1) Nonlinear Control

\[
\min \mathbb{E} \left[ \sum_{t=0}^{h-1} c(S_t, a_t) \left| S_{t+1} = f(S_t, a_t) \right. \right]
\]

In its full generality, this problem is hard to solve! Today we will learn an approach based on approximations.

In LQR (last lecture) we saw that the optimal policy did not depend on the disturbance \( w_t \). Since we are going to use an approximation based on LQR for the nonlinear problem, we consider deterministic dynamics.

**Assumption:**

- Dynamics \( f: S \times A \to S \)
- Cost \( c: S \times A \to \mathbb{R} \)

are differentiable and cost \( c \) is twice differentiable.

\[
\begin{align*}
\nabla_S f(s, a) & \quad \nabla_S c(s, a) \quad \nabla_S^2 c(s, a) \\
\nabla_A f(s, a) & \quad \nabla_A c(s, a) \quad \nabla_A^2 c(s, a) \quad \nabla_{AA} c(s, a)
\end{align*}
\]
Assumption: Either
1) we know the analytical form of \( f \) & \( c \), or
2) we have black-box access to \( f \) & \( c \), i.e.
we can observe \( s' = f(s, a) \)
and \( c = c(s, a) \).

IDEA: when dynamics are linear and costs are quadratic, we know how to find the optimal policy. So why not try
1) linearizing \( f \)
2) quadraticizing \( c \)
2) Linear/Quadratic Approximation

How can we find a good linear or quadratic approximation?

Recall Taylor Expansions: (in 1D)

\[ g(x) = g(x_0) + g'(x_0)(x-x_0) + \frac{1}{2}g''(x_0)(x-x_0)^2 + \ldots \]

When \( x \) is close to \( x_0 \), the higher order terms get vanishingly small

\[ \varepsilon^p \to 0 \text{ as } p \to \infty \text{ for } \varepsilon < 1 \]

E.g. \((0.001)^3 = 0.000000001 \quad 1e-3 \quad 1e-9 \]

**Linear Approximation**

\[ f(s,a) \approx f(s_0,a_0) + \nabla_s f(s_0,a_0)^T(s-s_0) + \nabla_a f(s_0,a_0)^T(a-a_0) \]

\( \nabla f(s,a) \in \mathbb{R}^{n_s \times n_s} \)

\( \nabla_a f(s,a) \in \mathbb{R}^{n_{ax} \times n_s} \)

Entry row \( i \) corresponds to \( f_j \) and \( s_i \).
\[ f(s, a) \approx f(s_0, a_0) + \nabla_s f(s_0, a_0)^T(s - s_0) + \nabla_a f(s_0, a_0)^T(a - a_0) \]

\[ = As + Ba + v \]

\[
A = \nabla_s f(s_0, a_0)^T
\]

\[
B = \nabla_a f(s_0, a_0)^T
\]

\[
v = f(s_0, a_0) + \nabla_s f(s_0, a_0)^T S_0 + \nabla_a f(s_0, a_0)^T a_0
\]

depend only on the point we are linearizing around.
Linear approximation doesn't encode (local) optima, so we use a quadratic (second-order) approximation for the cost function.

\[
C(s, a) \approx c(s_0, a_0) + \nabla_s c(s_0, a_0)^T (s - s_0) + \nabla_a c(s_0, a_0)^T (a - a_0) \\
+ \frac{1}{2} (s - s_0)^T \nabla_s^2 c(s_0, a_0) (s - s_0) \\
+ \frac{1}{2} (a - a_0)^T \nabla_a^2 c(s_0, a_0) (a - a_0) \\
+ (a - a_0)^T \nabla_{as} c(s_0, a_0) (s - s_0)
\]

\[
\nabla_s c(s, a) \in \mathbb{R}^{n_s} \quad \nabla_a c(s, a) \in \mathbb{R}^{n_a} \\
@ \text{index } i \quad \frac{dc}{ds_i} (s, a) \\
@ \text{index } i \quad \frac{dc}{da_i} (s, a)
\]

\[
\nabla_s^2 c(s, a) \in \mathbb{R}^{n_s \times n_s} \\
\begin{cases}
\nabla_{ii}^2 c(s, a) = \frac{d^2 c}{ds_i ds_j} (s, a) \\
\nabla_{ij}^2 c(s, a) = \frac{d^2 c}{da_i da_j} (s, a) \\
\end{cases} \\
\text{symmetric}
\]

\[
\nabla_{as} c(s, a) \in \mathbb{R}^{n_a \times n_s} \\
@ \text{index } i \quad \frac{d^2 c}{da_i ds_j} (s, a)
\]
\[ c(s,a) \approx c(s_0,a_0) + \nabla_s c(s_0,a_0)^T (s-s_0) + \nabla_a c(s_0,a_0)^T (a-a_0) + \frac{1}{2} (s-s_0)^T \nabla_s^2 c(s_0,a_0) (s-s_0) \]
\[ + \frac{1}{2} (a-a_0)^T \nabla_a^2 c(s_0,a_0) (a-a_0) + (a-a_0)^T \nabla_{as}^2 c(s_0,a_0) (s-s_0) \]

\[ = \mathbf{s}^T \bar{Q} \mathbf{s} + \mathbf{a}^T \bar{R} \mathbf{a} + \mathbf{a}^T \mathbf{M} \mathbf{s} \]
\[ + \mathbf{Q}^T \mathbf{s} + \mathbf{R}^T \mathbf{a} + c \]

**Practical Consideration:**

It doesn't make sense to minimize a downward facing quadratic.

