AA/EE547: Wi 22

Module 4: Linear Quadratic Regulator

Lecturer: L.J. Ratliff

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications, meaning you should take your own notes in class and review the provided references as opposed to taking these notes as your sole resource. I provide the lecture notes to you as a courtesy; it is not required that I do this. They may be distributed outside this class only with the permission of the Instructor.*

References. Chapter 11 and 15 [JH]; Chapter 8/8d, [C&D]. Note that 8d in [C&D] is the discrete time chapter on controllability and observability.

Contents

Introduction to Optimal Control Synthesis

The goal of this module is to introduce optimal control synthesis for linear systems given a quadratic cost function. We will review several methods for obtaining the optimal controller including dynamic programming, nonlinear programming, and the Hamiltonian approach. We will cover both discrete and continuous time. Discrete time provides a lot of intuition especially for dynamic programming and so we start with that.

1 M4-RL1: Nonlinear Programming Primer

This lecture's notes: basics of optimization (non-linear programming).

Goal: necessary and sufficient conditions for local optimality in non-linear programs

Refs: [Str14], pg. 29–41 [LSV12] Lewis Chapter 1

1.1 Unconstrained Non-Linear Programming

Suppose you are asked to minimize the function $F : \mathbb{R}^m \to \mathbb{R} : u \mapsto F(u)$, i.e. to *solve* the **nonlinear program** (NLP)

$$\min_{u\in\mathbb{R}^m} F(u)$$

F is called the **objective function**.

Definition 1. The following are definitions of minimizers/maximizers:

- We say that $u \in \mathbb{R}^m$ is a local minimizer for NLP if there exists an open set $V \subset \mathbb{R}^m$, containing u such that
$$F(u) \leq F(v), \quad \forall v \in V$$
- We say that u is a strict local minimizer if the inequality is strict ($<$).
- We say that u is a local maximizer if it is a local minimizer for
$$\min_{u\in\mathbb{R}^m} -F(u).$$

Consider cost $F : \mathbb{R} \to \mathbb{R}$. We can expand F about a nominal point u_0 using a Taylor approximation:

$$F(u) = F(u_0) + D_uF(u)|_{u=u_0}(u-u_0) + \frac{1}{2}D^2F(u)|_{u=u_0}(u-u_0)^2 + \underbrace{o((u-u_0)^3)}_{\text{h.o.t.}}$$

where h.o.t. stands for higher order terms. We can define perturbations in F and u:

$$\begin{aligned}\delta F &= F(u) - F(u_0)\\ \delta u &= u - u_0\end{aligned}$$

and neglecting h.o.t., the expansion leads to

$$\delta F = D_uF(u)|_{u=u_0}\delta u + \frac{1}{2}D_u^2F(u)|_{u=u_0}(\delta u)^2$$

If u_0 is a stationary point, $D_uF(u) = 0$ and the location variation in F is entirely due to second (and higher) derivatives of the function:

$$\delta F = \frac{1}{2}D_u^2F(u)|_{u=u_0}(\delta u)^2$$

Hence the variation of F in the neighborhood of the stationary point is well approximated as a quadratic form of the scalar variable δu. This observation will prove useful in later consideration of more complex problems.

This whole thing extends to the non-scalar case $F : \mathbb{R}^m \to \mathbb{R}$. The Taylor approximation is given by

$$F(u) = F(u_0) + D_uF(u)|_{u=u_0}(u-u_0) + \frac{1}{2}(u-u_0)^\top(D_u^2F(u)|_{u=u_0})(u-u_0) + o(\|u-u_0\|^3)$$

Note that here we are using the appropriate co-vector form of D_uF:

$$D_uF(u_0) = [D_{u_1}F(u_0) \ \cdots \ D_{u_m}F(u_0)]$$

The second derivative $D_u^2F(u)$ is just the Hessian:

$$D_u^2F(u_0) = \begin{bmatrix} D_{11}^2F(u) & D_{12}^2F(u) & \cdots & D_{1m}^2F(u)\\ D_{21}^2F(u) & \ddots & \ddots & D_{1m}^2F(u)\\ \vdots & \vdots & \ddots & \vdots\\ D_{m1}^2F(u) & D_{m2}^2F(u) & \cdots & D_{mm}^2F(u)\end{bmatrix}_{u=u_0}$$

F is critical at the point $u = u_0$ simultaneous variations in all the components of u have negligible effect on the value of F. The gradient of F must be zero at a stationary point:

$$D_u F(u_0) = 0$$

And of course, the Hessian is

- positive definite at a min
- negative definite at max
- neither at an inflection (saddle) point

Definition 2. We say $u_0 \in \mathbb{R}^m$ is a critical point for NLP if $DF(u_0) = 0$.

This assumes that $F \in C^1$ at u_0.

Theorem 3 (sufficient conditions for optimality)**.** A stationary point $u_0 \in \mathbb{R}^m$ is

- a strict local min if $D^2F(u_0) \succ 0$
- a strict local max if $D^2F(u_0) \prec 0$

Assumes $F \in C^2$ at u_0.

Theorem 4 (necessary conditions for optimality)**.** If $u_0 \in \mathbb{R}^m$ is a local minimizer for NLP:

- if $F \in C^1$ at u_0 then $DF(u_0) = 0$
- if $F \in C^2$ at u_0 then $D^2F(u_0) \succeq 0$

Actually defining the value of u that causes F to be minimized can be difficult if the dimension of u is high or if the shape of F is intricate.

In such a case, an iterative numerical technique such as **Newton-Raphson** can be used. If the second order approximation fits F exactly in the vicinity of the minimum, where $u = u^\star$, then the following should hold:

$$0 = D_u F(u^\star) = D_u F(u_0) + (u^\star - u_0)^\top D_u^2 F(u_0)$$

This leads to

$$u^\star = u_0 - (D_u^2 F(u_0))^{-1} D_u F(u_0)^\top$$

Example. Suppose second order approx. to F at $u_0 \in \mathbb{R}^m$ is exact, i.e.,

$$(F(u) - F(u_0)) = b^\top (u - u_0) + \frac{1}{2}(u - u_0)^\top C(u - u_0)$$

1. Determine necessary conditions on $b^\top \in \mathbb{R}^{1\times m}$, $C^\top = C$ for $u_0 \in \mathbb{R}^m$ to be local min:
   It is necessary that $DF(u_0) = b^\top + (u_0 - u_0)^\top C = b^\top = 0$, and $D^2F(u_0) = C = 0$.
2. Determine sufficient conditions on spec$DF(v)$ for $u_0 \in \mathbb{R}^m$ to be a strict local min:
   It is sufficient that $\forall\ \sigma \in D^2F(v) : \sigma > 0$.
3. Now assuming a strict local min, solve for u_0:
   In this case $D^2F(v)$ is invertible because all the eigenvalues are greater than 0, so
   $$u_0 = u - (D^2F(u))^{-1} DF = F(u)^\top$$
   is a strict local min for any $u \in \mathbb{R}^m$. This is called the Newton-Raphson formula.

