Model Predictive Control (H0E76A)

Exercise Session 6 | First-order optimization algorithms for MPC

leander.hemelhof@kuleuven.be
renzi.wang@kuleuven.be
brecht.evens@kuleuven.be

2024–2025

1 Problem description

In this session we will first-order optimization methods. These methods are well-studied and particularly suitable for real-time MPC purposes as they have relatively low memory footprints (they don't require solutions to linear systems or building/storing matrices) and are very easy to warm start.

1.1 Dynamics

In previous exercises, we already encountered the following linear model for a car driving in a straight line. Recall the following dynamics equation:

$$x_{k+1} = \underbrace{\begin{bmatrix} 1 & T_\mathrm{s} \\ 0 & 1 \end{bmatrix}}_{A} x_k + \underbrace{\begin{bmatrix} 0 \\ T_\mathrm{s} \end{bmatrix}}_{B} u_k$$

where we set $T_\mathrm{s} = 0.3$ s. As before, the state vector $x = [p\ v]^\top$ represents the position and the velocity of the vehicle and the input u is the acceleration.

1.2 Optimization problem

Consider the resulting optimal control problem:

$$\begin{aligned}
&\underset{x,u}{\text{minimize}}\ f(x,u), && && \text{(1a)}\\
&\text{s. t. } x_0 = \bar{x}_0, && && \text{(1b)}\\
&\quad x_{k+1} = Ax_k + Bu_k, && k = 0,\dots,N-1, && \text{(1c)}\\
&\quad u_k \in U, && k = 1,\dots,N-1. && \text{(1d)}
\end{aligned}$$

The objective function is our usual quadratic cost in terms of states and inputs, $f(x,u) = \frac{1}{2}x_N^\top P x_N + \sum_{k=0}^{N-1} \frac{1}{2}x_k^\top Q x_k + \frac{1}{2}u_k^\top R u_k$. Due to the input constraints (1d) (which are very common in practice for obvious reasons) this problem is not analytically solvable, in contrast to an equality-constrained QP (i.e., LQR). Hence, our need for numerical solvers.

1.3 Gradient projection

Let us start with the simplest first-order method to deal with the above problem: the gradient projection method. This method combines gradient descent steps on the objective function with projections onto the feasible set (here $U^N = \underbrace{U \times U \times \cdots \times U}_{N\text{times}}$). Before we can employ gradient projection though, we need to eliminate the system dynamics (1c) from the problem.

Exercise 1 | Explain why this is necessary for this particular algorithm.

We solve this problem by substituting all the states recursively as functions of the inputs (and with the initial state $\bar{x}_0$ as a parameter). The remaining decision variables are the control inputs, and this formulation is called *single shooting*.

2 Exercises

Exercise 2 | Convert the problem to a single shooting formulation, i.e. eliminate the state variables as a function of the inputs and the initial state as a parameter. The result should be an objective function as a function of the inputs and initial state only, i.e., $f(u, \bar{x}_0) = \frac{1}{2}u^\top Hu + g^\top u$. Use $Q = \text{diag}(10, 1)$, $R = 1$, $P = 5Q$, $N = 30$ and $\bar{x}_0 = [\,-100\ 0\,]^\top$.

Exercise 3 | Work out the gradient of the cost function and implement the gradient projection algorithm with a fixed step size, for [1].

- $U = [u_{\min}, u_{\max}] = [-10, 10]$;
- stepsize of $\gamma = \frac{1}{L_f}$ (with L_f the Lipschitz constant of ∇f);
- `max_it` = 200;
- Initial guess $u = 0$.

Plot the convergence rate, $\frac{f(u^k)-f(u^*)}{f(u^*)}$ as a function of iteration number k (use `semilogy`). **Hint.** You can use the final iteration u_T as an approximation for u^*. However, because of this approximation, it's best to plot only the errors of the first $T/2$ iterations, since the error in the last few iterations will most likely be in the order of magnitude of the approximation error $\|u^* - u_T\|$

Also take a look at the trajectory (using u^* perform the simulation loop) the car will take. What do you notice?

Exercise 4 | Accelerate the convergence using the Nesterov acceleration technique. Use $\alpha = \frac{\sqrt{\kappa_f}-1}{\sqrt{\kappa_f}+1}$, and compare the convergence with the regular projected gradient method. What are your observations?

Exercise 5 | Implement a line search on the step size γ. Start with $\gamma = \frac{100}{L_f}$. Does the gradient projection step with linesearch perform better or worse than without linesearch? Why is that?

Exercise 6 | Let's now add state constraints to the problem. Say that at every time step, the position should not exceed the origin, i.e.

$$\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}p\\v\end{bmatrix}_k \leq 0, \quad k = 0, ..., N. \tag{2}$$

Implement AMA in the sequential approach to solve this state constrained OCP. Use maxiter = 2000. What is the violation on the constraints?

[1] See `Problem` in `given.config`