MATLAB Implementation of Direct and Indirect Shooting Methods to Solve an Optimal Control Problem With State Constraints

Submitted: 3rd March 2020; accepted: 27th October 2020

Andrzej Karbowski

DOI: 10.14313/JAMRIS/1-2021/6

Abstract:
The paper presents a general procedure to solve numerically optimal control problems with state constraints. It is used in the case, when the simple time discretization of the state equations and expressing the optimal control problem as a nonlinear mathematical programming problem is too coarse. It is based on using in turn two multiple shooting BVP approaches: direct and indirect. The paper is supplementary to the earlier author's paper on direct and indirect shooting methods, presenting the theory underlying both approaches. The same example is considered here and brought to an end, that is two full listings of two MATLAB codes are shown.

Keywords: *optimal control, numerical methods, state constraints, shooting method, multiple shooting method, boundary value problem, MATLAB*

1. Introduction

We want to determine a piecewise continuous control function $u(t) \in \mathbb{R}$, $t_0 \leq t \leq t_f$, which minimizes the Mayer functional

$$J(u) = g(x(t_f)) \tag{1}$$

subject to the constraints

$$\dot{x}(t) = f(x(t), u(t), t), \quad t_0 \leq t \leq t_f \tag{2}$$

$$x(t_0) = x_0 \tag{3}$$

$$S(x(t), t) \leq 0, \quad t_0 \leq t \leq t_f \tag{4}$$

Here, $x(t) \in \mathbb{R}^n$ denotes a vector of state variables, constraint (3) describes initial conditions, (4) is a nonstationary inequality constraint on current values of state. It is assumed, that the function S is sufficiently continuously differentiable. The function $f(x(t), u(t), t)$ is allowed to be merely piecewise continuously differentiable with respect to time variable for $t \in [t_0, t_f]$. The final time t_f is fixed. Problems with free final time or problems with integral terms in the performance index (Bolza or Lagrange) can be easily transformed into a problem of the type (1)-(4) by means of additional state variables.

For the sake of simplicity of the presentation we do not consider constraints on control. However, the methodology presented both in [4] and here is general and may be used to solve also such problems.

The simplest method to solve the problem (1)-(4) consists in time discretization of the state equation (2) with equal length time stages. Then state and control trajectories are represented by vectors of real numbers (i.e., they are piecewise constant) and from the differential state equations difference equations are obtained. The latter are treated as a set of equality constraints in a static nonlinear programming problem with the performance index (1) and an additional set of inequality constraints stemming from the constraint on the current state (4). In this way the optimal control problem has been converted to a nonlinear mathematical programming problem. Unfortunately, there is a big class of problems, e.g. [8], in the aerospace, medical apparatus domain, chemical and nuclear reactors, robot manipulators, in which a high accuracy of the solution is crucial and the above approach is unacceptable as it delivers controls which are too coarse.

In such cases we apply direct and indirect shooting methods [8], [7]. Typically, the initial guesses of the optimal state and control trajectories are generated by applying the direct method [2]. On the basis of them the possible intervals of the activity of constraints are determined. Then the necessary conditions of optimality are analytically derived, what leads to a boundary value problem (BVP) [9] for ordinary differential equations with state and adjoint variables [10].

In the previous paper [4] the basic theory of the shooting approaches was presented. As an illustration an example taken from Jacobson and Lele's paper [3] was used. However, there was no space there to present the codes giving the presented results. In this paper MATLAB procedures which delivered the results described in [4] are shown. It should help the possible readers to understand the theory on both shooting approaches and to write their own codes.

2. The Optimal Control Problem

Let us consider the following optimal control problem taken from Jacobson and Lele's paper [3]:

$$\min_{u(\cdot)} \int_0^1 \left(x_1^2(t) + x_2^2(t) + 0.005u^2(t) \right) dt \tag{5}$$

where for $t \in [0, 1]$

$$u(t) \in \mathbb{R} \tag{6}$$

$$\dot{x}_1(t) = x_2(t), \qquad x_1(0) = 0 \tag{7}$$

$$\dot{x}_2(t) = -x_2(t) + u(t), \qquad x_2(0) = -1 \tag{8}$$

$$x_2(t) \leq 8(t-0.5)^2 - 0.5 \tag{9}$$

To apply shooting methods we have to rewrite this optimization problem as a Mayer problem, introducing an additional state variable x_3 governed by the state equation:

