Chapter Two
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 dynamical 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 we will often work 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.

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.
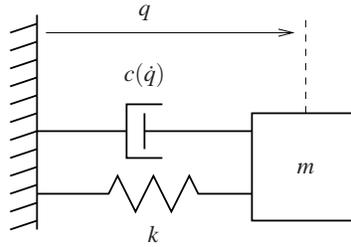
Figure 2.1: Spring-mass system, with nonlinear damping. The position of the mass is denoted by q, with $q = 0$ corresponding to the rest position of the spring. The forces on the mass are generated by a linear spring with spring constant k and a damper with force dependent on the velocity $\dot{q}$.

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 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.

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 spring-mass 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 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.

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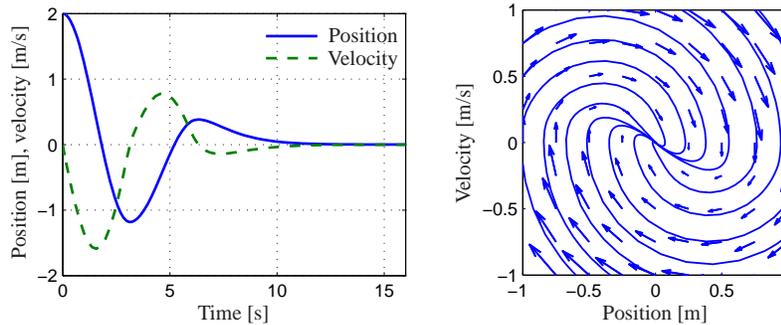
Figure 2.2: Illustration of a state model. A state model gives the rate of change of the state as a function of the state. The plot on the left shows the evolution of the state as a function of time. The plot on the right shows the evolution of the states relative to each other, with the velocity of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

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 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

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

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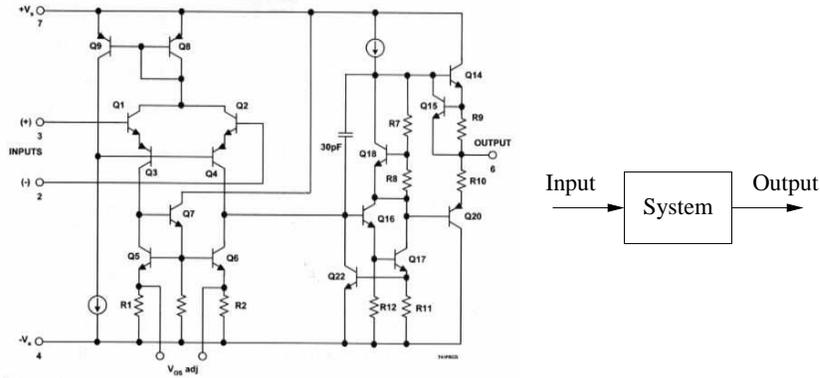
Figure 2.3: Illustration of the input/output view of a dynamical system. The figure on the left shows a detailed circuit diagram for an electronic amplifier; the one of the right its representation as a block diagram.

of inputs and outputs. Given an input signal $u(t)$ over some interval of time, 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.

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 very 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 frequency response is that we can completely
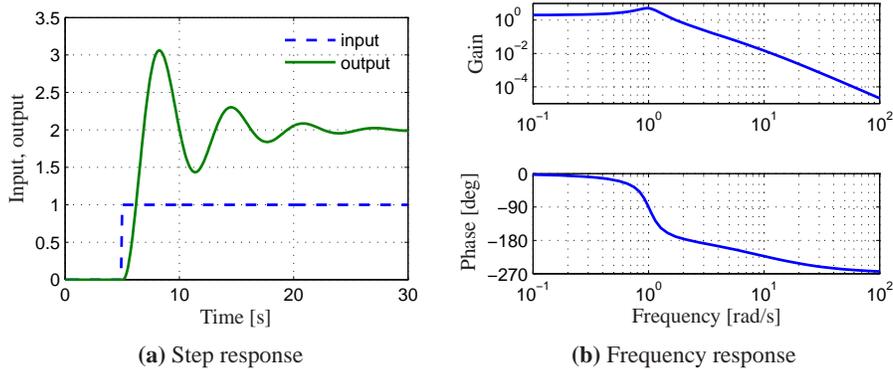
(a) Step response

(b) Frequency response

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.

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.

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. The emergence of space flight is a typical example, where precise control of the orbit of a spacecraft is essential. These two points of view gradually merged into what is today the state space representation of input/output systems.

The development of state space models involved modifying the models from mechanics to include external actuators and sensors, and utilizing more general forms of equations. In control, the model given by equation (2.2) was replaced by

$$\frac{dx}{dt} = f(x,u), \qquad y = h(x,u), \tag{2.3}$$

where x is a vector of state variables, u is a vector of control signals, and y a vector of measurements. The term dx/dt 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. For mechanical systems, the state consists of

the position and velocity of the system, so that $x = (q, \dot{q})$ in the case of a damped spring-mass system. Note that in the control formulation we model dynamics as first order differential equations, but we will see that this can capture the dynamics of higher order differential equations by appropriate definition of the state and the maps f and h.

Adding inputs and outputs 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 more realistic approach is to model disturbances as 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 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 systems 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 precompute 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 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 can be 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 model such multi-domain systems, we start by partitioning a system into smaller subsystems. Each subsystem is represented 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 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 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.

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 add the algebraic equation expressing that the voltages across the capacitors are the same.

Modelica is a language that has been developed to support component-based modeling. Differential algebraic equations are used as the basic description and 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 used 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 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 predicting the future. For a physical 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

$$\frac{dx}{dt} = f(x,u), \qquad y = h(x,u), \tag{2.5}$$

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 (2.5) is called *time-invariant* because the functions f and g do not depend explicitly on time t; there are more general time-varying systems where the functions do depend on time. The model consists of two functions: the function f gives the rate of change 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 a *linear* state space system if the functions f and g are linear in x and u. A linear state space system can thus be represented by

$$\frac{dx}{dt} = Ax + Bu, \qquad y = Cx + Du, \tag{2.6}$$

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

$$\frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \cdots + a_n y = u, \tag{2.7}$$

where t is the independent (time) variable, $y(t)$ is the dependent (output) variable, and $u(t)$ is the input. The notation $d^k y / dt^k$ is used to denote the kth derivative of y with respect to t, sometimes also written as $y^{(k)}$. The system (2.7) is said to be an

nth order system. This system can be converted into state space form by defining

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} d^{n-1}y/dt^{n-1} \\ d^{n-2}y/dt^{n-2} \\ \vdots \\ dy/dt \\ y \end{pmatrix}$$

and the state space equations become

$$\frac{d}{dt} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} -a_1 x_1 - \cdots - a_n x_n \\ x_2 \\ \vdots \\ x_{n-2} \\ x_{n-1} \end{pmatrix} + \begin{pmatrix} u \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}, \qquad y = x_n.$$

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

$$\frac{d}{dt} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} -a_1 & -a_2 & \cdots & -a_{n-1} & -a_n \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & & 1 & 0 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} u \qquad (2.8)$$

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

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
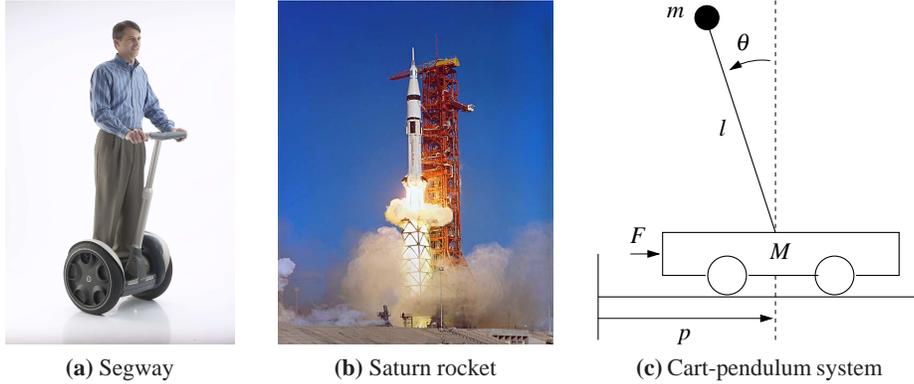
(a) Segway   **(b)** Saturn rocket   **(c)** Cart-pendulum system

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

their hand.

