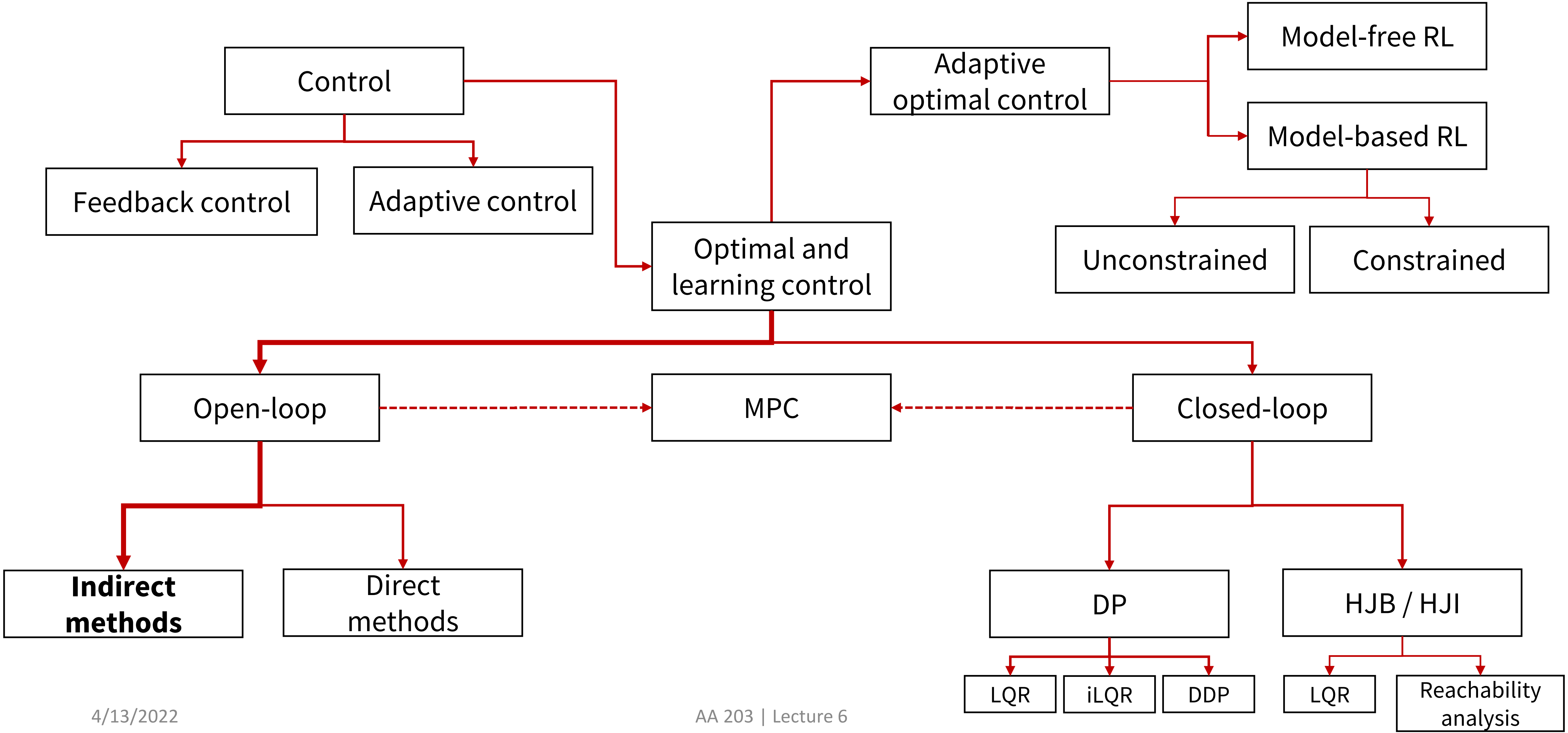
AA203
Optimal and Learning-based Control

Pontryagin's Minimum Principle (PMP);
Dynamic Programming

Stanford University

Autonomous Systems Lab

Roadmap



Control
Feedback control
Adaptive control
Optimal and learning control
Adaptive optimal control
Model-free RL
Model-based RL
Unconstrained
Constrained
Open-loop
MPC
Closed-loop
Indirect methods
Direct methods
DP
HJB / HJI
LQR
iLQR
DDP
LQR
Reachability analysis

Necessary conditions for optimal control
(with unbounded controls)

- The problem is to find an *admissible control* $\mathbf{u}^*$ which causes the system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$$

to follow an *admissible trajectory* $\mathbf{x}^*$ that minimizes the *functional*

$$J(\mathbf{u}) = h\big(\mathbf{x}(t_f), t_f\big) + \int_{t_0}^{t_f} g\,(\mathbf{x}(t), \mathbf{u}(t), t)\, dt$$

- Assumptions: $h \in C^2$, state and control regions are unbounded, t_0 and $\mathbf{x}(0)$ are fixed

Necessary conditions for optimal control
(with unbounded controls)

- Define the Hamiltonian

$$H(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}(t), t) \coloneqq g(\mathbf{x}(t), \mathbf{u}(t), t) + \mathbf{p}(t)^T \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)$$

- The necessary conditions for optimality (proof to follow) are

$$\left.\begin{aligned} \dot{\mathbf{x}}^*(t) &= \frac{\partial H}{\partial \mathbf{p}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t) \\ \dot{\mathbf{p}}^*(t) &= -\frac{\partial H}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t) \\ \mathbf{0} &= \frac{\partial H}{\partial \mathbf{u}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t) \end{aligned}\right\} \text{ for all } t \in [t_0, t_f]$$

with boundary conditions

$$\left[\frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^*(t_f), t_f) - \mathbf{p}^*(t_f)\right]^T \delta\mathbf{x}_f + \left[H(\mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f)\right]\delta t_f = 0$$

Necessary conditions for optimal control
(with bounded controls)

- So far, we have assumed that the admissible controls and states are not constrained by any boundaries
- However, in realistic systems, such constraints do commonly occur
    - control constraints often occur due to actuation limits
    - state constraints often occur due to safety considerations
- We will now consider the case with control constraints, which will lead to the statement of the Pontryagin's minimum principle

Why do control constraints complicate the analysis?

