15-424: Foundations of Cyber-Physical Systems

Lecture Notes on Differential Equations & Domains

André Platzer

Carnegie Mellon University
Lecture 2

1. Introduction

In Lecture 1, we have learned about the characteristic features of *cyber-physical systems* (CPS): they combine cyber capabilities (computation and/or communication as well as control) with physical capabilities (motion or other physical processes). Cars, aircraft, and robots are prime examples, because they move physically in space in a way that is determined by discrete computerized control algorithms that are adjusting the actuators (e.g., brakes) based on sensor readings of the physical state. Designing these algorithms to control CPSs is challenging due to their tight coupling with physical behavior. At the same time, it is vital that these algorithms be correct, since we rely on CPSs for safety-critical tasks like keeping aircraft from colliding.

Note 1 (Significance of CPS safety)**.** *How can we provide people with cyber-physical systems they can bet their lives on?* *– Jeannette Wing*

Since CPS combine cyber and physical capabilities, we need to understand both to understand CPS. It is not enough to understand both capabilities only in isolation, though, because we also need to understand how the cyber and the physics work together, i.e. what happens when they interface and interact, because this is what CPSs are all about.

Note 2 (CPS)**.** *Cyber-physical systems combine cyber capabilities with physical capabilities to solve problems that neither part could solve alone.*

You already have experience with models of computation and algorithms for the cyber part of CPS, because you have seen the use of programming languages for computer programming in previous courses. In CPS, we do not program computers, but

rather program CPSs instead. Hence, we program computers that interact with physics to achieve their goals. In this lecture, 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, that of ordinary differential equations.

While this lecture covers the most important parts of differential equations, it is not to be understood as doing complete diligence to the fascinating area of ordinary differential equations. The crucial part about differential equations that you need to get started with the course is an intuition about differential equations as well as an understanding of their precise meaning. This will be developed in today's lecture. Subsequently, we will be coming back to the topic of differential equations for a deeper understanding of differential equations and their proof principles a number of times at a later part of the course. The other important aspect that today's lecture develops is *first-order logic of real arithmetic* for the purpose of representing domains and domain constraints of differential equations.

You are advised to refer back to your differential equations course and follow the supplementary information[1] available on the course web page as needed during this course to refresh your knowledge of differential equations. We refer, e.g., to the book by Walter [Wal98] for details and proofs about differential equations. There is a lot of further background on differential equations in the literature [Har64, Rei71, EEHJ96].

These lecture notes are based on material on cyber-physical systems, hybrid programs, and logic [Pla12, Pla10, Pla08, Pla07]. Cyber-physical systems play an important role in numerous domains [Pre07, LS10, Alu11, LSC+12] with applications in cars [DGV96], aircraft [TPS98], robots [PKV09], and power plants [FKV04], chemical processes [RKR10, KGDB10], medical models [GBF+11, KAS+11, LSC+12], and even an importance for understanding biological systems [Tiw11].

More information about CPS can be found in [Pla10, Chapter 1]. Differential equations and domains are described in [Pla10, Chapter 2.2,2.3] in more detail.

The most important learning goals of this lecture 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 for 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 between syntax (which is notation) and semantics (what carries meaning), a core principle for computer science that continues to be crucial for CPS.

[1] http://symbolaris.com/course/fcps14-resources.html

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. 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 concisely describes 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. 1 shows the respective directions in which the system evolves by a vector at each point and illustrates one solution as a curve in space which follows those vectors everywhere. Of course, the figure would be rather cluttered if we would literally try to indicate the vector at each and every point, of which there are uncountably infinitely many. But this is a shortcoming only of our illustration. Differential equations actually define such a vector for the direction of evolution at every point in space.

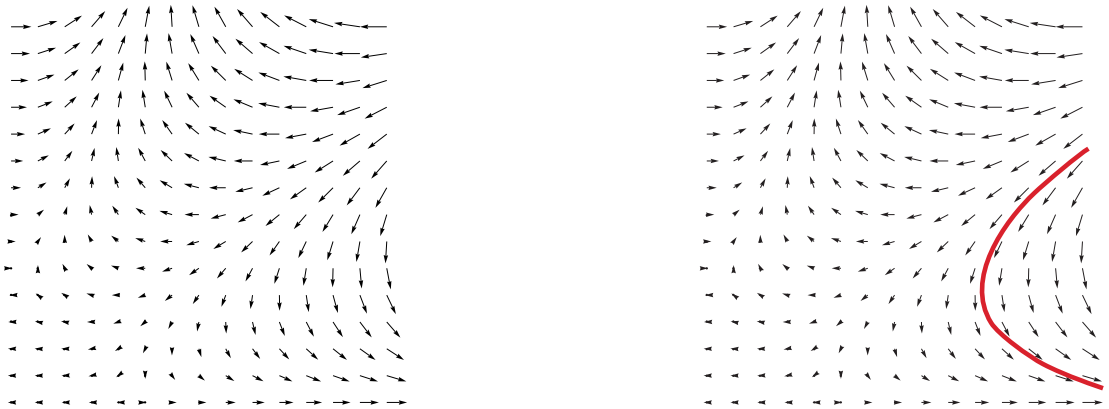

Figure 1: Vector field **(left)** and vector field with one solution of a differential equation **(right)**

As an example, suppose we have a car whose position is denoted by x. How the value of variable x changes over time depends on how fast the car is driving. Let v denote the velocity of the car. Since v is the velocity of the car, its position x changes such that its derivative x' is v, which we write by the differential equation $x' = v$. This differential equation is supposed to mean that the time-derivative of the position x is 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. If $v > 0$, then the position x keeps on increasing over time. How fast x increases depends on the value of v, bigger v give quicker changes in x.