$$\dot{x}_3(t) = x_1^2(t) + x_2^2(t) + 0.005u^2(t), \qquad x_3(0) = 0 \tag{10}$$

The objective function (5) will be replaced with

$$g(x(t_f)) = x_3(1) \tag{11}$$

which will be minimized.
So, now our problem has the following form:

$$\min_{u(.)} g(x(t_f)) = x_3(1) \tag{12}$$

where for $t \in [0,1]$

$$u(t) \in \mathbb{R} \tag{13}$$

$$\dot{x}(t) = f(x(t), u(t)) =$$

$$= \begin{bmatrix} x_2(t) \\ -x_2(t) + u(t) \\ x_1^2(t) + x_2^2(t) + 0.005u^2(t) \end{bmatrix} \tag{14}$$

$$S(x(t), t) = x_2(t) - 8(t-0.5)^2 + 0.5 \leq 0, \tag{15}$$

with

$$x(t_0) = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \tag{16}$$

3. Direct Multiple Shooting Technique

3.1. Formulation

In the direct approach a control interval is divided into a certain number of subintervals, on which a Cauchy problem is solved by an ordinary differential equation (ODE) solver. The initial conditions are generated iteratively by an optimizer, constraints on state are checked in the discretization points of the time interval.

The optimal control problem is transformed into a nonlinear programming problem [10], [4], [5] through a parametrization of control $u(t)$ on the subintervals of the control interval. For example, $u(t)$ may be: piecewise constant, piecewise linear or higher order polynomials, linear combination of some basis functions, e.g. B-splines. We apply the simplest type of the parametrization: piecewise constant, that is, we take:

$$t_0 < t_1 < t_2 < \dots t_{p-1} < t_p = t_f \tag{17}$$

$$u(t) = u_k, \quad t \in I_k = [t_k, t_{k+1}), \quad \text{for } k = 0, 1, \dots, p-1 \tag{18}$$

where $u_k \in \mathbb{R}$.

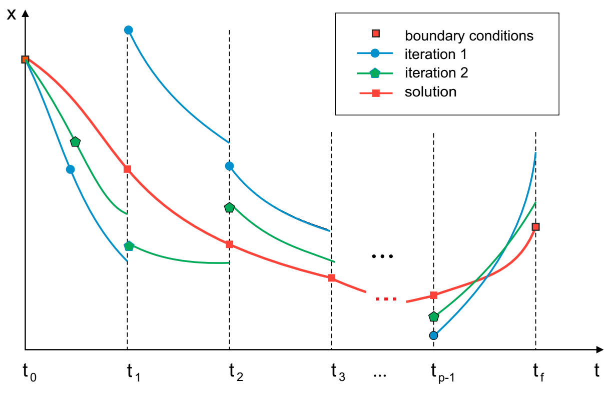

Fig. 1. Direct shooting method

The basic idea is to simultaneously integrate numerically the state equations (14) on the subintervals I_k for guess initial points

$$\chi_k = x(t_k) \tag{19}$$

Then the values obtained at the ends of subintervals - we will denote them by $x(t_{k+1}; \chi_k, u_k)$ - are compared with the guesses χ_{k+1}. The differential equations, initial and end points conditions and path constraints define the constraints of the nonlinear programming problem, that is the problem (1)-(4) is replaced with:

$$\min_{\chi, u} g(\chi_p) \tag{20}$$

$$\chi_{k+1} - x(t_{k+1}; \chi_k, u_k) = 0, \quad k = 0, \dots, p-1 \tag{21}$$

$$\chi_0 = x_0 \tag{22}$$

$$S(\chi_k, t_k) \leq 0, \quad k = 0, \dots, p \tag{23}$$

where $x(t_{k+1}; \chi_k, u_k)$ for $k = 0, \dots, p-1$ is the solution of ODE:

$$\dot{x}(t) = f(x(t), u_k, t), \quad t_k \leq t \leq t_{k+1}, \tag{24}$$

$$x(t_k) = \chi_k, \quad k = 0, \dots, p-1, \tag{25}$$

at $t = t_{k+1}$ (see Fig. 1).

