Native Optimization Features in OpenModelica

Vitalij Ruge

January 2, 2015

Abstract

OpenModelica supports native dynamic optimization of models. This allows users to define optimal control problems (OCP) using Modelica and Optimica (only partly supported) language specifications, and solve the underlying model formulation using collocation methods.

Motivation

In Modelica it’s easy to formulate models with the causality visualized in figure 1. But in some cases it is really hard to find the right input to get the right output like in figure 2. With the optimization in OpenModelica we can find the input!

Figure 1: Causality Modelica model

Figure 2: Unknown input, but a goal for the output
Contents

1 Optimal Control Problem 3

2 Extension 3

3 OCP Formulation 3
   3.1 Mayer-term 4
      3.1.1 Optimica-Mayer-Term 4
      3.1.2 Annotation-Mayer-Term 5
   3.2 Lagrange-Term 5
      3.2.1 Optimica-Lagrange-Term 5
      3.2.2 Annotation-Lagrange-Term 6
   3.3 Box-Constraints 6
   3.4 Constraints 7
      3.4.1 Optimica-Constraints 7
      3.4.2 Annotation-Constraints 7
      3.4.3 Annotation-Final-Constraints 8

4 Special Options 9
   4.1 Compiler Options 9
   4.2 Simulation Options 9

5 Examples 9
   5.1 Initialguess 9
   5.2 Warm Start 9
   5.3 Final Constraints 9
1 Optimal Control Problem

Mathematical formalization [4]:

\[
\min_{u(t)} J(x(t), u(t), t) = E(x(t_f), u(t_f), t_f) + \int_{t_0}^{t_f} L(x(t), u(t), t) \, dt
\]  

(1)

s.t.

\[
x(t_0) = x_0
\]  

(2)

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

(3)

\[
\dot{g}(x(t), u(t), t) \leq 0
\]  

(4)

\[
r(x(t_f)) = 0
\]  

(5)

where \( x(t) = [x^{(1)}(t), \ldots, x^{(nx)}(t)]^T \) and \( u(t) = [u^{(1)}(t), \ldots, u^{(nu)}(t)]^T \) are the state vector and control variable vector for \( t \in [t_0, t_f] \), respectively. The constraints \([2, 3, 4, 5]\) represent the initial conditions, the nonlinear dynamic model description based on differential algebraic equations (DAEs), the path constraints \( \dot{g}(x(t), u(t), t) \in \mathbb{R}^{nx} \) and the terminal constraints \([2]\). The path constraints can be split in the box constraints from \( \dot{g}(x(t), u(t), t) \leq 0 \), i.e.

\[
\begin{align*}
x_{\min} & \leq x(t) \leq x_{\max} \\
u_{\min} & \leq u(t) \leq u_{\max}
\end{align*}
\]

which can handling efficiently, it is possible to use the attributes \( \min \) and \( \max \) already available in Modelica for the description (hint: use StateSelect for transform nonlinear constraints in box constraints)\([3]\). And general constraints

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

which can formulate with Optimica \([\|]\) or special user annotation.

2 Extension

Extension for the formulation of OCP in OpenModelica:

<table>
<thead>
<tr>
<th>Optimica</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mayer-Term</td>
<td>objective = costM</td>
</tr>
<tr>
<td>Lagrange-Term</td>
<td>objectiveIntegrand = costL</td>
</tr>
<tr>
<td>constraints</td>
<td>con &lt;= 0</td>
</tr>
<tr>
<td>final constraints</td>
<td>not supported in OM</td>
</tr>
<tr>
<td></td>
<td>Real costM annotation(isMayer = true);</td>
</tr>
<tr>
<td></td>
<td>Real costL annotation(isLagrange = true);</td>
</tr>
<tr>
<td></td>
<td>Real con(max=0) annotation(isConstraint = true);</td>
</tr>
<tr>
<td></td>
<td>Real fcon(max=0) (isFinalConstraint = true);</td>
</tr>
</tbody>
</table>

Open issues: alias elimination for final constraints.

Note: OMEEdit support Modelica and not Optimica. It’s possible to use OMDNotebook.

3 OCP Formulation

At the beginning we start with a predator–prey equations in Modelica, which we want optimize!

