Chapter 2
Differential Equations & Domains

Synopsis The primary goal of this chapter is to obtain a solid working intuition for the continuous dynamics part of cyber-physical systems. It provides a brief introduction to differential equations with evolution domains as models of continuous physical processes. While focusing on an intuitive development, this chapter lays the foundation for an operational understanding of continuous processes. For reference, it discusses some of the elementary theory of differential equations. This chapter also introduces the first-order logic of real arithmetic as a language for describing the evolution domains to which continuous processes are restricted when forming hybrid systems.

2.1 Introduction

Cyber-physical systems combine cyber capabilities with physical capabilities. You already have experience with models of computation and algorithms for the cyber part of CPS if you have seen the use of programming languages for computer programming. In CPS, we do not program computers, though, but rather program CPSs instead. Hence, we program computers that interact with physics to achieve their goals. In this chapter, we study models of physics and the most elementary part of how they can interact with the cyber part. Physics by and large is obviously a deep subject. But for CPS, one of the most fundamental models of physics is sufficient at first, that of ordinary differential equations.

While this chapter covers the most important parts of differential equations, it is not to be misunderstood as a diligent coverage of the fascinating area of ordinary differential equations. What you need to get started with the book is an intuition about differential equations, as well as an understanding of their precise meaning. This will be developed in the present chapter. Subsequently, we will return to the topic of differential equations for a deeper understanding of differential equations and their proof principles a number of times in later chapters, especially Part II. The other important aspect that this chapter develops is *first-order logic of real arith-*

A. Platzer, *Logical Foundations of Cyber-Physical Systems*,
https://doi.org/10.1007/978-3-319-63588-0_2

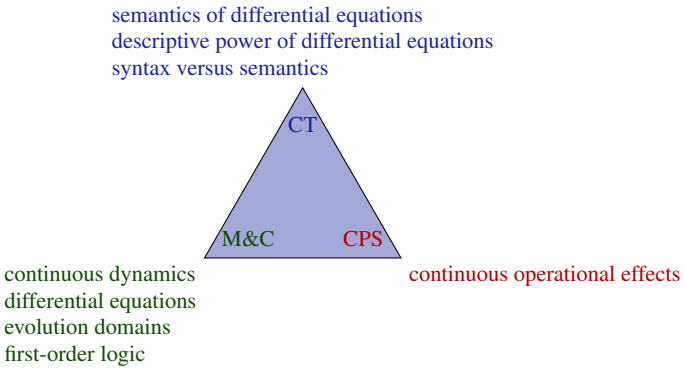
metic for the purpose of representing domains and domain constraints of differential equations, which is of paramount significance in hybrid systems. More detailed treatments of differential equations can be found, e.g., in the seminal book by Walter [10] or elsewhere [2, 4, 8, 9].

The most important learning goals of this chapter are:

Modeling and Control: We develop an understanding of one core principle behind CPS: the case of continuous dynamics and differential equations with evolution domains as models of the physics part of CPS. We introduce first-order logic of real arithmetic as the modeling language for describing evolution domains of differential equations.

Computational Thinking: Both the significance of meaning and the descriptive power of differential equations will play key roles, foreshadowing many important aspects underlying the proper understanding of cyber-physical systems. We will also begin to learn to carefully distinguish between syntax (which is notation) and semantics (what carries meaning), a core principle for both logic and computer science that continues to be crucial for CPS.

CPS Skills: We develop an intuition for the continuous operational effects of CPS and devote significant attention to understanding the exact semantics of differential equations, which has some subtleties in store for us.



2.2 Differential Equations as Models of Continuous Physical Processes

Differential equations model processes in which the state variables of a system evolve continuously in time. A differential equation describes quite concisely how the system evolves over time. It describes how the variables change locally, so it, basically, indicates the direction in which the variables evolve at each point in space.

Fig. 2.1 shows the respective directions in which the system evolves by a vector at each point and illustrates one solution as a curve in two-dimensional space that follows those vectors everywhere. Of course, the figure would be rather cluttered if we were to try to indicate the vector at literally each and every point, of which there are uncountably infinitely many. But this is a shortcoming only of our illustration, not of the mathematical realities. Differential equations actually define such a vector for the direction of evolution at *every* point in space.

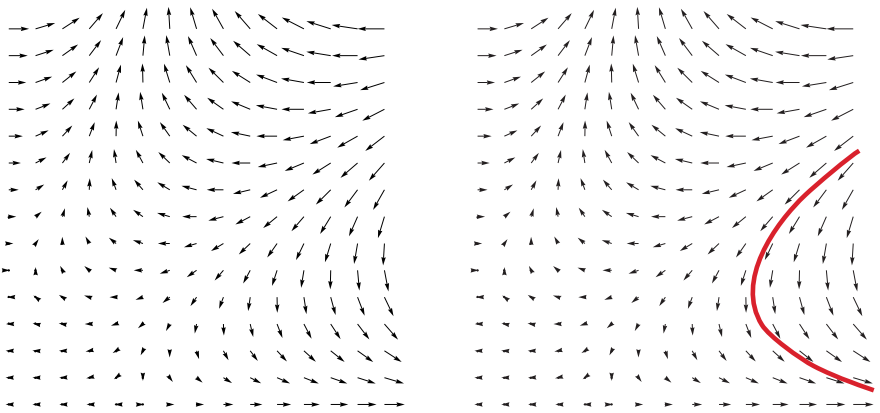


Fig. 2.1 Vector field **(left)** and vector field with one solution of a differential equation **(right)**

As an example, suppose we have a car whose position is denoted by x. Cars have a tendency to move, so the car's position x will change over time. How the value of variable x changes over time depends on how fast the car is driving. Let v denote the velocity of said car. Since v is the velocity of the car, its position x changes according to the velocity. So, the position x changes such that its derivative x' is v, which we denote by the differential equation $x' = v$. This differential equation means that the time-derivative of the position x equals the velocity v. So how x evolves depends on v. If the velocity is $v = 0$, then the position x does not change at all and the car might be parked or in a traffic jam. If $v > 0$, then the position x keeps on increasing over time. How fast x increases depends on the value of v, a bigger v give a quicker changes in x, because the time-derivative of x equals v in the differential equation $x' = v$.

Of course, the velocity v itself may also be subject to change over time. The car might accelerate, so let a denote its acceleration. Then the velocity v changes with time-derivative a, that is by the differential equation $v' = a$. Overall, the car then follows the differential equation (really a differential equation system):[1]

$$x' = v, v' = a$$

[1] Note that the value of x changes over time, so it is really a function of time. Hence, the notation $x'(t) = v(t), v'(t) = a$ is sometimes used. It is customary, however, to eloquently suppress the argument t for time and just write $x' = v, v' = a$ instead. In the physics literature, the notation $\dot{x}$ is often

That is, the position x of the car changes with time-derivative v, which, in turn, changes with time-derivative a.

What we mean by this differential equation, intuitively, is that the system always follows a direction (or vector field shown in Fig. 2.2) where, at all points (x, v), the direction vectors have their direction for positions point in a direction that equals the current v while their direction for velocities points in the same direction a. The system is always supposed to follow exactly in the direction of those direction vectors at every point. What does this mean exactly? How can we understand it doing that at all of the infinitely many points?

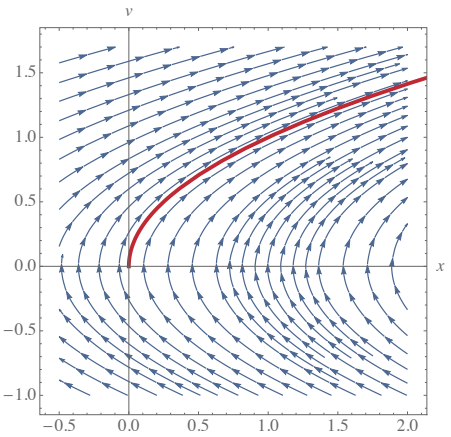


Fig. 2.2 Vector field with one solution of accelerated straight-line motion

To sharpen our intuition for this aspect, consider a one-dimensional differential equation with a position x that changes over time t starting at initial state 1 at initial time 0:

$$\begin{pmatrix} x'(t) = \frac{1}{4}x(t) \\ x(0) = 1 \end{pmatrix}$$

For a number of different time discretization steps $\Delta \in \{4, 2, 1, \frac{1}{2}\}$, Fig. 2.3 illustrates what an approximate pseudo-solution would look like that only respects the differential equation at the times that are integer multiples of Δ and is in blissful ignorance of the differential equation in between these grid points. Such a pseudo-solution corresponds to what is obtained by explicit Euler integration [3]. The true solution of the differential equation should, however, also respect the direction that the differential equation prescribes at all the other uncountably infinitely many time points in between. Because this differential equation is quite well behaved, the discretizations still approach the true continuous solution $x(t) = \mathrm{e}^{\frac{t}{4}}$ as Δ gets smaller. But differential equations come with a lot of surprises when anyone attempts to understand them from a discrete perspective. No matter how small a discretization

used when referring to the time-derivative of x. We prefer the mathematical notation x', because dots are more easily overlooked, especially on longer names, and are hard to typeset in ASCII.

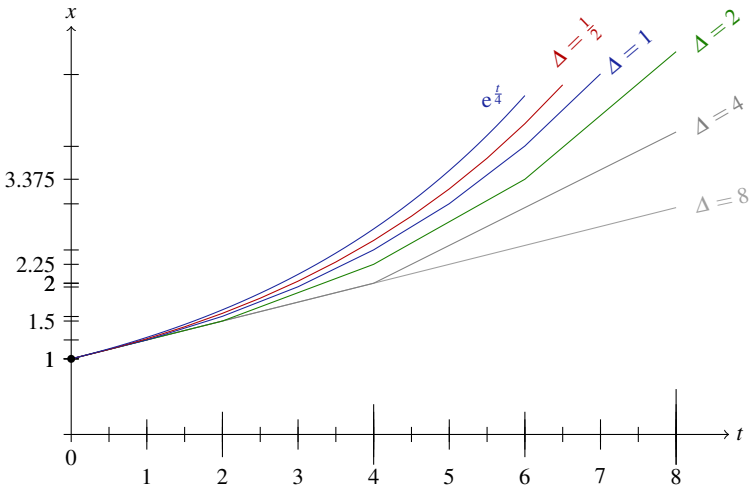


Fig. 2.3 Discretizations of differential equations with discretization time step Δ

$\Delta > 0$ we choose, that discretization will be arbitrarily far away from the true continuous solution for large t.

2.3 The Meaning of Differential Equations

We have obtained an intuitive understanding of how differential equations describe the direction of the evolution of a system as a vector field from Fig. 2.1. But some questions remain. What exactly is a vector field? What does it mean to describe directions of evolution at literally every point in space, of which there are uncountably infinitely many? Could these directions not *possibly contradict each other so that the description becomes ambiguous?* What is the exact meaning of a differential equation in the first place?

The only way to truly understand any system is to understand exactly what each of its pieces does. CPSs are demanding, and misunderstandings about their effect often have far-reaching consequences. So let us start by making the pieces of a CPS unambiguous. The first piece is differential equations.

Note 4 (Importance of meaning) The physical impacts of CPSs do not leave much room for failure. We want to immediately get into the habit of always studying the behavior and exact meaning of all relevant aspects of a CPS.

An ordinary differential equation in explicit form is an equation $y'(t) = f(t, y)$ where $y'(t)$ is meant to be the derivative of y with respect to time t and f is a function of time t and the current state y. A solution is a differentiable function Y of time that satisfies this equation when substituted into the differential equation, i.e.,

when substituting $Y(t)$ for y and the time-derivative $Y'(t)$ of Y at t for $y'(t)$. That is, the time-derivative of the solution at each time is equal to the differential equation's right-hand side, as illustrated for time $t = -1$ in Fig. 2.4.

