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.

Objectives

General Formulation

\[
\min_{u(t)} J[x(t)]
\]

such that

\[
x(t) = f(t, x(t), u(t)), \quad x(0) = x_0
\]

\[
\dot{h}_i(t, x(t), u(t)) = 0
\]

\[
g_i(t, x(t))u(t) \leq 0
\]

\[
x(t)^T \leq x(t) \leq x(T)
\]

\[
u(t)^T \leq u(t) \leq u(T)
\]

Problem Formulation

\[
\min_{u(t)} J = \int_0^T (x_1^2 + x_2^2 + 0.005u^2) dt
\]

such that

\[
x_1 = x_2
\]

\[
x_2 = -x_2 + u
\]

\[
0 \geq x_2 - 0.5(t - 0.5)^2 + 0.5
\]

\[
t_f = 1
\]

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

\[
x_1 = x_1^2 + x_2^2 + 0.005u^2, \quad x_1(0) = 0
\]

and rewrite the cost as

\[
\min_{u(t)} J = x_1(t_f)
\]

Discretisation of ODE’s

\[
x_{k+1}(t) = \sum_{k=0}^{K} \sum_{j=1}^{J} \phi_{ij}(t_j) (x_{ij}(0) + \int_{t_{i-1}}^{t_i} \sum_{j=1}^{J} \phi_{ij}(\tau) u_{ij}(\tau) d\tau)
\]

\[
u_{ij}(t) = \sum_{j=1}^{J} \phi_{ij}(t_j) (t_j - t_{i-1})
\]

in element \( i, i = 1, \ldots, NE \)

Graphical Interpretation

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

User-Interface

Objective Function

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

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

Constraints Function

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

c=x(2)-8*(t-0.5)^2+0.5;

ceq=[];

Dct=[-16*t+8];

Dcu=[0;0;0];

Dceq=[];

Dceqx=[];

Dcequ=[];

end

Optimisation

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

\[
\text{opt = optimset('LargeScale','off', ...}
\]

\[
\text{'Display','iter'});
\]

\[
\text{opt = optimset(opt,'GradObj','on', ...}
\]

\[
\text{'GradConstr','on');}
\]

\[
\text{opt = optimset(opt,'Type','TrustRegion', ...}
\]

\[
\text{'TrustRegion',1e-6);}
\]

\[
u = [13.4 -0.5 -2.0 0.0 0.5 2.0 0.0];
\]

\[
\text{time = 0.1*ones(10,1);}
\]

\[
\text{[x,fval,exitflag,output]=dynopt(...}
\]

\[
\text{1,2,4,2,time,1,u,[]}, ...}
\]

\[
\text{'objfun','confun','process',opt);}
\]

Conclusions

The orthogonal collocation method has been developed and implemented within MATLAB environment. It is freely available at [http://www.kirp.ctbh.stuba.sk/~fikar](http://www.kirp.ctbh.stuba.sk/~fikar). The package has been successfully tested on several examples from the literature.

Two different methods of gradient computation have been used.