ECE7121 Learning-based control – 2025 Fall

Optimal control

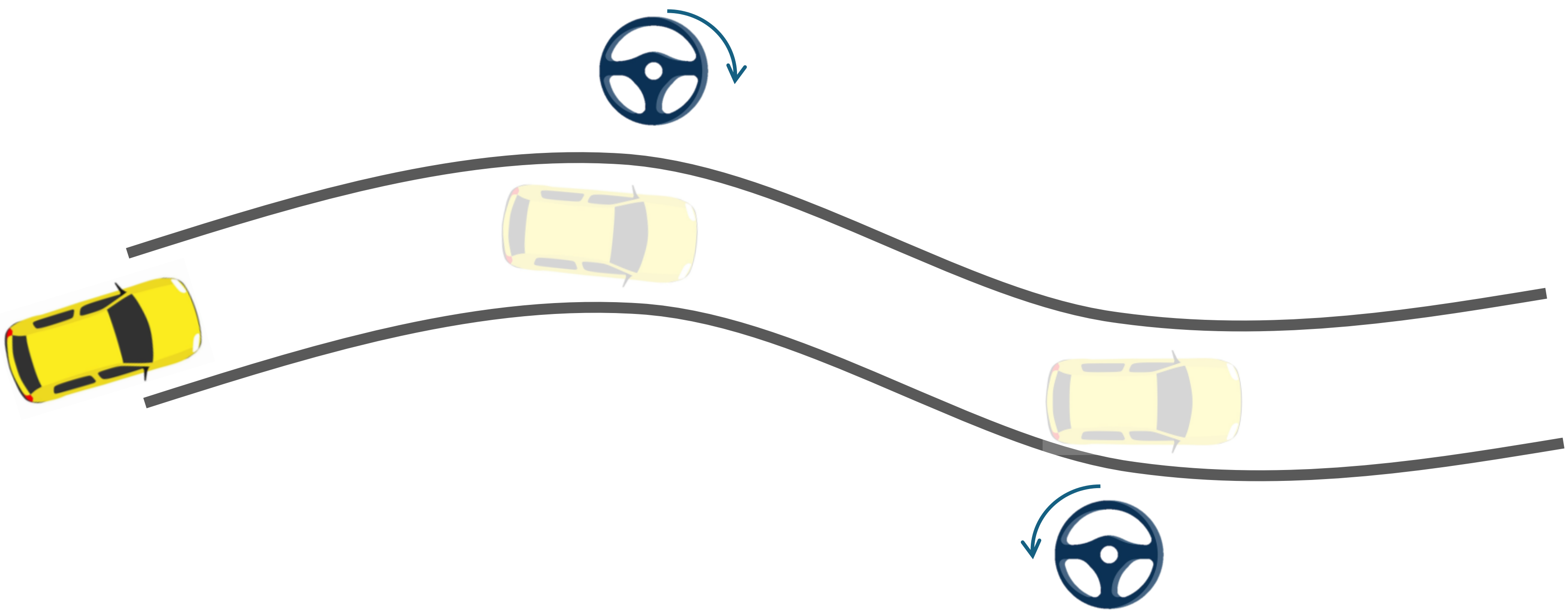
INHA UNIVERSITY

Overview

- Feedback control
- Optimization problem
- Linear system and LQR
- Trajectory optimization
- Dynamic programming

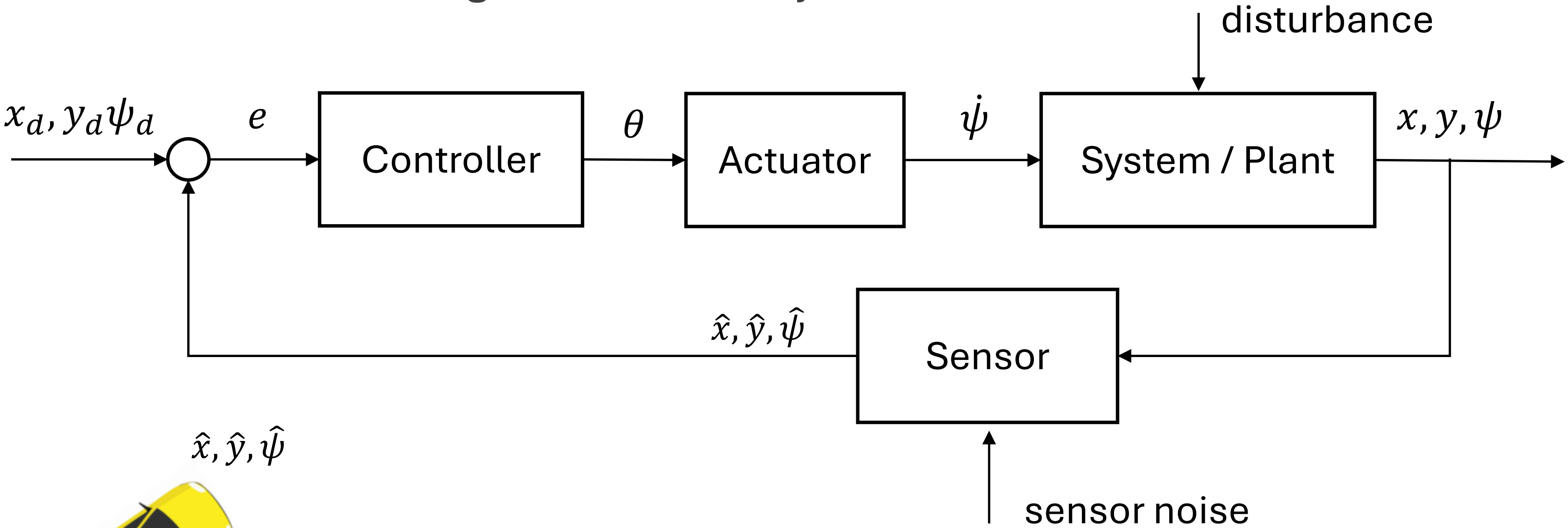
Feedback control

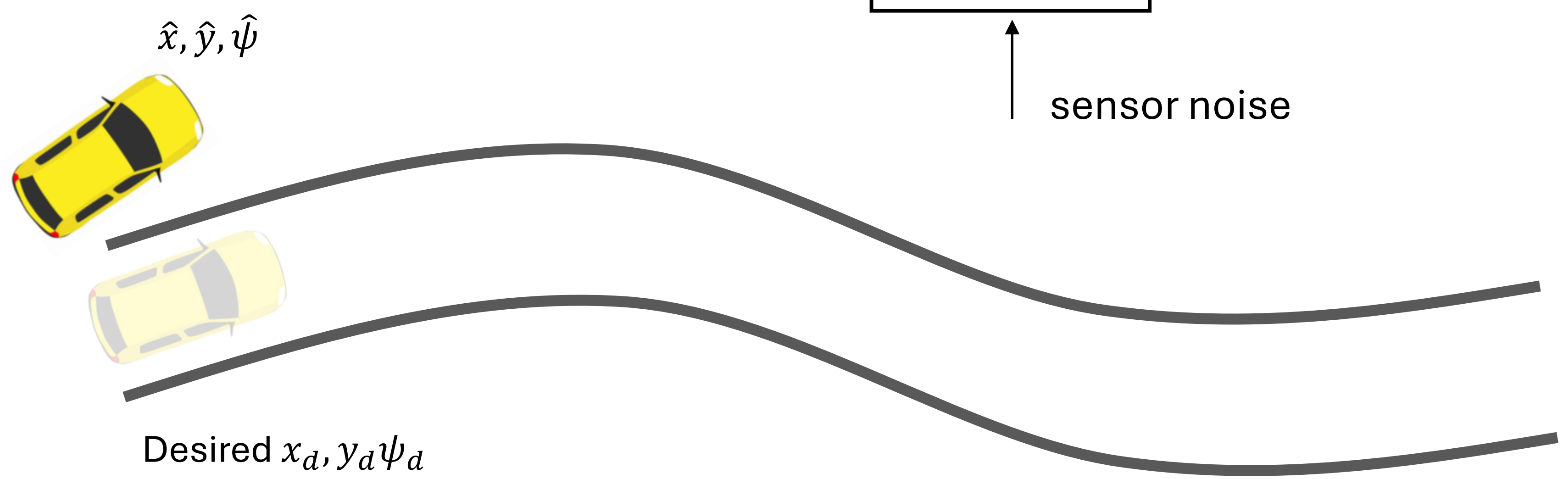
- Tracking a reference signal
  - Control input: steering wheel angle (θ)
  - Control output: vehicle yaw rate ($\dot{\psi}$)