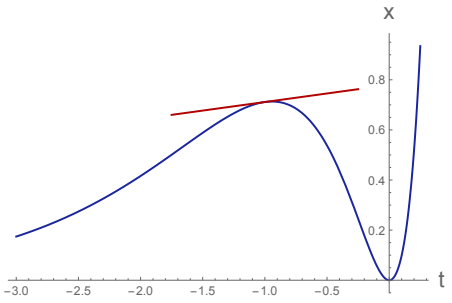


Fig. 2.4 Differential equation solution condition: time-derivative shown in red at $t = -1$ equals right-hand side of differential equation at all times

In the next chapter, we will study a more elegant definition of a solution of a differential equation that is well-attuned with the concepts in this book. But first, we consider the (equivalent) classical mathematical definition of a solution.

Definition 2.1 (Ordinary differential equation). Let $f : D \to \mathbb{R}^n$ be a function on a *domain* $D \subseteq \mathbb{R} \times \mathbb{R}^n$, i.e., an open and connected subset. The function $Y : J \to \mathbb{R}^n$ is a *solution* on the interval $J \subseteq \mathbb{R}$ of the *initial value problem*

$$\begin{pmatrix} y'(t) = f(t,y) \\ y(t_0) = y_0 \end{pmatrix} \tag{2.1}$$

with *ordinary differential equation* (*ODE*) $y' = f(t,y)$, if, for all times $t \in J$

1. solution Y is in the domain $(t, Y(t)) \in D$,
2. time-derivative $Y'(t)$ exists and is $Y'(t) = f(t, Y(t))$, and
3. initial value $Y(t_0) = y_0$ is respected at the initial time, also $t_0 \in J$.

If $f : D \to \mathbb{R}^n$ is continuous, then $Y : J \to \mathbb{R}^n$ is continuously differentiable, because its derivative $Y'(t)$ is $f(t, Y(t))$, which is continuous as f is continuous and Y is differentiable so continuous. Similarly if f is k-times continuously differentiable then Y is $k+1$-times continuously differentiable. The definition is analogous for higher-order differential equations, i.e., those involving higher-order derivatives such as $y''(t)$ or $y^{(n)}(t)$ for $n > 1$.

Let us consider the intuition for this definition. A differential equation (system) can be thought of as a vector field such as the one in Fig. 2.1, where, at each point, the vector shows in which direction the solution evolves. At every point, the vector corresponds to the right-hand side of the differential equation. A solution of a differential equation adheres to this vector field at every point, i.e., the solution (e.g., the solid curve in Fig. 2.1) *locally* follows the direction indicated by the vector of the right-hand side of the differential equation. There are many solutions of the differ-

ential equation corresponding to the vector field illustrated in Fig. 2.1. For the initial value problem (2.1), however, solutions also have to start at the prescribed position y_0 at the initial time t_0 and then follow the differential equations or vector field from this point. In general, there can still be multiple solutions for the same initial value problem, but not for well-behaved differential equations (Sect. 2.9.2).

2.4 A Tiny Compendium of Differential Equation Examples

While cyber-physical systems do not necessitate a treatment and understanding of every differential equation you could ever think of, they do still benefit from a working intuition about differential equations and their relationships to their solutions. The following list of examples indicate by a * which differential equations play an important rôle in this book (Example 2.4, Example 2.5 and Example 2.7), compared to the ones that are merely listed to support a general intuition about the different possibilities that might happen when working with differential equations.

Example 2.1 (A constant differential equation). Some differential equations are easy to solve, especially those with constant right-hand sides. The initial value problem

$$\begin{pmatrix} x'(t) = \frac{1}{2} \\ x(0) = -1 \end{pmatrix}$$

describes that x initially starts at -1 and always changes at the rate $1/2$. It has the solution $x(t) = \frac{1}{2}t - 1$ shown in Fig. 2.5. How can we verify that this is indeed a solution? This can be checked easily by inserting the solution into the differential equation and initial value equation and checking that they evaluate to the desired values according to the initial value problem:

$$\begin{pmatrix} (x(t))' = (\frac{1}{2}t - 1)' = \frac{1}{2} \\ x(0) = \frac{1}{2} \cdot 0 - 1 = -1 \end{pmatrix}$$

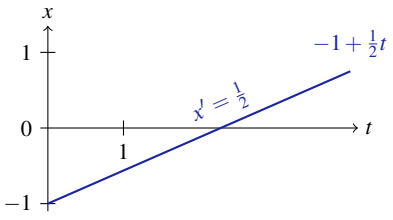


Fig. 2.5 Constant differential equation

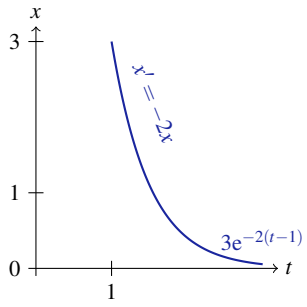


Fig. 2.6 Linear differential equation

Example 2.2 (A negative linear differential equation). Consider an initial value problem whose right-hand side is a linear function with a negative coefficient

$$\begin{pmatrix} x'(t) = -2x(t) \\ x(1) = 3 \end{pmatrix}$$

in which the rate of change of $x(t)$ depends on the current value of $x(t)$ and is in fact $-2x(t)$, so the rate of change gets smaller (more negative) as $x(t)$ gets bigger. This problem describes exponential decay and has the solution $x(t) = 3\mathrm{e}^{-2(t-1)}$ shown in Fig. 2.6, which starts at the initial time $t = 1$. The test, again, is to insert the solution into the (differential) equations of the initial value problems and check:

$$\begin{pmatrix} (3\mathrm{e}^{-2(t-1)})' = -6\mathrm{e}^{-2(t-1)} = -2x(t) \\ x(1) = 3\mathrm{e}^{-2(1-1)} = 3 \end{pmatrix}$$

Example 2.3 (A positive linear differential equation). The initial value problem

$$\begin{pmatrix} x'(t) = \frac{1}{4}x(t) \\ x(0) = 1 \end{pmatrix}$$

shown alongside different discretizations of it in Fig. 2.3 on p. 31 describes exponential growth and has the true continuous solution $x(t) = \mathrm{e}^{\frac{t}{4}}$, which can be checked in the same way as for the previous example:

$$\begin{pmatrix} (\mathrm{e}^{\frac{t}{4}})' = \mathrm{e}^{\frac{t}{4}}(\frac{t}{4})' = \mathrm{e}^{\frac{t}{4}}\frac{1}{4} = \frac{1}{4}x(t) \\ \mathrm{e}^{\frac{0}{4}} = 1 \end{pmatrix}$$

Of course, none of the discretizations actually satisfies these equations, except at the discretization points (the multiples of the respective discretization step Δ). Since the discretizations only satisfy the equation $x'(t) = \frac{1}{4}x(t)$ at integer multiples of Δ and nowhere else, they do not agree with the actual differential equation solution $\mathrm{e}^{\frac{t}{4}}$ anywhere other than at the initial time $t = 0$.

Example 2.4 (Accelerated motion in a straight line).* Consider the important differential equation system $x' = v, v' = a$ and the initial value problem

$$\begin{pmatrix} x'(t) = v(t) \\ v'(t) = a \\ x(0) = x_0 \\ v(0) = v_0 \end{pmatrix} \tag{2.2}$$

This differential equation states that the position $x(t)$ changes with a time-derivative equal to the respective current velocity $v(t)$, which, in turn, changes with a time-derivative equal to the acceleration a, which remains constant. The position and velocity start at the initial values x_0 and v_0. This initial value problem is a *symbolic initial value problem* with symbols x_0, v_0 as initial values (not specific numbers like 5 and 2.3). Moreover, the differential equation has a constant symbol a, and not a

specific number like 0.6, in the differential equation. When concrete numbers $x_0 = 0, v_0 = 0, a = 0.5$ are chosen, the initial value problem (2.2) becomes numerical and has the vector field shown in Fig. 2.2. The initial value problem (2.2) corresponds to a vectorial differential equation with vector $y(t) := (x(t), v(t))$ of dimension $n = 2$:

$$\begin{pmatrix} y'(t) = \begin{pmatrix} x \\ v \end{pmatrix}'(t) = \begin{pmatrix} v(t) \\ a \end{pmatrix} \\ y(0) = \begin{pmatrix} x \\ v \end{pmatrix}(0) = \begin{pmatrix} x_0 \\ v_0 \end{pmatrix} \end{pmatrix} \tag{2.3}$$

The solution of this initial value problem is

$$\begin{aligned} x(t) &= \frac{a}{2}t^2 + v_0 t + x_0 \\ v(t) &= at + v_0 \end{aligned}$$

We can show that this is the solution by inserting the solution into the (differential) equations of the initial value problems and checking:

$$\begin{pmatrix} (\frac{a}{2}t^2 + v_0 t + x_0)' = 2\frac{a}{2}t + v_0 = v(t) \\ (at + v_0)' = a \\ x(0) = \frac{a}{2}0^2 + v_0 0 + x_0 = x_0 \\ v(0) = a0 + v_0 = v_0 \end{pmatrix}$$

Example 2.5 (A two-dimensional linear differential equation for rotation).* In the important differential equation system $v' = w, w' = -v$ with initial value problem

$$\begin{pmatrix} v'(t) = w(t) \\ w'(t) = -v(t) \\ v(0) = 0 \\ w(0) = 1 \end{pmatrix} \tag{2.4}$$

the rate of change of $v(t)$ gets bigger as $w(t)$ gets bigger but, simultaneously, the rate of change of $w(t)$ is $-v(t)$ so it gets smaller as $v(t)$ gets bigger and vice versa. This differential equation describes a rotational effect (Fig. 2.7) with solution

$$\begin{aligned} v(t) &= \sin(t) \\ w(t) &= \cos(t) \end{aligned}$$

That this is the solution can also be checked by inserting the solution into the (differential) equations of the initial value problems and checking:

$$\begin{pmatrix} (\sin(t))' = \cos(t) = w(t) \\ (\cos(t))' = -\sin(t) = -v(t) \\ v(0) = \sin(0) = 0 \\ w(0) = \cos(0) = 1 \end{pmatrix}$$

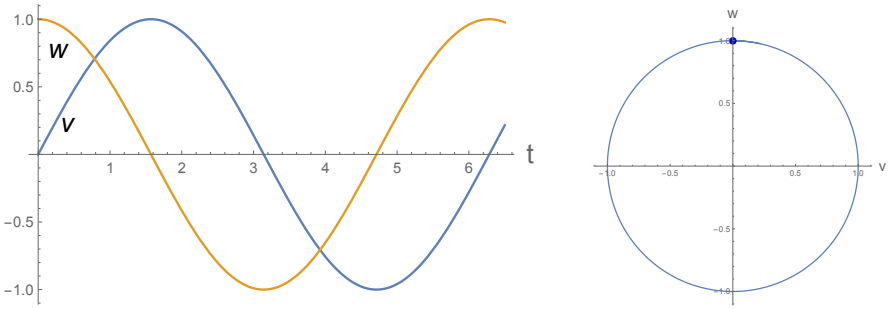


Fig. 2.7 A solution of the rotational differential equations v and w over time t **(left)** and in phase space with coordinates w over v **(right)**

Example 2.6 (A similar two dimensional linear differential equation). Consider the same differential equation system $v' = w, w' = -v$ from Example 2.5 but with different initial values than (2.4):

$$\begin{pmatrix} v'(t) = w(t) \\ w'(t) = -v(t) \\ v(0) = 1 \\ w(0) = 1 \end{pmatrix}$$

This differential equation still describes a rotational effect (Fig. 2.8), but the solution now is

$$\begin{aligned} v(t) &= \cos(t) + \sin(t) \\ w(t) &= \cos(t) - \sin(t) \end{aligned}$$

Showing that this is the solution amounts to inserting the solution into the (differential) equations of the initial value problems and checking:

$$\begin{pmatrix} (\cos(t) + \sin(t))' = -\sin(t) + \cos(t) = w(t) \\ (\cos(t) - \sin(t))' = -\sin(t) - \cos(t) = -v(t) \\ v(0) = \cos(0) + \sin(0) = 1 \\ w(0) = \cos(0) - \sin(0) = 1 \end{pmatrix}$$

Example 2.7 (An adjustable linear differential equation for rotation).* In the important differential equation system $v' = \omega w, w' = -\omega v$ with initial value problem

$$\begin{pmatrix} v'(t) = \omega w(t) \\ w'(t) = -\omega v(t) \\ v(0) = 0 \\ w(0) = 1 \end{pmatrix} \tag{2.5}$$

the rate of change of $v(t)$ gets bigger as $w(t)$ gets bigger but, simultaneously, the rate of change of $w(t)$ is $-v(t)$ so it gets smaller as $v(t)$ gets bigger and vice versa.

