Discrete-time optimal control—indirect approach

General nonlinear problem and intro to LQ optimal control

Zdeněk Hurák

DISCRETE-TIME optimal control will not only serve us in our course as a natural transition from the mathematical domain of finite-dimensional (nonlinear) optimization to the control-theoretic domain of optimal control but also will offer as a very useful and practical control design tool. After all, most control systems implemented today run on digital computers, that is, in discrete time. As a culmination of our presentation, the celebrated LQ optimal control will be introduced in quite some detail. First, we will present this classical result on a finite control horizon and then will extend the analysis to an infinite horizon.

1 Optimal control for a general nonlinear time-varying discrete-time dynamic system

We start by considering a general nonlinear and possibly time-varying discrete time dynamic system characterized by the state vector $\mathbf{x}_k$ whose evolution in discrete time k is described by the state-space model

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k), \tag{1}$$

for which the response is uniquely given by an initial value of the state $\mathbf{x}_i$ and a sequence of control inputs $\mathbf{u}_k$. For this system we want to find a control that extremizes a suitable optimization criterion. In some instances, minimizing a single quantity makes sense (such as minimizing the cost, rise time or steady-state error) while in some other, maximizing may seem more appropriate (such as maximizing the yield, bandwidth or robustness). In our course we will formulate the problem of optimal control design as a minimization problem. Another restriction that we will however relax shortly, will be that of control over a finite horizon $k \in [i, N]$. That is, we will look for a controller that minimizes a criterion of the following special structure

$$J_i(\mathbf{x}_k, \mathbf{u}_k) = \phi(\mathbf{x}_N, N) + \sum_{k=i}^{N-1} L_k(\mathbf{x}_k, \mathbf{u}_k). \tag{2}$$

The optimization criterion of this type turns out very flexible (and yet mathematically tractable as we will see shortly). It can be used to formulate the following control problems (we consider scalar control u_k and a scalar state x_k) here for notational simplicity)

- Minimum time problem — for $\phi = 1$ and $L_k = 1$, which gives $J = N - i$. Unfortunately, in this course we do not study tools for optimization in presence of integer variables. However, in principle some kind of binary search over the length of control interval is a straightforward solution. Furthermore, as we will see in one of the next lectures, once we switch from discrete-time to continuous-time systems, this control design formulation will be tractable for us even theoretically.
- Minimum fuel problem — for $\phi = 0$ and $L_k = |u_k|$, which gives $J = \sum_{k=i}^{N-1} |u_k|$.
- Minimum energy problem — for $\phi = \frac{1}{2} s_N x_N^2$ and $L_k = \frac{1}{2}(x_k^2 + u_k^2)$, which gives $J = \frac{1}{2} s_N x_N^2 + \frac{1}{2} \sum_{k=i}^{N-1}(x_k^2 + u_k^2)$. This type of an optimization cost is particularly popular. Both for the mathematical reasons (we all now appreciate the nice properties of quadratic functions) and for practical engineering reasons as it allows us to capture a trade-off between the control performance (penalty on x_k) and control effort (penalty on u_k)[1]. Furthermore, the state at the terminal time N is penalized separately just in order to allow another trade-off between the transient and terminal behavior. Our plan is to focus on this type of optimization criterion.

The two numbered expressions above specify an optimization fully. We are going to find a finite set of numbers (or n-tuples of numbers *aka* vectors) $\mathbf{x}_{i+1}, \mathbf{x}_{i+2}, \ldots, \mathbf{x}_N$ and $\mathbf{u}_i, \mathbf{u}_{i+1}, \ldots, \mathbf{u}_{N-1}$ (note that we do not include $\mathbf{u}_N$ as it has no influence over the interval $[i, N]$) that minimize the criterion (2) while satisfying the constraints (1). This fits nicely into the nonlinear optimization framework introduced in previous lectures. Before we start invoking a numerical solver for constrained optimization, let us investigate the problem a bit.

1.1 First-order necessary condition

We have identified a finite-dimensional constrained nonlinear optimization problem. We already know how to handle it. By introducing a set of auxiliary parameters $\boldsymbol{\lambda}_k$ called Lagrange multiplier we turn the constrained problem into an unconstrained one. The new (augmented) cost function is

$$J_i'(\mathbf{x}_k, \mathbf{u}_k, \boldsymbol{\lambda}_k) = \phi(\mathbf{x}_N, N) + \sum_{k=i}^{N-1} \left[L_k(\mathbf{x}_k, \mathbf{u}_k) + \boldsymbol{\lambda}_k^{\mathrm{T}} \left[\mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k) - \mathbf{x}_{k+1} \right] \right]. \quad (3)$$

From now on, you, the student, do not need any guide here. You are given an unconstrained optimization problem and its solution is just a few steps away. In particular, stationary points must be found. This calls for differentiating the above expression with respect to all the variables and setting these derivatives equal to zeros. The principles are clear. Nonetheless, some experience might be shared here if compact formulas are to be found. First such advice is to rename the variable(s) $\boldsymbol{\lambda}_k$ to $\boldsymbol{\lambda}_{k+1}$

[1]Without any rigorous theoretical background this can be checked for instance in the graphical design tool provided by Control System Toolbox for Matlab. Typing `sisotool` in Matlab prompt, *Control and Estimation Tools Manager* GUI opens. Click on the *Automated tuning* tab and select *LQG synthesis* in the *Design method.* Experiment with the slider specifying the *controller response* — more *aggressive* or more *robust*

$$J_i'(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\lambda}_{k+1}) = \phi(N,\mathbf{x}_N) + \sum_{k=i}^{N-1}\left[L_k(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{\boldsymbol{k+1}}^{\mathrm{T}}\ [\mathbf{f}_k(\mathbf{x}_k,\mathbf{u}_k) - \mathbf{x}_{k+1}]\right]. \tag{4}$$

This is really just a notational decision but thanks to it our resulting formulas will enjoy some symmetry. Maybe it would be more didactic to leave you to go on without this advice and only then to let you to figure out this remedy on your own.

Another notational advice is to make the above expression a bit shorter by introducing a new variable defined as

$$H_k(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\lambda}_{k+1}) = L_k(\mathbf{x}_k,\mathbf{u}_k) + \boldsymbol{\lambda}_{k+1}^{\mathrm{T}}\ \mathbf{f}_k(\mathbf{x}_k,\mathbf{u}_k), \tag{5}$$

which gives

$$J_i'(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\lambda}_{k+1}) = \phi(N,\mathbf{x}_N) + \sum_{k=i}^{N-1}\left[H_k(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\lambda}_{k+1}) - \boldsymbol{\lambda}_{k+1}^{\mathrm{T}}\ \mathbf{x}_{k+1}\right], \tag{6}$$

although this time the motivation for an introduction of a new symbol is not purely notational. We will call this new object Hamiltonian for the reasons that are to be discussed later in the course (when discussing continuous-time systems). Maybe you have already encountered the concept of Hamiltonian in physics or theoretical mechanics and now the structure of our expression must be familiar to you.

The final polishing of the expression (4) before starting calculation of the derivatives consists in bringing the terms that contain the "logically" related variables: the state $\mathbf{x}_N$ at the final time, the state $\mathbf{x}_i$ at the initial time, and the states, controls and Lagrange multipliers in the transient period

$$\begin{aligned}J_i'(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\lambda}_{k+1}) &= \underbrace{\phi(N,\mathbf{x}_N) - \boldsymbol{\lambda}_N^{\mathrm{T}}\ \mathbf{x}_N}_{\text{at terminal time}} + \underbrace{H_i(\mathbf{x}_i,\mathbf{u}_i,\boldsymbol{\lambda}_{i+1})}_{\text{at initial time}}\\ &\quad + \sum_{k=i+1}^{N-1}\left[H_k(\mathbf{x}_k,\mathbf{u}_k,\boldsymbol{\lambda}_{k+1}) - \boldsymbol{\lambda}_k^{\mathrm{T}}\ \mathbf{x}_k\right].\end{aligned} \tag{7}$$

This step was absolutely not important, it will only make it a bit more convenient once we start looking for the derivatives. And the time has come. Recall now our recommended procedure for finding derivatives of functions that feature matrices, vectors, their products, transposes and other operations — find the differential instead and identify the derivative in the result. The gradient is then (by convention) obtained as a transpose of the derivative. An example for refreshing

Example 1.1 (Finding a derivative of a matrix-vector expression)**.** *Find a gradient of a function* $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{x}$. *The differential* $\mathrm{d}f$ *is obtained using the identical rules that we apply when obtaining derivatives.*

$$\mathrm{d}f = \frac{1}{2}\mathrm{d}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathrm{d}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathrm{d}\mathbf{x} = \underbrace{(\mathbf{x}^{\mathrm{T}}\mathbf{Q} - \mathbf{b}^{\mathrm{T}})}_{\frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}}\mathrm{d}\mathbf{x}.$$

The derivative identified *in this way is the content of the brackets above. Now, although a part of the mathematical and control-theoretic community follows a different convention, we will the gradient* $\nabla f(\mathrm{x})$ *as a transpose of the derivative, that is,*

$$\nabla f(\mathbf{x}) = \mathbf{Q}\mathbf{x} - \mathbf{b}.$$

Following the "derivative-identification" procedure outlined in the example above, we anticipate the differential of the augmented cost function in the following form

$$\begin{aligned}\mathrm{d}J_i' &= (\qquad)^{\mathrm{T}}\,\mathrm{d}\mathbf{x}_N + (\qquad)^{\mathrm{T}}\,\mathrm{d}\mathbf{x}_i\\ &+ \sum_{k=i+1}^{N-1}(\qquad)^{\mathrm{T}}\,\mathrm{d}\mathbf{x}_k + \sum_{k=i}^{N-1}(\qquad)^{\mathrm{T}}\,\mathrm{d}\mathbf{u}_k + \sum_{k=i+1}^{N}(\qquad)^{\mathrm{T}}\,\mathrm{d}\boldsymbol{\lambda}_k.\end{aligned} \tag{8}$$

Filling in the empty brackets is the key task now. Its solution is documented below

$$\begin{aligned}\mathrm{d}J_i' &= \left(\nabla_{\mathbf{x}_N}\phi - \lambda_N\right)^{\mathrm{T}}\ \mathrm{d}\mathbf{x}_N + \left(\nabla_{\mathbf{x}_i}H_i\right)^{\mathrm{T}}\ \mathrm{d}\mathbf{x}_i\\ &+ \sum_{k=i+1}^{N-1}\left(\nabla_{\mathbf{x}_k}H_k - \boldsymbol{\lambda}_k\right)^{\mathrm{T}}\ \mathrm{d}\mathbf{x}_k + \sum_{k=i}^{N-1}\left(\nabla_{\mathbf{u}_k}H_k\right)^{\mathrm{T}}\ \mathrm{d}\mathbf{u}_k + \sum_{k=i+1}^{N}\left(\nabla_{\boldsymbol{\lambda}_k}H_{k-1} - \mathbf{x}_k\right)^{\mathrm{T}}\ \mathrm{d}\boldsymbol{\lambda}_k.\end{aligned} \tag{9}$$

The ultimate goal of this derivation was to obtain stationary points for the augmented cost function, that is, to find conditions under which $\mathrm{d}J_i' = 0$. In typical optimization problems, the optimization is conducted with respect to all the participating variables, which means that the corresponding differentials may be arbitrary and the only way to guarantee that the total differential of J_i' is zeros is to make the derivatives (the contents of the brackets) equal to zero. There are two exceptions to this rule in our case, though.

