ELEC4410

Control Systems Design

Lecture 11: State Space Equations

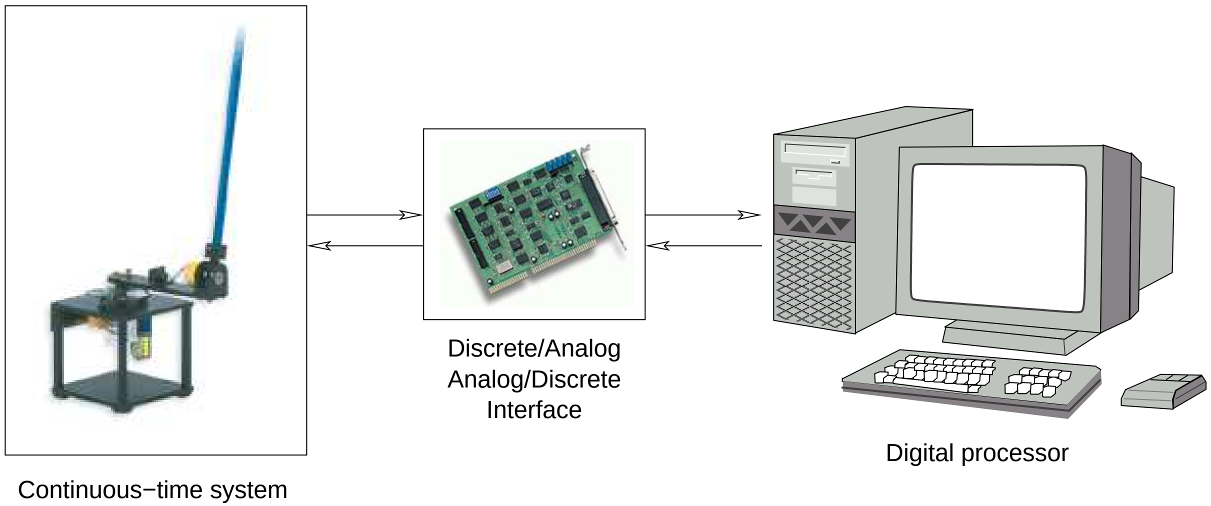
School of Electrical Engineering and Computer Science
The University of Newcastle

Outline

- Brief Review of Discrete-Time Systems
- Solution of LTI State Equations
  - Solution of Continuous-Time State Equations
  - The Matrix Exponential
  - Discretisation of LTI Systems
  - Solution of Discrete-Time State Equations

Brief Review of Discrete-Time Systems

Discrete-time systems are systems that are **digital** or arise from the **sampling** of a continuous-time system. An example, is the control of a continuous-time system through a digital processor.



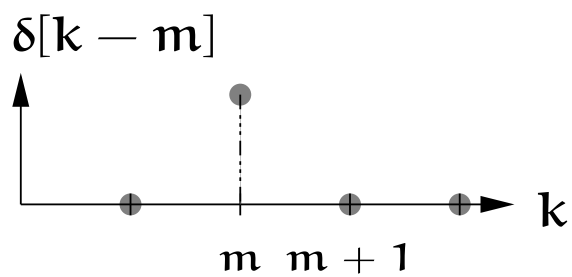
The continuous-time system, as seen from the discrete processor, is a discrete-time system.

Signals in a discrete-time system are not defined for *all* time $t \in \mathbb{R}$, but only for t in a **countable** (although maybe infinite) set. Thus, we can always assume $t = 0, 1, 2, 3, 4, \ldots$.

Brief Review of Discrete-Time Systems

Define the **impulse sequence** $\delta[k]$ as

$$\delta[k-m] = \begin{cases} 1 & \text{if } k = m \\ 0 & \text{if } k \neq m \end{cases} \quad \text{where } k \text{ and } m \text{ are integers.}$$



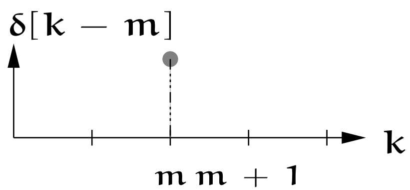
In the discrete-time case impulses are easy to implement physically, in contrast to the continuous-time case.

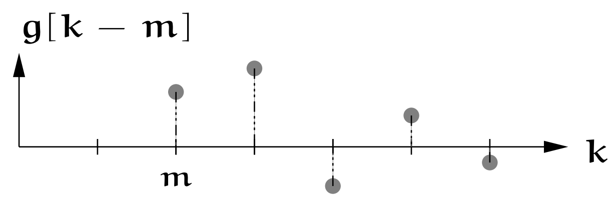
A sequence $u[k]$ can be represented by means of the series

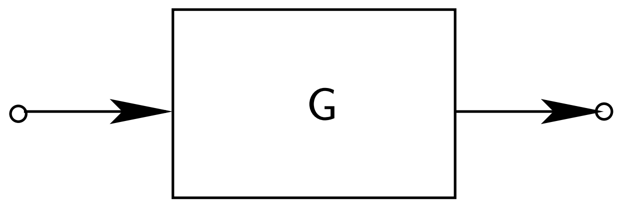
$$u[k] = \sum_{m=-\infty}^{\infty} u[m]\,\delta[k-m]\,.$$

Brief Review of Discrete-Time Systems

Let $g[k-m]$ denote the response of a causal, discrete-time **linear time-invariant** (LTI) system to a unit impulse applied at the instant m.







Then the output of the system to an arbitrary input sequence $u[k]$ is given the **discrete convolution**

$$
\begin{aligned}
y[k] &= \sum_{k=0}^{\infty} g[k-m]\,u[m] \\
&= \sum_{k=0}^{\infty} g[m]\,u[k-m].
\end{aligned}
$$

Brief Review of Discrete-Time Systems

The z-transform is an important tool in the study of LTI discrete-time systems. Denote by $Y(z)$ the z-transform of the sequence $y[k]$, defined as

$$Y(z) \triangleq \mathcal{Z}\{y[k]\} \triangleq \sum_{k=0}^{\infty} y[k]\, z^{-k}. \tag{Z}$$

Using the discrete convolution representation of $y[k]$ in (Z),

$$Y(z) = \sum_{k=0}^{\infty} \left(\sum_{m=0}^{\infty} g[k-m]\, u[m] \right) z^{-k}$$

Brief Review of Discrete-Time Systems

The *z*-transform is an important tool in the study of LTI discrete-time systems. Denote by $Y(z)$ the z-transform of the sequence $y[k]$, defined as

$$Y(z) \triangleq \mathcal{Z}\{y[k]\} \triangleq \sum_{k=0}^{\infty} y[k]\, z^{-k}. \tag{Z}$$

Using the discrete convolution representation of $y[k]$ in (Z),

$$Y(z) = \sum_{k=0}^{\infty} \left(\sum_{m=0}^{\infty} g[k-m]\, u[m] \right) z^{-k+m} z^{-m}$$

Brief Review of Discrete-Time Systems

The z-transform is an important tool in the study of LTI discrete-time systems. Denote by $Y(z)$ the z-transform of the sequence $y[k]$, defined as

$$Y(z) \triangleq \mathcal{Z}\{y[k]\} \triangleq \sum_{k=0}^{\infty} y[k]\, z^{-k}. \tag{Z}$$

Using the discrete convolution representation of $y[k]$ in (Z),

$$\begin{aligned} Y(z) &= \sum_{k=0}^{\infty} \left(\sum_{m=0}^{\infty} g[k-m]\, u[m] \right) z^{-k+m} z^{-m} \\ &= \sum_{m=0}^{\infty} \left(\sum_{k=0}^{\infty} g[k-m]\, z^{-(k-m)} \right) u[m]\, z^{-m} \end{aligned}$$

Brief Review of Discrete-Time Systems

The z-transform is an important tool in the study of LTI discrete-time systems. Denote by $Y(z)$ the z-transform of the sequence $y[k]$, defined as

$$Y(z) \triangleq \mathcal{Z}\{y[k]\} \triangleq \sum_{k=0}^{\infty} y[k]\, z^{-k}. \tag{Z}$$

Using the discrete convolution representation of $y[k]$ in (Z),

$$\begin{aligned}
Y(z) &= \sum_{k=0}^{\infty} \left(\sum_{m=0}^{\infty} g[k-m]\, u[m] \right) z^{-k+m} z^{-m} \\
&= \sum_{m=0}^{\infty} \left(\sum_{k=0}^{\infty} g[k-m]\, z^{-(k-m)} \right) u[m]\, z^{-m} \\
&= \underbrace{\left(\sum_{l=0}^{\infty} g[l]\, z^{-l} \right)}_{G(z)} \underbrace{\left(\sum_{m=0}^{\infty} u[m]\, z^{-m} \right)}_{U(z)}
\end{aligned}$$