Balance systems are a generalization of the spring-mass system we saw earlier. We can write the dynamics for a mechanical system in the general form

$$M(q)\ddot{q} + C(q,\dot{q}) + K(q) = B(q)u,$$

where $M(q)$ is the inertia matrix for the system, $C(q,\dot{q})$ represents the Coriolis forces as well as the damping, $K(q)$ gives the forces due to potential energy and $B(q)$ describes how the external applied forces couple into the dynamics. The specific form of the equations can be derived using Newtonian mechanics. Note that each of the terms depends on the configuration of the system q and these terms are often nonlinear in the configuration variables.

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

$$\begin{pmatrix} (M+m) & -ml\cos\theta \\ -ml\cos\theta & (J+ml^2) \end{pmatrix} \begin{pmatrix} \ddot{p} \\ \ddot{\theta} \end{pmatrix} + \begin{pmatrix} c\dot{p} + ml\sin\theta\,\dot{\theta}^2 \\ \gamma\dot{\theta} - mgl\sin\theta \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix}, \qquad (2.9)$$

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

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

the equations of motion then become

$$\frac{d}{dt} \begin{pmatrix} p \\ \theta \\ \dot{p} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \dot{p} \\ \dot{\theta} \\ \dfrac{-mls_\theta\dot{\theta}^2 + mg(ml^2/J_t)s_\theta c_\theta - c\dot{p} - \gamma lmc_\theta\dot{\theta} + u}{M_t - m(ml^2/J_t)c_\theta^2} \\ \dfrac{-ml^2 s_\theta c_\theta \dot{\theta}^2 + M_t gls_\theta - clc_\theta\dot{p} - \gamma(M_t/m)\dot{\theta} + lc_\theta u}{J_t(M_t/m) - m(lc_\theta)^2} \end{pmatrix}$$

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

where we have used the shorthand $c_\theta = \cos\theta$ and $s_\theta = \sin\theta$.

In many cases, the angle θ will be very close to 0 and hence we can use the approximations $\sin\theta \approx \theta$ 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

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

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

where $\mu = M_t J_t - m^2 l^2$. $\qquad\qquad \nabla$

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 are given by

$$\frac{d}{dt} \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \dot{\theta} \\ \dfrac{mgl}{J_t}\sin\theta - \dfrac{\gamma}{J_t}\dot{\theta} + \dfrac{l}{J_t}\cos\theta\, u \end{pmatrix}, \qquad y = \theta, \qquad (2.10)$$

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*. $\qquad \nabla$

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 $k = 0, 1, 2, \ldots$, then we can ask how the state of the system changes for each k. Just as in the case of differential equations, we 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

$$x[k+1] = f(x[k], u[k]), \qquad y[k] = h(x[k], u[k]) \tag{2.11}$$

where $x[k] \in \mathbb{R}^n$ is the state of the system at "time" k (an integer), $u[k] \in \mathbb{R}^p$ is the input and $y[k] \in \mathbb{R}^q$ is the output. As before, f and h are smooth mappings of the appropriate dimension. We call equation (2.11) 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 quantity; in the case of the latter we write $x_j[k]$ for the value of the jth 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

$$x[k+1] = Ax[k] + Bu[k], \qquad y[k] = Cx[k] + Du[k].$$

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 $u[0], \ldots, u[T]$ is given by

$$\begin{aligned} x[k] &= A^k x_0 + \sum_{j=0}^{k-1} A^{k-j-1} Bu[j] \\ y[k] &= CA^k x_0 + \sum_{j=0}^{k-1} CA^{k-j-1} Bu[j] + Du[k] \end{aligned} \qquad k > 0. \tag{2.12}$$

Difference equations are also useful as an approximation of differential equations, as we will show later.

Example 2.3 Predator-prey
As an example of a discrete time system, 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 interesting dynamics. Figure 2.6 shows a historical record taken over 50 years in a 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
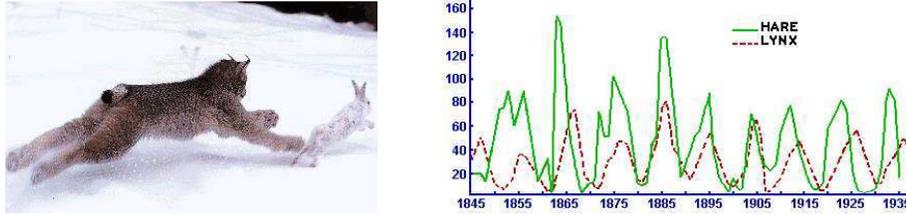
Figure 2.6: Predator versus prey. The photograph on the left shows a Canadian lynx and a snowshoe hare, the lynx's primary prey. The graph on the right shows the populations of hares and lynxes between 1845 and 1935 in a section of the Canadian Rockies [Mac37, MS93]. The data were collected on an annual basis over a period of 90 years. Photograph courtesy Rudolfo's Usenet Animal Pictures Gallery.

be the discrete time index (e.g., the day number), we can write

$$H[k+1] = H[k] + b_r(u)H[k] - aL[k]H[k]$$
$$L[k+1] = L[k] - d_f L[k] + cL[k]H[k],$$

(2.13)

where $b_r(u)$ is the hare birth rate per unit period and as a function of the food supply u, d_f is the lynx death rate, and a and c are the interaction coefficients. The interaction term $aL[k]H[K]$ models the rate of predation, which is assumed to be proportional to the rate at which predators and prey meet and is hence given by the product of the population sizes. The interaction term $cL[k]H[k]$ in the lynx dynamics has a similar form and represents the rate of growth of the lynx population. This model makes many simplifying assumptions—such as the fact that hares only decrease in numbers through predation by lynxes—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 at each time point from some initial population. This is done by starting with $x[0] = (H_0, L_0)$ and then using equation (2.13) to compute the populations in the following period. 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 and hence we can use the model to help explore the dynamics of the system. ∇

Example 2.4 Email Server
The IBM Lotus server is an collaborative software system that administers users' e-mail, documents and notes. Client machines interact with end users to provide access to data and applications. The server also handles other administrative tasks. In the early development of the system it was observed that the performance was poor when the CPU was overloaded because of too many service requests and mechanisms to control the load were therefore introduced.

The interaction between the client and the server is in the form of remote procedure calls (RPCs). The server maintains a log of statistics of completed requests.
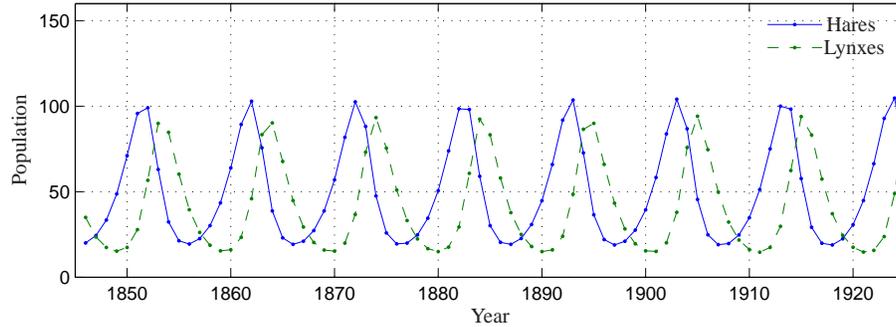
Figure 2.7: Discrete time simulation of the predator-prey model (2.13). Using the parameters $a = c = 0.007$, $b_r(u) = 0.7$ and $d = 0.5$ in equation (2.13), the period and magnitude of the lynx and hare population cycles approximately match the data in Figure 2.6.

The total number of requests being served, called RIS (RPCs in server), is also measured. The load on the server is controlled by a parameter called MaxUsers, which sets the total number of client connections to the server. This parameter is controlled by the system administrator. The server can be regarded as a dynamical system with MaxUsers as input and RIS as the output. The relationship between input and output was first investigated by exploring the steady state performance and was found to be linear.

