Advanced Robotics CS 287

Review by Geonhee Lee

Lecture 5: LQR, iterative LQR / Differential Dynamic Programming

Bellman's Curse of Dimensionality

- N-차원 state space
- N만큼 기하급수적으로 state의 수가 증가(좌표계당 descretization 레벨의 고정된 수)
- 실제로
  - 아래의 방법을 사용할 때, discretization은 5 혹은 6차원 state space까지만 계산적으로 실현 가능성이 고려된다.
    - Variable resolution discretization
    - Highly optimized implementations

This Lecture

- Linear Dynamical System 및 Quadratic cost(aka LQ setting, or LQR setting)에 대한 optimal control
  - 매우 특별한 경우: continous state-space optimal control 문제를 정확하게 풀 수 있고 linear algebra 연산만 요구한다.
  - Running time: $O(H n^3)$

Note 1: 좋은 참조 자료 Anderson and Moore, Linear Quadratic Methods
Note 2: Kalman filtering과 상당히 유사하며, 비록 일반적으로 closed form 솔루션이 아니거나 차원성을 좋지 않은 수치적 해이긴 하다.

Linear Quadratic Regulator(LQR)

- LQR setting는 아래와 같은 linear dynamic system이라 가정한다

$$x_{t+1} = Ax_t + Bu_t,$$

x_t: state at time t
u_t: input at time t

- Quadratic cost function:

$$g(x_t, u_t) = x_t^\top Q x_t + u_t^\top R u_t$$

with $Q \succ 0, R \succ 0$.

  - Square matrix X에 대해 X > 0인 것은 if and only if
    - 모든 vector z에 대해 $z^T X z > 0$.
  - 따라서, all-zeros state과 다르고, all-zero input과 다른 어떤 input에 대해서 non-zero cost.

Value Iteration

- i+1 step에 대한 Back-up

$$J_{i+1}(s) = \min_u g(s,u) + \sum_{s'} P(s'|s,u) J_i(s')$$

- LQR:

$$J_{i+1}(x) = \min_u x^\top Q x + u^\top R u + \sum_{x'=Ax+Bu} J_i(x')$$
$$= \min_u \left[x^\top Q x + u^\top R u + J_i(Ax+Bu)\right]$$

LQR value iteration: J_1

$$J_{i+1}(x) \leftarrow \min_u \left[x^\top Q x + u^\top R u + J_i(Ax+Bu)\right]$$

- $J_0(x) = x^T P_0 x$ 초기화

$$\begin{aligned} J_1(x) &= \min_u \left[x^\top Q x + u^\top R u + J_0(Ax+Bu)\right] \\ &= \min_u \left[x^\top Q x + u^\top R u + (Ax+Bu)^\top P_0 (Ax+Bu)\right] \quad (1) \end{aligned}$$

  - u에 대해 최솟값을 찾기 위해서 u에 관한 gradient를 0으로 초기화:

$$\nabla_u [\ldots] = 2Ru + 2B^\top P_0 (Ax+Bu) = 0,$$
$$\text{hence: } u = -(R + B^\top P_0 B)^{-1} B^\top P_0 A x \quad (2)$$

  - (2)를 (1)에 대입:

$$\begin{aligned} :\ J_1(x) &= x^\top P_1 x \\ \text{for: } P_1 &= Q + K_1^\top R K_1 + (A + BK_1)^\top P_0 (A + BK_1) \\ K_1 &= -(R + B^\top P_0 B)^{-1} B^\top P_0 A. \end{aligned}$$

- 요약:

$$J_0(x) = x^\top P_0 x$$
$$x_{t+1} = Ax_t + Bu_t$$
$$g(x,u) = u^\top R u + x^\top Q x$$

$$\begin{aligned} J_1(x) &= x^\top P_1 x \\ \text{for: } P_1 &= Q + K_1^\top R K_1 + (A + BK_1)^\top P_0 (A + BK_1) \\ K_1 &= -(R + B^\top P_0 B)^{-1} B^\top P_0 A. \end{aligned}$$