This nonlinear programming problem can be solved by any continuous constrained optimization solver.

3.2. MATLAB Solution

The problem (12)-(16) was solved by the direct shooting method, described in Sec. 3.1, for $p = 20$ time subintervals of equal length, with the help of two MATLAB functions: `ode45` (ODE solver; medium order Runge-Kutta method) and `fmincon` (constrained nonlinear multivariable optimization solver). The code implementing it in MATLAB has the name `shooDir.m` and is shown below. There T denotes the array of time instants, U is an array with (discretized in time) control trajectory, X with state trajectory. The vector z of decision variables is a concatenation of vectors representing, in turn, subsequent guesses of initial values of state variables at starting points of subintervals and control, that is:

$$z = [\chi_{1,0}, \chi_{1,1}, \ldots, \chi_{1,p}, \chi_{2,0}, \chi_{2,1}, \ldots, \chi_{2,p},$$
$$\chi_{3,0}, \chi_{3,1}, \ldots, \chi_{3,p}, u_0, u_1, \ldots, u_{p-1}]^T \qquad (26)$$

.

```
function [T,U,X,gopt]=shooDir(pval)
%
% Implementation of the direct
% shooting method
%
% PARAMETERS:
% pval - number of subintervals
%        (e.g. 20)
% T - array of subsequent time
%     points (i.e., times of shoots)
% U - trajectory of optimal control
%     (discretized)
% X - state trajectory
% gopt - optimal value
%        of objective function
%
% CALLING:
%   >> [T,U,X,gopt]=shooDir(pval)
%   For example:
%      >> [T,U,X,gopt]=shooDir(20)
%
   global p Dt U
   p = pval; Dt = 1/p;
   %
   % starting point for (time
   % discretized) trajectory
   % optimization
   %
   xs = zeros(p+1,1);
   us = zeros(p,1);
   zinit = [xs; xs; xs; us];
   U=[];
   %
   % initial conditions for state
   % equation
   %
   Aeq = zeros(3,4*p+3);
   Aeq(1,1) = 1; Aeq(2,p+2) = 1;
   Aeq(3,2*p+3) = 1;
   beq = [0;-1;0];
   %
   % inequality state constraint
   % discretized in time
   %
   A=zeros(p+1,4*p+3);
   for k=1:p+1
      A(k,p+1+k) = 1;
      b(k) = 8*((k-1)*Dt-0.5)^2-0.5;
   end
   tic
   [zo,gopt]=fmincon(@fun, zinit,...
                A, b, Aeq, beq,...
                [],[], @nonlcon,...
                optimset('Display',...
                'iter','TolFun',1.e-4,...
                'MaxFunEvals',100000));
   toc
   %
   % The lines below in this
   % function are only for output
   % purposes (including plots)
   %
   T=[];
   for k=1:p+1
      T(k) = Dt*(k-1);
   end
   f(1)=figure();
   X=[zo(1:p+1), zo(p+2:2*(p+1)), ...
                 zo(2*p+3:3*(p+1))];
   h1=plot(T,X(:,1),'b--',...
           T,X(:,2),'g-',...
           T,8*(T-0.5).^2-0.5,'m-.');
   grid on
   for i=1:3
      h1(i).LineWidth=3;
   end
   h1(2).Color=[0 0.6 0.3];
   legend('x_1','x_2','S(x,t)=0');
   xlabel('t'); ylabel('x');
   txt={'INFEASIBLE','            ',...
       ' x_2 REGION  '};
   text(0.4,0.8,txt,'Color','m');
   f(2)=figure();
   plot(T(1:end-1), U,'LineWidth',3);
   grid on;
   legend('u optimal - direct method'
     ↪ );
   xlabel('t'); ylabel('u');
end

function [c,ceq]=nonlcon(z)
%
% The function calculates
% for trajectories on subsequent
% subintervals the discrepancies
% between guesses of initial points
% and the end points of shoots
% (from the previous guessed points)
%
   global p Dt U
   for i=1:3
      for k=1:p+1
         xs(i,k) = z((i-1)*(p+1)+k);
      end
   end
   for k=1:p
      U(k) = z(3*(p+1)+k);
   end
   c=[]; ceq=[];j=1;
   for k=1:p
      [T,X]=ode45(@(t,x) f(t,x,...
           U(k)),[(k-1)*Dt k*Dt],...
```

