AUTOMATIC CONTROL

Andrea M. Zanchettin, PhD
Winter Semester, 2019

Discrete time linear systems
Discrete time linear systems

Modern control strategies are implemented on computers. Such systems we have several differences with respect to physical systems:

- **time** is a piecewise continuous (monotonic) function which takes integer values (typically multiple of the CPU clock ticks)
- **variables** (input, state, output) are discrete and takes value only within a finite set (floating point, fixed point, etc)

We are then interested in studying the behaviour of discrete time system, as we have done for continuous ones.
Discrete time linear systems – cont’d

Another motivation behind the study of discrete time system is that, assuming we are able to observe a continuous time system only at discrete time, how can we characterize the behaviour of what we observe?
Discrete time linear systems – cont’d

**Serie A Classifica**

<table>
<thead>
<tr>
<th>#</th>
<th>Squadra</th>
<th>PG</th>
<th>V</th>
<th>P</th>
<th>S</th>
<th>GF</th>
<th>GS</th>
<th>DG</th>
<th>PT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Juventus</td>
<td>36</td>
<td>27</td>
<td>4</td>
<td>5</td>
<td>72</td>
<td>26</td>
<td>46</td>
<td>85</td>
</tr>
<tr>
<td>2</td>
<td>Roma</td>
<td>36</td>
<td>26</td>
<td>3</td>
<td>7</td>
<td>82</td>
<td>33</td>
<td>49</td>
<td>81</td>
</tr>
<tr>
<td>3</td>
<td>Napoli</td>
<td>36</td>
<td>24</td>
<td>8</td>
<td>4</td>
<td>86</td>
<td>36</td>
<td>50</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>Lazio</td>
<td>36</td>
<td>21</td>
<td>7</td>
<td>8</td>
<td>72</td>
<td>45</td>
<td>27</td>
<td>70</td>
</tr>
<tr>
<td>5</td>
<td>Atalanta</td>
<td>36</td>
<td>19</td>
<td>9</td>
<td>8</td>
<td>60</td>
<td>41</td>
<td>19</td>
<td>66</td>
</tr>
</tbody>
</table>

Points each team has after k matches

\[
x_{k+1} = x_k + u_k
\]

Points updated with new ones obtained after the last match
Discrete time linear systems – cont’d

As we have already seen a discrete time systems has the generic form

\[ x(k + 1) = f(x(k), u(k), k) \quad x(k_0) = x_0 \]

\[ y(k) = h(x(k), u(k), k) \quad x \in \mathbb{R}^n \]

We are now interested in understanding how the concept we developed for linear continuous time system apply for discrete time ones.

\[ x(k + 1) = Ax(k) + Bu(k) \]

\[ y(k) = Cx(k) + Du(k) \]

These concepts are: movement, stability, transfer function and frequency characterization.
Discrete time linear systems – cont’d

Differently from continuous time system, computing the solution of a discrete time one is trivial. In fact given

\[ x(k + 1) = Ax(k) + Bu(k) \quad x(k_0) = x_0 \]

\[ y(k) = Cx(k) + Du(k) \]

The solution is (equivalent of Lagrange’s formula)

\[ x(1) = Ax_0 + Bu(0) \]

\[ x(2) = Ax(1) + Bu(1) = A^2x_0 + Bu(1) + ABu(0) \]

while for the output we have

\[ y(0) = Cx_0 + Du(0) \]

\[ y(1) = CAx_0 + CBu(0) + Du(1) \]
Discrete time linear systems – cont’d

In general we have

\[ x(k) = A^k x_0 + \sum_{i=0}^{k-1} A^{k-i-1} B u(i) \]

\[ y(k) = C A^k x_0 + C \sum_{i=0}^{k-1} A^{k-i-1} B u(i) + D u(k) \]

where we can again distinguish between free and force state and output motions!

As both the two motions are linear with respect to the initial state and the input, respectively, the superposition principle still holds.
Discrete time linear systems – cont’d

For a constant input, equilibria of a discrete time linear system correspond to

\[ \bar{x} = x(k+1) = x(k), \forall k \]

\[ \bar{x} = A\bar{x} + B\bar{u} \]

which can be computed as

\[ \bar{x} = (I - A)^{-1} B\bar{u} \]

\[ \bar{y} = \left[ C (I - A)^{-1} B + D \right] \bar{u} \]

provided that the matrix to be inverted is not singular.
Discrete time linear systems – cont’d

As for their continuous time counterpart, change of variables can be performed to show interesting properties of a given system.

In particular, consider the free motion $x_{free}(k) = A^k x_0$, through a generic change of variables we obtain

$$A^k = \left(T^{-1} \hat{A} T\right)^k = T^{-1} \hat{A}^k T$$

Assuming the matrix to be diagonalizable, we get

$$A^k = \left(T^{-1} \Lambda T\right)^k = T^{-1} \Lambda^k T = T^{-1} \begin{bmatrix} \lambda_1^k & 0 & \ldots & \ldots \\ 0 & \lambda_2^k & 0 & \ldots \\ 0 & \ldots & \ddots & 0 \\ 0 & \ldots & \ldots & \lambda_n^k \end{bmatrix} T$$
Discrete time linear systems – cont’d

Consider a first order discrete time autonomous linear system

\[ x(k + 1) = ax(k) \quad x(0) = x_0 \]

The corresponding free state motion is as follows

\[ x(k) = a^k x_0 \]

Depending on the value of \( a \), we can then distinguish at least four different situations

\[ a > 1 \quad 0 < a < 1 \quad -1 < a < 0 \quad a < -1 \]
Discrete time linear systems – cont’d

\[ a > 1 \]

\[ 0 < a < 1 \]

\[ -1 < a < 0 \]

\[ a < -1 \]
Discrete time linear systems – cont’d

We obtained a diagonal (complex) matrix which elements are power of the eigenvalues of $A$.

