

A differential extension of Descartes' foundational approach: a new balance between symbolic and analog computation

Pietro Milici¹

¹ Université de Bretagne Occidentale, Brest, France;
p.milici@gmail.com

Abstract

In *La Géométrie*, Descartes proposed a “balance” between geometric constructions and symbolic manipulation with the introduction of suitable ideal machines. In modern terms, that is a balance between analog and symbolic computation.

Descartes' geometric foundational approach (analysis without infinitary objects and synthesis with diagrammatic constructions) has been extended beyond the limits of algebraic polynomials in two different periods: by late 17th century *tractional motion* and by early 20th century *differential algebra*. This paper proves that, adopting these extensions, it is possible to define a new convergence of machines (analog computation), algebra (symbolic manipulations) and a well determined class of mathematical objects that gives scope for a constructive foundation of (a part of) infinitesimal calculus without the conceptual need of infinity. To establish this balance, a clear definition of the constructive limits of tractional motion is provided by a *differential universality theorem*.

1 Introduction

In *La Géométrie*, Descartes proposed a “balance” between geometric constructions and symbolic manipulation with the introduction of suitable ideal machines. In particular, Cartesian tools were polynomial algebra (analysis) and a class of diagrammatic constructions (synthesis). This setting provided a classification of curves, according to which only the algebraic ones were considered “purely geometrical.” Thanks to this approach, geometrical intuition was no longer necessary in proving new properties, because the “method” (suitable algorithms in the analysis) permitted to lead the thought: in modern term, there was the seed of automated reasoning. Descartes' limit was overcome with a general method by Newton and Leibniz introducing the infinity in the analytical part, whereas the synthetic perspective gradually lost importance with respect to the analytical one—geometry became a mean of visualization, no longer of construction.

Descartes' foundational approach (analysis without infinitary objects and synthesis with diagrammatic constructions) has, however, been extended be-

yond algebraic limits, albeit in two different periods. In the late 17th century, the synthetic aspect was extended by *tractional motion* (construction of transcendental curves with idealized machines). In the first half of the 20th century, the analytical part was extended by *differential algebra*, now a branch of computer algebra in which, informally speaking, the indeterminates are not numbers but continuous (and differentiable) functions. This paper seeks to prove that it is possible to obtain a new balance between these synthetic and analytical extensions of Cartesian tools for a class of transcendental problems.

A reason for a renewing of the Cartesian program concerns the historical evolution of mathematical objects. Mathematics can be considered as based on two cornerstones: arithmetics (symbolic manipulation of discrete elements) and geometry (constructions based on idealized continuous behaviors). The synthetic components of such approaches are respectively digital and analog computation, and the mutual relationship between these kind of constructions provides a cognitive richness that leads the main steps of mathematical evolution. When arithmetic and geometric strengths are unbalanced, their mutual conversions can constitute a challenge to overreach their own limits, as evinced from a linguistic perspective in [Kvasz, 2008, Ch. 1] (even though without distinguishing between the constructive and mere visual power of geometry). Being today mainstream mathematics too much oriented toward arithmetics, our aim is to resume the commitment toward geometric constructions. We are dealing with geometry instead of general analog computing because, according to Descartes' perspective, the primitive bases of our knowledge have to be intuitively clear, and the simple components of geometric ideal machines minimize the physical complexity and the cognitive requirements. From this perspective, the millennial endurance of Euclid's geometric paradigm can be justified by the wide application of its synthetic solutions, by the rigor of its analysis, but also by the concreteness of its constructive tools (segments and circles can be traced by ruler and compass).

Concerning the organization of this work, we show a new convergence of machines (analog computation, section 2), algebra (symbolic manipulations, section 3), and geometry (constructed mathematical objects, section 4) that, together with a problem solving method (section 5), gives scope for a foundation of (a part of) infinitesimal calculus without the conceptual need of infinity¹. To establish this balance, a clear (historically missing) definition of the constructive limits of tractional motion is provided by a *differential universality theorem*.

The peculiarity of this work lies in the attention to the constructive role of geometry as idealization of machines for foundational purposes. This approach, after the *de-geometrization* of mathematics, is far removed from the mainstream

¹An objection to such avoidance of infinity could be that we cannot really avoid the infinite in the analytic part, because to define continuous functions at the basis of differential algebra we need limits or similar tools. With regard to this objection, we claim that, even if one considers continuity expressible only through infinitary tools, the allowed operations in differential algebra remain in the field of a finitist symbolic manipulation (in fact, differential algebra is nowadays considered a field of computer algebra). The constructive role of infinity in differential algebra is avoidable as it is in the analysis of polynomial algebra. In classical algebra, indeterminates assume values on the field of the real numbers, the definition of which requires infinity, but algebra remains finite because it does not deal with general real numbers, one simply makes manipulations and can control only a countable subset of real numbers. Similarly, differential algebra does not deal with the definition of continuous functions: the underlying requirement is to manipulate symbols that represent such functions.

