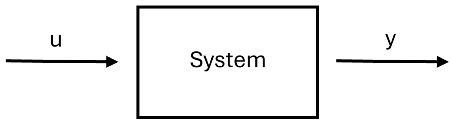
Exercise 2 - Modeling and Control Architectures

1 Modeling

As we said last week, we want to learn how to represent a dynamic control system in such a way that it can be treated effectively using mathematical tools. Precisely, we want to write down a set of equations that describe the system's behavior (**output**) as a function of the **input** and some other parameters.



Remember: All models are wrong, but some are useful!

As a first step, we divide the inputs into two categories:

- **Endogenous inputs**: inputs that we can manipulate, such as control inputs. E.g., the voltage applied to a motor, the force applied to a mass, etc.
- **Exogenous inputs**: inputs that we cannot control, but that affect the system. For example, wind, **disturbances**, etc.

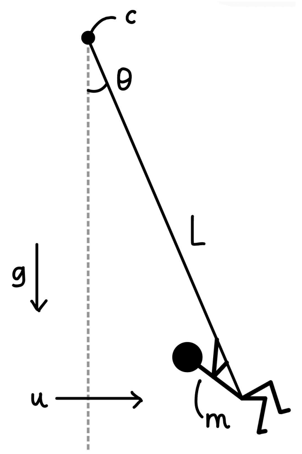
Furthermore, we can classify the outputs into:

- **Measured outputs**: quantities we can measure directly. E.g., the position of a mass, the temperature of a room, etc.
- **Performance outputs**: not directly measurable, but quantities we want to control. E.g., average fuel consumption.

Finally, the features that are specific to the system and don't change over time are called **parameters** (e.g., mass, resistance, etc.). On the other hand, the features that change over time and summarize everything that happens inside the system are called the **state** of the system.

Example 1: Swing/Pendulum (from the lecture)

We have a swing with a mass m and length L. The angle of the swing is $\theta(t)$. On the system acts gravity g and a friction coefficient c. We also apply a force u to the swing.

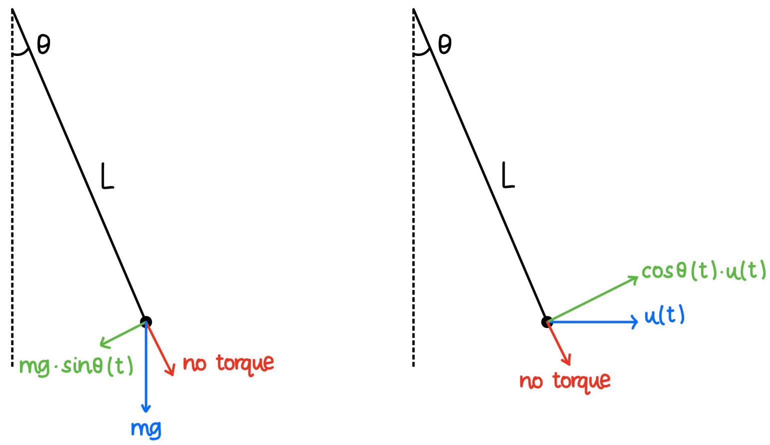


Given the balance of angular momentum (Mechanics III):

$$M_B = I_B \dot{w}$$

Then we can write the equation of motion of the system as:

$$mL^2\ddot{\theta}(t) = -mgL\sin(\theta(t)) - c\dot{\theta}(t) + L\cos(\theta(t))u(t)$$



And we can define the output as the angle of the swing:

$$y(t) = \theta(t)$$

In general, we want to write the equations in the following form:

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

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

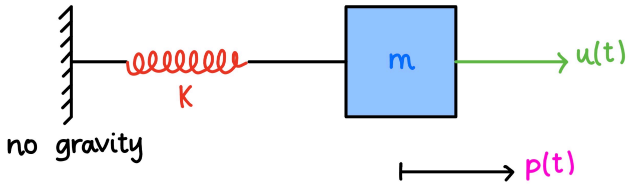
For our example, we can define the state as:

$$
\begin{aligned} x_1(t) &= \theta(t) \\ x_2(t) &= \dot{\theta}(t) \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \dot{x_1}(t) &= x_2(t) \\ mL^2\dot{x_2}(t) &= -mgL\sin(x_1(t)) - cx_2(t) + L\cos(x_1(t))u(t) \\ y(t) &= x_1(t) \end{aligned}
$$

Which is a system of first-order ordinary differential equations (ODEs) in the state space representation.