We can then formulate a result for stability of a discrete time linear system. Given a discrete time LTI system

**General result:**

- $\forall i, |\lambda_i| < 1$ the motion (system) is asymptotically stable
- $\exists i, |\lambda_i| > 1$ or $\exists i, |\lambda_i| = 1$ and those with unitary magnitude have index greater than one, the motion (system) is unstable
- $\forall i, |\lambda_i| \leq 1, \exists i, |\lambda_i| = 1$ and those with unitary magnitude have index one, the motion (system) is stable
Jury’s and Routh’s criteria

As for continuous time systems, stability of discrete time linear systems can be addressed without explicitly computing the eigenvalues, but analysing the characteristic polynomial (of which eigenvalues are roots).

$$\det (\lambda I - A) = \alpha_n \lambda^n + \alpha_{n-1} \lambda^{n-1} + \alpha_1 \lambda + \alpha_0$$

For discrete time linear systems there exists a criterion similar to the Routh’s one which is called Jury’s criterion.

The Jury’s criterion is the counterpart of the Routh’s criterion for discrete time systems.
Jury’s and Routh’s criteria – cont’d

**Definition:** the Jury table has $2n+1$ rows, the first two rows are defined based on the coefficient of the characteristic polynomials

\[
\begin{array}{cccc}
\alpha_0 & \alpha_1 & \ldots & \alpha_n \\
\alpha_n & \alpha_{n-1} & \ldots & \alpha_0 \\
\alpha_0 - \beta_1 \alpha_n & \alpha_1 - \beta_1 \alpha_{n-1} & \ldots & 0 \\
\alpha_{n-1} - \beta_1 \alpha_1 & \ldots & \alpha_0 - \beta_1 \alpha_n & 0 \\
\vdots & \vdots & \vdots & \vdots \\
\end{array}
\]

Coefficients (reverse order in the first row)

Difference between first and scaled second row, and same in reverse order with

\[
\beta_1 = \frac{\alpha_n}{\alpha_0}
\]

Then iterate with…

\[
\beta_2 = \frac{\alpha_{n-1} - \beta_1 \alpha_1}{\alpha_0 - \beta_1 \alpha_n}
\]
Jury’s and Routh’s criteria – cont’d

**General result:** a discrete time LTI system is asymptotically stable if and only if the elements of the first column have all the same sign (positive vs. negative).

**Notice:** the table is called undefined if some element of the first column is zero. In this case the elements does not have the same sign!

With respect to the Routh’s criterion, this procedure is tremendously tedious (and potentially confused with the Routh’s procedure).

We wonder whether we can apply the Routh criterion instead, after some manipulation fo the characteristic polynomial.
Jury’s and Routh’s criteria – cont’d

We are then looking of a mapping between eigenvalues such that

- eigenvalues in the unit circle, they are mapped somewhere in the left half-plane
- eigenvalues outside the unit circle are mapped somewhere in the right half-plane
- eigenvalues on the circle are mapped somewhere on the immaginary axis

This mapping (function) exists and it is called bilinear transformation between polynomials.
Jury’s and Routh’s criteria – cont’d

Such a function exists and is called **bilinear transformation**:

\[ \hat{\lambda} = b(\lambda) = \frac{1 + \lambda}{1 - \lambda} \]
Jury’s and Routh’s criteria – cont’d

Therefore, in order to address the stability of a discrete time system we:

• compute the characteristic polynomial
  \[ \alpha_n \hat{\lambda}^n + \cdots + \alpha_1 \hat{\lambda} + \alpha_0 \]

• apply the bilinear transformation
  \[ \alpha_n \left(\frac{1 + \lambda}{1 - \lambda}\right)^n + \cdots + \alpha_1 \left(\frac{1 + \lambda}{1 - \lambda}\right) + \alpha_0 \]

• perform some algebraic manipulation to obtain the form
  \[ \beta_n \lambda^n + \cdots + \beta_1 \lambda + \beta_0 \]

• apply the Routh’s criterion
Jury’s and Routh’s criteria – cont’d

Example: consider the discrete time linear system having

\[
A = \begin{bmatrix}
0.3 & 5.1 & 1.7 \\
0 & -0.1 & 3 \\
0 & 0 & 0
\end{bmatrix}
\]

The corresponding characteristic equation is the following on

\[
\hat{\lambda} \left( \hat{\lambda} - 0.3 \right) \left( \hat{\lambda} + 0.1 \right) = \hat{\lambda}^3 - 0.2\hat{\lambda}^2 - 0.03\hat{\lambda} = 0
\]

which clearly has roots within the unit circle. Let’s apply the bilinear transformation

\[
\left( \frac{1+\lambda}{1-\lambda} \right)^3 - 0.2 \left( \frac{1+\lambda}{1-\lambda} \right)^2 - 0.03 \left( \frac{1+\lambda}{1-\lambda} \right) = 0
\]
Jury’s and Routh’s criteria – cont’d

The corresponding polynomial is the following one

\[0.77\lambda^3 + 2.83\lambda^2 + 3.23\lambda + 1.17 = 0\]

