ME 7247: Advanced Control Systems **Supplementary notes**

The Linear Quadratic Regulator

v.1.0 (11.07.2022)

In these notes, we will derive the solution to the finite-horizon linear quadratic regulator (LQR) problem in several different ways. Fundamentally, LQR can be viewed as a large least-squares problem, but we are interested in the recursive solution because it can be efficiently computed (storage and computation scale linearly with the length of the time horizon).

1 The LQR problem

We consider the discrete-time finite-horizon version of the LQR problem. Consider the dynamical system with initial state x_0 and

$$x_{t+1} = Ax_t + Bu_t \qquad \text{for } t = 0, \dots, N-1 \tag{1}$$

The objective is to find a sequence of decisions $u_0, \dots, u_{N-1}$ that minimizes the quadratic cost

$$J = \sum_{t=0}^{N-1} \underbrace{\begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix}}_{\text{stage cost}} + \underbrace{x_N^\mathsf{T} Q_f x_N}_{\text{terminal cost}} \tag{2}$$

The only assumptions we make are that $\begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \succeq 0$, $Q_f \succeq 0$, and $R \succ 0$. These assumptions ensure that the cost will remain bounded. We first state result, and then we derive it in many ways.

Theorem 1. *The optimal decisions that solve the LQR problem are given by the state feedback policy* $u_t = K_t x_t$ *for* $t = 0, \dots, N-1$. *We can compute the optimal policy recursively in an offline fashion by starting at* $t = N$ *and working backwards to* $t = 0$. *The recursion is:*

$$P_N = Q_f \tag{3a}$$

$$P_t = A^\mathsf{T} P_{t+1} A + Q - (A^\mathsf{T} P_{t+1} B + S)(B^\mathsf{T} P_{t+1} B + R)^{-1}(B^\mathsf{T} P_{t+1} A + S^\mathsf{T}) \tag{3b}$$

$$K_t = -(B^\mathsf{T} P_{t+1} B + R)^{-1}(B^\mathsf{T} P_{t+1} A + S^\mathsf{T}) \tag{3c}$$

The optimal cost starting from initial condition x_0 *is given by* $J_\star = x_0^\mathsf{T} P_0 x_0$.

Note: We can make the state and cost matrices time-varying if we like, i.e. A_t, B_t, Q_t, S_t, R_t. The solution is exactly analogous. We just have to make the recursion time-varying. So:

$$P_t = A_t^\mathsf{T} P_{t+1} A_t + Q_t - (A_t^\mathsf{T} P_{t+1} B_t + S_t)(B_t^\mathsf{T} P_{t+1} B_t + R_t)^{-1}(B_t^\mathsf{T} P_{t+1} A_t + S_t^\mathsf{T})$$

$$K_t = -(B_t^\mathsf{T} P_{k+1} B_t + R_t)^{-1}(B_t^\mathsf{T} P_{k+1} A_t + S_t^\mathsf{T})$$

In fact, we can even *make the sizes of all matrices time-varying*! For example, the state x_t and input u_t could have different sizes as t changes.

1.1 Solution via dynamic programming

Define the cost-to-go (optimal value function) for $k = 0, \dots, N$ as

$$
\begin{aligned}
V_k(z) := \underset{u_k,\dots,u_{N-1}}{\text{minimize}} \quad & \sum_{t=k}^{N-1} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + x_N^\mathsf{T} Q_f x_N \\
\text{s.t.} \quad & x_{t+1} = Ax_t + Bu_t \quad \text{for } t = k, \dots, N-1 \\
& x_k = z
\end{aligned}
$$

Our ultimate goal is to find $V_0(x_0)$, but we will solve for all V_k for $k = 0, \dots, N$. By defining $w := u_k$ and decomposing the value function by separating the first decision at time k from all subsequent decisions, we can show that the following recursive equation (the Bellman equation) holds:

$$
V_k(z) = \min_w \left(\begin{bmatrix} z \\ w \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} + V_{k+1}(Az + Bw) \right) \qquad \text{for } k = 0, \dots, N-1. \tag{4}
$$