First, the state at the initial time is typically given, therefore it is not available for optimization. Then $\mathrm{d}\mathbf{x}_i = 0$ and the corresponding necessary condition is replaced by the statement of the initial value of the state $\mathbf{x}_i$ given.

Second, the state at the final time may also be given, in which case the corresponding condition is replaced by $\mathbf{x}_N$ given, but it makes a good sense to consider the final state as free for optimization, in which case corresponding necessary condition of optimality is given by the requirement that the content of the brackets must be equal to zero.

The ultimate set of first-order necessary conditions is given by the equations below, which contain in a compact way the discussion of the initial and final states initiated above

$$\mathbf{x}_{k+1} = \nabla_{\boldsymbol{\lambda}_{k+1}}H_k = \mathbf{f}_k(\mathbf{x}_k,\mathbf{u}_k), \quad k = i,\ldots,N-1, \tag{10}$$

$$\boldsymbol{\lambda}_k = \nabla_{\mathbf{x}_k}H_k = \nabla_{\mathbf{x}_k}\mathbf{f}_k\ \ \boldsymbol{\lambda}_{\mathbf{k+1}} + \nabla_{\mathbf{x}_k}L_k, \quad k = i+1,\ldots,N-1 \tag{11}$$

$$0 = \nabla_{\mathbf{u}_k}H_k = \nabla_{\mathbf{u}_k}\mathbf{f}_k\ \ \boldsymbol{\lambda}_{k+1} + \nabla_{u_k}L_k, \quad k = i,\ldots,N-1 \tag{12}$$

$$0 = \left(\nabla_{\mathbf{x}_N}\phi - \lambda_N\right)^{\mathrm{T}}\mathrm{d}\mathbf{x}_N, \tag{13}$$

$$0 = \left(\nabla_{\mathbf{x}_i}H_i\right)^{\mathrm{T}}\mathrm{d}\mathbf{x}_i. \tag{14}$$

Here in $\nabla\mathbf{f}$ we consider a vector function $\mathbf{f}$, therefore the resulting object is not just a gradient. Instead, it is a matrix whose columns are gradients of the individual components of the vector $\mathbf{f}$. It is a transpose of Jacobian. Note that here the notational conventions might differ in the literature.

The first three necessary conditions above can be made completely "symmetric" by running the second one from $k = i$ because the $\boldsymbol{\lambda}_i$ introduced this way does not influence the rest of the problem.

We have just derived a very important result. The first equation can be recognized as the original *state equation*. The second equation is called a *co-state equation* and the variable $\boldsymbol{\lambda}_k$ is called a *co-state variable.* The third equation is called an *stationarity equation.*

Now, let us see what can be the use of this result in a familiar situation of a linear system and a quadratic cost function.

2 LQ-optimal regulation over a finite horizon

We consider an LTI system described by the state-space model

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \qquad \mathbf{x}_0 \text{ given}, \tag{15}$$

and our goal is to find a control sequence $[u_0, u_1, \ldots, u_{N-1}]$ that minimizes

$$J_0 = \frac{1}{2}\mathbf{x}_N^{\mathrm{T}}\mathbf{S}_N\mathbf{x}_N + \frac{1}{2}\sum_{k=0}^{N-1}\left[\mathbf{x}_k^{\mathrm{T}}\mathbf{Q}\mathbf{x}_k + \mathbf{u}_k^{\mathrm{T}}\mathbf{R}\mathbf{u}_k\right], \tag{16}$$

where the quadratic cost function is parameterized the matrices which must be symmetric and at least positive semidefinite, otherwise the corresponding quadratic terms will not play a good role of a performance indicator, that is, a weighted distance from 0. For the reasons that will become clear soon, the matrix R must comply with an even stricter condition — it must be positive definite. Hence

$$\mathbf{S}_N \geq 0, \mathbf{Q} \geq 0, \mathbf{R} > 0. \tag{17}$$

Note also that in this task we decided to consider the initial time $i = 0$. This brings no harm to generality since we assume time-invariant system, that is, we can freely label the initial time as zero.

The Hamiltonian for our problem is

$$H_k = \frac{1}{2}\left(\mathbf{x}_k^{\mathrm{T}}\mathbf{Q}\mathbf{x}_k + \mathbf{u}_k^{\mathrm{T}}\mathbf{R}\mathbf{u}_k\right) + \boldsymbol{\lambda}_{k+1}^{\mathrm{T}}\left(\mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k\right). \tag{18}$$

Substituting into the general necessary conditions (10) we obtain

$$\mathbf{x}_{k+1} = \nabla_{\boldsymbol{\lambda}_{k+1}} H_k = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \tag{19}$$

$$\boldsymbol{\lambda}_k = \nabla_{\mathbf{x}_k} H_k = \mathbf{Q}\mathbf{x}_k + \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}, \tag{20}$$

$$\mathbf{0} = \nabla_{\mathbf{u}_k} H_k = \mathbf{R}\mathbf{u}_k + \mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}, \tag{21}$$

$$0 = \left(\mathbf{S}_N\mathbf{x}_N - \boldsymbol{\lambda}_N\right)^{\mathrm{T}} \mathrm{d}\mathbf{x}_N, \tag{22}$$

$$\mathbf{x}_0 = \mathbf{r}_0. \tag{23}$$

The last two equations represent boundary conditions. Note that here we have already fixed the initial state. If this is not appropriate in a particular scenario, go back to (14) and adjust the boundary equation accordingly.

The third equation above — the stationarity equation — can be used to extract the optimal control

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}. \tag{24}$$

The need for nonsingularity of $\mathbf{R}$ is now obvious. Upon substituting the recipe for the optimal $\mathbf{u}_k$ into the state and the co-state equations, two discrete-time equations result

$$\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\lambda}_k\end{bmatrix} = \begin{bmatrix}\mathbf{A} & -\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\\ \mathbf{Q} & \mathbf{A}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{x}_k\\ \boldsymbol{\lambda}_{k+1}\end{bmatrix} \tag{25}$$

This is a two-point boundary value problem. The problem is of order $2n$; in order to solve it we need $2n$ boundary values: n boundary values are provided by $\mathbf{x}_i$, which we are typically given, and n boundary values are given by the equation (22), from which $\boldsymbol{\lambda}_N$ must be extracted. Most of our subsequent discussion will revolve around this task.

An idea might come into one's mind that provided $\mathbf{A}$ is nonsingular, one can left-multiply the above equation by the inverse of $\mathbf{A}$ to obtain

$$\begin{bmatrix}\mathbf{x}_k\\ \boldsymbol{\lambda}_k\end{bmatrix} = \begin{bmatrix}\mathbf{A}^{-1} & \mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\\ \mathbf{Q}\mathbf{A}^{-1} & \mathbf{A}^{\mathrm{T}}+\mathbf{Q}\mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\lambda}_{k+1}\end{bmatrix} \tag{26}$$

This helped at least to have both variable evolving in the same "direction" in time but we do not know $\boldsymbol{\lambda}_N$ anyway... Nonetheless, do not forget this "trick" as we are going to invoke it later.

2.1 Zero-input cost

Before we delve into solution of the original problem, let us investigate a somewhat artificial problem when no control input is applied. We only calculate the cost of not controlling the system at all. This will give us some valuable insight.

We start by evaluating the cost at the terminal time N and then proceed backwards in time, that is, decrease time to $N-1$ and so on

$$J_N = \frac{1}{2}\mathbf{x}_N^{\mathrm{T}}\mathbf{S}_N\mathbf{x}_N \tag{27}$$

$$J_{N-1} = \frac{1}{2}\mathbf{x}_N^{\mathrm{T}}\mathbf{S}_N\mathbf{x}_N + \frac{1}{2}\mathbf{x}_{N-1}^{\mathrm{T}}\mathbf{Q}_{N-1}\mathbf{x}_{N-1} \tag{28}$$

$$= \frac{1}{2}\mathbf{x}_{N-1}^{\mathrm{T}}\left(\mathbf{A}^{\mathrm{T}}\mathbf{S}_N A + \mathbf{Q}\right)\mathbf{x}_{N-1} \tag{29}$$

$$J_{N-2} = \dots \tag{30}$$

Upon introducing a new name $\mathbf{S}_{N-1}$ for term $\mathbf{A}^{\mathrm{T}}\mathbf{S}_N\mathbf{A}+\mathbf{Q}$ and similarly for all lower time instances, we arrive at a difference Lyapunov equation

$$\mathbf{S}_k = \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}+\mathbf{Q}. \tag{31}$$

This is a very famous and well-investigated equation in systems and control theory. Its solution is given by

$$\mathbf{S}_k = (\mathbf{A}^{\mathrm{T}})^{N-k}\mathbf{S}_N\mathbf{A}^{N-k} + \sum_{i=k}^{N-1}\left(\mathbf{A}^{\mathrm{T}}\right)^{N-i-1}\mathbf{Q}\mathbf{A}^{N-i-1}. \tag{32}$$

Having the sequence of $\mathbf{S}_k$ at hand, the cost function can be readily evaluated as

$$J_k = \frac{1}{2}\mathbf{x}_k^{\mathrm{T}}\mathbf{S}_k\mathbf{x}_k. \tag{33}$$

We will come back to this observation in a few moments. Before we do that, note that if the plant is stable, the cost over $[-\infty, N]$ or, equivalently (thanks to the fact that the system is time invariant) $[0,\infty]$ is finite and given by

$$J_\infty = \frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{S}_\infty\mathbf{x}_0. \tag{34}$$

where the symbol $\mathbf{S}_\infty$ represents the limit

$$\mathbf{S}_\infty = \lim_{(N-k)\to\infty} \mathbf{S}_k. \tag{35}$$

When it comes to the computation of $\mathbf{S}_\infty$, besides the implementation of the limiting iterative process, we may exploit the fact that in the steady state

$$\mathbf{S}_k = \mathbf{S}_{k+1}, \tag{36}$$

which turns the difference Lyapunov equation into the even more famous Algebraic Lyapunov Equation (ALE)

$$\mathbf{S} = \mathbf{A}^{\mathrm{T}}\mathbf{S}\mathbf{A} + \mathbf{Q}. \tag{37}$$

Notoriously known facts about this equation (that is, studied by you in a previous course on linear systems) are

- if $\mathbf{A}$ stable and $\mathbf{Q} \geq 0$ then there is a solution to ALE satisfying $\mathbf{S} \geq 0$.