- $J_1(x)$은 $J_0(x)$와 마찬가지로 quadratic.
  - Value iteration update는 모든 시간에서 동일하고 이 특정한 continous state-space system 및 cost에 대해서는 closed form으로 수행될 수 있다.

$$\begin{aligned} J_2(x) &= x^\top P_2 x \\ \text{for: } P_2 &= Q + K_2^\top R K_2 + (A + BK_2)^\top P_1 (A + BK_2) \\ K_2 &= -(R + B^\top P_1 B)^{-1} B^\top P_1 A. \end{aligned}$$

Value iteration solution to LQR

Set $P_0 = 0$.
for $i = 1, 2, 3, \ldots$

$$\begin{aligned} K_i &= -(R + B^\top P_{i-1} B)^{-1} B^\top P_{i-1} A \\ P_i &= Q + K_i^\top R K_i + (A + BK_i)^\top P_{i-1} (A + BK_i) \end{aligned}$$

The optimal policy for a i-step horizon is given by:

$$\pi(x) = K_i x$$

The cost-to-go function for a i-step horizon is given by:

$$J_i(x) = x^\top P_i x.$$

- Fact: infinite horizon optimal control에 대해 수렴성을 증명하는 것은 if and only if
  - Dynamics (A, B)가 state를 0(zero)로 이동할 수 있는 정책이 존재하는 것을 충족(controllability?).
- 종종 모든 시간에 대해서 steady-state K를 사용하는 것이 매우 편리.

LQR assumptrions revisited

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t \\ g(x_t, u_t) &= x_t^\top Q x_t + u_t^\top R u_t \end{aligned}$$

= control 입력을 작게 유지하는 것을 선호하면서 선형시스템을 all-zero state로 유지하기 위해서

- Extensions은 더욱 일반적으로 응용가능하게 만든다(위의 state-space는 linear, time invariant이므로 특수 케이스에 해당).
  - Affine systems
  - Systems with stochasticity
  - Regulation around non-zero fixed point for non-linear systems
  - Penalization for change in control inputs
  - Linear time varying (LTV) systems
  - Trajectory following for non-linear systems

LQR Extension 0: Affine systems

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + c \\ g(x_t, u_t) &= x_t^\top Q x_t + u_t^\top R u_t \end{aligned}$$

- Optimal control policy는 linear하게 남아있고, optimal cost-to-go fucntion은 quadratic으로 남아있다.
  - c가 추가
- 유도하기 위한 두 가지 방안(avenues)
  - 1. 기본적인 설정에 대해 했던 것과 매우 유사한 update를 재유도(re-derive).
  - 2. state를 $z_t = [X_t; 1]$로 재정의하고, 다음을 갖는다.

$$z_{t+1} = \begin{bmatrix} x_{t+1} \\ 1 \end{bmatrix} = \begin{bmatrix} A & c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ 1 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_t = A' z_t + B' u_t$$

LQR Extension 1: Stochastic systems

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + w_t \\ g(x_t, u_t) &= x_t^\top Q x_t + u_t^\top R u_t \\ & w_t, t = 0, 1, \ldots \text{ are zero mean and independent} \end{aligned}$$

- Exercise: deterministic case에 대해진행했던 것 처럼 유사한 유도를 통해 진행.
  - 노이즈(w) 추가
- 결과:
  - 같은 optimal control policy
  - Cost-to-go 함수는 대부분 동일: noiso에서 variance에 의존하는 추가적인 term을 가진다(그리고 제어 입력의 선택에 의해서 영향을 받을 수 없다)

LQR Extension 2: non-linear systems

- Nonlinear system:

$$x_{t+1} = f(x_t, u_t)$$

- state x^*에서 system을 유지할 수 있다 iff

$$\exists u^* \text{s.t.} \quad x^* = f(x^*, u^*)$$

- x^* 주변에서 dynamics를 선형화하는 것(Talor series)은 다음과 같다:

$$x_{t+1} \approx f(x^*, u^*) + \underbrace{\frac{\partial f}{\partial x}(x^*, u^*)}_{A}(x_t - x^*) + \underbrace{\frac{\partial f}{\partial u}(x^*, u^*)}_{B}(u_t - u^*)$$

