Course Overview

- Topic 1: Introduction to State Space Modelling
- Topic 2: Lagrangian Mechanics
- Topic 3: Linearisation of Non-linear Differential Equations
- Topic 4: More State Space Modelling
- Topic 5: Modelling Multiple DOF Systems
- Topic 6: Modelling Distributed Parameter Systems
- Topic 7: Conversion between SS to TF and back again
- Topic 8: Solution to state equations, poles, zeros and stability
- Topic 9: Controllability and Observability
- Topic 10: Feedback Control & Pole Placement
- Topic 11: Optimal Control (LQR)

Course Overview cont.

- Topic 12: Observers (Estimators)
- Topic 13: Optimal Observers (Kalman-Bucy Filters, LQG)
- Topic 14: Reduced Order Observers
- Topic 15: Compensators
- Topic 16: Reference Input & Command Tracking
- Topic 17: Summary
- Bibliography

Course Objectives

- Have an understanding of basic control concepts such as controllability, observability, poles and zeros, stability
- Be able to construct state space models of a given dynamic system
- Be able to design a full-state control system
- Be able to design an optimal control system
- Be able to design and build a state estimator
- Be able to simulate state space systems in Matlab
- Have had experience with designing real control systems
Graduate Attributes

1. Ability to apply knowledge of basic science and engineering fundamentals assured through written examination and assignments
2. Ability to communicate effectively, not only with engineers but with the community at large developed through in-class discussion but not assured
3. In-depth technical competence in at least one engineering discipline assured through written examination and assignments
4. Ability to undertake problem definition, formulation and solution assured through written examination and assignments
5. Ability to utilise a systems approach to design and operational performance not assured
6. Ability to function effectively as an individual in multi-disciplinary and multi-cultural teams, with the capacity to be a leader or manager as well as an effective team member not assured
7. Understanding of professional and ethical responsibilities and commitment to them emphasized in lectures but not assured
8. Expectation of the need to undertake lifelong learning and the capacity to do so assured through the requirement to undertake additional reading and literature searches to complete some assignments.

Assessment of Graduate Attributes

- Online Tutorials (addressing attributes 1, 3, 4, 5 & 8)
- Individual Written Assignments (addressing attributes 1, 2, 3, 4, 5, 7 & 8)
- Individual Written Examination (addressing attributes 1, 2, 3, 4, 5, 7 & 8)

Assumed Knowledge

- Good knowledge in the following is essential*:
  - Differential calculus and the application of this to system dynamics (differential equations)
  - Laplace transform and the inverse transform
  - Fourier transform and the inverse transform
  - Linear algebra, including matrix manipulation, matrix solutions, eigen-analysis and singular value decomposition
  - Poles (stable and unstable) and zeros (minimum phase and non-minimum phase)
  - Frequency response functions, especially the Bode diagram
  - Root locus
  - Block diagrams, their construction and interpretation
  - Feedback and feedforward control
  - Response characterisation such as settling time and overshoot

* Note: If you did not receive a GPA of 65% in subjects covering this material it is likely you will find this subject extremely challenging.
**Topic 1: Introduction to State Space Modelling**

- Objectives are and understanding of:
  - What is state space, the advantages and disadvantages over classical control
  - Definitions such as states (variables), state vector, state space, state trajectory
  - Structure of the state equation, and the role of the state matrix $A$, input matrix $B$, output matrix $C$, and direct transmission matrix $D$.
  - How to derive the state equations for simple mechanical systems and draw an equivalent block diagram

- Reading:
  - Dorf and Bishop "Modern Control Systems", Chapt 3
  - Franklin, Powell and Emami-Naeini "Feedback Control of Dynamic Systems", Chapt 2.2 (& 2.3-2.5), Chapt 7.1-7.2
  - Nise "Control Systems Engineering", Chapt 3
  - Kuo "Automatic Control Systems", 7th Edn, Chapt 5

**A Brief History**

- "Classical Control"
  - Engineering-based
  - Also known as Frequency Domain Control
- Russians had a different point of view
  - Mathematics-based
  - Also known as Modern control, or Time domain control or State-space control

- Main problems with classical:
  - MIMO
  - Controllability / Observability
  - Optimality
  - Time-varying systems
  - Non-linear systems

**State Space Models - Basic Characteristics**

- Models constructed in the TIME DOMAIN
  - work directly with the governing D.E.
  - vs. frequency domain models of Classical Control
- Models are all matrix-based
  - first order matrix differential equations

**State Space Models - Basic Characteristics**

- Some Advantages:
  - multiple input/output models now possible
  - possible to minimise "error criteria" (optimal control)
  - possible to examine stability (yes/no/why) in more depth
  - ideally suited to computer-based design and analysis
  - System does not need to be LTI or have zero initial conditions
- Some Disadvantages
  - difficult to examine robustness (stability margins)
  - more work than classical control for "simple" problems
  - "optimal" systems require "optimal" error criteria
State Basics

- State of a System:
  - A set of quantities which completely determine the evolution of the response of a system (in the absence of external inputs).

- State Variables:
  - Set of variables that define the state (x1, x2, ...).
  - For future ref, these are not unique.

- State Vector:
  - The (column) vector of the n state variables:
  - \[ x(t) = [x_1(t) \ x_2(t) \ ... \ x_n(t)] \]
  - Note: system is of order n (ie, it is described by an nth order D.E.)

Examples of Common States:

<table>
<thead>
<tr>
<th>Energy Quantity (storage element)</th>
<th>State Variable (physical variable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass (kinetic)</td>
<td>Velocity</td>
</tr>
<tr>
<td>Moment of Inertia (kinetic)</td>
<td>Rotational Velocity</td>
</tr>
<tr>
<td>Spring (potential)</td>
<td>Displacement</td>
</tr>
<tr>
<td>Fluid Compressibility</td>
<td>Pressure</td>
</tr>
<tr>
<td>Fluid Capacitance</td>
<td>Height</td>
</tr>
<tr>
<td>Thermal Capacitance</td>
<td>Temperature</td>
</tr>
</tbody>
</table>

State Basics

- State Space:
  - The n-dimensional space in which the components of the state vector are the coordinate axes.

- State Trajectory:
  - The path in state space produced by the state vector as it changes with time.

- Selection of State Variables:
  - State variables are not unique.
  - In the first instance, it is often reasonable to choose something with "physical meaning", often something associated with system "energy":

State Space Models

Generic System

inputs (control, disturbance)

"Internal" Motion (homogenous response)

outputs (measurements)
State Space Models

State Equations:

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx + Du
\end{align*}
\]

- **A** = \((n \times n)\) state matrix
  - describes "internal" (homogeneous) motion
- **B** = \((n \times r)\) input matrix
  - describes how \(r\) inputs affect \(n\) states.
- **C** = \((m \times n)\) output matrix
  - describes how \(n\) states contribute to \(m\) outputs
- **D** = \((m \times r)\) direct transmission matrix
  - describes how \(r\) inputs are fed through to \(m\) outputs

Sometimes the two equations written as:

\[
\begin{bmatrix}
x \\ y
\end{bmatrix} =
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
x \\ u
\end{bmatrix}
\]

Two Steps in Modeling for Control System Design:

1. Obtain the fundamental relationships
   - motion, heat transfer, fluid flow, etc.
2. Express the fundamental relationships in a "standard form" for control system design.
   - These steps are required regardless of the problem.
3. In Matlab, the command to build a state space system is `ss(A, B, C, D)`

Note Again:

- Once systems are reduced to a differential equation description, they are treated similarly.
  - Question: what does your controller actually “see”? What “is” a DE?
- Control System Design Techniques are basically efficient ways of working with differential equations.
- The Techniques let you quickly analyse how the governing equations are modified, and what happens.
Translational Example 1

Let plant be a double integrator:

\[
\begin{align*}
\text{Plant} & \quad 1/s & 1/s \\
\text{control input} & \quad u & \quad x_2 & \quad x_1 \\
\text{output} & \quad y
\end{align*}
\]
Translational Example 1 Cont.

Therefore, the state is output system. The state vectors are 00 and 10. Therefore, let the variables be input 1 and 0, be variables 2 and 3.

\[
\begin{align*}
\mathbf{x} &= \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\
\mathbf{y} &= \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\mathbf{u} &= \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\
\mathbf{v} &= \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\end{align*}
\]

Translational Example 2

\[
\begin{align*}
\dot{x} &= \mathbf{f}(\mathbf{x}, \mathbf{u}) \\
\mathbf{y} &= \mathbf{g}(\mathbf{x}, \mathbf{u})
\end{align*}
\]

Translational Example 2 Cont.

\[
\begin{align*}
\mathbf{F} &= \begin{bmatrix} F \\ k \\ d \end{bmatrix} \\
\mathbf{M} &= \begin{bmatrix} M \end{bmatrix}
\end{align*}
\]

Translational Example 2 Cont.
Translational Example 2 Cont.

Translational Example 2 Cont.

Translational Example 3

Translational Example 3 Cont.
Translational Example 3 Cont.

Translational Example 3 Cont.

Translational Example 3 Cont.

Translational Example 4

\[ M_1 \xrightarrow{u} y \xrightarrow{d} z \xrightarrow{M_2} \]

\[ k \]
Translational Example 4 Cont.

Drawing the block diagram

- Consider the following generic second order differential equation

\[
\begin{bmatrix}
  x_1 \\
  x_2
\end{bmatrix} = \begin{bmatrix}
  A_1 & A_2 \\
  A_3 & A_4
\end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2
\end{bmatrix} + \begin{bmatrix}
  B_1 \\
  B_2
\end{bmatrix} u + \begin{bmatrix}
  C_1 \\
  C_2
\end{bmatrix} w
\]

\[
y = \begin{bmatrix}
  C_1 & C_2
\end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2
\end{bmatrix} + v
\]

\[
x = A_1 x_1 + A_2 x_2 + B_1 u + G w
\]

\[
y = C_1 x_1 + C_2 x_2 + D u + v
\]
**Topic 2: Lagrangian Mechanics**

- Objectives are an understanding of:
  - What is the Langrangian method and the advantages of this compared to the Newton-Euler method
  - Be able to apply the method to derive the equations of motion for multi-body translational and rotational systems

- Reading:
  - Vu and Esfandiari "Dynamic Systems: Modeling and Analysis", Section 4.9, McGraw Hill
  - Hansen and Snyder "Active control of noise and vibration", Section 2.2.2.5, E&FN Spon
  - Active Structures Laboratory Course Notes “Chapter 1 Lagrangian dynamics of mechanical systems”
    
**Lagrangian Method**

- Problem with the Newton-Euler method for deriving the equations of motions for mechatronic systems is that one has to deal with vector forces and accelerations
  - This can be extremely difficult when bodies involve rotational and translation motion
- An alternative approach is the Lagrangian method which involved only scalar energy terms

**Lagrange’s Equations: Generalised Coordinates**

- First define a set of *generalised coordinates* \( q_1, q_2, \ldots, q_n \)
  - to represent an \( n \)-degree-of-freedom system
  - Typically position coordinates (distances and/or angles)
- It must be possible to describe the total energy of the system (kinetic and potential) in terms of the generalised coordinates and their derivatives

**Lagrange’s Equations: Energy**

- In terms of the generalised coordinates (and derivatives), define the *Kinetic Energy*, \( T \), and the *Potential Energy*, \( V \)
  - In general, the kinetic energy will be a positive definite function of the generalised coordinates and their derivatives, whereas the potential energy is only a function of the generalised coordinates, ie \( T(q_1, q_2, \ldots, q_n, \dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n) \), \( V(q_1, q_2, \ldots, q_n) \)
  - Potential energies are the sum of elastic potential energies and gravitational potential energies
  - Kinetic energies will be the sum of translation and rotational kinetic energies
- In multi-body systems, the kinetic and potential energy can be computed for each body individually and then simply added together
Lagrange's Equations: Lagrangian

- Calculate the Lagrangian, which is defined as the difference between the kinetic and potential energies
  \[ L(q_1, q_2, ..., q_n, \dot{q}_1, \dot{q}_2, ..., \dot{q}_n) = T(q_1, q_2, ..., q_n, \dot{q}_1, \dot{q}_2, ..., \dot{q}_n) - V(q_1, q_2, ..., q_n) \]

- The equations of motion are expressed in terms of the Lagrangian of the following form (known as Lagrange's Equations):
  \[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = \frac{\partial T}{\partial \dot{q}_i} - \frac{\partial V}{\partial q_i} + \tau_i, \quad i = 1, 2, ..., n \]

  where \( \tau_i \) represents the generalised force (non-conservative force or torque) in the \( q_i \) direction. Non-conservative forces are forces not derivable from the Lagrangian.
  - These include external applied forces (or torques) and frictional forces (or torques).

Example 1

- Derive the DE for the system considered previously using the Lagrangian method
Example 2

Derive the DE for the system considered previously using the Lagrangian method. First consider the system in the absence of gravity, then account for gravity.
Example 3

- Derive the DEs for the system considered previously using the Lagrangian method.
Example 4

• Derive the DE for the simple pendulum system shown across for an applied torque $\tau$.
Example 4 Cont.

Example 4 Cont.

Example 5

• Derive the DEs for the reaction wheel pendulum system shown across for a torque \(\tau\) applied to the rotor.

Example 5 Cont.
**Topic 3: Linearisation of Non-linear Differential Equations**

- Objectives are an understanding of:
  - What a non-linear equation is
  - The need for linearisation
  - Linearisation techniques including a graphical approach, a Taylor series expansion and a Jacobian approach for MIMO systems
  - To be able to apply the above techniques to linearise non-linear algebraic and differential equations

- Reading:
  - Dorf and Bishop "Modern Control Systems", Section 2.3
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems", Section 2.6
  - Nise "Control Systems Engineering", Section 2.11
  - Close and Frederick "Modeling and Analysis of Dynamic Systems", Chapter 9

**Introduction**

- In practice many systems are inherently non-linear
- To work with these non-linear systems we have three choices:
  1. Attempt to solve the differential equations directly (constructive non-linear control).
  2. V. hard work. Difficult to find explicit solution
  3. Convergence for guaranteed
  4. Obtain computer solutions of the non-linear response for specific inputs, eg Simulink
  5. Derive a fixed linear approximation. We will cover three methods of linearisation:
     1. Graphical Approach
     2. Taylor Series Expansion
     3. Linearisation via Jacobian matrices: for multi-variable systems

**Graphical Approach**

- Consider a non-linear spring, with force displacement curve shown below
  ![Graphical Approach Diagram](image)
  - Operating point or nominal value: \( \bar{f} = f(\bar{x}) \)
  - Define \( x(t) = x + \dot{x}(t) \) where \( \dot{x}(t) \) is the incremental variable, and
  - Force is given by \( f(t) = \bar{f} + \dot{f}(t) \)

**Graphical Approach**

- The slope of the tangent is \( k = \frac{df}{dx} \bigg|_{\bar{x}} \)
- The linear approximation is thus \( f = \bar{f} + k(x - \bar{x}) \)
- May be rewritten as \( f - \bar{f} = k(x - \bar{x}) \)
- In terms of incremental variables we have \( \ddot{x} = x - \bar{x} \) & \( \dot{f} = f - \bar{f} \)
- Thus, the linearised expression is \( \dot{f} = k\ddot{x} \)
Taylor Series Expansion

- We can express the non-linear function in terms of its Taylor series expansion
  
  \[ f(x) = f(\bar{x}) + \frac{df}{dx}_{x=\bar{x}} (x - \bar{x}) + \frac{1}{2!} \frac{d^2f}{dx^2}_{x=\bar{x}} (x - \bar{x})^2 + \cdots \]

- A linear approximation is thus
  
  \[ f(x) \approx f(\bar{x}) + \frac{df}{dx}_{x=\bar{x}} (x - \bar{x}) \]

- Or, in terms of incremental variables
  
  \[ \hat{f} = f(x) - f(\bar{x}) = \frac{df}{dx}_{x=\bar{x}} (x - \bar{x}) = \frac{df}{dx}_{x=\bar{x}} \hat{x} \]

Example: Cubic spring

- Consider the non-linear spring given by \( f(x) = x^3 \). Linearise the equation about the nominal point of \( x = 1 \).