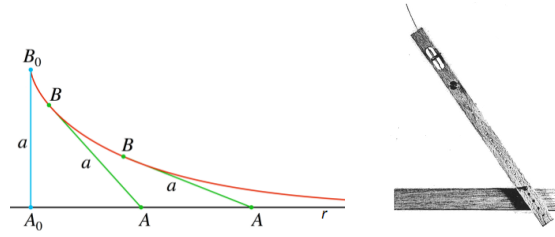


Figure 1: [Left] The heavy body is B , with initial position B_0 , the string is a , and the other end of the string is A , with initial position A_0 . Moving A along r , B describes the tractrix (obviously, the movement is not reversible because of the non-rigidity of the string). Note how a is tangent to the curve at every point. [Right] Reconstruction of Perks' instrument for the tractrix [Pedersen, 1963, p. 17], copyright license number 4518140353283. One can see the wheel taking the place of the load and a bar instead of the string.

discussions of mathematics, especially regarding foundations. However, though forgotten these days, the problem of defining appropriate canons of construction was very important in the early modern era, and heavily influenced the definition of mathematical objects and methods. According to Bos' definition in Bos [2001], these are *exactness problems* for geometry.

2 Machines (analog computation)

2.1 Brief history of tractional motion

The problem of extending geometry beyond Cartesian limits was dominant between 1650 and 1750 Bos [1988], and in this section, we shortly deal with it.

If direct tangent problems are present since the classical period, it was only in the second half of the 17th century that the inverse ones appeared. The main difference between direct and inverse tangent problems is the role of the curve: in the direct case it is given *a priori*, while in the second the curve is sought given some properties that its tangent has to satisfy. Even though beyond Cartesian geometry, to legitimate solutions of inverse tangent problems there was the introduction of certain machines, intended as both theoretical and practical instruments, able to trace such curves. The first documented curves constructed under tangent conditions were physically realized by the traction of a string tied to a load, which is why the study of these machines was named *tractional motion*. Consider the following example:

On a horizontal plane, a small heavy body (subjected to the friction on the plane) is tied with an ideally weightless non-elastic string, and imagine (slowly) pulling the other end of the string along a straight line drawn on the plane.

Because of the friction on the plane, the body offers resistance to the pulling of the string: if the motion is slow enough to neglect inertia, the curve described by the body is called a *tractrix*. Examining the left of Fig. 1, we can see how the curve is traced thanks to the property that the string is constantly tangent to the curve.

During this period, mathematicians like Huygens began to consider instruments that, like the handlebars of a bike, could guide the tangent of a curve (in analytical mechanics terms, they introduced *non-holonomic* constraints), in order to avoid inessential physical complications and so to consider tractional motion as “pure geometric.” Tractional motion suggested the possibility of constructing curves by imposing tangential conditions, generalizing (in a non-Cartesian way) the idea of geometrical objects, and constructing with new tools not only algebraic curves, but also some transcendental ones (seen as solutions of differential equations). During this period, the development of geometrical ideas often corresponded to the practical construction (or at least conception) of mechanical machines able to embody the theoretical properties, and thus able to trace the curves. Concerning practical machines, we recall those introduced in Perks [1706, 1714] (for the machine for the tractrix see the right of Fig. 1), which, for the first time, included a “rolling wheel” to guide the tangent. A more influential role for similar machines was played by the ones proposed in [Poleni, 1729, *Ad Iacobum Hermannum Epistola*]. An overview of such machines is visible in Crippa and Milici [2019]. As deepened below, the wheel is able to solve the inverse tangent problem because it avoids the lateral motion of its contact point. Furthermore, wheels imply less physical problems than dragging loads (e.g. inertia).

While questions about exactness in geometric constructions were so important in the early modern period, they disappeared in the 18th century because of the general affirmation of symbolic procedures, later considered autonomous from geometry. But, in contrast to what happened for algebraic curves, tractional motion did not reach a widely affirmed canon of constructions. Moreover, due to the change in paradigm, the geometric-mechanical ideas behind tractional machines remained forgotten for centuries, even for practical purposes, and were independently re-invented in the late 19th century, when they were used to build some grapho-mechanical instruments of integration (integragraphs) to analogically compute symbolically non-solvable problems (for further reading, see Blasjo [2017], Tournès [2009]).

2.2 Components of tractional motion machines

Leaving history behind, the goal of this part is to clearly define the components that can be used to obtain devices that implement certain tangent properties on a plane. Such clarification about the machines to be accepted in tractional constructions was historically missing: these ideal devices have only recently been defined with the so called *tractional motion machines* (or TMMs) Milici [2012a, 2015].

