------------------------------------
% -------------------- K elements settling --------------------
% -----------------------------------------------------------

% Initial values
```

```
x0 = [10, 5]';
R = 4;

% Calculate all Ki and get their elements
K = cell2mat(calculateK(A, B, Q, R, F, N)');
K11 = K(1:2:end, 1);
K12 = K(1:2:end, 2);
K21 = K(2:2:end, 1);
K22 = K(2:2:end, 2);

% Plot everything
fK = figure;
hold on;
stairs(K11, 'r');
stairs(K12, 'g');
stairs(K21, 'b');
stairs(K22, 'c');
title('K elements');
ylabel('K');
legend('K11', 'K12', 'K21', 'K22');
```

Do sprawdzania, czy macierze są symetrycznie i dodatnio określone (lub półokreślone) używana była funkcja `testMat` oraz funkcje pomocnicze `GreaterThan` i `GreaterThanEqual`.

Plik `testMat.m`:

```
function res = testMat(mat, op)
% Function testing if matrix 'mat' is symmetrical
% and positive-definite/semidefinite.
%
% mat - matrix to test
% op  - function returning true if the number satisfies
% the condition and false otherwise.
% Use GreaterThan(0) to test definiteness and GreaterEqualThan(0)
% to test semidefiniteness
% res - result (true or false)

    res = false;

    % Test if 'mat' is symmetrical
    if ~isequal(mat, mat')
```

```
        fprintf('Matrix %s is not symmetrical', inputname(1))
        return;
    end

    % Test definiteness/semidefinitenessh
    for i = 1:size(mat, 1)
        D = mat(1:i, 1:i);
        if ~op(det(D))
            fprintf('Matrix %s is not definite/semidefinite', inputname(1))
            return;
        end
    end

    res = true;
end
```

Plik `GreaterThan.m`:

```
function op = GreaterThan(val)
% Function returning pointer to function determining if its argument is
% greater than 'val'.

    op = @(x) x > val;
end
```

Plik `GreaterThanEqual.m`:

```
function op = GreaterThanEqual(val)
% Function returning pointer to function determining if its argument is
% greater than or equal 'val'.

    op = @(x) x >= val;
end
```

Do sprawdzenia sterowalności służyła funkcja `isControllable`.

Plik `isControllable.m`:

```
function res = isControllable(A, B)
% Function determining if system described by matrices A and B is
% controllable.
```

```

    n = size(A, 1);
    S = [];

    for i = 0:n-1
        S = [S, A^i*B];
    end

    if rank(S) == n
        res = true;
    else
        res = false;
    end
end
```

Macierze $\boldsymbol{K}_i$ wyznaczane były w funkcji `calculateK.m`.

Plik `calculateK.m`:

```
function K = calculateK(A, B, Q, R, F, N)
% Function calculating all K matrices

    K{N+1} = F;
    for i = N:-1:1
        K{i} = A'*(K{i+1} - K{i+1}*B*(R + B'*K{i+1}*B)^(-1)*B'*K{
            i+1})*A + Q;
        % If matrix is not symmetrical
        if ~isequal(K{i}, K{i}')
            K{i} = (K{i} + K{i}')/2;
        end
    end
end
```

Wskaźnik jakości wyliczany był w funkcji `perfIndex`.

Plik `perfIndex`:

```
function ret = perfIndex(x0, K0)
% Function returning optimum performance index

    ret = x0'*K0*x0 / 2;
end
```

Wartości $\boldsymbol{x}$ i $\boldsymbol{u}$ wyliczane były w funkcji `calcXandU`.

Plik `calcXandU.m`:

```
function [x, u] = calcXandU(x0, N, A, B, R, K)
    % Allocate memory for x and u for speed
    u{N} = 0;
    x{N+1} = 0;

    % Calculate x and u
    x{1} = x0;
    for i = 1:N
        u{i} = -(R + B'*K{i+1}*B)^(-1)*B'*K{i+1}*A*x{i};
        x{i+1} = A*x{i} + B*u{i};
    end

    x = [x{:}];
    u = [u{:}];
end
```

Dodatkowo do automatycznego tworzenia legendy wykorzystywana była funkcja `makeLegend`.

Plik `makeLegend`:

```
function legendArray = makeLegend(values, name)
% Function returning cell array of legend items basing on values
   set and name.

    for i = 1:length(values)
        str = string(num2str(values{i}));
        legendArray{i} = name + " = [";
        for j = 1:length(str)-1
            legendArray{i} = legendArray{i} + str(j) + ", ";
        end
        legendArray{i} = legendArray{i} + str(length(str)) + "]";
    end
end
```