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, so $v' = a$. Overall, the car then follows the differential equation (system):[2]

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

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 has a vector field where all vectors point into direction a. And that the system is always supposed to follow exactly in the direction of those vectors at every point. What does this mean exactly? How can we understand it doing that at all of the infinitely many points?

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.

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

For a number of different time discretization steps $\Delta \in \{4, 2, 1, \frac{1}{2}\}$, Fig. 2 illustrates what a 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. The true solution of the differential equation should, however, also have respected the direction that the differential equation prescribes at all the other uncountably infinitely time points in between. Because the differential equation is well-behaved, these discretizations still approach the true continuous solution $x(t) = e^{\frac{t}{4}}$ as Δ gets smaller.

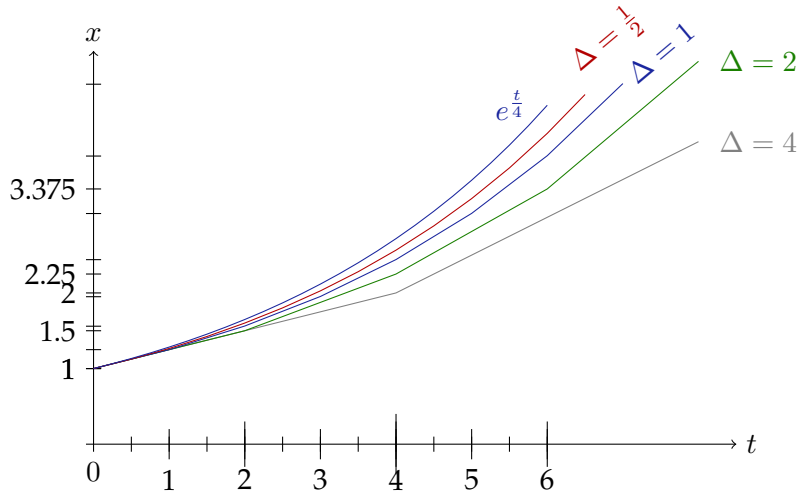

Figure 2: Discretizations of differential equations with discretization time step Δ

[2] 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 suppress the argument t for time and just write $x' = v, v' = a$ instead.

3. The Meaning of Differential Equations

We can relate an intuitive concept to how differential equations describe the direction of the evolution of a system as a vector field Fig. 1. But what exactly is a vector field? What does it mean to describe directions of evolutions at every point in space? 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.

Note 3 (Importance of meaning)**.** *The physical impacts of CPSs do not leave much room for failure, so we immediately want to get into the mood of consistently studying the behavior and exact meaning of all relevant aspects of 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 current state y. A solution is a differentiable function Y which satisfies this equation when substituted into the differential equation, i.e., when substituting $Y(t)$ for y and the derivative $Y'(t)$ of Y at t for $y'(t)$. In the next lecture, we will study an elegant definition of solution of differential equations that is well-attuned with the concepts in this class. But first, we consider the (equivalent) classical definition of solution.

Definition 1 (Ordinary differential equation)**.** Let $f : D \to \mathbb{R}^n$ be a function on a domain $D \subseteq \mathbb{R} \times \mathbb{R}^n$. The function $Y : I \to \mathbb{R}^n$ is a *solution* on the interval $I \subseteq \mathbb{R}$ of the *initial value problem*

$$\left[\begin{array}{ll} y'(t) = & f(t, y) \\ y(t_0) = & y_0 \end{array} \right] \tag{1}$$

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

1. $(t, Y(t)) \in D$,
2. time-derivative $Y'(t)$ exists and is $Y'(t) = f(t, Y(t))$,
3. $Y(t_0) = y_0$, especially $t_0 \in I$.

If $f : D \to \mathbb{R}^n$ is continuous, then it is easy to see that $Y : I \to \mathbb{R}^n$ is continuously differentiable, because its derivative $Y'(t)$ is $f(t, Y(t))$. Similarly if f is k-times continuously differentiable then Y is $k + 1$-times continuously differentiable. The definition is accordingly for higher-order differential equations, i.e., differential equations involving higher-order derivatives $y^{(n)}(t)$ 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. 1, where, at each point, the vector shows in which direction the solution evolves. At every point, the vector would

correspond 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. 1) locally follows the direction indicated by the vector of the right-hand side of the differential equation. There are many solutions of the differential equation corresponding to the vector field illustrated in Fig. 1. For the particular initial value problem, however, a solution also has to start at the prescribed position y_0 at time t_0 and then follow the differential equations or vector field from this point. In general, there could still be multiple solutions for the same initial value problem, but not for well-behaved differential equations (Appendix B).

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.

