Discrete-time signals and systems

By Finn Haugen, TechTeach
February 17 2005

This document provides a summary of the theory of discrete-time signals and dynamic systems. Whenever a computer is used in measurement, signal processing or control applications, the data (as seen from the computer) and systems involved are naturally discrete-time because a computer executes program code at discrete points of time. Theory of discrete-time dynamic signals and systems is useful in design and analysis of control systems, signal filters, and state estimators, and model estimation from time-series of process data (system identification).

It is assumed that you have basic knowledge about systems theory of continuous-time systems – specifically differential equations, transfer functions, block diagrams, and frequency response.[3]

If you have comments or suggestions for this document please send them via e-mail to finn@techteach.no.

The document may be updated any time. The date (written on the front page) identifies the version. Major updates from earlier versions will be described here.

FinnHaugen

Skien, Norway, February 2005

Contents

1 Introduction

Here is a brief description of the main sections of this document:

- Section 2, *Discrete-time signals*, defines discrete-time signals as sequences of .
- Section 3, *Sampling phenomena*, describes how sampling (in a analog-to-digital or DA-converter) converts a continuous-time signal to a discrete-time signal, resulting in a quantizing error which is a function of the number of bits used in the DA-converter. A particular phenomenon called aliasing is described. Aliasing occurs if the sampling frequency is too small, causing frequency components in the analog signal to appear as a low-frequent discrete-time signal.
- Section 4, *Difference equations*, defines the most basic discrete-time system model type – the difference equation. It plays much the same role for discrete-time systems as differential equations do for continuous-time systems. The section shown how difference equations can be represented using block diagrams.
- Section 5, *The z-transform*, shows how a discrete-time function is transformed to a z-valued function. This transformation is analogous to the Laplace-transform for continuous-time signals. The most important use of the z-transform is for defining z-transfer functions.
- Section 6, *z-transfer functions*, defines the z-transfer function which is a useful model type of discrete-time systems, being analogous to the Laplace-transform for continuous-time systems.
- Section 7, *Frequency response*, shows how the frequency response can be found from the z-transfer function.
- Section 8, *Discretizing continuous-time transfer functions*, explains how you can get a difference equation or a z-transfer function from a differential equation or a Laplace transfer function. Various discretization methods are described.
- Section 9, *State space models*, defines discrete-time state space models, which is just a set of difference equations written in a special way. Discretization of continuous-time state space models into discrete-time state space models is also described.
- Section 10, *Dynamics of discrete-time models*, analyzes the dynamics of basis systems, namely gains, integrators, first order systems and time-delays, with emphasis on the correspondence between pole and step-response.
- Section 11, *Stability analysis*, defines various stability properties of discrete-time systems: Asymptotic stability, marginal stability, and instability, and relates these stability properties to the pole placement in the complex plane. Finally, it is shown how stability analysis of feedback systems (typically control systems) is performed using frequency response based methods on the Nyquist stability criterion.

2 Discrete-time signals

A *discrete-time signal* is a sequence or a series of signal values defined in discrete points of time, see Figure 1. These discrete points of time can be

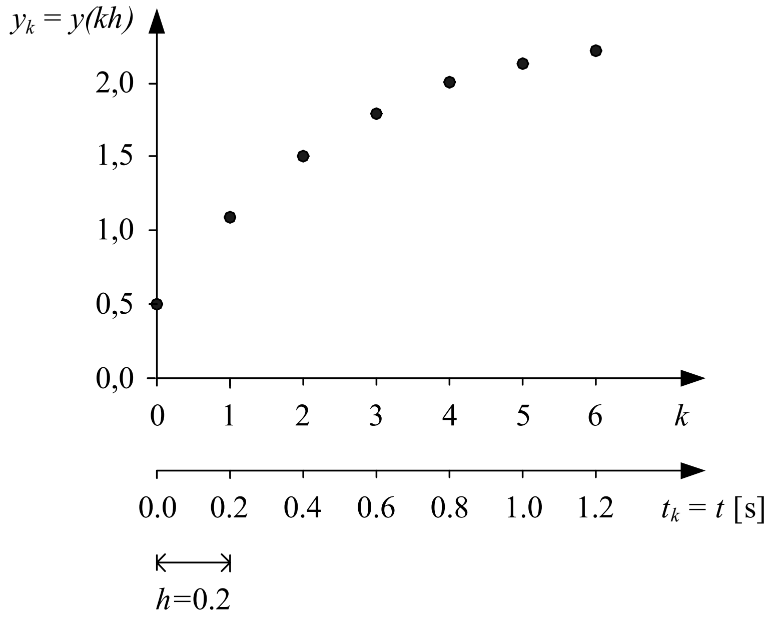

Figure 1: Discrete-time signal

denoted t_k where k is an integer time index. The distance in time between each point of time is the *time-step*, which can be denoted h. Thus,

$$h = t_k - t_{k-1} \tag{1}$$

The time series can be written in various ways:

$$\{x(t_k)\} = \{x(kh)\} = \{x(k)\} = x(0),\ x(1),\ x(2), \ldots \tag{2}$$

To make the notation simple, we can write the signal as $x(t_k)$ or $x(k)$.

Examples of discrete-time signals are logged measurements, the input signal to and the output signal from a signal filter, the control signal to a physical process controlled by a computer, and the simulated response for a dynamic system.

3 Sampling phenomena

3.1 Quantizing

The AD-converter (analog-digital) converts an analog signal $y_a(t)$, which can be a voltage signal from a temperature or speed sensor, to a digital signal,

$y_d(t_k)$, in the form of a number to be used in operations in the computer, see Figure 2. The AD-converter is a part of the interface between the computer

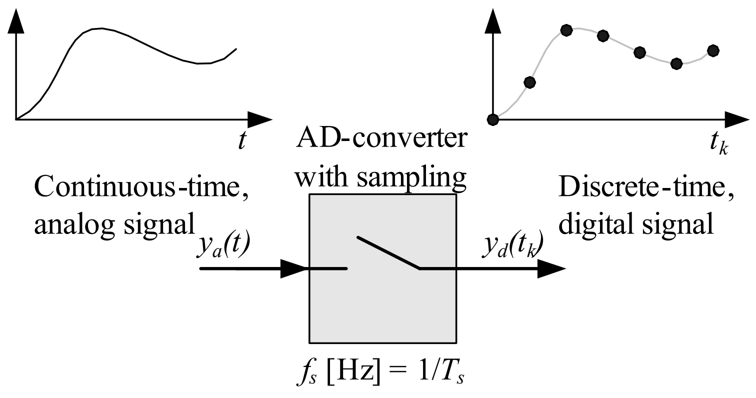
t
AD-converter with sampling
t_k
Continuous-time, analog signal
$y_a(t)$
$y_d(t_k)$
Discrete-time, digital signal
f_s [Hz] = $1/T_s$

Figure 2: Sampling

and the external equipment, e.g. sensors.

The digital signal is represented internally in the computer by a number of bits. One bit is the smallest information storage in a computer. A bit has two possible values which typically are denoted 0 (zero) and 1 (one). Assume that $y_a(t)$ has values in the range $[Y_{\min}, Y_{\max}]$ and that the AD-converter represents y_a in the given range using n bits. Then y_a is converted to a digital signal in the form of a set of bits:

$$y_d \sim b_{n-1}b_{n-2}...b_1b_0 \tag{3}$$

where each bit b_i has value 0 or 1. These bits are interpreted as weights or coefficients in a number with base 2:

$$y_d = b_{n-1}2^{n-1} + b_{n-2}2^{n-2} + ... + b_1 2^1 + b_0 2^0 \tag{4}$$

b_0 is the LSB (least significant bit), while b_{n-1} is the MSB (most significant bit).

Let us see how the special values $Y_{\min}$ and $Y_{\max}$ are represented in the computer. Assume that $n = 12$, which is typical in AD-converters. $y_d = Y_{\min}$ is then represented by

$$\begin{aligned} y_d &= Y\min & (5)\\ &= 0 \cdot 2^{11} + 0 \cdot 2^{10} + ... + 0 \cdot 2^1 + 0 \cdot 2^0 & (6)\\ &= 000000000000_2 = 0_2 & (7)\\ &= 0 \text{ (decimal)} & (8) \end{aligned}$$

where subindex 2 means number base 2. $y_d = Y_{\max}$ is represented by

$$
\begin{aligned}
y_d &= Y\max && (9)\\
&= 1\cdot 2^{11} + 1\cdot 2^{10} + ... + 1\cdot 2^1 + 1\cdot 2^0 && (10)\\
&= 111111111111_2 && (11)\\
&= 1000000000000_2 - 1 && (12)\\
&= 2^{12} - 1 && (13)\\
&= 4095 \text{ (decimal)} && (14)
\end{aligned}
$$

The *resolution* q, which is the interval represented by LSB, is

$$q = \frac{Y_{\max} - Y_{\min}}{\text{Number of intervals}} = \frac{Y_{\max} - Y_{\min}}{2^n - 1} \tag{15}$$

For a 12-bit AD-converter with $Y_{\max} = 10\text{V}$ and $Y_{\min} = 0\text{V}$, the resolution is

$$q = \frac{Y_{\max} - Y_{\min}}{2^n - 1} = \frac{10\text{V} - 0\text{V}}{2^{12} - 1} = \frac{10\text{V}}{4095} = 2.44\text{mV} \tag{16}$$

Variations smaller than the resolution may not be detected at all by the AD-converter.

Figure shows an example of an analog signal and the corresponding quantized signal for $n = 12$ bits and for $n = 4$ bits in the AD-converter. The low resolution is clear with $n = 4$.

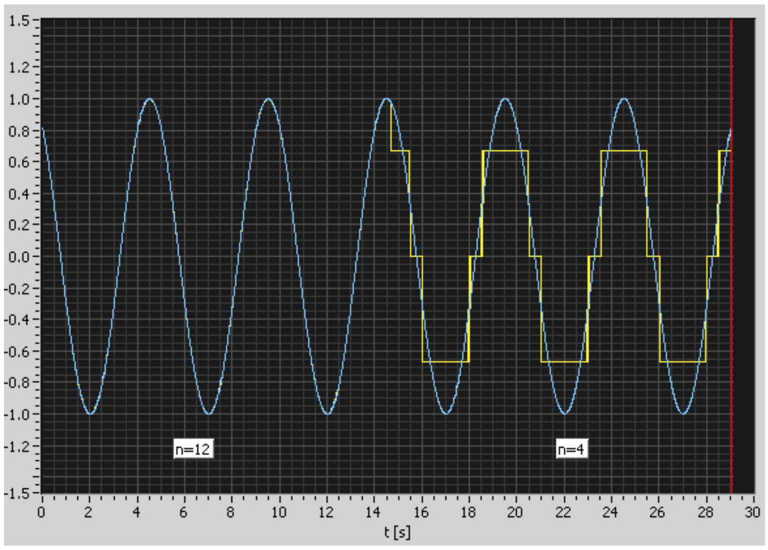

Figure 3: Analog signal and the corresponding quantized signal for $n = 12$ bits (up to 15s) and for $n = 4$ bits (after 15 s) in the AD-converter.

3.2 Aliasing

A particular phenomenon may occur when a continuous-time signal is sampled. Frequency components (i.e. sinusoidal signal components) in the analog signal may appear as a low-frequent sinusoid in the digital signal![1] This phenomenon is called *aliasing*[2], and it appears *if the sampling frequency is too small compared to the frequency of the sampled signal.* Figure 4 shows two examples, one without aliasing and one with aliasing. The sampling interval is different in the two examples. The continuous-time sinusoid in Figure 4 is given by

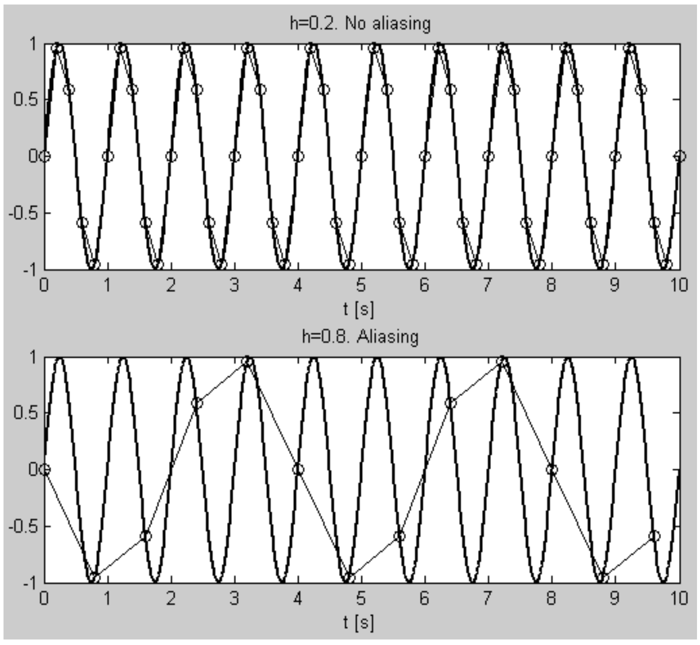

Figure 4: A continuous-time sinusoid $y(t) = \sin 2\pi t$ and the sampled sinusoids for two different sampling intervals

$$y(t) = \sin 2\pi t \tag{17}$$

having signal frequency

$$f_{con} = 1\text{Hz} \tag{18}$$

I have drawn straight lines between the discrete signal values to make these values more visible. The two cases are as follows:

1. Sampling interval $h = 0.2$s corresponding to sampling frequency

$$f_s = \frac{1}{h} = \frac{1}{0.2} = 5\text{Hz} \tag{19}$$

   The discrete signal has the same frequency as the continuous-time signal, see Figure 4. Thus, *there is is no aliasing.*

2. Sampling interval $h = 0.8$s corresponding to sampling frequency

$$f_s = \frac{1}{h} = \frac{1}{0.8} = 1.25\text{Hz} \tag{20}$$

[1] The amplitude is however not changed.
[2] Alias = who uses a false name

The discrete signal has a different (lower) frequency than the continuous-time signal, see Figure 4. Thus, *there is aliasing.*

What are the conditions for aliasing to occur? These conditions will not be derived here, but they are expressed in Figure 5.[3] The figure indicates that

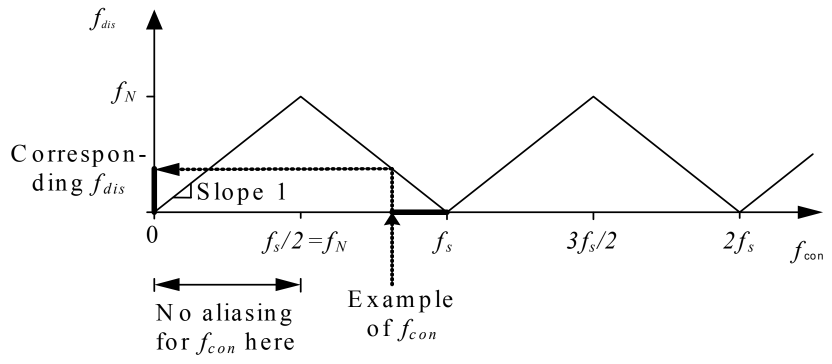

Figure 5: The correspondence between continuous-time signal frequency f_{con} and the sampled discrete-time signal frequency f_{dis}

(only) continuous-time signal frequency components having frequency f_{con} larger than half of the sampling frequency f_s are aliased, and when they are aliased, they appear as low-frequent sinusoids of frequency f_{dis}. Half the sampling frequency is defined as the Nyquist frequency:

$$f_N \overset{\text{def}}{=} \frac{f_s}{2} \tag{21}$$

Using f_N, the condition for aliasing can be stated as follows: *Continuous-time signal frequency components having frequency f_{con} larger than the Nyquist frequency f_N are aliased, and the resulting frequency, f_{dis}, of signals being aliased is found from Figure 5.*

Example 3.1 ***Aliasing***

Let us return to the two cases shown in Figure 4:

1. $h = 0.2$s: The sampling frequency is $f_s = 1/h = 5$Hz. The Nyquist frequency is

$$f_N = \frac{f_s}{2} = \frac{5}{2} = 2.5\text{Hz} \tag{22}$$

The continuous-time signal frequency is

$$f_{con} = 1\text{Hz} \tag{23}$$

Since $f_{con} < f_N$, there is no aliasing.

[3]The figure is inspired by [5].

2. $h = 0.8$s: The sampling frequency is $f_s = 1/h = 1.25$Hz. The Nyquist frequency is
$$f_N = \frac{f_s}{2} = \frac{1.25}{2} = 0.63\text{Hz} \tag{24}$$
The continuous-time signal frequency is
$$f_{con} = 1\text{Hz} \tag{25}$$
Since $f_{con} > f_N$, there is aliasing.

What is the frequency of the resulting signal, f_{dis}? Figure 6 shows how to find f_{dis}. The result is

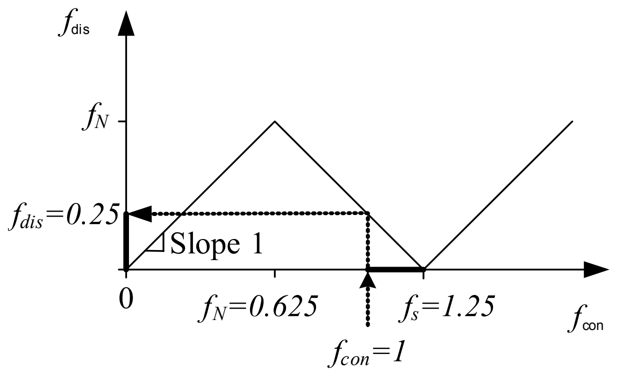

Figure 6: Example 3.1

$$f_{dis} = 0.25\text{Hz} \tag{26}$$
which gives period
$$T_{dis} = \frac{1}{f_{dis}} = \frac{1}{0.25\text{Hz}} = 4\text{s} \tag{27}$$

The above results are in accordance with the observations in Figure 4.

[End of Example 3.1]

Aliased frequency components may cause problems in audio applications and control applications. One way of reducing the aliasing problem is to pass the continuous-time signal through a lowpass filter before sampling. This anti-aliasing filter should be an analog filter which may be implemented using electronic components as resistors, capacitors and operational amplifiers.

4 Difference equations

4.1 Difference equation models

The basic model type of continuous-time dynamic systems is the differential equation. Analogously, the basis model type of discrete-time dynamic systems is the *difference equation*. Here is an example of a linear second order difference equation with u as input variable and y as output variable:

$$y(k) = -a_1 y(k-1) - a_0 y(k-2) + b_1 u(k-1) + b_0 u(k-2) \tag{28}$$

where a_i and b_j are *coefficients* of the difference equation, or model *parameters*. We can say that this difference equation is normalized since the coefficient of $y(k)$ is 1 (the other coefficients then have unique values).

The difference equation (28) may be written in other equivalent forms. One equivalent form is

$$y(k+2) + a_1 y(k+1) + a_0 y(k) = b_1 u(k+1) + b_0 u(k) \tag{29}$$

where there are no time delayed terms, only time advanced terms (or terms without any advance or delay). This form can be achieved by increasing each time index in (28) by 2.

Example 4.1 ***A lowpass filter as a difference equation***

The following difference equation implements a discrete-time lowpass filter. a is a filter parameter.

$$y(k) = ay(k-1) + (1-a)\,u(k) \tag{30}$$

[End of Example 4.1]

Example 4.2 ***A PI controller as a difference equation***

The following difference equation implements a discrete-time PI (proportional+integral) controller. K_p and T_i are controller parameters.

$$u(k) = u(k-1) + K_p\left(1 + \frac{h}{T_i}\right)e(k) - K_p e(k-1) \tag{31}$$

where u is the control signal generated by the controller, and e is the control error (which is the difference between the setpoint and the process measurement). K_p and T_i are controller parameters, and h is the sampling interval.

[End of Example 4.1]

Example 4.3 ***A simulation algorithm as a difference equation***

Given the following model of a first order system in the form of a differential equation:

$$\dot{y}(t) = -\frac{1}{T}y(t) + \frac{K}{T}u(t) \tag{32}$$

u is the input, y the output, K the gain and T the time constant. Applying the Euler forward method for numerical solution of this differential equation yields the following difference equation:

$$y(k) = \left(1 - \frac{h}{T}\right)y(k-1) + \frac{Kh}{T}u(k-1) \tag{33}$$

where h is the simulation time-step.

[End of Example 4.3]

4.2 Calculating responses

4.2.1 Calculating dynamic responses for difference equations

A difference equation is actually itself an *algorithm* or formula for calculating responses in the form of time functions.