Example: Cubic spring cont.

Example: Cubic spring and mass

- Consider the previous spring connected to a mass under the action of gravity
Example: Cubic spring and mass

Jacobian Linearisation

- In many situations, the plant is non-linear with respect to many variables, not just a single variable as in the previous cases
- *Jacobian linearisation* provides a systematic means of approaching the multi-variable linearisation problem

Jacobian Linearisation of State Equation

- Consider the non-linear equation $\dot{x} = f(x, t, u)$
  - where $x \in \mathbb{R}^N$ and $u \in \mathbb{R}^M$
  - and nominal & incremental states and inputs respectively
    - $x = \bar{x} + \delta x$, $u = \bar{u} + \delta u$
- Now the Taylor series expansion for each row (equation) of the function is given by
  $$
  \dot{\bar{x}}_i = f_i(\bar{x}, t, \bar{u}) + \sum_{j=1}^N \frac{\partial f_i}{\partial \bar{x}_j} \dot{\bar{x}}_j + \sum_{k=1}^M \frac{\partial f_i}{\partial \bar{u}_k} \dot{\bar{u}}_k + \text{higher order terms}
  $$

Jacobian Linearisation of State Equation

- Thus $\frac{d}{dt} \tilde{x}(t) \approx A(t)\tilde{x}(t) + B(t)\tilde{u}(t)$
- where the state matrix $A(t) \in \mathbb{R}^{N\times N}$, such that $A_y = \frac{\partial f_i}{\partial \bar{x}_j}|_{x, t, \bar{u}}$
- and the input matrix is $B(t) \in \mathbb{R}^{N\times M}$, such that $B_k = \frac{\partial f_i}{\partial \bar{u}_k}|_{x, t, \bar{x}}$
Jacobian Linearisation of Output Equation

- Similarly, the process can be repeated for non-linear outputs
- Consider the non-linear output equation $y = g(x, t, u)$
  - where $y \in \mathbb{R}^p$
  - and nominal & incremental outputs
    $$y = \tilde{y} + \hat{y}$$
- Now the Taylor series expansion for each row (equation) of the output function is given by
  $$y_i = g_i(x) = g_i(x, t, u) + \sum_{j=1}^{N} \frac{\partial g_i}{\partial x_j} \hat{x}_j + \sum_{k=1}^{M} \frac{\partial g_i}{\partial u_k} \hat{u}_k + \text{higher order terms}$$

Example: Linearisation of pendulum dynamics

- Consider the pendulum with applied torque $\tau$

Jacobian Linearisation of Output Equation

- The output equation is thus
  $$\dot{y}(t) \approx C(t)\dot{x}(t) + D(t)\dot{u}(t)$$
- where the output matrix is
  $C(t) \in \mathbb{R}^{p \times n}$, such that $C_{ij} = \frac{\partial g_i}{\partial x_j}$
- and the direct transmission matrix is
  $D(t) \in \mathbb{R}^{p \times m}$, such that $D_{ik} = \frac{\partial g_i}{\partial u_k}$

Example: Linearisation of pendulum dynamics
Example: Linearisation of pendulum dynamics
**Topic 4: More State Space Modelling**

- Objectives are an understanding of:
  - The derivation of the equations of motion for several different multiphysics problems including mechanical, electrical, thermal, hydraulic, and coupled systems
  - To write the equations of motion in state space form (linearising if necessary)

- Reading:
  - Dorf and Bishop "Modern Control Systems", Chapt 3
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems”, Chapt 2.2, Chapt 7.1-7.2
  - Nise "Control Systems Engineering", Chapt 3, 4.11, 5.7-5.8

---

**Rotational Mechanical Example 1**

- Derive the governing DE for the following single dof rotational system and write in state form

![Rotational Mechanical Example 1 Diagram]

---

**Rotational Mechanical Example 2**

- Derive the governing DE for the following 2 dof rotational system and write in state form

![Rotational Mechanical Example 2 Diagram]
Coupled Translational-Rotational Example

- Derive the governing DE for the following coupled translational-rotational system and write in state form.
Coupled Translational-Rotational Example

Coupled Translational-Rotational Example

Coupled Translational-Rotational Example

Coupled Translational-Rotational Example
Coupled Translational-Rotational Example
Electrical Example 1

• Derive the governing DE for the following 2\textsuperscript{nd} order electrical system and write in state form
**Electro-Mechanical System: Permanent Magnet DC Motor**

- Derive the governing DE for the following electro-mechanical system (dc motor) and write in state form.
Electro-Mechanical System

Hydraulic Example

- Derive the governing DE for the following 1st order hydraulic system and write in state form
Hydraulic Example

Thermal Example

- Derive the governing DE for the following 2\textsuperscript{nd} order thermal system and write in state form
Loudspeaker Example

- Consider the loudspeaker shown across. Derive the state equations in terms applied voltage to coil.
**Topic 5: Modelling Multiple DOF Systems**

- **Objectives are an understanding of:**
  - How one writes multi-dof systems in state space form including diagonal canonical form and modal/Jordan form
  - The fact that the state equations are not unique
  - The concept of eigenvectors (modes) and eigenvalues (natural frequencies)
  - Be able to apply these techniques to multi-dof system analysis

- **Reading:**
  - Dorf and Bishop "Modern Control Systems", Section 3.5
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems", Section 7.2.1
  - Nise "Control Systems Engineering", Section 5.7-5.8
  - Inman, "Engineering Vibration", Chapt 4 – Multiple degree of freedom systems

---

**Eigenvalues and Eigenvectors**

- Recall the eigen-analysis from maths
- **Eigenvalue equation for \( \mathbf{A} \):**
  - \( \det (\lambda I - \mathbf{A}) = 0 \)
  - Note that the eigenvalues, \( \lambda \), are also called poles
  - For second order systems
    - The magnitude of which is the natural frequency
    - The real part is equal to the product of the damping ratio and natural frequency
    - The imaginary part is equal to the resonance frequency
  - Eigenvectors (mode shapes) are obtained by solving for null space of
    - \( \lambda \mathbf{v} = \mathbf{Av} \)

---

**Uniqueness and the Similarity Transform**

Consider the state equations given by
\[
\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} \\
\mathbf{y} = \mathbf{Cx} + \mathbf{Du}
\]

Now, let us do the following
\[
\dot{\mathbf{x}} = \mathbf{AT}^{-1}\mathbf{x} + \mathbf{Bu}
\]
Premultiplying by \( \mathbf{T}^{-1} \) gives
\[
\mathbf{T}^{-1}\dot{\mathbf{x}} = \mathbf{T}^{-1}\mathbf{AT}^{-1}\mathbf{x} + \mathbf{T}^{-1}\mathbf{Bu}
\]
Define the following
\[
\dot{\mathbf{x}} = \tilde{\mathbf{x}} \quad \tilde{\mathbf{A}} = \mathbf{T}^{-1}\mathbf{AT}^{-1} \quad \tilde{\mathbf{B}} = \mathbf{T}^{-1}\mathbf{B} \quad \tilde{\mathbf{C}} = \mathbf{CT}
\]
We obtain the transformed state equation
\[
\dot{\mathbf{x}} = \tilde{\mathbf{A}}\mathbf{x} + \mathbf{Bu} \\
\mathbf{y} = \tilde{\mathbf{C}}\mathbf{x} + \mathbf{Du}
\]

---

**Example: 2 DOF Mass/Spring system**

- Consider the symmetric 2 dof system below. Derive the state equations, then using this determine the natural frequencies and mode shapes of the system.
Example: 2 DOF Mass/Spring system

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0
\end{bmatrix}
\]

Example: 2 DOF Mass/Spring system

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0
\end{bmatrix}
\]

Example: 2 DOF Mass/Spring system

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0
\end{bmatrix}
\]

Example: 2 DOF Mass/Spring system

\[
\begin{bmatrix}
A & B \\
C & D
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0
\end{bmatrix}
\]
### Diagonal Canonical Form

- We can use the eigenvectors to transpose the original state representation into a "diagonal canonical form" using

\[
\dot{x} = \tilde{A}x + \tilde{B}u \\
y = \tilde{C} \dot{x} + Du
\]

where

\[
\tilde{x} = T^{-1}x, \quad \tilde{A} = T^{-1}AT, \quad \tilde{B} = T^{-1}B, \quad \tilde{C} = CT
\]

and

\[
T = [v_1, v_2, \ldots, v_n] & \quad \tilde{A} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)
\]

- Note that this form contains complex elements in the matrices which are difficult to work with in practice.

### Jordan Canonical Form: Diagonal + Skew-Symmetric

- An alternative to the diagonal canonical form is the Jordan form which groups the modes into [2x2] parallel systems

\[
\tilde{A} = \begin{bmatrix}
Re(\lambda_i) & Im(\lambda_i) \\
-Im(\lambda_i) & Re(\lambda_i)
\end{bmatrix}
\]

An example of Jordan Block is:

\[
\begin{bmatrix}
-a \lambda & a \lambda \\
-a \lambda & a \lambda
\end{bmatrix}
\]

- Note that these matrices are entirely real.
- This form is also known as the modal form.
- Matlab command is:
  > csys = canon(sys,'modal')
  > [V2,D2] = cdf2rdf(V,D)

### Example: 2 DOF Mass/Spring system

- Consider the previous double mass system. Calculate the eigenvalues and eigenvectors and then write the state equations in diagonal and modal form.

\[
T = [Re(v_i), Im(v_i), Re(v_j), Im(v_j)]
\]

- Note that in this form, each mode in the state matrix is expressed as a second order system with a natural frequency and damping ratio.
Example: 2 DOF Mass/Spring system
Example: 2 DOF Mass/Spring system
Example: 2 DOF Mass/Spring system

\[ \begin{pmatrix} -k & 0 \\ 0 & -k \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \end{pmatrix} + \begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix} = 0 \]

Eqn (1) gives the eigenvalues...
Topic 6: Modelling Distributed Parameter Systems

Objectives are an understanding of:

- What a distributed parameter system is
- How this differs from a lumped parameter model
- The concept of modes, natural frequencies, damping ratio, modal mass, modal amplitude and modal force
- Different ways in which the state equation for modal systems can be expressed
- To be able to derive the state equations for a distributed parameter system

Reading:
- Inman, "Engineering Vibration", Chapt 6 – Distributed parameter systems
- Hatch, "Vibration simulation using Matlab and Ansys", The Whole Book
- Junkins and Kim, "Introduction to Dynamics and Control of Flexible Structures", AIAA Educational Series, Chapt 4

Multiple DOF and Continuous Systems

- Mechanical engineers are often interested in systems with second order response characteristics (i.e., things that move).
- These may be broadly classified as either
  - Lumped parameter (discrete) systems – lumped masses linked by springs for example
  - Distributed parameter (continuous) systems – beams and plates for example

Distributed or Lumped?

- Consider a tapered bar clamped at one end and pulled with load P at the other...

Distributed Vs Lumped parameter model

- Spatial variation
- ODE/PDE
- Example: Temperature distribution

\[
\begin{align*}
T(x, y, z) &= f(x, y, z) \\
\frac{\partial T}{\partial t} + \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} &= 0
\end{align*}
\]
Distributed Parameter System

- Consider a flexible structure, which is a distributed parameter system (vs. lumped parameter system).
- Response of a structural distributed parameter system:
  \[ m(r) \frac{\partial^2 w(r,t)}{\partial t^2} + d \frac{\partial w(r,t)}{\partial t} + k w(r,t) = f(r,t) \]
- Dynamics are the same as for discrete systems

Example: Modal response of a string

- For a structural dynamics problem, the standard way to model the “low” frequency response is in terms of modes.
- The response at any point is given by the sum of all mode shape functions (at that point) multiplied by the amplitude of the mode, ie
  \[ w(r,t) = \sum_{i=1}^{\infty} z_i(t) \varphi_i(r) \]
  where
  \[ w(r,t) = \text{response at location } r \text{ at time } t \]
  \[ z_i(t) = \text{amplitude of mode } i \text{ at time } t \]
  \[ \varphi_i(r) = \text{value of mode shape } i \text{ at location } r \]
- Each mode behaves like an independent second order system

Distributed Parameter System

- Consider a single mode. Response is standard second order:
  \[ \ddot{z}_i(t) + 2 \zeta_i \omega_i \dot{z}_i(t) + \omega_i^2 z_i(t) = f_i(t) / m_i \]
  - where \( \zeta_i \) is the damping ratio of mode \( i \)
  - where \( \omega_i \) is the natural frequency of mode \( i \)
  - where \( f_i \) is the “modal force”, and represents the effective force into mode \( i \)
  - where \( m_i = \int A(r) \rho(r) \varphi_i^2(r) dr \) is the “modal mass”, and represents the effective mass of mode \( i \)
- Second order - 2 states required
  - displacement and velocity of mode
Distributed Parameter System

- State equation for a single mode is

\[ \dot{x} = Ax + Bu \]
where

\[
\begin{bmatrix}
  x
  \\
  \dot{x}
\end{bmatrix} =
\begin{bmatrix}
  z_i \\
  \dot{z}_i
\end{bmatrix},
A =
\begin{bmatrix}
  0 & 1 \\
  -\omega_i^2 & -2\zeta_i\omega_i
\end{bmatrix},
B =
\begin{bmatrix}
  0 \\
  \psi_i(r_f)/m_i
\end{bmatrix}
\]

\( u = f \)

\( z_i \) = displacement of mode \( i \)
\( \dot{z}_i \) = velocity of mode \( i \)
\( f \) = applied force at \( r_f \)
\( f_i \) = generalised force (input into mode \( i \)) = \( \psi_i(r_f)f \)
\( m_i \) = modal mass of mode \( i \)

\[ y = Cx + Du \]

173

Distributed Parameter System

- Alternative state equation for a single mode:

\[ \dot{x} = Ax + Bu \]
where

\[
\begin{bmatrix}
  x
  \\
  \dot{x}
\end{bmatrix} =
\begin{bmatrix}
  z_i \\
  \dot{z}_i
\end{bmatrix},
A =
\begin{bmatrix}
  0 & \omega_i \\
  -\omega_i & -2\zeta_i\omega_i
\end{bmatrix},
B =
\begin{bmatrix}
  0 \\
  \psi_i(r_f)/m_i
\end{bmatrix}
\]

\( u = f \)

\( z_i \) = displacement of mode \( i \)
\( \dot{z}_i \) = velocity of mode \( i \)
\( f \) = applied force at location \( r_f \)
\( f_i \) = generalised force (input into mode \( i \)) = \( \psi_i(r_f)f \)
\( m_i \) = modal mass of mode \( i \)

\[ y = C\begin{bmatrix} 1/\omega_i & 0 \\ 0 & 1 \end{bmatrix}x + Du \]

