4F3: Optimal and Predictive Control

Need to Know

Mrinank Sharma

April 27, 2019

Please note that the margins of these notes can be used to check factual recall simply by covering up the right hand text.

1 Introduction

Convex Optimisation

A convex optimisation problem is written as follows:

$$\begin{aligned} &\min_x f_0(x) \\ &\text{s.t. } f_i(x) \leq b_i \qquad i = 1, \ldots, m \\ &\quad\; h_i(x) = 0 \qquad i = 1, \ldots, p \end{aligned} \tag{1}$$

where the objective and inequality constraint functions are convex i.e. $f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y)$ and the equality constraint functions are linear plus a constant for $\alpha + \beta = 1$, $\alpha \geq 0$ and $\beta \geq 0$.

Significance of Convex Optimisation

Whilst there is no analytical solution for this class of problems, there is a global minimum and there are reliable and efficient algorithms. There are many tricks for transforming problems into convex form.

Notation:

$$||y||_\infty = \max_t \sqrt{y^T(t)y(t)} \tag{2}$$

$$||y||_2^2 = \int_{-\infty}^{\infty} y^T(t)y(t)dt \tag{3}$$

2 Optimal Control and Dynamic Programming

2.1 Discrete Time Finite Horizon Optimal Control

State $x \in \mathcal{X}$, input $u \in \mathcal{U}$. Dynamics of the system are:

$$x_{k+1} = f(x_k, u_k) \text{ where } f(\cdot,\cdot) : \mathcal{X} \times \mathcal{U} \to \mathcal{X} \tag{4}$$

Cost Function

Given an initial condition, the input sequence deterministically generates a state-sequence. The cost function, in general, over a finite horizon h time-steps into the future, can be written as:

$$J(x_0, u_0, \ldots, u_{h-1}) = \sum_{k=0}^{h-1} \underbrace{c(x_k, u_k)}_{\text{stage cost}} + \overbrace{J_h(x_h)}^{\text{terminal cost}} \tag{5}$$

Our objective is to find the input sequence which minimises the above cost function for a given initial state. Note: in certain cases, J^* may not be well-defined and the optimal input sequence may not exist, or may be non-unique.

Bellman's Principle of Optimality

Assume that the optimal control sequence, $u_0^*, \ldots, u_{h-1}^*$, leads us from x_0 to x_k at step k. Then the truncated sequence $u_k^*, \ldots, u_{h-1}^*$ is a solution to the truncated problem:

$$u_k^*, \ldots, u_{h-1}^* = \arg \min_{u_k, \ldots, u_{h-1}} \sum_{i=1}^{h-1} c(x_i, u_i) + J_h(x_h) \tag{6}$$

Value Function

The *Value Function* is defined as:

$$V(x,k) \triangleq \min_{u_k,\dots,u_{h-1}} \sum_{i=1}^{h-1} c(x_i,u_i) + J_h(x_h) \tag{7}$$

where $x_i, i > k$ is generated using the system dynamics. Also known as the *cost-to-go*, this function is the optimal *additional cost* from the kth step.

Assume that $V(x,k+1)$ is known for all x. Then

Proof: Value Function Recursion

$$V(x,k) = \min_{u_k} \left\{ c(x,u_k) + V(x_{k+1},k+1) \right\} \tag{8}$$

Thus, the optimal control and cost can be found by solving the *Dynamic Programming Equation* (Eq. 8) starting with the final condition $V(x,h) = J_h(x)$. The optimal control is the sequence of $\{u_k\}$ minimising the cost at each stage of the dynamic programming equation.

Comments

The magic of dynamic programming has converted a minimisation over h inputs to a sequence of h minimisations over one input. Solving this equation gives the optimal control for all values of x_0 over this horizon length. This can always be solved if the state and input can only take a finite number of values.

2.2 Discrete-Time Finite Horizon Linear Quadratic Regulator

Discrete-time LQR

State $x \in \mathcal{X}$, input $u \in \mathcal{U}$. Dynamics of the system are:

$$x_{k+1} = Ax_k + Bu_k \tag{9}$$

The initial condition, x_0, is assumed given. The cost function is

$$J(x_0, u_0, \dots, u_{h-1}) = \sum_{k=1}^{h-1} x_k^T Q x_k + u_k^T R u_k + x_h^T X_h x_h \tag{10}$$

Q, R, X_h are symmetric matrices with $Q \geq 0$, $R > 0$ and $X_h \geq 0$. This implies that R^{-1} exists. Therefore, for the penultimate time-step:

$$\begin{aligned} V(x,h-1) &= \min_u \left\{ x^T Q x + u^T R u + (Ax+Bu)^T X_h (A_x + Bu)) \right\} \\ &= x^T \underbrace{(Q + A^T X_h A - A^T X_h B (R + B^T X_h B)^{-1} B^T X_h A)}_{X_{h-1}} x \end{aligned} \tag{11}$$

which is another quadratic form in x. Thus, the backwards difference equation

$$X_{k-1} = Q + A^T X_k A - A^T X_k B (R + B^T X_k B)^{-1} B^T X_k A) \tag{12}$$

is solved, and the optimal control is found as the minimiser at each time-step:

Prove the state feedback equation

$$u_k = -(R + B^T X_{k+1} B)^{-1} B^T X_{k+1} A x_k \tag{13}$$

which is state-feedback control.

2.3 Continuous Time Dynamic Programming

State $x \in \mathbb{R}^n$, input $u \in \mathcal{U} \subseteq \mathbb{R}^m$. Dynamics of the system are:

$$\dot{x} = f(x,u) \text{ where } f(\cdot,\cdot) : \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}^n \tag{14}$$

Given $x_0 \in \mathcal{X}$ and a horizon $T \geq 0$, each input function $u(\cdot) : [0,T] \to \mathcal{U}$ generates a state trajectory satisfying the above dynamics. The cost function is now:

Continuous Time: New Cost Function

$$J(x_0, u(\cdot)) = \int_0^T c(x(t),u(t))dt + J_t(x(T)) \tag{15}$$

The aim is to find the best input function:

$$u^*(\cdot) = \arg\min_{u(\cdot)} J(x_0, u(\cdot)) \tag{16}$$

Technical assumptions are placed on $f, \mathcal{U}, c, J_h$ to ensure that a unique trajectory exists, a unique optimal control exists and that there is a minimum of the cost function.