Example 2 (A constant differential equation). Some differential equations are easy to solve. The initial value problem

$$\left[\begin{array}{rl} x'(t) = & 5 \\ x(0) = & 2 \end{array}\right]$$

describes that x initially starts at 3 and always changes at the rate 5. It has the solution $x(t) = 5t + 2$. How could 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:

$$\left[\begin{array}{rl} (x(t))' = & (5t+2)' = 5 \\ x(0) = & 5 \cdot 0 + 2 = 2 \end{array}\right]$$

Example 3 (A linear differential equation). Consider the initial value problem

$$\left[\begin{array}{rl} x'(t) = & -2x(t) \\ x(1) = & 5 \end{array}\right]$$

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 as $x(t)$ gets bigger. This problem has the solution $x(t) = 5e^{-2(t-1)}$. The test, again, is to insert the solution into the (differential) equations of the initial value problems and check:

$$\left[\begin{array}{rl} (5e^{-2(t-1)})' = & -10e^{-2(t-1)} = -2x(t) \\ x(1) = & 5e^{-2(1-1)} = 5 \end{array}\right]$$

Example 4 (Another linear differential equation). The initial value problem

$$\left[\begin{array}{rl} x'(t) = & \frac{1}{4}x(t) \\ x(0) = & 1 \end{array}\right]$$

shown with different discretizations in Fig. 2 has the true continuous solution $x(t) = e^{\frac{t}{4}}$, which can be checked in the same way as for the previous example:

$$\left[\begin{array}{rl} (e^{\frac{t}{4}})' & = e^{\frac{t}{4}}(\frac{t}{4})' = e^{\frac{t}{4}}\frac{1}{4} = \frac{1}{4}x(t) \\ e^{\frac{0}{4}} & = 1 \end{array}\right]$$

Example 5 (Accelerated motion on a straight line). Consider the following important differential equation system $x' = v, v' = a$ and the initial value problem

$$\left[\begin{array}{rl} x'(t) = & v(t) \\ v'(t) = & a \\ x(0) = & x_0 \\ v(0) = & v_0 \end{array}\right]$$

This differential equation represents 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. Note that 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. In vectorial notation, the initial value problem with this differential equation system corresponds to a vectorial system when we denote $y(t) := (x(t), v(t))$, i.e., with dimension $n = 2$ in Def. 1:

$$\left[\begin{array}{rl} y'(t) = \begin{pmatrix} x \\ v \end{pmatrix}'(t) = & \begin{pmatrix} v(t) \\ a \end{pmatrix} \\ y(0) = \begin{pmatrix} x \\ v \end{pmatrix}(0) = & \begin{pmatrix} x_0 \\ v_0 \end{pmatrix} \end{array}\right]$$

The solution of this initial value problem is

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

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

$$\left[\begin{array}{rl} (\frac{a}{2}t^2 + v_0 t + x_0)' = & 2\frac{a}{2}t + v_0 = v(t) \\ (at + v_0)' = & a \\ x(0) = & \frac{a}{2}0^2 + v_0 0 + x_0 = x_0 \\ v(0) = & a0 + v_0 = v_0 \end{array}\right]$$

Example 6 (A two dimensional linear differential equation). Consider the differential equation system $x' = y, y' = -x$ and the initial value problem

$$\left[\begin{array}{rl} x'(t) = & y(t) \\ y'(t) = & -x(t) \\ x(0) = & 1 \\ y(0) = & 1 \end{array}\right]$$

in which the rate of change of $x(t)$ gets bigger as $y(t)$ gets bigger but, simultaneously, the rate of change of $y(t)$ is $-x(t)$ so it gets smaller as $x(t)$ gets bigger and vice versa. This differential equation describes a rotational effect (Fig. 3) with solution of this initial value being

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

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

$$\begin{bmatrix} (\cos(t) + \sin(t))' = & -\sin(t) + \cos(t) = y(t) \\ (\cos(t) - \sin(t))' = & -\sin(t) - \cos(t) = -x(t) \\ x(0) = & \cos(0) + \sin(0) = 1 \\ y(0) = & \cos(0) - \sin(0) = 1 \end{bmatrix}$$

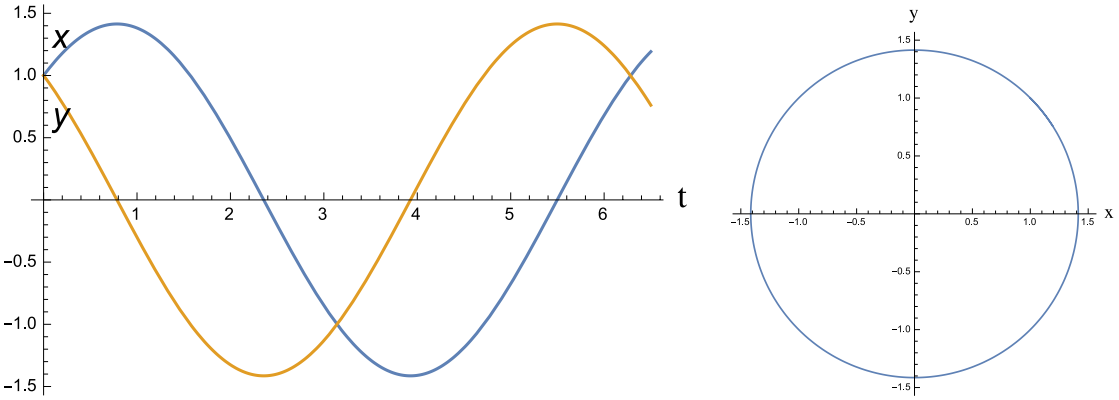

Figure 3: A solution of the rotational differential equations x and y over time t **(left)** and in phase space with coordinates y over x **(right)**

