CHEM-E7225 Advanced Process Control D — Winter 2026

Lecture 02 – Dynamic Systems: Models and simulation

Otacilio "Minho" Neto (otacilio.bneto@pm.me)

Dynamical systems are typically represented by continuous-time differential equations. These equations might arise from first-principles modelling, in which the dynamical behavior of a system is described using mass/energy/momentum conservation laws. In the digital era, the numerical routines (or algorithms) used to simulate and control dynamical systems are in discrete-time. For this reason, we are interested in approaches to convert *continuous-time* models into *discrete-time* models that can then be used in digital feedback controllers.

In the following, we briefly overview dynamical models, their properties, and numerical methods for their simulation. The focus is on deterministic systems and we omit the noise/disturbances–they will be discussed later in the course.

1 Nonlinear dynamical systems

A controlled dynamical system can be represented in general by a nonlinear ordinary differential equation (ODE),

$$\tfrac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), u(t)) \tag{1}$$

where $x(t) \in \mathbb{R}^{N_x}$ is the *state* and $u(t) \in \mathbb{R}^{N_u}$ are the *inputs (or controls)* deployed to the system, at time $t \in \mathbb{R}_{\geq 0}$. The function $f(\cdot)$ describes how the state changes, given its current value and the value of the control inputs; formally, it is a mapping $f : \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \to \mathbb{R}^{N_x}$. Since f itself does not depend on time, it defines a *time-invariant* model (as opposed to a *time-varying* model, when this dependency is explicit). We are interested in using this mapping to predict the state evolution $x(t)$ over some time interval $t \in [0, T]$, given the inputs $u(t)$ applied during this interval.

Example 1.1 ***An exothermic CSTR with constant volume [1]***

A CSTR of constant volume implements the exothermic reaction $A \xrightarrow{\kappa(T)} B$ in liquid phase (Figure 1).

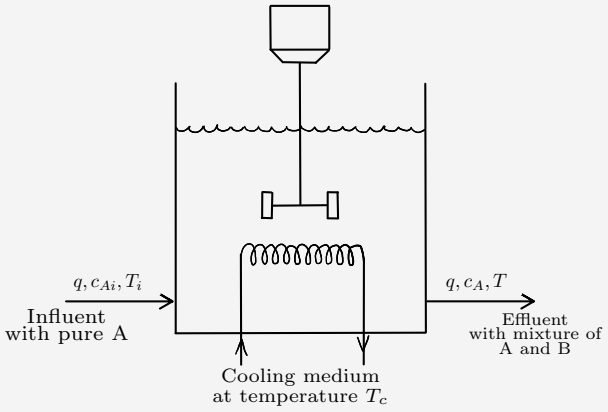

Figure 1. Schematic of the exothermic CSTR.

Using mass balance equations, the dynamics of this reactor are modelled as

$$V\,\tfrac{\mathrm{d}}{\mathrm{d}t}c_A = q(c_{Ai} - c_A) - V\kappa(T)c_A \tag{2a}$$

$$V\rho C\,\tfrac{\mathrm{d}}{\mathrm{d}t}T = \rho C q(T_i - T) + V(-\Delta H_R)\kappa(T)c_A + UA(T_c - T) \tag{2b}$$

with the reaction rate following Arrhenius law, $\kappa(T) = \kappa_0 \exp(-\frac{E}{RT})$. The terms $(V, c_{Ai}, T_i, k_0, \frac{E}{R}, -\Delta H_R, C, \rho, UA)$ are constant model parameters (see [1, §2.4.6] for their values).

In control notation, we let $x = (c_A, T)$ and $u = (q, T_c)$, so that Eq. (2a)–(2b) can be written in the form $\frac{\mathrm{d}}{\mathrm{d}t}x = f(x, u)$. Using this model, we want to simulate the response of this system to changes to its inputs.

1.1 Simulation – Initial value problems and transition maps

The task of simulating a dynamical system can be formally expressed as an *initial value problem* (IVP) of the form

$$\text{Find } x(t) \text{ such that } \begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), u(t)), & t \in [0, T], \\ x(0) = x_0. \end{cases} \tag{3}$$

The solution to the above IVP is the state signal restricted to the desired time interval, $x : [0, T] \to \mathbb{R}^{N_x}$. We note that $f(\cdot)$ must be Lipschitz continuous for Problem (3) to have an unique solution (see the Picard-Lindelöf theorem [2, §8.2]). While we don't study discontinuous systems in this course, it is important to be aware of this condition.

Example 1.2 ***An exothermic CSTR with constant volume (part 2)***

We simulate the Exothermic CSTR from before by solving an IVP given an initial state $x(0) = (0.5,\ 350)$ and the input signals $u_1(t) = 100 + 12\,\mathrm{sign}(t - 6)$ and $u_2(t) = 305 - 5\texttt{sign}(t - 18)$. The results (Figure 2) show that even a small change in the reactor temperature can cause a nonlinear state response.