The resulting Routh’s table is as follows

<table>
<thead>
<tr>
<th></th>
<th>0.77</th>
<th>3.23</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.83</td>
<td>1.17</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2.917</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.17</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Which guarantees the asymptotic stability of the original discrete time system.
Properties

Given an asymptotically stable discrete time system

\[ x(k + 1) = Ax(k) + Bu(k) \quad x(k_0) = x_0 \]

\[ y(k) = Cx(k) + Du(k) \]

- for a given constraint input the equilibrium state as well as the corresponding output are unique

\[ \bar{x} = (I - A)^{-1} B \bar{u} \quad \bar{y} = \left[ C (I - A)^{-1} B + D \right] \bar{u} \]

- as the free motion tends to zero, the overall motion tends to the force one

Moreover
- stability is a structural property
- a bounded input produces a bounded output (BIBO stability)
Stability and linearization

Consider a generic nonlinear system

\[ x^+ = f(x, u) \quad y = h(x, u) \]

assume the input is constant \( u(k) = \bar{u} \) and that there exists an equilibrium state \( \bar{x} \), i.e.

\[ \bar{x} = f(\bar{x}, \bar{u}) \]

We can then approximate the system behaviour in the neighborhood of the equilibrium with a linear system

\[ \delta x^+ = \left. \frac{\partial f}{\partial x} \right|_{\bar{x}, \bar{u}} \delta x + \left. \frac{\partial f}{\partial u} \right|_{\bar{x}, \bar{u}} \delta u \quad \delta y = \left. \frac{\partial h}{\partial x} \right|_{\bar{x}, \bar{u}} \delta x + \left. \frac{\partial h}{\partial u} \right|_{\bar{x}, \bar{u}} \delta u \]
Stability and linearization – cont’d

The system is now linear

\[ \delta x^+ = A \delta x + B \delta u \quad \delta y = C \delta x + D \delta u \]

Where

\[
A = \left. \frac{\partial f}{\partial x} \right|_{\bar{x}, \bar{u}} \quad B = \left. \frac{\partial f}{\partial u} \right|_{\bar{x}, \bar{u}} \quad C = \left. \frac{\partial h}{\partial x} \right|_{\bar{x}, \bar{u}} \quad D = \left. \frac{\partial h}{\partial u} \right|_{\bar{x}, \bar{u}}
\]

Similarly to continuous time systems, we are now interested in understanding whether the linear(ized) system reflects the stability property of the original (nonlinear).
Linearization and stability

As we have done for continuous time system, we can introduce the following

**General results:** given the equilibrium of a nonlinear discrete time system
- if the corresponding linearized system has all eigenvalues strictly inside the unit circle, then it is asymptotically stable
- if the corresponding linearized system has at least one eigenvalue outside the unit circle, then it is unstable

**Notice:** nothing can be said in other situations. Moreover, the stability property of a nonlinear system is strictly related to a particular equilibrium, rather than to the whole system.
Z-transform

Similarly to what we have done for continuous time systems, we now want to characterize the behaviour of a discrete time linear system in the frequency domain.

Let’s focus on a generic discrete time real signal

\[ v(k) = \{v(0), v(1), v(2), \ldots \} \]

we define the **Z-transform** as follows

\[ V(z) = \sum_{k=0}^{+\infty} v(k) z^{-k}, \ z \in \mathbb{C} \]
Z-transform – cont’d

**Example:** consider the following signal (discrete time exponential)

\[ v(k) = a^k, \quad a \in \mathbb{R} \]

its Z-transform is the following one

\[
V(z) = \sum_{k=0}^{+\infty} a^k z^{-k} = \sum_{k=0}^{+\infty} (az^{-1})^k = \frac{1}{1 - az^{-1}} = \frac{z}{z - a}
\]

The special case of the step function \( a = 1 \) gives us \( V(z) = \frac{z}{z - 1} \)

Before using the Z-transform, let’s have a brief overview of its properties.
Z-transform – cont’d

Properties of the Z-transform:

- Linearity $\mathcal{Z} [\alpha v_1 (k) + \beta v_2 (k)] = \alpha V_1 (z) + \beta V_2 (z)$
- Time shifts
  - ahead $\mathcal{Z} [v (k + 1)] = z (V (z) - v (0))$
  - behind $\mathcal{Z} [v (k - 1)] = z^{-1} V (z)$
- Derivative in the frequency domain $\mathcal{Z} [kv (k)] = -z \frac{dV (z)}{dz}$
- Convolution $\mathcal{Z} \left[ \sum_{h=0}^{k} v (k - h) u (h) \right] = V (z) U (z)$
Z-transform – cont’d

• Initial value $v(0) = \lim_{z \to +\infty} V(z)$

• Final value (if applicable) $\lim_{k \to +\infty} v(k) = \lim_{z \to 1} (z - 1) V(z)$

*Example*: compute the $Z$-transform of the ramp function

$$v(k) = \text{ramp}(k) = k, k \geq 0$$

We just have to notice that

$$v(k) = k\text{step}(k)$$

Then we obtain $V(z) = -z \frac{d}{dz} \frac{z}{z - 1} = \frac{z}{(z - 1)^2}$
Riemann-Fourier

As for the Laplace’s transform, the Z-transform has its anti transformation, which, in this case, is called Riemann-Fourier transform.

\[ v(k) = \frac{1}{2\pi j} \int_c V(z) z^{k-1} \, dz \]

where \( c \) denotes an anti-clockwise circular path around the origin.