- if $\mathbf{A}$ stable and $(\mathbf{A}, \sqrt{\mathbf{Q}})$ observable then there is a unique solution to ALE satisfying $\mathbf{S} > 0$.

Of course, if the system is unstable, the cost can be infinite, depending on $\mathbf{Q}$. As a trivial example, for $\mathbf{Q} = 0$, the cost will stay finite — exactly zero — disregarding the system blowing out.

Finally, concerning a numerical solution, Lyapunov equation is just a linear equation and as such can be reformulated into the standard $\mathbf{A}\mathbf{x} = \mathbf{b}$ form (using a trick based on Kronecker product, see **kron()**). Specialized algorithms exist and some of them are implemented in **dlyap()** function in Matlab.

2.2 Fixed final state and finite control horizon

Back to the nonzero control case. First we are going to investigate the scenario when the final requested state is given by the value $\mathbf{r}_N$. The optimal control problem turns into

$$\begin{aligned}
\operatorname*{minimize}_{\mathbf{x}_1,\ldots,\mathbf{x}_N,\mathbf{u}_0,\ldots,\mathbf{u}_{N-1}} \quad & \frac{1}{2}\sum_{k=0}^{N-1}\left[\mathbf{x}_k^T\mathbf{Q}\mathbf{x}_k + \mathbf{u}_k^T\mathbf{R}\mathbf{u}_k\right] \\
\text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \\
& \mathbf{x}_0 = \mathbf{r}_0, \\
& \mathbf{x}_N = \mathbf{r}_N, \\
& \mathbf{Q} \geq 0, \mathbf{R} > 0.
\end{aligned}$$

Note also that the term penalizing the final state is removed from the cost because it is always fixed and not subject to optimization.After eliminating the controls using

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}.$$

the two point boundary value problem specializes into

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}, \tag{38}$$

$$\boldsymbol{\lambda}_k = \mathbf{Q}\mathbf{x}_k + \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1} \tag{39}$$

$$\mathbf{x}_0 = \mathbf{r}_0, \tag{40}$$

$$\mathbf{x}_N = \mathbf{r}_N. \tag{41}$$

after the boundary condition (22) is replaced by

$$\mathbf{x}_N = \mathbf{r}_N. \tag{42}$$

Make sure that you understand how this comes to replace the original boundary condition. It is just that by fixing $\mathbf{x}_N$, the corresponding differential is zero

$$\mathrm{d}\mathbf{x}_N = 0. \tag{43}$$

In other words, $\mathbf{x}_N$ can no longer be used as an optimization parameter.

2.2.1 Two-point boundary value problem (assuming A nonsingular)

The problem at hand is called a *two-point boundary value problem* (BVP). The terminology is better established in the contrinuous time setting (for differential equations) but it makes perfect sense in the discrete-time setting too. It is called *two-point* because some of the variables are specified at one end of the time interval and some other variables are specified at the other end of the time interval. Well, in this particular case, it is only the $\mathbf{x}_k$ that is specified at both ends while $\boldsymbol{\lambda}_k$ is left unspecified.

But this is actually not the only problem with these equations. This set of recurrence equations is weird in the sense that from the perspective of $\mathbf{x}_k$ it evolves forward in time while from the perspective of $\boldsymbol{\lambda}_k$ it evolves backward in time.

$$\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\lambda}_k\end{bmatrix} = \begin{bmatrix}\mathbf{A} & -\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\\ \mathbf{Q} & \mathbf{A}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{x}_k\\ \boldsymbol{\lambda}_{k+1}\end{bmatrix} \tag{44}$$

There is not much we can do with these equations in this form. However, assuming that $\mathbf{A}$ is nonsingular, we can invoke the discrete-time Hamiltonian system (26), in which we reorganized the equations so that both state and co-state variables evolve backwards. For convenience we give it here again

$$\begin{bmatrix}\mathbf{x}_k\\ \boldsymbol{\lambda}_k\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{A}^{-1} & \mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\\ \mathbf{Q}\mathbf{A}^{-1} & \mathbf{A}^{\mathrm{T}}+\mathbf{Q}\mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\end{bmatrix}}_{\mathbf{H}}\begin{bmatrix}\mathbf{x}_{k+1}\\ \boldsymbol{\lambda}_{k+1}\end{bmatrix} \tag{45}$$

Obviously,

$$\begin{bmatrix}\mathbf{x}_0\\ \boldsymbol{\lambda}_0\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{A}^{-1} & \mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\\ \mathbf{Q}\mathbf{A}^{-1} & \mathbf{A}^{\mathrm{T}}+\mathbf{Q}\mathbf{A}^{-1}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\end{bmatrix}^N}_{\mathbf{M}:=\mathbf{H}^N}\begin{bmatrix}\mathbf{x}_N\\ \boldsymbol{\lambda}_N\end{bmatrix} \tag{46}$$

From the first equation we can get $\boldsymbol{\lambda}_N$. First, let's rewrite it here

$$\mathbf{M}_{12}\boldsymbol{\lambda}_N = \mathbf{x}_0 - \mathbf{M}_{11}\mathbf{x}_N,$$

from which (after substituting for the known initial and final states)

$$\boldsymbol{\lambda}_N = \mathbf{M}_{12}^{-1}(\mathbf{r}_0 - \mathbf{M}_{11}\mathbf{r}_N).$$

Having the final state and the co-state $[\mathbf{x}_N, \boldsymbol{\lambda}_N]^{\mathrm{T}}$, we could solve the Hamiltonian system backward to get the states and co-states on the whole time interval $[0, N-1]$.

2.2.2 Minimum energy control (assuming $\mathbf{Q} = \mathbf{0}$)

We can get some more insight into the problem if we further restrict the class of problems we can treat. Namely, we will assume

$$\mathbf{Q} = \mathbf{0}. \tag{47}$$

This is a significant restriction, nonetheless the resulting problem is still practically reasonable. On the other hand, here we do not need to assume that $\mathbf{A}$ is nonsingular. The cost function is then

$$J = \sum_{k=0}^{N} \mathbf{u}_k^{\mathrm{T}}\, \mathbf{u}_k, \tag{48}$$

which is why the problem is called *the minimum-energy problem.* Rewriting (25) with the new restriction $\mathbf{Q} = \mathbf{0}$ we get the state and co-state equation

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1} \tag{49}$$

$$\boldsymbol{\lambda}_k = \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}. \tag{50}$$

It is obvious why we wanted to enforce the $\mathbf{Q} = \mathbf{0}$ restriction — the co-state equation is now completely decoupled from the state equation and can be solved independently

$$\boldsymbol{\lambda}_k = (\mathbf{A}^{\mathrm{T}})^{N-k}\boldsymbol{\lambda}_N. \tag{51}$$

Now substitute this solution of the co-state equation into the state equation

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{N-k-1}\boldsymbol{\lambda}_N \tag{52}$$

Finding a solution to the state equation present no more difficult task then did the co-state equation — the second summand on the right is considered as a an “input”. The solution is then

$$\mathbf{x}_k = \mathbf{A}^k\mathbf{x}_0 - \sum_{i=0}^{k-1}\mathbf{A}^{k-1-i}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{N-i-1}\boldsymbol{\lambda}_N. \tag{53}$$

The last step reveals the motivation for all the previous steps — we are now expressing the state at the final time. Clearly, in this way we introduce some known quantity into the problem

$$\mathbf{x}_N = \mathbf{r}_N = \mathbf{A}^N\mathbf{x}_0 - \underbrace{\sum_{i=0}^{N-1}\mathbf{A}^{N-1-i}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{N-i-1}}_{G_{0,N,R}}\boldsymbol{\lambda}_N. \tag{54}$$

This enables us to calculate $\boldsymbol{\lambda}_N$ directly as a solution to a linear equation. To make the notation simpler, denote the sum in the expression above by $\mathbf{G}_{0,N,R}$ (we will discuss this particular object in a while

$$\boldsymbol{\lambda}_N = -\mathbf{G}_{0,N,R}^{-1}\,(\mathbf{r}_N - \mathbf{A}^N\mathbf{x}_0). \tag{55}$$

The rest is quite straightforward as the optimal control (24) depends on the co-state

$$\mathbf{u}_k = \mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{N-k-1}\mathbf{G}_{0,N,R}^{-1}\,(\mathbf{r}_N - \mathbf{A}^N\mathbf{x}_0). \tag{56}$$

This is the desired formula for computation of the optimal control. A few observations can be made

- The control is proportional to the difference $(\mathbf{r}_N - \mathbf{A}^N\mathbf{x}_0)$. The intuitive interpretation is that the further the requested final state is from the state into which the system would finally evolve without any control, the higher the control is needed.
- The control is proportional to the inverse of a matrix $\mathbf{G}_{0,N,R}$ which is called weighted reachability Gramian. The standard result from the theory of linear dynamic systems is that nonsingularity of a reachability Gramian is equivalent to reachability of the system. More on this below.

2.2.3 Weighted reachability Gramian

Recall that there is a matrix called discrete-time reachability Gramian

$$\mathbf{G} = \sum_{k=0}^{\infty} \mathbf{A}^k\mathbf{B}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^k \tag{57}$$

and the nonsingularity of this matrix serves as a test of reachability for stable discrete-time systems.

How does this classical object relate to the object $\mathbf{G}_{0,N,R}$ introduced in the previous paragraph? First consider the restriction of the summation from the infinite interval $[0,\infty]$ to $[0,N-1]$. In other words, we analyze the matrix

$$\mathbf{G}_{0,N} = \sum_{k=0}^{N-1} \mathbf{A}^{N-1-k}\mathbf{B}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{N-1-k} \tag{58}$$

Recall that Caley-Hamilton theorem tells us that every higher power of an $N\times N$ matrix can be expressed as a linear combination of powers of 0 through $N-1$. In other words, using higher order powers of A than $N-1$ cannot increase the rank of the matrix.

Finally, provided R is nonsingular (hence R^{-1} is nonsingular as well), the rank of the Gramian is not changed after introducing the weight

$$\mathbf{G}_{0,N,R} = \sum_{k=0}^{N-1} \mathbf{A}^{N-1-k}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{N-1-k}. \tag{59}$$

To conclude, the weighted Gramian defined on a finite discrete-time horizon is invertible if and only if the (stable) system is reachable. This conclusion is quite natural: if an optimal control is to be found, first it must be guaranteed that any control can be found which brings the system from an arbitrary initial state into an arbitrary final state on a finite time interval — the very definition of reachability.

