Chapter 2

System Modeling

> *... I asked Fermi whether he was not impressed by the agreement between our calculated numbers and his measured numbers. He replied, "How many arbitrary parameters did you use for your calculations?" I thought for a moment about our cut-off procedures and said, "Four." He said, "I remember my friend Johnny von Neumann used to say, with four parameters I can fit an elephant, and with five I can make him wiggle his trunk."*
>
> Freeman Dyson on describing the predictions of his model for meson-proton scattering to Enrico Fermi in 1953 [Dys04].

A model is a precise representation of a system's dynamics used to answer questions via analysis and simulation. The model we choose depends on the questions we wish to answer, and so there may be multiple models for a single physical system, with different levels of fidelity depending on the phenomena of interest. In this chapter we provide an introduction to the concept of modeling, and provide some basic material on two specific methods that are commonly used in feedback and control systems: differential equations and difference equations.

2.1 Modeling Concepts

A model is a mathematical representation of a physical, biological or information system. Models allow us to reason about a system and make predictions about how a system will behave. In this text, we will mainly be interested in models of dynamical systems describing the input/output behavior of systems and often in so-called "state space" form.

Roughly speaking, a dynamical system is one in which the effects of actions do not occur immediately. For example, the velocity of a car does not change immediately when the gas pedal is pushed nor does the temperature in a room rise instantaneously when a heater is switched on. Similarly, a headache does not vanish right after an aspirin is taken, requiring time to take effect. In business systems, increased funding for a development project does not increase revenues in the short term, although it may do so in the long term (if it was a good investment). All of these are examples of dynamical systems, in which the behavior of the system evolves with time.

Dynamical systems can be viewed in two different ways: the internal view or the external view. The internal view attempts to describe the internal workings of the system and originates from classical mechanics. The prototype problem was describing the motion of the planets. For this problem it was natural to give a complete characterization of the motion of all planets. This involves careful analysis of the effects of gravitational pull and the relative positions of the planets in a system. A major tool in the internal view is differential equations.

The other view on dynamics originated in electrical engineering. The prototype problem was to describe electronic amplifiers, where it was natural to view an amplifier as a device that transforms input voltages to output voltages and disregard the internal details of the amplifier. This resulted in the input/output, or external, view of systems. For this type of model, much more emphasis is placed on how a system is driven through and external input and how the system evolves in terms of a fixed set of sensed (or output) measurements. A major tool in the external view is the frequency response.

The two different views have been amalgamated in control theory. Models based on the internal view are called internal descriptions, state models, or white box models. The external view is associated with names such as external descriptions, input/output models or black box models. In this book we will mostly use the terms state models and input/output models.

In the remainder of this section we provide an overview of some of the key concepts in modeling. The mathematical details introduced here are explored more fully in the remainder of the chapter.

The Heritage of Mechanics

The study of dynamics originated in the attempts to describe planetary motion. The basis was detailed observations of the planets by Tycho Brahe and the results of Kepler, who found empirically that the orbits of the planets could be well described by ellipses. Newton embarked on an ambitious program to try to explain why the planets move in ellipses and he found that

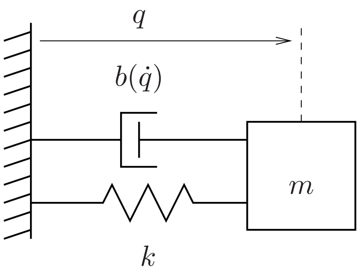


Figure 2.1: Mass and spring system, with nonlinear damping. The position of the mass is denoted by q, with $q = 0$ corresponding to the rest position of the spring.

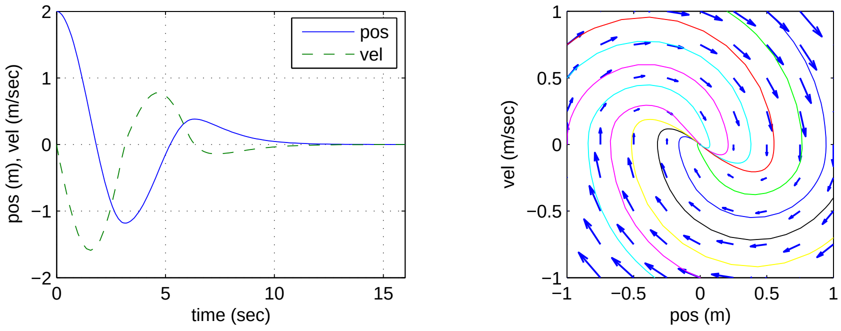
the motion could be explained by his law of gravitation and the formula that force equals mass times acceleration. In the process he also invented calculus and differential equations. Newton's result was the first example of the idea of reductionism, i.e. that seemingly complicated natural phenomena can be explained by simple physical laws. This became the paradigm of natural science for many centuries.

One of the triumphs of Newton's mechanics was the observation that the motion of the planets could be predicted based on the current positions and velocities of all planets. It was not necessary to know the past motion. The *state* of a dynamical system is a collection of variables that characterizes the motion of a system completely for the purpose of predicting future motion. For a system of planets the state is simply the positions and the velocities of the planets. We call the set of all possible states the *state space*.

A common class of mathematical models for dynamical systems is ordinary differential equations (ODEs). In mechanics, one of the simplest such differential equation is that of a mass and spring system, with damping:

$$m\ddot{q} + c(\dot{q}) + kq = 0. \tag{2.1}$$

This system is illustrated in Figure 2.1. The variable $q \in \mathbb{R}$ represents the position of the mass m with respect to its rest position. We use the notation $\dot{q}$ to denote the derivative of q with respect to time (i.e., the velocity of the mass) and $\ddot{q}$ to represent the second derivative (acceleration). The spring is assumed to be a satisfy Hooke's law, which says that the force is proportional to the displacement. The friction element (damper) is taken as a nonlinear function, $c(\dot{q})$, which can model effects such as stiction and viscous drag. The position q and velocity $\dot{q}$ represent the instantaneous "state" of the system. We say that this system is a *second order system* since the dynamics depend on the second derivative of q.

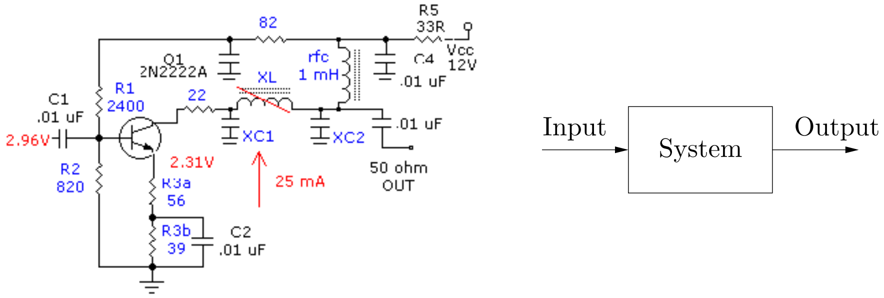


(a) (b)

Figure 2.2: Illustration of a state model. A state model gives the rate of change of the state as a function of the state. The plot on the left shows the evolution of the state as a function of time. The plot on the right shows the evolution of the states relative to each other, with the velocity of the state denoted by arrows.

The evolution of the position and velocity can be described using either a time plot or a phase plot, both of which are shown in Figure 2.2. The time plot, on the left, shows the values of the individual states as a function of time. The phase plot, on the right, shows the *vector field* for the system, which gives the state velocity (represented as an arrow) at every point in the state space. In addition, we have superimposed the traces of some of the states from different conditions. The phase plot gives a strong intuitive representation of the equation as a vector field or a flow. While systems of second order (two states) can be represented in this way, it is unfortunately difficult to visualize equations of higher order using this approach.

The ideas of dynamics and state have had a profound influence on philosophy, where they inspired the idea of predestination. If the state of a natural system is known at some time, its future development is completely determined. However, we know that for many natural systems it can be impossible to make predications of the detailed behavior of the system far into the future. This problem has been resolved in part by the advent of the theory of chaos. As the development of dynamics continued in the 20th century, it was discovered that there are simple dynamical systems that are extremely sensitive to initial conditions; small perturbations may lead to drastic changes in the behavior of the system. The behavior of the system could also be extremely complicated. The emergence of chaos thus resolved the problem of determinism: even if the solution is uniquely determined by the initial conditions, in practice it can be impossible to make predictions



(a)

(b)

Figure 2.3: Illustration of the input/output view of a dynamical system. The figure on the left shows a detailed circuit diagram for an electronic amplifier; the one of the right its representation as a block diagram.

because of the sensitivity of these initial conditions.

The differential equation (2.1) is called an autonomous system because there are no external influences. Such a model is natural to use for celestial mechanics, because it is difficult to influence the motion of the planets. In many examples, it is useful to model the effects of external disturbances or controlled forces on the system. One way to capture this is to replace equation (2.1) by

$$m\ddot{q} + c(\dot{q}) + kq = u \tag{2.2}$$

where u represents the effect of external influences. The model (2.2) is called a *forced* or *controlled differential equation*. The model implies that the rate of change of the state can be influenced by the *input*, $u(t)$. Adding the input makes the model richer and allows new questions to be posed. For example, we can examine what influence external disturbances have on the trajectories of a system. Or, in the case when the input variable is something that can be modulated in a controlled way, we can analyze whether it is possible to "steer" the system from one point in the state space to another through proper choice of the input.

The Heritage of Electrical Engineering

A very different view of dynamics emerged from electrical engineering, where the design of electronic amplifiers led to a focus on input/output behavior. A system was considered as a device that transformed inputs to outputs, as illustrated in Figure 2.3. Conceptually an input/output model can be viewed as a giant table of inputs and outputs. Given an input signal $u(t)$, the model should produce the resulting output $y(t)$.

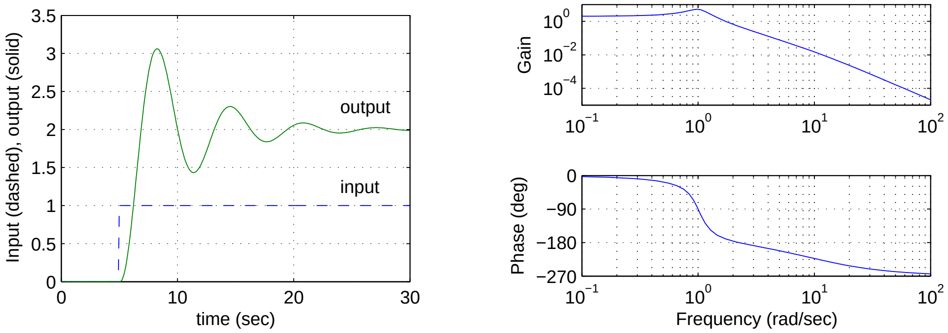
The input/output framework is used in many engineering systems since it allows us to decompose a problem into individual components, connected through their inputs and outputs. Thus, we can take a complicated system such as a radio or a television and break it down into manageable pieces, such as the receiver, demodulator, amplifier and speakers. Each of these pieces has a set of inputs and outputs and, through proper design, these components can be interconnected to form the entire system.

The input/output view is particularly useful for the special class of *linear, time-invariant* systems. This term will be defined more carefully later in this chapter, but roughly speaking a system is linear if the superposition (addition) of two inputs yields an output which is the sum of the outputs that would correspond to individual inputs being applied separately. A system is time-invariant if the output response for a given input does not depend on when that input is applied. (Chapter 5 provides a much more detailed analysis of linear systems.)

Many electrical engineering systems can be modeled by linear, time-invariant systems and hence a large number of tools have been developed to analyze them. One such tool is the *step response*, which describes the relationship between an input that changes from zero to a constant value abruptly (a “step” input) and the corresponding output. As we shall see in the latter part of the text, the step response is extremely useful in characterizing the performance of a dynamical system and it is often used to specify the desired dynamics. A sample step response is shown in Figure 2.4a.

