DYNOPT - Dynamic Optimisation Code for MATLAB

M. Čižniar*, D. Salhi†, M. Fikar*, M. A. Latifi†

*FCFT STU Radlinskeho 9, 812 37 Bratislava, Slovakia; †LSGC, CNRS-ENSIC, B. P. 451, 1 rue Grandville, 54001 Nancy Cedex, France; e-mail: miroslav.fikar@stuba.sk, cizniar@pobox.sk, latifi@ensic.inpl-nancy.fr

Objectives

- To implement orthogonal collocation method to dynamic optimisation problems.
- To develop a MATLAB package for solving DAE's, called *dynopt*.
- To implement analytical computation of gradients.
- To verify the ability of the *dynopt* package, to treat the problems of varying levels of difficulty.

Optimisation Method

General Formulation

$$\min_{\boldsymbol{u}(t)} J[\boldsymbol{x}(t_f)] \quad (1)$$

such that

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}[t, \boldsymbol{x}(t), \boldsymbol{u}(t)], \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \quad (2)$$

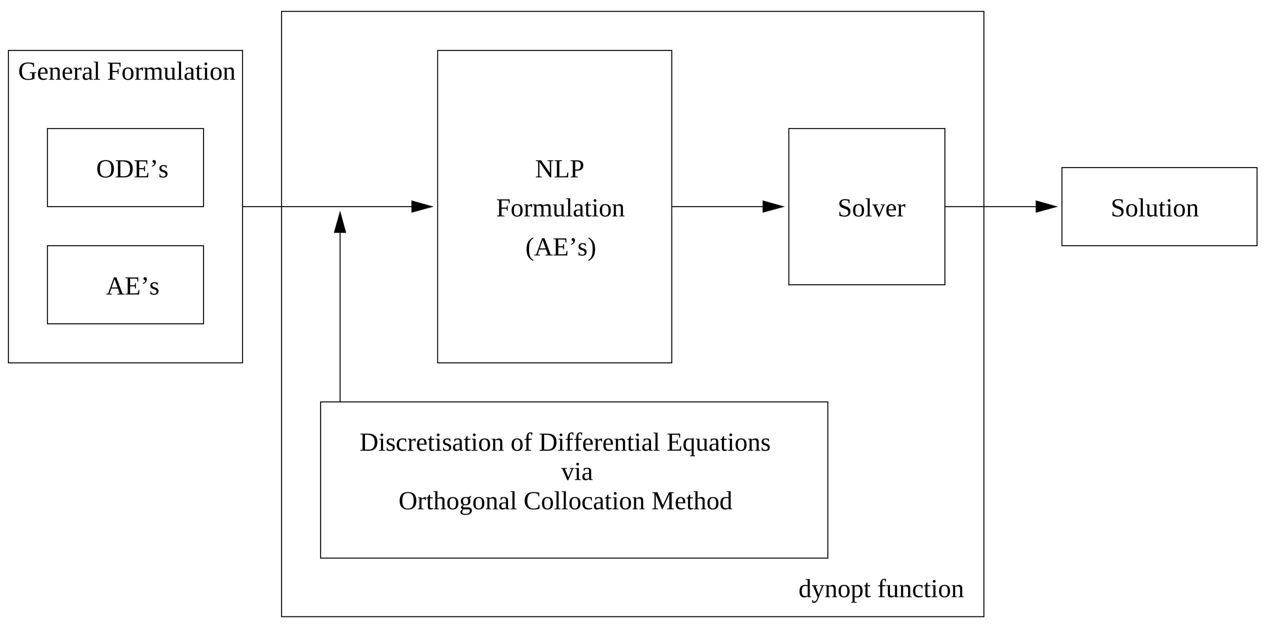
$$\boldsymbol{h}[t, \boldsymbol{x}(t), \boldsymbol{u}(t)] = \boldsymbol{0} \quad (3)$$

$$\boldsymbol{g}[t, \boldsymbol{x}(t)\boldsymbol{u}(t)] \le \boldsymbol{0} \quad (4)$$