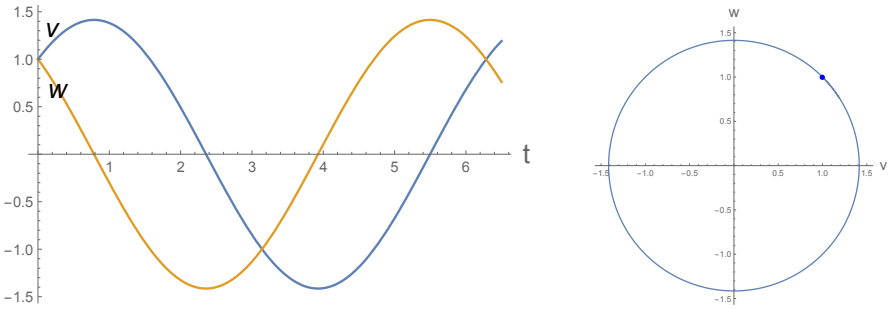


Fig. 2.8 Another solution of the rotational differential equations v and w over time t with initial values $v(0) = w(0) = 1$ **(left)** and in phase space with coordinates w over v **(right)**

But in all places, the rate of change is multiplied by a constant parameter ω, which represents the angular velocity. Bigger magnitudes of ω give faster rotations and positive ω gives clockwise rotations. This differential equation describes a rotational effect (Fig. 2.9) with solution

$$v(t) = \sin(\omega t)$$
$$w(t) = \cos(\omega t)$$

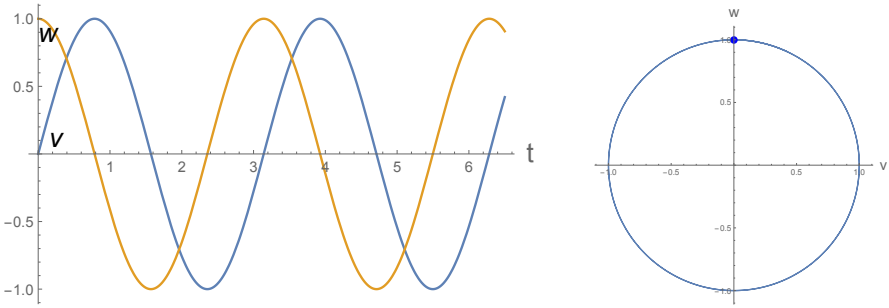


Fig. 2.9 A faster solution of the rotational differential equations v and w over time t with initial values $v(0) = 0, w(0) = 1$ and $\omega = 2$ **(left)** and in phase space with coordinates w over v **(right)**

Some differential equations mention the time variable t, which means that the required time-derivatives change over time.

Example 2.8 (Time square oscillator). Consider the following differential equation system $x'(t) = t^2 y, y'(t) = -t^2 x$, which explicitly mentions the time variable t, and the initial value problem

$$\begin{pmatrix} x'(t) = t^2 y \\ y'(t) = -t^2 x \\ x(0) = 0 \\ y(0) = 1 \end{pmatrix} \tag{2.6}$$

The solution shown in Fig. 2.10(left) illustrates that the system stays bounded but oscillates increasingly quickly. In this case, the solution is

$$\begin{pmatrix} x(t) = \sin\left(\frac{t^3}{3}\right) \\ y(t) = \cos\left(\frac{t^3}{3}\right) \end{pmatrix} \tag{2.7}$$

Note that there is no need to mention time variable t itself in the differential equa-

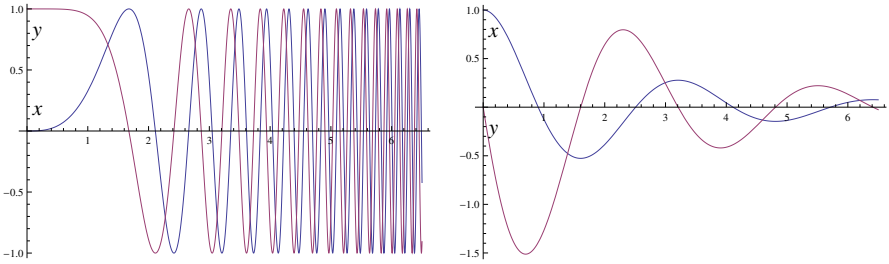


Fig. 2.10 A solution of the time square oscillator **(left)** and of the damped oscillator **(right)** up to time 6.5

tion. We could just as well have added an extra clock variable s with differential equation $s' = 1$ and initial value $s(0) = 0$ to serve as a proxy for time t. This leads to a system equivalent to (2.6) without explicit dependency on the time variable t, since the proxy s behaves in the same way as time t but is an ordinary state variable:

$$\begin{pmatrix} x'(t) = s^2 y \\ y'(t) = -s^2 x \\ s'(t) = 1 \\ x(0) = 0 \\ y(0) = 1 \\ s(0) = 0 \end{pmatrix}$$

This transformation to use s instead of t makes the differential equation *autonomous* because its right-hand side does not depend on the actual time variable t.

In this example, the solution oscillates increasingly quickly. The following example has no time-dependence and a constant frequency but the amplitude of the oscillation decreases over time.

Example 2.9 (Damped oscillator). Consider the linear differential equation $x' = y, y' = -4x - 0.8y$ and the initial value problem

$$\begin{pmatrix} x'(t) = y \\ y'(t) = -4x - 0.8y \\ x(0) = 1 \\ y(0) = 0 \end{pmatrix} \tag{2.8}$$

The solution shown in Fig. 2.10(right) illustrates that the dynamical system decays over time. In this case, the explicit global solution representing the dynamical system is more difficult to write down explicitly but a function that solves it still exists.

Note 5 (Descriptive power of differential equations) As a very general phenomenon, observe that solutions of differential equations can be much more involved than the differential equations themselves, which is part of the representational and descriptive power of differential equations. Pretty simple differential equations can describe quite complicated physical processes.

2.5 Domains of Differential Equations

Now we understand precisely what a differential equation is and how it describes a continuous physical process. In CPS, however, physical processes are not running in isolation, but interact with cyber elements such as computers or embedded systems. When and how do physics and cyber elements interact?

The first thing we need to understand for that is how to describe when physics stops so that the cyber elements take control of what happens next. Obviously, physics does not literally stop evolving, but rather keeps on evolving all the time. Yet, the cyber parts only take effect every now and then, because they only provide input into physics by way of changing the actuators every once in a while. So, our intuition may imagine physics "pauses" for a period of duration 0 and lets the cyber take action to influence the inputs on which physics is based. In fact, cyber may interact with physics over a period of time or after computing for some time to reach a decision. But the phenomenon is still the same. At some point, cyber is done sensing and deliberating and deems it time to act (if cyber never acts, it's boring and will be discarded). At this moment of time, physics needs to "pause" for a conceptual period of time of imaginary duration 0 to give cyber a chance to act.

The cyber and the physics can interface in more than one way. Physics might evolve and the cyber elements interrupt physics periodically to make measurements of the current state of the system in order to decide what to do next (Chap. 9). Or the physics might trigger certain conditions or events that cause cyber elements to compute their respective responses to these events (Chap. 8). Another way to look at this is that a differential equation that a system follows forever without further intervention by anything would not describe a particularly well-controlled system. If physics on its own were already to drive cars safely, we would not need any cyber or any control in the first place. But since physics has not quite passed the driver's license test yet, proper control is rather on the crucial side.

All those ways have in common that our model of physics not only needs to explain how the state changes over time with a differential equation, but it also needs to specify when physics stops evolving to give cyber a chance to perform its task. This information is what is called an *evolution domain* Q of a differential equation, which describes a region that the system cannot leave while following that particular continuous mode of the system. If the system were ever about to leave this region, it would stop evolving right away (for the purpose of giving the cyber parts of the system a chance to act) before it leaves the evolution domain. Of course, the overall idea will be for physics to resume once cyber is done inspecting and acting, but that is a matter for Chap. 3.

Note 6 (Evolution domain constraint) A *differential equation* $x' = f(x)$ *with evolution domain* Q is denoted by

$$x' = f(x) \,\&\, Q$$

using a conjunctive notation (&) between the differential equation and its evolution domain. This notation $x' = f(x) \,\&\, Q$ signifies that the system obeys *both* the differential equation $x' = f(x)$ *and* the evolution domain Q. The system follows this differential equation for any duration while inside the region Q, but is never allowed to leave the region described by Q. So the system evolution has to stop while the state is still in Q.

If, e.g., t is a time variable with $t' = 1$, then $x' = v, v' = a, t' = 1 \,\&\, t \leq \varepsilon$ describes a system that follows the differential equation at most until time $t = \varepsilon$ and not any further, because the evolution domain $Q \overset{\text{def}}{\equiv} (t \leq \varepsilon)$ would be violated after time ε. That can be a useful model of the kind of physics that gives the cyber elements a chance to act at the latest at time ε, because physics is not allowed to continue beyond time $t = \varepsilon$. The evolution domain $Q \overset{\text{def}}{\equiv} (v \geq 0)$, instead, restricts the system $x' = v, v' = a \,\&\, v \geq 0$ to nonnegative velocities. Should the velocity ever be about to become negative while following the differential equation $x' = v, v' = a$, then the system stops evolving before the velocity becomes negative. In a similar way $x' = v, v' = a \,\&\, v \leq 10$ describes a physics that has its velocity limited by an upper bound of 10. But, honestly, keeping the velocity of a car below 10 will also end up requiring some effort on the part of the cyber controller, not just the physics, which is a phenomenon that we will come back to in Chaps. 8 and 9.

In the first two scenarios illustrated in Fig. 2.11, the system starts at time 0 inside the evolution domain Q, which is depicted as a shaded green region in Fig. 2.11. Then the system follows the differential equation $x' = f(x)$ for any period of time, but has to stop before it leaves Q. Here, it stops at different choices for the stopping time r.