Feedback control

- Reference tracking with uncertainty

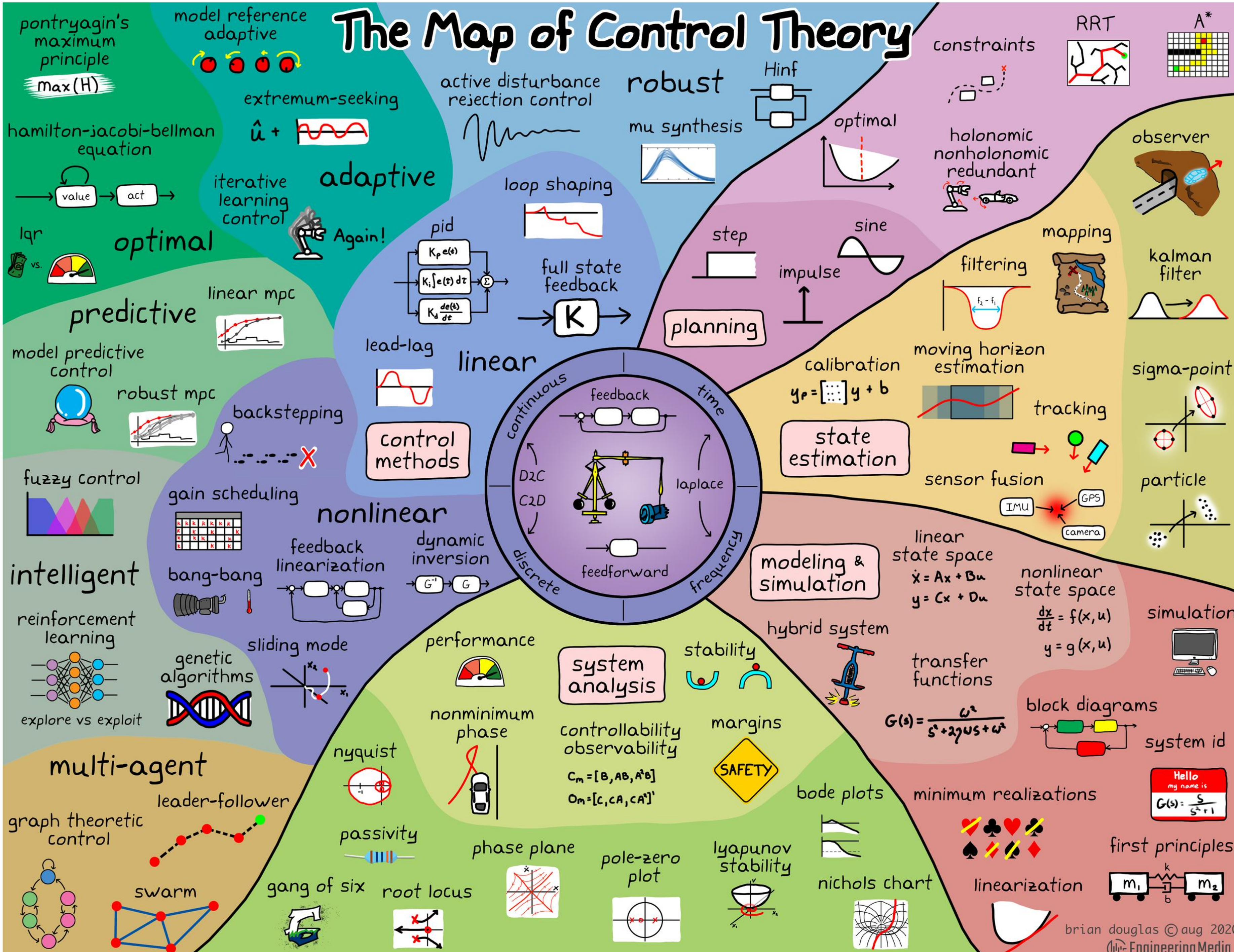




Feedback control

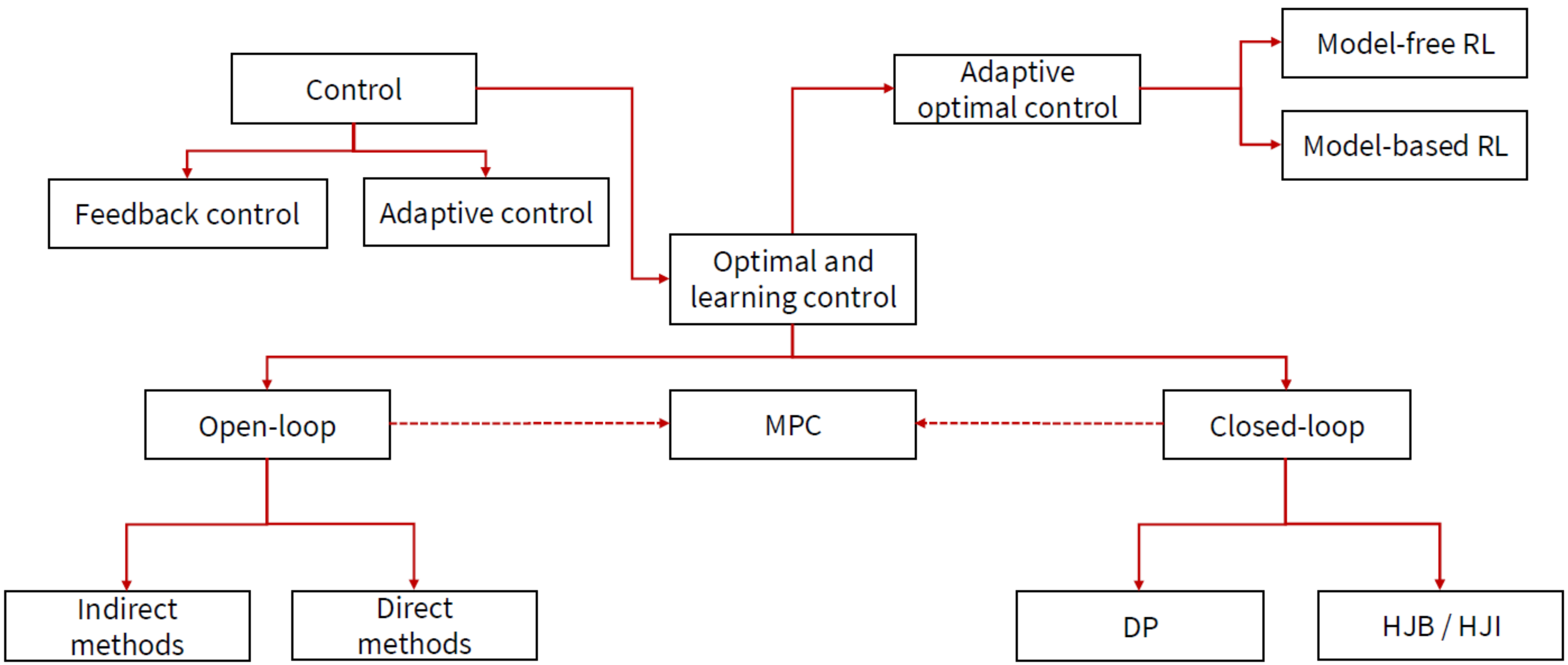
> Feedback: compare measurement with a desired value and use the difference to determine the control action

> Desiderata
  - stability: multiple notions; loosely system output is under control
  - tracking: the output should track the reference as closely as possible
  - disturbance rejection: the output should be insensitive to disturbance
  - robustness: controller should perform well up to some degree of model misspecification


