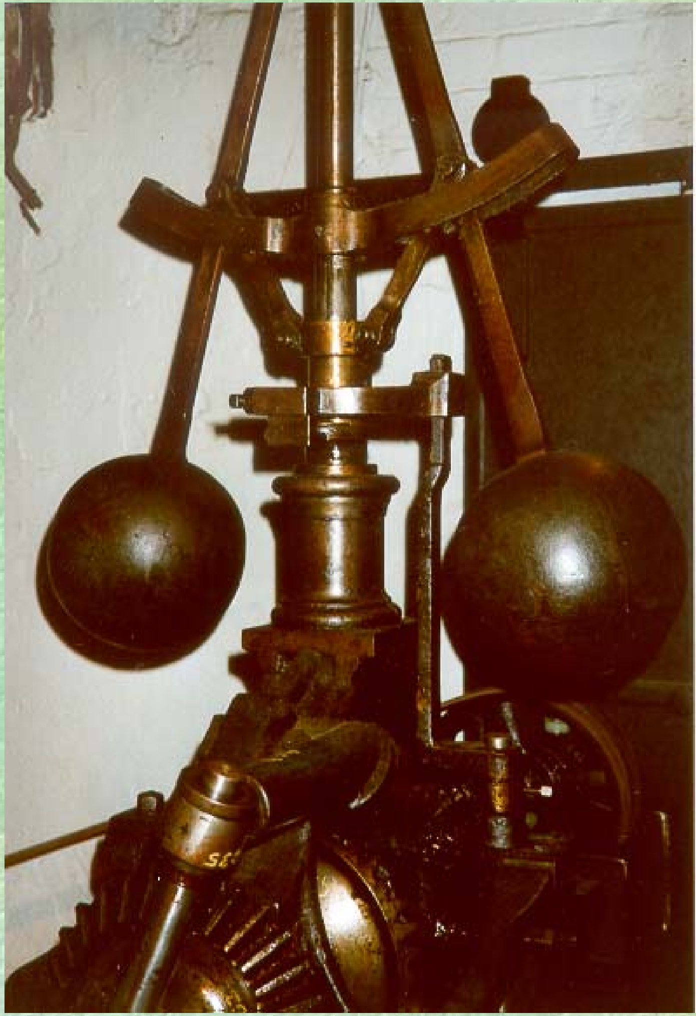
DIGITAL CONTROL SYSTEMS
&
COMPUTER AIDED DESIGN

Motivation for Control Engineering

Feedback control has a long history which began with the early desire of humans to harness the materials and forces of nature to their advantage. Early examples of control devices include clock regulating systems and mechanisms for keeping wind-mills pointed into the wind.

Modern industrial plants have sophisticated control systems which are crucial to their successful operation.



This flyball governor is in the same cotton factory in Manchester. However, this particular governor was used to regulate the speed of a water wheel driven by the flow of the river. The governor is quite large as can be gauged by the outline of the door frame behind the governor.

Improved control is a key enabling technology underpinning:

- enhanced product quality
- waste minimization
- environmental protection
- greater throughput for a given installed capacity
- greater yield
- deferring costly plant upgrades, and
- higher safety margins

Types of Control System Design

Control system design also takes several different forms and each requires a slightly different approach.

The control engineer is further affected by where the control system is in its lifecycle, e.g.:

- Initial "grass roots" design
- Commissioning and Tuning
- Refinement and Upgrades
- Forensic studies

System Integration

Success in control engineering depends on taking a holistic viewpoint. Some of the issues are:

- plant, i.e. the process to be controlled
- objectives
- sensors
- actuators
- communications
- computing
- architectures and interfacing
- algorithms
- accounting for disturbances and uncertainty

Plant

The physical layout of a plant is an intrinsic part of control problems. Thus a control engineer needs to be familiar with the "physics" of the process under study. This includes a rudimentary knowledge of the basic energy balance, mass balance and material flows in the system.

Objectives

Before designing sensors, actuators or control architectures, it is important to know the goal, that is, to formulate the control objectives. This includes

- what does one want to achieve (energy reduction, yield increase,...)
- what variables need to be controlled to achieve these objectives
- what level of performance is necessary (accuracy, speed,...)

Sensors

Sensors are the *eyes* of control enabling one to *see* what is going on. Indeed, one statement that is sometimes made about control is:

If you can measure it, you can control it.

Actuators

Once sensors are in place to report on the *state* of a process, then the next issue is the ability to affect, or actuate, the system in order to move the process from the current state to a desired state

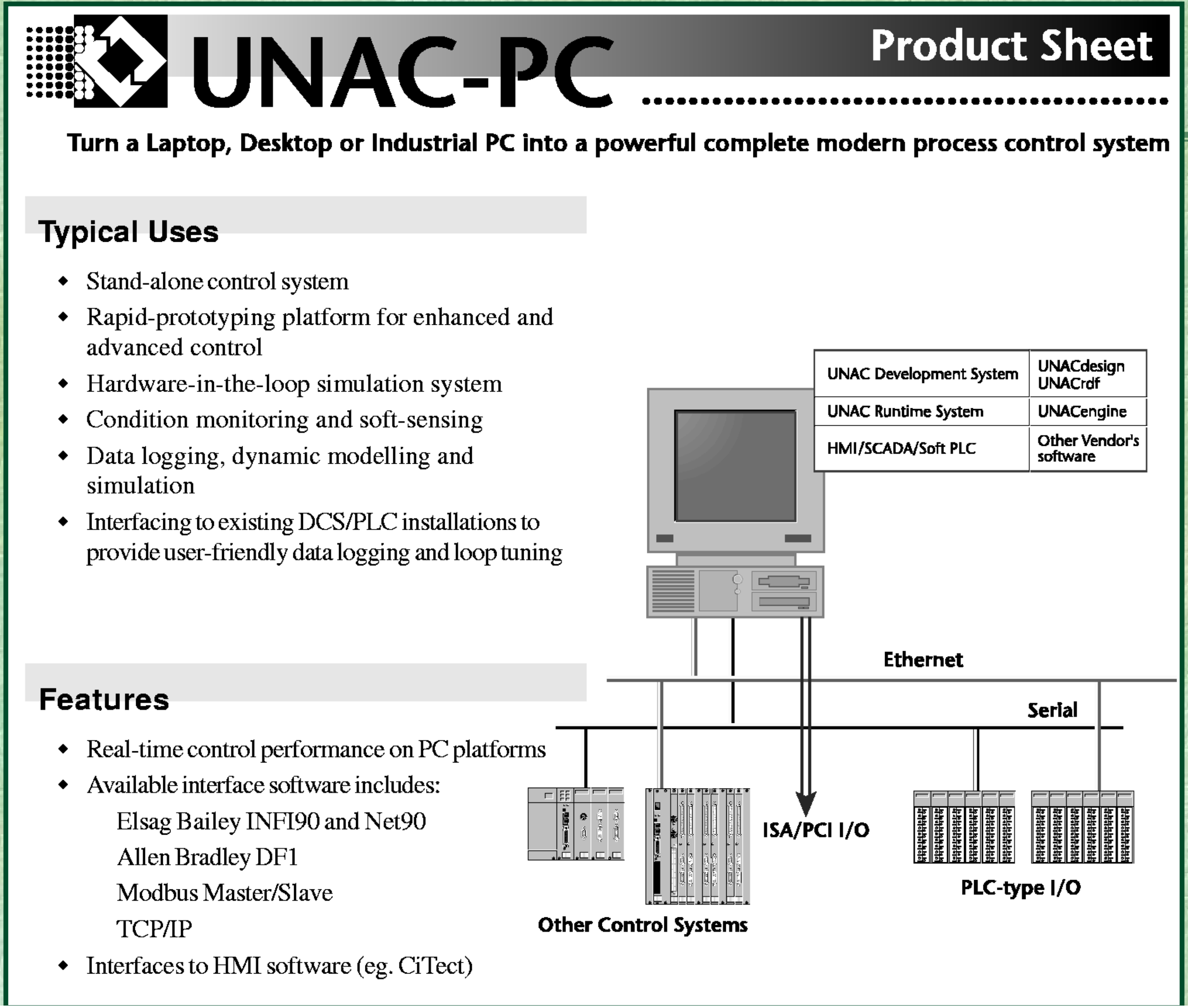
Communications

Interconnecting sensors to actuators, involves the use of communication systems. A typical plant can have many thousands of separate signals to be sent over long distances. Thus the design of communication systems and their associated protocols is an increasingly important aspect of modern control engineering.

Computing

In modern control systems, the connection between sensors and actuators is invariably made via a computer of some sort. Thus, computer issues are necessarily part of the overall design. Current control systems use a variety of computational devices including DCS's (Distributed Control Systems), PLC's (Programmable Logic Controllers), PC's (Personal Computers), etc.

A modern computer based rapid prototyping system



UNAC-PC

Product Sheet

Turn a Laptop, Desktop or Industrial PC into a powerful complete modern process control system

Typical Uses

- Stand-alone control system
- Rapid-prototyping platform for enhanced and advanced control
- Hardware-in-the-loop simulation system
- Condition monitoring and soft-sensing
- Data logging, dynamic modelling and simulation
- Interfacing to existing DCS/PLC installations to provide user-friendly data logging and loop tuning

Features

- Real-time control performance on PC platforms
- Available interface software includes:
  - Elsag Bailey INFI90 and Net90
  - Allen Bradley DF1
  - Modbus Master/Slave
  - TCP/IP
- Interfaces to HMI software (eg. CiTect)

UNAC Development System	UNACdesign UNACrdf
UNAC Runtime System	UNACengine
HMI/SCADA/Soft PLC	Other Vendor's software

Architectures and interfacing

The issue of what to connect to what is a non-trivial one in control system design. One may feel that the best solution would always be to bring all signals to a central point so that each control action would be based on complete information (leading to so called, centralized control). However, this is rarely (if ever) the best solution in practice. Indeed, there are very good reasons why one may not wish to bring all signals to a common point. Obvious objections to this include complexity, cost, time constraints in computation, maintainability, reliability, etc.

Algorithms

Finally, we come to the real *heart* of control engineering i.e. the algorithms that connect the sensors to the actuators. It is all to easy to underestimate this final aspect of the problem.

As a simple example from our everyday experience, consider the problem of playing tennis at top international level. One can readily accept that one needs good eye sight (sensors) and strong muscles (actuators) to play tennis at this level, but these attributes are not sufficient. Indeed eye-hand coordination (i.e. control) is also crucial to success.

Disturbances and Uncertainty

One of the things that makes control science interesting is that all real life systems are acted on by noise and external disturbances. These factors can have a significant impact on the performance of the system. As a simple example, aircraft are subject to disturbances in the form of wind-gusts, and cruise controllers in cars have to cope with different road gradients and different car loadings.

In order to make progress in control engineering (as in any field) it is important to be able to justify the associated expenditure. This usually takes the form of a cost benefit analysis.

Signals and systems terminology

	Tangible examples	*Examples of mathematical approximation*	*Examples of properties*
Signals	set point, control input, disturbances, measurements, ...	continuous function, sample-sequence, random process,...	analytic, stochastic, sinusoidal, standard deviations
Systems	process, controller, sensors, actuators, ...	differential equations, difference equations, transfer functions, state space models, ...	continuous time, sampled, linear, nonlinear, ...

Principles of Feedback

We have seen that feedback is a key tool that can be used to modify the behaviour of a system.

This behaviour altering effect of feedback is a key mechanism that control engineers exploit deliberately to achieve the objective of acting on a system to ensure that the desired performance specifications are achieved.

Performance specifications

The key performance goals for this problem are:

- *Safety*: Clearly, the mould level must never be in danger of overflowing or emptying as either case would result in molten metal spilling with disastrous consequences.
- *Profitability*: Aspects which contribute to this requirement include:
  - Product quality
  - Maintenance
  - Throughput

Modelling

To make progress on the control system design problem, it is first necessary to gain an understanding of how the process operates. This understanding is typically expressed in the form of a mathematical model.

Definition of the control problem

Abstracting from a particular problem, we can introduce:

Definition

The central problem in control is to find a technically feasible way to act on a given process so that the process behaves, as closely as possible, to some desired behaviour. Furthermore, this approximate behaviour should be achieved in the face of uncertainty of the process and in the presence of uncontrollable external disturbances acting on the process.

From open to closed loop architectures

Unfortunately, the open loop methodology does not lead to a satisfactory solution to the control problem unless:

- the model on which the design of the controller has been based is a very good representation of the plant,
- the model and its inverse are stable, and
- disturbances and initial conditions are negligible.

We are thus motivated to find an alternative solution to the problem which retains the key features but which does not suffer from the above drawbacks.

Figure 2.9: *Open loop controller*

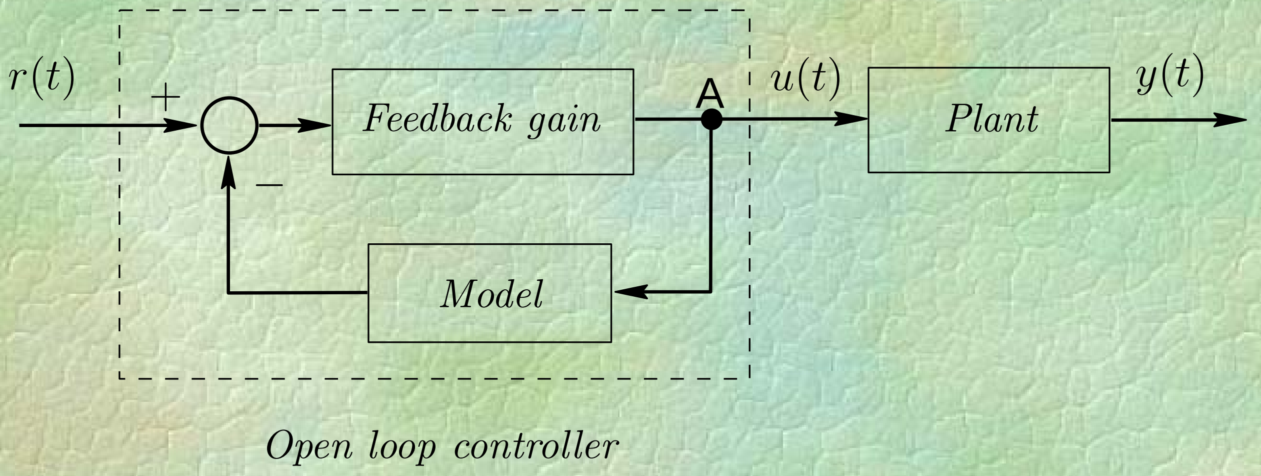
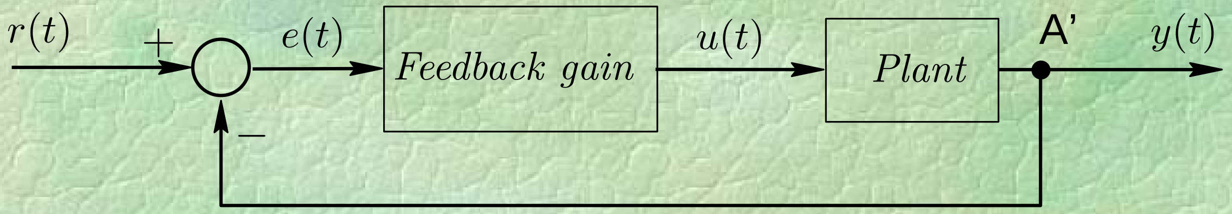


Figure 2.10: *Closed loop control*



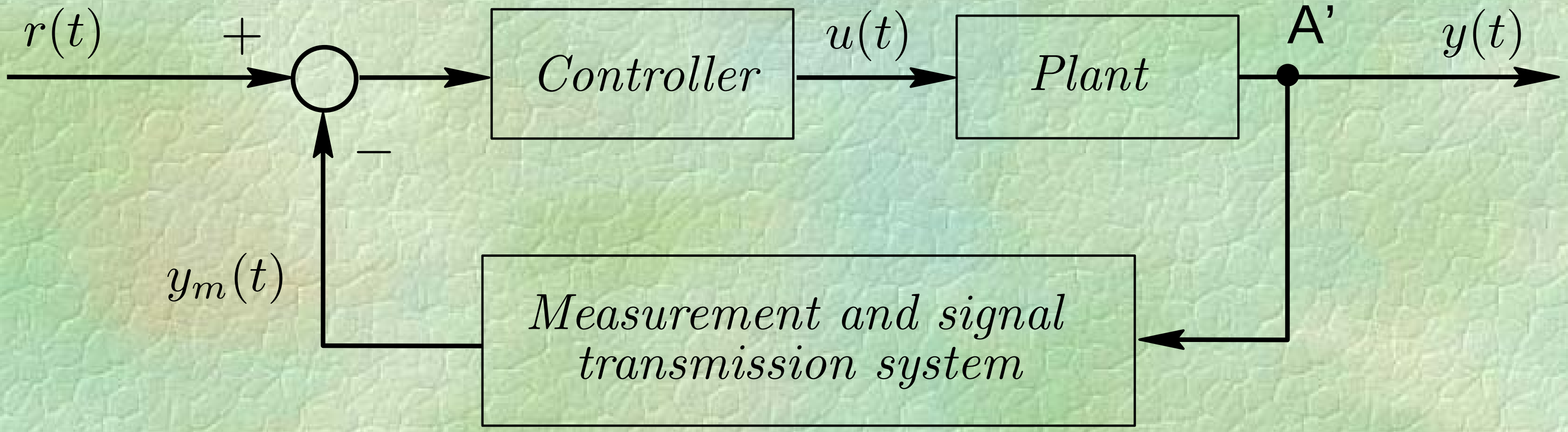
- The first thing to note is that, provided the model represents the plant exactly, and that all signals are bounded (i.e. the loop is stable), then both schemes are equivalent, regarding the relation between $r(t)$ and $y(t)$. The key differences are due to disturbances and different initial conditions.
- In the open loop control scheme the controller incorporates feedback internally, i.e. a signal at point A is fed back.

- In the closed loop scheme, the feedback signal depends on what is actually happening in the plant since the true plant output is used.

We have seen that this modified architecture has many advantages including:

- insensitivity to modelling errors;
- insensitivity to disturbances in the plant (*that are not reflected in the model*).

Figure 2.11: *Closed loop control with sensors*

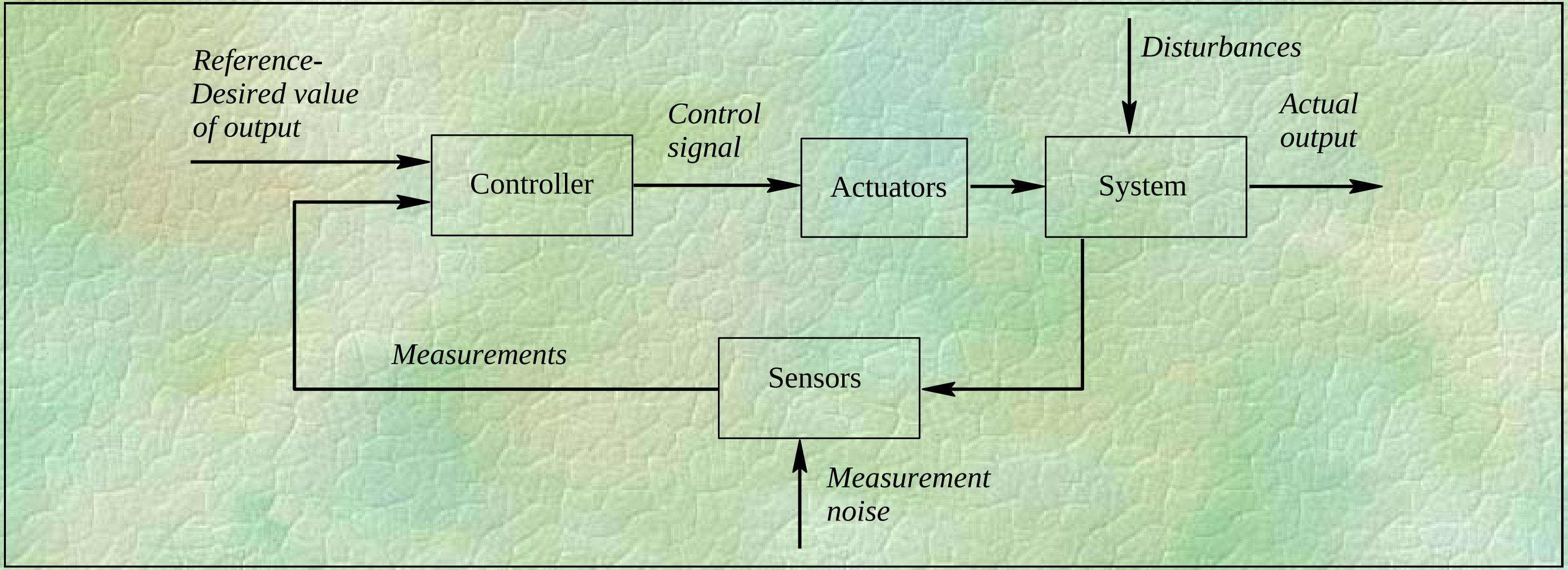


Desirable attributes of sensors

- *Reliability*. It should operate within the necessary range.
- *Accuracy*. For a variable with a constant value, the measurement should settle to the correct value.
- *Responsiveness*. If the variable changes, the measurement should be able to follow the changes. Slow responding measurements can, not only affect the quality of control but can actually make the feedback loop unstable. Loop instability may arise even though the loop has been designed to be stable assuming an exact measurement of the process variable.

