Note on Iterative Linear Quadratic Regulator

Wen Sun

1Department of Computer Science, Cornell University

March 2, 2021

1 Setting

Consider the following finite horizon nonlinear control problem:

\[
\min_{\pi} \sum_{t=0}^{H-1} c(x_t, a_t) + g(x_H),
\]

s.t., \( x_{t+1} = f(x_t, u_t), \ u_t = \pi(x_t), \forall t = 0, \ldots, H - 1 \)

where assume \( x_0 \) is given. Here we are going to assume that the transition \( f(x, u) \) is differentiable with respect to \( x, u \), and the cost functions \( c \) are twice differentiable with respect to both \( x \) and \( u \), and \( g \) is twice differential with respect to \( x \).

Note we will use subscript \( t \) instead of \( h \) to represent time step in this note.

2 Time inhomogeneous LQR

In class and HWs, we covered time homogeneous LQR. Here we study the time inhomogeneous setting.

\[
\min_{\pi} x_0^T P_H x_H + x_H^T q_H + c_H + \sum_{t=0}^{H-1} \left( x_t^T Q_t x_t + u_t^T R_t u_t + x_t^T M_t u_t + x_t^T q_t + u_t^T r_t + c_t \right),
\]

s.t., \( x_{t+1} = A_t x_t + B_t u_t + m_t, \ u_t = \pi(x_t), \forall t = 0, \ldots, H - 1, \ x_0 \sim \mathcal{N}(\bar{x}_0, \Sigma). \)

We will assume that \( Q_t \in \mathbb{R}^{d_x \times d_x} \) and \( R_t \in \mathbb{R}^{d_u \times d_u} \) are positive definite matrices, and \( q_t \in \mathbb{R}^{d_x} \) and \( r_t \in \mathbb{R}^{d_u} \), and \( c_t, m_t \in \mathbb{R} \). Note that our cost function not only contains the second order term, but now also contains the first order term and the zero-th order term as well.

To compute the optimal policy, we again will just use Dynamic Programming.

At the last time step \( H \), we have that \( V^*_H(x) \) just being the cost function at time step \( H \). Let us define \( V^*_H(x) \) as:

\[
V^*_H(x) = x^T P_H x + x^T p_H + z_H,
\]

where \( P_H = Q_H, p_H = q_H, z_H = c_H \).

Our goal is to use Dynamic programming to show that \( V^*_H(x) \) also has this quadratic formulation that contains second order term, first order term, and the zero-th order term:

\[
V^*_H(x) = x^T P_{H-1} x + x^T p_{H-1} + z_{H-1},
\]
where \( P_{H-1}, p_{H-1}, z_{H-1} \) are variables that depend on \( P_H, p_H, z_H \) and \( A_{H-1}, B_{H-1}, m_{H-1}, Q_{H-1}, q_{H-1}, R_{H-1}, r_{H-1}, c_{H-1} \).

In the HW problem, we will work out the detailed recursion update. Given \( V_h^* \) and \( Q_h^* \), we can derive the optimal policy as:

\[
\pi_h^*(x) = \arg\min_u Q_h^*(x, u).
\]

We will see in HW problem that this optimal policy again is a linear function of \( x \).

**Oracle** For the purpose of notation simplicity for iterative LQR on nonlinear control, Let us summarize the above DP procedure as the following black-box procedure:

\[
\{ \pi_0^*, \ldots, \pi_{H-1}^* \} \leftarrow \text{GeneralLQR} \left( Q_H, q_H, c_H, \{ Q_t, q_t, c_t, R_t, r_t, M_t, A_t, B_t, m_t \}_{t=0}^{H-1} \right).
\]

Namely, the general LQR black box takes the problem parameters (e.g., parameters of the cost functions, and parameters of the transitions) as input, outputs the optimal policy \( \pi_h^* \) for \( h \in [H] \).

As you will see in the HW problems, the optimal policy takes the following linear form:

\[
\pi_h^*(x) = -K_h^* x + k_h^*,
\]

where \( K_h^* \in \mathbb{R}^{d_u \times d_x} \) and \( k_h^* \in \mathbb{R}^{d_u} \).

