Numerical Optimal Control, August 2014

Exercise 8: Solving OCPs with a direct collocation method

Joel Andersson Joris Gillis Greg Horn Rien Quirynen Moritz Diehl

University of Freiburg – IMTEK, August 5th, 2014

In this exercise, we will solve an OCP (quadcopter flight) using a direct collocation method.

Tasks:

8.1 From the course website, obtain the quadcopter model file. Save it as `quadcopter.py` in your working directory, and after importing the file, instantiate the model with

```
model = Quadcopter()
```

The quadcopter model is described by an ode:

$$\dot{x} = f(x,u) \quad x, \dot{x} \in \mathbb{R}^{17}, u \in \mathbb{R}^4 \tag{1}$$

Use `Quadcopter?` or `help(Quadcopter)` to see a description of the statespace.

8.2 Consider the following OCP in which the aim is for the quadcopter to follow a trajectory starting from standstill:

$$\begin{aligned} \underset{x(t),u(t)}{\text{minimize}} \quad & \int_0^T \left[||p(t) - p^{\text{ref}}(t)||_2^2 + \alpha ||u(t)||_2^2 \right] dt \\ \text{subject to} \quad & \dot{x} = f(x,u) \\ & 0 \leq u \leq 0.5 \\ & p(0) = [0,0,0]^T \\ & v(0) = [0,0,0]^T \end{aligned} \tag{2}$$

In a direct collocation scheme, on each control interval, the state trajectories are approximated by polynomials. Each polynomial is parametrised with points that interpolate this polynomial. These points have the same dimensions as the state-space and are extra decision variables. In this exercise, we will using 1-degree polynomials, needing one extra decision variable y_k for each control interval, right in the middle.

The decision variable vector should look like:

$$w = [x_0, y_0, u_0, x_1, y_1, u_1, \ldots, x_{N-1}, y_{N-1}, u_{N-1}, x_N]^T \tag{3}$$

Complete the two missing parts in the following direct collocation transcription:

$$\begin{aligned}
\underset{x_\bullet, y_\bullet, u_\bullet}{\text{minimize}} \quad & \frac{T}{N}\left[\sum_{k=0}^{N} ||p_k - p_k^{\text{ref}}||_2^2 + \alpha \sum_{k=0}^{N-1} ||u_k||_2^2\right] \\
\text{subject to} \quad & x_{k+1} = ? \qquad \forall k = 0, 1, \ldots N-1 \quad \text{coupling constraints} \\
& ? = f(y_k, u_k) \qquad \forall k = 0, 1, \ldots N-1 \quad \text{collocation constraints} \\
& 0 \leq u_k \leq 0.5 \qquad \forall k = 0, 1, \ldots N-1 \\
& p_0 = [0, 0, 0]^T \\
& v_0 = [0, 0, 0]^T
\end{aligned} \tag{4}$$

.

Note that, in Moritz's slides, the collocation constraints instead read like $g(x_k, y_k, u_k) = 0$. For midpoint collocation, we can use either notation.

8.3 Formulate the above NLP in CasADi and solve it with IPOPT using initial guess $x_k = y_k =$ `model.x0` and $u_k =$ `model.u0`. Use a total time of $T = 3.0s$, 200 control intervals, and $\alpha = 0.05$.

Remember that formulating an NLP in CasADi starts with defining a single symbolic `MX` variable. Next, you construct expression using slices of this single variable:

```
W = MX.sym("W",(N+1)*17+N*17+N*4)
g = []
for k in range()
  g.append( stuff( W[(17+17+4)*k+17:(17+17+4)*k+17+17],...) ) # use y_k,...
g = vertcat(g)

nlp = MXFunction(nlpIn(x=W),nlpOut(f=f,g=g))
```

If all this indicing makes your head hurt, you are not alone. You should use the template solution found on the web page. It uses a nicer technique, where `W` is similar to an `MX`, but with convenient accessors.

8.4 **Extra:** The quadcopter model is in fact given by a fully implicit ode:

$$r(\dot{x}, x, u) = 0. \quad x, \dot{x} \in \mathbb{R}^{17}, u \in \mathbb{R}^4 \tag{5}$$

By using an inversion of $\frac{\partial r}{\partial \dot{x}}$, an explicit ode can be constructed that we used in the above exercise. Can you think of an alternative formulation that uses the implicit form $r(\dot{x}, x, u)$ directly?

8.5 **Extra:** Construct a time-optimal OCP with the same system. Again, a template is available from the website.