The Map of Control Theory
pontryagin's maximum principle
max(H)
model reference adaptive
extremum-seeking
û +
hamilton-jacobi-bellman equation
value
act
iterative learning control
Again!
adaptive
lqr
vs.
optimal
active disturbance rejection control
robust
Hinf
mu synthesis
loop shaping
pid
$K_p e(t)$
$K_i \int e(\tau) d\tau$
$K_d \frac{de(t)}{dt}$
full state feedback
K
linear mpc
predictive
model predictive control
robust mpc
lead-lag
linear
control methods
backstepping
fuzzy control
gain scheduling
nonlinear
intelligent
bang-bang
feedback linearization
dynamic inversion
G^{-1}
G
reinforcement learning
explore vs exploit
genetic algorithms
sliding mode
x_2
x_1
multi-agent
leader-follower
graph theoretic control
swarm
constraints
RRT
A*
optimal
holonomic
nonholonomic
redundant
step
sine
impulse
planning
observer
mapping
filtering
$f_2 - f_1$
kalman filter
calibration
$y_P = [\,] y + b$
moving horizon estimation
state estimation
sigma-point
tracking
sensor fusion
IMU
GPS
camera
particle
continuous
time
feedback
D2C
C2D
laplace
feedforward
discrete
frequency
modeling & simulation
linear state space
$\dot{x} = Ax + Bu$
$y = Cx + Du$
nonlinear state space
$\frac{dx}{dt} = f(x,u)$
$y = g(x,u)$
simulation
hybrid system
transfer functions
$G(s) = \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$
block diagrams
system id
Hello my name is
$G(s) = \frac{s}{s^2+1}$
minimum realizations
first principles
linearization
k
b
m_1
m_2
performance
system analysis
stability
nonminimum phase
controllability observability
$C_m = [B, AB, A^2B]$
$O_m = [C, CA, CA^2]'$
margins
SAFETY
nyquist
-1
bode plots
passivity
phase plane
pole-zero plot
lyapunov stability
nichols chart
gang of six
root locus
brian douglas © aug 2020
EngineeringMedia

The map of control theory

> Intertwining optimal control and learning-based control to understand the framework and context



The map of control theory

> Intertwining optimal control and learning-based control to understand the framework and context

Optimal control — with known dynamics model, trajectory optimization — iLQR, MPC

entire state space → Dynamic programming — use Bellman optimality, obtain value function and policy — VI, PI

no given model → model-based RL — model learning first, then optimize — Dyna, PETS

→ model-free RL — without the model — DQN, PPO

Problem formulation

> Mathematical description of the system to be controlled

- state variables: $x_1(t), x_2(t), \dots, x_n(t)$
- control inputs: $u_1(t), u_2(t), \dots, u_m(t)$
- model:

$$
\begin{aligned}
&\dot{x}_1(t) = f_1(x_1(t), x_2(t), \dots, x_n(t), u_1(t), u_2(t), \dots, u_m(t), t)\\
&\dot{x}_2(t) = f_2(x_1(t), x_2(t), \dots, x_n(t), u_1(t), u_2(t), \dots, u_m(t), t)\\
&\dots\\
&\dot{x}_n(t) = f_n(x_1(t), x_2(t), \dots, x_n(t), u_1(t), u_2(t), \dots, u_m(t), t)
\end{aligned}
$$

- In compact form: $\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)$

Problem formulation

> Constraints

- initial and final conditions (boundary conditions)

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0, \qquad \boldsymbol{x}(t_f) = \boldsymbol{x}_f$$

- constraints on state trajectories

$$\underline{X} \leq \boldsymbol{x}(t) \leq \overline{X}$$

- control authority

$$\underline{U} \leq \boldsymbol{u}(t) \leq \overline{U}$$

- and many more ...

Problem formulation

- Performance measure

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

h (terminal cost) and g (running cost) are scalar functions

- Problem: find an admissible control $\boldsymbol{u}^*$ which causes the system

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

to follow an admissible trajectory $\boldsymbol{x}^*$ that minimizes the performance measure

Optimization problem

> Finding the minimizer of a function subject to constraints:

$$\underset{x}{\text{minimize}}\, J(\boldsymbol{x})$$

$$\begin{aligned} s.t.\, f_i(\boldsymbol{x}) &\leq 0,\ \ i = \{1, \dots, k\} \\ h_j(\boldsymbol{x}) &= 0,\ \ j = \{1, \dots, l\} \end{aligned}$$

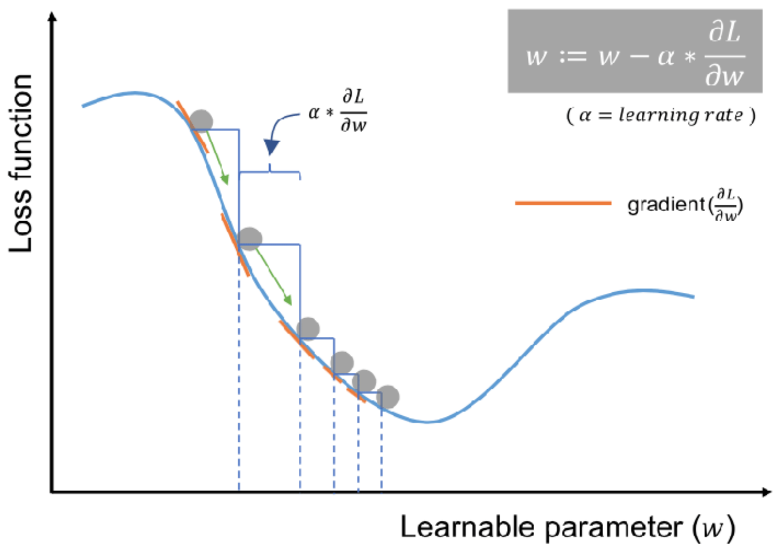
> Gradient: $\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \cdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$, Hessian: $\nabla^2 f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$

Optimization problem

> Unconstrained optimization
  - $\underset{x}{\text{minimize}}\, f(\boldsymbol{x})$
  - Necessary condition: $\nabla f(\boldsymbol{x}) = 0, \;\; \nabla^2 f(\boldsymbol{x}^*) \geq 0$
  - Sufficient condition: $\nabla f(\boldsymbol{x}) = 0, \;\; \nabla^2 f(\boldsymbol{x}^*) > 0$

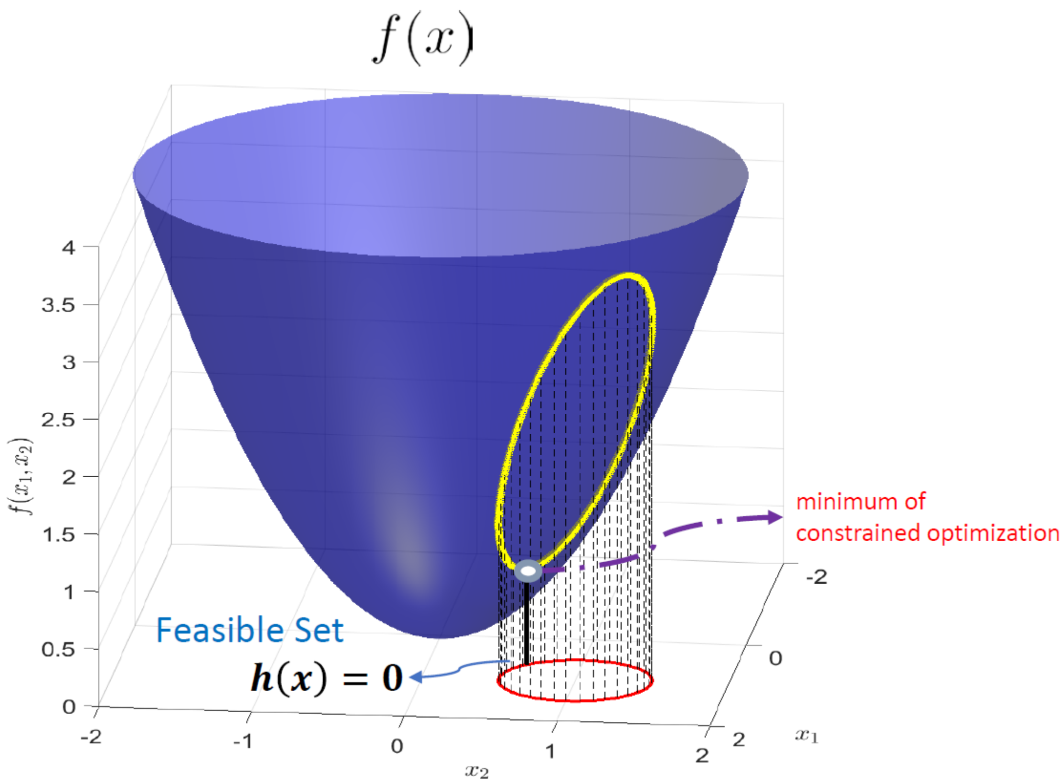
> Iterative gradient descent
  - $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \alpha_t \nabla f(\boldsymbol{x}_t)$