- By definition, the control $\mathbf{u}^*$ causes the functional J to have a relative minimum if

$$J(\mathbf{u}) - J(\mathbf{u}^*) = \Delta J \geq 0$$

for all admissible controls "close" to $\mathbf{u}^*$

- If we let $\mathbf{u} = \mathbf{u}^* + \delta\mathbf{u}$, the increment in J can be expressed as

$$\Delta J(\mathbf{u}^*, \delta\mathbf{u}) = \delta J(\mathbf{u}^*, \delta\mathbf{u}) + \text{higher order terms}$$

- The variation $\delta\mathbf{u}$ is arbitrary *only if* the extremal control is strictly within the boundary for all time in the interval $[t_0, t_f]$
- In general, however, an extremal control lies on a boundary during at least one subinterval of the interval $[t_0, t_f]$

Why do control constraints complicate the analysis?

- As a consequence, admissible control variations $\delta\mathbf{u}$ exist whose negatives $(-\delta\mathbf{u})$ are not admissible
- This implies that a necessary condition *for* $\mathbf{u}^*$ to minimize J is

$$\delta J(\mathbf{u}^*, \delta\mathbf{u}) \geq 0$$

for all admissible variations with $||\delta\mathbf{u}||$ small enough

Pontryagin's minimum principle

- Assuming bounded controls $\mathbf{u} \in U$, the necessary optimality conditions are (H is the Hamiltonian)

$$\dot{\mathbf{x}}^*(t) = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

$$\dot{\mathbf{p}}^*(t) = -\frac{\partial H}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

$$H(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t) \leq H(\mathbf{x}^*(t), \mathbf{u}(t), \mathbf{p}^*(t), t), \text{ for all } \mathbf{u}(t) \in U$$

for all $t \in [t_0, t_f]$

along with the boundary conditions:

$$\left[\frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^*(t_f), t_f) - \mathbf{p}^*(t_f)\right]^T \delta \mathbf{x}_f + \left[H(\mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f)\right] \delta t_f = 0$$

Pontryagin's minimum principle

- $\mathbf{u}^*(t)$ is a control that causes $H(\mathbf{x}^*(t), \mathbf{u}(t), \mathbf{p}^*(t), t)$ to assume its *global* minimum
- Harder condition in general to analyze
- Example: consider the system having dynamics:

$$\dot{x}_1(t) = x_2(t), \qquad \dot{x}_2(t) = -x_2(t) + u(t);$$

it is desired to minimize the functional

$$J = \int_{t_0}^{t_f} \frac{1}{2}[x_1^2(t) + u^2(t)]dt$$

subject to the control constraint $|u(t)| \leq 1$
with t_f fixed and the final state free.

Pontryagin's minimum principle

Solution:

- If the control is unconstrained,

$$u^*(t) = -p_2^*(t)$$

- If the control is constrained as $|u(t)| \leq 1$, then

$$u^*(t) = \begin{cases} -1 & \text{for } 1 < p_2^*(t) \\ -p_2^*(t), & -1 \leq p_2^*(t) \leq 1 \\ +1 & \text{for } p_2^*(t) < -1 \end{cases}$$

- To determine $u^*(t)$ explicitly, the state and co-state equations must still be solved

Additional necessary conditions

1. If the final time is fixed and the Hamiltonian does not depend explicitly on time, then

$$H\left(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t)\right) = c \quad \text{for all } t \in \left[t_0, t_f\right]$$

2. If the final time is free and the Hamiltonian does not depend explicitly on time, then

$$H\left(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t)\right) = 0 \quad \text{for all } t \in [t_0, t_f]$$

Minimum time problems

- Find the control input sequence

$$M_i^- \leq u_i(t) \leq M_i^+ \text{ for } i = 1, \dots, m$$

that drives the control affine system

$$\dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}, t) + B(\mathbf{x}, t)\mathbf{u}(t)$$

from an arbitrary state $\mathbf{x}_0$ to the origin, and minimizes time

$$J = \int_{t_0}^{t_f} dt$$

Minimum time problems

- Form the Hamiltonian

$$H = 1 + \mathbf{p}(t)^T\{\mathbf{a}(\mathbf{x},t) + B(\mathbf{x},t)\mathbf{u}(t)\}$$

$$= 1 + \mathbf{p}(t)^T\{\mathbf{a}(\mathbf{x},t) + [\mathbf{b}_1(\mathbf{x},t)\ \ \mathbf{b}_2(\mathbf{x},t) \cdots \mathbf{b}_m(\mathbf{x},t)]\mathbf{u}(t)\}$$

$$= 1 + \mathbf{p}(t)^T\mathbf{a}(\mathbf{x},t) + \sum_{i=1}^{m} \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i(t)$$

- By the PMP, select $u_i(t)$ to minimize H, which gives

$$u_i^*(t) = \begin{cases} M_i^+ \text{ if } \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t) < 0 \\ M_i^- \text{ if } \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t) > 0 \end{cases}$$