Bellman Optimality: Continuous Time

Assumption that the optimal control $u^*(\cdot) : [0, T] \to \mathcal{U}$ leads from $x(0)$ to $x(t)$ at $t < T$. Then the truncated control $u^*(\cdot) : [t, T] \to \mathcal{U}$ is a solution to the truncated problem:

$$\min_{u(\cdot)} \int_t^T c(x(\tau), u(\tau)) d\tau + J_t(x(T)) \tag{17}$$

Value Function

The value (or cost-to-go) function $V : \mathcal{X} \times [0, T] \to \mathbb{R}$ is defined as:

$$V(x(t), t) \triangleq \min_{u(\cdot)} \int_t^T c(x(\tau), u(\tau)) d\tau + J_t(x(T)) \tag{18}$$

Derive Value Function PDE

The recursive algorithm in discrete-time is converted to a partial differential equation known as the *Hamilton-Jacobi-Bellman PDE*:

$$-\frac{\partial V(x,t)}{\partial t} = \min_{u(\cdot)} \left\{ c(x,u) + \frac{\partial V(x,t)}{\partial x} f(x,u) \right\} \tag{19}$$

Solving the above PDE with condition $V(x, T) = J_t(x)$ yields the solution. The optimal cost is given by $V(x_0, 0)$ and the optimal input is:

$$u^*(t) = \arg\min_{u \in \mathcal{U}} \left\{ c(x(t), u) + \frac{\partial V(x,t)}{\partial x} f(x(t), u) \right\} \tag{20}$$

The optimisation over $u(\cdot)$ has been converted to a pointwise optimisation over $u \in \mathcal{U}$. **Note:** to solve the problem, a PDE needs to be solved, but there can be technical difficulties with this e.g. does a solution exist, and if so, in what sense? It is computable?

2.4 Continuous-Time LQR

$x \in \mathbb{R}^n, u \in \mathbb{R}^m, x(0) = x_0$. Horizon of t_1. System dynamics are:

$$\dot{x} = Ax + Bu \tag{21}$$

The cost function is:

$$J(x_0, u(\cdot)) = \int_0^{t_1} x^T Q x + u^T R u \, dt \; + x(T)^T X_{t_1} x(T) \tag{22}$$

Value Function Form

Fill in the missing steps...

with $R = R^T > 0$, $Q = Q^T \geq 0$ and $X_{t_1} = X_{t_1}^T \geq 0$. . Let $V(x, t) = x^T X(t) x$. Defining the vector gradient as a row vector, the HJB reads:

$$\begin{aligned} -x^T \dot{X}(t) x &= \min_u \left\{ x^T Q x + u^T R u + 2x^T X(t)(Ax + Bu) \right\} \\ &= x^T (Q + XA + A^T X - XBR^{-1} B^T X) x \qquad (23) \\ u^*(t) &= -R^{-1} B^T X(t) x(t) \qquad (24) \end{aligned}$$

Riccati Equation

Thus, the following ODE (a **Riccati equation**) must be solved *backwards in time*

$$-\dot{X} = Q + XA + A^T X - XBR^{-1} B^T X \tag{25}$$

Empirical Findings

with terminal condition $x_0^T X(0) x_0$. Solving the value function backwards allows the optimal control function to be found, which is the then integrated forwards. It is found that when simulating the solution to the Riccati equation backwards in time, the terminal cost causes a transient, beyond which (i.e. earlier in time) the matrix X appears to converge in value.

2.5 Infinite Horizon Continuous Time LQR

Setup

The problem is setup with

$$\dot{x} = Ax + Bu \qquad x(0) = x_0 \qquad z = \begin{bmatrix} Cx \\ u \end{bmatrix}$$

The cost function is defined as:

$$J(x_0, u(\cdot)) = \int_0^{\infty} z(t)^T z \; dt \tag{26}$$

It is assumed that (A, B) is controllable and (A, C) is observable. The infinite horizon is like a finite, but very long horizon. The solution is given by the Riccati equation:

Technical Assumptions

Intuition

$$-\dot{X} = C^T C + XA + A^T X - XBR^{-1}B^T X \tag{27}$$

for **any** final condition, as it does not have a large effect for long times. Therefore, we expect that:

$$u^*(t) = -B^T X x(t) \tag{28}$$

$$\dot{x} = Ax + Bu = (A - BB^T X)x \tag{29}$$

where $X = X^T$ solves the *Control Algebraic Riccati Equation (CARE)*:

$$C^T C + XA + A^T X - XBR^{-1}B^T X = 0 \tag{30}$$

It can be shown that the CARE has a unique, symmetric, positive definite solution $X = X^T \geq 0$ and this this solution is stabilising i.e. $A - BB^T$ has all eigenvalues in the left half plane. This solution can be obtained as $\lim_{t \to -\infty} X(t)$ where $X(t)$ solves Eq. 25 for **any** final condition.

Let $X = X^T$ be the stabilising solution to the CARE. Defining $V(t) = x(t)^T X x(t)$ and integrating $\dot{V} + z^T z$ gives, for matrix X solving the CARE:

Alternative Derivation

$$V(\infty) - V(0) + ||z||_2^2 = ||u + B^T X x||_2^2 \tag{31}$$

Assuming $x(t) \to 0$ as $t \to \infty$, meaning that $V(\infty) = 0$, we have:

$$\underbrace{||z||_2^2}_{J(x(0),u(\cdot))} = x(0)^T X x(0) + \underbrace{||u + B^T X x||_2^2}_{0 \text{ if } u=-B^T X x} \tag{32}$$

3 $\mathcal{H}_2$ Optimal Control

3.1 The $\mathcal{H}_2$ norm

Definition 3.1 ($\mathcal{H}_2$ norm) *The $\mathcal{H}_2$ norm of a system defined by its matrix transfer function, $G(s)$, is defined as:*

$$||G||_2^2 = \int_{-\infty}^{\infty} trace\{G(j\omega)^* G(j\omega)\} d\omega \tag{33}$$

Therefore:

Properties of the $\mathcal{H}_2$ norm

$$||G||_2^2 = \sum_i ||G_i||_2^2 \tag{34}$$

Assuming that $G(s)$ is a transfer function from u to y, it can be shown that:

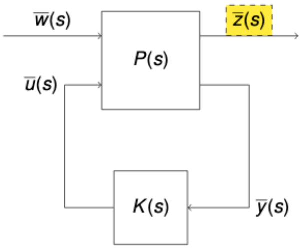
$$||y||_\infty \leq \frac{1}{\sqrt{2\pi}} ||G||_2 ||u||_2 \tag{35}$$

Consider the stable linear system

$$\dot{x} = Ax + Bu \qquad y = Cx$$

with transfer function $G(s) = C(sI - A)^{-1}B$. It can be shown that:

$$\frac{1}{\sqrt{2\pi}} ||G(s)||_2 = \sqrt{\text{trace}(B^T L B)}$$
$$\text{where } L = L^T \text{ solves } AL + LA + C^T C = 0 \tag{36}$$



3.2 Linear Fractional Transformations

Linear Fractional Transformations are a useful way of manipulating close-loop transfer functions and approach norm-optimal control problems.