> Newton's method
  - $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - (\nabla^2 f(x_t))^{-1} \nabla f(x_t)$



Optimization problem

> Optimization with equality constraints
  - $\underset{x}{\text{minimize}}\, f(\boldsymbol{x})$
    $s.t.\ \ h_i(\boldsymbol{x}) = 0, \quad i = 1, \dots, m$



Optimization problem

> Lagrange multipliers

- for a given local minimum $\boldsymbol{x}^*$, there exists scalars $\lambda_1, \dots, \lambda_m$ called Lagrange multipliers such that

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}^*) = 0$$

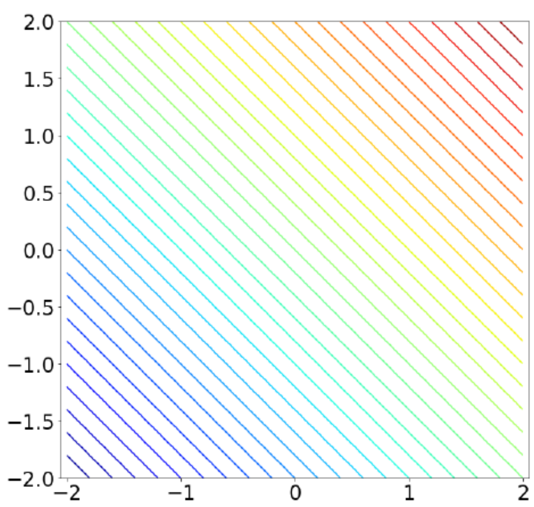
- If we had no constraints, the solution satisfies $\nabla f(\boldsymbol{x}^*) = 0$
- With equality constraints, we are not allowed to move in all directions
- We can move only along directions that stay on the constraint surface
- At an optimum, any movement along the constraint surface should not decrease the value of the objective. for direction d, $\nabla h_i(\boldsymbol{x}^*)^\top d = 0$
- If we move in a direction d, directional derivative of f along d must be zero (otherwise you could decrease f), $\nabla f(\boldsymbol{x}^*)^\top d = 0$
- The gradient of f must be orthogonal to the space of feasible directions
- $\nabla f(\boldsymbol{x}^*)$ must lie in the span of $\nabla h_i(\boldsymbol{x}^*)$, $\nabla f(\boldsymbol{x}^*) = \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}^*)$

Optimization problem

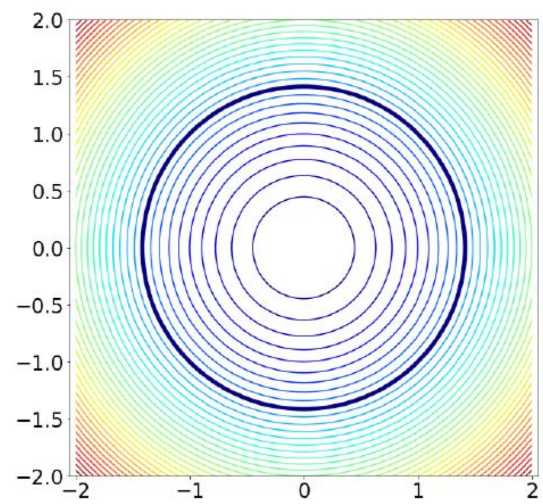
> Lagrange multipliers
  - for a given local minimum $\boldsymbol{x}^*$, there exists scalars $\lambda_1, \dots, \lambda_m$ called Lagrange multipliers such that

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}^*) = 0$$

$$f(\mathbf{x}) = x_1 + x_2$$



$$h(\mathbf{x}) = x_1^2 + x_2^2 - 2$$

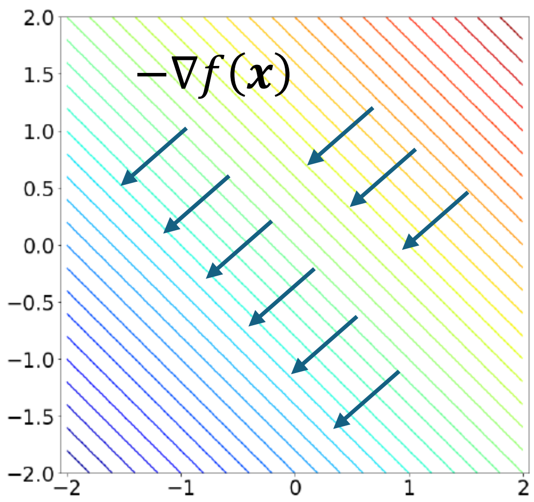


Optimization problem

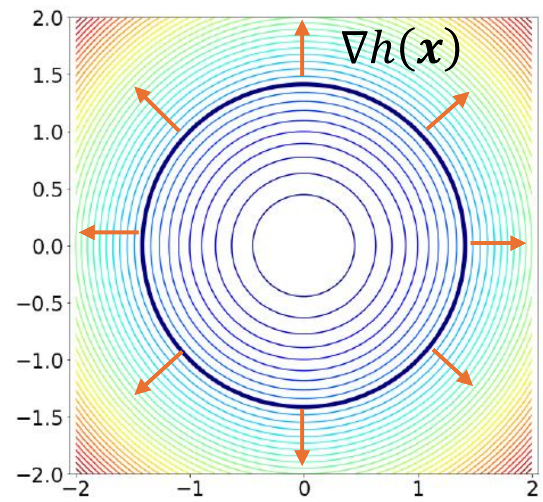
> Lagrange multipliers
  - for a given local minimum $\boldsymbol{x}^*$, there exists scalars $\lambda_1, \dots, \lambda_m$ called Lagrange multipliers such that

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}^*) = 0$$

$$f(\mathbf{x}) = x_1 + x_2$$



$$h(\mathbf{x}) = x_1^2 + x_2^2 - 2$$



Optimization problem

> Lagrange multipliers
  - for a given local minimum $\boldsymbol{x}^*$, there exists scalars $\lambda_1, \ldots, \lambda_m$ called Lagrange multipliers such that

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}^*) = 0$$

$$f(\mathbf{x}) = x_1 + x_2 \qquad h(\mathbf{x}) = x_1^2 + x_2^2 - 2$$

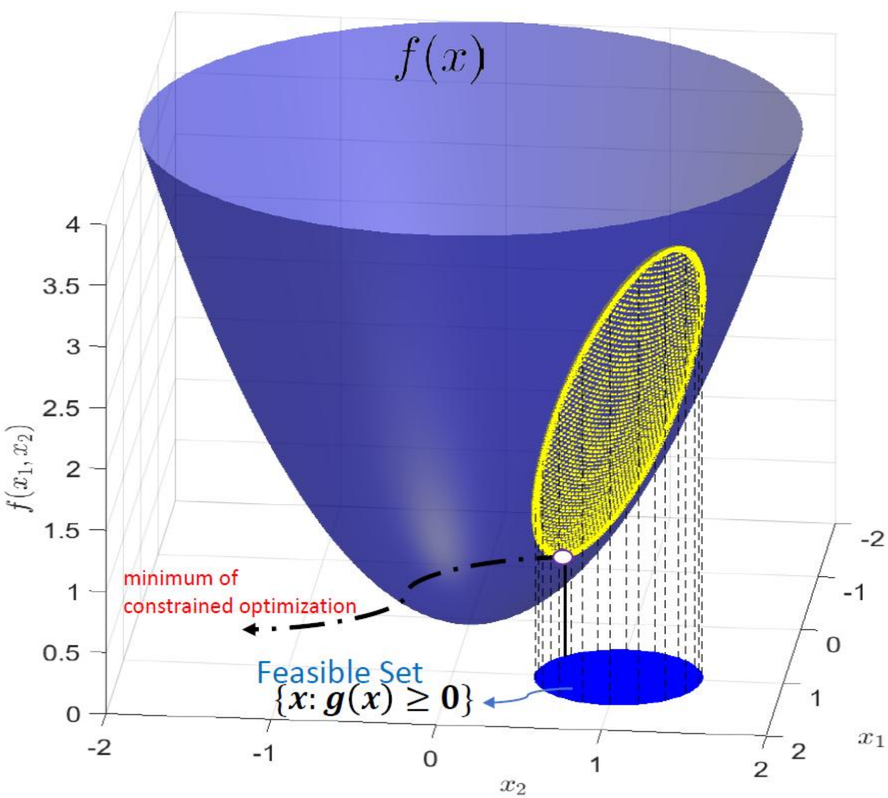
  - $\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \; x_1^2 + x_2^2 - 2 = 0$