Frankly speaking, the advantage of the just introduced optimal control design strategy seems to lie mainly in the new insight we gained. The outcome of the procedure is a pre-computed control sequence, which leads to an open-loop (pre-programmed) control strategy. This is hardly acceptable in practical applications because the presence of disturbances and modeling errors call for introduction of a feedback.

Furthermore, although the numerical computation is very simple, it appears that the approach does not offer too many opportunities for extensions. Mainly that we will not be able to include additional constraints such as bounds on the control signal

$$\mathbf{u}_{\min} \leq \mathbf{u}_k \leq \mathbf{u}_{\max}, \tag{60}$$

where the bounds are interpreted component-wise. Having been exposed to the basics of nonlinear optimization, it is a straightforward task to reformulate the optimal control problem as an instance of nonlinear programming. In particular, sticking to the quadratic cost function, a numerical solver for quadratic programming such as **quadprog()** in Optimization Toolbox for Matlab may be used to solve the problem with the bounds on the controls added.

2.3 Free final state and finite control horizon

The previous discussion revolved around the task of bringing the system to a given final state exactly. What if we relax this strict requirement and instead just request that the system be eventually brought to the close vicinity of the requested state? How close — this could be affected by the term corresponding to the terminal state in the general LQ optimization criterion.

The only change with respect to the previous development is just in the boundary condition. We have to go back to (22). Now the final state $\mathbf{x}_N$ can also be used as a parameter for our optimization. Hence $\mathrm{d}\mathbf{x}_N \neq 0$. We write down again the full necessary conditions including the new boundary conditions

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}, \tag{61}$$

$$\boldsymbol{\lambda}_k = \mathbf{Q}\mathbf{x}_k + \mathbf{A}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}, \tag{62}$$

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1}, \tag{63}$$

$$\mathbf{S}_N\mathbf{x}_N = \boldsymbol{\lambda}_N, \tag{64}$$

$$\mathbf{x}_0 = \text{given}. \tag{65}$$

We find ourselves in a pretty much similar trouble as before. The boundary condition (64) refers to the variables whose values we do not know. The solution is provided by the *insightful guess* called *sweep method*. The idea is to extend the validity of the linear relationship between the state and the co-state at the final time to all other discrete times

$$\mathbf{S}_k\mathbf{x}_k = \boldsymbol{\lambda}_k. \tag{66}$$

Let us try this an see if it gives a working solution. Substitute (66) into the state and co-state equations, (61) and (62), respectively. Let us start with the state equation

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1}. \tag{67}$$

Solving for $\mathbf{x}_{k+1}$ yields

$$\mathbf{x}_{k+1} = (\mathbf{I} + \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1})^{-1}\mathbf{A}\mathbf{x}_k. \tag{68}$$

Now perform the same *sweep* substitution into the co-state equation

$$\mathbf{S}_k\mathbf{x}_k = \mathbf{Q}\mathbf{x}_k + \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1} \tag{69}$$

and substitute from (68) into the last equation to get

$$\mathbf{S}_k\mathbf{x}_k = \mathbf{Q}\mathbf{x}_k + \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{I} + \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1})^{-1}\mathbf{A}\mathbf{x}_k. \tag{70}$$

Since this equation must hold for an arbitrary $\mathbf{x}_k$, we get an equation in the matrices $\mathbf{S}_k$

$$\mathbf{S}_k = \mathbf{Q} + \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{I} + \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1})^{-1}\mathbf{A}. \tag{71}$$

This is a famous equation called *Difference Riccati Equation.* When initialized with $\mathbf{S}_N$, it generates the sequence of matrices $\mathbf{S}_{N-1}, \mathbf{S}_{N-2}, \mathbf{S}_{N-3}, \ldots$ A noteworthy feature of this sequence is that the discrete-time now evolves backwards.

Once we have generated a sufficiently long sequence (down to $\mathbf{S}_1$), the optimal control is then computed using the stationary equation (63)

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\boldsymbol{\lambda}_{k+1} = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1}. \tag{72}$$

This suggests that the optimal control is generated using the state but the current scheme is noncausal. Turning this into a causal one is easy. Just substitute the state equation and we get

$$\mathbf{u}_k = -\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k). \tag{73}$$

Solving this equation for $\mathbf{u}_k$ gives

$$\mathbf{u}_k = \underbrace{(\mathbf{I} + \mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B})^{-1}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}}_{\mathbf{K}_k}\mathbf{x}_k. \tag{74}$$

Mission accomplished. This is our desired control. A striking observation is that although we made no specifications as for the controller structure, the state feedback popped out as the optimal control strategy! The feedback gain is time-varying and deserves a name after its inventor — Kalman gain. Incorporating the knowledge that R is nonsingular, a minor simplification of the lengthy expression can be made

$$\mathbf{K}_k = (\mathbf{R} + \mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}. \tag{75}$$

Before we move on, let us elaborate a bit more on the difference Riccati equation. Invoking a popular (but hard to reliably memorize) rule for inversion of a sum of two matrices called *matrix inversion formula*, which reads

$$(\mathbf{A}_{11}^{-1} + \mathbf{A}_{12}\mathbf{A}_{22}\mathbf{A}_{21})^{-1} = \mathbf{A}_{11} - \mathbf{A}_{11}\mathbf{A}_{12}(\mathbf{A}_{21}\mathbf{A}_{11}\mathbf{A}_{12} + \mathbf{A}_{22}^{-1})^{-1}\mathbf{A}_{21}\mathbf{A}_{11}, \tag{76}$$

the equation (71) can be rewritten (after multiplying the brackets out) into

$$\mathbf{S}_k = \mathbf{Q} + \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A} - \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}(\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B} + \mathbf{R})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}, \tag{77}$$

which we will regard as an alternative form of difference Riccati equation.

Observing that the steps of the computation of the Kalman gain $\mathbf{K}_k$ reappear in the computation of the solution of the Riccati equation (77), a more efficient arrangement of the computation in every iteration step is

$$\mathbf{K}_k = \left(\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B} + \mathbf{R}\right)^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A} \tag{78}$$

$$\mathbf{S}_k = \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{A} - \mathbf{B}\mathbf{K}_k) + \mathbf{Q}. \tag{79}$$

Finally, yet another equivalent version of Riccati equation is known as *Joseph's stabilized Riccati equation*

$$\mathbf{S}_k = (\mathbf{A} - \mathbf{B}\mathbf{K}_k)^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{A} - \mathbf{B}\mathbf{K}_k) + \mathbf{K}_k^{\mathrm{T}}\mathbf{R}\mathbf{K}_k + \mathbf{Q}. \tag{80}$$

Showing the equivalence is an exercise.

2.3.1 Second order sufficient conditions

So far we only found a solution that satisfies the first-order necessary equation but we have been warned at the introductory lessons to optimization that such solution need not necessarily constitute an optimum (minimum in our case). In order to check this, the second derivative (Hessian, curvature matrix) must be found and checked for positive definiteness. Our strategy will be to find the value of the optimal cost first and then we will identify its second derivative with respect to $\mathbf{u}_k$.

The trick to find the value of the optimal cost is from [1] and it is fairly technical. By this we want to express that it may be hard to learn a general lesson from these actual steps. Nonetheless we will need the result. The procedure is based on the observation that

$$\frac{1}{2}\sum_{k=0}^{N-1}(\mathbf{x}_{k+1}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1}-\mathbf{x}_k^{\mathrm{T}}\mathbf{S}_k\mathbf{x}_k)=\frac{1}{2}\mathbf{x}_N^{\mathrm{T}}\mathbf{S}_N\mathbf{x}_N-\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{S}_0\mathbf{x}_0. \tag{81}$$

Now consider our optimization criterion and add zero to it. The value of the cost function does not change. Weird procedure, right? Observing that zero can also be expressed as the right hand side minus the left hand side in the above equation, we get

$$J_0=\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{S}_0\mathbf{x}_0+\frac{1}{2}\sum_{k=0}^{N-1}\left[\mathbf{x}_{k+1}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1}+\mathbf{x}_k^{\mathrm{T}}(\mathbf{Q}-\mathbf{S}_k)\mathbf{x}_k+\mathbf{u}_k^{\mathrm{T}}\mathbf{R}\mathbf{u}_k\right]. \tag{82}$$

Substituting the state equation, the cost function transforms to

$$\begin{aligned}J_0=\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{S}_0\mathbf{x}_0+\frac{1}{2}\sum_{k=0}^{N-1}[\mathbf{x}_k^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}+\mathbf{Q}-\mathbf{S}_k)\mathbf{x}_k+\mathbf{x}_k^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}\mathbf{u}_k\\+\mathbf{u}_k^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}\mathbf{x}_k+\mathbf{u}_k^{\mathrm{T}}(\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}+\mathbf{R})\mathbf{u}_k].\end{aligned} \tag{83}$$

Substituting the Riccati equation (77) into the first term above

$$\begin{aligned}J_0=\frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{S}_0\mathbf{x}_0+\frac{1}{2}\sum_{k=0}^{N-1}[\mathbf{x}_k^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}(\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}+\mathbf{R})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A})\mathbf{x}_k+\mathbf{x}_k^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}\mathbf{u}_k\\+\mathbf{u}_k^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}\mathbf{x}_k+\mathbf{u}_k^{\mathrm{T}}(\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}+\mathbf{R})\mathbf{u}_k].\end{aligned} \tag{84}$$

The time-varying Hessian (second derivative of the optimization criterion) with respect to the control is

$$J_{uu,k}=\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}+\mathbf{R}. \tag{85}$$

Provided that $\mathbf{R}>0$, it can be seen that it is always guaranteed that $J_{uu}>0$. To prove this it must be shown that $\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}\geq 0$. As usual, let us make things more intuitive by switching to the scalar case. The previous expression simplifies to $b^2 s_{k+1}$. No matter what the value of b is, the square is always nonnegative. It remains to show that $s_{k+1}\geq 0$ (and in general $\mathbf{S}_{k+1}\geq 0$). This can be seen from the prescription for $\mathbf{S}_k$ given by the Riccati equation (71) using similar arguments for proving positive semidefiniteness of compound expressions.

To conclude, the solution provided by first-order conditions represented by the Riccati equation is always a minimizing solution.

We can work a bit more with the value of the optimal cost. Substituting the optimal control (74) we can see (after some careful two-line work) that

$$J_0 = \frac{1}{2}\mathbf{x}_0^{\mathrm{T}}\mathbf{S}_0\mathbf{x}_0. \tag{86}$$

The same conclusion can be obtained for any time instant k inside the interval $[0, N]$

$$J_k = \frac{1}{2}\mathbf{x}_k^{\mathrm{T}}\mathbf{S}_k\mathbf{x}_k. \tag{87}$$

This is a result that we have already seen in the no-control case: the optimal cost can be obtained as a quadratic function of the initial state using a matrix obtained as a solution to some iteration. We will use this result in the future derivations.