Figure 2.12: *Typical feedback loop*

In summary, a typical feedback loop (including sensor issues) is shown below.



Modelling

The power of a mathematical model lies in the fact that it can be simulated in hypothetical situations, be subject to states that would be dangerous in reality, and it can be used as a basis for synthesizing controllers.

Model Complexity

In building a model, it is important to bear in mind that all real processes are complex and hence any attempt to build an exact description of the plant is usually an impossible goal. Fortunately, feedback is usually very forgiving and hence, in the context of control system design, one can usually get away with rather simple models, provided they capture the essential features of the problem.

Building Models

A first possible approach to building a plant model is to postulate a specific model structure and to use what is known as a *black box* approach to modeling. In this approach one varies, either by trial and error or by an algorithm, the model parameters until the dynamic behavior of model and plant match sufficiently well.

An alternative approach for dealing with the modeling problem is to use physical laws (such as conservation of mass, energy and momentum) to construct the model. In this approach one uses the fact that, in any real system, there are *basic phenomenological laws* which determine the relationships between all the signals in the system.

In practice, it is common to combine both black box and phenomenological ideas to building a model.

State Space Models

For continuous time systems

$$\frac{dx}{dt} = f(x(t), u(t), t)$$

$$y(t) = g(x(t), u(t), t)$$

For discrete time systems

$$x[k+1] = f_d(x[k], u[k], k)$$

$$y[k] = g_d(x[k], u[k], k)$$

Linear State Space Models

$$\frac{dx(t)}{dt} = \mathbf{A}x(t) + \mathbf{B}u(t)$$

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

- Models are classified according to properties of the equation they are based on. Examples of classification include:

Model Attribute	*Contrasting Attribute*	*Asserts whether or not …*
Single input Single output	Multiple input multiple output	…the model equations have one input and one output only
Linear	Nonlinear	…the model equations are linear in the system variables
Time varying	Time invariant	…the model parameters are constant
Continuous	Sampled	…model equations describe the behavior at every instant of time, or only in discrete *samples* of time
Input-output	State space	…the model equations rely on functions of input and output variables only, or also include the so called *state variables*.
Lumped parameter	Distributed parameter	…the model equations are ordinary or partial differential equations

- In many situations nonlinear models can be linearised around a user defined operating point.

Continuous Time Signals

Mathematical Topics

Specific topics to be covered include:

- linear high order differential equation models
- Laplace transforms, which convert linear differential equations to algebraic equations, thus greatly simplifying their study
- methods for assessing the stability of linear dynamic systems
- frequency response.

Linear Continuous Time Models

The linear form of this model is:

$$\frac{d^n y(t)}{dt^n} + a_{n-1}\frac{d^{n-1}y(t)}{dt^{n-1}} + \ldots + a_0 y(t) = b_{n-1}\frac{d^{n-1}}{dt^{n-1}}u(t) + \ldots + b_0 u(t)$$

Introducing the Heaviside, or differential, operator ρ⟨∘⟩:

$$\rho\langle f(t)\rangle = \rho f(t) \triangleq \frac{df(t)}{dt}$$

$$\rho^n\langle f(t)\rangle = \rho^n f(t) = \rho\left\langle \rho^{n-1}\langle f(t)\rangle\right\rangle = \frac{df^n(t)}{dt^n}$$

We obtain:

$$\rho^n y(t) + a_{n-1}\rho^{n-1}y(t) + \ldots + a_0 y(t) = b_{n-1}\rho^{n-1}u(t) + \ldots + b_0 u(t)$$

Laplace Transforms

The study of differential equations of the type described above is a rich and interesting subject. Of all the methods available for studying linear differential equations, one particularly useful tool is provided by Laplace Transforms.

Transfer Functions

Taking Laplace Transforms converts the differential equation into the following algebraic equation

$$s^{n}Y(s) + a_{n-1}s^{n-1}Y(s) + ::: + a_{0}Y(s)$$

$$= b_{n-1}s^{n-1}U(s) + ::: + b_{0}U(s) + f(s; x_{o})$$

where $$Y(s) = G(s)U(s)$$

and $$G(s) = \frac{B(s)}{A(s)}$$

$G(s)$ is called the *transfer function*.

$$A(s) = s^{n} + a_{n-1}s^{n-1} + \ldots + a_{0}$$

$$B(s) = b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \ldots + b_{0}$$

Often practical systems have a time delay between input and output. This is usually associated with the transport of material from one point to another. For example, if there is a conveyor belt or pipe connecting different parts of a plant, then this will invariably introduce a delay.

The transfer function of a pure delay is of the form:

$$H(s) = e^{-sT_d}$$

where T_d is the delay (in seconds). T_d will typically vary depending on the transportation speed.

Systems with Delay

The transfer function from input to the output is approximately of the form:

$$H(s) = \frac{Ke^{-sT_d}}{(\tau s + 1)}$$

Summary

Transfer functions describe the input-output properties of linear systems in algebraic form.

Stability of Transfer Functions

We say that a system is stable if any bounded input produces a bounded output for all bounded initial conditions. In particular, we can use a partial fraction expansion to decompose the total response of a system into the response of each pole taken separately. For continuous-time systems, we then see that stability requires that the poles have strictly negative real parts, i.e., they need to be in the open left half plane (OLHP) of the complex plane s . This implies that, for continuous time systems, the stability boundary is the imaginary axis.

Steady State Step Response

The steady state response (provided it exists) for a unit step is given by

$$\lim_{t\to\infty} y(t) = y_\infty = \lim_{s\to 0} s\, G(s)\frac{1}{s} = G(0)$$

where $G(s)$ is the transfer function of the system.

We define the following indicators:

Steady state value, y_∞: the final value of the step response (this is meaningless if the system has poles in the CRHP).

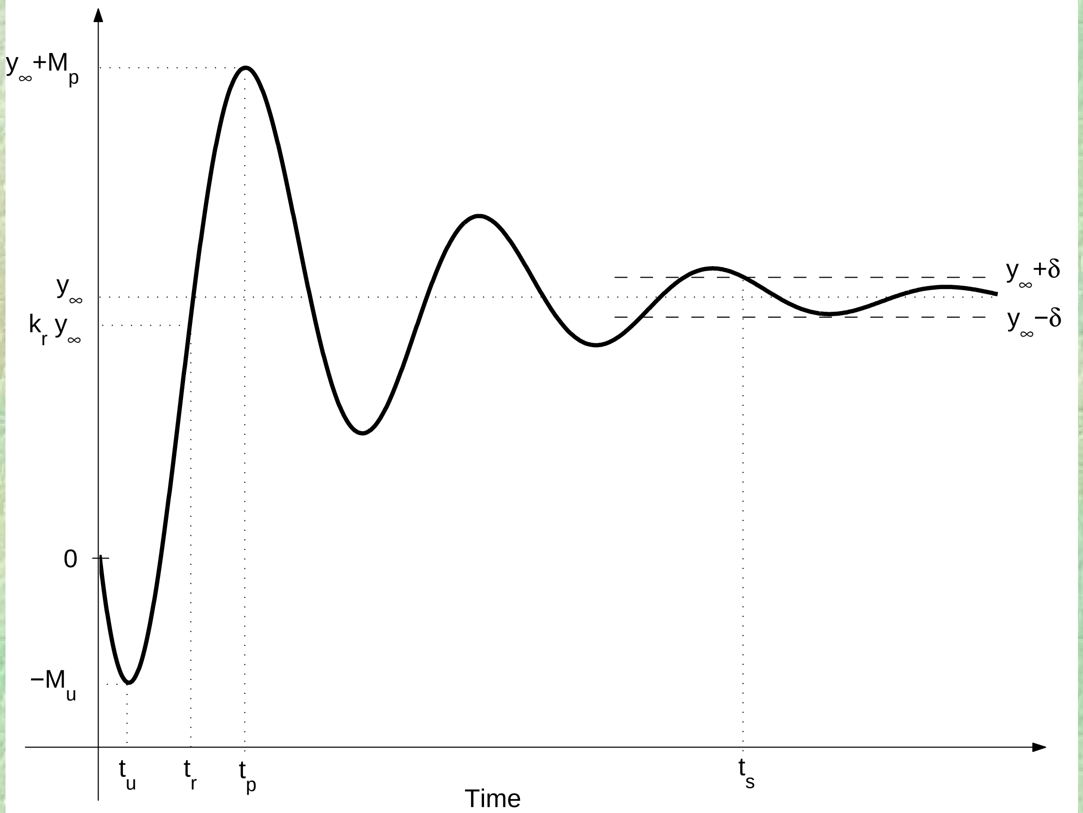
Rise time, t_r: The time elapsed up to the instant at which the step response reaches, for the first time, the value $k_r y_\infty$. The constant k_r varies from author to author, being usually either 0.9 or 1.

Overshoot, M_p: The maximum instantaneous amount by which the step response exceeds its final value. It is usually expressed as a percentage of $\boldsymbol{y_\infty}$

Undershoot, M_u: the (absolute value of the) maximum instantaneous amount by which the step response falls below zero.

Settling time, t_s: the time elapsed until the step response enters (without leaving it afterwards) a specified deviation band, ±δ, around the final value. This deviation δ, is usually defined as a percentage of y_∞, say 2% to 5%.

Figure 4.3: *Step response indicators*



Poles, Zeros and Time Responses

We will consider a general transfer function of the form

$$H(s) = K\frac{\prod_{i=1}^{m}(s-\beta_i)}{\prod_{l=1}^{n}(s-\alpha_l)}$$

β_1, β_1,…, β_m and α_1, α_2, ,,, α_n are the zeros and poles of the transfer function, respectively. The relative degree is $n_r \overset{\Delta}{=} n - m$.

Poles

Recall that any scalar rational transfer function can be expanded into a partial fraction expansion, each term of which contains either a single real pole, a complex conjugate pair or multiple combinations with repeated poles.

First Order Pole

A general first order pole contributes

$$H_1(s) = \frac{K}{\tau s + 1}$$

The response of this system to a unit step can be computed as

$$y(t) = \mathcal{L}^{-1}\left[\frac{K}{s(\tau s + 1)}\right] = \mathcal{L}^{-1}\left[\frac{K}{s} - \frac{K\tau}{\tau s + 1}\right] = K(1 - e^{-\frac{t}{\tau}})$$

A Complex Conjugate Pair

For the case of a pair of complex conjugate poles, it is customary to study a *canonical second order system* having the transfer function.

$$H(s) = \frac{\omega_n^2}{s^2 + 2\psi\omega_n s + \omega_n^2}$$

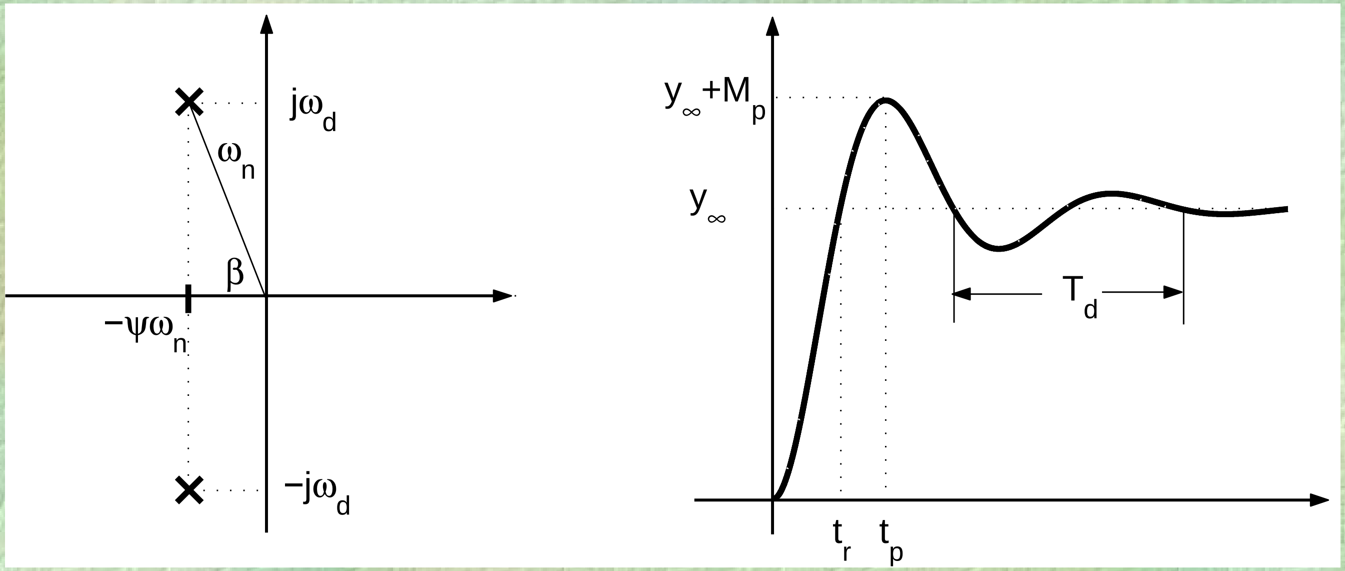
Step Response for Canonical Second Order Transfer Function

$$\begin{aligned} Y(s) &= \frac{1}{s} - \frac{s+\psi\omega_n}{(s+\psi\omega_n)^2+\omega_d^2} - \frac{\psi\omega_n}{(s+\psi\omega_n)^2+\omega_d^2} \\ &= \frac{1}{s} - \frac{1}{\sqrt{1-\psi^2}}\left[\sqrt{1-\psi^2}\frac{s+\psi\omega_n}{(s+\psi\omega_n)^2+\omega_d^2} - \psi\frac{\omega_d}{(s+\psi\omega_n)^2+\omega_d^2}\right] \end{aligned}$$

On applying the inverse Laplace transform we finally obtain

$$y(t) = 1 - \frac{e^{-\psi\omega_n t}}{\sqrt{1-\psi^2}}\sin(\omega_d t + \beta)$$

Figure 4.5: *Pole location and unit step response of a canonical second order system.*



Zeros

The effect that zeros have on the response of a transfer function is a little more subtle than that due to poles. One reason for this is that whilst poles are associated with the states in isolation, zeros rise from additive interactions amongst the states associated with different poles. Moreover, the zeros of a transfer function depend on where the input is applied and how the output is formed as a function of the states.

Frequency Response

We next study the system response to a rather special input, namely a sine wave. The reason for doing so is that the response to sine waves also contains rich information about the response to other signals.

Let the transfer function be

$$H(s) = K \frac{\sum_{i=0}^{m} b_i s^i}{s^n + \sum_{k=1}^{n-1} a_k s^k}$$

Then the steady state response to the input sin(*wt*) is

$$y(t) = |H(jw)|\sin(wt + \phi(w))$$

where

$$H(j\omega) = |H(j\omega)|e^{j\phi(\omega)}$$

In summary:

A sine wave input forces a sine wave at the output with the same frequency. Moreover, the amplitude of the output sine wave is modified by a factor equal to the magnitude of H(jw) and the phase is shifted by a quantity equal to the phase of H(jw).

Bode Diagrams

Bode diagrams consist of a pair of plots. One of these plots depicts the magnitude of the frequency response as a function of the angular frequency, and the other depicts the angle of the frequency response, also as a function of the angular frequency.

Summary

- There are two key approaches to linear dynamic models:
  - the, so-called, time domain, and
  - the so-called, frequency domain
- Although these two approaches are largely equivalent, they each have their own particular advantages and it is therefore important to have a good grasp of each.

- With respect to the important characteristic of stability, a continuous time system is
  - stable if and only if the real parts of all poles are strictly negative
  - marginally stable if at least one pole is strictly imaginary and no pole has strictly positive real part
  - unstable if the real part of at least one pole is strictly positive
  - non-minimum phase if the real part of at least one zero is strictly positive.

Analysis of SISO Control Loops

Topics to be covered

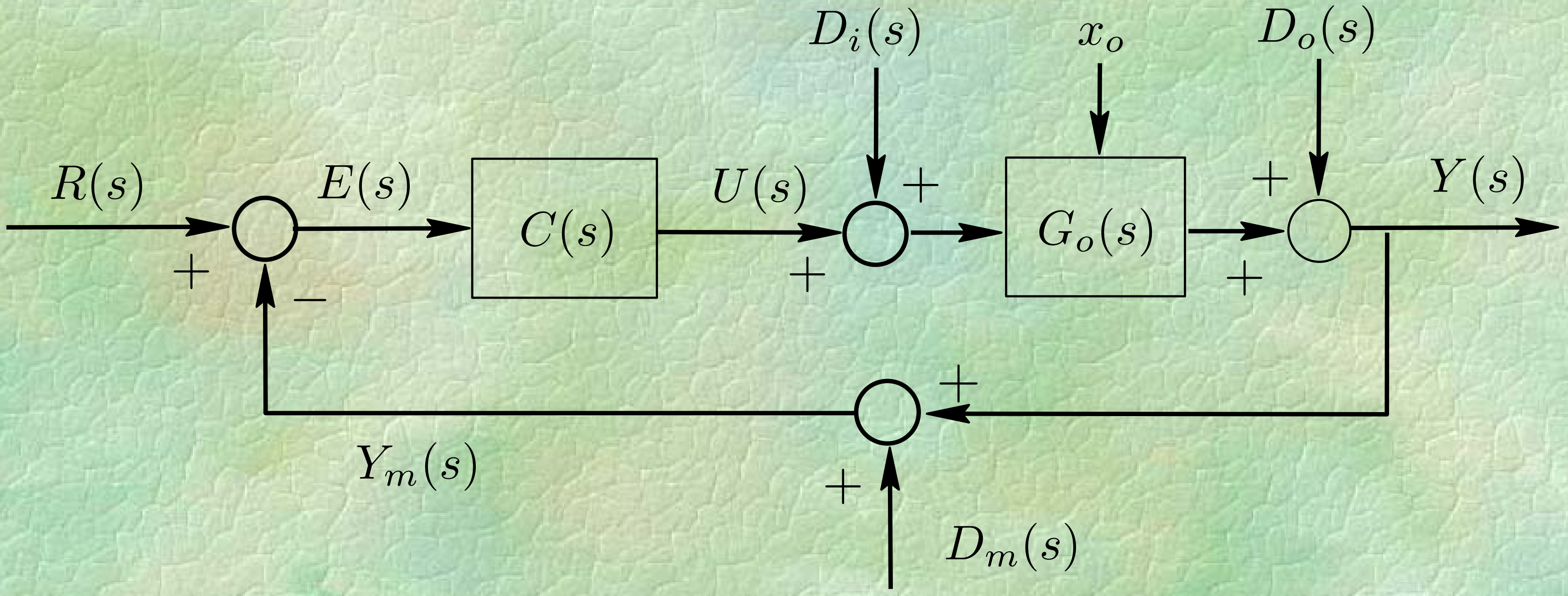
For a given controller and plant connected in feedback we ask and answer the following questions:

- Is the loop stable?
- What are the sensitivities to various disturbances?
- What is the impact of linear modeling errors?
- How do small nonlinearities impact on the loop?