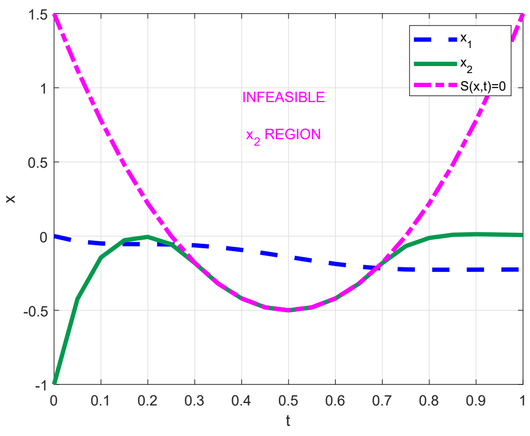

Fig. 2. Optimal state trajectories obtained from the direct shooting method

```
                    [xs(1,k);xs(2,k);...
                    xs(3,k)]);
            for  i=1:3
                ceq(j) = X(end,i)-xs(i,k+1);
                j=j+1;
            end
        end
end

function dxdt = f(t,x,u)
%
%   state transformation function
%   (rhs of the state equation);
%   here x indices denote coordinates
%
    dxdt = zeros(3,1);
    dxdt(1) = x(2);
    dxdt(2) = -x(2)+u;
    dxdt(3) = x(1)^2+x(2)^2+0.005*u^2;
end

function fz=fun(z)
global p
    fz = z(3*(p+1));
end
```

The calculations were performed on a PC with Intel Core i7-2600K CPU@3.40 GHz processor under MATLAB R2020b. After about 50 s we obtained the performance index value equal 0.1708. The resulting trajectories of the state variables x_1 and x_2 are presented in Fig. 2, of the state variuable x_3 in Fig. 6 and of the optimal control in Fig. 5.

Analyzing the resulting x_2 state variable trajectory we may see, that it contains one boundary arc. To find its precise course the indirect shooting method will be used.

4. Indirect Multiple Shooting Technique

4.1. Formulation

In the indirect approach BVP concerns not only state equations, but also the equations describing adjoint variables $\eta(t)$. It means, that for an optimal control problem, before using a solver, we have to make a kind of preprocessing on the paper, based on the appropriate theory [1]. In particular, we have to determine the number of switching points s, where the state trajectory enters and leaves the constraint boundary. The optimal control at a given time instant is a function of the current value of state and adjoint variables (i.e., we provide a control law). Only general formulas should be given, their parameters: Lagrange multipliers, initial values of adjoint variables, their jumps and the concrete values of switching points (i.e., times) will be the subject of optimization.

The basic idea of the numerical treatment of such problems by multiple shooting technique is to consider the switching conditions as boundary conditions to be satisfied at some interior multiple shooting nodes [6]. Thus, the problem is transformed into a classical multipoint BVP [9], [6]:

Determine a piecewise smooth vector function $y(t) = [x(t), \eta(t)]$, which satisfies

$$\dot{y}(t) = f(y(t), u(t), t), \quad t_0 \le t \le t_f, \tag{27}$$

$$u = u_k(y(t), t), \quad \tau_k \le t < \tau_{k+1}, \quad k = 0, \ldots, s, \tag{28}$$

$$y(\tau_k^+) = h_k(y(\tau_k^-), \gamma_k), \quad k = 1, \ldots, s, \tag{29}$$

$$y(t_0) = \begin{bmatrix} x_0 \\ \eta_0 \end{bmatrix} \tag{30}$$

$$r_i(y(t_f)) = 0, \quad i = 1, \ldots, n_f, \tag{31}$$

$$\tilde{r}_k(\tau_k, y(\tau_k^-)) = 0, \quad k = 1, ..., s. \tag{32}$$

In this formulation, $\eta_0, \gamma, \tau_k, k = 1, \ldots, s$ are unknown parameters of the problem, where the latter satisfy

$$t_0 =: \tau_0 < \tau_1 < \tau_2 < \cdots < \tau_s < \tau_{s+1} := t_f \tag{33}$$

The trajectory may possess jumps of size given by Eq. (29). If a coordinate of $y(t)$ is continuous (as all state variables), then its h_k is identity. The boundary conditions and the switching conditions are described by Eqs. (30)-(32).