Equivalently:

$$x_{t+1} - x^* \approx A(x_t - x^*) + B(u_t - u^*)$$

- $z_t = x_t - x^*$, $v_t = u_t - u^*$이라면, 다음과 같다:

$$z_{t+1} = Az_t + Bv_t, \qquad \text{cost} = z_t^\top Q z_t + v_t^\top R v_t \quad \text{[=standard LQR]}$$
$$v_t = Kz_t \Rightarrow u_t - u^* = K(x_t - x^*) \Rightarrow u_t = u^* + K(x_t - x^*)$$

LQR Extension 3: Penalize for Change in Control Inputs

- Standard LQR:

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t \\ g(x_t, u_t) &= x_t^\top Q x_t + u_t^\top R u_t \end{aligned}$$

- 실제 시스템에서 위와 같은 형태를 이용할 때, 자주 high frequency control input들이 생성된다. 일반적으로 매우 좋지 않으며 좋지 않은 제어 성능을 도출한다.
- Why?
- **Solution**: cost function의 frequency shaping. filter를 시스템을 추가하여 구동할 수 있으며, filter output은 quadratic cost에서 사용될 수 있다.(see, e.g., Anderson and Moore.)
  - 실전에서 잘동작하는 간단한 특별한 경우: 제어 입력의 변화에 대해 페널티를 제공.

- 제어의 변화를 cost/ reward function에 어떻게 병합?
  - Soln. method A: 과거의 제어 입력 벡터, 그리고 마지막 두 개의 제어 입력 벡터 차이를 가지고 state를 추가하여 state를 explicitly 병합 .
  - Soln. method B: 제어 입력 변수를 변경.

- 제어 Δu의 변화 소개

$$\underbrace{\begin{bmatrix} x_{t+1} \\ u_t \end{bmatrix}}_{x'_{t+1}} = \underbrace{\begin{bmatrix} A & B \\ 0 & I \end{bmatrix}}_{A'} \underbrace{\begin{bmatrix} x_t \\ u_{t-1} \end{bmatrix}}_{x'_t} + \underbrace{\begin{bmatrix} B \\ I \end{bmatrix}}_{B'} \underbrace{\Delta u_t}_{u'_t}$$

$$\text{cost} = -(x'^\top Q' x' + \Delta u^\top R' \Delta u) \qquad Q' = \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix}$$

R' = penalty for change in controls

[If R'=0, then "equivalent" to standard LQR.]

LQR Extension 4: Linear Time Varying (LTV) Systems

$$\begin{aligned} x_{t+1} &= A_t x_t + B_t u_t \\ g(x_t, u_t) &= x_t^\top Q_t x_t + u_t^\top R_t u_t \end{aligned}$$

Set $P_0 = 0$.
for $i = 1, 2, 3, \ldots$

$$\begin{aligned} K_i &= -(R_{H-i} + B_{H-i}^\top P_{i-1} B_{H-i})^{-1} B_{H-i}^\top P_{i-1} A_{H-i} \\ P_i &= Q_{H-i} + K_i^\top R_{H-i} K_i + (A_{H-i} + B_{H-i} K_i)^\top P_{i-1} (A_{H-i} + B_{H-i} K_i) \end{aligned}$$

The optimal policy for a i-step horizon is given by:

$$\pi(x) = K_i x$$

The cost-to-go function for a i-step horizon is given by:

$$J_i(x) = x^\top P_i x.$$

LQR Extension 5: Trajectory Following for Non-Linear Systems

Regulation control이 아닌 tracking control.

- State sequence $x_0^*, x_1^*, \ldots, x_H^*$는 feasible target trajectory iff

$$\exists u_0^*, u_1^*, \ldots, u_{H-1}^* : \forall t \in \{0, 1, \ldots, H-1\} : x_{t+1}^* = f(x_t^*, u_t^*)$$

- Problem statement

$$\min_{u_0, u_1, \ldots, u_{H-1}} \sum_{t=0}^{H-1} (x_t - x_t^*)^\top Q (x_t - x_t^*) + (u_t - u_t^*)^\top R (u_t - u_t^*)$$
$$\text{s.t.} \quad x_{t+1} = f(x_t, u_t)$$