Q: Why does open set matter for local optimality?
A: It doesnt really in the sense that you can define a local optimum on a non-empty compact (closed, bounded in $\mathbb{R}^n$). However, we define it on an open set as I mentioned due to the fact we define local optimality with respect to derivatives and those need to be define on more than just a point (that is the idea with derivatives, that we are looking at what the function does for small perturbations).

1.2 Constrained NLP

Goal: Sufficient conditions for local optimality in NLP subject to constraints.

Generally, a constrained NLP is given by the following:

$$\min_{u\in\mathbb{R}^m} F(u) \quad \text{s.t.} \quad f(u) = 0$$

where $F : \mathbb{R}^l \to \mathbb{R}$, $f : \mathbb{R}^l \to \mathbb{R}^n$.

Let us first build up to this.

Our function $F : \mathbb{R}^\ell \to \mathbb{R} : u \mapsto F(u)$ from before can represent a scalar cost function of the m dimensional control vector u; hence, the problem of finding the value of u that minimizes F provides an introduction to optimal control.

In the unconstrained case, ℓ components of u must be defined in order to specify the minimum of F.

Now, if we add a **scalar equality constraint**

$$f(u) = 0$$

then there are at most $\ell - 1$ independent components of u. That is the order of the optimization problem has been reduced by one.

If we add a **vector equality constraint**

$$f(u) = 0$$

where $f(u) \in \mathbb{R}^{\ell_1}$ and the dimension of u is $\ell = \ell_1 + \ell_2$. Then ℓ_1 components of u are specified as functions of the remaining ℓ_2 terms which are varied until $F(u)$ is at a minimum.

One option is to partition u as $u = u_1 + u_2$ with $u_i \in \mathbb{R}^{\ell_i}$. Then we are interested in solving

$$\begin{array}{ll} \min_{u\in\mathbb{R}^m} & F(u_1, u_2) \\ \text{s.t.} & f(u_1, u_2) = 0 \end{array}$$

Alternatively, we can make the following changes in notation (getting closer to optimal control!):

$$\begin{aligned} x &= u_1 \in \mathbb{R}^n \\ u &= u_2 \in \mathbb{R}^m \end{aligned}$$

Then we want to solve

$$\begin{array}{ll} \min_{u\in\mathbb{R}^l} & F(x, u) \\ \text{s.t.} & f(x, u) = 0 \end{array}$$

Of course, we can think of x as the **state vector** and u as the **control vector**.

Coming back to our original constrained NLP:

$$\begin{array}{ll} \min_{u\in\mathbb{R}^m} & F(w) \\ \text{s.t.} & f(w) = 0 \end{array}$$

If we split $w \in \mathbb{R}^l$ into $w = (x, u) \in \mathbb{R}^n \times \mathbb{R}^n$, solve equation $f(x, u) = 0$ for x in terms of $u, x : \mathbb{R}^m \to \mathbb{R}^n$, then could equivalently solve

$$\min_{u\in\mathbb{R}^m} F(x(u), u)$$

In general it is hard to solve $f(x, u) = 0$.

Note. You may have also seen inequality constraints such as $f(w) \geq 0$. We will not deal with those for now.

Example: We can show how in principle to reduce this to an unconstrained NLP.

Constraints can be dealt with in a number of ways including the following naïve way. You did this on your homework hopefully. Consider the cost

$$F(u_1, u_2) = u_1^2 - 2u_1u_2 + 3u_2^2 - 40$$

Suppose this represents some bowl shaped valley. There is a straight path through the valley defined by

$$0 = f(u_1, u_2) = -u_2 + u_1 + 2$$

What is the lowest point on the path?

First this can be written as a constrained problem:

$$\min_{u_1,u_2} \left\{u_1^2 - 2u_1u_2 + 3u_2^2 - 40 \mid 0 = -u_2 + u_1 + 2\right\}$$

Q: what is the naïve thing to do?

A: solve the constraint for one variable, say u_2, and plug it in to the cost.

$$\min_{u_1} \left\{u_1^2 - 2u_1(u_1 + 2) + 3(u_1 + 2)^2 - 40\right\}$$

We can solve this now **unconstrained** problem in the usual way. First order conditions:

$$0 = \frac{dF}{du_1} \quad = \quad 2u_1 - 4u_1 - 4 + 6(u_1 + 2) = 4u_1 + 8 \implies u_1^\star = -2,\ u_2^\star = 0,\ F(u_1^\star, u_2^\star) = -36$$

Q (DIY): How does this compare with the optimal solution of the unconstrained problem?

An alternative view. Instead of reducing the unconstrained NLP, we augment the cost function to be

$$\tilde{F}(x, u, \lambda) = F(x, u) + \lambda f(x, u),$$

where there are $n + m + n$ unknowns:

$$x \in \mathbb{R}^n,\ u \in \mathbb{R}^m,\ \lambda \in \mathbb{R}^{1\times n}.$$

- n equations to define λ
- n equations to find x
- m equations to find u

Defn. $\lambda \in \mathbb{R}^n$ are called the **Lagrange multipliers**. (sometimes called adjoint vector or costate vector). $\lambda f(x, u) \in \mathbb{R}$ is a scalar and it always equals zero when $f(x, u) = 0$ is satisfied.

History Lesson. Giuseppe Lodovico Lagrangia (aka Joseph Lagrange) (1736–1813) derived methods for solving variational problems in classical mechanics. He was an Italian Enlightenment Era mathematician and astronomer. He was so good that Euler, Maupertuis, and d'Alembert tried to encourage him to come to Berlin. But, he wouldn't go until Euler left.

As in the unconstrained case, for (x_0, u_0, λ_0) to be local min for $\tilde{F}$, it is necessary that $D\tilde{F}(x_0, u_0, \lambda_0) = 0$.

Perturbations in x and u lead to perturbations in $\tilde{F}$. To first order,

$$\delta\tilde{F} = D_x\tilde{F}\big|_{(x,u)=(x_0,u_0)}\delta x + D_u\tilde{F}\big|_{(x,u)=(x_0,u_0)}\delta u$$

where

$$\begin{aligned} D_x\tilde{F} &= D_xF + \lambda D_xf \\ D_u\tilde{F} &= D_uF + \lambda D_uf \end{aligned}$$

Note. The partial derivative of f wrt u is the $n \times m$ Jacobian matrix with elements $D_{u_j} f_i$. And, the partial derivative of f wrt x is the $n \times n$ Jacobian matrix with elements $D_{x_j} f_i$.

History Lesson. Carl G. J. Jacobi (1804–1851)

Q: Determine necessary condition on λ_0, assuming $D_x f(x_0, u_0)$ is invertible.

A: $D_x \tilde{F} = D_x F + \lambda D_x f$, so if $D_x f(x_0, u_0)$ invertible then its necessary that

$$\lambda_0 = -D_x F(x_0, u_0)[D_x f(x_0, u_0)]^{-1}$$

(i.e. $D_x \tilde{F} \equiv 0$ when $\lambda = \lambda_0$ is chosen as above).