3 **Iterative Linear Quadratic Regulator**

Now we consider the general nonlinear control problem that we posed at the beginning of this lecture note. We are going to use an iterative procedure to compute a locally optimal policy for the general nonlinear control problem. In high level, at each iteration, given the current trajectory, what we will do is that we are going to perform second-order taylor expansion on the cost functions \( c \) at the current trajectory state-action pairs; we are going to perform first order taylor expansion on the nonlinear dynamics at the current trajectory state-action pairs. With taylor expansion, we get quadratic cost functions—one for each time step, and we get linear dynamical systems—one for each time step.

3.1 **Taylor Expansion**

Let us consider the Talyor expansion procedure here which is used to get a quadratic cost function and a linear dynamical system.

Assume that we are given a nominal trajectory \( \{ \bar{x}_0, \bar{u}_0, \bar{x}_1, \bar{u}_1, \ldots, \bar{x}_{H-1}, \bar{u}_{H-1}, \bar{x}_H \} \). We first perform first-order Taylor expansion on the nonlinear dynamical system. Linearize \( f \) at \( \bar{x}_t, \bar{u}_t \), we get:

\[
f(x, u) \approx f(\bar{x}_t, \bar{u}_t) + \nabla_x f(\bar{x}_t, \bar{u}_t) (x - \bar{x}_t) + \nabla_u f(\bar{x}_t, \bar{u}_t) (u - \bar{u}_t),
\]

where \( \nabla_x f(x, u) \in \mathbb{R}^{d_x \times d_x} \) is the partial gradient of \( f \) with respect to \( x \) (it is also called Jacobian), i.e.,

\[
\nabla_x f(x, u)[i, j] = \frac{\partial f[i]}{\partial x[j]}(x, u),
\]

and \( \nabla_u f(x, u) \in \mathbb{R}^{d_x \times d_u} \) is the partial gradient of \( f \) with respect to \( u \), i.e.,

\[
\nabla_u f(x, u)[i, j] = \frac{\partial f[u]}{\partial u[j]}(x, u).
\]

Perform linearization for all \( t \), we get a sequence of time-inhomogeneous linear dynamical systems, where:

\[
A_t := \nabla_x f(\bar{x}_t, \bar{u}_t), \quad B_t := \nabla_u f(\bar{x}_t, \bar{u}_t), \quad m_t := f(\bar{x}_t, \bar{u}_t) - A_t \bar{x}_t - B_t \bar{u}_t, \quad \forall t = 0, \ldots, H - 1. \quad (1)
\]
We can perform similar operation on the non-linear cost functions, but we will perform second-order taylor expansion instead (recall that in LQR formulation, we can handle quadratic cost functions). For all $t$, perform second-order Taylor expansion at $(\bar{x}_t, \bar{u}_t)$, we get:

$$c(x, u) \approx c(\bar{x}_t, \bar{u}_t) + \nabla_x c(\bar{x}_t, \bar{u}_t)^\top (x - \bar{x}_t) + \nabla_u c(\bar{x}_t, \bar{u}_t)^\top (u - \bar{u}_t)$$

$$+ \frac{1}{2} (x - \bar{x}_t)^\top \nabla_x^2 c(\bar{x}_t, \bar{u}_t)(x - \bar{x}_t) + \frac{1}{2} (u - \bar{u}_t)^\top \nabla_u^2 c(\bar{x}_t, \bar{u}_t)(u - \bar{u}_t) + (x - \bar{x}_t)^\top \nabla_{x,u} c(\bar{x}_t, \bar{u}_t)(u - \bar{u}_t),$$

where $\nabla_x c(x, u) \in \mathbb{R}^{d_x}$ is the gradient of $c(x, u)$ with respect to $x$, i.e., $\nabla_x c(x, u)[i] = \frac{\partial c}{\partial x^i}(x, u)$ (similarly for $\nabla_u c(x, u) \in \mathbb{R}^{d_u}$, and $\nabla_x^2 c(x, u) \in \mathbb{R}^{d_x \times d_x}$ is the Hessian with respect to $x$, i.e., $\nabla_x^2 c(x, u)[i, j] = \frac{\partial^2 c}{\partial x^i \partial x^j}(x, u)$ (similarly for $\nabla_u^2 c(x, u)$), and $\nabla_{x,u} c(x, u) \in \mathbb{R}^{d_x \times d_u}$).

