AUTOMATIC CONTROL AND SYSTEM THEORY

DISCRETE TIME OPTIMAL CONTROL & THE KALMAN FILTER

Gianluca Palli

Dipartimento di Ingegneria dell'Energia Elettrica e dell'Informazione (DEI)
Università di Bologna
Email: *gianluca.palli@unibo.it*

Discrete Time Optimal Control

- **Problem statement**

Let us consider the system:

$$x(k+1) = A\,x(k) + B\,u(k) \;,\; x(0) = x_a$$

where $x \in R^n$, $u \in R^p$ *and* (A, B) stabilizable.

Performance index:

$$J = \frac{1}{2}\,x^T(k_b)\,S_b\,x(k_b) +$$

$$+ \frac{1}{2}\sum_{k=0}^{k_b-1}\left[x^T(k)\,Q\,x(k) + u^T(k)\,R\,u(k) + 2\,x^T(k)\,N\,u(k)\right]\,dt$$

where

$$S_b \geq 0 \;,\; R > 0 \;,\; \begin{bmatrix} Q & N \\ N^T & R \end{bmatrix} \geq 0$$

Also in this case the problem can be be defined with infinite-time horizon or with finite-time horizon.

Discrete Time Optimal Control

- **Definitions**

The Hamiltonian of the system is:

$$H = \frac{1}{2}\left(x^T Q x + u^T R u + 2x^T N u\right) + \lambda^T (k+1)(Ax + Bu)$$

where $\lambda \in R^n$ is the costate (or adjoint variable) vector.

The solution to the problem is obtained by considering the following adjoint system equations

$$\lambda(k) = \frac{\partial H}{\partial x} = A^T \lambda(k+1) + Qx(k) + Nu(k)$$

If the matrix A is invertible, this last equation can be rewritten as (step-forward formulation):

$$\lambda(k+1) = A^{-T}\lambda(k) - A^{-T} Q x(k) - A^{-T} N u(k)$$

Discrete Time Optimal Control

- **Definitions**

In the discrete-time case, the stationarity conditions are defined as:

$$\frac{\partial H}{\partial u} = R\,u(k) + N^T x(k) + B^T \lambda(k+1) = 0$$

If the matrix A is invertible, we obtain the extended dynamic system:

$$\begin{bmatrix} x(k+1) \\ \lambda(k+1) \end{bmatrix} = \begin{bmatrix} A & 0 \\ -A^{-T}Q & A^{-T} \end{bmatrix} \begin{bmatrix} x(k) \\ \lambda(k) \end{bmatrix} + \begin{bmatrix} B \\ -A^{-T}N \end{bmatrix} u(k)$$

$$0 = \begin{bmatrix} -BA^{-T}Q + N^T & B^T A^{-T} \end{bmatrix} \begin{bmatrix} x(k) \\ \lambda(k) \end{bmatrix} + \left[R - BA^{-T}N\right]\, u(k)$$

The costate initial conditions $\lambda(0)$ are not known!!!

Discrete Time Optimal Control

- **Solution**

It is important to note that, if the matrix *A* is invertible, the problem has been reformulated as computing the discrete-time control law that makes the output of the extended dynamic system always null. If *R>0* it is possible to compute *u(k)* from the stationarity condition:

$$u(k) = (R - B^T A^{-T} N)^{-1} \left[(B^T A^{-T} Q + N^T)\, x(k) - B^T A^{-T} \lambda(k) \right]$$

By sustituting the above expression of *u(k)* into the extended dynamic system the 2*n*-order (finite-difference) Hamiltonian system is obtained.

It is possible to show that the system presents eigenvalues couples that satisfy the relation

$$\lambda_i \ , \ 1/\lambda_i \ , \quad i = 1, \ldots, n$$

Discrete Time Optimal Control

- **General solution**

Similarly to the continuous-time case, we define a relation between the state and the costate at the time step k of the type

$$\lambda(k) = S(k)x(k)$$

$$\rightarrow \quad \lambda(k+1) = S(k+1)x(k+1)$$

From the stationarity condition

$$\frac{\partial H}{\partial u} = R\,u(k) + N^T x(k) + B^T \lambda(k+1) = 0$$

$$\begin{aligned} R\,u(k) &= -N^T x(k) - B^T S(k+1)x(k+1) \\ R\,u(k) &= -N^T x(k) - B^T S(k+1)(Ax(k) + Bu(k)) \\ u(k) &= -(R + B^T S(k+1)B)^{-1}(N^T + B^T S(k+1)A)x(k) \\ u(k) &= K(k)x(k) \end{aligned}$$