Example 4.4 ***Calculating the dynamic responses for a difference equation***

See Example 4.3. Assume the following parameter values:

$$h = 0.1;\ T = 1;\ K = 2 \tag{34}$$

The difference equation becomes

$$\begin{aligned} y(k) &= \left(1 - \frac{0.1}{1}\right) y(k-1) + \frac{2 \cdot 0.1}{1} u(k-1) \qquad (35) \\ &= 0.9y(k-1) + 0.2u(k-1) \qquad (36) \end{aligned}$$

Assume that u is a step of amplitude U at discrete time $k = 0$, and that the initial value of y is y_0. From (36) we can calculate the first two response in y as follows:

$$\begin{aligned} y(1) &= 0.9y(0) + 0.2u(0) \qquad (37) \\ &= 0.9y_0 + 0.2U \qquad (38) \end{aligned}$$

$$\begin{aligned} y(2) &= 0.9y(1) + 0.2u(1) \qquad (39) \\ &= 0.9\left[0.9y_0 + 0.2U\right] + 0.2 \cdot 0 \qquad (40) \\ &= 0.81y_0 + 0.18U \qquad (41) \end{aligned}$$

[End of Example 4.3]

4.2.2 Calculating static responses for difference equation

By *static response* we mean the constant steady-state value of the output variable of the model when the input variables have constant values. The static response can be calculated from the static version of the difference equation. The static version is found by neglecting all time-dependencies in the differential equation. For example, a term as $y(k-1)$ is replaced by y_s where subindex s is for static.

Example 4.5 ***Calculating static response for a difference equation***

Eq. (30) in Example 4.1 is a lowpass filtering algorithm. It is repeated here for convenience:

$$y(k) = ay(k-1) + (1-a)\,u(k) \tag{42}$$

Let us calculate the static response in the filter output y. The input u is assumed to be a constant of amplitude U. The static version of the difference equation (42) is

$$\begin{aligned} y_s &= a y_s + (1-a)\,u_s && (43)\\ &= a y_s + (1-a)\,U && (44)\end{aligned}$$

Solving for y_s yields

$$y_s = \frac{(1-a)\,U}{1-a} = U \tag{45}$$

(So, the output is equal to the input, statically. This is to be expected for a lowpass filter.)

[End of Example 4.5]

4.3 Block diagram of difference equation models

A *block diagram* gives a graphical representation of a mathematical model. The block diagram shows the structure of the model, e.g. how subsystems are connected. Furthermore, block diagram models can be represented directly in graphical simulation tools such as SIMULINK and LabVIEW.

Figure 7 shows the most frequently used blocks – or the *elementary blocks* – used in block diagrams of difference equation models.

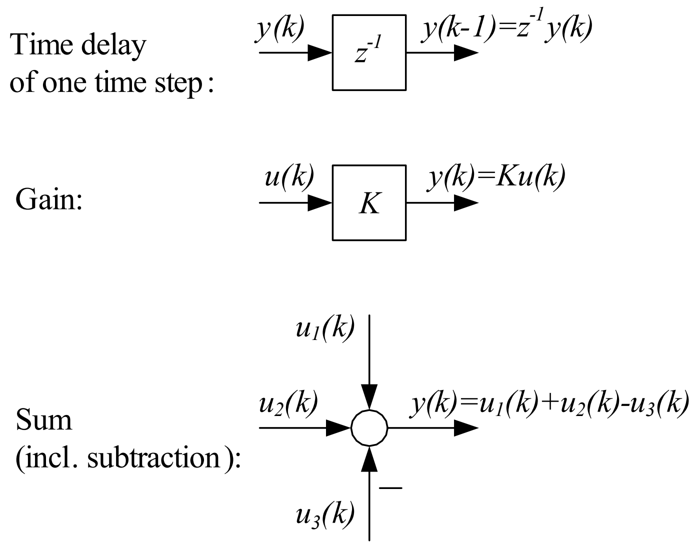

Figure 7: Elementary blocks for drawing block diagrams of difference equation models

A comment about the time delay block: The output $y(k)$ is equal to the time delayed input, $y(k-1)$:

$$y(k-1) = z^{-1} y(k) \tag{46}$$

The operator z^{-1} is here a time-step delay operator, and it can be regarded as an operator of the time-step delay, cf. Section A. (z^{-1} is also the transfer function of a time-step delay, cf. Section 6.7.)

Example 4.6 ***Block diagram of a difference equation***

Eq. (30) in Example 4.1 is a lowpass filtering algorithm. It is repeated here:

$$y(k) = ay(k-1) + (1-a)\,u(k) \tag{47}$$

Using the elementary blocks shown in Figure 7 a block diagram of this difference equation can be drawn as shown in Figure 8.

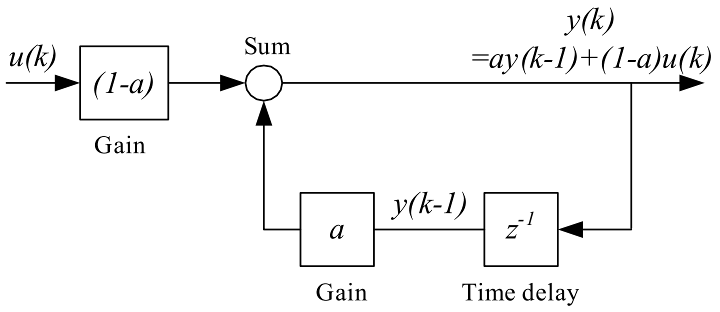

Figure 8: The block diagram corresponding to (47)

[End of Example 4.6]

5 The z-transform

5.1 Definition of the z-transform

The z-transform of discrete-time signals plays much the same role as the Laplace transform for continuous-time systems.

The z-transform of the discrete-time signal $\{y(k)\}$, or just $y(k)$, is defined as follows:

$$\mathcal{Z}\{y(k)\} = \sum_{k=0}^{\infty} y(k)z^{-k} \tag{48}$$

For simplicity, I will use the symbol $y(z)$ for $\mathcal{Z}\{y(k)\}$ when it can not be misunderstood. Strictly, a different variable name must be used, for example $Y(z)$.

Example 5.1 ***z-transform of a constant***

Assume that the signal $y(k)$ has constant value A. This signal can be regarded a step of amplitude A at time-step 0. z-transforming $y(k)$ gives

$$y(z) = \sum_{k=0}^{\infty} y(k)z^{-k} = \sum_{k=0}^{\infty} Az^{-k} = \frac{A}{1-z^{-1}} = \frac{Az}{z-1} \tag{49}$$

[End of example 5.1]

5.2 Properties of the z-transform

Below are the most important properties of the z-transform. These properties can be used when calculating the z-transform of composite signals. A more complete list of z-transform properties are shown in Appendix A.1.

- **Linearity:**
$$k_1y_1(z) + k_2y_2(z) \Longleftrightarrow k_1y_1(k) + k_2y_2(k) \tag{50}$$
- **Time delay:** Multiplication by z^{-n} means time delay of n time-steps:
$$z^{-n}y(z) \Longleftrightarrow y(k-n) \tag{51}$$
- **Time advancing**: Multiplication by z^n means time advancing by n time-steps:
$$z^n y(z) \Longleftrightarrow y(k+n) \tag{52}$$

Example 5.2 ***z-transformation of a composite signal***

Given the following discrete-time function:

$$y(k) = Ba^{k-n} \tag{53}$$

(which is a time delayed time exponential). The inverse z-transform of $y(k)$ can be calculated using (353) together with (50) and (51). The result becomes

$$y(z) = Bz^{-n}\frac{z}{z-a} = B\frac{z^{1-n}}{z-a} \tag{54}$$

[End of example 5.2]

5.3 Inverse transform

Inverse z-transformation of a given z evaluated function, say $Y(z)$, is calculating the corresponding time function, say $y(k)$. The inverse transform may be calculated using a complex integral[4], but this method is not very practical. Another method is to find a proper combination of the z-transformation pairs listed in Appendix A.2, possibly in combination with some of the z-transform properties in Appendix A.1.

In most cases where you need to calculate a time signal $y(k)$, its z-transform $Y(z)$ stems from a transfer function excited by some discrete-time input signal. You may then calculate $y(k)$ by first transferring the transfer function to a corresponding difference equation, and then calculating $y(k)$ iteratively from this difference equation as explained in 4.2.

[4] $y(k) = \frac{1}{2\pi j}\oint Y(z)z^k\frac{dz}{z}$, where the integration path must be in the area of convergence of $Y(z)$.[1]

6 z-transfer functions

6.1 Introduction

Models in the form of difference equations can be z-transformed to z-transfer *functions*, which plays the same role in discrete-time systems theory as s transfer functions do in continuous-time systems theory. More specific:

- The combined model of systems in a serial connection can be found my simply multiplying the individual z-transfer functions.
- The frequency response can be calculated from the transfer function.
- The transfer function can be used to represent the system in a simulator or in computer tools for analysis and design (as SIMULINK, MATLAB or LabVIEW)

6.2 How to calculate z-transfer functions

As an example we will calculate the z-transfer function from input u to output y for the difference equation (28), which is repeated here:

$$y(k) = -a_1 y(k-1) - a_0 y(k-2) + b_1 u(k-1) + b_0 u(k-2) \tag{55}$$

First we take the z-transform of both sides of the difference equation:

$$\mathcal{Z}\{y(k)\} = \mathcal{Z}\{-a_1 y(k-1) - a_0 y(k-2) + b_1 u(k-1) + b_0 u(k-2)\} \tag{56}$$

Using the linearity property (50) and the time delay property (51) (56) can be written as

$$\mathcal{Z}\{y(k)\} = -\mathcal{Z}\{a_1 y(k-1)\} - \mathcal{Z}\{a_0 y(k-2)\} + \mathcal{Z}\{b_1 u(k-1)\} + \mathcal{Z}\{b_0 u(k-2)\} \tag{57}$$

and

$$y(z) = -a_1 z^{-1} y(z) - a_0 z^{-2} y(z) + b_1 z^{-1} u(z) + b_0 z^{-2} u(z) \tag{58}$$

which can be written as

$$y(z) + a_1 z^{-1} y(z) + a_0 z^{-2} y(z) = b_1 z^{-1} u(z) + b_0 z^{-2} u(z) \tag{59}$$

or

$$\left[1 + a_1 z^{-1} + a_0 z^{-2}\right] y(z) = \left[b_1 z^{-1} + b_0 z^{-2}\right] u(z) \tag{60}$$

$$y(z) = \underbrace{\frac{b_1 z^{-1} + b_0 z^{-2}}{1 + a_1 z^{-1} + a_0 z^{-2}}}_{H(z)} u(z) \tag{61}$$

$$\underbrace{\frac{b_1 z + b_0}{z^2 + a_1 z^1 + a_0}}_{H(z)} u(z) \tag{62}$$

where $H(z)$ is the z-transfer function from u to y:

$$H(z) = \frac{y(z)}{u(z)} = \frac{b_1 z^{-1} + b_0 z^{-2}}{1 + a_1 z^{-1} + a_0 z^{-2}} = \frac{b_1 z + b_0}{z^2 + a_1 z^1 + a_0} \tag{63}$$

Hence, z-transfer functions can be written both with positive and negative exponents of z.[5]

Example 6.1 ***Calculating the transfer function***

Eq. (30) in Example 4.1 is a lowpass filtering algorithm. It is repeated here:

$$y(k) = ay(k-1) + (1-a)\,u(k) \tag{64}$$

Let us calculate the transfer function from u to y. Taking the z-transform of (64) gives

$$y(z) = az^{-1}y(z) + (1-a)\,u(z) \tag{65}$$

which can be written as

$$y(z) = \underbrace{\frac{1-a}{1-az^{-1}}}_{H(z)} u(z) \tag{66}$$

$$= \underbrace{\frac{(1-a)\,z}{z-a}}_{H(z)} u(z) \tag{67}$$

where $H(z)$ is the transfer function (written in two ways).

[End of Example 6.1]

The following example shows that the transfer function is the same as the z-transformed impulse response.

Example 6.2 ***Transfer function is the z-transformed impulse response***

Given a system with transfer function $H(z)$. Assume that the input u is an impulse, which is a signal having value 1 at time index $k = 0$ and value zero for other points of time. According to (351) $u(z) = 1$. Then the z-transformed impulse response is

$$y(z) = H(z)u(z) = H(z) \cdot 1 = H(z) \tag{68}$$

(as stated).

[End of example 6.2]

[5] Within the area of signal processing transfer functions with negative exponents of z are common, while in control theory transfer functions with positive exponents are more common.

6.3 From z-transfer function to difference equation

In the above Section we derived a z-transfer function from a difference equation. We may go the opposite way, too, to derive a difference equation from a given z-transfer function. Some applications of this are

- Deriving a filtering algorithm from a filtering transfer function
- Deriving a control function from a given controller transfer function
- Deriving a simulation algorithm from the transfer function of the system to be simulated

The procedure will be illustrated via a concrete example. Assume given the following transfer function:

$$H(z) = \frac{b_1 z + b_0}{z^2 + a_1 z + a_0} = \frac{y(z)}{u(z)} \tag{69}$$

We start by cross multiplying (69):

$$\left(z^2 + a_1 z + a_0\right) y(z) = (b_1 z + b_0)\, u(z) \tag{70}$$

which can be written as

$$z^2 y(z) + a_1 z y(z) + a_0 y(z) = b_1 z u(z) + b_0 u(z) \tag{71}$$

Taking the inverse transform of the above expression gives

$$\underbrace{z^2 y(z)}_{y(k+2)} + \underbrace{a_1 z y(z)}_{a_1 y(k+1)} + \underbrace{a_0 y(z)}_{a_0 y(k)} = \underbrace{b_1 z u(z)}_{b_1 u(k+1)} + \underbrace{b_0 u(z)}_{b_0 u(k)} \tag{72}$$

Reducing each of the time indexes by 2 yields

$$y(k) + a_1 y(k-1) + a_0 y(k-2) = b_1 u(k-1) + b_0 u(k-2) \tag{73}$$

Usually it is practical to have the output variable alone on the left side:

$$y(k) = -a_1 y(k-1) - a_0 y(k-2) + b_1 u(k-1) + b_0 u(k-2) \tag{74}$$

This difference equation can be used as a simulation algorithm, cf. Section 4.2.

6.4 Poles and zeros

Poles and *zeros* of z-transfer functions are defined in the same way as for s transfer functions: The zeros of the transfer function are the z-roots of numerator polynomial, and the poles are the z-roots of the denominator polynomial. The poles determine the stability property of the system, as explained in Section 11.2.

Example 6.3 ***Poles and zeros***

Given the following z-transfer function:

$$H(z) = \frac{(z-b)}{(z-a_1)(z-a_2)} \tag{75}$$

The poles are a_1 and a_2, and the zero is b.

[End of Example 6.3]

6.5 Calculating time responses in z-transfer function models

Assume given a transfer function, say $H(z)$, with input variable u and output variable y. Then,

$$y(z) = H(z)u(z) \tag{76}$$

If $u(z)$ is given, the corresponding time response in y can be calculated in several ways:

1. By finding a proper transformation pair in Appendix A.2, possibly combined with some of the z-transform properties in Appendix A.1.
2. By deriving a differential equation corresponding to the transfer function and then calculating $y(k)$ iteratively according to the difference equation. The procedure of deriving a differential equation corresponding to a given transfer function is explained in Section 6.3, and the calculation of time responses for s difference equation is described in Section 4.2.

6.6 Static transfer function and static response

The static version H_s of a given transfer function $H(z)$ will now be derived. Using the static transfer function the static response can easily be calculated. Assume that the input variable u is a step of amplitude U. The stationary response can be found using the final value theorem:

$$\begin{aligned}
\lim_{k\to\infty} y(k) = y_s &= \lim_{z\to 1}(z-1)y(z) && (77)\\
&= \lim_{z\to 1}(z-1)H(z)u(z) && (78)\\
&= \lim_{z\to 1}(z-1)H(z)\frac{zU}{z-1} && (79)\\
&= H(1)U && (80)
\end{aligned}$$

Thus, we have the following static transfer function:

$$H_s = \lim_{z\to 1} H(z) = H(1) \tag{81}$$

Using the static transfer function the static response can be calculated by

$$y_s = H_s U \tag{82}$$

Example 6.4 ***Static transfer function and static response***

Eq. (66) defines the transfer function of a given lowpass filter. The transfer function is repeated here:

$$H(z) = \frac{y(z)}{u(z)} = \frac{1-a}{1-az^{-1}} \tag{83}$$

The corresponding static transfer function is

$$H_s = \lim_{z \to 1} H(z) = \lim_{z \to 1} \frac{1-a}{1-az^{-1}} = \frac{1-a}{1-a \cdot 1} = 1 \tag{84}$$

Assume that the input is U (constant). The corresponding static response in y is

$$y_s = H(1) \cdot U = U \tag{85}$$

(which is the same as calculated using the static version of the filter difference equation (42) in Example 4.5).

[End of Example 6.4]

6.7 Block diagrams

6.7.1 Basic blocks

Block diagrams of z-transfer function models can be drawn using the same elementary blocks as for difference equation models, see Figure 7, except that the signals are z-transformed. In addition to these elementary blocks, a block containing any transfer function can be used, see Figure 9.

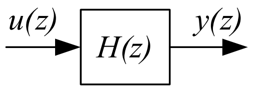

Figure 9: A block containing the transfer function $H(z)$

6.7.2 Block diagram manipulation

The most common rules for block diagram manipulation are shown (and proven) in Figure 10. The rules are the same as for Laplace transfer functions.

Example 6.5 ***Block diagram manipulation***

Figure 11 shows a block diagram of a feedback control system for a physical process. The blocks contains z-transfer functions (however unspecified in this example). The process can be for example a liquid tank where the level is to be controlled, or a motor where the speed is to be controlled.

The transfer function $H_{y_{SP},y}(s)$ from setpoint y_{SP} to process output variable y in the block diagram shown in Figure 11 can be found by simplifying the block diagram until it consists of one block, and the transfer function is then the contents of this single block. Figure 12 shows the individual steps of the block diagram manipulation. The transfer function becomes

$$H_{y_{SP},y}(z) = \frac{y(z)}{y_{SP}(z)} = \frac{H_{sm}(z)H_c(z)H_u(z)}{1 + H_s(z)H_c(z)H_u(z)} \tag{86}$$

[End of Example 6.5]

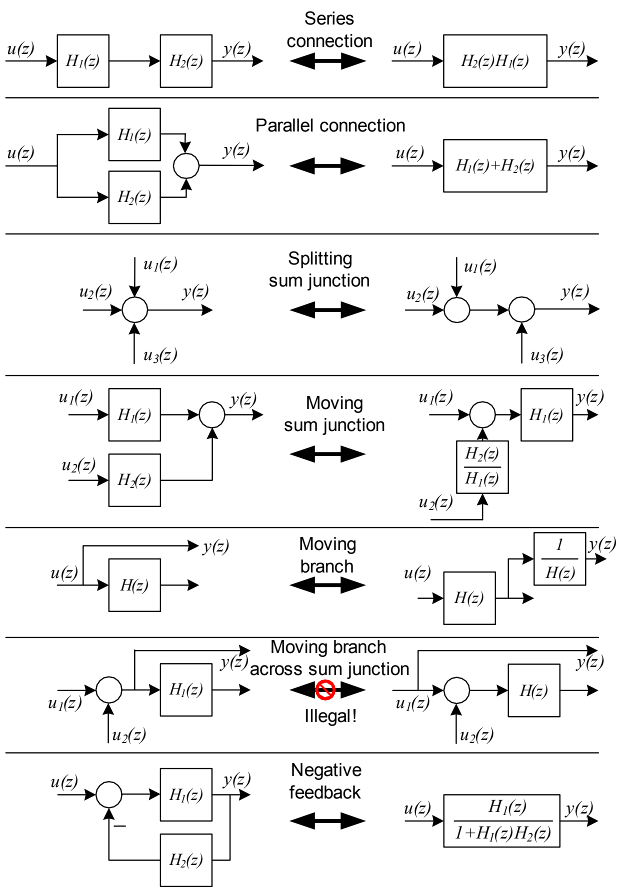

Figure 10: Rules for block diagram manipulation

6.7.3 Calculating the transfer function from the block diagram without block diagram manipulation

You do not have to manipulate the block diagram to calculate the transfer function from it. You can just write down the proper relations or equations from the block diagram, and then calculate the transfer function from these equations directly.

Example 6.6 ***Calculating the transfer function without block diagram manipulation***

See Example 6.5. From the block diagram in Figure 11 we can write down the following equation for $y(z)$:

$$y(z) = H_u(z)H_c(z)\left[H_{sm}(z)y_{SP}(z) - H_s(z)y(z)\right]$$

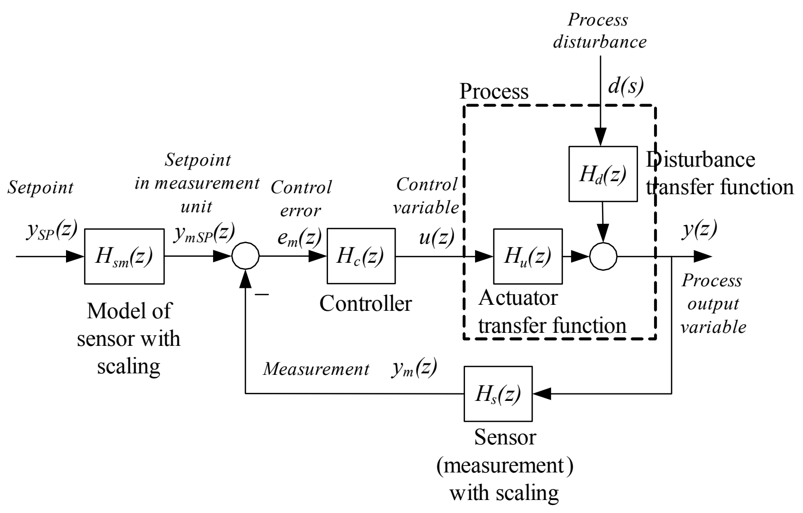

Figure 11: Block diagram of a feedback control system for a physical process. The subsystems are represented by transfer function blocks.

