User Manual

OCDES (v1.4): OCDE Simulator

Xiao Zhao
Email: xiao.zhao@outlook.de.
07.2020

1. Overview

OCDES is a MATLAB-based tool that performs numerical integration to solve Optimization-Constrained Differential Equations (OCDE):

$$\dot{x} = f(x, v),\ x(0) = x_0, \tag{1a}$$
$$v \in \arg\min_{v} g(x, v), \tag{1b}$$
$$s.t.\ h_i(x, v) = 0,\ i = 1, \dots, M, \tag{1c}$$
$$l_j(x, v)\ \geq 0,\ \ j = 1, \dots, N \tag{1d}$$

$x \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. $f: \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^m$, $g: \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$, $h_i: \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$, $l_j: \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$ are at least second order continuously differentiable. OCDES provides efficient numerical solution of OCDE by using local optimality-based solution method. A sequence of DAE systems are generated and classical index-1 DAE simulator is applied to solve the derived DAE systems.

2. Basic Requirements

(1) Matlab, version 2014 or higher.
(2) Matlab Symbolic Toolbox

3. Installation

The simulator needs Matlab environment.

4. How to use OCDES

Please refer to the example demo1.m. General steps of using OCDES are:

(1) Define state variables x and optimization variables v in symbolic format.

(2) Define functions f, g, h, l in symbolic format.

(3) Give initial condition x(0) and initial guess of v(0).

(5) Specify options for solving the inner NLP for initialization

`opt_init.tol_act`	tolerance to check active inequality constraints
`opt_init.optimoptions`	Optimization options, cf. MATLAB optimoptions
`opt_sol.MaxNoUptActiveSet`	maximum number of updating active set
`opt_sol.tol_feasible`	feasibility tolerance

(6) Specify options for integration

tstart	starting time of simulation
tfinal	ending time of simulation
opt_sol.integrator	Selected integration
opt_sol.opt_integrator	Integration options

(7) Call `sOCDE_main.m` to solve the OCDE.

5. Function description

```
[tout,yout,teout,yeout,ieq_actout,miu_ieq, miu_eq]=sOCDE_main(f, obj,heq,
hieq,x,x0,v,v0,tstart, tfinal,opt_init,opt_sol)
%This function solves OCDE by calling OCDE_sol_init.m for initialization and
%OCDE_sol.m for integration

%Input:
%f: state equations of the dynamic part (sym)
%obj: objective function (sym)
%heq: left hand side of equality constraints (sym)
%hieq: left hand side of inequality (>=) constraints (sym)
%x:  state varaibles(sym)
%x0: initial point of x (real column vector). x0 must be given and fixed.
%v:  inner optimization variables (sym)
%v0: initial guess of inner optimization variables (real column vector). v0
can be empty.
%opt_init: options for initilization
%opt_sol:  options for integration

%Outputs:
%tout: integration steps of t (real vector)
%yout: integration results of [x;v;miu_ieq;miu_eq] (real matrix)
%teout: the time, when the switching of active set happens (real vector)
%yeout: states of [x;v;miu_ieq;miu_eq], when the switching of active set
happens (real matrix)
%ieq_actout: record the different values of active sets along the solution
trajectory (real matrix)
%miu_ieq: L. multipliers for inequality constraints
%miu_eq:  L. multipliers for   equality constraints


function [KKT,miu_ieq, miu_eq,
ieq_act,KKT_fun,event_fun,init]=OCDE_sol_init(obj,heq, hieq,x,x0,v,v0,opt)
%This function:
%(1) formulates the KKT optimality condition of the inner NLP
%in Matlab symbolic format
%(2) generates m-files of KKT optimality condition and associated function
handles
%(3) provides initial conditions for state variables, inner optimization
%variables, Lagrange multipliers, initial active set of inequality
constraints,
%(4) define event functions for detecting the switching of active set.
```

```matlab
%Inputs:
%obj: objective function (sym)
%heq: left hand side of equality constraints (sym)
%hieq: left hand side of inequality (>=) constraints (sym)
%x:  state varaibles(sym)
%x0: initial point of x (real column vector). x0 must be given and fixed.
%v:  inner optimization variables (sym)
%v0: initial guess of inner optimization variables (real column vector). v0
can be empty.
%opt: options of initilizations
%opt.MaxTry: the number of multi-start times when solving NLP for
initialization (integer), default 10.
%opt.tol_act: tolerance of active inequality (real scalar), default 1e-6.
%opt.optimoptions=: options for used optimizer. default is empty.


%Outputs:
%KKT: KKT optimality condition of inner optimization (sym)
%miu_ieq: Lagrange multipliers of inequality constraints (sym)
%miu_eq: Lagrange multipliers of equality constraints (sym)
%ieq_act: index for active inequality constraint (sym). If ieq_act(j)=1: the
j-th inequality constraint is active, otherwise non-active.
%KKT_fun([x;v;miu_ieq;miu_eq],ieq_act): function handle, which
%returns the evaluated value of KKT
%event_fun: event function for detecting the switching of active inequalities
%init: initial values of state variables, inner optimization variables,
%Lagrange multipliers and the active set


function [tout,yout,teout,yeout,ieq_actout]=OCDE_sol(ODE_fun, ...
    KKT_fun,x0, v0, miu_ieq0, miu_eq0, ieq_act0, tstart, tfinal, ...
    event_fun,Jac_fun,option)
%This function integrates the OCDE by using KKT optimality conditions. It
%solves the following quasi-DAE system
%dx=ODE_fun(x,v);
%0=KKT_fun([x;v;miu_ieq;miu_eq], ieq_act)
%x=x, z=[v;miu_ieq;miu_eq], y=[x;z]

%Inputs:
%ODE_fun: the right hand side of dx=f(x,v) (function handle)
%KKT_fun([x;v;miu_ieq;miu_eq], ieq_act): return the right hand side of 0=KKT.
%(function handle).
%x0:initial value of x (real vector)
%v0: initial value of v, i.e. the optimal solution of inner NLP (real vector)
%miu_ieq0: initial value of Lagrange multipliers miu_ieq (real vector)
%miu_eq0: initial value of Lagrange multipliers miu_ieq (real vector)
%ieq_act0: initial value of the active set index ieq_act (real vector)
%tstart: start time of integration (real scalar)
%tfinal: end time of integeration (real scalar)
%event_fun([x;v;miu_ieq;miu_eq]): function handle to detect the switching of
%active set. It is define to be miu_ieq-hieq(x,v). (function handle)
%Jac_fun(t,[x;v;miu_ieq;miu_eq]), ieq_act): Jacobian matrix df(x,z)/dxz. If
empty, it is approximated by finite difference methods.
%option: options

%Outputs:
```