We recall several analysis tools; specifically

- Root locus

Figure 5.1: *Simple feedback control system*



In the loop shown in Figure 5.1 we use transfer functions and Laplace transforms to describe the relationships between signals in the loop. In particular, $C(s)$ and $G_0(s)$ denote the transfer functions of the controller and the nominal plant model respectively, which can be represented in fractional form as:

$$C(s) = \frac{P(s)}{L(s)}$$

$$G_o(s) = \frac{B_o(s)}{A_o(s)}$$

Link to Characteristic Equation

Lemma

Consider the nominal closed loop depicted in Figure 5.1. Then the nominal closed loop is internally stable if and only if the roots of the nominal closed loop characteristic equation

$$A_o(s)L(s) + B_o(s)P(s) = 0$$

all lie in the open left half plane. We call $A_0L + B_0P$ the nominal closed-loop characteristic polynomial.

Stability and Polynomial Analysis

Consider a polynomial of the following form:

$$p(s) = s^n + a_{n-1}s^{n-1} + \ldots + a_1 s + a_0$$

The problem to be studied deals with the question of whether that polynomial has any root with nonnegative real part. Obviously, this equation can be answered by *computing the n roots* of $p(s)$. However, in many applications it is of special interest to study the interplay between the location of the roots and certain polynomial coefficients.

Root Locus (RL)

A classical tool used to study stability of equations of the type given above is root locus. The root locus approach can be used to examine the location of the roots of the characteristic polynomial as one parameter is varied.

Consider the following equation

$$1 + \lambda F(s) = 0 \qquad \text{where} \qquad F(s) = \frac{M(s)}{D(s)}$$

with $\lambda \geq 0$ and *M*, *N* have degree *m*, *n* respectively.

Nominal Stability using Frequency Response

A classical and lasting tool that can be used to assess the stability of a feedback loop is Nyquist stability theory. In this approach, stability of the closed loop is predicted using the open loop frequency response of the system. This is achieved by plotting a polar diagram of the product $G_0(s)C(s)$ and then counting the number of encirclements of the (-1,0) point. We show how this works below.

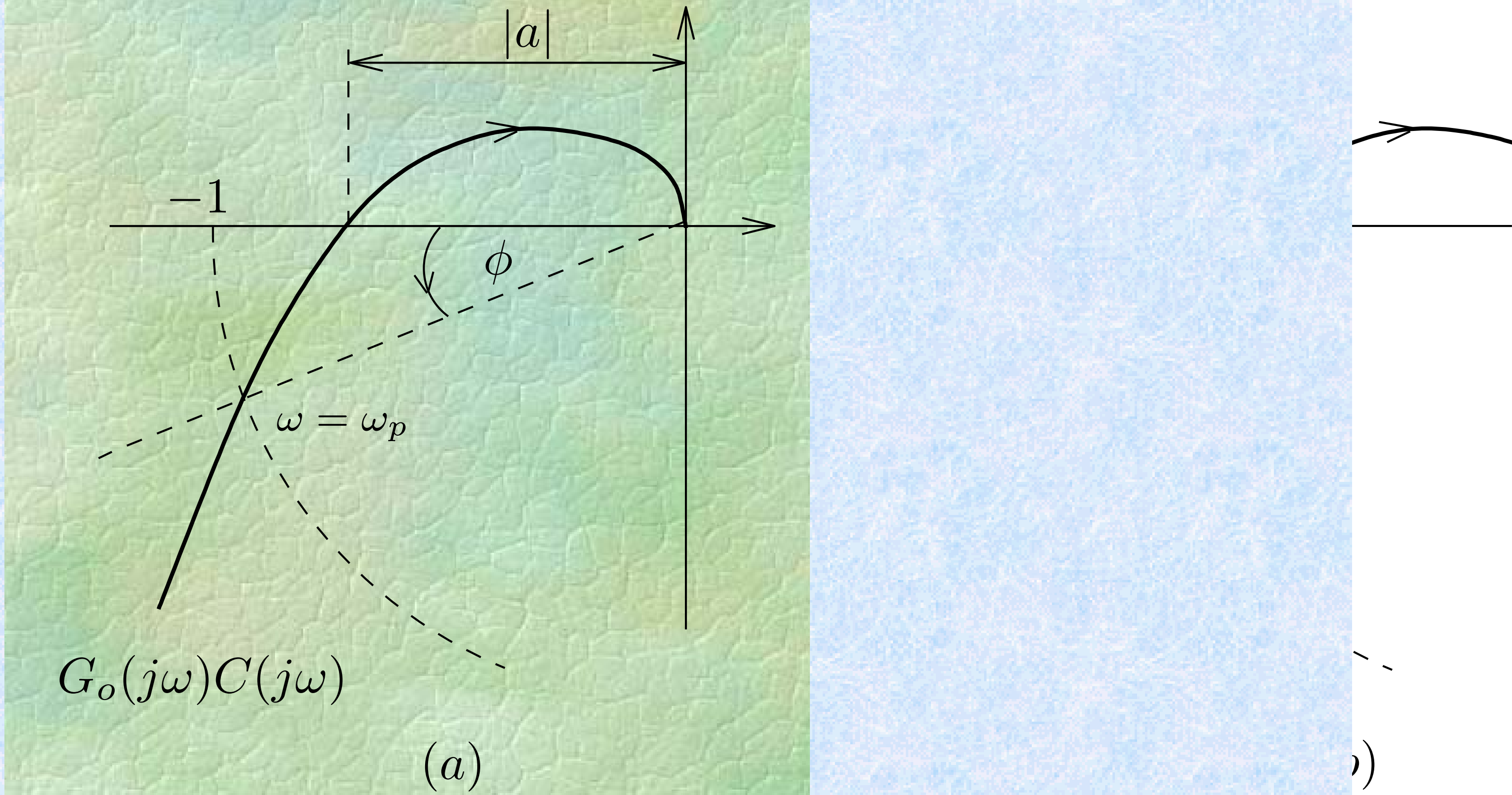
Discussion

- If the system is open loop stable, then for the closed loop to be internally stable it is necessary and sufficient that no unstable cancellations occur and that the Nyquist plot of $G_0(s)C(s)$ *does not encircle the point* (-1,0).

Relative Stability: Stability margins

In control system design, one often needs to go beyond the issue of closed loop stability. In particular, it is usually desirable to obtain some quantitative measures of how far from instability the nominal loop is, i.e. to quantify relative stability. This is achieved by introducing measures which describe the distance from the nominal open loop frequency response to the critical stability point (-1,0).

Figure 5.7: *Stability margins*



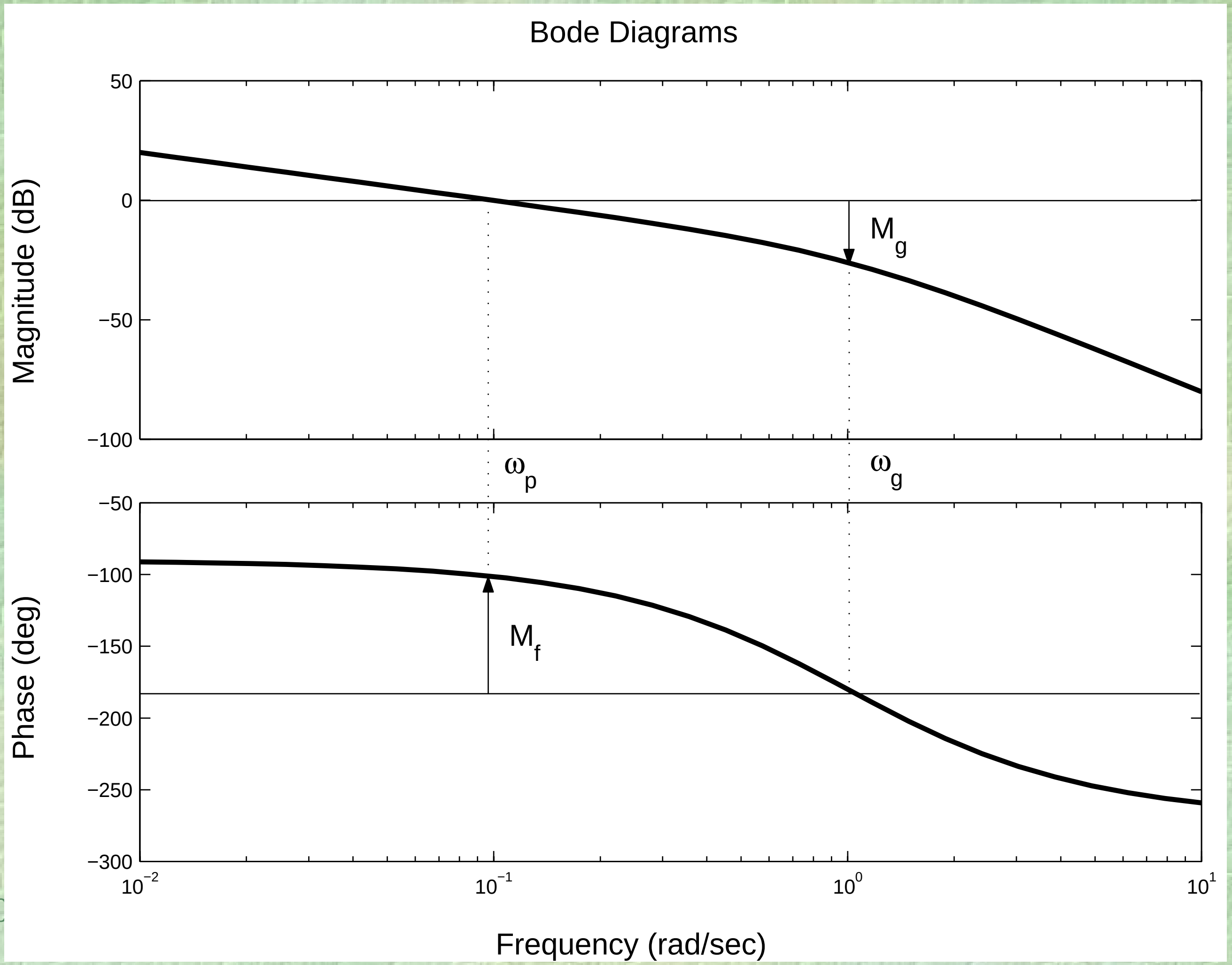
Gain and Phase Margins

- The gain margin, M_g, and the phase margin M_f are defined as follows (see Figure 5.7):

$$M_g \triangleq -20\log_{10}(|a|)$$

$$M_f \triangleq \phi$$

Figure 5.8: *Stability margins in Bode diagrams*



- Well designed, feedback can
  - make an unstable system stable;
  - increase the response speed;
  - decrease the effects of disturbances;
  - decrease the effects of system parameter uncertainties, and more.

Classical PID Control

This lecture examines a particular control structure that has become almost universally used in industrial control. It is based on a particular fixed structure controller family, the so-called PID controller family. These controllers have proven to be robust and extremely beneficial in the control of many important applications.

PID stands for: **P** (*Proportional*)

I (*Integral*)

D (*Derivative*)

The Current Situation

*Despite the abundance of sophisticated tools, including advanced controllers, the Proportional, Integral, Derivative (PID controller) is still the most widely used in modern industry, controlling more that 95% of closed-loop industrial processes**

* Åström K.J. & Hägglund T.H. 1995, "New tuning methods for PID controllers", *Proc. 3rd European Control Conference*, p.2456-62; and

*Yamamoto & Hashimoto 1991, "Present status and future needs: The view from Japanese industry", Chemical Process Control, *CPCIV, Proc. 4th Inter-national Conference on Chemical Process Control*, Texas, p.1-28.

PID Structure

Consider the simple SISO control loop shown in Figure 6.1:

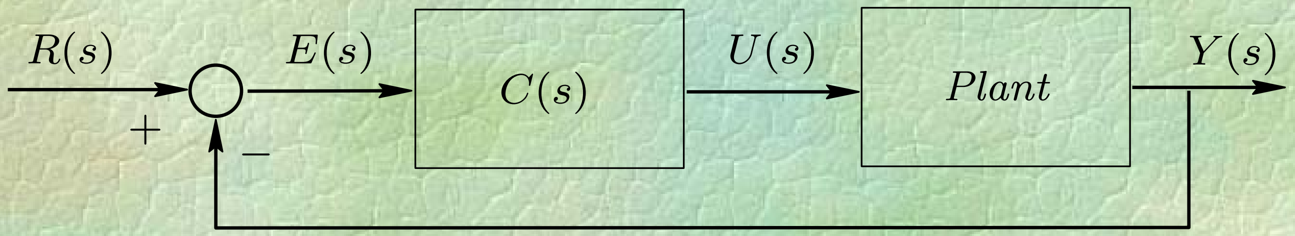


Figure 6.1: *Basic feedback control loop*

The *standard form* PID are:

Proportional only: $C_P(s) = K_p$

Proportional plus Integral: $C_{PI}(s) = K_p\left(1 + \frac{1}{T_r s}\right)$

Proportional plus derivative: $C_{PD}(s) = K_p\left(1 + \frac{T_d s}{\tau_D s + 1}\right)$

Proportional, integral and derivative: $C_{PID}(s) = K_p\left(1 + \frac{1}{T_r s} + \frac{T_d s}{\tau_D s + 1}\right)$

Tuning of PID Controllers

Because of their widespread use in practice, we present below several methods for tuning PID controllers. *Actually these methods are quite old and date back to the 1950's*. Nonetheless, they remain in widespread use today.

In particular, we will study.

- *Ziegler-Nichols Oscillation Method*

Ziegler-Nichols (Z-N) Oscillation Method

This procedure is only valid for open loop stable plants and it is carried out through the following steps

- Set the true plant under proportional control, with a very small gain.
- Increase the gain until the loop starts oscillating. Note that linear oscillation is required and that it should be detected at the controller output.

- Record the controller critical gain $K_p = K_c$ and the oscillation period of the controller output, P_c.
- Adjust the controller parameters according to Table 6.1 (*next slide*); there is some controversy regarding the PID parameterization for which the Z-N method was developed, but the version described here is, to the best knowledge of the authors, applicable to the parameterization of standard form PID.

Table 6.1: *Ziegler-Nichols tuning using the oscillation method*

	$\mathbf{K_p}$	$\mathbf{T_r}$	$\mathbf{T_d}$
P	$0.50K_c$		
PI	$0.45K_c$	$\frac{P_c}{1.2}$	
PID	$0.60K_c$	$0.5P_c$	$\frac{P_c}{8}$

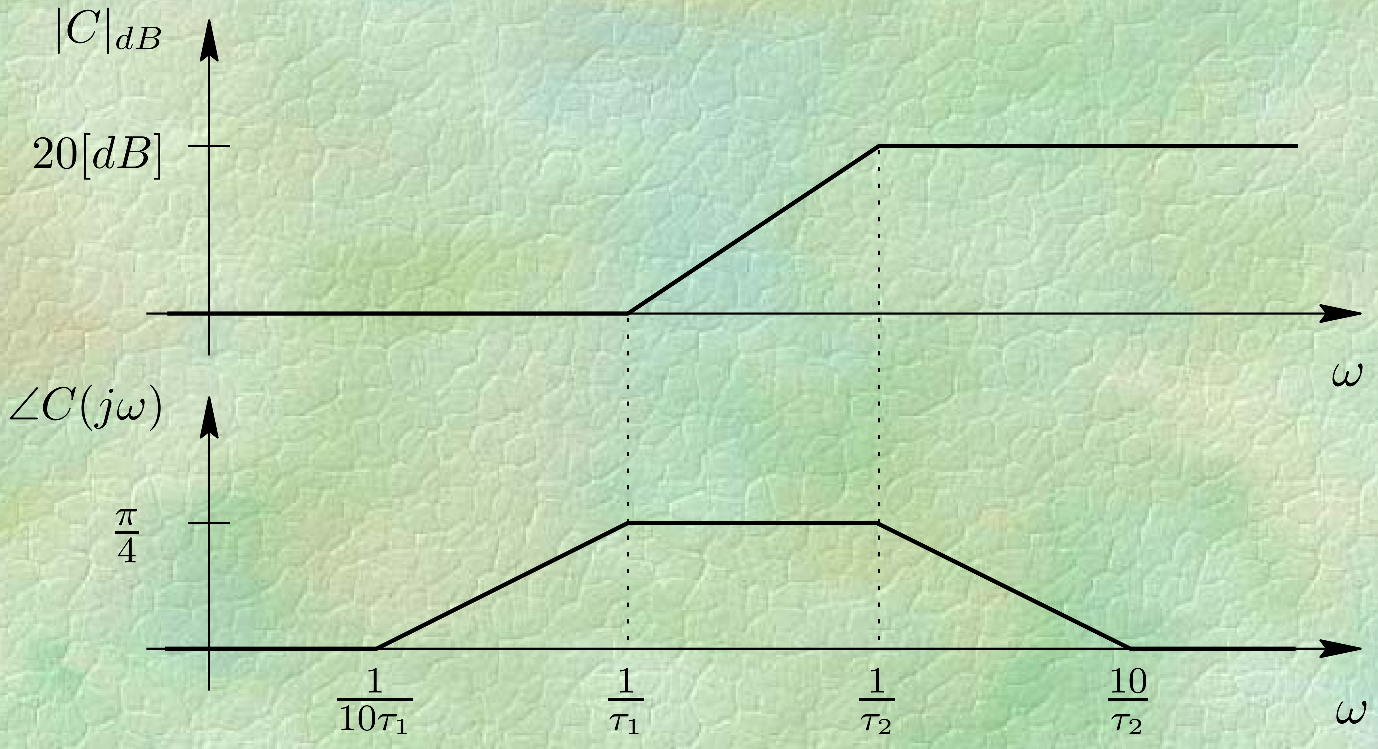
Lead-lag Compensators

Closely related to PID control is the idea of lead-lag compensation. The transfer function of these compensators is of the form:

$$C(s) = \frac{\tau_1 s + 1}{\tau_2 s + 1}$$

If $\tau_1 > \tau_2$, then this is a *lead network* and when $\tau_1 < \tau_2$, this is a *lag network*.

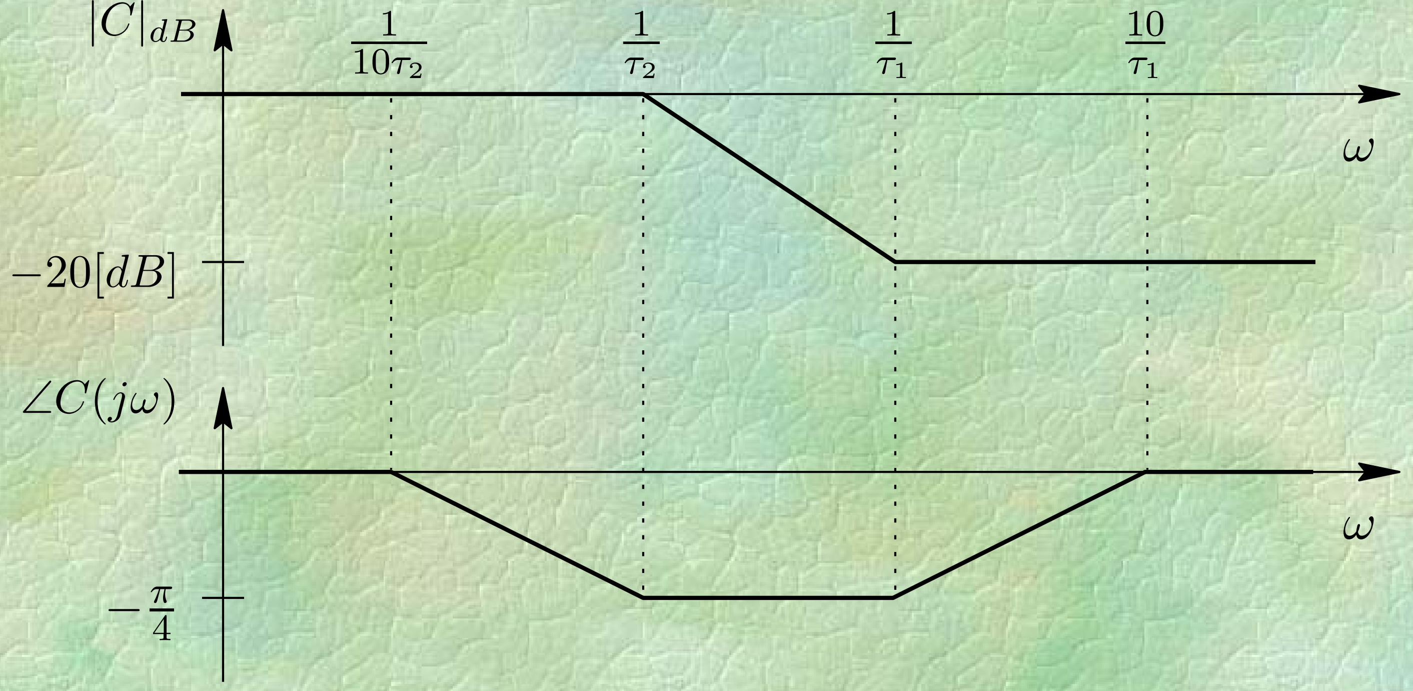
Figure 6.9: *Approximate Bode diagrams for* ***lead networks*** *(τ_1=10τ_2)*



Observation

We see from the previous slide that the lead network gives phase advance at $\omega = 1/\tau_1$ without an increase in gain. Thus it plays a role similar to derivative action in PID.