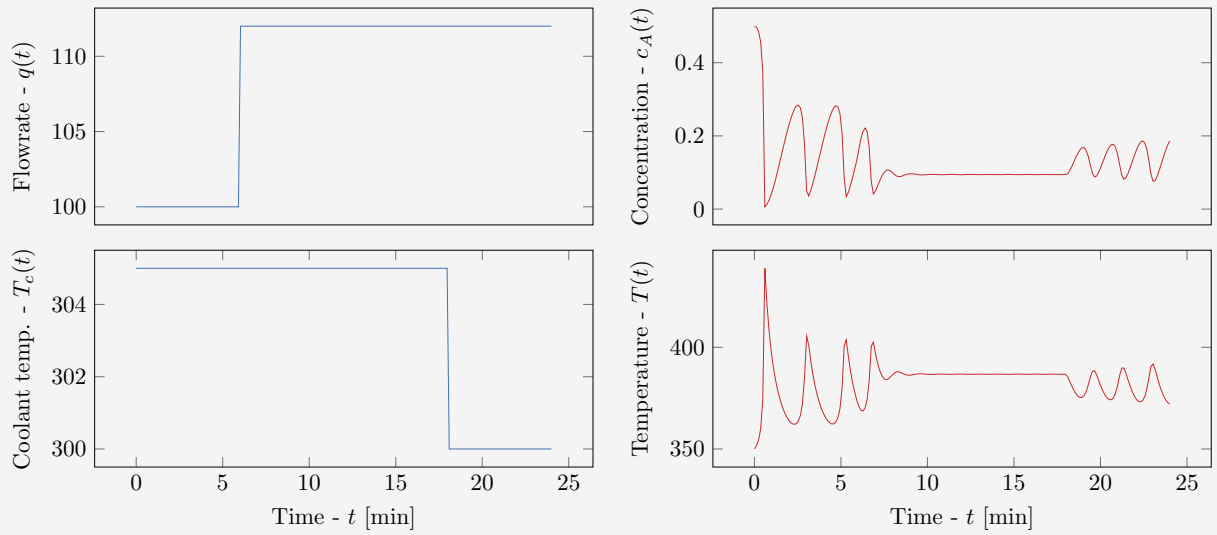

Figure 2. Simulation of the exothermic CSTR for step changes in its inputs.

Discrete-time nonlinear systems

In digital process control, the inputs are a *piecewise constant signal* with a frequency $1/\Delta t$ (see Figure 3). Instead of computing the whole continuous trajectory $x(t)$, $t \in [0, T]$, we are more interested in knowing the state at the *timesteps* in which the input changes. Specifically, we want to compute $x(t)$ for timesteps $t \in [\Delta t, 2\Delta t, \ldots, \lfloor T/\Delta t \rfloor]$. Let $x[n] = x(n\Delta t)$ denote the value of the state at each timestep (similarly, $u[n] = u(n\Delta t)$). Due to causality, and the input being piecewise constant, the IVP (3) can be decomposed into $N = \lfloor T/\Delta t \rfloor$ subproblems of the form

$$\text{Find } x[n{+}1] \text{ such that } \begin{cases} \frac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), u[n]), & t \in [n\Delta t,\ (n{+}1)\Delta t], \\ x[n] = x_n, \end{cases} \tag{4}$$

for each timestep $n = 1, 2, \ldots, N$. A solution to Problem (4), if it exists, is structurally identical for each nth timestep. This allows us to formulate a discrete-time version of the dynamical model in Eq. (1),

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), u(t)) \quad \xrightarrow[\text{w/ period } \Delta t]{\text{Discretization}} \quad x[n{+}1] = f_{\Delta t}(x[n], u[n]), \tag{5}$$

where the *transition mapping* $f_{\Delta t} : \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \to \mathbb{R}^{N_x}$ corresponds to a formula/algorithm that solves Problem (4). The simulation of a discrete-time model is easy: we recursively compute $x[n{+}1] = f_{\Delta t}(x[n], u[n])$, for $n = 0, 1, \ldots, N$, starting from an initial state $x[0] = x_0$. Explicitly, we can represent this routine using a *solution map* $x[n] = f^n_{\Delta t}(x_0, \mathbf{u}_n)$, interpreted as n recursive applications of $f_{\Delta t}(\cdot)$ given x_0 and an action sequence $\mathbf{u}_n = (u[0], u[1], \ldots, u[n{-}1])$.