Example 2: Oscillator

We have a mass m attached to a spring with spring constant k. We apply a force $u(t)$ to the system. The position of the mass is $p(t)$.



The equation of motion is given by Newton's second law and Hooke's law:

$$m\ddot{p}(t) = -kp(t) + u(t)$$

If we look at the 2nd order ODE, we can see that the system is linear. We can rewrite it as a system of first-order ODEs (see Lineare Algerbra II): Susbstituting we obtain:

$$x_1(t) = p(t)$$

$$x_2(t) = \dot{p}(t)$$

Then we can write:

$$\dot{x_1}(t) = x_2(t)$$

$$m\dot{x_2}(t) = -kx_1(t) + u(t)$$

That can also be written in matrix form:

$$\begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u(t)$$

This is something we already know from Linear Algebra II. The only thing left is to define the output. Let's say we want to measure the velocity of the mass:

$$y(t) = \dot{p}(t) = x_2(t)$$

Now we can write the output equation in matrix form as well:

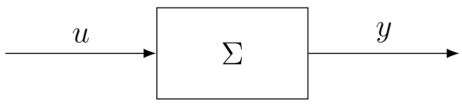
$$y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

The vector $x = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$ contains all state variables of the system. The states describe how a system changes internally over time. It can be seen as a memory, containing a summary of how the system behaved in the past.
Given the internal states and the current input, we can uniquely predict any future behavior.
This system of equations is called the **state space representation** of the system:

$$\boxed{\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \\ y(t) &= h(x(t), u(t)) \end{aligned}}$$

2 Control Architectures

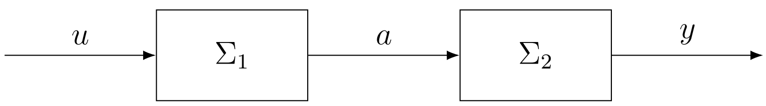
Block diagrams are a graphical representation of a system. They consist of blocks, which represent mathematical operations or functions, and arrows, which indicate the flow of signals between the blocks.
Let's start with an input-output model:



Here Σ maps the input u to the output y. We can write this as:

$$\boxed{y = \Sigma u}$$

We can also have systems that are connected in series or parallel. For example, if we have two systems Σ_1 and Σ_2 connected in series, we can represent it as:



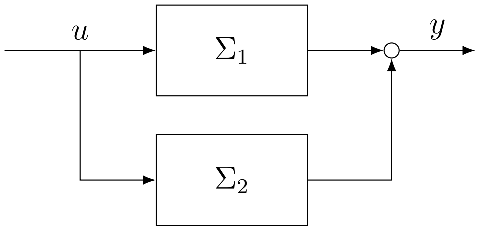
In this case, we can define an intermediate signal a such that:

$$y = \Sigma_2 a$$
$$a = \Sigma_1 u$$

Combining the two, we can write:

$$\boxed{y = \Sigma_2 \Sigma_1 u}$$

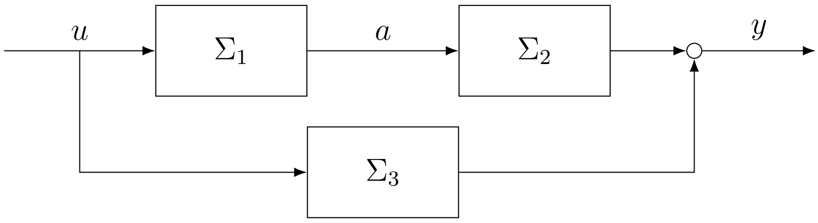
Furthermore, we can have two systems in parallel, as shown below:



Here, we can write:

$$\boxed{y = \Sigma_1 u + \Sigma_2 u = (\Sigma_1 + \Sigma_2)u}$$

If we combine both series and parallel connections, we can have something like this:

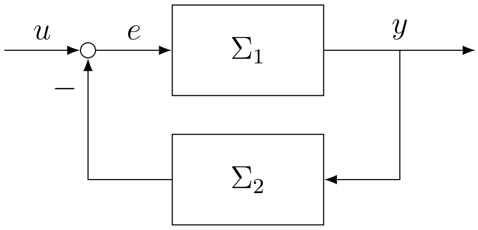


$$y = \Sigma_2 a + \Sigma_3 u$$
$$a = \Sigma_1 u$$

In the end we get:

$$\boxed{y = (\Sigma_2\Sigma_1 + \Sigma_3)u}$$

Finally, we can also have (negative) feedback:



Following the same procedure as before, we can write:

$$y = \Sigma_1 e$$
$$e = u - \Sigma_2 y$$
$$y = \Sigma_1(u - \Sigma_2 y) = \Sigma_1 u - \Sigma_1\Sigma_2 y$$

Rearranging the terms, we get:

$$y + \Sigma_1\Sigma_2 y = \Sigma_1 u$$

And finally:

$$\boxed{y = \tfrac{\Sigma_1}{1+\Sigma_1\Sigma_2}u}$$

This only if Σ_1 and Σ_2 are scalars! If they are not, we need to use the matrix inversion.