Brief Review of Discrete-Time Systems

The z-transform is an important tool in the study of LTI discrete-time systems. Denote by $Y(z)$ the z-transform of the sequence $y[k]$, defined as

$$Y(z) \triangleq \mathcal{Z}\{y[k]\} \triangleq \sum_{k=0}^{\infty} y[k]\, z^{-k}. \tag{Z}$$

Using the discrete convolution representation of $y[k]$ in (Z),

$$\begin{aligned} Y(z) &= \sum_{k=0}^{\infty}\left(\sum_{m=0}^{\infty} g[k-m]\, u[m]\right) z^{-k+m} z^{-m} \\ &= \sum_{m=0}^{\infty}\left(\sum_{k=0}^{\infty} g[k-m]\, z^{-(k-m)}\right) u[m]\, z^{-m} \\ &= \underbrace{\left(\sum_{l=0}^{\infty} g[l]\, z^{-l}\right)}_{G(z)} \underbrace{\left(\sum_{m=0}^{\infty} u[m]\, z^{-m}\right)}_{U(z)} = G(z)U(z). \end{aligned}$$

Brief Review of Discrete-Time Systems

- The equation

$$\boxed{\mathbf{Y}(z) = \mathbf{G}(z)\mathbf{U}(z)}$$

  is the discrete counterpart of the transfer function representation $\mathbf{Y}(s) = \mathbf{G}(s)\mathbf{U}(s)$ for continuous-time systems.

- The function $\mathbf{G}(z)$ is the z-transform of the impulse response sequence $\mathbf{g}[k]$ and is called the **discrete transfer function**.

- Both the discrete convolution and transfer function describe the system assuming **zero initial conditions**.

Brief Review of Discrete-Time Systems

Example. Consider the unit-sampling-time delay system defined by

$$y[k] = u[k-1].$$

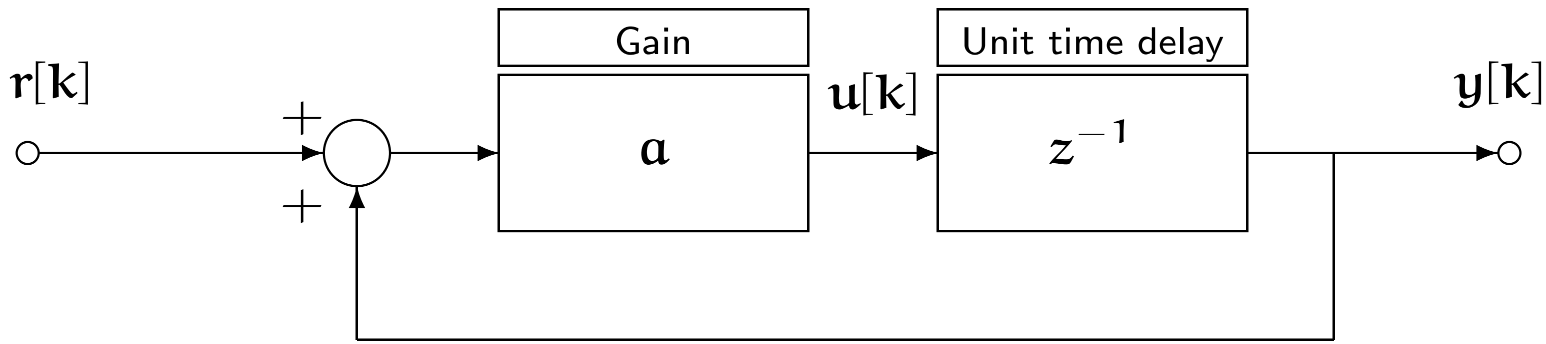
The output equals the input delayed by one sampling period. Its impulse response sequence is $g[k] = \delta[k-1]$ and its discrete transfer function is

$$G(z) = \mathcal{Z}\{\delta[k-1]\} = z^{-1} = \frac{1}{z}.$$

It is a rational function of z. Note that every continuous-time system involving a time-delay is a distributed system. This is not so in discrete-time systems. $\square$

Brief Review of Discrete-Time Systems

Example. Consider the discrete-time system of the block diagram below.



If the unit-sampling-time delay is replaced by its discrete transfer function z^{-1}, then the discrete transfer function from r to y can be computed as

$$G(z) = \frac{az^{-1}}{1 - az^{-1}} = \frac{a}{z - a}$$

Brief Review of Discrete-Time Systems

Example (continuation). On the other hand, let the reference input r be a unit impulse $\delta[k]$. By assuming $y[0] = 0$, we have

$$y[0] = 0, \quad y[1] = a, \quad y[2] = a^2, \quad y[2] = a^3, \quad \dots$$

Thus,

$$y[k] = g[k] = a\delta[k-1] + a^2\delta[k-2] + \cdots = \sum_{m=0}^{\infty} a^m \delta[k-m].$$

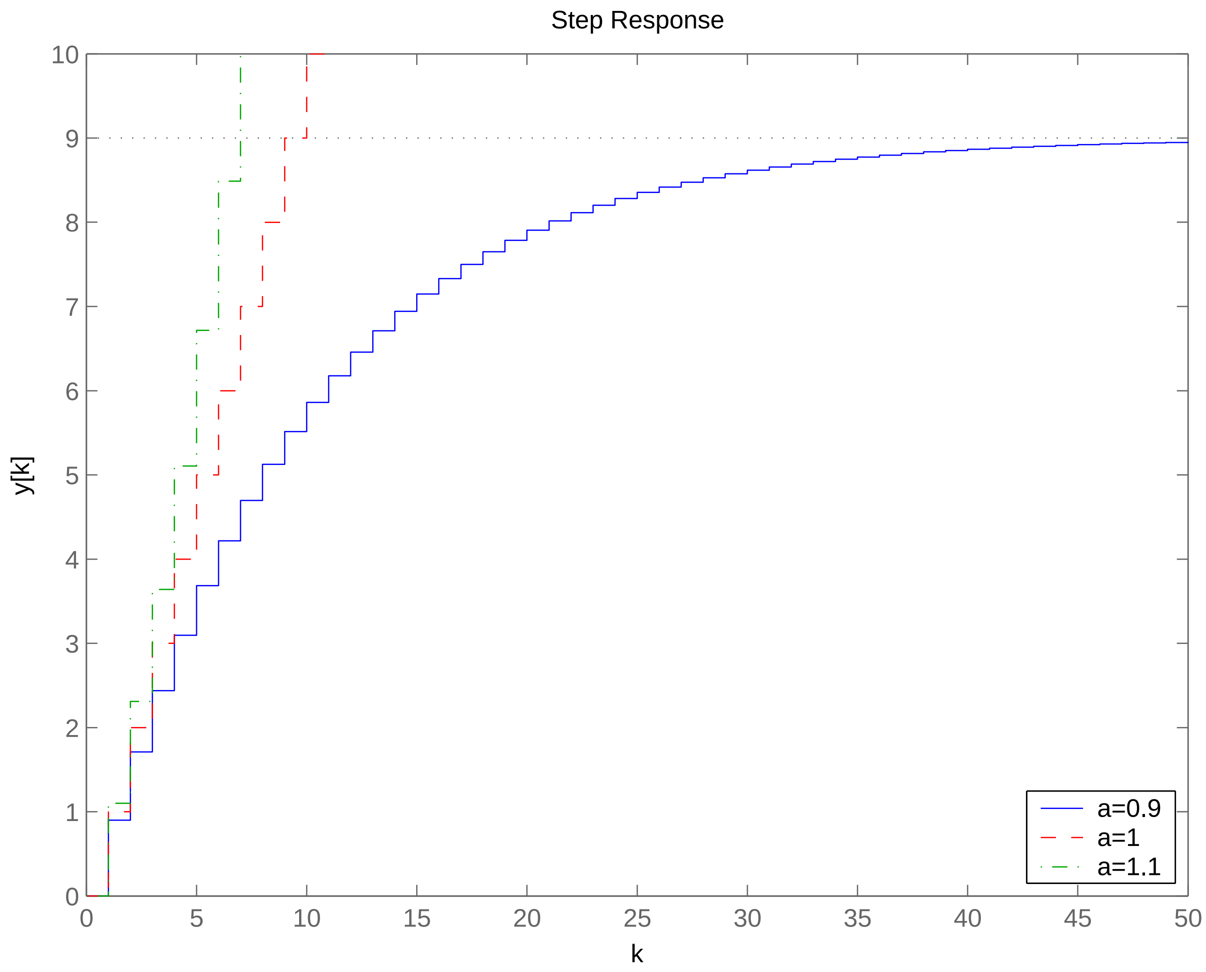
Because $\mathcal{Z}\{\delta[k-m]\} = z^{-m}$, the transfer function of the system is

$$\begin{aligned} G(z) &= \mathcal{Z}\{g[k]\} = az^{-1} + a^2z^{-2} + a^3z^{-3} + \cdots \\ &= az^{-1}\sum_{m=0}^{\infty}(az^{-1})^m = \frac{az^{-1}}{1 - az^{-1}}, \end{aligned}$$

the same result as before.