2.3.2 Numerical example with a scalar and first-order system

Example 2.1. *As usual, some practical insight can be developed by analyzing the things when restricted to the scalar case. For this, consider a first order system described by the first-order state equation*

$$x_{k+1} = ax_k + bu_k \tag{88}$$

and the optimization criterion in the form

$$J_0 = \frac{1}{2}s_N x_N^2 + \frac{1}{2}\sum_{k=0}^{N-1}\left[qx_k^2 + ru_k^2\right]. \tag{89}$$

The scalar Riccati equation simplifies to

$$s_k = a^2 s_{k+1} - \frac{a^2b^2s_{k+1}^2}{b^2s_{k+1}+r} + q \tag{90}$$

or

$$s_k = \frac{a^2rs_{k+1}}{b^2s_{k+1}+r} + q. \tag{91}$$

Obviously the final state is not particularly close to zero, which is the desired final value. However, increasing the s_N term we can bring the system arbitrarily close, as the next simulation in Fig. 2 confirms.

The last outputs suggests that both s_N and K_k stay constant for most of the control interval and only change dramatically towards the end of the control interval. This is indeed the case as highlighted in the next simulation output in Fig. 3 when the control interval was stretched a bit.

The observation in the example poses a question of how much is lost after replacing the optimal control represented by the sequence $\mathbf{K}_k$ by a constant value $\mathbf{K}$. A natural candidate is the steady-state value that $\mathbf{K}_k$ has as the beginning of the control interval, that is at $k = 0$ in our case.

Obviously, on a finite-horizon there is not much to be investigated, the constant feedback gain is just *suboptimal*, but things are somewhat more involved as the control horizon stretches to infinity, that is, $N \to \infty$. Note that allowing this is just a formal mathematical relaxation—now the optimization is done over the set of infinite control sequences but how fast or slow the response of our controlled system is depends on the choice of the weighting matrices. We will investigate this next.

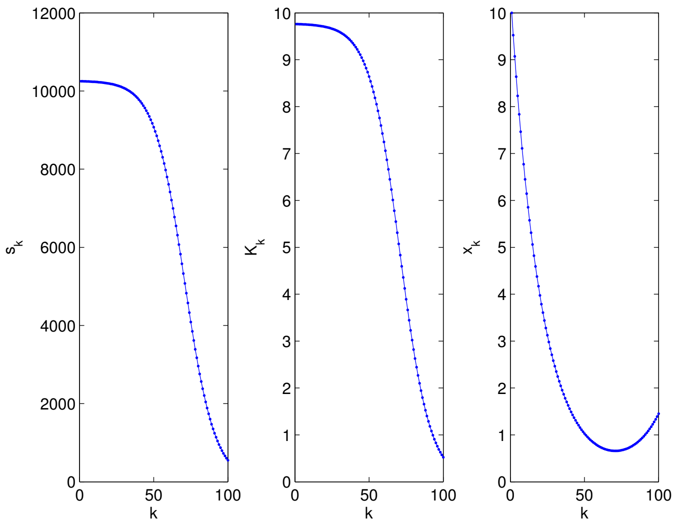

Figure 1: The optimal values for the solution sequence s_k or Riccati equation, the state feedback gain K_k and the states x_k.

3 LQ-optimal control over an infinite horizon—algebraic Riccati equation (ARE)

The design decision we make now is to replace the time-varying LQ-optimal state feedback gain $\mathbf{K}_k$ by a constant gain $\mathbf{K}$. This was motivated by the observation that on a sufficiently long control interval, it can be observed that $\mathbf{K}_k$ appears constant for most of the interval and it starts evolving dramatically only towards the end of the control interval (see the example at the end of the previous lecture). The irresistible idea is that the time-varying sequence of state-feedback gains $\mathbf{K}_k$ could be replaced by the steady state value of $\mathbf{K}_k$. If we now extend the control horizon to infinity, that is, if $N \to \infty$, the steady state value of the state-feedback gain matrix can be denoted as $\mathbf{K}_\infty$.

Some comment may be needed here to explain the notation. Remeber that we decided to consider the time $k = 0$ as the beginning of our control interval and the time N as the end of the control interval. Stretching the interval, that is, making N to approach ∞, the steady-state value of $\mathbf{K}_k$ is achieved toward the beginnning of the control interval, that is, at $k = 0$. It could be perhaps more appropriate to denote the steady-state value of $\mathbf{K}_k$ as $\mathbf{K}_0$. But thanks to time invariance, we can also fix the final time to $k = N$ and strech the interval by moving its beginning toward $-\infty$. The steady state of the sequence $\mathbf{K}_k$ can be then considered at $k = -\infty$. Would not it be then more appropriate to call the steady-state value $\mathbf{K}_{-\infty}$? Could be. Nonetheless, the commonly accepted notation for the steady state value found in the literature is $\mathbf{K}_\infty$, that is

$$\mathbf{K}_\infty \triangleq \lim_{k\to-\infty} \mathbf{K}_k. \tag{92}$$

Identical notational convention holds for the steady-state value of $\mathbf{S}_k$, which also evolves backwards in time and from which the above state-feedback gain matrix is

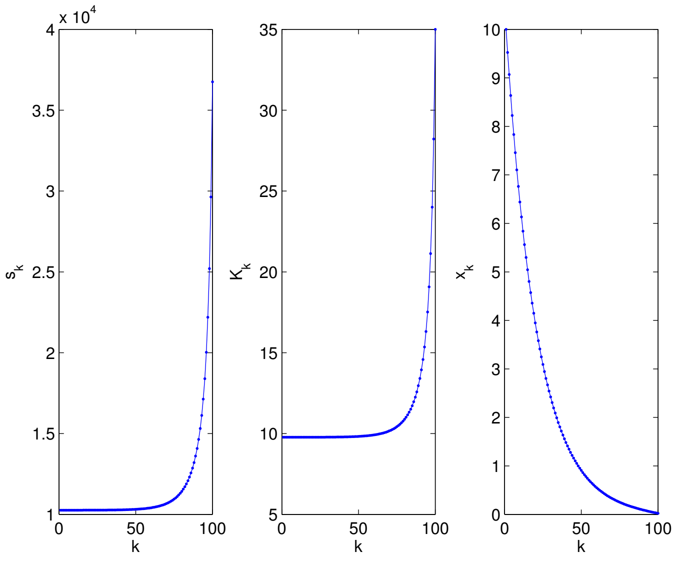

Figure 2: The optimal values for the solution sequence s_k or Riccati equation, the state feedback gain K_k and the states x_k.

actually derived

$$\mathbf{S}_\infty \triangleq \lim_{k\to-\infty} \mathbf{S}_k. \tag{93}$$

Leaving aside for the moment the important question whether and under which conditions such a limit exists, the immediate question is how to compute such limit. One straightforward strategy is to run the recurrent scheme (Riccati equation) generating the sequence $\mathbf{S}_N, \mathbf{S}_{N-1}, \mathbf{S}_{N-2}, \ldots$ so long as there is a nonnegligible improvement, that is, once $\mathbf{S}_k \approx \mathbf{S}_{k+1}$, stop iterating.

Another idea is to apply the steady-state condition

$$\mathbf{S}_\infty = \mathbf{S}_k = \mathbf{S}_{k+1} \tag{94}$$

to the Riccati equation. The resulting equation

$$\mathbf{S}_\infty = \mathbf{A}^{\mathrm{T}}\left[\mathbf{S}_\infty - \mathbf{S}_\infty\mathbf{B}(\mathbf{B}^{\mathrm{T}}\mathbf{S}_\infty\mathbf{B}+\mathbf{R})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_\infty\right]\mathbf{A} + \mathbf{Q} \tag{95}$$

is called discrete-time *Algebraic Riccati Equation* (DARE) and it is one of the most important equations in the field of computational control design. The equation may look quite “messy” and offers hardly any insight. Remember the good advice to switch to the scalar case while studying similar matrix-vector expressions. Our ARE simplifies to

$$s_\infty = a^2 s_\infty - \frac{a^2b^2s_\infty^2}{b^2 s_\infty + r} + q \tag{96}$$

This equation can be solved by solving the corresponding quadratic (in s_∞) equation

$$b^2 s_\infty^2 + (r - a^2b^2 - b^2 q)s_\infty - qr = 0. \tag{97}$$

Voilà! A quadratic equation for which the solution(s) can be found readily. There is a caveat here, though. Quadratic equation can have two (or none) real solutions.

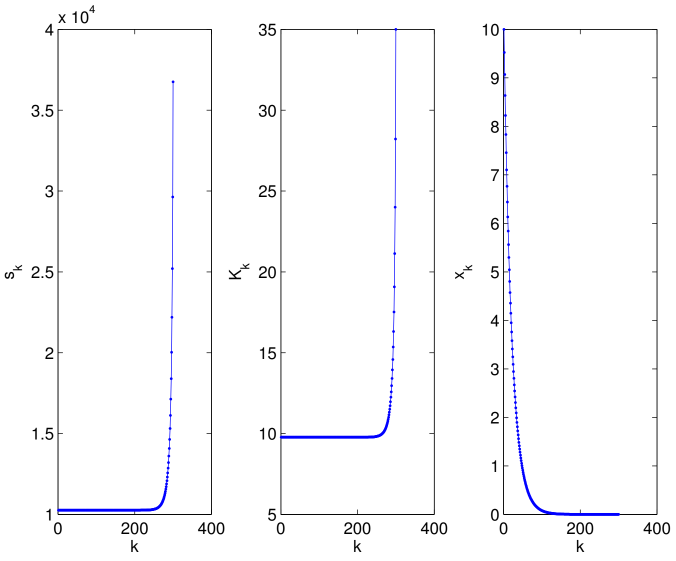

Figure 3: The optimal values for the solution sequence s_k or Riccati equation, the state feedback gain K_k and the states x_k.

But the sequence produced by recurrent Riccati equation is determined uniquely! What's up? How are the solutions to ARE related to the limiting solution of recurrent RE?

Answering this question will keep us busy for most of this lecture. We will structure this broad question into several sub-questions

1. under which conditions it is guaranteed that there exists a (bounded) limiting solution $\mathbf{S}_\infty$ to the recurrent Riccati equation for all initial (actually final) values $\mathbf{S}_N$?
2. under which conditions is the limit solution unique for arbitrary $\mathbf{S}_N$?
3. under which conditions is it guaranteed that the time-invariant feedback gain $\mathbf{K}_\infty$ computed from $\mathbf{S}_\infty$ stabilizes the system (on the infinite control interval)?