This gives us a sequence of quadratic cost functions,

$$Q_H = \nabla_x^2 g(\bar{x}_H)/2, \quad q_H = \nabla_x g(\bar{x}_H) - \nabla_x^2 g(\bar{x}_H)\bar{x}_H, \quad c_H = g(\bar{x}_H) - q_H^\top \bar{x}_H + \frac{1}{2} \bar{x}_H^\top \nabla_x^2 g(\bar{x}_H)\bar{x}_H,$$

$$\forall t = 0, \ldots, H - 1 :$$

$$Q_t = \nabla_x^2 c(\bar{x}_t, \bar{u}_t)/2, \quad R_t = \nabla_u^2 c(\bar{x}_t, \bar{u}_t)/2,$$

$$M_t = \nabla_{x,u} c(\bar{x}_t, \bar{u}_t),$$

$$q_t = \nabla_x c(\bar{x}_t, \bar{u}_t) - \nabla_x^2 c(\bar{x}_t, \bar{u}_t)\bar{x}_t - \nabla_{x,u} c(\bar{x}_t, \bar{u}_t)\bar{u}_t,$$

$$r_t = \nabla_u c(\bar{x}_t, \bar{u}_t) - \nabla_u^2 c(\bar{x}_t, \bar{u}_t)\bar{u}_t - \nabla_{x,u} c(\bar{x}_t, \bar{u}_t)\bar{x}_t,$$

$$c_t = c(\bar{x}_t, \bar{u}_t) - \nabla_x c(\bar{x}_t, \bar{u}_t)^\top \bar{x}_t - \nabla_u c(\bar{x}_t, \bar{u}_t)^\top \bar{u}_t + \frac{1}{2} \bar{x}_t^\top Q_t \bar{x}_t + \frac{1}{2} \bar{u}_t^\top R_t \bar{u}_t + \bar{x}_t^\top M_t \bar{u}_t;$$

So after this procedure, we are almost there to call the general LQR black box oracle to compute $\pi_0^*, \ldots, \pi_{H-1}^*$, using $A_t, B_t, M_t, Q_t, q_t, R_t, M_t, r_t, c_t$ for $t \in [H]$. But we need one more step to process $Q_t, R_t$ for all $t$.

Note that while the above derivation looks tedious, all we did is basically Taylor expansion and re-arrange terms to get the form that we want.

### 3.2 Making Quadratized cost function convex

Now that our original nonlinear cost function might not even be convex. So after performing second order taylor expansion at some $(\bar{x}_t, \bar{u}_t), Q_t$ and $R_t$ might not be PSD. To use LQR, we need to use PSD $Q_t$ and $R_t$. We can perform the following approximation on $Q_t$ and $R_t$ to force them to be PSD.

Denote the eigen-decomposition of $Q_t$ as follow: $Q_t = \sum_{i=1}^{d_u} \sigma_i v_i v_i^\top$, where $\sigma_i$ is the $i$-th eigenvalue and $v_i$ is the corresponding eigenvector. For any negative eigenvalues, we simply set them to zero, i.e., we construct a PSD matrix to approximate $Q_t$ as follows:

$$Q_t \leftarrow \sum_{i=1}^{d_u} \sigma_i v_i v_i^\top 1\{\sigma_i \geq 0\} + \lambda I, \quad \forall t = 0, \ldots, H,$$

where $\lambda \in \mathbb{R}^+$ is the regularization to make $Q_t$ truly positive definite.