Beside its definition, in case \( V(z) \) is rational and strictly proper, its Riemann-Fourier transform can be computed through the Heaviside’s method or another method (that we will see during seminars).
Transfer function

Given a generic discrete time linear system

\[ x^{+} = Ax + Bu \quad y = Cx + Du \quad x(0) = x_0 \]

as we have done for continuous time systems, we are interested in deriving a relationship between the Z-transform of the input and the Z-transform of the output, which we will call transfer function.

In other words we are interested in characterizing the relationship between

\[ U(z) = Z[u(k)] \quad Y(z) = Z[y(k)] \]

In order to do so, let’s apply the Z-transform to the equations of the system.
Transfer function – cont’d

We obtain

\[ \mathcal{Z} \left( x(k+1) \right) = z(XX(z) - x_0) = AX(z) + BU(z) \]

or equivalently

\[(zI - A)X(z) = BU(z) + zx_0\]

\[X(z) = (zI - A)^{-1} BU(z) + (zI - A)^{-1} zx_0\]

as for the output equation we get

\[Y(z) = CX(z) + DU(z)\]

and finally obtain

\[Y(z) = \left[ C(zI - A)^{-1} B + D \right] U(z) + C(zI - A)^{-1} zx_0\]
Transfer function – cont’d

We then define the transfer function of a discrete time system the following quantity

\[ G(z) = C(zI - A)^{-1}B + D \]

Which represents, for a null initial state, the relationship between the Z-transform of the input and the Z-transform of the output, i.e.

\[ Y(z) = G(z)U(z) \]

Similarly to the continuous time SISO case, the transfer function is the ratio between polynomials, i.e.

\[ G(z) = \frac{\alpha_m z^m + \alpha_{m-1} z^{m-1} + \cdots + \alpha_1 z + \alpha_0}{\beta_n z^n + \beta_{n-1} z^{n-1} + \cdots + \beta_1 z + \beta_0} \]
Transfer function – cont’d

As for the continuous time case, the transfer function has some interesting properties.

**Properties and facts:**

- numerator and denominator are polynomials with real coefficients;
- the degree of the denominator $n$ is always greater or equal to the degree of the numerator $m$ (their difference is called **relative degree**);
- the root of those polynomials (real or complex conjugate) are called **zeros** (numerator) and **poles** (denominator)
- we call type the number of poles in $z = 1$
- when $D = 0$, the relative degree is at least one (the system is called strictly proper), otherwise it is zero (proper);
- poles are also eigenvalues of matrix $A$
Frequency response

As we have done with continuous time linear system, we want to study the behaviour of a discrete time LTI SISO system in response to harmonic oscillations.

\[ U(z) \overset{G(z)}{\rightarrow} Y(z) = G(z)U(z) \]

where \( u(k) = A \sin(\theta k + \psi), k \geq 0 \).

**Frequency response theorem:** given an asymptotically stable discrete time system having transfer function \( G(z) \) the output, subject to the given input, is such that

\[
\lim_{k \to +\infty} y_R(k) - y(k) = 0
\]

where

\[
y_R(k) = |G(e^{j\theta})| A \sin(\theta k + \psi + \angle G(e^{j\theta}))
\]
Frequency response – cont’d

The complex function $G(e^{j\theta}) \in \mathbb{C}$, $\theta \in [0, \pi]$ is then called frequency response of the system.

The frequency response is obtain by evaluating the transfer function on the upper half-circumference of unit radius, centered in the origin.

As for continuous time system, in the discrete time case the frequency response has several useful representations, such as the polar plot and the Bode diagrams.
Stability of feedback systems

For completeness we here list the available results for closed-loop stability assessment of discrete time system.

Consider the following closed-loop system:

\[ L(z) = \frac{n(z)}{d(z)} \]

then the stability of the closed-loop system can be checked
- by computing closed-loop poles, i.e. solutions of \( n(z) + d(z) = 0 \)
- with the Routh’s or Jury’s criteria
- with the Nyquist criterion
- with root locus
Stability of feedback systems – cont’d

The last two methods deserve a more detailed explanation.

**Nyquist criterion**

as usual N is the number of encirclements around -1, P is the number of poles with magnitude greater than 1, the Nyquist path is around the unit circle centered in the origin with the same “trick” of circumventing poles on with unit magnitude (on the circle).

**Root locus**

the drawing rules are the same, but the stability limit is now represented by the unit circle centered in the origin (and no longer by the imaginary axis).
Digital control systems

We have introduced two methodologies (root locus and the one based on Bode diagrams) to design a controller in terms of a continuous time transfer function for a generic LTI continuous time system, described in terms of a transfer function.

Time ago, the resulting controller, typically in one of the forms we have seen (lead, lag, PID, etc.) was implemented through an analog electronic circuit.
Digital control systems – cont’d

“Modern” control systems are implemented through digital technologies such as computers (e.g. PC, ePC, PLC, DSP, μC).
Digital control systems – cont’d

As already introduced, there are two main differences between the “traditional” analog implementation of a controller and what can be done through a “modern” digital technology:

- **time** is a discrete variable, typically multiple of the clock tick;
- **numbers** are quantized, typically through fixed-point or floating-point arithmetics.

Therefore, in modern control systems, two words coexists:

- the “analog world”, the physical one
- the “digital world”, the one of computers

Next we will try to understand *how to merge those two worlds* and how they can exchange information.
Digital control systems – cont’d

The first component we introduce is used to “convert” signals from continuous time to discrete time: it is called sampler.

\[ e^*(k) = e(kT_s) \quad k \in \mathbb{N} \]

It is represented as a switch which automatically triggers every \( T_s \) seconds. This parameters is called sampling time. Alternatively we can also define

\[
\begin{align*}
    f_s &= \frac{1}{T_s} \\
    \omega_s &= 2\pi f_s = \frac{2\pi}{T_s}
\end{align*}
\]
Digital control systems – cont’d

The second component is called quantizer and represent the inherent discretization of number in any digital unit.