A matrix is better conditioned in this form

174

Distributed Parameter System

- Input Expression

\[ \text{Actuator} \quad \rightarrow \quad \text{Structure} \]
\[ \text{Mode shape } i \]

Suppose the actuator provides an input force, and that when the force input is 1N the modal force input is \( f_f \). Then the input is given by,

\[
\begin{bmatrix}
  f_f \\
  m_i
\end{bmatrix} =
\begin{bmatrix}
  0 \\
  \psi_i(r_f)/m_i
\end{bmatrix}
\]

where \( r_f \) is the location of the applied force

175

Distributed Parameter System

- Output Expression

\[ \text{Sensor} \quad \rightarrow \quad \text{Structure} \]
\[ \text{Mode shape } i \]

Suppose the sensor measures displacement, and that when the amplitude of mode \( i \) is 1, the output is \( c_i \). Then,

\[
y = \left[ \begin{array}{c} c_i \\ \psi_i(r_s)/m_i \end{array} \right] + \frac{0}{Du}f
\]

where \( c_i = \psi_i(r_s) \), and \( r_s \) is the location of the sensor

176
Distributed Parameter System: Multiple Modes

\[
\begin{bmatrix}
\dot{z}_1 \\
\dot{z}_2 \\
\vdots \\
\dot{z}_n
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
-\omega_n^2 - 2\zeta \omega_n & 0 & 0 & \cdots & 0 \\
0 & -\omega_n^2 - 2\zeta \omega_n & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & -\omega_n^2 - 2\zeta \omega_n
\end{bmatrix}
\begin{bmatrix}
z_1 \\
z_2 \\
\vdots \\
z_n
\end{bmatrix} + \begin{bmatrix}
0 \\
\psi_1 / m_1 \\
0 \\
\vdots \\
0 \\
\psi_n / m_n
\end{bmatrix} \cdot f
\]

\[y = \begin{bmatrix} c_1 & 0 & \cdots & 0 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\
z_2 \\
\vdots \\
z_n
\end{bmatrix} + [0] \cdot f\]

Example: First two modes of a simply supported beam

Consider the simply supported Euler-Bernoulli Beam below. Derive the state equations for the system by considering the first two modes, given that the force input is a 30% along the beam and the displacement sensor is 40% along the beam. Also assume

- \( L = 1.5 \text{m}, b = 0.150 \text{m}, h = 0.006 \text{m} \)
- \( E = 209 \text{GPa} \)
- \( \rho = 7800 \text{kg/m}^3 \)
- \( \zeta = 10\% \)
Example: First two modes of a simply supported beam
Example: First two modes of a simply supported beam
**Topic 7: Conversion between SS to TF and back again**

- Objectives are an understanding of:
  - How to convert a state space equation into the classical transfer function
  - How to convert a transfer function into one of the "Standard" state forms such as control canonical, observer canonical, diagonal and Jordan forms, balanced realisation

- Reading:
  - Dorf and Bishop "Modern Control Systems", Chapt 3.5 & 3.6
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems”, Chapt 7.2.1, 7.2.2, 7.5.1
  - Nise "Control Systems Engineering", Chapt 3.5, 3.6, 5.7-5.8
  - Kuo, "Automatic Control Systems", Chapt 5.5 – 5.9

---

**Transfer Functions**

- The short hand notation for the state-space representation is:
  \[
  \begin{bmatrix}
  \dot{x} \\
  y
  \end{bmatrix} = \begin{bmatrix}
  A & B \\
  C & D
  \end{bmatrix} \begin{bmatrix}
  x \\
  u
  \end{bmatrix}
  \]

- The equivalent for the transfer function is:
  \[
  G(s) = \begin{bmatrix}
  A & B \\
  C & D
  \end{bmatrix} = C(sI - A)^{-1}B + D
  \]

- The Matlab command is:
  - `tf(plant)`
  - `zpk(plant)`

---

**Convert SS to TF Models**

- Why?
  - Because it is easier / better to assess some things using classical techniques, such as gain and phase margin.

- How?
  - Standard transfer function formulation procedure:
    - 1. Laplace transform of output
    - 2. Laplace transform of input
    - 3. Transfer function = \( \mathcal{L}(\text{output}) / \mathcal{L}(\text{input}) \)

---

**Convert SS to TF Models**

- Output:
  - \( y = Cx + Du \)
  - \( \mathcal{L}(\text{output}): \ y(s) = Cx(s) + Du(s) \)

- Input:
  - \( \dot{x} = Ax + Bu \)
  - \( \mathcal{L}(\text{input}): \ (sI-A)x(s) = Bu(s) \)

- Transfer function:
  - \[ G(s) = \frac{y(s)}{u(s)} = C(sI-A)^{-1}B + D \]
Example: State-Space to Transfer function

- Convert the state and output equations below to a transfer function

\[
\begin{bmatrix}
-4 & -1.5 \\
4 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
+ \begin{bmatrix}
2
\end{bmatrix}u(t)
\]

\[
y = \begin{bmatrix}
1.5 & 0.625
\end{bmatrix}x
\]

Example: State-Space to Transfer function cont.

Control System Design Methodology

- What are we doing?
  - Modeling for control system design

- What do we have so far?
  - Statement of the fundamental relationship
    - part 1 of two part process

- What do we need to do?
  - Express the fundamental relationship in a “standard form”
    - part 2 of the process

Convert TF to SS Models

- Why?
  - Usually, because original model was in the form of a transfer function.

- Unique-ness
  - State Space models are not unique. Therefore, there are several possible state space models for a given transfer function. These are classified according to their form (how the matrices in the state equations appear). Common forms include:
    - Control canonical - controllability
    - Observer canonical - observability
    - Jordan / Modal / Block Diagonal - understanding modes
    - Balanced - for model truncation
**Controllability / Observability**

- **Controllable if**
  - State variable(s) can be moved from initial location to desired final location, in finite time, using a input
  - Matlab command: `ct_mat=ctrb(sys.a,sys.b)`
  - `rank(ct_mat)==rank(sys.a)`..Controllable!

- **Observable if**
  - State variable(s) can be determined using input and output measured over finite time
  - Matlab command: `ob_mat=obsv(sys.a,sys.c)`
  - `rank(ob_mat)==rank(sys.a)`..Observable!

---

**Which TF can be converted?**

- Transfer function which is **strictly proper** can be easily transferred into state space control canonical form
  - Degree of numerator < Degree of denominator

- Transfer function which is **proper** can also be transferred
  - Degree of numerator ≤ Degree of denominator
  - Separate out the strictly proper TF from the constant TF
  - \( G(s) = G_{SP}(s) + G(\infty) \)

---

**Model 1: Control Canonical Form (Upper Companion)**

\[
G(s) = y(s) = \frac{b_n s^{n-1} + b_{n-1} s^{n-2} + \ldots + b_2 s + b_1}{s^n + a_{n-1} s^{n-1} + \ldots + a_2 s + a_1} u(s)
\]

Let \( y/u = (y/z) \times (z/u) \), where:

\[
\frac{z}{u} = \frac{1}{s^n + a_{n-1} s^{n-1} + \ldots + a_2 s + a_1}
\]

\[
\frac{y}{z} = b_n s^{n-1} + b_{n-1} s^{n-2} + \ldots + b_2 s + b_1
\]

To derive control canonical form, re-express the above as differential equations.

---

**Model 1: Control Canonical Form (Upper Companion)**

\[
A = \begin{bmatrix}
-a_1 & -a_2 & \cdots & -a_{n-1} & -a_n \\
1 & 0 & \cdots & 0 & 0 \\
0 & 1 & \cdots & 0 & 0 \\
0 & 0 & \cdots & 1 & 0 \\
0 & 0 & \cdots & 0 & 1 \\
\end{bmatrix}
\]

\[
B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ \vdots \end{bmatrix}
\]

\[
C = [b_1 \ b_2 \ \cdots \ b_{n-1} \ b_n], \ D = 0
\]

- **Note:**
  - (i) The characteristic equation is in the upper row
  - (ii) When state variables are the dependent variable and successive derivatives of dependent variable, then they are called phase variables.
Model 1: Control Canonical Form (Upper Companion)

- Note: that all (feedback from) states connect to the control input

Example: Control Canonical (Upper Companion)

- Write the following transfer function in upper control canonical form:
  \[ G(s) = \frac{s+4}{(s+1)(s+2)(s+5)} \]

Example: Control Canonical (Upper Companion) cont.

Model 1: Control Canonical Form (Lower Companion)

- Note:
  1. The characteristic equation is in the lower row
  2. When state variables are the dependent variable and successive derivatives of dependent variable, then they are called phase variables.
**Example: Control Canonical (Lower Companion)**

- Write the following transfer function in lower control canonical form:
  \[ G(s) = \frac{(s+4)}{[(s+1)(s+2)(s+5)]} \]

**Model 2: Observer Canonical Form (Left Companion)**

- Rearranging the previous expression gives the following DE:
  \[ y^n(t) + a_n y^{n-1}(t) + \ldots + a_2 y(t) + a_1 y(t) = b_1 u^{n-1}(t) + \ldots + b_n u(t) \]

- Rearranging gives
  \[ y^n = (-a_2 y^{n-1} + b_1 u^{n-1}) + \ldots + (-a_n y' + b_n u') + (-a_n y + b_n u) \]

- Note that the characteristic equation is in the left hand column

\[ G(s) = y(s) = \frac{b_1 s^{n-1} + b_2 s^{n-2} + \ldots + b_{n-1} s + b_n}{s^n + a_1 s^{n-1} + \ldots + a_{n-1} s + a_n} \]

To derive the form graphically, note that:

- \[ b_n u - a_n y = (y \text{ terms}) - (u \text{ terms}) \]

\[ x = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_{n-1} \\ z_n \end{bmatrix}, \quad A = \begin{bmatrix} -a_1 & 1 & \cdots & 0 & 0 \\ -a_2 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -a_{n-1} & 0 & \cdots & 0 & 1 \\ -a_n & 0 & \cdots & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \end{bmatrix}, \quad D = 0 \]
Model 2: Observer Canonical Form (Left Companion)

System, in observer canonical form

*** note that feedback to states is from the output

Example: Observer Canonical (Left Companion) cont.

Example: Observer Canonical (Left Companion)

• Write the following transfer function in left observer canonical form:
  \[ G(s) = \frac{s+4}{(s+1)(s+2)(s+5)} \]

Model 2: Observer Canonical Form (Right Companion)

\[
\begin{bmatrix}
  z_1 \\
  z_2 \\
  \vdots \\
  z_{n-1} \\
  z_n
\end{bmatrix} =
\begin{bmatrix}
  0 & 0 & \cdots & 0 & -a_n \\
  1 & 0 & \cdots & 0 & -a_{n-1} \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  0 & 0 & \cdots & 0 & -a_2 \\
  0 & 0 & \cdots & 1 & -a_1
\end{bmatrix}
\begin{bmatrix}
  x \n\end{bmatrix} +
\begin{bmatrix}
  b_n \\
  b_{n-1} \\
  \vdots \\
  b_2 \\
  b_1
\end{bmatrix}
\]

\[ C = [0 \ 0 \ \cdots \ 0 \ 1], \ D = 0 \]

• Note that the characteristic equation is in the right hand column
Example: Observer Canonical (Right Companion)

- Write the following transfer function in right observer canonical form:
  \[ G(s) = \frac{s+4}{(s+1)(s+2)(s+5)} \]

Model 3: Jordan Form (block diagonal, modal or parallel form)

- Diagonal is the (complex) eigenvalues (poles) of the system
- Jordan/Modal Form - diagonal is (1x1) or (2x2) blocks of real modal state matrices
- Eg
  \[
  A = \begin{bmatrix}
  \lambda_1 & 1 & 0 & 0 & 0 \\
  0 & \lambda_1 & 1 & 0 & 0 \\
  0 & 0 & \lambda_1 & 0 & 0 \\
  0 & 0 & 0 & \lambda_2 & 0 \\
  0 & 0 & 0 & 0 & \lambda_3 \\
  \end{bmatrix}
  \]

  where \( \lambda_1 \) is a third order eigenvalue, and distinct eigenvalues \( \lambda_2 \) and \( \lambda_3 \).
- The Matlab command is `canon(plant, 'modal')`

Model 3: Jordan Form (block diagonal, modal or parallel form)

Suppose that the transfer function was split into partial fractions:

\[\text{eg, } G(s) = \frac{1}{s} + \frac{2s + 3}{s^2 + 4s + 2} + \ldots\]

Then you have a lot of parallel states, with:

\[
A = \begin{bmatrix}
[A_1] & 0 & 0 & \vdots \\
0 & [A_2] & 0 & \vdots \\
0 & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & [A_n]
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
B_1 \\
B_2 \\
\vdots \\
B_n
\end{bmatrix}
\]

C = [C_1 C_2 ... C_n]

Example: Diagonal Canonical Form

- Write the following transfer function in diagonal canonical form:
  \[ G(s) = \frac{s+4}{(s+1)(s+2)(s+5)} \]
Example: Diagonal Canonical Form
Example: Diagonal Canonical Form

Model 4: Balanced Realisation

- **Asymptotically stable** if system has Eigenvalues with negative real part
  - An asymptotically stable system will not "blow up" / give an unbounded output for fixed-finite input
  - E.g. step input results in decaying oscillations in the output and the output will tend to settle, asymptotically, to a final steady-state value

- **Minimal realisation** defines the system with minimum number of states
- **Controllability Gramian** predicts controllability
- **Observability Gramian** predicts observability

Model 4: Balanced Realisation Definitions

- A **balanced realisation** is an asymptotically stable minimal realisation in which the controllability (P) and observability (Q) Gramians are equal and diagonal

The system \((A,B,C,D)\) is balanced if the solutions to the Lyapunov equations:

\[ AP + PA^T + BB^T = 0 \quad \text{and} \quad A^TQ + QA + C^TC = 0 \]

are \(P=Q=\text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n) = \Sigma\), where \(P\) and \(Q\) are the controllability and observability Gramians (see later)

Therefore \(\sigma_i = \sqrt{\text{eig}_i(PQ)}\)

Model 4: Balanced Realisation

- In a balanced system, each state is as controllable as it is observable.
- In a balanced realisation, the size of \(\sigma_i\) is a relative measure of the contribution state \(x_i\) makes to the input-output of the system
  - E.g., if \(\sigma_i >> \sigma_j\), then the state \(x_i\) affects the input-output behaviour much more than \(x_j\).
- Commonly used to determine which states are unimportant, allowing model truncation
- In Matlab the command is \([\text{sysb},g]=\text{balreal}(\text{sys})\)
  - \(\text{sysb}\) is the equivalent realization for which the diagonal entries of the controllability and observability Gramians form the vector 'g'. Small entries in 'g' indicates the states that can be removed to truncate the model
Example: Balanced Realisation

- Write the following transfer function in balanced form:
  \[ G(s) = \frac{s+4}{(s+1)(s+2)(s+5)} \]
Solution to State Equations

• Homogeneous State Equations: \( \dot{x} = Ax \)
  \[ x(t) = e^{At}x(0) \]

• Note: \( e^{At} \)
  \( \rightarrow \) is a matrix exponential
  \( \rightarrow \) is also known as the state transition matrix
  \( \rightarrow \) is sometimes labelled as \( \phi(t) = e^{At} \)
  \( \rightarrow \) In Matlab \( \phi = \expm(A\cdot t) \)
  \( \rightarrow \) is difficult to calculate. Two common ways:
  \( \rightarrow \) Expansion:
    \[ \phi(t) = e^{At} = I + At + \frac{A^2t^2}{2!} + \ldots + \frac{A^kt^k}{k!} + \ldots \]
  \( \rightarrow \) Inversion:
    \[ \phi(t) = L^{-1}\{\Phi(s)\} = L^{-1}\{(sI - A)^{-1}\} \]

Solution to State Equations

• The forced response (in-homogeneous solution) is given by (assuming \( t_0 = 0 \)):
  \[ x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \]

• This expression is known as the state transition equation

• The output is therefore given by:
  \[ y(t) = Cx(t) + Du(t) \]

• The Matlab command is \texttt{lsim(plant,U,T)}
Example: Stationary Object

- Consider a stationary object with initial position at 5m. Determine the temporal response of the object.