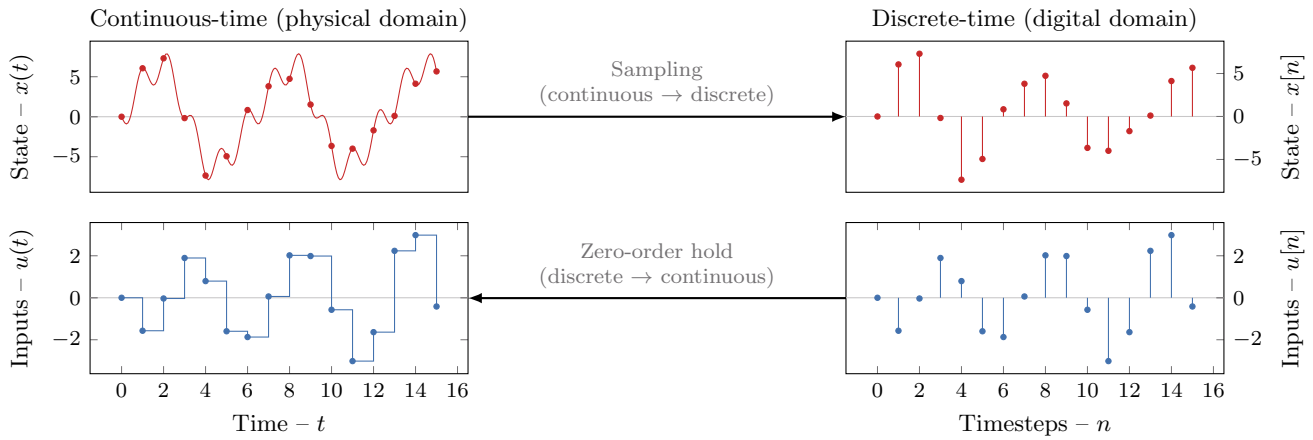

Figure 3. The physical domain is modelled in continuous-time, but digital controllers operate in discrete-time. In practice, we *sample* the continuous-time state signal $x(\cdot)$ to obtain discrete-time data $x[\cdot]$ which is fed to the controller. The controller computes discrete-time inputs $u[\cdot]$ which can be translated back into continuous-time $u(\cdot)$ using a zero-order hold. In the above illustration, $\Delta t = 1$ [units-of-time].

In the next lectures, we will work exclusively in the digital domain–but you should always remember that these models are obtained by discretizing a continuous-time model. For nonlinear systems, there is no general analytical formula for the transition function $f_{\Delta t}$ (that is, a solution to the IVP (4)); so we'll proceed using a numerical approach.

Numerical integration methods

Considering the ODE in Problem (4) with $t_n = n\Delta t$, the transition function $f_{\Delta t}(\cdot)$ has the general form

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), u[n]) \quad \Rightarrow \quad \int_{t_n}^{t_{n+1}} \frac{\mathrm{d}}{\mathrm{d}t}x(t)dt = \int_{t_n}^{t_{n+1}} f(x(t), u[n])dt \tag{6a}$$

$$\Rightarrow \quad x(t_{n+1}) - x(t_n) = \int_{t_n}^{t_{n+1}} f(x(t), u[n])dt \tag{6b}$$

$$\Rightarrow \quad x[n+1] = \underbrace{x[n] + \int_0^{\Delta t} f(x(t), u[n])dt}_{f_{\Delta t}(x[n], u[n])} \tag{6c}$$

where we use the time-invariance of $f(\cdot)$ to change the integration limits. The transition function $f_{\Delta t}(\cdot)$ is computed by numerically approximating the integral in Eq. (6c). For this reason, a formula/routine to approximate $f_{\Delta t}(\cdot)$ is often called an *integrator* or *integration method*. In this course, we consider two very popular methods:

- The *Explicit Euler* (EE) method leads to the discrete-time nonlinear system

$$x[n+1] \approx x[n] + f(x[n], u[n])\Delta t. \tag{7}$$

  That is, it approximates the integral in Eq. (6c) with a rectangle of height $f(x[n], u[n])$ and length Δt. Clearly, this is a very crude approximation; in fact, the EE is a *first-order* method with a global error of order $\mathcal{O}(\Delta t)$. Regardless, it is still an important integrator for its simplicity and for being the basis for more complex methods.

- The *Runge-Kutta of 4th-Order* (RK4) method leads to the discrete-time nonlinear system

$$x[n+1] \approx x[n] + \frac{\Delta t}{6}\big(\kappa_1 + 2\kappa_2 + 2\kappa_3 + \kappa_4\big), \quad \text{with} \quad \begin{cases} \kappa_1 = f(x[n], u[n]), \\ \kappa_2 = f(x[n] + \kappa_1 \frac{\Delta t}{2}, u[n]), \\ \kappa_3 = f(x[n] + \kappa_2 \frac{\Delta t}{2}, u[n]), \\ \kappa_4 = f(x[n] + \kappa_3 \Delta t, u[n]). \end{cases} \tag{8}$$

  In this case, the integral in Eq. (6c) is approximated by a weighted average of four auxiliary increments. The RK4 is more competitive than the EE, while still being computationally cheap. It is a fourth-order method, that is, it achieves a global approximation error of order $\mathcal{O}(\Delta t^4)$. For this reason, this integrator is widely used in practice and usually the first method practitioners try before moving to more complex alternatives.

