16.323 Lecture 15

Model Predictive Control

Planning in Lecture 7 was effectively “open-loop”

- Designed the control input sequence \( u(t) \) using an assumed model and set of constraints.

- Issue is that with modeling error and/or disturbances, these inputs will not necessarily generate the desired system response.

Need a “closed-loop” strategy to compensate for these errors.

- Approach called Model Predictive Control

- Also known as receding horizon control

Basic strategy:

- At time \( k \), use knowledge of the system model to design an input sequence
  \[
  u(k|k+1), u(k|k+2), u(k|k+3), \ldots, u(k|k+N)
  \]
  over a finite horizon \( N \) from the current state \( x(k) \)

- Implement a fraction of that input sequence, usually just first step.

- Repeat for time \( k + 1 \) at state \( x(k + 1) \)
• Note that the control algorithm is based on numerically solving an optimization problem at each step
  – Typically a constrained optimization

• Main advantage of MPC:
  – Explicitly accounts for system constraints.
    ◇ Doesn’t just design a controller to keep the system away from them.
  – Can easily handle nonlinear and time-varying plant dynamics, since the controller is being explicitly a function of the model that can be modified in real-time (and plan time)

• Many commercial applications that date back to the early 1970’s, see http://www.che.utexas.edu/~qin/cpcv/cpcv14.html
  – Much of this work was in process control - very nonlinear dynamics, but not particularly fast.

• As computer speed has increased, there has been renewed interest in applying this approach to applications with faster time-scale: trajectory design for aerospace systems.
Basic Formulation

• Given a set of plant dynamics (assume linear for now)

\[ x(k+1) = Ax(k) + Bu(k) \]
\[ z(k) = Cx(k) \]

and a cost function

\[ J = \sum_{j=0}^{N} \left\{ \|z(k+j|k)\|_{R_{zz}} + \|u(k+j|k)\|_{R_{uu}} \right\} + F(x(k+N|k)) \]

- \( \|z(k+j|k)\|_{R_{zz}} \) is just a short hand for a weighted norm of the state, and to be consistent with earlier work, would take

\[ \|z(k+j|k)\|_{R_{zz}} = z(k+j|k)^T R_{zz} z(k+j|k) \]

- \( F(x(k+N|k)) \) is a terminal cost function

• Note that if \( N \to \infty \), and there are no additional constraints on \( z \) or \( u \), then this is just the discrete LQR problem solved on page 3-14.

  - Note that that could have been written as just an input control sequence, but we choose to write it as a linear state feedback.
  
  - In the nominal case, there is no difference between the two.
  
  - But with modeling errors and disturbances, the state feedback form is much less sensitive.

  \[ \Rightarrow \] This is the main reason for using feedback.

• Issue: When limits on \( x \) and \( u \) are added, we can no longer find the general solution in analytic form \( \Rightarrow \) must solve it numerically.
• However, solving for a very long input sequence:
  – Does not make sense if one expects that the model is wrong and/or there are disturbances, because it is unlikely that the end of the plan will be implemented (a new one will be made by then)
  – Longer plans have more degrees of freedom and take much longer to compute.

• Typically design using a small $N \Rightarrow$ short plan that does not necessarily achieve all of the goals.
  – Classical hard question is how large should $N$ be?
  – If plan doesn’t reach the goal, then must develop an estimate of the remaining cost-to-go

• Typical problem statement: for finite $N$ ($F = 0$)

\[
\min_u J = \sum_{j=0}^{N} \left\{ \|z(k+j|k)\|_{R_{zz}} + \|u(k+j|k)\|_{R_{uu}} \right\}
\]

s.t. \[ x(k+j+1|k) = A x(k+j|k) + B u(k+j|k) \]
\[ x(k|k) \equiv x(k) \]
\[ z(k+j|k) = C x(k+j|k) \]

and \[ |u(k+j|k)| \leq u_m \]
- Consider converting this into a more standard optimization problem.

\[
z(k|k) = Cx(k|k)
\]

\[
z(k + 1|k) = Cx(k + 1|k) = C(Ax(k|k) + Bu(k|k))
= CAx(k|k) + CBu(k|k)
\]

\[
z(k + 2|k) = Cx(k + 2|k)
= C(Ax(k + 1|k) + Bu(k + 1|k))
= CA(Ax(k|k) + Bu(k|k)) + CBu(k + 1|k)
= CA^2x(k|k) + CABu(k|k) + CBu(k + 1|k)
\]

\[
\vdots
\]

\[
z(k + N|k) = CA^Nx(k|k) + CA^{N-1}Bu(k|k) + \cdots + CBu(k + (N - 1)|k)
\]

- Combine these equations into the following:

\[
\begin{bmatrix}
z(k|k) \\
z(k + 1|k) \\
z(k + 2|k) \\
\vdots \\
z(k + N|k)
\end{bmatrix}
= \begin{bmatrix}
C \\
CA \\
CA^2 \\
\vdots \\
CA^N
\end{bmatrix}
\begin{bmatrix}
x(k|k)
\end{bmatrix}
\]

\[
+ \begin{bmatrix}
0 & 0 & 0 & \cdots & 0 \\
CB & 0 & 0 & \cdots & 0 \\
CAB & CB & 0 & \cdots & 0 \\
\vdots \\
CA^{N-1}B & CA^{N-2}B & CA^{N-3}B & \cdots & CB
\end{bmatrix}
\begin{bmatrix}
u(k|k) \\
u(k + 1|k) \\
u(k + 2|k) \\
\vdots \\
u(k + N - 1|k)
\end{bmatrix}
\]
• Now define

\[
 Z(k) \equiv \begin{bmatrix}
 z(k|k) \\
 \vdots \\
 z(k+N|k)
\end{bmatrix} \quad U(k) \equiv \begin{bmatrix}
 u(k|k) \\
 \vdots \\
 u(k+N-1|k)
\end{bmatrix}
\]

then, with \( x(k|k) = x(k) \)

\[
 Z(k) = Gx(k) + HU(k)
\]