Brief Review of Discrete-Time Systems

Example (continuation). The plot shows the step response of the system for different values of a.



Brief Review of Discrete-Time Systems

Every discrete-time, finite dimensional, linear system can be represented by state space **difference equations**, as in

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k] \\ y[k] &= Cx[k] + Du[k]\,. \end{aligned}$$

The relation between discrete transfer function representation and state space representation is identical to the continuous-time case,

$$\hat{G}(z) = C(zI - A)^{-1}B + D\,,$$

and the same MATLAB functions can be used to define systems, e.g.,

```
G1 = ss(A,B,C,D,T);
G2 = tf(Num,Den,T);
```

Summary on Discrete-Time Systems

- Most of the state space concepts for linear continuous-time systems directly translate to discrete-time systems, described by **linear difference equations**. In this case the time variable t only takes values a set like $\{0, 1, 2, \dots\}$.

Summary on Discrete-Time Systems

- Most of the state space concepts for linear continuous-time systems directly translate to discrete-time systems, described by **linear difference equations**. In this case the time variable t only takes values a set like $\{0, 1, 2, \dots\}$.
- When the discrete-time system is obtained by sampling a continuous-time system, we have that $t = kT$, $k = 0, 1, 2, \dots,$ where T is the **sampling period**. We denote the discrete-time variables (sequences) as $u[k] \triangleq u(kT)$.

Summary on Discrete-Time Systems

- Most of the state space concepts for linear continuous-time systems directly translate to discrete-time systems, described by **linear difference equations**. In this case the time variable t only takes values a set like $\{0, 1, 2, \dots\}$.
- When the discrete-time system is obtained by sampling a continuous-time system, we have that $t = kT$, $k = 0, 1, 2, \dots$, where T is the **sampling period**. We denote the discrete-time variables (sequences) as $u[k] \triangleq u(kT)$.
- Finite dimensionality, causality, linearity and the superposition principle for responses to initial conditions and inputs are exactly the same as those in the continuous-time case.

Summary on Discrete-Time Systems

- Most of the state space concepts for linear continuous-time systems directly translate to discrete-time systems, described by **linear difference equations**. In this case the time variable t only takes values a set like $\{0, 1, 2, \dots\}$.
- When the discrete-time system is obtained by sampling a continuous-time system, we have that $t = kT$, $k = 0, 1, 2, \dots,$ where T is the **sampling period**. We denote the discrete-time variables (sequences) as $u[k] \triangleq u(kT)$.
- Finite dimensionality, causality, linearity and the superposition principle for responses to initial conditions and inputs are exactly the same as those in the continuous-time case.
- One difference though: pure delays in discrete-time do not give raise to an infinite-dimensional system, as is the case for continuous-time systems, if the delay is a multiple of the sampling period T.

Outline

- Brief Review of Discrete-Time Systems
- Solution of LTI State Equations
  - Solution of Continuous-Time State Equations
  - The Matrix Exponential
  - Discretisation of LTI Systems
  - Solution of Discrete-Time State Equations

Solution of LTI State Equations

As we have seen, linear systems can be represented by means of a convolution integral and, if they are finite-dimensional, also by means of state space equations.

We are interested in obtaining $y(t)$ for $t \geq t_0$, given the value of $u(t)$ for all $t \in [t_0, t]$.

- There is no simple analytical form to solve the convolution integral

$$y(t) = \int_{t_0}^{t} g(t, \tau) u(\tau) d\tau.$$

Solution of LTI State Equations

As we have seen, linear systems can be represented by means of a convolution integral and, if they are finite-dimensional, also by means of state space equations.

We are interested in obtaining $y(t)$ for $t \geq t_0$, given the value of $u(t)$ for all $t \in [t_0, t]$.

- There is no simple analytical form to solve the convolution integral

$$y(t) = \int_{t_0}^{t} g(t, \tau) u(\tau) d\tau.$$

- Probably, the simplest way would be to compute it numerically, for which we would need first to approximate it by performing a discretisation.

Solution of LTI State Equations

When the system has finite dimensions, the most efficient way to compute $y(t)$ is to obtain a representation in state equations of the convolution integral (that is, a *state space realisation*) and solve the equations

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \qquad \text{(SE)}$$

$$y(t) = C(t)x(t) + D(t)u(t)\,. \qquad \text{(OE)}$$

Solution of LTI State Equations

When the system has finite dimensions, the most efficient way to compute $y(t)$ is to obtain a representation in state equations of the convolution integral (that is, a *state space realisation*) and solve the equations

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \qquad \text{(SE)}$$

$$y(t) = C(t)x(t) + D(t)u(t)\,. \qquad \text{(OE)}$$

We will only consider the LTI case, i.e., when A, B, C, D are constant matrices. We start by looking for the solution $x(t)$ to the equation

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

with a given initial state $x(0)$ and input $u(t), t \geq 0$.

Solution of LTI State Equations

One way to find the solution in this case, because it is simple, is to postulate a candidate solution $x(t)$ and then check that it satisfies the equation.

Solution of LTI State Equations

One way to find the solution in this case, because it is simple, is to postulate a candidate solution $x(t)$ and then check that it satisfies the equation.

We know that for a *scalar* ($x(t) \in \mathbb{R}$) equation

$$\dot{x}(t) = ax(t)$$

the solution has the form $x(t) = e^{at}x(0)$.

Solution of LTI State Equations

One way to find the solution in this case, because it is simple, is to postulate a candidate solution $x(t)$ and then check that it satisfies the equation.

We know that for a *scalar* ($x(t) \in \mathbb{R}$) equation

$$\dot{x}(t) = ax(t)$$

the solution has the form $x(t) = e^{at}x(0)$. Thus, we can reasonably assume that $x(t)$ in the matrix equation

$$\dot{x}(t) = Ax(t)$$

will involve the matrix exponential e^{At}.

We make a brief detour from the solution of the state equation to review a few facts about the matrix exponential.

The Matrix Exponential

- For any square matrix $\mathbf{M}$, the matrix exponential $e^{\mathbf{M}}$ is a square matrix function. In MATLAB, $e^{\mathbf{A}}$ is computed with the function `expm(M)`, which uses the *Padé approximation*.

The Matrix Exponential

- For any square matrix $\mathbf{M}$, the matrix exponential $e^{\mathbf{M}}$ is a square matrix function. In MATLAB, $e^{\mathbf{A}}$ is computed with the function `expm(M)`, which uses the *Padé approximation*.
- Note the difference with the MATLAB function `exp(M)`, which computes the matrix of exponentials of the elements of $\mathbf{M}$.

The Matrix Exponential

- For any square matrix M, the matrix exponential e^{M} is a square matrix function. In MATLAB, e^{A} is computed with the function `expm(M)`, which uses the *Padé approximation*.
- Note the difference with the MATLAB function `exp(M)`, which computes the matrix of exponentials of the elements of M.
- Because the Taylor expansion $e^{\lambda t} = 1 + \lambda t + \frac{\lambda^2 t^2}{2!} + \cdots + \frac{\lambda^n t^n}{n!} + \cdots$ converges for all finite λ and t, we have that for matrices

$$\boxed{e^{At} = I + tA + \frac{t^2}{2!}A + \cdots = \sum_{k=0}^{\infty} \frac{t^k}{k!}A^k}. \qquad \text{(TE)}$$

The Matrix Exponential

By using the Taylor expansion (TE) it's easy to show the following first three important properties of the matrix exponential e^{At}

$$e^{0} = I\,, \tag{P1}$$

$$e^{A(t_1+t_2)} = e^{At_1}e^{At_2}\,, \tag{P2}$$

$$\frac{d}{dt}e^{At} = Ae^{At} = e^{At}A\,, \tag{P3}$$

$$\left(e^{At}\right)^{-1} = e^{-At}\,. \tag{P4}$$

Exercise: Prove property (P4). Note that in general $e^{(A+B)t} \neq e^{At}e^{Bt}$ (Why?).

Matrix differentiation and integration applies **element-wise**.

