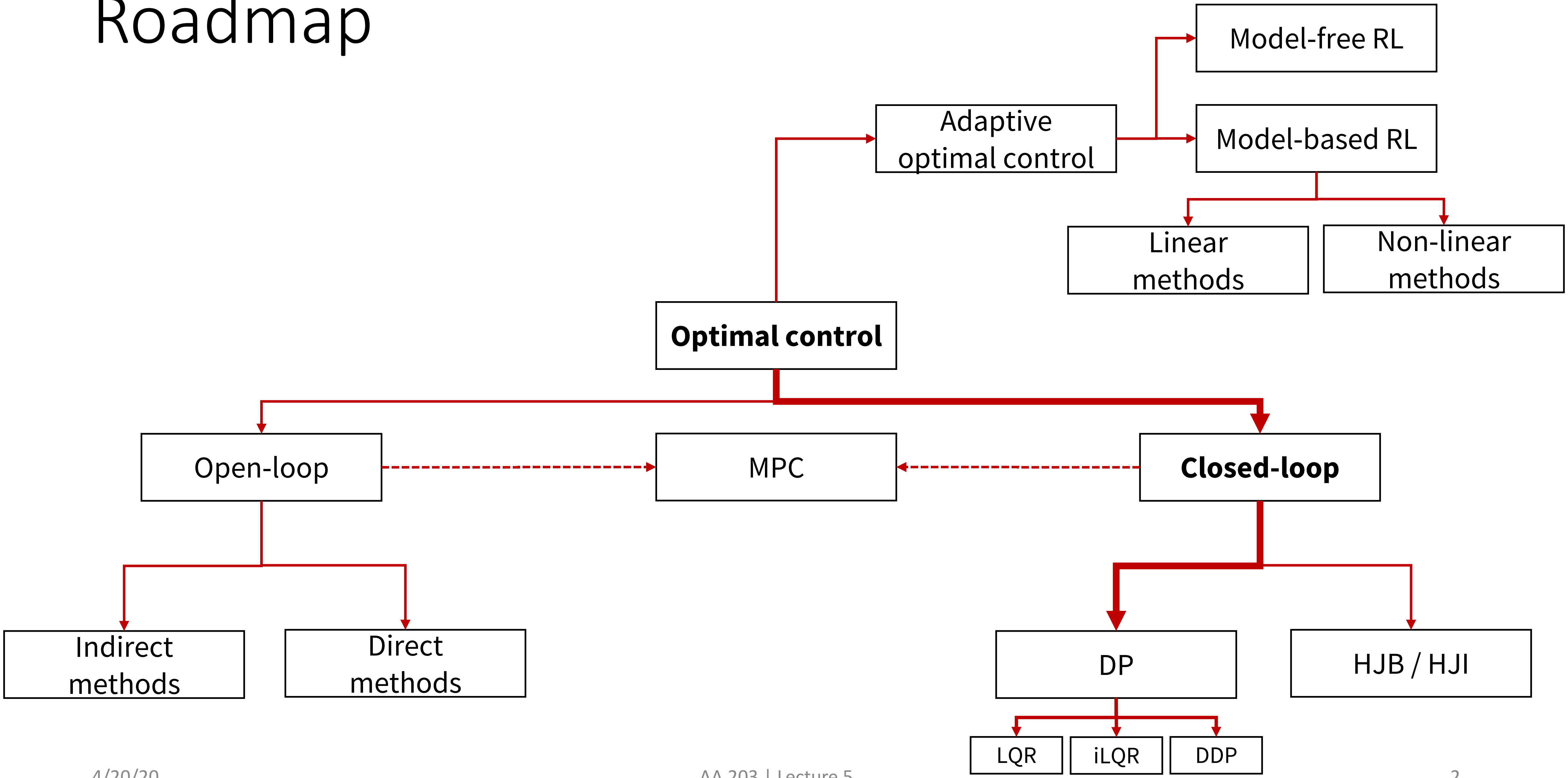
AA203
Optimal and Learning-based Control

Iterative LQR and differential dynamic programming

Stanford University

Autonomous Systems Lab

Roadmap



Model-free RL
Adaptive optimal control
Model-based RL
Linear methods
Non-linear methods
Optimal control
Open-loop
MPC
Closed-loop
Indirect methods
Direct methods
DP
HJB / HJI
LQR
iLQR
DDP