"Bang-bang" control

Minimum time problems

- Note: we showed what to do when $\mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x}, t) \neq 0$
- Not obvious what to do if $\mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x}, t) = 0$
- If $\mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x}, t) = 0$ for some finite time interval, then the coefficient of $u_i(t)$ in the Hamiltonian is zero, so the PMP provides no information on how to select $u_i(t)$
- The treatment of such a *singular condition* requires a more sophisticated analysis
- The analysis in the linear case is significantly easier, see Kirk Sec. 5.4

Minimum fuel problems

- Find the control input sequence

$$M_i^- \leq u_i(t) \leq M_i^+ \text{ for } i = 1, \ldots, m$$

that drives the control affine system

$$\dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}, t) + B(\mathbf{x}, t)\mathbf{u}(t)$$

from an arbitrary state $\mathbf{x}_0$ to the origin in a fixed time, and minimizes

$$J = \int_{t_0}^{t_f} \sum_{i=1}^{m} c_i \, |u_i(t)| \, dt$$

Minimum fuel problems

- Form the Hamiltonian

$$H = \sum_{i=1}^{m} c_i\,|u_i(t)| + \mathbf{p}(t)^T\{\mathbf{a}(\mathbf{x},t) + B(\mathbf{x},t)\mathbf{u}(t)\}$$

$$= \sum_{i=1}^{m} c_i\,|u_i(t)| + \mathbf{p}(t)^T\mathbf{a}(\mathbf{x},t) + \sum_{i=1}^{m}\mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i(t)$$

$$= \sum_{i=1}^{m}[c_i\,|u_i(t)| + \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i(t)] + \mathbf{p}(t)^T\mathbf{a}(\mathbf{x},t)$$

- By the PMP, select $u_i(t)$ to minimize H, that is

$$\sum_{i=1}^{m}[c_i\,|u_i^*(t)| + \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i^*(t)] \leq \sum_{i=1}^{m}[c_i\,|u_i(t)| + \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i(t)]$$

Minimum fuel problems

- Since the components of $\mathbf{u}(t)$ are independent, then one can just look at

$$c_i\,|u_i^*(t)| + \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i^*(t) \le c_i\,|u_i(t)| + \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i(t)$$

- The resulting control law is

$$u_i^*(t) = \begin{cases} M_i^- & \text{if } c_i < \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t) \\ 0 & \text{if } -c_i < \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t) < c_i \\ M_i^+ & \text{if } \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t) < -c_i \end{cases}$$

"Bang-off-bang" control

Minimum energy problems

- Find the control input sequence

  $M_i^- \leq u_i(t) \leq M_i^+$ for $i = 1, \dots, m$

  that drives the control affine system

  $$\dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}, t) + B(\mathbf{x}, t)\mathbf{u}(t)$$

  from an arbitrary state $\mathbf{x}_0$ to the origin in a fixed time, and minimizes

  $$J = \frac{1}{2}\int_{t_0}^{t_f} \mathbf{u}(t)^T R \mathbf{u}(t) dt,$$

  where $R > 0$ and diagonal

Minimum energy problems

- Form the Hamiltonian

$$H = \frac{1}{2}\mathbf{u}(t)^T R\mathbf{u}(t) + \mathbf{p}(t)^T\{\mathbf{a}(\mathbf{x},t) + B(\mathbf{x},t)\mathbf{u}(t)\}$$

$$= \frac{1}{2}\mathbf{u}(t)^T R\mathbf{u}(t) + \mathbf{p}(t)^T B(\mathbf{x},t)\mathbf{u}(t) + \mathbf{p}(t)^T\mathbf{a}(\mathbf{x},t)$$

- By the PMP, we need to solve

$$\mathbf{u}^*(t) = \arg\min_{\mathbf{u}(t)\in U}\left[\sum_{i=1}^{m}\frac{1}{2}R_{ii}u_i(t)^2 + \mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x},t)u_i(t)\right]$$

Minimum energy problems

- As in the first example today, in the unconstrained case, the optimal solution for each component of $\mathbf{u}(t)$ would be

$$\hat{u}_i(t) = -R_{ii}^{-1}\,\mathbf{p}(t)^T\mathbf{b}_i(\mathbf{x}, t)$$

- Considering the input constraints, the resulting control law is

$$u^*(t) = \begin{cases} M_i^- & \text{if} \quad \hat{u}_i(t) < M_i^- \\ \hat{u}_i(t) & \text{if} \quad M_i^- < \hat{u}_i(t) < M_i^+ \\ M_i^+ & \text{if} \quad M_i^+ < \hat{u}_i(t) \end{cases}$$