Example 7 (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

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

The solution shown in Fig. 4(left) illustrates that the system stays bounded but oscillates increasingly fast. In this case, the solution is

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

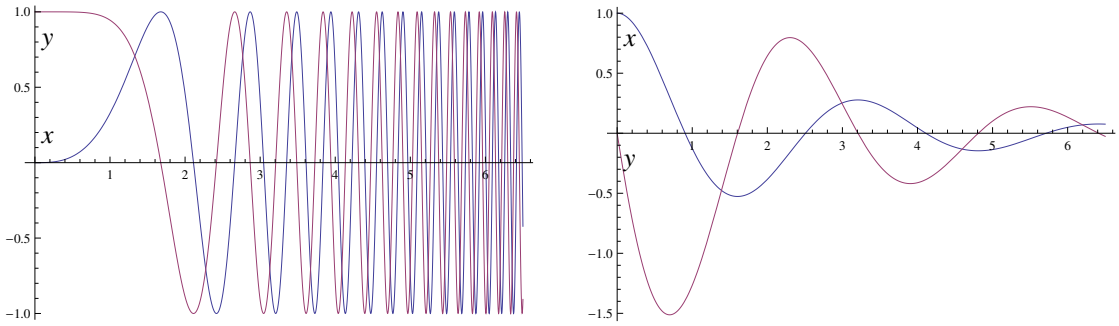

Figure 4: A solution of the time square oscillator **(left)** and of the damped oscillator **(right)** up to time 6.5

Note that there is no need to mention time variable t itself directly as 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):

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

Example 8 (Damped oscillator). Consider the linear differential equation $x' = y, y' = -4x - 0.8y$ and the initial value problem

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

The solution shown in Fig. 4(right) illustrates that the dynamical system decays over time. In this case, the explicit global solution representing the dynamical system is more difficult.

Note 5 (Descriptive power of differential equations)**.** *As a 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.*

5. Domains of Differential Equations

Now we understand exactly 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 it only provides input into physics by way of its 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 that physics is based on. 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. At which 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 could interface in more than one way. Physics might evolve and the cyber elements interrupt to inspect measurements about the state of the system periodically to decide what to do next. Or the physics might trigger certain conditions or events that cause cyber elements to compute their respective responses to these events. Another way to look at that is that a differential equation that a system follows forever without further intervention by anything would not describe a particularly well-controlled system. All those ways have in common that our model of physics needs to specify when it stops evolving to give cyber a chance to perform its task.

This information is what is a 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.

Note 6 (Evolution domain constraints). *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. That is, 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 for the kind of physics that gives the cyber elements a chance to act at the latest at time ε. 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 become negative while following the differential equation $x' = v, v' = a$, then the system stops before that happens.

In the left two scenarios illustrated in Fig. 5, the system starts at time 0 inside the evolution domain Q that is depicted as a shaded green region in Fig. 5. 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 time r_0 (left) or r (middle, right) respectively.

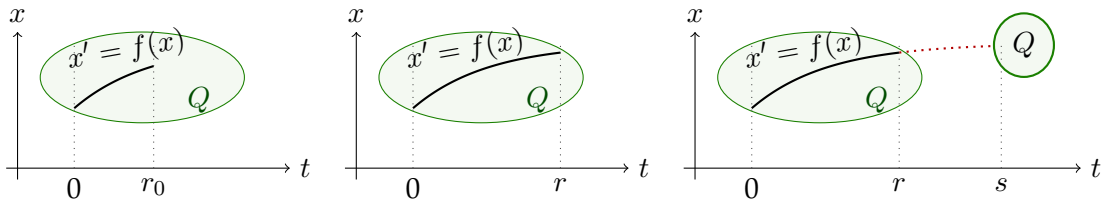

Figure 5: System $x' = f(x) \,\&\, Q$ follows the differential equation $x' = f(x)$ for any duration r but cannot leave the (shaded) evolution domain Q.

In contrast, consider the scenario shown on the right of Fig. 5. The system is *not* allowed to evolve until time s, because—even if the system were 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 allowed. Consequently, the continuous evolution on the right of Fig. 5 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. Its most critical bit are which points satisfy Q and which ones doesn't, which is what logic is good at making precise.

6. Continuous Programs: Syntax

After these preparations for understanding differential equations and domains, we start developing a programming language for cyber-physical systems. Ultimately, this programming language of *hybrid programs* will contain more features than just differential equations. But this most crucial feature is what we start with in this lecture. This course develops this programming language and its understanding and its analysis in layers one after the other. We will discuss the principles behind its design in the next lecture in more details and just start with continuous programs for now.

Continuous Programs. The first element of the syntax of hybrid programs are purely continuous programs.