Listing 1: Modelica forest model

```model forest
```
Optimization

OpenModelica

parameter Real \( g_r = 4 \times 10^{-2} \) "Natural growth rate for rabbits";
parameter Real \( g_{fr} = 1 \times 10^{-1} \) "Efficient in growing foxes from rabbits";

parameter Real \( d_{rf} = 5 \times 10^{-3} \) "Death rate of rabbits due to foxes";
parameter Real \( d_{rh} = 5 \times 10^{-2} \) "Death rate of rabbits due to hunters";
parameter Real \( d_f = 9 \times 10^{-2} \) "Natural deathrate for foxes";
parameter Real \( d_{fh} = 9 \times 10^{-2} \) "Death rate of foxes due to hunters";

Real rabbits (start = 700) "Rabbits, (R) with start population 700";
Real foxes (start = 10) "Foxes, (F) with start population 10";

input Real hunter_rabbits;
input Real hunter_foxes;

equation
\[
\text{der (rabbits)} = g_r \times \text{rabbits} - d_{rf} \times \text{rabbits} \times \text{foxes} - d_{rh} \times \text{hunter_rabbits};
\]
\[
\text{der (foxes)} = g_{fr} \times d_{rf} \times \text{rabbits} \times \text{foxes} - d_f \times \text{foxes} - d_{fh} \times \text{hunter_foxes};
\]
end forest;

The forest-model depict a forest with foxes, rabbits and hunters. The hunters should control the population. This example we can extends with our optimization goals.

3.1 Mayer-term

For Example we wish to minimized the difference:

\[
\frac{(\text{foxes}(t_f) - 5)^2}{10} + \frac{(\text{rabbits}(t_f) - 500)^2}{100}
\]

In order to get a wished population in the forest at the end. In this case we can use the mayer term for the formulation.

3.1.1 Optimica-Mayer-Term

We can use the Optimica-Extension with two new keywords \{optimization,objective\}, which currently not part of Modelica.

Listing 2: Modelica\&Optimica forest model, mayert-term

```
optimization forestMayer (objective = goalRabbits + goalFoxes)
"goal:
closing balance rabbits = 500
closing balance foxes = 5
"extends forest;
Real goalRabbits = (rabbits - 500)^2/100 "goal for rabbits ";
Real goalFoxes = (foxes-5)^2/10 "goal for rabbits ";
```
end foresMayer;

The keyword `optimization` is alternative to `model` and signals this is not typical model for simulation.
The keyword `objective` is an attribute of the class `optimization` and contains the object function for the endpoint.

### 3.1.2 Annotation-Mayer-Term

Alternative it is possible to use `annotation` to say the optimizer, which variable express the mayer-term.

Listing 3: Modelica forest model, mayert-term

```model forestMayer
" goal:
closing balance rabbits = 500
closing balance foxes = 5
"
extends forest;
Real goalRabbits(nominal = 1e2) = (rabbits - 500)^2 "goal for rabbits" annotation(isMayer = true);
Real goalFoxes(nominal = 1e1) = (foxes-5)^2 "goal for rabbits 
annotation(isMayer = true);
end foresMayer;
```

*Note:* `nominal` used for scale the object!

In both formulations 2 and 3, OpenModelica find the inputs(`hunter_rabbits`, `hunter_foxes`).

### 3.2 Lagrange-Term

Of the other hand we want minimize the number of the hunters in each time point, because we pay for the guys. In this case we can use the lagrange-term (or include additional state, which affect the mayer-term).

\[
\begin{align*}
\text{min!} \quad & \int \text{cost}(t)dt \quad \text{or} \\
\text{min!} \quad & \text{Cost}(t_f) \quad \text{where Cost = cost}(t) \text{ and Cost}(t_0) = 0
\end{align*}
\]

*Note:* The numeric and convergence of both formulation is not the same.

