Analytical Solution of Constrained LQ Optimal Control for Horizon 2

José De Doná

September 2004
1. A Motivating Example
   - Cautious Design
   - Serendipitous Design
   - Tactical Design

2. Explicit Solutions
   - Explicit vs Implicit

3. A Simple Problem with $N = 2$
   - Dynamic Programming

4. Motivating Example Revisited

5. Conclusions
Consider the double integrator plant:

\[ \frac{d^2 y(t)}{dt^2} = u(t). \]

The zero-order hold discretisation with sampling period 1 is:

\[ x_{k+1} = Ax_k + Bu_k, \]
\[ y_k = Cx_k, \]

with

\[ A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}. \]

Assume the actuator has a maximum and minimum allowed value (saturation) of ±1. Thus, the controller has to satisfy the constraint: \( |u_k| \leq 1 \) for all \( k \).
A Motivating Example

The schematic of the feedback control loop is shown in the Figure.

**Figure:** Feedback control loop.

“sat” represents the actuator modelled by the *saturation function*

\[
\text{sat}(u) = \begin{cases} 
1 & \text{if } u > 1, \\
 u & \text{if } |u| \leq 1, \\
-1 & \text{if } u < -1.
\end{cases}
\]
Suppose the initial condition is $x_0 = \begin{bmatrix} -6 & 0 \end{bmatrix}^T$.

We start with a “low gain” linear state feedback controller (LQR), computed so as to minimise the objective function:

$$V_\infty(\{x_k\}, \{u_k\}) = \frac{1}{2} \sum_{k=0}^{\infty} \left( x_k^T Q x_k + u_k^T R u_k \right),$$

with weighting matrices $Q = C^T C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ and $R = 20$.

The solution is then computed by solving the algebraic Riccati equation $P = A^T P A + Q - K^T (R + B^T P B) K$, where $K = (R + B^T P B)^{-1} B^T P A$. We obtain the linear state feedback law:

$$u_k = -K x_k = -\begin{bmatrix} 0.1603 & 0.5662 \end{bmatrix} x_k.$$
“Slow gain” control: \( Q = C^T C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \), \( R = 20 \), \( u_k = -Kx_k \).

The resulting input and output sequences are shown in the Figure.

We can see from the figure that the input \( u_k \) satisfies the given constraint \( |u_k| \leq 1 \), for all \( k \); for this initial condition.

The response is rather slow. The “settling time” is of the order of 8 samples.
“High gain” control: \( Q = C^T C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad R = 2, \quad u_k = \text{sat}(-Kx_k) \).

The resulting input and output sequences are shown in the Figure.

We can see from the figure that the input \( u_k \) satisfies the given constraint \( |u_k| \leq 1 \), for all \( k \). The controller makes better use of the available control authority.

The amount of overshoot is essentially the same. The “settling time” is of the order of 5 samples.
“High gain” control: $Q = C^T C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$, $R = 0.1$, $u_k = \text{sat}(-Kx_k)$.

The resulting input and output sequences are shown in the Figure.

We can see from the figure that the input $u_k$ satisfies the given constraint $|u_k| \leq 1$, for all $k$. The control sequence stays saturated much longer.

Significant overshoot. The “settling time” blows out to 12 samples.
Recapitulation

- Going from \( R = 20 \rightarrow 2 \): same overshoot, faster response.
- Going from \( R = 2 \rightarrow 0.1 \): large overshoot, long settling time.

Let us examine the state space trajectory corresponding to the serendipitous strategy \( R = 0.1 \), \( u_k = \text{sat}( -Kx_k ) \):

The control law \( u = -\text{sat}(Kx) \) partitions the state space into three regions. Hence, the controller can be characterised as a \textit{switched} control strategy:

\[
 u = \mathcal{K}(x) = \begin{cases} 
 -Kx & \text{if } x \in R_0, \\
 1 & \text{if } x \in R_1, \\
 -1 & \text{if } x \in R_2.
\end{cases}
\]
Heuristically, we can think, in this example, of $x^2$ as “velocity” and $x^1$ as “position.” Now, in our attempt to change the position rapidly (from $-6$ to $0$), the velocity has been allowed to grow to a relatively high level ($+3$). This would be fine if the braking action were unconstrained. However, our input (including braking) is limited to the range $[-1, 1]$. Hence, the available braking is inadequate to “pull the system up,” and overshoot occurs.
We will now start “afresh” with a formulation that incorporates constraints from the beginning in the design process.