Example: Object with constant velocity

- Consider an object with initial position at 5m and initial velocity of 2m/s. Determine the temporal response of the object.
Example: Object with constant acceleration

- Consider an object with initial position at 5m and initial velocity of 2m/s under the influence of a gravitational acceleration of -10m/s². Determine the temporal response of the object.

Example: Homogenous response of a 2nd order system

- Determine the homogenous response of a mass $m = 2kg$ attached to a spring with stiffness $k = 100N/m$ and dashpot with damping coefficient of $c = 1Ns/m$. 
Example: Homogenous response of a 2\textsuperscript{nd} order system
Example: Homogenous response of a 2\textsuperscript{nd} order system
Recap: Converting SS to TF Models

Output: \[ y = Cx + Du \]
\[ \mathcal{L}(\text{output}): \quad y(s) = Cx(s) + Du(s) \]
Input: \[ \dot{x} = Ax + Bu \]
\[ \mathcal{L}(\text{input}): \quad (sI-A)x(s) = Bu(s) \]

Transfer function:
\[ G(s) = \frac{y(s)}{u(s)} = C(sI-A)^{-1}B + D \]

Relationship Between Poles and Eigenvalues

- Important Conclusion from SS to TF Conversion Exercise
  - Poles of the SS system defined by: \( \det (sI-A) = 0 \)

- Eigenvalue equation for \( A \):
  - \( \det (s I - A) = 0 \)

- Poles defined by:
  - \( \det (sI - A) = 0 \)

- Conclusion:
  - Eigenvalues of \( A \) are the poles of the system.
  - Eigenvectors are the "mode shapes"
  - In Matlab the command is `eig(A)` or `damp(A)` or `pole(plant)`

Example: Poles of the SS system

- Consider the transfer function \( T(s) = \frac{s + 1}{s^2 + 2s + 3} \)

Zeros of the SS system

- Defined by: \[ \det \begin{bmatrix} sI - A & -B \\ C & D \end{bmatrix} = 0 \]

- Proof:

  - In Matlab type `zero(plant)` where `plant` is the state space model.
Zeros of the SS system

• Proof cont.

Example: Zeros of the SS system

• Consider the transfer function \( T(s) = \frac{s + 1}{s^2 + 2s + 3} \)

Example: Zeros of the SS system

A Few Words on Stability

• As always, the poles of a given system must be on the left side of the s-plane for the system to be stable
• In state-space format: system poles = eigenvalues of state matrix \( A \)
• Therefore, the eigenvalues of \( A \) must be negative for the system to be stable
• With modern computing packages, the validity of this requirement for a given system is easily checked
A Few Words on Stability

- Pre-computers, one technique used for stability checking was Lyapunov's method (second method for LTI systems)
- Lyapunov's method can be formulated using concepts of "energy" and "trajectory"

Lyapunov Stability

- Consider the homogenous state equation:
  \[ \dot{x} = Ax \]
- This describes the path taken by each state \( x \) from \( x(0) \) to equilibrium \( x_e \)
- The "distance" between where the (total) system is at time \( t \) and equilibrium can be found by the Euclidean norm


Lyapunov Stability

- If the state variables are phase variables, then the plot is referred to as the phase plane.
  - Typical trajectory for a two-state underdamped (stable) system.
    - Successive radii will be smaller for a stable system.
    - If the state variables are phase variables, then the plot is referred to as the phase plane.

Lyapunov Stability

- Suppose the distance (norm) between equilibrium and \( x(0) \) is bounded by a radius \( r \):
  \[ \|x(0)\| = \|x(0) - x_e\| \leq r \]
- and the radius at time \( t > 0 \) is bounded by \( R \):
  \[ \|x(t)\| = \|x(t) - x_e\| \leq R \]
- If the system is stable, and the trajectories decaying towards equilibrium, then \( R < r \). If this relationship holds true for all initial conditions \( x(0) \), then the system is asymptotically stable at large
Lyapunov Stability

To derive the Lyapunov stability criterion, consider system energy; if the system energy is decreasing, then the system is going to be stable. The problem is, how to calculate "generic system energy" (call it $V$)

Note a few properties:
1. $V(x_e) = 0$ (equilibrium)
2. $V(x) > 0$ for $||x|| \neq 0$
3. $V$ is continuous, and has continuous derivatives with respect to all states in $x$
4. $V \leq 0$ along the trajectories, in other words, system energy decreasing with time.

This leads to the requirement:
$$A^T P + PA = -Q$$
where $Q$ is any positive definite matrix, and $P$ is positive definite if the system is stable

Generally let $Q = I$. Then solve for $P$

Allows us to check stability without solving full equations or finding eigen-values (both of which are much more difficult)

The Matlab command is $P = \text{lyap}(A,Q)$

Positive Definiteness
Example: Lyapunov Stability

- Investigate the stability of the following spring-mass system
Example: Lyapunov Stability

- Investigate the stability of system with: \( A = \begin{bmatrix} -\alpha & \beta \\ -\beta & -\alpha \end{bmatrix} \)
**Topic 9: Controllability and Observability**

- **Objectives** are an understanding of:
  - Controllability
    - how one qualitatively determines such using graphical means
    - How one quantifies such using the controllability matrix
  - Observability
    - how one qualitatively determines such using graphical means
    - How one quantifies such using the observability matrix
  - Condition number of a matrix

- **Reading:**
  - Dorf and Bishop "Modern Control Systems", Chapt 11.1-11.3
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems", Chapt 7.2 and 7.5
  - Nise "Control Systems Engineering", Chapt 12.1-12.3 & 12.5-12.6
  - Skogestad and Postlethwaite, "Multivariable Feedback Control", Section 4.2-4.6

---

**Controllability**

- **What is it?**
  - Physically, it is a test of the ability of the actuators
  - A system is controllable if it is possible to transfer any state with any set of initial conditions to any final state in some finite time period
  - Alternatively, a system is only controllable if every mode (or state) is connected to the control input
  - A system is referred to as "stabilizable" so long as we can state control all unstable modes
  - Might mean that there are some stable uncontrollable states
  - Strictly speaking the dynamical system described by the pair \((A;B)\) is said to be (state-feedback) stabilizable if there exists a state feedback \(u = -Kx\) such that \(A + BK\) is stable.

- **How do we test for it?**
  - For an LTI system, the “Controllability Matrix*” must be full rank:
    - controllability matrix, \(C_o = [B \ AB \ A^2B \ ... \ A^{n-1}B]\)
    - ie, the controllability matrix must be invertible
    - Note the difference between rank and determinant
  - Often in uncontrollable systems, part of the system is unconnected from input
  - The Matlab command is \(Co = ctrb(A,B), \text{then det}(Co)\) or \(\text{rank}(Co)\)

*The derivation of the controllability matrix can be found in Appendix D, “Feedback Control of Dynamic Systems”, Franklin, Powell and Emami-Naeini
Example 1: Controllability

- Investigate the controllability of: \( A = \begin{bmatrix} -1 & 0 \\ 1 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \)

Example 2: Controllability

- Investigate the controllability of: \( A = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \)

Example 2: Controllability

Example 3: Controllability

- Investigate the controllability of:
  \[
  Y(s) = T(s) = \frac{s^2 + a_2s + a_1}{s^3 + a_2s^2 + a_1s + a_0}
  \]
Example 4: Controllability

- Investigate the controllability of:
  \[ \dot{x}_1 = x_2 - 2x_1 + u \]
  \[ \dot{x}_2 = -x_2 \]
  \[ y = x_1 \]

Example 5: 2 DOF Mass/Spring system

- Consider the symmetric 2 dof system below. Determine the conditions necessary for the system to be controllable.
Example 5: 2 DOF Mass/Spring system
Example 5: 2 DOF Mass/Spring system

Example 5: 2 DOF Mass/Spring system

Example 5: 2 DOF Mass/Spring system

Example 5: 2 DOF Mass/Spring system
Example 6: Controllability from block diagrams

- Determine the controllability of these 3 systems, noting the rank of the controllability matrix.

Observability

- What is it?
  - Physically, it is a test of the ability of the sensors.
  - A system is observable if every initial state \( x(0) \) can be determined by observing the system output over some finite time period.
  - If a system is referred to as "detectable" if all unstable modes are state observable.
  - May mean system has unobservable states which are stable.
  - Strictly speaking the pair \((C;A)\) is said to be detectable if there exists a matrix \( L \) such that \( A + LC \) is stable.

Observability

- How do we test for it?
  - For an LTI system, the "Observability Matrix" must be full rank:
    - Observability matrix, \( Q_0 = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} \)
    - ie, the observability matrix must be invertible.
    - The Matlab command is: \( \text{Ob = obsv(A,C)} \)
    - Test for observability
      \[ \text{Rank(Ob) = full rank} \]
      \[ \text{Det(Ob) \neq 0} \]

- Additional tests are to show that the observability Gramian \( Q \) is positive definite, where \( Q \) may be found by the solution to the Lyapunov equation: \( A^TQ + QA^- = -C^TC \)

- Alternatively, \( Q \equiv \int_0^\infty e^{A^Tt}C^TCe^{At}dt \)
**Example 1: Observability**

- Investigate the observability of: \( A = \begin{bmatrix} -1 & 0 \\ 1 & -2 \end{bmatrix} \),
  with \( C = [1 \ 0] \) and \( C = [0 \ 1] \)

---

**Example 2: Observability**

- Investigate the observability of: \( A = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \), \( C = [1 \ 3] \)

---

**Example 3: Observability**

- Investigate the observability of:
  \[
  Y(s) = T(s) = \frac{1}{s^3 + a_2s^2 + a_1s + a_0}
  \]

---
Example 4: Observability

- Investigate the observability of:
  \[ \dot{x}_1 = -2x_1 + 3u \]
  \[ \dot{x}_2 = -x_2 + u \]
  \[ y = x_1 \]
**Example 5: Observability from block diagrams**

- Determine the observability of these 3 systems, noting the rank of the observability

**Controllability and Observability**

- Conceptually, a system can be divided into 4 sub-systems:
  - completely controllable, completely observable
  - completely controllable, unobservable
  - completely observable, uncontrollable
  - uncontrollable and unobservable

**C&O: Additional Points**

- To find the uncontrollable / unobservable mode, re-express the equations in terms of normal modes and look for the '0' terms in the input matrix $B$ (controllability) or output matrix $C$ (observability). These modes will be the uncontrollable / unobservable ones

$$\dot{x} = Ax + Bu, \quad y = Cx + Du$$

- If a system has unobservable modes, then it must also have uncontrollable modes (intuitively, if you can't see it, you can't control it). Sensor positioning is one of the most important aspects of control system design, especially with complex systems.

A transfer function model considers only this part.
Problems with C&O

- Controllability and observability are "binary" concepts; a system either is, or is not, controllable and/or observable.

**Condition number**

- In numerical analysis, the condition number associated with a problem is a measure of that problem's amenability to digital computation, that is, how numerically well-posed the problem is.
- A problem with a low condition number (close to unity) is said to be well-conditioned, while a problem with a high condition number is said to be ill-conditioned.
- For example, the condition number associated with the linear equation \( Ax = b \) gives a bound on how inaccurate the solution \( x \) will be after approximate solution.
  - Note that this is before the effects of round-off error are taken into account;
  - conditioning is a property of the matrix, not the algorithm or floating point accuracy of the computer used to solve the corresponding system.
  - The condition number effectively amplifies the error present in \( b \).

**Problems with C&O: Condition number**

- The questions "how controllable / observable" and of the "best" (usually better / worse) locations for sensor and actuator placement is not straightforward. Most methods rely on assessing some measure of the condition number of the controllability / observability matrix.
  - Condition number = ratio (max singular value) / (min singular value).
  - Singular values are given by: \( \Lambda = V \Lambda U^T \) where \( \Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n \end{bmatrix} \)
  - The Matlab command is \( \text{cond}(C_o) \)
**Example: Condition number**

**Condition number**

- The smaller the condition number, the more "uniform" is the controllability / observability amongst modes, and therefore the better is the placement.
  - This limits the need for large dynamic ranges on the sensor or actuator

- The apparent lack of a full rank controllability / observability matrix can also be due to numerical conditioning problems with $A$
**Topic 10: Feedback Control & Pole Placement**

- **Objectives are an understanding of:**
  - The concept of full state feedback
  - Controller design using control canonical form
  - Ackermann's formula
  - Robust assignment of controller gains

- **Reading:**
  - Dorf and Bishop, 11.5 & 11.6
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems”, Chapt 7.3-7.4
  - Nise “Control Systems Engineering”, Chapt 12.2-12.4

---

**State Feedback**

Control signal $u$ is a linear combination of plant states.

Full state feedback requires that all states are known.

---

**Pole Placement - Introduction**

- **Similar in concept to classical control system design:**
  - Formulate desired pole locations to satisfy some performance criteria, then formulate control gains to make this happen

- **Design assumes that ALL states can be measured and used in control implementation; this is called **FULL STATE FEEDBACK**

---

**Pole Placement - Introduction**

- **Control input (or signal):**
  \[
  u = -kx
  \]
  where $k = \text{control vector}$ (or matrix) of proportional control gains applied to each state given by:
  \[
  k = [k_1 \ k_2 \ldots \ k_n]
  \]

- **Essentially, we are trying to modify the underlying differential equations.**
Pole Placement - Where are the poles of the CL system?

- Consider the state equation:
  - Input: \( \dot{x} = Ax + Bu = Ax - Bkx = (A - Bk)x \)

- Laplace Transform of Input: \([sl - A + Bk]x(s) = 0\)

- Closed Loop Poles defined by: \( \text{det}[sI - A] = 0 \)
  - cf. open loop poles: \( \text{det}[sI - A] = 0 \)

- Output: \( y = Cx + Du = Cx - Dkx = (C - Dk)x \)

Example 1(a)

- Consider an undamped oscillator with frequency \( \omega_n \) and a SS model given by
  \[
  \begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2
  \end{bmatrix} =
  \begin{bmatrix}
  0 & 1 \\
  -\omega_n^2 & 0
  \end{bmatrix}
  \begin{bmatrix}
  x_1 \\
  x_2
  \end{bmatrix} +
  \begin{bmatrix}
  0 \\
  1
  \end{bmatrix} u
  \]

- Find the controller that places the both CL poles of the system at \(-2\omega_n\). In other words, you want to double the natural frequency and increase the damping ratio \( \zeta \) from 0 to 1.

Example 1(a) cont.

- Hence \( K^{(2)} \) and \( K^{(1)} \) Comparing \( K^{(2)} \) and \( K^{(1)} \) therefore \( \text{Desired} 2-\text{poles} \)
Example 1(b)

Example 1(b) cont.