In contrast, consider the scenario shown on the right of Fig. 2.11. The system stops at time r and is *not* allowed to evolve until time s, because—even if the system would be back in the evolution domain Q at that time—it has already left the evolution domain Q between time r and s (indicated by dotted lines), which is not

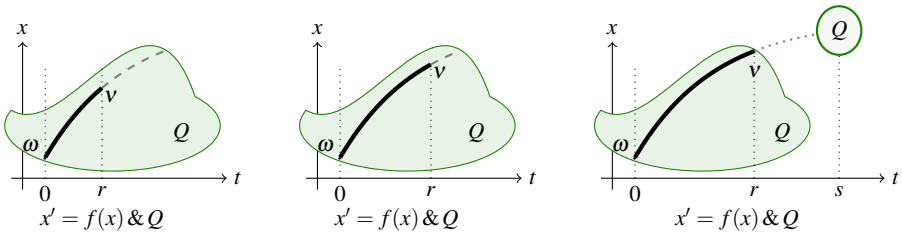


Fig. 2.11 System $x' = f(x)\,\&\,Q$ follows the differential equation $x' = f(x)$ for any duration r but cannot leave the (green) evolution domain Q

allowed. Physics $x' = f(x)\,\&\,Q$ cannot sneak out of its respective evolution domain Q hoping that we will not notice, not even temporarily. Consequently, the continuous evolution on the right of Fig. 2.11 will also stop at time r at the latest and cannot continue any further.

Now that we know what the evolution domain constraint Q of a differential equation is supposed to do, the question is how we can properly describe it in a CPS model? We will need some logic for that. For one thing, we should start getting precise about how to describe the evolution domain Q for a differential equation, just as we have become precise about the understanding of a differential equation itself. The most critical bit of an evolution domain is the question of which points satisfy Q and which ones doesn't, which is what logic is good at making precise.

2.6 Syntax of Continuous Programs

After these preparations for understanding differential equations and domains, we start developing a precise mathematical model. The differential equations with their evolution domains will ultimately need a way of interfacing with discrete computer control programs, because hybrid system models of cyber-physical systems combine discrete dynamics and continuous dynamics. The conceptually easiest and most compositional way to make that happen is to integrate continuous dynamics seamlessly into the computer control programs. This book develops the programming language of *hybrid programs*, which contain discrete features in addition to the differential equations. Hybrid programs and their analysis is developed in layers one after another. For now, we focus on the first layer of this programming language, which contains only the most crucial feature of continuous dynamics.

2.6.1 Continuous Programs

The first element of the syntax of hybrid programs is purely continuous programs.

Note 7 (Continuous programs) Layer 1 of *hybrid programs* (HPs) comprises *continuous programs*. If x is a variable, e any term possibly containing x, and Q a formula of first-order logic of real arithmetic, then continuous programs are of the form

$$\alpha ::= x' = e \,\&\, Q$$

Continuous programs consist of a single statement of the form $x' = e \,\&\, Q$. In later chapters, we will add more statements to form hybrid programs, but we just focus on differential equations for the continuous dynamics for now. The *continuous evolution* $x' = e \,\&\, Q$ expresses that, from the present value of variable x, the system follows the differential equation $x' = e$ for some amount of time within the *evolution domain constraint* Q. What form formula Q can take will be defined below. But it has to enable an unambiguous definition of the set of points that satisfy Q, because the continuous evolution is not allowed to ever leave that region. Further x is a variable, but is also allowed to be a vector of variables and, then, e is a vector of terms of the same dimension. This corresponds to the case of differential equation systems such as

$$x' = v, v' = a \,\&\, (v \geq 0 \wedge v \leq 10)$$

Differential equations are allowed without an evolution domain constraint Q as well:

$$x' = y, y' = x + y^2$$

which corresponds to choosing *true* for Q, since the formula *true* is true everywhere and, thus, actually imposes no condition on the state whatsoever, because every state easily satisfies the formula *true* with flying colors. Of course, we will have to be more precise about what terms e are allowed and what formulas Q are allowed, which is what we will pursue next.

2.6.2 Terms

A rigorous definition of the syntax of hybrid programs also depends on defining what a term e and what a formula Q of first-order logic of real arithmetic are. Terms e occur on the right-hand side of differential equations, and formulas Q are their evolution domains.

Definition 2.2 (Terms). A *term* e is a polynomial term defined by the grammar (where $e, \tilde{e}$ are terms, x is a variable, and c is a rational number constant)

$$e, \tilde{e} ::= x \mid c \mid e + \tilde{e} \mid e \cdot \tilde{e}$$

This grammar[2] means that a term e (or a term $\tilde{e}$) is either a variable x, or a rational number constant $c \in \mathbb{Q}$ such as 0 or 1 or $5/7$, or a sum of terms $e, \tilde{e}$, or it is

[2] From a formal languages perspective, it would be fine to use the equivalent grammar

a product of terms $e, \tilde{e}$, which are again built in this way recursively. The cases of variables x or constants c are called *atomic terms*. The other cases take two terms as input to produce more complex terms and are called *compound terms*. The addition term $e+\tilde{e}$, for example, consists of two terms e and $\tilde{e}$. So does the multiplication term $e \cdot \tilde{e}$. Subtraction $e-\tilde{e}$ is another useful case, but it turns out that it is already included, because the subtraction term $e-\tilde{e}$ is already definable by the term $e+(-1)\cdot\tilde{e}$. That is why we will not worry about subtraction in developing the theory, but use it in our examples regardless. Unary negation $-e$ is helpful, too, but also already included as $0-e$. For example, $4+x\cdot 2$ and $x\cdot 2+y\cdot y$ are terms and $4\cdot x-y\cdot y+1$ will also be considered as a term even if it really should have been written as $((4\cdot x)+(((-1)\cdot y)\cdot y))+1$. Definition 2.2 yields all polynomials. The polynomial x^3+5x^2-x+4 can, e.g., be represented by the term $x\cdot x\cdot x+5\cdot x\cdot x+(-1)\cdot x+4$.

If you were to implement the syntax of terms in a computer program, you could implement the four cases of the syntax of terms in Definition 2.2 as constructors of a data type. An atomic term can either be constructed by providing a variable x or a computer representation of a rational number $c \in \mathbb{Q}$. Compound terms can be constructed with the sum constructor $e+\tilde{e}$ when providing two previously constructed terms $e, \tilde{e}$. Or they can be constructed with the product constructor $e\cdot\tilde{e}$. That way, every concrete term such as $x\cdot 2+y\cdot y$ can be represented in the data structure by calling the respective constructors with the appropriate arguments.

2.6.3 First-Order Formulas

The formulas of first-order logic of real arithmetic are defined as usual in first-order logic, except that, being the logic for real arithmetic, it also uses the specific language of real arithmetic, for example $e \geq \tilde{e}$ for greater-or-equal. First-order logic supports the logical connectives not ($\neg$), and ($\wedge$), or ($\vee$), implies ($\rightarrow$), and bi-implication alias equivalence ($\leftrightarrow$), as well as quantifiers for all ($\forall$) and exists ($\exists$). In the first-order logic of *real* arithmetic, $\forall, \exists$ quantify over the reals $\mathbb{R}$.

Definition 2.3 (Formulas of first-order logic of real arithmetic). The *formulas* of *first-order logic of real arithmetic* are defined by the following grammar (where P, Q are formulas of first-order logic of real arithmetic, $e, \tilde{e}$ are terms, and x is a variable):

$$P, Q ::= e \geq \tilde{e} \mid e = \tilde{e} \mid \neg P \mid P \wedge Q \mid P \vee Q \mid P \rightarrow Q \mid P \leftrightarrow Q \mid \forall x P \mid \exists x P$$

$$e ::= x \mid c \mid e+e \mid e\cdot e$$

with a single nonterminal e. We use the slightly more verbose form with two redundant nonterminals $e, \tilde{e}$ just to emphasize directly that a term can be a sum $e+\tilde{e}$ of any arbitrary and possibly different terms $e, \tilde{e}$ and does not have to consist of sums $e+e$ of one and the same term e. The two presentations of the grammar are equivalent.

The usual abbreviations are allowed, such as $e \leq \tilde{e}$ for $\tilde{e} \geq e$ and $e < \tilde{e}$ for $\neg(e \geq \tilde{e})$ and will be used in examples even if the theory does not need to be bothered with them.

Example formulas we saw before as evolution domains are $t \leq \varepsilon$, which we mean as a bound on time t, as well as $v \geq 0$, which we used as a velocity bound. But these first-order logic formulas themselves do not tell us that t is a time and v a velocity, which is the rôle of the differential equations. However, the formula $v \geq 0$ quite precisely instructs us where it is true (which is what it semantics will define rigorously). Whenever the value of velocity v is greater-or-equal 0, the formula $v \geq 0$ will be *true*, otherwise *false*. The formula $t \leq \varepsilon \wedge v \geq 0$, which is the conjunction of the formulas $t \leq \varepsilon$ and $v \geq 0$, is used as a domain for systems that evolve at most till time ε and also cannot move backwards. An implication $t \geq 2 \rightarrow v \leq 5$ may be used as an evolution domain to say that if the system is beyond time 2, then the velocity is at most 5.

The quantifiers for all ($\forall$) and exists ($\exists$) over the reals are less relevant for evolution domains and become more important in later chapters of the book. They are already shown here because they are a characteristic part of first-order logic. For example, $\forall x \exists y\,(y > x)$ expresses that for all real numbers x there is a real number y that is larger than x. The formula $\exists x\,(x \cdot x = 2)$ expresses that there is a real number whose square is 2, which is true thanks to the real number $\sqrt{2}$. But the formula $\exists x\,(x \cdot x = -1)$ is not true over the reals, because the squares of all real numbers are nonnegative and the imaginary unit i, which satisfies $\mathrm{i}^2 = -1$, is not a real number. The formula $\exists y\,(x < y \wedge y < x + 1)$ is also true, no matter what the value of x is, but would be false over the integers, which suffer from a clear lack of numbers between x and $x + 1$.

Expedition 2.1 (Naming conventions)

In this book, we generally follow these naming conventions (programs, function, and predicate symbols will be introduced in later chapters):

Letters	Convention
x, y, z	variables
$e, \tilde{e}$	terms
P, Q	formulas
α, β	programs
c	constant symbols
f, g, h	function symbols
p, q, r	predicate symbols

In any application context, it may be better to deviate from this convention and follow the naming conventions of the application. For example, x is often used for position, v for velocity, and a for acceleration, even if all are variables.

2.7 Semantics of Continuous Programs

This is the first of many occasions in this textbook where we observe a distinct dichotomy between syntax and semantics. The syntax defines the notation, i.e., what questions can be written down and how. But to the innocent bystander and any self-respecting logician, the syntax just provides a long list of funny, arbitrary symbols, until their intended meaning has been clarified. The syntax is only given meaning by the semantics, which defines what real or mathematical object each element of the syntax stands for. The symbols that the syntax uses are arbitrary and completely meaningless until the semantics defines what the symbols mean. Of course, the syntax is cleverly chosen to already remind us of what it is supposed to stand for.

Note 8 (Syntax versus semantics) Syntax just defines arbitrary notation. Its meaning is defined by the semantics.

It is by way of this clear distinction of syntactic and semantic objects that we will ultimately develop a nuanced and accurate understanding of the relationships between mathematical meaning and computer-based insights. Without such a clear distinction, there can be no subsequent alignment and relation. Numerous mistakes in reasoning can be traced back to the lack of a clear separation of the syntactic object-level and semantic meta-level elements of a development. Object-level expressions are expressions in the language (for example first-order logic). Meta-level statements are statements about the language, phrased, for example, in mathematics or English.

2.7.1 Terms

The meaning of a continuous evolution $x' = e \,\&\, Q$ depends on understanding the meaning of term e. A term e is a syntactic expression. The meaning of a term e is given by the real number to which it evaluates. In a term, e, the symbol $+$, of course, means addition of real numbers, $\cdot$ means multiplication, and the meaning of the constant symbol $5/7$ is the rational number five sevenths.