2 Linear dynamical systems

A fundamental class of systems are those whose dynamics can be represented by time-invariant affine ODEs,

$$\tfrac{\mathrm{d}}{\mathrm{dt}}x(t) = Ax(t) + Bu(t) + c \tag{9}$$

where $A \in \mathbb{R}^{N_x \times N_x}$ and $B_u \in \mathbb{R}^{N_x \times N_u}$ are the state-to-state and input-to-stat matrices, respectively, and $c \in \mathbb{R}^{N_x}$ is the affine term. We focus on the case of $c = 0$, when these become *linear time-invariant* (LTI) systems. These models have a general, intuitive, interpretation: considering the entries in the vectors/matrices from Eq. (9),

$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{dt}}x_1(t) \\ \frac{\mathrm{d}}{\mathrm{dt}}x_2(t) \\ \vdots \\ \frac{\mathrm{d}}{\mathrm{dt}}x_{N_x}(t) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N_x} \\ a_{21} & a_{22} & \cdots & a_{2N_x} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N_x1} & a_{N_x2} & \cdots & a_{N_xN_x} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_{N_x}(t) \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1N_u} \\ b_{21} & b_{22} & \cdots & b_{2N_u} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N_x1} & b_{N_x2} & \cdots & b_{N_xN_u} \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_{N_u}(t) \end{bmatrix} \tag{10}$$

the rate of change of any state component corresponds to a linear combination of all state and input components, that is, $\frac{\mathrm{d}}{\mathrm{dt}}x_i(t) = \sum_j a_{ij}x_j(t) + \sum_j b_{ij}u_j(t)$. The coefficient a_{ij} (resp. b_{ij}) quantifies how the state component $x_j(\cdot)$ (input component $u_j(\cdot)$) affects $x_i(\cdot)$. If the coefficient $a_{ij} = 0$ and/or $b_{ij} = 0$, then these components do not interact.

2.1 Discrete-time linear dynamical systems

The initial value problem associated with a linear time-invariant problem,

$$\text{Find } x(t) \text{ such that } \begin{cases} \frac{\mathrm{d}}{\mathrm{dt}}x(t) = Ax(t) + Bu(t), & t \in [0, T], \\ x(0) = x_0. \end{cases} \tag{11}$$

has the well-known analytical solution

$$x(t) = \exp(At)x_0 + \int_0^t \exp(A(t-\tau))Bu(\tau)d\tau, \qquad \forall t \in \mathbb{R}\,. \tag{12}$$

Note that the second term in the solution is the *convolution* $\Phi * u(t)$ with $\Phi(t) = \exp(At)B$ being the *impulse response* of the system. From Eq. (12), a discrete-time dynamical model can be obtained directly as

$$x[n+1] = A_{\Delta t}x[n] + B_{\Delta t}u[n], \tag{13}$$

with $A_{\Delta t} = \exp(A\Delta t)$ and $B_{\Delta t} = \int_0^{\Delta t} \Phi(\Delta t - \tau)d\tau$, which can be computed *offline* (see Appendix A for a trick). Therefore, the transition map of linear systems is analytical, $f_{\Delta t}(x[n], u[n]) = A_{\Delta t}x[n] + B_{\Delta t}u[n]$. Note that the time discretization of Eq. (9) is also a linear system, that is, its properties are fully determined by $(A_{\Delta t}, B_{\Delta t})$. Finally, linear systems also have an analytical solution map,

$$f^n_{\Delta t}(x_0, \mathbf{u}_n) = A^n_{\Delta t}x_0 + \sum_{k=0}^{n-1} A_{\Delta t}^{(n-1)-k} B_{\Delta t}u[k], \qquad \forall k \in \mathbb{N}, \tag{14}$$

which is easily obtained by applying $f_{\Delta t}(\cdot)$ recursively. (Also, note the similarity between Eq. (12) and Eq. (14)).

2.2 Linearization of nonlinear systems

The dynamics of a nonlinear system in the vicinity of an operating point $p_{\text{ref}} = (x_{\text{ref}}, u_{\text{ref}})$ can be approximated by an affine model. Using the Taylor series expansion of $f(\cdot)$ at the point $(x(t), u(t)) = (x_{\text{ref}}, u_{\text{ref}})$, we have that

$$\tfrac{\mathrm{d}}{\mathrm{dt}}x(t) \approx f(x_{\text{ref}}, u_{\text{ref}}) + \frac{\partial}{\partial x}f(x_{\text{ref}}, u_{\text{ref}})(x(t) - x_{\text{ref}}) + \frac{\partial}{\partial u}f(x_{\text{ref}}, u_{\text{ref}})(u(t) - u_{\text{ref}}), \tag{15}$$