Definition 3.2 *The lower LFT $\mathcal{F}_l(P(s),K(s))$ is defined as the closed loop transfer function from $\bar{w}(s)$ to $\bar{z}(s)$ i.e.*

$$\mathcal{F}_l(P(s),K(s)) = T_{\bar{w}\to\bar{z}} \tag{37}$$

Generalised Plant

$P(x)$ is known as the *Generalised Plant* and has the following block transfer function representation:

$$\begin{bmatrix}\bar{z}(s)\\ \bar{y}(s)\end{bmatrix} = \underbrace{\begin{bmatrix}P_{11}(s) & P_{12}(s)\\ P_{21}(s) & P_{22}(s)\end{bmatrix}}_{P(s)}\begin{bmatrix}\bar{u}(s)\\ \bar{w}(s)\end{bmatrix} \tag{38}$$

Interpreting w and z

Note that w usually represents some sort of control input, for example, driving disturbance noise, either in output or state. z is the control output and typically represents some sort of performance measure.

Note that it can be shown that:

Show form of LFT

$$\mathcal{F}_l(P(s),K(s)) = P_{11}(s) + P_{12}(s)K(s)(I - P_{22}(s)K(s))^{-1}P_{21}(s) \tag{39}$$

3.3 $\mathcal{H}_2$ Optimal State Feedback Control

Plant Setup

Consider the following generalised plant:

$$\begin{bmatrix}\dot{x}\\ z\\ y\end{bmatrix} = \begin{bmatrix} A & B_1 & B_2\\ \begin{bmatrix}C_1\\ 0\end{bmatrix} & 0 & \begin{bmatrix}0\\ I\end{bmatrix}\\ I & 0 & 0\end{bmatrix}\begin{bmatrix}x\\ w\\ u\end{bmatrix} \tag{40}$$

w represents an external disturbance and z penalises deviations from 0 in the state and using input energy. Note that since $y = x$, this is state feedback i.e. the controller has **direct access to the state**. We make the technical assumptions that the pair (A, B_2) is controllable and (A, C_1) is observable.

Objective

The objective is as follows:

$$\min_{K(s)\text{ stabilising}} ||\mathcal{F}_l(P(s),K(s))||_2 = \min_{K(s)\text{ stabilising}} \sqrt{2\pi}\sqrt{\sum_i \left|\left| z(t)|_{w(t)=e_i\delta(t)}\right|\right|_2^2} \tag{41}$$

Choose $X = X^T$ as the stabilising solution to the CARE equation. Setting $\bar{u}(s) = \underbrace{-B_2^T X}_{K(s)}\bar{x}(x)$ gives the optimal solution:

$$\frac{1}{2\pi}||\mathcal{F}_l(P(s),K(s))||_2^2 = \text{trace}(B_1^T X B_1) \tag{42}$$

3.4 $\mathcal{H}_2$ Optimal Output Feedback Control

Problem Setup

The driving noise, w_1, is now considered with observation noise, w_2, altering the generalised plant, P.

$$\begin{bmatrix}\dot{x}\\ z\\ y\end{bmatrix} = \begin{bmatrix} A & \begin{bmatrix}B_1 & 0\end{bmatrix} & B_2\\ \begin{bmatrix}C_1\\ 0\end{bmatrix} & 0 & \begin{bmatrix}0\\ I\end{bmatrix}\\ C_2 & \begin{bmatrix}0 & I\end{bmatrix} & 0\end{bmatrix}\begin{bmatrix}x\\ w\\ u\end{bmatrix} \tag{43}$$

In addition to the prior assumptions (controllable (A, B_2) and observable (A, C_1)), the additional assumptions that (A, B_1) is controllable and (A, C_2) is observable are made. This assumptions are appropriate for the dual problem.

Derivation

Choose $Y = Y^T$ as the stabilising solution (i.e. $A - YC_2^T C_2$ has eigenvalues in the LHP) to the **Filter Algebraic Riccati Equation(FARE)**:

FARE

$$0 = YA^T + AY + B_1 B_1^T - YC_2^T C_2 Y \tag{44}$$

Now consider the observer. The system dynamics are:

$$\begin{aligned} \dot{\tilde{x}} &= A^T \tilde{x} + F^T \tilde{w} + C_2^T \tilde{u} && (45) \\ \tilde{y} &= B_2^T \tilde{x} + \tilde{w} && (46) \end{aligned}$$

Thus, setting

$$\dot{\tilde{x}}_k = A^T \tilde{x}_k + F^T(\tilde{y} - B_2^T \tilde{x}_k) + C_2^T \tilde{u} \tag{47}$$

gives $\tilde{x}_k = \tilde{x} \Rightarrow \dot{\tilde{x}}_k = \dot{\tilde{x}}$, so provided the initial condition is the same for both, the observer trackers $\tilde{x}$. Then to minimise the additional cost, set

$$\tilde{u} = -\underbrace{C_2 Y}_{H^T} \tilde{x}_k \tag{48}$$

Therefore, the optimal K^T has the realisation:

$$\begin{bmatrix} \dot{\tilde{x}}_k \\ \tilde{u} \end{bmatrix} = \begin{bmatrix} A^T - F^T B_2^T - C_2^T H^T & F \\ -H^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{x}_k \\ \tilde{y} \end{bmatrix} \tag{49}$$

which gives the following optimal K.

$$\begin{bmatrix} \dot{x}_k \\ u \end{bmatrix} = \begin{bmatrix} A - B_2 F - HC_2 & -H \\ F & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y \end{bmatrix} \tag{50}$$

Note that there is an alternative realisation of this K which can be implemented in observer form.

Thus, nothing that the initial steps of the output feedback derivation remain valid, the optimal K achieves:

$$\min_{K(s)\text{ stabilising}} ||\mathcal{F}_l(P(s), K(s))||_2 = \sqrt{2\pi}\sqrt{\text{trace}(B_1^T X B_1)} + \sqrt{2\pi}\sqrt{\text{trace}(FYF^T)} \tag{51}$$

The closed loop poles are $\lambda_i(A - B_2 F) \bigcup \lambda_i(A - HC_2)$

Closed Loop Poles

4 $\mathcal{H}_\infty$ Optimal Control

4.1 $\mathcal{H}_\infty$ Norm

Definition 4.1 *The $\mathcal{H}_\infty$ norm of a stable linear system, $G(s)$, between signals u and y has two interpretations:*

(a): The Maximum Singular Value of $G(j\omega)$ i.e.

$$||G||_\infty = \max_\omega \bar{\sigma}(G(j\omega)) \tag{52}$$

(b): Signal Energy Bound:

$$||G||_\infty = \max_{\hat{u} \neq 0} \frac{||G\hat{u}||_2}{||\hat{u}||_2} = \max_{u \neq 0} \frac{||y||_2}{||u||_2} \tag{53}$$

Consider the system:

Calculating the $\mathcal{H}_\infty$ norm

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx && (54) \end{aligned}$$

with $x(0) = 0$. If (and only if) the Riccati equation:

$$A^T X + XA + C^T C + \frac{1}{\gamma^2} XBB^T X = 0 \tag{55}$$

has a solution, $X = X^T > 0$, then $||G||_\infty \leq \gamma$. This condition can be checked easily algebraically, so a bisection algorithm can be used to find the smallest value of γ which has a solution to this equation with would be the $\mathcal{H}_\infty$ norm.