Figure 6.10: *Approximate Bode diagrams for* ***lag networks*** *(τ_2=10τ_1)*



Observation

We see from the previous slide that the lag network gives low frequency gain increase. Thus it plays a role similar to integral action in PID.

Summary

- PI and PID controllers are widely used in industrial control.

- From a modern perspective, a PID controller is simply a controller of (up to second order) containing an integrator. Historically, however, PID controllers were tuned in terms of their **P**, **I** and **D** terms.

- It has been empirically found that the PID structure often has sufficient flexibility to yield excellent results in many applications.

- The basic term is the proportional term, **P**, which causes a corrective control actuation proportional to the error.
- The integral term, **I** gives a correction proportional to the integral of the error. This has the positive feature of ultimately ensuring that sufficient control effort is applied to reduce the tracking error to zero. However, integral action tends to have a destabilizing effect due to the increased phase shift.

- The derivative term, **D**, gives a predictive capability yielding a control action proportional to the rate of change of the error. This tends to have a stabilizing effect but often leads to large control movements.
- Various empirical tuning methods can be used to determine the PID parameters for a given application. They should be considered as a first guess in a search procedure.

Synthesis of SISO Controllers

Pole Assignment

In the previous chapter, we examined PID control. However, the tuning methods we used were essentially ad-hoc. Here we begin to look at more formal methods for control system design. In particular, we examine the following key synthesis question:

> *Given a model, can one systematically synthesize a controller such that the closed loop poles are in predefined locations*?

This lecture will show that this is indeed possible. We call this *pole assignment*, which is a fundamental idea in control synthesis.

Polynomial Approach

In the nominal control loop, let the controller and nominal model transfer functions be respectively given by:

$$C(s) = \frac{P(s)}{L(s)} \qquad G_o(s) = \frac{B_o(s)}{A_o(s)}$$

with

$$P(s) = p_{n_p}s^{n_p} + p_{n_p-1}s^{n_p-1} + \ldots + p_0$$

$$L(s) = l_{n_l}s^{n_l} + l_{n_l-1}s^{n_l-1} + \ldots + l_0$$

$$B_o(s) = b_{n-1}s^{n-1} + b_{n-2}s^{n-2} + \ldots + b_0$$

$$A_o(s) = a_n s^n + a_{n-1}s^{n-1} + \ldots + a_0$$

Consider now a desired closed loop polynomial given by

$$A_{cl}(s) = a^c_{n_c} s^{n_c} + a^c_{n_c-1} s^{n_c-1} + \ldots + a^c_0$$

Goal

Our objective here will be to see if, for given values of B_0 and A_0, P and L can be designed so that the closed loop characteristic polynomial is $A_{cl}(s)$.

We will see that, under quite general conditions, this is indeed possible.

PI and PID Synthesis Revisited using Pole Assignment

The reader will recall that PI and PID controller synthesis using classical methods were reviewed in Lecture 6.

During laboratory sessions we place these results in a more modern setting by discussing the synthesis of lead/lag networks and PID controllers based on pole assignment techniques (via root locus analysis).

- The key synthesis question is:
  Given a model, can one synthesize a controller such that the closed loop poles (i.e. sensitivity poles) are in pre-defined locations.

- Stated mathematically:
  Given polynomials $A_0(s)$, $B_0(s)$ (defining the model) and given a polynomial $A_{cl}(s)$ (defining the desired location of closed loop poles), is it possible to find polynomials $P(s)$ and $L(s)$ such that $A_0(s)L(s) + B_0(s)P(s) = A_{cl}(s)$? This lecture shows that this is indeed possible.

After Continuous-Time Control…

… Models for Sampled Data Systems

Motivation

Up to this point, we have assumed that the control systems we have studied operate in continuous time and that the control law is implemented in analogue fashion. Certainly in the early days of control, all control systems were implemented via some form of analogue equipment. Typically controllers were implemented using one of the following formats:

- hydraulic
- pneumatic
- analogue electronic

However, in recent times, almost all analogue controllers have been replaced by some form of computer control.

This is a very natural move since control can be conceived as the process of making computations based on past observations of a system's behavior so as to decide how one should change the manipulated variables to cause the system to respond in a desirable fashion.

The most natural way to make these computations is via some form of computer.

A huge array of control orientated computers are available in the market place.

A typical configuration includes:

- some form of central processing unit (*to make the necessary computations*)

- analogue to digital converters (*to read the analogue process signals into the computer*).

  (We call this the process of **SAMPLING**)

- digital to analogue converters (*to take the desired control signals out of the computer and present them in a form whereby they can be applied back onto the physical process*).

  (We call this the process of **SIGNAL RECONSTRUCTION**)

Why Study Digital Control?

A simple (*engineering*) approach to digital control is to sample quickly and then to make some reasonable approximation to the derivatives of the digital data. For example, we could approximate the derivative of an analogue signal, $y(t)$, as follows:

$$\frac{d}{dt}y(t) \approx \frac{y(t)-y(t-\Delta)}{\Delta}$$

where Δ is the sampling period.

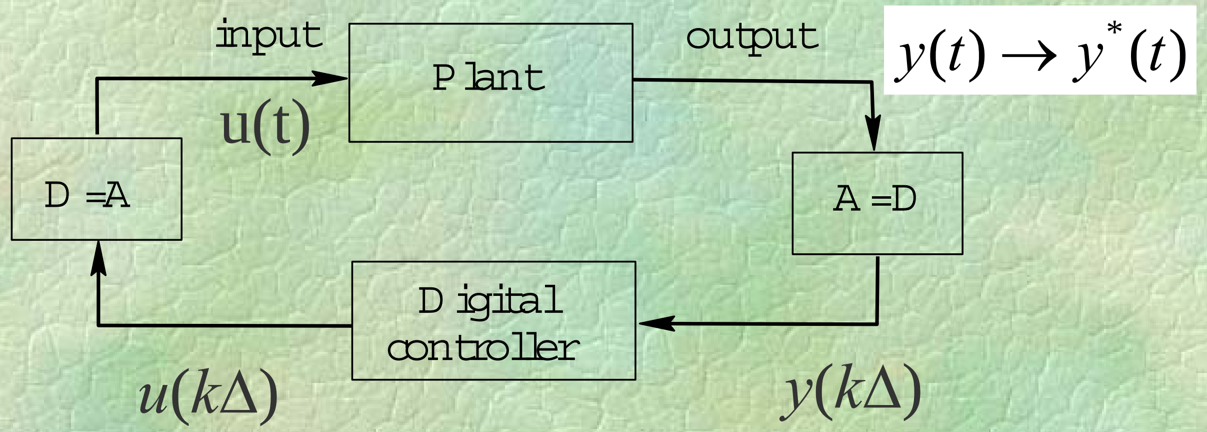
The remainder of the design might then proceed exactly as for continuous time signals and systems using the continuous model.

Actually, the above strategy turns out to be quite good and it is certainly very commonly used in practice.

However, there are some unexpected traps for the unwary. These traps have lead to negative experiences for people naively trying to do digital control by simply mimicking analogue methods. Thus it is important to know when such simple strategies make sense and what can go wrong. We will illustrate by a simple example below.

General Digital Control Scheme

The set-up for digital control of this system is shown schematically below:



The objective is to cause the output $y(t)$, to follow a given reference signal, $y^*(t)$.

Modelling

Since the control computations will be done inside the computer, it seems reasonable to first find a model relating the sampled output

$\{y(k\Delta); k = 0, 1, \dots \}$

to the sampled input signals generated by the computer, which we denote by

$\{u(k\Delta), k = 0, 1, \dots \}$.

Here Δ is the sample period.

Modelling Issues

This lecture is principally concerned with **modelling issues**, i.e. how to relate samples of the output of a physical system to the sampled data input.

Specific topics to be covered are:

- **Discrete-time signals**
- **Z-transforms** and Delta transforms
- **Sampling and reconstruction**
- Aliasing and anti-aliasing filters
- **Sampled-data control systems**

Sampling

The result of sampling a continuous time signal is shown below:

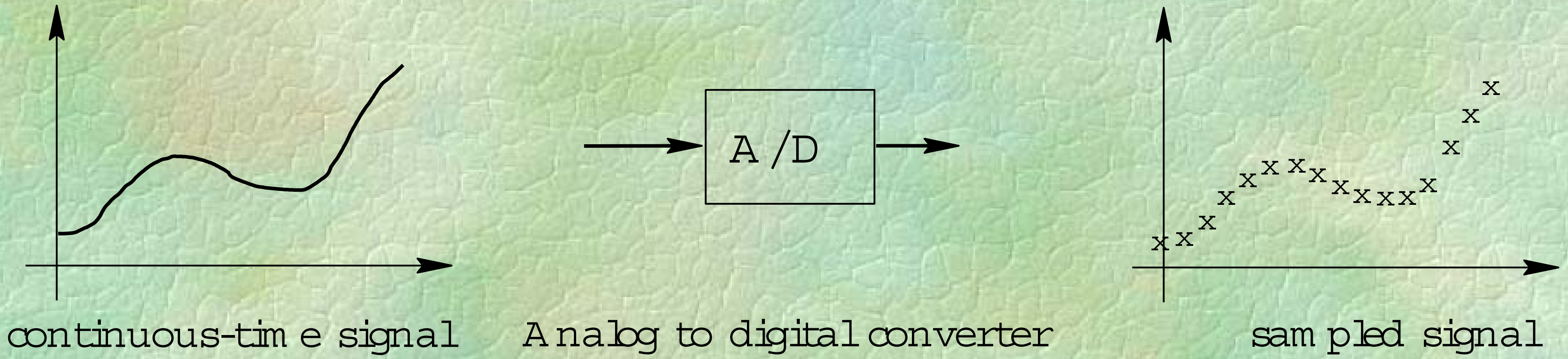


Figure 12.10: *The result of sampling*

There will always be loss of information due to sampling. However, the extent of this loss depends on the sampling method and the associated parameters. For example, assume that a sequence of samples is taken of a signal $f(t)$ every Δ seconds, then the sampling frequency needs to be large enough in comparison with the maximum rate of change of $f(t)$. Otherwise, high frequency components will be mistakenly interpreted as low frequencies in the samples sequence.

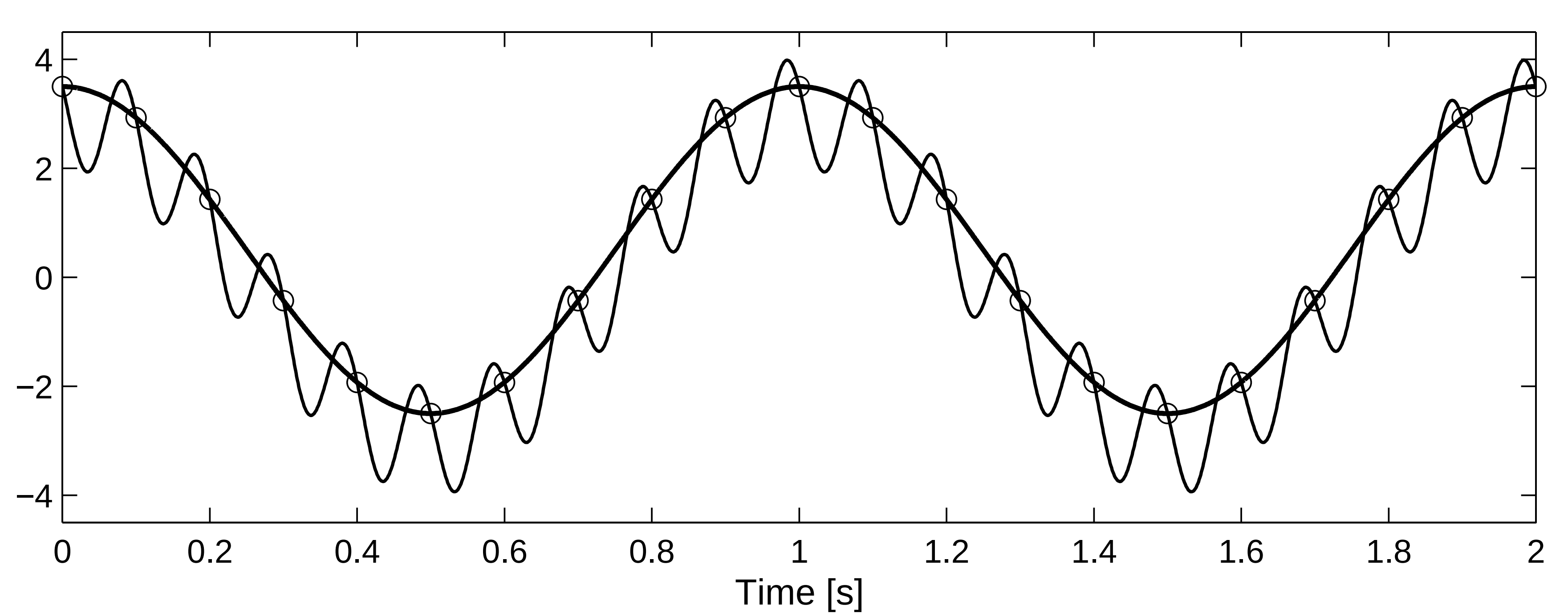


Figure 12.1: *Aliasing effect when using low sampling rate*

Signal Reconstruction

The output of a digital controller is another sequence of numbers $\{u[k]\}$ which are the sample values of the intended control signal. These sample values need to be converted back to continuous time functions before they can be applied to the plant. Usually, this is done by interpolating them into a staircase function $u(t)$ as illustrated in Figure 12.2.

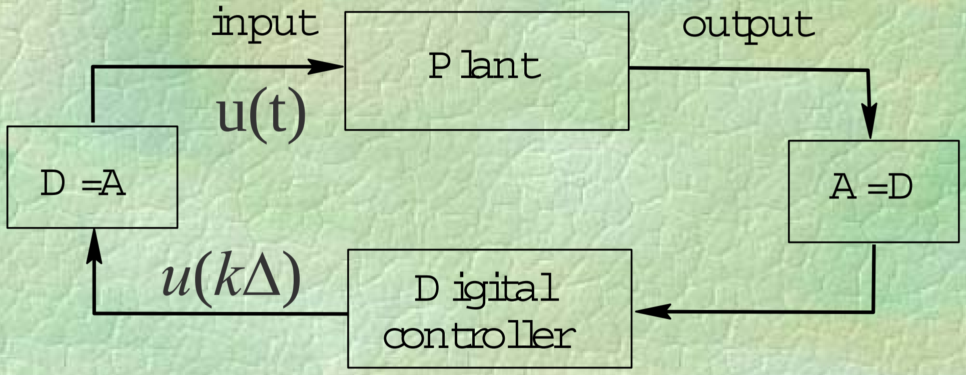


Illustration of Signal Reconstruction

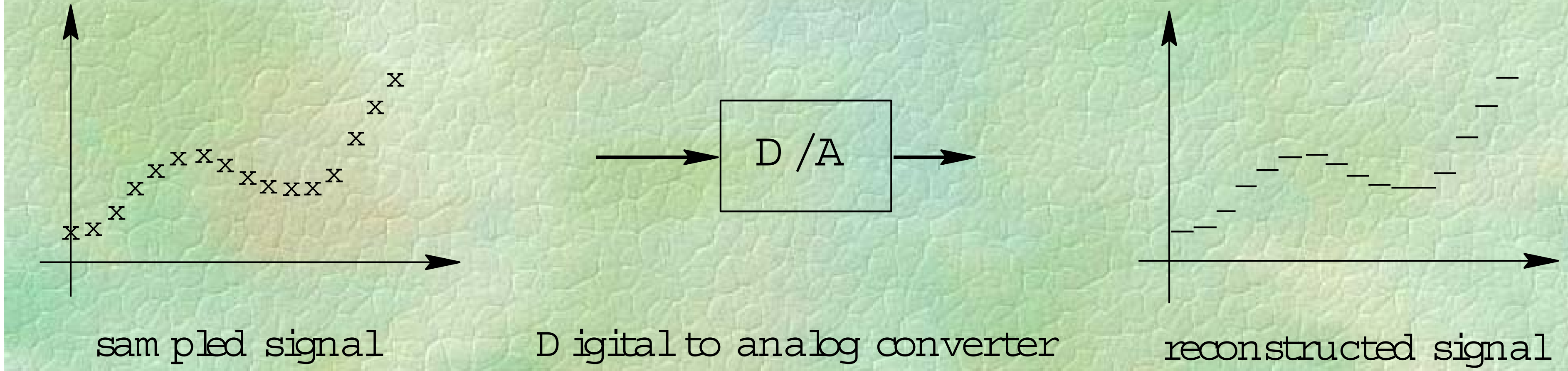


Figure 12.2: *The result of reconstruction*

Linear Discrete Time Models

A useful discrete time model of the type referred to above is the linear version of the high order difference equation model. In the discrete case, this model takes the form:

$$y[k+n] + \overline{a}_{n-1} y[k+n-1] + \cdots + \overline{a}_0 y[k]$$
$$= \overline{b}_{n-1} u[k+n-1] + \cdots + \overline{b}_0 u[k]$$

Note that we saw a special form of this model in relation to the example presented earlier.

The Shift Operator

Forward shift operator

$$\boxed{q(f[k]) \triangleq f[k+1]}$$

In terms of this operator, the model given earlier becomes:

$$q^n y[k] + \overline{a}_{n-1} q^{n-1} y[k] + \cdots + \overline{a}_0 y[k] = \overline{b}_m q^m u[k] + \cdots + \overline{b}_0 u[k]$$

For a discrete time system it is also possible to have discrete state space models. In the shift domain these models take the form:

$$\begin{aligned} qx[k] &= \mathbf{A}_q x[k] + \mathbf{B}_q u[k] \\ y[k] &= \mathbf{C}_q x[k] + \mathbf{D}_q u[k] \end{aligned}$$

Z-Transform

Analogously to the use of Laplace Transforms for continuous time signals, we introduce the Z-transform for discrete time signals.

Consider a sequence $\{y[k]; k = 0, 1, 2, \ldots\}$. Then the Z-transform pair associated with $\{y[k]\}$ is given by

$$\mathcal{Z}\left[y[k]\right] = Y(z) = \sum_{k=0}^{\infty} z^{-k} y[k]$$

$$\mathcal{Z}^{-1}\left[Y(z)\right] = y[k] = \frac{1}{2\pi j} \oint z^{k-1} Y(z) dz$$

How do we use Z-transforms ?

We saw earlier that Laplace Transforms have a remarkable property that they convert differential equations into algebraic equations.

Z-transforms have a similar property for discrete time models, namely they convert difference equations (expressed in terms of the shift operator q) into algebraic equations.

Discrete Transfer Functions

Taking Z-transforms on each side of the high order difference equation model leads to

$$A_q(z)Y_q(z) = B_q(z)U_q(z) + f_q(z, x_o)$$

where $Y_q(z)$, $U_q(z)$ are the Z-transform of the sequences $\{y[k]\}$ and $\{u[k]\}$ respectively, and

$$A_q(z) = z^n + a_{n-1}z^{n-1} + \cdots + a_o$$
$$B_q(z) = b_m z^m + b_{m-1}z^{m-1} + \cdots + b_o$$

We then see that (*ignoring the initial conditions*) the Z-transform of the output $Y(z)$ is related to the Z-transform of the input by $Y(z) = G_q(z)U(\mathrm{z})$ where

$$G_q(z) \triangleq \frac{B_q(z)}{A_q(z)}$$

$G_q(z)$ is called the ***discrete (shift form)*** **transfer function.**

Discrete Time Models

We next examine several properties of discrete time models, beginning with the issue of stability.

