Optimal Control, LQR, Trajectory Optimization
What will you take home today?

Intro Optimal Control
  Principle of Optimality
  Bellman Equation
  Deriving LQR
Trajectory Optimization
  Paper
Optimal Control and Reinforcement Learning from a unified point of view

Optimal Control Problem

\[ x_{t+1} = f(x_t, u_t) \rightarrow \text{system dynamics} \]
\[ J \rightarrow \text{cost function} \]

Goal: Find a sequence of control inputs \( u_0, ..., u_T \) that minimizes the cost function \( J \) given the system dynamics

DQR: Assume linear system dynamics & quadratic cost
Principle of Optimality – Example: Graph Search problem
Quiz
Forward Search
Backward Search
Principle of Optimality

If path $ABCDE$ is optimal, then $BCDE$ is optimal for the truncated problem.

In "control" terms: $U_0^*, \ldots, U_n^*, U_{n+1}^*, \ldots U_N^* \Rightarrow U_n^*, U_{n+1}^*, \ldots U_N^*$
Problem setup

System Dynamics

\[ x_{n+1} = f_n(x_n, u_n), \quad n \in \{0, 1, \ldots, N-1\} \]

Cost function

\[ J = \alpha^N \Phi(x_N) + \sum_{k=0}^{N-1} \alpha^k L_k(x_k, u_k) \]

where

- \( n \) is the discrete time index,
- \( x_n \) is the state of the system at time \( n \),
- \( u_n \) is the control input at time \( n \) and
- \( f_n \) is the state transition equation.

Policy

\[ \mu = \{u_0, u_1, \ldots, u_{N-1}\} \]

Decay

\[ 0 \leq \alpha \leq 1 \]

Goal: \( \mu^* \)
Goal

\[ \mu^* = \arg \min_u J \]
Formalize Cost-to-Go / Value function

\[
V^M (n, x) = \alpha^{N-n} \phi (x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} d_k (x_k, u_k)
\]

\[X_n = n\]
\[X_{k+1} = f_k (x_k, u_k) \sim \text{Dynamic model}\]
\[u_k = m (k, x_k)\]
\[V^M (N, x) = \phi (x)\]
Optimal Value function = V with lowest cost

\[ V^*(n, \mathbf{x}) = \min_{\mu} V^\mu(n, \mathbf{x}) \]
\[ = \min_{\mathbf{u}_n, \ldots, \mathbf{u}_{N-1}} \{ \alpha^{N-n} \Phi(\mathbf{x}_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(\mathbf{x}_k, \mathbf{u}_k) \} \]
Deriving the Bellman Equation

\[ V^\mu(n, x) = \alpha^{N-n} \Phi(x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \]

\[ V^M(n, x) = d_n(x, u_n) + \alpha^{N-n} \Phi(x_N) + \sum_{k=n+1}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \]

\[ V^M(n, x) = d_n(x, u_n) + \alpha^{N-n} \Phi(x_N) + \sum_{k=n+1}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \]

Bellman Equation
Comparing Optimal Bellman and Value Function

\[ V^*(n, x) = \min_{\mu} V^\mu(n, x) \]

\[ = \min_{u_n, \ldots, u_{N-1}} \{ \alpha^{N-n} \Phi(x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \} \]

\[ V^*(n, x) = \min_{u_n} [L_n(x, u_n) + \alpha V^*(n + 1, f_n(x, u_n))] \]
Infinite time horizon, deterministic system

\[ J = \alpha^N \Phi(x_N) + \sum_{k=0}^{N-1} \alpha^k L_k(x_k, u_k) \]

\[ J = \sum_{k=0}^{\infty} \alpha^k L(x_k, u_k), \quad \alpha \in [0, 1] \]
Infinite time horizon, deterministic system

\[ V^*(n, x) = \min_{\mu} V^\mu(n, x) \]

\[ = \min_{u_n, \ldots, u_{N-1}} \left\{ \alpha^{N-n} \Phi(x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \right\} \]
Infinite time horizon, deterministic system

\[ x_{t+n} = f_n(x_t, u_t) \]

\[ V^*(n, x) = \min_{u_n} [L_n(x, u_n) + \alpha V^*(n + 1, f_n(x, u_n))] \]

\[ V^*(x) = \min_u \{L(x, u) + \alpha V^*(f(x, u))\}, \]
Finite Horizon, Stochastic system

Stochastic System Dynamics

\[ x_{n+1} = f_n(x_n, u_n) + w_n \quad n \in \{0, 1, ..., N - 1\} \]

where

\( n \) is the discrete time index,

\( x_n \) is the state of the system at time \( n \),

\( u_n \) is the control input at time \( n \) and

\( f_n \) is the state transition equation.

\[ w_n \sim p_w(\cdot | x_n, u_n) \]

Cost function

\[ J = \mathbb{E} \left[ \alpha^N \phi(x_N) + \sum_{k=0}^{N-1} \alpha^k \lambda_k(x_k, u_k) \right] \]
Finite Horizon, Stochastic system

\[
V^\mu(n, x) = E \left[ \alpha^{N-n} \Phi(x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \right]
\]