In [HDPT04] a dynamic model in the form of a first order difference equation is used to capture the dynamic behavior of this system. Using system identification techniques they construct a model of the form

$$y[k+1] = ay[k] + bu[k],$$

where $u = \texttt{MaxUsers} - \overline{\texttt{MaxUsers}}$ and $y = \texttt{RIS} - \overline{\texttt{RIS}}$. The parameters $a = 0.43$ and $b = 0.47$ are parameters that describe the dynamics of the system around the operating point and $\overline{\texttt{MaxUsers}} = 165$ and $\overline{\texttt{RIS}} = 135$ represent the nominal operating point of the system. The number of requests was averaged of the sampling period which was 60 s.   ∇

Simulation and Analysis

State space models can be used to answer many questions. One of the most common, as we have seen 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.

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.
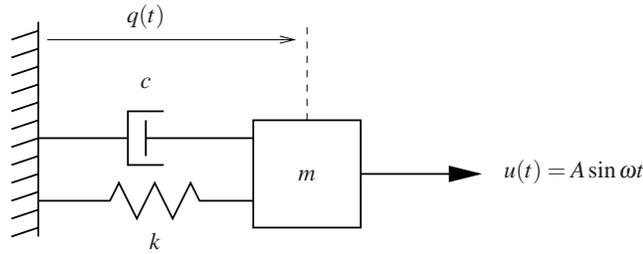
Figure 2.8: A driven spring-mass system, with damping. Here we use a linear damping element with coefficient of viscous friction c. The mass is driven with a sinusoidal force of amplitude A.

We choose to model the system with 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

$$m\ddot{q} + c\dot{q} + kq = u, \tag{2.14}$$

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 u is the applied force. In state space form, using $x = (q, \dot{q})$ as the state and choosing $y = q$ as the output, we have

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

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 a 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 $h > 0$ later by assuming that the rate of change of $f(x, u)$ is constant over the interval t to $t + h$. This gives

$$x(t + h) = x(t) + hf(x(t), u(t)). \tag{2.15}$$

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 h represent the time increment and write $x[k] = x(kh)$. Although modern simulation tools such as MATLAB and Mathematica use 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 (2.15), along with the analytical computation. We see that as
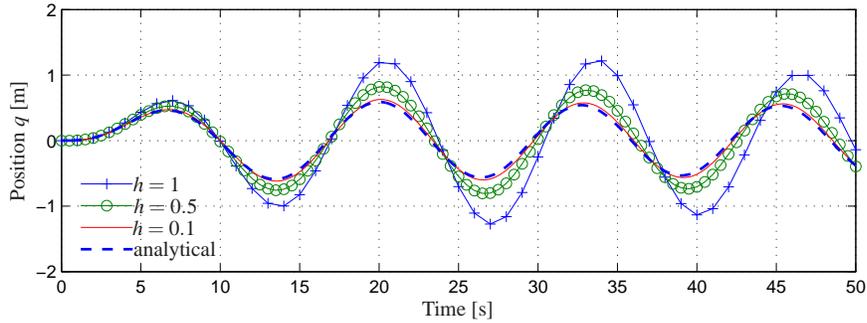
Figure 2.9: Simulation of the forced spring-mass system with different simulation time constants. The darker dashed line represents that analytical solution. The solid lines represent the approximate solution via the method of Euler integration, using decreasing step sizes.

h gets smaller, the computed solution converges to the exact solution. The form of the solution is also worth noticing: after an initial transient, the system settles into a periodic motion. The portion of the response after the transient is called the *steady state response* to the input.

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

Returning to the damped spring-mass system, the equations of motion with no input forcing are given by

$$\frac{dx}{dt} = \begin{pmatrix} x_2 \\ -\dfrac{c}{m}x_2 - \dfrac{k}{m}x_1 \end{pmatrix}, \tag{2.16}$$

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,

$$V(x) = \frac{1}{2}kx_1^2 + \frac{1}{2}mx_2^2. \tag{2.17}$$

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

$$\frac{dV}{dt} = kx_1\dot{x}_1 + mx_2\dot{x}_2 = kx_1x_2 + mx_2(-\frac{c}{m}x_2 - \frac{k}{m}x_1) = -cx_2^2,$$

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 later, 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 (2.16) to obtain:

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

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.

Another 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:

$$m\ddot{q} + c\dot{q} + kq = u. \tag{2.18}$$

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

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

We will see how to do this analytically in Chapter 6, 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 (2.18) with input $u(t)$, then applying an input $2u(t)$ will give a solution $2q(t)$ (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

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

where $g(\omega)$ is called the *gain* of the system and $\varphi(\omega)$ is called the *phase* (or 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.
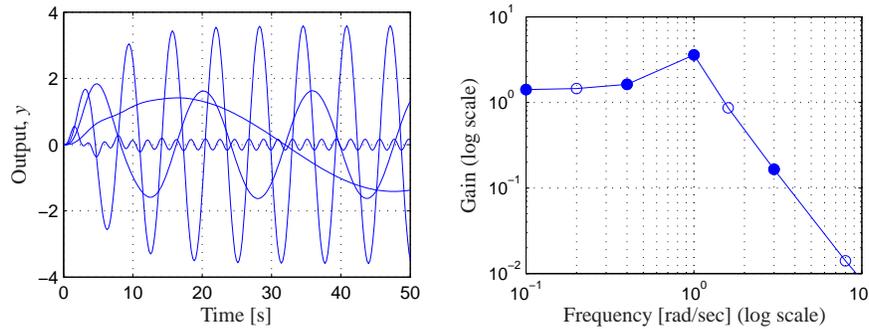
Figure 2.10: A frequency response (magnitude only) computed by measuring the response of individual sinusoids. The figure on the left shows the response of the system as a function of time to a number of different unit magnitude inputs (at different frequencies). The figure on the right shows this same data in a different way, with the magnitude of the response plotted as a function of the input frequency. The filled circles correspond to the particular frequencies shown in the time responses.

2.3 MODELING METHODOLOGY

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 individual components. Examples of such diagrams are shown in Figure 2.11. Schematic diagrams are useful because they give an overall picture of a system, showing different subprocesses and their interconnection, and indicating variables that can be manipulated and signals that can be measured.

Block Diagrams

A special graphical representation called a *block diagram* has been developed in control engineering. The purpose of a block diagram 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.

Figure 2.12 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 represented as a rectangle with the system name (or mathematical description) in the block. Two special cases are a proportional gain, which scales the
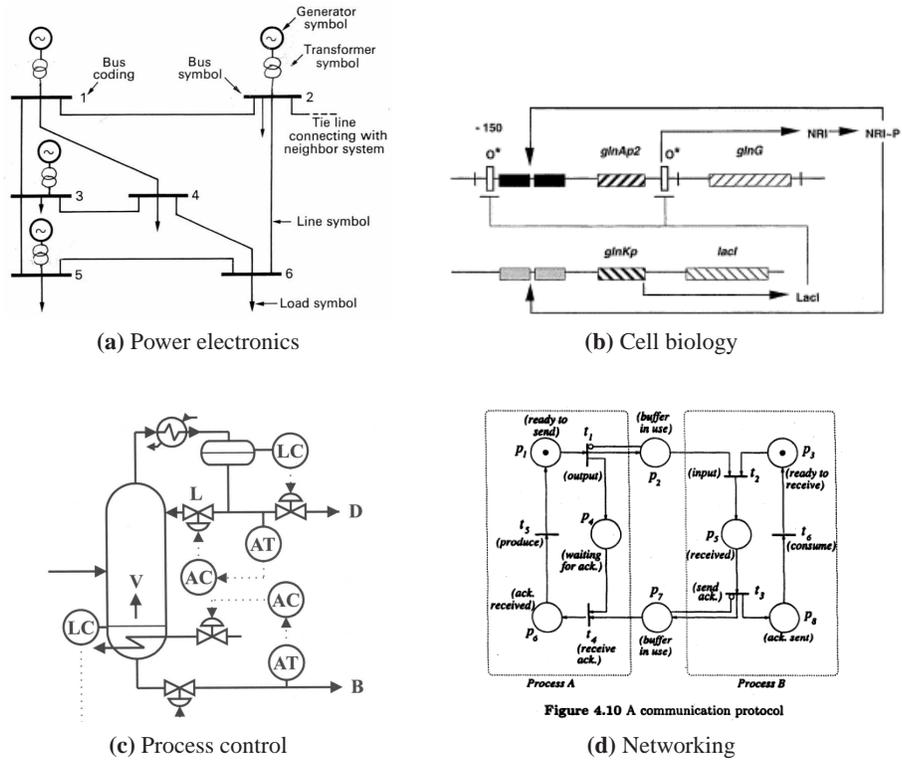
(a) Power electronics