LQR-style algos for optimal control

- Linear tracking problems
- LQR with cross-quadratic cost and affine dynamics
- Non-linear tracking problems
- Using LQR techniques to solve non-linear optimal control problems
  - Iterative LQR
  - Differential dynamic programming

- Readings: lecture notes and references therein

Linear tracking problems

- Imagine you are given a *nominal trajectory*

$$(\bar{\boldsymbol{x}}_0, \ldots, \bar{\boldsymbol{x}}_N), (\bar{\boldsymbol{u}}_0, \ldots, \bar{\boldsymbol{u}}_{N-1})$$

- Assume nominal trajectory satisfies linear dynamics
- Linear tracking problem: find policy to minimize cost

$$\frac{1}{2}\left(\boldsymbol{x}_N - \bar{\boldsymbol{x}}_N\right)^T H(\boldsymbol{x}_N - \bar{\boldsymbol{x}}_N) + \frac{1}{2}\sum_{k=0}^{N-1}\left[\left(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k\right)^T Q(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k) + \left(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k\right)^T R(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)\right]$$

- Then define dev*iation variables*

$$\delta\boldsymbol{x}_k := \boldsymbol{x}_k - \bar{\boldsymbol{x}}_k \text{ and } \delta\boldsymbol{u}_k := \boldsymbol{u}_k - \bar{\boldsymbol{u}}_k$$

and solve standard LQR with respect to deviation variables

LQR with cross-quadratic cost & affine dynamics

- Consider the LQR problem with the generalized cost

$$\frac{1}{2}\boldsymbol{x}_k^T Q_k \boldsymbol{x}_k + \frac{1}{2}\boldsymbol{u}_k^T R_k \boldsymbol{u}_k + \boldsymbol{u}_k^T H_k \boldsymbol{x}_k + \boldsymbol{q}_k^T \boldsymbol{x}_k + \boldsymbol{r}_k^T \boldsymbol{u}_k + c_k$$

  and dynamics

$$\boldsymbol{x}_{k+1} = A_k \boldsymbol{x}_k + B_k \boldsymbol{u}_k + \boldsymbol{d}_k$$

- We can derive an *affine* optimal feedback law for this system via DP recursion

LQR with cross-quadratic cost & affine dynamics

- The cost-to-go at time k takes the form

$$\frac{1}{2}\boldsymbol{x}_k^T P_k \boldsymbol{x}_k + \boldsymbol{p}_k^T \boldsymbol{x}_k + p_k$$

- Optimal control takes the form $\boldsymbol{u}_k^* = \boldsymbol{l}_k + L_k \boldsymbol{x}_k$ with

$$\boldsymbol{l}_k = -(R_k + B_k^T P_{k+1} B_k)^{-1}(\boldsymbol{r}_k + \boldsymbol{p}_{k+1}^T B_k + \boldsymbol{d}_k P_{k+1} B_k)$$

$$L_k = -(R_k + B_k^T P_{k+1} B_k)^{-1}(B_k^T P_{k+1} A_k + H_k)$$

- Equations for the constant/linear/quadratic cost-to-go terms are unwieldy but not hard to derive, and are given in the lecture notes

Nonlinear tracking problems

- Imagine you are given a *feasible nominal trajectory*

$$(\bar{\boldsymbol{x}}_0, \dots, \bar{\boldsymbol{x}}_N), (\bar{\boldsymbol{u}}_0, \dots, \bar{\boldsymbol{u}}_{N-1})$$

- The tracking cost is still quadratic, but the dynamics are now nonlinear

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k)$$

- To apply LQR, we can linearize around the nominal trajectory

$$\boldsymbol{x}_{k+1} \approx f(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k) + \underbrace{\frac{\partial f}{\partial \boldsymbol{x}}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k)}_{A}\overbrace{(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k)}^{\delta\bar{\boldsymbol{x}}_k} + \underbrace{\frac{\partial f}{\partial \boldsymbol{u}}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k)}_{B}\overbrace{(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)}^{\delta\bar{\boldsymbol{u}}_k}$$

- And apply LQR to the deviation variables (with dynamics $\delta\bar{\boldsymbol{x}}_{k+1} = A\delta\bar{\boldsymbol{x}}_k + B\delta\bar{\boldsymbol{u}}_k$)

Non-linear optimal control problem

- Consider now non-linear optimal control problem

$$\min_{\mathbf{u}} \sum_{k=0}^{N-1} c(\mathbf{x}_k, \mathbf{u}_k)$$
$$\text{subject to } \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k)$$

- Can we apply LQR-techniques to approximately solve it?