"Saturating" control

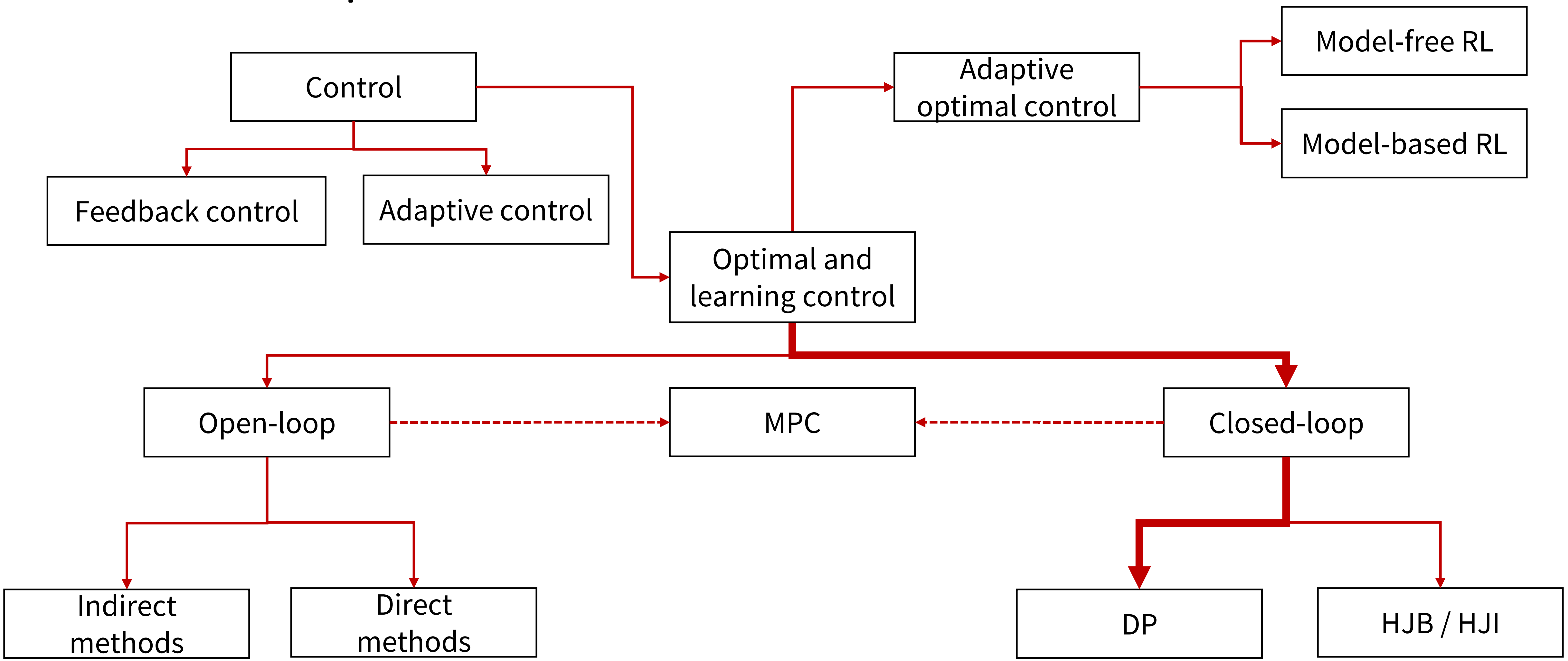
Uniqueness and existence

- Note: uniqueness and existence are not in general guaranteed!

- Example 1 (non uniqueness): find a control sequence $u(t)$ to transfer the system $\dot{x}(t) = u(t)$ from an arbitrary initial state x_0 to the origin, and such that the functional $J = \int_0^{t_f} |u(t)| dt$ is minimized. The final time is free, and the admissible controls are $|u(t)| \leq 1$

- Example 2 (non existence): find a control sequence $u(t)$ to transfer the system $\dot{x}(t) = x(t) + \ u(t)$ from an arbitrary initial state x_0 to the origin, and such that the functional $J = \int_{t_0}^{t_f} |u(t)| dt$ is minimized. The final time is free, and the admissible controls are $|u(t)| \leq 1$

Roadmap

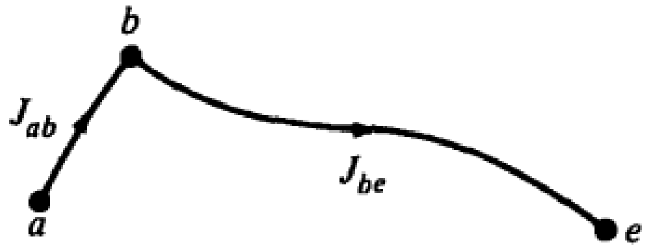


Control
Feedback control
Adaptive control
Optimal and learning control
Adaptive optimal control
Model-free RL
Model-based RL
Open-loop
MPC
Closed-loop
Indirect methods
Direct methods
DP
HJB / HJI

Principle of optimality

The key concept behind the dynamic programming approach is the principle of optimality

Suppose optimal path for a multi-stage decision-making problem with additive cost structure is



- first decision yields segment $a - b$ with cost J_{ab}
- remaining decisions yield segments $b - e$ with cost J_{be}
- optimal cost is then $J_{ae}^* = J_{ab} + J_{be}$

Principle of optimality

- Claim: If $a - b - e$ is optimal path from a to e, then $b - e$ is optimal path from b to e

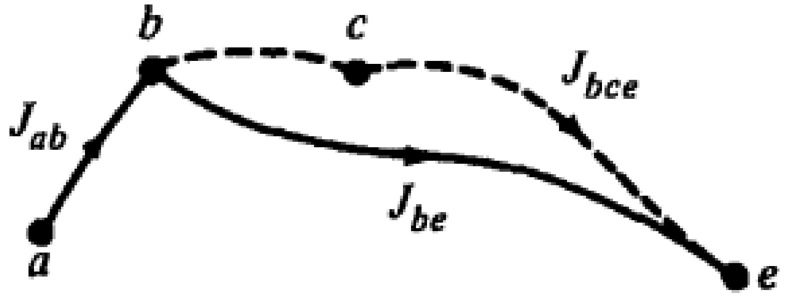
Principle of optimality

- Claim: If $a - b - e$ is optimal path from a to e, then $b - e$ is optimal path from b to e
- *Proof*: Suppose $b - c - e$ is the optimal path from b to e. Then

$$J_{bce} < J_{be}$$

and

$$J_{ab} + J_{bce} < J_{ab} + J_{be} = J_{ae}^*$$



Contradiction!