When $k = z$, we have $V_N(z) = z^\mathsf{T} Q_f z$. We can show by induction that $V_k(z)$ is a positive semidefinite quadratic for all $k \le N$. Suppose that $V_t(z) = z^\mathsf{T} P_t z$ with $P_t \succeq 0$ for $t = k+1$. We will prove that this holds for $t = k$ as well. Substitute into Eq. (4) and obtain

$$
V_k(z) = \min_w \left(\begin{bmatrix} z \\ w \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} + (Az + Bw)^\mathsf{T} P_{k+1} (Az + Bw) \right) \tag{5}
$$

$$
= \min_w \begin{bmatrix} z \\ w \end{bmatrix}^\mathsf{T} \begin{bmatrix} A^\mathsf{T} P_{k+1} A + Q & A^\mathsf{T} P_{k+1} B + S \\ B^\mathsf{T} P_{k+1} A + S^\mathsf{T} & B^\mathsf{T} P_{k+1} B + R \end{bmatrix} \begin{bmatrix} z \\ w \end{bmatrix} \tag{6}
$$

This is a standard quadratic optimization problem. Due to our assumption that $P_{k+1} \succeq 0$ and $R \succ 0$, the solution is

$$
\begin{aligned}
w^\star &= -(B^\mathsf{T} P_{k+1} B + R)^{-1} (B^\mathsf{T} P_{k+1} A + S^\mathsf{T}) z \\
V_k(z) &= z^\mathsf{T} \left(A^\mathsf{T} P_{k+1} A + Q - (A^\mathsf{T} P_{k+1} B + S)(B^\mathsf{T} P_{k+1} B + R)^{-1} (B^\mathsf{T} P_{k+1} A + S^\mathsf{T}) \right) z
\end{aligned}
$$

We deduce that $V_k(z)$ is also quadratic, and P_k satisfies the recursion (3a)–(3b) Since $w = u_k$ and $z = x_k$, we also find that the optimal policy is a state-feedback policy of the form $u_t = K_t x_t$, where K_t is given by (3c). The cost associated with using the optimal control policy starting from the state x_0 is the cost to go $V_0(x_0)$, which is given by $x_0^\mathsf{T} P_0 x_0$.

Note. We assumed $\begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \succeq 0$ and $R \succ 0$, so we can prove by induction that since $P_N = Q_f \succeq 0$, each $V_t(z) = z^\mathsf{T} P_t z$ is the minimum of a positive definite quadratic function (5), and is therefore positive semidefinite, and we have $P_t \succeq 0$ for all t.

The above dynamic programming approach works even when the system matrices are time-varying or even have different sizes as a function of time.

1.2 Solution via completing the square

Consider the cost we are trying to minimize:

$$J(x_0) = \sum_{t=0}^{N-1} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + x_N^\mathsf{T} Q_f x_N$$

Let's introduce a set of matrices $P_0, P_1, \dots, P_N$ and include them into the sum as follows.

$$J(x_0) = x_0^\mathsf{T} P_0 x_0 + \sum_{t=0}^{N-1} \left(x_{t+1} P_{t+1} x_{t+1} - x_t^\mathsf{T} P_t x_t + \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} \right) + x_N^\mathsf{T} (Q_f - P_N) x_N.$$

Note that all the P_t's cancel out, so the above expression is equal to $J(x_0)$ *no matter what values we pick* for the P_t's. Start by substituting $x_{t+1} = Ax_t + Bu_t$ in the sum and it becomes

$$J(x_0) = x_0^\mathsf{T} P_0 x_0 + \sum_{t=0}^{N-1} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} A^\mathsf{T} P_{t+1} A - P_t + Q & A^\mathsf{T} P_{t+1} B + S \\ B^\mathsf{T} P_{t+1} A + S^\mathsf{T} & B^\mathsf{T} P_{t+1} B + R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + x_N^\mathsf{T} (Q_f - P_N) x_N.$$