As we have introduced the quantizer, we can define a “block” representing the conversion from the “analog world” to the “digital” one: the so called analog to digital (A2D) converter.
Digital control systems – cont’d

Another component we need to introduce is adopted to “convert” signals from discrete to continuous time: it is called (zero order) holder (ZOH).

\[ u^* (k) \xrightarrow{ZOH} u (t) \]

\[ u (t) = u^* (k), \quad t \in [kT_s, (k + 1) T_s) \]

It maintains its output constant for the whole sampling interval.

There are slightly different holder, for example the first order holder (FOH) maintains the slope of its output constant for the whole sampling time.
Digital control systems – cont’d

Before characterizing each of the components we have introduced, let’s try to complete the puzzle of our hybrid (discrete/continuous time) control system.
Sampled signals

Starting from the sampler, let’s analyse the behaviour of each component.
Sampled signals – cont’d

When a signal is sampled, it is necessary to select the sampling frequency so that no information about the signal is lost.

\[ v_1(t) = \sin(0.6\pi t) \quad v_2(t) = -\sin(-1.4\pi t) \quad v_1^*(k) = v_2^*(k), T_s = 1 \]
Sampled signals – cont’d

To do so, we are interested in comparing the Fourier transform of the continuous time system, and the (discrete) Fourier transform of the sampled one.

Therefore, letting \( V(j\omega) = \mathcal{F}[v(t)] \) and \( V^*(e^{j\theta}) = \mathcal{F}^*[v(kT_s)] \)

**Theorem:** assuming \( v(t) \) to be continuous at each sampled time, then

\[
V^*(e^{j\omega T_s}) = \frac{1}{T_s} \sum_{h=-\infty}^{h=+\infty} V\left(j \left(\omega + h \frac{2\pi}{T_s}\right)\right)
\]

Notice that \( V^*(e^{j\omega T_s}) \) is symmetric and periodic of period \( \omega_s \).
Sampled signals – cont’d

Example: for simplicity, assume the continuous time signal $v(t)$ has a real valued Fourier transform $V(j\omega)$ like the one shown below

As we have seen from the last theorem, the Fourier transform of the sampled signal is obtained by summing up the periodic repetition of the original transfer function
Sampled signals – cont’d

What we obtain is then the following

$$|V^*(j\omega)|$$

Therefore, within the bandwidth $\omega = [0, \omega_s/2]$ all the frequency components of the original are preserved and the original signal can be reconstructed by filtering out higher frequencies (we will see this).
Sampled signals – cont’d

On the other hand, as we have already seen, there are situations in which this is not possible.

Back to the Example
Sampled signals – cont’d

Assume we are able to filter out all the components outside the bandwidth $\omega = [0, \omega_s/2]$. In the two examples we obtain:

\[ |V^*(j\omega)| \]

\[ \omega_s/2 \]

\[ \omega \]

\[ |V^*(j\omega)| \]

\[ \omega_s/2 \qquad \omega_s \]
Sampled signals – cont’d

As we might guess, aliasing is caused by the combination of the sampling frequency and the bandwidth of the signal.

**Shannon’s theorem:** given a signal \( v(t) \) with limited bandwidth \( \omega \in [0, \omega_{max}] \). If

\[
\omega_{max} < \omega_N = \frac{\omega_s}{2} = \frac{\pi}{T_s}
\]

and the signal is continuous at each sampling instant, then, it is possible to be exactly reconstructed from its samples

\[
v^*(k) = v(kT_s), k \in \mathbb{N}
\]

The limiting bandwidth \( \omega_N \) is called **Nyquist frequency**.
Sampled signals – cont’d

In the second example we are not able to reconstruct the original signal through low-pass filtering. This phenomenon, is called **aliasing**. The aliasing effect is also known as **wagon-wheel effect** (from western movies of mid 1950’s). Shannon’s theorem was actually published at the end of 1940’s.

https://www.youtube.com/watch?v=6XwgbHjRo30
Sampled signals – cont’d

Moreover, in order to relax the assumption of a band limited signal, it is usually necessary to low-pass filter (called anti-aliasing filter) the analog signal before sampling it.

Therefore, the complete model of an A2D converter is as follows
Sampled signals – cont’d

When it comes to reconstruct a continuous time behaviour from a sampled signal, Shannon suggested the following ideal formula:

\[ v(t) = \sum_{k=-\infty}^{+\infty} v^*(k) \frac{\sin(\omega_N t - k\pi)}{\omega_N t - k\pi} = \sum_{k=-\infty}^{+\infty} v^*(k) \text{sinc}(\omega_N t - k\pi) \]

This formula is ideal as it involves an infinite number of samples. Moreover it is not practically applicable “online” as it requires also future samples.

Practical implementations, as we have already seen, involves a holder. The main important one is the zero order holder (ZOH) which simply maintains the value constant for the whole sampling interval.
Sampled signals – cont’d

In the following a signal is reported and compared to its sampled version and its ZOH reconstruction. Notice the delay between the original signal and the piece-wise constant one.
Sampled signals – cont’d

Where does this delay come from? Let’s analyse the behaviour of the ZOH element. In order to address this problem we consider the Z-transform of the input of a ZOH and try to relate it to its output.