Another possibility to describe a linear, time-invariant system is to represent the system by its response to sinusoidal input signals. This is called the *frequency response* and a rich powerful theory with many concepts and strong, useful results has emerged. The results are based on the theory of complex variables and Laplace transforms. The basic idea behind the frequency response is that we can completely characterize the behavior of a system by its steady state response to sinusoidal inputs. Roughly speaking, this is done by decomposing any arbitrary signal into a linear combination of sinusoids (e.g., by using the Fourier transform) and then using linearity to compute the output by combining the response to the individual frequencies. A sample frequency response is shown in Figure 2.4b.

The input/output view lends itself naturally to experimental determination of system dynamics, where a system is characterized by recording its response to a particular input, e.g. a step or a sweep across a range of frequencies.



(a)

(b)

Figure 2.4: Input/output response of a linear system. The step response (a) shows the output of the system due to an input that changes from 0 to 1 at time $t = 5$ s. The frequency response (b) shows the amplitude gain and phase change due to a sinusoidal input at different frequencies.

The Control View

When control theory emerged as a discipline in the 1940s, the approach to dynamics was strongly influenced by the electrical engineering (input/output) view. A second wave of developments in control, starting in the late 1950s, was inspired by mechanics, where the state space perspective was used. In addition, there was a shift over this period from autonomous systems (with no inputs) to those where inputs to the process were available to modify the dynamics of the process. The emergence of space flight is a typical example, where precise control of the orbit is essential.

The models from mechanics were thus modified to include external control forces and sensors, and more general forms of equations were considered. In control, the model given by equation (2.2) was replaced by

$$\begin{aligned} \frac{dx}{dt} &= f(x, u) \\ y &= h(x, u), \end{aligned} \tag{2.3}$$

where x is a vector of "state" variables, u is a vector of control signals, and y a vector of measurements. As before, $\dot{x}$ represents the derivative of x with respect to time, now considered as a vector, and f and h are mappings of their arguments to vectors of the appropriate dimension.

This viewpoint has added to the richness of the classical problems and led to many new concepts. For example it is natural to ask if possible states x can be reached with the proper choice of u (reachability) and if the measurement y contains enough information to reconstruct the state (observability) (these topics will be addressed in greater detail in Chapters 6 and 7).

A final development in building the control point of view was the emergence of disturbance and model uncertainty as critical elements in the theory. The simple way of modeling disturbances as deterministic signals like steps and sinusoids has the drawback that such signals can be predicted precisely. A much more realistic approach is to model disturbances like random signals. This viewpoint gives a natural connection between prediction and control. The dual views of input/output representations and state space representations are particularly useful when modeling uncertainty, since state models are very convenient to describe a nominal model but uncertainties are easier to describe using input/output models (often via a frequency response description). Uncertainty will be a constant theme throughout the text and will be studied in particular detail in Chapter 12.

An interesting experience in design of control system is that feedback systems can often be analyzed and designed based on comparatively simple models. The reason for this is the inherent robustness of feedback systems. However, other uses of models may require more complexity and more accuracy. One example is feedforward control strategies, where one uses a model to pre-compute the inputs that will cause the system to respond in a certain way. Another area is in system validation, where one wishes to verify that the detailed response of the system performs as it was designed. Because of these different uses of models, it is therefore common to use a hierarchy of models having different complexity and fidelity.

Multi-Domain Modeling

Modeling is an essential element of many disciplines, but traditions and methods from individual disciplines in can be very different from each other, as illustrated by the previous discussion of mechanical and electrical engineering. A difficulty in systems engineering is that it is frequently necessary to deal with heterogeneous systems from many different domains, including chemical, electrical, mechanical and information systems.

To deal with such multi-domain systems, we start by cutting a system into smaller subsystems. Each subsystem is modeled either by balance equations for mass, energy, and momentum or by appropriate descriptions of the information processing in the subsystem. The behavior at the interfaces is captured by describing how the variables of the subsystem behave when the subsystems are interconnected. These interfaces often act by constraining variables within the individual subsystems to be equal (such as mass, energy or momentum fluxes). The complete model is then obtained by combining the descriptions of the subsystems and the interfaces.

Using this methodology it is possible to build up libraries of subsystems that correspond to physical, chemical and informational components. The procedure mimics the engineering approach where systems are built from subsystems that are themselves built from smaller components. As experience is gained, the components and their interfaces can be standardized and collected in model libraries. In practice, it takes several iterations to obtain a good library that can be reused for many applications.

State models or ordinary differential equations are not suitable for component based modeling of this form because states may disappear when components are connected. This implies that the internal description of a component may change when it is connected to other components. As an illustration we consider two capacitors in an electrical circuit. Each capacitor has a state corresponding to the voltage across the capacitors, but one of the states will disappear if the capacitors are connected in parallel. A similar situation happens with two rotating inertias, each of which are individually modeled using the the angle of rotation and the angular velocity. Two states will disappear when the inertias are joined by a rigid shaft.

This difficulty can be avoided by replacing differential equations by *differential algebraic equations*, which have the form

$$F(z, \dot z) = 0$$

where $z \in \mathbb{R}^n$. A simple special case is

$$\dot x = f(x, y), \qquad g(x, y) = 0 \tag{2.4}$$

where $z = (x, y)$ and $F = (\dot x - f(x, y), g(x, y))$. The key property is that the derivative $\dot z$ is not given explicitly and there may be pure algebraic relations between the components of the vector z.

A differential equation is an *imperative* description: if it tells how to calculate $\dot x$ from x. The differential algebraic equation is a *declarative* description: it gives a relation between z and $\dot z$, without explicitly describing how to compute $\dot z$. The model (2.4) captures the examples of the parallel capacitors and the linked rotating inertias. For example, when two capacitors are connected we simply include the algebraic equation expressing that the voltages across the capacitors are the same.

A practical difficulty with component-based declarative descriptions is a that the model may contain many auxiliary variables. This was a severe limitation for hand calculations, but fortunately there are methods for symbolic calculation that can be used to eliminate the auxiliary variables. Symbolic calculations can also be used to transform and simplify the models.

Modelica is a language that has been developed to support component based modeling. Differential algebraic equations are used as the basic description, object-oriented programming is used to structure the models. Modelica is used to model the dynamics of technical systems in domains such as, mechanical, electrical, thermal, hydraulic, thermo-fluid, and control subsystems. Modelica is intended to serve as a standard format so that models arising in different domains can be exchanged between tools and users. A large set of free and commercial *Modelica* component libraries are available and are utilized by a growing number of people in industry, research and academia. For further information about *Modelica*, see `http://www.modelica.org`.

2.2 State Space Models

In this section we introduce the two primary forms of models that we use in this text: differential equations and difference equations. Both of these make use of the notions of state, inputs, outputs and dynamics to describe the behavior of a system.

Ordinary Differential Equations

The state of a system is a collection of variables that summarize the past of a system for the purpose of prediction the future. For an engineering system the state is composed of the variables required to account for storage of mass, momentum and energy. A key issue in modeling is to decide how accurately this storage has to be represented. The state variables are gathered in a vector, $x \in \mathbb{R}^n$, called the *state vector*. The control variables are represented by another vector $u \in \mathbb{R}^p$ and the measured signal by the vector $y \in \mathbb{R}^q$. A system can then be represented by the differential equation

$$\begin{aligned} \frac{dx}{dt} &= f(x, u) \\ y &= h(x, u), \end{aligned} \tag{2.5}$$

where $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ and $h : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^q$ are smooth mappings. We call a model of this form a *state space model*.

The dimension of the state vector is called the order of the system. The system is called time-invariant because the functions f and g do not depend explicitly on time t. It is possible to have more general time-varying systems where the functions do depend on time. The model thus consists of two functions: the function f gives the velocity of the state vector as a function of state x and control u, and the function g gives the measured values as functions of state x and control u.

A system is called linear if the functions f and g are linear in x and u. A linear state space system can thus be represented by

$$\begin{aligned}\frac{dx}{dt} &= Ax + Bu\\ y &= Cx + Du,\end{aligned}$$

where A, B, C and D are constant matrices. Such a system is said to be linear and time-invariant, or LTI for short. The matrix A is called the *dynamics matrix*, the matrix B is called the *control matrix*, the matrix C is called the *sensor matrix* and the matrix D is called the *direct term*. Frequently systems will not have a direct term, indicating that the control signal does not influence the output directly.

A different form of linear differential equations, generalizing the second order dynamics from mechanics, is an equation of the form

$$\frac{d^nq}{dt^n} + a_1\frac{d^{n-1}q}{dt^{n-1}} + \cdots + a_nq = u, \tag{2.6}$$

where t is the independent (time) variable, $q(t)$ is the dependent (output) variable, and $u(t)$ is the input. This system is said to be an nth order system. This system can be converted into state space form by defining

$$x = \begin{pmatrix} x_1\\ x_2\\ \vdots\\ x_n\end{pmatrix} = \begin{pmatrix} d^{n-1}q/dt^{n-1}\\ \vdots\\ dq/dt\\ q\end{pmatrix}$$

and the state space equations become

$$\frac{d}{dt}\begin{pmatrix} x_1\\ x_2\\ \vdots\\ x_{n-1}\\ x_n\end{pmatrix} = \begin{pmatrix} -a_1x_1 - \cdots - a_nx_n\\ x_1\\ x_2\\ \vdots\\ x_{n-1}\end{pmatrix} + \begin{pmatrix}1\\ 0\\ \vdots\\ 0\end{pmatrix}$$

$$y = x_n.$$

With the appropriate definition of A, B, C and D, this equation is in linear state space form.

An even more general system is obtained by letting the output be a linear combination of the states of the system, i.e.

$$y = b_1 x_1 + b_2 x_2 + \cdots + b_n x_n + du$$

This system can be modeled in state space as

$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} -a_1 & -a_2 & \dots & & -a_n \\ 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & & & \ddots & \\ 0 & & & 1 & 0 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} u$$

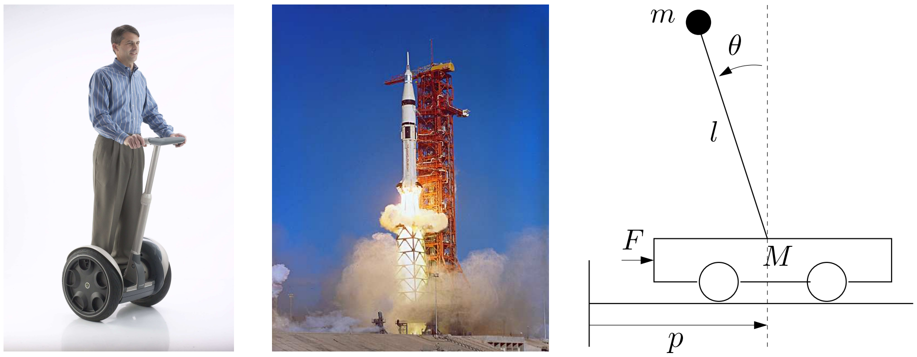
$$y = \begin{pmatrix} b_1 & b_2 & \dots & b_n \end{pmatrix} x + du.$$

This particular form of a linear state space system is called reachable canonical form and will be studied in more detail in later chapters.

Example 2.1 (Balance systems)**.** An example of a class of systems that can be modeled using ordinary differential equations is the class of "balance systems." A balance system is a mechanical system in which the center of mass is balanced above a pivot point. Some common examples of balance systems are shown in Figure 2.5. The Segway human transportation system (Figure 2.5a) uses a motorized platform to stabilize a person standing on top of it. When the rider leans forward, the vehicle propels itself along the ground, but maintains its upright position. Another example is a rocket (Figure 2.5b), in which a gimbaled nozzle at the bottom of the rocket is used to stabilize the body of the rocket above it. Other examples of balance systems include humans or other animals standing upright or a person balancing a stick on their hand.