Example 1(c)

Pole Placement - When in Control Canonical Form

- What happens if we move poles to $s = -\omega_n$?

- Suppose we have some desired pole locations: $p_1, p_2, p_3, \ldots$

- Then the desired characteristic equation is: $(s-p_1) (s-p_2) \ldots (s-p_n) = 0$.

- This can be expanded to (desired characteristic eqn): $s^n + \alpha_1 s^{n-1} + \ldots + \alpha_{n-1}s + \alpha_n = 0$. 
**Pole Placement - When in Control Canonical Form**

- Now suppose that the state space equations are in control canonical (upper companion) form:

\[
\begin{bmatrix}
-a_1 & -a_2 & \cdots & -a_{n-1} & -a_n \\
1 & 0 & \cdots & 0 & 0 \\
0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 1 & 0
\end{bmatrix}, \quad \begin{bmatrix}
0 & 0 & \cdots & 0 & 1
\end{bmatrix}
\]

- The poles of the feedback system are defined by the expression:

\[
\det(sI - A + Bk) = 0.
\]

**Pole Placement - Generalised Strategy**

- Therefore, one possible methodology for control system design using pole placement:

1) Transform state equations into control canonical form

2) Calculate control gains by comparison with the desired characteristic equation

3) Transform back to original states.

**Conversion to Control Canonical Form**

- **Step 1.** Transform state equations into control canonical form. Transformation has the form:

\[
\begin{align*}
\dot{x}' &= T^{-1}x' \quad (x' = \text{transformed state vector}) \\
A' &= T^{-1}AT, \quad B' = T^{-1}B \quad (A' = \text{transformed state matrix})
\end{align*}
\]

where \(T\) is the transformation matrix: \(T = C_0N\) and \(C_0 = \text{controllability matrix} = [B \ AB \ \cdots \ A^{n-1}B]\)

\[
N = \begin{bmatrix}
1 & a_1 & \cdots & a_{n-1} \\
0 & 1 & \cdots & a_{n-2} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1
\end{bmatrix}
\]

- where \(a_1, a_2, \ldots\) are from the characteristic equation of the uncontrolled system: \(s^n + a_1s^{n-1} + \ldots + a_n = 0\)
Conversion to Control Canonical Form Cont.

- Note:
  - \( T \) must be invertible
  - \( T^{-1} = N^{-1} C_0 \)\(^{-1} \)
  - To invert \( C_0 \), the system must be controllable

- Conclusion:
  - The system must be controllable for pole placement to be possible

Step 3: Transform the calculated control gains back to use with original states.
- Transform has the form:
  \[ k_{\text{original state}} = k_{\text{control canonical}} T^{-1} \]
- The Matlab command is:
  \[ k = \text{place}(A, B, \text{Poles}) \]

Example 1(d): When in Control Canonical Form

- Consider an undamped oscillator with frequency \( \omega_n \) and a SS model given by:
  \[
  \begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2
  \end{bmatrix} =
  \begin{bmatrix}
  0 & 1 \\
  -\omega_n^2 & 0
  \end{bmatrix}
  \begin{bmatrix}
  x_1 \\
  x_2
  \end{bmatrix} +
  \begin{bmatrix}
  0 \\
  1
  \end{bmatrix} u
  \]

- Find the controller that places the both CL poles of the system at \(-2\omega_n\). In other words, you want to double the natural frequency and increase the damping ratio \( \zeta \) from 0 to 1.
Pole Placement using Ackermann’s Formula

- **For SISO systems** the control gains using Ackermann’s Formula are

\[ k = [0 \ 0 \ \cdots \ 0 \ 1] C_0^{-1} \gamma(A) \]

where \( C_0 \) = controllability matrix (note inversion again), and

\[ \gamma(A) = A^n + \alpha_1 A^{n-1} + \alpha_2 A^{n-2} + \cdots + \alpha_n I \]

\( A \) = state matrix,
\( \alpha_1, \alpha_2, \ldots \) = coefficients of the desired characteristic eqn.

The Matlab command is \( K = \text{acker}(A,B,Poles) \)

Example 1(e): Pole Placement using Ackermann’s Formula

- Consider an undamped oscillator with frequency \( \omega_n \) and a SS model given by

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
-\omega_n^2 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
+ \begin{bmatrix}
0 \\
1
\end{bmatrix} u
\]

- Find the controller that places the both CL poles of the system at -2\( \omega_n \). In other words, you want to double the natural frequency and increase the damping ratio \( \zeta \) from 0 to 1.

Example 2(a): Pole Placement - Direct

- Consider the SS system

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
-1 & 0 \\
1 & -2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
+ \begin{bmatrix}
1 \\
2
\end{bmatrix} u
\]

- Design a control system to move the system poles to \( s = -1 \pm j \)
Example 2(b): Pole Placement
- Via Control Canonical

Example 2(c): Pole Placement
- Via Ackermann’s Formula

Example 2(c): Pole Placement
- Via Ackermann’s Formula
Robust Eigenvalue Assignment: Pole Placement for Multi-Input

- When placing closed loop poles for a system with multiple inputs there will be an infinite number of solutions
  - In other words, if each actuator can place all the poles, then the solution involving multiple actuators is not unique
- How do we decide how much “effort” each actuator provides?
  - We use one of many “Robust Eigenvalue Assignment” routines which aim for a well-conditioned solution
  - Conditioning of eigenvector matrix is a measure of the robustness of the eigenvalues
  - Matlab place command uses Kautsky et al, "Robust pole assignment in linear state feedback", *IEC, 41*(5), 1129-1155, 1985
Robust eigenvalue assignment for 3 mass-spring system
**Topic 11: Optimal Control (LQR)**

- Objectives are an understanding of:
  - The concept of optimal control, cost functions, the state weighting matrix and effort weighting matrix
  - The linear quadratic regulator
  - The continuous algebraic Riccati equation

- Reading:
  - Dorf and Bishop, 11.4 (9th Edn) or 11.9 (10th Edn)
  - Franklin, Powell and Emami-Naeini Feedback Control of Dynamic Systems”, Chapt 7.4.2
  - Skogestad and Postlethwaite, “Multivariable feedback control”, Section 9.2.
  - Bosgra, Kwakernaak & Meinsma,”Design Methods for Control Systems”, Sections 4.1 and 4.2 (4.2.1-4.2.3, 4.2.6 & 4.2.7)
  - Anderson and Moore, “Linear Optimal Control”, Section 3.1, 5.3, 5.4

**Some final comments / questions regarding pole placement**

- If a system is controllable, we can place the closed loop poles anywhere we want. What else might we want?

- We can place poles with a single control input. What happens when there are multiple inputs?

- How do we pick pole locations?

- What happens when we have actuators of finite bandwidth and strength?

- Why use state-space methods for pole placement (ie, are there advantages to be found over pole placement)?

**Optimal Control - Introduction**

- The answers to these questions lead us to the idea of "Optimal Control".

- In optimal control, we introduce a measure of the performance into the problem, which is used to (inherently) guide pole placement.

- Note the following:
  - Optimal control is a large area of control, and it constitutes an area of study in its own right (this is why it is completely ignored in most student texts). It can be mathematically expensive. We will look at the basics only, from the standpoint of the problem formulation for computer solution.

**“Standard” Control Problem**

- We have a dynamic system whose response is governed by the state equation:
  \[ \dot{x} = Ax + Bu \]
  - We are considering the general MIMO (multiple input, multiple output) case.

- We want to modify the dynamic response characteristics by introducing a control input:
  \[ u = -Kx \]

- The problem is to determine the gain matrix \( K \) which will satisfy our requirements.
The Optimal Control Problem

• For optimal control, instead of seeking a gain matrix $K$ which places the poles in certain locations, we will seek a gain matrix which minimises some performance index, or cost function, $J$:

$$J = \int_0^T \left[ x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau) \right] d\tau$$

This term relates to achieving a “desired outcome”; it is a “penalty” for deviation from the equilibrium position

This term relates to a “penalty” for using large control inputs

Some comments:

➢ $T =$ final (terminal) time, $t =$ present (initial) time. If $T = \infty$, then we have a "steady state optimal control problem" (i.e., a regulator design problem). This will be our main interest

➢ $Q$ and $R$ weighting matrices; $Q$ is the positive definite state weighting matrix, and $R$ is the positive definite control (or effort) weighting matrix

➢ The controller will be optimal only in terms of minimising the previous mathematical expression (is this really the objective?) for the given $Q$ and $R$ (how do we get these?)

The Optimal Control Problem

• The same comments regarding optimality apply to the selection of the effort weighting matrix, $R$

• $R$ is used to limit the control signal size

• If $R$ is “too small”, this may result in actuator saturation (not necessarily a bad thing if designed for)

• If $R$ is “too large”, this may limit performance to a point where the design objectives are not met.

Ratio of $Q/R$ and associated trade off

• Larger the value of $Q$: $Q/R$ increases
  ➢ Larger the value of gain $K$
  ➢ Larger the control input - could saturate the actuator
  ➢ Better performance – settling time improves
  ➢ Damped system – implies less overshoot

• Larger the value of $R$: $Q/R$ reduces
  ➢ Smaller the value of gain $K$
  ➢ Smaller the control input – reduced control effort
  ➢ Limited performance – design objectives not achieved

• Trade off: Performance of system v/s control effort
  ➢ Faster performance requires bigger control effort
Optimal Control: Choice of $Q$ & $R$

• Suppose state 1 ($x_1$) is displacement, and the other states are successive derivatives (phase variables): $x_2 = \dot{x}_1$, $x_3 = \ddot{x}_1$, ...

• If only displacement is of concern then we might select

$$Q = \begin{bmatrix}
1 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & 0 \\
0 & 0 & \cdots & 0
\end{bmatrix}$$

so that the performance index is $J = \int_0^\infty [x_1^2 + u^T Ru] dt$

Optimal Control: Choice of $Q$ & $R$ – Bryson's Rule

• One common point for designing an optimal controller is to choose diagonal $Q$ and $R$ such that

  - $Q_{ii} = 1/(\text{max acceptable value of } [x_i]^2)$
  - $R_{ii} = 1/(\text{max acceptable value of } [u_i]^2)$

• The weighting matrices are then modified in subsequent iterations to achieve an acceptable tradeoff between control performance and control effort.

• Thus the cost function is:

$$J = \int_0^\infty \left[ \sum_{i=1}^N x_i^2 \left( \frac{x_i}{x_{i,max}} \right)^2 + \sum_{i=1}^N u_i^2 \left( \frac{u_i}{u_{i,max}} \right)^2 \right] dt$$

Solving for the Optimal Control Gain Matrix

• Given the optimal control performance index, there are a number of ways to derive an expression which defines the optimal gain matrix (which minimises the performance index).

  - We will use the second method of Lyapunov to derive the expression.

• For this course, the method is not as important as the end result.
**Evaluation of the performance index for the homogenous case**

- Consider a system without a controller, described by the homogeneous state equation: \( \dot{x} = Ax \)

- If the system is perturbed, what is the value of the performance index? 

- We can show (see Hansen & Snyder or Dorf & B.) for \( P \) defined by:
  \[
  A^T P + PA = -Q
  \]

  - Note: 
    - \( x(0) \) is the initial state (at start time).
    - System is stable for positive definite \( Q \).
    - Results could be used for assessing the influence of system parameter changes (in \( A \)).

**Now Introduce Control**

- Consider a system with a controller, described by:
  \( \dot{x} = Ax - BKx = (A-BK)x \)

- The performance index is now:
  \[
  J = \int_0^\infty (x^T Q x) \, d\tau
  \]

  - It can be shown that for \( P \) defined by:
    \[
    (A-BK)^T P + P (A-BK) = -(Q + K^T R K)
    \]

- We now need to use this relationship to derive a gain matrix \( K \) which minimises the performance index. Steps are (see texts):
  - Factor \( R = T T^T \) (eg Cholesky), where \( T \) is a non-singular matrix.
  - Expand defining equation for \( P \).
  - Minimise both sides of the expression - it will produce a minimum \( P \), which will minimise the value of the error criterion.
  - Minimum \( P \) is defined by: \( T^T K = T^T B^T P \)
  - Therefore, optimal gain matrix is:
    \[
    K = R^{-1} B^T P
    \]

**Algebraic Riccati Equation**

- We now need to use this relationship to derive a gain matrix \( K \) which minimises the performance index. Steps are (see texts):
  - Factor \( R = T T^T \) (eg Cholesky), where \( T \) is a non-singular matrix.
  - Expand defining equation for \( P \).
  - Minimise both sides of the expression - it will produce a minimum \( P \), which will minimise the value of the error criterion.
  - Minimum \( P \) is defined by: \( T^T K = T^T B^T P \)
  - Therefore, optimal gain matrix is:
    \[
    K = R^{-1} B^T P
    \]

**A few comments on the optimal control gain derivation exercise**

- For the solution to exist, \( (A-BK) \) must be stable. It is possible to show that this requirement is met if:
  - the system (matrix pair \( [A, B] \)) is controllable.
  - If this requirement is met, then the algebraic Riccati equation has a unique, positive definite solution \( P \) which minimises the performance index.
  - Solving the Riccati equation is not a trivial task, and normally requires some form of iterative approach.
  - Note also that the optimal control problem is often referred to as the Linear Quadratic Regulator (LQR) problem.
  - The Matlab commands are \( P = \text{CARE}(A, B, Q, R) \) and \( K = \text{lqr}(A, B, Q, R) \).
Example 1: Optimal Control

- Consider the following first order system,
  \[ \dot{x} = x + u \]
  ➢ Design an optimal controller given that \( Q = 1, \; R = 0, \; x(0) = \sqrt{2} \)

Example 1: Cont.

- Consider the following first order system,
  \[ \dot{x} = x + u \]
  ➢ Design an optimal controller given that \( Q = 1, \; R = \lambda \).
Example 2: Cont.

Example 3: Optimal Control

- Consider the following first order system,
  \[ \dot{x} = x + u \]
  ➔ Design an optimal controller given that \( Q = 1, R = \lambda \) using the Riccati equation.

Example 3: Cont.

Example 4: Optimal Control

- Consider the dynamic cart-bucket system.
  Derive an optimal control law.
Example 4: Optimal Control

Given the system described in (5), we can substitute into function (6) to obtain the optimal control. Let

\[ J = \int_{0}^{T} \left( g(x) + u^2(x) \right) \, dx \]

be the cost function. Assume the equations (3) and (2) from the previous slide.

Continued...

Assume (4) and (1) are given, then equation (2) can be used to model the space and state. The differentiations in (2) and (3) lead to the Riccati equation.

\[ \frac{dP}{dt} = \frac{M}{g} \left( B - \frac{M}{g} P \right) \]

and

\[ \frac{dP}{dt} = \frac{M}{g} \left( C - \frac{M}{g} P \right) \]

The Riccati equation is a differential equation that can be solved to find the optimal control and state feedback gains. The solution can be found by solving the characteristic equation

\[ \lambda^2 + \lambda + \frac{1}{g} = 0 \]

which yields the poles

\[ \lambda_{1,2} = \frac{-1 \pm \sqrt{1 - \frac{4}{g}}}{2} \]

These poles determine the closed-loop system. The values of 'g' and 'M' are determined by solving the Riccati equation.