    $\frac{2}{\lambda^2} = 2, \rightarrow \lambda = \pm 1 \rightarrow (x_1, x_2) = (\pm 1, \pm 1)$

  - Optimal point: $(x_1^*, x_2^*) = (-1, -1)$

Optimization problem

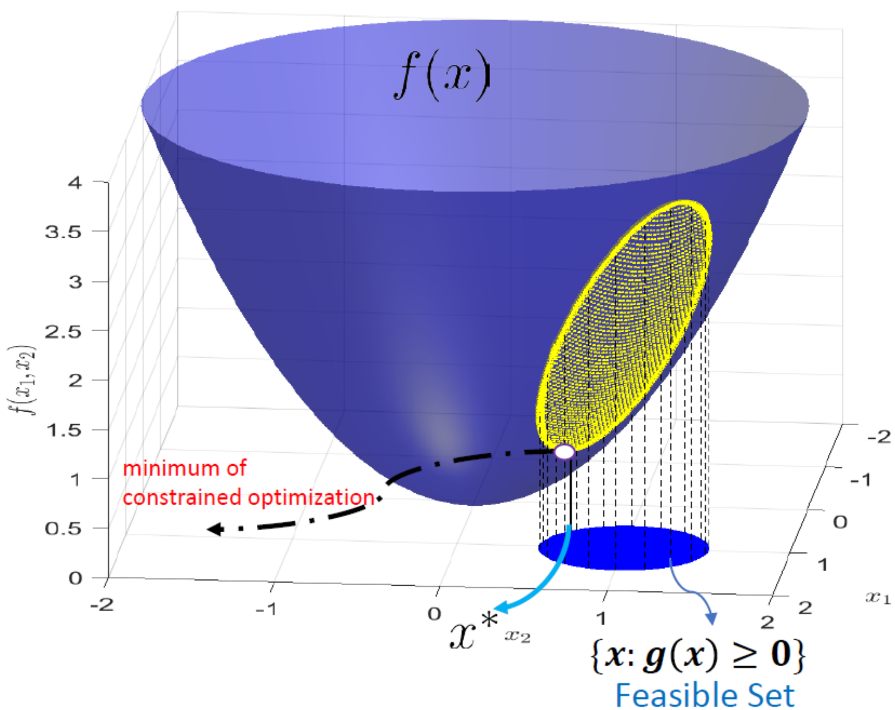
> Optimization with inequality constraints
  - $\underset{x}{\text{minimize}}\, f(\boldsymbol{x})$
    $s.t. \quad g_i(\boldsymbol{x}) \geq 0, \quad i = 1, \dots, r$



Optimization problem

> It is similar to equality constrained optimization
   - $g(\boldsymbol{x}^*) = 0$
   - $\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{r} \mu_i \nabla g_i(\boldsymbol{x}^*) = 0$

Case 1: x^* is on the boundary of the feasible region.

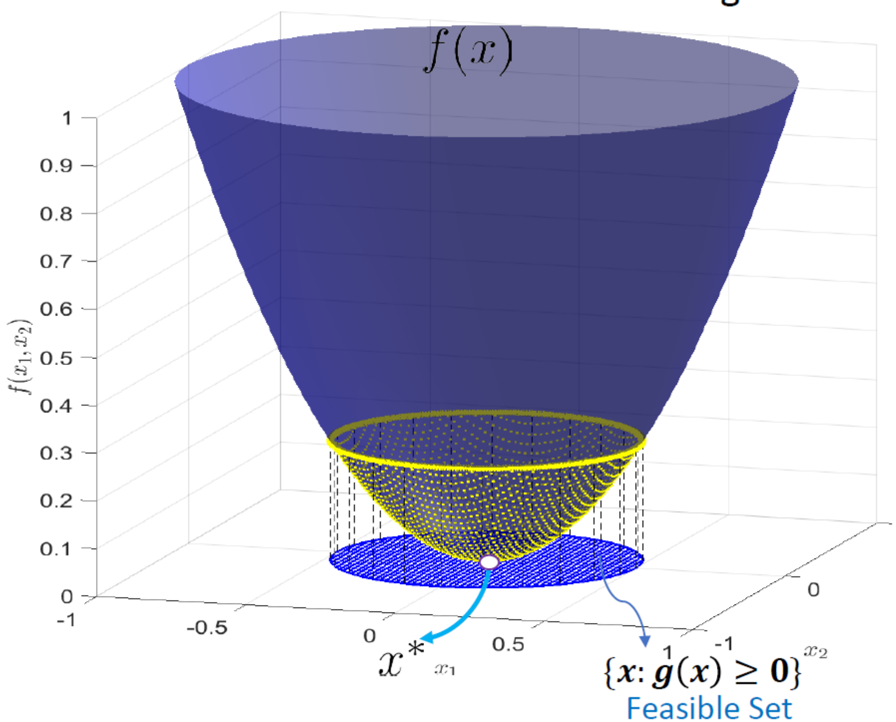


if the constraint $g_i(\boldsymbol{x}) \geq 0, \mu_i \leq 0$
if the constraint $g_i(\boldsymbol{x}) \leq 0, \mu_i \geq 0$

Optimization problem

> It is similar to unconstrained optimization
  - $g(\boldsymbol{x}^*) > 0$
  - $\nabla f(\boldsymbol{x}^*) = 0$
  - equivalent to $\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{r} \mu_i \nabla g_i(\boldsymbol{x}^*) = 0\,,\ \mu_i = 0$

Case 2: x^* is inside the feasible region.



Optimization problem

- $$\underset{x}{\text{minimize}}\, f(\boldsymbol{x})$$
  $$s.t.\, h_i(\boldsymbol{x}) = 0, \quad i = 1, \dots, m$$
  $$g_i(\boldsymbol{x}) \leq 0, \quad i = 1, \dots, r$$

- Define Lagrange function
  $$L(\boldsymbol{x}, \lambda, \mu) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i h_i(\boldsymbol{x}) + \sum_{j=1}^{r} \mu_j g_j(\boldsymbol{x})$$

  - $\nabla_x L(x, \lambda, \mu) = 0$ $\quad \rightarrow \nabla f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i \nabla h_i(\boldsymbol{x}) + \sum_{j=1}^{r} \mu_j \nabla g_j(\boldsymbol{x}) = 0$
  - $\mu_i \geq 0$
  - $\mu_i^* g_i(\boldsymbol{x}^*) = 0$
  - $h_i(\boldsymbol{x}) = 0$
  - $g_i(\boldsymbol{x}) \leq 0$

  Necessary optimality condition called **Karush-Kuhn-Tucker (KKT) condition**

Linear dynamical system

> Model: $\boldsymbol{x}_{t+1} = A_t\boldsymbol{x}_t + B_t\boldsymbol{u}_t, \quad \boldsymbol{y}_t = C_t\boldsymbol{x}_t$

> Linear quadratic control

- $$\begin{aligned} \underset{\boldsymbol{u}}{\text{minimize}}\, & J_{\boldsymbol{y}} + \rho J_{\boldsymbol{u}} \\ s.t.\quad & \boldsymbol{x}_{t+1} = A_t\boldsymbol{x}_t + B_t\boldsymbol{u}_t, \quad t = 1, \dots, T-1 \\ & \boldsymbol{x}_1 = \boldsymbol{x}_{init}, \ \boldsymbol{x}_T = \boldsymbol{x}_{des} \end{aligned}$$

  $$J_{\boldsymbol{y}} = \sum_{t=1}^{T} \|\boldsymbol{y}_t\|^2 = \|C\boldsymbol{x}_t\|^2, \qquad J_{\boldsymbol{u}} = \sum_{t=1}^{T} \|\boldsymbol{u}_t\|^2$$

- first constraint imposes the linear dynamics equations
- second set of constraints specifies the initial and final state
- $\boldsymbol{y}$ would be the error state (the difference from the desired state)