But the overall value of term e also depends on how we interpret the variables appearing in the term e. What values those variables have changes depending on the current state of the CPS. A state needs to let us know what real values all variables of e have, before we are able to say what real value term e evaluates to in that state. In fact, a *state* ω *is* nothing but a mapping from variables to real numbers, such that it associates a real value $\omega(x)$ with each variable x. The *set of states* is denoted $\mathcal{S}$. In other words, if $\mathcal{V}$ is the set of all variables, then a state $\omega \in \mathcal{S}$ is a function $\omega : \mathcal{V} \to \mathbb{R}$ whose value $\omega(x) \in \mathbb{R}$ at $x \in \mathcal{V}$ indicates the real value that variable x has in state ω.

Since the value of a term depends on the state, we will use the notation $\omega[\![e]\!]$ for the real value that the term e evaluates to in state ω. This notation is reminiscent of

function application $\omega(e)$, but when a state is a function $\omega : \mathscr{V} \to \mathbb{R}$, then $\omega(e)$ is only defined if e is a variable, not if it is a term $x+7$. The notation $\omega[\![e]\!]$, thus, lifts to terms e the value that the state ω assigns only to variables $x \in \mathscr{V}$.

Definition 2.4 (Semantics of terms). The *value of term* e in state $\omega \in \mathscr{S}$ is a real number denoted $\omega[\![e]\!]$ and is defined by induction on the structure of e:

$$\begin{aligned} \omega[\![x]\!] &= \omega(x) && \text{if } x \text{ is a variable} \\ \omega[\![c]\!] &= c && \text{if } c \in \mathbb{Q} \text{ is a rational constant} \\ \omega[\![e+\tilde{e}]\!] &= \omega[\![e]\!] + \omega[\![\tilde{e}]\!] \\ \omega[\![e \cdot \tilde{e}]\!] &= \omega[\![e]\!] \cdot \omega[\![\tilde{e}]\!] \end{aligned}$$

That is, the value of a variable x in state ω is given directly by the state ω, which is a mapping from variables to real numbers. A rational constant c such as 0.5 evaluates to itself. The value of a sum term of the form $e+\tilde{e}$ in a state ω is the sum of the values of the subterms e and $\tilde{e}$ in ω, respectively. Those lines of Definition 2.4 already explain that $\omega[\![x+y]\!] = \omega(x)+\omega(y)$. Likewise, the value of a product term of the form $e \cdot \tilde{e}$ in a state ω is the product of the values of the subterms e and $\tilde{e}$ in ω, respectively. Each term has a value in every state, because each case of the syntactic form of terms (Definition 2.2) has been given a semantics in Definition 2.4. The semantics of every term, thus, is a mapping from states $\omega \in \mathscr{S}$ to the real value $\omega[\![e]\!]$ that the term e evaluates to in the respective state ω.

The value of a variable-free term like $4+5 \cdot 2$ does not depend on the state ω at all. In this case, the value is 14. The value of a term with variables, like $4+x \cdot 2$, depends on what value the variable x has in state ω. Suppose $\omega(x) = 5$, then the value is also $\omega[\![4+x \cdot 2]\!] = 4+\omega(x) \cdot 2 = 14$. However, for $\nu(x) = 2$, it evaluates to $\nu[\![4+x \cdot 2]\!] = 4+\nu(x) \cdot 2 = 8$, instead. While, technically, the state ω is a mapping from all variables to real numbers, the values that ω gives to most variables are immaterial; only the values of the variables that actually occur in the term have any influence (Sect. 5.6.5). So while the value of $4+x \cdot 2$ very much depends on the value of x, it does not depend on the value that variable y has since y does not even occur in the term $4+x \cdot 2$. This is in contrast to the term $x \cdot 2+y \cdot y$ whose value depends on the values of both x and y but not on z, so for $\omega(x) = 5$ and $\omega(y) = 4$, it evaluates to $\omega[\![x \cdot 2+y \cdot y]\!] = \omega(x) \cdot 2+\omega(y) \cdot \omega(y) = 26$.

The way that Definition 2.4 defines a semantics for terms directly corresponds to the definition of a recursive function in a functional programming language by distinguishing the respective constructors of the data types for terms. For each constructor, there is a corresponding case that defines its value in the argument state. And if that function makes a recursive call, as in the cases of $\omega[\![e+\tilde{e}]\!]$ and $\omega[\![e \cdot \tilde{e}]\!]$, it does so on terms that are smaller to make sure the function terminates and is well defined on all inputs.

Expedition 2.2 (Semantic brackets $[\![\cdot]\!] : \text{Trm} \to (\mathscr{S} \to \mathbb{R})$)

There are multiple equivalent ways of understanding Definition 2.4. The most elementary understanding is, as written, to understand it as defining the real value $\omega[\![e]\!]$ in a state ω for each term e by an inductive definition on the structure of e. If e is a variable, the first line is applicable, if e is a rational constant symbol the second line. If e is a sum term, then the third line, and the fourth line is applicable if e is a product term. Since every term fits to exactly one of those four shapes and the right-hand sides use the definition on smaller subterms of e that have already received a value by this definition, the definition is well defined.

More eloquently, Definition 2.4 can be read as defining an operator $[\![\cdot]\!]$ that defines the semantics of a term e as a mapping $[\![e]\!] : \mathscr{S} \to \mathbb{R}$ from states to real numbers such that the real value $\omega[\![e]\!]$ is computed according to the equations in Definition 2.4. That is, the function $[\![e]\!]$ is defined by induction on the structure of e:

$$\begin{aligned}
[\![x]\!] &: \mathscr{S} \to \mathbb{R};\ \omega \mapsto \omega(x) && \text{if } x \text{ is a variable}\\
[\![c]\!] &: \mathscr{S} \to \mathbb{R};\ \omega \mapsto c && \text{if } c \in \mathbb{Q} \text{ is a rational constant}\\
[\![e+\tilde{e}]\!] &: \mathscr{S} \to \mathbb{R};\ \omega \mapsto \omega[\![e]\!] + \omega[\![\tilde{e}]\!]\\
[\![e\cdot\tilde{e}]\!] &: \mathscr{S} \to \mathbb{R};\ \omega \mapsto \omega[\![e]\!] \cdot \omega[\![\tilde{e}]\!]
\end{aligned}$$

The notation for evaluating $[\![e]\!]$ at state ω is still $\omega[\![e]\!]$. For example, the last line defines the function $[\![e\cdot\tilde{e}]\!]$ as the function $[\![e\cdot\tilde{e}]\!] : \mathscr{S} \to \mathbb{R}$ that maps state ω to the real value given by the product $\omega[\![e]\!] \cdot \omega[\![\tilde{e}]\!]$ of the values $\omega[\![e]\!]$ and $\omega[\![\tilde{e}]\!]$.

The two ways of understanding Definition 2.4 are equivalent. The former is more elementary. The latter generalizes more directly to defining a semantics for other syntactic objects when choosing a different semantic domain than $\mathscr{S} \to \mathbb{R}$. When Trm is the set of terms, the semantic brackets for terms define an operator $[\![\cdot]\!] : \text{Trm} \to (\mathscr{S} \to \mathbb{R})$ that defines the meaning $[\![e]\!]$ for each term $e \in \text{Trm}$, which, in turn, defines the real value $\omega[\![e]\!] \in \mathbb{R}$ for each state $\omega \in \mathscr{S}$. In a functional programming language, the difference between the two styles of definition of the semantics of terms is exactly currying, i.e., translating a function that takes multiple arguments into a sequence of functions each of which only takes a single argument.

2.7.2 First-Order Formulas

Unlike for terms, the value of a logical formula is not a real number but instead *true* or *false*. Whether a logical formula evaluates to *true* or *false* still depends on the interpretation of its symbols. In first-order logic of real arithmetic, the meaning of all symbols except the variables is fixed. The meaning of terms and of formulas of first-order logic of real arithmetic is as usual in first-order logic, except that $+$

really means addition, $\cdot$ means multiplication, $\geq$ means greater or equals, and the quantifiers $\forall x$ and $\exists x$ quantify over the reals. As always in first-order logic, the meaning of $\wedge$ is conjunction and that of $\vee$ is disjunction, etc. The meaning of the variables in the formula is again determined by the state ω of the CPS.

In direct analogy to the real-valued semantics $\omega[\![e]\!] \in \mathbb{R}$ of terms e, we might define a boolean-valued semantics $\omega[\![P]\!] \in \{true, false\}$ for formulas P that defines what truth-value (*true* or *false*) the formula P has in state ω (Exercise 2.10). However, our interest is ultimately in understanding which formulas are true, because the complement then also already tells us which formulas are false. That is why it simplifies matters if we define the semantics via the set of states $[\![P]\!]$ in which formula P is true. Then $\omega \in [\![P]\!]$ will say that formula P is true in state ω. The opposite $\omega \notin [\![P]\!]$ says that formula P is not true so false in state ω. For consistency with other books, this chapter uses the *satisfaction relation* notation $\omega \models P$ instead of $\omega \in [\![P]\!]$, but they mean the same thing.

Definition 2.5 (First-order logic semantics). The first-order formula P is true in state ω, is written $\omega \models P$, and is defined inductively as follows:

- $\omega \models e = \tilde{e}$ iff $\omega[\![e]\!] = \omega[\![\tilde{e}]\!]$
  That is, an equation $e = \tilde{e}$ is true in a state ω iff the terms e and $\tilde{e}$ evaluate to the same number in ω according to Definition 2.4.
- $\omega \models e \geq \tilde{e}$ iff $\omega[\![e]\!] \geq \omega[\![\tilde{e}]\!]$
  That is, a greater-or-equals inequality is true in a state ω iff the term on the left evaluates to a number that is greater than or equal to the value of the right term.
- $\omega \models \neg P$ iff $\omega \not\models P$, i.e., if it is not the case that $\omega \models P$
  That is, a negated formula $\neg P$ is true in state ω iff the formula P itself is not true in ω.
- $\omega \models P \wedge Q$ iff $\omega \models P$ and $\omega \models Q$
  That is, a conjunction is true in a state iff both conjuncts are true in said state.
- $\omega \models P \vee Q$ iff $\omega \models P$ or $\omega \models Q$
  That is, a disjunction is true in a state iff either of its disjuncts is true in said state.
- $\omega \models P \rightarrow Q$ iff $\omega \not\models P$ or $\omega \models Q$
  That is, an implication is true in a state iff either its left-hand side is false or its right-hand side is true in said state.
- $\omega \models P \leftrightarrow Q$ iff ($\omega \models P$ and $\omega \models Q$) or ($\omega \not\models P$ and $\omega \not\models Q$)
  That is, a bi-implication is true in a state iff both sides are true or both sides are false in said state.
- $\omega \models \forall x P$ iff $\omega_x^d \models P$ for all $d \in \mathbb{R}$
  That is, a universally quantified formula $\forall x P$ is true in a state iff its *kernel* P is true in all variations of the state, no matter what real number d the quantified variable x evaluates to in the variation ω_x^d defined below.

- $\omega \models \exists x P$ iff $\omega_x^d \models P$ for some $d \in \mathbb{R}$
  That is, an existentially quantified formula $\exists x P$ is true in a state iff its kernel P is true in some variation of the state, for a suitable real number d that the quantified variable x evaluates to in the variation ω_x^d.

If $\omega \models P$, then we say that P is true at ω or that ω is a model of P. Otherwise, i.e., if $\omega \not\models Q$, we say that P is false at ω. A formula P is *valid*, written $\vDash P$, iff $\omega \models P$ for all states ω. Formula P is called *satisfiable* iff there is a state ω such that $\omega \models P$. Formula P is *unsatisfiable* iff there is no such ω. The set of states in which formula P is true is written $[\![P]\!] = \{\omega : \omega \models P\}$.