The input of a ZOH can be seen as a train of impulses. If we consider just the first impulse, we have:

\[ v^* (k) = v^* (0) \text{imp}(k) \]

If we feed our ZOH with such an input we obtain

\[ v(t) = v^* (0) [\text{step}(t) - \text{step}(t - T_s)] \]

\[ \mathcal{L} [v(t)] = v^* (0) \frac{1 - e^{-sT_s}}{s} \]
Sampled signals – cont’d

Hence, the quantity $H_0(s) = \frac{1 - e^{-sT_s}}{s}$ represents the relationship between the Z-transform of the input of a ZOH

$$v^*(k) = v^*(0) \text{imp}(k) \quad \mathcal{Z}[v^*(0) \text{imp}(k)] = v^*(0)$$

and the Laplace transform of its output

$$v(t) = v^*(0) [\text{step}(t) - \text{step}(t - T_s)] \quad \mathcal{L}[v(t)] = v^*(0) \frac{1 - e^{-sT_s}}{s}$$

In the more general case of a train of impulses

$$v^*(k) = v^*(0) \text{imp}(k) + v^*(1) \text{imp}(k - 1) + v^*(2) \text{imp}(k - 2) + \cdots$$

we obtain the following relationship

$$V(s) = H_0(s) V^*(e^{sT_s})$$
Sampled signals – cont’d

Finally, considering the following scheme

\[
\begin{align*}
T_s & \\
v_0(t) & \rightarrow v^*(k) \rightarrow \text{ZOH} \rightarrow v(t)
\end{align*}
\]

if we use one of the previous result, corresponding to an impulse we have

\[
V(j\omega) = H_0(j\omega) V^*(e^{j\omega T_s}) = H_0(j\omega) \frac{1}{T_s} V_0(j\omega)
\]

Therefore, with some “algebraic trick”, we obtain

\[
H_0(j\omega) = \frac{1 - e^{-j\omega T_s}}{j\omega T_s} = e^{-j\omega T_s} e^{\frac{T_s}{2} - e^{-\frac{j\omega T_s}{2}}} = e^{-j\omega T_s} \sin\left(\frac{T_s}{\omega} \right)
\]

Euler formula
Sampled signals – cont’d

The exact frequency response is then the following one

\[
\begin{align*}
\left| \frac{H_0(j\omega)}{T_s} \right| &= \sin \left( \frac{\omega T_s}{2} \right) \\
\frac{\omega T_s}{2} &\quad \left\langle \frac{H_0(j\omega)}{T_s} \right\rangle = -\frac{\omega T_s}{2}
\end{align*}
\]

For small frequencies, however, we can introduce the following (widely adopted) approximation

\[
\begin{align*}
\left| \frac{H_0(j\omega)}{T_s} \right| &\approx 1 \\
\left\langle \frac{H_0(j\omega)}{T_s} \right\rangle &= -\frac{\omega T_s}{2}
\end{align*}
\]

which approximately corresponds to frequency response of a delay

\[
\begin{align*}
\frac{H_0(j\omega)}{T_s} &\approx e^{-j\frac{\omega T_s}{2}}
\end{align*}
\]
Discrete time realization

So far we have characterized the behaviour of a digital control system and analysed the two main components: sampler and holder.

Now, we want to address the following questions:

1. what is the suitable sampling time for the discrete time implementation of a given controller?
2. given a continuous time controller, in terms e.g. of a transfer function, how can we transform it into a discrete time one?
Discrete time realization – cont’d

As for the selection of a suitable sampling time, we should account for the following consideration:

- the controller must be able to compute and update the control action within one sampling interval
  - the faster, the better
- the sampling interval must be small enough to ensure that any relevant information is lost (i.e. we have to obey to the Shannon’s theorem)
- relevant information contained e.g. in the controlled variable are in the control-loop bandwidth, i.e. \( \omega \in [0, \omega_c] \), therefore at least we have

\[
\omega_s > 2\omega_c \quad \omega_N > \omega_c \quad \frac{\pi}{T_s} > \omega_c
\]
Discrete time realization – cont’d

- moreover, we have seen that the ZOH+sampling can be seen as an equivalent delay

\[
\frac{H_0(j\omega)}{T_s} \approx e^{-j\omega \frac{T_s}{2}}
\]

which might cause the designed phase margin to decrease

\[
\psi_m = \psi_m^{\text{desgin}} - \omega_c \frac{180 T_s}{\pi} \frac{1}{2}
\]

As we should have understood that the sampling time and the crossover frequency must be somehow related, let’s assume

\[
\alpha \omega_N = \alpha \frac{\pi}{T_s} = \omega_c, \alpha < 1
\]
Discrete time realization – cont’d

Decrement to the phase margin can be then computed as follows

\[ \psi_m = \psi_m^{design} - \omega_c \frac{180 T_s}{\pi} \frac{1}{2} = \psi_m^{design} - \alpha \frac{\pi}{T_s} \frac{180 T_s}{\pi} \frac{1}{2} = \psi_m^{design} - 90 \alpha \]

A rule of thumb \( \alpha \approx 0.1, \Delta \psi_m \approx 9^\circ \) practically means to select the Nyquist frequency one decade after the design crossover frequency.

\[ \omega_N \approx 10 \omega_c \]

Remind that an anti-aliasing filter might further decrease the phase margin.
Discrete time realization – cont’d

Once we have selected the sampling time, we want to traduce our controller, designed with continuous time tools, into an algorithm to be implemented on a computer, or, in other terms into a discrete time transfer function.

\[ R(s) \xrightarrow{T_s} R^*(z) \]

Ideally, since \( z^{-1} \) is a unit delay in discrete-time, it would be enough to set \( z^{-1} \rightarrow e^{-sT_s} \)

This might imply to substitute

\[ s = \frac{1}{T_s} \log z \quad \Rightarrow \quad R^*(z) = R\left(\frac{1}{T_s} \log z\right) \]
Discrete time realization – cont’d

The ideal solution is not practical as it would transform a rational (continuous time) transfer into a non rational discrete time one.