(b) Cell biology

(c) Process control

(d) Networking

Figure 2.11: Schematic diagrams in different disciplines. Each diagram is used to illustrate the dynamics of a feedback system: (a) electrical schematics for a power system, (b) a biological circuit diagram for a synthetic clock circuit [ASMN03], (c) process diagram for a distillation column and (d) Petri net description of a communication protocol [**?**].
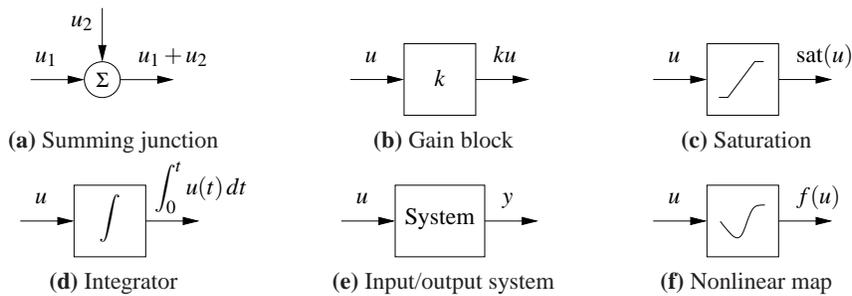

(a) Summing junction

(b) Gain block

(c) Saturation

(d) Integrator

(e) Input/output system

(f) Nonlinear map

Figure 2.12: Standard block diagram elements. The arrows indicate the the inputs and outputs of each element, with the mathematical operation corresponding to the blocked labeled at the output. The system block (e) represents the full input/output response of a dynamical system.
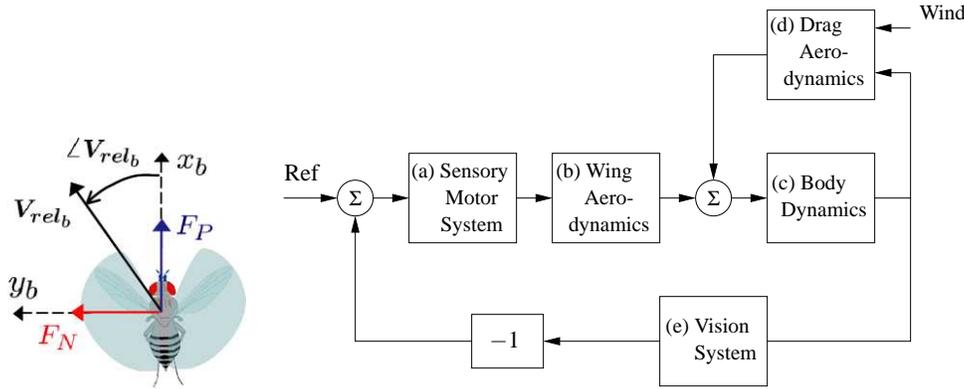
Figure 2.13: A block diagram representation of the flight control system for an insect flying against the wind. The mechanical portion of the model consists of the rigid body dynamics of the fly, the drag due to flying through the air and the forces generated by the wings. The motion of the body causes the visual environment of the fly to change, and this information is then used to control the motion of the wings (through the sensory motor system), closing the loop.

input by a multiplicative factor, and an integrator, which outputs the integral of the input signal.

Figure 2.13 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 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 2.13, 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 (e) and generates muscle commands that attempt to steer the fly so that the point of contraction is centered. These muscle commands are converted into forces through the flapping of the wings (b) 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 (c) 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 system blocks as inputs.

Each of the blocks in the diagram can itself be a complicated subsystem. For example, the fly visual system of a 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 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 give the step response of the system and 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 the change of the control signal we can also determine ranges where the process can be approximated by a linear system.

Example 2.5 Identification of a spring-mass system
Consider the spring-mass system from Section 2.1, whose dynamics are given by

$$m\ddot{q} + c\dot{q} + kq = u. \tag{2.19}$$

We wish to determine the constants m, c 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 $c^2 < 4km$, the step response for this system from the rest configuration is given by

$$q(t) = \frac{F_0}{k}\left(1 - \exp\left(-\frac{ct}{2m}\right)\sin(\omega_d t + \varphi)\right) \qquad \begin{aligned} \omega_d &= \frac{\sqrt{4km - c^2}}{2m} \\ \varphi &= \tan^{-1}\left(\sqrt{4km - c^2}\right). \end{aligned}$$

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.
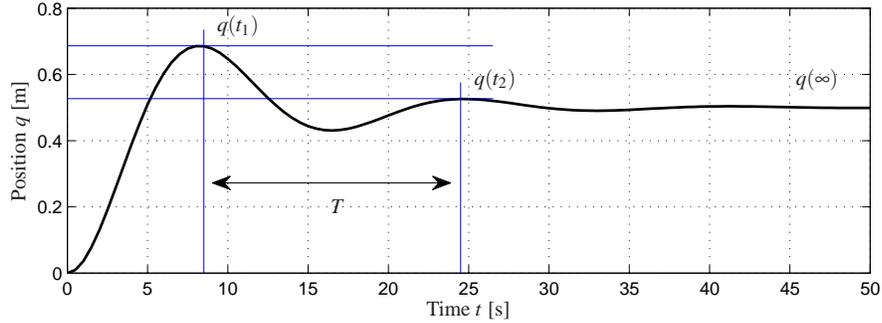
Figure 2.14: Step response for a spring-mass system. The magnitude of the step input is $F_0 = 20$ N. The period of oscillation, T is determined by looking at the time between two subsequent local maxima in the response. The period combined with the steady state value $q(\infty)$ and the relative decrease between local maxima can be used to estimate the parameters in a model of the system.

Figure 2.14 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:

$$q(\infty) = \frac{F_0}{k}, \tag{2.20}$$

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

$$\frac{2\pi}{T} = \frac{\sqrt{4km - c^2}}{2m}. \tag{2.21}$$

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 2.5)

$$\log\big(q(t_1) - F_0/k\big) - \log\big(q(t_2) - F_0/k\big) = \frac{c}{2m}(t_2 - t_1). \tag{2.22}$$

Using this set of three equations, we can solve for the parameters and determine that for the step response in Figure 2.14 we have $m \approx 250$ kg, $c \approx 60$ N s/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 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.

Normalization and Scaling

Having obtained a model, it is often useful to scale the variables by introducing dimension free variables. Such a procedure can often simplify the equations for a system by reducing the number of parameters and reveal interesting properties of the model. Scaling can also improve the numerical conditioning of the model to allow faster and more accurate simulations.

The procedure of scaling is straightforward: simply choose units for each independent variable and introduce new variables by dividing the variables with the chosen normalization unit. We illustrate the procedure with two examples.

Example 2.6 Spring-mass system
Consider again the spring-mass system introduced earlier. Neglecting the damping, the system is described by

$$m\ddot{q} + kq = u.$$

The model has two parameters m and k. To normalize the model we introduce dimension free variables $x = q/l$ and $\tau = \omega_0 t$, where $\omega_0 = \sqrt{k/m}$ and l is the chosen length scale. We scale force by $ml\omega_0^2$ and introduce $v = u/(ml\omega_0^2)$. The scaled equation then becomes

$$\frac{d^2 x}{d\tau^2} = \frac{d^2 q/l}{d(\omega_0 t)^2} = \frac{1}{\omega_0^2 lm}(-kq + u) = -x + v,$$

which is the normalized undamped spring-mass system. Notice that the normalized model has no parameters while the original model had two parameters m and k. Introducing the scaled, dimension-free state variables $z_1 = x = q/l$ and $z_2 = dx/d\tau = \dot{q}/(l\omega_0)$ the model can be written as

$$\frac{d}{dt} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} + \begin{pmatrix} 0 \\ v \end{pmatrix}.$$

This simple linear equation describes the dynamics of any spring-mass system, independent of the particular parameters, and hence gives us insight into the fundamental dynamics of this oscillatory system. To recover the physical frequency of oscillation or its magnitude, we must invert the scaling we have applied.   ∇