Recall the completion of squares formula (LDU factorization):

$$\begin{bmatrix} x \\ u \end{bmatrix}^\mathsf{T} \begin{bmatrix} A & B \\ B^\mathsf{T} & C \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = x^\mathsf{T} \left(A - BC^{-1} B^\mathsf{T} \right) x + \left(u - C^{-1} B^\mathsf{T} x \right)^\mathsf{T} C \left(u - C^{-1} B^\mathsf{T} x \right)$$

Applying this to the quadratic form in the sum, we obtain:

$$\begin{aligned} J(x_0) &= x_0^\mathsf{T} P_0 x_0 \\ &\quad + \sum_{t=0}^{N-1} x_t^\mathsf{T} \left(A^\mathsf{T} P_{t+1} A - P_t + Q - (A^\mathsf{T} P_{t+1} B + S)(B^\mathsf{T} P_{t+1} B + R)^{-1} (B^\mathsf{T} P_{t+1} A + S^\mathsf{T}) \right) x_t \\ &\qquad + \sum_{t=0}^{N-1} (u_t - K_t x_t)^\mathsf{T} (B^\mathsf{T} P_{t+1} B + R)(u_t - K_t x_t) + x_N^\mathsf{T} (Q_f - P_N) x_N \end{aligned}$$

where we defined K_t as in (3c). Again, remember that this expression for $J(x_0)$ *does not depend on the choice of the* P_t's. So we can choose them however we like. In particular, if we choose P_t so that it satisfies (3a)–(3b), the sum simplifies greatly to

$$J(x_0) = x_0^\mathsf{T} P_0 x_0 + \sum_{t=0}^{N-1} (u_t - K_t x_t)^\mathsf{T} (B^\mathsf{T} P_{t+1} B + R)(u_t - K_t x_t). \tag{7}$$

We also have $P_t \succeq 0$ for all t (see the note at the end of Section 1.1). Therefore each term in the sum is nonnegative. We can minimize $J(x_0)$ by picking $u_t = K_t x_t$, which leaves us with the optimal cost $J_\star = x_0^\mathsf{T} P_0 x_0$.

Note. If we use a *suboptimal* policy $\hat{K}_t$ instead of the optimal K_t, then the formula (7) reveals exactly the extra cost we will have to pay. In particular,

$$J_{\text{extra}} = \sum_{t=0}^{N-1} x_t^\mathsf{T} (\hat{K}_t - K_t)^\mathsf{T} (B^\mathsf{T} P_{t+1} B + R)(\hat{K}_t - K_t) x_t$$

1.3 Solution via block elimination

We will make use of *block variable elimination.* Here is a useful result that is easy to prove.

Proposition 1 (block elimination)**.** *Suppose we have linear equations of the form*

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} p \\ 0 \end{bmatrix},$$

where D is square and invertible. If we solve for y in the second equation and substitute the result into the first equation, we obtain

$$(A - BD^{-1}C)x = p \qquad \textit{and} \qquad y = -D^{-1}Cx.$$

We will make use of this result throughout the following derivation.

Write out the objective and all constraints as a large optimization problem. Here, we treat both the states and inputs as variables, and we include the state dynamics as constraints.

$$\begin{aligned} \underset{\substack{x_1,\dots,x_N,\\ u_0,\dots,u_{N-1}}}{\text{minimize}} \quad & \sum_{t=0}^{N-1} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + x_N^\mathsf{T} Q_f x_N \\ \text{s.t.} \quad & x_{t+1} = Ax_t + Bu_t \quad \text{for } t = 0, \dots, N-1 \end{aligned}$$