$D_u \tilde{F} = D_u F + \lambda D_u f$, so necessary that

$$D_u F(x_0, u_0) + \lambda_0 D_u f(x_0, u_0) = 0$$

$D_\lambda \tilde{F} = f$, so also necessary that $f(x_0, u_0) = 0$

We now have enough equations to specify stationary points for constrained NLP.

Definition 5 (Constrained NLP Stationary Point)**.** $(x_0, u_0) \in \mathbb{R}^{n \times m}$ is a stationary point for

$$\begin{array}{ll} \min_{(x,u) \in \mathbb{R}^{n \times m}} & F(x, u) \\ \text{s.t.} & f(x, u) = 0 \end{array}$$

if

- $D_u F(x_0, u_0) + \lambda_0 D_u f(x_0, u_0) = 0$
- $f(x_0, u_0) = 0$
- $D_x f(x_0, u_0)$ is invertible
- $\lambda_0 = -D_x F(x_0, u_0)[D_x f(x_0, u_0)]^{-1}$

The second order condition for optimality: First, write

$$\tilde{F}(x, u, \lambda) = F(x, u) + \lambda f(x, u)$$

so that

$$\tilde{F}_x = D_x \tilde{F} = D_x F + \lambda D_x f = F_x + \lambda f_x$$

Then the Taylor expansion gives

$$D\tilde{F} = \begin{bmatrix} F_x^\top & F_u^\top \end{bmatrix} \begin{bmatrix} dx \\ du \end{bmatrix} + \frac{1}{2} \begin{bmatrix} dx^\top & du^\top \end{bmatrix} \begin{bmatrix} F_{xx} & F_{xu} \\ F_{ux} & F_{uu} \end{bmatrix} \begin{bmatrix} dx \\ du \end{bmatrix} + O(3)$$

(where $O(k)$ denotes higher order terms in the expansion of order k and above) and

$$Df = \begin{bmatrix} f_x^\top & f_u^\top \end{bmatrix} \begin{bmatrix} dx \\ du \end{bmatrix} + \frac{1}{2} \begin{bmatrix} dx^\top & du^\top \end{bmatrix} \begin{bmatrix} f_{xx} & f_{xu} \\ f_{ux} & f_{uu} \end{bmatrix} \begin{bmatrix} dx \\ du \end{bmatrix} + O(3)$$

where, e.g.,

$$F_{xx} = D_x^2 F, \ F_{xu} = D^2 xu F, \ f_{xx} = D_x^2 f, \ f_{xu} = D_{xu}^2 f$$

and so on and the 'big O' notation just means 'higher order terms' for our purpose. Thus,

$$\begin{bmatrix} 1 & \lambda \end{bmatrix} \begin{bmatrix} DF \\ Df \end{bmatrix} = \begin{bmatrix} \tilde{F}_x^\top & \tilde{F}_u^\top \end{bmatrix} \begin{bmatrix} dx \\ du \end{bmatrix} + \frac{1}{2} \begin{bmatrix} dx^\top & du^\top \end{bmatrix} \begin{bmatrix} \tilde{F}_{xx} & \tilde{F}_{xu} \\ \tilde{F}_{ux} & \tilde{F}_{uu} \end{bmatrix} \begin{bmatrix} dx \\ du \end{bmatrix} + O(3)$$

At critical points (stationary points) $\tilde{F}_x = 0$, $\tilde{F}_u = 0$, and $Df = 0$ (necessarily) so that the above equation for Df implies

$$dx = -f_x^{-1} f_u du + O(2)$$

We can plug this in the above expression to get that

$$D\tilde{F} = \frac{1}{2} du^\top \begin{bmatrix} -f_u^\top f_x^{-\top} & I \end{bmatrix} \begin{bmatrix} \tilde{F}_{xx} & \tilde{F}_{xu} \\ \tilde{F}_{ux} & \tilde{F}_{uu} \end{bmatrix} \begin{bmatrix} -f_x^{-1} f_u \\ I \end{bmatrix} du + O(3)$$

Note that dx is replaced with an expression of du due to the first-order optimality conditions.

To ensure a minimum, $D\tilde{F}$ above should be positive for all increments du. This is guaranteed if the **curvature matrix** with constant f equal to zero, i.e.

$$\begin{aligned} D_u^2 \tilde{F} = \tilde{F}_{uu} &= \begin{bmatrix} -f_u^\top f_x^{-\top} & I \end{bmatrix} \begin{bmatrix} \tilde{F}_{xx} & \tilde{F}_{xu} \\ \tilde{F}_{ux} & \tilde{F}_{uu} \end{bmatrix} \begin{bmatrix} -f_x^{-1} f_u \\ I \end{bmatrix} \\ &= F_{uu} - f_u^\top f_x^{-\top} F_{xu} - F_{ux} f_x^{-1} f_u + f_u^\top f_x^{-\top} F_{xx} f_x^{-1} f_u, \end{aligned}$$

is positive definite. Note that if the constraint $f(x, u)$ is identically zero for x, u then the above reduces to the unconstrained case (unsurprisingly).

Moreover, if $\tilde{F}_{uu}$ is negative definite (indefinite), then the stationary point is a constrained maximum (saddle point).

Theorem 6 (Sufficient Conditions for Optimality)**.** A stationary point $(x_0, y_0) \in \mathbb{R}^{n \times m}$ is:

1. A strict local min if $D_u^2 \tilde{F}(x_0, u_0) > 0$.
2. A strict local max if $D_u^2 \tilde{F}(x_0, u_0) < 0$.

References

[Str14] S. Strogatz, 'Nonlinear Dynamics and Chaos,' *Westview Press*, Ed: 2nd , 2014.

[LVS12] F.L. Lewis, D.L. Varbie, and V.L. Syrmos, "Optimal Control," *John Wiley & Sons, Inc.*, 2012.

2 M4-RL2: Dynamic Programming and Discrete Time LQR

Goal: Learn the basics of dynamic programming via discrete time linear quadratic regulator (LQR)

2.1 Bellman's Principle

In previous lectures, we have discussed the Lyapunov equation and NLP. We will see how these two things combine to enable us to solve optimal control problems.

We will start with discrete time in order to give some intuition for dynamic programming and to better connect with the NLP framework we recently discussed.

Suppose we have a DT difference equation

$$x^+ = f(x, u), \ x \in \mathbb{R}^n, \ u \in \mathbb{R}^m$$

and we want to choose inputs over time $u : [0, t] \to \mathbb{R}^m$ to minimize some cost

$$V(x, u) = \underbrace{\ell(t, x(t))}_{\text{'final' cost}} + \sum_{k=0}^{t-1} \underbrace{L(k, x(k), u(k))}_{\text{'running' cost}}$$

History Lesson: Richard Bellman developed the key insight for 'recursion' in optimal control in 1957. That is, the optimal control at time t depends only on $x(t)$ (i.e. not on previous states).