Solving this expression for $y(z)$ yields

$$y(z) = \underbrace{\frac{H_{sm}(z)H_c(z)H_u(z)}{1 + H_s(z)H_c(z)H_u(z)}}_{=H_{y_{SP},y}(z)} y_{SP}(z) \tag{87}$$

which gives the same transfer function $H_{y_{SP},y}(s)$ as found by block diagram manipulation in Example 6.5.

[End of Example 6.6]

7 Frequency response

7.1 Calculating frequency response from transfer function

As for continuous-time systems, the frequency response of a discrete-time system can be calculated from the transfer function: Given a system with z-transfer function $H(z)$. Assume that input signal exciting the system is the sinusoid

$$u(t_k) = U \sin(\omega t_k) = U \sin(\omega k h) \tag{88}$$

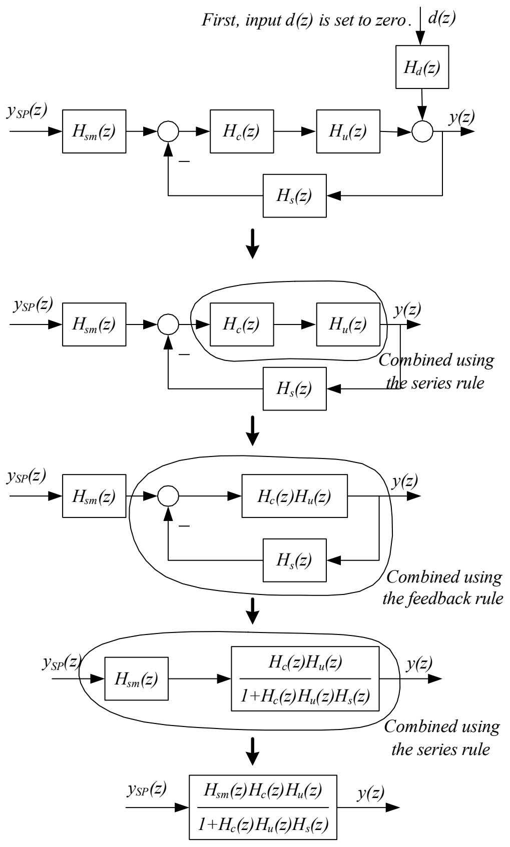
First, input d(z) is set to zero.
d(z)
H_d(z)
y_SP(z)
H_sm(z)
H_c(z)
H_u(z)
y(z)
H_s(z)
−
Combined using the series rule
H_c(z)H_u(z)
Combined using the feedback rule
H_c(z)H_u(z) / 1+H_c(z)H_u(z)H_s(z)
Combined using the series rule
H_sm(z)H_c(z)H_u(z) / 1+H_c(z)H_u(z)H_s(z)

Figure 12: Manipulating a block diagram to find the transfer function from y_{SP} to y

where ω is the signal frequency in rad/s. It can be shown that the stationary response on the output of the system is

$$\begin{aligned} y(t_k) &= Y\sin(\omega kh + \phi) && (89)\\ &= UA\sin(\omega kh+\phi) && (90)\\ &= \underbrace{U\overbrace{\left|H(e^{j\omega h})\right|}^{A}}_{Y}\sin\left[\omega t_k + \underbrace{\arg H(e^{j\omega h})}_{\phi}\right] && (91)\end{aligned}$$

where $H(e^{j\omega h})$ is the *frequency response* which is calculated with the following substitution:

$$H(e^{j\omega h}) = H(z)|_{z=e^{j\omega h}} \quad (92)$$

where h is the time-step. The *amplitude gain function* is

$$A(\omega) = |H(e^{j\omega h})| \quad (93)$$

The *phase lag function* is

$$\phi(\omega) = \arg H(e^{j\omega h}) \quad (94)$$

$A(\omega)$ and $\phi(\omega)$ can be plotted in a Bode diagram.

Figure 13 shows as an example the Bode plot of the frequency response of the following transfer function (time-step is 0.1s):

$$H(z) = \frac{b}{z-a} = \frac{0,0952}{z-0,9048} \tag{95}$$

Note that the plots in Figure 13 are drawn only up to the Nyquist frequency which in this case is

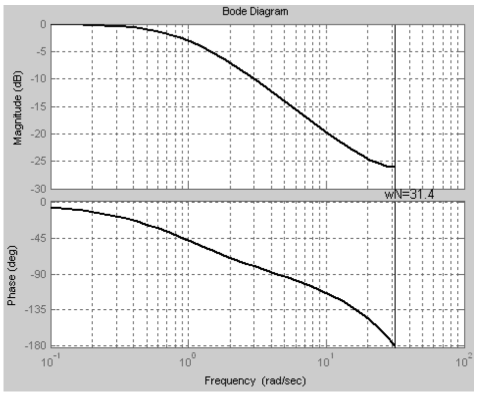

Figure 13: Bode plot of the transfer function (95). $\omega_N = 31.4$ rad/s is the Nyquist frequency (sampling time h is 0.1s).

$$\omega_N = \frac{\omega_s}{2} = \frac{2\pi/h}{2} = \frac{\pi}{h} = \frac{\pi}{0.1} = 10\pi \approx 31.4\text{rad/s} \tag{96}$$

The plots are not drawn (but they exist!) above the Nyquist frequency because of symmetry of the frequency response, as explained in the following section.

Example 7.1 ***Calculating the frequency response manually from the z-transfer function***

Given the z-transfer function

$$H(z) = \frac{b}{z-a} \tag{97}$$

The frequency response becomes

$$H(e^{j\omega h}) = \frac{b}{e^{j\omega h} - a} \tag{98}$$

$$= \frac{b}{\cos\omega h + j\sin\omega h - a} \tag{99}$$

$$= \frac{b}{\underbrace{(\cos\omega h - a)}_{\text{Re}} + j\underbrace{\sin\omega h}_{\text{Im}}} \tag{100}$$

$$= \frac{b}{\sqrt{(\cos\omega h - a)^2 + (\sin\omega h)^2}e^{j\arctan[(\sin\omega h)/(\cos\omega h - a)]}} \tag{101}$$

$$= \frac{b}{\sqrt{(\cos\omega h - a)^2 + (\sin\omega h)^2}}e^{j\left[-\arctan\left(\frac{\sin\omega h}{\cos\omega h - a}\right)\right]} \tag{102}$$

The amplitude gain function is

$$A(\omega) = |H(e^{j\omega h})| = \frac{b}{\sqrt{(\cos\omega h - a)^2 + (\sin\omega h)^2}} \tag{103}$$

and the phase lag function is

$$\phi(\omega) = \arg H(e^{j\omega h}) = -\arctan\left(\frac{\sin\omega h}{\cos\omega h - a}\right) \text{ [rad]} \tag{104}$$

[End of Example 7.1]

Even for the simple example above, the calculations are cumbersome, and prone to errors. Therefore you should use some computer tool for calculating the frequency response, as MATLAB's Control System Toolbox or LabVIEW's Control Design Toolkit.

7.2 Symmetry of frequency response

It can be shown that the frequency response is symmetric as follows: $|H(e^{j\omega h})|$ and $\arg H(e^{j\omega h})$ are unique functions in the frequency interval $[0, \omega_N]$ where ω_N is the Nyquist frequency.[6] In the following intervals $[m\omega_s, (m+1)\omega_s]$ (m is an integer) the functions are mirrored as indicated in Figure 14 which has a logarithmic frequency axis. (The Bode plots in this section are for the transfer function (95).) The symmetry appears clearer in the Bode plots in Figure 15 where the frequency axis is linear.

Due to the symmetry of the frequency response, it is strictly not necessary to draw more of frequency response plots than of the frequency interval $[0, \omega_N]$.

[6] The symmetry is related to aliasing, cf. Section 3.2.

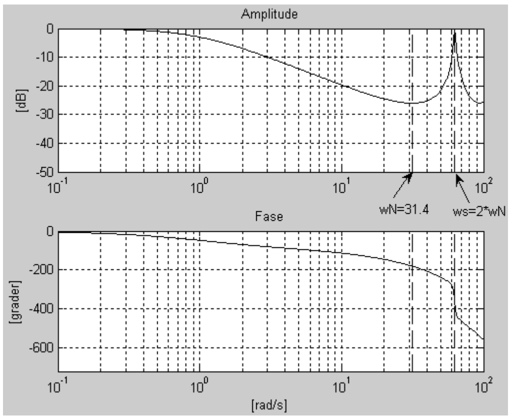

Figure 14: Bode plots of frequency response of (63). The frequency axis is logarithmic.

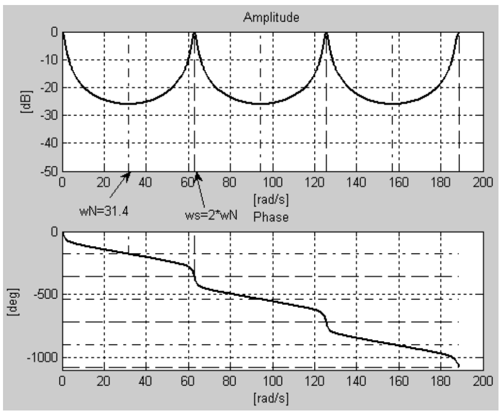

Figure 15: Bode plots of frequency response of (63). The frequency axis is linear to make the symmetries if the frequency responses clearer.

8 Discretizing continuous-time transfer functions

8.1 Introduction

In some cases you need to find a discrete-time z-transfer function from a given continuous-time s transfer function:

- In accurate model based design of a discrete controller for a process originally in the form of a continuous-time s transfer function, $H_p(s)$. The latter should be discretized to get a discrete-time process model before the design is started.
- Implementation of continuous-time control and filtering functions in a

computer program.

There are several methods for discretization of an s transfer function. The methods can be categorized as follows, and they are described in the following sections:

1. Discretization based on having a *zero order hold element* on the input of the system. This method should be used in controller design of a process which has a sample and hold element on its input, as when a physical process is controlled by a computer via a DA converter (digital to analog). Zero order hold means that the input signal is held constant during the time-step or sampling interval. Figure 16 shows a block diagram of a continuous-time process with transfer function model $G(s)$ having a zero order hold element on its input.

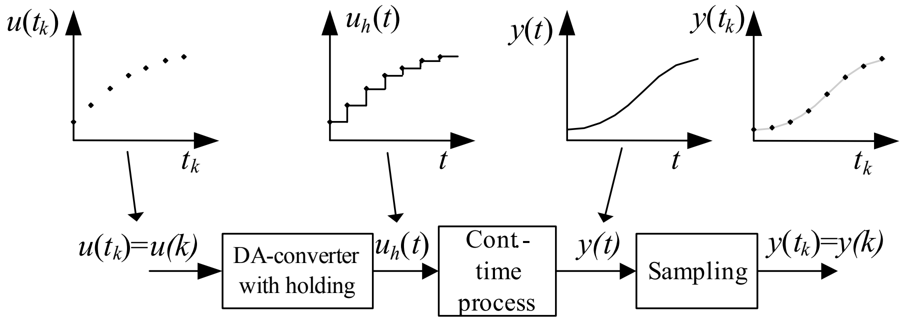

Figure 16: Block diagram of a process having a zero order hold element on its input. u_h is the piecewise constant (held) input signal.

   The discretization can be realized in several ways:

   (a) By calculating the z-transfer function for $G(s)$ with a zero order hold element on its input.

   (b) By first deriving a canonical continuous-time state space model corresponding to $G(s)$, and then discretizing this state space model assuming a hold element on its input.

2. Using an integral numerical approximation[7], typically Euler's forward method, Euler's backward method or Tustin's method. These methods should be used when a continuous-time controller transfer function or filter transfer function are discretized to get an algorithm ready for programming. In such cases the input signal is a discrete-time signal with no holding. The discretization is based on some numerical method so that the behaviour of the discrete-time transfer function is similar to that of the continuous-time transfer function.

[7] Alternatively, differential approximations may be used as the basis for the derivation of the discretization procedures.

8.2 Discretization with zero order hold element on the input

8.2.1 Discretization involving the inverse Laplace transform

Below are two alternative (but equivalent) discretization formulas:

$$H(z) = \mathcal{Z}\left[\mathcal{L}^{-1}\left\{ \frac{G(s)}{s}\left(1 - e^{-hs}\right) \right\}\Big|_{t=kh} \right] \tag{105}$$

and

$$H(z) = \left(1 - z^{-1}\right)\mathcal{Z}\left[\mathcal{L}^{-1}\left\{ \frac{G(s)}{s} \right\}\Big|_{t=kh} \right] \tag{106}$$

where h is the sampling interval (or time-step). These formulas may be derived by first assuming a discrete-time impulse in $u(k)$ and calculating the impulse response in $y(k)$, and then calculating the transfer function as the z transform of this transfer function, cf. Example 6.2.

Example 8.1 ***Discretizing a first order system***

We will discretize the following continuous-time transfer function having a zero order hold element at its input:

$$G(s) = \frac{K}{Ts+1} = \frac{\frac{K}{T}}{s + \frac{1}{T}} \tag{107}$$

(106) becomes

$$H(z) = \left(1 - z^{-1}\right)\mathcal{Z}\left[\mathcal{L}^{-1}\left\{ \frac{\frac{K}{T}}{(s + \frac{1}{T})s} \right\}\Big|_{t=kh} \right] \tag{108}$$

Here

$$\mathcal{L}^{-1}\left\{ \frac{\frac{K}{T}}{(s + \frac{1}{T})s} \right\}\Big|_{t=kh} = K\left(1 - e^{-t/T}\right)\Big|_{t=kh} = K\left(1 - e^{-kh/T}\right) \tag{109}$$

From (354) with $a = e^{-h/T}$ and (343) we get

$$H(z) = \frac{K\left(1 - e^{-h/T}\right)}{z - e^{-h/T}} \tag{110}$$

[End of Example 8.1]

8.2.2 Discretizing using canonical state space model

The discretization method described in this section involves state space models and discretization of such models. These topics are described in a later section (9) of this document, but references to these sections are given in the present section.

In Section 8.2.1 the discretization method involves the Laplace transform, and it may be difficult to find the transfer function if the original continuous-time transfer function is complicated or has a special form. I will now present a method that may work fine even for complicated transfer functions. It is based on discretization a canonical state space model corresponding to the given continuous-time transfer function. The method is as follows:

1. Given a continuous-time transfer function from input u to output y on the following general form:

$$\frac{y(s)}{u(s)} = H(s) = \frac{b_n s^n + b_{n-1}s^{n-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} \tag{111}$$

   Note that the coefficient a_n of the term s^n in the denominator is equal to 1.

2. Derive a continuous-time *canonical* state space model corresponding to $H(s)$. A canonical model is one with a defined structure among the infinitely number of state space models having equal transfer function. Here we choose a suitable model form called the *controllable canonical* form:

$$\begin{aligned} \dot{x}_1 &= x_2 && (112)\\ \dot{x}_2 &= x_3 && (113)\\ &\vdots \\ \dot{x}_n &= -a_0 x_1 - a_1 x_2 - \cdots - a_{n-1}x_n + u \\ y &= (b_0 - b_n a_0)\, x_1 + (b_1 - b_n a_1)\, x_2 + \cdots \\ &\quad + (b_{n-1} - b_n a_{n-1})\, x_n + b_n u && (114) \end{aligned}$$

3. Discretize the continuous-time canonical state space model above assuming zero order hold on the system's input. The discretization procedure is described in Section 9.2.2. The result is a discrete-time state space model on the form

$$x(k+1) = A_d x(k) + B_d u(k) \tag{115}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{116}$$

4. Calculate the discrete-time transfer function from the state space model using e.g. the formula (228) which is repeated here:

$$H(z) = \frac{y(z)}{u(z)} = C_d(zI - A_d)^{-1}B_d + D_d \tag{117}$$

Example 8.2 ***Discretizing a first order transfer function***

1. Given the following continuous-time transfer function:

$$H(s) = \frac{y(s)}{u(s)} = \frac{K}{Ts+1} = \frac{\frac{K}{T}}{s + \frac{1}{T}} = \frac{b_0}{s + a_0} \tag{118}$$

2. The continuous-time canonical state space model is found from (112) – (114):

$$\dot{x} = -\frac{1}{T_1}x + u = Ax + Bu \tag{119}$$

$$y = \frac{K}{T}x = Cx + Du \tag{120}$$

3. The discrete-time state space model is (assuming time-step h)

$$x(k+1) = A_d x(k) + B_d u(k) \tag{121}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{122}$$

where

$$\begin{aligned} A_d &= \left[\mathcal{L}^{-1}\left\{(sI - A)\right\}\right]_{t=h} && (123)\\ &= \left[\mathcal{L}^{-1}\left\{\left(s + \frac{1}{T}\right)\right\}\right]_{t=h} && (124)\\ &= e^{-h/T} && (125)\end{aligned}$$

$$\begin{aligned} B_d &= \left[\mathcal{L}^{-1}\left\{(sI - A)^{-1}\frac{1}{s}B\right\}\right]_{t=h} && (126)\\ &= h\left(1 - e^{-h/T}\right) && (127)\end{aligned}$$

4. The discrete-time transfer function can be calculated from (228):

$$\begin{aligned} H(z) &= \frac{y(z)}{u(z)} = D(zI - A_d)^{-1}B_d && (128)\\ &= \frac{K}{h}\left(z - e^{-h/T}\right)^{-1}\left[h\left(1 - e^{-h/T}\right)\right] && (129)\\ &= \frac{K\left(1 - e^{-h/T}\right)}{z - e^{-h/T}} && (130)\end{aligned}$$

which is the same as the result in Example (8.1).

[End of Example 8.2]

8.3 Discretizing using Euler's and Tustin's numerical approximations

8.3.1 The substitution formulas

We will now derive substitution formulas for substituting s in continuous-time transfer functions by a functions of z to get a discrete-time z-transfer function. The derivation will be based on numerical approximations to time integrals. Let us start with the following general differential equation:

$$\dot{x} = f(x,\ u) \tag{131}$$

where u is input variable and x is output variable. To simplify the notation we let $f(k)$ represent $f[x(t_k),\ u(t_k)] = f[x(k),\ u(k)]$. In general $f()$ is a nonlinear, vectorized[8] function. By integrating the differential equation (131), we get the

[8] which means that the variables are vectors, e.g. $x = [x_1, x_2]^T$

following solution:

$$x(t_k) = x(t_{k-1}) + \int_{t_{k-1}}^{t_k} f \, d\tau \tag{132}$$

or

$$x(k) = x(k-1) + \int_{(k-1)h}^{kh} f \, d\tau \tag{133}$$

The integral in (133) can be approximated in several ways:

- Euler's forward method:

$$x(k) \approx x(k-1) + hf(k-1) \tag{134}$$

- Euler's backward method:

$$x(k) \approx x(k-1) + hf(k) \tag{135}$$

- Tustin's method:

$$x(k) \approx x(k-1) + \frac{h}{2}\left[f(k) + f(k-1)\right] \tag{136}$$

The above integral approximations are illustrated in Figure 17.

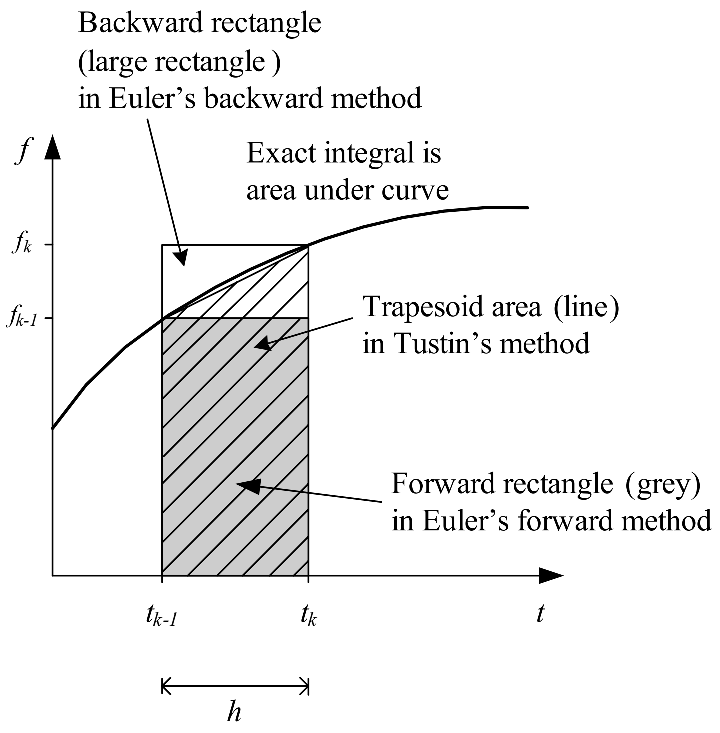

Figure 17: Numerical approximations to the integral in (133)

We will now study the simple model

$$\dot{x} = u \equiv f \tag{137}$$

(which is an integrator with u as input and x as output), and apply the above various integral approximations to the solution of this differential equation. By relating the z-transfer function from u to x and the s transfer function from u to x, which is

$$\frac{x(s)}{u(s)} = \frac{1}{s} \tag{138}$$

we will (soon) get the substitution formulas we are after. Setting $f = u$ in (134) – (136) yields the following differential equations. The z-transfer functions from u to x is derived and shown for each case.