Let’s focus of the meaning of the Laplace operator. We know that

\[ \mathcal{L} [\dot{x} (t)] = sX (s) \]

Hence we can transform the Laplace operator (derivator) into the rate change, i.e.

\[ \dot{x} (t) = \lim_{T_s \to 0} \frac{x (t + T_s) - x (t)}{T_s} \approx \frac{x (t + T_s) - x (t)}{T_s} \]

If we take the Z-transform of the last term we get

\[ z \left[ \frac{x (t + T_s) - x (t)}{T_s} \right] = \frac{z - 1}{T_s} X (z) \]
Discrete time realization – cont’d

Therefore, in order to achieve our goal of transforming a continuous time transfer function into a discrete time one, we can perform the following substitution

\[ s \rightarrow \frac{z - 1}{T_s} \quad R(s) \rightarrow R^*(z) = R\left(\frac{z - 1}{T_s}\right) \]

We have now a simple rule to implement our digital controller.

Let’s see how it works in practice…
Discrete time realization – cont’d

*Example:* assume the following system

\[ G(s) = \frac{10}{1 + s} \]

has to be controlled and that we have designed the following lead+integral controller

\[ R(s) = \frac{1 + s}{s \cdot 0.01s + 1} \quad \omega_c = 10, \psi_m = 84^\circ \]

We then have selected the following sampling time

\[ T_s = 0.03, \omega_N \approx 10\omega_c \quad \Delta\psi_m \approx 9^\circ \]

\[ R^*(z) = R \left( \frac{z - 1}{T_s} \right) = \frac{3}{z - 1} \cdot \frac{z - 0.97}{z - 2} \]

something strange happened!
Discrete time realization – cont’d

In the last example, due to the discretization process, starting from a marginally stable continuous time controller, we obtained an unstable discrete time one.

It means that our mapping between continuous time and discrete time

\[ s \rightarrow \frac{z - 1}{T_s} \]

can transform stable (continuous time) poles into unstable (discrete time) ones, i.e.

\[ s \rightarrow \frac{z - 1}{T_s} \]

\[ \Re \left[ s_p \right] < 0 \quad \frac{T_s}{\rightarrow} \quad \left| z_p \right| > 1 \]
Discrete time realization – cont’d

Consider $s + \alpha + j\beta$, $\alpha > 0$ and apply the transformation $s \rightarrow \frac{z - 1}{T_s}$

\[
\frac{z - 1 + T_s\alpha + j\beta T_s}{T_s} = \frac{z = 1 - T_s\alpha - j\beta T_s}{T_s}
\]

The region in which we preserve stability is then $\sqrt{(1 - T_s\alpha)^2 + \beta^2 T_s^2} < 1$ which is a circle centered in $(-1/Ts,0)$ and radius $1/Ts$. 
Discrete time realization – cont’d

Sometimes, our first guess concerning the need of a discretization method might fail. We should look for something more robust, at least capable of preserving stability regardless the sampling time.

Consider one of the property of the Laplace transform

\[
\mathcal{L} \left[ \int_0^t f(\tau) \, d\tau \right] = \frac{1}{s} F(s)
\]

What we can do is to analyse different formulas to compute integrals and then compute its reciprocal.

\[
\int_0^{T_s} f(\tau) \, d\tau = T_s (1 - \alpha) f(0) + T_s \alpha f(T_s), 0 \leq \alpha \leq 1
\]
Discrete time realization – cont’d

One of the mostly adopted generic integration formula is the following one

$$\int_{0}^{T_s} f(\tau)\,d\tau = T_s \left(1 - \alpha\right) f(0) + T_s \alpha f(T_s), \quad 0 \leq \alpha \leq 1$$
Discrete time realization – cont’d

One of the mostly adopted generic integration formula is the following one

\[ y(T_s) - y(0) = \int_0^{T_s} f(\tau) \, d\tau \approx T_s (1 - \alpha) f(0) + T_s \alpha f(T_s), \quad 0 \leq \alpha \leq 1 \]

which generally corresponds, through the Z-transform, to the following substitution

\[ Y(z)(z - 1) = T_s (1 - \alpha) F(z) + zT_s \alpha F(z) \]

\[ Y(s) = \frac{F(s)}{s} \]

\[ s \rightarrow \frac{z - 1}{T_s (1 - \alpha) + zT_s \alpha} \]
Discrete time realization – cont’d

Considering the three cases we have introduced a while ago, we have

\[ \alpha = 0, s \to \frac{z - 1}{T_s} \]  \hspace{1cm} \text{Forward Euler}

\[ \alpha = 1, s \to \frac{z - 1}{T_s z} \]  \hspace{1cm} \text{Backward Euler}

\[ \alpha = 0.5, s \to \frac{2}{T_s} \frac{z - 1}{z + 1} \]  \hspace{1cm} \text{Tustin (or bilinear method)}
Discrete time realization – cont’d

Another possibility to derive a discrete-time version of the controller is to apply the Lagrange formula.