A sensible idea would seem to be to try to “look ahead” and take account of future input constraints (that is, the limited braking authority available).

We now consider a Model Predictive Control law with prediction horizon $N = 2$. At each sampling instant $i$ and for the current state $x_i$, the two-step objective function:

$$V_2(\{x_k\}, \{u_k\}) = \frac{1}{2} x_{i+2}^T P x_{i+2} + \frac{1}{2} \sum_{k=i}^{i+1} \left( x_k^T Q x_k + u_k^T R u_k \right),$$

is minimised subject to the equality and inequality constraints:

$$x_{k+1} = A x_k + B u_k, \quad k = i, i + 1,$$
$$|u_k| \leq 1, \quad k = i, i + 1.$$
In the objective function (1), we set, as before, $Q = C^T C$, $R = 0.1$. The terminal state weighting matrix $P$ is taken to be the solution of the algebraic Riccati equation $P = A^T PA + Q - K^T (R + B^T PB) K$, where $K = (R + B^T PB)^{-1} B^T PA$.

As a result of minimising (1) subject the constraints (2), we obtain an optimal fixed-horizon control sequence

$$\{u_{i}^{\text{opt}}, u_{i+1}^{\text{opt}}\}$$

We then apply the resulting value of $u_{i}^{\text{opt}}$ to the system in a receding horizon control form.

We can see that this strategy has the ability to “look ahead” by considering the constraints not only at the current time $i$, but also one step ahead $i + 1$. 
MPC: \( Q = C^T C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \ R = 0.1, \ \text{Horizon} \ N = 2 \)

The resulting input and output sequences are shown in the Figure.

Dashed line: control \( u_k = -Kx_k \).
Solid line: MPC

We can see from the figure that the output trajectory with constrained input now has minimal overshoot. Thus, the idea of “looking ahead” has paid dividends.
Recapitulation

As we will see in this part of the course, the receding horizon control strategy (MPC) we have used can be described as a partition of the state space into different regions in which affine control laws hold.

- **Serendipitous strategy**
  \[ R = 0.1, \quad u_k = \text{sat}(-Kx_k). \]

- **Receding horizon tactical design**
  \[ R = 0.1, \quad N = 2. \]
Before we start studying how to find an explicit characterisation of the MPC solution, let us first define what procedures we would call explicit and which ones we would call implicit.

- **Explicit solution**
  \( p = \text{parameter, } a, b = \text{constants} \).

  \( f_p(z) = z^2 + 2apz + bp \)

  \( \frac{\partial f_p}{\partial z} = 0 \Rightarrow z_{\text{opt}}(p) = -ap \)

- **Implicit (numerical) solution**
  \( p = \text{parameter, } a, b = \text{constants} \).

  \( f_p(z) = z^2 + 2apz + bp \)
We consider the discrete time system

$$x_{k+1} = Ax_k + Bu_k,$$

where $x_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}$. The pair $(A, B)$ is assumed to be stabilisable. We consider the following fixed horizon optimal control problem:

$$\mathcal{P}_N(x) : \quad V_N^{\text{opt}}(x) = \min V_N(\{x_k\}, \{u_k\}),$$

subject to:

- $x_0 = x$,
- $x_{k+1} = Ax_k + Bu_k$ for $k = 0, 1, \ldots, N - 1$,
- $u_k \in \mathbb{U} = [-\Delta, \Delta]$ for $k = 0, 1, \ldots, N - 1$,

where $\Delta > 0$ is the input constraint level. The objective function is:

$$V_N(\{x_k\}, \{u_k\}) = \frac{1}{2} x_N^T P x_N + \frac{1}{2} \sum_{k=0}^{N-1} \left( x_k^T Q x_k + u_k^T R u_k \right).$$
Let the control sequence that achieves the minimum in (3) be:

\[ \{u_0^{\text{OPT}}, u_1^{\text{OPT}}, \ldots, u_{N-1}^{\text{OPT}}\} \]

The Receding horizon control law, which depends on the current state \( x = x_0 \), is

\[ \mathcal{K}_N(x) = u_0^{\text{OPT}}. \]