Exam Problem: HS23

Problem: Consider the interconnected system shown in Figure 1. The input-output relation for each system Σ_i is given by $y_i = \Sigma_i u_i$, that is, each of the systems Σ_i represents a simple scalar gain.

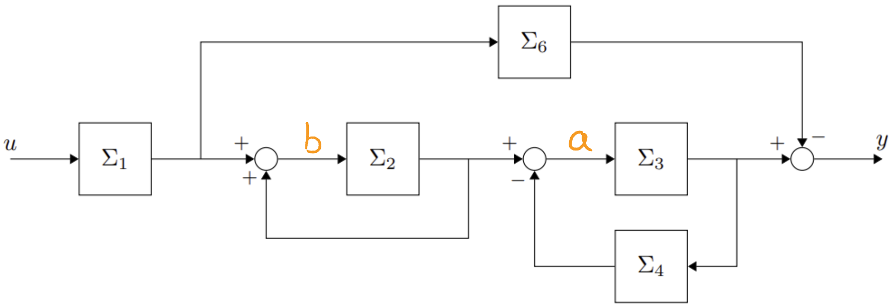


Figure 1: Interconnected system.

Q4 **(1.5 Points)** Derive the transfer function Σ from u to y for the interconnected system shown in Figure 1. **Simplify the result as much as possible.**

Let's do it together:

$$Y = a\Sigma_3 - \Sigma_1\Sigma_6 u$$

$$a = b\Sigma_2 - a\Sigma_3\Sigma_4$$

$$b = b\Sigma_2 + \Sigma_1 u \iff b(1-\Sigma_2) = \Sigma_1 u$$

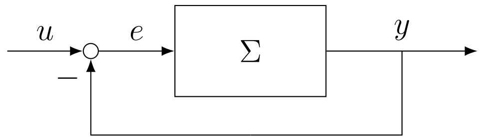
$$b = \frac{\Sigma_1}{1-\Sigma_2} u$$

$$a(1+\Sigma_3\Sigma_4) = \frac{\Sigma_1\Sigma_2}{1-\Sigma_2} u$$

$$a = \frac{\Sigma_1\Sigma_2}{(1+\Sigma_3\Sigma_4)(1-\Sigma_2)} u$$

$$Y = \frac{\Sigma_1\Sigma_2\Sigma_3}{(1+\Sigma_3\Sigma_4)(1-\Sigma_2)} u - \Sigma_1\Sigma_6 u = u\left(\frac{\Sigma_1\Sigma_2\Sigma_3}{(1+\Sigma_3\Sigma_4)(1-\Sigma_2)} - \Sigma_1\Sigma_6\right)$$

Let's go back and look quickly at a simple feedback system with Σ being a scalar gain.



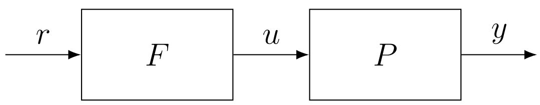
The input-output relation is:

$$y = \frac{\Sigma}{1+\Sigma} u$$

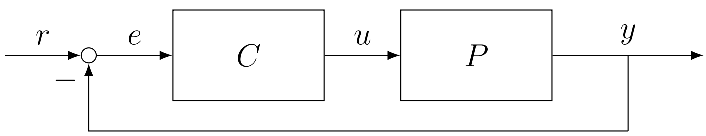
Let's analyze what happens when we choose Σ to be -1.
The output y will go to infinity, independent of the input u! This is called **instability**. Means we have to be careful when designing feedback systems, or our system might become unstable.

In conclusion, we can watch at some basic control architectures:

- **Feed-forward**: relies on a precise knowledge of the plant.



- **Feedback**: allows to stabilize unstable plants, reject disturbances and handle uncertainties. But can introduce **instability**.



- **Two degrees of freedom**: better transient behavior, good tracking of rapidly changing references.