Assign the Lagrange multiplier λ_{t+1} to the equality constraints for $t = 0, \dots, N-1$. The Lagrangian for the problem is therefore:

$$L(x, u, \lambda) = \frac{1}{2} \sum_{t=0}^{N-1} \begin{bmatrix} x_t \\ u_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} Q & S \\ S^\mathsf{T} & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} + \frac{1}{2} x_N^\mathsf{T} Q_f x_N - \sum_{t=0}^{N-1} \lambda_{t+1}^\mathsf{T} (x_{t+1} - Ax_t - Bu_t)$$

The factors of $\frac{1}{2}$ are there to make the algebra nicer. The KKT necessary conditions for optimality are $\nabla_x L = 0$, $\nabla_u L = 0$, and $\nabla_\lambda L = 0$. Evaluating these gradients, we obtain the equations

$$\begin{aligned} Qx_t + Su_t + A^\mathsf{T}\lambda_{t+1} - \lambda_t &= 0 \qquad \text{for } t = 0, \dots, N-1 \\ Q_f x_N - \lambda_N &= 0 \\ S^\mathsf{T} x_t + Ru_t + B^\mathsf{T}\lambda_{t+1} &= 0 \qquad \text{for } t = 0, \dots, N-1 \\ Ax_t + Bu_t - x_{t+1} &= 0 \qquad \text{for } t = 0, \dots, N-1 \end{aligned}$$

Merging these together as a single set of linear equations, we obtain:

$$\lambda_N = Q_f x_N \tag{8a}$$

$$\begin{bmatrix} \lambda_t \\ 0 \\ x_{t+1} \end{bmatrix} = \begin{bmatrix} Q & S & A^\mathsf{T} \\ S^\mathsf{T} & R & B^\mathsf{T} \\ A & B & 0 \end{bmatrix} \begin{bmatrix} x_t \\ u_t \\ \lambda_{t+1} \end{bmatrix} \qquad \text{for } t = 0, \dots, N-1 \tag{8b}$$

We will prove by induction that $\lambda_t = P_t x_t$ for all t. From (8a), the result holds for $t = N$ with $P_N = Q_f$. Suppose it holds for $t+1$. Substitute $\lambda_{t+1} = P_{t+1}x_{t+1}$ into (8b) and obtain:

$$\begin{bmatrix} \lambda_t \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} Q & S & A^\mathsf{T} P_{t+1} \\ S^\mathsf{T} & R & B^\mathsf{T} P_{t+1} \\ A & B & -I \end{bmatrix} \begin{bmatrix} x_t \\ u_t \\ x_{t+1} \end{bmatrix} \tag{9}$$

Apply Proposition 1 to eliminate x_{t+1} from (9), which leads to:

$$\begin{bmatrix}\lambda_t \\ 0\end{bmatrix} = \begin{bmatrix} A^\mathsf{T} P_{t+1} A + Q & A^\mathsf{T} P_{t+1} B + S \\ B^\mathsf{T} P_{t+1} A + S^\mathsf{T} & B^\mathsf{T} P_{t+1} B + R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix}$$

Apply Proposition 1 once more to eliminate u_t, which leads to:

$$\begin{aligned} \lambda_t &= \left(A^\mathsf{T} P_{t+1} A + Q - (A^\mathsf{T} P_{t+1} B + S)(B^\mathsf{T} P_{t+1} B + R)^{-1}(B^\mathsf{T} P_{t+1} A + S^\mathsf{T}) \right) x_t \\ u_t &= -(B^\mathsf{T} P_{t+1} B + R)^{-1}(B^\mathsf{T} P_{t+1} A + S^\mathsf{T}) x_t \end{aligned}$$

Therefore, we have $\lambda_t = P_t x_t$, which is what we wanted to prove, and the recursion for P_t and the expression for K_t are precisely the solution we previously found in Eq. (3).