This key insight leads naturally to working backward from final time to determine optimal *policy*.

Let $V_k^\star(x(k))$ denote the lowest (i.e. optimal) cost achievable from state $x(k)$ at time k,

$$V_k^\star(x(k)) = \min_{u(k)\in\mathbb{R}^m} \big(L(k, x(k), u(k)) + V_{k+1}^\star(x(k+1))\big)$$

where

$$x(k+1) = f(x(k), u(k))$$

depends on $u(k)$.

This is referred to as the **Bellman Equation** and it enables us (in principal) to determine optimal control inputs by solving a sequence of NLP in backward time!

2.2 DT Linear Quadratic Regulator

Consider

$$x(t+1) = Ax(t) + Bu(t), \; x(0) = x_0$$

and

$$J(u) = x_N^T Q_f x_N + \sum_{k=0}^{N-1} \big(x(k)^T Q x(k) + u(k)^T R u(k)\big)$$

where $u = (u(0), \ldots, u(N-1))$ and $Q = Q^T \geq 0$, $Q_f = Q_f^T \geq 0$, $R = R^T > 0$ are the given state cost, final state cost, and input cost matrices.

- N is the time horizon
- first term measures state deviation
- second term measures input size or actuator authority
- last term measures final state deviation
- Q, R set relative weights of state deviation and input usage
- $R > 0$ means any (non-zero) input adds cost to J

LQR Problem: find $u^\star$ that minimizes $J(u)$.

Q: how does this compare to other problems you may be familiar with?

2.2.1 Comparison to Least Norm

Consider the least norm input problem where you must determine the least norm input that steers x to $x_N = 0$:

- no cost attached to $x_0, \ldots, x_{N-1}$
- x_N must be exactly zero

We can approximate this problem in the above framework by letting $R = I$, $Q = 0$ and $Q_f >> I$ (e.g., $Q_f = 10^8 I$).

2.2.2 Multi-Objective Interpretation

Common form for Q and R:

$$R = \rho I, \; Q = Q_f = C^T C$$

where $C \in \mathbb{R}^{p \times n}$ (p size of output), $\rho \in \mathbb{R}$ with $\rho > 0$. Fix $x(0) = x_0$.

The cost is then

$$J(u) = \underbrace{\sum_{k=0}^{N} \|y(k)\|^2}_{\text{output cost: } J_o(u)} + \rho \underbrace{\sum_{k=0}^{N-1} \|u(k)\|^2}_{\text{input cost: } J_i(u)}$$

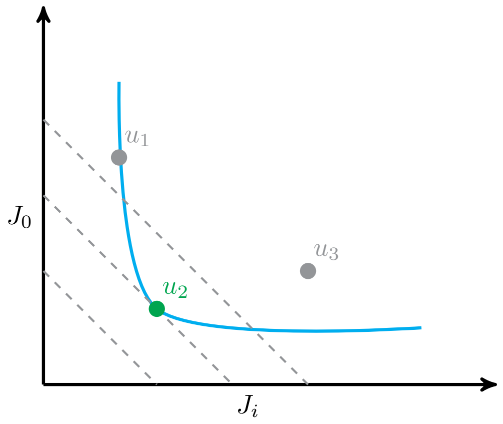
where

$$y = Cx$$

Here, $\sqrt{\rho}$ gives the relative weighting of output norm and input norm.

The input and output costs $J_i(u)$ and $J_o(u)$ respectively are competing objectives. We want both of them to be *small*.

Then the LQR cost is $J_o(u) + \rho J_i(u)$.



- Above the cyan curve shows (J_i, J_o) achieved by some u
- below the same cure shows (J_i, J_o) not achieved by any u
- Three samples inputs u_1, u_2, u_3 are shown
- u_3 is worse than u_2 on both parameters J_i, J_o
- u_1 is better than u_2 in J_i but worse in J_o

Where $J = J_o + \rho J_i =$ constant corresponds to lines with slope $-\rho$ on the (J_i, J_o) plot.

LQR optimal input is at boundary of shaded region, just touching the line of smallest possible J. Hence, u_2 in the plot is optimal for ρ. By varying ρ from 0 to $+\infty$, can sweep optimal tradeoff curve.

2.3 LQR as least squares

LQR can be formulated and solved as a least squares problem. Let $X = (x_0, \dots, x_N)$ where $x_k = x(k)$. The state is a linear function of x_0 and $u = (u_0, \dots, u_{N-1})$:

$$\begin{bmatrix} x_0 \\ \vdots \\ x_N \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \cdots & \cdots & \cdots \\ B & 0 & \cdots & \cdots \\ AB & B & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} u_0 \\ \vdots \\ u_{N-1} \end{bmatrix}}_{u} + \underbrace{\begin{bmatrix} I \\ A \\ \vdots \\ A^N \end{bmatrix}}_{H} x_0$$

Then we can express $X = Gu + Hx_0$ where $G \in \mathbb{R}^{Nn \times Nm}$ and $H \in \mathbb{R}^{Nn \times m}$

Thus the LQR cost is

$$J(u) = \left\| \text{diag}(Q^{1/2}, \dots, Q^{1/2}, Q_f^{1/2})(Gu + Hx_0) \right\|^2 + \left\| \text{diag}(R^{1/2}, \dots, R^{1/2}u \right\|^2$$

This is simply a big least squares problem! The solution method requires forming and solving a Least squares problem with size $N(n+m) \times Nm$. Using a naïve method (e.g., QR factorization), cost is $O(N^2nm^2)$.

2.4 Dynamic Programming Solution

- givens an efficient, recursive method to solve LQR least-squares problem; cost is $O(Nn^3)$
- (but in fact, a less naive approach to solve the LQR least-squares problem will have the same complexity)
- DP is a useful and important idea on its own (it is applied in a number of domains including search, RL, MDPs, etc.)

Definition 7. For $t = 0, \dots, N$ define the **value function** $V_t : \mathbb{R}^n \to \mathbb{R}$ by

$$V_t(z) = \min_{u_t, \dots, u_{N-1}} \sum_{k=t}^{N-1} (x_k^T Q x_k + u_k^T R u_k) + x_N^T Q_f x_N$$

subject to

$$x_t = z, \ x_{k+1} = Ax_k + Bu_k, \ k = t, \dots, N$$

- $V_t(z)$ gives the **minimum LQR cost-to-go**, starting from state z at time t
- $V_0(x_0)$ is minimum LQR cost (from state x_0 at time 0

We will see that

- V_t is quadratic, i.e. $V_t(z) = z^T P_t z$, where $P_t = P_t^T \geq 0$ (this should look super familiar, like a Lyapunov function and hence we can tie this back to what we initially learned at the beginning of the quarter)
- P_t can be found recursively, working backward from $t = N$
- the LQR optimal u is easily expressed in terms of P_t

The **cost to go** with no time left is just the final state cost

$$V_N(z) = z^T Q_f z$$

Thus, we have that $P_N = Q_f$

2.4.1 Dynamic Programming Principle

Q: Suppose we know $V_{t+1}(z)$. What is the optimal choice for u_t?