- Euler's forward method:

$$x(k) = x(k-1) + hu(k-1) \Rightarrow \frac{x(z)}{u(z)} = \frac{h}{z-1} \tag{139}$$

- Euler's backward method:

$$x(k) = x(k-1) + hu(k) \Rightarrow \frac{x(z)}{u(z)} = \frac{hz}{z-1} \tag{140}$$

- Tustin's method:

$$x(k) = x(k-1) + \frac{h}{2}[u(k) + u(k-1)] \Rightarrow \frac{x(z)}{u(z)} = \frac{h}{2}\frac{z+1}{z-1} \tag{141}$$

By relating each of the z-transfer functions (139) – (141) with the s transfer function (138), we get the following substitution formulas:

$$\textbf{Euler forward}: \quad s \longleftarrow \frac{z-1}{h} \quad \text{or} \quad z \longleftarrow 1 + hs \tag{142}$$

$$\textbf{Euler backward}: \quad s \longleftarrow \frac{z-1}{hz} \quad \text{or} \quad z \longleftarrow \frac{1}{1-hs} \tag{143}$$

$$\textbf{Tustin}: \quad s \longleftarrow \frac{2}{h}\frac{z-1}{z+1} \quad \text{or} \quad z \longleftarrow \frac{1+\frac{h}{2}s}{1-\frac{h}{s}s} \tag{144}$$

Example 8.3 ***Discretizing a first order system with Tustin's method***

Given the following transfer function

$$\frac{y(s)}{u(s)} = H_{cont}(s) = \frac{1}{\frac{s}{\omega_b}+1} \tag{145}$$

which is the transfer function of a continuous-time first order lowpass filter with bandwidth ω_b [rad/s]. The filter will now be discretized using Tustin's method. Using the substitution (144) in (145) gives the following z-transfer function:

$$\frac{y(z)}{u(z)} = H_{disc}(z) = \frac{1}{\frac{2(z-1)}{h\omega_b(z+1)} + 1} = \frac{h\omega_b(z+1)}{(h\omega_b + 2)z + (h\omega_b - 2)} \tag{146}$$

Let us also derive the filtering algorithm: From (146),

$$(h\omega_b + 2)zy(z) + (h\omega_b - 2)y(z) = h\omega_b zu(z) + h\omega_b u(z) \tag{147}$$

Taking the inverse z-transform yields

$$(h\omega_b + 2)y(k+1) + (h\omega_b - 2)y(k) = h\omega_b u(k+1) + h\omega_b u(k) \tag{148}$$

Reducing the time indices by one and then solving for the output y yields the following filtering algorithm:

$$y(k) = \frac{2 - h\omega_b}{2 + h\omega_b}y(k-1) + \frac{h\omega_b}{2 + h\omega_b}u(k) + \frac{h\omega_b}{2 + h\omega_b}u(k-1) \tag{149}$$

[End of Example 8.3]

A few comments about the Tustins's method:

- In signal processing literature the Tustin's method is frequently denoted the *bilinear transformation method*. The term bilinear is related to the fact that the imaginary axis in the complex s-plane for continuous-time systems is mapped or transformed onto the unity circle for the corresponding discrete-time system. In addition, the poles are transformed so that the stability property is preserved (this is not guaranteed in the Euler's methods).
- In general the frequency responses of $H_{cont}(s)$ and of $H_{disc}(z)$ are not equal at the same frequencies. Tustins method can be modified or enhanced so that you can obtain equal frequency response of both $H_{cont}(s)$ and $H_{disc}(z)$ at one or more user-defined critical frequencies. This is done by modifying – *prewarping* – the critical frequencies of $H_{cont}(s)$ so that the frequency responses are equal after discretization by Tustin's method. However, it my experience that prewarping is seldom necessary, so I will not describe the prewarping procedure in further detail here [1].

8.3.2 Which discretization method should you choose?

Euler's forward method, Euler's backward method or Tustin's method? In general, Tustin's method is the most accurate of these three methods, so I suggest it is the default choice. However, typically it is not much difference between the Tustins' method and the Euler's backward method.

The Euler's forward method is the least accurate method. But this method may be a good choice for discretizing a nonlinear differential equation model since it gives an explicit formula for the output variable. Tustin's method and

Euler's backward methods are implicit methods since the output variable, $y(k)$, appears on both the left and the right side of the discretized expression, and it is in general necessary to solve the discretized expression for $y(k)$ at each time-step, and this may be impractical. See Section 9.2.4 for an example of discretizing a nonlinear model.

It turns out that Euler's backward method is quite commonly used for discretizing continuous-time control functions (i.e. the PID control function) in commercial control equipment, so this method can therefore be our choice in control applications.

8.3.3 Guidelines for choosing the time-step h

In general it is important that the time-step h of the discrete-time function is relatively small, so that the discrete-time function behaves approximately similar to the original continuous-time system. For the Euler's forward method a (too) large time-step may even result in an unstable discrete-time system!

The guidelines (rule of thumbs) below expresses either rules for the time-step h or for the sampling frequency f_s .They are related by

$$h = \frac{1}{f_s} \tag{150}$$

The guidelines depend somewhat on the application:

- **Filters**:
$$f_s \equiv \frac{1}{h} \geq 5f_H \text{ [Hz]} \tag{151}$$
Here, f_s is the sampling frequency and f_H is the highest frequency where you want the discrete-time filter to have almost the same characteristics as the original continuous-time filter. For example, for a lowpass filter f_H may be 5 times the bandwidth so that the filter characteristics up to 25 times the bandwidth will be similar to that of the original continuous-time filter.

  Above, the frequency unit is Hz. However, the relation (151) is the same if the frequency unit is rad/s:
$$\omega_s \equiv \frac{2\pi}{h} \geq 5\omega_H \text{ [rad/s]} \tag{152}$$

- **Controllers:** [4] The DA converter (digital to analog) which is always between the discrete-time control function and the continuous-time process to be controlled, implements holding of the calculated control signal during the time-step (sampling interval). This holding implies that the control signal is *time delayed* approximately $h/2$, see Figure 18. The delay influences the stability of the control loop. Suppose we have tuned a continuous-time PID controller, and apply these PID parameters on a discrete-time PID controller. Then the control loop will get reduced stability because of the approximate delay of h/s. As a rule of thumb (this can be confirmed from a frequency response based analysis), the

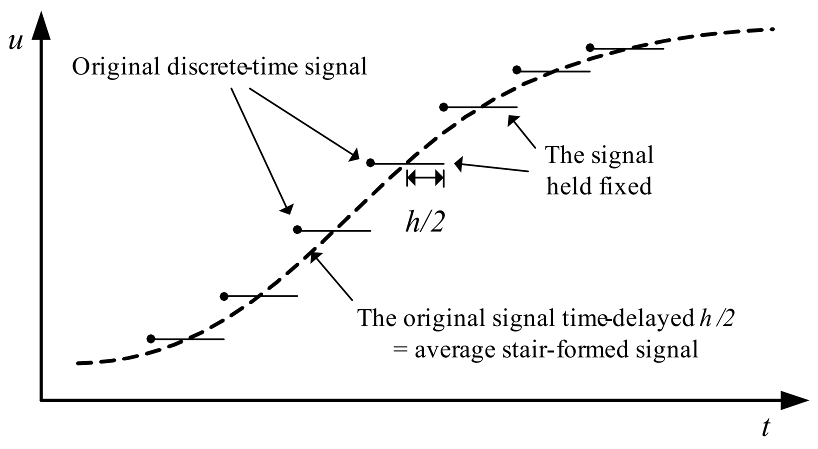
u
Original discrete-time signal
The signal held fixed
h/2
The original signal time-delayed h /2 = average stair-formed signal
t

Figure 18: The DA-converter holds the calculated control signal throyghout the sampling interval, thereby introducing an approximate time-delay of $h/2$.

stability reduction is small and tolerable if the time delay is less than *one tenth of the response-time* of the control system as it would have been with a continuous-time controller or a controller having very small sampling time:

$$\frac{h}{2} \leq \frac{T_r}{10} \tag{153}$$

which gives

$$h \leq \frac{T_r}{5} \tag{154}$$

The response time is here the 63% rise time which can be read off from the setpoint step response. For a system the having dominating time constant T, the response-time is approximately equal to this time constant. If the bandwidth of the control system is ω_b [rad/s] (assuming that the PID parameters have been found using a continuous-time PID controller), the response-time of the control system can be estimated by

$$T_r \approx \frac{1}{\omega_b} \tag{155}$$

- **Simulators:**

$$h \leq \frac{0.1}{|\lambda|_{\max}} \tag{156}$$

Here $|\lambda|_{\max}$ is the largest of the absolute values of the eigenvalues of the model, which is the eigenvalues of the system matrix A in the state-space model $\dot{\underline{x}} = A\underline{x} + B\underline{u}$. For transfer function models you can consider the poles in stead of the eigenvalues (the poles and the eigenvalues are equal for most systems not having pol-zero cancellations). If the model is nonlinear, it must be linearized before calculating eigenvalues or poles.

Above, maximum values of the time-step h, or minimum values of the sampling frequency f_s, were presented. However, you may also use a trial-and-error method for choosing h (or f_s): *Reduce h until there is a negligible change of the response of the system if h is further reduced.* If possible, you should use a simulator of your system to test the importance of the value of h before implementation.

Example 8.4 ***Sampling frequency for a filter***

In Example 8.3 a first order lowpass filter was discretized. Assume that the bandwidth is $\omega_b = 100\text{rad/s}$. Let us set $\omega_H = 5\omega_b = 500\text{rad/s}$, and use the lower limit of ω_s as given by (152):

$$\omega_s = 5\omega_H = 25\omega_b = 2500\text{rad/s} \tag{157}$$

which gives

$$h = \frac{2\pi}{\omega_s} = \frac{2\pi}{\omega_s} = 2.51\text{ms} \tag{158}$$

Figure 19 shows the frequency responses of the continuous-time filter $H_{cont}(s)$ given by (145) and the discrete-time filter $H_{disc}(z)$ given by (146). The two

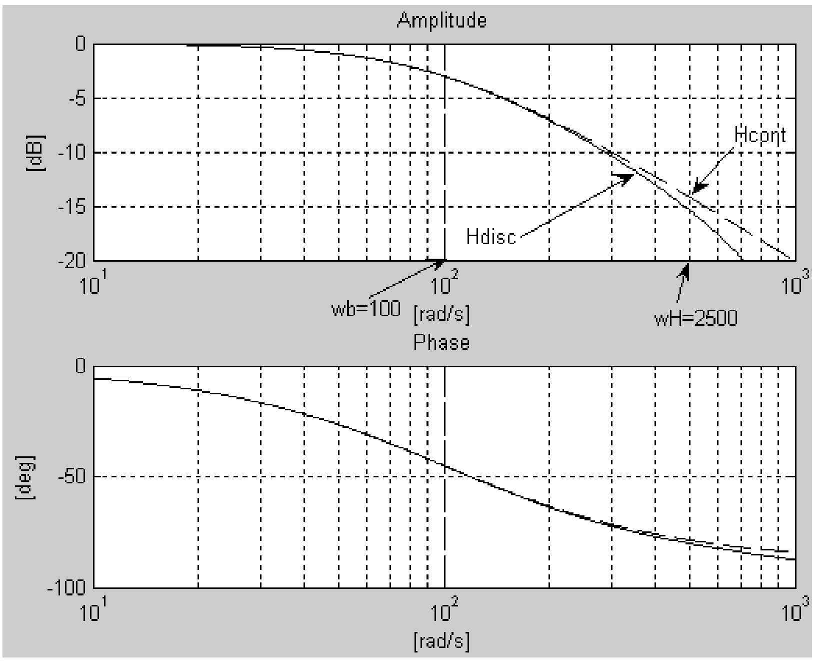

Figure 19: Example 8.4: Frequency responses of $H_{cont}(s)$ given by (145) and $H_{disc}(z)$ given by (146) with sampling frequency (157)

filters have quite equal frequency responses up to ω_H, as specified.

[End of Example 8.4]

8.4 The relation between continuous-time poles and discrete-time poles

In design of discrete-time control systems and discrete-time state estimators based on pole placement specifications it may be useful to know the above pole transformations. Specifications in the form of s poles can then be transformed to z poles, and the in e.g. pole placement controller design and pole placement controller design [2]. Knowing pole transformations may also be useful in stability analysis.

What is the relation between the poles $\{s_i\}$ of the original continuous-time system and the poles $\{z_i\}$ of the corresponding discrete-time system? The answer depends on the discretization method used:

- Discretization using zero order hold element:
$$z_i = e^{s_i h} \tag{159}$$
And reversely:
$$s_i = \frac{\ln z_i}{h} \tag{160}$$
(159) can be explained from (105) or from (106): The transfer function $G(s)$ of the continuous-time system has the poles $\{s_i\}$. Taking the inverse Laplace transform of $\frac{G(s)}{s}$ in (105) yields a linear combination of time functions of terms $e^{s_i t}$ which is $e^{s_i kh} = \left(e^{s_i h}\right)^k$ when we use $t = kh$. The z-transform of $\left(e^{s_i h}\right)^k$ is $\frac{z}{z-e^{s_i h}}$, cf. (353), and $\frac{z}{z-e^{s_i h}}$ has pole $z_i = e^{s_i k}$. Hence, (159) is proven.

- Discretization using Euler's methods or Tustin's method: The relation between the poles are given by (142) – (144). Hence, we can calculate the z poles by inserting the s poles in the proper expression on the right side of (142) – (144).

Example 8.5 ***Pole transformations***

Assume given that a continuous-time system with pole
$$s_1 = -2 \tag{161}$$

1. The system is discretized assuming a zero order hold element on the systems input. The time-step is $h = 0.1$s. What is the pole of the resulting discrete-time system? It is given by (159):
$$z_1 = e^{s_1 h} = e^{-2\cdot 0.1} = e^{-0.2} = 0.8187 \tag{162}$$

2. What is the pole if in stead the Tustin's method is used? It is given by (144):
$$z_1 = \frac{1 + \frac{0.1}{2}(-2)}{1 - \frac{0.1}{2}(-2)} = \frac{1 + \frac{0.1}{2}(-2)}{1 - \frac{0.1}{2}(-2)} = 0.8182 \tag{163}$$
(Thus, a slight difference.)

[End of Example 8.5]

9 State space models

9.1 General form and linear form of state space models

The general form of a discrete-time state space model is

$$x(k+1) = f[x(k),\ u(k)] \tag{164}$$

$$y(k) = g[x(k),\ u(k)] \tag{165}$$

where x is the state variable, u is the input variable which may consist of control variables and disturbances (in this model definition there is no difference between these two kinds of input variables). y is the output variable. f and g are functions – linear or nonlinear. $x(k+1)$ in (164) means the state one time-step ahead (relative to the present state $x(k)$). Thus, the state space model expresses how the systems' state (variables) and output variables evolves along the discrete time axis.

The variables in (164) – (165) may actually be vectors, e.g.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{166}$$

where x_i is a (scalar) state variable, and if so, f and/or g are vector evaluated functions.

A special case of the general state space model presented above is the *linear* state space model:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{167}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{168}$$

where A_d (subindex d for discrete-time) is the transition matrix, B_d is the input matrix, C_d is the output matrix and D_d is the direct output matrix (in most cases, $D_d = 0$).

9.2 Discretization of linear continuous-time state space models

9.2.1 Introduction

The needs for discretizing a continuous-time state space model are similar to the needs for a discrete-time transfer function model, cf. Section 8.1:

- In accurate model based design of a discrete controller for a process originally represented by a continuous-time state space model. This state space model must be discretized before the design work.
- Implementation of continuous-time control and filtering functions in computer program.

The underlying principles of the various discretization methods are also the same as for discretizing transfer function models:

- Discretization based on having a zero order hold element on the input of the system which is the case when the physical process is controlled by a computer via a DA converter (digital to analog). See Figure 16.
- Using a integral numerical approximation, typically Euler's forward method, Euler's backward method or Tustin's method.

9.2.2 Discretization with zero order hold element on the input

Assume given the following continuous-time state space model:

$$\dot{x} = Ax + Bu \tag{169}$$

$$y = Cx + Du \tag{170}$$

having a zero order hold element on its input, see Figure 16. It can be shown that the following discrete-time state space model expresses how the state x evolves along a discrete time axis:[9]

$$x(k+1) = A_d x(k) + B_d u(k) \tag{171}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{172}$$

where

$$A_d = e^{Ah} \tag{173}$$

$$B_d = \int_0^h e^{A\tau}\, d\tau B \tag{174}$$

$$C_d = C \tag{175}$$

$$D_d = D \tag{176}$$

In (173) and (174), e is the natural base. $A_d = e^{Ah}$ is denoted the transition matrix. It is a matrix exponential which can be calculated as described below. One interpretation of A_d is that it tells how the transition from state $x(k)$ to state $x(k+1)$ takes place in an autonomous system (having $u \equiv 0$). This transition is given by $x(k+1) = A_d x(k)$.

A_d given by (173) and B_d given by (174) can be calculated in several ways. Below are described two methods:

- **Exact calculation:**

$$A_d = \mathcal{L}^{-1}\{(sI - A)^{-1}\}\big|_{t=h} = e^{At}\big|_{t=h} = e^{Ah} \tag{177}$$

$$B_d = \mathcal{L}^{-1}\left\{(sI - A)^{-1}\frac{1}{s}B\right\}\bigg|_{t=h} = \int_0^h e^{A\tau}\, d\tau B \tag{178}$$

  where $\mathcal{L}^{-1}$ means the inverse Laplace transform.

[9] The discrete-time state space model can be derived by using the Laplace transform to calculate the state $x(t_{k+1}) = x(k+1)$.

- **Approximative (numerical) calculation:**

$$\begin{aligned} A_d &= I + Ah + \frac{A^2h^2}{2!} + \frac{A^3h^3}{3!} + \cdots \frac{A^nh^n}{n!} \cdots && (179) \\ &= I + A \underbrace{\left(Ih + \frac{Ah^2}{2!} + \frac{A^2h^3}{3!} + \cdots \frac{A^{n-1}h^n}{n!} \cdots \right)}_{S} && (180) \\ &= I + AS && (181) \end{aligned}$$

$$\begin{aligned} B_d &= \underbrace{\left(Ih + \frac{Ah^2}{2!} + \frac{A^2h^3}{3!} + \cdots \frac{A^{n-1}h^n}{n!} \cdots \right)}_{S} B && (182) \\ &= SB && (183) \end{aligned}$$

where I is the identity matrix of the same dimension as A, and S is just a common series term in A_d and B_d. Of course, the larger order n, the more accurate resulting values of A_d and B_d.[10] On the other hand, for a given n, the discrete-time model is less accurate the larger h, and the model may even become unstable if h is too large. (156) may be used as a guideline for choosing h.

Example 9.1 ***Discretization of a double integrator***

In this example the state space model

$$\dot{x} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u \tag{184}$$

$$y = \underbrace{[1]}_{C} x + \underbrace{[0]}_{D} u \tag{185}$$

where $x = [x_1 \;\; x_2]^T$ is the state vector, will be discretized using the exact method and the numerical method with order $n = 1$. This is a state space model of the double integrator $\ddot{y} = u$ where $y = x_1$. The time-step (sampling interval) is $h = 0.5$s.