The definition of the semantics of quantifiers uses *state modifications*, i.e., ways of changing a given state ω around by changing the value of a variable x but leaving the values of all other variables alone. The notation $\omega_x^d \in \mathcal{S}$ denotes the state that agrees with state $\omega \in \mathcal{S}$ except for the interpretation of variable x, which is changed to the value $d \in \mathbb{R}$. That is, the state ω_x^d has the same values that the state ω has for all variables other than the variable x, and the value of the variable x in ω_x^d is d:

$$\omega_x^d(y) \stackrel{\text{def}}{=} \begin{cases} d & \text{if } y \text{ is the modified variable } x \\ \omega(y) & \text{if } y \text{ is another variable} \end{cases} \tag{2.9}$$

The formula $x > 0 \wedge x < 1$ is satisfiable, because all it takes for it to be true is a state ω in which, indeed, the value of x is a real number between zero and one, such as 0.592. The formula $x > 0 \wedge x < 0$ is unsatisfiable, because it is kind of hard (read: impossible) to find a state which satisfies both conjuncts at once. The formula $x > 0 \vee x < 1$ is valid, because there is no state in which it would not be true, because, surely, x will either be positive or smaller than one.

In the grand scheme of things, the most exciting formulas are the ones that are valid, i.e., $\vDash P$, because that means they are true no matter what state a system is in. Valid formulas, and how to find out whether a formula is valid, will keep us busy quite a while in this textbook. Consequences of a formula set Γ are also amazing, because, even if they may not be valid per se, they are true whenever Γ is. For this chapter, however, it is more important which formulas are true in a given state, because that is what we need to make sense of an evolution domain of a continuous evolution, which would be futile if it were true in all states.

For example, the formula $\exists y\,(y > x)$ is valid, so $\vDash \exists y\,(y > x)$, because it is true, i.e., $\omega \in [\![\exists y\,(y > x)]\!]$, in all states ω, as there always is a real number y that is a little larger than the value of x, whatever real value x might have in ω. The formula $t \leq \varepsilon$ is not valid. Its truth-value depends on the value of its variables t and ε. In a state ω with $\omega(t) = 0.5$ and $\omega(\varepsilon) = 1$, the formula is true, so $\omega \in [\![t \leq \varepsilon]\!]$. But in a state ν with $\nu(t) = 0.5$ and $\nu(\varepsilon) = 0.1$, the formula is false, so $\nu \notin [\![t \leq \varepsilon]\!]$. In a state ω with $\omega(t) = 0.5$ and $\omega(\varepsilon) = 1$ and $\omega(v) = 5$, even the formula $t \leq \varepsilon \wedge v \geq 0$ is true, so $\omega \in [\![t \leq \varepsilon \wedge v \geq 0]\!]$, because both $\omega \in [\![t \leq \varepsilon]\!]$ and $\omega \in [\![v \geq 0]\!]$.

As will be elaborated in Sect. 5.6.5, the only relevant information from the state is the values of the free variables, i.e., those that occur outside the scope of quan-

tifiers for that variables. For example the truth-value of $\exists y\,(y^2 \leq x)$ depends on the value that its free variable x has in the state, but does not depend on the value of variable y, because the existential quantifier $\exists y$ will give y a new value anyhow. For example, $\omega \in [\![\exists y\,(y^2 \leq x)]\!]$ for a state ω with $\omega(x) = 5$, regardless of what $\omega(y)$ is and regardless of $\omega(z)$, because z does not occur at all and y only occurs in the scope of a quantifier. But $\nu \notin [\![\exists y\,(y^2 \leq x)]\!]$ in state ν with $\nu(x) = -1$, because it is impossible to find a real number whose square is less than or equal to -1. Come to think of it, $x \geq 0$ would have been an easier way to state $\exists y\,(y^2 \leq x)$, because the two formulas are equivalent, i.e., have the same truth-value in every state. All states define real values for all variables, because states are (total) functions from the variables to the reals. But the only relevant values are the values of the free variables of the formula.

A formula P is valid iff the formula $\forall x P$ is valid, because validity means truth in all states, which includes all real values for all variables, in particular the variable x. Likewise, P is satisfiable iff the formula $\exists x P$ is satisfiable, because satisfiability means truth in some state, which provides a real value for all variables, even variable x. In a similar way, we could prefix universal quantifiers explicitly for all free variables when asking for validity, because that implicitly quantifies over all real values of all variables. We could also prefix existential quantifiers explicitly for the free variables when asking for satisfiability.

Of course, which quantifier we implicitly mean for a formula with free variables such as $\forall y\,(y^2 > x)$ depends. If we ask whether $\forall y\,(y^2 > x)$ is true in the state ω with $\omega(x) = -1$, there is no implicit quantifier, because we ask about that specific state, and the answer is yes, so $\omega \in [\![\forall y\,(y^2 > x)]\!]$. When asking whether $\forall y\,(y^2 > x)$ is true in the state ν with $\nu(x) = 0$, then the answer is no, since $\nu \notin [\![\forall y\,(y^2 > x)]\!]$. If we ask whether $\forall y\,(y^2 > x)$ is valid, we do not provide a specific state, because validity requires truth in all states, so implicitly quantifies all free variables universally. The answer is no, because $\forall y\,(y^2 > x)$ is not valid as witnessed by the above state ν. The variation $\forall y\,(y^2 \geq -x^2)$ is valid, written $\models \forall y\,(y^2 \geq -x^2)$. If we ask whether $\forall y\,(y^2 > x)$ is satisfiable, we do not provide a specific state either, because the question is whether there is a state ω with $\omega \in [\![\forall y\,(y^2 > x)]\!]$, so the free variables are implicitly quantified existentially. The answer is yes as witnessed by the above state ω.

With the semantics, we now know how to evaluate whether an evolution domain Q of a continuous evolution $x' = e \,\&\, Q$ is true in a particular state ω or not. If $\omega \in [\![Q]\!]$, then the evolution domain Q holds in that state. Otherwise (i.e., if $\omega \notin [\![Q]\!]$), Q does not hold in ω. Yet, in which states ω do we even need to check the evolution domain? We need to find some way of saying that the evolution domain constraint Q is checked for whether it is true (i.e., $\omega \in [\![Q]\!]$) in all states ω along the solution of the differential equation. That will be the next item on our agenda.

2.7.3 Continuous Programs

The semantics of continuous programs surely depends on the semantics of their pieces, which include terms and formulas. The latter have now both been defined, so the next step is giving continuous programs themselves a proper semantics.

In order to keep things simple, all we care about for now is the observation that running a continuous program $x' = e \,\&\, Q$ takes the system from an initial state ω to a new state ν. And, in fact, one crucial aspect to notice is that there is not only one state ν that $x' = e \,\&\, Q$ can reach from ω just as there is not only one solution of the differential equation $x' = e$. Even in cases where there is a unique solution of maximal duration, there are still many different solutions differing only in the duration of the solution. Thus, the continuous program $x' = e \,\&\, Q$ can lead from initial state ω to more than one possible state ν. Which states ν are reachable from an initial state ω along the continuous program $x' = e \,\&\, Q$ exactly? Well, these should be the states ν to which ω can be connected with a solution of the differential equation $x' = e$ that remains entirely within the set of states where the evolution domain constraint Q holds true, as illustrated in Fig. 2.12. Giving this a precise meaning requires going back and forth between syntax and semantics carefully.

Definition 2.6 (Semantics of continuous programs). The state ν is reachable from initial state ω by the continuous program $x_1' = e_1, \ldots, x_n' = e_n \,\&\, Q$ iff there is a *solution* φ of some duration $r \geq 0$ along $x_1' = e_1, \ldots, x_n' = e_n \,\&\, Q$ from state ω to state ν, i.e., a function $\varphi : [0, r] \to \mathscr{S}$ such that:

- initial and final states match: $\varphi(0) = \omega, \varphi(r) = \nu$;
- φ respects the differential equations: For each variable x_i, the value $\varphi(\zeta)[\![x_i]\!] = \varphi(\zeta)(x_i)$ of x_i at state $\varphi(\zeta)$ is continuous in ζ on $[0, r]$ and, if $r > 0$, has a time-derivative of value $\varphi(\zeta)[\![e_i]\!]$ at each time $\zeta \in [0, r]$, i.e.,

$$\frac{\mathsf{d}\varphi(t)(x_i)}{\mathsf{d}t}(\zeta) = \varphi(\zeta)[\![e_i]\!]$$

- the value of other variables $y \notin \{x_1, \ldots, x_n\}$ remains constant throughout the continuous evolution, that is $\varphi(\zeta)[\![y]\!] = \omega[\![y]\!]$ for all times $\zeta \in [0, r]$;
- and φ respects the evolution domain at all times: $\varphi(\zeta) \in [\![Q]\!]$ for each $\zeta \in [0, r]$.

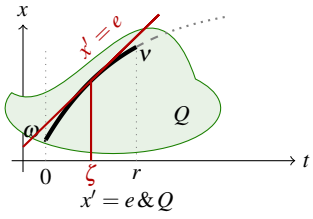


Fig. 2.12 Illustration of the dynamics of continuous programs

Observe that this definition is explicit about the fact that variables without differential equations do not change during a continuous program. The semantics is *explicit change*: nothing changes unless a program statement specifies how. Further observe the explicit passing from syntax to semantics[3] by the use of the semantics function $[\![\cdot]\!]$ in Definition 2.6. Finally note that for duration $r = 0$, no condition is imposed on the time-derivative, because there are no time-derivatives for a function that is only defined at 0. Consequently, the only conditions that Definition 2.6 imposes for duration 0 are that the initial state ω and final state ν agree and that the evolution domain constraint Q is respected at that state: $\omega \in [\![Q]\!]$. Later chapters will have a slightly refined understanding, but Definition 2.6 is sufficient for Part I.

2.8 Summary

This chapter gave a precise semantics to differential equations and presented first-order logic of real arithmetic, which we use for the evolution domain constraints within which differential equations are supposed to stay. The operators in first-order logic of real arithmetic and their informal meaning is summarized in Table 2.1.

While this chapter provided an important continuous foundation, all its elements will be revisited in more detail in subsequent chapters. The semantics of continuous programs will be revisited in their interaction with discrete programs in Chap. 3. First-order logic will be substantially generalized in more detail in Chap. 4. In fact, even the set of terms will be extended in Part II but will be with us throughout the book.

Table 2.1 Operators and meaning in first-order logic of real arithmetic (FOL)

FOL	Operator	Meaning
$e = \tilde{e}$	equals	true iff values of e and $\tilde{e}$ are equal
$e \geq \tilde{e}$	greater or equals	true iff value of e greater-or-equal to $\tilde{e}$
$\neg P$	negation / not	true iff P is false
$P \wedge Q$	conjunction / and	true iff both P and Q are true
$P \vee Q$	disjunction / or	true iff P is true or if Q is true
$P \rightarrow Q$	implication / implies	true iff P is false or Q is true
$P \leftrightarrow Q$	bi-implication / equivalent	true iff P and Q are both true or both false
$\forall x P$	universal quantifier / for all	true iff P is true for all values of real variable x
$\exists x P$	existential quantifier / exists	true iff P is true for some values of real variable x

[3] This important aspect is often overlooked. Informally, one might say that x obeys $x' = e$, but this cannot mean that the equation $x' = e$ holds true, because it is not even clear what the meaning of x' would be. A syntactic variable x has a meaning in a single state, but a time-derivative cannot. The semantical value of x along a function φ, instead, can have a well-defined derivative at time ζ. This requires passing back and forth between syntax and semantics.

2.9 Appendix

For your reference, this appendix already contains a short primer on some important results about differential equations from the literature [10]. While not crucial for the immediate technical development in subsequent chapters, this appendix is a helpful resource to come back to as desired for important meta-results for the general theory of differential equations. This appendix also lists useful counterexamples highlighting that the assumptions of the respective theorems are necessary.

