Hands-on CasADi course on optimal control, Yacoda

Exercise: Optimal Control

In this exercise, we'll be working with a toy discrete optimal control problem. The discrete system consist of a scalar state x, a scalar control u, and a discrete state propagation function $F : (x, u) \mapsto x^2 + u$

Tasks:

1. Using `opti`, implement and solve (with `Ipopt`) the following multiple shooting transcription for $N = 4$:

$$\begin{aligned}
\underset{x_1,x_2,\dots x_{N+1},u_1,u_2,\dots,u_N}{\text{minimize}} \quad & \sum_{k=1}^{N} u_k^2 + \sum_{k=1}^{N+1} x_k^2 \\
\text{subject to} \quad & F(x_k, u_k) = x_{k+1}, \quad k = 1 \dots N \\
& x_1 = 2 \\
& x_{N+1} = 3 \\
& x_k \geq 0, \quad k = 1 \dots N+1
\end{aligned} \tag{1}$$

   Start from:

```
X = opti.variable(N+1);
U = opti.variable(N);
```

   Don't provide an initial guess here. The default guess (zero) is fine.

   Verify that the optimal control sequence is given by `[-2.7038;-0.5430;0.2613;0.5840]`.

2. Adapt your implementation to a single shooting transcription: the only decision variables now are $u_1, u_2, \dots, u_N$, the states should be eliminated from the problem.

   Verify that you get the same numerical answer as the previous implementation.

3. Plot the sparsities of the objective's Hessian and constraint Jacobian for both the multiple shooting (MS) and single shooting approaches (SS).

   Give an intuitive explanation for the patterns you see.

4. (extra) Rework the multiple shooting implementation to yield a banded constraint Jacobian. Does this change the NLP iterations?

5. Taking into account the facts below, which transcription scales best for large N, SS or MS?

   - The Hessian will be part of the KKT matrix, which will be factorized/inverted by the NLP solver.

- Inverting a dense system of shape N-by-N costs $O(N^3)$.

6. Compare the numerical values of the constraint Jacobian for MS and SS, evaluated at the initial guess. You may do this by passing a second argument to `sol.value`: `opti.initial`.

   For SS, can you explain intuitively the large numbers? What would happen with the magnitudes when N is further increased?

7. (extra) Assuming all constraints are active at the initial guess, construct the KKT matrix for both SS and MS at the initial guess, and compare condition numbers. You'll find `2.805e+20` and `16.5` respectively. Hint: first take some care in creating constraints to avoid obvious LICQ violations.

   Can you explain using the answer of the previous question?

8. Compare the number of iterations between SS and MS. The difference is fairly typical.

   Our system happens to have a polynomial dynamics. What order of polynomial is $F(x, 0)$? What order of polynomial is $F(F(x, 0), 0)$?

   Bearing in mind that each iteration involves a linearization, can you explain the iteration difference intuitively?

9. (extra) Suppose you have a reasonable initial guess for the state trajectory available. How could you exploit this in MS and SS?

10. Perform a multiple-shooting with a system $x \in \mathbb{R}^3, u \in \mathbb{R}^2$, $N = 4$, and

$$F(x, u) = \begin{bmatrix} 1 & 0.1 & 0.2 \\ 2 & 0.3 & 0.4 \\ 6 & 1 & 3 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 2 & 1 \end{bmatrix} u$$

$$\begin{aligned} \underset{x_1, x_2, \ldots x_{N+1}, u_1, u_2, \ldots, u_N}{\text{minimize}} \quad & \sum_{k=1}^{N} u_k^2 + \sum_{k=1}^{N+1} x_k^2 \\ \text{subject to} \quad & F(x_k, u_k) = x_{k+1}, \quad k = 1 \ldots N \\ & x_1 = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}^T \\ & x_{N+1} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \end{aligned} \tag{2}$$

    Verify that you obtain $u_1 =$ `[-4.2894;-4.1930]`. Explain the amount of iterations it takes. Without implementing it, what number of iterations would SS take?

11. (extra) Suppose you need to perform optimal control on a system with 100 states and 1 control, stable dynamics with $\frac{\partial F}{\partial x}$ dense, no path constraints and $N = 10000$. Would you pick MS or SS? You can get some inspiration from the sparsity pattern of the previous question, but don't attempt to increase N and the state-space that big!