1 Nonlinear Model Predictive Control

1.1 Introduction

In the first section we looked at the following simple system.

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= u \\
y &= x_1
\end{align*}
\]

The goal is to keep the output at a given setpoint \( y_{sp} \) using the control action \( u(\cdot) \), which is bounded \(-M \leq u \leq M\). Figure 1 shows the result of designing a control using LQR plus saturation at \( M \), i.e.

\[ u = \text{sat}(-Kx, M) \]

We observe that as the value of \( M \) decreases this simple strategy fails to keep the system stable. The reason is that although the controller does know about the system dynamics, the controller design ignores the constraint, being over optimistic about its ability to slow down the system once the speed is high.

Intuitively, it is clear that better results would be achieved if we made the controller see not only the dynamics but also the barrier. Model Predictive Control (MPC) offers that framework. Indeed, Figure 2 shows the result of applying MPC to this problem. We observe that stability is not lost as \( M \) decreases. Model Predictive Control has largely conquered industrial applications by means of being both systematic and intuitive.

1.2 Main Theoretical Elements

Model predictive control is nowadays probably the most popular method for handling disturbances and forecast changes. The main ingredients of the method are

1. Plant model

\[ \dot{x} = f(t, x, u), \quad x(0) = x_0 \]
Figure 1: LQR plus saturation for double integrator
1. Constraints

\[ g(t, x, u) = 0 \]
\[ g_i(t, x, u) \leq 0 \]

1. Objective function

\[ J[x_0, u(\cdot)] = \int_t^{t+T} f_0(t, x(\tau), u(\tau)) d\tau + F(x(t + T), u(t + T)) \]

Resembling a chess game as played by a computer algorithm, the method works in iterations comprising the following steps.

1. Evaluate position (=measurement) and estimate system state

2. Calculate the effect on the plant of (possible) sequences of actuator moves

3. Select the best control sequence (mathematical algorithm, optimization)

4. Implement the first move (new actuator set-point)
5. Restart after the opponents move (plant/process reaction to actuator move)

Remark 1

1. The model is used to predict the system behavior into the future.

2. The method requires solution of optimization problem at every sampling time.

3. Additional constraints on the actuators can be added. For instance, that the actuators are to remain constant during the last “N” steps.

4. Normally linear or quadratic cost functions are used. These functions represent trade off among deviations from setpoints, actuator action costs, economical considerations, etc.

5. For nonlinear systems, one has to deal with risks like loss of convexity, local minima, increased computational time, etc. Still, the optimization problems to be solved online are highly sparse. This allows for efficiency gains of several orders of magnitudes.

6. Useful functionality is added when mathematical models in the “Mixed Logical Dynamical Systems” framework are used. In this case, logical constraints and states can be included in the mathematical model,
fact that is extremely useful when dealing with plant wide planning and scheduling problems.

1.3 Example: Tank Level Reference Tracking

Problem Statement
The process consists of a tank with a measurable inflow disturbance and a controllable outflow in form of a pump. Constraints are set on the tank level and the pump capacity. A level reference target is given.

Model Variables
1. Inputs of the model are the known but not controllable tank inflow and the controlled valve opening variation.
2. States are the tank level and the current valve opening:
3. Outputs are the tank level, and the outflow
Model Equations

State dynamics

\[ \text{vol}(t + 1) = \text{vol}(t) + f_{\text{in}}(t) - f_{\text{out}}(t) \]

\[ f_{\text{out}}(t) = \alpha \text{level}(t) u(t) \]

\[ \text{level}(t) = \frac{\text{volume}(t)}{\text{area}} \]

\[ u(t + 1) = u(t) + du(t) \]

Outputs

\[ y_{\text{level}}(t) = \text{level}(t) \]

\[ y_{\text{out}}(t) = f_{\text{out}}(t) \]

Inputs

- \( f_{\text{in}} \): inflow, measured but not controlled
- \( u \): valve opening, controlled via its derivative \( du \).

Cost Function

\[ J[x_0, u(\cdot)] = \sum_{t=0}^{T} q \left[ y_{\text{level}}(t) - y_{\text{ref}}(t) \right]^2 + r du^2 \]

Problem Constraints

1. Level within min/max
2. Outflow within min/max
3. Max valve variation per step

Model Representation

Industry has created dedicated software tools aimed to facilitate the tasks of modelling and control design. As a rule, the engineer is provided tools for graphical representation of the model. Figure 5 shows the tank model equations depicted in a typical industrial software.

There are also standard displays for fast visualization of the plant dynamics. In Figure 6 we see such standard representation. The graphic is a grid of graphs, with as many rows as outputs and as many columns as inputs. Each column represents the response of each output to a step in a given input.
Figure 5: Tank Model in Graphical Package

**Results Discussion**

Figure ?? shows optimization results obtained with a receding horizon of $T = 20$ steps. Note how the Model Predictive Controller is able to nicely bring the level to the desired setpoint.