Figure 2.5c shows a simplified diagram for a balance system. To model this system, we choose state variables that represent the position and velocity of the base of the system, p and $\dot{p}$, and the angle and angular rate of the structure above the base, θ and $\dot{\theta}$. We let F represent the force applied at the base of the system, assumed to be in the horizontal direction (aligned with p), and choose the position and angle of the system as outputs. With this set of definitions, the dynamics of the system can be computed using Newtonian mechanics and has the form

$$\begin{pmatrix} (M+m) & -ml\cos\theta \\ -ml\cos\theta & (J+ml^2) \end{pmatrix} \begin{pmatrix} \ddot{p} \\ \ddot{\theta} \end{pmatrix} + \begin{pmatrix} c\dot{p} + ml\sin\theta\,\dot{\theta}^2 \\ mgl\sin\theta + \gamma\dot{\theta} \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix}, \qquad (2.7)$$



(a) (b) (c)

Figure 2.5: Balance systems: (a) Segway human transportation systems, (b) Saturn rocket and (c) simplified diagram. Each of these examples uses forces at the bottom of the system to keep it upright.

where M is the mass of the base, m and J are the mass and moment of inertia of the system to be balanced, l is the distance from the base to the center of mass of the balanced body, c and γ are coefficients of viscous friction, and g is the acceleration due to gravity.

We can rewrite the dynamics of the system in state space form by defining the state as $x = (p, \theta, \dot{p}, \dot{\theta})$, the input as $u = F$ and the output as $y = (p, \theta)$. If we define the total mass and total inertia as

$$M_t = M + m \qquad J_t = J + ml^2,$$

respectively, the equations of motion then become

$$\frac{d}{dt}\begin{pmatrix} p \\ \theta \\ \dot{p} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \dot{p} \\ \dot{\theta} \\ \dfrac{-ml\sin\theta\dot{\theta}^2 + mg(ml^2/J_t)\sin\theta\cos\theta - c\dot{p} + u}{M_t - m(ml^2/J_t)\cos^2\theta} \\ \dfrac{-ml^2\sin\theta\cos\theta\dot{\theta}^2 + M_t gl\sin\theta + cl\cos\theta\dot{p} + \gamma\dot{\theta} + l\cos\theta u}{J_t(M_t/m) - m(l\cos\theta)^2} \end{pmatrix}$$

$$y = \begin{pmatrix} p \\ \theta \end{pmatrix}.$$

In many cases, the angle θ will be very close to 0 and hence we can approximate $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. Furthermore, if $\dot{\theta}$ is small, we can ignore quadratic and higher terms in $\dot{\theta}$. Substituting these approximations into our equations, we see that we are left with a *linear* state space equation

$$\frac{d}{dt}\begin{pmatrix} p \\ \theta \\ \dot{p} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{m^2l^2g}{\mu} & \frac{-cJ_t}{\mu} & 0 \\ 0 & \frac{M_tmgl}{\mu} & \frac{clm}{\mu} & \frac{\gamma m}{\mu} \end{pmatrix} \begin{pmatrix} p \\ \theta \\ \dot{p} \\ \dot{\theta} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \frac{J_t}{\mu} \\ \frac{lm}{\mu} \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} x,$$

where $\mu = M_tJ_t - m^2l^2g$. ∇

Example 2.2 (Inverted pendulum)**.** A variation of this example is one in which the location of the base, p, does not need to be controlled. This happens, for example, if we are only interested in stabilizing a rocket's upright orientation, without worrying about the location of base of the rocket. The dynamics of this simplified system is given by

$$\begin{aligned} \frac{d}{dt}\begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} &= \begin{pmatrix} \dot{\theta} \\ \dfrac{mgl}{J_t}\sin\theta - \dfrac{\gamma}{J_t}\dot{\theta} + \frac{l}{J_t}\cos\theta\, u \end{pmatrix} \\ y &= \begin{pmatrix} 1 & 0 \end{pmatrix} x, \end{aligned} \tag{2.8}$$

where γ is the coefficient of rotational friction, $J_t = J + ml^2$ and u is the force applied at the base. This system is referred to as an *inverted pendulum.* ∇

Difference Equations

In some circumstances, it is more natural to describe the evolution of a system at discrete instants of time rather than continuously in time. If we refer to each of these times by an integer $k = 0, 1, 2, \dots,$ then we can ask how the state of the system changes for each k. Just as in the case of differential equations, we shall define the state to be those sets of variables that summarize the past of the system for the purpose of predicting its future. Systems described in this manner are referred to as *discrete time systems.*

The evolution of a discrete time system can written in the form

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ y_k &= h(x_k, u_k) \end{aligned} \tag{2.9}$$

where $x_k \in \mathbb{R}^n$ is the state of the system at "time" k (an integer), $u_k \in \mathbb{R}^m$ is the input and $y_k \in \mathbb{R}^p$ is the output. As before, f and h are smooth mappings of the appropriate dimension. We call equation (2.9) a *difference equation* since it tells us now x_{k+1} differs from x_k. The state x_k can either be a scalar or a vector valued quanity; in the case of the latter we use superscripts to denote a particular element of the state vector: x_k^i is the value of the ith state at time k.

Just as in the case of differential equations, it will often be the case that the equations are linear in the state and input, in which case we can write the system as

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k + Du_k. \end{aligned}$$

As before, we refer to the matrices A, B, C and D as the dynamics matrix, the control matrix, the sensor matrix and the direct term. The solution of a linear difference equation with initial condition x_0 and input $u_1, \dots, u_T$ is given by

$$\begin{aligned} x_k &= A^k x_0 + \sum_{i=0}^{k} A^i B u_i \\ y_k &= CA^k x_0 + \sum_{i=0}^{k} CA^i B u_i + Du_k \end{aligned} \tag{2.10}$$

Example 2.3 (Predator prey)**.** As an example of a discrete time system, we consider a simple model for a predator prey system. The predator prey problem refers to an ecological system in which we have two species, one of which feeds on the other. This type of system has been studied for decades and is known to exhibit very interesting dynamics. Figure 2.6 shows a historical record taken over 50 years in the population of lynxes versus hares [Mac37]. As can been seen from the graph, the annual records of the populations of each species are oscillatory in nature.

A simple model for this situation can be constructed using a discrete time model by keeping track of the rate of births and deaths of each species. Letting H represent the population of hares and L represent the population of lynxes, we can describe the state in terms of the populations at discrete periods of time. Letting k be the discrete time index (e.g., the day number), we can write

$$\begin{aligned} H_{k+1} &= H_k + b_r(u) H_k - a L_k H_k \\ L_{k+1} &= L_k - d_f L_k + a L_k H_k, \end{aligned} \tag{2.11}$$

where $b_r(u)$ is the hare birth rate per unit period and as a function of the

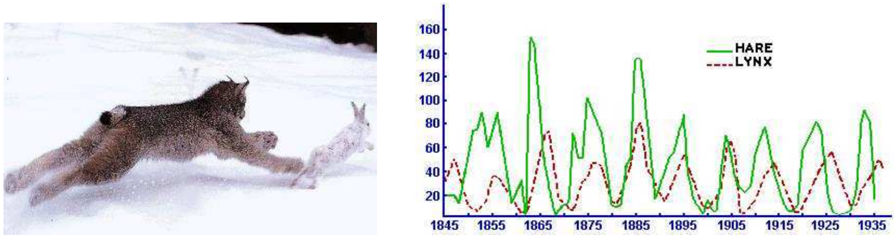


Figure 2.6: Predator versus prey. The photograph shows a Canadian lynx and a snowshoe hare. The graph on the right shows the populations of hares and lynxes between 1845 and 1935 [MS93]. Photograph courtesy Rudolfo's Usenet Animal Pictures Gallery.

food supply u, d_f is the lynx death rate, and a is the interaction term. The interaction term models both the rate at which lynxes eat hares and the rate at which lynxes are produced by eating hares. This model makes many simplifying assumptions—such as the fact that hares never die of old age or causes other than being eaten—but it often is sufficient to answer basic questions about the system.

To illustrate the usage of this system, we can compute the number of lynxes and hares from some initial population. This is done by starting with $x_0 = (H_0, L_0)$ and then using equation (2.11) to compute the populations in the following year. By iterating this procedure, we can generate the population over time. The output of this process for a specific choice of parameters and initial conditions is shown in Figure 2.7. While the details of the simulation are different from the experimental data (to be expected given the simplicity of our assumptions), we see qualitatively similar trends

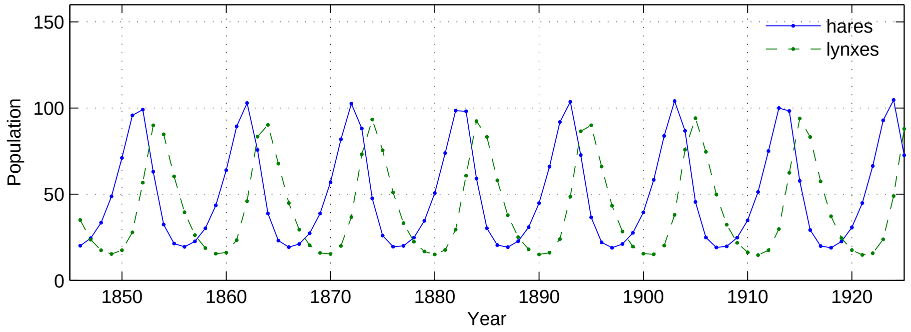


Figure 2.7: A simulation of the predator prey model with $a = 0.007$, $b_r(u) = 0.7$ and $d = 0.5$.

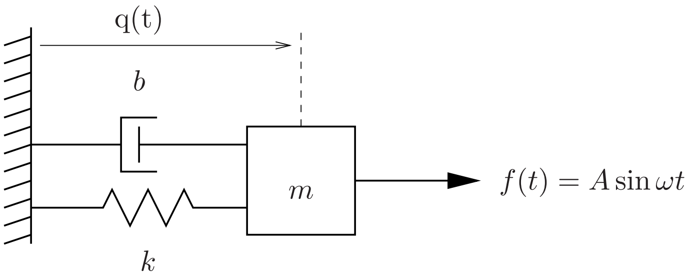


Figure 2.8: A driven mass spring system, with damping.

and hence we can use the model to help explore the dynamics of the system. ∇

Simulation and Analysis

State space models can be used to answer many questions. One of the most common, as we saw in the previous examples, is to predict the evolution of the system state from a given initial condition. While for simple models this can be done in closed form, more often it is accomplished through computer simulation. One can also use state space models to analyze the overall behavior of the system, without making direct use of simulation. For example, we can ask whether a system that is perturbed from an equilibrium configuration will return to that configuration; such a system is said to be *stable*. While one could in principle answer this question by simulating many trajectories, it turns out that we can use analysis techniques to answer this much more easily and completely. We illustrate some of the concepts of simulation and analysis through a series of examples; a more formal treatment is provided in the next chapter.

Example 2.4 (Damped spring mass system)**.** Consider again the damped spring mass system from Section 2.1, but this time with an external force applied, as shown in Figure 2.8. We wish to predict the motion of the system for a periodic forcing function, with a given initial condition, and determine the amplitude, frequency, and decay rate of the resulting motion.

We choose to model the system using a linear ordinary differential equation. Using Hooke's law to model the spring and assuming that the damper exerts a force that is proportional to the velocity of the system, we have

$$m\ddot{q} + c\dot{q} + kq = f(t), \tag{2.12}$$

where m is the mass, q is the displacement of the mass, c is the coefficient of viscous friction, k is the spring constant and f is the applied force. In state space form, using $x = (q, \dot{q})$ as the state, $u = f$ as the input and choosing $y = q$ as the output, we have

$$\begin{aligned}\frac{dx}{dt} &= \begin{pmatrix} x_2 \\ -\frac{c}{m}x_2 - \frac{k}{m}x_1 + u/m \end{pmatrix} \\ y &= x_1.\end{aligned}$$

We see that this is a linear, second order differential equation with one input and one output.