Alternative elimination ordering. If we eliminate the variables in a different order, we get different (but equivalent) expressions for the P_t recursion and for K_t. Specifically, if we start from (9) but apply Proposition 1 to eliminate u_t first, we obtain:

$$\begin{aligned} \begin{bmatrix}\lambda_t \\ 0\end{bmatrix} &= \begin{bmatrix} Q - SR^{-1}S^\mathsf{T} & A^\mathsf{T} P_{t+1} - SR^{-1}B^\mathsf{T} P_{t+1} \\ A - BR^{-1}S^\mathsf{T} & -I - BR^{-1}B^\mathsf{T} P_{t+1} \end{bmatrix} \begin{bmatrix} x_t \\ x_{t+1} \end{bmatrix} \\ u_t &= -R^{-1}\big(S^\mathsf{T} x_t + B^\mathsf{T} P_{t+1} x_{t+1}\big) \end{aligned}$$

To ease the notation, define:

$$E := A - BR^{-1}S^\mathsf{T} \qquad G := BR^{-1}B^\mathsf{T} \qquad \bar{Q} := Q - SR^{-1}S^\mathsf{T}$$

Based on our original problem assumptions, we have $G \succeq 0$ and $\bar{Q} \succeq 0$. Using our new variable definitions, the equations simplify to:

$$\begin{aligned} \begin{bmatrix}\lambda_t \\ 0\end{bmatrix} &= \begin{bmatrix} \bar{Q} & E^\mathsf{T} P_{t+1} \\ E & -(I + GP_{t+1}) \end{bmatrix} \begin{bmatrix} x_t \\ x_{t+1} \end{bmatrix} \\ u_t &= -R^{-1}\big(S^\mathsf{T} x_t + B^\mathsf{T} P_{t+1} x_{t+1}\big) \end{aligned}$$

Now apply Proposition 1 to eliminate x_{t+1} and obtain:

$$\begin{aligned} \lambda_t &= \left(\bar{Q} + E^\mathsf{T} P_{t+1} (I + GP_{t+1})^{-1} E \right) x_t \\ u_t &= -R^{-1}\big(S^\mathsf{T} + B^\mathsf{T} P_{t+1} (I + GP_{t+1})^{-1} E\big) x_t \\ x_{t+1} &= (I + GP_{t+1})^{-1} E x_t \end{aligned}$$

This yields new (but equivalent!) formulas for the optimal controller (3) and the optimal closed-loop matrix $A + BK_t$.

$$\boxed{\begin{aligned} P_N &= Q_f \\ P_t &= \bar{Q} + E^\mathsf{T} P_{t+1} (I + GP_{t+1})^{-1} E \\ K_t &= -R^{-1}\big(S^\mathsf{T} + B^\mathsf{T} P_{t+1} (I + GP_{t+1})^{-1} E\big) \\ A + BK_t &= (I + GP_{t+1})^{-1} E \end{aligned}} \tag{10}$$

1.4 Solution via adjoint equations

This approach is similar to the block elimination approach of Section 1.3. We start with (8):

$$\lambda_N = Q_f x_N \tag{11a}$$

$$\begin{bmatrix} \lambda_t \\ 0 \\ x_{t+1} \end{bmatrix} = \begin{bmatrix} Q & S & A^\mathsf{T} \\ S^\mathsf{T} & R & B^\mathsf{T} \\ A & B & 0 \end{bmatrix} \begin{bmatrix} x_t \\ u_t \\ \lambda_{t+1} \end{bmatrix} \qquad \text{for } t = 0, \dots, N-1 \tag{11b}$$

Eliminate u_t right away using Proposition 1 and use the same new variables as in Section 1.3:

$$E := A - BR^{-1}S^\mathsf{T} \qquad G := BR^{-1}B^\mathsf{T} \qquad \bar{Q} := Q - SR^{-1}S^\mathsf{T}$$

This yields the so-called *adjoint equations*:

$$\lambda_N = Q_f x_N \tag{12a}$$

$$\begin{bmatrix} x_{t+1} \\ \lambda_t \end{bmatrix} = \begin{bmatrix} E & -G \\ \bar{Q} & E^\mathsf{T} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_{t+1} \end{bmatrix} \tag{12b}$$