In every time stage k the numerical integration over the interval $[\tau_k, \tau_{k+1}]$ is done by any conventional Cauchy problem (aka initial value problem - IVP) ODE solver with stepsize control. The resulting system of nonlinear equations (29)-(32) can be solved numerically by a quasinewton method, e.g., from the Broyden family.

The derivation of optimality conditions for our problem (12)-(16) is presented in [4]. They lead to the following set of equations (here: $s = 2, t_1 \triangleq \tau_1, t_2 \triangleq$

τ_2):

$$\dot{x}_1(t) = x_2(t) \tag{34}$$
$$\dot{x}_2(t) = -x_2(t) + u(t) \tag{35}$$
$$\dot{x}_3(t) = x_1^2(t) + x_2^2(t) + 0.005u^2(t) \tag{36}$$
$$\dot{\eta}_1(t) = 2x_1(t) \tag{37}$$
$$\dot{\eta}_2(t) = 2x_2(t) - \eta_1(t) + \eta_2(t) - \mu_S(t) \tag{38}$$

where

$$u(t) = \begin{cases} x_2(t) + 16(t-0.5), & t \in [t_1, t_2] \\ 100\eta_2(t), & t \notin [t_1, t_2] \end{cases} \tag{39}$$

$$\mu_S(t) = \begin{cases} \eta_2(t) - 0.01 \cdot u(t), & t \in [t_1, t_2] \\ 0, & t \notin [t_1, t_2] \end{cases} \tag{40}$$

with

$$x_1(0) = 0 \tag{41}$$

$$x_2(0) = -1 \tag{42}$$

$$x_3(0) = 0 \tag{43}$$

$$x_1(t_1^+) = x_1(t_1^-), \;\; x_2(t_1^+) = x_2(t_1^-), \;\; x_3(t_1^+) = x_3(t_1^-) \tag{44}$$

$$x_1(t_2^+) = x_1(t_2^-), \;\; x_2(t_2^+) = x_2(t_2^-), \;\; x_3(t_2^+) = x_3(t_2^-) \tag{45}$$

$$\eta_1(t_2^+) = \eta_1(t_2^-), \;\; \eta_2(t_2^+) = \eta_2(t_2^-) \tag{46}$$

The junction (jump) conditions will be as follows:

$$\eta_1(t_1^+) = \eta_1(t_1^-) \tag{47}$$
$$\eta_2(t_1^+) = \eta_2(t_1^-) + \gamma \tag{48}$$

and the final ones:

$$\eta_1(1) = 0 \tag{49}$$

$$\eta_2(1) = 0 \tag{50}$$

The switching (tangency) condition in our problem has the form:

$$S(x_2(t_1^-), t_1^-) = x_2(t_1^-) - 8(t_1^- - 0.5)^2 + 0.5 = 0 \tag{51}$$

4.2. MATLAB Solution

Putting all these things together and expressing them in the format required by the indirect shooting method, described in [4], we get a system of nonlinear equations, with the vector of unknowns:

$$z = [\eta_{10}, \eta_{20}, t_1, t_2, \gamma] \tag{52}$$

to be satisfied at final and switching points, resulting from the solution of ODEs with time functions: $x_1(t), x_2(t), x_3(t), \eta_1(t), \eta_2(t)$, defined on three intervals: $[0, t_1]$, $[t_1, t_2]$, $[t_2, 1]$, where starting points are defined by initial and junction/jump conditions (41)-(48). The set of nonlinear equations to be solved was made of conditions (49)-(51):

$$F(z) = F_{err}(z) = \begin{bmatrix} S(x_2(t_1^-), t_1^-) \\ \eta_1(1) \\ \eta_2(1) \end{bmatrix} = 0 \tag{53}$$

