

ELP7100: Autonomous Systems & Control Lab

Assignment - 1

Semester I 2026-2027

Course Instructors: Subashish Datta Subashish.Datta@ee.iitd.ac.in

Course TAs: Anvisha Chaturvedi, Ashtha Tiwari, Arpita Dey, S.A. Yogeswari

1. Controller Canonical Transformation:

Consider a system:

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

where $x \in \mathbb{R}^n$ is the state vector and $u \in \mathbb{R}^m$ is the control input. Assume that the pair (A, B) is controllable. Transform this system to an equivalent one:

$$\dot{\bar{x}} = \bar{A}\bar{x} + \bar{B}u$$

such that $\bar{A} = PAP^{-1}$, $\bar{B} = PB$ are in controller canonical form.

Steps to obtain $\bar{A}$ & $\bar{B}$:

(a) Compute the controllability matrix:

$$\mathcal{C} = [B \ AB \ A^2B \ \cdots \ A^{n-1}B]$$

(b) Starting from the left and moving towards the right, select m independent columns of $\mathcal{C}$. This is always possible since the pair (A, B) is controllable.

$$\mathcal{C} = [b_1 \ b_2 \ \cdots \ b_m \ | \ Ab_1 \ Ab_2 \ \cdots \ Ab_m \ | \ \cdots \ | \ A^{n-1}b_1 \ A^{n-1}b_2 \ \cdots \ A^{n-1}b_m]$$

(c) Reorder these columns by taking first $b_1, Ab_1, \dots$ until all columns involving b_1 have been used. Then take $b_2, Ab_2, \dots$ till all columns involving b_2 are used and then lastly use $b_m, Ab_m, \dots$

(d) Construct the following matrix:

$$\mathcal{C}' = [b_1 \ Ab_1 \ \cdots \ A^{\mu_1-1}b_1 \ | \ b_2 \ Ab_2 \ \cdots \ A^{\mu_2-1}b_2 \ | \ \cdots \ | \ b_m \ Ab_m \ \cdots \ A^{\mu_m-1}b_m]$$

where μ_i are the controllability indices of the system.

$$\mu := \max_i \mu_i$$

is called the "controllability index" of the system.

(e) Define

$$\sigma_k = \sum_{i=1}^k \mu_i \quad k = 1, 2, \dots, m$$

i.e.

$$\sigma_1 = \mu_1$$

$$\sigma_2 = \mu_1 + \mu_2$$

$$\vdots$$

$$\sigma_m = \mu_1 + \mu_2 + \cdots + \mu_m = n$$

- (f) Compute $[\bar{\mathcal{C}}]^{-1}$, and denote q_k as the σ_k^{th} row of $[\bar{\mathcal{C}}]^{-1}$.
- (g) Construct the transformation matrix P as:

$$\begin{bmatrix}
 \dots & q_1 & \dots \\
 \dots & q_1 A & \dots \\
 \vdots & \vdots & \vdots \\
 \dots & q_1 A^{\mu_1-1} & \dots \\
 \hline
 \dots & q_2 & \dots \\
 \dots & q_2 A & \dots \\
 \vdots & \vdots & \vdots \\
 \dots & q_2 A^{\mu_2-1} & \dots \\
 \hline
 \vdots & \vdots & \vdots \\
 \hline
 \dots & q_m & \dots \\
 \dots & q_m A & \dots \\
 \vdots & \vdots & \vdots \\
 \dots & q_m A^{\mu_m-1} & \dots
 \end{bmatrix}$$

- (h) Finally, compute:

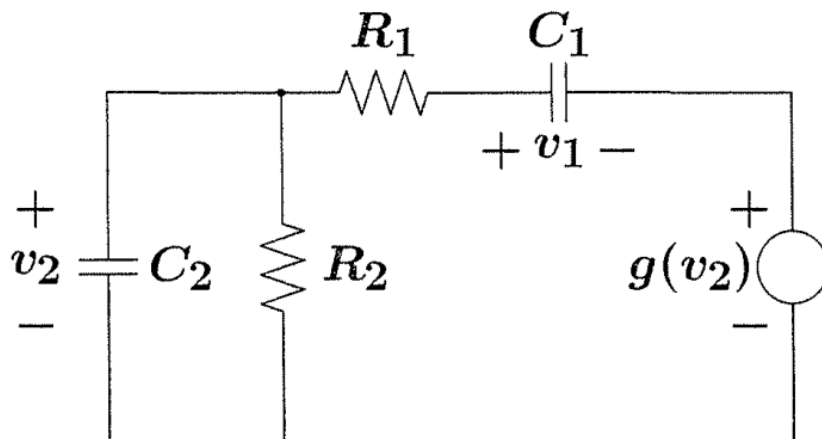
$$\bar{A} = PAP^{-1}, \quad \bar{B} = PB$$

Programming Task: Write code in **MATLAB**, **Scilab**, and **Python** to compute the controller canonical form.