- **Exact discretization**: The matrices in the resulting discrete-time state

[10] With $n = 1$ the resulting state space model is actually the same as obtained using Euler's forward method, cf. Section 9.2.3.

space model (171) – (172) becomes, using (177) – (178),

$$
\begin{aligned}
A_d &= \mathcal{L}^{-1}\left\{(sI-A)^{-1}\right\}\Big|_{t=h} \\
&= \mathcal{L}^{-1}\left\{\begin{bmatrix} s & -1 \\ 0 & s \end{bmatrix}^{-1}\right\}\Bigg|_{t=h} \\
&= \mathcal{L}^{-1}\left\{\begin{bmatrix} \frac{1}{s} & \frac{1}{s^2} \\ 0 & \frac{1}{s} \end{bmatrix}\right\}\Bigg|_{t=h} \\
&= \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}\Bigg|_{t=h} \\
&= \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} \qquad (186)
\end{aligned}
$$

$$
\begin{aligned}
B_d &= \mathcal{L}^{-1}\left\{(sI-A)^{-1}\frac{1}{s}B\right\}\Big|_{t=h} \qquad (187) \\
&= \mathcal{L}^{-1}\left\{\begin{bmatrix} s & -1 \\ 0 & s \end{bmatrix}^{-1}\frac{1}{s}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\right\}\Bigg|_{t=h} \\
&= \mathcal{L}^{-1}\left\{\begin{bmatrix} \frac{1}{s^3} \\ \frac{1}{s^2} \end{bmatrix}\right\}\Bigg|_{t=h} \qquad (188) \\
&= \begin{bmatrix} \frac{h^2}{2} \\ h \end{bmatrix} \\
&= \begin{bmatrix} 0.125 \\ 0.5 \end{bmatrix} \qquad (189)
\end{aligned}
$$

$$C_d = C = [1] \qquad (190)$$

$$D_d = D = [0] \qquad (191)$$

- **Approximative (numerical) discretization**: The matrices in the resulting discrete-time state space model becomes, using (179) – (182) with $n = 1$,

$$
\begin{aligned}
x(k+1) &= (I+Ah)x(k) + Bhu(k) \qquad (192) \\
&= \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ h \end{bmatrix} u(k) \qquad (193) \\
&= \underbrace{\begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix}}_{A_d} x(k) + \underbrace{\begin{bmatrix} 0 \\ 0.5 \end{bmatrix}}_{B_d} u(k) \qquad (194)
\end{aligned}
$$

Figure 20 shows simulated responses in the output variable y for a ramp in u for the two discrete-time models and for the original continuous-time model. The initial state is $x(0) = [0, 0]^T$. It is the ramp after the zero order hold element (this signal is stair-formed in Figure 20) that is used to excite the continuous-time system. We observe the following:

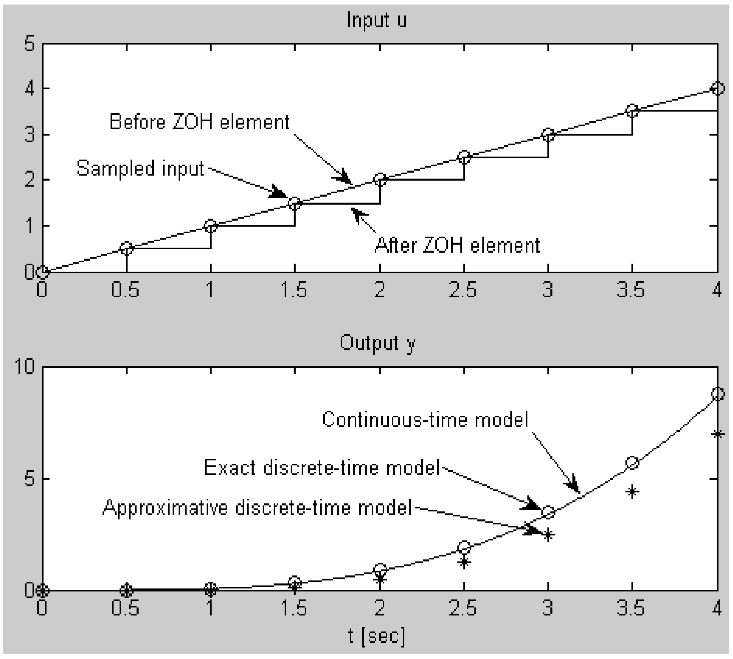

Figure 20: Example 9.1: Simulated responses for continuous-time system and two different discrete-time models

- In the discrete points of time the response in the exactly discretized model is equal to the response in the continuous-time model, as expected.
- The response of the approximative discrete-time model differs from the exact response, as expected.

[End of Example 9.1]

9.2.3 Discretizing using Euler's and Tustins' numerical approximations

The discretization methods described in this section do not assume zero order hold on the system's input. The basis of the methods is to get a discrete-time model which hopefully behaves similar to the original continuous-time state space model, which here is assumed linear:

$$\dot{x} = \underbrace{Ax + Bu}_{f()} \tag{195}$$

$$y = Cx + Du \tag{196}$$

where u is input variable and x is output variable. To simplify the notation we let $f(k)$ represent $f[x(t_k),\ u(t_k)] = f[x(k),\ u(k)]$. Actually, the discretization formulas were developed in Section 8.3. All we have to do now is inserting $Ax(k) + Bu(k)$ for $f(k)$ in (134) – (136), and then solve for $x(k)$. The resulting discretizing formulas are shown below:

- **Euler's forward method**:

$$x(k) = (I + Ah)x(k-1) + Bhu(k-1) \tag{197}$$

  or (equivalently)

$$x(k+1) = (I + Ah)x(k) + Bhu(k) \tag{198}$$

- **Euler's backward method**:

$$x(k) = (I - Ah)^{-1}x(k-1) + (I - Ah)^{-1}BTu(k) \tag{199}$$

- **Tustin's method**:

$$x(k) = \left(I - \frac{Ah}{2}\right)^{-1}\left(I + \frac{Ah}{2}\right)x(k-1) \tag{200}$$

$$+\left(I - \frac{Ah}{2}\right)^{-1}\frac{Bh}{2}[u(k) + u(k-1)] \tag{201}$$

A guideline for choosing a proper sampling time h is (156) in Section 8.3.3.

9.2.4 Discretizing nonlinear state space models

A general form of a nonlinear continuous-time state space model is

$$\dot{x} = f(x, u) \tag{202}$$

where $f()$ is a nonlinear function of x and/or u. It may be difficult to apply Tustins's method or Euler's backward method for discretizing (202) since these methods are implicit, which means that they do not immediately give an explicit or isolated formula for the state variable $x(k)$. For example, using Euler's backward method (135) on the model

$$\dot{x} = -\sqrt{x} + 2u \tag{203}$$

yields

$$x(k) \approx x(k-1) + hf\left[x(k), u(k)\right] \tag{204}$$

$$= x(k-1) + h\left[-\sqrt{x(k)} + 2u(k)\right] \tag{205}$$

which is a nonlinear implicit equation for $x(k)$. In general such equations are not easy to solve, and an iterative procedure may be necessary.

In practical applications[11] where you actually need a discretized version of a nonlinear continuous-time model Euler's forward method (134) is commonly used. Inserting (202) into (134) yields the Euler's forward discretizing formula:

$$x(k) = x(k-1) + hf(k-1) \tag{206}$$

[11] As developing state estimators (Kalman filters)

Example 9.2 ***Discretization of nonlinear model using Euler's backward method***

Using (206) on (134) yields

$$x(k) = x(k-1) + h\left[-\sqrt{x(k-1)} + 2u(k-1)\right] \tag{207}$$

[End of Example 9.2]

9.3 Linearizing nonlinear state space models

The formulas for linearizing nonlinear discrete-time state space models are presented without derivation below. They can be derived in the same way as for linearizing nonlinear continuous-time models [3]. In the formulas below it assumed a second order system. I guess it is clear how the formulas can be generalized to higher orders.

Given the following continuous-time nonlinear state space model

$$\begin{aligned} x_1(k) &= f_1[x_1(k),\ x_2(k),\ u_1(k),\ u_2(k)] \\ x_2(k) &= f_2[x_1(k),\ x_2(k),\ u_1(k),\ u_2(k)] \end{aligned} \tag{208}$$

$$\begin{aligned} y_1(k) &= g_1[x_1(k),\ x_2(k),\ u_1(k),\ u_2(k)] \\ y_2(k) &= g_2[x_1(k),\ x_2(k),\ u_1(k),\ u_2(k)] \end{aligned} \tag{209}$$

where f_1 and f_2 are nonlinear functions. The corresponding linear model, which defines the system's dynamic behaviour about a specific operating point, is

$$\begin{aligned} \Delta x_1(k+1) &= \left.\tfrac{\partial f_1}{\partial x_1}\right|_{\rm op} \Delta x_1(k) + \left.\tfrac{\partial f_1}{\partial x_2}\right|_{\rm op} \Delta x_2(k) + \left.\tfrac{\partial f_1}{\partial u_1}\right|_{\rm op} \Delta u_1(k) + \left.\tfrac{\partial f_1}{\partial u_2}\right|_{\rm op} \Delta u_2(k) \\ \Delta x_2(k+1) &= \left.\tfrac{\partial f_2}{\partial x_1}\right|_{\rm op} \Delta x_1(k) + \left.\tfrac{\partial f_2}{\partial x_2}\right|_{\rm op} \Delta x_2(k) + \left.\tfrac{\partial f_2}{\partial u_1}\right|_{\rm op} \Delta u_1(k) + \left.\tfrac{\partial f_2}{\partial u_2}\right|_{\rm op} \Delta u_2(k) \end{aligned} \tag{210}$$

$$\begin{aligned} \Delta y_1(k) &= \left.\tfrac{\partial g_1}{\partial x_1}\right|_{\rm op} \Delta x_1(k) + \left.\tfrac{\partial g_1}{\partial x_2}\right|_{\rm op} \Delta x_2(k) + \left.\tfrac{\partial g_1}{\partial u_1}\right|_{\rm op} \Delta u_1(k) + \left.\tfrac{\partial g_1}{\partial u_2}\right|_{\rm op} \Delta u_2(k) \\ \Delta y_2(k) &= \left.\tfrac{\partial g_2}{\partial x_1}\right|_{\rm op} \Delta x_1(k) + \left.\tfrac{\partial g_2}{\partial x_2}\right|_{\rm op} \Delta x_2(k) + \left.\tfrac{\partial g_2}{\partial u_1}\right|_{\rm op} \Delta u_1(k) + \left.\tfrac{\partial g_2}{\partial u_2}\right|_{\rm op} \Delta u_2(k) \end{aligned} \tag{211}$$

or

$$\Delta x(k+1) = A_d \Delta x(k) + B_d \Delta u(k) \tag{212}$$

$$\Delta y(k) = C_d \Delta x(k) + D_d \Delta u(k) \tag{213}$$

where[12]

$$A_d = \left.\left[\begin{array}{cc} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{array}\right]\right|_{\rm op} = \left.\frac{\left[\begin{array}{c} \partial f_1 \\ \partial f_2 \end{array}\right]}{\left[\begin{array}{cc} \partial x_1 & \partial x_2 \end{array}\right]}\right|_{\rm op} = \left.\frac{\partial f}{\partial x^T}\right|_{\rm op} \tag{214}$$

[12] Partial derivative matrices are denoted Jacobians.

$$B_d = \left. \frac{\partial f}{\partial u^T} \right|_{\text{op}} \tag{215}$$

$$C_d = \left. \frac{\partial g}{\partial x^T} \right|_{\text{op}} \tag{216}$$

$$D_d = \left. \frac{\partial g}{\partial u^T} \right|_{\text{op}} \tag{217}$$

In the formulas above the subindex *op* is for operating point, which is a particular set of values of the variables. Typically the operating point is an equilibrium (or static) operating point, which means that all variables have constant values.

9.4 Calculating responses in discrete-time state space models

9.4.1 Calculating dynamic responses

Calculating responses in discrete-time state space models is quite easy. The reason is that the model *is* the algorithm! For example, assume that Euler's forward method has been used to get the following discrete-time state space model:

$$x(k) = x(k-1) + hf(k-1) \tag{218}$$

This model constitutes the algorithm for calculating the response $x(k)$.

9.4.2 Calculating static responses

The static response is the response when all input variables have constant values and all output variables have converged to constant values. Assume the following possibly nonlinear state space model:

$$x(k+1) = f_1\left[x(k), u(k)\right] \tag{219}$$

where f_1 is a possibly nonlinear function of x and u. Let us write x_s and u_s for static values. Under static conditions (219) becomes

$$x_s = f_1\left[x_s, u_s\right] \tag{220}$$

which is an algebraic equation from which we can try to solve for unknown variables.

If the model is linear:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{221}$$

a formula of the steady-state solution can be calculated as follows. The steady-state version of (221) is

$$x_s = A_d x_s + B_d u_s \tag{222}$$

Solving for x_s gives

$$x_s = (I - A_d)^{-1} B_d u_s \tag{223}$$

Note that only asymptotically stable systems have a feasible static operating point. Stability of discrete-time systems is analyzed in Section 11.

9.5 From state space model to transfer function

Given the following linear discrete-time state space model:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{224}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{225}$$

We will now calculate a general formula for the z-transfer function $H(z)$ from input u to output y. (If u and/or y are vectors, the transfer function we calculate is actually a transfer function matrix.) Taking the z-transform of (224) yields

$$zx(z) = zIx(z) = A_d x(z) + B_d u(z) \tag{226}$$

where I is the identity matrix having the same order as the number of state variables. Solving for $x(z)$ gives

$$x(z) = (zI - A_d)^{-1} B_d u(z) \tag{227}$$

Combining with (225) gives the following formula(s) for the transfer function:

$$\begin{aligned} H(z) &= \frac{y(z)}{u(z)} = C_d (zI - A_d)^{-1} B_d + D_d & (228) \\ &= C_d \frac{\operatorname{adj}(zI - A_d)}{\det(zI - A_d)} B_d + D_d & (229) \end{aligned}$$

which is the formula for $H(z)$. *adj* means adjoint, and *det* means determinant. $(zI - A_d)$ is denoted the *characteristic matrix* of the system.

Example 9.3 ***Transfer function of the double integrator***

In Example 9.1 the discrete-time state space model of a double integrator was derived The model is on the form (224) – (225) with system matrices as given by (186) – (191). We will now calculate the z-transfer function from u to $y = x_1$ using (229). We start by calculating the characteristic matrix $zI - A_d$. Here, A_d is given by (186). We get

$$(zI - A_d) = \underbrace{\begin{bmatrix} z & 0 \\ 0 & z \end{bmatrix}}_{zI} - \underbrace{\begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}}_{A_d} = \begin{bmatrix} z-1 & -h \\ 0 & z-1 \end{bmatrix} \tag{230}$$

which gives

$$\operatorname{adj}(zI - A_d) = \operatorname{adj}\begin{bmatrix} z-1 & -h \\ 0 & z-1 \end{bmatrix} = \begin{bmatrix} z-1 & h \\ 0 & z-1 \end{bmatrix} \tag{231}$$

and

$$\det(zI - A_d) = \det\begin{bmatrix} z-1 & -h \\ 0 & z-1 \end{bmatrix} = z^2 - 2z + 1 \tag{232}$$

B_d is given by (188). C_d is [1 0], and $D_d = 0$. The transfer function becomes

$$\frac{y(z)}{u(z)} = D_d(zI - A_d)^{-1}B_d = D_d\frac{\mathrm{adj}(zI - A_d)}{\det(zI - A_d)}B_d \tag{233}$$

$$= [1 \;\; 0] \cdot \frac{1}{z^2 - 2z + 1} \cdot \begin{bmatrix} z-1 & h \\ 0 & z-1 \end{bmatrix} \cdot \begin{bmatrix} \frac{h^2}{2} \\ h \end{bmatrix} \tag{234}$$

$$= \frac{h^2}{2}\frac{z+1}{z^2 - 2z + 1} \tag{235}$$

[End of Example 9.3]

9.6 From transfer function to state space model

Assume that you need a state space model which has the following transfer function,

$$\frac{y(z)}{u(z)} = H(z) = \frac{b_n z^n + b_{n-1}z^{n-1} + \cdots + b_1 z + b_0}{z^n + a_{n-1}z^{n-1} + \cdots + a_1 z + a_0} \tag{236}$$

Note that the coefficient in the z_n term in the denominator is 1 $(= a_n)$. Actually, there are an infinite number of state space models having the above transfer function. Therefore it may be wise to choose one of the so-called *canonical* state space models which have defined structures. One such canonical model is presented here. It can be shown that (236) is the transfer function from u to y in the (canonical) block diagram shown in Figure 21. The

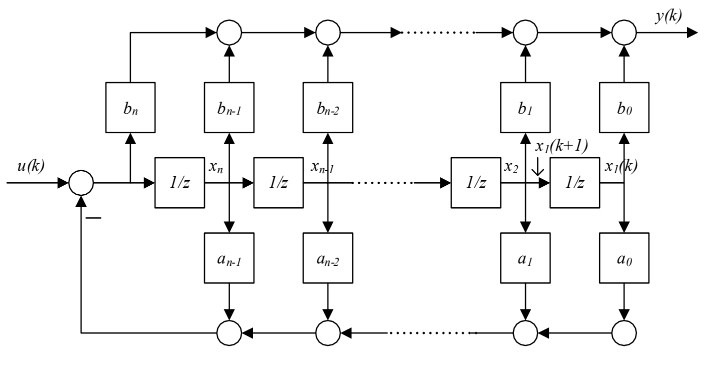

Figure 21: The transfer function from u to y in this (canonical) block diagram is (236).

variables on the outputs of the time-step delay blocks (containing $1/z$) are *canonical state variables*. From the block diagram we can write down the following canonical state space model:

$$\begin{array}{lcl}
x_1(k+1) & = & x_2(k) \\
x_2(k+1) & = & x_3(k) \\
 & \vdots & \\
x_{n-1}(k+1) & = & x_n(k) \\
x_n(k+1) & = & -a_0x_1(k) - a_1x_2(k) - \cdots - a_{n-1}x_n(k) \quad + u(k) \\
 & & \\
y(k) & = & (b_0 - b_na_0)x_1(k) + \cdots + (b_{n-1} - b_na_{n-1})x_n(k) + b_nu(k)
\end{array} \tag{237}$$

which is a linear state space model which compactly can be written as (however, the matrices will not be shown here):

$$x(k+1) = A_dx(k) + B_du(k) \tag{238}$$

$$y(k) = C_dx(k) + D_du(k) \tag{239}$$

Example 9.4 ***Canonical state space model***

We will find a state space model having the following transfer function:

$$\frac{y(z)}{u(z)} = H(z) = \frac{2z^{-3}}{1 - 0,5z^{-1}} \tag{240}$$

First we write the the transfer function on the standard form (236):

$$H(z) = \frac{2}{z^3 - 0,5z^2} = \frac{b_3z^3 + b_2z^2 + b_1z + b_0}{z^3 + a_2z^2 + a_1z + a_0} \tag{241}$$

hence, the coefficients are $b_3 = 0 = b_2 = b_1$, $b_0 = 2$, $a_2 = -0.5$ and $a_1 = 0 = a_0$. From (237) we get the following state space model:

$$\begin{array}{lcl}
x_1(k+1) & = & x_2(k) \\
x_2(k+1) & = & x_3(k) \\
x_3(k+1) & = & 0,5x_3(k) + u(k) \\
 & & \\
y(k) & = & 2x_1(k)
\end{array} \tag{242}$$

[End of Example 9.4]

10 Dynamics of discrete-time systems

I this section we will study a few important types of discrete-time dynamic systems represented in the form of transfer functions, namely gain, integrator, first order system, and system with time delay.

10.1 Gain

A discrete-time gain has the transfer function

$$\frac{y(z)}{u(z)} = H(z) = K \tag{243}$$

where K is the (proportional) *gain*, u is the input y is the output. The gain has no poles and no zeros and has therefore no "dynamics". In the time domain the relation between u and y is given by

$$y(k) = Ku(k) \tag{244}$$

Thus, the output is proportional to the input.

10.2 Integrator

We start with the continuous-time integrator represented by the following integral equation

$$y(t) = y(0) + \int_0^t K_i u(\tau)\, d\tau \tag{245}$$

which corresponds to the following differential equation

$$\dot{y} = K_i u \tag{246}$$

The s transfer function from u to y is

$$\frac{y(s)}{u(s)} = H(s) = \frac{K_i}{s} \tag{247}$$

K_i is the integral gain. This integrator can be discretized in several ways, cf. Section 8. Let us confine ourselves to discretization using zero order hold element on the input, cf. Section 8.2.1. It can be shown[13] that the discrete-time transfer function becomes

$$\frac{y(z)}{u(z)} = H(z) = \frac{K_i h}{z-1} \tag{248}$$

where h is the time-step. The pole is

$$p = 1 \tag{249}$$

The step response (unit step at the input) of $H(z)$ can be calculated as follows: Assume that the step in input u has amplitude U, which z-transformed is, cf. (352),

$$u(z) = \frac{Uz}{z-1} \tag{250}$$

The z-transform of the step response becomes

$$y_{step}(z) = H(z)u(z) = \frac{K_i h}{z-1} \cdot \frac{Uz}{z-1} \tag{251}$$

[13] You can e.g. apply (106) together with (355) with $a = 1$.

Taking the inverse transform using (355) yields

$$y_{step}(h) = K_i U h k \tag{252}$$

Figure 22 shows the pole and the step response (unity step) for with $K_i = 1$ and time-step $h = 1$.

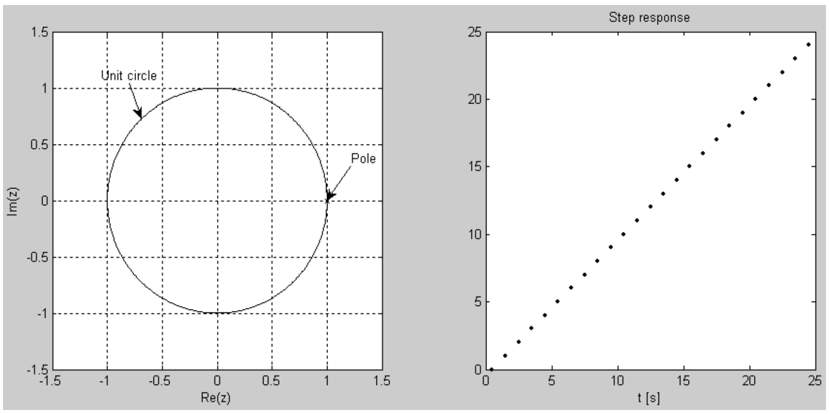

Figure 22: Pole and step response (unity step) for the integrator (248) with $K_i = 1$ and time step $h = 1$

In some cases a given discrete-time transfer function $H(z)$ should be modified so that it gets integrating action. One example is design of a control function with integrating action. Integrator action can be added (included) by multiplying the original transfer function, say $H_0(z)$, by the factor

$$H_{int}(z) = \frac{h}{z-1} \tag{253}$$

so that the resulting transfer function is

$$H(z) = H_0(z) H_{int} = H_0(z) \frac{h}{z-1} \tag{254}$$

(thereby adding the pole +1).