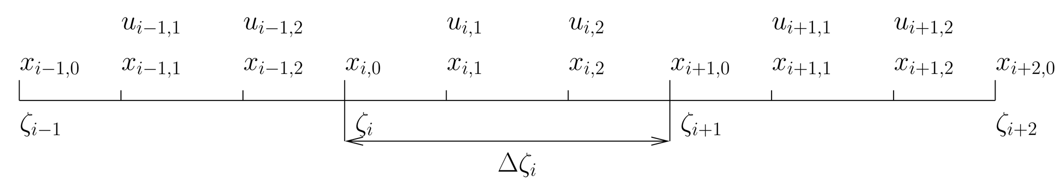
$$\boldsymbol{x}(t)^L \le \boldsymbol{x}(t) \le \boldsymbol{x}(t)^U \quad (5)$$

$$\boldsymbol{u}(t)^L \le \boldsymbol{u}(t) \le \boldsymbol{u}(t)^U \quad (6)$$

General Scheme



Discretisation of ODE's



Lagrange Functions

$$\boldsymbol{x}_{K+1}(t) = \sum_{j=0}^{K} \boldsymbol{x}_{ij}\phi_j(t); \quad \phi_j(t) = \prod_{k=0,j}^{K} \frac{(t - t_{ik})}{(t_{ij} - t_{ik})} \quad (7)$$

$$\boldsymbol{u}_K(t) = \sum_{j=1}^{K} \boldsymbol{u}_{ij}\theta_j(t); \quad \theta_j(t) = \prod_{k=1,j}^{K} \frac{(t - t_{ik})}{(t_{ij} - t_{ik})} \quad (8)$$

in element i, $i = 1, \ldots, \mathrm{NE}$

Example

Problem Formulation

$$\min_{\boldsymbol{u}(t)} J = \int_0^{t_f} (x_1^2 + x_2^2 + 0.005u^2)\mathrm{d}t \quad (9)$$

such that

$$\dot{x}_1 = x_2 \qquad x_1(0) = 0 \quad (10)$$

$$\dot{x}_2 = -x_2 + u \qquad x_2(0) = -1 \quad (11)$$

$$0 \ge x_2 - 8(t - 0.5)^2 + 0.5 \qquad \forall t \quad (12)$$

$$t_f = 1 \quad (13)$$

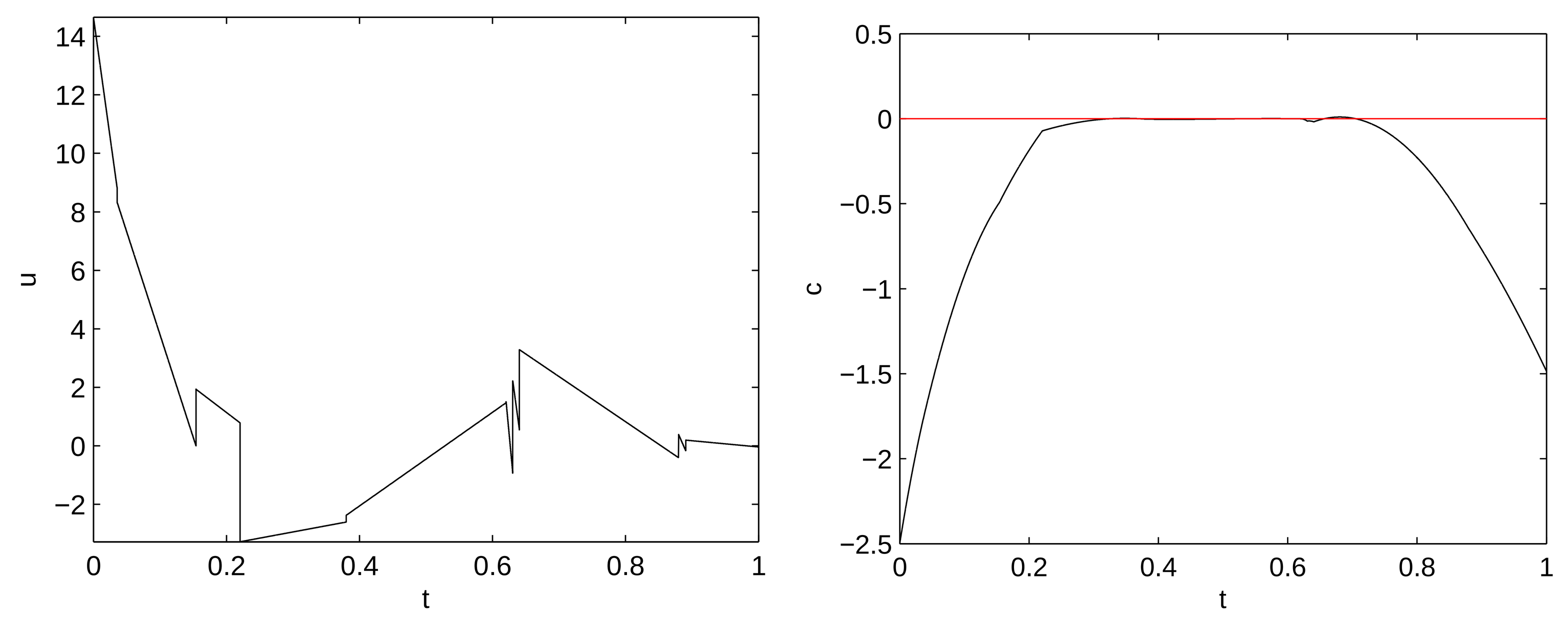
As the objective function is not in the Meyer form needed by *dynopt*, we define an additional differential equation