On the other hand, Figure 8 shows the same optimization obtained with a receding horizon of $T = 2$ steps. In this case, the performance considerably deteriorates: stability is lost!

In the last example we look at the problem of controlling 3 interconnected tanks. The problem to solve is the same as before, namely, to keep the tank levels at given setpoints. The problem is now not just larger, but also more complicated due to the interaction among the tanks. We design 2 controllers, one that knows about the true plant dynamics, and one that treats the tanks as if they were independent.

Figure 10 shows the step response of this model. Note how the effect of each actuators propagates in the system from one tank to the next.

In Figure 11 we represent the closed loop responses of 2 different controllers:

- one that knows about the interconnection among the tanks and can better predict the system behavior and
Figure 6: Tank Model Step Response
Figure 7: Tank Trajectories for T=20
Figure 8: Tank Example Trajectories for T=2
Figure 9: Three Interconnected Tanks
Figure 10: Step Response in the Interconnected Tanks Case
one that treats each tanks as independent entities, where the valve is used to control the level.

As expected we observe that the multivariable controller is able to solve the problem in a much more efficient fashion than the "single input single output" one, showing the benefits of the model predictive control approach over the idea of cascaded SISO controllers.

1.4 Chronology of Model Predictive Control

Below we find some major milestones in the journey leading to the current MPC approach:

1. 1970’s: Step response models, quadratic cost function, ad hoc treatment of constraints
2. 1980’s: linear state space models, quadratic cost function, linear constraints on inputs and output
3. 1990’s: constraint handling: hard, soft, ranked
4. 2000’s: full blown nonlinear MPC

1.5 Stability of Model Predictive Controllers

When obtaining stability results for MPC based controllers, one or several of the following assumptions are made

1. Terminal equality constraints
2. Terminal cost function
3. Terminal constraint set
4. Dual mode control (infinite horizon): begin with NMPC with a terminal constraint set, switch then to a stabilizing linear controller when the region of attraction of the linear controller is reached.

In all these cases, the idea of the proofs is to convert the problem cost function into a Lyapunov function for the closed loop system.
Figure 11: MPC and SISO Responses in the Interconnected Tanks Case
Let us consider at least one case in details. For that, we introduce the MPC problem for discrete time systems. Note that in practice, this is the form that is actually used.

Consider the time invariant system

1. Plant model
   \[ x(k + 1) = f(x(k), u(k)), \quad x(0) = x_0 \]

1. Constraints
   \[ g(k, x, u) = 0, \quad k = 0 : \infty \]
   \[ g_i(k, x, u) \leq 0, \quad k = 0 : \infty \]

1. Objective function
   \[ J[x_0, u(\cdot)] = \sum_{l=k}^{k+N} L(l, x(l), u(l)) \]

The optimal control is a function

\[ u^*(\cdot) = \arg \min J[x_0, u(\cdot)], \quad u^*(l), \quad l = k : k + N \]

**Theorem 1** Consider an MPC algorithm for the discrete time plant, where \( x = 0 \) is an equilibrium point for \( u = 0 \), i.e. \( f(0, 0) = 0 \).
Let us assume that

- The problem contains a terminal constraint \( x(k + N) = 0 \)
- The function \( L \) in the cost function is positive definite in both arguments.

Then, if the optimization problem is feasible at time \( k \), then the coordinate origin is a stable equilibrium point.

**Proof.** We use the Lyapunov result on stability of discrete time systems introduced in the Lyapunov stability lecture. Indeed, consider the function

\[ V(x) = J^*(x), \]

where \( J^* \) denotes the performance index evaluated at the optimal trajectory. We note that:
Figure 12: Double Integrator loses stability lost for short horizon

- $V(0) = 0$
- $V(x)$ is positive definite.
- $V(x(k+1)) - V(x(k)) < 0$. The later is seen by noting the following argument. Let $u^*_k(l), \ l = k : k + N$ be the optimal control sequence at time $k$. Then, at time $k + 1$, it is clear that the control sequence $u(l), \ l = k + 1 : N + 1$, given by
  
  $u(l) = u^*_k(l), \ l = k + 1 : N$
  $u(N + 1) = 0$

  generates a feasible albeit suboptimal trajectory for the plant. Then, we observe
  
  $V(x(k+1)) - V(x(k)) < J(x(k+1), u(\cdot)) - V(x(k)) = -L(x(k), u^*(k)) < 0$

  which proves the theorem.

**Remark 2**  Stability can be lost when receding horizon is too short, see Figure 12.

**Remark 3**  Stability can also be lost when the full state is not available for control and an observer must be used. More on that topic in the Observers Lecture.