10.3 First order system

Here is a general first order discrete-time transfer function:

$$H(z) = \frac{y(z)}{u(z)} = \frac{b}{z-p} \tag{255}$$

where p is the pole, which is on the real axis. Let us study the step response for different values of p. Assume that the step in input u has amplitude U, which z-transformed is, cf. (352),

$$u(z) = \frac{Uz}{z-1} \tag{256}$$

The z-transform of the step response becomes

$$y_{step}(z) = H(z)u(z) = \frac{b}{z-p} \cdot \frac{Uz}{z-1} \tag{257}$$

Taking the inverse transform using (354) yields

$$y_{step}(k) = \frac{bU}{1-p}\left(1-p^k\right) \tag{258}$$

Let us study the following different cases, given by the pole p:

- $p = 1$: The transfer function (255) is then

  $$H(z) = \frac{b}{z-1} \tag{259}$$

  and the system is an integrator. This case is covered in Section 10.2.

- $0 < p < 1$: As an example, let us consider the transfer function (255) having $p = 0.82$ and $b = 0.18$. This transfer function happens to be the discretized version of the continuous-time transfer function

  $$G(s) = \frac{K}{Ts+1} \tag{260}$$

  with time-step $h = 0.2$s (the discretization was performed in 8.1). Figure

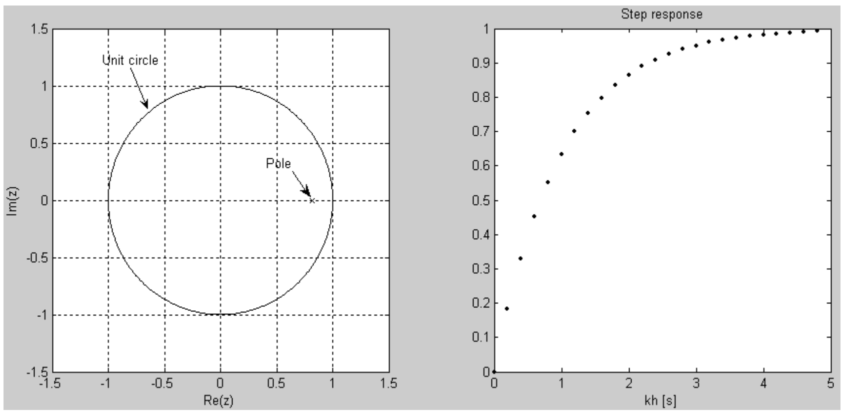

Figure 23: Pole placement and the step response (unit step at the input) of $H(z)$ given by (255) for $p = 0.82$ and $b = 0.18$.

  23 shows the pole placement and the step response (unit step at the input) of $H(z)$ given by (255). The step response has an exponential convergence towards its steady-state value.

  Assuming that $H(z)$ is the discretized $G(s)$, the discrete-time pole is

  $$p = e^{-h/T} \tag{261}$$

  from which we can conclude that the a smaller p (smaller time constant T) gives a faster response. This can also be seen from (258).

- $p = 0$: The transfer function (255) becomes

$$H(z) = \frac{b}{z} = bz^{-1} \tag{262}$$

which is the transfer function of a delay of one time-step, cf. (51). The pole is in *origin*. Discrete-time time delay is described in more detail in Section 10.4.

- $-1 < p < 0$: The transfer function is still (255). The step response is still (258). Assume as an example that $p = -0.8$ and $b = 1$. Figure 24 shows the pole and the step response. The step response now *oscillates*, but it

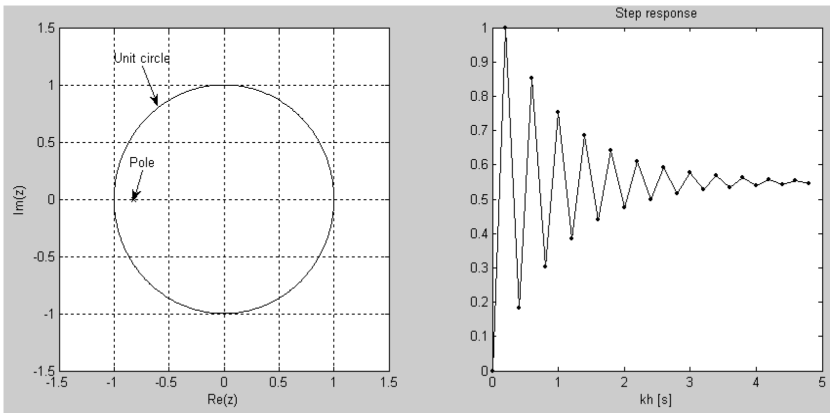

Figure 24: Pole and step response for $p = -0.8$ and $b = 1$

converges towards a steady-state value. (Continuous-time first order systems can not oscillate, but discrete-time first order systems can.)

- $p = -1$: The transfer function (255) becomes

$$H(z) = \frac{b}{z+1} \tag{263}$$

The step response is (258). Assume that $b = 1$. Figure 25 shows the pole and the step response which shows undamped oscillations of period 2 time-steps. The oscillations is denoted "ringing".

10.4 System with time delay

According to (344) multiplying a z-transform by z^{-n} means time delay by n time-steps, i.e. $n \cdot h$. Thus, the transfer function of a time delay of n time-steps is

$$H(z) = z^{-n} = \frac{1}{z^n} \tag{264}$$

The transfer function has n poles in origin. Figure 26 shows the poles and the step response for $n = 5$. The time-step is $h = 0.5$.

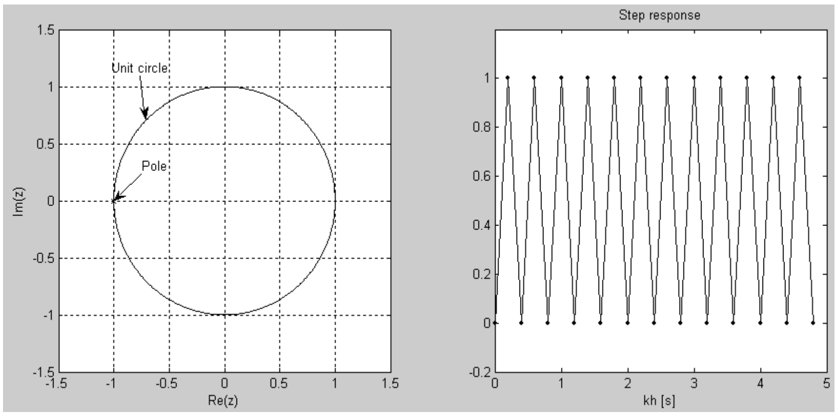

Figure 25: Pole and step response for $p = -1$ and $b = 1$

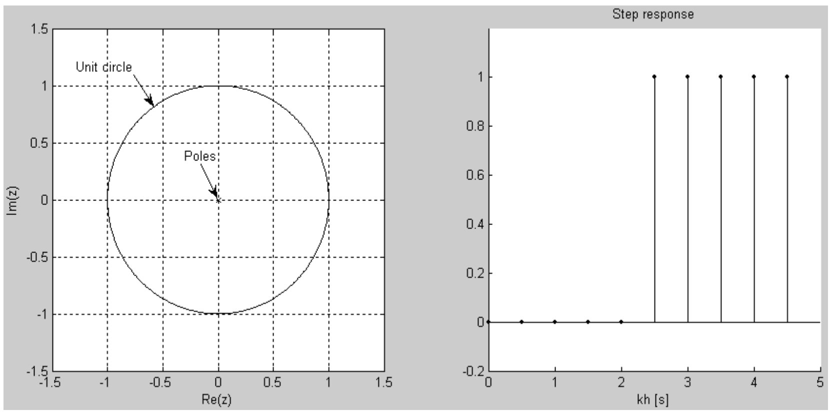

Figure 26: Poles and the step response for $n = 5$. The time step is $h = 0.5$.

11 Stability analysis

11.1 Stability properties

Assume given a dynamic system with input u and output y. The stability property of a dynamic system can be defined from the *impulse response*[14] of a system as follows:

- **Asymptotic stable system:** The steady state impulse response is zero:

$$\lim_{k\to\infty} y_\delta(k) = 0 \tag{265}$$

- **Marginally stable system:** The steady state impulse response is

[14] There is an impulse $\delta(0)$ on the input.

different from zero, but limited:

$$0 < \lim_{k\to\infty} y_\delta(k) < \infty \tag{266}$$

- **Unstable system:** The steady state impulse response is unlimited:

$$\lim_{k\to\infty} y_\delta(k) = \infty \tag{267}$$

The impulse response for the different stability properties are illustrated in Figure 27. (The simulated system is defined in Example 11.1.)

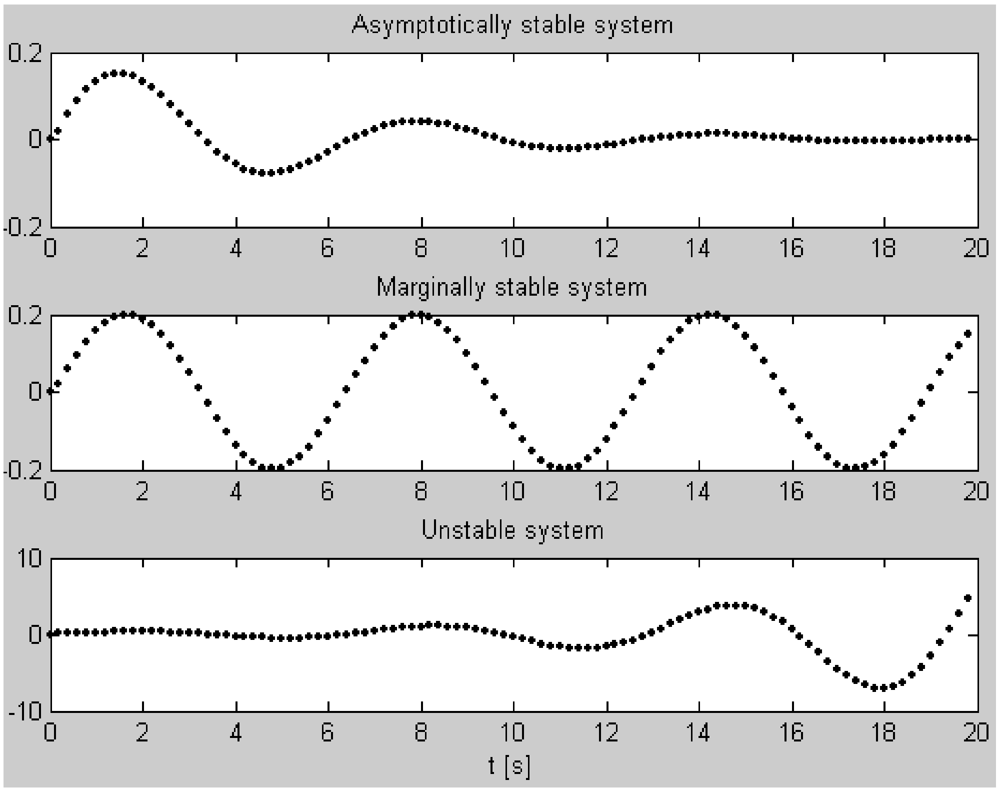

Figure 27: Impulse repsonse and stability properties

11.2 Stability analysis of transfer function models

The impulse response $y_\delta(k)$ is determined by the poles of the system's poles and zeros since the impulse responses is the inverse z-transform of the transfer function, cf. Example 6.2:

$$y_\delta(k) = \mathcal{Z}^{-1}\{H(z)\} \tag{268}$$

Consequently, the stability property is determined by the poles and zeros of $H(z)$. However, we will soon see that only the poles determine the stability.

We will now derive the relation between the stability and the poles by studying the impulse response of the following system:

$$H(z) = \frac{y(z)}{u(z)} = \frac{bz}{z-p} \tag{269}$$

The pole is p. Do you think that this system is too simple as a basis for deriving general conditions for stability analysis? Actually, it is sufficient because we can always think that a given z-transfer function can be partial fractionated in a sum of partial transfer functions or terms each having one pole. Using the superposition principle we can conclude about the stability of the original transfer function.

In the following, cases having of multiple (coinciding) poles will be discussed, but the results regarding stability analysis will be given.

From Example 6.2 we know that the z-transform of the impulse response of a system is the transfer function of the system. The system given by (269) has the following impulse response calculated below. It is assumed that the pole in general is a complex number which may be written on polar form as

$$p = me^{j\theta} \tag{270}$$

where m is the magnitude and θ the phase. The impulse response is

$$\begin{aligned}
y_\delta(k) &= \mathcal{Z}^{-1}\left\{\frac{bz}{z-p}\right\} && (271)\\
&= \mathcal{Z}^{-1}\left\{\frac{p}{1-pz^{-1}}\right\} && (272)\\
&= \mathcal{Z}^{-1}\left\{b\sum_{k=0}^{\infty} p^k z^{-k}\right\} && (273)\\
&= bp^k && (274)\\
&= b|m|^k e^{jk\theta} && (275)
\end{aligned}$$

From (275) we see that it is the *magnitude* m which determines if the steady state impulse response converges towards zero or not. From (275) we can now state the following relations between stability and pole placement (the statements about multiple poles have however not been derived here):

- **Asymptotic stable system:** All poles lie inside (none is on) the unit circle, or what is the same: all poles have magnitude less than 1.
- **Marginally stable system:** One or more poles – but no multiple poles – are on the unit circle.
- **Unstable system:** At least one pole is outside the unit circle. Or: There are multiple poles on the unit circle.

The "stability areas" in the complex plane are shown in Figure 28.

Let us return to the question about the relation between the zeros and the stability. We consider the following system:

$$H_1(z) = \frac{y(z)}{u(z)} = \frac{b(z-c)}{z-p} = (z-c)H(z) \tag{276}$$

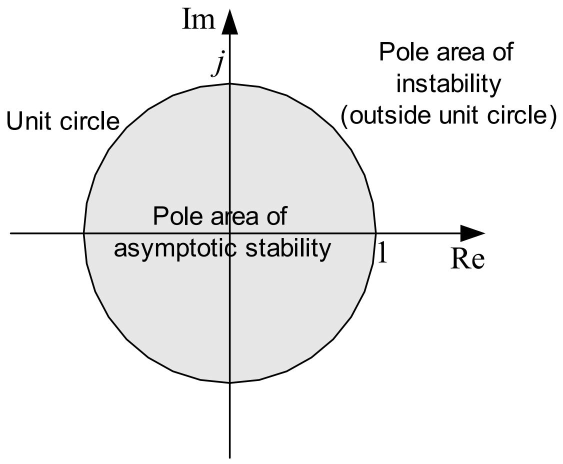
Im
j
Pole area of instability (outside unit circle)
Unit circle
Pole area of asymptotic stability
1
Re

Figure 28: "Stability areas" in the complex plane

where $H(z)$ is it the "original" system (without zero) which were analyzed above. The zero is c. $H_1(z)$ can be written as

$$H_1(z) = \frac{bz}{z-p} + \frac{-bc}{z-p} \tag{277}$$

$$= H(z) - cz^{-1}H(z) \tag{278}$$

The impulse response of $H_1(z)$ becomes

$$y_{\delta_1}(k) = y_\delta(k) - cy_\delta(k-1) \tag{279}$$

where $y_\delta(k)$ is the impulse response of $H(z)$. We see that the zero does not influence wether the steady state impulse response converges towards to zero or not. We draw the conclusion that eventual zeros in the transfer function do not influence the stability of the system.

Example 11.1 ***Stability analysis of discrete-time system***

The three responses shown in Figure 27 are actually the impulse responses in three systems each having a transfer function on the form

$$\frac{y(z)}{u(z)} = H(z) = \frac{b_1 z + b_0}{z^2 + a_1 z + a_0} \tag{280}$$

The parameters of the systems are given below:

1. Asymptotically stable system: $b_1 = 0.019$, $b_0 = 0.0190$, $a_1 = -1.885$ and $a_0 = 0.923$. The poles are

$$z_{1,\,2} = 0.94 \pm j0.19 \tag{281}$$

   They are shown in Figure 29 (the zero is indicated by a circle). The poles are inside the unity circle.

2. Marginally stable system: $b_1 = 0.020$, $b_0 = 0.020$, $a_1 = -1.96$ and $a_0 = 1.00$. The poles are

$$z_{1,\,2} = 0.98 \pm j0.20 \tag{282}$$

They are shown in Figure 29. The poles are on the unity circle.

3. Unstable system: $b_1 = 0.021$, $b_0 = 0.021$, $a_1 = -2.04$ and $a_0 = 1.08$. The poles are

$$z_{1,\,2} = 1.21 \pm j0.20 \tag{283}$$

They are shown in Figure 29. The poles are outside the unity circle.

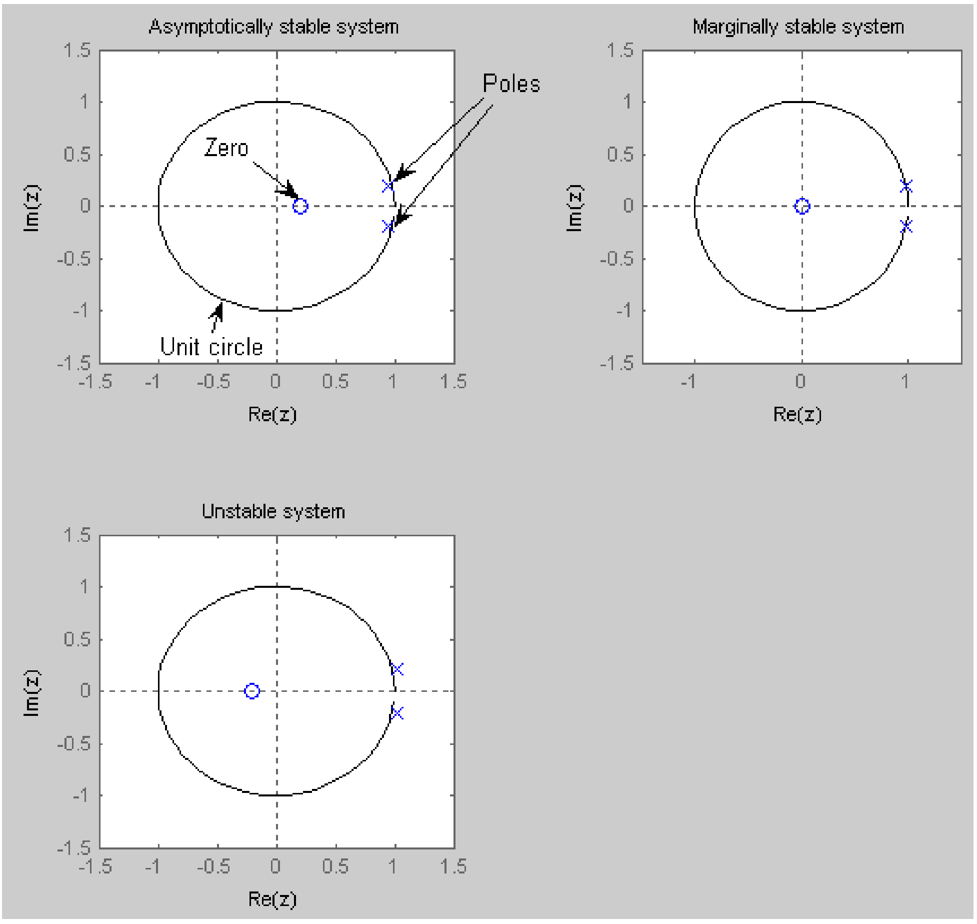

Figure 29: Example 11.1: Poles (and zeros) for the three systems each having different stability property

[End of Example 11.1]

11.3 Stability analysis of state space models

Assume that the system has the following state space model:

$$x(k+1) = A_d x(k) + B_d u(k) \tag{284}$$

$$y(k) = C_d x(k) + D_d u(k) \tag{285}$$

In Section 9.5 we derived a formula for the transfer function from $u(z)$ to $y(z)$. It is repeated here:

$$\begin{aligned} H(z) &= \frac{y(z)}{u(z)} = C_d(zI - A_d)^{-1}B_d + D_d && (286) \\ &= C_d \frac{\text{adj}(zI - A_d)}{\det(zI - A_d)} B_d + D_d && (287) \end{aligned}$$

The poles are the roots of the characteristic equation:

$$\det(zI - A_d) = 0 \tag{288}$$

The stability property is determined by the placement of the poles in the complex plane. Therefore, to determine the stability property of a state space model, we may just compute the poles of the system, and conclude about the stability as explained in Section 11.2.

The equation (288) actually defines the *eigenvalues* of the system: The eigenvalues are the z-solutions to 288. Therefore, the poles are equal to the eigenvalues, and the relation between stability properties and eigenvalues are the same relation as between stability properties and poles, cf. Section 11.2.