We now wish to compute the response of the system to an input of the form $u = A\sin\omega t$. Although it is possible to solve for the response analytically, we instead make use of computational approach that does not rely on the specific form of this system. Consider the general state space system

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

Given the state x at time t, we can approximate the value of the state at a short time $\epsilon > 0$ later by assuming that x and u are constant over the interval ϵ. This gives us that

$$x(t + \epsilon) = x(t) + \epsilon f(x(t), u(t)). \tag{2.13}$$

Iterating this equation, we can thus solve for x as a function of time. This approximation is known as Euler integration, and is in fact a difference equation if we let ϵ represent the time increment and write $x_k = x(k\epsilon)$. Although modern simulation tools use much more accurate methods than Euler integration, it still illustrates some of the basic tradeoffs.

Returning to our specific example, Figure 2.9 shows the results of computing $x(t)$ using equation (2.13), along with the analytical computation. We see that as h gets smaller, the compute solution converges to the exact solution. The form of the solution is also worth noticing: after an initial transient, the system settles into a period motion that is the same frequency as the input term, but at a different amplitude and slightly shifted in time. The portion of the response after the transient is called the *steady state response* to the input. ∇

In addition to performing simulations, models can also be used to answer other types of questions. Two that are central to the methods described in this text are stability of an equilibrium point and the input/output frequency

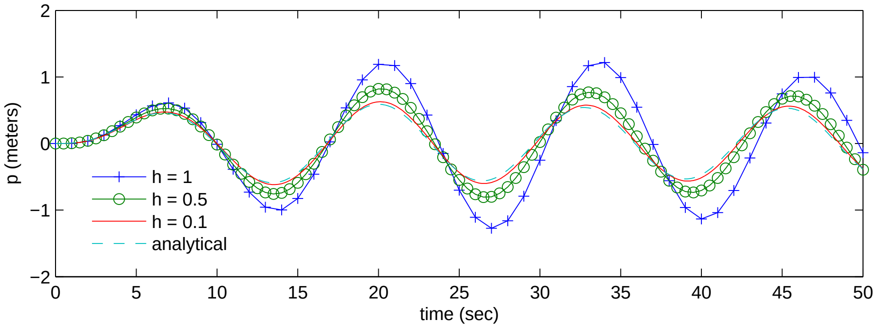


Figure 2.9: Simulation of the forced spring mass system with different simulation time constants.

response. We illustrate these two computations through the examples below, and return to the general computations in later chapters.

Example 2.5 (Stability)**.** Consider the damped spring mass system given in the previous example, but with no input forcing. The equations of motion are given by

$$\frac{dx}{dt} = \begin{pmatrix} x_2 \\ -\frac{b}{m}x_2 - \frac{k}{m}x_1 \end{pmatrix}, \tag{2.14}$$

where x_1 is the position of the mass (relative to the rest position) and x_2 its velocity. We wish to show that if the initial state of the system is away from the rest position, the system will return to the rest position eventually (we will later define this situation to mean that the rest position is *asymptotically stable*). While we could heuristically show this by simulating many, many initial conditions, we seek instead to prove that this is true for *any* initial condition.

To do so, we construct a function $V : \mathbb{R}^n \to \mathbb{R}$ that maps the system state to a positive real number. For mechanical systems, a convenient choice is the energy of the system,

$$V(x) = \frac{1}{2}kx_1^2 + \frac{1}{2}m\dot{x}_2^2. \tag{2.15}$$

If we look at the time derivative of the energy function, we see that

$$\begin{aligned} \frac{dV}{dt} &= kx_1\dot{x}_1 + mx_2\dot{x}_2 \\ &= kx_1x_2 + mx_2(-\frac{b}{m}x_2 - \frac{k}{m}x_1) \\ &= -bx_2^2, \end{aligned}$$

which is always either negative or zero. Hence $V(x(t))$ is never increasing and, using a bit of analysis that we will see formally in the next chapter, the individual states must remain bounded.

If we wish to show that the states eventually return to the origin, we must use a more slightly more detailed analysis. Intuitively, we can reason as follows: suppose that for some period of time, $V(x(t))$ stops decreasing. Then it must be true that $\dot{V}(x(t)) = 0$, which in turn implies that $x_2(t) = 0$ for that same period. In that case, $\dot{x}_2(t) = 0$ and we can substitute into the second line of equation (2.14) to obtain:

$$0 = \dot{x}_2 = -\frac{b}{m}x_2 - \frac{k}{m}x_1 = \frac{k}{m}x_1.$$

Thus we must have that x_1 also equals zero and so the only time that $V(x(t))$ can stop decreasing is if the state is at the origin (and hence this system is at its rest position). Since we know that $V(x(t))$ is never increasing (since $\dot{V} \leq 0$), we therefore conclude that the origin is stable (for *any* initial condition.

This type of analysis, called Lyapunov analysis, is considered in detail in Chapter 4 but shows some of the power of using models for analysis of system properties. ∇

Example 2.6 (Frequency response)**.** A second type of analysis that we can perform with models is to compute the output of a system to a sinusoidal input. We again consider the spring mass system, but this time keeping the input and leaving the system in its original form:

$$m\ddot{q} + c\dot{q} + kq = f(t). \tag{2.16}$$

We wish to understand what the response of the system is to a sinusoidal input of the form

$$f(t) = A \sin \omega t.$$

We will see how to do this analytically in Chapter 8, but for now we make use of simulations to compute the answer.

We first begin with the observation that if $q(t)$ is the solution to equation (2.16) with input $f(t)$, then applying an input $f'(t) = 2f(t)$ will give a solution $q'(t) = 2q(t)$ (this is easily verified by substitution). Hence it suffices to look at an an input with unit magnitude, $A = 1$. A second observation, which we will prove in Chapter 5, is that the long term response of the system to a sinusoidal input is itself a sinusoid (at the same frequency) and so the output has the form

$$q(t) = g(\omega) \sin(\omega t + \varphi(\omega)),$$

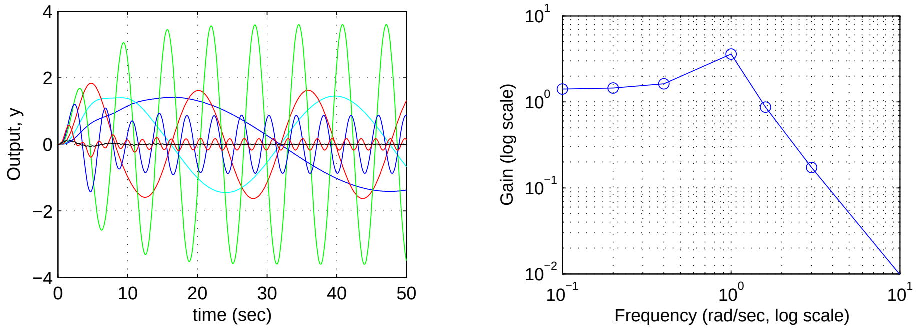


Figure 2.10: A frequency response (magnitude only) computed by measuring the response of individual sinusoids. The figure on the left shows the response of the system to a number of different unit magnitude inputs (at different frequencies). The figure on the right shows this same data in a different way, with the magnitude of the response plotted as a function of the input frequency.

where $g(\omega)$ is the "gain" of the system and $\varphi(\omega)$ is the phase offset.

To compute the frequency response numerically, we can simply simulate the system at a set of frequencies $\omega_1, \ldots, \omega_N$ and plot the gain and phase at each of these frequencies. An example of this type of computation is shown in Figure 2.10. ∇

Modeling from Experiments

Since control systems are provided with sensors and actuators it is also possible to obtain models of system dynamics from experiments on the process. The models are restricted to input/output models since only these signals are accessible to experiments, but modeling from experiments can also be combined with modeling from physics through the use of feedback and interconnection.

A simple way to determine a system's dynamics is to observe the response to a step change in the control signal. Such an experiment begins by setting the control signal to a constant value, then when steady state is established the control signal is changed quickly to a new level and the output is observed. The experiment will thus directly give the step response of the system. The shape of the response gives useful information about the dynamics. It immediately gives an indication of the response time and it tells if the system is oscillatory or if the response in monotone. By repeating the experiment for different steady state values and different amplitudes of

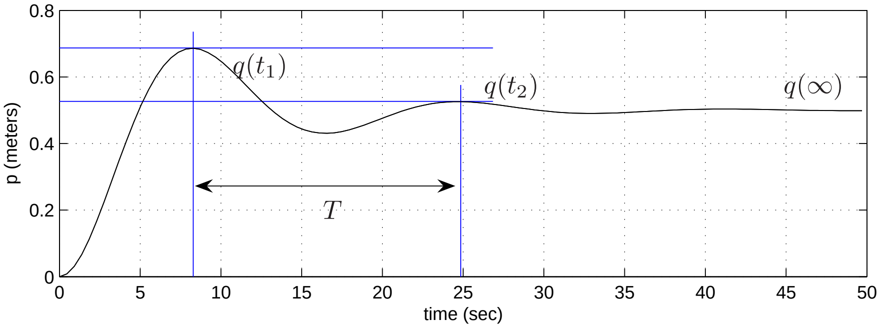


Figure 2.11: Step response for a spring mass system. The magnitude of the step input is $F_0 = 20$ N.

the change of the control signal we can also determine ranges where the process can be approximated by a linear system.

Example 2.7 (Identification of a spring mass system)**.** Consider the spring mass system from Section 2.1, whose dynamics are given by

$$m\ddot{q} + b\dot{q} + kq = u. \tag{2.17}$$

We wish to determine the constants m, b and k by measuring the response of the system to a step input of magnitude F_0.

We will show in Chapter 5 that when $b^2 < 4km$, the step response for this system from the rest configuration is given by

$$q(t) = \frac{F_0}{k}\left(1 - e^{-\frac{bt}{2m}}\left[\cos(\tfrac{\sqrt{4km-b^2}}{2m}\,t) - \frac{1}{\sqrt{4km-b^2}}\sin(\tfrac{\sqrt{4km-b^2}}{2m}\,t)\right]\right)$$

From the form of the solution, we see that the form of the response is determined by the parameters of the system. Hence, by measuring certain features of the step response we can determine the parameter values.