Discrete System Stability

Relationship to Poles

We have seen that the response of a discrete system (in the shift operator) to an input $U(z)$ has the form

$$Y(z) = G_q(z)U(z) + \frac{f_q(z;x_o)}{(z-\alpha_1)(z-\alpha_2)\cdots(z-\alpha_n)}$$

where $\alpha_1 \ldots \alpha_n$ are the poles of the system.

We then know, via a partial fraction expansion, that $Y(z)$ can be written as

$$Y(z) = \sum_{j=1}^{n} \frac{\beta_j z}{z-\alpha_j} + \text{terms depending on } U(z)$$

where, for simplicity, we have assumed non repeated poles.

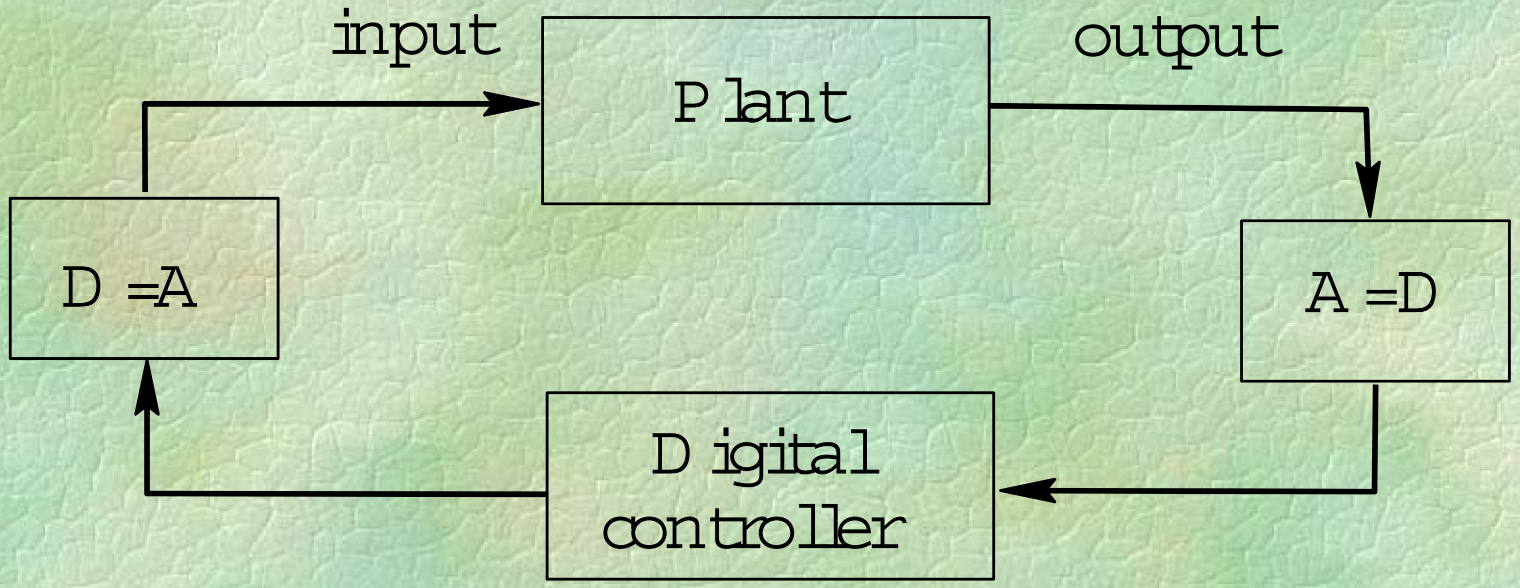
The corresponding time response is

$$y[k] = {}_{j}[\ {}_{j}]^{k} + \text{term s depending on the input}$$

Stability requires that $[\alpha_j]^k \to 0$, which is the case if $[\alpha_j] < 1$.

Hence stability requires the poles to have magnitude less than 1, i.e. to lie inside a unit circle centered at the origin.

Digital control of a continuous time plant



Details of how the plant input is reconstructed

When a zero order hold is used to reconstruct *u*(*t*), then

$$u(t) = u[k] \qquad \text{for} \quad k \qquad t < \ (k + 1)$$

Note that this is the staircase signal shown earlier in Figure 12.2. Discrete time models typically relate the sampled signal *y*[*k*] to the sampled input *u*[*k*]. Also a digital controller usually evaluates *u*[*k*] based on *y*[*j*] and *r*[*j*], where $\{r(k\Delta)\}$ is the reference sequence and $j \leq$ k.

Using Continuous Transfer Function Models

We observe that the generation of the staircase signal $u(t)$, from the sequence $\{u(k)\}$ can be modeled as in Figure 12.5.

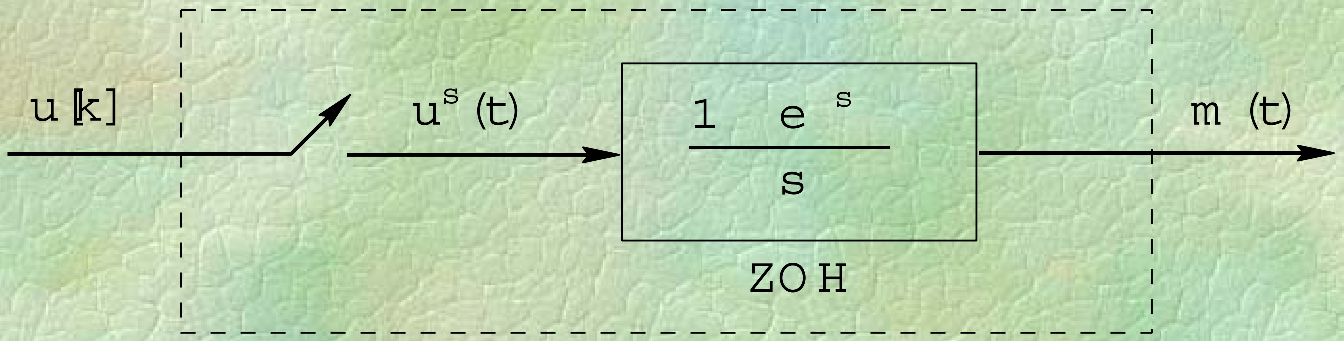
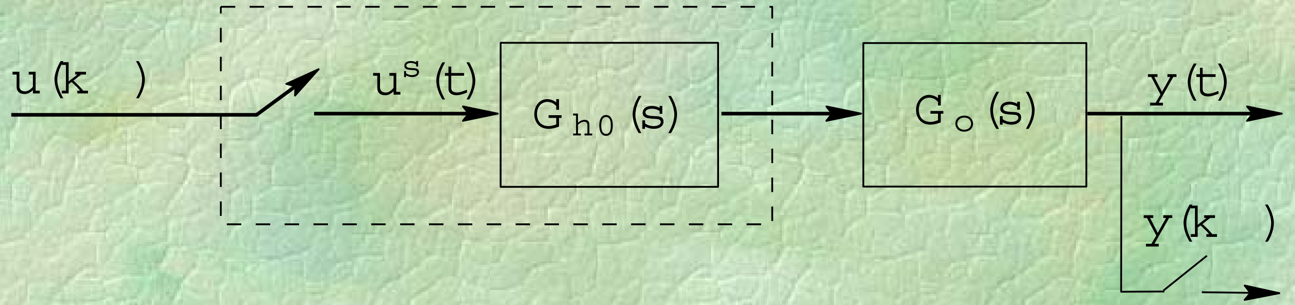


Figure 12.5: *Zero order hold*

Figure 12.6: *Discrete time equivalent model with zero order hold*

Combining the circuit on the previous slide with the plant transfer function $G_0(s)$, yields the equivalent connection between input sequence, $u(k\Delta)$, and sampled output $y(k\Delta)$ as shown below:



We saw earlier that the transfer function of a discrete time system, in Z-transform form is the Z-transform of the output (the sequence $\{y[k]\}$) when the input, $u[k]$, is a Kronecker delta, with zero initial conditions. We also have, from the previous slide, that if $u[k] = \delta_K[k]$, then the input to the continuous plant is a Dirac Delta, i.e. $u^s(t) = \delta(t)$. If we denote by $H_{eq}(z)$ the transfer function from $U_q(z)$ to $Y_q(z)$, we then have the following result.

$$H_{oq}(z) = \mathcal{Z}\,[\text{the sampled impulse response of } G_{h0}(s)G_o(s)]$$

$$= (1 - z^{-1})\,\mathcal{Z}\,[\text{the sampled step response of } G_o(s)]$$

Frequency Response of Sampled Data Systems

We evaluate the frequency response of a linear discrete time system having transfer function $H_q(z)$. Consider a sine wave input given by

$$u(k) = \sin(!k) = \sin\ 2\ k\frac{!}{!_s} = \frac{1}{2j}\ e^{j2\ k\frac{!}{!_s}}\ \ e^{\ j2\ k\frac{!}{!_s}}$$

where $\omega_s = \frac{2\pi}{\Delta}$.

Following the same procedure as in the continuous time case (see Lecture 4) we see that the system output response to the input is

$$y(k) = (!)\sin(!k + (!))$$

where

$$H_q(e^{j!}) = (!)e^{j\ (!)}$$

Figure 12.7: *Periodicity in the frequency response of sampled data systems.*

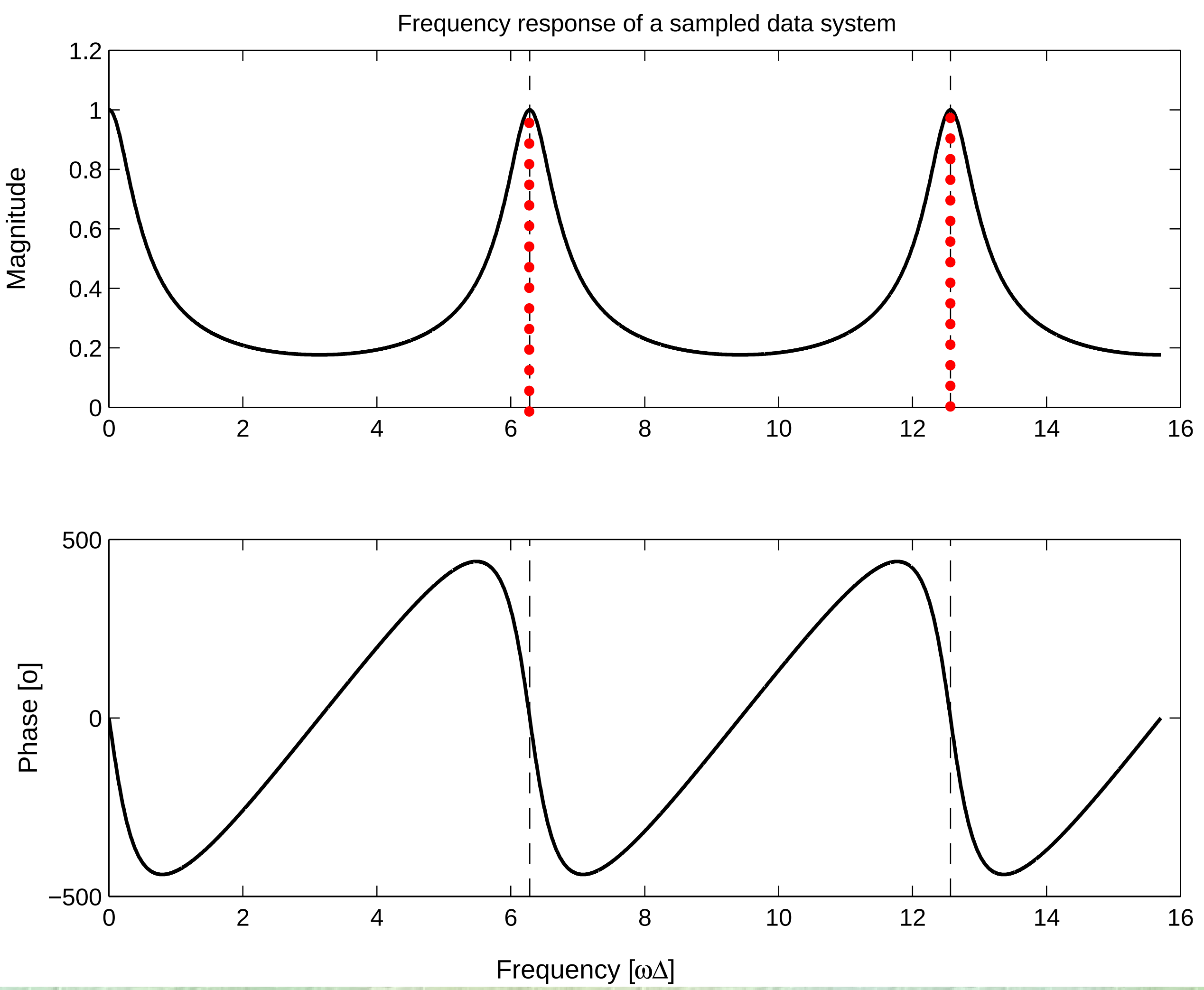
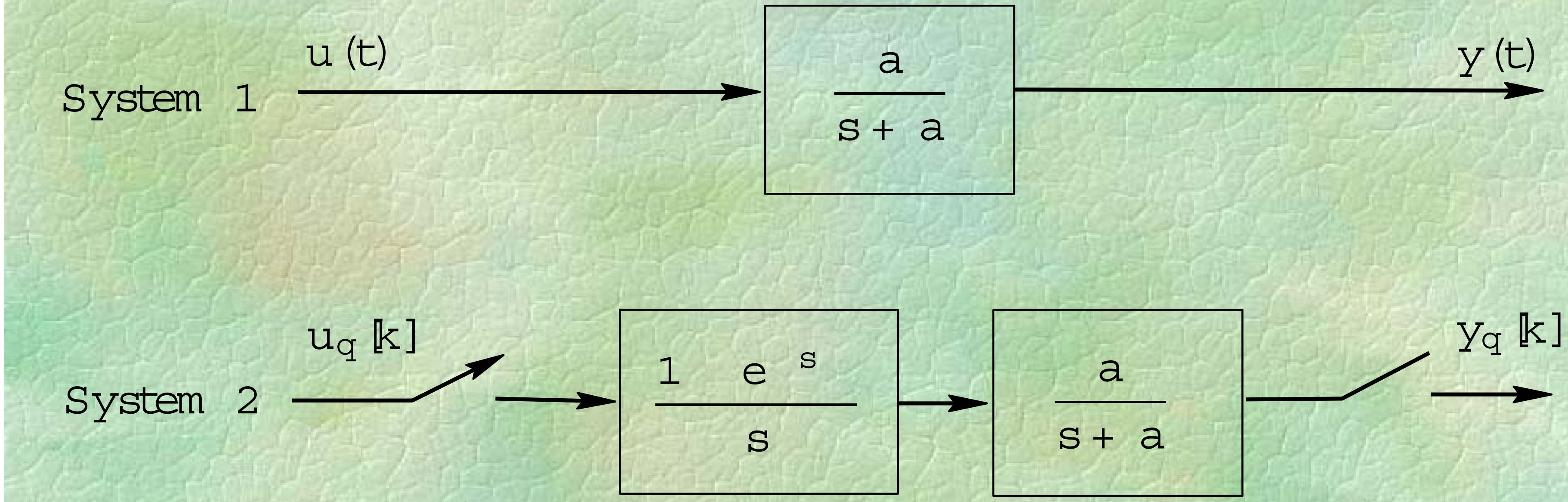
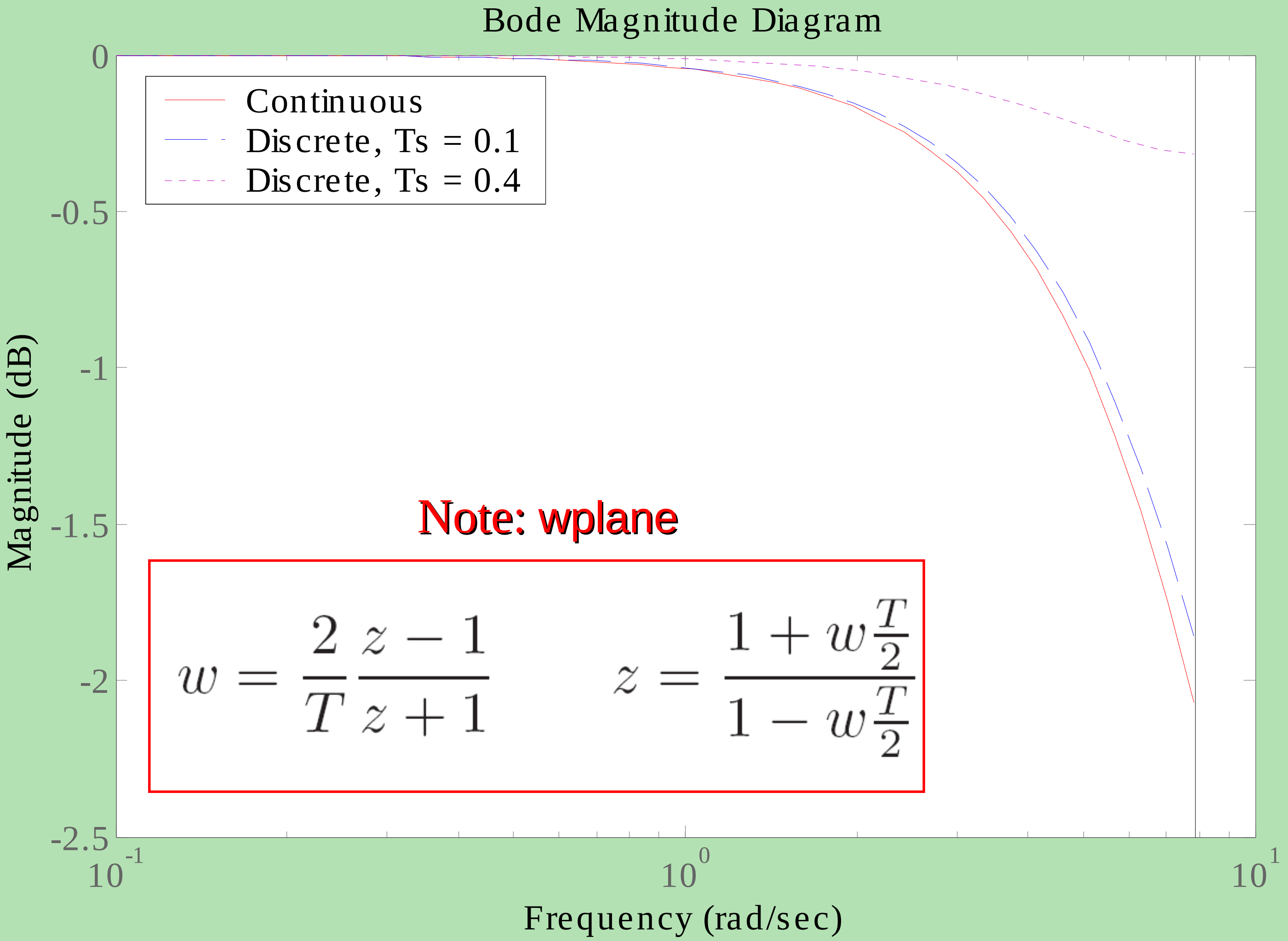


Figure 12.8: *Continuous and sampled data systems*





Note: wplane

$$w = \frac{2}{T}\frac{z-1}{z+1} \qquad z = \frac{1+w\frac{T}{2}}{1-w\frac{T}{2}}$$

Figure 12.9: ***Asymptotic behavior of a sampled data transfer function***

Causes of the Poor Response

It turns out that there are many reasons for the poor response. Some of these are:

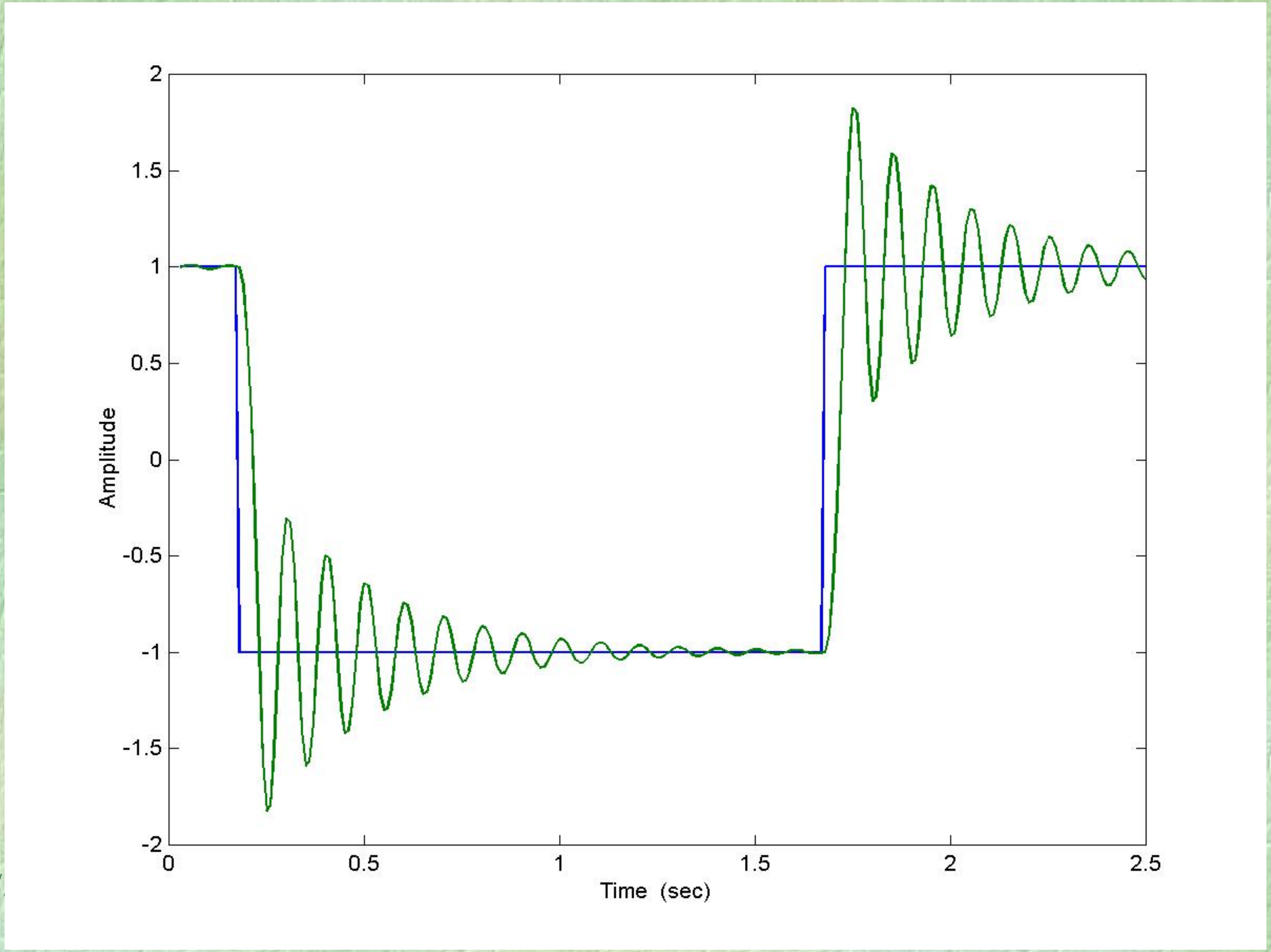
1. *Intersample issues*
2. *Noise*

The purpose of this chapter is to understand these issues. To provide motivation for the reader we will briefly examine these issues for a simple example.

1. Intersample Issues

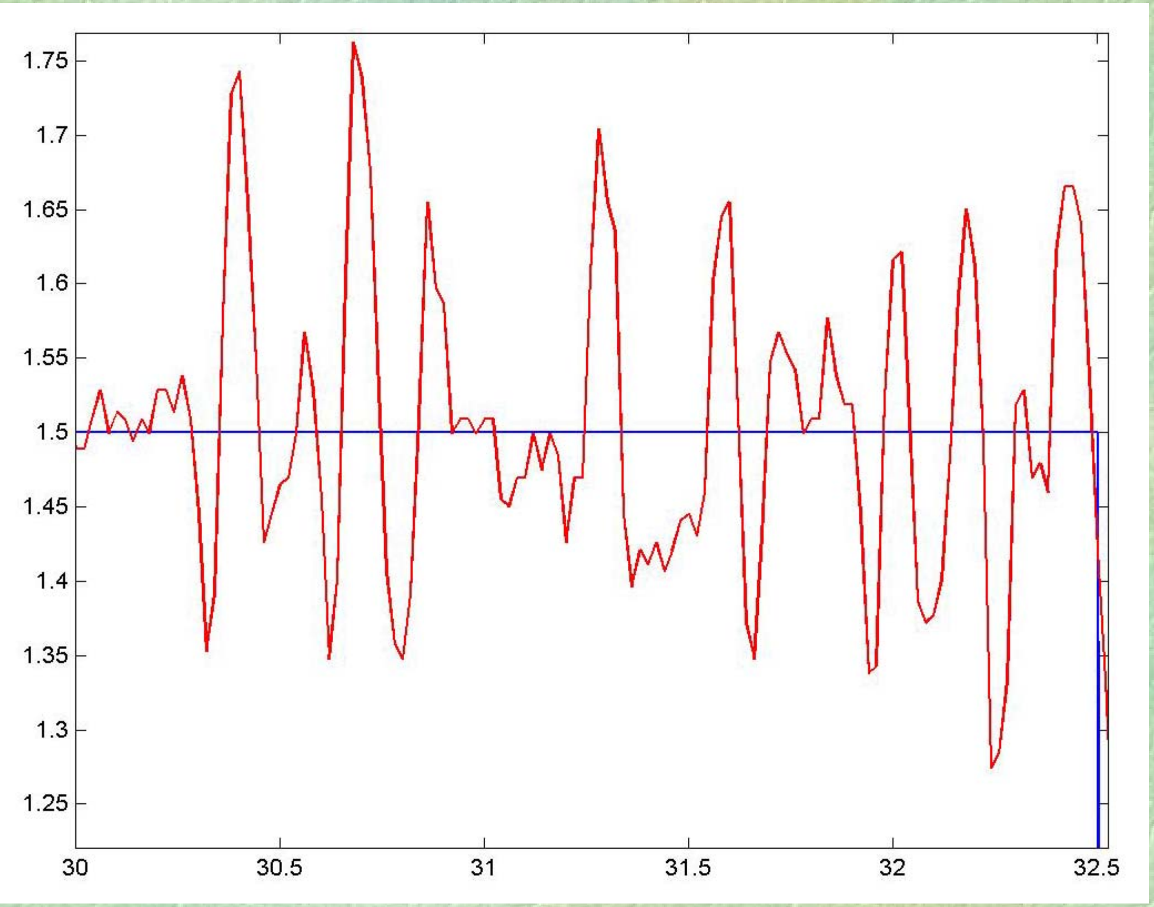
If we look at the output response at a rate faster than the control sampling rate then we see that the actual response is as shown on the next slide.

Simulation result showing full continuous output response



2. Noise

One further point that we have overlooked is that causing $y(t)$ to approach y^* as quickly as possible gives a very wide bandwidth controller. However, it should be clear that such a controller will necessarily magnify noise. Indeed, if we look at the steady response of the system (*see the next slide*) then we can see that noise is indeed causing problems.


1.75
1.7
1.65
1.6
1.55
1.5
1.45
1.4
1.35
1.3
1.25
30
30.5
31
31.5
32
32.5

Summary

- Very few plants encountered by the control engineer are digital, most are continuous. That is, the control signal applied to the process, as well as the measurements received from the process, are usually continuous time.
- Modern control systems, however, are almost exclusively implemented on digital computers.
- Compared to the historical analog controller implementation, the digital computer provides
  - much greater ease of implementing complex algorithms,
  - convenient (graphical) man-machine interfaces,
  - logging, trending and diagnostics of internal controller and
  - flexibility to implement filtering and other forms of signal processing operations.

- Digital computers operate with sequences in time, rather than continuous functions in time.
  Therefore,
  - input signals to the digital controller-notably process measurements - must be sampled;
  - outputs from the digital controller-notably control signals - must be interpolated from a digital sequence of values to a continuous function in time.
- Sampling (see next slide) is carried out by A/D (analog to digital converters.

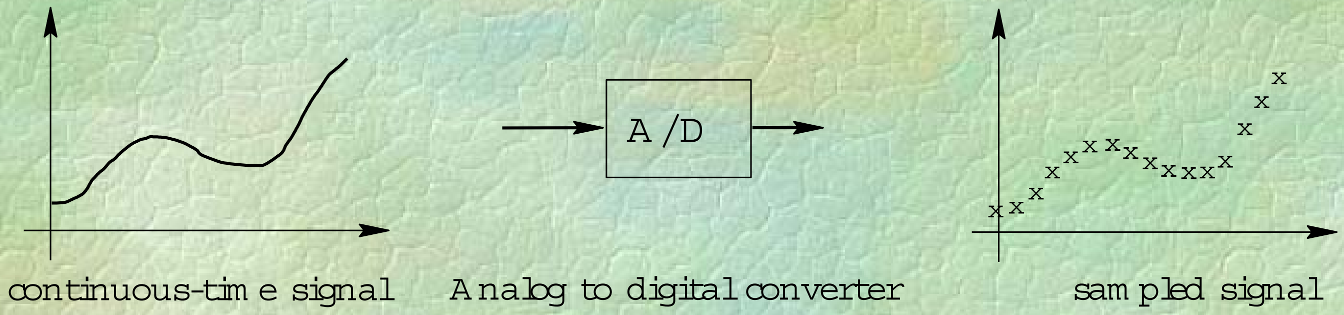


Figure 12.10: *The result of sampling*

- The converse, reconstructing a continuous time signal from digital samples, is carried out by D/A (digital to analog) converters. There are different ways of interpolating between the discrete samples, but the so called zero-order hold (see next slide) is by far the most common.

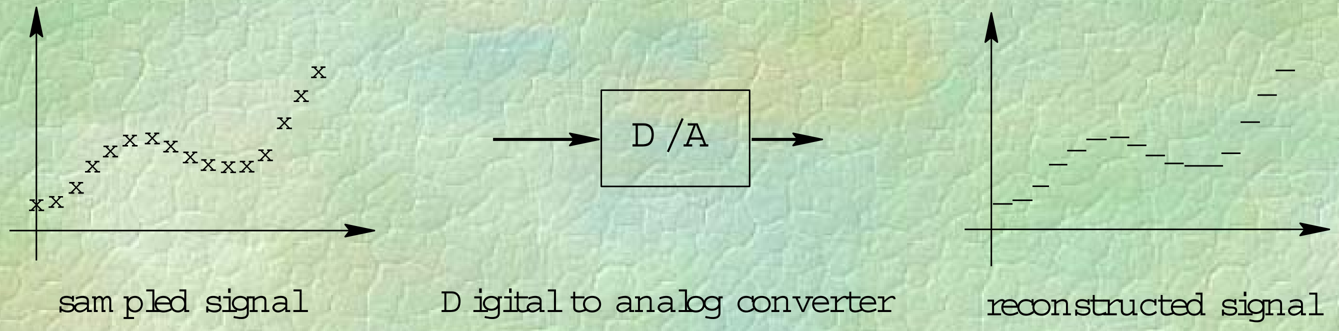


Figure 12.11: *The result of reconstruction*

- When sampling a continuous time signal,
  - an appropriate sampling rate must be chosen
  - an anti-aliasing filter (low-pass) should be included to avoid frequency folding.
- Analysis of digital systems relies on discrete time versions of the continuous operators.

- The chapter has introduced the discrete operator:
  - the shift operator, q, defined by $qx[k] \triangleq x[k+1]$

- The shift operator, q,
  - is the traditional operator;
  - is the operator many engineers feel more familiar with;
  - is used in the majority of the literature.

- Analysis of digital systems relies on discrete time versions of the continuous operators:
  - the discrete version of the differential operator is difference operator;
  - the discrete version of the Laplace Transform is the Z-transform (associated with the shift operator).
- With the help of these operators,
  - continuous time differential equation models can be converted to discrete time difference equation models;
  - continuous time transfer or state space models can be converted to discrete time transfer or state space models in either the shift or δ operators.

Digital Control

A key idea is that if one is only interested in the at-sample response, these samples can be described by discrete time models in the delta operator. For example, consider the sampled data control loop shown below

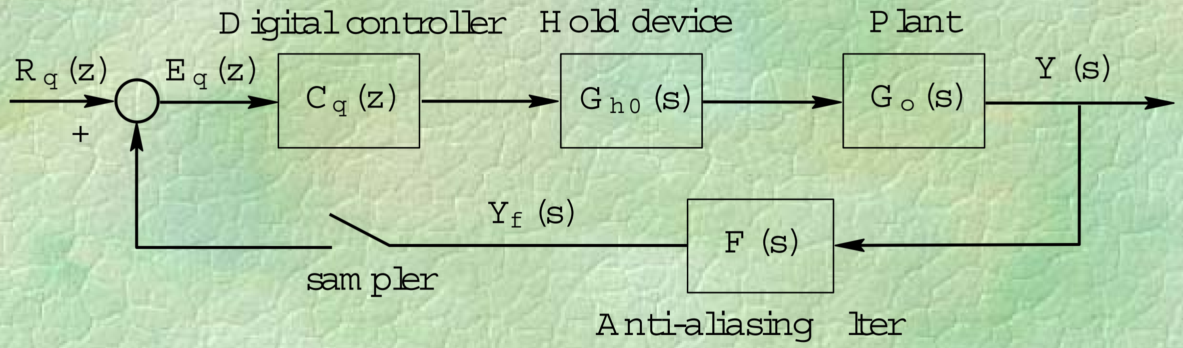


Figure 13.1: *Sampled data control loop*

If we focus only on the sampled response then it is straightforward to derive an equivalent discrete model for the at-sample response of the hold-plant-anti-aliasing filter combination.

We use the transfer function form, and recall the following forms for the discrete time model:

(a) With anti-aliasing filter F

$$[FG_0G_{h0}]_q(z),\ Z\{\text{sampled impulse response of } F(s)G_0(s)G_{h0}(s)\}$$

(b) Without anti-aliasing filter

$$[G_0G_{h0}]_q(z),\ Z\{\text{sampled impulse response of } G_0(s)G_{h0}(s)\}$$

Are there special features of digital control models?

Many ideas carry directly over to the discrete case. For example, one can easily do discrete pole assignment. Of course, one needs to remember that the discrete stability domain is different from the continuous stability domain. However, this simply means that the desirable region for closed loop poles is different in the discrete case.

We are led to ask if there are any real conceptual differences between continuous and discrete.

Continuous-Discrete Poles

Functions converge to the underlying continuous time descriptions. In particular, the relationship between continuous and discrete poles is as follows:

$$p_i^d = e^{p_i T} \quad \text{or} \quad p_i^d \cong p_i T + 1, \quad i = 1, \cdots, n$$

where p_i^d, p_i denote the discrete (*z*-domain) poles and continuous time poles, respectively.

Continuous-Discrete Zeros

The relationship between continuous and discrete zeros is more complex. Perhaps surprisingly, all discrete time systems turn out to have relative degree 1 irrespective of the relative degree of the original continuous system.

Hence, if the continuous system has n poles and $m(< n)$ zeros then the corresponding discrete system will have n poles and $(n\text{-}1)$ zeros. Thus, we have n-m+1 extra discrete zeros. We therefore (*somewhat artificially*) divide the discrete zeros into two sets.

In the control of discrete time systems special care needs to be taken with the sampling zeros. For example, these zeros can be non-minimum phase even if the original continuous system is minimum phase.

Is a Dedicated Digital Theory Really Necessary?

We could well ask if it is necessary to have a separate theory of digital control or could one simply map over a continuous design to the discrete case. The possible design options are:

1) Design the controller in continuous time, discretise the result for implementation and ensure that the sampling constraints do not significantly affect the final performance.
2) Work in discrete time by doing an exact analysis of the *at-sample* response and ensure that the intersample response is not too surprising.

We will analyze and discuss these possibilities below.

1. Approximate Continuous Designs

Given a continuous controller, $C(s)$, we mention the methods drawn from the digital signal processing literature for determining an *equivalent* digital controller.

1.1 Simply take a continuous time controller expressed in terms of the Laplace variable, s and then replace every occurrence of s by the corresponding shift domain operator z. This leads to the following digital control law:

$$\boxed{\overline{C}_1(z) = C(s)\big|_{s=\frac{z-1}{T}}}$$

where $C(s)$ is the transfer function of the continuous time controller and where $\overline{C}_1(z)$ is the resultant transfer function of the discrete time controller in the shift form.

1.2 Convert the controller to a zero order hold discrete equivalent. This is called a *step invariant transformation,* Hold Equivalence (HE). This leads to

$$\overline{C}_2(\ ) = D\ [\text{sampled impulse response of } \{C(s)G_{h0}(s)\}]$$

where $C(s)$, $G_{h0}(s)$ and $\overline{C}_2(\gamma)$ are the transfer functions of the continuous time controller, zero order hold and resultant discrete time controller respectively.

1.2.1 Convert the controller to a simple discrete equivalent. This leads to

$$\boxed{\overline{C}_{2.1}(\gamma) = \mathrm{D}\left[\text{sampled impulse response of } \{\mathrm{C(s)}\}\right]}$$

where $C(s)$ and $\overline{C}_{2.1}(\gamma)$ are the transfer functions of the continuous time controller and the resultant discrete time controller, respectively.

1.3 We could use a more sophisticated mapping from *s* to γ. For example, we could carry out the following transformation, commonly called a *bilinear transformation with pre-warping*. We first let

$$s = \frac{}{\frac{}{2} \quad + 1} (\ ) \qquad = \frac{s}{\frac{}{2}s}$$

The discrete controller is then defined by

$$\overline{C}_3 (\ ) = C\,(s)\mathrm{j}_{s = \frac{}{\frac{}{2} \ + 1}}$$

2. At-Sample Digital Design

The next option we explore is that of doing an exact digital control system design *for the sampled response*.

We recall that the sampled response is exactly described by appropriate discrete-time-models (expressed in either the shift operator z).

Time Domain Design

Any algebraic technique (*such as pole assignment*) has an immediate digital counterpart. Essentially all that is needed is to work with z (*or* γ) instead of the Laplace variable, s, and to keep in mind the different region for closed loop stability.

Frequency Domain Design

Automatic design techniques can be exploited for frequency domain design.

Common frequency domain design tools are:

- Bode plots;
- Root locus;
- Nyquist diagrams.

- See laboratory experiences and practical applications…

Summary

- There are a number of ways of designing digital control systems:
  - design in continuous time and discretise the controller prior to implementation;
  - model the process by a digital model and carry out the design in discrete time.
- Continuous time design which is discretised for implementation:
  - Continuous time signals and models are utilized for the design;
  - Prior to implementation, the controller is replaced by an equivalent discrete time version;
  - Equivalent means to simply map s to z (where z is the shift operator);

  - Caution must be exercised since the analysis was carried out in continuous time and the expected results are therefore based on the assumption that the sampling rate is high enough to mask sampling effects;
  - If the sampling period is chosen carefully, in particular with respect to the open and closed loop dynamics, then the results should be acceptable.

- Discrete design based on a discretised process model:
  - First the model of the continuous process is discretised;
  - Then, based on the discrete process, a discrete controller is designed and implemented;
  - Caution must be exercised with so called intersample behavior: the analysis is based entirely on the behavior as observed at discrete points in time, but the process has a continuous behavior also between sampling instances;

  - Problems can be avoided by refraining from designing solutions which appear feasible in a discrete time analysis, but are known to be unachievable in a continuous time analysis.

- The following rules of thumb will help avoid intersample problems if a purely digital design is carried out:
  - sample 10 times the desired closed loop bandwidth;
  - always check the intersample response.