where we have truncated the higher-order terms $\mathcal{O}((x(t)-x_{\text{ref}})^2)$ and $\mathcal{O}((u(t)-u_{\text{ref}})^2)$. The matrices $\frac{\partial}{\partial x}f(\cdot) \in \mathbb{R}^{N_x \times N_x}$ and $\frac{\partial}{\partial u}f(\cdot) \in \mathbb{R}^{N_x \times N_u}$ are the Jacobian matrices with entries (the arguments are omitted to simplify notation)

$$\frac{\partial}{\partial x}f = \begin{bmatrix} \frac{\partial}{\partial x_1}f_1 & \frac{\partial}{\partial x_2}f_1 & \cdots & \frac{\partial}{\partial x_{N_x}}f_1 \\ \frac{\partial}{\partial x_1}f_2 & \frac{\partial}{\partial x_2}f_2 & \cdots & \frac{\partial}{\partial x_{N_x}}f_2 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_1}f_{N_x} & \frac{\partial}{\partial x_2}f_{N_x} & \cdots & \frac{\partial}{\partial x_{N_x}}f_{N_x} \end{bmatrix} \quad \text{and} \quad \frac{\partial}{\partial u}f = \begin{bmatrix} \frac{\partial}{\partial u_1}f_1 & \frac{\partial}{\partial u_2}f_1 & \cdots & \frac{\partial}{\partial u_{N_u}}f_1 \\ \frac{\partial}{\partial u_1}f_2 & \frac{\partial}{\partial u_2}f_2 & \cdots & \frac{\partial}{\partial u_{N_u}}f_2 \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial}{\partial u_1}f_{N_x} & \frac{\partial}{\partial u_2}f_{N_x} & \cdots & \frac{\partial}{\partial u_{N_u}}f_{N_x} \end{bmatrix} \tag{16}$$

Note that the higher-order terms we ignored are small if the distances $\|x(t) - x_{\text{ref}}\|$ and $\|u(t) - u_{\text{ref}}\|$ are also small. In other words, the linear terms dominate the dynamics $f(\cdot)$ when the state-input pair $(x(t), u(t))$ is close to the operating point $(x_{\text{ref}}, u_{\text{ref}})$. The *linearization* (or, actually, *affinization*) of the nonlinear dynamics can thus represented as

The Jacobian matrices $A = \frac{\partial}{\partial x} f(\cdot)$ and $B = \frac{\partial}{\partial u} f(\cdot)$ can be computed efficiently and accurately via automatic differentiation, even for large-scale systems with state dimension $N_x > 1000$.

$$\frac{\mathrm{d}}{\mathrm{d}t} x(t) = f(x(t), u(t)) \xrightarrow[\text{at } (x_{\text{ref}}, u_{\text{ref}})]{\text{Linearization}} \frac{\mathrm{d}}{\mathrm{d}t} x^{\delta}(t) = A x^{\delta}(t) + B u^{\delta}(t) + c \qquad (17)$$

given $A = \frac{\partial}{\partial x} f(x_{\text{ref}}, u_{\text{ref}})$, $B = \frac{\partial}{\partial u} f(x_{\text{ref}}, u_{\text{ref}})$, and $c = f(x_{\text{ref}}, u_{\text{ref}})$, and *deviation signals* $x^{\delta}(t) = x(t) - x_{\text{ref}}$ and $u^{\delta}(t) = u(t) - u_{\text{ref}}$. Using the deviation signals, we say that this is a valid approximation of the original dynamics when both $\|x^{\delta}(t)\|$ and $\|u^{\delta}(t)\|$ are "sufficiently" small. Note that ODE (17) is given in terms of the perturbations $x^{\delta}(\cdot)$, not in terms of the original state $x(\cdot)$; nevertheless, they are related by $x(t) = x^{\delta}(t) + x_{\text{ref}}$ (similarly, $u(t) = u^{\delta}(t) + u_{\text{ref}}$). Interestingly, this model has the interpretation that the coefficient $a_{ij} = \frac{\partial}{\partial x_j} f_i(\cdot)$ (resp. $b_{ij} = \frac{\partial}{\partial u_j} f_i(\cdot)$) quantifies how perturbations to the jth state component $x_j^{\delta}(\cdot)$ (input component $u_j^{\delta}(\cdot)$) affect the ith state component $x_i^{\delta}(\cdot)$.

3 Stability

In the following, for simplicity, we restrict ourselves to autonomous systems whose models are the form $\frac{\mathrm{d}}{\mathrm{d}t} x(t) = f(x(t))$. Specifically, let us assume that there is a controller implementing a policy $u(t) = K(x(t))$ such that the inputs have been 'incorporated' to the model, that is, $f(x) = \widetilde{f}(x, K(x))$ given a controlled system with model $\widetilde{f}(x, u)$.