4.2 $\mathcal{H}_\infty$ Optimal Control

Problem Setup

Consider the generalised plant with realisation:

$$\begin{bmatrix}\dot{x}\\ z\\ y\end{bmatrix} = \begin{bmatrix} A & \begin{bmatrix}B_1 & 0\end{bmatrix} & B_2 \\ \begin{bmatrix}C_1\\ 0\end{bmatrix} & 0 & \begin{bmatrix}0\\ I\end{bmatrix} \\ C_2 & \begin{bmatrix}0 & I\end{bmatrix} & 0 \end{bmatrix}\begin{bmatrix}x\\ w\\ u\end{bmatrix} \tag{56}$$

It is assumed that (A, B_2) is controllable and (A, C_2) is observable, which are appropriate to the state-feedback problem. Additionally, we assume that (A, B_1) is controllable and (A, C_2) is observable, appropriate for the estimation problem. Note: **we will also assume that** $x(0) = 0$.

Objective

The objective is to find a stabilising controller, $K(s)$ such that

$$||\mathcal{F}_l(P(s), K(s))||_\infty \leq \gamma \tag{57}$$

Consider $V = x^T X x$ for some $X = X^T$. Then:

$$\begin{aligned}\frac{dV}{dt} + z^T z - \gamma^2 w^T w = {} & x^T(A^T X + XA + C^T C + \frac{1}{\gamma^2} XBB^T X)x \\ & + (u + B_2^T X x)^T (u + B_2^T X x) \\ & - \gamma^2 \left(w - \frac{1}{\gamma^2}\begin{bmatrix}B_1^T\\ 0\end{bmatrix} X x\right)^T \left(w - \frac{1}{\gamma^2}\begin{bmatrix}B_1^T\\ 0\end{bmatrix} X x\right)\end{aligned} \tag{58}$$

Choosing X

Now, if $X = X^T$ is chosen to satisfy:

(a): $A^T X + XA + C^T C + \frac{1}{\gamma^2} XBB^T X = 0$.

(b): Closed loop stability in the absence of disturbance i.e. when $u = -B_2^T X x$ with $u = 0$ meaning that $A - B_2 B_2^T X$ (the closed loop 'A' matrix) is stable.

(c): Closed loop stability in the worst case disturbance i.e. $A - B_2 B_2^T X + \gamma^{-2} B_1 B_1^T X$ stable. This occurs when:

$$\begin{aligned} u &= -B_2^T X x \\ w &= \frac{1}{\gamma^2}\begin{bmatrix}B_1^T\\ 0\end{bmatrix} X x \end{aligned}$$

Note at most one solution to (a) also satisfies (c) and if there exists a stabilising $K(s)$ with the desired $\mathcal{H}_\infty$ norm, then a solution to these equations exists. Choosing this X and integrating yields:

$$||z||_2^2 - \gamma^2 ||w||_2^2 = \Bigg|\Bigg|\underbrace{u + B_2^T X x}_{v}\Bigg|\Bigg|_2^2 - \gamma^2 \Bigg|\Bigg|\underbrace{w - \frac{1}{\gamma^2}\begin{bmatrix}B_1^T\\ 0\end{bmatrix} X x}_{r}\Bigg|\Bigg|_2^2 \tag{59}$$

since $x(\infty) = x(0) = 0$.

State Feedback

If we are able to choose $u = -B_2^T X x$, then we immediately have $||T_{w\to z}||_\infty \leq \gamma$ noting the signs in the above equation.

Output Feedback

However, if we cannot take $u = -B_2^T X x$, note that $||\mathcal{F}(P, K)||_\infty \leq \gamma \Rightarrow ||T_{r\to v}||_\infty \leq \gamma$.

Output Feedback Derivation

Optimal Observer

Consider:

$$\dot{\tilde{x}}_k = \hat{A}^T \tilde{x}_k + F^T \underbrace{(\tilde{y} - B_2 \tilde{x}_k)}_{\tilde{r}_{\text{est}}} - C_2^T C_2 Y \tilde{x}_k \tag{60}$$

For this system, $\tilde{x}_k = \tilde{x} \Rightarrow \dot{\tilde{x}}_k = \dot{\tilde{x}}$. Define $H^T = C_2 Y$. The following optimal K is.

$$\begin{bmatrix}\dot{x_k}\\ u\end{bmatrix} = \begin{bmatrix}\hat{A} - B_2 F - H C_2 & -H \\ F & 0\end{bmatrix}\begin{bmatrix}x_k\\ y\end{bmatrix} \tag{61}$$

5 Convex Optimisation & LMIs for Control Design

Consider the stable linear system, $G(s)$, with state-space realisation:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{62}$$

Stability of the system can be shown by finding $V = x^T X x, X = X^T > 0$ such that $\dot{V} < 0$ for $u = 0$ i.e. by finding a *Lyapunov function* since $V(x(t)) \to 0$ as $t \to \infty$ implies $x(t) \to 0$. This is equivalent to:

$$\dot{V} = x^T(A^T X + XA)x < 0 \Leftrightarrow A^T X + XA < 0 \tag{63}$$

This is a *Linear Matrix Inequality (LMI)*.

Note that:

$$Q \geq 0 \text{ iff } R^T Q R \geq 0 \tag{64}$$

for any invertible matrix R. This then gives:

Schur Complement

$$\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0 \quad \text{iff} \quad R \geq 0 \text{ and } Q - SR^{-1}S^T \geq 0 \tag{65}$$

provided $Q = Q^T$ and $R = R^T$ is invertible. This is easy to show by considering:

Proof of Schur Complement

$$\begin{bmatrix} I & 0 \\ -R^{-1}S^T & I \end{bmatrix}^T \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \begin{bmatrix} I & 0 \\ -R^{-1}S^T & I \end{bmatrix} \tag{66}$$

There exist efficient algorithms to solve LMIs.

5.1 Design of Stabilising Controllers

Consider the system:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ u &= Kx \end{aligned} \tag{67}$$

As before, we seek $X = X^T > 0$ and K such that $V = x^T X x$ and $\dot{V} < 0$. This is equivalent to finding:

First Inequality

$$(A + BK)^T X + X(A + BK) \leq 0 \tag{68}$$

However, this is not a LMI as there are terms corresponding to the product of K and X. To convert this into an LMI, multiply left and right by $Y = X^{-1}$ and write $Z = KY$. We then form the following LMI:

Convert to LMI

$$YA^T + Z^T B^T + AY + BZ \leq 0 \tag{69}$$

After solving for Y and Z, then $X = Y^{-1}$ and $K = ZX$.