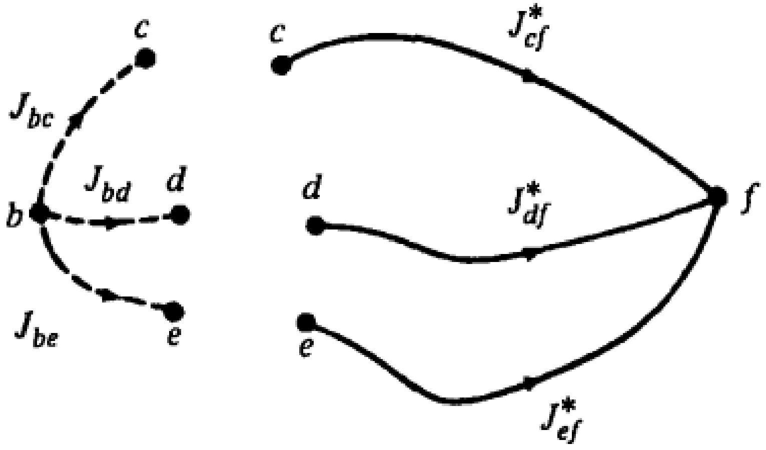
Principle of optimality

Principle of optimality (for discrete-time systems):
Let $\pi^* := \{\pi_0^*, \pi_1^*, \ldots, \pi_{N-1}^*\}$ be an optimal policy. Assume state $\mathbf{x}_k$ is reachable. Consider the subproblem whereby we are at $\mathbf{x}_k$ at time k and we wish to minimize the cost-to-go from time k to time N. Then the truncated policy $\{\pi_k^*, \pi_{k+1}^*, \ldots, \pi_{N-1}^*\}$ is optimal for the subproblem.

- tail policies optimal for tail subproblems
- notation: $\pi_k^*(\mathbf{x}_k) = \pi^*(\mathbf{x}_k, k)$

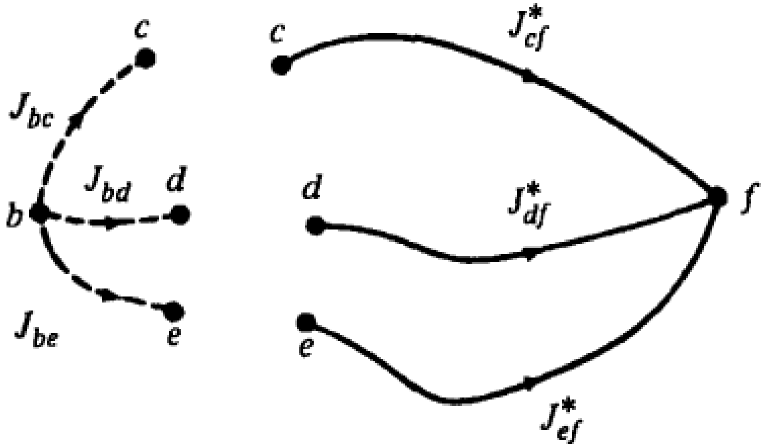
Applying the principle of optimality

Principle of optimality: if $b - c$ is the initial segment of the optimal path from b to f, then $c - f$ is the terminal segment of this path



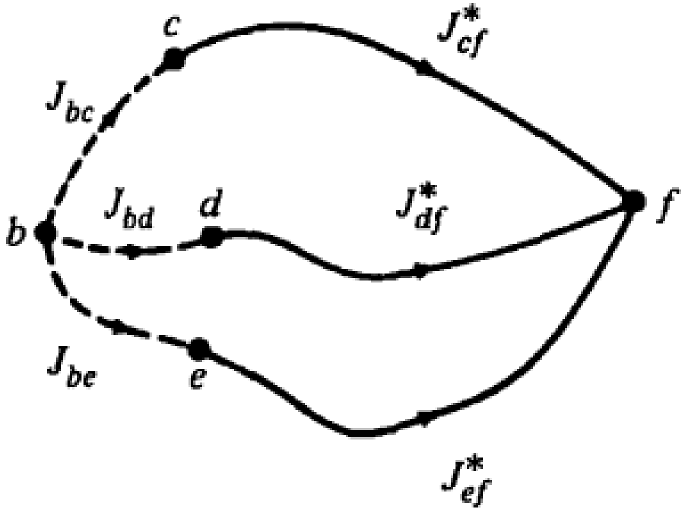
Applying the principle of optimality

Principle of optimality: if $b-c$ is the initial segment of the optimal path from b to f, then $c-f$ is the terminal segment of this path



Hence, the optimal trajectory is found by comparing:

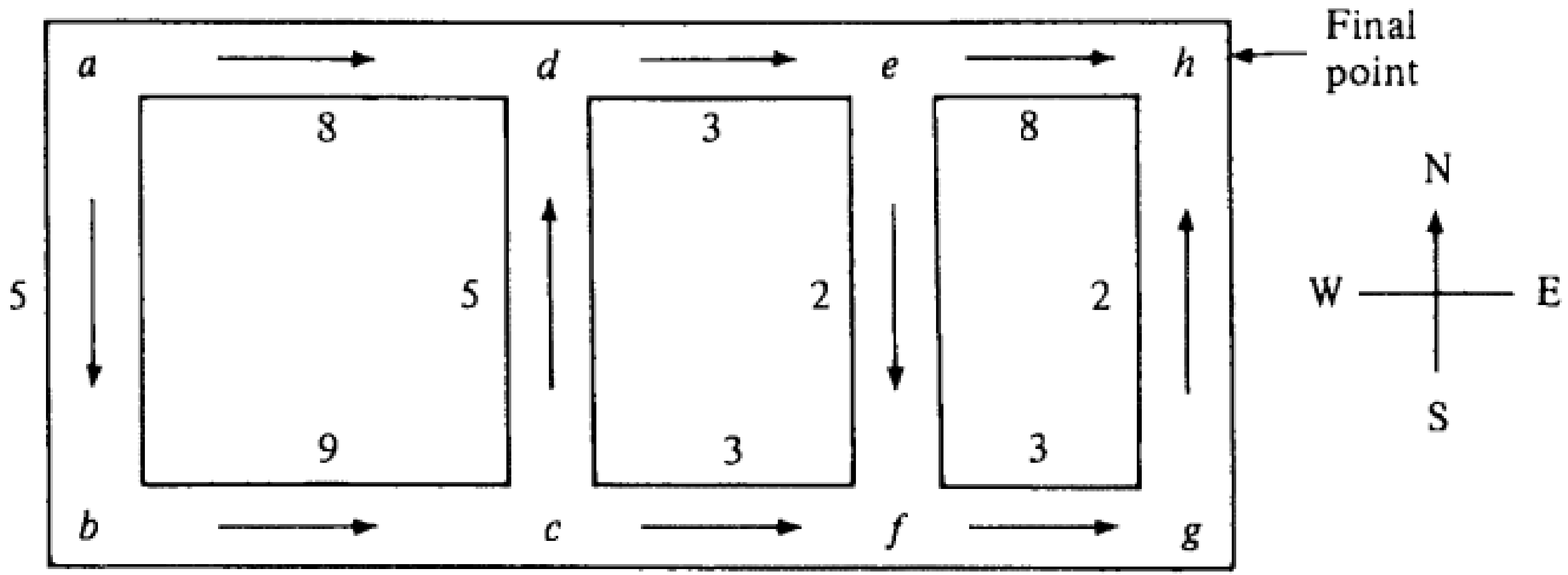
$$C_{bcf} = J_{bc} + J^*_{cf}$$
$$C_{bdf} = J_{bd} + J^*_{df}$$
$$C_{bef} = J_{be} + J^*_{ef}$$



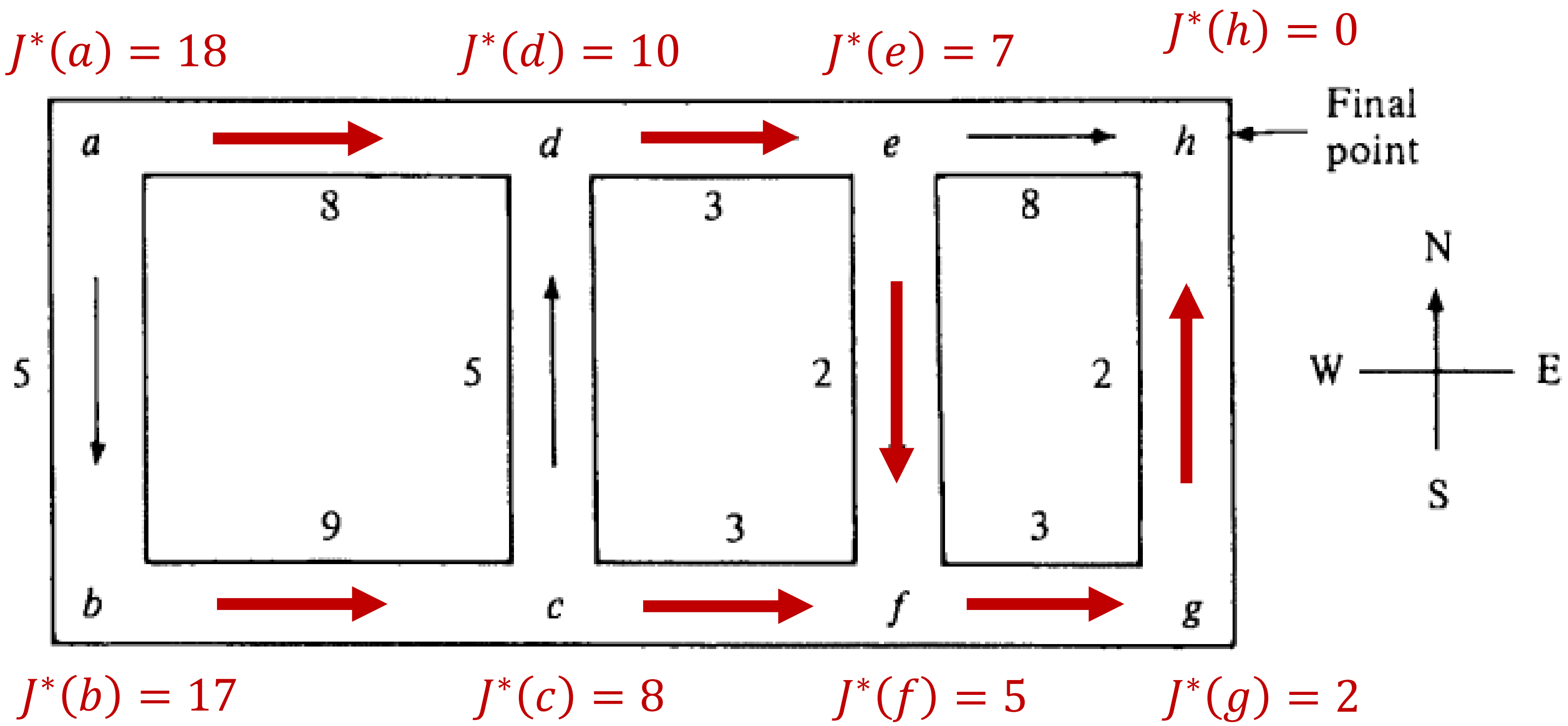
Applying the principle of optimality

- Need only to compare the concatenations of immediate decisions and optimal decisions → significant decrease in computation/possibilities
- In practice: carry out this procedure backward in time