- Linear time varying case(LTV)로 변환:

$$x_{t+1} \approx f(x_t^*, u_t^*) + \underbrace{\frac{\partial f}{\partial x}(x_t^*, u_t^*)}_{A_t}(x_t - x_t^*) + \underbrace{\frac{\partial f}{\partial u}(x_t^*, u_t^*)}_{B_t}(u_t - u_t^*)$$
$$x_{t+1} - x_{t+1}^* \approx A_t(x_t - x_t^*) + B_t(u_t - u_t^*)$$

- 이제 standard LQR back-up iteration을 실행할 수 있다.
- 마지막에서부터 i time-step에서의 최종적인 policy:

$$u_{H-i} - u_{H-i}^* = K_i (x_{H-i} - x_{H-i}^*)$$

- 만약이 target trajectory은 이 기술을 적용하기 위한 feasible을 필요로 하지는 않지만, infeasible하다면 linearization은 방문될(visited) (state, input) pair 주위는 아니다.

Most General Cases

- 일반적인(Generic) optimal control 문제를 풀기 위한 방법

$$\min_u \sum_{t=0}^{H} g(x_t, u_t)$$
$$\text{subject to} \quad x_{t+1} = f(x_t, u_t) \ \forall t$$

  - 반복적으로 근사함으로써 linear quadratic formulation이 풀기 싶다는 이점이 있다.

Iteratively Apply LQR

- (a) A control policy $\pi^{(0)}$ 혹은(b) A sequence of states $x_0^{(0)}, x_1^{(0)}, \ldots, x_H^{(0)}$과 control input 중에 하나를 선택하여 알고리즘을 초기화한다
- (a)를 초기화하면 Step (1)에서 부터 시작. (b)를 초기화하면 Step (2)에서 시작한다.

- Iterate the following:
  1. curren policy $\pi^{(i)}$ 를 실행하고 결과 state-input trajectory $x_0^{(i)}, u_0^{(i)}, \ldots, x_H^{(i)}, u_H^{(i)}$를 기록한다.
  2. Dynamic 모델의 1차 Tayer expansion, 그리고 cost function의 2차 Taylor expansion을 계산하여 위에서 획득한 state, input trajectory에 대해 optimal control 문제의 LQ approximation을 계산.
  3. Step (2)에서 얻은 LQ approximation에 대해 optimal control policy $\pi^{(i+1)}$ 에 대해 풀기 위해 LQR back-ups를 사용
  4. i = i + 1을 설정하고 Step (1)로 가서 반복.

Iterative LQR in Standard LTV Format

Standard LTV is of the form $z_{t+1} = A_t z_t + B_t v_t$, $g(z, v) = z^\top Q z + v^\top R v$. Linearizing around $(x_t^{(i)}, u_t^{(i)})$ in iteration i of the iterative LQR algorithm gives us (up to first order!):

$$x_{t+1} = f(x_t^{(i)}, u_t^{(i)}) + \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)})(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Subtracting the same term on both sides gives the format we want:

$$x_{t+1} - x_{t+1}^{(i)} = f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)} + \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)})(x_t - x_t^{(i)}) + \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)})(u_t - u_t^{(i)})$$

Hence we get the standard format if using:

$$\begin{aligned} z_t &= [x_t - x_t^{(i)} \quad 1]^\top \\ v_t &= (u_t - u_t^{(i)}) \\ A_t &= \begin{bmatrix} \frac{\partial f}{\partial x}(x_t^{(i)}, u_t^{(i)}) & f(x_t^{(i)}, u_t^{(i)}) - x_{t+1}^{(i)} \\ 0 & 1 \end{bmatrix} \\ B_t &= \begin{bmatrix} \frac{\partial f}{\partial u}(x_t^{(i)}, u_t^{(i)}) \\ 0 \end{bmatrix} \end{aligned}$$

A similar derivation is needed to find Q and R.

Iteratively Apply LQR: Convergence

- 공식화 된 것처럼 수렴할 필요가 없다!
  - 이유: LQ approximation의 optimal policy는 LQ Approximation이 Taylor expansion에 의해 계산된 지점(point)들의 sequence에 가까이 있지 않을 수 있다.