Note 7. *Layer 1 of* hybrid programs *(HPs) are* continuous programs. *These are defined by the following grammar (α is a HP, x is a variable, e is any term possibly containing x, and Q a formula of first-order logic of real arithmetic):*

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

This means that a hybrid program α consists of a single statement of the form $x' = e \,\&\, Q$. In later lectures, we will add more statements to hybrid programs, but focus on differential equations for now. The formula Q is called *evolution domain constraint* of the *continuous evolution* $x' = e \,\&\, Q$. What form Q can take will be defined below. But it has to enable an unambiguous definition of which points satisfy Q and which points do not. 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:

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

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

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

which corresponds to choosing $true$ for Q, since the formula $true$ is true everywhere and, thus, actually imposes no condition on the state whatsoever.

Terms. A rigorous definition of the syntax of hybrid programs also depends on defining what a term e is and what a formula Q of first-order logic of real arithmetic is.

Definition 9 (Terms)**.** A *term* e is a polynomial term defined by the grammar (where $e, \tilde{e}$ are terms, x a variable, and c a rational number constant):

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

This means that a term e (or a term $\tilde{e}$)[3] is either a variable x, or a rational number constant $c \in \mathbb{Q}$ such as 0 or 1 or $\frac{5}{7}$, or a sum of terms $e, \tilde{e}$, or a product of terms $e, \tilde{e}$, which are again built of this form recursively. 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

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

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

We use the slightly more verbose form just to emphasize 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.

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.

First-order Formulas. The formulas of first-order logic of real arithmetic are defined as usual in first-order logic, except that it 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 ($\rightarrow$), biimplication or equivalence ($\leftrightarrow$), as well as quantifiers for all ($\forall$) and exists ($\exists$).

Definition 10 (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 a variable):

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

The usual abbreviations are allowed, such as $e \leq \tilde{e}$ for $\tilde{e} \geq e$ and $e < \tilde{e}$ for $\neg(e \geq \tilde{e})$.

7. Continuous Programs: Semantics

Note 10 (Syntax vs. Semantics)**.** *Syntax just defines arbitrary notation. Its meaning is defined by the semantics.*

Terms. The meaning of a continuous evolution $x' = e \,\&\, Q$ depends on understanding the meaning of terms e. A term e is a syntactic expression. Its value depends on the interpretation of the variables appearing in the term e. What values those variables have changes depending on the state of the CPS. A *state* ω is a mapping from variables to real numbers. The set of states is denoted $\mathcal{S}$.

Definition 11 (Semantics of terms)**.** The *value of term* e in state $\omega \in \mathcal{S}$ is a real number denoted $[\![e]\!]\omega$ and is defined by induction on the structure of term e:

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

That is, the value of a variable x in state ω is defined by the state ω, which is a mapping from variables to real numbers. And the value of a 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. Likewise, the value of a 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 (Def. 9) has been given a semantics. That is, the semantics of a term is a mapping from states to the real value that the term evaluates to in the respective state.

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 $[\![4 + x \cdot 2]\!]\omega = 14$. If $\nu(x) = 2$, then $[\![4 + x \cdot 2]\!]\nu = 8$. While, technically, the state is a mapping from all variables to real numbers, it turns out that the values it gives to most variables are immaterial, only the values of its free variables have any influence [Pla15]. 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.

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* 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 that the quantifiers $\forall x$ and $\exists x$ quantify over the reals. The meaning of the variables is again determined by the state of the CPS.

For the definition of the semantics, we need 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. Let $\omega_x^d \in \mathcal{S}$ denote the state that agrees with state $\omega \in \mathcal{S}$ except for the interpretation of variable x, which is changed to the value $d \in \mathbb{R}$:

$$\omega_x^d(y) = \begin{cases} d & \text{if } y \text{ is the variable } x \\ \omega(y) & \text{otherwise} \end{cases}$$

We write $\omega \in [\![F]\!]$ to indicate that F evaluates to *true* in state ω and define it as follows.

Definition 12 (First-order logic semantics)**.** The *satisfaction relation* $\omega \in [\![P]\!]$ for a first-order formula P of real arithmetic in state ω is defined inductively:

- $\omega \in [\![(e = \tilde{e})]\!]$ iff $[\![e]\!]\omega = [\![\tilde{e}]\!]\omega$
  That is, an equation is true in a state ω iff the terms on both sides evaluate to the same number.
- $\omega \in [\![(e \geq \tilde{e})]\!]$ iff $[\![e]\!]\omega \geq [\![\tilde{e}]\!]\omega$
  That is, a greater-or-equals inequality is true in a state ω iff the term on the left evaluate to a number that is greater or equal to the value of the right term.
- $\omega \in [\![\neg P]\!]$ iff $\omega \notin [\![P]\!]$, i.e. if it is not the case that $\omega \in [\![P]\!]$
  That is, a negated formula $\neg P$ is true in state ω iff the formula P itself is not true in ω.
- $\omega \in [\![P \wedge Q]\!]$ iff $\omega \in [\![P]\!]$ and $\omega \in [\![Q]\!]$
  That is, a conjunction is true in a state iff both conjuncts are true in said state.
- $\omega \in [\![P \vee Q]\!]$ iff $\omega \in [\![P]\!]$ or $\omega \in [\![Q]\!]$
  That is, a disjunction is true in a state iff either of its disjuncts is true in said state.
- $\omega \in [\![P \rightarrow Q]\!]$ iff $\omega \notin [\![P]\!]$ or $\omega \in [\![Q]\!]$
  That is, an implication is true in a state iff either its left-hand side is false or its right-hand side true in said state.
- $\omega \in [\![P \leftrightarrow Q]\!]$ iff ($\omega \in [\![P]\!]$ and $\omega \in [\![Q]\!]$) or ($\omega \notin [\![P]\!]$ or $\omega \notin [\![Q]\!]$)
  That is, a biimplication is true in a state iff both sides are true or both sides are false in said state.
- $\omega \in [\![\forall x\, P]\!]$ iff $\omega^d_x \in [\![P]\!]$ 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^d_x.
- $\omega \in [\![\exists x\, P]\!]$ iff $\omega^d_x \in [\![P]\!]$ 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^d_x.