5.2 $\mathcal{H}_\infty$ Optimal Control

Consider the system G represented as follows:

Computing the $\mathcal{H}_\infty$ norm

$$\begin{aligned} \dot{x} &= Ax + Bu + B_w w \\ z &= Cx \end{aligned} \tag{70}$$

From the previous section, $||G||_\infty \leq \gamma$ if for some $X = X^T > 0$, $V = x^T X x$ satisfies:

$$\dot{V} + z^T z - \gamma^2 w^T w \leq 0 \tag{71}$$

This can be written as a LMI:

Show this Expression

$$\begin{bmatrix} x \\ w \end{bmatrix}^T \begin{bmatrix} A^T X + XA + C^T C & XB_w \\ B_w^T X & -\gamma^2 I \end{bmatrix} \begin{bmatrix} x \\ w \end{bmatrix} \leq 0 \tag{72}$$

Thus to compute $||G||_\infty$, find $X = X^T$ and $\min \gamma$ such that:

$$\begin{bmatrix} A^T X + XA + C^T C & XB_w \\ B_w^T X & -\gamma^2 I \end{bmatrix} \leq 0 \tag{73}$$

For the system:

Computing the Optimal State-Feedback Controller

$$\begin{aligned} \dot{x} &= Ax + Bu + B_w w \\ z &= Cx + Du \end{aligned} \tag{74}$$

we want to find a controller, $u = Kx$ that minimises $||T_{w\to z}||_\infty$. Applying the same trick as usual yields:

$$\begin{bmatrix} (A+BK)^T X + X(A+BK) + (C+DK)^T(C+DK) & XB_w \\ B_w^T X & -\gamma^2 I \end{bmatrix} \leq 0 \tag{75}$$

However, this is not a LMI. Multiplying left and right by $\begin{bmatrix} Y & 0 \\ 0 & I \end{bmatrix}$ gives:

$$\begin{bmatrix} Y(A+BK)^T + (A+BK)Y + Y(C+DK)^T(C+DK)Y & B_w \\ B_w^T & -\gamma^2 I \end{bmatrix} \leq 0 \tag{76}$$

Then, clearly:

$$\begin{bmatrix} Y(A+BK)^T + (A+BK)Y + Y(C+DK)^T(C+DK)Y & 0 & B_w \\ 0 & -I & 0 \\ B_w^T & 0 & -\gamma^2 I \end{bmatrix} \leq 0 \tag{77}$$

Then, applying the Schur complement:

$$\begin{bmatrix} Y(A+BK)^T + (A+BK)Y & Y(C+DK)^T & B_w \\ (C+DK)Y & -I & 0 \\ B_w^T & 0 & -\gamma^2 I \end{bmatrix} \leq 0 \tag{78}$$

As before, write $Z = KY$ to give:

$$\begin{bmatrix} YA^T + Z^T B^T + AY + BKZ & YC^T + Z^T D^T & B_w \\ CY + DZ & -I & 0 \\ B_w^T & 0 & -\gamma^2 I \end{bmatrix} \leq 0 \tag{79}$$

which is a LMI in Y and Z. Similarly, solve for Z and Y, then use these to calculate $X = Y^{-1}$ and $K = ZY^{-1} = ZX$. Note that we are also minimising over γ.

Predictive Control

6 Introduction to Predictive Control

Predictive Control is a different approach to control; the control input to the plant is the solution to an optimisation problem computed at discrete time-steps. The advantages of predictive control include:

1. Systematic method of handling constraints.
2. Can operate close to constraints.
3. Easy to tune.

All systems have constraints, such as physical constraints (e.g. an actuator limit), a safety constraint or a performance constraint.

Predictive control works as follows. At each sampling instant, a predictive controller:

How does Predictive Control Work

1. Takes a measurement of the system state.
2. Computes a sequence of inputs over a finite time horizon using an internal model to predict states are future times and minimising some cost function of future states and inputs. The controller checks that no constraints are violated on states and inputs.
3. Implements the first part of the optimal sequence.

Note that this is a feedback control law. Each new measurement is used to calculate a new input, and the prediction horizon recedes over time. For example, with a linear model, quadratic costs without constraints, this is exactly a LQR problem, but optimisation occurs directly in the loop in real time. Note that the choice of cost function is more flexible, and that typically the optimisation is constrained.

Compared to Optimal Control

However, there is no guarantees about performance or stability in the long run.

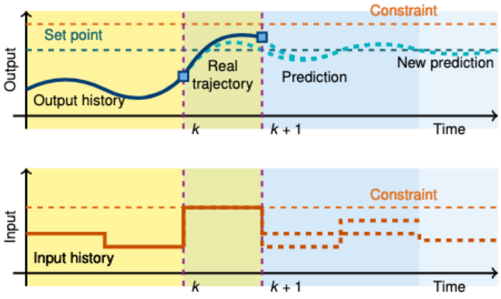


Figure 1: Receding Horizon Principle

7 Unconstrained Predictive Control

Consider a discrete time system:

$$x(k+1) = Ax(k) + Bu(k) \tag{80}$$

Assumptions

It is assumed that we can apply full state feedback (i.e. we have state measurements). The goal is to regulate states around the origin and there are no delays, noise, disturbances, model errors etc.

Task

In this situation, the task is to find the finite horizon input sequence with minimises the finite horizon cost function:

$$V(x, u_0, \dots, u_{N-1}) = \sum_{i=0}^{h-1} \left(x_i^T Q x_i + u_i^T R u_i \right) + x_N^T P x_N \tag{81}$$

$R > 0$ penalises non-zero inputs, $Q \geq 0$ penalises non-zero states. This is precisely the LQR problem, and thus the receding horizon controller with state feedback is:

$$u = -(R + B^T X_1 B)^{-1} B^T X_{k+1} A x_1 \tag{82}$$

where X_1 is found by solving a backwards difference equation.

Alternative Derivation

Recalling the dynamics of the system, note that:

$$x_i = A^i x_0 + A^{i-1} B u_0 + A^{i-2} B u_1 + \dots B u_{i-1} \tag{83}$$

Define the stacked vectors, $\mathbf{u} \in \mathbb{R}^{Nm}$ and $\mathbf{x} \in \mathbb{R}^{Nn}$

$$\mathbf{u} = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \tag{84}$$

Then:

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}}_{\Phi} x_0 + \underbrace{\begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}}_{\Gamma} \underbrace{\begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}}_{\mathbf{u}} \tag{85}$$

i.e. $\mathbf{x} = \Phi x_0 + \Gamma \mathbf{u}$.