Consider the transfer function of the controller

\[ R(s) = C_R (sI - A_R)^{-1} B_R + D_R \]

then

\[ x_R(k+1) = \exp(A_R T_s) x_R(k) + \int_0^{T_s} \exp(A_R (T_s - \tau)) B_R e(\tau) d\tau = \]

\[ = H_R x_R(k) + \left[ \int_0^{T_s} \exp(A_R (T_s - \tau)) d\tau \right] B_R e(k) \]

and finally

\[ u(k) = C_R x_R(k) + D_R e(k) \]
Discrete time realization – cont’d

*Example*: back to our example

\[ R(s) = \frac{1}{s} \frac{1 + s}{0.01s + 1} \]

Let’s use the **Tustin method** instead

\[ s \rightarrow \frac{2}{T_s} \frac{z - 1}{z + 1} \]

We finally obtain

\[ R^*(z) = \frac{0.609}{z - 1} \frac{(z + 1)(z - 0.97)}{(z + 0.2)} \]
Discrete time realization – cont’d

After discretization, we might have obtained the following form

\[ R^*(z) = \frac{\alpha_m z^{m-n} + \alpha_{m-1} z^{m-n-1} + \cdots + \alpha_1 z^{1-n} + \alpha_0 z^{-n}}{\beta_n + \beta_{n-1} z^{-1} + \cdots + \beta_1 z^{1-n} + \beta_0 z^{-n}} \]

which can be rewritten as

\[
(\beta_n + \beta_{n-1} z^{-1} + \cdots + \beta_1 z^{1-n} + \beta_0 z^{-n}) U(z) = \\
(\alpha_m z^{m-n} + \alpha_{m-1} z^{m-n-1} + \cdots + \alpha_1 z^{1-n} + \alpha_0 z^{-n}) E(z)
\]

By computing the antitrasformation, we obtain

\[
\beta_n u(k) + \beta_{n-1} u(k-1) + \cdots + \beta_1 u(k-n+1) + \beta_0 u(k-n) = \\
\alpha_m y(k-n+m) + \alpha_{m-1} e(k-n+m+1) + \cdots + \alpha_1 e(k-n+1) + \alpha_0 e(k-n)
\]
Discrete time realization – cont’d

We can then compute the control input based on:

- the error and its previous samples
- previous samples of the control input itself

\[
u(k) = -\frac{\beta_{n-1}}{\beta_n}u(k-1) - \cdots - \frac{\beta_1}{\beta_n}u(k-n+1) - \frac{\beta_0}{\beta_n}u(k-n) + \frac{\alpha_m}{\beta_n}e(k-n+m) + \frac{\alpha_{m-1}}{\beta_n}e(k-n+m+1) + \cdots + \frac{\alpha_0}{\beta_n}e(k-n)\]

and iterate this simple algorithm at each sampling time.
Discrete time realization – cont’d

*Example:* in our last example we had

\[ R^*(z) = \frac{0.609 \times (z + 1)(z - 0.97)}{z - 1} = \frac{0.609z^2 + 0.01827z - 0.5907}{z^2 - 0.8z - 0.2} \]

Let’s try to handle all the implementation part until some executable algorithm (code).

The given transfer function can be better written in terms of negative powers of z as follows

\[ R^*(z) = \frac{0.609 + 0.01827z^{-1} - 0.5907z^{-2}}{1 - 0.8z^{-1} - 0.2z^{-2}} \]
Discrete time realization – cont’d

From this form, it is easy to compute the input-output relationship

\[(1 - 0.8z^{-1} - 0.2z^{-2}) U(z) = (0.609 + 0.01827z^{-1} - 0.5907z^{-2}) E(z)\]

which can be easily transformed back in time domain

\[u(k) - 0.8u(k - 1) - 0.2u(k - 2) = 0.609e(k) + 0.01827e(k - 1) - 0.5907e(k - 2)\]

or equivalently into the following “algorithm”

\[u(k) = 0.8u(k - 1) + 0.2u(k - 2)\]

\[0.609e(k) + 0.01827e(k - 1) - 0.5907e(k - 2)\]

which can be easily implemented.
Discrete time realization – cont’d

We can then update the control law with the following very simple code (ANSI/C syntax):

```c
double u_pre = 0, u_pre_pre = 0;
double e_pre = 0, e_pre_pre = 0;

double mycontrol(double e) {
    u = .8*u_pre + .2*u_pre_pre + .609*e + .01827*e_pre - .5907*e_pre_pre;
    u_pre_pre = u_pre;
    u_pre = u;
    e_pre_pre = e_pre;
    e_pre = e;
    return u;
}
```

\[ u(k) = 0.8u(k-1) + 0.2u(k-2) \\
       0.609e(k) + 0.01827e(k-1) - 0.5907e(k-2) \]
Discrete time realization – cont’d

We have the following overall control loop:

```c
double u_pre = 0, u_pre_pre = 0;
double e_pre = 0, e_pre_pre = 0;

double mycontrol(double e) {
    u = 0.8*u_pre + ...;
    u_pre_pre = u_pre;
    u_pre = u;
    e_pre_pre = e_pre;
    e_pre = e;
    return u;
}
```

Let’s verify whether we have done everything correctly…
Discrete time realization – cont’d

Output step response: continuous time vs discrete time.
Discrete time realization – cont’d

Control input step response (continuous time vs discrete time), notice the \textbf{piece-wise constant behaviour} due to the ZOH.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure}
\caption{Digital control law vs analog control law}
\end{figure}