Closed-loop poles are

\[ \lambda_{1,2} = \frac{-1 \pm \sqrt{1 - \frac{4}{g}}}{2} \]

The Riccati equation can be used to find the optimal control and state feedback gains.
Example 4: Optimal Control
Example 4: Optimal Control

Limiting behaviour of LQR

- Expensive control (when $R \to \infty$)
  - Penalises control energy
  - Does not move stable poles (in the LH s-plane)
  - RH poles are moved to their mirror location in the LH plane
    - Does this to stabilise the system with minimal effort
    - Consequently the speed of the controlled system (poles) is the same as the uncontrolled system.

- Cheap control (when $R \to 0$)
  - Control energy is no object
  - Control law moves some of the closed loop poles right on top of the minimum-phase zeros (LH plane)
  - All other poles are moved to infinity
  - Non minimum-phase zeros are moved to their mirror location in the LH plane
Example: Expensive and Cheap Control

LQRY Control

- In many applications it is typical to want to minimise the output signals.
- An appropriate cost function to do this is the standard quadratic regulator cost function:

  \[ J = \int \left[ y^T(\tau)Qy(\tau) + u^T(\tau)Ru(\tau) \right] d\tau \]

  which minimises the weighted output in an optimal tradeoff with control effort.
- Since \( y(t) = Cx(t) \) it is simple to show that

  \[ J = \int \left[ x^T(\tau)(C^TQC)x(\tau) + u^T(\tau)Ru(\tau) \right] d\tau \]

  which may be solved using the CARE such that

  \[ A^T P + PA - PB^T B^T P + C^T QC = 0 \]

LQRY Control of acceleration

- In active control of civil structures it may be desired to control the acceleration at a number of locations.
- Consider the second-order LTI model for a structural system:

  \[ M\ddot{v}(t) + C\dot{v}(t) + K\ddot{v}(t) = F(t) \]

  where \( v \) is the displacement of the structure (with derivatives velocity and acceleration), and \( M, K \) and \( C \) are the global mass, stiffness and damping matrices respectively.
- In state space form:

  \[ \dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -M & -C \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ M^{-1} \end{bmatrix} F(t) \]

  where the state vector \( x(t) = [\dot{v}(t), \ddot{v}(t)]^T \) is comprised of structural displacements and velocities respectively.
LQRY Control of acceleration

- It is relatively easy to show that the structural acceleration is given by
  \[ \ddot{v}(t) = -M^{-1}Kv(t) - M^{-1}Cv(t) + M^{-1}F(t) \]
  \[ = \left[-M^{-1}K - M^{-1}C\right]x(t) + M^{-1}F(t) \]
  \[ = C_1x(t) + M^{-1}F(t) \]
  where \( C_1 = [-M^{-1}K - M^{-1}C] \)

- Therefore, if we let our output be only the acceleration due to internal motion and external forces (and not control forces), i.e.,
  \[ y(t) = \ddot{v}(t) - M^{-1}F(t) = C_1x(t) \]
then the solution to the CARE will minimise this acceleration (such that the output matrix is \( C_1 \)).

- Using a similar approach, it is also possible to derive an expression that minimises jerk (which relates to occupant comfort).

LQR Robustness

- For optimal control with full-state feedback (with diagonal weights \( R \)):
  - The Kalman Inequality (which also applies to unstable plants) says \(|S|<=1\) for all frequencies, where \( S \) is the sensitivity function.
  - Yields a loop transfer function with one excess pole (one more pole than it has zeros).
  - There are no non-minimum phase zeros (in the RHP).

LQR Robustness

- It can be shown that the optimal control gain (assuming full-state feedback and diagonal weights \( R \)) yields a system with
  - A phase margin of at least ±60 degrees (minimum)
  - An infinite gain margin
    - In other words, gain matrix \( K \) can be increased by a very large scalar and stability will be assured.
  - A "downsize" gain margin (reduction in gain) of 1/2 or 6B
    - In other words, gain matrix \( K \) can be decreased a factor of 2 and stability will be maintained.
Topic 12: Observers (Estimators)

- Objectives are an understanding of:
  - What an observer (estimator) is and why it is necessary
  - The concept of measurement noise
  - Why closed loop (feedback) is necessary for observers and how to determine the observer feedback gain
  - Calculation of observer gains using pole placement and Ackermann’s formula

- Reading:
  - Dorf and Bishop "Modern Control Systems", 10th Edn, Chapt 11.6
  - Driels, "Linear Control Systems Engineering", Module 25
  - Nise, "Control Systems Engineering", 12.5
  - Franklin, Powell & Workman, "Digital Control of Dynamic Systems", 8.2
  - Franklin, Powell & Emami-Naeini, "Feedback Control of Dynamic Systems", 7.5

Why do we need them?

- State space control systems work on the idea that all states will be available to be fed back for controller implementation. This is not always (usually) the case:
  - sensors are expensive
  - states are not always measurable
  - measurements may be noisy

- Therefore, in order to implement the control system, we must estimate (observe) the values of the states, and use these to derive a control signal.

- This is the function of the observer (aka estimator):
  - To estimate the value of some / all of the state variables.

A control system usually has two parts:

- Controller:
  - Takes measurements and/or estimates of the state variables, multiplies them by the control gains, and produces the control signal.
  - design by pole placement or optimal control

- Observer / estimator:
  - Estimates some or all of the states of the system.
  - Two types:
    - Linear observers (pole placement)
    - Optimal observers (Kalman filters)
  - We will start with linear observers.

How might you build an observer?

- System described by: \( \dot{x} = Ax + Bu, \quad y = Cx \)
- Assume that we know \( A, B, C, u \).
- Possibilities for extracting \( x \):
  - 1. Somehow work with the defining equations for output? Invert \( C \)? Too noisy, \( C \) might be singular, etc.
  - 2. Construct a second linear system (a model of the target system), using the known parameters (\( A, B, C, D, u \)) of the target system, which predicts the (measurable) target system output. If the predicted output is acceptably close to the actual output, then we can use the estimated states in place of the actual states.
Open-Loop Observers: Block Diagram

plant \[ x = Ax + Bu \]
control input \[ u \]
actual states \[ x \]
actual output \[ y \]
model (observer) \[ \hat{x} = A\hat{x} + Bu \]
model (eg, in software)
estimated states \[ \hat{x} \]
predicted output \[ \hat{y} \]

Observers

- So, we want to minimise the difference between the actual and predicted states. What will the difference be?
- Define error in estimate of state, \( x_e = x - \hat{x} \) which converges to zero if \( A \) is stable.
- Open loop dynamics:
  \[ x_e = \dot{x} - \dot{\hat{x}} = (Ax + Bu) - A(x - x_e) - Bu = Ax_e \]
  Characteristic eq: \( \det(sI - A) = 0 \)
- Convergence of the error is governed by the poles of the open loop system. What do we do if the convergence is too slow?

Add feedback to improve observer performance

- Observer (model) system response is now:
  \[ \dot{x} = (Ax + Bu) + L(y - \hat{y}) = (Ax + Bu) + LC(x - \hat{x}) \]
- The dynamics of the error become:
  \[ \dot{x}_e = x - \hat{x} = Ax + Bu - [A\hat{x} + Bu] + LC(x - \hat{x}) \]
  \[ = Ax - A\hat{x} - LC(x - \hat{x}) \]
  But \( (x - \hat{x}) = x_e \), therefore
  \[ \dot{x}_e = A\hat{x} - LC\hat{x}_e \]
  \[ \dot{x}_e = x_e(A - LC) \]
- Therefore, the char. equation of the close loop observer is now:
  \[ \det(sI - (A - LC)) = 0 \]
  ie, we can change convergence speed by adding feedback.
- Note: Feedback not only speeds up the response, it also reduces the estimation error when:
  - there are unknown inputs (disturbances),
  - unknown initial conditions and,
  - errors in the state model.
Observer and controller

The observer is identical to the original plant except for the observer feedback.

Observer feedback gains

- We now need to determine acceptable observer feedback gains \((L)\). How?
  1. Pole Placement
  2. Optimal calculation

  Similar to the controller design (aka the "duality" of control)

- Note in advance: the system must be observable (as opposed to controllable) for this to work

Observer Pole Placement via Conversion to Observer Canonical Form

Step 1.

- Suppose we have some desired (observer error) response characteristics, defined by the pole locations: \(p_1, p_2, p_3, \ldots\)

- Then the desired characteristic equation is:

\[
(s - p_1) (s - p_2) \ldots (s - p_n) = 0.
\]

- This can be expanded to:

\[
s^n + \alpha_1 s^{n-1} + \ldots + \alpha_{n-1} s + \alpha_n = 0.
\]

Observer Pole Placement via Conversion to Observer Canonical Form

Step 2.

- Now suppose that the state space equations are in observer canonical form:

\[
A = \begin{bmatrix}
-a_1 & 1 & \cdots & 0 \\
-a_2 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
-a_{n-1} & 0 & \cdots & 1 \\
-a_n & 0 & \cdots & 0
\end{bmatrix},
B = \begin{bmatrix} b_1 \\
b_2 \\
\vdots \\
b_{n-1} \\
b_n \end{bmatrix},
C = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \end{bmatrix}
\]

Characteristic equation from the Open loop plant
Observer Pole Placement via Conversion to Observer Canonical Form

Step 2.

With observer feedback, the poles are defined by the expression: \( \det(sI - A + LC) = 0 \).

\[
\begin{bmatrix}
s + a_1 + l_1 & -1 & \cdots & 0 & 0 \\
a_2 + l_2 & s & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
a_{n-1} + l_{n-1} & 0 & \cdots & s & -1 \\
a_n + l_n & 0 & \cdots & 0 & s \\
\end{bmatrix}
\]

\( \det(sI - A + LC) = s^n + (a_1 + l_1)s^{n-1} + (a_2 + l_2)s^{n-2} + \ldots + (a_n + l_n) = 0 \)

Compare this to the "desired" characteristic equation:

\( s^n + \alpha_1 s^{n-1} + \alpha_2 s^{n-2} + \ldots + \alpha_{n-1} s + \alpha_n = 0. \)

The conclusion is that when the (observer error) system is in observer canonical form, then observer gains can be calculated by simple comparison of coefficients:

\[ l_i = \alpha_i - a_i. \]

Example: Observer Design

Compute the estimator (observer) gain matrix which will place both estimator poles at \( -10\omega_n \) given

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
-\omega_n^2 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
+ \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \quad y = \begin{bmatrix} 1 & 0 \end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
\]

Example: Observer Design Cont.
Generalised Observer Methodology

• Therefore, one possible methodology for observer design using pole placement:

1. Transform state equations into observer canonical form

2. Calculate observer gains by comparison with the desired characteristic equation

3. Transform back to original states

Conversion to Observer Canonical Form (right companion)

• **Step 1.** Transform state equations into observer canonical form. Transformation has the form: 
  \[ x' = T^{-1}x \]  (\( x' \) = transformed state vector)

  \[ A' = T^{-1}AT, \quad C' = T^{-1}C \]  (\( A' \) = transformed state matrix)

  where

  \[
  A' = T^{-1}AT = \begin{bmatrix}
  0 & 0 & \cdots & 0 & -a_n \\
  1 & 0 & \cdots & 0 & -a_{n-1} \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  0 & 0 & \cdots & 0 & -a_2 \\
  0 & 0 & \cdots & 1 & -a_1 \\
  \end{bmatrix}
  \]

  where \( a_1, a_2, \ldots \) are from the characteristic equation of the “uncontrolled” observer: \( s^n+a_1s^{n-1}+\ldots+a_n=0 \)

Conversion to Observer Canonical Form (left companion)

• **Step 1 cont.**

  \( T \) is the transformation matrix: 
  \[ T = (NO_b)^{-1} \]

  \[ O_b = \text{observability matrix} = [C^T \quad A^T \quad C^T \quad \ldots \quad (A^T)^{n-1} C^T]^T \]

  \[ N = \text{the following Toeplitz matrix} : \]

  \[
  \begin{bmatrix}
  a_{n-1} & a_{n-2} & \cdots & a_1 & 1 \\
  a_{n-2} & a_{n-3} & \cdots & 1 & 0 \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  a_1 & 1 & \cdots & 0 & 0 \\
  1 & 0 & \cdots & 0 & 0 \\
  \end{bmatrix}
  \]

  where \( a_1, a_2, \ldots \) are from the characteristic equation of the “uncontrolled” observer: \( s^n+a_1s^{n-1}+\ldots+a_n=0 \)
Conversion to Observer Canonical Form (left companion)

• **Step 1 cont.**
  
  $T$ is the transformation matrix: $T = (N_0)^{-1}$

  $O_b = \text{observability matrix} = \left[ C^T A^T C^T \ldots (A^T)^{n-1} C^T \right]^T$

  $N = \text{the following Toeplitz matrix:}

  \[
  \begin{bmatrix}
  1 & 0 & \cdots & 0 & 0 \\
  a_1 & 1 & \cdots & 0 & 0 \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  a_{n-2} & a_{n-3} & \cdots & 1 & 0 \\
  a_{n-1} & a_{n-2} & \cdots & a_1 & 1
  \end{bmatrix}
  \]

  where $a_1, a_2, \ldots$ are from the characteristic equation of the "uncontrolled" observer: $s^n + a_1 s^{n-1} + \ldots + a_n = 0$

Conversion to Observer Canonical Form Cont.

• Note:
  
  $\begin{align*}
  &\text{T must be invertible} \\
  &\text{T} = O_b^{-1} N^{-1} \\
  &\text{To invert} \ O_b, \ \text{the system must be observable}
  \end{align*}$

  
  Conclusion:

  $\begin{align*}
  &\text{The system must be observable for pole placement observer gain calculation to be possible}
  \end{align*}$

Conversion to Observer Canonical Form Cont.

• **Step 3:** Transform the calculated control gains back to use with original states.
  
  $\begin{align*}
  &\text{Transform has the form:} \\
  &L_{\text{original state}} = T \ast L_{\text{observer canonical}}
  \end{align*}$

Example: Observer Design using Observer Canonical

• Compute the estimator (observer) gain matrix which will place both estimator poles at $-10 \omega_p$ given

  $\begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2
  \end{bmatrix} =
  \begin{bmatrix}
  0 & 1 \\
  \omega_p^2 & 0
  \end{bmatrix}
  \begin{bmatrix}
  x_1 \\
  x_2
  \end{bmatrix} +
  \begin{bmatrix}
  0 \\
  1
  \end{bmatrix} u, \ y = [1 \ 0]
  \begin{bmatrix}
  x_1 \\
  x_2
  \end{bmatrix}$
Example: Observer Design using Observer Canonical

Pole Placement using Ackermann’s Formula

- For SISO systems the observer gains using Ackermann’s Formula are
  \[ L = \gamma(A)O_b^{-1}[0 \ 0 \ \ldots \ 0 \ 1]^T \]
  where \( O_b \) = observability matrix (note inversion again), and
  \[ \gamma(A) = A^n + \alpha_2 A^{n-2} + \cdots + \alpha_n I \]
  \( A \) = state matrix,
  \( \alpha_1, \alpha_2, \ldots \) = coefficients of the desired characteristic eqn.