The cost function can be written as:

$$V(x, \mathbf{u}) = x_0^T Q x_0 + \mathbf{x}^T \underbrace{\begin{bmatrix} Q & & & \\ & \ddots & & \\ & & Q & \\ & & & P \end{bmatrix}}_{\Omega} \mathbf{x} + \mathbf{u}^T \underbrace{\begin{bmatrix} R & & & \\ & R & & \\ & & \ddots & \\ & & & R \end{bmatrix}}_{\Psi} \mathbf{u} \tag{86}$$

$$= x^T Q x + \mathbf{x}^T \Omega \mathbf{x} + \mathbf{u}^T \Psi \mathbf{u} \tag{87}$$

Note that $\Omega \geq 0$ and $\Psi > 0$ due to the assumptions placed on Q, P and R. Substituting in yields:

$$V(x,\mathbf{u}) = \frac{1}{2}\mathbf{u}^T \underbrace{\left\{2\Psi + 2\Gamma^T\Omega\Gamma\right\}}_{G} \mathbf{u} + \mathbf{u}^T \underbrace{\left\{2\Gamma^T\Omega\Phi\right\}}_{F} \mathbf{x} + x^T(Q + \phi^T\Omega\phi)x \tag{88}$$

which is a quadratic cost, which has the minimiser:

$$\mathbf{u}^*(x) = -G^{-1}Fx \tag{89}$$

The RHC law is the first part of $\mathbf{u}^*(x)$ i.e.

$$u_0^* = \underbrace{-\begin{bmatrix} I_m & 0 & \cdots & 0 \end{bmatrix} G^{-1}F}_{K_{\text{RHC}}}\, x \tag{90}$$

Alternative Formulas

which is a time invariant linear control law. A common alternative formulation is to optimise over predicted input changes with large penalties on rapid control fluctuations.

Stability by Tuning

A simple experiment with fixed $Q = P = I$ and then computes the spectral radius, $\rho(A + BK_{\text{RHC}})$ shows there is no clear pattern in determining stability when changing R and N - some values do not guarantee a stable closed loop system.

Guaranteeing Stability

Let $Q = C^TC$ and assume that (A, C) is detectible. If we choose $P = X$ to be the solution of the discrete time algebraic Ricatti equation:

$$X = Q + A^TXA - A^TXB(R + B^TX_kB)^{-1}B^TXA \tag{91}$$

then:

$$V^*(x) = \min_{\mathbf{u}} \left\{ x_N^T P x_n + \sum_{i=1}^{N-1} \left(x_i^T Q x_i + u_i^T R u_i \right) \right\} = x^T P x \tag{92}$$

for **any choice of** N, effectively giving stability. The controller:

$$u = -(R + B^TXB)^{-1}B^TXAx \tag{93}$$

Why can we choose P?

is guaranteed to be stabilising. We are free to choose the terminal cost since we are using a receding horizon controller over an indefinite timespan, so the terminal cost does not have a real meaning.

Output Feedback

We do not always have access to the state of the system. In this situation, an observer is used to provide estimates of the state, $\hat{x}$, using the output, y. The RHC law is the same with x replaced with it's current estimate.

8 Predictive Control with Constraints

Input Saturation

Many systems have constraints, and predictive control provides an excellent method of accounting for these. Also note that input saturation is a common system non-linearity which can be easily transformed to a constraint on inputs.

On an infinite horizon, finding the optimal set of inputs:

$$\arg\min_{\{u\}} \sum_{i=0}^{\infty} \left(x_i^T Q x_i + u_i^T R u_i \right) \tag{94}$$

whilst guaranteeing the constraints are satisfied for all time is impossible to solve explicitly. Predictive control provides an approximate solution to this problem, but typically RHC laws with constraints are non-linear.

Recall that, over a finite horizon, we wrote the unconstrained LQR problem as:

$$\mathbf{x} = \Phi x_0 + \Gamma\mathbf{u} \tag{95}$$

$$V(x,\mathbf{u}) = \frac{1}{2}\mathbf{u}^T G\mathbf{u} + \mathbf{u}^T FX + x^T(Q + \Phi^T\Omega\Phi)x \tag{96}$$

Typically, we may have a set of linear inequality constraints on the predicted states and inputs:

$$\begin{aligned} M_i x_i + E_i u_i &\leq b_i \qquad \forall\ i = 0, 1, \ldots, N-1 \\ M_n x_N &\leq b_n \end{aligned} \tag{97}$$

These constraints can be written in the following form:

Rewriting in Standard Form

$$\underbrace{\begin{bmatrix} M_0 \\ 0 \\ \cdots \\ 0 \end{bmatrix}}_{\mathcal{D}} x_0 + \underbrace{\begin{bmatrix} 0 & \cdots & 0 \\ M_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & M_n \end{bmatrix}}_{\mathcal{M}} \mathbf{x} + \underbrace{\begin{bmatrix} E_0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & E_{N-1} \\ 0 & \cdots & 0 \end{bmatrix}}_{E} \mathbf{u} \leq \underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \cdots \\ b_N \end{bmatrix}}_{c} \tag{98}$$

i.e. $\mathcal{D}x + \mathcal{M}\mathbf{x} + \mathcal{E}\mathbf{u} \leq c$. Linear constraints on the states are transformed into linear constraints on the inputs, which by substitution via the prediction matrices yields:

Define J and W

$$J\mathbf{u} \leq c + Wx \tag{99}$$

This is now a *Quadratic Programming* problem, with linear constraints and a quadratic cost function (note that the problem is convex). Note that if the quadratic matrix (G) in our case is > 0, then the optimisation problem is strictly convex and a global minimiser can always be found. The global minimiser is also unique.

Quadratic Programming

When the quadratic problem is solved for each timestep, the resulting control law is non-linear. Typically, there may be a number of regions in which the controller is linear.

An alternative predictive control formulation is:

Alternative Formulation

$$\theta = \begin{bmatrix} u_0 \\ x_1 \\ u_1 \\ x_2 \\ \vdots \\ u_{N-1} \\ x_N \end{bmatrix}, \qquad \min_{\theta} \theta^T \begin{bmatrix} R & & & & & & \\ & Q & & & & & \\ & & R & & & & \\ & & & Q & & & \\ & & & & \ddots & & \\ & & & & & R & \\ & & & & & & P \end{bmatrix} \theta \tag{100}$$

subject to:

$$\begin{bmatrix} B & -I & & & & \\ & A & B & -I & & \\ & & A & B & -I & \\ & & & \vdots & \ddots & \vdots \\ & & & A & B & -I \end{bmatrix} \theta = \begin{bmatrix} -Ax(k) \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{101}$$

and

$$\begin{bmatrix} E_0 & & & & & & \\ & M_1 & E_1 & & & & \\ & & & M_2 & E_2 & & \\ & & & & \ddots & & \\ & & & & M_{N-1} & E_{N-1} & \\ & & & & & & M_N \end{bmatrix} \theta \leq \begin{bmatrix} -M_0 x(k) + b \\ b \\ b \\ \vdots \\ b \\ b_N \end{bmatrix} \tag{102}$$

9 Feasibility and Stability in Predictive Control

9.1 Feasibility

Assume that (A,B) is stabilizable and $Q = C^T C$ with (A,C) detective. Let $P \geq 0$ be the unique non-negative solution to the DARE:

$$P = Q + A^T PA - A^T PB(R + B^T PB)^{-1} B^T PA \tag{103}$$

We will consider a solution where we minimise the finite horizon cost function subject to the state transition constraints as well as the input constraints. The key idea is what **we will assume that we use** $u_k = Kx_k$ where $K = -(R+B^T PB)^{-1} B^T PA$ from $k = N$ onwards, and we will add an extra constraint on the terminal position i.e.