Similarly, denote the eigendecomposition of $R_t$ as $R_t = \sum_{i=1}^{d_u} \xi_i u_i u_i^\top$, and we construct a PSD matrix by throwing aways negative eigenvalues to approximate $R_t$ as follows:

$$R_t \leftarrow \sum_{i=1}^{d_u} \xi_i u_i u_i^\top 1\{\xi_i \geq 0\} + \lambda I, \quad \forall t = 0, \ldots, H - 1$$

With the above approximation procedures on $Q_t$ and $R_t$, we must have both $Q_t$ and $R_t$ being positive definite matrices.
3.3 The Iterative Procedure

We start from the following iterative procedure. Let us initialize a sequence of nominal controls, i.e., initialize $\bar{u}^0_0, \bar{u}^0_1, \ldots, \bar{u}^0_{H-1}$ (i.e., maybe we can just set $\bar{u}^0_t = 0$ for all $t$ as one kind of initialization). We execute $\bar{u}^0_0, \bar{u}^0_1, \ldots, \bar{u}^0_{H-1}$ and get a sequence of corresponding states $\bar{x}^0_0, \ldots, \bar{x}^0_H$, where $\bar{x}^0_{t+1} = f(\bar{x}^0_t, \bar{u}^0_t)$.

Repeat the following procedure for $k = 0 \rightarrow \infty$:

1. Define $\tilde{c}_t(x,u) := c(x,u) + \lambda \|u - \bar{u}^k_t\|_2^2$ where $\lambda \in \mathbb{R}^+$, $\forall t = 0, H - 1$, and define $\tilde{c}_H(x) = g(x)$.

2. Following Eq 4, we quadratic $\tilde{c}_t$ around $(\bar{x}^k_t, \bar{u}^k_t)$ to get $Q_t, R_t, M_t, q_t, r_t, c_t$ for $t = 0, \ldots, H - 1$, and quadratize $\tilde{c}_H$ at $\bar{x}^k_H$ to get $Q_H, q_H, c_H$.

3. Following Eq. 1, we linearize $f(x,u)$ at $(\bar{x}^k_t, \bar{u}^k_t)$ to get $A_t, B_t, m_t$, for all $t = 0, \ldots, H - 1$.

4. Following Eq. 11 and 12, make $Q_0, \ldots, Q_H$ and $R_0, \ldots, R_{H-1}$ positive definite matrices.

5. $\pi_0, \ldots, \pi_{H-1} \leftarrow \text{GeneralLQR} \left( Q_H, q_H, c_H, \{Q_t, q_t, c_t, R_t, r_t, M_t, A_t, B_t, m_t\}_{t=0}^{H-1} \right)$

6. Starting at $\bar{x}^{k+1}_0 = \bar{x}_0$, we compute $\bar{u}^{k+1}_t = \pi_t(\bar{x}^{k+1}_t), \bar{x}^{k+1}_{t+1} = f(\bar{x}^{k+1}_t, \bar{u}^{k+1}_t)$ for $t = 0, \ldots, H - 1$

Basically, we iteratively linearize and quadratize around the current state-action trajectory; we then call our general LQR oracle to compute a sequence of new policies, and re-generate the new state-action trajectory starting at $x_0$, and repeat.

Control Damping In the class, we talked about using the line search trick to ensure stable convergence. Here, we present a another popular approach. Note that in Item 1 we add a regularization term $\lambda \|u - \bar{u}^k_t\|_2^2$ to $c(x,u)$ with $\lambda \in \mathbb{R}^+$. This term ensures that we penalize the deviation from the current control $\bar{u}^k_t$. This forces that our new state-action trajectory is not that far away from the current one. This regularization is necessary since the taylor expansion is only valid in the region that is near the expansion point $(\bar{x}^k_t, \bar{u}^k_t)$. In the extreme case, setting $\lambda = \infty$, we will never move away from the current trajectory $(\bar{x}^k_t, \bar{u}^k_t)$. In practice, we need to tune $\lambda$ to get the best empirical performance (we want to make fast improvement, but we also want to be cautious as linearization and quadratization induces approximation errors). This regularization trick is often called control damping.