Optimal Value function

\[
V^*(n, x) = \min_\mu E \left[ \alpha^{N-n} \Phi(x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \right]
\]

Optimal Policy

\[
\mu^* = \arg \min_\mu E \left[ \alpha^{N-n} \Phi(x_N) + \sum_{k=n}^{N-1} \alpha^{k-n} L_k(x_k, u_k) \right]
\]
Finite Horizon, Stochastic system

Bellman Equation

\[
V^\mu(n, x) = L_n(x, u_n) + \alpha \mathbb{E}_{x' \sim P_f(\cdot | x, u_n)} \left[ V^\mu(n + 1, x') \right]
\]

Optimal Bellman Equation

\[
V^*(n, x) = \min_{u_n} \left[ L_n(x, u_n) + \alpha \mathbb{E}_{x' \sim P_f(\cdot | x, u_n)} \left[ V^*(n + 1, x') \right] \right]
\]
Infinite Horizon, Stochastic system

- Combining formulation from infinite horizon - discrete system with stochastic system derivation
- See lecture notes on webpage
Continuous time systems

Hamilton-Jacobi-Bellman Equation
How do you solve these equations?

- Non-linear dynamics model → no analytical solution
- Numerical methods
  - Approximate the complex, full problem with a set of tractable subproblems
  - Iterate
Sequential Quadratic Programming

\[
\begin{align*}
\min_{x} & \quad f(x) \quad x \in \mathbb{R}^n \\
\text{s.t.} & \quad f_j(x) \leq 0 \\
& \quad g_i(x) = 0
\end{align*}
\]

Approximate \( f(x) \) with a second-order Taylor Exp.

\( f_j, g_j \) with a first-order
Sequential Quadratic Programming

\[ \begin{align*}
\min_{x} f(x) \quad & x \in \mathbb{R}^n \\
\text{s.t.} \quad & f_j(x) \leq 0, \quad j = 1, \ldots, N \\
& h_j(x) = 0, \quad j = 1, \ldots, N
\end{align*} \]

\[ f(x) \approx f(\tilde{x}_0) + (x - \tilde{x}_0)^T \nabla f(\tilde{x}_0) + \frac{1}{2} (x - \tilde{x}_0)^T \nabla^2 f(\tilde{x}_0)(x - \tilde{x}_0) \]

\[ f_j(x) \approx f_j(\tilde{x}_0) + (x - \tilde{x}_0)^T \nabla f_j(\tilde{x}_0) \]

\[ h_j(x) \approx h_j(\tilde{x}_0) + (x - \tilde{x}_0)^T \nabla h_j(\tilde{x}_0) \]
Example – Newton-Raphson Method

\[ x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)} \]
Sequential Linear Quadratic Programming

\[
\min_{x_0} \left[ \phi(x(N)) + \sum_{n=0}^{N-1} d_n(x(n), u(n)) \right]
\]

s.t.
\[
x(n+1) = f(x(n), u(n))\]
\[
x(0) = x_0
\]
\[
u(n, x) = \mu(n, x)
\]
1. Guess $\mu^0(n,x)$
2. Roll out using $f(x,u)$
   \[ X_k = d(x(0)...x(N)) \]
   \[ U_k = q(u(0)...u(N)) \]
   \[ \text{Forward pass} \]
3. Approximate value function with a quadratic function $f(x(N-1), u(N-1))$
4. Solve the Bellman Eq.: $u^*(W-1)$ with $J^*$
5. Backward Pass: repeat 3, 4 for every pair along this trajectory
   \[ \sum_{k=1}^{W-1} \nabla u_k \hat{a}_k(x) \cdot \nabla u_k \]
   \[ \mu_{k+1} = \mu_k + \alpha_k \cdot \nabla u_k \]
6. Repeat Step 2-5 with $\mu_{k+1}$
Iterative Linear Quadratic Controller: ILQC

\[ x_{n+1} = f_n(x_n, u_n) \]

\[ J = \phi(x_N) + \sum_{n=0}^{N-1} d_n(x_n, u_n) \]

\[ u = \{u_0, \ldots, u_{N-1}\} \]
ILQC – Linearizing Dynamics

\[ \delta x_n \triangleq x_n - \bar{x}_n \]
\[ \delta u_n \triangleq u_n - \bar{u}_n \]

\[ x_{n+1} + \delta x_{n+1} = f_n(x_n + \delta x_n, u_n + \delta u_n) \]
\[ \approx f_n(x_n, u_n) + \frac{\partial f(x_n, u_n)}{\partial x} \delta x_n + \frac{\partial f(x_n, u_n)}{\partial u} \delta u_n \]

\[ \delta x_{n+1} \approx A_n \delta x_n + B_n \delta u_n \]

\[ A_n = \frac{\partial f(x_n, u_n)}{\partial x} \]
\[ B_n = \frac{\partial f(x_n, u_n)}{\partial u} \]
ILQC - Quadratizing the cost

\[ J \approx q_N + \delta x_N^T q_N + \frac{1}{2} \delta x_N^T Q_N \delta x_N \]

\[ + \sum_{n=0}^{N-1} \{ q_n + \delta x_n^T q_n + \delta u_n^T r_n + \frac{1}{2} \delta x_n^T Q_n \delta x_n + \frac{1}{2} \delta u_n^T R_n \delta u_n + \delta u_n^T P_n \delta x_n \} \]