Solution of LTI State Equations

We now return to the solution of the state equation

$$\dot{x}(\tau) = Ax(\tau) + Bu(\tau).$$

Following the scalar case, we multiply (from the right) both sides of the equation by $e^{-A\tau}$ to obtain

$$\begin{aligned} & e^{-A\tau}\dot{x}(\tau) - e^{-A\tau}Ax(\tau) = e^{-A\tau}Bu(\tau) \\ \Leftrightarrow \quad & \frac{d}{d\tau}\left(e^{-A\tau}x(\tau)\right) = e^{-A\tau}Bu(\tau), \qquad \text{by (P3).} \end{aligned}$$

Solution of LTI State Equations

We now return to the solution of the state equation

$$\dot{\mathbf{x}}(\tau) = \mathbf{A}\mathbf{x}(\tau) + \mathbf{B}\mathbf{u}(\tau).$$

Following the scalar case, we multiply (from the right) both sides of the equation by $e^{-\mathbf{A}\tau}$ to obtain

$$\begin{aligned} e^{-\mathbf{A}\tau}\dot{\mathbf{x}}(\tau) - e^{-\mathbf{A}\tau}\mathbf{A}\mathbf{x}(\tau) &= e^{-\mathbf{A}\tau}\mathbf{B}\mathbf{u}(\tau) \\ \Leftrightarrow \qquad \frac{d}{d\tau}\left(e^{-\mathbf{A}\tau}\mathbf{x}(\tau)\right) &= e^{-\mathbf{A}\tau}\mathbf{B}\mathbf{u}(\tau), \qquad \text{by (P3).} \end{aligned}$$

Integration of the last equation between 0 and t yields

$$e^{-\mathbf{A}\tau}\mathbf{x}(\tau)\Big|_{\tau=0}^{t} = \int_0^t e^{-\mathbf{A}\tau}\mathbf{B}\mathbf{u}(\tau)d\tau.$$

Solution of LTI State Equations

In other words, from the last equation we have that

$$\mathbf{e}^{-\mathbf{A}t}\mathbf{x}(t) - \mathbf{e}^{0}\mathbf{x}(0) = \int_0^t \mathbf{e}^{-\mathbf{A}\tau}\mathbf{B}\mathbf{u}(\tau)\,\mathbf{d}\tau.$$

Solution of LTI State Equations

In other words, from the last equation we have that

$$e^{-At}x(t) - e^{0}x(0) = \int_0^t e^{-A\tau}Bu(\tau)d\tau.$$

Because the inverse of e^{-At} is e^{At} and $e^{0} = I$,

Solution of LTI State Equations

In other words, from the last equation we have that

$$e^{-At}x(t) - e^{0}x(0) = \int_0^t e^{-A\tau}Bu(\tau)d\tau.$$

Because the inverse of e^{-At} is e^{At} and $e^{0} = I$, we finally have that the solution of the state equation is given by

$$\boxed{x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau.} \qquad \text{(PVF)}$$

Solution of LTI State Equations

In other words, from the last equation we have that

$$e^{-At}x(t) - e^{0}x(0) = \int_0^t e^{-A\tau}Bu(\tau)d\tau.$$

Because the inverse of e^{-At} is e^{At} and $e^0 = I$, we finally have that the solution of the state equation is given by

$$\boxed{x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau.} \qquad \text{(PVF)}$$

Equation (PVF) is the *general solution of the state equation* (SE), and is sometimes referred to as the Parameter Variation Formula.

Solution of LTI State Equations

Now that we have the solution to the equation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t),$$

we conclude by replacing $\mathbf{x}(t)$ into the algebraic output equation

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t),$$

$$\boxed{\mathbf{y}(t) = \mathbf{C}e^{\mathbf{A}t}\mathbf{x}(0) + \mathbf{C}\int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)d\tau + \mathbf{D}\mathbf{u}(t)}$$

(OR)

the response to initial conditions

and the response to the input.

Solution of LTI State Equations

Now that we have the solution to the equation

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

we conclude by replacing $x(t)$ into the algebraic output equation

$$y(t) = Cx(t) + Du(t),$$

and obtain

$$y(t) = Ce^{At}x(0) + C\int_0^t e^{A(t-\tau)}Bu(\tau)d\tau + Du(t) \qquad \text{(OR)}$$

the response to initial conditions

and the response to the input.

Solution of LTI State Equations

Now that we have the solution to the equation

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

we conclude by replacing $x(t)$ into the algebraic output equation

$$y(t) = Cx(t) + Du(t),$$

and obtain

$$\boxed{y(t) = Ce^{At}x(0) + C\int_0^t e^{A(t-\tau)}Bu(\tau)d\tau + Du(t)} \qquad \text{(OR)}$$

the response to initial conditions and the response to the input.

Solution of LTI State Equations

Now that we have the solution to the equation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t),$$

we conclude by replacing $\mathbf{x}(t)$ into the algebraic output equation

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t),$$

and obtain

$$\boxed{\mathbf{y}(t) = \mathbf{C}e^{\mathbf{A}t}\mathbf{x}(0) + \mathbf{C}\int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)d\tau + \mathbf{D}\mathbf{u}(t)} \qquad \text{(OR)}$$

Notice the superposition of the response to initial conditions and the response to the input.

Solution of LTI State Equations

Now that we have the solution to the equation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t),$$

we conclude by replacing $\mathbf{x}(t)$ into the algebraic output equation

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t),$$

and obtain

$$\boxed{\mathbf{y}(t) = \mathbf{C}e^{\mathbf{A}t}\mathbf{x}(0) + \mathbf{C}\int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)d\tau + \mathbf{D}\mathbf{u}(t)} \qquad \text{(OR)}$$

Notice the superposition of the response to initial conditions and the response to the input.

Solution of LTI State Equations

Now that we have the solution to the equation

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t),$$

we conclude by replacing $\mathbf{x}(t)$ into the algebraic output equation

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t),$$

and obtain

$$\boxed{\mathbf{y}(t) = \mathbf{C}e^{\mathbf{A}t}\mathbf{x}(0) + \mathbf{C}\int_0^t e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)d\tau + \mathbf{D}\mathbf{u}(t)} \qquad \text{(OR)}$$

Notice the superposition of the response to initial conditions and the response to the input.

Solution of LTI State Equations

Now that we have the solution to the equation

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

we conclude by replacing $x(t)$ into the algebraic output equation

$$y(t) = Cx(t) + Du(t),$$

and obtain

$$\boxed{y(t) = Ce^{At}x(0) + C\int_0^t e^{A(t-\tau)}Bu(\tau)d\tau + Du(t)} \qquad \text{(OR)}$$

Notice the superposition of the response to initial conditions and the response to the input.

Solution of LTI State Equations

An alternative way to compute the solution of the state space equation is via the Laplace Transform.

- Apply the Laplace Transform to the state and output equations (SE) and (OE) to obtain

$$\mathbf{X}(s) = (s\mathbf{I} - \mathbf{A})^{-1}[x(0) + \mathbf{B}\mathbf{U}(s)]$$

$$\mathbf{Y}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}[x(0) + \mathbf{B}\mathbf{U}(s)] + \mathbf{D}\mathbf{U}(s).$$

Solution of LTI State Equations

An alternative way to compute the solution of the state space equation is via the Laplace Transform.

- Apply the Laplace Transform to the state and output equations (SE) and (OE) to obtain

$$\begin{aligned} \mathbf{X}(s) &= (s\mathbf{I} - \mathbf{A})^{-1}\left[x(0) + \mathbf{B}\mathbf{U}(s)\right] \\ \mathbf{Y}(s) &= \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\left[x(0) + \mathbf{B}\mathbf{U}(s)\right] + \mathbf{D}\mathbf{U}(s). \end{aligned}$$

- Then solve the above *algebraic* equations to compute $\mathbf{Y}(s)$.

Solution of LTI State Equations

An alternative way to compute the solution of the state space equation is via the Laplace Transform.

- Apply the Laplace Transform to the state and output equations (SE) and (OE) to obtain

$$\begin{aligned} \mathbf{X}(s) &= (s\mathbf{I} - \mathbf{A})^{-1}\left[x(0) + \mathbf{B}\mathbf{U}(s)\right] \\ \mathbf{Y}(s) &= \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\left[x(0) + \mathbf{B}\mathbf{U}(s)\right] + \mathbf{D}\mathbf{U}(s). \end{aligned}$$

- Then solve the above *algebraic* equations to compute $\mathbf{Y}(s)$.
- Finally, anti-transform $\mathbf{Y}(s)$ to go back to the time domain and obtain $y(t)$.