Example



Example



Optimal cost: 18; Optimal path: $a \to d \to e \to f \to g \to h$

DP Algorithm

- Model: $\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, k), \quad \mathbf{u}_k \in U(\mathbf{x}_k)$
- Cost: $J(\mathbf{x}_0) = h_N(\mathbf{x}_N) + \sum_{k=0}^{N-1} g(\mathbf{x}_k, \pi_k(\mathbf{x}_k), k)$

DP Algorithm: For every initial state $\mathbf{x}_0$, the optimal cost $J^*(\mathbf{x}_0)$ is equal to $J_0^*(\mathbf{x}_0)$, given by the last step of the following algorithm, which proceeds backward in time from stage $N-1$ to stage 0:

$$J_N^*(\mathbf{x}_N) = h_N(\mathbf{x}_N)$$

$$J_k^*(\mathbf{x}_k) = \min_{\mathbf{u}_k \in U(\mathbf{x}_k)} g(\mathbf{x}_k, \mathbf{u}_k, k) + J_{k+1}^*\big(f(\mathbf{x}_k, \mathbf{u}_k, k)\big), \qquad k = 0, \dots, N-1$$

Furthermore, if $\mathbf{u}_k^* = \pi_k^*(\mathbf{x}_k)$ minimizes the right hand side of the above equation for each $\mathbf{x}_k$ and k, the policy $\{\pi_0^*, \pi_1^*, \dots, \pi_{N-1}^*\}$ is optimal

Comments

- discretization (from differential equations to difference equations)
- quantization (from continuous to discrete state variables / controls)
- global minimum
- constraints, in general, simplify the numerical procedure
- optimal control in closed-loop form
- curse of dimensionality

Discrete LQR

- Canonical application of dynamic programming for control
- One case where DP can be solved analytically (in general, DP algorithm must be performed numerically)

Discrete LQR: select control inputs to minimize

$$J_0(\mathbf{x}_0) = \frac{1}{2}\mathbf{x}_N^T Q_N \mathbf{x}_N + \frac{1}{2}\sum_{k=0}^{N-1}\left(\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + 2\mathbf{x}_k^T S_k \mathbf{u}_k\right)$$

subject to the dynamics

$$\mathbf{x}_{k+1} = A_k \mathbf{x}_k + B_k \mathbf{u}_k, \qquad k \in \{0, 1, \ldots, N-1\}$$

assuming

$$Q_k = Q_k^T \succeq 0, \quad R_k = R_k^T \succ 0, \quad \begin{bmatrix} Q_k & S_k \\ S_k^T & R_k \end{bmatrix} \succeq 0 \quad \forall k$$

Discrete LQR

Many important extensions, some of which we'll cover later in this class

- Tracking LQR: $\mathbf{x}_k, \mathbf{u}_k$ represent small deviations ("errors") from a nominal trajectory (possibly with nonlinear dynamics)
- Cost with linear terms, affine dynamics: can consider today's analysis with augmented dynamics

$$\mathbf{y}_{k+1} = \begin{bmatrix} \mathbf{x}_{k+1} \\ 1 \end{bmatrix} = \begin{bmatrix} A_k & c_k \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x}_k \\ 1 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \mathbf{u}_k = \tilde{A}\mathbf{y}_k + \tilde{B}\mathbf{u}_k$$

Discrete LQR – brute force

Rewrite the minimization of

$$J_0(\mathbf{x}_0) = \frac{1}{2}\mathbf{x}_N^T Q_N \mathbf{x}_N + \frac{1}{2}\sum_{k=0}^{N-1}\left(\mathbf{x}_k^T Q_k \mathbf{x}_k + \mathbf{u}_k^T R_k \mathbf{u}_k + 2\mathbf{x}_k^T S_k \mathbf{u}_k\right)$$

subject to dynamics

$$\mathbf{x}_{k+1} = A_k \mathbf{x}_k + B_k \mathbf{u}_k, \qquad k \in \{0, 1, \ldots, N-1\}$$

as…

Discrete LQR – brute force

$$\min_{\mathbf{x}_k,\mathbf{u}_k} \quad \frac{1}{2}\begin{bmatrix}\mathbf{x}_0\\ \mathbf{u}_0\\ \mathbf{x}_1\\ \mathbf{u}_1\\ \vdots\\ \mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\\ \mathbf{x}_N\end{bmatrix}^T \begin{bmatrix} Q_0 & S_0 & & & & & & \\ S_0^T & R_0 & & & & & & \\ & & Q_1 & S_1 & & & & \\ & & S_1^T & R_1 & & & & \\ & & & & \ddots & & & \\ & & & & & Q_{N-1} & S_{N-1} & \\ & & & & & S_{N-1}^T & R_{N-1} & \\ & & & & & & & Q_N \end{bmatrix} \begin{bmatrix}\mathbf{x}_0\\ \mathbf{u}_0\\ \mathbf{x}_1\\ \mathbf{u}_1\\ \vdots\\ \mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\\ \mathbf{x}_N\end{bmatrix}$$

$$\text{s.t.} \quad \begin{bmatrix} -I & & & & & & \\ A_0 & B_0 & -I & & & & \\ & & A_1 & B_1 & -I & & \\ & & & & \ddots & & \\ & & & & & A_{N-1} & B_{N-1} & -I \end{bmatrix} \begin{bmatrix}\mathbf{x}_0\\ \mathbf{u}_0\\ \mathbf{x}_1\\ \mathbf{u}_1\\ \mathbf{x}_2\\ \vdots\\ \mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\\ \mathbf{x}_N\end{bmatrix} + \begin{bmatrix}\mathbf{x}_0\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix} = \mathbf{0}$$