Example 2.7 Balance system
Consider the balance system described in Section 2.1. Neglecting damping by putting $c = 0$ and $\gamma = 0$ in equation (2.9) the model can be written as

$$(M + m)\frac{d^2 q}{dt^2} - ml\cos\theta\frac{d^2\theta}{dt^2} + ml\sin\theta\left(\frac{dq}{dt}\right)^2 = F$$

$$-ml\cos\theta\frac{d^2 q}{dt^2} + (J + ml^2)\frac{d^2\theta}{dt^2} - mgl\sin\theta = 0$$

Let $\omega_0 = \sqrt{mgl/(J + ml^2)}$, choose the length scale as l, the time scale as $1/\omega_0$, the force scale as $(M + m)l\omega_0^2$ and introduce the scaled variables $\tau = \omega_0 t$, $x = q/l$
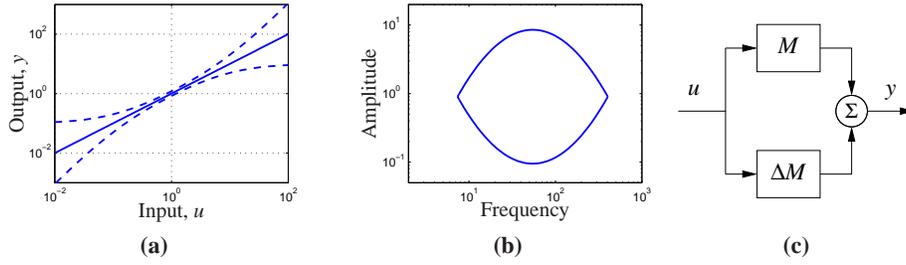
Figure 2.15: Characterization of model uncertainty. Uncertainty of a static system is illustrated in (a). The uncertainty lemon in (b) is one way to capture uncertainty in dynamical systems emphasizing that a model is only valid in some amplitude and frequency ranges. In (c) a model is represented by a nominal model (M) and another model ΔM representing the uncertainty analogous to representation of parameter uncertainty.

and $u = F/((M+m)l\omega_0^2)$. The equations then become

$$\frac{d^2x}{d\tau^2} - \alpha\cos\theta\frac{d^2\theta}{d\tau^2} + \alpha\left(\frac{d\theta}{d\tau}\right)^2 = u$$

$$-\beta\cos\theta\frac{d^2x}{d\tau^2} + \frac{d^2\theta}{d\tau^2} - \sin\theta = 0,$$

where $\alpha = m/(M+m)$ and $\beta = ml^2/(J+ml^2)$. Notice that the original model has five parameters m, M, J, l and g but the normalized model has only two parameters α and β. If $M \gg m$ and $ml^2 \gg J$ we get $\alpha \approx 0$ and $\beta \approx 1$ and the model can be approximated by

$$\frac{d^2x}{d\tau^2} = u, \quad \frac{d^2\theta}{d\tau^2} - \sin\theta = u\cos\theta.$$

The model can be interpreted as a mass combined with an inverted pendulum driven by the same input. ∇

Model Uncertainty

Reducing uncertainty is one of the main reasons for using feedback and it is therefore important to characterize uncertainty. When making measurements there is a good tradition to assign both a nominal value and a measure of uncertainty. It is useful to apply same principle to modeling, but unfortunately it is often difficult to express the uncertainty of a model quantitatively.

For a static system whose input-output relation can be characterized by a function, uncertainty can be expressed by an uncertainty band as illustrated in In Figure 2.15a. At low signal levels there are uncertainties due to sensor resolution, friction and quantization. Some models for queuing systems or cells are based on averages that exhibit significant variations for small populations. At large signal levels there are saturations or even system failures. The signal ranges where a model is reasonably accurate varies dramatically between applications but it is rare to find models that are accurate for signal ranges larger than 10^4.

Characterization of uncertainty of dynamic model is much more difficult. We can try to capture uncertainties by assigning uncertainties to parameters of the model but this is often not sufficient. There may be errors due to phenomena that have been neglected, for example small time delays. In control the ultimate test is how well a control system based on the model performs and time delays can be important. There is also a frequency aspect. There are slow phenomena, such as aging, that can cause changes or drift in the systems. There are also high frequency effects: a resistor will no longer be a pure resistance at very high frequencies and a beam has stiffness and will exhibit additional dynamics when subject to high frequency excitation. The *uncertainty lemon* shown in Figure 2.15b is one way to conceptualize the uncertainty of a system. It illustrates that a model is only valid in certain amplitude and frequency ranges.

We will introduce some formal tools for representing uncertainty in Chapter 12 using figures such as the one shown in Figure 2.15c. These tools make use of the concept of a transfer function, which describes the frequency response of an input/output system. For now, we simply note that one should always be careful to recognize the limits of a model and not to make use of models outside their range of applicability. For example, one can describe the uncertainty lemon and then check to make sure that signals remain in this region.

2.4 MODELING EXAMPLES

In this section we introduce 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 applications that serve as running examples throughout the text are given in the next chapter.

Motion Control Systems

Motion control systems involve the use of computation and feedback to control the movement of a mechanical system. Motion control systems range from nanopositioning 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 brakes, suspension control, traction control), to air and space flight control systems (airplanes, satellites, rockets and planetary rovers).

Example 2.8 Vehicle steering—the bicycle model
A common problem in motion control is to control the trajectory of a vehicle through an actuator that causes a change in the orientation. A steering wheel on an automobile or the front wheel of a bicycle are two examples, but similar dynamics occur in steering of ships or control of the pitch dynamics of an aircraft. In many
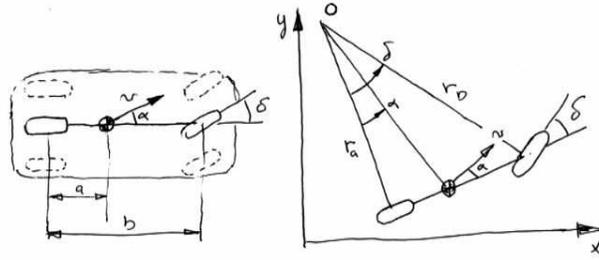
Figure 2.16: Vehicle steering dynamics. The left figure shows an overhead view of a vehicle with four wheels. By approximating the motion of the front and rear pairs of wheels by a single front and rear wheel, we obtain an abstraction called the "bicycle model", shown on the right. The wheel base is b and the center of mass at a distance a forward of the rear wheels. The steering angle is δ and the velocity at the center of mass has the angle α relative the length axis of the vehicle. The position of the vehicle is given by (x,y) and the orientation (heading) by θ.

cases, we can understand the basic behavior of these systems through the use of a simple model that captures the basic geometry of the system.

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 steering angle δ. To be specific, consider the velocity v at the center of mass, a distance a from the rear wheel, and let b be the wheel base, as shown in Figure 2.16. Let x and y be the coordinates of the center of mass, θ the heading angle and α the angle between the velocity vector v and the centerline of the vehicle. Since $b = r_a \tan\delta$ and $a = r_a \tan\alpha$ it follows that $\tan\alpha = (a/b)\tan\beta$ and we get the following relation between α and the steering angle δ:

$$\alpha(\delta) = \arctan\left(\frac{a\tan\delta}{b}\right). \tag{2.23}$$

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

$$\begin{aligned}\frac{dx}{dt} &= v\cos(\alpha+\theta) = v_0\frac{\cos(\alpha+\theta)}{\cos\alpha}\\ \frac{dy}{dt} &= v\sin(\alpha+\theta) = v_0\frac{\sin(\alpha+\theta)}{\cos\alpha}.\end{aligned} \tag{2.24}$$

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

$$\frac{d\theta}{dt} = \frac{v_0}{r_a} = \frac{v_0}{b}\tan\delta. \tag{2.25}$$

Equations (2.23)–(2.25) can be used to model an automobile under the assumptions that there is no slip between the wheels and the road and that the two front
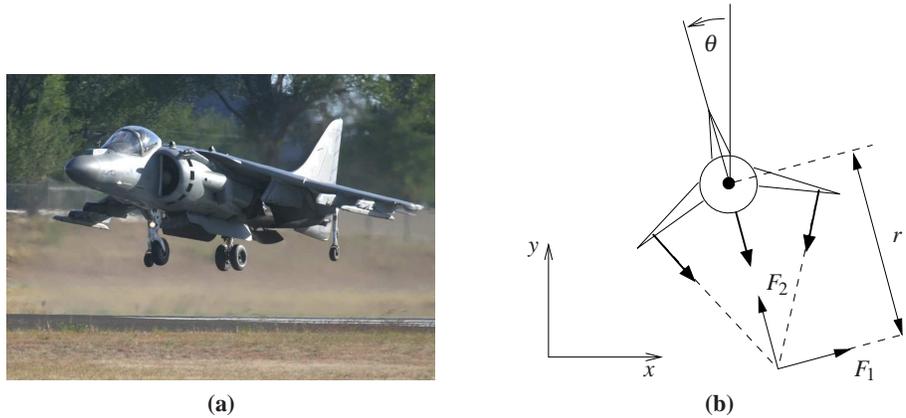
Figure 2.17: Vectored thrust aircraft. The Harrier AV-8B military aircraft (a) redirects its engine thrust downward so that it can "hover" above the ground. Some air from the engine is diverted to the wing tips to be used for maneuvering. As shown in (b), the net thrust on the aircraft can be decomposed into a horizontal force F_1 and a vertical force F_2 acting at a distance r from the center of mass.