Linear dynamical system

> Can be written as

$$\underset{u}{\text{minimize}} \left\|\tilde{A}\boldsymbol{z} - \tilde{\boldsymbol{b}}\right\|^2$$
$$s.t.\quad \tilde{C}\boldsymbol{z} = \tilde{\boldsymbol{d}}$$

$$\tilde{A} = \left[\begin{array}{ccc|ccc} C_1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & C_T & 0 & \cdots & 0 \\ \hline 0 & \cdots & 0 & \sqrt{\rho}I & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & \sqrt{\rho}I \end{array}\right], \qquad \tilde{b} = 0$$

> KKT conditions in matrix form

- $$\begin{bmatrix} 2\tilde{A}^\top \tilde{A} & \tilde{C}^\top \\ \tilde{C} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{z} \\ \lambda \end{bmatrix} = \begin{bmatrix} 2\tilde{A}^\top \tilde{\boldsymbol{b}} \\ \tilde{\boldsymbol{d}} \end{bmatrix}$$

$$\tilde{C} = \left[\begin{array}{cccccc|cccc} A_1 & -I & 0 & \cdots & 0 & 0 & B_1 & 0 & \cdots & 0 \\ 0 & A_2 & -I & \cdots & 0 & 0 & 0 & B_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & A_{T-1} & -I & 0 & 0 & \cdots & B_{T-1} \\ \hline I & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & I & 0 & 0 & \cdots & 0 \end{array}\right], \qquad \tilde{d} = \left[\begin{array}{c} 0 \\ 0 \\ \vdots \\ 0 \\ \hline x^{\text{init}} \\ x^{\text{des}} \end{array}\right]$$

> Solve the KKT conditions to find candidate solutions
then, evaluate the original cost function to determine the optimum

Linear dynamical system

> Linear quadratic regulator (LQR)

- with linear dynamics model: $\boldsymbol{x}_{t+1} = A_t \boldsymbol{x}_t + B_t \boldsymbol{u}_t$ and quadratic cost f.

- $$\underset{\boldsymbol{u}}{\text{minimize}} \frac{1}{2}\boldsymbol{x}_T^\top L \boldsymbol{x}_T + \sum_{t=0}^{T-1} \frac{1}{2}\boldsymbol{x}_t^\top Q \boldsymbol{x}_t + \frac{1}{2}\boldsymbol{u}_k^\top R \boldsymbol{u}_k$$
  $$s.t. \quad \boldsymbol{x}_{t+1} = A_t \boldsymbol{x}_t + B_t \boldsymbol{u}_t, \quad t = 1, \ldots, T-1$$
  $$\boldsymbol{x}_1 = \boldsymbol{x}_{init}, \ \boldsymbol{x}_T = \boldsymbol{x}_{des}$$

- L, Q, R are positive definite weight matrices
- We want to stabilize the system in the end, so we prioritize the terminal state

Linear dynamical system

> Constrained LQR

- $$\begin{aligned} &\underset{\boldsymbol{u}}{\text{minimize}} \frac{1}{2}\boldsymbol{x}_T^\top L \boldsymbol{x}_T + \textstyle\sum_{t=0}^{T-1} \frac{1}{2}\boldsymbol{x}_t^\top Q \boldsymbol{x}_t + \frac{1}{2}\boldsymbol{u}_k^\top R \boldsymbol{u}_k \\ &s.t. \quad \boldsymbol{x}_{t+1} = A_t \boldsymbol{x}_t + B_t \boldsymbol{u}_t, \quad t = 1, \dots, T-1 \\ &\qquad\quad \boldsymbol{x}_1 = \boldsymbol{x}_{init}, \ \boldsymbol{x}_T = \boldsymbol{x}_{des} \\ &\qquad\quad \mathbf{u}_{\min} \leq \boldsymbol{u}_t \leq \boldsymbol{u}_{max}, \ \boldsymbol{x}_{min} \leq \boldsymbol{x} \leq \boldsymbol{x}_{max} \end{aligned}$$

  (inequality constraints)

- In reality, there are always control input and state boundaries

> It is a special case of quadratic program (QP)

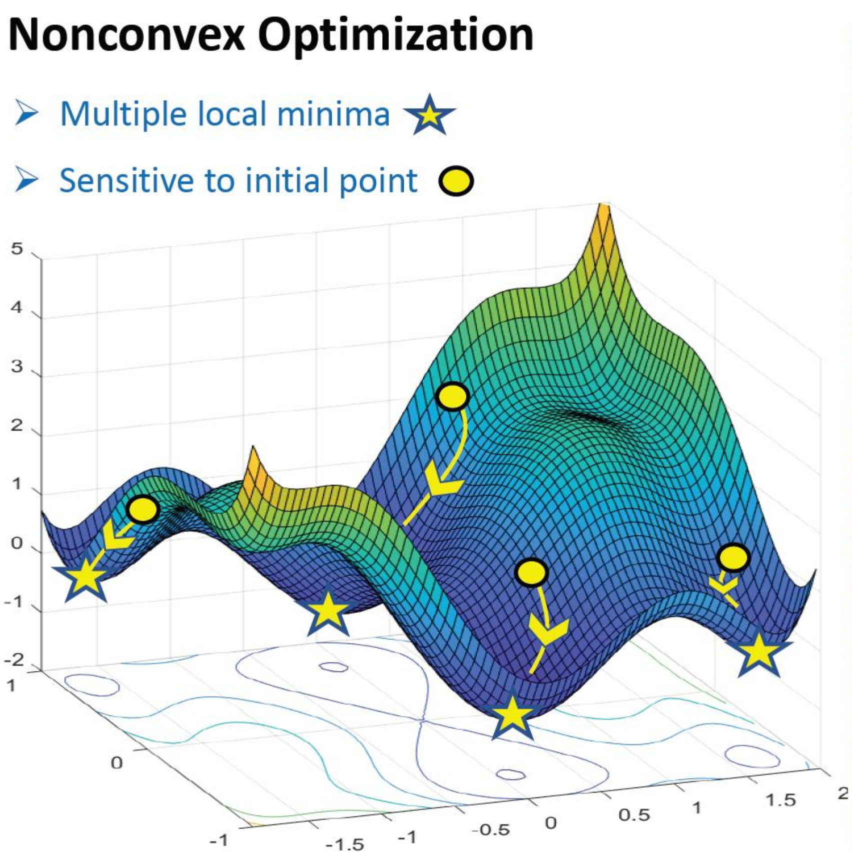
- $$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} \frac{1}{2}\boldsymbol{x}^\top Q \boldsymbol{x} + q\boldsymbol{x} \\ &s.t. \quad A_{in}\boldsymbol{x} \leq b_{in} \\ &\qquad\ A_{eq}\boldsymbol{x} = b_{eq} \\ &\qquad\ lb \leq \boldsymbol{x} \leq ub \end{aligned}$$

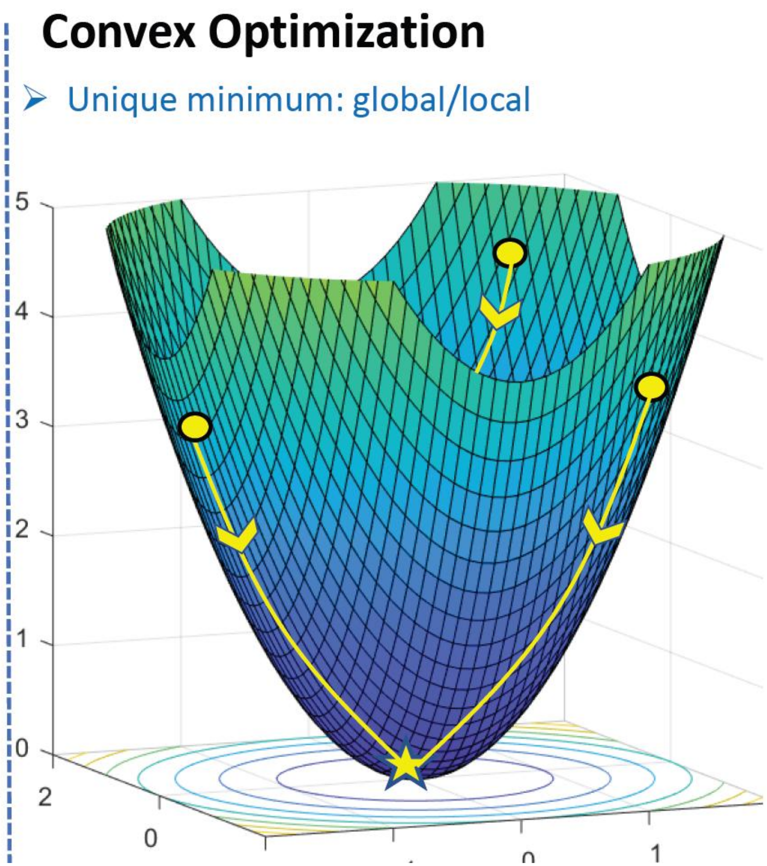
  Q is positive definite.