$$\rightarrow \quad K(k) = -(R + B^T S(k+1)B)^{-1}(B^T S(k+1)A + N^T)$$

Discrete Time Optimal Control

- **General solution**

From

$$\lambda(k+1) = S(k+1)x(k+1)$$

by substituting the system equations

$$\begin{aligned} A^T\lambda(k+1) &= A^T S(k+1)x(k+1) \\ A^T(\lambda(k) - Qx(k) - Nu(k)) &= A^T S(k+1)(Ax(k)+Bu(k)) \\ S(k)x(k) - Qx(k) - Nu(k) &= A^T S(k+1)(Ax(k)+Bu(k)) \end{aligned}$$

Finally, by substituting the expression of the control law

$$\begin{aligned} S(k) &= A^T S(k+1)A + \\ &\quad -(A^T S(k+1)B+N)(R+B^T S(k+1)B)^{-1}(B^T S(k+1)A+N^T)+Q \\ S(k_b) &= S_b \end{aligned}$$

This is the *finite difference algebraic Riccati equation*, that must be solved backward in time.

Discrete Time Optimal Control

- **Infinite time horizon**

In the infinite time horizon case, the problem converges to the steady state solution:

$$S = A^T SA - (A^T SB + N)(R + B^T SB)^{-1}(B^T SA + N^T) + Q$$

that is the Discrete-time Algebraic Riccati Equation (DARE).

Similarly to the continuous time case, the value of the preformance index is:

$$J = \frac{1}{2} x_a^T S\, x_a$$

In Matlab, the instruction *[S, L, K] = dare(A,B,Q,R,N)* can be used to compute the state feedback matrix *K* and the matrix *S*. Oppositely to the *care* instruction, that requires that *R* must be non-singular, the *dare* admits also *R=0* and *N=0*.

Discrete Time Optimal Control

- **Iterative solution**

The DARE can also be solved by means of the iterative relation

$$S_0 = I_n$$
$$S_{i+1} = A^T S_i A - (A^T S_i B + N)(R + B^T S_i B)^+ (B^T S_i A + N^T) + Q$$

that converges to the steady state solution S (note the use of the pseudo-inverse).

State Estimation from Noisy Measurements

- Suppose we have the discrete-time linear system with additive Gaussian white noise in both the process and the measurement equation

$$\begin{aligned} x(k+1) &= A\,x(k) + B\,u(k) + v(k) \quad , \quad x(0) = x_a \\ y(k) &= Cx(k) + w(k) \end{aligned}$$

- We want to use the available measurements *y* to estimate the state of the system *x*
- We know how the system behaves according to the state equation and we have partial state information thruogh the output
- How can we determine the best estimate of the state *x*?

The Kalman Filter – Definition

- First, we want the average value of our state estimate to be equal to the average value of the true state. That is, we don't want our estimate to be biased one way or another.
- Second, we want a state estimate that varies from the true state as little as possible. That is, we want to find the estimator with the smallest possible error variance.
- The *Kalman filter* is the estimator that satisfies these two criteria if certain assumptions about the noise that affects the system are satisfied
- We have to assume that the average value of v(k) is zero and the average value of w(k) is zero (*white Gaussian processes*).
- We have to further assume that no correlation exists between v(k) and w(k). That is, at any time k, v(k), and w(k) are *independent* random variables.

The Kalman Filter – Design by Duality

- Process noise V(k) and measurement noise W(k) covariance matrices

$$V(k) = E(v(k)v(k)^T), \quad W(k) = E(w(k)w(k)^T)$$

By duality with the design of the discrete-time optimal control problem

$$L(k) = AP(k)C^T(W(k) + CP(k)C^T)^{-1}$$

$$\hat{x}(k+1) = A\,\hat{x}(k) + B\,u(k) + L(k)(y(k) - C\hat{x}(k))$$

$$P(k+1) = AP(k)A^T - AP(k)C^T(W(k) + CP(k)C^T)^{-1}CP(k)A^T + V(k)$$

$$P(0) = E(x(0)x(0)^T)$$

- These equations define the so called *Kalman filter*
  - They can be solved iteratively at each time step
  - It consists in a state observer that minimize the estimation error due to measurement and process noises