The most crucial insights are that continuous differential equations have solutions (Sect. 2.9.1) and locally Lipschitz continuous differential equations (such as continuously differentiable differential equations) have unique solutions (Sect. 2.9.2).

2.9.1 Existence Theorems

Classical theorems guarantee existence and/or uniqueness of solutions of differential equations (not necessarily closed-form solutions with elementary functions, though). The existence theorem is due to Peano [6]. A proof can be found in the literature [10, Theorem 10.IX].

Theorem 2.1 (Peano's existence theorem). *Let $f : D \to \mathbb{R}^n$ be a continuous function on an open, connected domain $D \subseteq \mathbb{R} \times \mathbb{R}^n$. Then, the initial value problem (2.1) with $(t_0, y_0) \in D$ has a solution. Every solution of (2.1) can be continued arbitrarily close to the boundary of D.*

Peano's theorem only proves that a solution exists, not for what duration it exists. Still, it shows that every solution can be *continued arbitrarily close to the boundary* of the domain. That is, the closure of the graph of the solution is not a compact subset of D, which means [10, §6.VII] that the solution exists either globally on $[0, \infty)$ or on a bounded interval $[0, r)$ with the solution approaching the boundary of D or an infinite norm at r. The *graph* $\operatorname{graph}(y)$ of a function $y : J \to D$ is the subset $\{(t, y(t)) \,:\, t \in J\}$ of $J \times D$.

Peano's theorem shows the existence of solutions of continuous differential equations on open, connected domains, but there can still be multiple solutions.

Example 2.10 (Nonunique solutions). The initial value problem with the following continuous differential equation

$$\begin{pmatrix} y' = \sqrt[3]{|y|} \\ y(0) = 0 \end{pmatrix}$$

has multiple solutions, for example

$$y(t) = 0$$

$$y(t) = \left(\frac{2}{3}t\right)^{\frac{3}{2}}$$

$$y(t) = \begin{cases} 0 & \text{for } t \leq s \\ \left(\frac{2}{3}(t-s)\right)^{\frac{3}{2}} & \text{for } t > s \end{cases}$$

where $s \geq 0$ can be any nonnegative real number.

2.9.2 Uniqueness Theorems

As usual in mathematics, $C^k(D, \mathbb{R}^n)$ denotes the space of k times continuously differentiable functions from domain D to $\mathbb{R}^n$. The Euclidean norm of a vector $v = (v_1, \ldots, v_n)$ is denoted by $\|v\| = \sqrt{\sum_{i=1}^n v_i^2}$.

If (the right-hand side of) the differential equation is continuously differentiable, then the Picard-Lindelöf theorem gives a stronger result than Peano's theorem. It shows that the solution is unique. For this, recall that a function $f : D \to \mathbb{R}^n$ with $D \subseteq \mathbb{R} \times \mathbb{R}^n$ is called *Lipschitz continuous* with respect to y iff there is an $L \in \mathbb{R}$ such that for all $(t, y), (t, \bar{y}) \in D$,

$$\|f(t, y) - f(t, \bar{y})\| \leq L\|y - \bar{y}\|.$$

If the partial derivative $\frac{\partial f(t,y)}{\partial y}$ exists and is bounded on D, then f is Lipschitz continuous with $L = \max_{(t,y)\in D} \|\frac{\partial f(t,y)}{\partial y}\|$ by the mean value theorem. Similarly, f is *locally Lipschitz continuous* iff for each $(t, y) \in D$, there is a neighborhood in which f is Lipschitz continuous. In particular, if f is continuously differentiable, i.e. $f \in C^1(D, \mathbb{R}^n)$, then f is locally Lipschitz continuous.

The Picard-Lindelöf theorem [5], which is also known as the Cauchy-Lipschitz theorem, guarantees existence and uniqueness of solutions (except, of course, that the restriction of any solution to a sub-interval is again a solution). A proof can be found in the literature [10, Theorem 10.VI]

Theorem 2.2 (Picard-Lindelöf uniqueness theorem). *Let $f : D \to \mathbb{R}^n$ be a continuous function on an open, connected domain $D \subseteq \mathbb{R} \times \mathbb{R}^n$ that is locally Lipschitz continuous with respect to y (e.g., $f \in C^1(D, \mathbb{R}^n)$). Then the initial value problem (2.1) with $(t_0, y_0) \in D$ has a unique solution.*

The Picard-Lindelöf theorem does not indicate the duration of the solution. It only shows that the solution is unique on a nonempty open interval. Under the assumptions of the Picard-Lindelöf theorem, every solution can be extended to a solution of maximal duration arbitrarily close to the boundary of D by Peano's theorem.

Example 2.11 (Quadratic). The initial value problem

$$\begin{pmatrix} y' = y^2 \\ y(0) = 1 \end{pmatrix}$$

has the unique solution $y(t) = \frac{1}{1-t}$ of maximal duration on the domain $t < 1$. It cannot be extended to include its singularity at $t = 1$, though, but can get arbitrarily close. At the singularity it converges to the boundary $\pm\infty$ of the domain $\mathbb{R}$.

The following global uniqueness theorem shows a stronger property of a global solution on $[0,a]$ when the domain is a global stripe $[0,a] \times \mathbb{R}^n$. It is a corollary to Theorems 2.1 and 2.2, but used prominently in the proof of Theorem 2.2, and is of independent interest. A direct proof of the following global version of the Picard-Lindelöf theorem can be found in the literature [10, Proposition 10.VII].

Corollary 2.1 (Global uniqueness theorem of Picard-Lindelöf). *Let $f : [t_0,a] \times \mathbb{R}^n \to \mathbb{R}^n$ be a continuous function that is Lipschitz continuous with respect to y. Then, there is a unique solution of the initial value problem (2.1) on $[t_0,a]$.*

As Example 2.11 illustrates, local Lipschitz continuity is not enough to guarantee the existence of a global solution that Corollary 2.1 concludes from global Lipschitz continuity.

2.9.3 Linear Differential Equations with Constant Coefficients

For the common class of linear differential equation systems with constant coefficients there is a well-established theory for obtaining closed-form solutions of initial value problems using classical techniques from linear algebra.

Proposition 2.1 (Linear differential equations with constant coefficients). *For a constant matrix $A \in \mathbb{R}^{n\times n}$, the initial value problem*

$$\begin{pmatrix} y'(t) = Ay(t) + b(t) \\ y(\tau) = \eta \end{pmatrix} \tag{2.10}$$

has the (unique) solution

$$y(t) = \mathrm{e}^{A(t-\tau)}\eta + \int_\tau^t \mathrm{e}^{A(t-s)} b(s)\, ds$$

where exponentiation of matrices is defined by the usual power series (generalized to matrices):

$$\mathrm{e}^{At} = \sum_{n=0}^{\infty} \frac{1}{n!} A^n t^n$$

A proof, more details, and generalizations can be found in the literature [10, §18.VI]. If the matrix A is *nilpotent*, i.e., $A^n = 0$ for some $n \in \mathbb{N}$, and the terms $b(t)$ are polynomials in t, then the solution of the initial value problem is a polynomial function, because the exponential series stops at A^n and is then a finite polynomial in t

$$\mathrm{e}^{At} = \sum_{k=0}^{\infty} \frac{1}{k!} A^k t^k = \sum_{k=0}^{n-1} \frac{1}{k!} A^k t^k$$

Since products and sums of polynomials are polynomials (polynomials form what is known as an algebra [1]) and polynomials in the variable t are closed under integration (meaning integrating a univariate polynomial in the variable t will yield another such polynomial), the solution identified in Proposition 2.1 is a polynomial. Furthermore, this solution is unique by Theorem 2.2. Such polynomials are especially useful for formal verification, because the resulting arithmetic is decidable. But the assumptions needed to guarantee such simple solutions are quite strong.

Example 2.12 (Accelerated motion in a straight line). In the initial value problem from Example 2.4 on p. 34, we guessed the solution of the differential equation system and then checked that it is the right solution by inserting it into the differential equations. But how do we compute the solution in the first place without having to guess? The differential equations $x' = v, v' = a$ from (2.2) are linear with constant coefficients. In vectorial notation where we denote $y(t) := (x(t), v(t))$, the vectorial equivalent (2.3) of (2.2) can be rewritten in explicit linear form (2.10) as follows:

$$\left(\begin{array}{l} y'(t) = \begin{pmatrix} x \\ v \end{pmatrix}' (t) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x(t) \\ v(t) \end{pmatrix} + \begin{pmatrix} 0 \\ a \end{pmatrix} =: Ay(t) + b(t) \\ y(0) = \begin{pmatrix} x \\ v \end{pmatrix} (0) = \begin{pmatrix} x_0 \\ v_0 \end{pmatrix} =: \eta \end{array} \right)$$

This linear differential equation system has the form of Proposition 2.1 with a constant coefficient matrix A. First, we compute the exponential series for the matrix A, which terminates quickly because $A^2 = 0$:

$$\begin{aligned} \mathrm{e}^{At} &= \sum_{n=0}^{\infty} \frac{1}{n!} A^n t^n = A^0 + At + \frac{1}{2!} \underbrace{A^2}_{0} t^2 + \underbrace{A^2}_{0} \sum_{n=3}^{\infty} \frac{1}{n!} A^{n-2} t^n \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} t = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix} \end{aligned}$$

Now Proposition 2.1 can be used to compute a solution of this differential equation:

$$
\begin{aligned}
y(t) &= \mathrm{e}^{At}\eta + \int_0^t \mathrm{e}^{A(t-s)} b(s)\, ds \\
&= \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ v_0 \end{pmatrix} + \int_0^t \begin{pmatrix} 1 & t-s \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ a \end{pmatrix} ds \\
&= \begin{pmatrix} x_0 + v_0 t \\ v_0 \end{pmatrix} + \int_0^t \begin{pmatrix} at - as \\ a \end{pmatrix} ds \\
&= \begin{pmatrix} x_0 + v_0 t \\ v_0 \end{pmatrix} + \begin{pmatrix} \int_0^t (at - as)\, ds \\ \int_0^t a\, ds \end{pmatrix} \\
&= \begin{pmatrix} x_0 + v_0 t \\ v_0 \end{pmatrix} + \begin{pmatrix} ats - \frac{a}{2}s^2 \\ as \end{pmatrix} \Bigg|_{s=0}^{s=t} \\
&= \begin{pmatrix} x_0 + v_0 t \\ v_0 \end{pmatrix} + \begin{pmatrix} at^2 - \frac{a}{2}t^2 \\ at \end{pmatrix} - \begin{pmatrix} a \cdot 0^2 - \frac{a}{2} \cdot 0^2 \\ a \cdot 0 \end{pmatrix} \\
&= \begin{pmatrix} x_0 + v_0 t + \frac{a}{2}t^2 \\ v_0 + at \end{pmatrix}
\end{aligned}
$$

The last equation is exactly the solution we guessed and checked in Example 2.4. Now we have computed it constructively. An alternative way of computing solutions of differential equations is by proof [7].

2.9.4 Continuation and Continuous Dependency

Occasionally it is helpful to know that solutions of differential equations can be continued by concatenation toward a maximal interval of existence. The following result is a componentwise generalization of a classical result [10, Proposition 6.VI] to vectorial differential equations and can be used to extend solutions.

Proposition 2.2 (Continuation of solutions). *Let $f : D \to \mathbb{R}^n$ be a continuous function on the open, connected domain $D \subseteq \mathbb{R} \times \mathbb{R}^n$. If φ is a solution of differential equation $y' = f(t,y)$ on $[0,b)$ whose image $\varphi([0,b))$ lies within a compact set $A \subseteq D$, then φ can be continued to a solution on $[0,b]$. Furthermore, if φ_1 is a solution of differential equation $y' = f(t,y)$ on $[0,b]$ and φ_2 is a solution of $y' = f(t,y)$ on $[b,c]$ with $\varphi_1(b) = \varphi_2(b)$, then their concatenation*

$$
\varphi(t) := \begin{cases} \varphi_1(t) & \text{for } 0 \le t \le b \\ \varphi_2(t) & \text{for } b < t \le c \end{cases}
$$

is a solution on $[0,c]$.