  - Solution: 각각 iteration에서, cost function을 조정하여, 즉 이경우에는 아래와 같은 cost function을 사용:

$$(1-\alpha) g(x_t, u_t) + \alpha(\|x_t - x_t^{(i)}\|_2^2 + \|u_t - u_t^{(i)}\|_2^2)$$

    - g가 bouded라고 가정하고, α가 1에 충분히 가깝다면, 두 번째 term은 우세(dominate)할 것이며 linearization이 LQR에 의해 찾아진 solution trajectory에 근방에 대해 좋은 근사인 확인 할 수 있다.

Iteratively Apply LQR: Practicalities

- f는 non-linear이므로, non-convex optimization 문제이다. local optima에 stuck 될 수 있으며, 좋은 초기값(initialization)이 필요.

- g는 non-convex가 될 수 있다: 그러면 LQ 근사는 positive-definite cost 행렬을 가질 수 없다.
  - Practical fix: 만약 Q_t 혹은 R_t 가 positive definite가 아니라면 → 결과 Q_t 와 R_t가 positive definite 일때까지, 현재 state와 input $(x^{(i)}_t, u^{(i)}_t)$에서 벗어나는 penalty가 증가한다.

Differential Dynamic programming(DDP)

- 종종 느슨하게 반복적인 LQR 절차를 참조하기 위해 사용된다.
- 보다 정확하게: 직접적으로 Bellman back-up 방정식의 2차 Taylor expansion을 수행 [Dynamics를 선형화하고 cost를 2차 근사보다 정확]
- iterative LQR 접근에서 버려진 back-up equation에서 term을 유지한다는 것이 드러남.
- [Dynamic model에서 quadratic term이라면, cost가 convex이라도 결과 LQ problem은 non-convex가 된다.

[Reference: Jacobson and Mayne, "DifferenBal dynamic programming," 1970]

- scale 제어 입력 u 경우를 고려하자:

DDP

$$\begin{aligned} J_i(f(x,u)) \approx\ & J_i(f(x,\bar{u})) \\ & + J_i'(f(x,\bar{u})) f_u(x,\bar{u})(u - \bar{u}) \\ & + J_i''(f(x,\bar{u})) f_u(x,\bar{u}) f_u(x,\bar{u})(u - \bar{u})^2 \\ & + \boxed{J_i'(f(x,\bar{u})) f_{uu}(x,\bar{u})(u - \bar{u})^2} \end{aligned}$$

Iterative LQR

$$\begin{aligned} J_i(x_{t+1}) \approx\ & J_i(\bar{x}_{t+1}) \\ & + J_i'(\bar{x}_{t+1})(x_{t+1} - \bar{x}_{t+1}) \\ & + \frac{1}{2} J_i''(\bar{x}_{t+1})(x_{t+1} - \bar{x}_{t+1})^2 \\ \approx\ & J_i(\bar{x}_{t+1}) \\ & + J_i'(f(\bar{x}_{t+1}) f_u(x,\bar{u})(u - \bar{u}) \\ & + \frac{1}{2} J_i''(\bar{x}_{t+1}) (f_u(x,\bar{u})(u - \bar{u}))^2 \end{aligned}$$

$$x_{t+1} = f(x,u)$$
$$\bar{x}_{t+1} = f(x,\bar{u})$$

Can We Do Even Better(더 좋은 방법)?

- YES!
- iLQR(iterative LQR) 과 DDP의 수렴지점에서, 알고리즘이 수렴되는 (state, input) trajectory에 근방에서 linearization으로 마무리.

- 실전: perturbation/ initial state being off/ dynamics model being off 때문에 시스템이 이 trajectory에 있을 수 없을 것이다
- Solution: 제어 입력 u_t를 생성하기 위해서 요청하는 time t에서, H동안 time step t에 걸쳐 iLQR 혹은 DDP를 사용하여 control 문제를 재해결 할 수 있다.
- 전체 경로를 재설계하는 것은 종종 실용적이지 않다 → 실전: horizon h에 걸쳐 재설계.
  - = receding horizon control
    - 이것은 모든 미래 cost를 설명하는 cost to go J(t+h)를 제공하는 것을 요구한다. 이것은 offline iLQR이나 DDP 실행에서 가져올 수 있다.