\( \forall n \in \{0, \ldots, N-1\} : \)

\[ q_n = L_n(\bar{x}_n, \bar{u}_n), \quad q_n = \frac{\partial L(\bar{x}_n, \bar{u}_n)}{\partial x}, \quad Q_n = \frac{\partial^2 L(\bar{x}_n, \bar{u}_n)}{\partial x^2} \]

\[ P_n = \frac{\partial^2 L(\bar{x}_n, \bar{u}_n)}{\partial u \partial x}, \quad r_n = \frac{\partial L(\bar{x}_n, \bar{u}_n)}{\partial u}, \quad R_n = \frac{\partial^2 L(\bar{x}_n, \bar{u}_n)}{\partial u^2} \]

\( n = N : \)

\[ q_N = \Phi(\bar{x}_N), \quad q_N = \frac{\partial \Phi(\bar{x}_N)}{\partial x}, \quad Q_N = \frac{\partial^2 \Phi(\bar{x}_N)}{\partial x^2} \]
Deriving the Value function and Bellman Equation

\[ V^*(n + 1, \delta x_{n+1}) = s_{n+1} + \delta x_{n+1}^T s_{n+1} + \frac{1}{2} \delta x_{n+1}^T S_{n+1} \delta x_{n+1} \]

\[ V^*(n, \delta x_n) = \min_{u_n} [L_n(x_n, u_n) + V^*(n + 1, \delta x_{n+1})] \]
Incremental Updates

\[ \delta u_n = -H_n^{-1}g_n - H_n^{-1}G_n \delta x_n \]

\[ g_n \triangleq r_n + B_n^T s_{n+1} \]

\[ G_n \triangleq P_n + B_n^T S_{n+1} A_n \]

\[ H_n \triangleq R_n + B_n^T S_{n+1} B_n \]

\[ S_n = Q_n + A_n^T S_{n+1} A_n + K_n^T H_n K_n + K_n^T G_n + G_n^T K_n \]

\[ s_n = q_n + A_n^T s_{n+1} + K_n^T H_n \delta u_n^{ff} + K_n^T g_n + G_n^T \delta u_n^{ff} \]

\[ s_n = q_n + s_{n+1} + \frac{1}{2} \delta u_n^{ff}^T H_n \delta u_n^{ff} + \delta u_n^{ff}^T g_n \]

\[ S_N = Q_N, \quad s_N = q_N, \quad s_N = q_N \]
Linear Dynamical Systems, Quadratic cost – Linear Quadratic Regulator (LQR)

\[ x_{n+1} = A_n x_n + B_n u_n \]

\[ J = \frac{1}{2} x_N^T Q x_N x_n + \sum_{n=0}^{N-1} \frac{1}{2} x_n^T Q_n x_n + \frac{1}{2} u_n^T R_n u_n + u_n^T P_n x_n \]

\[ \text{min} J \]
Linear Dynamical Systems, Quadratic cost – Linear Quadratic Regulator (LQR)

\[ S_n = Q_n + A_n^T S_{n+1} A_n - (P_n^T + A_n^T S_{n+1} B_n)(R_n + B_n^T S_{n+1} B_n)^{-1}(P_n + B_n^T S_{n+1} A_n) \]

Discrete time Ricatti Equation

\[ \mu^*(n, x) = -(R_n + B_n^T S_{n+1} B_n)^{-1}(P_n + B_n^T S_{n+1} A_n)x \]
Trajectory Optimization
Motion Planning with Sequential Convex Optimization and Convex Collision Checking

John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, Pieter Abbeel

Abstract—We present a new optimization-based approach for robotic motion planning among obstacles. Like CHOMP, our algorithm can be used to find collision-free trajectories from naive, straight-line initializations that might be in collision. At the core of our approach are (i) a sequential convex optimization procedure, which penalizes collisions with a hinge loss and increases the penalty coefficients in an outer loop as necessary, and (ii) an efficient formulation of the no-collisions constraint that directly considers continuous-time safety. Our algorithm is implemented in a software package called TrajOpt.

We report results from a series of experiments comparing TrajOpt with CHOMP and randomized planners from OMPL, with regard to planning time and path quality. We consider motion planning for 7 DOF robot arms, 18 DOF full-body robots, statically stable walking motion for the 34 DOF Atlas humanoid robot, and physical experiments with the 18 DOF PR2. We also apply TrajOpt to plan curvature-constrained steerable objects in contact with the environment.
The robot has converged to a trajectory without collisions.