Iterative LQR

- Imagine you are given a *feasible nominal trajectory*

$$(\bar{\boldsymbol{x}}_0, \dots, \bar{\boldsymbol{x}}_N), (\bar{\boldsymbol{u}}_0, \dots, \bar{\boldsymbol{u}}_{N-1})$$

- Linearize the dynamics around feasible trajectory

$$\boldsymbol{x}_{k+1} \approx f(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k) + \frac{\partial f}{\partial \boldsymbol{x}}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k)(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k) + \frac{\partial f}{\partial \boldsymbol{u}}(\bar{\boldsymbol{x}}_k, \bar{\boldsymbol{u}}_k)(\boldsymbol{u}_k - \bar{\boldsymbol{u}}_k)$$

- And Taylor expand cost function around feasible trajectory

$$c(\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k) = c_k + \underbrace{c_{\boldsymbol{x},k}^T}_{\boldsymbol{q}_k} \delta \boldsymbol{x}_k + \underbrace{c_{\boldsymbol{u},k}^T}_{\boldsymbol{r}_k} \delta \boldsymbol{u}_k + \frac{1}{2} \delta \boldsymbol{u}_k^T \underbrace{c_{\boldsymbol{uu},k}^T}_{R_k} \delta \boldsymbol{u}_k + \frac{1}{2} \delta \boldsymbol{x}_k^T \underbrace{c_{\boldsymbol{xx},k}^T}_{Q_k} \delta \boldsymbol{x}_k + \delta \boldsymbol{u}_k^T \underbrace{c_{\boldsymbol{ux},k}^T}_{H_k} \delta \boldsymbol{x}_k$$

Iterative LQR

- By optimizing over deviation variables (using results for LQR with cross-quadratic cost & affine dynamics), we obtain new solution:

$$\{\bar{\boldsymbol{x}}_k + \delta \boldsymbol{x}_k^*\} \text{ and } \{\bar{\boldsymbol{u}}_k + \delta \boldsymbol{u}_k^*\}$$

- We can then re-linearize and Taylor expand around this new trajectory, and iterate!

Iterative LQR

- Backward pass ($k = N$ to 0):
  - Compute locally linear dynamics, locally quadratic cost around nominal trajectory
  - Solve local approximation of DP recursion to compute control law
  - Compute cost-to-go
- Forward pass ($k = 0$ to N):
  - Use control law to update nominal trajectory
- Iterate until convergence