Multiplicative Noise

- 관심있는 많은 시스템에서, 시스템으로 제어 입력에 곱해지는(추가되는) 노이즈가 존재, i.e.:

$$x_{t+1} = Ax_t + (B + B_w w_t) u_t$$

- Exercise: LQR derivation for this setting

[optional related reading:Todorov and Jordan, nips 2003]

Lyapunov's linearization method

controller를 설계하면, autonomous system, $x_{t+1} = f(x_t)$을 얻는다

Defn. x^*는 만약 $\|x - x^*\| \le \varepsilon$을 만족하는 모든 초기 state x에 대해 만족하는 $\varepsilon > 0$이 존재한다면, asymptotically stable equilibrium point.

자세히 다루지는 않는다, 그러나 다음은 기본적인 결과:

- x*가 f(x)에 대해서 equilibrium point라고 가정하자, i.e., x* = f(x*)
- 만약 x*가 선형화된 시스템에 대한 asymptotically stable equilibrium point라면, non-linear system에대해 asymptotically stable이다.
- 만약 x*가 linear 시스템에 대해 unstable이라면, non-linear system에대해 unstable.
- 만약 x*가 linear 시스템에 대해 marginally stable이라면, 결과는 이끌어 낼 수 없다
  - 성형 제어 설계 기법에 대한 추가적인 증명

Controllability

- A system is t-time-steps controllable if from any start state, x_0, we can reach any target state, x^*, at time t.

- For a linear time-invariant systems, we have:

$$x_t = A^t x_0 + A^{t-1} B u_0 + A^{t-2} B u_1 + \ldots + AB u_{t-2} + B u_{t-1}$$

hence the system is t-time-steps controllable if and only if the above linear system of equations in $u_0, \ldots, u_{t-1}$ has a solution for all choices of x_0 and x_t. This is the case if and only if

$$\text{rank}\left[A^{t-1}B \quad A^{t-2}B \quad \cdots \quad A^2B \quad AB \quad B\right] = n$$

The Cayley-Hamilton theorem says that for all A, for all t, n : $\exists w \in \mathbb{R}^n, \quad A^t = \sum_{i=0}^{n-1} w_i A^i$

Hence we obtain that the system (A,B) is controllable for all times t>=n, if and only if

$$\text{rank}\left[A^{n-1}B \quad A^{n-2}B \quad \cdots \quad A^2B \quad AB \quad B\right] = n$$

Feedback Linearization

Consider system of the form:

$$\dot{x} = f(x) + g(x)u$$

If $g(x)$ is square (i.e., number of control inputs = number of state variables) and it is invertible, then we can linearize the system by a change of input variables:

$$v = f(x) + g(x)u$$

gives us:

$$\dot{x} = v$$

Prototypical example: fully actuated manipulators:

$$H(q)\ddot{q} + b(q, \dot{q}) + g(q) = \tau$$

Feedback linearize by using the following transformed input:

$$v = H^{-1}(q)(\tau - g(q) - b(q, \dot{q}))$$

which results in

$$\ddot{q} = v$$

Lagrangian Dynamics

- Newton: F = ma
  - 꽤 일반적으로 응용.
  - 이 이용은 constraint//internal force를 가진 multi-body 시스템에서 꽤 다루기 힘들 것.

- Lagrangian dynamics
  - 처음부터 internal force를 제거하고 system의 DOF에 관해서 dynamics를 표현.

- r_i: generalized coordinates
- T: total kinetic energy
- U: total potential energy
- Q_i : generalized forces $\quad Q_i = \sum_j F_j \frac{dr_i}{dq_j}$
- Lagrangian L = T – U

→ Lagrangian dynamic equations: $\quad \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i$

[Nice reference: Goldstein, Poole and Satko, "Classical Mechanics"]

[Reference]

1. [CS287 - Lecture5](https://people.eecs.berkeley.edu/~pabbeel/cs287-fa15/slides/lecture5-LQR.pdf)
2. [EE363](http://web.stanford.edu/class/ee363/index.html)