3.1 Suboptimal solutions

Before we start answering the three important question posed above, let us investigate the cost of any (!) suboptimal control $\mathbf{K}_k$ (including the constant gain). In order to to this, we will invoke the same trick as we used for finding the const of the optimal control in the previous question. Namely, the observation that

$$\frac{1}{2}\sum_{k=i}^{N-1}(\mathbf{x}_{k+1}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1}-\mathbf{x}_k^{\mathrm{T}}\mathbf{S}_k\mathbf{x}_k)=\frac{1}{2}\mathbf{x}_N^{\mathrm{T}}\mathbf{S}_N\mathbf{x}_N-\frac{1}{2}\mathbf{x}_i^{\mathrm{T}}\mathbf{S}_i\mathbf{x}_i. \tag{98}$$

Adding the left hand side to and subtracting the right hand side to from the cost function (in other words, adding zero to the cost function) we get

$$J_i=\frac{1}{2}\mathbf{x}_i^{\mathrm{T}}\mathbf{S}_i\mathbf{x}_i+\frac{1}{2}\sum_{k=i}^{N-1}\left[\mathbf{x}_{k+1}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{x}_{k+1}+\mathbf{x}_k^{\mathrm{T}}(\mathbf{Q}-\mathbf{S}_k)\mathbf{x}_k+\mathbf{u}_k^{\mathrm{T}}\mathbf{R}\mathbf{u}_k\right]. \tag{99}$$

Substituting the state equation and the state feedback control $\mathbf{u}_k = -\mathbf{K}_k\mathbf{x}_k$ into the above, we get

$$J_i = \frac{1}{2}\mathbf{x}_i^{\mathrm{T}}\mathbf{S}_i\mathbf{x}_i + \frac{1}{2}\sum_{k=0}^{N-1}\mathbf{x}_k^{\mathrm{T}}\left[(\mathbf{A}-\mathbf{B}\mathbf{K}_k)^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{A}-\mathbf{B}\mathbf{K}_k)+\mathbf{Q}-\mathbf{S}_k+\mathbf{K}_k^{\mathrm{T}}\mathbf{R}\mathbf{K}_k\right]\mathbf{x}_k. \tag{100}$$

However, at this point the sequence $\mathbf{S}_k$ is undefined. At this moment these are just some symbols for us. Maybe we should have used a symbol different from the solution to Riccati equation for it. Now, let us define this sequence as

$$\mathbf{S}_k = (\mathbf{A}-\mathbf{B}\mathbf{K}_k)^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{A}-\mathbf{B}\mathbf{K}_k)+\mathbf{Q}+\mathbf{K}_k^{\mathrm{T}}\mathbf{R}\mathbf{K}_k \tag{101}$$

With this choice the cost function (the optimization criterion) simplifies to

$$J_i = \frac{1}{2}\mathbf{x}_i^{\mathrm{T}}\mathbf{S}_i\mathbf{x}_i. \tag{102}$$

For the third time in this course we observe that the cost of the control when starting at a given discrete time i can be calculated as a quadratic matrix form with the coefficient matrix obtained from some matrix recurrent scheme. Make sure you understand now that (101) is not Riccati equation. Instead, it is just Lyapunov equation since $\mathbf{K}_k$ is fixed. It can be even constant.

Example 3.1. *In this example we analyze numerically what an impact on the cost function the shift from the time-varying to the time-invariant state feedback is. We stick to the scalar first-order assignment used previously. See the Fig. 4 and 5. The parameters of the first-order LTI system and the LQ optimal control are*

```
a=1.05; b=0.01;
q=5; r=5; sN=5;
x0=10;
N=100;
```

For these parameters, the recurrent scheme(s) given in the previuos lecture give the sequences s_k as in the left column of Fig. 5. Corresponding to the optimal solution is the blue sequence while the green one is suboptimal on the finite interval (and optimal in steady state). The red sequence s_k on the left is is generated for"some" constant stabilizing feedback gain, the trick that we used to find an upper bound on the optimal s_k.

The state trajectories in the right column confirm the already discussed well-known fact that the optimization only does what it was asked for. In this case, it simply minimizes the chosen quadratic criterion. The suboptimal solution which was obtained by solving ARE can then easily turn out more appropriate although it is only optimal for an infinite time horizon.

Looking at the evolution of costs in Fig. 5, one notices immediately that the differences between the cost function of the optimal control and the suboptimal one (obtained by solving ARE) are very small. This suggests that our losses in optimality introduced by using a steady-state solution to Riccati equaton are tolerable.

3.2 Boundedness of the solution to recurrent Riccati equation

Let us state the answer first: the system $\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k$ must be stabilizable in order to guarantee existence of a bounded limiting solution $\mathbf{S}_\infty$ to Riccati solution.

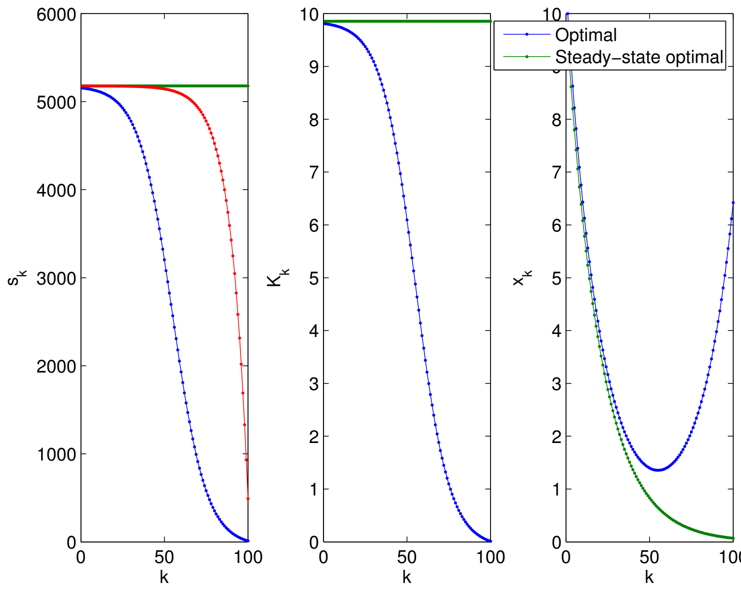

Figure 4: Comparison of cost functions for the optimal (blue) and suboptimal (green) feedback gains (optimal in steady state). The red sequence s_k on the left is is generated for“some” constant feedback gain.

To see this, note that for a stabilizable system, we can find some time-invariant feedback gain, which guarantes that $\mathbf{x}_k \to 0$ as $k \to \infty$ (note, once again, that for LTI systems the situation with fixed and finite N and k going toward $-\infty$ can be view equivalenty as if k goes from 0 to ∞.) Knowing this and recalling also that our cost is

$$J_i = \frac{1}{2}\mathbf{x}_N^{\mathrm{T}}\mathbf{S}_N\mathbf{x}_N + \frac{1}{2}\sum_{k=i}^{N-1}\mathbf{x}_k^{\mathrm{T}}\mathbf{Q}\mathbf{x}_k + \mathbf{u}_k^{\mathrm{T}}\mathbf{R}\mathbf{u}_k, \tag{103}$$

we can argue that $\lim_{i\to-\infty} J_i$ is finite. At the same moment, the cost function J_i for this any suboptimal feedback gain must be at any i higher or equal to the cost function for the optimal state feedback gain; this is the very definition of an optimal solution. Let us use temporarily the symbol $*$ to denote the objects related to the optimal control. Hence

$$J_i \geq J_i^*. \tag{104}$$

Therefore

$$\mathbf{x}_i^{\mathrm{T}}\mathbf{S}_i\mathbf{x}_i \geq \mathbf{x}_i^{\mathrm{T}}\mathbf{S}_i^*\mathbf{x}_i, \tag{105}$$

which can also be written as

$$\mathbf{S}_i \geq \mathbf{S}_i^*. \tag{106}$$

In other words, we have just shown that for a stabilizable system the optimal sequence $\mathbf{S}_k$ is bounded at every k.

Two more properties of the optimal $\mathbf{S}_k$ can be identified upon consulting one of the version of Riccati equation or the other.

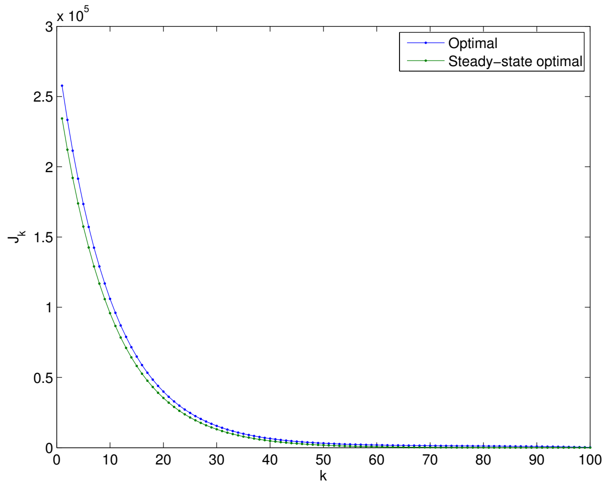

Figure 5: Comparison of cost functions for the optimal (blue) and suboptimal (green) feedback gains (optimal in steady state).

1. $\mathbf{S}_k$ is symmetric. This is obvious once we observe that by transposing the Riccati equation derived in (77)

$$\mathbf{S}_k = \mathbf{Q} + \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A} - \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}(\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{B}+\mathbf{R})^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1}\mathbf{A}, \tag{107}$$

   we obtain the same expression.

2. $\mathbf{S}_k$ is positive semidefinite provided $\mathbf{S}_N \geq 0$. This is obvious from the other form of Riccati equation that we have derived in 71 and we display here for convenience

$$\mathbf{S}_k = \mathbf{Q} + \mathbf{A}^{\mathrm{T}}\mathbf{S}_{k+1}(\mathbf{I}+\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{S}_{k+1})^{-1}\mathbf{A}. \tag{108}$$

   Just use the couple of rules such as that a squared matrix is always positive semidefinite, or that a sum of two semidefinite matrices is a semidefinite matrix. As usual, resorting to scalars will lend some insight.

3. $\mathbf{S}_\infty$ solves the ARE. This is obvious.