Figure 2.11 shows the response of the system to a step of magnitude $F_0 =$ 20 N, along with some measurements. We start by noting that the steady state position of the mass (after the oscillations die down) is a function of the spring constant, k:

$$q(\infty) = \frac{F_0}{k}, \tag{2.18}$$

where F_0 is the magnitude of the applied force ($F_0 = 1$ for a unit step input). The period of the oscillation can be measured between two peaks and must satisfy

$$\frac{2\pi}{T} = \frac{\sqrt{4km-b^2}}{2m}. \tag{2.19}$$

Finally, the rate of decay of the oscillations is given by the exponential factor in the solution. Measuring the amount of decay between two peaks, we have (using Exercise 2)

$$\log(q(t_1) - F_0/k) - \log(q_(t_2) - F_0/k) = \frac{b}{2m}(t_2 - t_1) \tag{2.20}$$

Using this set of three equations, we can solve for the parameters and determine that for the step response in Figure 2.11 we have $m \approx 250$ kg, $b \approx 60$ N-sec/m and $k \approx 40$ N/m. ∇

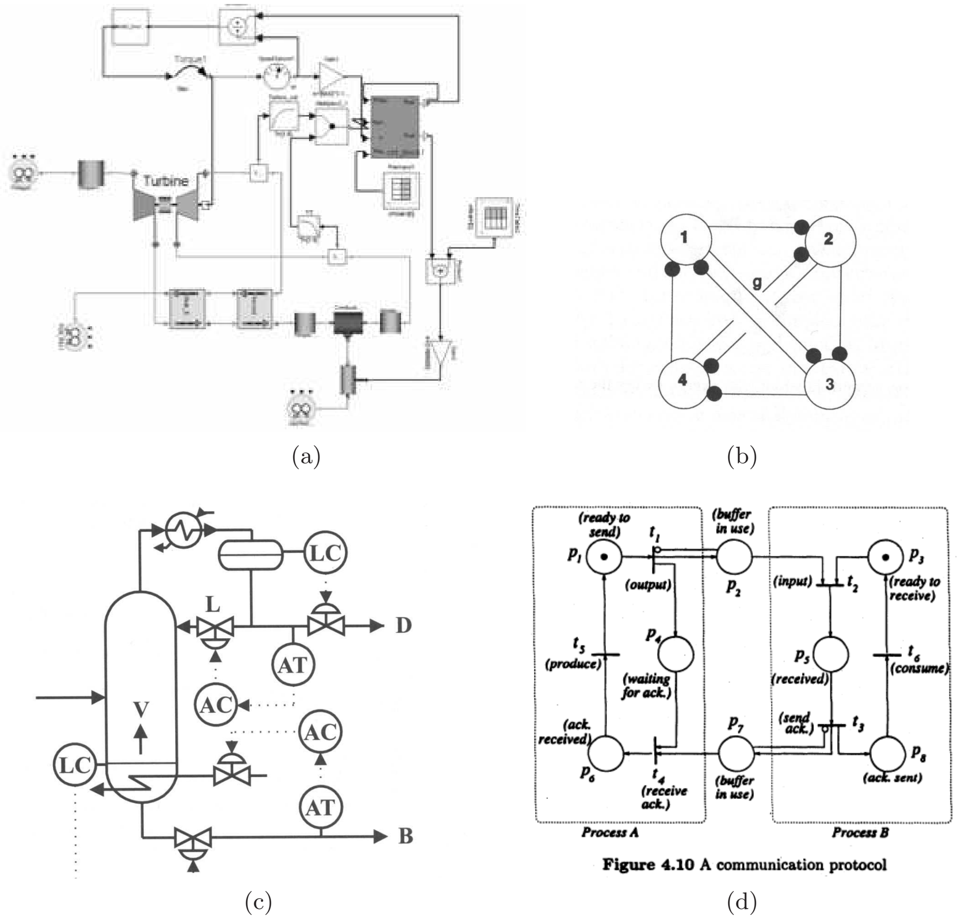
Modeling from experiments can also be done using many other signals. Sinusoidal signals are commonly used particularly for systems with fast dynamics and very precise measurements can be obtained by exploiting correlation techniques. An indication of nonlinearities can be obtained by repeating experiments with input signals having different amplitudes.

2.3 Schematic Diagrams

To deal with large complex systems, it is useful to have different representations of the system that capture the essential features and hide irrelevant details. In all branches of science and engineering, it is common practice to use some graphical description of systems. They can range from stylistic pictures to drastically simplified standard symbols. These pictures make it possible to get an overall view of the system and to identify the physical components. Examples of such diagrams are shown in Figure 2.12. Schematic diagrams are useful because they give an overall picture of a system, showing different physical processes and their interconnection, and indicating variables that can be manipulated and signals that can be measured.

Block Diagrams

A special graphical representation called *block diagrams* has been developed in control engineering. The purpose of block diagrams is to emphasize the information flow and to hide details of the system. In a block diagram, different process elements are shown as boxes and each box has inputs denoted by lines with arrows pointing toward the box and outputs denoted by lines with arrows going out of the box. The inputs denote the variables that influence a process and the outputs denote signals that we are interested in or signals that influence other subsystems. Block diagrams can also be organized in hierarchies, where individual blocks may themselves contain more detailed block diagrams.



(a) (b)

(c) (d)

Figure 2.12: Examples of schematic descriptions: (a) schematic picture of an micro gas turbine using Modelica, (b) neuronal network for respiratory control, (c) process and instrumentation diagram and (d) Petri net description of a communication protocol.

Figure 2.13 shows some of the notation that we use for block diagrams. Signals are represented as lines, with arrows to indicate inputs and outputs. The first diagram is the representation for a summation of two signals. An input/output response is represent as a rectangle with the system name (or mathematical description) in the block. Two special cases are a proportional gain, which scales the input by a multiplicative factor, and an integrator, which outputs the integral of the input signal.

Figure 2.14 illustrates the use of a block diagram, in this case for modeling the flight response of a fly. The flight dynamics of an insect are

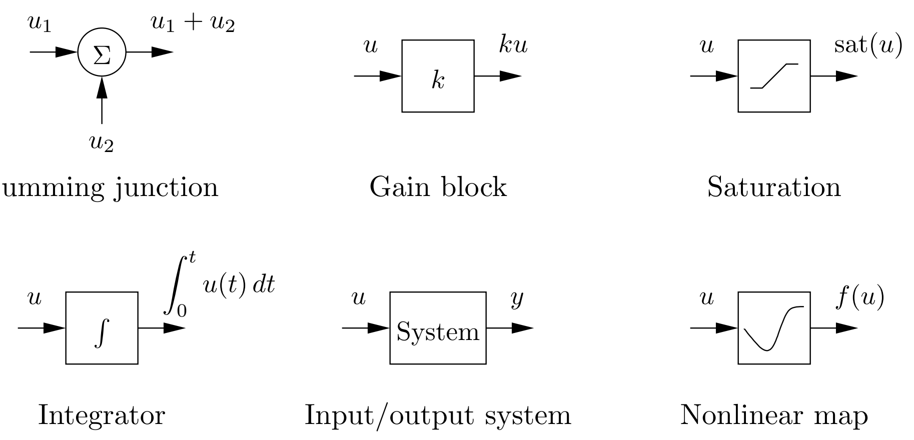


Figure 2.13: Some standard notation for block diagrams.

incredibly intricate, involving a careful coordination of the muscles within the fly to maintain stable flight in response to external stimuli. One known characteristic of flies is their ability to fly upwind by making use of the optical flow in their compound eyes as a feedback mechanism. Roughly speaking, the fly controls its orientation so that the point of contraction of the visual field is centered in its visual field.

To understand this complex behavior, we can decompose the overall dynamics of the system into a series of interconnected subsystems (or "blocks"). Referring to Figure 2.14, we can model the insect navigation system through an interconnection of five blocks. The sensory motor system (a) takes the information from the visual system (b) and generates muscle commands that attempt to steer the fly so that the point of contraction is centered.

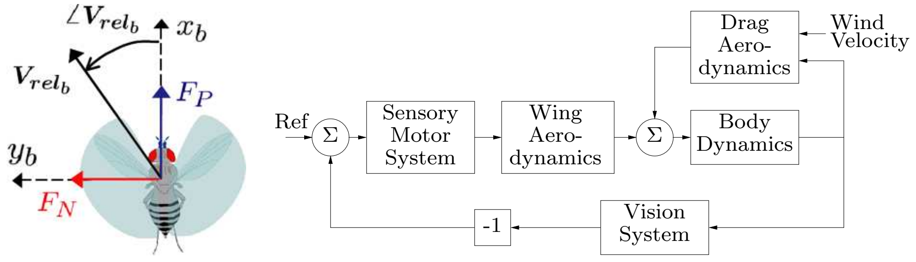


Figure 2.14: A block diagram representation of the flight control system for an insect flying against the wind.

These muscle commands are converted into forces through the flapping of the wings (c) and the resulting aerodynamic forces that are produced. The forces from the wings are combined with the drag on the fly (d) to produce a net force on the body of the fly. The wind velocity enters through the drag aerodynamics. Finally, the body dynamics (e) describe how the fly translates and rotates as a function of the net forces that are applied to it. The insect position, speed and orientation is fed back to the drag aerodynamics and vision systems blocks as inputs.

Each of the blocks in the diagram can itself be a very complicated subsystem. For example, the fly visual system of a tiny fruit fly consists of two complicated compound eyes (with about 700 elements per eye) and the sensory motor system has about 200,000 neurons that are used to process that information. A more detailed block diagram of the insect flight control system would show the interconnections between these elements, but here we have used one block to represent how the motion of the fly affects the output of the visual system and a second block to represent how the visual field is processed by the fly's brain to generate muscle commands. The choice of the level of detail of the blocks and what elements to separate into different blocks often depends on experience and the questions that one wants to answer using the model. One of the powerful features of block diagrams is their ability to hide information about the details of a system that may not be needed to gain an understanding of the essential dynamics of the system.

Modeling Tools

One of the reasons that block diagrams have emerged as a common representation of a model is the development of software tools for manipulating these diagrams. Modern modeling environments provide libraries of standard elements that can be interconnected to form more complex systems. We briefly describe two such environments here.

SIMULINK is a toolbox for MATLAB that allows the user to make use of either pre-defined or custom blocks that represent input/output components. Blocks can themselves be constructed from other blocks, allowing very complex models to be manipulated. SIMULINK allows continuous-time and discrete-time blocks to be interspersed, which is useful when building models of computer-controlled systems. Standard blocks include linear and nonlinear ordinary differential equations, summation and gain blocks, and common mathematical operations. Optional toolboxes allow SIMULINK diagrams to be compiled into machine executable code, so that controllers can

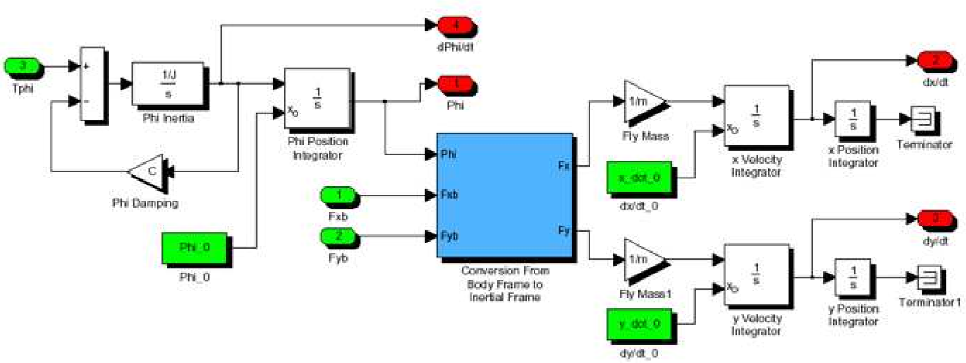


Figure 2.15: An example of a SIMULINK block diagram, corresponding to the body dynamics block of Figure 2.14.

be simulated in SIMULINK and then transferred to a hardware platform for implementation on a physical system. An example of a SIMULINK block diagram is shown in Figure 2.15. This diagram represents the insect body dynamics block of the larger block diagram in Figure 2.14.

LabVIEW is a graphical programming language developed by National Instruments that can be executed directly on a wide range of computer platforms and embedded targets. The Simulation Module is an add-on numerical simulation package that includes continuous and discrete-time blocks, nonlinear blocks, and various mathematical operations. All LabVIEW functions and toolkits can be used with the Simulation Module, allowing for both offline simulation and real-time control implementation of complex models. LabVIEW also has a scripting language, *MathScript*, and toolboxes which can be used run many of the examples in this book.

Models for large systems are often built by combining models of different subsystems. Block diagram modeling has severe drawbacks in this context, as discussed in Section 2.1. There is software for modeling and simulation tailored to specific domains that overcome these difficulties. Typical examples are SPICE for electrical circuits, Adams for multi-body mechanical systems, AUTOSIM for cars and SBML (Systems Biology Markup Language) for biological systems. *Modelica* that was discussed in Section 2.1 covers several domains. Dymola from Dynasim is an environment for modeling and simulation of complex systems based on *Modelica*.