If $\omega \in [\![P]\!]$, then we say that P is true at ω or that ω is a model of P. A formula P is *valid*, written $\models P$, iff $\omega \in [\![P]\!]$ for all states ω. A formula P is a *consequence* of a set of formulas Γ, written $\Gamma \models P$, iff, for each ω: $\omega \in [\![Q]\!]$ for all $Q \in \Gamma$ implies that $\omega \in [\![P]\!]$.

The most exciting formulas are the ones that are valid, i.e., $\models P$, 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 course. 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 today's lecture, however, it is more important which formulas are true in a given state.

With the semantics, we 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.

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

There is more than one way to define the meaning of a program, including defining a denotational semantics, an operational semantics, a structural operational semantics, an axiomatic semantics. We will be in a better position to appreciate several nuances of these aspects in later lectures. 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 like 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 that can be connected from ω by a solution of the differential equation $x' = e$ that remains entirely within the set of states where the evolution domain constraint Q holds true. Giving this a precise meaning requires going back and forth between syntax and semantics carefully.

Definition 13 (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 (or *flow*) 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 $\varphi : [0, r] \to \mathcal{S}$ such that:

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

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

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

The next lecture will introduce a notation for this and just write

$$(\omega, \nu) \in [\![x_1' = e_1, \ldots, x_n' = e_n \,\&\, Q]\!]$$

to indicate that state ν is reachable from initial state ω by the continuous program $x_1' = e_1, \ldots, x_n' = e_n \,\&\, Q$.

Observe that this definition is explicit about the fact that variables without differential equations do not change during a continuous program. The semantics of HP is *explicit change*: nothing changes unless (an assignment or) a differential equation specifies how. Also observe the explicit passing from syntax to semantics[4] by the use of the valuation function $[\![\cdot]\!]$ in Def. 13.

Finally note that for duration $r = 0$, the condition on respecting the differential equation is trivially satisfied, because Def. 13 only requires the time-derivative of the value of x_i to match with its right-hand side e_i in the open interval $(0, r)$, which, for $r = 0$, is the empty interval. Observe that this is a good choice for the semantics, because for $r = 0$, the meaning of a derivative at the only point in time 0 would not even be well-defined, so it would not be meaningful to refer to it. Consequently, the only conditions that Def. 13 imposes for duration 0 are that the initial state ω and final state ν are the same and that the evolution domain constraint Q is respected at that state: $\omega \in [\![Q]\!]$.

[4] This important aspect is often overlooked. Informally, one might say that x obeys $x' = e$, but this certainly cannot mean that the equation $x' = e$ holds true, because it is not even clear what the meaning of x' would be, nor does e have a single value, because it is a syntactic term whose value depends on the state by Def. 11. A syntactic variable x has a meaning in a state but x' does not. The semantical valuation of x along a function φ, instead, can have a well-defined derivative. This requires passing back and forth between syntax and semantics.

Note 14 (Operators and (informal) 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*
$e \geq \tilde{e}$	*equals*	*true iff value of* e *greater-or-equal to* $\tilde{e}$
$\neg\phi$	*negation / not*	*true if* ϕ *is false*
$\phi \wedge \psi$	*conjunction / and*	*true if both* ϕ *and* ψ *are true*
$\phi \vee \psi$	*disjunction / or*	*true if* ϕ *is true or if* ψ *is true*
$\phi \rightarrow \psi$	*implication / implies*	*true if* ϕ *is false or* ψ *is true*
$\phi \leftrightarrow \psi$	*bi-implication / equivalent*	*true if* ϕ *and* ψ *are both true or both false*
$\forall x\, \phi$	*universal quantifier / for all*	*true if* ϕ *is true for all values of variable* x
$\exists x\, \phi$	*existential quantifier / exists*	*true if* ϕ *is true for some values of variable* x

8. Summary

This lecture 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 Note 14.

A. Existence Theorems

For your reference, this appendix contains a short primer on some important results about differential equations [Pla10, Appendix B].

There are several classical theorems that 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 [Pea90]. A proof can be found in [Wal98, Theorem 10.IX].