#### 3.2.1 Optimica-Lagrange-Term

Listing 4: Modelica&Optimica forest model, lagrange-term

```optimization forestLagrange(objective = goalRabbits + goalFoxes,
objectiveIntegrand = 1e-2*costHuntersFoxes + 1e-2*costHuntersRabbits)
"`
goal:
  closing balance rabbits = 500
  closing balance foxes = 5
  minimize cost for hunters

extends forest;
Real goalRabbits = (rabbits - 500)^2 "goal for rabbits ";
Real goalFoxes = (foxes-5)^2 "goal for rabbits ";
Real costHuntersFoxes = hunter_foxes^2;
Real costHuntersRabbits = hunter_rabbits^2;
end forestLagrange;

The keyword \texttt{objectiveIntegrand} is an attribute of the class \texttt{optimization} and contains the object function over the time.

3.2.2 Annotation-Lagrange-Term

Listing 5: Modelica forest model, lagrange-term

model forestLagrange

 goal:
  closing balance rabbits = 500
  closing balance foxes = 5
  minimize cost for hunters

extends forest;
Real goalRabbits(nominal = 1e2) = (rabbits - 500)^2 "goal for rabbits ", annotation(isMayer = true);
Real goalFoxes(nominal = 1e1) = (foxes-5)^2 "goal for rabbits ", annotation(isMayer = true);
Real costHuntersFoxes(nominal = 1e2) = hunter_foxes^2, annotation(isLagrange = true);
Real costHuntersRabbits(nominal = 1e2) = hunter_rabbits^2, annotation(isLagrange = true);
end forestLagrange;

3.3 Box-Constraints

The bounds for the states and inputs will be handling as box constraints.

Listing 6: forest model, box constraints

... forestBoxConstarints

box constraints
  0 <= foxis <= 700
  0 <= rabbits <= 200
  0 <= hunter_foxes
  0 <= hunter_rabbits
extends forest(
    foxis(min = 0, max = 700),
    rabbits(min =0, max = 200),
    hunter_foxes(min = 0),
    hunter_rabbits(min = 0)
);
end forestBoxConstarints;

3.4 Constraints

3.4.1 Optimica-Constraints

Listing 7: Modelica & Optimica forest model, constraints

optimization forestConstarints
    
    box constraints
    0 <= foxis <= 700
    0 <= rabbits <= 200
    0 <= hunter_foxes
    0 <= hunter_rabbits

    constraints
    2 <= hunter_foxes + hunter_rabbits <= 30
    foxis <= 7 * rabbits;

end forestConstarints;

With constraint we have a new section like equation where we can formulate constraints.

Note: constraint is only support inside optimization.

3.4.2 Annotation-Constraints

Listing 8: Modelica forest model, constraints

model forestConstarints
    
    box constraints
    0 <= foxis <= 700
    0 <= rabbits <= 200

end forestConstarints;
0 <= hunter_foxes
0 <= hunter_rabbits

constraints
2 <= hunter_foxes + hunter_rabbits <= 30
foxis <= 7 * rabbits;

extends forest(
  foxis(min = 0, max = 700),
  rabbits(min =0, max = 200),
  hunter_foxes(min = 0),
  hunter_rabbits(min = 0)
);
Real con1(min=2, max = 30) annotation(isConstraint = true);
Real con2(max = 0) = foxis - 7 * rabbits annotation(
  isConstraint = true);
equation
  con1 = hunter_foxes + hunter_rabbits;
end forestConstarints;

3.4.3 Annotation-Final-Constraints

Listing 9: Modelica forest model, final constraints

model forestFinalConstarints
  
  box constraints
  0 <= foxis <= 700
  0 <= rabbits <= 200
  0 <= hunter_foxes
  0 <= hunter_rabbits

constraints
2 <= hunter_foxes + hunter_rabbits <= 30
foxis <= 7 * rabbits;
final constraint
  hunter_foxes == 0;
  hunter_rabbits == 0;

extends forest(
  foxis(min = 0, max = 700),
  rabbits(min =0, max = 200),
  hunter_foxes(min = 0),
  hunter_rabbits(min = 0)
);
parameter Real noAlias = 1.0;
Real con1(min=2, max = 30) annotation(isConstraint = true);
Real con2(max = 0) = foxis - 7 * rabbits annotation(
  isConstraint = true);
Real fcon1(min=0,max=0) = noAlias*hunter_foxes annotation(
  isFinalConstraint = true);
Real fcon2(min=0, max=0) = noAlias*hunter_rabbits annotation(
  isFinalConstraint = true);
equation
  con1 = hunter_foxes + hunter_rabbits;
end forestFinalConstarints;

4 Special Options

4.1 Compiler Options

<table>
<thead>
<tr>
<th>numberOfIntervals</th>
<th>e.g. 50</th>
<th>collocation intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime, stopTime</td>
<td>time horizon</td>
<td></td>
</tr>
<tr>
<td>tolerance</td>
<td>e.g. 1e-8</td>
<td>solver/optimizer tolerance</td>
</tr>
<tr>
<td>simflags</td>
<td>(see 4.2)</td>
<td>simulation flags</td>
</tr>
</tbody>
</table>

4.2 Simulation Options

<table>
<thead>
<tr>
<th>-lv</th>
<th>LOG_IPOPT, LOG_IPOPT_ERROR</th>
<th>collocation intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ipopt_hesse</td>
<td>CONST, BFGS, NUM</td>
<td>hessian approximation</td>
</tr>
<tr>
<td>-ipopt_max_iter</td>
<td>e.g. 100</td>
<td>maximal number of iteration for ipopt</td>
</tr>
<tr>
<td>-exInputFile</td>
<td>externalInput.csv</td>
<td>input guess</td>
</tr>
<tr>
<td>-optimizerNP</td>
<td>1 or 3</td>
<td>number of collocation points</td>
</tr>
<tr>
<td>-ipopt_warm_start</td>
<td>e.g. 8</td>
<td>scale for initial guess (prototype)</td>
</tr>
</tbody>
</table>

5 Examples

Some example can be found in [https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/](https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/).

5.1 OMNotebook and Shell


Initial guess

[https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/BRinitialGuess.mos](https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/BRinitialGuess.mos)

Warm Start

[https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/DMwarm.mos](https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/DMwarm.mos)

Final Constraints

[https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/TFC.mos](https://trac.openmodelica.org/OpenModelica/browser/trunk/testsuite/openmodelica/cruntime/optimization/basic/TFC.mos)
References