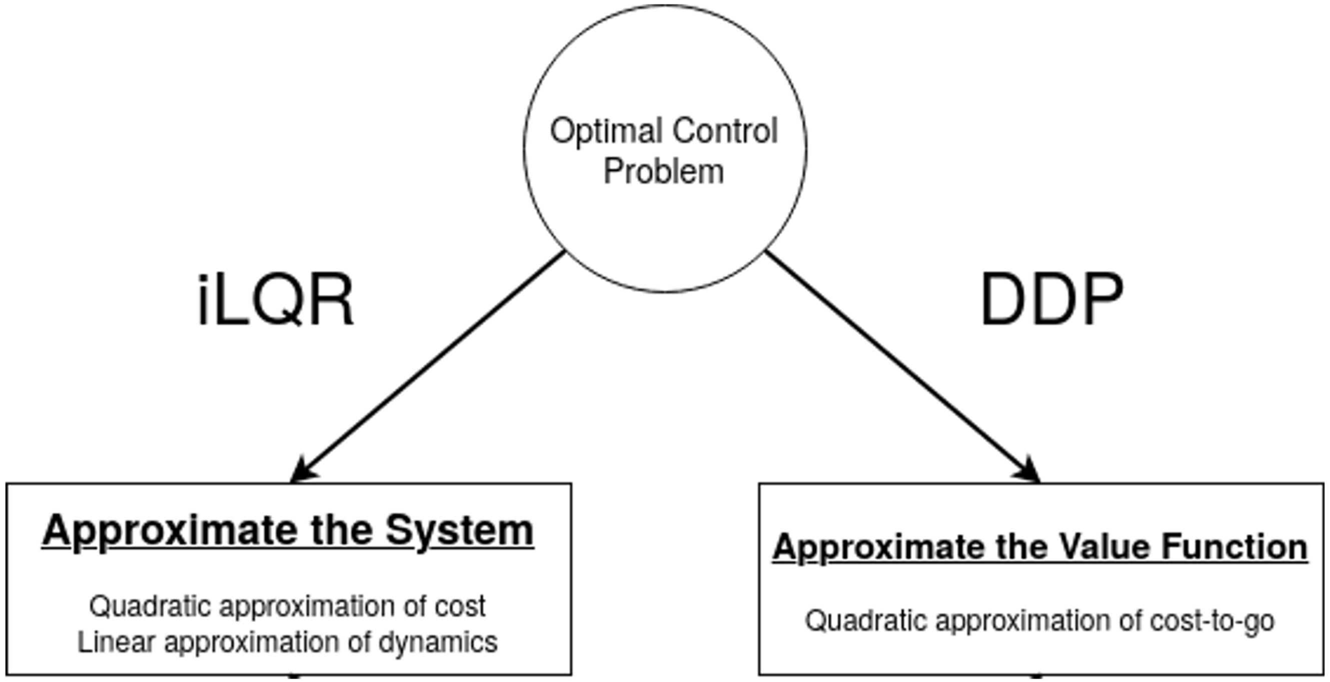
Algorithmic details

- Need to make sure that the new state / control stay close to the linearization point
  - Add extra penalty on deviations
- Need to decide on termination criterion
  - For example, one can stop when improvement is “small”
- Method can get stuck in local minima → “good” initialization is often critical
- Cost matrices may not be positive definite
  - Regularize them until they are

Differential Dynamic Programming (DDP)

- iLQR first approximates dynamics and cost, then performs exact DP recursion

- DDP instead approximates DP recursion directly



Differential Dynamic Programming(DDP)

In detail, define the change in cost to go at timestep *k* under a perturbation $(\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k)$ as:

$$Q_k(\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k) := c(\overline{\boldsymbol{x}}_k + \delta \boldsymbol{x}_k, \overline{\boldsymbol{u}}_k + \delta \boldsymbol{u}_k) + J_{k+1}(f(\overline{\boldsymbol{x}}_k + \delta \boldsymbol{x}_k, \overline{\boldsymbol{u}}_k + \delta \boldsymbol{u}_k))$$

Using a 2nd order Taylor Expansion,

$$Q_k(\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k) \approx Q_k(0,0) + \underbrace{\nabla Q_k^\top (\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k)}_{\text{first order terms}} + \underbrace{\frac{1}{2}(\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k)^\top \nabla^2 Q_k (\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k)}_{\text{second order terms}}$$

Differential Dynamic Programming(DDP)

The optimal control perturbation is

$$\delta \boldsymbol{u}_k^* = \mathrm{argmin}_{\delta \boldsymbol{u}} Q(\delta \boldsymbol{x}_k, \delta \boldsymbol{u})$$

Expanding the approximation, one gets

$$Q_k(\delta \boldsymbol{x}_k, \delta \boldsymbol{u}_k) \approx Q_k(0,0) + \underbrace{Q_{x,k}^\top \delta \boldsymbol{x}_k + Q_{u,k}^\top \delta \boldsymbol{u}_k}_{\text{first order terms}}$$

$$+ \underbrace{\frac{1}{2}\delta \boldsymbol{x}_k^\top Q_{xx,k} \delta \boldsymbol{x}_k + \frac{1}{2}\delta \boldsymbol{u}_k^\top Q_{uu,k} \delta \boldsymbol{u}_k + \delta \boldsymbol{x}_k^\top Q_{xu,k} \delta \boldsymbol{u}_k}_{\text{second order terms}}$$

Differential Dynamic Programming(DDP)

Find where the derivative is equal to zero:

$$Q_{u,k} + Q_{ux,k}\delta \boldsymbol{x}_k + Q_{uu,k}\delta \boldsymbol{u}_k = 0$$

$$\implies \delta \boldsymbol{u}_k^* = -Q_{uu,k}^{-1} Q_{u,k} - Q_{uu,k}^{-1} Q_{ux,k}\delta \boldsymbol{x}_k$$

As was the case with LQR, the optimal control has the form

$$\delta \boldsymbol{u}_k^* = \boldsymbol{l}_k + L_k \delta \boldsymbol{x}_k$$

Algorithm proceeds via same forward/backward passes as iLQR

Next time

- Intro to RL