Can we obtain an \textbf{explicit} expression for \( \mathcal{K}_N(\cdot) \) defined above, as a function of the parameter \( x \)?
Consider now the following fixed horizon optimal control problem with prediction horizon $N = 2$:

$$\mathcal{P}_2(x) : \quad V_2^{\text{opt}}(x) = \min V_2(\{x_k\}, \{u_k\}), \quad (5)$$

subject to:  
$$x_0 = x$$
$$x_{k+1} = Ax_k + Bu_k \quad \text{for } k = 0, 1,$$
$$u_k \in U = [-\Delta, \Delta] \quad \text{for } k = 0, 1.$$

The objective function in (5) is

$$V_2(\{x_k\}, \{u_k\}) = \frac{1}{2} x_2^T P x_2 + \frac{1}{2} \sum_{k=0}^{1} \left( x_k^T Q x_k + u_k^T R u_k \right).$$
A Simple Problem with $N = 2$

Objective function:

$$V_2(\{x_k\}, \{u_k\}) = \frac{1}{2} x_2^T P x_2 + \frac{1}{2} \sum_{k=0}^{1} \left( x_k^T Q x_k + u_k^T R u_k \right). \quad (6)$$

The matrices $Q$ and $R$ are assumed positive semi-definite and definite respectively. $P$ is taken as the solution to the algebraic Riccati equation

$$P = A^T PA + Q - K^T \bar{R} K,$$

where $K = \bar{R}^{-1} B^T PA$, $\bar{R} = R + B^T P B$.

Let the control sequence that minimises (6) be

$$\{u_0^{OPT}, u_1^{OPT}\}.$$

Then the RHC law is given by the first element of the optimal sequence (which depends on the current state $x_0 = x$), that is,

$$K_2(x) = u_0^{OPT}.$$
We will use **Dynamic Programming** to find the analytical solution for this problem. We define the partial value functions as:

\[
V_0^{\text{OPT}}(x_2) = \frac{1}{2} x_2^T P x_2
\]

\[
V_1^{\text{OPT}}(x_1) = \min_{u_1 \in \mathbb{U}} \left\{ \frac{1}{2} x_2^T P x_2 + \frac{1}{2} x_1^T Q x_1 + \frac{1}{2} u_1^T R u_1 \mid x_2 = A x_1 + B u_1 \right\}
\]

\[
V_2^{\text{OPT}}(x) = \min_{u_k \in \mathbb{U}} \left\{ \frac{1}{2} x_{k+1}^T P x_{k+1} + \frac{1}{2} \sum_{k=0}^{1} \left( x_k^T Q x_k + u_k^T R u_k \right) \mid x_{k+1} = A x_k + B u_k \right\}
\]
The dynamic programming algorithm makes use of the **Principle of Optimality**, which states that *any portion of the optimal trajectory is itself an optimal trajectory*:

$$V_{k}^{\text{OPT}}(x) = \min_{u \in U} \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + V_{k-1}^{\text{OPT}}(A x + B u),$$

where $u$ and $x$ denote, $u = u_k$ and $x = x_k$, respectively.

In the sequel we will use the saturation function:

$$\text{sat}_\Delta(u) = \begin{cases} 
\Delta & \text{if } u > \Delta, \\
 u & \text{if } |u| \leq \Delta, \\
-\Delta & \text{if } u < -\Delta.
\end{cases}$$
Theorem (RHC Characterisation for $N = 2$)

The RHC law has the form

$$K_2(x) = \begin{cases} -\text{sat}_\Delta(Gx + h) & \text{if } x \in \mathbb{Z}^- , \\ -\text{sat}_\Delta(Kx) & \text{if } x \in \mathbb{Z} , \\ -\text{sat}_\Delta(Gx - h) & \text{if } x \in \mathbb{Z}^+ , \end{cases}$$

(7)

where the gains $K, G \in \mathbb{R}^{1 \times n}$ and the constant $h \in \mathbb{R}$ are

$$K = \bar{R}^{-1} B^\top PA, \quad G = \frac{K + KBKA}{1 + (KB)^2}, \quad h = \frac{KB}{1 + (KB)^2}\Delta,$$

and the sets $(\mathbb{Z}^-, \mathbb{Z}, \mathbb{Z}^+)$ are defined by

$$\mathbb{Z}^- = \{x : K(A - BK)x < -\Delta\} , \quad \mathbb{Z} = \{x : |K(A - BK)x| \leq \Delta\} ,$$

$$\mathbb{Z}^+ = \{x : K(A - BK)x > \Delta\} .$$
Dynamic Programming for $N = 2$

Outline of the Proof:

(i) The partial value function $V_0^{\text{OPT}}$:
Here $x = x_2$. By definition, the partial value function at time $k = N = 2$ is

$$V_0^{\text{OPT}}(x) = \frac{1}{2} x^T P x \quad \text{for all } x \in \mathbb{R}^n.$$ 

(ii) The partial value function $V_1^{\text{OPT}}$:
Here $x = x_1$ and $u = u_1$. By the principle of optimality, for all $x \in \mathbb{R}^n$,

$$V_1^{\text{OPT}}(x) = \min_{u \in \mathcal{U}} \left\{ \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + V_0^{\text{OPT}}(Ax + Bu) \right\}$$

$$= \min_{u \in \mathcal{U}} \left\{ \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + \frac{1}{2} (Ax + Bu)^T P (Ax + Bu) \right\}$$

$$= \min_{u \in \mathcal{U}} \left\{ \frac{1}{2} x^T P x + \frac{1}{2} \bar{R} (u + Kx)^2 \right\}.$$
Dynamic Programming for $N = 2$

$$V_1^{OPT}(x) = \min_{u \in U} \left\{ \frac{1}{2} x^T P x + \frac{1}{2} \bar{R}(u + K x)^2 \right\}.$$ 

From the convexity of the function $\bar{R}(u + K x)^2$ it then follows that the constrained ($u \in U$) optimal control law is given by

$$u_1^{OPT} = \text{sat}_\Delta(-K x) = -\text{sat}_\Delta(K x) \quad \text{for all } x \in \mathbb{R}^n.$$
Substituting the optimal control

\[ u_{1}^{\text{opt}} = -\text{sat}_{\Delta}(Kx) \quad \text{for all } x \in \mathbb{R}^n. \]

into the minimisation problem

\[ V_{1}^{\text{opt}}(x) = \min_{u \in U} \left\{ \frac{1}{2} x^T P x + \frac{1}{2} \bar{R} (u + Kx)^2 \right\}. \]

we have that the partial value function at time \( k = N - 1 = 1 \) is

\[ V_{1}^{\text{opt}}(x) = \frac{1}{2} x^T P x + \frac{1}{2} \bar{R} [Kx - \text{sat}_{\Delta}(Kx)]^2 \quad \text{for all } x \in \mathbb{R}^n. \]
(iii) The partial value function $V_{2}^{\text{opt}}$:
Here $x = x_0$ and $u = u_0$. By the principle of optimality, we have that, for all $x \in \mathbb{R}^n$,

$$V_{2}^{\text{opt}}(x) = \min_{u \in \mathcal{U}} \left\{ \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + V_{1}^{\text{opt}}( Ax + Bu) \right\}$$

$$= \min_{u \in \mathcal{U}} \left\{ \frac{1}{2} x^T Q x + \frac{1}{2} u^T R u + \frac{1}{2} ( Ax + Bu)^T P ( Ax + Bu) \right.$$

$$\left. + \frac{1}{2} \bar{R} [ K ( Ax + Bu ) - \text{sat}_{\Delta}(K ( Ax + Bu ))]^2 \right\}$$

$$= \frac{1}{2} \min_{u \in \mathcal{U}} \left\{ x^T P x + \bar{R} ( u + Kx)^2 \right.$$

$$\left. + \bar{R} [ KAx + KBu - \text{sat}_{\Delta}(K Ax + KBu)]^2 \right\}.$$
Dynamic Programming for $N = 2$

$$u_0^{\text{opt}} = \arg\min_{u \in U} \left\{ \tilde{R}(u + Kx)^2 + \tilde{R} [KAx + KBu - \text{sat}_\Delta(KAx + KBu)]^2 \right\}.$$ 