Key Idea

$$M_N x_N \leq b_N \tag{104}$$

which ensures that $u = Kx$ is a feasible control policy for all future time steps. The actual input supplied later on should then perform better, ensuring stability.

Invariant Set

Definition 9.1 (Invariant Set) *The set $S\subset\mathbb{R}^n$ is called an invariant set for the system:*

$$x(k+1)=f(x(k)) \tag{105}$$

iff $x(0)\in S$ implies that $f(x(k))\in S\forall\ k\geq 0$

Constraint Admissible Set

Definition 9.2 (Constraint Admissible Set) *Given the control law, $u=\kappa(x)$, a set of states $S\subset\mathbb{R}^n$, and a set of constraints $Z\subset\mathbb{R}^n\times\mathbb{R}^m$, S is constraint admissible iff*

$$(x,\kappa(x))\in Z\ \forall\ x\in S \tag{106}$$

For our problem, $Z=\{(x,u):Mx+Eu\leq b\}$.

Constructing a feasible control law

Given a feedback control law, K, such that $\rho(A+BK)<1$, we choose a matrix M_N and b_N such that:

$$S=\{x\in\mathbb{R}^n : M_N x\leq b_N\} \tag{107}$$

such that S is invariant for the closed loop system:

$$x(k+1)=(A+BK)x(k) \tag{108}$$

and constraint admissible for the control law $u=Kx$ and the constraint set Z i.e. we require that for every $x\in S$:

$$M_N(A+BK)x\leq b_N \tag{109}$$

$$(M+EK)x\leq b \tag{110}$$

To find this set, let $S_0=\{Mx+EKx\leq b\}$ and define $S_n=\{x\in S_0:(A+BK)^k x\in S_0, k=1,\ldots,n\}$. Eventually, $S_{n+1}=S_n$ for some n, which then gives the set S_n as being invariant and constraint admissible. Then choosing $M_N=(M+EK)(A+BK)^n$ and $b_N=b$ yields the desired solution. (?)

9.2 Stability

Consider the discrete time system:

$$x(k+1)=f(x(k))$$

with $f(0)=0$ and f continuous.

Stability

Definition 9.3 (Stability) *The origin is a **stable** equilibrium point if, for any $\epsilon>0$, there exists $\delta>0$ such that if $||x(0)||<\delta$ then $||x(k)||<\epsilon$ for all $k>0$.*

Definition 9.4 (Asymptotic Stability) *The origin is **asymptotically stable** if $||x(k)||\to 0$ as $k\to\infty$.*

Lyapunov Function

Definition 9.5 (Lyapunov Function for Discrete Time Systems) *A continuous function $V:S\to\mathbb{R}$ defined on a region $S\subset\mathbb{R}^n$ containing the origin in its interior is called a **Lyapunov Function** if:*

1. *$V(0)=0$.*
2. *$V(x)>0\ \forall\ x\in S, x\neq 0$.*
3. *$V(f(x)-V(x))\leq 0\ \forall x\in S$.*

If there exists a Lyapunov function such that

$$V(f(x)-V(x))<0\ \forall x\in S,\text{ with } x\neq 0$$

then the origin the origin is an asymptotically stable equilibrium point with the region of attraction S. If S is the whole space and $V(x)\to\infty$ as $||x||\to\infty$ then the system is globally asymptotically stable.

Note that, if we choose $P>0$ such that:

$$(A+BK)^T P(A+BK)-P\leq -Q-K^T RK \tag{111}$$

Then the system is stable. **Question:** *Apparently: ?!*Since $u=Kx$ is optimal from N onwards:

$$V(x)=\min_u\sum_{i=0}^{\infty}x_i^T Qx_i+u_i^T Ru_i \tag{112}$$

subject to the usual constraints. Then,

$$V(Ax+Bu_0^*)=\min_u\sum_{i=1}^{\infty}x_i^T Qx_i+u_i^T Ru_i<V(x) \tag{113}$$

meaning that $V(x)$ is a Lyapunov function, guaranteeing stability.

Reinforcement Learning

10 Dynamic Programming Revisited

Recall in order to minimise the finite horizon cost,

$$J(x_0, u_0, \dots, u_{h-1}) = \sum_{k=1}^{h-1} \underbrace{c(x_k, u_k)}_{\text{stage cost}} + \overbrace{J_h(x_h)}^{\text{terminal cost}} \tag{114}$$

Value Function

we define the value, or cost-to-go, function as:

$$V(x,k) \triangleq \min_{u_k,\dots,u_{h-1}} \left\{ \sum_{i=k}^{h-1} c(x_i, u_i) + J_h(x_k) \right\} \tag{115}$$

We find the value function by solving the *Dynamic Programming Equation*:

$$V(x,k) = \min_u \left\{ c(x,u) + V(f(x,u), k+1) \right\} \tag{116}$$

Application to Infinite Horizon Problems

with the final condition, $V(x,h) = J_h(x)$. This can be applied to infinite horizon problems, for which the cost function is defined as:

$$J(x_0) = \sum_{k=0}^{\infty} \lambda^k c(x_k, u_k) \tag{117}$$

Bellman Optimality: Infinite Horizon

where $\lambda \leq 1$ is known as the discount factor. The *Bellman Optimality* condition becomes:

$$V(x) = \min_u \{ c(x,u) + \lambda V(f(x,u)) \} \tag{118}$$

Episodic Problems

Episodic Problems are problems with $\lambda = 1$ which are finite horizon problems where the finishing time is not specified by rather there exists a stopping set, X_s, such that there exists a u such that $f(x,u) \in X_s$ and $c(x,u) = 0$ thus guaranteeing a finite cost.

Value Iteration

Value iteration uses the Bellman Optimality equation to form an update rule:

$$V_{k+1}(x) = \min_u \{ c(x,u) + \lambda V_k(f(x,u)) \} \tag{119}$$

It can be shown that this is guaranteed to converge for any initial guess, $V_0(x)$. This single equation effectively combines a step evaluating the value of a policy, and then applying a greedy update.