This problem was solved under MATLAB with the help of two MATLAB functions: `ode45` (mentioned above) and `fsolve` (a solver of systems of nonlinear equations of several variables). The resulting code has the name `shooIndir.m` and is presented below:

```
function [T,U,Y,gopt]=shooIndir
%
% Implementation of the indirect
% shooting method
%
% PARAMETERS:
% T - array of subsequent time
%     points of control interval
% U - trajectory of optimal control
%     (discretized)
% Y - state and adjoint variables
%     trajectories (discretized)
% gopt - optimal value
%        of objective function
%
% CALLING:
%   >> [T,U,Y,gopt]=shooIndir
%
  global TTot YTot

  zs=[ 0; 0;  0.33; 0.66; 1];
  tic
  [zo,Fo] = fsolve(@ShooF, zs,
      ↪ optimset('Display','iter'));
  T=TTot; Y=YTot;
  toc
  %
  % The lines below in this
  % function are only for output
  % purposes (including plots)
  %
  stT=size(T);
  for t=1:stT
     if T(t) < zo(3) || T(t) > zo(4)
        U(t)= 100*Y(t,5);
        MuS(t)= 0.;
     else
        U(t)= Y(t,2)+16*(T(t)-0.5);
        MuS(t)= Y(t,5)-0.01*U(t);
     end
  end
  gopt=Y(end,3);
  f(1)=figure();
  h1=plot(T, Y(:,1),'b--',...
          T, Y(:,2),'g-',...
          T,8*(T-0.5).^2-0.5,'m-.');
  for i=1:3
```

```
        h1(i).LineWidth=3;
    end
    h1(2).Color=[0 0.6 0.3];
    grid on;
    legend('x_1','x_2','S(x,t)=0');
    xlabel('t'); ylabel('x');
    txt={'INFEASIBLE','           ',...
       ' x_2 REGION  '};
    text(0.4,0.8,txt,'Color','m');

    f(2)=figure();
    h2=plot(T, Y(:,4),'r-',...
            T, Y(:,5),'b--',...
            T, MuS','g-.');
    h2(3).Color=[0 0.6 0.3];
    grid on;
    legend('\eta_1','\eta_2','\mu_S');
    xlabel('t'); ylabel('\eta,\mu_S');
    for i=1:3
        h2(i).LineWidth=3;
    end

    f(3)=figure();
    plot(T, U,'LineWidth',3);
    grid on;
    legend('u optimal - indirect
      ↪ method');
    xlabel('t'); ylabel('u');
end

function Ferr=ShooF(z)
%
% The function calculates lhs
% of the ultimate set of (static)
% nonlinear equations:
%       Ferr(z)=0
%
    global TTot YTot

    eT10 = z(1);
    eT20 = z(2);
    t1 = z(3);
    t2 = z(4);
    gamma = z(5);
    TTot=[]; YTot=[];
    %
    % Region 1 (state constraint
    % inactive)
    %
     [T,Y]=ode45(@(t,y) f(t,y,1),...
              [0 t1],...
              [0; -1; 0; eT10; eT20]);
    TTot = T; YTot = Y;
    x11 = Y(end,1);
    x21 = Y(end,2);
    x31 = Y(end,3);
    eT11 = Y(end,4);
    eT21 = Y(end,5);
    x2b1 = 8*(t1-0.5)^2-0.5;
    Ferr(1) = x21-x2b1;
    %
    % Region 2 (state constraint
    % active)
    %
     [T,Y]=ode45(@(t,y) f(t,y,2),...
              [t1 t2],...
              [x11; x21; x31; ...
               eT11; eT21+gamma]);
    TTot =[TTot; T]; YTot =[YTot; Y];
    x12 = Y(end,1);
    x22 = Y(end,2);
    x32 = Y(end,3);
    eT12 = Y(end,4);
    eT22 = Y(end,5);
    %
    % Region 3 (state constraint
    % inactive)
    %
     [T,Y]=ode45(@(t,y) f(t,y,3),...
              [t2 1],...
              [x12; x22; x32; ...
               eT12; eT22]);
    TTot =[TTot; T]; YTot =[YTot; Y];
    Ferr(2) = Y(end,4);
    Ferr(3) = Y(end,5);
end

function dydt = f(t,y,region)
%
% extended state transformation
% function (rhs of the 1st order
% ODE in state and adjoint
% variables space);
% here: y=[x1, x2, x3, eta1, eta2]
%
    dydt = zeros(5,1);
    dydt(1) = y(2);
    switch region
        case {1, 3}
            u = 100*y(5);
            muS = 0;
        case 2
            u = y(2)+16*(t-0.5);
            muS = y(5)-0.01*u;
    end
    dydt(2) = -y(2)+u;
    dydt(3) = y(1)^2+y(2)^2+0.005*u^2;
    dydt(4) = 2*y(1);
    dydt(5) = 2*y(2)-y(4)+y(5)-muS;
end
```