Recall we had
\[ \bar{Q}, \bar{R} > 0 \]
positive definite.

Therefore we

1) put all negative eigenvalues to 0

2) add regularization \( \lambda I \)

If \( \bar{Q} = \sum_{i=1}^{n_s} \sigma_i \mathbf{v}_i \mathbf{v}_i^T \), \( \bar{Q} = \sum_{i=1}^{n_s} \max(\sigma_i, 0) \mathbf{v}_i \mathbf{v}_i^T + \lambda I \)

same for \( \bar{R} \to R \)
Black Box Access

What if we don’t know the analytical forms of $f$ & $c$ and can only observe $s’ = f(s, a)$, $c = c(s, a)$ for $s, a$ that we query?

Finite differencing:

For scalar $g'(x) \approx \frac{g(x+\delta) - g(x-\delta)}{2\delta}$

For multivariate:

\[
\frac{df_i}{ds_j} \approx \frac{f(s+\delta e_j, a) - f(s-\delta e_j, a)}{2\delta}
\]

\[
f = \begin{bmatrix} f_1 \\ \vdots \\ f_{hs} \end{bmatrix}
\]

\[
e_j = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \leftarrow j\text{th entry}
\]

Similar for $\frac{dc}{ds_j}$, $\frac{df_i}{da_j}$, $\frac{dc}{da_j}$.

For second derivatives:

\[
\frac{dc}{da_ida_j} = \frac{d}{da_i} \left[ \frac{dc}{ds_j} \right]
\]

First estimate $dc/\delta_j$ and then another with respect to $a_i$. 


3. Local LQR Control

Setting: minimize distance from a goal state/action $s_x, a_x$  $c(s, a) = d(s, s_x) + d(a, a_x)$

e.g. cart-pole (HW 1)

\[
\begin{align*}
\dot{\theta} &= \omega \\
\theta &= \text{angle} \\
f &= \text{force} \\
x &= \text{position} \\
\dot{x} &= v
\end{align*}
\]

\[
S = \begin{bmatrix}
\theta \\
\omega \\
x \\
v
\end{bmatrix}
\]

$s_x = 0, a_x = 0$

objective: balance upright

Approach: Locally linearize $f$ around $(s_x, a_x)$ and 2nd order approximation of $c$ around $(s_x, a_x)$

1) use finite differencing to compute approximate

\[a) \nabla_s f(s_x, a_x), \nabla_a f(s_x, a_x), \nabla^2_s c(s_x, a_x), \nabla_a c(s_x, a_x), \nabla^2_a c(s_x, a_x)\]

\[b) A_1, B_1, V\]

2) use formulas above to compute

\[a) A_1 B_1 V\]

\[b) Q_1 R_1 M_1 q_1 r_1 c\]

we will call this procedure

\[A_1 B_1 V, Q_1 R_1 M_1 q_1 r_1 c = \text{Approx} (f, c, (s_x, a_x))\]
\[
\min_{S_0 \succeq 0} \mathbb{E} \left[ \sum_{t=0}^{H-1} S_t^T Q S_t + a_t^T R a_t + q_t^T M S_t + q_t^T s_t + r_t^T a_t + c \right]
\]

Generalization of the LQR problem we discussed last lecture (HW1)

Results in quadratic \( V^* \) and affine \( \pi^* \)

\[
\pi_t^*(s) = K_t^* s + k_t^*
\]

\[
K_0^* \ldots, K_{H-1}^*, k_0^* \ldots, k_{H-1}^* = \text{LQR}(A, B, V, Q, R, M, q, r, c)
\]

Today we abstract this computation.

In HW1 you will see that this works quite well for balancing the cart-pole.

**Problem:** The approximations fail when \( s, a \) are far from \( s_*, a_* \).
4) Iterative LQR for trajectory optimization

IDEA: given a trajectory $\{S_t, A_t\}_{t=0}^{H-1}$ we can approximate around $(\bar{S}_t, \bar{A}_t)$ at time $t$.

This leads to a time-varying LQR problem with $A_t, B_t, V_t$ and $Q_t, R_t, M_t, g_t, r_t, c_t$ still results in $A_t^* = K_t^* S_t + K_t^*$

However, which trajectory should we approximate around? Iterate!

Alg: iLQR:

Initialize $\bar{a}_0, \ldots, \bar{a}_{H-1}$ and $S_0 \sim \mu_0$

generate nominal trajectory $t_0^* = f(S_0, A_0)$ for $i = 0, 1, \ldots$

$\{A_t, B_t, V_t, Q_t, R_t, g_t, r_t, c_t\}_{t=0}^{H-1}$

$\{S_t, A_t\}_{t=0}^{H-1} = \text{APPROX} (f, c, T_i)$

$LQR(\{A_t, B_t, V_t, Q_t, R_t, g_t, r_t, c_t\}_{t=0}^{H-1})$

generate $T_{i+1} = \{S_{t+1}, A_{t+1}(S_{t+1})^T\}$, $S_{t+1} = f(S_t, T_t(S_t))$