Zero-Input Response

We discuss a general property of the zero-input response $e^{\mathbf{A}t}\mathbf{x}(0)$. Suppose that we have a matrix $\mathbf{A}$ whose **Jordan form** is

$$\bar{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q} = \begin{bmatrix} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{bmatrix}$$

where $\mathbf{Q}$ is a nonsingular matrix that makes the change of coordinates that brings $\mathbf{A}$ to $\bar{\mathbf{A}}$. (Given any matrix $\mathbf{A}$, there is always a nonsingular matrix $\mathbf{Q}$ that gives its Jordan form $\bar{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q}$ as above.)

Zero-Input Response

We discuss a general property of the zero-input response $e^{\mathbf{A}t}\mathbf{x}(0)$. Suppose that we have a matrix $\mathbf{A}$ whose **Jordan form** is

$$\bar{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q} = \begin{bmatrix} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{bmatrix}$$

where $\mathbf{Q}$ is a nonsingular matrix that makes the change of coordinates that brings $\mathbf{A}$ to $\bar{\mathbf{A}}$. (Given any matrix $\mathbf{A}$, there is always a nonsingular matrix $\mathbf{Q}$ that gives its Jordan form $\bar{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q}$ as above.)

The scalars λ_1 and λ_2 are the **eigenvalues** of $\bar{\mathbf{A}}$, which are also those of $\mathbf{A}$. The matrix exponential of a matrix in its Jordan form is easy to compute. For the above example we have

$$e^{\bar{\mathbf{A}}t} = \begin{bmatrix} e^{\lambda_1 t} & te^{\lambda_1 t} & 0 \\ 0 & e^{\lambda_1 t} & 0 \\ 0 & 0 & e^{\lambda_2 t} \end{bmatrix}$$

Zero-Input Response

The matrix exponential of $\mathbf{A}$ is obtained from that of $\bar{\mathbf{A}}$ by changing back the coordinates,

$$e^{\mathbf{A}t} = \mathbf{Q}e^{\bar{\mathbf{A}}t}\mathbf{Q}^{-1}.$$

Zero-Input Response

The matrix exponential of A is obtained from that of $\bar{A}$ by changing back the coordinates,

$$e^{At} = Qe^{\bar{A}t}Q^{-1}.$$

Thus, we see that the general response of the system to initial conditions is a linear combination of the terms $e^{\lambda_1 t}$, $te^{\lambda_1 t}$ and $e^{\lambda_2 t}$,

$$x(t) = e^{At}x(0) = Q \begin{bmatrix} e^{\lambda_1 t} & te^{\lambda_1 t} & 0 \\ 0 & e^{\lambda_1 t} & 0 \\ 0 & 0 & e^{\lambda_2 t} \end{bmatrix} Q^{-1}x(0)$$

Zero-Input Response

The matrix exponential of A is obtained from that of $\bar{A}$ by changing back the coordinates,

$$e^{At} = Qe^{\bar{A}t}Q^{-1}.$$

Thus, we see that the general response of the system to initial conditions is a linear combination of the terms $e^{\lambda_1 t}$, $te^{\lambda_1 t}$ and $e^{\lambda_2 t}$,

$$x(t) = e^{At}x(0) = Q \begin{bmatrix} e^{\lambda_1 t} & te^{\lambda_1 t} & 0 \\ 0 & e^{\lambda_1 t} & 0 \\ 0 & 0 & e^{\lambda_2 t} \end{bmatrix} Q^{-1}x(0)$$

If all eigenvalues of A have **negative real parts**, the system response to initial conditions will decay to zero as $t \to \infty$. Otherwise, the response may grow unbounded.

Computing the Matrix Exponential

Formulas (PVF) or (OR) require the matrix exponential $e^{\mathbf{A}t}$. The Taylor expansion (TE) could be a way to compute $e^{\mathbf{A}t}$, since it only involves matrix multiplications and sums, although an infinite number of them.

However, there are several better ways to compute the matrix exponential, among others:

- the Laplace Transform method

Computing the Matrix Exponential

Formulas (PVF) or (OR) require the matrix exponential $e^{\mathbf{A}t}$. The Taylor expansion (TE) could be a way to compute $e^{\mathbf{A}t}$, since it only involves matrix multiplications and sums, although an infinite number of them.

However, there are several better ways to compute the matrix exponential, among others:

- the Laplace Transform method
- the Jordan decomposition method

Computing the Matrix Exponential

Formulas (PVF) or (OR) require the matrix exponential $e^{\mathbf{A}t}$. The Taylor expansion (TE) could be a way to compute $e^{\mathbf{A}t}$, since it only involves matrix multiplications and sums, although an infinite number of them.

However, there are several better ways to compute the matrix exponential, among others:

- the Laplace Transform method
- the Jordan decomposition method
- the Cayley-Hamilton Theorem method

Computing the Matrix Exponential

Formulas (PVF) or (OR) require the matrix exponential $e^{\mathbf{A}t}$. The Taylor expansion (TE) could be a way to compute $e^{\mathbf{A}t}$, since it only involves matrix multiplications and sums, although an infinite number of them.

However, there are several better ways to compute the matrix exponential, among others:

- the Laplace Transform method
- the Jordan decomposition method
- the Cayley-Hamilton Theorem method

See e.g., C.-T. Chen, *Linear System Theory and Design*. Oxford University Press, 1999.

Computing the Matrix Exponential

Formulas (PVF) or (OR) require the matrix exponential $e^{\mathbf{A}t}$. The Taylor expansion (TE) could be a way to compute $e^{\mathbf{A}t}$, since it only involves matrix multiplications and sums, although an infinite number of them.

However, there are several better ways to compute the matrix exponential, among others:

- the Laplace Transform method
- the Jordan decomposition method
- the Cayley-Hamilton Theorem method

See e.g., C.-T. Chen, *Linear System Theory and Design*. Oxford University Press, 1999. We will now have a look at the first method.

Matrix Exponential Via Laplace Transform

From property (P3), we have that

$$\frac{d}{dt}e^{At} = Ae^{At}, \quad \text{with } e^{A0} = I.$$

The Laplace transform of this equation yields

$$\mathcal{L}\left\{\frac{d}{dt}e^{At}\right\} = s\mathcal{L}\{e^{At}\} - I = A\mathcal{L}\{e^{At}\},$$

Matrix Exponential Via Laplace Transform

From property (P3), we have that

$$\frac{d}{dt}e^{At} = Ae^{At}, \quad \text{with } e^{A0} = I.$$

The Laplace transform of this equation yields

$$\mathcal{L}\left\{\frac{d}{dt}e^{At}\right\} = s\mathcal{L}\{e^{At}\} - I = A\mathcal{L}\{e^{At}\},$$

hence,

$$\begin{aligned} & (sI - A)\,\mathcal{L}\{e^{At}\} = I \\ \Leftrightarrow \quad & \mathcal{L}\{e^{At}\} = (sI - A)^{-1} \\ \Leftrightarrow \quad & \boxed{e^{At} = \mathcal{L}^{-1}\left\{(sI - A)^{-1}\right\}} \end{aligned}$$

Matrix Exponential Via Laplace Transform

Example. We consider the LTI equation

$$\dot{x}(t) = \begin{bmatrix} 0 & -1 \\ 1 & -2 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

Matrix Exponential Via Laplace Transform

Example. We consider the LTI equation

$$\dot{x}(t) = \begin{bmatrix} 0 & -1 \\ 1 & -2 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t)$$

Its solution is given by Equation (PVF). We compute e^{At} via the Laplace Transform method. The inverse of $sI - A$ is

$$(sI - A)^{-1} = \begin{bmatrix} s & 1 \\ -1 & s+2 \end{bmatrix}^{-1} = \frac{1}{(s+1)^2} \begin{bmatrix} s+2 & -1 \\ 1 & s \end{bmatrix}$$

$$= \begin{bmatrix} (s+2)/(s+1)^2 & -1/(s+1)^2 \\ 1/(s+1)^2 & s/(s+1)^2 \end{bmatrix}$$

Matrix Exponential Via Laplace Transform

Example (continuation). The matrix $e^{\mathbf{A}t}$ is the Laplace anti-transform of $(s\mathbf{I} - \boldsymbol{A})^{-1}$, which we obtain by performing an expansion in simple fractions and using a table of Laplace Transform pairs (or in MATLAB, with the *symbolic tool-box*, by using the function `ilaplace`).