Policy Definition

Consider a policy, $\pi(x)$, such that $u = \pi(x)$. The value function of that policy is:

$$V^{\pi}(x) = c(x, \pi(x)) + \lambda V^{\pi}(f(x, \pi(x))) \tag{120}$$

Policy Evaluation

The following iterative scheme can be used to evaluate the value function of the policy:

$$V^{\pi}_{k+1}(x) = c(x, \pi(x)) + \lambda V^{\pi}_k(f(x, \pi(x))) \tag{121}$$

The Bellman equation means that $V^{\pi}(x)$ is a fixed point for this update rule.

Policy Iteration

Policy iteration repeatedly evaluates a policy and applies a greedy update to the policy as follows:

1. Initialise policy, $\pi(x)$.
2. Compute value of this policy, $V^{\pi}(x)$.
3. Update π to be the greedy policy, assuming that after the next step policy π will be followed:

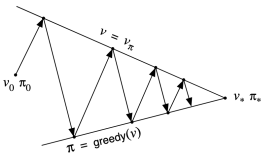
   $$\pi(s) \leftarrow \arg\min_u c(x,u) + \lambda V^{\pi}(f(x,u)) \tag{122}$$

   The intuition behind is if that it is better to choose input u in state x and then policy $\pi(x)$ thereafter, it should be better to choose input u every time we are in state x.

Generalised Policy Iteration

Policy iteration consists of two simultaneous processes which are run to convergence after each iteration. However this is not strictly speaking necessary; in value iteration, we only apply one sweep of policy evaluation before improving the policy function again. In *Generalised Policy Iteration*, we allow the policy-evaluation and policy-improvement processes to interact. Note that if both processes stabilise (i.e. no longer produce changes), then the value function and policy must be optimal: the value function stabilises only when it is consistent with the current policy, and the policy stabilises only when it is greedy with respect to the current value function. Thus both process stabilise when a policy has been found which is greedy with respect to its evaluation function, implying that the the Bellman optimality equation holds.

Note that making the policy greedy with respect to the value function typically makes the value function incorrect for the changed policy and making the value function consistent with the policy typically means that policy is no longer greedy.



Policy Iteration

11 Learning from Samples

Action-Value Function

Definition 11.1 (Action-Value Function) *The action-value function, $Q(x,u)$, is defined as:*

$$Q(x,u) = c(x,u) + \lambda V(f(x,u)) \tag{123}$$

i.e. the cost of applying input u in the current state and applying the optimal input thereafter.

Note that:

$$V(x) = \min_u Q(x,u) \tag{124}$$

Q-function Recursion

and the optimal input is the minimiser of the above. This gives:

$$Q(x,u) = c(x,u) + \lambda \min_v Q(f(x,u),v) \tag{125}$$

Q-Learning

Given a sample, (x_i, u_i, c_i, x_{i+1}), the Q-function can be updated as:

$$Q_{k+1}(x_i,u_i) = c_i + \lambda \min_u Q_k(x_{i+1},u) \tag{126}$$

with $Q_{k+1}(x,u) = Q_k(x,u)$ for all other x,u. Provided each state and input is visited infinitely often, then $Q_k \to Q$ as k increases. If the dimension of the state-space and the number of inputs is small, then this recursion can be solved by tabulation.

Application to Stochastic Problems

For stochastic problems where x_{i+1} is not deterministically given, the update rule is modified:

$$Q_{k+1}(x_i,u_i) = Q_k(x_i,u_i) + \alpha\Big[c_i + \lambda \min_u Q_k(x_{i+1},u) - Q_k(x_i,u_i)\Big] \tag{127}$$

α is known as the learning rate, and it must be decreased gradually to zero (in a complex manner) to guarantee convergence.

Functional Approximation

In many possible problems, it is not possible to discretise over states. Typically, a *functional approximation*, $Q_\theta(x,u)$ may be used in order to approximate the action-value function. For instance, deep neural networks are often used for this task which typically, given the state, return the Q-function for every possible (discretised) input.

Optimisation Problem

Denote an experience of the agent as $e_t = (x_t, u_t, c_t, x_{t+1})$. A *replay buffer* collects the agent's experiences, $\mathcal{D}_t = \{e_0, \dots, e_t\}$. We seek to minimise:

$$\begin{aligned}
\text{minimise } L(\theta) &= \mathbb{E}[(y_t - Q_\theta(x_t,u_t))^2] \\
&\simeq \frac{1}{N}\sum_{k=1}^{N}(y_k - Q_\theta(x_k,u_k))^2 && (128) \\
\text{where } y_t &= c_t + \lambda \min_u Q_\theta(x_{t+1},u) && (129)
\end{aligned}$$

We fix the target values, optimise over θ, and then re-evaluate the target values. Note that the sum is typically taken over a mini-batch of N experiences, randomly sampled from the replay buffer. For a neural network, gradients can be found by using back propagation. Q-learning is an *off-policy* method where the experiences need not come from the current policy.

ϵ-greedy

Sometimes, an ϵ-greedy policy may be used:

$$u_t = \begin{cases} \arg\min_u Q(x_t, y) & \text{with probability } 1-\epsilon \\ \text{random exploratory action} & \text{with probability } \epsilon \end{cases} \tag{130}$$

This attempts to resolve the exploration-exploitation trade-off. Note that in certain situations,a target neural network may also be used to calculate y_k which tracks Q_θ.

Continuous Input Spaces

For continuous input spaces, which are very common in control problems, it is possible to discretise over inputs but this is not usually feasible. u could be an input to a neural network, but this means that it is very difficult to find the minimiser of $Q(x,u)$, which unfortunately is required very often.

Actor-Critic Methods

A solution to this problem is to add a second neural network, Π_w, which represents the policy ('actor'). The Q-network is the 'critic' and is used to evaluate the policy. Then, the optimisation problem becomes:

$$\begin{aligned}
\text{minimise } L_Q(\theta) &= \mathbb{E}[(y_t - Q_\theta(x_t, u_t))^2] \\
&\simeq \frac{1}{N}\sum_{k=1}^{N}(y_k - Q_\theta(x_k, u_k))^2 \\
\text{and minimise } L_\Pi(w) &= \mathbb{E}[Q_\theta(x_t, \Pi_w(x_t))] \\
&\simeq \frac{1}{N}\sum_{k=1}^{N} Q_\theta(x_k, \Pi_w(x_t)) \qquad (131) \\
\text{where } y_t &= c_t + \lambda Q_\theta(x_{t+1}, \Pi_w(x_{t+1}))
\end{aligned}$$

The algorithm alternates between updating θ to reduce $L_Q(\theta)$ which updates the value function for the policy and updating w to make the policy closer to optimal. Stochastic gradient descent is one technique for achieving this. Note that when calculating targets, it is assumed that the policy is optimal.