The calculations took only about 1.5 s, the obtained optimal value of the performance index was 0.1698. The resulting state x_1, x_2 and adjoint variables η_1, η_2 trajectories are presented in Figs. 3-4, and the optimal control u and state variable x_3 trajectories obtained from both methods are depicted, respectively, in Fig. 5, and 6. One may see, that indeed, the solution delivered by the indirect shooting method is much more precise than that of the direct one, despite that the number of unknowns in the indirect method was much smaller than the dimension of the decision vector in the direct method (5 vs. 83) and the time of calculations was

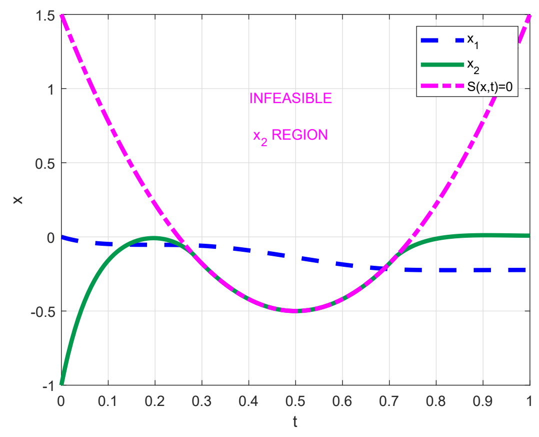

Fig. 3. Optimal trajectories of the state variables x_1 and x_2 obtained from the indirect shooting method.

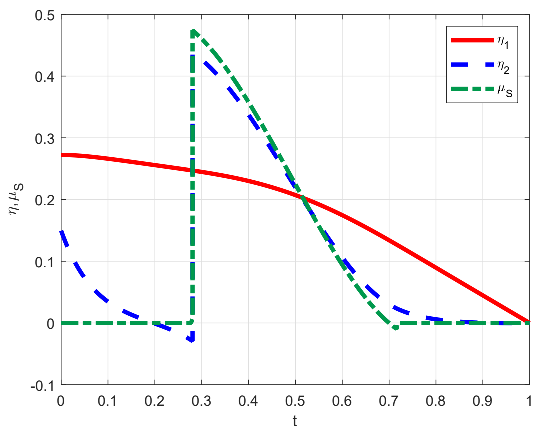

Fig. 4. Optimal trajectories of the adjoint variables η_1, η_2 and the Lagrange multiplier function for state constraint μ_S obtained from the indirect shooting method.

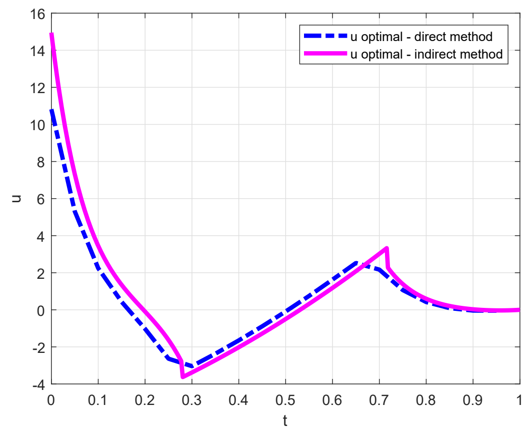

Fig. 5. Optimal control trajectories obtained from both direct and indirect shooting methods

much shorter (1.5 s vs. 50 s).