- **Input:** Matrices A, B
- **Output:**
 - (a) Canonical matrices $\bar{A}, \bar{B}$
 - (b) Controllability indices $\mu_1, \dots, \mu_m$

2. Wien-Bridge Oscillator Analysis:

An equivalent circuit of the Wien-Bridge oscillator is shown below:



The non-linear voltage controlled voltage source is given by:

$$g(v_2) = 3.23v_2 - 2.195v_2^3 + 0.666v_2^5$$

Let $x_1 = v_1$ and $x_2 = v_2$. Then, a state-space model of the circuit is:

$$\begin{aligned}\dot{x}_1 &= \frac{1}{C_1 R_1}(-x_1 + x_2 - g(x_2)) \\ \dot{x}_2 &= -\frac{1}{C_2 R_1}(-x_1 + x_2 - g(x_2)) - \frac{1}{C_2 R_2}x_2\end{aligned}$$

Assume the component values: $C_1 = C_2 = R_1 = R_2 = 1$. Then:

- (i) Solve the above set of equations to obtain x_1 and x_2 using numerical **integration** methods. **Do not use any existing ODE solver command.** Let the solutions be $\bar{x}_1$ and $\bar{x}_2$.
- (ii) Solve these equations for x_1 and x_2 using existing commands (e.g., built-in ODE solvers). Plot x_1 and x_2 . Also plot $\bar{x}_1$ and $\bar{x}_2$ (obtained from part (i)). Compare these plots.

Programming Task: For both parts (i) and (ii), use **MATLAB**, **Scilab**, and **Python**.

[Example taken from Book: Nonlinear Systems, by Khalil, Chapter 2]

3. Modeling and Simulation with OpenModelica:

Using **OpenModelica**, perform modeling and simulation of the following systems:

- Simple electric circuits
- Mass-Spring Damper system
- An electro-mechanical system

Analysis Task: Comment on the type of model you have obtained in comparison to the ordinary state-space model.

4. Numerical Integration of a Simple Pendulum

Consider the following model of a simple pendulum:

$$\ddot{\theta} + \sin \theta = 0$$

Define the state variables as $x_1 = \theta$ and $x_2 = \dot{\theta}$. This converts the second-order differential equation into two first-order differential equations:

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\sin(x_1)\end{aligned}$$

We will now explore numerical integration of these equations.

The simplest method is the ****Euler method****, which approximates the derivative as:

$$\dot{x}_i \approx \frac{x_i(t+h) - x_i(t)}{h}$$

where h is a small time step. Using this approximation, we can write the two first-order differential equations as the following difference equations:

$$\begin{aligned}x_1(t+h) &= x_1(t) + hx_2(t) \\ x_2(t+h) &= x_2(t) - h\sin(x_1(t))\end{aligned}$$

These difference equations can be numerically solved for any initial condition $x_1(0)$, $x_2(0)$, over a finite time interval $[0, T]$. We divide the time interval into T/h steps, each of size h . Let $x_i(nh) \equiv x_i[n]$ for $i = 1, 2$ and $n = 0, \dots, T/h$.

We now compute the approximate trajectories for:

- Initial conditions: $x_1(0) = 1, x_2(0) = 0$
- Total time: $T = 10$
- Step sizes: $h = 0.1, 0.01, 0.001$

Observations: As the step size h decreases, the numerical solution becomes more accurate and stable. Larger step sizes may lead to noticeable numerical errors or instability in the solution, especially for nonlinear systems like the pendulum.

Comparison with ODE Solvers: Compare the results obtained from the Euler method with a standard ODE solver such as `ode45` in MATLAB or `scipy.integrate.solve_ivp` in Python. These solvers typically use adaptive step-size Runge–Kutta methods and offer better accuracy and stability than the simple Euler method.

5. Numerical Solution of the 1D Heat Equation Many control applications involve heat transfer. The one-dimensional heat equation is given by

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2}. \quad (1)$$

For the spatial domain $0 \leq x \leq 1$ and time interval $0 \leq t \leq 1$, with initial condition $T(x, 0) = 37$ and boundary conditions $T(0, t) = 25$ and $T(1, t) = 25$, we wish to compute the spatiotemporal temperature profile $T(x, t)$.

Consider the following finite-difference discretization of the heat equation:

$$\frac{1}{k}(T_{i,j+1} - T_{i,j}) = \frac{1}{h^2}(T_{i+1,j} - 2T_{i,j} + T_{i-1,j}), \quad (2)$$

where h is the step size in the spatial (x) direction, k is the step size in the temporal (t) direction, and the notation $T(ih, jk) \equiv T_{i,j}$ is used.

Simulate the system using this scheme for different values of h and k . Observe the evolution of the temperature distribution over time and analyze how the choice of h and k affects numerical accuracy and stability.

Note from TA's

If you faced challenges, don't be discouraged - that's part of the learning process.
Happy learning!