SSY280 Model Predictive Control

Assignment 3
Using code generation tools for MPC

The assignment is voluntary

Division of Systems and Control
Department of Signals and Systems
Chalmers University of Technology

February 2017

1 Objectives

The objective of this assignment is to introduce some of the tools that are now available to deploy MPC algorithms in embedded control systems for applications with high demands on real-time performance. The assignment is based on Assignment 2 and should be fairly straightforward to complete if you have access to a laptop with Matlab together with a supported compiler.

2 Introduction

In Assignment 2, you have coded an MPC algorithm including both state estimation and target calculations. For the QP solver, we have so far resorted to Matlab's inbuilt solver quadprog. This is, however, typically not an option when an embedded implementation of an MPC algorithm is desired. Moreover, a tailored QP solver can be expected to perform at least an order of magnitude more efficiently than quadprog.

There are several ways to approach the problem, if you want to avoid coding the optimization algorithm at the lowest level yourself. One is to go for a standard solver, available in source code form, and then to integrate this with your own MPC code. Another possibility is to use a code generation tool to have your solver being tailored to your specific problem. Several services that have emerged over the last few years have made the latter alternative a viable option; among these are the following:

- FORCES Pro (https://www.embotech.com/FORCES-Pro)
- ECOS (https://www.embotech.com/ECOS)
- CVXGEN (http://cvxgen.com/docs/index.html)
- ACADO (http://acado.github.io/index.html)
- FiOrdOs (http://fiordos.ethz.ch/dokuwiki/doku.php?id=start)

Several of these services fit the MPC needs well. The basic idea is that you describe your optimization problem in a way that resembles what you have already done in Assignment 2 using Matlab scripts. In return, you get a piece of code that can either be built into your embedded controller, or – as in our case – be called from within Matlab or Simulink.

3 Preparations

We have chosen to use the services offered by Embotech (a spinoff from ETH in Zürich) via the product FORCES Pro. They provide both a 30-day trial (to be used here) and academic licenses for free. The code generation is initiated by a call to their server from within Matlab, and a solver that is tailored to your problem is automatically downloaded to your computer and can then be called in your MPC code instead of quadprog.

The following steps need to be carried out to prepare your Matlab environment for the assignment:

1. Check Mathworks' homepage to make sure you have a supported compiler for MEX-file compilation, compatible with your Matlab version, installed on your computer. If not, download and install.

2. Download the free trial version of the Forces Pro Matlab client at https://www.embotech.com/FORCES-Pro/Download. In this step, you need to register with your Chalmers email address; we have communicated with Embotech, who supports this activity, and there should be no problem to register.

3. Install using instructions at https://www.embotech.com/FORCES-Pro/Installation

4. You also need to download YALMIP, see https://www.embotech.com/FORCES-Pro/User-Manual/Y2F-Yalmip-interface. Then setup YALMIP according to the instructions in the readme.txt file.

4 Tasks

A simple MPC example

With the preparations above being done, you should be ready to start trying out the features offered by Forces Pro. It is suggested that you start out with the simple MPC example found at
https://www.embotech.com/Examples/Yalmip-interface-Y2F/Basic-example.
The example is for a double integrator plant, an example that is often used to illustrate MPC features in the literature.

(a) Download and open the example Matlab script to inspect the Matlab code. As can be seen, these are the main steps that the Matlab code includes:

1. Definition of the data (i.e. constants and matrices) that are used in setting up the MPC formulation; this includes e.g. prediction horizon, plant model matrices, cost function matrices, and control and state constraint matrices. It can be noticed that the example uses the infinite LQ cost as terminal cost, and that lower and upper bounds for both control and states are used.
2. Definition of the MPC setup. This can be done in different ways; in the example, a format called Yalmip is used. The syntax should be clear from the example, at least for simple cases like this. Notice the indexing: X(:,1) is the initial state and U(:,1) is the control vector to be applied to the process.
3. A call to Embotech's server, which will result in the generation of a tailor-made solver, and the subsequent downloading of files to be used.
4. A simulation with calls to the downloaded solver.
5. Plotting results.

Now run the Matlab script to see that everything runs as expected.

Chemical reactor example

The main task in this Assignment is to use Forces Pro to include a code-generated solver to replace the call to quadprog in your Matlab script from Assignment 2.

(b) By following the steps in the simple example, modify your script from Assignment 2 by including the MPC definition in Yalmip code and the call to generate the solver, and then replace the call to quadprog with the call to your own solver. Run the modified script to see that it works properly. Inspect the execution times of the solver to verify its real-time capabilities for the two cases $N = 10$, $M = 3$ (as in Assignment 2) and $M = N = 10$, respectively.

(c) Now we would like to include control constraints into the MPC algorithm. To solve this, you need to observe the following: the code you have generated in (b) is for a specific setup, including fixed parameters for cost functions etc. To be able to experiment with different parameters to tune the MPC performance, without having to re-generate the solver, it is possible to define *parameters*, in addition to the initial state, when generating the solver; see https://www.embotech.com/Examples/Yalmip-interface-Y2F/Parametric-cost. This is also useful when implementing

the constraints. If you want to have bounds on the absolute control vector, for example, then the bounds to be respected by the solver will be a function of the steady state target: $u_{min} - u_s \leq \delta u \leq u_{max} - u_s$. Since the steady state target is recomputed at every sampling instant, this means that the bounds vary with time; this can be handled by treating the bounds as parameters. Now include some suitable control constraints to see how the solver performance is affected!

More features of Forces Pro

There are several additional features of Forces Pro that can be investigated. Here are a few suggestions:

(d) Among the files that are generated and downloaded by Forces Pro is a Simulink block, which can be found in the Interface subfolder. Hence, you can build a Simulink model to simulate your MPC controller, including scopes to view different signals in the loop.

(e) There are many ways you can configure the detailed functions of the solver, see https://www.embotech.com/FORCES-Pro/User-Manual/Low-level-Interface/Solver-Options.

Report

The assignment is voluntary and no report needs to be filed.