3.3 Stabilizing solution of ARE

Now let's skip the second of our three original questions (the one about uniqueness) temporarily and focus on the third question. In order to answer it, let us extend our state-space model with the "artificial" output equation

$$\mathbf{y}_k = \begin{bmatrix}\mathbf{C}\\ \mathbf{0}\end{bmatrix}\mathbf{x}_k + \begin{bmatrix}\mathbf{0}\\ \mathbf{D}\end{bmatrix}\mathbf{u}_k, \quad k=0,1,\ldots,N-1, \qquad \mathbf{y}_N = \begin{bmatrix}\mathbf{C}\\ \mathbf{0}\end{bmatrix}\mathbf{x}_N, \tag{109}$$

where $\mathbf{Q}=\mathbf{C}^{\mathrm{T}}\mathbf{C}$, $\mathbf{R}=\mathbf{D}^{\mathrm{T}}\mathbf{D}$ and $\mathbf{S}_N = \mathbf{C}_N^{\mathrm{T}}\mathbf{C}_N$. With this new somewhat artificial system, our original optimization criterion can be rewritten

$$J_i = \frac{1}{2}\mathbf{y}_N^{\mathrm{T}}\mathbf{y}_N + \frac{1}{2}\sum_{k=i}^{N-1}\mathbf{y}_k^{\mathrm{T}}\mathbf{y}_k = \frac{1}{2}\sum_{k=i}^{N}\mathbf{y}_k^{\mathrm{T}}\mathbf{y}_k. \tag{110}$$

From the previous analysis we know that thanks to stabilizability the optimal cost function is always bounded (by a finite cost for some suboptimal stabilizing controller)

$$J_\infty^* = \frac{1}{2}\sum_{k=i}^{\infty} \mathbf{y}_k^{\mathrm{T}}\mathbf{y}_k < \infty. \tag{111}$$

Having a bounded sum of an infinite number of squared terms, it follows that $\mathbf{y}_k \to 0$ as $k \to \infty$ by which we mean a rigorous statement $\|\mathbf{y}_k\| \to 0$ as $k \to \infty$. The crucial question now is: does the fact that $\mathbf{C}\mathbf{x}_k \to 0$ and $\mathbf{D}\mathbf{u}_k \to 0$ for $k \to \infty$ also imply that $\mathbf{x}_k \to 0$ and $\mathbf{u}_k \to 0$?

Since $|\mathbf{R}| \neq 0$, $\mathbf{u}_k \to 0$. But we made no such restrictive assumption about $\mathbf{Q}$. In the very extreme case assume that $\mathbf{Q} = \mathbf{0}$. What will happen with an unstable system is that our optimization criterion only contains the control sequence $\mathbf{u}_k$ and it naturally minimizes the total cost by setting $\mathbf{u}_k = 0$. The result is catastrophic — the system goes unstable untill it blows out. The fact that the states $\mathbf{x}_k$ of the system diverge goes unnoticed by the optimization criterion. The easiest fix of this situation is to require $\mathbf{Q}$ nonsingular as we did for $\mathbf{R}$ (for other reasons), but this is a way too restrictive. We will shortly see an example where nonsingular weighting matrix $\mathbf{Q}$ might be useful.

The ultimate answer is that the condition under which the blowing out of the system states is always reflected in the optimization cost is *detectability* of the artificial system given by the matrix pair $(\mathbf{A}, \mathbf{C})$. In practice we may check for observability (similarly as we do for controllability instead of detectability) but doing so we request more then is needed. Note that unlike in the fixed final state here we need neither controllability nor observability to have a stabilizing controller.

3.4 Uniqueness of the stabilizing solution

The last issue that we have to solve is uniqueness. Why do we need to care? We have already discussed that the ARE, being a quadratic equation, can have more than just one solution. In the scalar case it can have two real solutions (or no real because both complex). In general there are then several posibilities

1. None of them is nonnegative. Bad luck, no stabilizing solution can be found. This can be, however, excluded if the system $(\mathbf{A}, \mathbf{B})$ is stabilizable as discussed before.

2. Only one of them is nonnegative. We are lucky because this is our stabilizing solution.

3. Both solutions are nonnegative. Which one we shall pick? Both seem to be acceptable candidates but only one of them truly corresponds to the optimal solution. Is it possible to exclude this scenario?

Our goal is to study if we can exclude the last scenario — multiple positive semidefinite solutions of ARE. Provided the system is stabilizable, we know that one of them is our optimal solution but we will have troubles to identify the correct solution. In other words, we are asking if there is a unique stabilizing solution to ARE. We will use ARE in the Joseph stabilized form

$$\mathbf{S} = (\mathbf{A} - \mathbf{B}\mathbf{K})^{\mathrm{T}}\mathbf{S}(\mathbf{A} - \mathbf{B}\mathbf{K}) + \mathbf{K}^{\mathrm{T}}\mathbf{R}\mathbf{K} + \mathbf{Q}. \tag{112}$$

Using our factorization of $\mathbf{Q}$ and $\mathbf{R}$ we can write the Riccati equation as

$$\mathbf{S} = (\mathbf{A} - \mathbf{B}\mathbf{K})^{\mathrm{T}}\mathbf{S}(\mathbf{A} - \mathbf{B}\mathbf{K}) + \begin{bmatrix}\mathbf{C}^{\mathrm{T}} & \mathbf{K}\mathbf{D}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{C}\\ \mathbf{D}\mathbf{K}\end{bmatrix}. \tag{113}$$

Observe that for a fixed (optimal) $\mathbf{K}$ (we stop using ∞ in the lower index for notational brevity) the above equation is actually a Lyapunov equation for an equivalent system with the state matrices $(\mathbf{A} - \mathbf{B}\mathbf{K}, \begin{bmatrix}\mathbf{C}\\ \mathbf{D}\mathbf{K}\end{bmatrix})$. We have already refreshed the well-known facts about Lypunov equation that provided the system is stable (and in our case it is since $(\mathbf{A} - \mathbf{B}\mathbf{K})$ matrix is) and observable, the equation has a unique positive definite solution. If observability is not guranteeed, there is a positive definite solution.

Clearly, what remains to be shown is that the system $(\mathbf{A} - \mathbf{B}\mathbf{K}, \begin{bmatrix}\mathbf{C}\\ \mathbf{D}\mathbf{K}\end{bmatrix})$ is observable. Let us invoke one of the tests of observability — the popular PBH test. For a system $(\mathbf{A}, \mathbf{C})$ it consists in creating a matrix $\begin{bmatrix}z\mathbf{I} - \mathbf{A}\\ \mathbf{C}\end{bmatrix}$ and checking if it is full rank for every complex z. In our case this test specializes to checking the rank of

$$\begin{bmatrix}z\mathbf{I} - (\mathbf{A} - \mathbf{B}\mathbf{K})\\ \mathbf{C}\\ \mathbf{D}\mathbf{K}\end{bmatrix}, \tag{114}$$

which can be shown to be equal to the rank of

$$\begin{bmatrix}z\mathbf{I} - \mathbf{A}\\ \mathbf{C}\\ \mathbf{D}\mathbf{K}\end{bmatrix}. \tag{115}$$

This expresses the well-known fact from linear systems that state feedback preserves observability. If $\mathbf{K}$ can be arbitrary, the only way to keep this rank full is to guarantee that

$$\begin{bmatrix}z\mathbf{I} - \mathbf{A}\\ \mathbf{C}\end{bmatrix} \tag{116}$$

is full rank. In other words, the system $(\mathbf{A}, \mathbf{C})$, where $\mathbf{C} = \sqrt{\mathbf{Q}}$, must be observable.

Let us summarize the findings: although detectability of $(\mathbf{A}, \mathbf{C})$ is enough to guarantee the existence of a positive semidefinite $\mathbf{S}$ which stabilizes the system, if the detectability condition is made stronger by requiring observability, it is guaranteed that there will be a unique positive definite solution to ARE.

Why do we need to care about positive definiteness of $\mathbf{S}$? Let us consider a scalar case for illustration.

Example 3.2 (Solution to scalar ARE). *For a first-order system $x_{k+1} = ax_k + bu_k$ and the standard cost $J = \frac{1}{2}\sum_{k=0}^{\infty}[qx_k^2 + ru_k^2]$ the corresponding ARE is*

$$s = q + a^2 s - \frac{a^2b^2s}{b^2s + r}, \tag{117}$$

which can be turned into

$$s(b^2s + r) = q(b^2s + r) + a^2s(b^2s + r) - a^2b^2s. \tag{118}$$

Grouping together the coefficients with equal powers of s yields the quadratic equation in the standard form

$$(b^2a^2 - b^2)s^2 + (b^2q - r + ra^2 - a^2b^2)s + rq = 0. \tag{119}$$

Trivial analysis shows for $q = 0$, one of the roots is $s_1 = 0$ and the other is always $s_2 < 0$. Hence the solution of ARE that represents the steady-state solution of the recurrent Riccati equation is $s = 0$. As a consequence, the optimal state-feedback gain is $k = 0$. For an unstable system this would be unacceptable but for a stable system this makes sense: the system is stable even withouth the control, therefore, when the state is not penalized in the criterion at all ($q = 0$), the optimal strategy is not regulating at all. Mathematically correct. Nonethelesss, from an engineering viewpoint we may be quite unhappy because the role of the feedback regulator is also to attenuate the influence of external disturbances. Our optimal state-feedback regulator does not help at all in these situation. That is why we may want to require positive definite solution of ARE.

4 Computational example using Matlab

Our task is to design an LQ-optimal state feedback regulator for an F16 aircraft. This assignment is based on Example 5.3-1 from Stevens & Lewis 2003, page 413. Linear model of lateral-directional dynamics of F16 trimmed at: VT=502ft/s, 302psf dynamic pressure, cg @ 0.35cbar. The model includes dynamics of ailerons and rudders and a washout filter. The model can be created in Matlab by running the following code.

```
% beta    ... sideslip angle
% phi     ... bank angle
% p       ... roll rate
% r       ... yaw rate
%
% delta_a  ... aileron deflection
% delta_r  ... rudder deflection
%
% r_w      ... filtered yaw rate

A = [-0.3220,   0.0640,   0.0364,   -0.9917,   0.0003,   0.0008   0;
     0,         0,        1,        0.0037,    0,        0,       0;
     -30.6492,  0,        -3.6784,  0.6646,    -0.7333,  0.1315,  0;
     8.5396,    0,        -0.0254,  -0.4764,   -0.0319,  -0.062,  0;
     0,         0,        0,        0,         -20.2,    0,       0;
     0,         0,        0,        0,         0,        -20.2,   0;
     0,         0,        0,        57.2958,   0,        0,       -1];

B = [0,     0;
     0,     0;
     0,     0;
     0,     0;
     20.2,  0;
     0,     20.2;
     0,     0];

C = [0,        0,        0,        57.2958,  0,   0,   -1;
     0,        0,        57.2958,  0,        0,   0,   0;
     57.2958,  0,        0,        0,        0,   0,   0;
     0,        57.2958,  0,        0,        0,   0,   0];


G = ss(A,B,C,zeros(4,2));

set(G, 'StateName',{'beta','phi','p','r','delta_a','delta_r','r_w'});
set(G, 'OutputName',{'r_w','p','beta','phi'});
set(G,'InputName',{'u_a','u_r'});

Ts = 0.1;           % sampling period
Gd = c2d(G,Ts);     % discretized system
[A,B,C,D] = ssdata(Gd);
```

The key characteristic of this model is that it includes the dynamics of the servos that deflect the control surfaces (ailerons and ruders). These are represented by the state variables δ_a and δ_r. The measured yaw rate is filtered using a washout filter. The structure of the model is shown in Fig.6.