wheels can be a approximated by a single wheel at the center of the car. The assumption of no slip can be relaxed by adding an extra state variable, giving a more realistic model. Such a model also describes the steering dynamics of ships as well as the pitch dynamics of aircraft and missiles. It is also possible to place the coordinates of the car at the rear wheels (corresponding to setting $\alpha = 0$), a model which is often referred to as the *Dubins car* [Dub57].

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, which is equivalent to a vehicle with rear-wheel steering.

$$\nabla$$

Example 2.9 Vectored thrust aircraft
Consider the motion of vectored thrust aircraft, such as the Harrier "jump jet" shown Figure 2.17a. The 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_2 acting at a distance r below the aircraft (determined by the geometry of the thrusters).

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
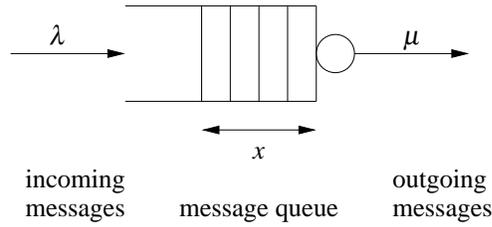
Figure 2.18: Schematic diagram of a queuing system. Messages arrive at rate λ and are stored in a queue. Messages are processed and removed from the queue at rate μ. The average size of the queue is given by $x \in \mathbb{R}$.

vehicle are given by

$$
\begin{aligned}
m\ddot{x} &= F_1 \cos\theta - F_2 \sin\theta - c\dot{x} \\
m\ddot{y} &= F_1 \sin\theta + F_2 \cos\theta - mg - c\dot{y} \\
J\ddot{\theta} &= rF_1.
\end{aligned}
\tag{2.26}
$$

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 $u_2 = F_2 - mg$, the equations become

$$
\begin{aligned}
m\ddot{x} &= -mg\sin\theta - c\dot{x} + u_1 \cos\theta - u_2 \sin\theta \\
m\ddot{y} &= mg(\cos\theta - 1) - c\dot{y} + u_1 \sin\theta + u_2 \cos\theta \\
J\ddot{\theta} &= ru_1.
\end{aligned}
\tag{2.27}
$$

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

Information Systems

Information systems range from communication systems like the Internet to software systems that manipulate data or manage enterprise wide resources. Feedback is present in all these systems, and design of strategies for routing, flow control and buffer management are typical problems. Many results in queuing theory emerged from design of telecommunication systems and later from development of the Internet and computer communication systems [BG87, Kle75, Sch87]. Management of queues to avoid congestion is a central problem and we will therefore start by discussing modeling of queuing systems.

Example 2.10 Queuing systems
A schematic picture of a simple queue is shown in Figure 2.18. Requests arrive and are then queued and processed. There can be large variations in arrival rates and service rates and the queue length builds up when the arrival rate is larger than the service rate. When the queue becomes too large, service is denied using an admission control policy.

   The system can be modeled in many different ways. One way is to model each incoming request, which leads to an event-based model where the state is

an integer that represents the queue length. The queue changes when a request arrives or a request is serviced. The statistics of arrival and servicing are typically modeled as random processes. In many cases it is possible to determine statistics of quantities like queue length and service time but the computations can be quite complicated.

A significant simplification can be obtained by using a *flow model*. Instead of keeping track of each request we instead view service and requests as flows, similar to what is done when replacing molecules by a continuum when analyzing fluids. Assuming that the average queue length x is a continuous variable and that arrivals and services are flows with rates λ and μ, the system can be modeled by the first order differential equation

$$\frac{dx}{dt} = \lambda - \mu = \lambda - \mu_{\max} f(x), \quad x \geq 0, \tag{2.28}$$

where $\mu_{\max}$ is the maximum service rate and $f(x)$ is a number between 0 and 1 that describes the effective service rate as a function of the queue length.

It is natural to assume that the effective service rate depends on the queue length because larger queues require more resources. In steady state we have $f(x) = \lambda/\mu_{\max}$ and we assume that the queue length goes to zero when $\lambda/\mu_{\max}$ goes to zero and that it goes to infinity when $\lambda/\mu_{\max}$ goes to 1. This implies that $f(0) = 0$ and that $f(\infty) = 1$. In addition if we assume that the effective service rate deteriorates monotonically with queue length then the function $f(x)$ is monotone and concave. A simple function that satisfies the basic requirements is $f(x) = x/(1+x)$, which gives the model

$$\frac{dx}{dt} = \lambda - \mu_{\max} \frac{x}{x+1}. \tag{2.29}$$

This model was proposed by Agnew [Agn76]. It can be shown that if arrival and service processes are Poisson processes the average queue length is given by equation (2.29) and that equation (2.29) is a good approximation even for short queue lengths; see Tipper [TS90].

To explore the properties of the model (2.29) we will first investigate the equilibrium value of the queue length when the arrival rate λ is constant. Setting the derivative dx/dt to zero in equation (2.29) and solving for x we find that the queue length x approaches the steady state value

$$x_e = \frac{\lambda}{\mu_{\max} - \lambda}. \tag{2.30}$$

Figure 2.19a shows the steady state queue length as a function of $\lambda/\mu_{\max}$, the effective service rate excess. Notice that the queue length increases rapidly as λ approaches $\mu_{\max}$. To have a queue length less than 20 requires $\lambda/\mu_{\max} < 0.95$. The average time to service a request is $T_s = (x+1)/\mu_{\max}$ and it also increases dramatically as λ approaches $\mu_{\max}$.

Figure 2.19b illustrates the behavior of the server in a typical overload situation. The maximum service rate is $\mu_{\max} = 1$, and the arrival rate starts at $\lambda = 0.5$. The
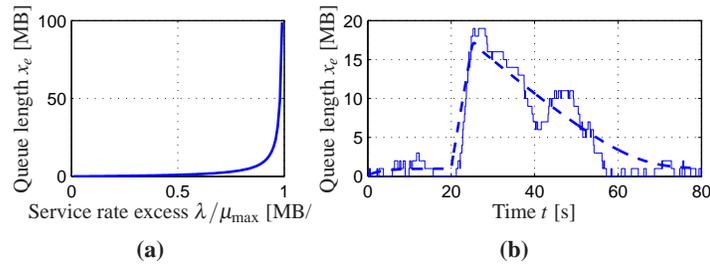
Figure 2.19: Queuing dynamics. The figure on the left shows steady state queue length as a function of $\lambda/\mu_{\max}$, and the figure on the right shows the behavior of the queue length when there is a temporary overload in the system. The full line shows a realization of an event based simulation and the dashed line shows the behavior of the flow model (2.29).

arrival rate is increased to $\lambda = 4$ at time 20, and it returns to $\lambda = 0.5$ at time 25. The figure shows that the queue builds up quickly and clears very slowly. Since the response time is proportional to queue length, it means that the quality of service is poor for a long period after an overload. This behavior is called the *rush-hour effect* and has been observed in web servers and many other queuing systems such as automobile traffic.

The dashed line in Figure 2.19b shows the behavior of the flow model, which describes the average queue length. The simple model captures behavior qualitatively, but there are significant variations from sample to sample when the queue length is short.

$$\nabla$$

Queuing problems of the type illustrated in Example 2.10 have been observed in many different situations. The following example illustrates an early example of the difficulty and it also describes how it can be avoided by using a simple feedback scheme.