A *steady state* or *equilibrium point* is a state x_{ss} at which the system will remain (that is, $x(t) = x_{\text{ss}}$ for all $t > 0$) after it has been reached ($x(0) = x_{\text{ss}}$). Consequently, it is a point at which the (closed-loop) dynamics vanish,

$$f(x_{\text{ss}}) = 0 \quad \Longleftrightarrow \quad \frac{\mathrm{d}}{\mathrm{d}t} x(t) = 0. \qquad (18)$$

We are often interested in operating a process around a specified equilibrium point. Therefore, we want to determine if a particular x_{ref} is *stable*, in that sense that the controlled system converges back to it (or remains at its vicinity) if it is pushed away (e.g., by some external disturbance). If so, we might say that the system self-regulates back to x_{ss}. The region of the state-space from which the system can self-regulate back to x_{ss} is called its *region of attraction*.

Example 3.1 ***Stability of second-order systems***

The stability properties of dynamical systems are perhaps better visualized for two-dimensional systems. Let us consider the controlled 2^{nd}-order damped harmonic oscillator, which can be modelled as

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\sin(x_1) - 0.5 x_2 + K(x) \end{bmatrix}. \qquad (19)$$

Using a control law $u = K(x) = 0$, its *phase portrait* (a.k.a., vector field) is plotted in Figure 4, alongside two simulations from different initial states. Black circles (filled or not) indicate the equilibrium points.

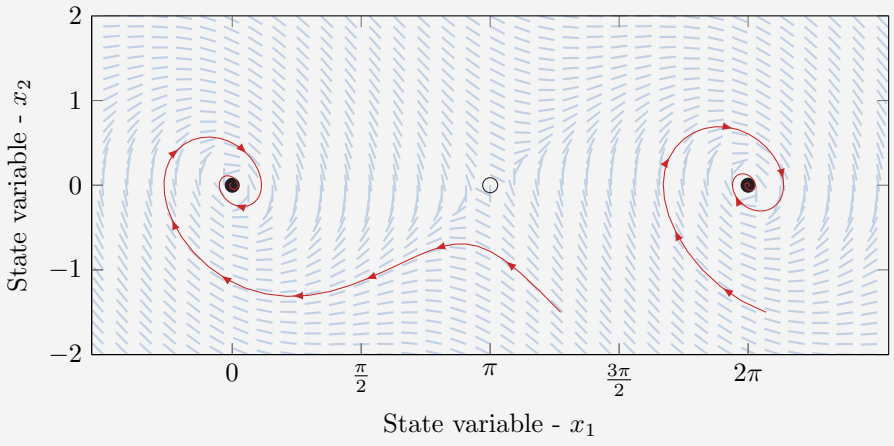

Figure 4. Phase portrait of a 2^{nd}-order system with constant control $K(x) = 0$.

The visualization shows that, the system might converge to $(x_1, x_2) = (0, 0)$ or $(x_1, x_2) = (2\pi, 0)$, depending on the initial state. These equilibria, marked with filled circles, are *asymptotically stable*: they are both stationary and attractive [3]. Conversely, the system can never reach the point $(x_1, x_2) = (\pi, 0)$. These equilibria, marked with open circles, are *unstable*: they repel the state, preventing the system to remain at its vicinity. The attractive and repulsive properties of these points are also visible on the vector field itself.

Lyapunov's linearization method

For linear systems $f(\cdot) = Ax(t)$, stability analysis is simple. If the system is nonsingular (i.e., A is full-rank), then the origin $x_{\text{ss}} = 0$ is the only equilibrium point–otherwise, there are infinitely many equilibrium points, all contained in the subspace $\text{null}(A) = \{x_{\text{ss}} \mid Ax_{\text{ss}} = 0\}$. The stability of the origin can be easily determined using the formula for the response of linear systems, $x(t) = \exp(At)x(0)$, for any arbitrary initial state $x(0) \in \mathbb{R}^{N_x}$. The matrix exponential decays, leading to $x(t) = \exp(At)x(0) \to 0$ when $t \to \infty$, iff the eigenvalues of A all have negative real parts. If at least one eigenvalue has a positive real part, then the system is unstable. Formally, we can state that

$$\text{Re}(\lambda_i) < 0, \text{ for all } \lambda_1, \ldots, \lambda_{N_x} \iff \text{The system } \tfrac{\text{d}}{\text{dt}}x(t) = Ax(t) \text{ is exponentially stable (w.r.t the origin)}$$

where $\lambda_1, \ldots, \lambda_{N_x}$ are the eigenvalues of the matrix A.