2.4 Examples

In this section we introduce some additional examples that illustrate some of the different types of systems for which one can develop differential equation and difference equation models. These examples are specifically chosen from a range of different fields to highlight the broad variety of systems to which feedback and control concepts can be applied. A more detailed set of examples that serve as running examples throughout the text are given in the next chapter.

Motion Control Systems

Motion control system involve the use of computation and feedback to control the movement of a mechanical system. Motion control systems range from nano-positioning systems (atomic force microscopes, adaptive optics), to control systems for the read/write heads in a disk drive of CD player, to manufacturing systems (transfer machines and industrial robots), to automotive control systems (anti-lock breaks, suspension control, traction control), to air and space flight control systems (for airplanes, satellites, rockets and planetary rovers).

Example 2.8 (Vehicle steering)**.** Consider a vehicle with two wheels as shown in Figure 2.16. For the purpose of steering we are interested in a model that describes how the velocity of the vehicle depends on the steer angle δ. To be specific, we will consider the velocity at a point A at the distance a from the rear wheel. We take the wheel base to be b and let θ denote the heading angle and x and y be the coordinates of the point A as shown in Figure 2.16. Since $b = r_0 \tan u$ and $a = r_0 \tan \delta$ we get the following relation between α and the steer angle δ

$$\alpha = \arctan\Big(\frac{a \tan \delta}{b}\Big). \tag{2.21}$$

Assume that the wheels are rolling without slip, and that the velocity of the rear wheel is v_0. The vehicle speed at A is $v = v_0/\cos\alpha$ and we find that the velocity of point A on the vehicle is given by

$$\begin{aligned}\frac{dx}{dt} &= v\cos(\alpha+\theta) = v_0\frac{\cos(\alpha+\theta)}{\cos\alpha}\\ \frac{dy}{dt} &= v\sin(\alpha+\theta) = v_0\frac{\sin(\alpha+\theta)}{\cos\alpha}.\end{aligned} \tag{2.22}$$

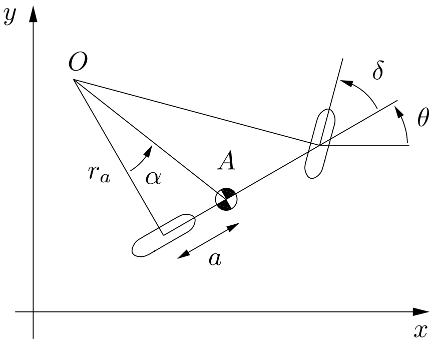


Figure 2.16: Schematic figure of a vehicle with two wheels. The steer angle is δ, and the heading angle is θ.

To see how the angle θ is influenced by the steer angle we observe from Figure 2.16 that the vehicle rotates with the angular velocity v_0/r_0 around the point O. Hence

$$\frac{d\theta}{dt} = \frac{v_0}{b}\tan\delta, \tag{2.23}$$

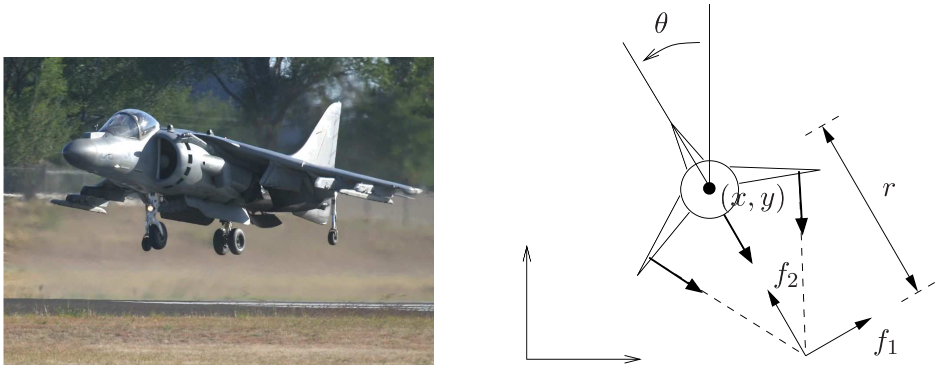
where α is a function of θ given by equation (2.21).

The simple kinematics model given by equations (2.21), (2.22) and (2.23) captures the essence of steering for many vehicles, including an automobile (with the approximate that the two front wheels can be a approximate by a single wheel at the center of the car). The assumption of no slip can be relaxed by adding an extra state variable gives a more realistic model. Such a model describes steering dynamics of cars and ships and pitch dynamics of aircrafts and missiles.

The situation in Figure 2.16 represents the situation when the vehicle moves forward and has front-wheel steering. The case when the vehicle reverses is obtained simply by changing the sign of the velocity. Changing the sign of the velocity also represents a vehicle with rear-wheel steering.

The simple kinematics model captures the essence of steering for many vehicles, including an automobile (with the approximate that the two front wheels can be a approximate by a single wheel at the center of the car). The assumption of no slip can be relaxed by adding an extra state variable gives a more realistic model. Such a model describes steering dynamics of cars and ships and pitch dynamics of aircrafts and missiles. ∇

Example 2.9 (Vectored thrust aircraft)**.** Consider the motion of vectored thrust aircraft, such as the Harrier "jump jet" shown Figure 2.17a. The



(a)

(b)

Figure 2.17: Vectored thrust aircraft: (a) Harrier AV-8B military aircraft and (b) a simplified planar model.

Harrier is capable of vertical takeoff by redirecting its thrust downward and through the use of smaller maneuvering thrusters located on its wings. A simplified model of the Harrier is shown in Figure 2.17b, where we focus on the motion of the vehicle in a vertical plane through the wings of the aircraft. We resolve the forces generated by the main downward thruster and the maneuvering thrusters as a pair of forces f_1 and f_s acting at a distance r below the aircraft (determined by the geometry of the engines).

Let (x, y, θ) denote the position and orientation of the center of mass of aircraft. Let m be the mass of the vehicle, J the moment of inertia, g the gravitational constant, and c the damping coefficient. Then the equations of motion for the fan are given by:

$$\begin{aligned} m\ddot{x} &= f_1 \cos\theta - f_2 \sin\theta - c\dot{x} \\ m\ddot{y} &= f_1 \sin\theta + f_2 \cos\theta - mg - c\dot{y} \\ J\ddot{\theta} &= r f_1. \end{aligned}$$

It is convenient to redefine the inputs so that the origin is an equilibrium point of the system with zero input. Letting $u_1 = f_1$ and $u_2 = f_2 - mg$, then the equations become

$$\begin{aligned} m\ddot{x} &= -mg\sin\theta - c\dot{x} + u_1 \cos\theta - u_2 \sin\theta \\ m\ddot{y} &= mg(\cos\theta - 1) - c\dot{y} + u_1 \sin\theta + u_2 \cos\theta \qquad (2.24) \\ J\ddot{\theta} &= r u_1. \end{aligned}$$

These equations described the motion of the vehicle as a set of three coupled second order differential equations. ∇

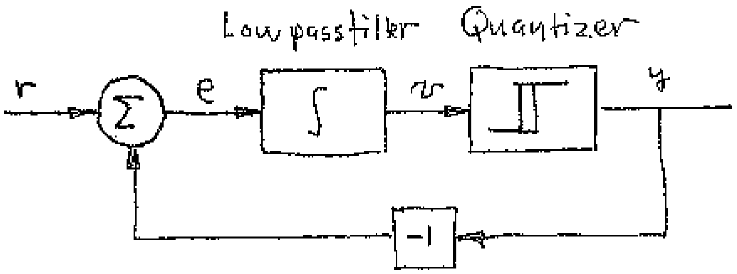


Figure 2.18: Schematic diagram of a delta-sigma converter.

Electronics and Instrumentation

Black's invention of the negative feedback amplifier paved the way for the use of feedback in electronic circuits. Electronics are ubiquitous in the world around us and many electronic devices involve feedback and control systems at a variety of levels. Some of the most common examples include video and audio systems, instrumentation systems, and a whole host control systems in transportation, manufacturing and communication systems.

Example 2.10 (Delta-sigma converters)**.** Delta-sigma converters are used for analog to digital conversion in high-quality audio and communication. Common examples are one-bit AD converters and digital audio amplifiers. Delta-sigma converters are also used to generate pulse-width modulated signals for motor drives. The converter generates an output signal with quantized amplitude that resembles the input signal in the sense that the filtered output is close to the input. In the extreme case of a one-bit converter the output has only two levels.

A schematic diagram of a delta-sigma converter is shown in Figure 2.18. The system is a feedback loop with a quantizer and a low pass filter. A particularly simple version is when the quantizer is a relay with hysteresis and the filter is an integrator. Figure 2.19 shows a simulation of such a converter when the input is a sinusoid. A feedback loop will normally act to make the error small. In this particular case the instantaneous value cannot be made small because the output switches between -1 and 1. The integral of the error is however small because it can be shown that

$$\int_{t_1}^{t_2} \Big| V_{in}(t) - V_{out}(t) \Big| dt \leq a, \tag{2.25}$$

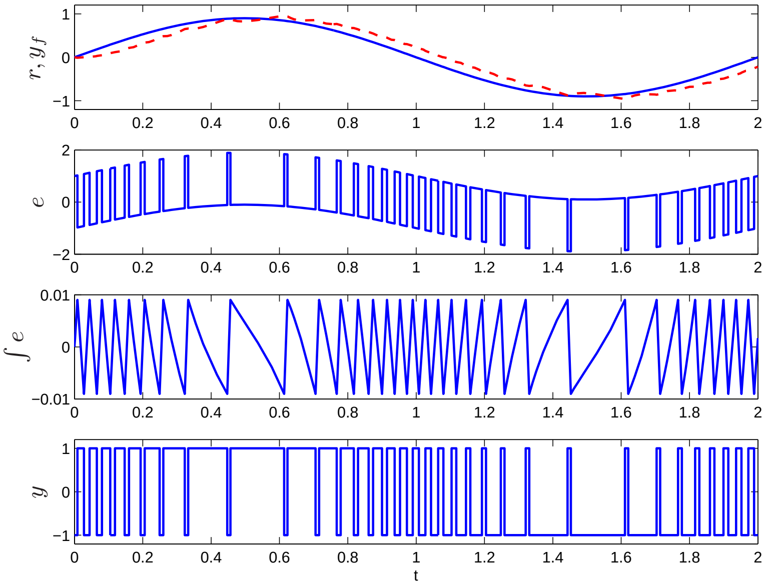


Figure 2.19: Simulation of a delta-sigma converter. The upper curve shows the input r (full) and the filtered output y_f (dashed), the next curves show the error e, the filtered error v and the converter output y. The loop filter is an integrator, the quantizer a relay with hysteresis $a = 0.009$. The pulse output y is filtered with a second order low-pass filter with time constant $T = 0.04s$.

where the interval (t_1, t_2) covers a full cycle, and a is the hysteresis of the relay. The filtered output y_f is also close to the input r.