Example 2.11 Virtual memory paging control
An early example of use of feedback in computer systems was applied in operating system OS/VS for the IBM 370 [BG68, Cro75]. The system used virtual memory, which allows programs to address more memory than is physically available as fast memory. Data in current fast memory (RAM) is accessed directly but data that resides in slower memory (disk) is automatically loaded into fast memory. The system is implemented in such a way that it appears to the programmer as a single large section of memory. The system performed very well in many situations but very long execution times were encountered in overload situations, as shown in Figure 2.20a. The difficulty was resolved with a simple discrete feedback system. The load of the central processing unit (CPU) was measured together with the number of page swaps between fast memory and slow memory. The operating region was classified as being in one of three states: normal, underload or overload. The normal state is characterized by high CPU activity, the underload state is characterized by low CPU activity and few page replacements, the
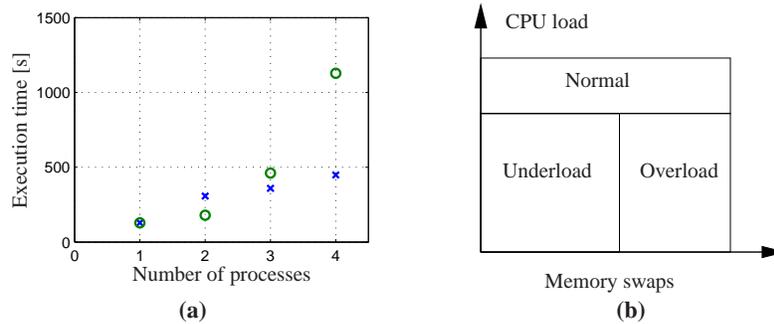
Figure 2.20: Illustration of feedback in the virtual memory system of IBM/370. The left figure (a) shows the effect of feedback on execution times in a simulation, following [BG68]. Results with no feedback are shown with ○ and with feedback with ×. Notice the dramatic decrease in execution time for the system with feedback. The right figure (b) illustrates how the three states were obtained based on process measurements.

overload state has moderate to low CPU load but many page replacements, see Figure 2.20a. The boundaries between the regions and the time for measuring the load were determined from simulations using typical loads. The control strategy was to do nothing in the normal load condition, to exclude a process from memory in an overload condition and to allow a new process or a previously excluded process in the underload condition. Figure 2.20a shows the effectiveness of the simple feedback system in simulated loads. Similar principles are used in many other situations, for example in fast, on-chip cache memory.   ∇

Example 2.12 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, 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 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 two groups of agents are completely isolated from each other). Figure 2.21a shows a simple situation of this type.

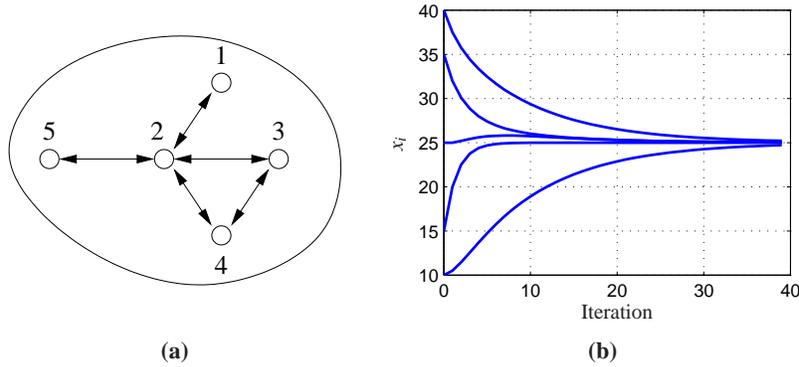We model the connectivity of the sensor network using a graph, with nodes

Figure 2.21: Consensus protocols for sensor networks. A simple sensor network with five nodes is shown on the left. In this network, node 1 communicates with node 2, node 2 communicates with notes 1, 3, 4 and 5, etc. A simulation demonstrating the convergence of the consensus protocol (2.31) to the average value of the initial conditions is shown on the right.

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 2.21a, 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

$$x_i[k+1] = x_i[k] + \gamma \sum_{j \in \mathcal{N}_i} (x_j[k] - x_i[k]). \tag{2.31}$$

This protocol attempts to compute the average by updating the local state of each agent based on the value of its neighbors. 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 $x_e[k+1] = x_e[k]$. It is easy to show that $x_e = (\alpha, \alpha, \dots, \alpha)$ is an equilibrium 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

$$W[k] = \frac{1}{N} \sum_{i=1}^{n} x_i[k]$$

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 quantity we are trying to compute. $W[k]$ is given by the difference equation

$$W[k+1] = \frac{1}{N} \sum_{i=1}^{n} x_i[k+1] = \frac{1}{N} \sum_{i=1}^{n} \left(x_i[k] + \gamma \sum_{j \in \mathcal{N}_i} (x_j[k] - x_i[k]) \right).$$

Since $i \in \mathcal{N}_j$ implies that $j \in \mathcal{N}_i$, it follows that each term in the second summation 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 connected 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 2.21b.

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 or computing higher order statistical moments of a distributed quantity [OSFM07]. $\qquad \nabla$

Biological Systems

Biological systems 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.
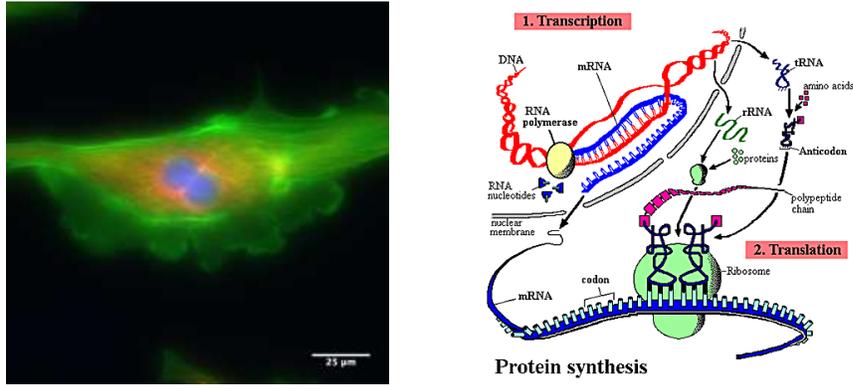
Figure 2.22: Biological circuitry. The cell on the left is a bovine pulmonary cell, stained so that the the nucleus, actin and chromatin are visible. The figure on the right gives an overview of the process by which proteins in the cell are made. RNA is transcribed from DNA by an RNA polymerase enzyme. The RNA is then translated into a protein by an organelle called the ribosome.

Example 2.13 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 an mRNA transcript from DNA. The mRNA is then translated into a protein according to its nucleotide sequence. This process is illustrated in Figure 2.22.

A simple model of the transcriptional regulation process is through 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

$$\frac{dm_A}{dt} = -\gamma m_A + \frac{\alpha}{1 + k_B p_B^n} + \alpha_0, \qquad \frac{dp_A}{dt} = \beta m_A - \delta p_A, \qquad (2.33)$$

where $\alpha + \alpha_0$ is the unregulated transcription rate, γ represents the rate of degradation of mRNA, α and n are parameters that describe how B represses A, β represents the rate of production of the protein from its corresponding mRNA and δ represents the rate of degradation of the protein A. The parameter α_0 describes the "leakiness" of the promoter and n is called the Hill coefficient and relates to the cooperativity of the promoter.