Hybrid Control…

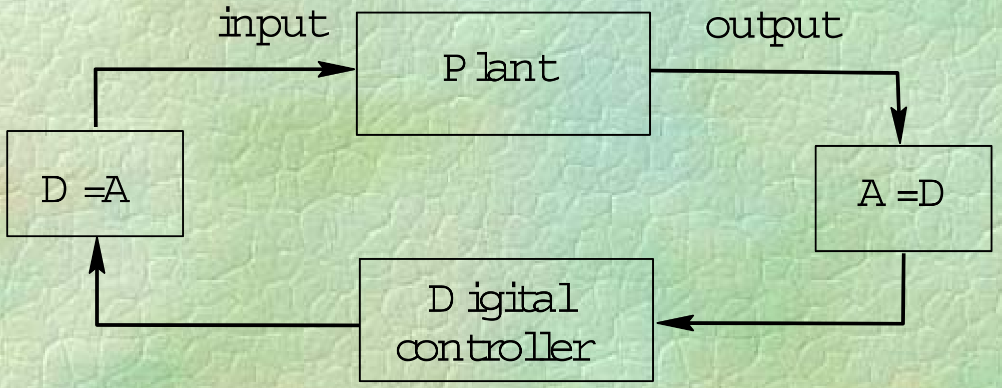
Final Comments, Remarks and Conclusion

Motivation

In this lecture we will introduce the concept of *Hybrid Control*. By this terminology we mean the combination of a *digital* control law with a *continuous-time* system. We will be particularly interested in analysing the continuous response and the connections with the sampling points.

We recall the motivations and the main design concepts presented in the slides for the previous lectures.

The set-up for digital control of this system is shown schematically below:



The objective is to cause the output shaft position, $y(t)$, to follow a given reference signal, $y^*(t)$.

$$u\left(k\Delta\right)=\frac{y^*\left(\overline{k+1\Delta}\right)-\alpha_1 y\left(\overline{k-1\Delta}\right)-\alpha_2 y\left(\overline{k-2\Delta}\right)-\beta_2 u\left(\overline{k-\Delta}\right)-\beta_3 u\left(\overline{k-2\Delta}\right)}{\beta_1}$$

Notice that the above control law expresses the current control $u(k\Delta)$ as a function of

- the reference, $y^*\left(\overline{k+1\Delta}\right)$
- past output measurements, $y\left(\overline{k-1\Delta}\right), y\left(\overline{k-2\Delta}\right)$
- past control signals, $u\left(\overline{k-1\Delta}\right), u\left(\overline{k-2\Delta}\right)$

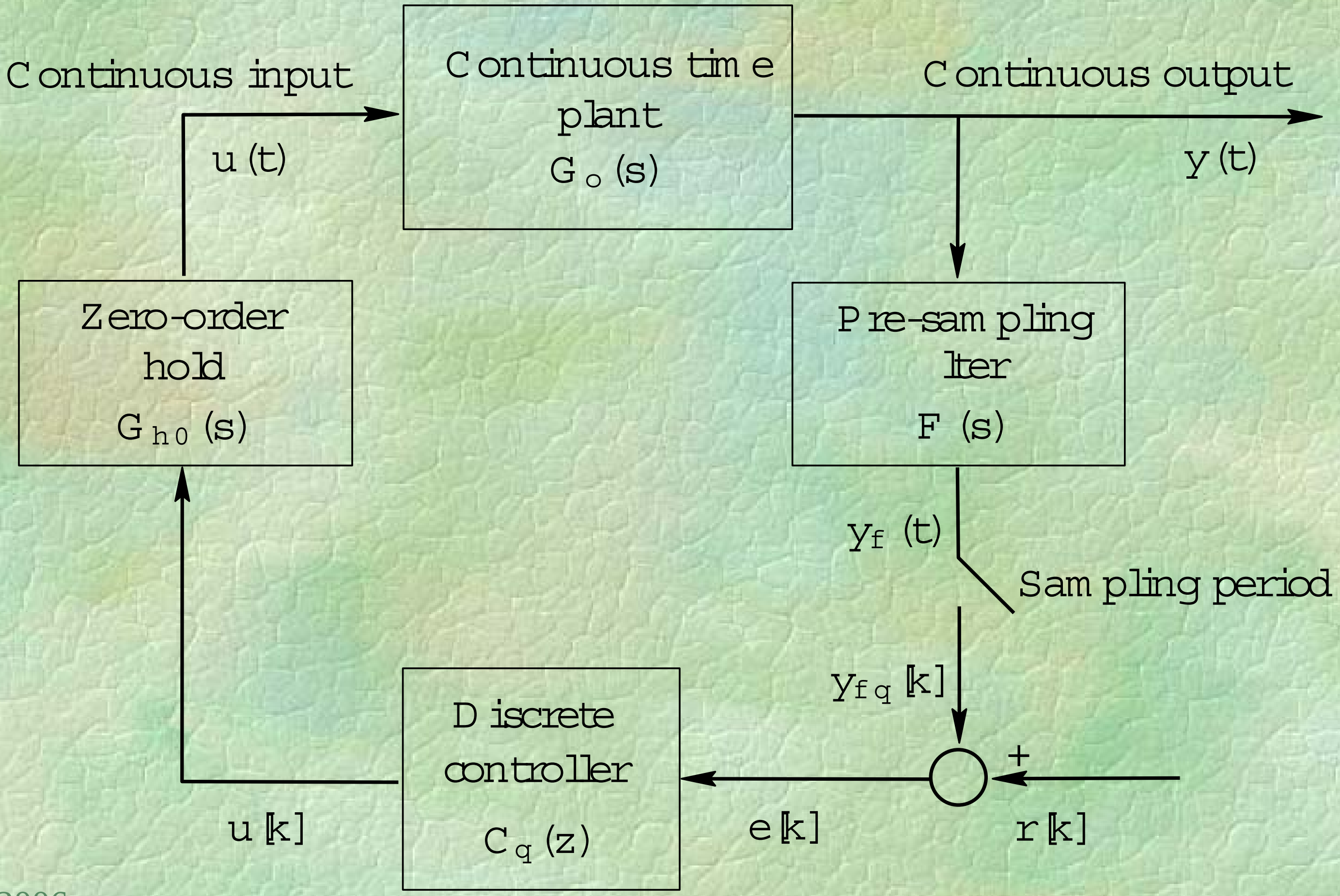
Models for Hybrid Control Systems

A hybrid control loop containing both continuous and discrete time elements is shown in Figure 14.1.

We denote the discrete equivalent transfer function of the combination {zero order hold + Continuous Plant + Filter} as $[FG_0G_{h0}]_q$. We have

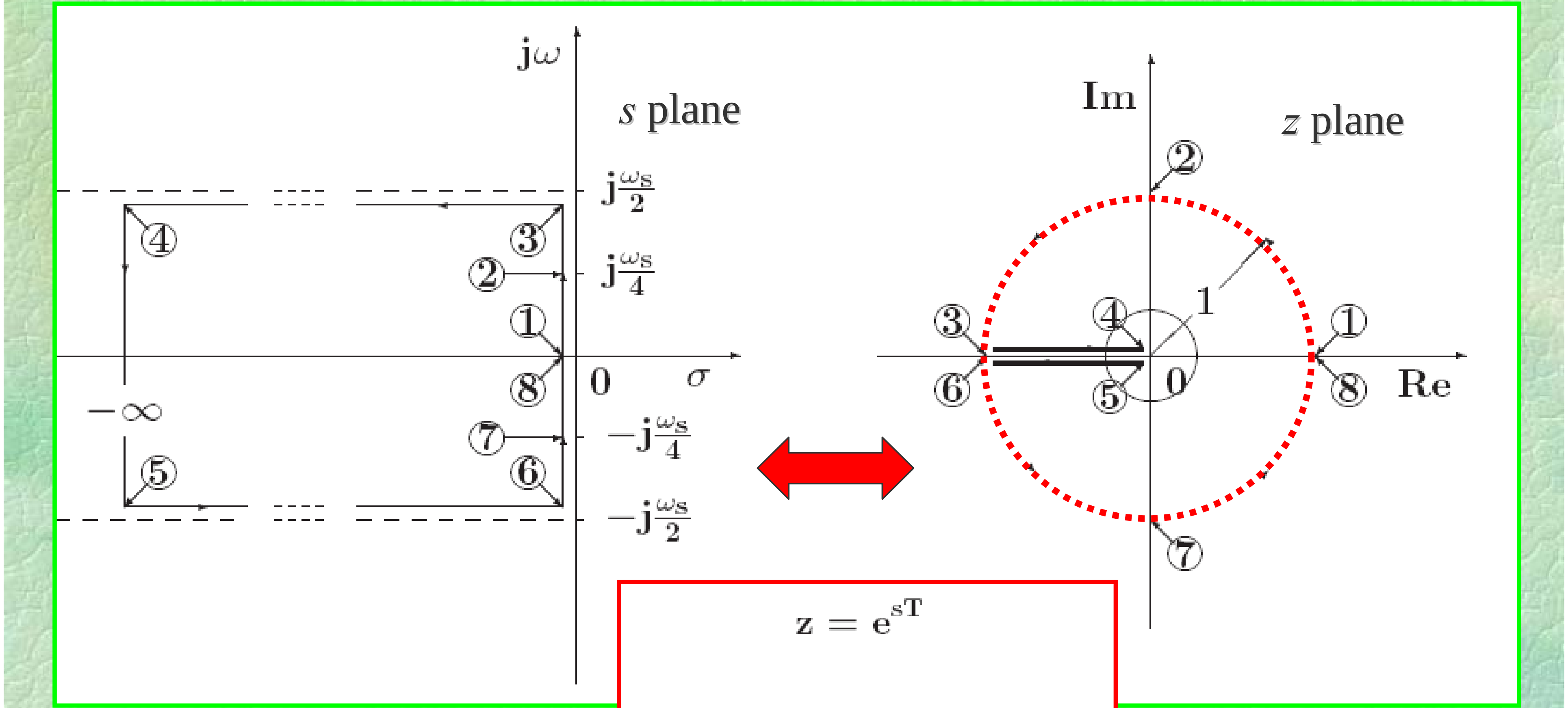
$$[FG_oG_{h_0}]_q = \mathcal{Z}\{\text{sampled impulse response of } F(s)G_o(s)G_{h_0}(s)\}$$

Figure 14.1: *Sampled data control loop. Block form*



Design Remarks and Recalling...

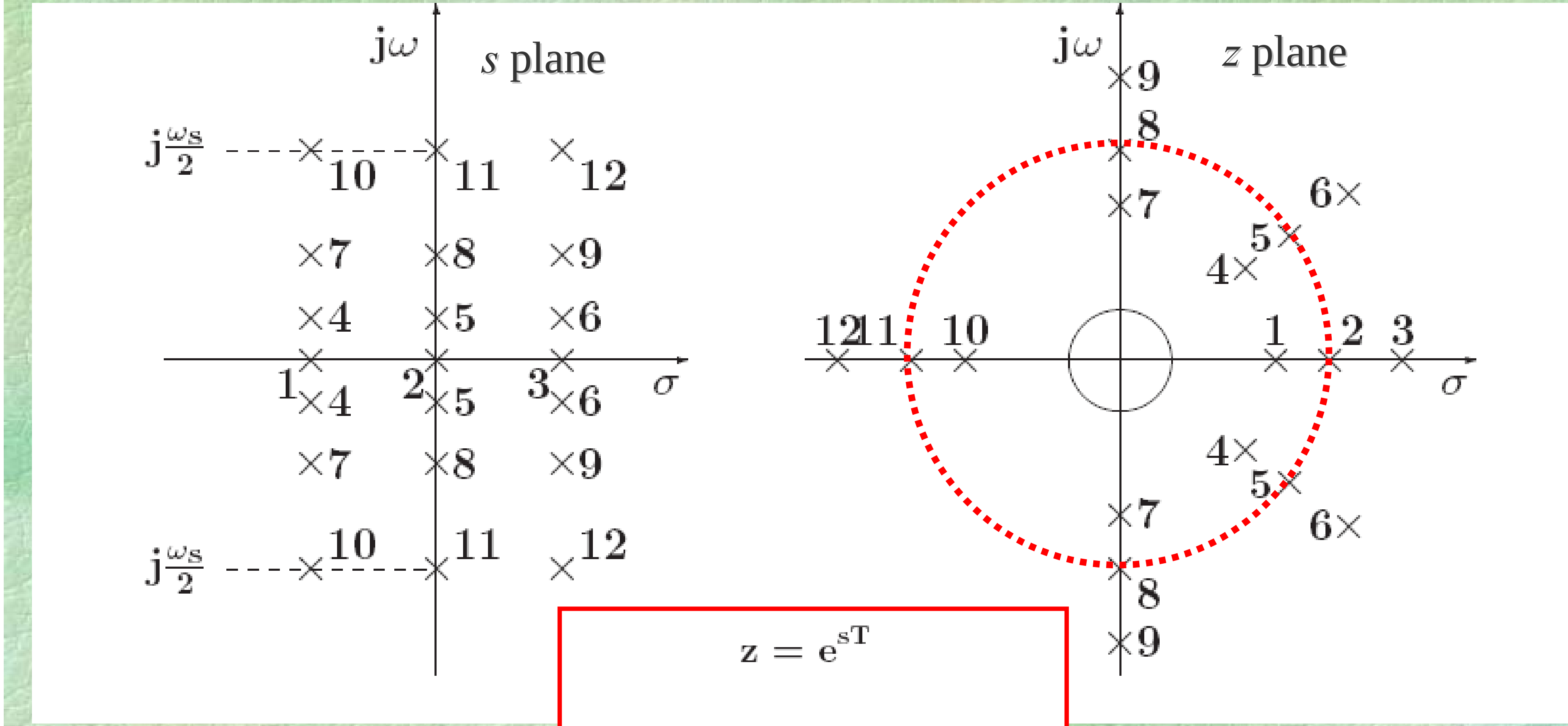
Link between *z* and *s* planes *(1)*



$$z = e^{sT}$$

$$z = e^{T(\sigma+j\omega)} = e^{T\sigma}e^{jT\omega}$$

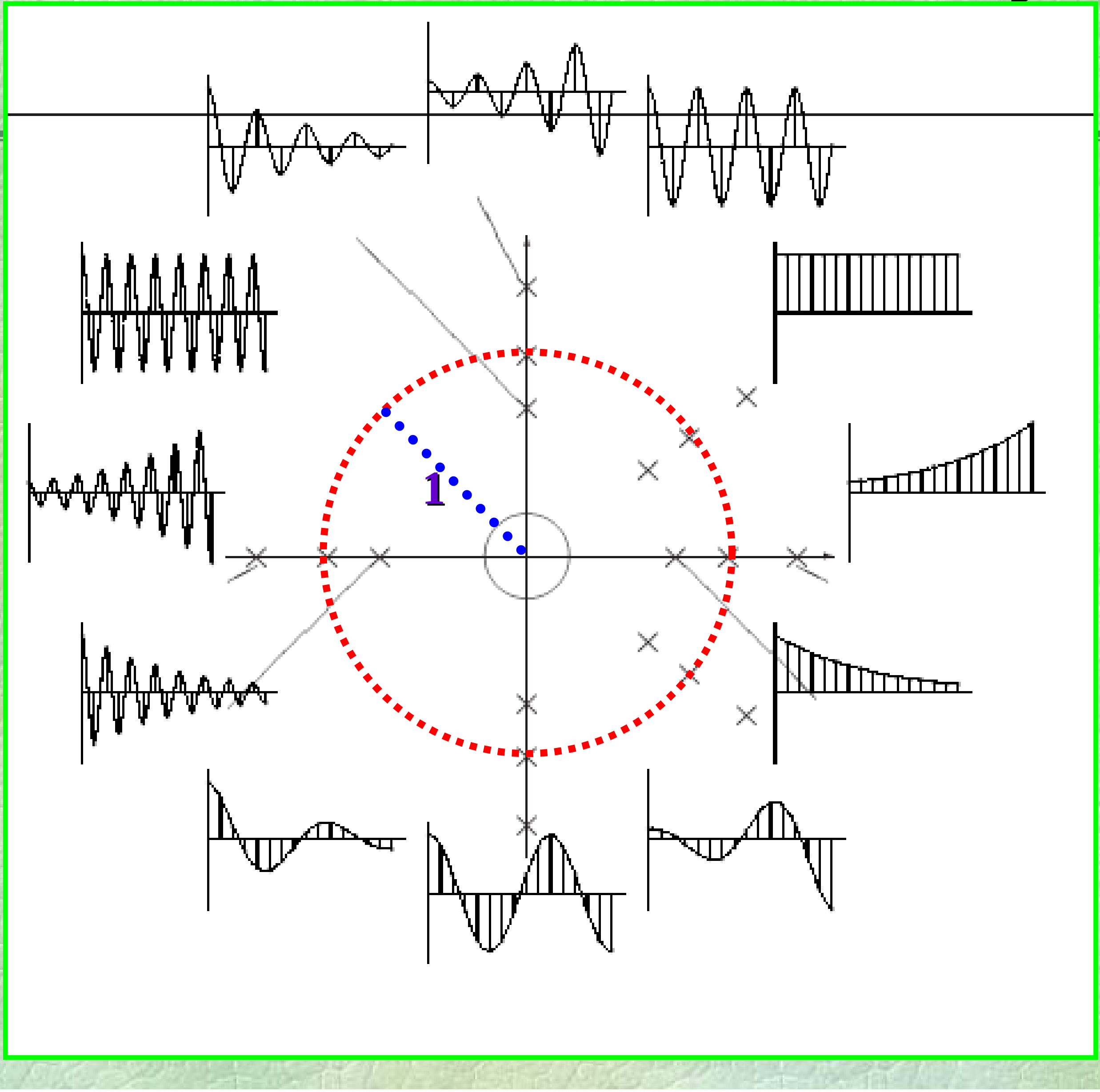
Link between *z* and *s* planes (2)



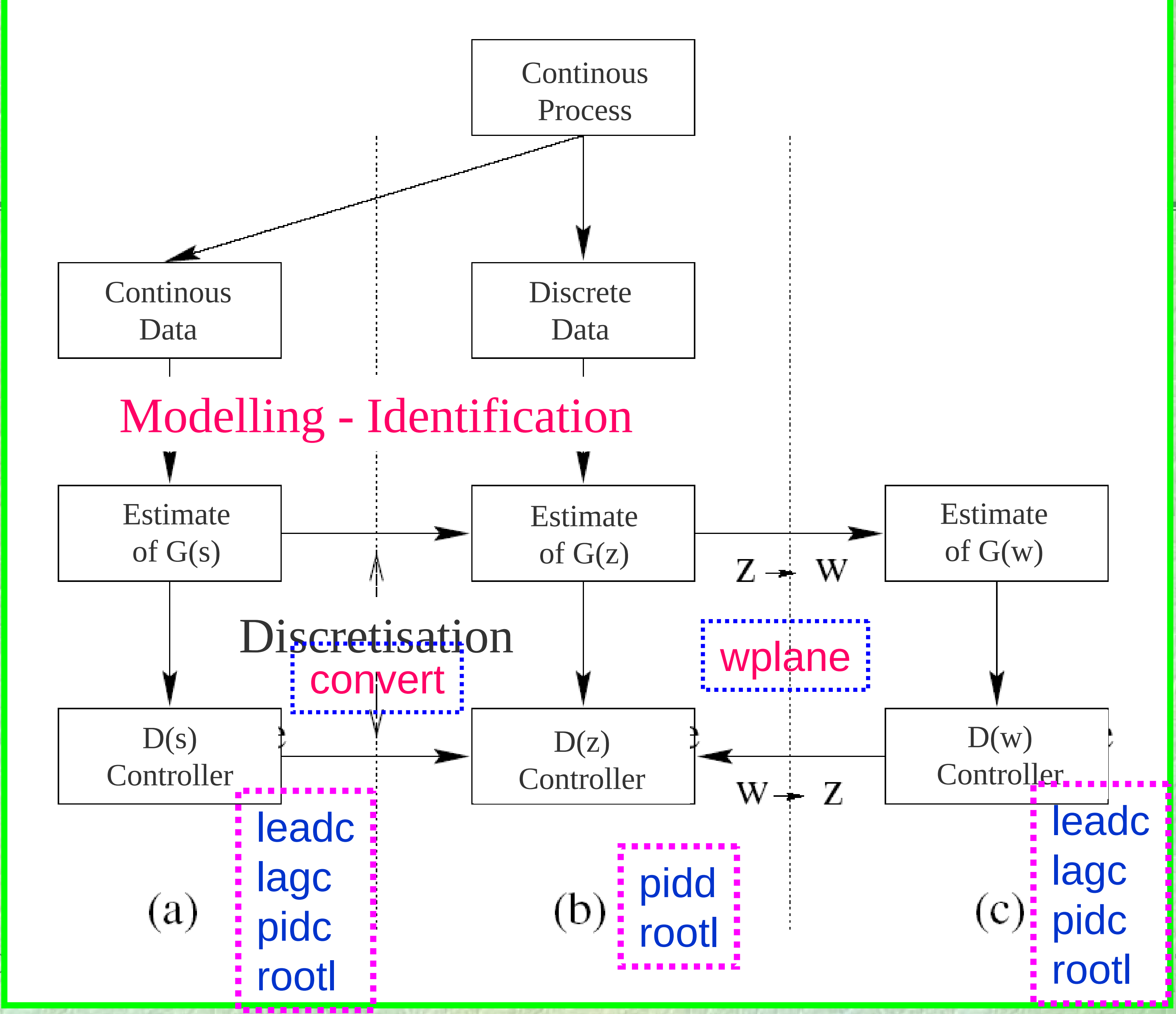
$$z = e^{sT}$$

$$z = e^{T(\sigma + j\omega)} = e^{T\sigma} e^{jT\omega}$$

Discrete Model Stability



Design Strategy Overview



Continous Process

Continous Data

Discrete Data

Modelling - Identification

Estimate of G(s)