choice of u_t impacts current cost incurred (through $u_t^T R u_t$) and where we end up, i.e. x_{t+1} (hene, the min-cost-to-go from x_{t+1}

Definition 8. The dynamic programming principle is

$$V_t(z) = \min_w (z^T Q z + w^T R w + V_{t+1}(Az + Bw)$$

where

- $z^T Q z + w^T R w$ is cost incurred at time t if $u_t = w$
- $V_{t+1}(Az + Bu)$ is min cost-to-go from where you land at $t+1$

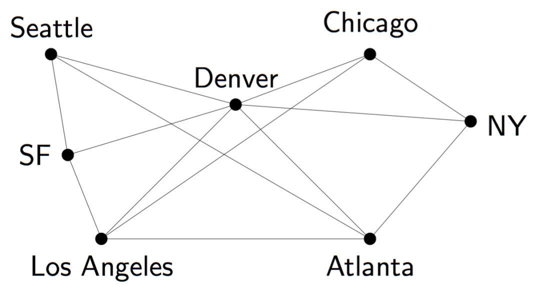
This follows from the fact that we can minimize in any order:

$$\min_{w_1,\ldots,w_k} f(w_1,\ldots,w_k) = \min_{w_1} \underbrace{\left(\min_{w_2,\ldots,w_k} f(w_1,\ldots,w_k) \right)}_{\text{a fn of } w_1}$$

in words: min cost-to-go from where you are = min over (current cost incurred + min cost-to-go from where you land)

Example: path optimization

- edges show possible flights and each has some cost (weight)
- e.g., want to find min cost route or path from Seattle to Atlanta



Q: what is the DP solution in this context?

- $V(i)$ is min cost from airport i to ATL, over all possible paths
- to find min cost from city i to ATL: minimize sum of flight cost plus min cost to ATL from where you land, over all flights out of city i (gives optimal flight out of city i on way to ATL)
- if we can find $V(i)$ for each i, we can find min cost path from any city to ATL
- **DP principle**: $V(i) = \min_j (c_{ji} + V(j))$, where c_{ji} is cost of flight from i to j, and minimum is over all possible flights out of i

2.4.2 HJ equation for LQR

$$V_t(z) = z^T Q z + \min_w (w^T R w + V_{t+1}(Az + Bw))$$

- called DP, Bellman, or Hamilton-Jacobi equation
- gives V_t recursively in terms of V_{t+1}
- any minimizing w gives optimal u_t:

$$u_t^\star = \arg\min_w (w^T R w + V_{t+1}(Az + Bw))$$

Let us assume that $V_{t+1}(z) = z^T P_{t+1} z$, with $P_{t+1} = P_{t+1}^T \geq 0$. We will show that V_t has the same form!

By DP,

$$V_t(z) = z^T Q z + \min_w \left(w^T R w + (Az + Bw)^T P_{t+1}(Az + Bw) \right)$$

Here comes the NLP! we can solve by setting the derivative wrt w to zero:

$$2w^T R + 2(Az + Bw)^T P_{t+1} B = 0$$

Hence the optimal input is

$$w^\star = -(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A z$$

and after a bunch of algebra (check your self!) we get

$$\begin{aligned} V_t(z) &= z^T Q z + (w^\star)^T R w^\star + (Az + Bw^\star)^T P_{t+1}(Az + Bw^\star) \\ &= z^T (Q + A^T P_{t+1} A - A^T P_{t+1} B (R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A) z \\ &= z_t^P z \end{aligned}$$

where

$$P_t = Q + A^T P_{t+1} A - A^T P_{t+1} B (R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A$$

This should look very familiar (discrete time Lyapunov?). Hence, it is easy to show that $P_t^T = P_t \geq 0$ (do this your self!)

2.5 Summary of LQR via DP

step 1 set $P_N = Q_f$

step 2 for $t = N, \ldots, 1$, do

$$P_{t-1} = Q + A^T P_t A - A^T P_t B (R + B^T P_{t+1} B)^{-1} B^T P_t A$$

step 3 for $t = 0, \ldots, N-1$, define

$$K_t = -(R + B^T P_{t+1} B)^{-1} B^T P_{t+1} A$$

step 4 for $t = 0, \ldots, N-1$, optima u is given by

$$u_t^\star = K_t x_t$$

What else do we notice?

- optimal u is a linear function of the state (called linear state feedback)
- recursion for min cost-to-go runs backward in time

2.6 Final Comments on DT LQR

The following are some additional comments:

- another name for the recursion for P_t is the Riccati recursion
- usually P_t rapidly converges as t decreases below N (remember we are going backward in time)
- the limit or steady-state (ss) value P_{ss} satisfies the following Riccati equation:

$$P_{ss} = Q + A^\top P_{ss} A - A^\top P_{ss} B (R + B^\top P_{ss} B)^{-1} B^\top P_{ss} A$$

  i.e. the DT algebraic Riccati equation (ARE)
- for t not close to the horizon N, LQR optimal input is approximately a linear constant state feedback

$$u_t = K_{ss} x_t, \;\; K_{ss} = -(R + B^\top P_{ss} B)^{-1} B^\top P_{ss} A$$

3 M4-RL3: LQR via NLP

Goal: Convert the DT LQR problem to a constrained NLP and solve via the adjoint method.

References: [Str14] §3.4; [LVS12] Lewis §6

3.1 LQR via constrained NLP

Matrix inversion identities:

1. $(I + AB)^{-1} = I - A(I + AB)^{-1} B$
2. $A(I + AB)^{-1} = (I + BA)^{-1} B$
3. $(I + AC^{-1}B)^{-1} = I - A(C + BA)^{-1} B$

Recall the following matrix inversion lemma from linear algebra or [510]:

Lemma 9. For appropriately dimensioned matrices, the following holds:

$$(A + BC)^{-1} = A^{-1} - A^{-1} B (I + CA^{-1} B)^{-1} CA^{-1}$$

This allows us to convert the Riccati recursion into a different form:

$$\begin{aligned}
P_{t-1} &= Q + A^\top P_t A - A^\top P_t B (R + B^\top P_t B)^{-1} B^\top P_t A \\
&= Q + A^\top P_t (I - B(R + B^\top P_t B)^{-1} B^\top P_t A \\
&= Q + A^\top P_t (I - B \left((I + B^\top P_t B R^{-1}) R \right)^{-1} B^\top P_t A \\
&= Q + A^\top P_t (I - BR^{-1} (I + B^\top P_t B R^{-1})^{-1} B^\top P_t) A \\
&= Q + A^\top P_t (I + BR^{-1} B^\top P_t)^{-1} A \\
&= Q + A^\top (I + P_t B R^{-1} B^\top)^{-1} P_t A
\end{aligned}$$

which can be converted to the following symmetric form

$$P_{t-1} = Q + A^\top P_t^{1/2} (I + P_t^{1/2} B R^{-1} B^\top P_t^{1/2})^{-1} P_t^{1/2} A$$

3.2 Recall NLP's

Recall the constrained NLP's we looked at before the midterm:

$$\begin{aligned} \min_x \;& F(x) \\ \text{s.t. } & Gx = h \end{aligned}$$

where $F : \mathbb{R}^n \to \mathbb{R}$ and $G \in \mathbb{R}^{m\times n}$.