Matrix Exponential Via Laplace Transform

Example (continuation). The matrix e^{At} is the Laplace anti-transform of $(sI - A)^{-1}$, which we obtain by performing an expansion in simple fractions and using a table of Laplace Transform pairs (or in MATLAB, with the *symbolic tool-box*, by using the function `ilaplace`).

$$\mathcal{L}^{-1}\left\{\begin{bmatrix} \frac{(s+2)}{(s+1)^2} & \frac{-1}{(s+1)^2} \\ \frac{1}{(s+1)^2} & \frac{s}{(s+1)^2} \end{bmatrix}\right\} = \begin{bmatrix} (1+t)e^{-t} & -te^{-t} \\ te^{-t} & (1-t)e^{-t} \end{bmatrix}.$$

Matrix Exponential Via Laplace Transform

Example (continuation). The matrix e^{At} is the Laplace anti-transform of $(s\mathbf{I} - A)^{-1}$, which we obtain by performing an expansion in simple fractions and using a table of Laplace Transform pairs (or in MATLAB, with the *symbolic tool-box*, by using the function `ilaplace`).

$$\mathcal{L}^{-1}\left\{\begin{bmatrix} \frac{(s+2)}{(s+1)^2} & \frac{-1}{(s+1)^2} \\ \frac{1}{(s+1)^2} & \frac{s}{(s+1)^2} \end{bmatrix}\right\} = \begin{bmatrix} (1+t)e^{-t} & -te^{-t} \\ te^{-t} & (1-t)e^{-t} \end{bmatrix}.$$

Finally, by using the Parameter Variation Formula (PVF)

$$x(t) = \begin{bmatrix} (1+t)e^{-t}x_1(0) - te^{-t}x_2(0) \\ te^{-t}x_1(0) + (1-t)e^{-t}x_2(0) \end{bmatrix} + \begin{bmatrix} -\int_0^t (t-\tau)e^{-(t-\tau)}u(\tau)d\tau \\ \int_0^t [1-(t-\tau)]e^{-(t-\tau)}u(\tau)d\tau \end{bmatrix}.$$

$\square$

Outline

- Brief Review of Discrete-Time Systems
- Solution of LTI State Equations
  - Solution of Continuous-Time State Equations
  - The Matrix Exponential
  - Discretisation of LTI Systems
  - Solution of Discrete-Time State Equations

Discretisation

The operation by which a continuous-time model is converted into a discrete-time one is called discretisation.

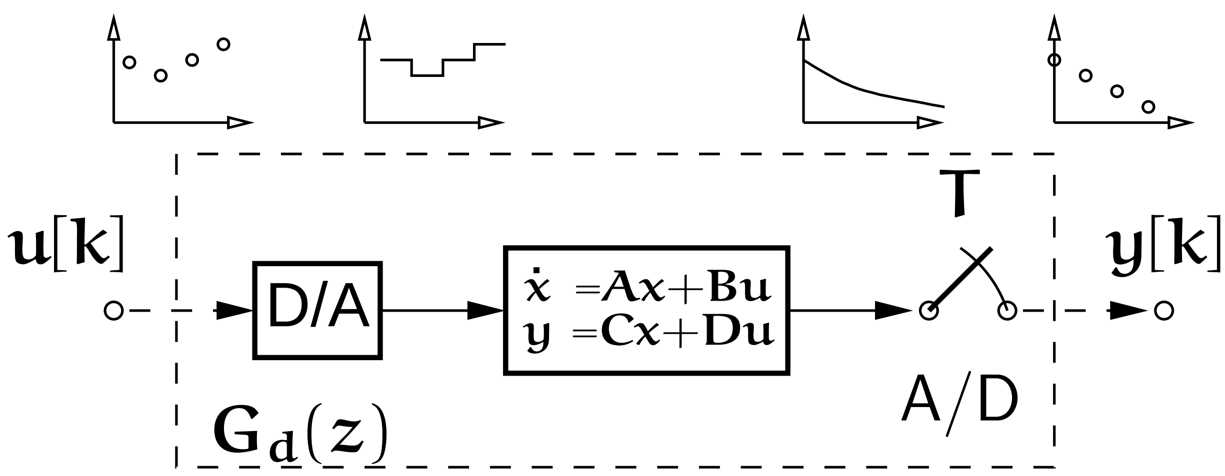
A discrete-time model is often needed, for example to simulate it with a digital computer; or to design a discrete-time controller, which is also implemented in some kind of digital computer.

Discretisation

The operation by which a continuous-time model is converted into a discrete-time one is called discretisation.

A discrete-time model is often needed, for example to simulate it with a digital computer; or to design a discrete-time controller, which is also implemented in some kind of digital computer.

The Parameter Variation Formula yields a direct method for discretisation of a continuous-time system state space model.



Discretisation

Consider a continuous-time, LTI system $\mathbf{G}$ represented by the state equations

$$\mathbf{G} \triangleq \begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)\,. \end{cases}$$

We are after a discrete-time state equation representation

$$\mathbf{G}_d \triangleq \begin{cases} \mathbf{x}[k+1] = \mathbf{A}_d\mathbf{x}[k] + \mathbf{B}_d\mathbf{u}[k] \\ \mathbf{y}[k] = \mathbf{C}_d\mathbf{x}[k] + \mathbf{D}_d\mathbf{u}[k]\,. \end{cases}$$

assuming that the plant has zero order hold at its input and a sampler at its output. We will see two methods:

- Simple (but approximate) discretisation

Discretisation

Consider a continuous-time, LTI system $\mathbf{G}$ represented by the state equations

$$\mathbf{G} \triangleq \begin{cases} \dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u(t) \\ y(t) = \mathbf{C}x(t) + \mathbf{D}u(t)\,. \end{cases}$$

We are after a discrete-time state equation representation

$$\mathbf{G}_d \triangleq \begin{cases} x[k+1] = \mathbf{A}_d x[k] + \mathbf{B}_d u[k] \\ y[k] = \mathbf{C}_d x[k] + \mathbf{D}_d u[k]\,. \end{cases}$$

assuming that the plant has zero order hold at its input and a sampler at its output. We will see two methods:

- Simple (but approximate) discretisation
- Exact discretisation

Simple (But Approximate) Discretisation

This is the most intuitive approach. The simplest way to obtain a discrete model from a continuous-time system regularly sampled with period T is by using Euler's approximation,

$$\dot{x}(t) \approx \frac{x(t+T) - x(t)}{T},$$

to obtain $x(t+T) = x(t) + Ax(t)T + Bu(t)T$.

Simple (But Approximate) Discretisation

This is the most intuitive approach. The simplest way to obtain a discrete model from a continuous-time system regularly sampled with period T is by using Euler's approximation,

$$\dot{x}(t) \approx \frac{x(t+T) - x(t)}{T},$$

to obtain $x(t+T) = x(t) + Ax(t)T + Bu(t)T$. If we are only interested in the evolution of the system *at the sampling instants*, $t = kT$, $k = 0, 1, 2 \ldots$, we arrive to the model

$$x[k+1] = \underbrace{(I + AT)}_{A_d} x[k] + \underbrace{BT}_{B_d} u[k].$$

Simple (But Approximate) Discretisation

This is the most intuitive approach. The simplest way to obtain a discrete model from a continuous-time system regularly sampled with period T is by using Euler's approximation,

$$\dot{x}(t) \approx \frac{x(t+T) - x(t)}{T},$$

to obtain $x(t+T) = x(t) + Ax(t)T + Bu(t)T$. If we are only interested in the evolution of the system *at the sampling instants*, $t = kT$, $k = 0, 1, 2 \ldots,$ we arrive to the model

$$x[k+1] = \underbrace{(I + AT)}_{A_d} x[k] + \underbrace{BT}_{B_d} u[k].$$

This discrete model is simple to obtain, although inexact even at the sampling instants.