The Kalman Filter – Origins and History

- The Kalman filter is essentially a set of mathematical equations that implement a predictor-corrector type estimator that is optimal in the sense that it minimizes the estimated error covariance
- Since the time of its introduction in 1960 by Rudolph E. Kalman, the Kalman filter has been the subject of extensive research and application, particularly in the area of spacecraft autonomous or assisted navigation
- This is likely due in large part to advances in digital computing that made the use of the filter practical, but also to the relative simplicity and robust nature of the filter itself
- In practice, rarely do the conditions necessary for optimality actually exist, and yet the filter apparently works well for many applications in spite of this situation
- Many different forms and implementations of the Kalman filter exist
- It applies also to non-linear systems (*Extended Kalman Filter, EKF*)

The Kalman Filter – Classic Formulation

- The Kalman filter addresses the general problem of trying to estimate the state of a discrete-time controlled process that is governed by the linear stochastic difference equations

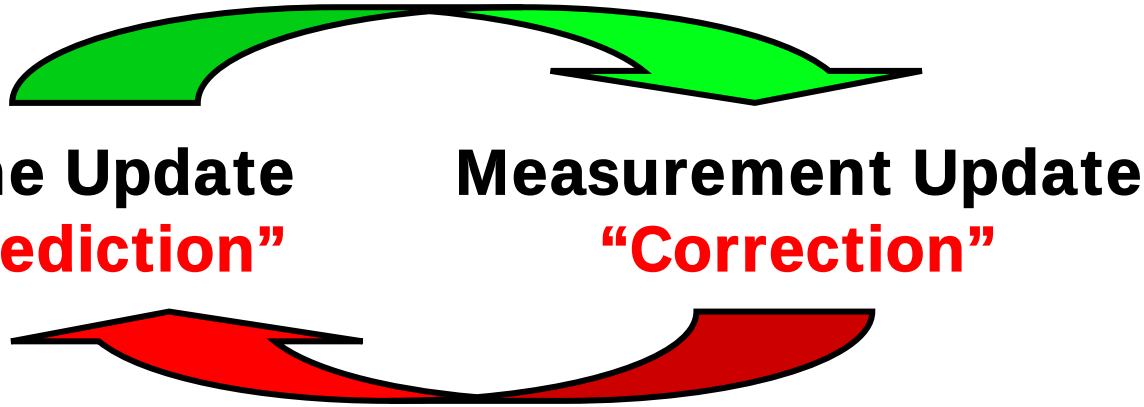
$$\begin{aligned} x_k &= A\,x_{k-1} + B\,u_k + v_k \\ y_k &= Cx_k + w_k \end{aligned}$$

- v(k) and w(k) represent the process and measurement noise respectively. They are assumed to be independent (of each other), white and Gaussian with covariance Q and R respectively.
- We define $\hat{x}_k^-$ (note the "super minus") to be our *a priori* state estimate at step *k* given knowledge of the process prior to step *k*, and $\hat{x}_k$ to be our *a posteriori* state estimate at step *k* given measurement y_k .
- We can then define *a priori* and *a posteriori* estimate errors and their covariance as

$$\begin{aligned} e_k^- &= x_k - \hat{x}_k^- & \qquad P_k^- &= E[e_k^- e_k^{-T}] \\ e_k &= x_k - \hat{x}_k & \qquad P_k &= E[e_k e_k^T] \end{aligned}$$

The Discrete Kalman Filter Algorithm

- The Kalman filter estimates the process state at some time and then obtains feedback in the form of (noisy) measurements
- The equations for the Kalman filter fall into two groups: *time update* equations and *measurement update* equations.
- The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the *a priori* estimates for the next time step.
- The measurement update equations are responsible for the feedback—i.e. for incorporating a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate.



The Discrete Kalman Filter Algorithm

- Discrete Kalman filter *time update equations:*

$$\begin{aligned} \hat{x}_k^- &= A\hat{x}_{k-1} + B\,u_k \\ P_k^- &= AP_{k-1}A^T + Q \end{aligned}$$

the time update equations project the state and covariance estimates forward from time step k-1 to step k.

- Discrete Kalman filter *measurement update equations:*

$$\begin{aligned} L_k &= P_k^- C^T (CP_k^- C^T + R)^{-1} \\ \hat{x}_k &= \hat{x}_k^- + L_k(y_k - C\hat{x}_k^-) \\ P_k &= (I - L_k C)P_k^- \end{aligned}$$

Innovation

the measurement update computes first the Kalman gain matrix L_k, then the measure y_k is used to generate an *a posteriori* state estimate. The final step is to obtain an *a posteriori* error covariance estimate.

The Discrete Kalman Filter Algorithm

- After each time and measurement update pair, the process is repeated with the previous *a posteriori* estimates used to project or predict the new *a priori* estimates.
- This recursive nature of the Kalman filter makes its practical implementation very easy and effective