The Lagrangian is given by

$$L(x,\lambda) = F(x) + \lambda^\top (h - Gx)$$

If x is optimal, then

$$D_x L = DF(x) - G^\top \lambda = 0,\; D_\lambda L = h - Gx = 0$$

Interesting connection to the linear algebra we saw last quarter ([510]): the first equation implies that $DF(x) = G^\top \lambda$ for some λ. This is true if and only if $DF(x)^\top \in \mathcal{R}(G^\top)$ if and only if $DF(x)^\top \perp \mathcal{N}(G)$ (behold! the many applications of FROL—Finite Rank Operator Lemma).

3.3 Recall Descent Directions

Suppose x is current, feasible point (i.e. $Gx = h$). Consider a small step in the direction v, to $x + \gamma v$ (γ small, positive step size).

Q: When is $x + \gamma v$ better than x?

A: We need the following:

1. $x + \gamma v$ to be feasible:
$$G(x + \gamma v) = h + \gamma G v = h \implies Gv = 0$$
Hence, $v \in \mathcal{N}(G)$ is called a **feasible direction**.
2. $x + \gamma v$ to have a smaller objective than x:
$$F(x + \gamma v) \simeq F(x) + \gamma DF(x) v < F(x)$$
so we need $DF(x)v < 0$ (i.e. **descent direction**)
Note that if $DF(x)v0$, $-v$ is a descent direction, so we need only $DF(x)v \neq 0$.

x is not optimal if there exists a feasible descent direction.

if x is optimal, every feasible direction satisfies $DF(x)v = 0$. Hence,

$$\begin{aligned} Gv = 0 \implies DF(x)v = 0 \iff & \mathcal{N}(G) \subset \mathcal{N}(DF(x)) \\ \iff & \mathcal{R}(G^\top) \supset \mathcal{R}(DF(x)^\top) \\ \iff & DF(x)^\top \in \mathcal{R}(G^\top) \\ \iff & DF(x)^\top = G^\top \lambda \;\text{ for some } \lambda \in \mathbb{R}^n \\ \iff & DF(x)^\top \perp \mathcal{N}(G) \end{aligned}$$

3.4 LQR as Constrained Minimization Problem

Consider the following

$$\begin{aligned} \min_u \;\; & J = \frac{1}{2}\sum_{t=0}^{N-1}\left(x_t^\top Q x_t + u_t^\top R u_t\right) + \frac{1}{2} x_N^\top Q_f x_N \\ \text{s.t. } \;\; & x_{t+1} = Ax_t + Bu_t,\; t = 0, \ldots, N-1 \end{aligned}$$

- variables are $u_0, \ldots, u_{N-1}$ and $x_1, \ldots, x_N$ (x's can actually be specified in terms of u's via the constraint as we have seen before).
- x_0 is given
- the objective is quadratic (i.e. it is convex)

First step is to introduce Lagrange multipliers.

Q: How many do we need?

A: $\lambda_i \in \mathbb{R}^n$ for $i = 1, \ldots, N$.

Next step is to form the Lagrangian:

$$L = J + \sum_{t=0}^{N-1} \lambda_{t+1}^\top (Ax_t + Bu_t - x_{t+1})$$

Optimality Conditions. **Q**: What are the optimality conditions?

A: First, x_0 is given and we know that

$$x_{t+1} = Ax_t + Bu_t,\ t = 0, \ldots, N-1$$

We can take the derivative of L with respect to u_t to get

$$D_{u_t} L = Ru_t + B^\top \lambda_{t+1} = 0,\ t = 0, \ldots, N-1$$

Hence,

$$u_t = -R^{-1} B^\top \lambda_{t+1}$$

Similarly, we can write first order conditions for x_t:

$$D_{x_t} L = Qx_t + A^\top \lambda_{t+1} - \lambda_t = 0,\ t = 0, \ldots, N-1$$

Hence

$$\lambda_t = A^\top \lambda_{t+1} + Qx_t$$

Finally,

$$D_{x_N} L = Q_f x_N - \lambda_N = 0$$

so that $\lambda_N = Q_f x_N$

This is a set of linear equations in the variables

$$u_0, \ldots, u_{N-1},\ x_1, \ldots, x_N,\ \lambda_1, \ldots, \lambda_N$$

Co-State Equations. **Q**: What do you notice about these equations?

A: we have a forward equation for x and a backward equation for λ...

Optimality conditions break into two parts:

$$x_{t+1} = Ax_t + Bu_t,\ x_0$$

this recursion for state x runs forward in time, with initial condition

$$\lambda_t = A^\top \lambda_{t+1} + Qx_t,\ \lambda_N = Q_f x_N$$

this recursion for **co-state** λ runs backward in time, with final condition

As mentioned before, the recursion for λ is sometimes called the **adjoint system**.

Solution via Riccati Recursion. We will see that $\lambda_t = P_t x_t$ where P_t is the min-cost-to-go matrix defined by the Riccati recursion we saw last time. This implies that the Riccati recursion gives a useful way to solve this set of linear equations.

Q: How do we show this?

A: via induction like last time.

First, it holds for $t = N$, since $P_N = Q_f$ and $\lambda_N = Q_f x_N$

Now, suppose it holds for $t+1$, that is, $\lambda_{t+1} = P_{t+1}x_{t+1}$. Let's show it holds for t. Using the state equation $x_{t+1} = Ax_t + Bu_t$ and $u_t = -R^{-1}B^\top \lambda_t$ from our optimality conditions, we get that

$$\lambda_{t+1} = P_{t+1}(Ax_t + Bu_t) = P_{t+1}(Ax_t - BR^{-1}B^\top \lambda_{t+1})$$

so that

$$\lambda_{t+1} = (I + P_{t+1}BR^{-1}B^\top)^{-1}P_{t+1}Ax_t$$

We can then use the co-state equation $\lambda_t = A^\top \lambda_{t+1} + Qx_t$ to get that

$$\lambda_t = A^\top (I + P_{t+1}BR^{-1}B^\top)^{-1}P_{t+1}Ax_t + Qx_t = P_t x_t$$

since by the Riccati recursion we know that

$$P_t = Q + A^\top (I + P_{t+1}BR^{-1}B^\top)^{-1}P_{t+1}A.$$

This completes the induction argument so that $\lambda_t = P_t x_t$.

