TEMPO Spring School: Theory and Numerics for Nonlinear Model Predictive Control

Exercise 4: Model-predictive control

J. Andersson M. Diehl J. Gillis J. Rawlings M. Zanon

University of Freiburg, March 27, 2015

NMPC with direct multiple shooting and Gauss-Newton SQP

In nonlinear model predictive control (NMPC), we repeatedly solve an optimal control problem (OCP) with changing data in order to derive an optimal feedback strategy for a controller. Both direct collocation methods and direct multiple shooting methods are good candidates for transcribing the OCP since in both cases we are able to efficiently exploit the similarity between consecutive problems by initializing the NLP solver. To solve the NLP, we can use either sequential quadratic programming (SQP) or interior-point methods (IP).

Since solving an NLP is an expensive operation, there is often a tradeoff between finding a better solution to the NLP or returning feedback to the system more frequently. In the most extreme case, we just do one iteration of the NLP solver for every feedback time. In the case of an SQP solver, this means solving a single QP.

We continue to work with the simple OCP from Exercise 2:

$$
\begin{aligned}
\underset{x,u}{\text{minimize}} \quad & \int_0^T x_0(t)^2 + x_1(t)^2 + u(t)^2 \, dt \\
\text{subject to} \quad & \dot{x}_0(t) = (1 - x_1(t)^2)\, x_0(t) - x_1(t) + u(t), && x_0(0) = 0, \\
& \dot{x}_1(t) = x_0(t), && x_1(0) = 1, \\
& -1 \le u(t) \le 1, \quad \text{for } t \in [0, T],
\end{aligned}
\tag{1}
$$

where $T = 10$ as earlier and its multiple shooting discretization:

$$
\begin{aligned}
\underset{w}{\text{minimize}} \quad & J(w) := \sum_{k=0}^{N} \|X_k\|_2^2 + \sum_{k=0}^{N-1} U_k^2 \\
\text{subject to} \quad & X_0 = [0, 1]^\intercal, \\
& X_{k+1} = F(X_k, U_k, h), \quad -1 \le U_k \le 1, \qquad k = 0, \dots, N-1,
\end{aligned}
\tag{2}
$$

with $N = 20$.

Tasks:

4.1 As in Exercise 3, identify the least squares objective $R(w)$ such that $J(w) = \frac{1}{2} \|R(w)\|_2^2$ (on paper, no coding needed).

4.2 Implement a CasADi `MXFunction` G that calculates the equality constraint function $g(x)$ and another `MXFunction` Jg as in Task 3.2 and 3.3. Evaluate them numerically and confirm that you get the expected result.

4.3 Implement a Gauss-Newton method to solve the problem as in Exercise 3. Do 20 iterations.

4.4 Go to the course website and download the files `nmpc_ipopt.py`. This is an implementation of an NMPC controller using direct multiple shooting and IPOPT as an NLP solver. The file depends on `plotter.py` to make nice plots, so downlaod this as well. You should recognize the code from Exercise 2. Instead of working with indices to access different parts of the NLP variable vector, this code uses a concept in Python called *structures*. You can read more about this feature in the user guide. Go through the script and make sure that you understand the code. Run the script.

4.5 **Extra**: Replace the OCP solver with your own Gauss-Newton SQP method. Just take a single iteration of the SQP method. Compare the result with Task 4.4 above

4.6 **Extra**: When just solving a single QP per NMPC iteration, it often make sense to divide the solution code into a *preparation phase* and a *feedback phase*. The preparation phase contains the part of the algorithm that can be calculated before we obtain measurements for the state of the system (i.e. the initial conditions of the ODE). This allows the controller to return feedback to the system faster. What part of the algorithm can be made part of preparation phase?

4.7 **Extra**: Modify the solution to take more than one SQP iteration per NMPC iteration. Does it improve the controller?