Internal and external stability Actually, if there are pole-zero-cancellations in (287), the cancelled poles are not among the eigenvalues, and the conclusion about stability based on the poles may be different from the conclusion based on the eigenvalues. For example, assume that an unstable pole (lying outside the unity circle) has been cancelled by a zero of same value when calculating (287), and that the uncanceled rest poles are asymptotically stable. The set of eigenvalues are equal to the set of cancelled and uncanceled poles. So the eigenvalue based analysis would say "unstable" while the pole based analysis would say "asymptotically stable".

Generally speaking, the eigenvalues determines the internal stability property, while the poles determines the external or input-output stability property of the system. In an internally unstable but internally asymptotically stable system there will one or more state variables that shows an unlimited response as time goes to infinity, while the impulse response in the output goes to zero.

For most systems there are no pole-zero cancellations, and, hence, the internal and external stability is the same.

11.4 Stability analysis of feedback (control) systems

The contents of this section (and subsections) follows the description of stability analysis of continuous-time feedback systems in Section 6.4 in the book PID Control [4].

Although it is assumed here that the feedback system is a control system, the stability analysis methods described can be applied to any (linear) feedback system.

11.4.1 Defining the feedback system

Stability analysis is important in control theory. Figure 30 shows a block diagram of a feedback control system. To analyze the stability of the feedback

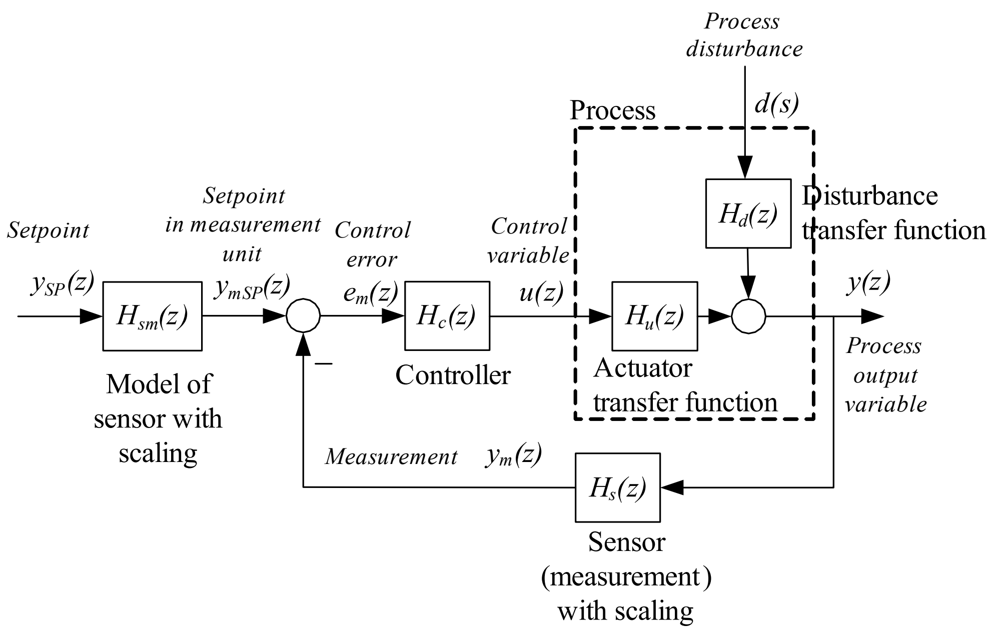

Figure 30: Feedback control system where the subsystems are represented by transfer functions

system (loop) it is sufficient to study an extracted part of the block diagram shown in Figure 30. Figure 31 shows this extracted part and how it can be made into a compact block diagram. The stability analysis will be based this compact block diagram. It has one single transfer function block containing the *loop transfer function*, $L(z)$, which is the product of the transfer functions in the loop:

$$L(z) = H_c(z)\underbrace{H_u(z)H_s(z)}_{H_p(z)} = H_c(z)H_p(z) \tag{289}$$

The closed loop transfer function is the transfer function from input (setpoint) $y_{m_{SP}}$ to output y_m (process measurement), and it denoted the *tracking transfer function*:

$$T(z) = \frac{L(z)}{1+L(z)} = \frac{\frac{n_L(z)}{d_L(z)}}{1+\frac{n_L(z)}{d_L(z)}} = \frac{n_L(z)}{d_L(z)+n_L(z)} \tag{290}$$

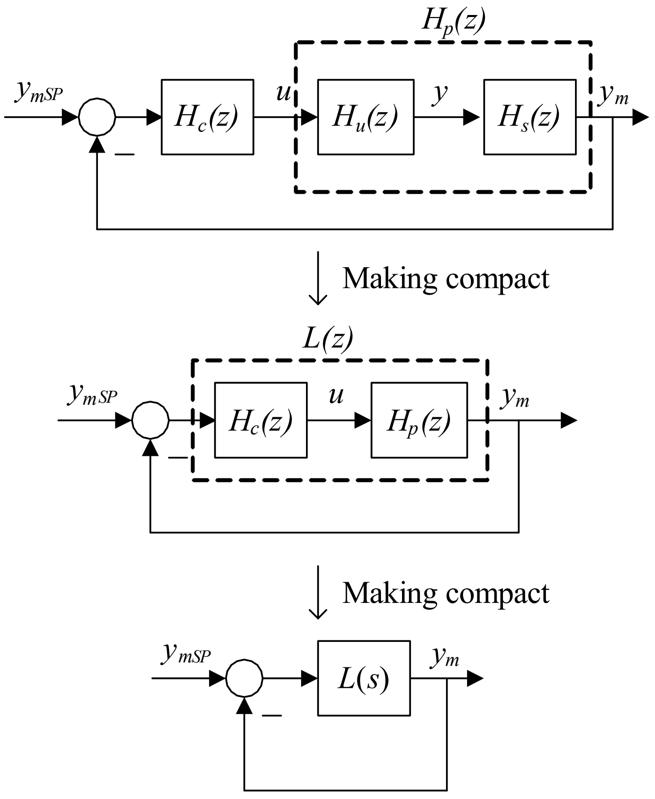

Figure 31: Converting an extracted part of the detailed block diagram in Figure 30 into a compact block diagram. $L(z)$ is the loop transfer function.

where $n_L(z)$ and $d_L(z)$ are the numerator and denominator polynomials of $L(z)$, respectively. The stability of the feedback system is determined by the stability of $T(z)$.

11.4.2 Pole placement based stability analysis

The *characteristic polynomial* of the tracking transfer function (290) is

$$c(z) = d_L(z) + n_L(z) \tag{291}$$

Hence, the stability of the control system is determined by the placement of the roots of (291) in the complex plane.

Example 11.2 ***Pole based stability analysis of feedback system***

Assume given a control system where a P controller controls an integrating process. The controller transfer function is

$$H_c(z) = K_p \tag{292}$$

and the process transfer function is, cf. (248),

$$H_p(z) = \frac{K_i h}{z-1} \tag{293}$$

We assume that $K_i = 1$ and $h = 1$. The loop transfer function becomes

$$L(z) = H_c(z)H_p(z) = \frac{K_p}{z-1} = \frac{d_L(z)}{n_L(z)} \tag{294}$$

We will calculate the range of values of K_p that ensures asymptotic stability of the control system.

The characteristic polynomial is, cf. (291),

$$c(z) = d_L(z) + n_L(z) = K_p + z - 1 \tag{295}$$

The pole is

$$p = 1 - K_p \tag{296}$$

The feedback system is asymptotically stable if p is inside the unity circle or has magnitude less than one:

$$|p| = |1 - K_p| < 1 \tag{297}$$

which is satisfied with

$$0 < K_p < 2 \tag{298}$$

Assume as an example that $K_p = 1.5$. Figure 32 shows the step response in y_m for this value of K_p.

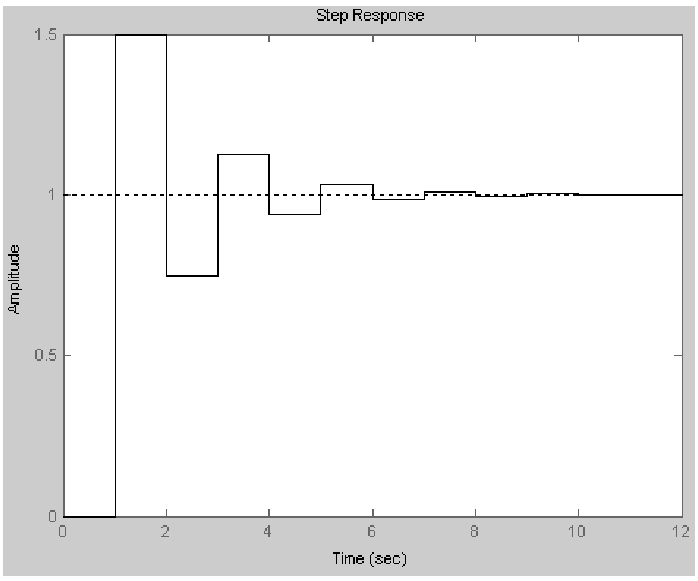

Figure 32: Example 11.2: Step resonse in y_m. There is a step of amplitude $y_{m_{SP}}$.

[End of Example 11.2]

11.4.3 Nyquist's stability criterion for feedback systems

The Nyquist's stability criterion is a graphical tool for stability analysis of feedback systems. The traditional stability analysis based on the frequency response of $L(z)$ in a Bode diagram is based on Nyquist's stability criterion. In the following the Nyquist's stability criterion for discrete-time systems will be described briefly. Fortunately, the principles and methods of the stability analysis are much the same as for continuous-time systems [4].

In Section 11.4.1 we found that the characteristic polynomial of a feedback system is

$$c(z) = d_L(z) + n_L(z) \tag{299}$$

The poles of the feedback system are the roots of the equation $c(z) = 0$. These roots are the same as the roots of the following equation:

$$\frac{d_L(z) + n_L(z)}{d_L(z)} = 1 + \frac{n_L(z)}{d_L(z)} \equiv 1 + L(z) = 0 \tag{300}$$

which we denote the characteristic equation. In the discrete-time case, as in the continuous-time case, the stability analysis is about determining the number of *unstable* poles of the feedback system. Such poles lie *outside the unit circle* in the z plane.

(300) is the equation from which the Nyquist's stability criterion will be derived. In the derivation we will use the Argument Variation Principle:

Argument Variation Principle: Given a function $f(z)$ where z is a complex number. Then $f(z)$ is a complex number, too. As with all complex numbers, $f(z)$ has an angle or argument. If z follows a closed contour Γ (gamma) in the complex z-plane which encircles a number of poles and a number of zeros of $f(z)$, see Figure 33, then the following applies:

$$\underset{\Gamma}{\arg} f(z) = 360^\circ \cdot (\text{number of zeros minus number of poles of } f(z) \text{ inside } \Gamma) \tag{301}$$

where $\arg_\Gamma f(z)$ means the change of the angle of $f(z)$ when z has followed Γ once in positive direction of circulation .

For the purpose of stability analysis of feedback systems, we let the function $f(z)$ in the Argument Variation Principle be

$$f(z) = 1 + L(z) \tag{302}$$

The Γ contour must encircle the entire complex plane outside the unit circle in the z-plane, so that we are certain that all poles and zeros of $1 + L(z)$ are encircled. From the Argument Variation Principle we have (below UC is unit circle):

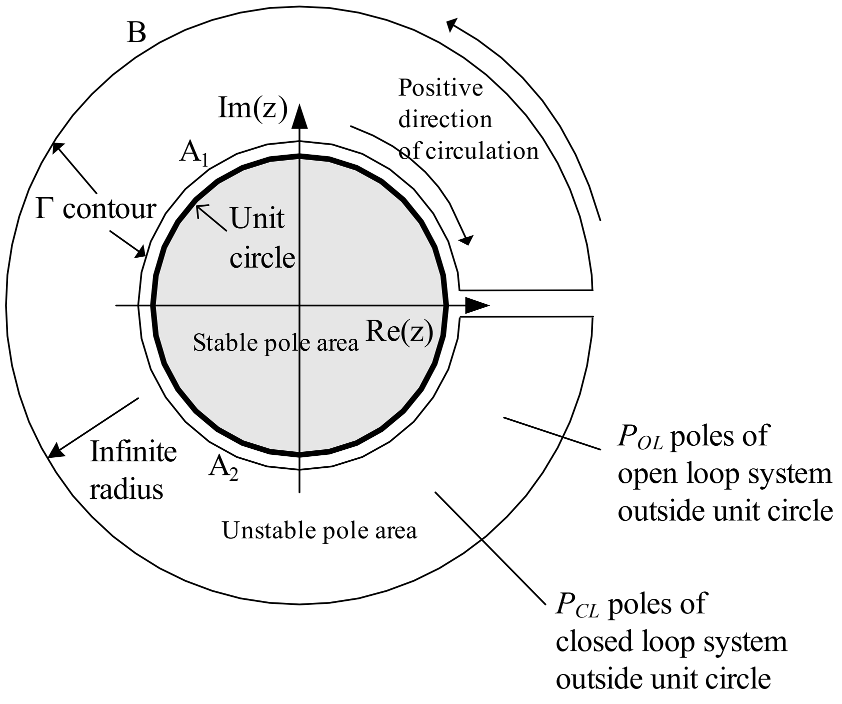
B
Im(z)
Positive direction of circulation
A1
Γ contour
Unit circle
Re(z)
Stable pole area
Infinite radius
A2
Unstable pole area
P_{OL} poles of open loop system outside unit circle
P_{CL} poles of closed loop system outside unit circle

Figure 33: In the Nyquist's stability criterion for discrete-time systems, the Γ-contour in the Argument Variation Principle [4] must encircle the whole area outide the unit circle. The letters A and B identifies parts of the Γ-contour (cf. the text).

$$
\begin{aligned}
\arg_{\Gamma}[1+L(z)] &= \arg_{\Gamma}\frac{d_L(z)+n_L(z)}{d_L(z)} && (303)\\
&= 360^\circ\cdot(\text{number of roots of } (d_L+n_L) \text{ outside UC}^{15}\\
&\quad \text{minus number roots of } d_L \text{ outside UC}) && (304)\\
&= 360^\circ\cdot(\text{number poles of closed loop system outside UC}\\
&\quad \text{minus number poles of open system outside UC})\\
&= 360^\circ\cdot(P_{CL}-P_{OL}) && (305)
\end{aligned}
$$

By "open system" we mean the (imaginary) system having transfer function $L(z) = n_L(z)/d_L(z)$, i.e., the original feedback system with the feedback broken. The poles of the open system are the roots of $d_L(z) = 0$.

Finally, we can formulate the Nyquist's stability criterion. But before we do that, we should remind ourselves what we are after, namely to be able to determine the number poles P_{CL} of the closed loop system outside the unit circle. These poles determines whether the closed loop system (the control system) is asymptotically stable or not. *If $P_{CL} = 0$ the closed loop system is asymptotically stable.*

Nyquist's Stability Criterion: Let P_{OL} be the number of poles of the open system outside the unit circle, and let $\arg_\Gamma [1 + L(z)]$ be the angular change of the vector $[1 + L(z)]$ as z have followed the Γ contour once in positive direction of circulation. Then, the number poles P_{CL} of the closed loop system outside the unit circle, is

$$P_{CL} = \frac{\arg_\Gamma [1 + L(z)]}{360^\circ} + P_{OL} \tag{306}$$

If $P_{CL} = 0$, the closed loop system is asymptotically stable.

Let us take a closer look at the terms on the right side of (306): P_{OL} are the number of the roots of $d_L(z)$, and there should not be any problem calculating that number. What about determining the angular change of the vector $1 + L(z)$? Figure 34 shows how the vector (or complex number) $1 + L(z)$ appears in a *Nyquist diagram* for a typical plot of $L(z)$. A Nyquist diagram is simply a Cartesian diagram of the complex plane in which L is plotted. $1 + L(z)$ is the vector from the point $(-1,\ 0j)$, which is denoted the *critical point*, to the Nyquist curve of $L(z)$.

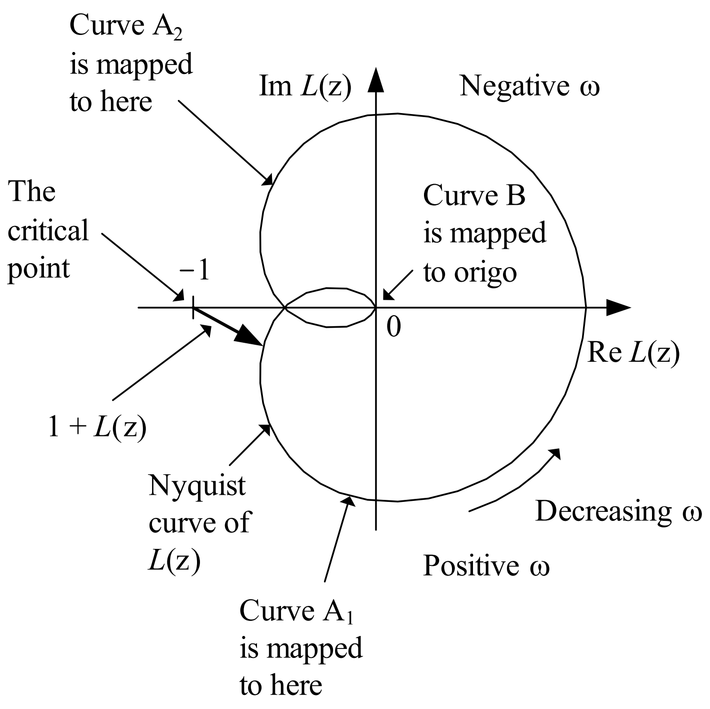

Figure 34: Typical Nyquist curve of $L(z)$. The vector $1 + L(z)$ is drawn.

More about the Nyquist curve of $L(z)$ Let us take a more detailed look at the Nyquist curve of L as z follows the Γ contour in the z-plane, see Figure 33. In practice, the denominator polynomial of $L(z)$ has higher order than the numerator polynomial. This implies that $L(z)$ is mapped to the origin of the

Nyquist diagram when $|z| = \infty$, which corresponds to contour B in Figure 33. The unit circle, contour A_1 plus contour A_2 in Figure 33, constitutes (most of) the rest of the Γ contour. There, z has the value

$$z = e^{j\omega h} \tag{307}$$

Consequently, the loop transfer function becomes $L(e^{j\omega h})$ when z is on the unit circle. For z on A_1 in Figure 33, ω is positive, and hence $L(e^{j\omega h})$ is the *frequency response* of L. A consequence of this is that we can determine the stability property of a feedback system by just looking at the frequency response of the loop transfer function, $L(e^{j\omega h})$. For z on A_2 in Figure 33 ω is negative, and the frequency response has a pure mathematical meaning. From general properties of complex functions,

$$|L(e^{j(-\omega)h})| = |L(e^{j(+\omega)h})| \tag{308}$$

and

$$\arg L(e^{j(-\omega)h}) = -\arg L(e^{j(+\omega)h}) \tag{309}$$

Therefore the Nyquist curve of $L(z)$ for $\omega < 0$ will be identical to the Nyquist curve for $\omega > 0$, but mirrored about the real axis. Thus, we only need to know how $L(e^{j\omega h})$ is mapped for $\omega \geq 0$. The rest of the Nyquist curve then comes by itself! Actually we need not draw more of the Nyquist curve (for $\omega > 0$) than what is sufficient for determining if the critical point is encircled or not.

If $L(z)$ has poles on the unit circle, i.e. if $d_L(z)$ has roots in the unit circle, we must let the Γ contour pass outside these poles, otherwise the function $1 + L(z)$ is not analytic on Γ. Assume the common case that $L(z)$ contain a pure integrator (which may be the integrator of the PID controller). This implies that $L(z)$ contains $1/(z-1)$ as factor. We let the Γ contour pass just outside the point $z = 1$ in such a way that the point is not encircled by Γ. It may be shown that this passing maps z onto an infinitely large semicircle encircling the right half plane in Figure 34.

11.4.4 Nyquist's special stability criterion

In most cases the open system is stable, that is, $P_{OL} = 0$. (306) then becomes

$$P_{CL} = \frac{\arg_\Gamma[L(z)]}{360^\circ} \tag{310}$$

This implies that the feedback system is asymptotically stable if the Nyquist curve does not encircle the critical point. This is the *Nyquist's special stability criterion* or the *Nyquist's stability criterion for open stable systems.*

The Nyquist's special stability criterion can also be formulated as follows: *The feedback system is asymptotically stable if the Nyquist curve of L has the critical point on its left side for increasing ω.*

Another way to formulate Nyquist's special stability criterion involves the following characteristic frequencies: *Amplitude crossover frequency* ω_c and *phase crossover frequency* ω_{180}. ω_c is the frequency at which the $L(e^{j\omega h})$

curve crosses the unit circle, while ω_{180} is the frequency at which the $L(e^{j\omega h})$ curve crosses the negative real axis. In other words:

$$|L(e^{j\omega_c h})| = 1 \tag{311}$$

and

$$\arg L(e^{j\omega_{180} h}) = -180^\circ \tag{312}$$

See Figure 35. Note: The Nyquist diagram contains no explicit frequency axis.