- $\boldsymbol{x}^*$ exists and is unique (QP is convex and feasible) unless non-empty constraint subset

Convex optimization

> With convex cost function, convex inequality constraints, and affine equality constraints





Linear dynamical system

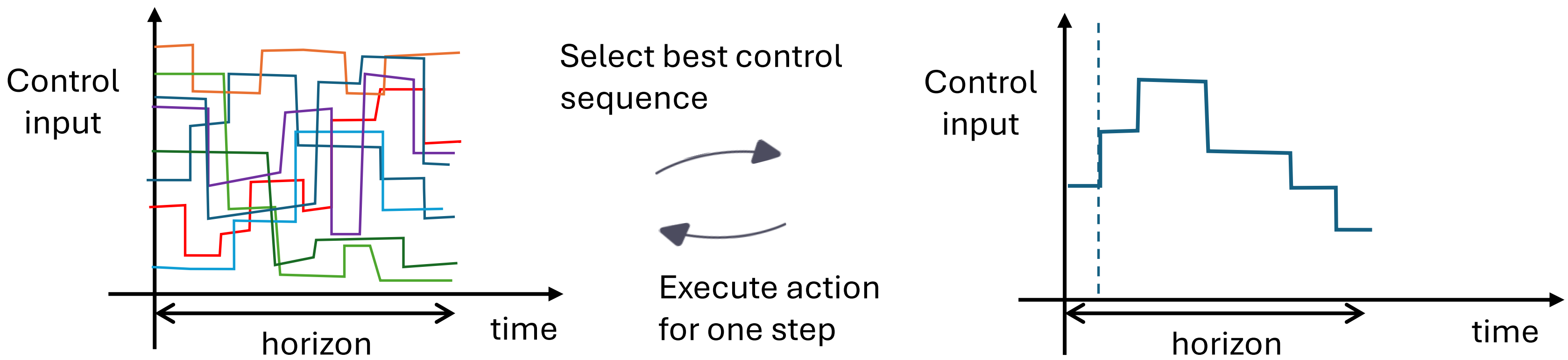
> LQR solution gives all the sequences of state and input

> Applying the control input sequence leads to an open-loop control policy
  - there is no feedback along the way
  - LQR is agnostic to future changes during execution
  - it can’t handle the model error and estimation error

> Instead of executing everything, just execute the first action and do the optimization again = MPC
  - it needs quick optimization (we need it every timestep)
  - we have a very complicated feedback, it is difficult to analyze it
  - still, works very well in practice

Model predictive control

> With finite horizon, iterative planning framework



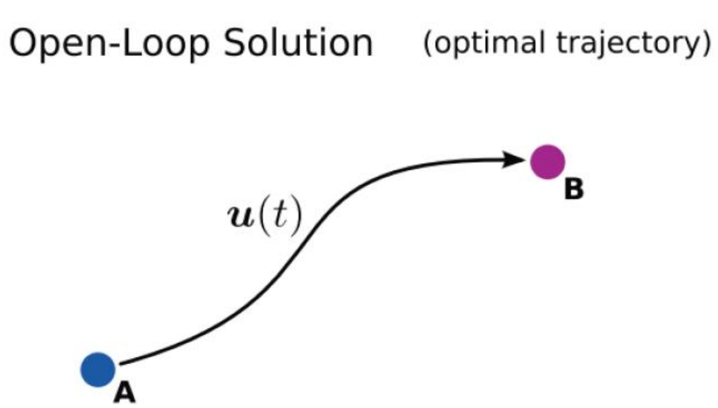
- Depending on the optimization solver
  - nonlinear program (nonlinear solver) = nonlinear MPC
  - quadratic program = QP-MPC = linear MPC (recommended!)
  - sampling-based optimization >> model-based RL

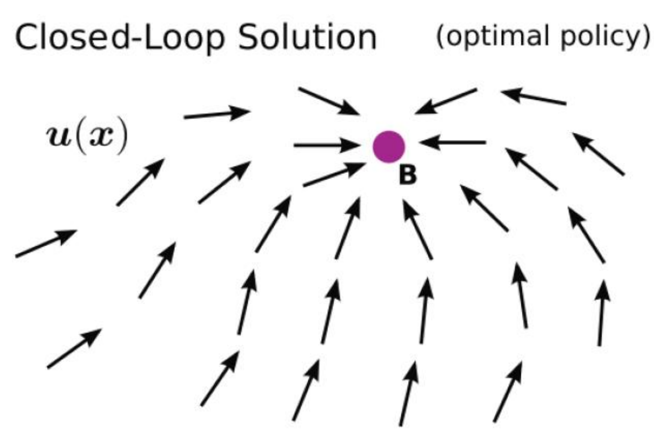
Trajectory optimization

> TO is a collection of techniques that are used to find open-loop solutions to an optimal control problem

- $$\begin{aligned} &\underset{\boldsymbol{u}}{\text{minimize}}\, N(\boldsymbol{x}_T) + \int_0^T L(\boldsymbol{x}_t, \boldsymbol{u}_t)dt \\ &s.t. \quad \dot{\boldsymbol{x}} = f(\boldsymbol{x}_t, \boldsymbol{u}_t, t) \\ &\qquad\quad \boldsymbol{x}(0) = \boldsymbol{x}_{init},\ \boldsymbol{x}(T) = \boldsymbol{x}_{des} \\ &\qquad\quad + \textit{additional constraints} \end{aligned}$$

- additional constraints include path constraints and bounds on $\boldsymbol{x}, \boldsymbol{u}$





Trajectory optimization

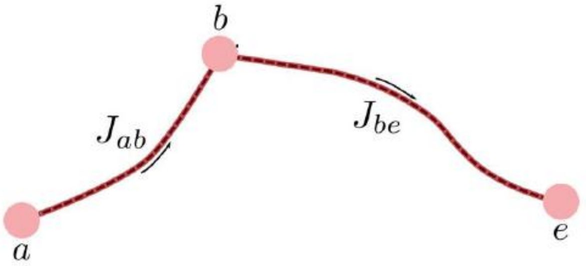
- Indirect methods (optimize -> discretize)
  - analytically construct the conditions for optimality
  - Euler-Lagrange eqn., Hamilton-Jacobi-Bellman eqn.
  - more accurate, harder to pose and solve
  - usually used in aerospace industry
- Direct methods (discretize -> optimize)
  - numerically optimizing it as a nonlinear program
  - more recent and more widely used in robotics
  - less accurate, easier to pose and solve

- Solving strategies
  - shooting: only solve for inputs and simulate the state
  - collocation: parametrize the problem via many trajectory segments\
  - These are connected to model-based RL in future

Dynamic programming

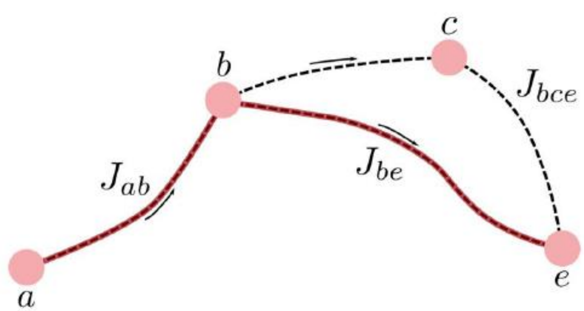
> Focus is now on finding optimal closed-loop policies
  - cost: $J = h\left(\boldsymbol{x}\left(t_f\right), t_f\right) + \int_{t_0}^{t_f} g(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt$
  - model: $\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)$
  - control constraints: $\boldsymbol{u}(t) \in U(\boldsymbol{x}(t))$
  - Find $\boldsymbol{u}^*(t) = \pi^*(\boldsymbol{x}_t)$

> Principle of optimality
  - the key concept behind the DP is the principle of optimality
  - cost a-e = cost a-b + cost b-e
    $J_{ae} = J_{ab} + J_{be}$



Dynamic programming

> If $a - b - e$ is optimal path from a to e, then $\mathrm{b} - e$ is optimal path from b to e
  - proof:
    suppose $\mathrm{b} - \mathrm{c} - e$ is the optimal path from b to e
    then, $J_{bce} < J_{be}$
    and $\mathrm{J}_{ab} + \mathrm{J}_{bce} < J_{ab} + J_{be} = J_{ae}^{*}$ (contradiction !)