Digital signals are formed by sampling in time and by quantization in amplitude. The delta-sigma modulator shows that a good digital representation can be obtained with a very crude quantization of the amplitude, only 0 and 1, provided that the time resolution is sufficiently high (oversampling). The pulsed output signal interesting signal form. It encodes the original continuous signal into a pulse-width modulated signal where the average value corresponds to the signal amplitude. The pulse width is proportional to the rate of change of the continuous signals. It is interesting to note that pulsed signals are common in biological systems. ∇

Example 2.11 (Josephson junction)**.** Josephson received the Nobel Prize in Physics 1973 for discovery of the Josephson effect which occurs in two superconducting layers separated by an insulating oxide. Under certain conditions current can pass through the insulator through tunneling of Cooper pairs

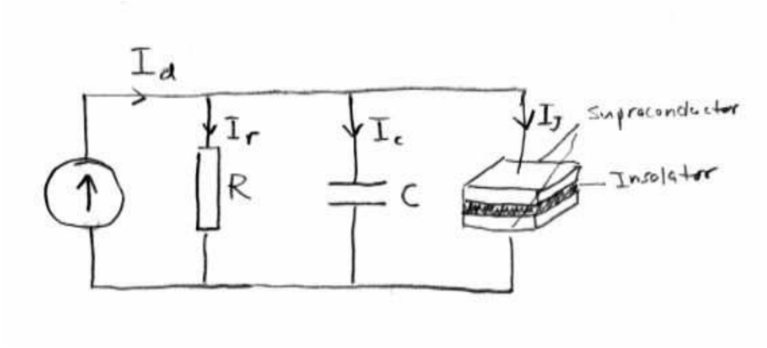


Figure 2.20: An electric circuit with a Josephson junction.

of electrons. The effect has been used to design superconducting quantum interference devices (SQUID), because switching is very fast, in the order of picoseconds. Tunneling in the Josephson junctions is very sensitive to magnetic fields and can therefore be used to measure extremely small magnetic fields, the threshold is as low as 10^{-14} T. Josephson junctions are also used for other precision measurements. The standard volt is now defined as the voltage required to produce a frequency of 483,597.9 GHz in a Josephson junction oscillator.

A schematic diagram of a circuit with a Josephson junction is shown in Figure 2.20. The quantum effects can be modeled by the Schrödinger equation. In spite of this it turns out that the circuit can be modeled as a system with lumped parameters. Let φ be the flux which is the integral of the voltage V across the device, hence

$$V = \frac{d\varphi}{dt}. \tag{2.26}$$

It follows from quantum theory, see Feynman [Fey70], that the current I through the device is a function of the flux φ

$$I = I_0 \sin k\varphi, \tag{2.27}$$

where I_0 is a device parameter, and the Josephson parameter k is given by

$$k = 4\pi\frac{e}{h}\ \mathrm{V}^{-1}\mathrm{s}^{-1} = 2\frac{e}{h}\ \mathrm{HzV}^{-1}, \tag{2.28}$$

where $e = 1.602 \times 10^{-19}$ C is the charge of an electron and $h = 6.62 \times 10^{-34}$ $\mathrm{V}^{-1}\mathrm{s}^{-1}$ is Planck's constant.

The circuit in Figure 2.20 has two storage elements the capacitor and the Josephson junction. We choose the states as the voltage V across the capacitor and the flux φ of the Josephson junction. Let I_R, I_C and I_J be the currents through the resistor, the capacitor and the Josephson junction. We have

$$I_R = \frac{V}{R}, \quad I_C = C\frac{dV}{dt}, \quad I_J = I_0 \sin k\varphi,$$

and a current balance gives

$$I_R + I_C + I_J = I_d,$$

which can be rewritten as

$$C\frac{dV}{dt} = I_d - \frac{V}{R} - I_0 \sin k\varphi.$$

Combining this equation with equation (2.26) gives the following state equation for the circuit

$$\begin{aligned} \frac{d\varphi}{dt} &= V \\ C\frac{dV}{d} &= -I_0 \sin k\varphi - \frac{V}{R} + I_d. \end{aligned} \tag{2.29}$$

Notice that apart from parameter values equation (2.29) is identical to the equation for the inverted pendulum given in equation (2.8). ∇

Information Systems

Information systems can range from communication systems for transmitting data from one location to another, to software systems that manipulate data or manage enterprise-wide resources, to economies and financial markets, that use prices to reflect current and future value. Feedback is present in all of these systems, although it is often not directly visible.

Example 2.12 (Congestion control)**.** The Internet was created to obtain a large, highly decentralized, efficient, and expandable communication system. The system consists of a large number of interconnected gateways. A message is split into several packets that are transmitted over different paths in the network. The packages are joined to recover the message at the receiver. A message is sent back to the sender when a packet is received. The operation of the system is governed by a simple but powerful decentralized control structure which evolved over time.

The system is governed by two control mechanisms, called protocols: the Transmission Control Protocol (TCP) for end-to-end network communication and the Internet Protocol (IP) for routing packets and for host-to-gateway or gateway-to-gateway communication. The current protocols evolved after some spectacular congestion collapses in the mid 1980s, when throughput unexpectedly could drop by a factor of 1000. The control mechanism in TCP is based on conserving the number of packets in the loop from sender to receiver and back to the sender. The sending rate is increased exponentially when there is no congestion and it is dropped drastically to a very low level when there is congestion.

A simple model for congestion control between N computers connected by a single router is given by the differential equation

$$\begin{aligned} \frac{dx_i}{dt} &= -b\frac{x_i^2}{2} + (b_{\max} - b) \\ \frac{db}{dt} &= \sum_{i=1}^{N} x_i - c, \end{aligned} \tag{2.30}$$

where $x_i \in \mathbb{R}$, $i = 1, \ldots, N$ are the transmission rates for the sources of data, $b \in \mathbb{R}$ is the current buffer size of the router, $b_{\max} > 0$ is the maximum buffer size, and $c > 0$ is the capacity of the link connecting the router to the computers. The $\dot{x}_i$ equation represents the control law that the individual computers use to determine how fast to send data across the network (this version is motivated by a protocol called "Reno") and the $\dot{b}$ equation represents the rate at which the buffer on the router fills up.

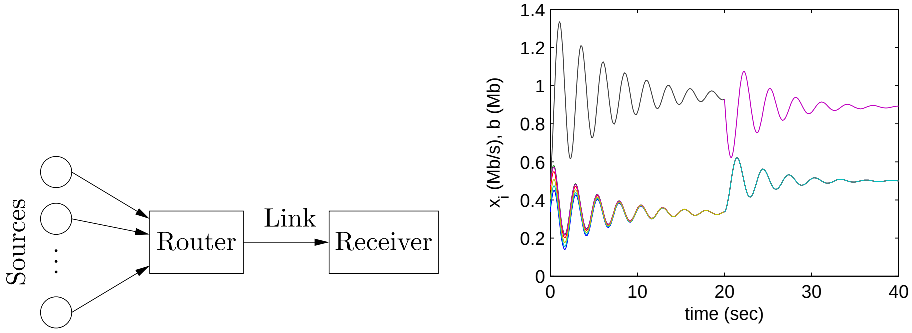
The nominal operating point for the system can be found by setting $\dot{x}_i = \dot{b} = 0$:

$$\begin{aligned} 0 &= \frac{x_i^2}{2} + \left(1 - \frac{b_{\max}}{b}\right) \quad \text{for all } i \\ 0 &= \sum_{i=1}^{N} x_i - c \end{aligned}$$

From the first equation we notice that the equilibria for all the x_i should be the same and it follows that there is a unique equilibrium

$$\begin{aligned} x_i^* &= \frac{c}{N} \quad \text{for all } i \\ b^* &= \frac{2N^2 b_{\max}}{2N^2 + c^2}, \end{aligned}$$

which corresponds to each of the sources sending data at rate c/N and the buffer size in the router staying constant.



(a)

(b)

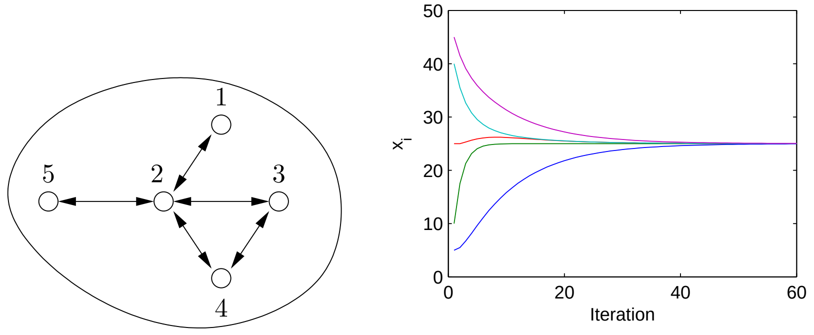
Figure 2.21: Congestion control simulation: (a) Multiple sources attempt to communicate through a router across a single link. (b) Simulation with 6 sources starting random rates, with 2 sources dropping out at $t = 20$ s.

Figure 2.21 shows a simulation of 6 sources communicating across a single link, with two sources dropping out at $T = 1$ s and the remaining courses increasing their rates to compensate. Note that the solutions oscillate before approaching their equilibrium values, but that the transmission rates and buffer size automatically adjust depending on the number of sources.

A good presentation of the ideas behind the control principles for the Internet are given by one of its designers in [Jac88]. The paper [Kel85] is an early effort of analysis of the system. The book [HDPT04] gives many interesting examples of control of computer systems. ∇

Example 2.13 (Consensus protocols in sensor networks)**.** Sensor networks are used in a variety of applications where we want to collect and aggregate information over a region of space using multiple sensors that are connected together via a communications network. Examples include monitoring environmental conditions in a geographical area (or inside a building), monitoring movement of animals or vehicles, or monitoring the resource loading across a group of computers. In many sensor networks the computational resources for the system are distributed along with the sensors and it can be important for the set of distributed agents to reach a consensus about a certain property across the network, such as the average temperature in a region or the average computational load amongst a set of computers.

To illustrate how such a consensus might be achieved, we consider the problem of computing the average value of a set of numbers that are locally available to the individual agents. We wish to design a "protocol" (algorithm) such that all agents will agree on the average value. We consider the



(a)

(b)

Figure 2.22: Consensus protocols for sensor networks: (a) a simple sensor network with five notes; (b) simulation demonstrating convergence of the network to the average value of the initial conditions.

case in which all agents cannot necessarily communicate with each other directly, although we will assume that the communications network is connected (meaning that no agents are completely isolated from the group). Figure 2.22a shows a simple situation of this type.

We model the connectivity of the sensor network using a graph, with nodes corresponding to the sensors and edges corresponding to the existence of a direct communications link between two nodes. For any such graph, we can build an *adjacency matrix*, where each row and column of the matrix corresponds to a node and a 1 in the respective row and column indicates that the two nodes are connected. For the network shown in Figure 2.22a, the corresponding adjacency matrix is

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

We also use the notation $\mathcal{N}_i$ to represent the set of neighbors of a node i. For example, $\mathcal{N}_2 = \{1, 3, 4, 5\}$ and $\mathcal{N}_3 = \{2, 4\}$.

To solve the consensus problem, we let x^i be the state of the ith sensor, corresponding to that sensor's estimate of the average value that we are trying to compute. We initialize the state to the value of the quantity measured by the individual sensor. Our consensus protocol can now be realized as a local update law of the form

$$x^i_{k+1} = x^i_k + \gamma \sum_{i \in \mathcal{N}_i} (x^j_k - x^i_k). \tag{2.31}$$

This protocol attempts to compute the average by updating the local state of each agent based on the value of its neigbors. The combined dynamics of all agents can be written in the form

$$x_{k+1} = x_k - \gamma(D - A)x_k \tag{2.32}$$

where A is the adjacency matrix and D is a diagonal matrix whose entries correspond to the number of neighbors of the corresponding node. The constant γ describes the rate at which we update our own estimate of the average based on the information from our neighbors. The matrix $L := D - A$ is called the *Laplacian* of the graph.