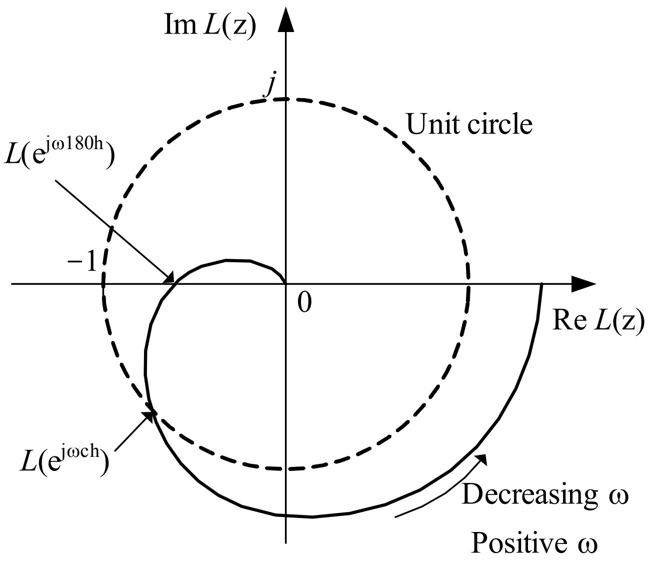

Figure 35: Definition of amplitude crossover frequency ω_c and phase crossover frequency ω_{180}

We can now determine the stability properties from the relation between these two crossover frequencies:

- Asymptotically stable closed loop system: $\omega_c < \omega_{180}$
- Marginally stable closed loop system: $\omega_c = \omega_{180}$
- Unstable closed loop system: $\omega_c > \omega_{180}$

11.4.5 Stability margins: Gain margin GM and phase margin PM

An asymptotically stable feedback system may become marginally stable if the loop transfer function changes. The *gain margin GM* and the *phase margin PM* [radians or degrees] are *stability margins* which in their own ways expresses how large parameter changes can be tolerated before an asymptotically stable system becomes marginally stable. Figure 36 shows the stability margins defined in the Nyquist diagram. GM is the (multiplicative,

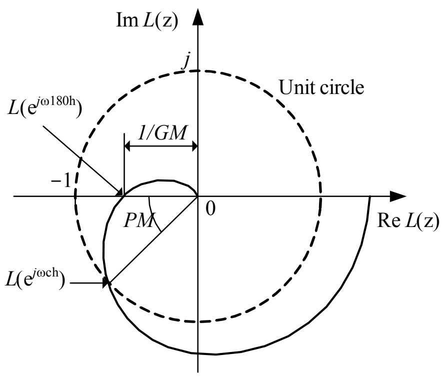

Figure 36: Gain margin GM and phase margin PM defined in the Nyquist diagram

not additive) increase of the gain that L can tolerate at ω_{180} before the L curve (in the Nyquist diagram) passes through the critical point. Thus,

$$\left|L(e^{j\omega_{180}h})\right| \cdot GM = 1 \tag{313}$$

which gives

$$GM = \frac{1}{|L(e^{j\omega_{180}h})|} = \frac{1}{|\mathrm{Re}\, L(e^{j\omega_{180}h})|} \tag{314}$$

(The latter expression in (314) is because at ω_{180}, $\mathrm{Im}\, L = 0$ so that the amplitude is equal to the absolute value of the real part.)

If we use decibel as the unit (like in the Bode diagram which we will soon encounter), then

$$GM \text{ [dB]} = -\left|L(e^{j\omega_{180}h})\right| \text{ [dB]} \tag{315}$$

The phase margin PM is the phase reduction that the L curve can tolerate at ω_c before the L curve passes through the critical point. Thus,

$$\arg L(e^{j\omega_c h}) - PM = -180^\circ \tag{316}$$

which gives

$$PM = 180^\circ + \arg L(e^{j\omega_c h}) \tag{317}$$

We can now state as follows: The feedback (closed) system is asymptotically stable if

$$GM > 0\text{dB} \;= 1 \text{ and } PM > 0^\circ \tag{318}$$

This criterion is often denoted the *Bode-Nyquist stability criterion.*

Reasonable ranges of the stability margins are

$$2 \approx 6\text{dB} \leq GM \leq 4 \approx 12\text{dB} \tag{319}$$

and

$$30^\circ \leq PM \leq 60^\circ \tag{320}$$

The larger values, the better stability, but at the same time the system becomes more sluggish, dynamically. If you are to use the stability margins as design criterias, you can use the following values (unless you have reasons for specifying other values):

$$GM \geq 2.5 \approx 8\text{dB and } PM \geq 45^\circ \tag{321}$$

For example, the controller gain, K_p, can be adjusted until one of the inequalities becomes an equality.[16]

It can be shown that for $PM \leq 70^\circ$, the damping of the feedback system approximately corresponds to that of a second order system with relative damping factor

$$\zeta \approx \frac{PM}{100^\circ} \tag{322}$$

For example, $PM = 50^\circ \sim \zeta = 0.5$.

Example 11.3 ***Stability analysis in Nyquist diagram***

Given the following continuous-time process transfer function:

$$H_p(s) = \frac{y_m(z)}{u(z)} = \frac{K}{\left(\frac{s}{\omega_0}\right)^2 + 2\zeta\frac{s}{\omega_0} + 1}e^{-\tau s} \tag{323}$$

with parameter values

$$K = 1;\ \zeta = 1;\ \omega_0 = 0.5\text{rad/s};\ \tau = 1\text{s} \tag{324}$$

The process is controlled by a discrete-time PI-controller having the following z-transfer function, which can be derived by taking the z-transform of the PI control function (31),

$$H_c(z) = \frac{K_p\left(1 + \frac{h}{T_i}\right)z - K_p}{z - 1} \tag{325}$$

where the time-step (or sampling interval) is

$$h = 0.2\text{s} \tag{326}$$

Tuning the controller with the Ziegler-Nichols' closed-loop method [4] in a simulator gave the following controller parameter settings:

$$K_p = 2.0;\ T_i = 5.6\text{s} \tag{327}$$

To perform the stability analysis of the discrete-time control system $H_p(s)$ is discretized assuming first order hold. The result is

$$H_{p_d}(z) = \frac{0.001209z + 0.001169}{z^2 - 1.902z + 0.9048}z^{-10} \tag{328}$$

The loop transfer function is

$$L(z) = H_c(z)H_{p_d}(z) \tag{329}$$

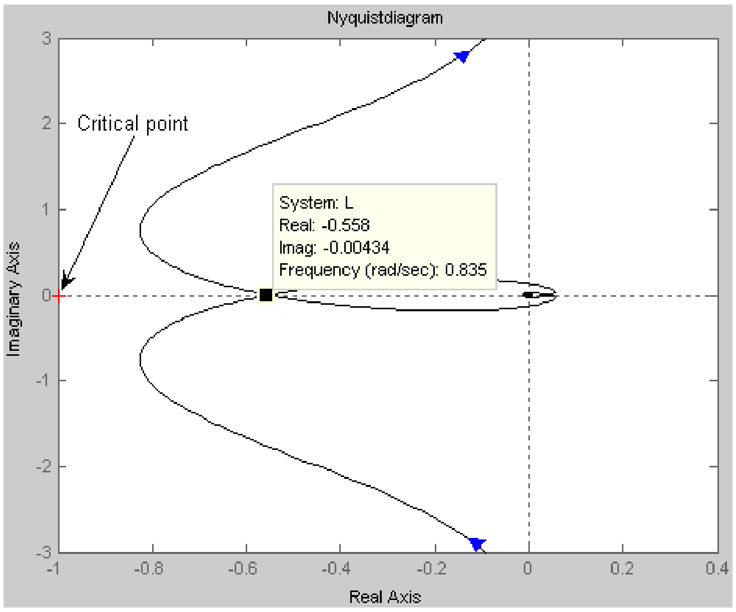

Figure 37: Example 11.3: Nyquist diagram of $L(z)$

Figure 37 shows the Nyquist plot of $L(z)$. From the Nyquist diagram we read off

$$\omega_{180} = 0.835 \text{rad/s} \tag{330}$$

and

$$\operatorname{Re} L(e^{j\omega_{180}h}) = -0.558 \tag{331}$$

which gives the following gain margin, cf. (314),

$$GM = \frac{1}{|\operatorname{Re} L(e^{j\omega_{180}h})|} = \frac{1}{|-0.558|} = 1.79 = 5.1\text{dB} \tag{332}$$

The phase margin can be found to be

$$PM = 35^\circ \tag{333}$$

Figure 38 shows the step response in y_m (unity step in setpoint $y_{m_{SP}}$).

[End of Example 11.3]

11.4.6 Stability margin: Maximum of sensitivity function

The sensitivity (transfer) function S is frequently used to analyze feedback control systems in various aspects. [4] It is

$$S(z) = \frac{1}{1 + L(z)} \tag{334}$$

[16] But you should definitely check the behaviour of the control system by simulation, if possible.

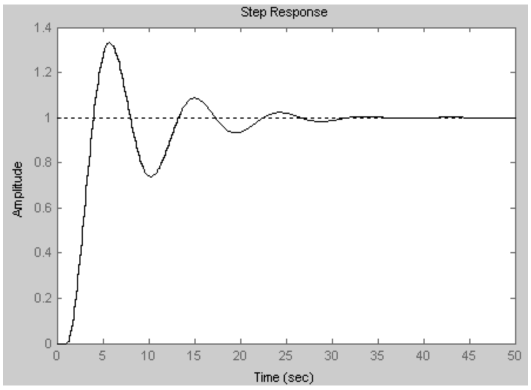

Figure 38: Example 11.3: Step response in y_m (unity step in setpoint $y_{m_{SP}}$)

where $L(z)$ is the loop transfer function. S has various interpretations: One is being the transfer function from setpoint $y_{m_{SP}}$ to control error e in the block diagram in Figure 30:

$$S(z) = \frac{e_m(z)}{y_{m_{SP}}(z)} \tag{335}$$

A Bode plot of $S(z)$ can be used in frequency response analysis of the *setpoint tracking* property of the control system. One other interpretation of S is being the ratio of the z-transform of the control error in closed en loop control and the control error in open loop control, when this error is caused by an excitation in the disturbance d, cf. Figure 30. Thus,

$$S(z) = \frac{e_{\text{disturb}}(z)_{\text{closed loop system}}}{e_{\text{disturb}}(z)_{\text{open loop system}}} \tag{336}$$

(336) expresses the *disturbance compensation* property of the control system.

Back to the stability issue: A measure of a stability margin alternative to the gain margin and the phase margin is the minimum distance from the $L(e^{j\omega h})$ curve to the critical point. This distance is $\left|1 + L(e^{j\omega h})\right|$, see Figure 39. So, we can use the minimal value of $\left|1 + L(e^{j\omega h})\right|$ as a stability margin. However, it is more common to take the inverse of the distance: Thus, a stability margin is the *maximum* value of $1/\left|1 + L(e^{j\omega h})\right|$. And since $1/[1 + L(z)]$ is the sensitivity function $S(z)$ [4], then $\left|S(e^{j\omega h})\right|_{\max}$ represents a stability margin. Reasonable values are in the range

$$1.5 \approx 3.5\text{dB} \leq \left|S(e^{j\omega h})\right|_{\max} \leq 3.0 \approx 9.5\text{dB} \tag{337}$$

If you use $\left|S(e^{j\omega h})\right|_{\max}$ as a criterion for adjusting controller parameters, you can use the following criterion (unless you have reasons for some other specification):

$$\left|S(e^{j\omega h})\right|_{\max} = 2.0 \approx 6\text{dB} \tag{338}$$

Example 11.4 $|S|_{\max}$ ***as stability margin***

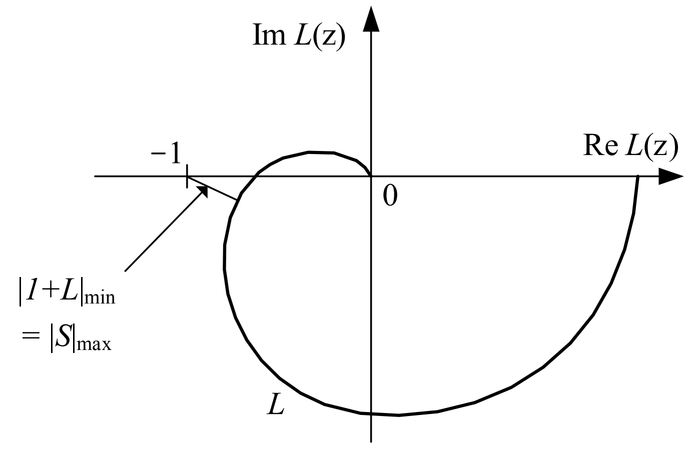

Figure 39: The minimum distance between the $L(e^{j\omega h})$ curve and the critical point can be interpreted as a stability margin. This distance is $|1+L|_{\min} = |S|_{\max}$.

See Example 11.3. It can be shown that

$$|S|_{\max} = 8.9\text{dB} = 2.79 = \frac{1}{0.36} = \frac{1}{|1+L|_{\min}} \tag{339}$$

[End of Example 11.4]

Frequency of the sustained oscillations There are sustained oscillations in a marginally stable system. *The frequency of these oscillations is* $\omega_c = \omega_{180}$.This can be explained as follows: In a marginally stable system, $L(e^{j(\pm\omega_{180})h}) = L(e^{j(\pm\omega_c)h}) = -1$. Therefore, $d_L(e^{j(\pm\omega_{180})h}) + n_L(e^{j(\pm\omega_{180})h}) = 0$, which is the characteristic equation of the closed loop system with $e^{j(\pm\omega_{180})h}$ inserted for z. Therefore, the system has $e^{j(\pm\omega_{180})h}$ among its poles. The system usually have additional poles, but they lie in the left half plane. The poles $e^{j(\pm\omega_{180})h}$ leads to sustained sinusoidal oscillations. Thus, ω_{180} (or ω_c) is the frequency of the sustained oscillations in a marginally stable system. This information can be used for tuning a PID controller with the Ziegler-Nichols' frequency response method [4].

11.4.7 Stability analysis in a Bode diagram

It is most common to use a Bode diagram, not a Nyquist diagram, for frequency response based stability analysis of feedback systems. The Nyquist's Stability Criterion says: The closed loop system is marginally stable if the Nyquist curve (of L) goes through the critical point, which is the point $(-1, 0)$. But where is the critical point in the Bode diagram? The critical point has phase (angle) $-180°$ and amplitude $1 = 0$dB. The critical point therefore constitutes two lines in a Bode diagram: The 0dB line in the amplitude diagram and the $-180°$ line in the phase diagram. Figure 40 shows typical L

curves for an asymptotically stable closed loop system. In the figure, GM, PM, ω_c and ω_{180} are indicated.

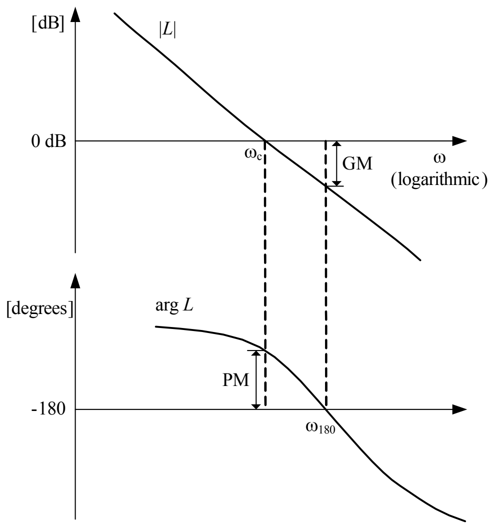

Figure 40: Typical L curves of an asymptotically stable closed loop system with GM, PM, ω_c and ω_{180} indicated

Example 11.5 ***Stability analysis in Bode diagram***

See Example 11.3. Figure 41 shows a Bode plot of $L(e^{j\omega h})$. The stability margins are shown in the figure. They are

$$GM = 5.12\text{dB} = 1.80 \tag{340}$$

$$PM = 35.3^\circ \tag{341}$$

which is in accordance with Example 11.3.

[End of Example 11.5]

A z-transform

In this appendix the capital letter $F(z)$ is used for the z-transformed time function $f(kT) = f(k)$. In other sections of this document lowercase letters are used for both the time function $f(k)$ and its z-transformed function $f(z)$.

A.1 Properties of the z-transform

Linear combination:

$$k_1F_1(z) + k_2F_2(z) \Longleftrightarrow k_1f_1(k) + k_2f_2(k) \tag{342}$$

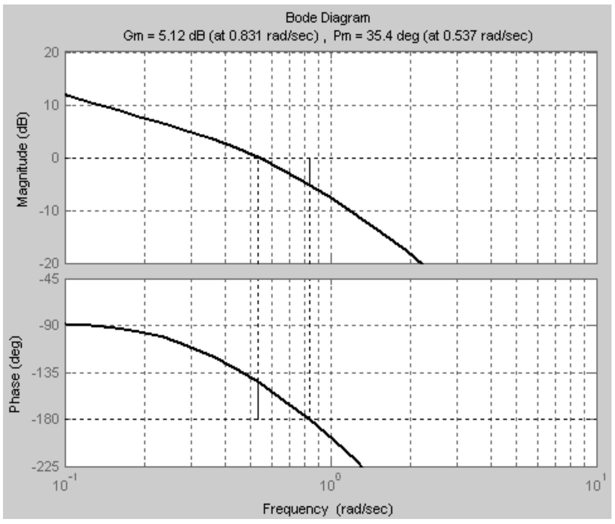

Figure 41: Example 11.5: Bode plot of L

Special case:

$$k_1F(z) \leftrightarrow k_1f(k) \tag{343}$$

Time delay (time shift backward) n time-steps:

$$z^{-n}F(z) \quad \Longleftrightarrow \quad f(k-n) \tag{344}$$

Time advance (time shift forward) n time-steps:

$$z^{n}F(z) \quad \Longleftrightarrow \quad f(k+n) \tag{345}$$

Convolution:

$$F_1(z)F_2(z) \quad \Longleftrightarrow \quad f_1(k) * f_2(k) = \sum_{l=-\infty}^{\infty} f_1(k-l)f_2(l) \tag{346}$$

Final value theorem:

$$\lim_{z\to 1}(z-1)F(z) \quad \Longleftrightarrow \quad \lim_{k\to\infty} f(k) \tag{347}$$

$$-z\frac{dF(z)}{dz} \quad \Longleftrightarrow \quad k \cdot f(k) \tag{348}$$

$$\frac{z}{z-1}\,F(z) \quad \Longleftrightarrow \quad \sum_{n=0}^{k} f(n) \tag{349}$$

A.2 z-transform pairs

Below are several important z-transform pairs showing discrete-time functions and their corresponding z-transformations. The time functions are defined for $k \geq 0$.

$$\text{Unity impulse at time-step } k\text{: } \delta(k) \quad \Longleftrightarrow \quad z^k \tag{350}$$

$$\text{Unity impulse at time-step } k=0\text{: } \delta(0) \quad \Longleftrightarrow \quad 1 \tag{351}$$

$$\text{Unity step at time-step } k=0\text{: } 1 \Longleftrightarrow \quad \frac{z}{z-1} \tag{352}$$

$$\text{Time exponential: } a^k \quad \Longleftrightarrow \quad \frac{z}{z-a} \tag{353}$$

$$1-a^k \quad \Longleftrightarrow \quad \frac{z\,(1-a)}{(z-1)\,(z-a)} \tag{354}$$

$$kha^k \quad \Longleftrightarrow \quad \frac{zha}{(z-a)^2} \tag{355}$$

$$a_1{}^k - a_2{}^k \quad \Longleftrightarrow \quad \frac{(a_1-a_2)\,z}{(z-a_1)\,(z-a_2)} \tag{356}$$

$$e^{-akh}\cos bkh \quad \Longleftrightarrow \quad \frac{z\left(z-e^{-ah}\cos bh\right)}{z^2-(2e^{-ah}\cos bh)\,z+e^{-2ah}} \tag{357}$$

$$e^{-akh}\sin bkh \quad \Longleftrightarrow \quad \frac{ze^{-ah}\sin bh}{z^2-(2e^{-ah}\cos bh)\,z+e^{-2ah}} \tag{358}$$

$$1-e^{-akT}\left(\cos bkT+\frac{a}{b}\sin bkT\right) \Longleftrightarrow \frac{z\,(Az+B)}{(z-1)\,(z^2-(2e^{-aT}\cos bT)\,z+e^{-2aT})} \tag{359}$$

where

$$A = 1-e^{-aT}\cos bT - \frac{a}{b}e^{-aT}\sin bT \tag{360}$$

and

$$B = e^{-2aT}+\frac{a}{b}e^{-aT}\sin bT - e^{-aT}\cos bT \tag{361}$$

References

[1] G. Franklin and J. D. Powell, **Digital Control of Dynamic Systems**, Addison Wesley, 1980

[2] F. Haugen: Advanced Control, **Tapir Academic Press** (Norway), 2004 (to appear)

[3] F. Haugen: Dynamic Systems – Modeling, Analysis and Simulation, **Tapir Academic Press**, 2004

[4] F. Haugen: PID Control, **Tapir Academic Press**, 2004

[5] G. Olsson: **Industrial Automation**, Lund University, 2003