Example: Observer Design using Ackermann’s Formula

- Compute the estimator (observer) gain matrix which will place both estimator poles at \(-10\omega_p\) given
  \[
  \begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2
  \end{bmatrix} = \begin{bmatrix}
  0 & 1 \\
  -\omega_p^2 & 0
  \end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2
  \end{bmatrix} + \begin{bmatrix}
  0 \\
  1
  \end{bmatrix} u,
  \begin{bmatrix}
  y \\
  \hat{y}_1
  \end{bmatrix} = \begin{bmatrix}
  1 & 0
  \end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2
  \end{bmatrix}
  \]
Observer/Controller Duality

- Pole placement for control and estimation are mathematically equivalent
- Control Observer
  \[
  \begin{align*}
  A & \rightarrow A^T \\
  B & \rightarrow C^T \\
  C & \rightarrow B^T
  \end{align*}
  \]

  A comment on Matlab: Note that the pole placement exercise for the observer is identical to that for the controller, with $A^T$ used in place of $A$ (in the controller) and $C^T$ used in place of $B$. Matlab requires you to use this symmetry for pole placement in observers; it does not have an explicit routine for the observers, only the controllers.

  The Matlab command is
  \[
  L = \text{transpose}(\text{place}(A',C',\text{Poles}))
  \]
  \[
  \text{estimator} = \text{estim}(\text{plant},L)
  \]

Additional Comments

- Where should observer poles be placed?
  - We want the observer to converge faster than the controller. Therefore, stated rule of thumb is make the observer poles 4-10 times faster than the controller poles.
  - Ideally these will have heavily damped poles since it is the magnitude of the real part of the pole that determines settling time.
Lecture 13: Optimal Observers
(Kalman-Bucy Filters, LQG)

- Objectives are an understanding of:
  - What an optimal observer is
  - Concept of process noise (disturbances) and measurement noise
  - Duality of control and estimation

- Reading:
  - Hansen and Snyder, “Active Control of Noise and Vibration”, Section 5.11
  - Stengel, “Optimal Control and Estimation”, Chapter 4
  - Maybeck, “Stochastic models, estimation and control”, Volume 1, Chapter
  - Gelb, “Applied optimal estimation”, Section 4.3
  - Bosgra, Kwakernaak & Meinsma,”Design Methods for Control Systems”, Sections 4.1 and
    4.3 (4.3.2 & 4.3.3)
  - Anderson and Moore, “Linear Optimal Control”, Section 8.4,
    http://users.rsise.anu.edu.au/~john/papers/BOOK/B01.PDF
  - Anderson and Moore, “Optimal Control”, Section 7.3,
    http://users.rsise.anu.edu.au/~john/papers/BOOK/B03.PDF

Optimal Observers

- As with controller design, there are two observer design approaches:
  1. Pole placement, where pole location is explicitly targeted.
  2. Optimal observers (Kalman filters), where an error criterion is
     minimised.

- Question: What is "optimal" in an observer?
  - Answer: there is no "effort" in an observer, as it’s a software
    construct.

- The key to determining what is optimal is to look at "noise" in the system.

- Note: We are only going to consider LTI systems.

Two Sources of Noise:

- The estimator error equation with these additional inputs is:

  \[ \hat{x} = (A - LC)\hat{x} + Gw - Lv \]

- Note that the sensor noise is multiplied by the observer gain, \( L \),
  whereas the process noise is not.

  
  - Hence, measurement noise is amplified by observer gain
  - Means there is a compromise between good disturbance rejection
    (high observer gain) and filtering sensor noise (low observer gain)!

Two Sources of Noise:

- Process Noise:
  - Noise that is "really there"
  - Plant will respond to this noise

- Sensor Noise:
  - Noise which is "not really there"; purely from sensor
  - Plant will not respond to this noise (provided it does not
    circulate through controller)

- State Equations with noise included:
  - plant with process noise:
    \[ x = Ax + Bu + Gw \]
  - measurement with sensor noise:
    \[ y = Cx + Du + v \]
  where \( w = \) process noise, \( v = \) sensor noise
Optimal Observers

- Problem: Given what we know about the system, find the optimum observer which minimises the estimation error variance.
- Assume that the spectral density functions of noise processes $w$, $v$ are known (ie, we can mathematically describe $w$, $v$), such that:
  - they are white,
  - they are Gaussian
  - have zero mean: $E\{w\} = E\{v\} = 0$
  - have known covariance: $E\{ww^T\} = Q$, $E\{vv^T\} = R$
  - and are independent: $E\{wv\} = 0$

Form of observer (state estimate eqn):

$$\hat{x} = A\hat{x} + Bu + L(y - C\hat{x} - Du)$$

It can be shown that:

$$L = PC^TR^{-1}$$

where $P$ is defined by:

$$AP + PA^T - PC^TR^{-1}CP + GQG^T = 0$$

Comments

- When deriving the Riccati equation it has been assumed that:
  - the plant is linear time-invariant ($A$, $B$ & $C$ are not functions of time)
  - the driving noise statistics are stationary ($Q$ & $R$ are not functions of time)
  - Complete observability ensures the existence of a steady-state solution
  - Complete controllability ensures that the steady-state solution is unique
- In the steady-state, the rate at which the uncertainty builds (positive semi-definite $GQG^T$) is balanced by the rate at which new information enters the system (negative semi-definite $-PC^TR^{-1}CP$) and the system dissipation due to damping (arising from $A$)

The larger the disturbances ($Q$), the greater the rate of growth in the uncertainty
The larger the measurement noise (the smaller the $R^{-1}$ term), the less the measurements contribute to the reduction in the uncertainty ($-PC^TR^{-1}CP$)

Consequently, as the measurement noise increases, the observer gain decreases ($L = PC^TR^{-1}$)
This is intuitive since we will be less likely to trust the actual measurements to estimate the states if the measurement noise is large, so the filter will converge slowly to the actual measurements.
And vice-versa, if the measurement noise is extremely low, then we want the filter to converge quickly as we have more confidence in the measurements.
Duality for Optimal Control/Estimation

- Comparing the Riccati equation for the optimal sensing and optimal control laws, the similarity between the two solutions is apparent
- Control Observer
  \[
  A \rightarrow A^T \\
  B \rightarrow C^T \\
  C \rightarrow B^T \\
  Q \rightarrow GQG^T \\
  R \rightarrow R
  \]
- The Matlab command is
  \[
  L = \text{kalman}(\text{plant},Q,R) \quad \text{or} \\
  \]

Example: Optimal Observer

- Consider the following first order system,
  \[
  \dot{x} = x + u + w, \quad y = x + v
  \]
  - Design an optimal observer given that \( Q = 1, R = \varepsilon \)

Example: Optimal Observer Cont.
Consider an object which on average remains stationary but is subject to small random (RMS) velocity fluctuations of $\sigma_w$. The position of the object is measured by a sensor with noise standard deviation of $\sigma_v$. Determine the Kalman gains and the estimator transfer function.
Example: Optimal Observer 2

Example: Multiple measurements of same output

- Consider an object which on average remains stationary but is subject to small random (RMS) velocity fluctuations of $\sigma=1\text{m/s}$. The position of the object is measured by two sensors with noise standard deviation of $\sigma_1=1\text{m}$ and $\sigma_2=2\text{m}$. Determine the Kalman gains and the estimator $\hat{x}$. 

Example: Multiple measurements of same output
"Qualitative" Thoughts on an Optimal Observer

- The observer must balance between "speed" of obtaining a good estimate (in the face of system perturbations due to process noise) and "contamination" of estimates due to sensor noise.
- Poles of the observer are defined by: \[ \text{det} [sI - A + LC] = 0 \]
  - ie, poles defined by observer gains
- If poles are:
  - High Frequency:
    - speed of response is increased, as is bandwidth (hence susceptibility to sensor noise contamination is also up)
  - Low Frequency:
    - speed of response is down, and so is bandwidth (hence susceptibility to sensor noise contamination is also down)
  - Therefore:
    - If have low sensor noise, should use fast observer poles
    - If have high sensor noise, should use slow observer poles

A few problems ...

- Optimal observers (Kalman filters) are derived based upon the process and sensor noise being white, with known probability density functions (PDF's). In practice, however:
  1. They may not be white
  2. They may not have known PDF's
  3. Sensors may be of variable qualities, and so the sensor noise spectral density matrix \( R \) may be very poorly conditioned
- Therefore, may want to construct a reduced order observer, purely for estimating the states / observations which satisfy the assumptions
Topic 14: Reduced Order Observers

- Objectives are an understanding of:
  - What is a reduced order observer, their advantages and disadvantages
  - The block diagram of these observers
  - How to determine the observer gains using pole placement and Ackermann’s formula

- Reading:
  - Driels, “Linear Control Systems Engineering”, Module 25
  - Nise, “Control Systems Engineering”, 12.5
  - Franklin, Powell & Workman, “Digital Control of Dynamic Systems”, 8.2
  - Franklin, Powell & Emami-Naeini, “Feedback Control of Dynamic Systems”, 7.5.2
  - Anderson and Moore, “Linear Optimal Control”, Section 8.3,
  - Anderson and Moore, “Optimal Control”, Section 7.2,

Reduced Order Observers

- General Philosophy
  - Estimate only what is required; measure whatever can be measured (with good fidelity).

- Approach:
  1. Divide system into what can / cannot be measured.
  2. Set up "observer-like" equations, by substituting in the:
     (i) state matrix ($A$) and control input ($B$)
     (ii) known input (was $u$)
     (iii) known measurement (was $y$)
     of the unknown part of the system (ie, construct observer equations based upon the unknown part)
  3. Place the observer gains as before.

Reduced Order Observers: Details

- Original system: $\dot{x} = Ax + Bu$, $y = Cx$, (assume $D = 0$)

- Let: $x_a = \text{measurable states, } x_b = \text{states to be estimated.}$

- So $x = \begin{bmatrix} x_a \\ x_b \end{bmatrix}$, $A = \begin{bmatrix} A_{aa} & A_{ab} \\ A_{ba} & A_{bb} \end{bmatrix}$, $B = \begin{bmatrix} B_a \\ B_b \end{bmatrix}$, $C = \begin{bmatrix} I & 0 \end{bmatrix}$

- Matrix of zeros

- We can therefore write the state equations as

$$\begin{bmatrix} \dot{x}_a \\ \dot{x}_b \end{bmatrix} = \begin{bmatrix} A_{aa} & A_{ab} \\ A_{ba} & A_{bb} \end{bmatrix} \begin{bmatrix} x_a \\ x_b \end{bmatrix} + \begin{bmatrix} B_a \\ B_b \end{bmatrix} u$$

$$y = x_a = \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} x_a \\ x_b \end{bmatrix}$$

Reduced Order Observers: Details

- If we now look at the governing equations for the states to be estimated:

$$\begin{align*}
\dot{x}_b &= A_{bb} x_b + A_{ba} x_a + B_b u = A_{bb} x_b + [A_{ba} B_b] u \\
\text{State Matrix Vector} & \quad \text{State Input Matrix} \\
\text{Known Input} & \quad \text{Known Measurement}
\end{align*}$$

- The measured output is the dynamics of the known states:

$$y = A_{aa} y + A_{ab} x_b + B_a u$$

- or

$$A_{ab} x_b = \hat{y} - A_{aa} y - B_a u$$

Kinda like, $Cx = y$ where $A_{ab}$ is like the $C$ matrix.
Reduced Order Observers: Details

• Substitute into observer equations:
  \[ x_b \rightarrow x \quad \text{state vector} \]
  \[ A_{ab} \rightarrow A \quad \text{state matrix} \]
  \[ [A_{bb} \ B_b] \rightarrow B \quad \text{state input matrix} \]
  \[ y \rightarrow u \quad \text{input vector} \]
  \[ y - A_{ab}y - B_u u \rightarrow \text{measured output} \]
  \[ A_{ab} \rightarrow C \quad \text{output vector} \]

  This produces the reduced-order observer equation:
  \[
  \begin{bmatrix}
  \dot{x}_b \\
  y - A_{ab}y - B_u u
  \end{bmatrix}
  = A_{bb} \begin{bmatrix}
  \dot{x}_b \\
  y - A_{ab}y - B_u u
  \end{bmatrix}
  + L \begin{bmatrix}
  \dot{y} - A_{ab}y - B_u u - A_{ab} \dot{x}_b
  \end{bmatrix}
  \]

Reduced Order Observers: Poles

• The error in the estimator is given by the difference between the actual state and the estimated state, \( \tilde{x}_b = x_b - \hat{x}_b \)

• Differentiating the error and substituting
  \[
  \dot{\tilde{x}}_b = A_{bb}(x_b - \hat{x}_b) - LA_{ab}(x_b - \hat{x}_b)
  \]

  \[
  = (A_{bb} - LA_{ab}) \tilde{x}_b
  \]

• Poles of error equation: \( \det(sI - (A_{bb} - LA_{ab})) \)

Reduced Order Observers: Design

• Since the poles of the reduced order observer are given by \( \det(sI - (A_{bb} - LA_{ab})) \)

• this is equivalent to having a State Matrix:
  \[
  A_{Reduced\ Order} = A_{bb}
  \]

• and a State Output Matrix:
  \[
  C_{Reduced\ Order} = A_{ab}
  \]

• Given this, then the reduced order observer gains can be designed using previous observer design techniques.

Reduced Order Observer Gains Via Ackermann’s Formula

• For SISO systems the observer gains using Ackermann’s Formula are
  \[
  L = \gamma(A_{bb}) \hat{O}_b^{-1} \begin{bmatrix}
  0 & 0 & \ldots & 0 & 1
  \end{bmatrix}^T
  \]

  where
  \[
  \hat{O}_b = \begin{bmatrix}
  A_{ab} & A_{ab}A_{bb} & \cdots & A_{ab}A_{bb}^{n-2} \\
  A_{ab}A_{bb} & A_{ab}A_{bb}^2 & \cdots & A_{ab}A_{bb}^{n-2} \\
  \vdots & \vdots & \ddots & \vdots \\
  A_{ab}A_{bb}^{n-3} & A_{ab}A_{bb}^{n-4} & \cdots & A_{ab}A_{bb}^{n-2} \\
  \end{bmatrix}
  \]

  and
  \[
  A_{bb} = \text{state matrix, } \alpha_1, \alpha_2, \ldots = \text{coefficients of the desired characteristic eqn.}
  \]

  \[
  \gamma(A_{bb}) = A_{bb}^n + \alpha_1 A_{bb}^{n-1} + \alpha_2 A_{bb}^{n-2} + \cdots + \alpha_n
  \]
Reduced Order Observers: Details

• Problem: Previous form contains derivative of output, i.e. \( \dot{y} \)
  ➢ want to avoid taking derivatives (prone to high frequency noise)
• Solution: Define a new state: \( x_c = \dot{x}_b - Ly \)
  ➢ In terms of this, the reduced order observer equation becomes:
  \[
  \dot{x}_c = (A_{bb} - LA_{ab})\dot{x}_b + (A_{ba} - LA_{aa})y + (B_b - LB_b)u
  \]
  \[
  \dot{x}_c = \underbrace{(A_{bb} - LA_{ab})x_c}_{Ly} + \underbrace{(A_{ba} - LA_{aa} + A_{bb}L - LA_{ab}L)y + (B_b - LB_b)u}_{Bu}
  \]
  ➢ No longer is the derivative of the plant output present!

