5 Iterative Optimization, Maximum Principle, and Two Point Boundary Value Problems

5.1 The Maximum Principle

We have just shown a very, *very* limited version of what is called the Pontryagin Maximum Principle. We know that for the LQ problem

$$\begin{aligned} J &= \int_0^T \frac{1}{2}x^T Q x + \frac{1}{2}u^T R u dt + \frac{1}{2}x(T)^T P_1 x(T) \\ &\text{such that} \\ \dot{x} &= A(t)x + B(t)u \quad x(0) = x_0, \end{aligned}$$

we get

$$\begin{aligned} \dot{x} &= A(t)x + B(t)u & x(0) &= x_0 \\ \dot{p} &= -A(t)^T p - Q(t)x & p(T) &= p_1 \\ 0 &= B(t)^T p(t) + R(t)u. \end{aligned}$$

A notational restatement of this can be achieved by using the *Hamiltonian* formalism. Define

$$H = \ell + p^T f$$

where $\ell = \frac{1}{2}x^T Q x + \frac{1}{2}u^T R u$, p is the adjoint variable, and $f = A(t)x + B(t)u$. Then these equations are equivalent to Hamilton's equations

$$\begin{aligned} \dot{x} &= D_p H^T \\ \dot{p} &= -D_x H^T \\ 0 &= D_u H^T \end{aligned}$$

(We will also sometimes use the notation $D_p H = H_p$, $-D_x H = -H_x$, and $D_u H = H_u$.) For the case of the LQ problem, we have already shown that these equations provide the optimal control as a Two Point Boundary Value Problem. What is amazing is that these equations hold for *any* nonlinear system and for any objective function! This result, called the *Pontryagin Maximum Principle*, can be stated as the following.

Theorem 1. *Given $J = \int_0^T \ell(t, x(t), u(t))dt + m(x(T))$ and f sufficiently differentiable, and $H = \ell + p^T f$, the optimal control u satisfies the following relation.*

$$\begin{aligned} \dot{x} &= f(t, x(t), u(t)) & x(0) &= x_0 \\ \dot{p} &= -f_x(t, x(t), u(t))^T p - \ell_x(t, x(t), u(t)) & p(T) &= p_1 = m_x(T) \\ 0 &= f_u(t, x(t), u(t))^T p(t) + \ell_u(t, x(t), u(t)). \end{aligned}$$

We will use this fact many times in later sections.

5.2 Back to iterative optimization

This is basically the same thing that we did in the last section! That is, minimizing this linear quadratic problem is *very* similar to optimizing when there is a reference signal in the Lagrangian ℓ. To see this, define

$$(a(t)^T, b(t)^T) = D\ell(\xi).$$

Now, the Hamiltonian for the system with ζ as the optimization variable is

$$H = \underbrace{a^T z + b^T v}_{D\ell(\xi)\cdot\zeta} + \frac{1}{2}\underbrace{(z^T Q z + v^T R v) + p^T(Az + Bv)}_{||\zeta||^2}$$

(where Q and R are whatever we have chosen for the quadratic model) so that the optimality conditions become

$$\begin{aligned}
\dot{z} &= H_p^T = Az + Bv \\
\dot{p} &= -H_z^T = -a - Qz - A^T p \\
0 &= H_v^T = b + Rv + B^T p
\end{aligned}$$

so that $v = -R^{-1}(B^T p + b)$. The key here is that the a and b terms both typically depend on the *error* between the actual trajectory and the desired trajectory, so those terms will be responsible for locally driving the system cost down.