This is a difference equation with the state x_t equation evolving forward in time and co-state λ_t equation evolving backward in time. There is also a boundary condition that couples the variables at the terminal timestep. From here, we could prove $\lambda_t = P_t x_t$ using induction as we did in Section 1.3. Another approach is to rearrange (12) so that both equations go forward in time, which yields

$$\lambda_N = Q_f x_N \tag{13a}$$

$$\begin{bmatrix} I & G \\ 0 & E^\mathsf{T} \end{bmatrix} \begin{bmatrix} x_{t+1} \\ \lambda_{t+1} \end{bmatrix} = \begin{bmatrix} E & 0 \\ -\bar{Q} & I \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t \end{bmatrix} \tag{13b}$$

If E is invertible, we can invert the matrix on the left-hand side and write the equations as

$$\lambda_N = Q_f x_N$$

$$\begin{bmatrix} x_{t+1} \\ \lambda_{t+1} \end{bmatrix} = \begin{bmatrix} E + GE^{-\mathsf{T}}\bar{Q} & -GE^{-\mathsf{T}} \\ -E^{-\mathsf{T}}\bar{Q} & E^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t \end{bmatrix}$$

The 2×2 block matrix above is a *symplectic matrix* and has some useful properties, such as if λ is an eigenvalue, so is λ^{-1}. Such matrices play an important role in the study of Algebraic Riccati Equations. Consider a set of matrices $P_0, P_1, \dots, P_N$ and write:

$$\begin{aligned}
&\begin{bmatrix} x_{t+1} \\ \lambda_{t+1} - P_{t+1}x_{t+1} \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ -P_{t+1} & I \end{bmatrix} \begin{bmatrix} x_{t+1} \\ \lambda_{t+1} \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ -P_{t+1} & I \end{bmatrix} \begin{bmatrix} E + GE^{-\mathsf{T}}\bar{Q} & -GE^{-\mathsf{T}} \\ -E^{-\mathsf{T}}\bar{Q} & E^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ -P_{t+1} & I \end{bmatrix} \begin{bmatrix} E + GE^{-\mathsf{T}}\bar{Q} & -GE^{-\mathsf{T}} \\ -E^{-\mathsf{T}}\bar{Q} & E^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} I & 0 \\ P_t & I \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t - P_t x_t \end{bmatrix} \\
&= \begin{bmatrix} E + GE^{-\mathsf{T}}\bar{Q} - GE^{-\mathsf{T}}P_t & -GE^{-\mathsf{T}} \\ -P_{t+1}E - P_{t+1}GE^{-\mathsf{T}}\bar{Q} + P_{t+1}GE^{-\mathsf{T}}P_t + E^{-\mathsf{T}}P_t - E^{-\mathsf{T}}\bar{Q} & P_{t+1}GE^{-\mathsf{T}} + E^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t - P_t x_t \end{bmatrix}
\end{aligned}$$

Note that this holds for *any* choice of the P_t, since we added and subtracted it without changing anything. Consider the $(2,1)$ block of the transition matrix:

$$\begin{aligned}&-P_{t+1}E - P_{t+1}GE^{-\mathsf{T}}\bar{Q} + P_{t+1}GE^{-\mathsf{T}}P_t + E^{-\mathsf{T}}P_t - E^{-\mathsf{T}}\bar{Q}\\ &= -P_{t+1}E + (P_{t+1}G + I)E^{-\mathsf{T}}(P_t - \bar{Q})\end{aligned}$$

This can be made zero if we choose $P_t = \bar{Q} + E^{\mathsf{T}}P_{t+1}(I + GP_{t+1})^{-1}E$, which is precisely the alternative form for the solution we derived in (10). With this choice, our adjoint equations become:

$$\begin{bmatrix} x_{t+1} \\ \lambda_{t+1} - P_{t+1}x_{t+1} \end{bmatrix} = \begin{bmatrix} E + GE^{-\mathsf{T}}\bar{Q} - GE^{-\mathsf{T}}P_t & -GE^{-\mathsf{T}} \\ 0 & P_{t+1}GE^{-\mathsf{T}} + E^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t - P_t x_t \end{bmatrix}$$