Theorem 14 (Existence theorem of Peano). *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* (1) *with* $(t_0, y_0) \in D$ *has a solution. Further, every solution of* (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 D. That is, the closure of the graph of the solution, when restricted to $[0, 0] \times \mathbb{R}^n$, is not a compact subset of D. In particular, there is a global solution on the interval $[0, \infty)$ if $D = \mathbb{R}^{n+1}$ then.

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

Example 15. The initial value problem with the following continuous differential equation

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

has multiple solutions:

$$y(t) = 0$$

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

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

where $s \geq 0$ is any nonnegative real number.

B. Existence and Uniqueness Theorems

As usual, $C^k(D, \mathbb{R}^n)$ denotes the space of k times continuously differentiable functions from domain D to $\mathbb{R}^n$.

If we know that the differential equation (its right-hand side) is continuously differentiable on an open, connected domain, then the Picard-Lindelöf theorem gives a stronger result than Peano's theorem. It shows that there is a unique solution (except, of course, that the restriction of any solution to a sub-interval is again a solution). 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, for instance, $\frac{\partial f(t,y)}{\partial y}$ 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 mean value theorem. Similarly, f is *locally Lipschitz continuous* iff for each $(t, y) \in D$, there is a neighbourhood 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.

Most importantly, Picard-Lindelöf's theorem [Lin94], which is also known as the Cauchy-Lipschitz theorem, guarantees existence and uniqueness of solutions. As restrictions of solutions are always solutions, we understand uniqueness up to restrictions. A proof can be found in [Wal98, Theorem 10.VI]

Theorem 16 (Uniqueness theorem of Picard-Lindelöf)**.** *In addition to the assumptions of Theorem 14, let f be locally Lipschitz continuous with respect to y (for instance, $f \in C^1(D, \mathbb{R}^n)$ is sufficient). Then, there is a unique solution of the initial value problem (1).*

Picard-Lindelöf's theorem does not show the duration of the solution, but shows only that the solution is unique. Under the assumptions of Picard-Lindelöf's theorem,

every solution can be extended to a solution of maximal duration arbitrarily close to the boundary of D by Peano's theorem, however. The solution is unique, except that all restrictions of the solution to a sub-interval are also solutions.

Example 17. The initial value problem

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

has the unique maximal solution $y(t) = \frac{1}{1-t}$ on the domain $t < 1$. This solution cannot be extended to include the singularity at $t = 1$.

The following global uniqueness theorem shows a stronger property when the domain is $[0, a] \times \mathbb{R}^n$. It is a corollary to Theorems 14 and 16, but used prominently in the proof of Theorem 16, and is of independent interest. A direct proof of the following global version of the Picard-Lindelöf theorem can be found in [Wal98, Proposition 10.VII].

Corollary 18 (Global uniqueness theorem of Picard-Lindelöf)**.** *Let $f : [0, a] \times \mathbb{R}^n \to \mathbb{R}^n$ be a continuous function that is Lipschitz continuous with respect to y. Then, there is a unique solution of the initial value problem (1) on $[0, a]$.*

Exercises

Exercise 1. Subtraction $e - \tilde{e}$ is already included as a term, because it is definable. What about negation $-e$? What about division $e/\tilde{e}$ and powers $e^{\tilde{e}}$?

Exercise 2. Review the basic theory of ordinary differential equations and examples.

Exercise 3. Review the syntax and semantics of first-order logic.

Exercise 4. A number of differential equations and some suggested solutions are listed in Table 1. Are these all solutions? Are there other solutions? In what ways are the solutions be considered more complicated than their differential equations?

Exercise 5 (**). What exactly would change and/or go wrong in which cases if Def. 13 were to demand that the derivative condition of the differential equation is respected at all times $\zeta \in [0, r]$ rather than at all times $\zeta \in (0, r)$ in an open interval?

References

[Alu11] Rajeev Alur. Formal verification of hybrid systems. In Chakraborty et al. [CJBF11], pages 273–278.

[CJBF11] Samarjit Chakraborty, Ahmed Jerraya, Sanjoy K. Baruah, and Sebastian Fischmeister, editors. *Proceedings of the 11th International Conference on Embedded Software, EMSOFT 2011, part of the Seventh Embedded Systems Week, ESWeek 2011, Taipei, Taiwan, October 9-14, 2011*. ACM, 2011.

Table 1: A list of some differential equations and some solutions

ODE	Solution
$x' = 1, x(0) = x_0$	$x(t) = x_0 + t$
$x' = 5, x(0) = x_0$	$x(t) = x_0 + 5t$
$x' = x, x(0) = x_0$	$x(t) = x_0 e^t$
$x' = x^2, x(0) = x_0$	$x(t) = \frac{x_0}{1 - t x_0}$
$x' = \frac{1}{x}, x(0) = 1$	$x(t) = \sqrt{1 + 2t}$
$y'(x) = -2xy, y(0) = 1$	$y(x) = e^{-x^2}$
$x'(t) = tx, x(0) = x_0$	$x(t) = x_0 e^{\frac{t^2}{2}}$
$x' = \sqrt{x}, x(0) = x_0$	$x(t) = \frac{t^2}{4} \pm t\sqrt{x_0} + x_0$
$x' = y, y' = -x, x(0) = 0, y(0) = 1$	$x(t) = \sin t, y(t) = \cos t$
$x' = 1 + x^2, x(0) = 0$	$x(t) = \tan t$
$x'(t) = \frac{2}{t^3} x(t)$	$x(t) = e^{-\frac{1}{t^2}}$ non-analytic
$x'(t) = e^{t^2}$	non-elementary

[DGV96] Akash Deshpande, Aleks Göllü, and Pravin Varaiya. SHIFT: A formalism and a programming language for dynamic networks of hybrid automata. In Panos J. Antsaklis, Wolf Kohn, Anil Nerode, and Shankar Sastry, editors, *Hybrid Systems*, volume 1273 of *LNCS*, pages 113–133. Springer, 1996.