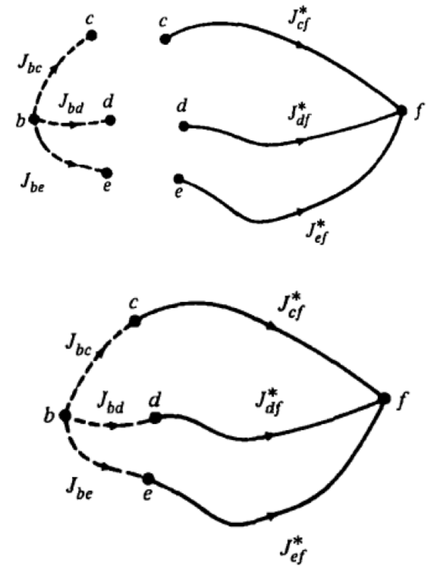
> Principle of optimality: tail of optimal sequences is optimal for tail subproblems

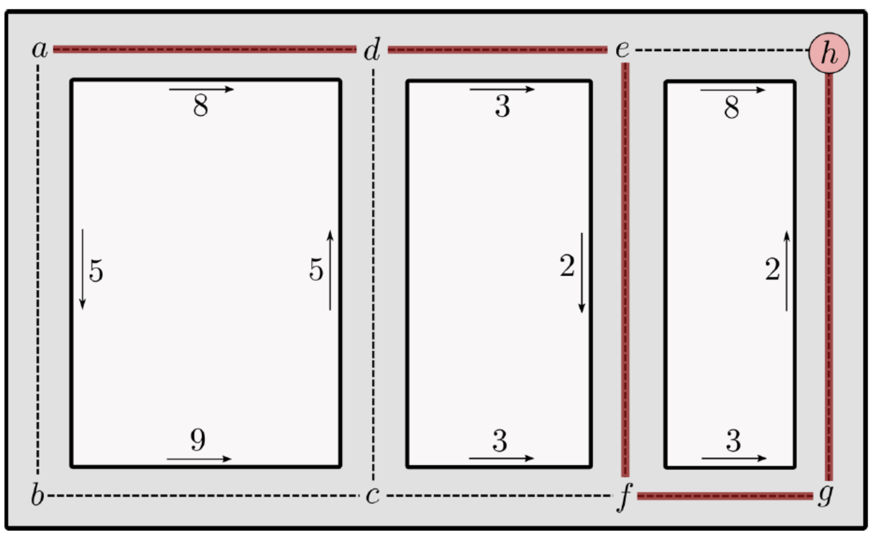
Dynamic programming

> Applying the principle of optimality
  - the optimal trajectory is found by comparing

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

  - carry out this procedure backward in time





Dynamic programming

> In most cases, DP algorithm needs to be performed numerically

> A few case can be solved analytically (LQR)

- $$\underset{\boldsymbol{u}}{\text{minimize}} \frac{1}{2}\boldsymbol{x}_T^\top L \boldsymbol{x}_T + \sum_{t=0}^{T-1} \frac{1}{2}\boldsymbol{x}_t^\top Q \boldsymbol{x}_t + \frac{1}{2}\boldsymbol{u}_k^\top R \boldsymbol{u}_k$$
  $$s.t.\quad \boldsymbol{x}_{t+1} = A_t \boldsymbol{x}_t + B_t \boldsymbol{u}_t, \quad t = 1, \dots, T-1$$
  $$\boldsymbol{x}_1 = \boldsymbol{x}_{init}, \ \boldsymbol{x}_T = \boldsymbol{x}_{des}$$

- $J_T^*(\boldsymbol{x_T}) = \frac{1}{2}\boldsymbol{x}_T^\top L \boldsymbol{x}_T$
- going backward

$$\begin{aligned} J_{T-1}(\boldsymbol{x_{T-1}}) &= \underset{\boldsymbol{u}_{T-1}}{\text{minimize}} \frac{1}{2}\boldsymbol{x}_{T-1}^\top Q \boldsymbol{x}_{T-1} + \frac{1}{2}\boldsymbol{u}_{T-1}^\top R \boldsymbol{u}_{T-1} + \frac{1}{2}\boldsymbol{x}_T^\top L \boldsymbol{x}_T \\ &= \underset{\boldsymbol{u}_{T-1}}{\text{minimize}} \frac{1}{2}\boldsymbol{x}_{T-1}^\top Q \boldsymbol{x}_{T-1} + \frac{1}{2}\boldsymbol{u}_{T-1}^\top R \boldsymbol{u}_{T-1} + \\ &\qquad \frac{1}{2}(A_{T-1}\boldsymbol{x}_{T-1} + B_{T-1}\boldsymbol{u}_{T-1})^\top H (A_{T-1}\boldsymbol{x}_{T-1} + B_{T-1}\boldsymbol{u}_{T-1}) \end{aligned}$$

Dynamic programming

- Taking derivative

$$\frac{\partial J^*_{T-1}(\boldsymbol{x}_{T-1})}{\partial \boldsymbol{u}_{T-1}} = R\boldsymbol{u}_{T-1} + B^\top_{T-1} H(A_{T-1}\boldsymbol{x}_{T-1} + B_{T-1}\boldsymbol{u}_{T-1}) = 0$$

$$\boldsymbol{u}^*_{T-1} = -(R + B^\top_{T-1} H B_{T-1})^{-1} B^\top_{T-1} H A_{T-1}\boldsymbol{x}_{T-1} = F_{T-1}\boldsymbol{x}_{T-1}$$

$$\begin{aligned} J_{T-1}(\boldsymbol{x}_{T-1}) &= \frac{1}{2}\boldsymbol{x}^\top_{T-1}\{Q + F^\top_{T-1} R F_{T-1} + \\ &\qquad (A_{T-1} + B_{T-1}F_{T-1})^\top H (A_{T-1} + B_{T-1}F_{T-1})\}\boldsymbol{x}_{T-1} \\ &= \boldsymbol{x}^\top_{T-1} P_{T-1} \boldsymbol{x}_{T-1} \end{aligned}$$

- Proceeding by induction
  - $\boldsymbol{u}^*_t = F_t \boldsymbol{x}_t$ where $F_t = -(R + B_t^\top P_{t+1} B_t)^{-1} B_t^\top P_{t+1} A_t$
  - $J_t(\boldsymbol{x}_t) = \frac{1}{2}\boldsymbol{x}_t^\top P_t \boldsymbol{x}_t^\top$ where $P_t = Q + F_t^\top R F_t + (A_t + B_t F_t)^\top P_{t+1}(A_t + B_t F_t)$
  - At the end we could get $J_0(\boldsymbol{x}_0) = \frac{1}{2}\boldsymbol{x}_0^\top P_0 \boldsymbol{x}_0$

Dynamic programming

- We can do same things in continuous-time setting
  - DP leads to Hamilton-Jacobi-Bellman (HJB), Hamilton-Jacobi-Issacs (HJI) equations
  - HJB: $\frac{\partial J}{\partial t} + \min_{\mathrm{u}} [g(\boldsymbol{x}, \boldsymbol{u}, t) + \frac{\partial J}{\partial \boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{u}, t)] = 0$
  - but, solving PDEs is hard except in simple cases
  - for high-dimensional systems, numerical solutions become infeasible

What if the problem is nonconvex?

- Local optimization method
  - sequential quadratic/convex programming (SQP, SCP)
- Dynamic programming + local optimization
  - Iterative linear quadratic regulator (iLQR)
  - Differential dynamic programming (DDP)
- Sampling-based methods
  - Model predictive path integral control (MPPI)
  - Cross-entropy method (CEM)
  - Covariance matrix adaptation evolutionary strategy (CMA-ES)