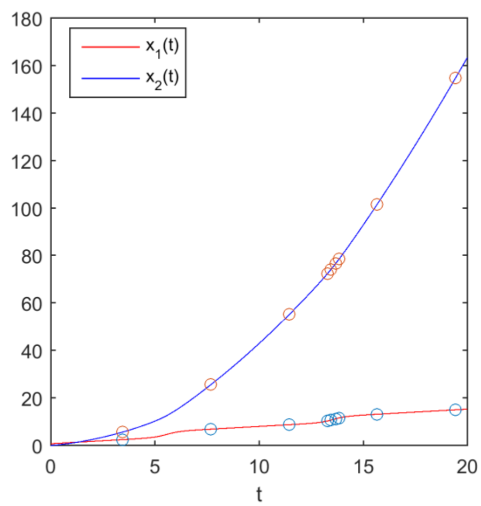
```
%tout: integration steps of t (real vector)
%yout: integration results of [x;v;miu_ieq;miu_eq] (real matrix)
%teout: the time, when the switching of active set happens (real vector)
%yeout: states of [x;v;miu_ieq;miu_eq], when the switching of active set
happens (real matrix)
%ieq_actout: record the different values of active sets along the solution
trajectory (real matrix)
```

Example (demo1.m)

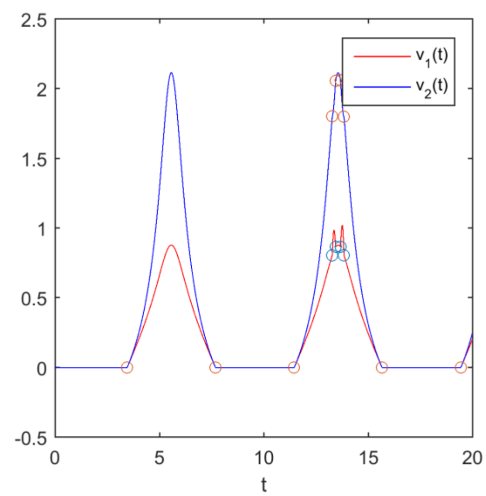
The demo example solves the following OCDE

$$\dot{x_1} = 0.5 + v_1 v_2,\ x_1(0) = {}^{\pi}/_{4}, \tag{2a}$$
$$\dot{x_2} = x_1,\ x_2(0) = 0, \tag{2b}$$
$$v = (v_1, v_2) \in arg \min_{v} {v_1}^2 + {v_2}^2 \quad ,$$
$$s.t.\ 1.7\sin(x_1) + v_2 - e^{-v_1} \geq 0, \tag{2c}$$
$$0.2\cos(x_2) + 2 - v_2 \geq 0. \tag{2d}$$

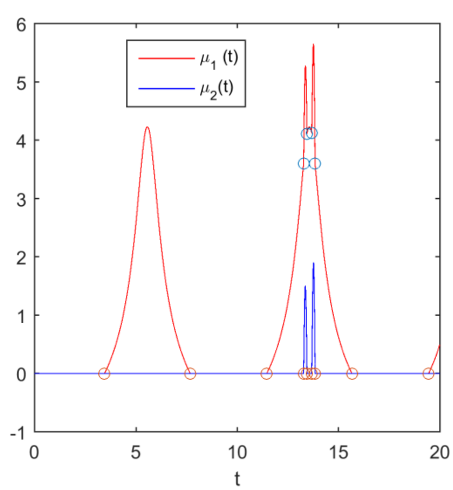
Simulation results:

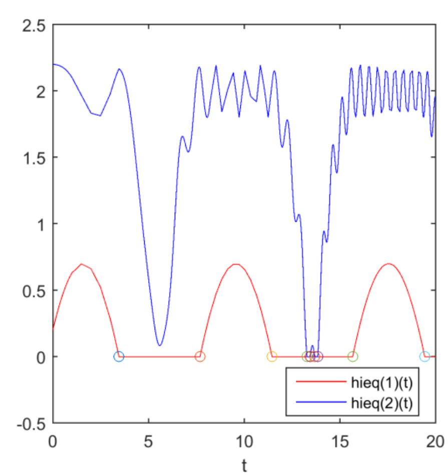


(a) State variables



(b) Inner optimization variables





(3) Lagrange multipliers *(4) inequality constraints*

Fig. 1. Simulation results of OCDE (2). Small Cycles refer to the switching time of active set.

6. **Citation**

Please cite [1], if you use the code.

7. License

Author: Xiao Zhao, Email: Xiao.Zhao@outlook.de

References

[1] Zhao, Ploch, Noack, Wiechert, Mitsos, von Lieres, Analysis of local well-posedness of optimization-constrained differential equations by local optimality conditions, AIChE J., DOI:10.1002/aic.16548.