The equilibrium points of equation (2.32) are the set of states such that $x^*_{k+1} = x^*_k$. It is easy to show that $x^* = \alpha(1, 1, \dots, 1)$ is an equilibrum state for the system, corresponding to each sensor having an identical estimate α for the average. Furthermore, we can show that α is the precisely the average value of the initial states. To see this, let

$$W_k = \frac{N}{\sum}{}^{n}_{i=1} x^i_k$$

where N is the number of nodes in the sensor network. W_0 is the average of the initial states of the network, which is the average quantity we are trying to compute. W_k is given by the difference equation

$$W_{k+1} = \frac{1}{N} \sum_{i=1}^{n} x^i_{k+1} = \frac{1}{N} \sum_{i=1}^{n} \bigl(x^i_k + \gamma \sum_{j \in \mathcal{N}_i} (x^j_k - x^i_k) \bigr).$$

Since $i \in \mathcal{N}_j$ implies that $j \in \mathcal{N}_i$, it follows that each term in the second sum occurs twice with opposite sign. Thus we can conclude that $W_{k+1} = W_k$ and hence $W_k = W_0$ for all k, which implies that at the equilibrium point α must be W_0, the average of the initial states. W is called an *invariant* and the use of invariants is an important technique for verifying correctness of computer programs.

Having shown that the desired consensus state is an equilibrium point for our protocol, we still must show that the algorithm actually converges to this state. Since there can be cycles in the graph, it is possible that the state of the system could get into an “infinite loop” and never converge to the desired consensus state. A formal analysis requires tools that will be introduced later in the text, but it can be shown that for any given graph, we can always find a γ such that the states of the individual agents converge to the average. A simulation demonstrating this property is shown in Figure 2.22b.

Although we have focused here on consensus to the average value of a set of measurements, other consensus states can be achieved through choice of appropriate feedback laws. Examples include finding the maximum or minimum value in a network, counting the number of nodes in a network, and computing higher order statistical moments of a distributed quantity. ∇

Biological Systems

Biological systems are filled with feedback loops and provide perhaps the richest source of feedback and control examples. The basic problem of homeostasis, in which a quantity such as temperature or blood sugar level is regulated to a fixed value, is but one of the many types of complex feedback interactions that can occur in molecular machines, cells, organisms and ecosystems.

Example 2.14 (Transcriptional regulation)**.** Transcription is the process by which mRNA is generated from a segment of DNA. The promoter region of a gene allows transcription to be controlled by the presence of other proteins, which bind to the promoter region and either repress or activate RNA polymerase (RNAP), the enzyme that produces mRNA from DNA. The mRNA is then translated into a protein according to its nucleotide sequence.

A simple model of the transcriptional regulation process is the use of a Hill function [dJ02, Mur04]. Consider the regulation of a protein A with concentration given by p_A and corresponding mRNA concentration m_A. Let B be a second protein with concentration p_B that represses the production of protein A through transcriptional regulation. The resulting dynamics of p_A and m_A can be written as

$$\begin{aligned}\frac{dm_A}{dt} &= -\tau m_A + \frac{\alpha}{1+p_B^n} + \alpha_0 \\ \frac{dp_A}{dt} &= \beta(m_A - p_A),\end{aligned} \tag{2.33}$$

where $\alpha + \alpha_0$ is the basal transcription rate, τ represents the rate of degredation of mRNDA, α and n are parameters that describe how B represses

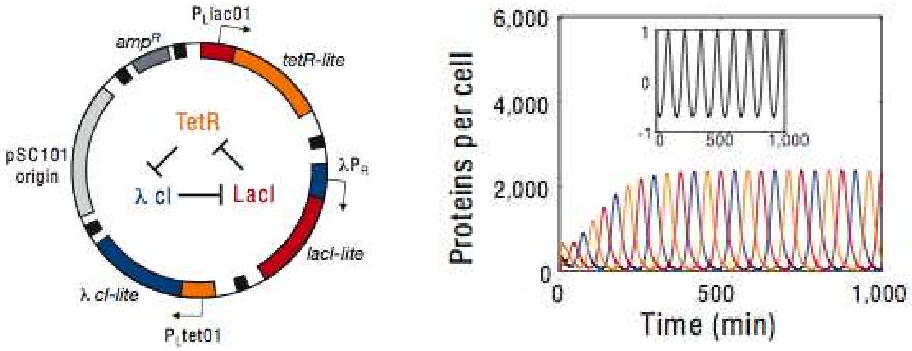


Figure 2.23: The repressilator genetic regulatory network: (a) a schematic diagram of the repressilator, showing the layout of the genes in the plasmid that holds the circuit as well as the circuit diagram (center); (b) simulation of a simple model of the repressilator.

A and β represents both the rate of production of the protein from its corresponding mRNA and also the rate of degradation of A. The parameter α_0 describes the "leakiness" of the promotor and n is called the Hill coefficient and relates to the cooperativity of the promotor. For simplicity we will assume that $\tau = 1$, which corresponds to choosing units of time that correspond to the mRNA decay rate.

A similar model can be used when a protein activates the production of another protein, rather than repressing it. In this case, the equations have the form

$$\begin{aligned}\frac{dm_A}{dt} &= -\tau m_A + \frac{\alpha p_B^n}{1+p_B^n} + \alpha_0 \\ \frac{dp_A}{dt} &= \beta(m_A - p_A),\end{aligned} \tag{2.34}$$

where the variables are the same as described. Note that in the case of the activator, if p_B is zero then the production rate is α_0 (versus $\alpha + \alpha_0$ for the repressor). As p_B gets large, the second term in the expression for $\dot{m}_A$ approaches 1 and the transcription rate becomes $\alpha + \alpha_0$ (versus α_0 for the repressor). Thus we see that the activator and repressor act in opposite fashion from each other.

As an example of how these models can be used, we consider the model of a "repressilator", originally due to Elowitz and Leibler [EL00]. The repressilator is a synthetic circuit in which three proteins each repressor another in a cycle. This is shown schematically in Figure 2.23a, where the three proteins are tetR, λ cI and LacI. The basic idea of the repressilator is that if tetR is present then it represses the production of λ cI. If λ cI is represent, then LacI is produced (at the basal transcription rate), which in turn represses TetR. Once TetR is repressed then λ cI is no longer repressed and so on. If the dynamics of the circuit are designed properly, the resulting protein concentrations will oscillate.

We can model this system using three copies of equation (2.33), with A and B replaced by the appropriate combination of TetR, cI and LacI. The state of the system is then given by $x = (m_{\text{TetR}}, p_{\text{TetR}}, m_{\text{cI}}, p_{\text{cI}}, m_{\text{LacI}}, p_{\text{LacI}})$. Figure 2.23b shows the traces of the three protein concentrations for parameters $\alpha_0 = 0$, $\alpha = 50$, $\beta = 0.2$ and $n = 2$ and initial conditions $x(0) = 0.2, 0.1, 0.1, 0.4, 0.3, 0.5)$ (from [EG05]). ∇

Example 2.15 (Hodgkin-Huxley equations[1])**.** The dynamics of the membrane potential in a cell is a fundamental mechanism in discussing signaling in cells. The Hodgkin-Huxley equations provide a simple model for studying propagation waves in networks of neurons. The model for a single neuron has the form

$$C\frac{dV}{dt} = -I_{Na} - I_K - I_{leak} + I_{input}$$

where V is the membrane potential, C the capacitance, I_{Na} and I_K the current caused by transport of sodium and potassium across the cell membrane, I_{leak} is a leakage current ant I_{input} is the external stimulation of the cell. Each current obeys Ohms law, i.e.

$$I = g(V - E)$$

where g is the conductance and E the equilibrium voltage. The equilibrium voltage is given by Nernst's law

$$E = \frac{RT}{xF}\log(C_{out}/C_{in})$$

where R is Boltzmann's constant, T the absolute temperature, F Faraday's constant, C_{out} and C_{in} the ion concentrations outside and inside the cell. At 20° we have $RT/F = 20$ mV.

The Hodgkin-Huxley model was originally developed as a means to predict the quantitative behavior of the squid giant axon [**?**]. Hodgkin and Huxley shared the 1963 Nobel Prize in Physiology (along with J. C. Eccles) for analysis of the electrical and chemical events in nerve cell discharge. ∇

[1] H. R. Wilson, *Spikes, Decisions and Actions—Dynamical Foundations of Neuroscience.* Oxford University Press.

2.5 Further Reading

Modeling is ubiquitous in engineering and science and has a long history in applied mathematics. For example, the Fourier series was introduced in connection with modeling of heat conduction in solids. Models of dynamics have been developed in many different fields, including mechanics [Gol53], heat conduction [CJ59], fluids![BS60], vehicles [Abk69, Bla91, Ell94], circuit theory [Gui63], acoustics [Ber54] and micromechanical systems [Sen01]. Control theory requires modeling from many different domains and most texts control theory contain several chapters on modeling using ordinary differential equations and difference equations (see, for example, [FPEN05]).

A classic book on modeling of physical systems, especially mechanical, electrical and thermo-fluid systems, is Cannon's *Dynamics of Physical Systems* [Can03]. Two of the authors' favorite books on modeling of biological systems are *Mathematical Biology* by J. D. Murray [Mur04] and *Spikes, Decision and Actions: The Dynamical Foundations of Neuroscience* by H. R. Wilson [Wil99]. For readers interested in learning more about object oriented modeling and Modelica, the edited volume by Tiller [Til01] provides an excellent introduction.

2.6 Exercises

1. Use the equations of motion for a balance system to derive a dynamic model for the inverted pendulum described in Example 2.2 and verify that for small θ they are approximated by equation (2.8).

2. (Second order system identification) Verify that equation (2.20) in Example 2.7 is correct and use this formula and the others in the example to compute the parameters corresponding to the step response in Figure 2.11.

3. ☡ (Least squares system identification) Consider a nonlinear differential equation that can be written in the form

$$\frac{dx}{dt} = \sum_{i=1}^{M} \alpha_i f_i(x)$$

where $f_i(x)$ are known nonlinear functions and α_i are unknown, but constant, parameters. Suppose that we have measurements (or estimates) of the state x at time instants $t_1, t_2, \ldots, t_N$, with $N > M$.

Show that the parameters α_i can be determined by finding the least squares solution to a linear equation of the form

$$H\alpha = b$$

where $\alpha \in \mathbb{R}^M$ is the vector of all parameters and $H \in \mathbb{R}^{N\times M}$ and $b \in \mathbb{R}^N$ are appropriately defined.

4. Consider the following discrete time system

$$\begin{aligned} z_{k+1} &= Az_k + Bu_k \\ y_k &= Cz_k \end{aligned}$$

where

$$z = \begin{pmatrix} z^1 \\ z^2 \end{pmatrix} \qquad A = \begin{pmatrix} a_{11} & a_{12} \\ 0 & a_{22} \end{pmatrix} \qquad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad C = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

In this problem, we will explore some of the properties of this discrete time system as a function of the parameters, the initial conditions, and the inputs.

(a) Assume that the off diagonal element $a_{12} = 0$ and that there is no input, $u = 0$. Write a closed form expression for the output of the system from a nonzero initial condition $z_0 = (z_0^1, z_0^2)$ and give conditions on a_{11} and a_{22} under which the output gets smaller as k gets larger.

(b) Now assume that $a_{12} \neq 0$ and write a closed form expression for the response of the system from a nonzero initial conditions. Given a condition on the elements of A under which the output gets smaller as k gets larger.

(c) Write a MATLAB program to plot the output of the system in response to a unit step input, $u[k] = 1$, $k \geq 0$. Plot the response of your system with $z_0 = 0$ and A given by

$$A = \begin{pmatrix} 0.5 & 1 \\ 0 & 0.25 \end{pmatrix}$$

5. Consider the delta-sigma converter in Example 2.10. Propose a way to obtain an estimate of the instantaneous value of the reference signal and its derivative from the pulsed output.

6. Consider the linear ordinary differential equation (2.6). Show that by choosing a state space representation with $x_1 = y$, the dynamics can be written as

$$A = \begin{pmatrix} 0 & 1 & & 0 \\ 0 & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \\ -a_n & -a_{n-1} & & -a_1 \end{pmatrix} \qquad B = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}$$

$$C = \begin{pmatrix} 1 & \dots & 0 & 0 \end{pmatrix}$$

This canonical form is called *chain of integrators* form.