The solution of a Lipschitz continuous initial value problem depends continuously on the initial values and permits error estimation from the Lipschitz constants. A proof and generalizations are elsewhere [10, Proposition 12.V].

Proposition 2.3 (Lipschitz estimation). *Let $f : D \to \mathbb{R}^n$ be a continuous function that is Lipschitz continuous with Lipschitz constant L with respect to y on the open, connected domain $D \subseteq J \times \mathbb{R}^n$ with an interval J. Let y be a solution on J of the initial value problem $y'(t) = f(t, y(t)), y(0) = y_0$ and z an approximate solution in the sense that*

$$\|z(0) - y(0)\| \leq \gamma, \quad \|z'(t) - f(t, z(t))\| \leq \delta$$

then for all $t \in J$:

$$\|y(t) - z(t)\| \leq \gamma e^{L|t|} + \frac{\delta}{L}(e^{L|t|} - 1)$$

Here J is any interval with $0 \in J$ and $graph(y), graph(z) \subseteq D$.

Exercises

2.1. Suppose $\omega(x) = 7$, then explain why the value of $4 + x \cdot 2$ in ω is $\omega[\![4 + x \cdot 2]\!] = 18$. What is the value of the same term $4 + x \cdot 2$ in a state ν with $\nu(x) = -4$? What is the value of the term $4 + x \cdot 2 + x \cdot x \cdot x$ in the same state ν? How does its value change in a state where $\nu(x) = -4$ but also $\nu(y) = 7$? Suppose $\omega(x) = 7$ and $\omega(y) = -1$, explain why $x \cdot 2 + y \cdot y$ evaluates to $\omega[\![x \cdot 2 + y \cdot y]\!] = 15$. What is the value of $x \cdot 2 + y \cdot y$ in a state ν with $\nu(x) = -4$ and $\nu(y) = 7$?

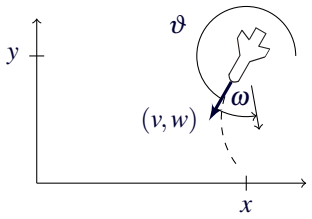
2.2. Subtraction $e - \tilde{e}$ is already implicitly available as a term, because it is definable via $e + (-1) \cdot \tilde{e}$. In practice, we can, thus, pretend $e - \tilde{e}$ is in the syntax, while theoretical investigations can ignore $e - \tilde{e}$ as it is not an official part of the syntax. What about negation $-e$? Is negation implicitly already available as well? What about division $e/\tilde{e}$ and powers $e^{\tilde{e}}$?

2.3 (Speeding). Consider a car that is driving on a road with a speed limit of either 35 mph or 50 km/h when a deer darts onto the road at a distance in front of the car that is just enough for the car to stop when driving at the speed limit. Suppose the car was speeding at 45 mph or 70 km/h, instead. How fast is the car still going when it meets the surprised deer? You may assume brakes of effective deceleration $a = -6m/s^2$, which is typical for some road conditions. How does the answer change when the driver needs a reaction time of 2 seconds?

2.4 (Changing accelerations). Some settings of idealized physics benefit from considering not just position x, its rate of change velocity v, and its rate of change acceleration a, but also the continuous rate of change of the acceleration a, which is called jolt or jerk j. Jolt may happen, for example, when changing gears abruptly or when not releasing the brakes of a car when it is about to come to a standstill. Solve the resulting differential equation of accelerated acceleration in a straight line:

$$x' = v, v' = a, a' = j$$

2.5 (Robot moving along a planar circular curve). This exercise develops differential equations for the continuous dynamics of a robot that is moving in the two-dimensional plane. Consider a robot at a point with coordinates (x,y) that is facing in direction (v,w). While the robot is moving along the dashed curve, this direction (v,w) is simultaneously rotating with angular velocity ω.



Develop a differential equation system describing how the position and direction of the robot change over time. Build your way up to that differential equation by first considering just the rotation of (v,w), then considering the motion of (x,y) in a fixed direction (v,w), and then putting both types of behavior together. Can you subsequently generalize the dynamics to also include an acceleration of the linear ground speed when the robot is speeding up?

2.6. A number of differential equations and some suggested solutions are listed in the following table. Are these correct solutions? Are these all solutions? Are there other solutions? In what ways are the solutions characteristically more complicated than their differential equations?

ODE	Solution
$x' = 1, x(0) = x_0$	$x(t) = x_0 + t$
$x' = 5, x(0) = x_0$	$x(t) = x_0 + 5t$
$x' = x, x(0) = x_0$	$x(t) = x_0 \mathrm{e}^t$
$x' = x^2, x(0) = x_0$	$x(t) = \frac{x_0}{1 - t x_0}$
$x' = \frac{1}{x}, x(0) = 1$	$x(t) = \sqrt{1 + 2t}$
$y'(x) = -2xy, y(0) = 1$	$y(x) = \mathrm{e}^{-x^2}$
$x'(t) = tx, x(0) = x_0$	$x(t) = x_0 \mathrm{e}^{\frac{t^2}{2}}$
$x' = \sqrt{x}, x(0) = x_0$	$x(t) = \frac{t^2}{4} \pm t\sqrt{x_0} + x_0$
$x' = y, y' = -x, x(0) = 0, y(0) = 1$	$x(t) = \sin t, y(t) = \cos t$
$x' = 1 + x^2, x(0) = 0$	$x(t) = \tan t$
$x'(t) = \frac{2}{t^3} x(t)$	$x(t) = \mathrm{e}^{-\frac{1}{t^2}}$ non-analytic
$x'(t) = \mathrm{e}^{t^2}$	no elementary closed form

2.7 (**). Would the differential equation semantics defined in Definition 2.6 change when we only require the differential equation to be respected at all times $\zeta \in (0,r)$ in an open interval rather than at all times $\zeta \in [0,r]$ in a closed interval?

2.8. Review the theory of ordinary differential equations. Investigate which theorems from this chapter's appendix apply to the example differential equations given in this chapter.

2.9 (Valid quantified formulas). Using the semantics of quantifiers, show that the following first-order logic formulas are valid, i.e., true in all states:

$$(\forall x\, p(x)) \rightarrow (\exists x\, p(x))$$
$$(\forall x\, p(x)) \rightarrow p(e)$$
$$\forall x\,(p(x) \rightarrow q(x)) \rightarrow (\forall x\, p(x) \rightarrow \forall x\, q(x))$$

In the second formula, e is any term and $p(e)$ should be understood here as resulting from the formula $p(x)$ by replacing all (free) occurrences of variable x by term e. An occurrence of x in $p(x)$ within the scope of another quantifier for x is not free but bound, so will not be substituted. Some care is needed to avoid capture, i.e., that x does not occur in the scope of a quantifier binding a variable of e, because replacing x with e would then bind a free variable of e.

2.10 (* Two-valued semantics). The semantics of terms is defined by a real-valued mapping $[\![e]\!] : \mathscr{S} \rightarrow \mathbb{R}$ in Expedition 2.2. The semantics of formulas was given in Definition 2.5 by defining the set of states $[\![P]\!]$ in which formula P is true. Give an equivalent definition of the semantics of first-order formulas using a function $[\![P]\!]_{\mathbb{B}} : \mathscr{S} \rightarrow \{true, false\}$ that defines the truth-value $[\![P]\!]_{\mathbb{B}}(\omega)$ of formula P in each state ω. Prove equivalence of the semantics by showing that the new truth-value semantics evaluates to the truth-value *true* if and only if the formula P is true at ω:

$$[\![P]\!]_{\mathbb{B}}(\omega) = true \quad \text{iff} \quad \omega \in [\![P]\!]$$

2.11 (Term interpreter). In a programming language of your choosing, fix a recursive data structure for terms from Definition 2.2 and fix some finite representation for states where all variables have rational values instead of reals. Write a term interpreter as a computer program that, given a state ω and a term e, computes the value of $\omega[\![e]\!]$ by implementing Definition 2.4 as a recursive function. Write a similar interpreter for first-order formulas from Definition 2.3 that, given a state ω and a formula P, reports "yes" if and only if $\omega \in [\![P]\!]$. Which cases are problematic?

2.12 (Set-valued semantics).** There are at least two styles of giving a meaning to a logical formula. One way is to inductively define a satisfaction relation $\models$ that holds between a state ω and a dL formula P, written $\omega \models P$, whenever the formula P is true in the state ω. Its definition includes, among other cases, the following notational variant of Definition 2.5:

$$\omega \models e \geq \tilde{e} \text{ iff } \omega[\![e]\!] \geq \omega[\![\tilde{e}]\!]$$
$$\omega \models P \wedge Q \text{ iff } \omega \models P \text{ and } \omega \models Q$$

The other way is to inductively define, for each dL formula P, the set of states, written $[\![P]\!]$, in which P is true. Its definition will include, among other cases, the

following:

$$\begin{array}{rcl} [\![e \geq \tilde{e}]\!] & = & \{\omega : \omega[\![e]\!] \geq \omega[\![\tilde{e}]\!]\} \\ [\![P \wedge Q]\!] & = & [\![P]\!] \cap [\![Q]\!] \end{array}$$

Complete both styles of defining the semantics and prove that they are equivalent. That is, $\omega \models P$ iff $\omega \in [\![P]\!]$ for all states ω and all first-order formulas P.

Such a proof can be conducted by induction on the structure of P. That is, consider each case, say $P \wedge Q$, and prove $\omega \models P \wedge Q$ iff $\omega \in [\![P \wedge Q]\!]$ from the inductive hypothesis that the conjecture already holds for the smaller subformulas:

$$\omega \models P \text{ iff } \omega \in [\![P]\!]$$
$$\omega \models Q \text{ iff } \omega \in [\![Q]\!]$$

2.13. Explain which formulas you expect to be particularly common as evolution domain constraints in cyber-physical systems.

References

[1] Nicolas Bourbaki. *Algebra I: Chapters 1–3*. Elements of mathematics. Berlin: Springer, 1989.

[2] Kenneth Eriksson, Donald Estep, Peter Hansbo, and Claes Johnson. *Computational Differential Equations*. Cambridge: Cambridge University Press, 1996.

[3] Leonhard Euler. *Institutionum calculi integralis*. St Petersburg: Petropoli, 1768.

[4] Philip Hartman. *Ordinary Differential Equations*. Hoboken: John Wiley, 1964.

[5] M. Ernst Lindelöf. Sur l'application de la méthode des approximations successives aux équations différentielles ordinaires du premier ordre. *Comptes rendus hebdomadaires des séances de l'Académie des sciences* **114** (1894), 454–457.

[6] Giuseppe Peano. Demonstration de l'intégrabilité des équations différentielles ordinaires. *Mathematische Annalen* **37**(2) (1890), 182–228. DOI: 10.1007/BF01200235.

[7] André Platzer. A complete uniform substitution calculus for differential dynamic logic. *J. Autom. Reas.* **59**(2) (2017), 219–265. DOI: 10.1007/s10817-016-9385-1.

[8] William T. Reid. *Ordinary Differential Equations*. Hoboken: John Wiley, 1971.

[9] Gerald Teschl. *Ordinary Differential Equations and Dynamical Systems*. Providence: AMS, 2012.

[10] Wolfgang Walter. *Ordinary Differential Equations*. Berlin: Springer, 1998. DOI: 10.1007/978-1-4612-0601-9.