Check for consistency across the two methods.

$$\begin{aligned}
u_t &= -R^{-1}B^\top \lambda_{t+1}\\
&= -R^{-1}B^\top (I + P_{t+1}BR^{-1}B^\top)^{-1}P_{t+1}Ax_t\\
&= -R^{-1}(I + B^\top P_{t+1}BR^{-1})^{-1}B^\top P_{t+1}Ax_t\\
&= -\left((I + B^\top P_{t+1}BR^{-1})R\right)^{-1}B^\top P_{t+1}Ax_t\\
&= -(R + B^\top P_{t+1}B)^{-1}B^\top P_{t+1}Ax_t
\end{aligned}$$

4 M4-RL4: CT LQR

Goal: Understand CT LQR.

References: Stengel §3.4 (DP solution, pg. 274); Lewis §3 (pg. 135, adjoint approach)

Consider the CT system

$$\dot{x} = Ax + Bu,\ x(0) = x_0$$

The problem is to choose $u : [0,T] \to \mathbb{R}^m$ so as to minimize

$$J = \int_0^T (x(\tau)^\top Qx(\tau) + u(\tau)^\top Ru(\tau))\, d\tau + x(T)^\top Q_f x(T)$$

We have a similar set up:

- T is horizon
- $Q = Q^\top \succeq 0$, $Q_f = Q_f^\top \succeq 0$, $R = R^\top \succ 0$ are the state cost, final state cost and input cost matrices

This is now an *infinite dimensional* problem since $u : [0,T] \to \mathbb{R}^m$ is the variable.

4.1 DP solution

As in the DT case, we define a **value function** $V_t : \mathbb{R}^n \to \mathbb{R}$ by

$$V_t(z) = \min_u \int_t^T \left(x(\tau)^\top Q x(\tau) + u(\tau)^\top R u(\tau)\right)\, d\tau + x(T)^\top Q_f x(T)$$

subject to $x(t) = z$, $\dot{x} = Ax + Bu$

Some notes:

- The minimization is taken over all possible signals $u : [t, T] \to \mathbb{R}^m$.
- $V_t(z)$ gives the minimum LQR cost-to-go, starting from state z at time t
- $V_T(z) = z^\top Q_f z^T$

Note. V_t is quadratic, i.e., $V_t(z) = z^\top P_t z$, where $P_t = P_t^T \geq 0$

Moreover, like the DT case: P_t can be found from a differential equation running backward in time from $t = T$. And, the LQR optimalu is easily expressed in terms of P_t.

4.2 DP Approach

Start with $x(t) = z$. Let us take $u(t) = w \in \mathbb{R}^m$, a constant, over the time interval $[t, t+h]$, where $h > 0$ is small.

Just as with the usual DP approach, we want to approximate the value at time t forward as the *instantaneous cost* incurred where we are plus the cost to go over the remainder of the horizon.

We will do this in an approximate manner.

First, the cost incurred over $[t, t+h]$ is

$$\int_t^{t+h} \left(x(\tau)^\top Q x(\tau) + u(\tau)^\top R u(\tau)\right)\, d\tau \simeq h(z^\top Q + w^\top R w)$$

and we end up at $x(t+h) \simeq z + h(Az + Bw)$.

Second, the min-cost-to-go from where we end-up is approximately given by

$$\begin{aligned} V_{t+h}(z + h(Az + Bw)) &= (z + h(Az + Bw))^\top P_{t+h}(z + h(Az + Bw)) \\ &\simeq (z + h(Az + Bw))^\top (P_t + h\dot{P}_t)(z + h(Az + Bw)) \\ &\simeq z^\top P_t z + h\left((Az + Bw)^\top P_t z + z^\top \dot{P}_t(Az + Bw) + z^\top \dot{P}_t z\right) \end{aligned}$$

(of course, dropping h^2 and higher order terms).

Adding them up, the cost incurred plus min-cost-to-go is approximately given by

$$z^\top P_t z + h\left(z^\top Q z + w^\top R w + (Az + Bw)^\top P_t z + z^\top P_t(Az + Bw) + z^\top \dot{P}_t z\right)$$

Now, minimize over w to get (approximately) optimal w:

$$2hw^\top R + 2Hz^\top P_t B = 0$$

so that

$$w^\star = -R^{-1}B^\top P_t z$$

(boy does that look similar!)

Thus optimal u is time-varying linear state feedback is given by

$$u^\star(t) = K_t x(t),\ K_t = -R^{-1}B^\top P_t$$

Ok, so this gives us the optimal control in terms of P_t. But, what is P_t?

4.3 (Approximate) Hamilton-Jacobi Equation

Let's substitute $w^\star$ into the (approx.) HJ equation

$$z^\top P_t z \simeq z^\top P_t z + h\left(z^T Q z + (w^\star)^\top R w^\star + (Az + Bw^\star)^\top P_t z + z^\top P_t (Az + Bw^\star) + z^\top \dot{P}_t z\right)$$

After you plug things in and simplify we get

$$-\dot{P}_t = A^\top P_t + P_t A - P_t B R^{-1} B^\top P_t Q$$

which is the Riccati differential equation for the LQR problem.

We can solve it (numerically) using the final condition $P_T = Q_f$.

Summary:

step 1 solve Riccati differential equation

$$-\dot{P}_t = A^\top P_t + P_t A - P_t B R^{-1} B^\top P_t Q,\ P_T = Q_f$$

backward in time

step 2 optimal u is $u^\star(t) = K_t x(t)$, $K_t = -R^{-1}B^\top P_t$

4.4 Some Additional comments

- DP method readily extends to time-varying A, B, Q, R, and tracking problem
- usually P_t rapidly converges as t decreases below T (same comment as DT)
- limit P_{ss} satisfies (cts-time) algebraic Riccati equation (ARE)

$$A^T P + PA - PBR^{-1}B^\top P + Q = 0$$

  which is a quadratic matrix equation
- P_{ss} can be found by (numerically) integrating the Riccati differential equation, or by direct methods (you will do this in your project)
- for t not close to horizon T, LQR optimal input is approximately a linear, constant state feedback

$$u(t) = K_{ss} x(t),\ K_{ss} = -R^{-1}B^\top P_{ss}$$

There is a similar co-state formulation for continuous time. Please see Lewis chapter 3 for this.

5 M4-RL5: CT LQR via Lagrange

In this Module 4 section we revisit CT LQR from a classical optimal control perspective using Lagrangians and nonlinear programming (NLP).

Consider the constrained NLP:

$$\begin{aligned} \min_u \ & \frac{1}{2}\int_0^T x(\tau)^\top Q x(\tau)\ d\tau \quad =: J + \frac{1}{2}x(T)^\top Q_f x(T) \\ \text{s.t.} \ & \dot{x}(t) = Ax(t) + Bu(t),\ t \in [0,T] \end{aligned}$$

Introduce Lagrange multiplier (function) $\lambda : [0,T] \to \mathbb{R}^n$ and write

$$L = J + \int_0^T \lambda(\tau)^\top (Ax(\tau) + Bu(\tau) - \dot{x}(\tau))\ d\tau$$

We can compute the derivatives (recall: you need distributions or test function to actually compute the derivatives here. . .)