Case (a): $x \in \mathbb{Z}^-$

$x \in \mathbb{Z}^-$

$\iff$

$KAx + KB(-Kx) < -\Delta$
We conclude that

\[ u_0^{\text{opt}} = \arg \min_{u \in U} \left\{ \tilde{R}(u + Kx)^2 + \tilde{R} \left[ KA x + KBu - \text{sat}_\Delta(KA x + KBu) \right]^2 \right\} \]

\[ = \arg \min_{u \in U} \left\{ \tilde{R}(u + Kx)^2 + \tilde{R} \left[ KA x + KBu + \Delta \right]^2 \right\} \]

Hence

\[ u_0^{\text{opt}} = -\text{sat}_\Delta(Gx + h) \quad \text{for all } x \in \mathbb{Z}^- , \]

where

\[ G = \frac{K + KBKA}{1 + (KB)^2}, \quad h = \frac{KB}{1 + (KB)^2} \Delta. \]
Dynamic Programming for $N = 2$

$$u_0^{\text{OPT}} = \arg \min_{u \in U} \left\{ \bar{R} (u + Kx)^2 + \bar{R} \left[ KAx + KBu - \text{sat}_\Delta (KAx + KBu) \right]^2 \right\}.$$  

Case (b): $x \in \mathbb{Z}$

$$x \in \mathbb{Z} \iff |KAx + KB(-Kx)| \leq \Delta$$
We conclude that

\[ u_0^{\text{opt}} = \arg \min_{u \in U} \left\{ \bar{R}(u + Kx)^2 + \bar{R} \left[ KA_x + KBu - \text{sat}_\Delta(KAx + KBu) \right]^2 \right\} \]

\[ = \arg \min_{u \in U} \left\{ \bar{R}(u + Kx)^2 \right\} \]

Hence

\[ u_0^{\text{opt}} = -\text{sat}_\Delta(Kx) \quad \text{for all } x \in \mathbb{Z}. \]
Dynamic Programming for $N = 2$

$$u_0^{\text{opt}} = \arg \min_{u \in U} \left\{ R(u + Kx)^2 + R [KAx + KBu - \text{sat}_\Delta(KAx + KBu)]^2 \right\}.$$  

Case (c): $x \in \mathbb{Z}^+$  

\[
x \in \mathbb{Z}^+ \iff KAx + KB(-Kx) > \Delta
\]
We conclude that

\[ u_0^{\text{opt}} = \arg\min_{u \in U} \left\{ \tilde{R} (u + Kx)^2 + \tilde{R} [KAx + KBu - \text{sat}_\Delta(KAx + KBu)]^2 \right\} \]

\[ = \arg\min_{u \in U} \left\{ \tilde{R} (u + Kx)^2 + \tilde{R} [KAx + KBu - \Delta]^2 \right\} \]

Hence

\[ u_0^{\text{opt}} = -\text{sat}_\Delta(Gx - h) \quad \text{for all } x \in \mathbb{Z}^+, \]

where

\[ G = \frac{K + KBKA}{1 + (KB)^2}, \quad h = \frac{KB}{1 + (KB)^2} \Delta. \]
Motivating Example Revisited

We have, from the previous result, that the control law is:

\[ K_2(x) = \begin{cases} 
- \text{sat}_\Delta(Gx + h) & \text{if } x \in \mathbb{Z}^-, \\
- \text{sat}_\Delta(Kx) & \text{if } x \in \mathbb{Z}, \\
- \text{sat}_\Delta(Gx - h) & \text{if } x \in \mathbb{Z}^+, 
\end{cases} \]

or, equivalently:

\[ K_2(x) = \begin{cases} 
- \Delta & \text{if } x \in R_1, \\
- Gx - h & \text{if } x \in R_2, \\
- Kx & \text{if } x \in R_3, \\
- Gx + h & \text{if } x \in R_4, \\
\Delta & \text{if } x \in R_5. 
\end{cases} \]
Conclusions

Note that we have obtained, for the case $N = 2$, the following expression for the receding horizon control law

$$K_2(x) = \begin{cases} 
-\Delta & \text{if } x \in R_1, \\
-Gx - h & \text{if } x \in R_2, \\
-Kx & \text{if } x \in R_3, \\
-Gx + h & \text{if } x \in R_4, \\
\Delta & \text{if } x \in R_5.
\end{cases}$$

That is, the control law can be characterised as a **piecewise affine function** of the state $x$.

The calculation using Dynamic Programming can be extended to longer prediction horizons, $N > 2$. However, we will instead reconsider this same problem, and further extensions, using **geometric arguments** later in the course.