• Note that

\[
 \sum_{j=0}^{N} z(k+j|k)^T R_{zz} z(k+j|k) = Z(k)^T W_1 Z(k)
\]

with an obvious definition of the weighting matrix \( W_1 \)

• Thus

\[
 Z(k)^T W_1 Z(k) + U(k)^T W_2 U(k) \\
 = (Gx(k) + HU(k))^T W_1 (Gx(k) + HU(k)) + U(k)^T W_2 U(k) \\
 = x(k)^T H_1 x(k) + H_2^T U(k) + \frac{1}{2} U(k)^T H_3 U(k)
\]

where

\[
 H_1 = G^T W_1 G, \quad H_2 = 2(x(k)^T G^T W_1 H)^T, \quad H_3 = 2(H^T W_1 H + W_2)
\]

• Then the MPC problem can be written as:

\[
 \min_{U(k)} \tilde{J} = H_2^T U(k) + \frac{1}{2} U(k)^T H_3 U(k)
\]

s.t. \[
 \begin{bmatrix}
 I_N \\
 -I_N
\end{bmatrix} U(k) \leq u_m
\]
**Toolboxes**

- **Key point:** the MPC problem is now in the form of a standard **quadratic program** for which standard and efficient codes exist.

```
QUADPROG Quadratic programming. 
X=QUADPROG(H,f,A,b) attempts to solve the 
quadratic programming problem:

min 0.5*x'*H*x + f'*x  subject to:  A*x <= b
x

X=QUADPROG(H,f,A,b,Aeq,beq) solves the problem %
above while additionally satisfying the equality% 
constraints Aeq*x = beq.
```

- Several Matlab toolboxes exist for testing these ideas
  - MPC toolbox by Morari and Ricker — extensive analysis and design tools.
  - MPCtools¹ enables some MPC simulation and is free
    [www.control.lth.se/user/johan.akesson/mpctools/](http://www.control.lth.se/user/johan.akesson/mpctools/)

---

Observations

• Current form assumes that full state is available - can hookup with an estimator

• Current form assumes that we can sense and apply corresponding control immediately
  – With most control systems, that is usually a reasonably safe assumption
  – Given that we must re-run the optimization, probably need to account for this computational delay - different form of the discrete model - see F&P (chapter 2)

• If the constraints are not active, then the solution to the QP is that

$$U(K) = -H_3^{-1}H_2$$

which can be written as:

$$u(k|k) = -\begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix} (H^T W_1 H + W_2)^{-1} H W_1 G x(k)$$

$$= -K x(k)$$

which is just a state feedback controller.

  – Can apply this gain to the system and check the eigenvalues.
• What can we say about the stability of MPC when the constraints are active? ²
  – Depends a lot on the terminal cost and the terminal constraints. ³
• Classic result: ⁴ Consider a MPC algorithm for a linear system with constraints. Assume that there are terminal constraints:
  – \( x(k + N) = 0 \) for predicted state \( x \)
  – \( u(k + N) = 0 \) for computed future control \( u \)
If the optimization problem is feasible at time \( k \), then \( x = 0 \) is stable.

**Proof:** Can use the performance index \( J \) as a Lyapunov function.
  – Assume there exists a feasible solution at time \( k \) and cost \( J_k \)
  – Can use that solution to develop a feasible candidate at time \( k+1 \), by simply adding \( u(k + N + 1) = 0 \) and \( x(k + N + 1) = 0 \).
  – **Key point:** can estimate the candidate controller performance
    
    \[
    \tilde{J}_{k+1} = J_k - \{\|z(k|k)\|_{R_{zz}} + \|u(k|k)\|_{R_{uu}}\} \leq J_k - \{\|z(k|k)\|_{R_{zz}}\}
    \]
  – This candidate is suboptimal for the MPC algorithm, hence \( J \) decreases even faster \( J_{k+1} \leq \tilde{J}_{k+1} \)
  – Which says that \( J \) decreases if the state cost is non-zero (observability assumptions) \( \Rightarrow \) but \( J \) is lower bounded by zero.

• Mayne et al. [2000] provide an excellent review of other strategies for proving stability
  – Terminal cost and constraint sets

---

Example: Helicopter

- Consider a system similar to the Quansar helicopter\(^5\)

- There are 2 control inputs – voltage to each fan \(V_f, V_b\)

- A simple dynamics model is that:

\[
\begin{align*}
\ddot{\theta}_e &= K_1(V_f + V_b) - T_g/J_e \\
\dot{\theta}_r &= -K_2 \sin(\theta_p) \\
\dot{\theta}_p &= K_3(V_f - V_b)
\end{align*}
\]

and there are physical limits on the elevation and pitch:

\[-0.5 \leq \theta_e \leq 0.6 \quad -1 \leq \theta_p \leq 1\]

- Model can be linearized and then discretized \(T_s = 0.2\) sec.

---

\(^5\)ISSN 02805316 ISRN LUTFD2/TFRT- -7613- -SE MPCtools 1.0 Reference Manual Johan Akesson Department of Automatic Control Lund Institute of Technology January 2006
Figure 4: Response with $N = 3$

Figure 5: Response with $N = 10$

Figure 6: Response with $N = 25$