Substituting for P_t and simplifying, we obtain:

$$\begin{bmatrix} x_{t+1} \\ \lambda_{t+1} - P_{t+1}x_{t+1} \end{bmatrix} = \begin{bmatrix} (I + GP_{t+1})^{-1}E & -GE^{-\mathsf{T}} \\ 0 & (I + P_{t+1}G)E^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t - P_t x_t \end{bmatrix}$$

Now recall from (10) that $A + BK_t = (I + GP_{t+1})^{-1}E$, so we have:

$$\begin{bmatrix} x_{t+1} \\ \lambda_{t+1} - P_{t+1}x_{t+1} \end{bmatrix} = \begin{bmatrix} A + BK_t & -GE^{-\mathsf{T}} \\ 0 & (A + BK_t)^{-\mathsf{T}} \end{bmatrix} \begin{bmatrix} x_t \\ \lambda_t - P_t x_t \end{bmatrix}$$

From here, we easily see that if $\lambda_{t+1} = P_{t+1}x_{t+1}$, then we must also have $\lambda_t = P_t x_t$ and this completes the proof. The equations also simplify to $x_{t+1} = (A + BK_t)x_t$, which are the closed-loop equations we expected to see.

Infinte-horizon LQR. This formulation using the adjoint equation is particularly useful when solving the infinite-horizon LQR problem. In the infinite-horizon setting, we have $P_t = P_{t+1} = P$, so the transformation of the symplectic matrix preserves eigenvalues, and we have:

$$\begin{bmatrix} I & 0 \\ -P & I \end{bmatrix} \underbrace{\begin{bmatrix} E + GE^{-\mathsf{T}}\bar{Q} & -GE^{-\mathsf{T}} \\ -E^{-\mathsf{T}}\bar{Q} & E^{-\mathsf{T}} \end{bmatrix}}_{M} \begin{bmatrix} I & 0 \\ P & I \end{bmatrix} = \begin{bmatrix} A + BK & -GE^{-\mathsf{T}} \\ 0 & (A + BK)^{-\mathsf{T}} \end{bmatrix}. \tag{14}$$

This observation is the key to solving the Discrete Algebraic Riccati Equation (DARE): eigenvalues of the symplectic matrix M are the eigenvalues of the LQR-optimal closed-loop map (stable) and their conjugate inverses (unstable). Multiply (14) by $\left[\begin{smallmatrix} I & 0 \\ P & I \end{smallmatrix}\right](\dots)\left[\begin{smallmatrix} I \\ 0 \end{smallmatrix}\right]$ and obtain

$$M \begin{bmatrix} I \\ P \end{bmatrix} = \begin{bmatrix} I \\ P \end{bmatrix} (A + BK). \tag{15}$$

The stable eigenvalues of M are the eigenvalues of $(A + BK)$. So if we diagonalize M and collect all stable eigenvalues in the diagonal matrix Λ, we can write the eigenvalue decomposition

$$M \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \Lambda.$$

Under suitable assumptions, V_1 will be invertible. Multiply on the right by V_1^{-1} and obtain

$$M \begin{bmatrix} I \\ V_2 V_1^{-1} \end{bmatrix} = \begin{bmatrix} I \\ V_2 V_1^{-1} \end{bmatrix} \left(V_1 \Lambda V_1^{-1}\right).$$

Note the similarity with (15). It takes some work to prove the details, but it turns out that $P = V_2 V_1^{-1}$ is the (unique) stabilizing solution to the DARE, and $V_1 \Lambda V_1^{-1} = A + BK$ is the LQR-optimal closed-loop map.