5. Conclusion

In the paper an optimal control problem with one state constraint was considered. The proposed solution procedure consists in the application of two shooting approaches together: first the direct shooting method to find an approximation of the optimal control trajectory and then - on the basis of the assessment of the intervals of the activity of the state constraint, obtained from the direct method and the analysis of conditions of optimality - the indirect shooting method. The latter may seem quite complicated at the beginning and it may be difficult for the novices to imagine how to translate them into a computer code. The two listings of the corresponding, working codes in MATLAB, given in the paper, should help in this.

Both files ('shooDir.m' and 'shooIndir.m') may be downloaded from the author's Web page: http://www.ia.pw.edu.pl/~karbowsk/shooting.

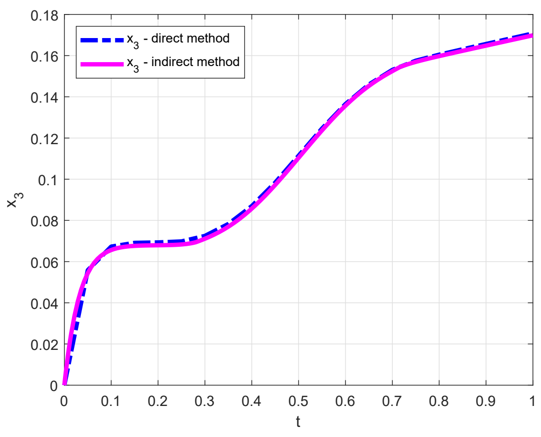

Fig. 6. Optimal trajectories of the (artificial, representing the objective function growth) state variable x_3 obtained from both direct and indirect shooting methods

AUTHOR
Andrzej Karbowski – Institute of Control and Computation Engineering, Warsaw University of Technology, Warsaw, Poland, e-mail: andrzej.karbowski@pw.edu.pl, www: http://www.ia.pw.edu.pl/~karbowsk.

REFERENCES

[1] A. E. Bryson, W. F. Denham, and S. E. Dreyfus, "Optimal Programming Problems with Inequality Constraints I: Necessary Conditions for Extremal Solutions", *AIAA Journal*, vol. 1, no. 11, 1963, 2544–2550, 10.2514/3.2107.

[2] M. Gerdts, "Direct Shooting Method for the Numerical Solution of Higher-Index DAE Optimal Control Problems", *Journal of Optimization Theory and Applications*, vol. 117, no. 2, 2003, 267–294, 10.1023/A:1023679622905.

[3] D. Jacobson and M. Lele, "A transformation technique for optimal control problems with a state variable inequality constraint", *IEEE Transactions on Automatic Control*, vol. 14, no. 5, 1969, 457–464, 10.1109/TAC.1969.1099283.

[4] A. Karbowski, "Shooting Methods to Solve Optimal Control Problems with State and Mixed Control-State Constraints". In: R. Szewczyk, C. Zieliński, and M. Kaliczyńska, eds., *Challenges in Automation, Robotics and Measurement Techniques*, Springer: Cham, 2016, 189–205, 10.1007/978-3-319-29357-8_17.

[5] H. Maurer and W. Gillessen, "Application of multiple shooting to the numerical solution of optimal control problems with bounded state variables", *Computing*, vol. 15, no. 2, 1975, 105–126, 10.1007/BF02252860.

[6] H. J. Oberle, "Numerical solution of minimax optimal control problems by multiple shooting technique", *Journal of Optimization Theory and Applications*, vol. 50, no. 2, 1986, 331–357, 10.1007/BF00939277.

[7] R. Pytlak, J. Blaszczyk, A. Karbowski, K. Krawczyk, and T. Tarnawski, "Solvers chaining in the IDOS server for dynamic optimization". In: *52nd IEEE Conference on Decision and Control*, 2013, 7119–7124, 10.1109/CDC.2013.6761018.

[8] S. Sager, *Numerical methods for mixed-integer optimal control problems*, Ph.D. Thesis, University of Heidelberg, 2005.

[9] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, Springer New York: New York, NY, 2002, 10.1007/978-0-387-21738-3.

[10] O. von Stryk and R. Bulirsch, "Direct and indirect methods for trajectory optimization", *Annals of Operations Research*, vol. 37, no. 1, 1992, 357–373, 10.1007/BF02071065.