Discrete LQR – brute force

Defining suitable notation, this is

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{1}{2}\mathbf{z}^T W \mathbf{z} \\ \text{s.t.} \quad & C\mathbf{z} + \mathbf{d} = \mathbf{0} \end{aligned}$$

with solution from applying NOC (also SOC in this case, due to problem convexity):

$$\begin{bmatrix} \mathbf{z}^* \\ \boldsymbol{\lambda}^* \end{bmatrix} = \begin{bmatrix} W & C^T \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ -\mathbf{d} \end{bmatrix}$$

Discrete LQR – dynamic programming

First step:

$$J_N^*(\mathbf{x}_N) = \frac{1}{2}x_N^T Q_N x_N = \frac{1}{2}x_N^T P_N x_N$$

Going backward:

$$J_{N-1}^*(\mathbf{x}_{N-1}) = \min_{\mathbf{u}_{N-1}} \frac{1}{2}\left(\begin{bmatrix}\mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\end{bmatrix}^T \begin{bmatrix}Q_{N-1} & S_{N-1}\\ S_{N-1}^T & R_{N-1}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\end{bmatrix} + \mathbf{x}_N^T P_N \mathbf{x}_N\right)$$

$$= \min_{\mathbf{u}_{N-1}} \frac{1}{2}\left(\begin{bmatrix}\mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\end{bmatrix}^T \begin{bmatrix}Q_{N-1} & S_{N-1}\\ S_{N-1}^T & R_{N-1}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{N-1}\\ \mathbf{u}_{N-1}\end{bmatrix} + \right.$$

$$\left.(A_{N-1}\mathbf{x}_{N-1} + B_{N-1}\mathbf{u}_{N-1})^T P_N (A_{N-1}\mathbf{x}_{N-1} + B_{N-1}\mathbf{u}_{N-1})\right)$$

Discrete LQR – dynamic programming

Unconstrained NOC:

$$\begin{aligned}\nabla_{u_{N-1}} J_{N-1}(\mathbf{x}_{N-1}) = R_{N-1}\mathbf{u}_{N-1} + S_{N-1}^T \mathbf{x}_{N-1} + \\ B_{N-1}^T P_N (A_{N-1}\mathbf{x}_{N-1} + B_{N-1}\mathbf{u}_{N-1}) = \mathbf{0} \\ \implies \mathbf{u}_{N-1}^* = -(R_{N-1} + B_{N-1}^T P_N B_{N-1})^{-1}(B_{N-1}^T P_N A_{N-1} + S_{N-1}^T)\mathbf{x}_{N-1} \\ := F_{N-1} x_{N-1}\end{aligned}$$

Note also that:

$$\nabla_{u_{N-1}}^2 J_{N-1}(\mathbf{x}_{N-1}) = R_{N-1} + B_{N-1}^T P_N B_{N-1} \succ 0$$

Discrete LQR – dynamic programming

Plugging in the optimal policy:

$$
\begin{aligned}
J^*_{N-1}(\mathbf{x}_{N-1}) &= \frac{1}{2}\mathbf{x}_{N-1}^T \big(Q_{N-1} + A_{N-1}^T P_N A_{N-1} - \\
&\qquad (A_{N-1}^T P_N B_{N-1} + S_{N-1})(R_{N-1} + B_{N-1}^T P_N B_{N-1})^{-1}(B_{N-1}^T P_N A_{N-1} + S_{N-1}^T)\big)\, \mathbf{x}_{N-1} \\
&:= \frac{1}{2}\mathbf{x}_{N-1}^T P_{N-1} \mathbf{x}_{N-1}
\end{aligned}
$$

Algebraic details aside:

- Cost-to-go (equivalently, “value function”) is a quadratic function of the state at each step
- Optimal policy is a time-varying linear feedback policy

Discrete LQR – dynamic programming

Proceeding by induction, we derive the Riccati recursion:

1. $P_N = Q_N$
2. $F_k = -(R_k + B_k^T P_{k+1} B_k)^{-1}(B_k^T P_{k+1} A_k + S_k^T)$
3. $P_k = Q_k + A_k^T P_{k+1} A_k -$
   $$(A_k^T P_{k+1} B_k + S_k)(R_k + B_k^T P_{k+1} B_k)^{-1}(B_k^T P_{k+1} A_k + S_k^T)$$
4. $\pi_k^*(\mathbf{x}_k) = F_k \mathbf{x}_k$
5. $J_k^*(\mathbf{x}_k) = \frac{1}{2}\mathbf{x}_k^T P_k \mathbf{x}_k$

Compute policy backwards in time, apply policy forward in time.

Next time

Stochastic dynamic programming

$$V^*(x) = \max_u \left(R(x,u) + \gamma \sum_{x' \in \mathcal{X}} T(x'|x,u)\, V^*(x') \right)$$