- with respect to (wrt) u:

$$D_{u(t)}L = Ru(t) + B\lambda^\top(t) = 0 \quad \Longrightarrow \quad u = -R^{-1}B^\top\lambda(t)$$

- wrt x: use the fact that (just basic fundamental theorem of calculus)

$$\int_0^T \lambda(\tau)^\top \dot{x}(\tau) + \dot{\lambda}(\tau)^\top x(\tau)\ d\tau = \lambda(T)^\top x(T) - \lambda(0)^\top x(0)$$

$$D_{x(t)}L = Qx(t) + A^\top\lambda(t) + \dot{\lambda}(t) = 0 \quad \Longrightarrow \quad \dot{\lambda}(t) = -A^\top\lambda(t) - Qx(t)$$

- wrt $x(T)$:

$$D_{x(T)}L = Q_f x(T) - \lambda(T) = 0 \quad \Longrightarrow \quad \lambda(T) = Q_f x(T)$$

The above gives us optimality conditions just as in the DT case:

$$\dot{x} = Ax + Bu,\ x(0) = x_0, \quad \dot{\lambda} = -A^\top\lambda - Qx,\ \lambda(T) = Q_f x(T),\ u(t) = -R^{-1}B^\top\lambda(t)$$

5.1 Adjoint or Co-state system

Using $u(t) = -R^{-1}B^\top\lambda(t)$, we can write a system of equations (two-point boundary value problem):

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix} = \underbrace{\begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix}}_{\text{Hamiltonian}} \begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix} \tag{1}$$

You can argue that $\lambda(t) = P_t x(t)$. That is, use the Riccati DE with final time condition $P_T = Q_f$ and the co-state equation to show that $\lambda(t) = P_t x(t)$ as above is a solution to the costate equation.

5.2 Solving the Riccati DE via Hamiltonian

Consider

$$-\dot{P} = A^\top P + PA - PBR^{-1}B^\top P + Q$$

and recall (1). These two differential equations are related.

The fact that $\lambda(t) = P_t x(t)$ suggests that P should have the form $\lambda(t)x(t)^{-1}$ but of course this only makes sense if they are scalars.

So what do we do? How do we generalize this intuition?

Consider the matrix version of the Hamiltonian DE:

$$\frac{d}{dt}\begin{bmatrix} X(t) \\ Y(t) \end{bmatrix} = \underbrace{\begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix}}_{\text{Hamiltonian}} \begin{bmatrix} X(t) \\ Y(t) \end{bmatrix} \tag{2}$$

where $X, Y \in \mathbb{R}^{n\times n}$.

Then $Z(t) = Y(t)X(t)^{-1}$ satisfies the Riccati DE:

$$-\dot{Z} = A^\top Z + ZA - ZBR^{-1}B^\top Z + Q$$

Check your self using the fact that

$$\begin{aligned} \frac{d}{dt}(F(t)G(t)) &= \dot{F}(t)G(t) + F(t)\dot{G}(t) \\ \frac{d}{dt}(F(t)^{-1}) &= -F(t)^{-1}\dot{F}(t)F(t)^{-1} \qquad \text{(we saw this earlier in the quarter in [HW1]!!)} \end{aligned}$$

And, the fact that

$$\begin{aligned} \dot{Z} = \frac{d}{dt}YX^{-1} &= \dot{Y}X^{-1} - YX^{-1}\dot{X}X^{-1} \\ &= (-QX - AY)X^{-1} - YX^{-1}(AX - BR^{-1}B^\top Y)X^{-1} \\ &= -Q - A^\top Z - ZA + ZBR^{-1}B^\top Z \end{aligned}$$

Hence, we can solve the Riccati DE by solving a linear matrix (Hamiltonian) DE with final conditions $X(T) = I, Y(T) = Q_f$ and forming $P(t) = Y(t)X(t)^{-1}$ after.

5.3 Solving ARE via Hamiltonian

Recall the ARE (which gives you the steady state solution for the Riccati DE and also gives you the infinite horizon LQR solution):

$$Q + A^\top P + PA - PBR^{-1}B^\top P = 0$$

with $P \succeq 0$.

Aside:
Note $u^\star = Kx = -BR^{-1}B^\top Px$ and the closed loop dynamics look like

$$\dot{x} + Ax + Bu = (A + BK)x$$

It turns out that the closed loop system is stable when (Q, A) is observable and (A, B) is controllable (that is, finding an LQR solution will stabilize the system if these two conditions are met).

Let the eigenvalues of the closed loop system be

$$\lambda_1, \ldots, \lambda_n$$

And with the assumptions above, $\mathrm{Re}(\lambda_i) < 0$.

Now we also have that

$$\begin{bmatrix} I & 0 \\ -P & I \end{bmatrix}\begin{bmatrix} A & -BR^{-1}B^\top \\ -Q & -A^\top \end{bmatrix}\begin{bmatrix} I & 0 \\ P & 0 \end{bmatrix} = \begin{bmatrix} A + BK & -BR^{-1}B^\top \\ 0 & -(A+BK)^\top \end{bmatrix} \tag{3}$$

where the zero comes from the ARE. Note

$$\begin{bmatrix} I & 0 \\ P & 0 \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \\ -P & I \end{bmatrix}$$

Now, looking at the diagonal of (3), the eigenvalues of the Hamiltonian are $\lambda_1, \ldots, \lambda_n$ and $-\lambda_1, \ldots, -\lambda_n$. Hence, closed-loop eigenvalues are the eigenvalues of H with negative real part (under assumptions above).

You can show that if $A + BK$ is diagonalizable, i.e.

$$M^{-1}(A + BK)M = \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n),$$

then

$$M^\top(-A - BK)^\top M^{-\top} = -\Lambda$$

so that

$$\begin{bmatrix} M^{-1} & 0 \\ 0 & M^\top \end{bmatrix} \begin{bmatrix} A + BK & -BR^{-1}B^\top \\ 0 & -(A + BK)^\top \end{bmatrix} \begin{bmatrix} M & 0 \\ 0 & M^{-\top} \end{bmatrix} = \begin{bmatrix} \Lambda & -M^{-1}BR^{-1}B^\top M^{-\top} \\ 0 & -\Lambda \end{bmatrix}$$

Combine this with (3) to get

$$H \begin{bmatrix} M \\ PM \end{bmatrix} = \begin{bmatrix} M \\ PM \end{bmatrix} \Lambda$$

so that the n columns of

$$\begin{bmatrix} M \\ PM \end{bmatrix}$$

are the eigenvectors of H associated with the stable eigenvalues $\lambda_1, \ldots, \lambda_n$.

Thus, to solve the ARE, take the n stable eigenvalues (there will be n stable and n unstable ones) and finding the eigenvectors associated with the n stables ones. Stack them up as columns in a matrix and partition that matrix as

$$\begin{bmatrix} v_1 & \cdots v_n \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix} \in \mathbb{R}^{2n \times n}$$

Then $P = YX^{-1}$ is the unique PSD solution of the ARE.