[EEHJ96] Kenneth Eriksson, Donald Estep, Peter Hansbo, and Claes Johnson. *Computational Differential Equations*. Cambridge University Press, 1996.

[FKV04] G. K. Fourlas, K. J. Kyriakopoulos, and C. D. Vournas. Hybrid systems modeling for power systems. *Circuits and Systems Magazine, IEEE*, 4(3):16 – 23, quarter 2004.

[GBF+11] Radu Grosu, Grégory Batt, Flavio H. Fenton, James Glimm, Colas Le Guernic, Scott A. Smolka, and Ezio Bartocci. From cardiac cells to genetic regulatory networks. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *CAV*, volume 6806 of *LNCS*, pages 396–411. Springer, 2011. doi:10.1007/978-3-642-22110-1_31.

[Har64] Philip Hartman. *Ordinary Differential Equations*. John Wiley, 1964.

[KAS+11] BaekGyu Kim, Anaheed Ayoub, Oleg Sokolsky, Insup Lee, Paul L. Jones, Yi Zhang, and Raoul Praful Jetley. Safety-assured development of the gpca infusion pump software. In Chakraborty et al. [CJBF11], pages 155–164. doi:10.1145/2038642.2038667.

[KGDB10] Branko Kerkez, Steven D. Glaser, John A. Dracup, and Roger C. Bales. A hybrid system model of seasonal snowpack water balance. In Karl Henrik Johansson and Wang Yi, editors, *HSCC*, pages 171–180. ACM, 2010. doi:10.1145/1755952.1755977.

[Lin94] 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*, 114:454–457, 1894.

[LS10] Insup Lee and Oleg Sokolsky. Medical cyber physical systems. In Sachin S. Sapatnekar, editor, *DAC*, pages 743–748. ACM, 2010.

[LSC+12] Insup Lee, Oleg Sokolsky, Sanjian Chen, John Hatcliff, Eunkyoung Jee, BaekGyu Kim, Andrew L. King, Margaret Mullen-Fortino, Soojin Park, Alex Roederer, and Krishna K. Venkatasubramanian. Challenges and research directions in medical cyber-physical systems. *Proc. IEEE*, 100(1):75–90, 2012. doi:10.1109/JPROC.2011.2165270.

[Pea90] Giuseppe Peano. Demonstration de l'intégrabilité des équations différentielles ordinaires. *Mathematische Annalen*, 37(2):182–228, 1890.

[PKV09] Erion Plaku, Lydia E. Kavraki, and Moshe Y. Vardi. Hybrid systems: from verification to falsification by combining motion planning and discrete search. *Form. Methods Syst. Des.*, 34(2):157–182, 2009.

[Pla07] André Platzer. Differential dynamic logic for verifying parametric hybrid systems. In Nicola Olivetti, editor, *TABLEAUX*, volume 4548 of *LNCS*, pages 216–232. Springer, 2007. doi:10.1007/978-3-540-73099-6_17.

[Pla08] André Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reas.*, 41(2):143–189, 2008. doi:10.1007/s10817-008-9103-8.

[Pla10] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010. doi:10.1007/978-3-642-14509-4.

[Pla12] André Platzer. Logics of dynamical systems. In *LICS*, pages 13–24. IEEE, 2012. doi:10.1109/LICS.2012.13.

[Pla15] André Platzer. A uniform substitution calculus for differential dynamic logic. In Amy Felty and Aart Middeldorp, editors, *CADE*, volume 9195 of *LNCS*, pages 467–481. Springer, 2015. doi:10.1007/978-3-319-21401-6_32.

[Pre07] President's Council of Advisors on Science and Technology. Leadership under challenge: Information technology R&D in a competitive world. An Assessment of the Federal Networking and Information Technology R&D Program, Aug 2007.

[Rei71] William T. Reid. *Ordinary Differential Equations*. John Wiley, 1971.

[RKR10] Derek Riley, Xenofon Koutsoukos, and Kasandra Riley. Reachability analysis of stochastic hybrid systems: A biodiesel production system. *European Journal on Control*, 16(6):609–623, 2010.

[Tiw11] Ashish Tiwari. Logic in software, dynamical and biological systems. In *LICS*, pages 9–10. IEEE Computer Society, 2011. doi:10.1109/LICS.2011.20.

[TPS98] Claire Tomlin, George J. Pappas, and Shankar Sastry. Conflict resolution for air traffic management: a study in multi-agent hybrid systems. *IEEE T. Automat. Contr.*, 43(4):509–521, 1998.

[Wal98] Wolfgang Walter. *Ordinary Differential Equations*. Springer, 1998.