$$\dot{x}_3 = x_1^2 + x_2^2 + 0.005u^2, \quad x_3(0) = 0 \quad (14)$$

and rewrite the cost as

$$\min_{\boldsymbol{u}(t)} J = x_3(t_f) \quad (15)$$

Graphical Interpretation



left: optimal control profile, for given example, right: appropriate profile of constraints over full time interval

User–Interface

Process Function

```
function sys = process(t,x,flag,u)
switch flag,
   case 0
      sys = [x(2);
             -x(2)+u;
             x(1)^2+x(2)^2+0.005*u^2];
   case 1
      sys = [0 0 2*x(1);
             1 -1 2*x(2);
             0 0 0];
   case 2
      sys = [0 1 0.01*u];
   case 3
      sys = [0 0 0];
   case 4
      sys = [0;-1;0];
   otherwise
      error(['unhandled flag = ', ...
              num2str(flag)]);
end
```

Objective Function

```
function [f,Dft,Dfx,Dfu] = objfun(t,x,u)

f=x(3);
Dft=[];
Dfx=[0;0;1];
Dfu=[];
```

Constraints Function

```
function [c,ceq,Dct,Dcx,Dcu,Dceqt,Dceqx, ...
Dcequ] = confun(t,x,u)

c=x(2)-8*(t-0.5)^2+0.5;
ceq=[];
Dct=[-16*t+8];
Dcx=[0;1;0];
Dcu=[];
Dceqt=[];
Dceqx=[];
Dcequ=[];
```

Optimisation

After the problem has been defined by the above mentioned functions (*process*,*objfun*,*confun*), the user calls the *dynopt* function as follows:

```
opt = optimset('LargeScale','off', ...
      'Display','iter');
opt = optimset(opt,'GradObj', 'on', ...
      'GradConstr','on');
opt = optimset(opt,'TolFun',1e-5);
opt = optimset(opt,'TolCon',1e-5);
opt = optimset(opt,'TolX',1e-5);

u = [13 4 -0.5 -2 -1.8 0 0.5 2 0 0];
time = 0.1*ones(10,1);
[x,fval,exitflag,output]=dynopt( ...
1,2,4,2,time,1,u,[],[], ...
'objfun','confun','process',opt);
```

Conclusions

- The orthogonal collocation method has been developed and implemented within MATLAB environment. It is freely available at (http://www.kirp.chtf.stuba.sk/∼fikar)
- Two different methods of gradient computation have been used.
- The package has been succesfully tested on several examples from the literature.

Acknowledgements: The authors are pleased to acknowledge the financial support of the Scientific Grant Agency of the Slovak Republic under grant No. 1/0135/03. Part of this work has been completed under the French-Slovak program Stefanik (Dynamic and Global Optimisation of Processes) and within the framework of the Cultural and Educational Grant Agency of the Slovak Republic under the grant No. 3/3121/05.