For general nonlinear systems, we can always make the change of variable $x^\delta(t) = x(t) - x_{\text{ss}}$ and discuss the stability of the equivalent system $\widetilde{f}(x^\delta(t)) = f(x(t) - x_{\text{ss}})$ with respect to the origin $x^\delta_{\text{ss}} = 0$. Thus, we will save some words and instead write "*the stability of the system*" instead of "*the stability of the system w.r.t. the origin*". While stability analysis of nonlinear system is more involving, the *Lyapunov's indirect method* [3] establishes that

$$\text{The linearization } \tfrac{\text{d}}{\text{dt}}x^\delta(t) = Ax^\delta(t) \text{ is exponentially stable (w.r.t the origin)}$$
$$\downarrow$$
$$\text{The equilibrium } x_{\text{ss}} \text{ is locally asymptotically stable for } \tfrac{\text{d}}{\text{dt}}x(t) = f(x(t))$$

This is an outstanding fact, as it provides a mathematical argument in favour of controlling nonlinear systems using simplified, linearized, models. Specifically, to control a nonlinear system, we (i) choose a desired steady state x_{ss}, (ii) linearize $f(\cdot)$ around x_{ss}, (iii) design a control policy based on the linearized model, then (iv) deploy the controller to the real system. If the controller stabilizes the linear model, then it will also stabilize the actual nonlinear system around the equilibrium point–as long as the noise/disturbances don't push it outside its region of attraction. For this reason, we will focus on designing controllers based on linear(ized) models. Lyapunov shall protect us!

Linear control still has limitations: sometimes we cannot guarantee that the closed-loop system reaches (and remains) at the desired region of attraction. In such cases, we need nonlinear control methods.

Example 3.2 ***Stability of second-order systems (part 2)***

A powerful application of feedback is its ability to change the stability properties of a state-space. For instance, applying the control law $u = K(x) = \frac{\pi}{2} + \sin(x_1) - 0.5x_1$ leads to the phase portrait in Figure 5.

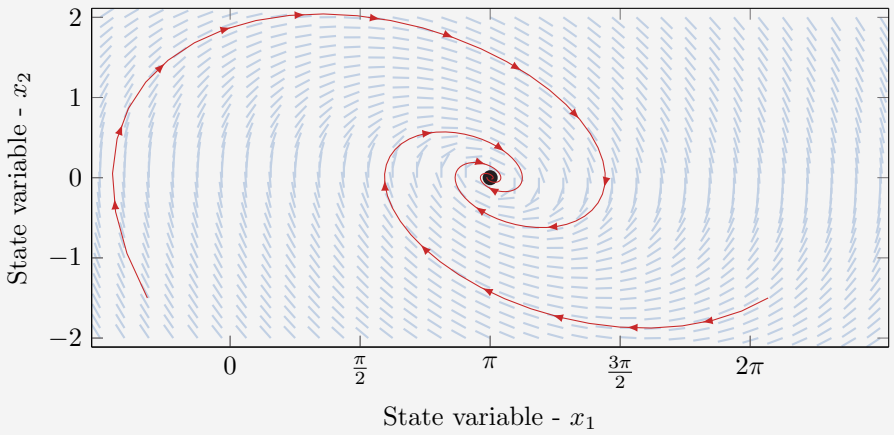

Figure 5. Phase portrait of a 2nd-order system with $K(x) = \frac{1}{\pi} + \sin(x_1) - 0.5x_2$.

The closed-loop system now has the unique equilibrium $(x_1, x_2) = (\pi, 0)$, which is stable. Surprisingly, this equilibrium was unstable when we the system was subject to a constant input (without feedback). Therefore, feedback controllers have the power of *stabilizing* a dynamical system even to (open-loop) unstable steady-states. Finally, if we plug this control law into the model (Eq. 19) it is possible to attest the stability of the (closed-loop) system by linearizing the system around this new equilibrium point.

References

[1] D. E. Seborg et al. *Process dynamics and control*. John Wiley & Sons, 2016.

[2] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model Predictive Control: Theory, Computation and Design*. 2nd ed. Nob Hill Publ., LLC., 2020. ISBN: 978-0-9759377-5-4.

[3] J.-J. E. Slotine. "Applied nonlinear control". In: *Prentice Hall* 2 (1991).

A Tips & Tricks – Linear systems

Converting affine systems to linear systems

The affine system representation $\frac{\mathrm{d}}{\mathrm{d}t}x(t) = Ax(t) + Bu(t) + c$ is a generalization of linear systems, but we can still convert an affine model to a linear model. Specifically, we can simply define the equivalent linear model

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = Ax(t) + \underbrace{\begin{bmatrix} B & I_{N_x} \end{bmatrix}}_{\widetilde{B}} \underbrace{\begin{bmatrix} u(t) \\ c \end{bmatrix}}_{\widetilde{u}(t)}. \tag{20}$$

While this is useful for computational purposes (and for simplifying notation), one must be careful. Many properties of linear systems would assume that the input signal $\widetilde{u} : \mathbb{R} \to \mathbb{R}^{N_u+N_x}$ can be chosen arbitrarily; specifically, that the input signal is absolutely integrable ($\widetilde{u} \in \mathcal{L}_1$). If $c \neq 0$, this assumption does not hold and we might conclude wrongly about the properties of the system. Therefore, we always prefer the original affine representation to perform analysis, and use the linear representation only to perform computations.