Exact Discretisation

An exact discrete model of the continuous time system may be obtained by using the PVF. Note that the output of the *zero order hold* (D/A) is kept constant during each sampling period T until the new sample arrives,

$$u(t) = u(kT) \triangleq u[k] \quad \text{para } t : kT \leq t < (k+1)T.$$

Exact Discretisation

An exact discrete model of the continuous time system may be obtained by using the PVF. Note that the output of the *zero order hold* (D/A) is kept constant during each sampling period T until the new sample arrives,

$$u(t) = u(kT) \triangleq u[k] \quad \text{para } t : kT \leq t < (k+1)T.$$

Now, for this *sectionally constant* input, we evaluate the state of the continuous-time system at the sampling instant $t = (k+1)T$,

$$x[k+1] \triangleq x((k+1)T) = e^{A(k+1)T}x(0) + \int_0^{(k+1)T} e^{A((k+1)T-\tau)}Bu(\tau)d\tau$$

$$= e^{AT}\underbrace{\left(e^{AkT}x(0) + \int_0^{kT} e^{A(kT-\tau)}Bu(\tau)d\tau\right)}_{x[k]} + \int_{kT}^{(k+1)T} e^{A((k+1)T-\tau)}Bu[k]d\tau$$

$$= e^{AT}x[k] + \left(\int_0^T e^{A\sigma}d\sigma\right)Bu[k]. \qquad \text{(where we used } \sigma = (k+1)T - \tau\text{)}.$$

Exact Discretisation

Thus, we have arrived at the discrete-time model

$$\begin{aligned} x[k+1] &= A_d x[k] + B_d u[k] \\ y[k] &= C_d x[k] + D_d u[k], \end{aligned}$$

where

$$\boxed{A_d \triangleq e^{AT}}, \quad \boxed{B_d \triangleq \int_0^T e^{A\tau} d\tau B}, \quad \boxed{C_d \triangleq C}, \quad \boxed{D_d \triangleq D}.$$

Exact Discretisation

Thus, we have arrived at the discrete-time model

$$\begin{aligned} x[k+1] &= A_d x[k] + B_d u[k] \\ y[k] &= C_d x[k] + D_d u[k], \end{aligned}$$

where

$$\boxed{A_d \triangleq e^{AT}}, \quad \boxed{B_d \triangleq \int_0^T e^{A\tau} d\tau B}, \quad \boxed{C_d \triangleq C}, \quad \boxed{D_d \triangleq D}.$$

This discrete model gives the exact value of the variables at time $t = kT$. In MATLAB the function `[Ad,Bd] = c2d(A,B,T)` computes A_d and B_d using the above expressions.

Exact Discretisation

Thus, we have arrived at the discrete-time model

$$x[k+1] = A_d x[k] + B_d u[k]$$
$$y[k] = C_d x[k] + D_d u[k],$$

where

$$\boxed{A_d \triangleq e^{AT}}, \quad \boxed{B_d \triangleq \int_0^T e^{A\tau} d\tau B}, \quad \boxed{C_d \triangleq C}, \quad \boxed{D_d \triangleq D}.$$

This discrete model gives the exact value of the variables at time $t = kT$. In MATLAB the function `[Ad,Bd] = c2d(A,B,T)` computes A_d and B_d using the above expressions.

By using the equality $A \int_0^T e^{A\tau} d\tau = e^{AT} - I$, if A is non singular, a quick way to compute B_d is from the formula

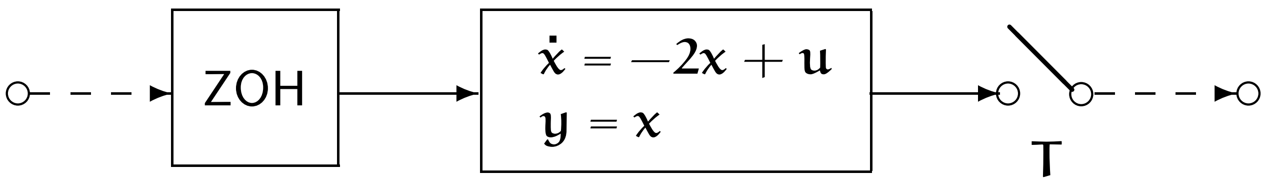
$$\boxed{B_d = A^{-1}(A_d - I)B}, \qquad \text{if } \mathbf{det}\{A\} \neq 0.$$

Exact Discretisation

Example. Consider the scalar system

$$\dot{x}(t) = -2x(t) + u(t), \quad y(t) = x(t).$$

We wish to obtain a discrete-time model of the system sampled with period T, and assuming a ZOH at its input.

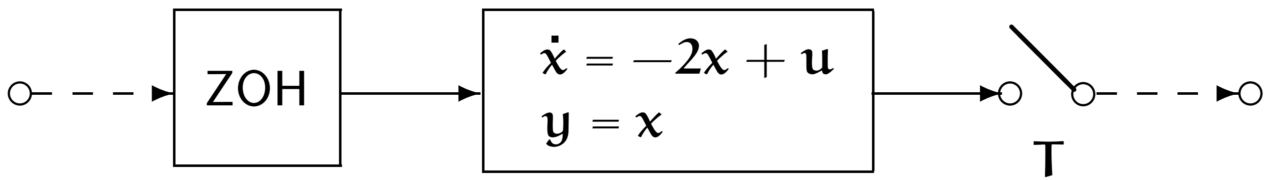


Exact Discretisation

Example. Consider the scalar system

$$\dot{x}(t) = -2x(t) + u(t), \quad y(t) = x(t).$$

We wish to obtain a discrete-time model of the system sampled with period T, and assuming a ZOH at its input.



The **approximate** discretisation from Euler's formula yields

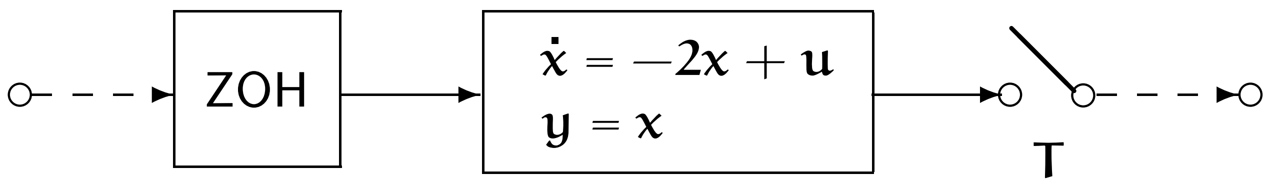
$$x[k+1] = (1-2T)x[k] + Tu[k],$$

Exact Discretisation

Example. Consider the scalar system

$$\dot{x}(t) = -2x(t) + u(t), \quad y(t) = x(t).$$

We wish to obtain a discrete-time model of the system sampled with period T, and assuming a ZOH at its input.



The **approximate** discretisation from Euler's formula yields

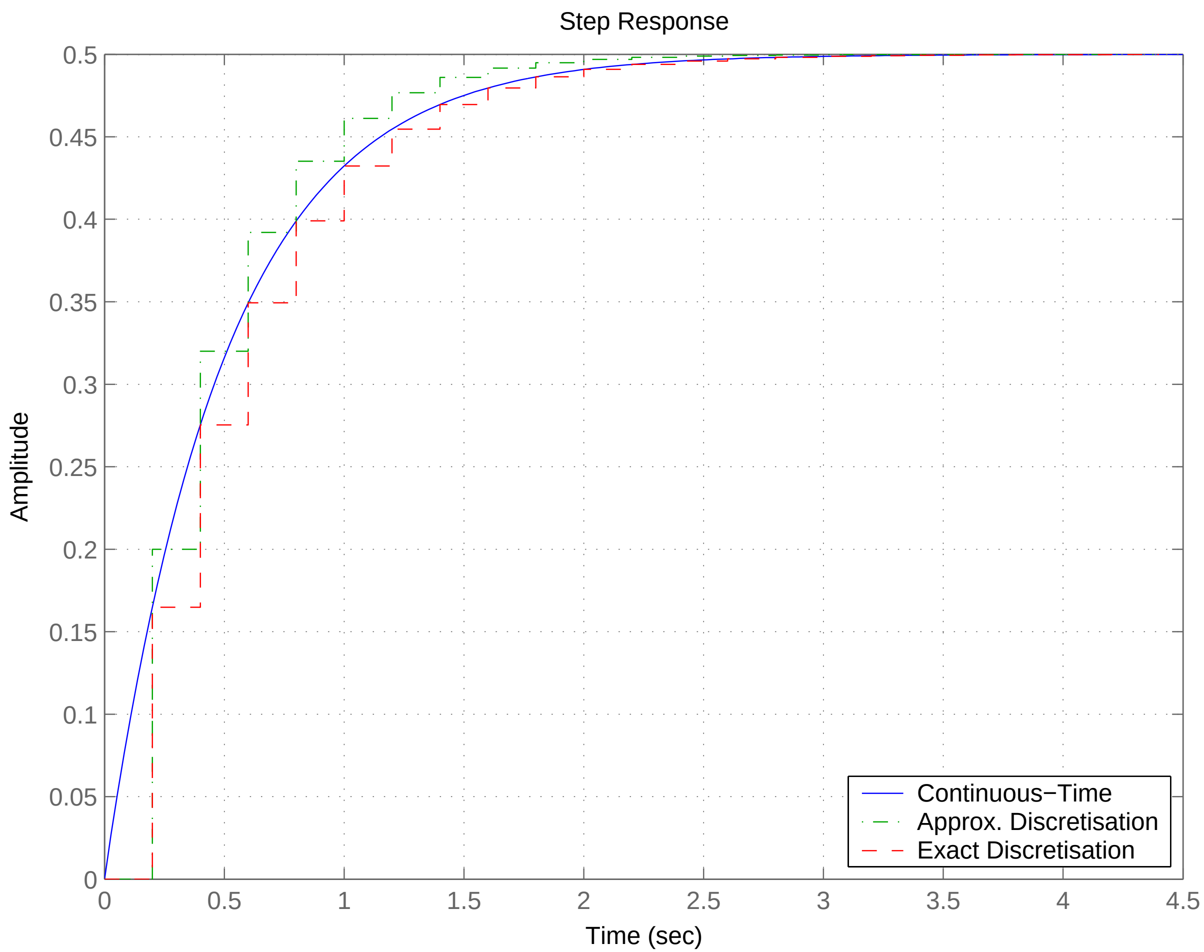
$$x[k+1] = (1-2T)x[k] + Tu[k],$$

while the **exact** discretisation via the PVF yields

$$x[k+1] = e^{-2T}x[k] + \left(\frac{1-e^{-2T}}{2}\right)u[k].$$

Exact Discretisation

Example (continuation). The plot shows the step response of the original continuous-time system, and that of its approximate and exact discretisations with $T = 0.2$.