Reduced Order Observer Block Diagram

Example: Reduced Order Observer

• Compute the reduced order estimator gain which will place the estimator pole at \(-10\omega_n\) given

  \[
  \begin{bmatrix}
  \dot{x}_1 \\
  \dot{x}_2 \\
  \end{bmatrix} = \begin{bmatrix}
  0 & 1 \\
  -\omega_n^2 & 0 \\
  \end{bmatrix} \begin{bmatrix}
  x_1 \\
  x_2 \\
  \end{bmatrix} + \begin{bmatrix}
  1 \\
  0 \\
  \end{bmatrix} u,
  y = [1 \ 0] \begin{bmatrix}
  x_1 \\
  x_2 \\
  \end{bmatrix}
  \]
Example: Reduced Order Observer

Example: Reduced Order Observer

Example: Reduced Order Observer

Example: Reduced Order Observer via Ackermann’s

- Compute the reduced order estimator gain which will place the estimator pole at $-10\omega_n$ given

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} =
\begin{bmatrix}
0 & 1 \\
-\omega_n^2 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} +
\begin{bmatrix}
0 \\
1
\end{bmatrix} u,
\]
\[
y = [1 \ 0]
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
\]
Lecture 15: Compensators

- Objectives are an understanding of:
  - What a compensator is
  - The block diagram of the plant and compensator
  - The transfer function of the compensator
  - The poles of the compensator, and the poles of the entire system
  - Concept of strongly stabilisable

- Reading:
  - Driels, "Linear Control Systems Engineering", Module 25
  - Franklin, Powell & Workman, "Digital Control of Dynamic Systems", 8.3 & 8.4
  - Franklin, Powell & Emani-Naeini, "Feedback Control of Dynamic Systems", 7.6, 7.7 & 7.9

Regulated Control System

- Observer state and output equations are given by:
  \[
  \dot{x} = (A\hat{x} + Bu) - BK\hat{x} + L(y - \hat{y})
  \]
  \[
  \hat{y} = C\hat{x} + D(-K\hat{x})
  \]

- Plant state and output equations with feedback \(K\) are given
  \[
  x = Ax + Bu - BK\hat{x} = (A - BK)x + BKx_e + Bu
  \]
  \[
  y = Cx + Du + D(-K\hat{x}) = (C - DK)x + Du + DKx_e
  \]

Complete Controller

Governing equations:
**Governing equations:**

- The previous equations can be combined to produce the following augmented state equation:

\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_e
\end{bmatrix} =
\begin{bmatrix}
A & -BK \\
LC & A-BK-LC
\end{bmatrix}
\begin{bmatrix}
x \\
x_e
\end{bmatrix} +
\begin{bmatrix}
B \\
0
\end{bmatrix} u
\]

- The output equation remains unchanged:

\[
y = [C \ -DK] \begin{bmatrix} x \\ \hat{x} \end{bmatrix} + Du = Cx - DK \hat{x} + Du
\]

**Separation Principle**

- Characteristic Equation of entire system (controller & plant):

\[
\det(sI - A + BK) \cdot \det(sI - A + LC) = 0
\]

- Therefore, poles_{overall} = poles_{controller} AND poles_{observer}

  ➢ This is referred to as the *Separation Principle.*

**Separation Principle Derivation**

- Subtracting the bottom row from the top row we get an expression in terms of the (states and) state error, \( x_e = x - \hat{x} \):

\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_e
\end{bmatrix} =
\begin{bmatrix}
A - BK & BK \\
0 & A - LC
\end{bmatrix}
\begin{bmatrix}
x \\
x_e
\end{bmatrix} +
\begin{bmatrix}
B \\
0
\end{bmatrix} u
\]

- The output equation remains unchanged:

\[
y = [C - DK \ DK] \begin{bmatrix} x \\ x_e \end{bmatrix} + Du = (C - DK)x + DKx_e + Du
**Regulator Transfer Function**

- The compensator state equation is
  \[
  \dot{x} = (A - BK - LC + LDK)x + Bu + Ly
  \]
  \[
  u = -K \frac{\dot{x}}{c}
  \]

- Compensator Transfer Function:

  \[
  G(s) = \frac{U(s)}{Y(s)} = \frac{-K}{c} \left( sI - \left[ A - BK - LC + LDK \right] \right)^{-1} L
  \]

- Note that we never specified the roots of the above expression.
  This means that the poles of the Compensator may not be stable!

---

**Compensator poles from block diagram**

\[
\text{poles} = \text{eig}(A - BK - LC + LDK)
\]

---

**Derivation of Entire System Characteristic Equation**

---

**Derivation of Entire System Characteristic Equation**
Example: Design of a Regulator

- Consider the SS system

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} = \begin{bmatrix}
-1 & 0 \\
1 & -2
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} + \begin{bmatrix}
1 \\
2
\end{bmatrix} u
\]

\[\begin{bmatrix}
1 \\
1
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}
\]

- Design a regulator to move the closed loop system poles to \(s = -2, -4\). Use an observer with poles at \(s = -4, -8\).
Example: Design of a Regulator

Compensators with Reduced Order Observers

• The control law for a system with a reduced order observer is:

\[ u = -k x = -[k_a \ k_b] \begin{bmatrix} x_a \\ \hat{x}_b \end{bmatrix} = -k_a y - k_b \hat{x}_b \]

• Note: The above expression assumes that the output matrix of the plant is the identity matrix, i.e.,

\[ y = x_a = I x_a \]

Compensators with Reduced Order Observers: State Eqn

• The state equations of the compensator are given by:

- State Eqn: \( \dot{x}_c = A_c x_c + B_c y \)
- Output Eqn: \( u = C_r x_c + D_r y \)

- where

\[
\begin{align*}
A_r &= A_{bb} - L A_{ab} - (B_b - L B_a) k_b \\
B_r &= A_r L + A_{ba} - L A_{aa} - (B_b - L B_a) k_a \\
C_r &= -k_b \\
D_r &= -k_a - k_b L
\end{align*}
\]

- Note: Strictly speaking these equations are for a regulator since there is no command input.

State Eqn for Plant and Reduced Order Observer

• The equations for the plant and observer can be combined to produce the following augmented state equation:

\[
\begin{bmatrix}
\dot{x}_a \\
\dot{x}_b \\
\dot{x}_c
\end{bmatrix} = \begin{bmatrix}
A_{aa} + B_a D_r & A_{ab} & B_a C_r \\
A_{ba} + B_b D_r & A_{bb} & B_b C_r \\
B_r & 0 & A_r
\end{bmatrix} \begin{bmatrix}
x_a \\
x_b \\
x_c
\end{bmatrix} + \begin{bmatrix}
B_a \\
B_b \\
0
\end{bmatrix} u
\]

Known States

Unknown States

Pseudo State Estimates
Reduced Order Observer and Controller Implementation

Compensators

- Summary of steps
  1. Model the system, $A$.
  2. Place the control inputs, check for controllability, $B$.
  3. Derive the control gains, $K$.
  4. Place the sensors, check for observability, $C$.
  5. Design an observer, $L$.

Strongly Stabilisable

- A system is called strongly stabilisable if it can be stabilised by a stable controller
  - Stable controllers are preferable, especially for stable plants, since we want to maintain stability in the event of actuator/sensor failure
- A plant is strongly stabilisable iff it has an even number of real poles between every pair of real zeros in RHP (including $+\infty$)
  - This property is called the parity interlacing property
  - Note complex zeros are not counted
  - Zeros at $+\infty$ are included when the plant is strictly proper

Example 1: Strongly stabilisable

- Example (1): Let $P(s) = \frac{s-1}{s(s-2)}$

- Example (2): Let $P(s) = \frac{(s-1)^2(s^2-s+1)}{(s-2)^2(s+1)^3}$

Strongly Stabilisable

\[ P(s) = \frac{s-1}{s(s-2)} \]
Example 2a: Strongly stabilisable

- Consider the inverted pendulum on a cart system seen previously with only a position sensor, with transfer function given by

\[ P(s) = \frac{Y(s)}{U(s)} = \frac{s^2 - \frac{g}{l}}{M(s^2 - \frac{g}{l} (1 + \frac{m}{M}))s^2} \]

Example 2b: Strongly stabilisable

- Consider the inverted pendulum on a cart system seen previously but now with a sensor that measures the displacement at the bob, with transfer function given by

\[ P(s) = \frac{Z(s)}{U(s)} = \frac{-\frac{g}{l}}{M(s^2 - \frac{g}{l} (1 + \frac{m}{M}))s^2} \]
**Topic 16: Reference Input & Command Tracking**

- Objectives are an understanding of:
  - What reference tracking (setpoint following) is and the need for it
  - How it is achieved
  - State augmentation

- Reading:
  - Driels, "Linear Control Systems Engineering", Module 25
  - Franklin, Powell & Workman, "Digital Control of Dynamic Systems", 8.3 & 8.4
  - Franklin, Powell & Emani-Naeini, "Feedback Control of Dynamic Systems", 7.6, 7.7 & 7.9
  - Kuo, "Automatic Control Systems", Section 10-14
  - Nise, "Control Systems Engineering", Section 12.8

**Command Tracking**

- Up to now we have been concerned with regulator design, where the main criterion is "good" disturbance rejection
- We now want something which can provide "good" command following

**Why?**
- For command tracking (also known as setpoint following)

**How?**
- Introduce a Reference Signal Into the State Space Controller

---

**Command Tracking: Reference Signal**

- Possibility 1: Simply Introduce \( r \):

\[
\begin{align*}
\text{plant} & \xrightarrow{C} y \\
\text{K} & \xrightarrow{\Sigma} r
\end{align*}
\]

- Not good! You can have a non-zero steady-state (ss) error with a step command input

**Command Tracking**

- Two Techniques we will look at:
  - Open Loop
    - Solution is to simply rescale the reference input
    - Is not robust (as plant gain may vary)
  - Closed Loop
    - Add integral controller (like I controller in PID)
    - This adds new states (called augmented states)
Reference Input, Part 1: Full State Feedback (no observer)

- Objective is to have \( y_{ss} = r_{ss} \) at steady-state.
  - Also at steady-state: \( x = x_{ss}, u = u_{ss} \) where \( x_{ss} \) are the desired steady-state states.

- The steady state solution is given by:

\[
\begin{bmatrix}
\dot{x} \\
y
\end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_{ss} \\
u_{ss} \end{bmatrix} \Rightarrow \begin{bmatrix} 0 \\
0 \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x_{ss} \\
u_{ss} \end{bmatrix}
\]

- To solve, let \( x_{ss} = N_x r_{ss}, u_{ss} = N_u r_{ss} \) (where \( N_x, N_u = \text{matrix of gains} \)).

Example: Reference Input – Full State Feedback

- Determine the reference gain for the following closed-loop system

\[
A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}, D = 0 \text{ & } K = \begin{bmatrix} 2 & 2 \end{bmatrix}
\]
Example: Reference Input – Full State Feedback

Reference Input, Part 2: Compensator With Observer

- Several ways to introduce reference input. However, most common is to introduce it in such a way that the observer is unaffected:

  - Equations:
    - Plant: \( \dot{x} = Ax + Bu, \ y = Cx + Du \)
    - Compensator: \( \dot{x} = (A - BK - LC)x + Ly, \ u = -K\dot{x} \)
    - Reference: \( N = N_u + KN_r \) (as before)
    - Input to Output TF:
      \[
      Y(s) = NC[sl - (A - BK)]^{-1}B
      \]
      \[
      R(s)
      \]

Robust tracking using integral control

- The previous reference input method is not robust as it is an open-loop approach
- Only a closed-loop approach will be robust and always remove steady-state errors
- Integral control is used
  - We add new states which are the integral of the output error
  - This is known as state augmentation
Integral Control: State Augmentation:

- Add state(s) which is the integral of the output error:
  \[ x_i = \int (y - r) dt \], or
  \[ x_i = y - r - e = Cx - r \]

- Augmented plant model is:
  \[
  \begin{bmatrix}
  \dot{x} \\
  \dot{x}_i
  \end{bmatrix} =
  \begin{bmatrix}
  A & 0 \\
  C & 0
  \end{bmatrix}
  \begin{bmatrix}
  x \\
  x_i
  \end{bmatrix}
  +
  \begin{bmatrix}
  B \\
  \end{bmatrix}
  u
  \]

- Control law for full-state feedback
  \[
  u = -K_0 \begin{bmatrix} x \end{bmatrix} - K_i \begin{bmatrix} x_i \end{bmatrix}
  \]

where \( K_0 \) are the control gains from the plant, and \( K_i \) are the control gains for the error integral.

Rearranging the previous expression and feeding back the control signal into the augmented plant model gives:

\[
\begin{bmatrix}
  \dot{x} \\
  \dot{x}_i
  \end{bmatrix} =
  \begin{bmatrix}
  A - BK_0 & -BK_i \\
  C - DK_0 & -DK_i
  \end{bmatrix}
  \begin{bmatrix}
  x \\
  x_i
  \end{bmatrix}
  +
  \begin{bmatrix}
  0 \\
  r
  \end{bmatrix}
  \]

Desired outputs become new input

where \( K_0 \) are the control gains from the estimator, and \( K_i \) are the control gains for the error integral.

Integral Control with Full State Feedback

- Control gains \( K = [K_0 \ K_i] \) are calculated using previous techniques.

Example: Integral Control

- Consider a motor-speed system given by \( G(s) = 1/(s+3) \). Design a state controller using integral control such that the poles are \( s = -5, -5 \) rad/s.
Compared to (2) and (1) in terms of coefficients.

Desired...

Taking into account...

Thus, input control

Augmenting model space state to

Converting

The system input is not "u" anymore. It is the reference "r".
Integral Control with Full State Estimation

- Augmented plant model with full state estimator is:

\[
\begin{bmatrix}
    \dot{x} \\
    \dot{x}_i \\
    \dot{\hat{x}}
\end{bmatrix} =
\begin{bmatrix}
    A & 0 & 0 \\
    C & 0 & 0 \\
    LC & 0 & A - LC
\end{bmatrix}
\begin{bmatrix}
    x \\
    x_i \\
    \hat{x}
\end{bmatrix} +
\begin{bmatrix}
    B \\
    D u - l r \\
    B
\end{bmatrix} \begin{bmatrix}
    0
\end{bmatrix}
\]

- Control law for full-state feedback:

\[
u = -\begin{bmatrix} 0 & K_i & K_0 \end{bmatrix} \begin{bmatrix} x \\ x_i \\ \hat{x} \end{bmatrix}
\]

where \( K_0 \) are the control gains from the estimator, and \( K_i \) are the control gains for the error integral.
Integral Control with Full State Estimation

- Rearranging the previous expression and feeding back the control signal into the augmented plant model gives:

\[
\begin{bmatrix}
\dot{x} \\
\dot{x}_f \\
\hat{x}
\end{bmatrix} =
\begin{bmatrix}
A & -BK_f & -BK_0 \\
C & -DK_f & -DK_0 \\
LC & -BK_f & A-LC-BK_0
\end{bmatrix}
\begin{bmatrix}
x \\
x_f \\
\hat{x}
\end{bmatrix}
+ \begin{bmatrix}
0 \\
1 \\
0
\end{bmatrix} r
\]

where \(K_0\) are the control gains from the estimator, and \(K_f\) are the control gains for the error integral.
**Topic 17: Summary**

**Homogeneous Open-Loop**

\[ \dot{x} = Ax \]

**Inhomogeneous Open-Loop**

\[ y = Cx + Du \]

**Closed-Loop using Full-State Feedback**

\[ x = (A - K)x \]

\[ y = Cx + Du - Kx \]
Closed-Loop using Full-State Feedback & Integral Tracking

State Feedback Control & Full-State Estimation

State Feedback Control, Full-State Estimation & Integral Tracking

State Feedback Control & Reduced-Order Estimation
State Feedback Control, Reduced-Order Estimation & Integral Tracking

Bibliography

- Junkins and Kim (1993) "Introduction to Dynamics and Control of Flexible Structures", AIAA Educational Series
- Anderson and Moore (1971) "Linear Optimal Control", Prentice Hall