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

$$\frac{dm_A}{dt} = \frac{\alpha k_B p_B^n}{1 + k_B p_B^n} + \alpha_0 - \gamma m_A, \qquad \frac{dp_A}{dt} = \beta m_A - \delta p_A, \qquad (2.34)$$
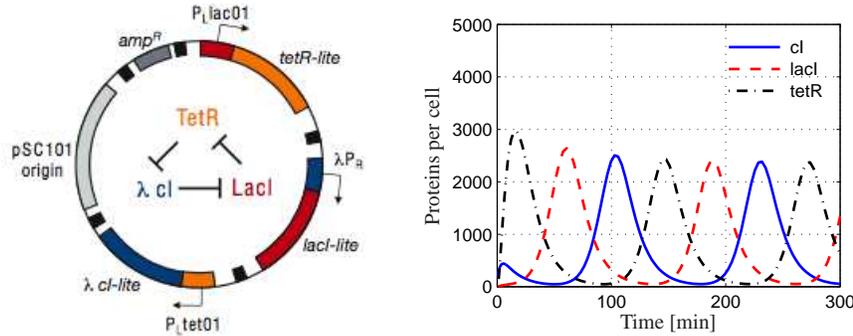
Figure 2.23: The repressilator genetic regulatory network. A schematic diagram of the re-pressilator is given on the left, showing the layout of the genes in the plasmid that holds the circuit as well as the circuit diagram (center). A simulation of a simple model for the repressilator is shown on the right, showing the oscillation of the individual protein concentrations. Parameter values taken from [EL00].

where the variables are the same as described previously. 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 first 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 repress 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 absent, then LacI is produced (at the unregulated 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 $n = 2$, $\alpha = 0.5$, $k = 6.25 \times 10^{-4}$, $\alpha_0 = 5 \times 10^{-4}$, $\gamma = 5.8 \times 10^{-3}$, $\beta = 0.12$ and $\delta = 1.2 \times 10^{-3}$ with initial conditions $x(0) = (1, 0, 0, 200, 0, 0)$ (following [EL00]). ∇

Example 2.14 Wave propagation in neuronal networks
The dynamics of the membrane potential in a cell are a fundamental mechanism in understanding signaling in cells, particularly in neurons and muscle cells. The Hodgkin-Huxley equations give a simple model for studying propagation waves in networks of neurons. The model for a single neuron has the form

$$C\frac{dV}{dt} = -I_{\text{Na}} - I_{\text{K}} - I_{\text{leak}} + I_{\text{input}},$$

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} a leakage current and I_{input} the external stimulation of the cell. Each current obeys Ohm's 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

$$E = \frac{RT}{nF} \log(c_e/c_i),$$

where R is Boltzmann's constant, T the absolute temperature, F Faraday's constant, n is the charge (or valence) of the ion, and c_i and c_e are the ion concentrations inside the cell and in the external fluid. At 20 °C 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 [HH52]. 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. The voltage clamp described in Section 1.3 (see Figure 1.8) was a key element in Hodgkin and Huxley's experiments.   ∇

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 by Fourier when he modeled heat conduction in solids [Fou07]. Models of dynamics have been developed in many different fields, including mechanics [Arn78, Gol53], heat conduction [CJ59], fluids [BRS60], vehicles [Abk69, Bla91, Ell94], circuit theory [Gui63], acoustics [Ber54] and micromechanical systems [Sen01]. Control theory requires modeling from many different domains and most control theory texts 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 [Can03]. The book by Aris [Ari94] is highly original and has a detailed discussion of the use of dimension free variables. Two of the authors' favorite books on modeling of biological systems are J. D. Murray [Mur04] and Wilson [Wil99]. For readers interested in learning more about object oriented modeling and Modelica, Tiller [Til01] provides an excellent introduction.

EXERCISES

2.1 Consider the linear ordinary differential equation (2.7). Show that by choosing a state space representation with $x_1 = y$, the dynamics can be written as

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

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

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

2.2 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 the dynamics are approximated by equation (2.10).

2.3 Consider the following discrete time system

$$x[k+1] = Ax[k] + Bu[k]$$
$$y[k] = Cx[k]$$

where

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad A = \begin{pmatrix} a_{11} & a_{12} \\ 0 & a_{22} \end{pmatrix} \qquad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad C = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

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) For the case when $a_{12} = 0$ and $u = 0$, give a closed for expression for the output of the system.

(b) A discrete system is in *equilibrium* when $x[k+1] = x[k]$ for all k. Let $u = r$ be a constant input and compute the resulting equilibrium point for the system. Show that if $|a_{ii}| < 1$ for all i, all initial conditions give solutions that converge to the equilibrium point.

(c) Write a computer 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 $x[0] = 0$ and A given by

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

2.4 Keynes' simple model for an economy is given by

$$Y[k] = C[k] + I[k] + G[k],$$

where Y, C, I and G are gross national product (GNP), consumption, investment and government expenditure for year k. Consumption and investment are modeled

by difference equations of the form

$$C[k+1] = aY[k], \qquad I[k+1] = b(C[k+1] - C[k]),$$

where a and b are parameters. The first equation implies that consumption increases with GNP but that the effect is delayed. The second equation implies that investment is proportional to the rate of change of consumption.

Show that the equilibrium value of the GNP is given by

$$Y_e = \frac{1}{1-a}(I_e + G_e),$$

where the parameter $1/(1-a)$ is the Keynes multiplier (the gain from I or G to Y). With $a = 0.25$ an increase of government expenditure will result in a fourfold increase of GNP. Also show that the model can be written as the following discrete time state model

$$\begin{pmatrix} C[k+1] \\ I[k+1] \end{pmatrix} = \begin{pmatrix} a & a \\ ab-a & ab \end{pmatrix} \begin{pmatrix} C[k] \\ I[k] \end{pmatrix} + \begin{pmatrix} a \\ ab \end{pmatrix} G[k]$$

$$Y[k] = C[k] + I[k] + G[k].$$

2.5 (Second order system identification) Verify that equation (2.22) in Example 2.5 is correct and use this formula and the others in the example to compute the parameters corresponding to the step response in Figure 2.14.

2.6 (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.

2.7 (Normalized oscillator dynamics) Consider a damped spring-mass system with dynamics

$$m\ddot{q} + c\dot{q} + kq = u.$$

Let $\omega_0 = \sqrt{k/m}$ be the undamped natural frequency and $\zeta = c/(2\sqrt{km})$ be the relative damping.

(a) Show that by rescaling the equations, we can write the system dynamics in the form

$$\ddot{q} + 2\zeta\omega_0\dot{z} + \omega_0^2 q = w \qquad\qquad (2.35)$$

where $u = F/m$. This form of the dynamics is that of a linear oscillator with natural frequency ω_0 and damping coefficient ζ.

(b) Show that the system can be further normalized and written in the form

$$\frac{dz_1}{dt} = z_2, \qquad \frac{dz_2}{dt} = -z_1 - 2\zeta z_2 + v.$$

We thus see that the essential dynamics of the system are governed by a single damping parameter, ζ.

2.8 An electric generator connected to a strong power grid can be modeled by a momentum balance for the rotor of the generator:

$$J\frac{d^2\varphi}{dt^2} = P_m - P_e = P_m - \frac{EV}{X}\sin\varphi,$$

where J is the effective moment of inertia of the generator, φ the angle of rotation, P_m the mechanical power that drives the generator, E is the generator voltage, V the grid voltage and X the reactance of the line. P_e is the active electrical power and, assuming that the line dynamics are much faster than the rotor dynamics, it is given by $P_e = VI = (EV/X)\sin\varphi$, where I is the current component in phase with the voltage E and φ is the phase angle between voltages E and V.

Show that the dynamics of the electric generate have the same normalized form as the inverted pendulum (note that damping has been neglected in the model above).

2.9 Show that the dynamics for a balance system using normalized coordinates can be written in state space form as

$$\frac{dx}{dt} = \begin{pmatrix} x_3 \\ x_4 \\ \dfrac{-\alpha x_4^2 - \alpha \sin x_2 \cos x_2 + u}{1 - \alpha\beta \cos^2 x_2} \\ \dfrac{-\alpha\beta \cos x_2 x_4^2 - \sin x_2 + \beta \cos x_2 u}{1 - \alpha\beta \cos^2 x_2} \end{pmatrix},$$

where $x = (q/l, \theta, \dot{q}/l, \dot{\theta})$.

2.10 Consider the dynamics of two repressors connected together in a cycle, as shown below:
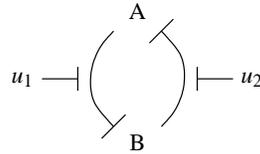
Using the models from Example 2.13, under the assumption that the parameters are the same for both genes, and further assuming that the mRNA concentrations

reach steady state quickly, show that the dynamics for this system can be written as

$$\frac{dz_1}{d\tau} = \frac{\mu}{1+z_2^n} - z_1 - v_1, \qquad \frac{dz_2}{d\tau} = \frac{\mu}{1+z_1^n} - z_2 - v_2. \qquad (2.36)$$

where z_1 and z_2 represent scaled versions of the protein concentrations and the time scale has been changed. Show that $\mu \approx 2.16$ using the parameters in Example 2.13.