Estimate of G(z)

Estimate of G(w)

Z → W

Discretisation

convert

wplane

D(s) Controller

D(z) Controller

D(w) Controller

W → Z

leadc
lagc
pidc
rootl

(a)

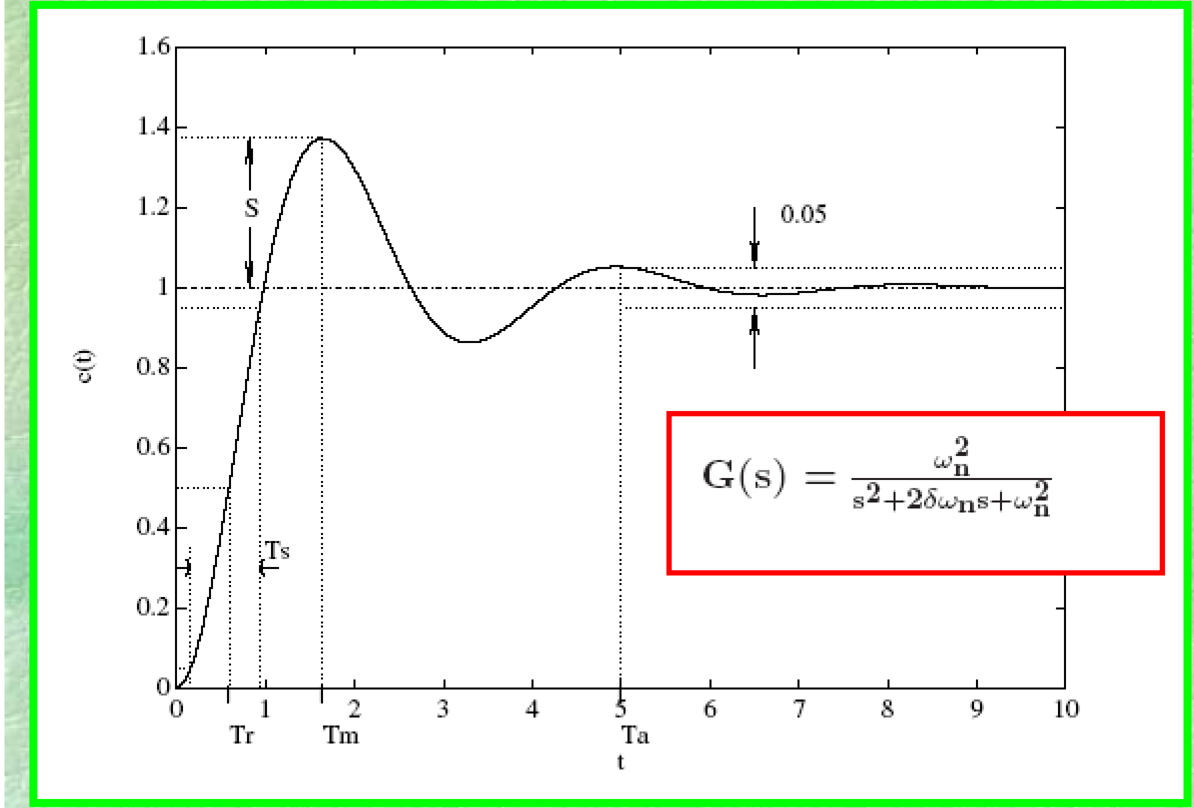
(b)

pidd
rootl

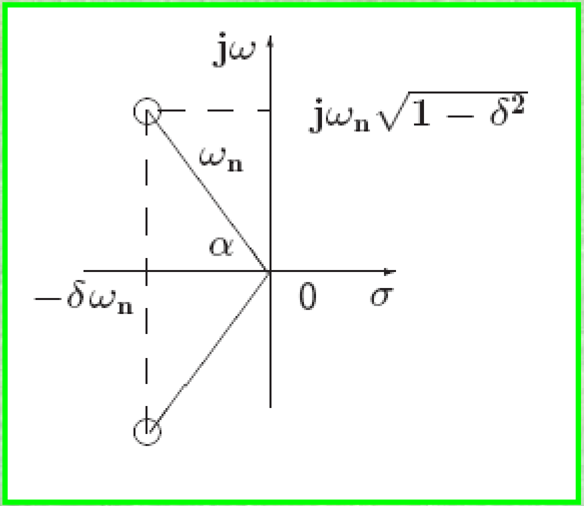
(c)

leadc
lagc
pidc
rootl

2nd order system Step Response *(1)*



$$G(s) = \frac{\omega_n^2}{s^2+2\delta\omega_n s+\omega_n^2}$$

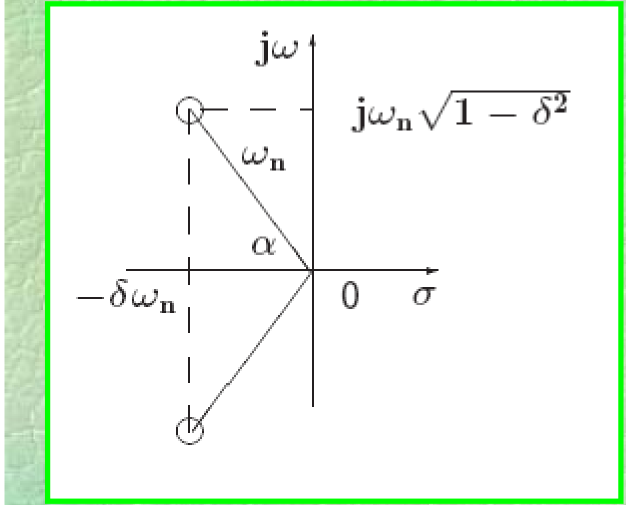


(1)

$$T_a = \frac{3}{\delta\omega_n}$$

Settling Time

2nd order system Step Response *(2)*

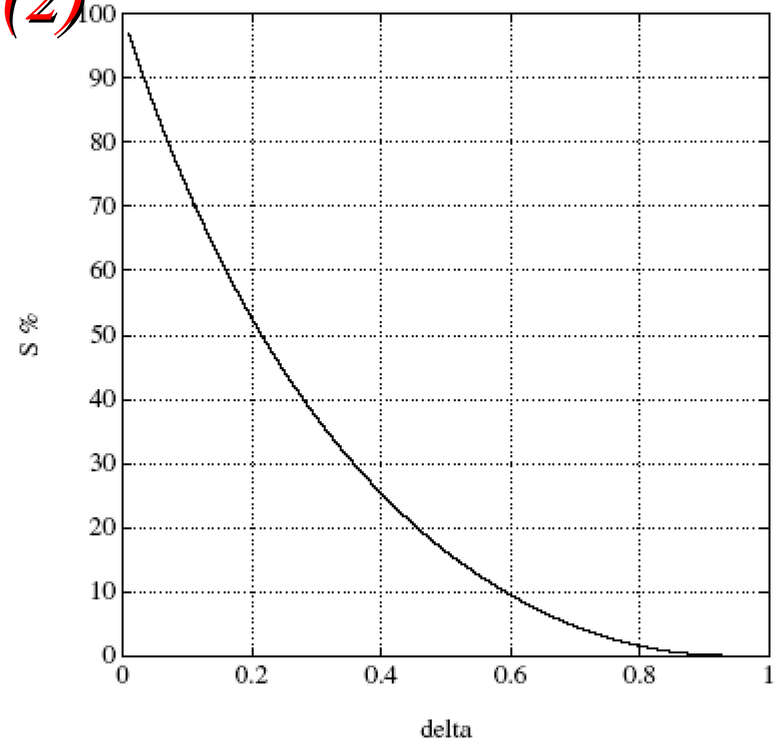


Overshoot

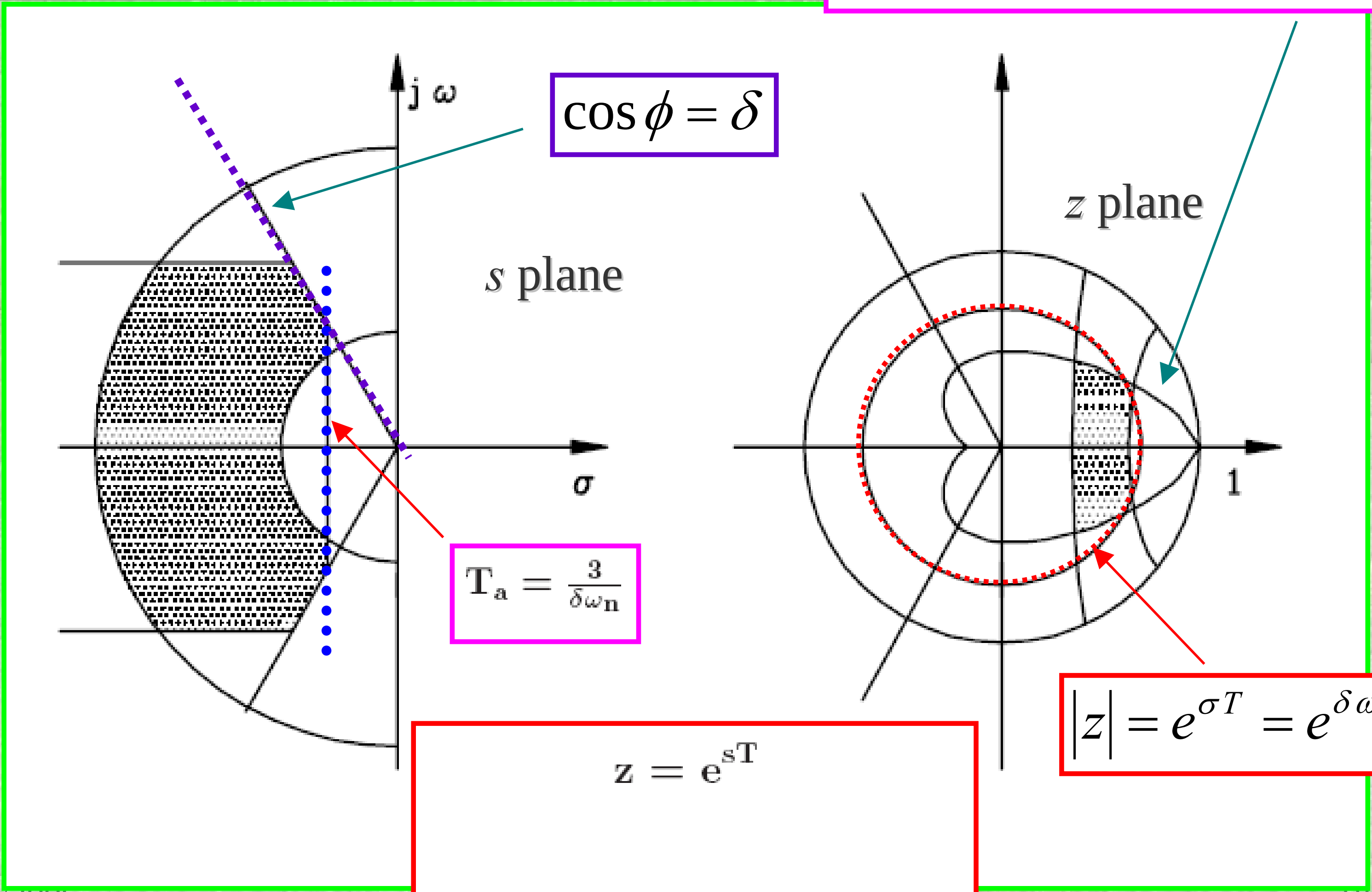
$$G(s) = \frac{\omega_n^2}{s^2+2\delta\omega_n s+\omega_n^2}$$

$$S = 100[c(T_m) - 1] = 100\, e^{-\frac{\delta\pi}{\sqrt{1-\delta^2}}}$$

(2)



$$S = 100[c(T_m) - 1] = 100\, e^{-\frac{\delta\pi}{\sqrt{1-\delta^2}}}$$



$$\cos\phi = \delta$$

$$T_a = \frac{3}{\delta\omega_n}$$

$$|z| = e^{\sigma T} = e^{\delta\omega_n T}$$

$$z = e^{sT}$$

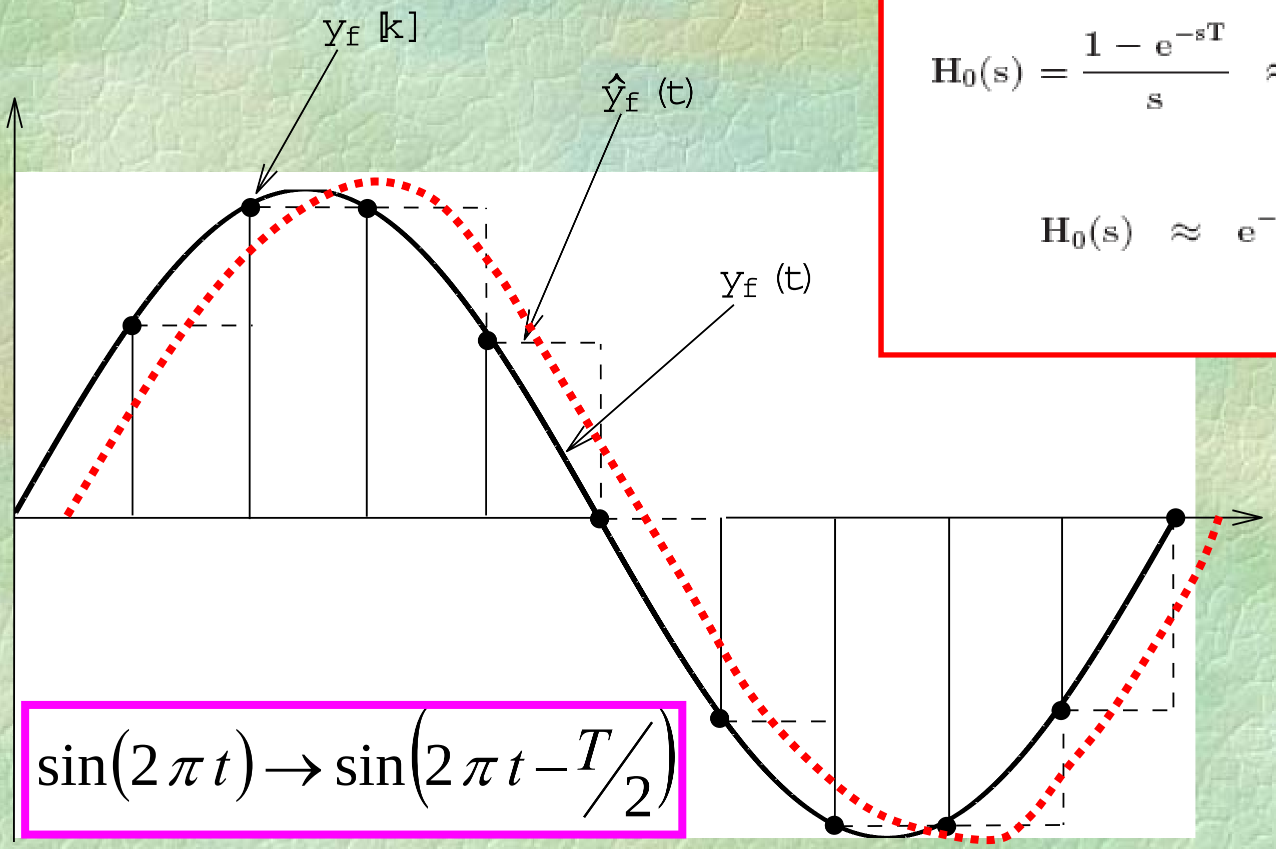
$$z = e^{T(\sigma+j\omega)} = e^{T\sigma} e^{jT\omega}$$

Zero Order Hold Effects...

Figure 14.2: *Connections between* $y_f(t)$, $y_f[k]$ *and* $\hat{y}_f(t)$ *for* $y_f(t) = \sin(2\pi t)$, $\Delta=0.1$

$$H_0(s) = \frac{1-e^{-sT}}{s} \approx \frac{T}{\frac{T}{2}s+1}$$

$$H_0(s) \approx e^{-sT/2}$$



$$\sin(2\pi t) \rightarrow \sin\left(2\pi t - T/2\right)$$

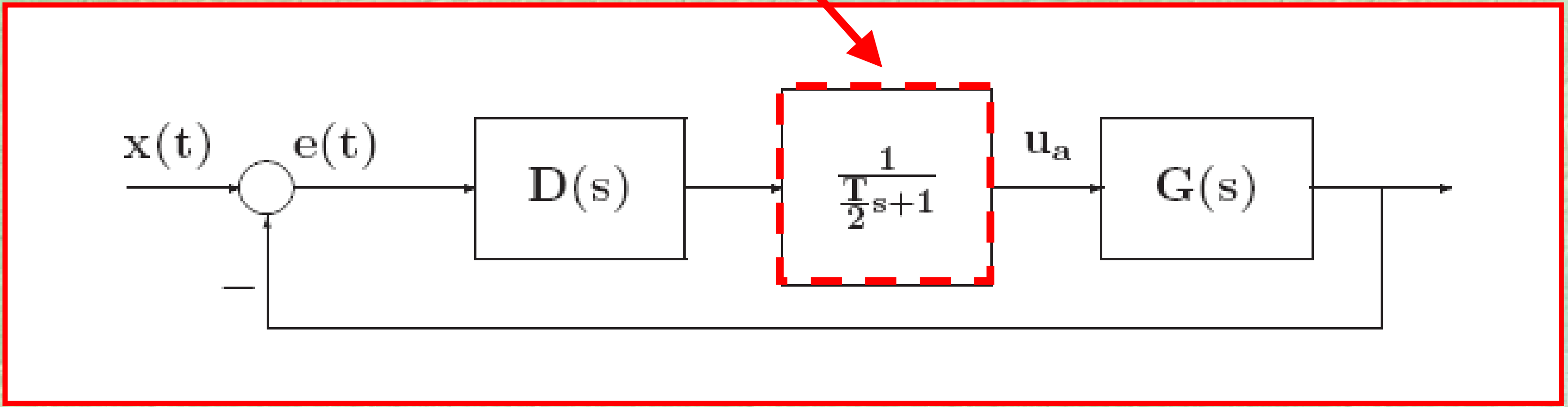
Phase Margin Degradation!

$$H_0(s) = \frac{1 - e^{-sT}}{s} \approx \frac{T}{\frac{T}{2}s + 1}$$

$$H_0(s) \approx e^{-sT/2}$$

$$\boxed{M_f^{(h0)} \cong M_f - T/2}$$

!!!



Discretisation Techniques...

$$D(z) = D(s)\big|_{s=\frac{z-1}{T}}$$

Euler

$$D(z) = \mathcal{Z}\left[\mathcal{L}^{-1}[D(s)]\right]$$

Sampled Impulse Rensponse Discretisation

$$D(z) = (1-z^{-1})\mathcal{Z}\left[\frac{D(s)}{s}\right] = \mathcal{Z}\left[\frac{1-e^{-sT}}{s}D(s)\right]$$

Hold Equivalence

Summary

- Hybrid analysis allows one to mix continuous and discrete time systems properly.
- Hybrid analysis should always be utilized when design specifications are particularly stringent and one is trying to push the limits of the fundamentally achievable.