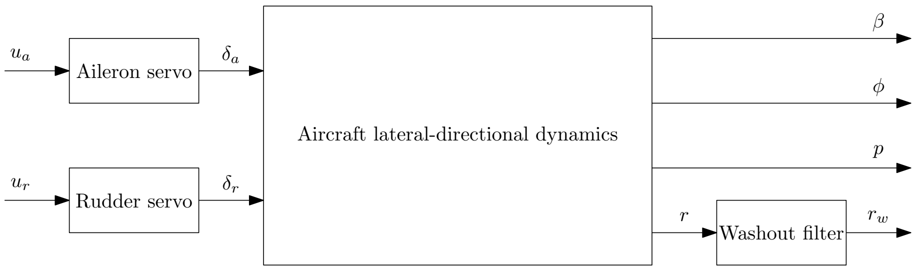

Figure 6: Structure of a model of directional-lateral dynamics of F16.

The task is to bring the system states to zero from a nonzero initial states. This assignment also covers the situation in which the system is exposed to external disturbances. Their impact on the system can be modeled as if they were setting a system into some nonzero initial state.

We formulate this and an LQ-optimal control design with the criterion

$$J = \sum_{k=0}^{\infty} [\mathbf{x}_k^{\mathrm{T}} \mathbf{Q} \mathbf{x}_k + \mathbf{u}_k^{\mathrm{T}} \mathbf{R} \mathbf{u}_k], \tag{120}$$

where our initial structure of the weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ might be

$$\mathbf{Q} = q\mathbf{I} = \begin{bmatrix} q & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q \end{bmatrix} \tag{121}$$

and

$$\mathbf{R} = \mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{122}$$

which would leave us with just a single design parameter q. That is, all the states are penalized equally. Remember also that the tradoff between supressing the regulation error and the keeping the control error small is expressed by the ratio between the entries of $\mathbf{Q}$ and $\mathbf{R}$, that is why $\mathbf{R}$ is set to an identity matrix here. This one-parameter setting, however, does not leave enough freedom for the control design.

Therefore the next natural choice might be

$$\mathbf{Q} = \begin{bmatrix} q_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q_5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q_6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q_7 \end{bmatrix} \tag{123}$$

and

$$\mathbf{R} = \begin{bmatrix} r_1 & 0 \\ 0 & r_2. \end{bmatrix} \tag{124}$$

Here we are in the opposite situation — each state variable and the control signal have their own weights. We may find the number of degrees of freedom too high. There are too many parameters to tune, although their interprettation is very intuitive. For example, the large the q_1, the faster the corresponding state variable — the sideslip angle — goes to zero.

There is one general rule of thumb, sometimes called Bryson's rule, which sais that q_i should be set to $\frac{1}{(\text{maximum acceptable } x_i)^2}$. In other words, the state variables that are and similarly r_i should be set to $\frac{1}{(\text{maximum acceptable } u_i)^2}$. This generally represents a good initial setting of the weighing matrices.

There is one special feature of our model, that will nicely illustrate the full power of the analysis presented in this lecture. Namely, note that although δ_a and δ_r are included in the system state vector, they also directly correspond to the control inputs u_a and u_r. As the Fig.6 explains, they are just smoothed control inputs. And the inputs are penalized using the **R** matrix. Why then penalized these states using the corresponding entries of the **Q** matrix? Therefore it seems to a good idea to set the corresponding matrices to zero. That is

$$\mathbf{Q} = \begin{bmatrix} q_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q_7 \end{bmatrix} \tag{125}$$

Unless we have enough physical insight into the problem which guides us, for instance through the Bryson's rule mentioned above, we start by set all the values to some arbitrary value, say 1

```
q_beta = 1;
q_phi = 1;
q_p = 1;
q_r = 1;
q_rw = 1;

r_a = 1;
r_r = 1;

Q = diag([q_beta q_phi q_p q_r 0 0 q_rw]);
R = diag([r_a r_r]);
```

We were warned in the previous sections that we must now check if the important condition $(\mathbf{A}, \sqrt{\mathbf{Q}})$ stabilizable is still satisfied. We will do it by testing for observability, which is a stronger property but easier to test.

```
rank(obsv(A,sqrt(Q)))   % check the observability condition
ans =
     7
```

The actual optimal control design is now a simple job. Using a dedicated solver **dare()** in *Control System Toolbox for Matlab*[2]

```
[S,E,K] = dare(A,B,Q,R);
```

Finally we build the closed-loop system and simulate a response to nonzero initial conditions. In particular, nonzero initial sideslip

```
G_closed = ss(A-B*K,B,eye(7),zeros(7,2));
set(G_closed,'OutputName',{'beta','phi','p','r','delta_a','delta_r','r_w'});
initial(G_closed,[1 0 0 0 0 0 0],10)
```

The responses are in Fig.7.

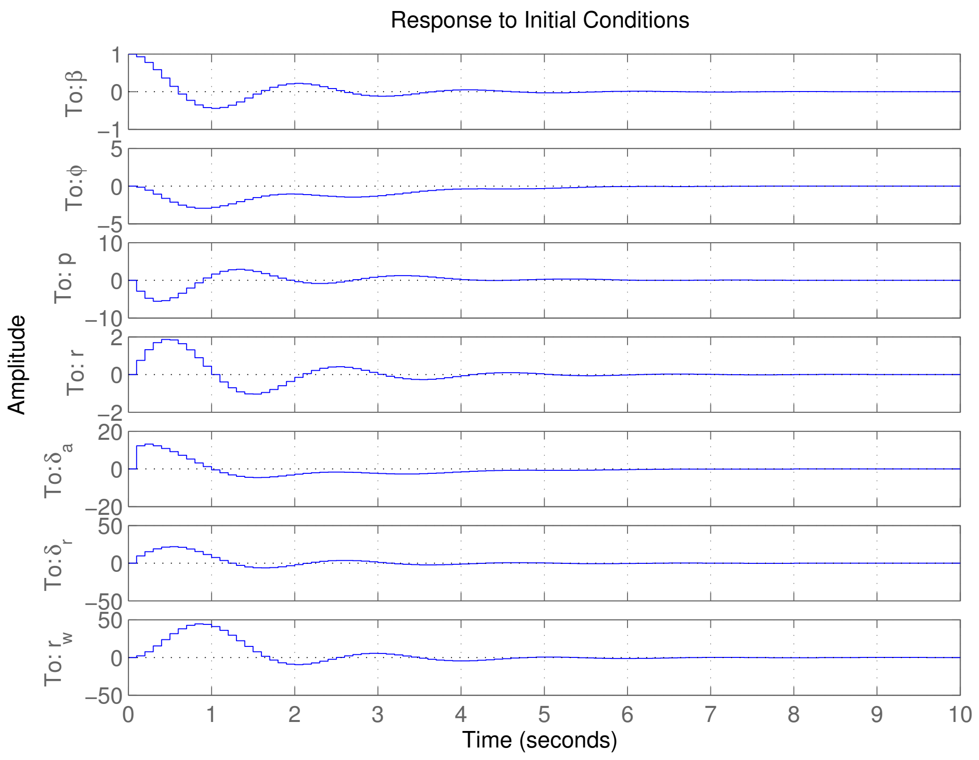

Figure 7: Response to initial conditions (nonzero sideslip β) with an LQ regulator.

Note that although our initial setting of the weighting matrices was somewhat arbitrary, we already received a reasonable-looking response. This should not be taken for granted! A few more iterations of finetuning the diagonal entries of the

[2]Solvers for discrete-time ARE can be found in every other software package for engineering computations: Mathematica, Maple, Scilab, Octave. Choose you own tool.

weighting matrices might be needed. As we will see later, this nice behaviour is guaranteed for arbitrary selections of $\mathbf{Q}$ and $\mathbf{R}$. Just compare it with the process of finding the coefficients for a PID controller — for some choices the response of the systems can be easily unacceptable, even unstable.

5 Summary

In this lecture we have considered the general discrete-time optimal control problem. Nonlinear and time-varying systems were allowed. Necessary first-order conditions optimality were derived which determined an optimal discrete-time control signal. The conditions came in the form of a boundary value problem. Although such problem is difficult to solve ingeneral, we considered a specialization of the result to the case of a linear time-invariant (LTI) system and a popular quadratic cost function (penalizing both the states and the controls). We analyzed two different situations—first, when the final state was required to attain some prespecified value, and then, with the state at the final state unspecified. Although even with the relaxed final state we can force the system to come arbitrarily close to the desired final value, the solutions to the two problems are strikingly different. For the fixed final state, the optimization computes offline the optimal control sequence, that is, we get an open-loop (or preprogrammed) control strategy. If instead we force the state at the final time to the desired value only indirectly—through a penalization in the criterion, the solution to the optimal control problem yields a time-varying state feedback! The feedback gains were obtained by solving a difference Riccati equation.

We then continued by observing that the solution to the difference Riccati equation (henc the state feedback gains too) remain constant for most of time and they only change dramatically toward the very end of the control interval. We therefore investigated the possibility to replace the optimal time-varying state feedback gains with some suboptimal but constant gains. Computationally this was handled using difference algebraic Riccati equation (DARE).

What we did not cover:

- we only considered the so-called regulation task, that is, the optimization criterion penalized deviation of the state from zero. An example of such task might be inertial stabilization of cameras where the task is to keep the inertial angular rate close to zero. There are, however, other control scenarios where the task is to make the states follow certain nonzero values. The optimization criterion must be changed accordingly. More on this in the fourth chapter of [1] or elsewhere under the name LQ optimal reference tracking.
- so-far we have only been investigating systems for which all the state variables were available for control. When this is not the case, we either need to reformulate the problem so that the outputs enter the optimization cost instead of the states (the task becomes tremendously difficult, see chapter eight of [1]), or we need to combine the optimal state feedback with an (optimal) observer. The latter will be investigate in one of the future lectures under the name of LQG optimal control.

6 Tracking

7 Further reading

This lecture was prepared to a major extent with the help of the second chapter of [1]. There are dozens of texts on LQ optimal control, but they mainly treat the continuous-time case.

References

[1] Frank L. Lewis and Vassilis L. Syrmos. *Optimal Control.* Wiley-Interscience, 2nd edition, October 1995.