Assume that $p = Pz + r$ just like the last section. Taking derivatives with respect to time, we find that

$$\begin{aligned}
p &= Pz + r \\
\dot{p} &= \dot{P}z + P\dot{z} + \dot{r} \\
-a - Qz - A^T p &= \dot{P}z + PAz + PBv + \dot{r} \\
-a - Qz - A^T Pz - A^T r &= \dot{P}z + PAz + PBv + \dot{r} \\
-a - Qz - A^T Pz - A^T r &= \dot{P}z + PAz + PB(-R^{-1}(B^T p + b)) + \dot{r} \\
-a - Qz - A^T Pz - A^T r &= \dot{P}z + PAz - PBR^{-1}B^T p - PBR^{-1}b + \dot{r} \\
-a - Qz - A^T Pz - A^T r &= \dot{P}z + PAz - PBR^{-1}B^T Pz - PBR^{-1}B^T r - PBR^{-1}b + \dot{r} \\
0 &= (\dot{P} + PA + A^T P - PBR^{-1}B^T P + Q)z + (\dot{r} + A^T r - PBR^{-1}B^T r + a - PBR^{-1}b) \\
0 &= (\dot{P} + PA + A^T P - PBR^{-1}B^T P + Q)z + (\dot{r} + (A - BR^{-1}B^T P)^T r + a - PBR^{-1}b).
\end{aligned}$$

So we have two differential equations to solve:

$$\begin{aligned}
0 &= \dot{P} + PA + A^T P - PBR^{-1}B^T P + Q \\
0 &= \dot{r} + (A - BR^{-1}B^T P)^T r + a - PBR^{-1}b.
\end{aligned}$$

These are subject to terminal boundary conditions, of course. Remember, that we know what $p(T)$ is, and given z_0 we know what $z(T)$ is. As a consequence, we know what $P(T)$ and $r(T)$ must be as well.

We will obtain the *descent direction* defined by these equations by solving differential equations. What is the descent direction? It is $\zeta = (z, v)$ satisfying

$$\begin{aligned}
\dot{z} &= Az + Bv \\
v &= -R^{-1}B^T Pz - R^{-1}B^T r - R^{-1}b
\end{aligned}$$

where we have not yet defined what the initial condition of z should be. There are two "natural" choices of initial condition. One is that $z(0) = 0$, so that the perturbation at time zero is zero, making it easier to regulate at time $t = 0$. The other possibility is to choose an optimal $z(0)$—to minimize the objective with respect to z_0.

$$\min_{z_0} g(\zeta) = \min_{z_0} \int_0^T D_2\ell(t,\xi)\cdot\zeta + \frac{1}{2}\|\zeta\|^2 dt$$

The solution to this problem is $z(0) = -P^{-1}(0)r(0)$, but this isn't easy to show. The standard method involves computing the "cost to go" function; we will discuss this in the final section of these notes—the derivation of this extraordinarily compact result is quite involved—but for now all you really need to know is that $z(0) = -P^{-1}(0)r(0)$ is a good choice in your code.

5.3 The iLQR algorithm

To recap, last lecture we derived an optimal controller for a linear time-varying system. It was shown that a backwards propagating Riccati equation could be solved to obtain a optimal controller with respect to any quadratic cost:

Given the cost

$$J(u(\cdot)) = \frac{1}{2}\int_0^T x(t)^{\mathrm{T}}Q(t)x(t) + u(t)^{\mathrm{T}}R(t)u(t)\ \mathrm{d}t + \frac{1}{2}x(T)^{\mathrm{T}}P_1x(T),$$

the optimal system behavior is given as

$$\begin{aligned}
\dot{x}(t) &= (A(t) - B(t)K(t))x(t), \quad x(t_0) = x_0,\\
-\dot{P}(t) &= P(t)A(t) + A(t)^{\mathrm{T}}P(t) - P(t)B(t)R(t)^{-1}B(t)^{\mathrm{T}}P(t) + Q(t), \quad P(T) = P_1,\\
K(t) &= R(t)^{-1}B(t)^{\mathrm{T}}P(t).
\end{aligned}$$

Then, in this lecture, the methodology was extended to the case when the system is governed by nonlinear dynamics. However, instead of deriving a globally optimal controller the optimal descent direction was obtained. That is, we found the best way to change the current trajectory, $\xi = (x, u)$, to obtain a more optimal one. The descent direction, $\zeta = (z, v)$, is computed as:

Given the cost and nonlinear dynamics

$$J(u(\cdot)) = \int_0^T l(t,(x,u))\ \mathrm{d}t \quad \text{and} \quad \dot{x}(t) = f(x,u),$$

the descent direction, $\zeta = (z, v)$, is computed as

$$\begin{aligned}
\dot{z}(t) &= A(t)z(t) + B(t)v(t), \quad x(t_0) = x_0,\\
v(t) &= -R(t)^{-1}B(t)^{\mathrm{T}}P(t)z(t) - R(t)^{-1}B^{\mathrm{T}}(t)r(t) - R(t)^{-1}b(t),\\
-\dot{P}(t) &= P(t)A(t) + A(t)^{\mathrm{T}}P(t) - P(t)B(t)R(t)^{-1}B(t)^{\mathrm{T}}P(t) + Q(t),\\
-\dot{r}(t) &= (A - BR^{-1}B^{\mathrm{T}}P)^{\mathrm{T}}r + a - PBR^{-1}b.
\end{aligned}$$

where $(a(t)^{\mathrm{T}}, b(t)^{\mathrm{T}}) = D_2 l(t,\xi)$, $A(t) = D_1 f(x,u)$, and $B(t) = D_2 f(x,u)$.

A gradient descent algorithm can now be created based on this result.

Algorithm 1 Iterative Linear Quadratic Control

Data:

$\xi_0 = (x_0, u_0)$ Initial Trajectory (Initial Guess)

$J(\xi_0)$ Considered Cost

$\alpha = (0, \frac{1}{2})$, $\beta = (0, 1)$, and ϵ (a small number)

$i = 0$

while $||\zeta|| > \epsilon$ **do**

compute descent direction, $\zeta_i = (z_i, v_i)$

$\zeta_i = \arg\min_{\zeta} DJ(\xi_i) \circ \zeta + \frac{1}{2}||\zeta||^2$

compute line search using descent direction ζ_i

n = 0

while $J(\xi_i^+) > J(\xi_i) + \alpha\gamma DJ(\xi_i) \circ \zeta_i$ **do**

$u_i^+ = u_i + \gamma v_i$

$x_i^+ = x(t_0) + \int_0^T f(t, (x_i^+, u_i^+))\ \mathrm{d}t$

$\xi_i^+ = (x_i^+, u_i^+)$

$n = n + 1$

$\gamma = \beta^N$

end while

$\xi_{i+1} = \xi_i^+$

i = i + 1

end while

The algorithm is iterative since it successively updates the trajectory based on the computed descent direction. Note that the updated state trajectory, x_i^+, is not computed directly from the state descent direction, z_i. Instead, only the control signal is directly updated. Therefore, it cannot be assumed that the new state trajectory follows the computed descent direction ($x_i^+ \neq x_i + \gamma z_i$). Indeed, $\xi_i^+ = \xi_i + \gamma\zeta$ may not even be a dynamically feasible trajectory. In the next couple of lectures projection operators are introduced to address this issue.

5.4 Solving Optimal Control Problems Using Two Point Boundary Value Problems

From the Maximum Principle, we can also obtain a numerics-based approach to solving an optimal control problem. That is, we can just try to solve the two point boundary value problem directly (e.g., MATLAB, Mathematica, python have built-in solvers). If the solver returns an answer, you have your solution!

Exercises

Show numerically that the solution to the Riccati equation (obtained by solving the matrix-valued ordinary differential equation) and the solution to the Two Point Boundary Value Problem (obtained using an appropriate numerical solver) are nearly, but not exactly, the same. Use the example from last lecture as the example.

Use the Armijo line search for a single iterate of iLQR.

Apply iLQR to the vehicle example, using a semi-circle as an initial trajectory and an infeasible straight line corresponding to parallel parking as a reference trajectory.