Discretization

The time discretization of a linear system requires us to compute the matrices

$$A_{\Delta t} = \exp(A\Delta t) \quad \text{and} \quad B_{\Delta t} = \int_0^{\Delta t} \exp(A\Delta t - \tau)B d\tau. \tag{21}$$

While computing $A_{\Delta t}$ is easy, computing $B_{\Delta t}$ is prone to some numerical issues. Thankfully, there exists a "trick" to compute $B_{\Delta t}$ accurately and efficiently. If we augment the state with the input, $\widetilde{x}(t) = \texttt{col}(x(t), u(t))$, we have that

$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}x(t) \\ \frac{\mathrm{d}}{\mathrm{d}t}u(t) \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \quad \rightsquigarrow \quad \frac{\mathrm{d}}{\mathrm{d}t}\widetilde{x}(t) = \widetilde{A}\widetilde{x}(t) \tag{22}$$

This is a purely linear system (i.e., there are no affine terms on the augmented state). Thus, its transition map takes the simple form $\widetilde{x}[n+1] = \exp(\widetilde{A}\Delta t)\widetilde{x}[n]$, where it can be shown that

$$\exp\left(\begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix} \Delta t\right) = \begin{bmatrix} A_{\Delta t} & B_{\Delta t} \\ 0 & I_{N_u} \end{bmatrix}. \tag{23}$$

Therefore, to compute a time discretization of a linear system, we can simply take the exponential matrix of this augmented system, then extract the matrices $(A_{\Delta t}, B_{\Delta t})$ accordingly.

B A word of caution about noise/disturbances

Until now, we have discussed deterministic models which do not account for the effect of external noise/disturbances. In real-world problems, these cannot be ignored. So, let's take a quick moment to discuss *stochastic systems*.

A more realistic dynamical model is the ordinary differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}x(t) = f(x(t), u(t), w(t)) \tag{24}$$

in which the noise/disturbance signal $w(\cdot)$ is a stochastic process with mean $\mu_w(t) = \mathrm{E}w(t)$ and autocorrelation $R_w(t_1, t_2) = \mathrm{E}(w(t_1)w(t_2)^\mathsf{T})$. In this case, Eq. (24) is a *stochastic differential equation* (SDE) describing a system whose evolution is random, and requires a statistical treatment. In general, the analysis and numerics of SDEs is much more involving than their deterministic counterpart—with some notable exceptions—,and thus we will only discuss very specific stochastic systems during this course. In practice, feedback control provides us with some "protection" against noise/disturbances, as mentioned in the first lecture. We will see that control laws designed using deterministic models can still enforce our desired operational objectives, even in the presence of noise/disturbances–usually, they only make the closed-loop system "zig-zag" around the desired objective (Figure 6). Therefore, for process control purposes, we will (carefully) proceed using deterministic dynamical models. For safety-critical systems or highly-stochastic systems, the fields of *stochastic and robust control* are concerned with designing more advanced controllers that explicitly account for the uncertainty in the models.

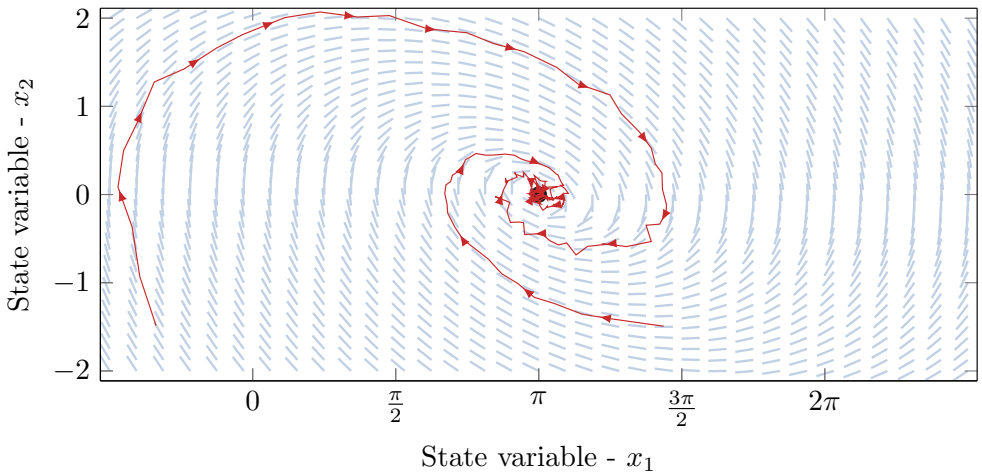

Figure 6. Phase portrait of a 2nd-order system with $K(x) = \frac{\pi}{2} + \sin(x_1) - 0.5x_1$, while also being affected by noise. The closed-loop system is unable to converge asymptotically to the equilibrium. However, thanks to feedback, it still remains at its vicinity.