MATLAB *code to generate the plot*

```
% Discrete.m
% Matlab script for an example to compare exact with approximate
% discretisation
% Sampling time
T=0.2;
% Continuous-time system
G=ss(-2,1,1,0);
% Approximate discretisation
G1=ss((1-2*T),T,1,0,T);
% Exact discretisation
G2=ss(exp(-2*T),(1-exp(-2*T))/2,1,0,T);
% Step responses
step(G,'b',G1,'g-.',G2,'r--')
legend('Continuous-Time','Approx. Discretisation',['Exact ' ...
                    'Discretisation'],4)
hold off
```

Solution of Discrete-Time State Equations

The solution of discrete-time state equations is considerably simpler that that of continuous-time state equations. From

$$\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k]$$

we have

$$\begin{aligned}
\mathbf{x}[1] &= \mathbf{A}\mathbf{x}[0] + \mathbf{B}\mathbf{u}[0] \\
\mathbf{x}[2] &= \mathbf{A}\mathbf{x}[1] + \mathbf{B}\mathbf{u}[1] = \mathbf{A}^2\mathbf{x}[0] + \mathbf{A}\mathbf{B}\mathbf{u}[0] + \mathbf{B}\mathbf{u}[1].
\end{aligned}$$

Solution of Discrete-Time State Equations

The solution of discrete-time state equations is considerably simpler that that of continuous-time state equations. From

$$\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k]$$

we have

$$\begin{aligned}\mathbf{x}[1] &= \mathbf{A}\mathbf{x}[0] + \mathbf{B}\mathbf{u}[0] \\ \mathbf{x}[2] &= \mathbf{A}\mathbf{x}[1] + \mathbf{B}\mathbf{u}[1] = \mathbf{A}^2\mathbf{x}[0] + \mathbf{A}\mathbf{B}\mathbf{u}[0] + \mathbf{B}\mathbf{u}[1].\end{aligned}$$

By proceeding forward we readily obtain, for $k > 0$,

$$\boxed{\mathbf{x}[k] = \mathbf{A}^k\mathbf{x}[0] + \sum_{m=0}^{k-1} \mathbf{A}^{k-1-m}\mathbf{B}\mathbf{u}[m]}$$

$$\boxed{\mathbf{y}[k] = \mathbf{C}\mathbf{A}^k\mathbf{x}[0] + \sum_{m=0}^{k-1} \mathbf{C}\mathbf{A}^{k-1-m}\mathbf{B}\mathbf{u}[m] + \mathbf{D}\mathbf{u}[k]}.$$

Discrete-Time Zero-Input Response

We now discuss the zero-input response $\mathbf{x}[k] = \mathbf{A}^k\mathbf{x}[0]$ of a discrete-time system. Consider a matrix $\mathbf{A}$ whose Jordan form is

$$\bar{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q} = \begin{bmatrix} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{bmatrix}$$

Discrete-Time Zero-Input Response

We now discuss the zero-input response $\mathbf{x}[k] = \mathbf{A}^k \mathbf{x}[0]$ of a discrete-time system. Consider a matrix $\mathbf{A}$ whose Jordan form is

$$\bar{\mathbf{A}} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q} = \begin{bmatrix} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{bmatrix}$$

Then we have that $\bar{\mathbf{A}}^k = \begin{bmatrix} \lambda_1^k & k\lambda_1^{k-1} & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{bmatrix}$.

So the system response to initial conditions

$$\mathbf{x}[k] = \mathbf{A}^k \mathbf{x}(0) = \mathbf{Q}\bar{\mathbf{A}}^k \mathbf{Q}^{-1}\mathbf{x}(0)$$

will be a linear combination of the terms λ_1^k, $k\lambda_1^{k-1}$ and λ_2^k.

Discrete-Time Zero-Input Response

We now discuss the zero-input response $x[k] = A^k x[0]$ of a discrete-time system. Consider a matrix A whose Jordan form is

$$\bar{A} = Q^{-1}AQ = \begin{bmatrix} \lambda_1 & 1 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{bmatrix}$$

Then we have that $\bar{A}^k = \begin{bmatrix} \lambda_1^k & k\lambda_1^{k-1} & 0 \\ 0 & \lambda_1^k & 0 \\ 0 & 0 & \lambda_2^k \end{bmatrix}$.

So the system response to initial conditions

$$x[k] = A^k x(0) = Q\bar{A}^k Q^{-1} x(0)$$

will be a linear combination of the terms λ_1^k, $k\lambda_1^{k-1}$ and λ_2^k.

If all eigenvalues of A are **strictly within the unit circle**, the system response to initial conditions will decay to zero as $t \to \infty$. Otherwise, the response may grow unbounded.

Summary

- We have presented the general solution of the state equation for continuous and discrete-time LTI systems.

Summary

- We have presented the general solution of the state equation for continuous and discrete-time LTI systems.
- In the continuous-time case, the solution is given by the **Parameter Variation Formula**. In the discrete-time case, the solution is a simple recursion based on the powers of $\mathbf{A}$.

Summary

- We have presented the general solution of the state equation for continuous and discrete-time LTI systems.
- In the continuous-time case, the solution is given by the **Parameter Variation Formula**. In the discrete-time case, the solution is a simple recursion based on the powers of A.
- The PVF requires the computation of the **matrix exponential** e^{At}, and we have presented a method based on the Laplace transform:

$$\mathbf{e}^{\mathbf{A}t} = \mathcal{L}^{-1}\left\{(s\mathbf{I} - \mathbf{A})^{-1}\right\}$$

Summary

- We have presented the general solution of the state equation for continuous and discrete-time LTI systems.
- In the continuous-time case, the solution is given by the **Parameter Variation Formula**. In the discrete-time case, the solution is a simple recursion based on the powers of A.
- The PVF requires the computation of the **matrix exponential** $e^{\mathbf{A}t}$, and we have presented a method based on the Laplace transform:

$$\mathbf{e}^{\mathbf{A}\mathbf{t}} = \mathcal{L}^{-1}\left\{(\mathbf{s}\mathbf{I} - \mathbf{A})^{-1}\right\}$$

- We have seen two applications of the PVF to
  - the analysis of the zero-input response of the system
  - the discretisation of a continuous-time LTI systems

Summary

- We have presented the general solution of the state equation for continuous and discrete-time LTI systems.
- In the continuous-time case, the solution is given by the **Parameter Variation Formula**. In the discrete-time case, the solution is a simple recursion based on the powers of A.
- The PVF requires the computation of the **matrix exponential** e^{At}, and we have presented a method based on the Laplace transform:

$$\mathbf{e}^{\mathbf{A}\mathbf{t}} = \mathcal{L}^{-1}\left\{(\mathbf{sI} - \mathbf{A})^{-1}\right\}$$

- We have seen two applications of the PVF to
  - the analysis of the zero-input response of the system
  - the discretisation of a continuous-time LTI systems

Summary

- We have presented the general solution of the state equation for continuous and discrete-time LTI systems.
- In the continuous-time case, the solution is given by the **Parameter Variation Formula**. In the discrete-time case, the solution is a simple recursion based on the powers of A.
- The PVF requires the computation of the **matrix exponential** e^{At}, and we have presented a method based on the Laplace transform:

$$\mathbf{e}^{\mathbf{A}\mathbf{t}} = \mathcal{L}^{-1}\left\{(\mathbf{s}\mathbf{I} - \mathbf{A})^{-1}\right\}$$

- We have seen two applications of the PVF to
  - the analysis of the zero-input response of the system
  - the discretisation of a continuous-time LTI systems