We define the mechanical components that are allowed in a modern interpretation of tractional motion: we adopt these because they seem to give a good compromise between the simplicity of the components (two instruments and two constraints) and that of the assembled machines (even if the proposed components are not minimal [Milici, 2012a, Section 2]). Machines obtained assembling these components (to be considered on a plane that can be infinitely extended) can be considered as an extension of Kempe’s linkages: Kempe [1876] stated the so-called Kempe’s *universality theorem*, that every bounded portion of a planar algebraic curve can be traced by linkages made of jointed finite-length rods (the proof was flawed, a corrected proof is in Kapovich and Millson

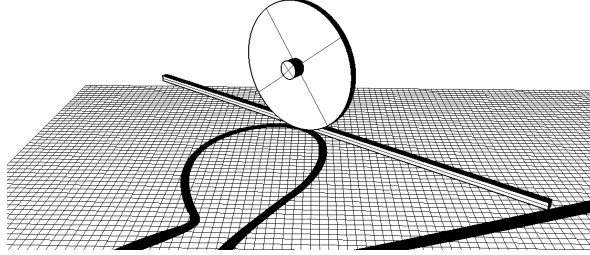


Figure 2: A wheel rolling while following any regular curve has the property that its own *direction* (represented in the picture by a bar) is always tangent to the curve.

[2002]). In section 5.2 we provide a generalization of Kempe’s result for TMMs with a *differential universality theorem*. Now, let us define the components of *Tractional Motion Machines*.

- We adopt **rods**, and assume these have perfect straightness and negligible width. They can be finite or infinitely extensible: in both cases they are different from the Euclidean segments and straight lines, because they are not statically traced objects but planar rigid bodies (mechanical entities with three degrees of freedom, two characterizing the position of a specific point and the third identifying the slope with respect to a fixed line).
- It is possible to put some **carts** on a rod, each one using the rod as a rail: a cart has one degree of freedom once placed on a rod (the cart can only move up and down the rod).
- The **joint** is a constraint between fixed points of two (or more) different objects (here, “object” refers to the plane, a rod, or a cart). Once the joint has been applied, jointed objects can only rotate around their common point (note that, in general, the junction point does not have to be fixed on the plane).
- Finally, we have the non-holonomic constraint, the **wheel**: once a rod r and a point S on r have been selected, we can set a wheel at S that prevents S itself moving perpendicularly to r (considering the motion of S with respect to the plane). Technically, this is as if we put a fixed caster (oriented like r) at S , with its wheel rotating without slipping on the plane. As evinced since the construction of the tractrix, the avoidance of lateral motion in the rod at a point is strongly related to the tangent. If we consider the caster wheel as a disk rolling perpendicularly to the base plane, the projection of the disk surface is always tangent to the curve described by the disk contact point (see Fig. 2). Thus, the rod is tangent to the curve traced by the wheeled point, having the same direction as the caster wheel.

Like Kempe’s linkages, even TMM tools are assumed to be ideal (we do not care about physical inaccuracies), and we do not consider problems related to collisions of rods or of different carts on the same rod. Once specified such details, these components can be used to assemble machines whose motion on

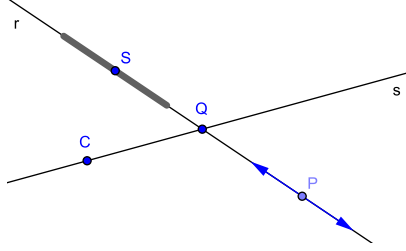


Figure 3: Schematic representation of the components: there are two rods (r and s) joined at Q . On r , there is also a cart P (the arrows stand for the possible motions the cart can have) and a wheel S (the gray thick line ideally represents the projection of a wheel).

a plane is purely kinematic (just kinematic constraints, with no attention to other physical interrelations). For a diagrammatic representation of assembled components, see Fig. 3.

As an important remark note that, differently from the general setting of linkages, we are not only looking for a mechanical method to define geometrical objects, but for a computational model not involving infinity from a foundational perspective. That means that we cannot accept any general distance between points fixed on a rod, because that would imply the introduction of real numbers and so of uncountable sets. As deepened in the following section, a simple solution is to introduce an arbitrary unit length and, for any point P fixed on a rod r , to admit the constructability of the points on r distant one unit from P . This unit-distance primitive operation is necessary because in our model there is no compass available to transfer lengths.

2.3 A code for constructions

Once introduced the components, we have to define how to properly assemble them with an adequate language (as an exemplary formalization of geometric constructions in a computational language see Huckenbeck [1989]). In our language, points are enumerated by natural numbers ($P_0, P_1, \dots$), and the *index* of the point P_n is n . These points, if not differently imposed by other constraints, can freely move on the plane. However, to construct a TMM, we have to start from a certain number of given points that are fixed on the plane. For a minimal definition, we consider as given two distinct fixed points P_0 and P_1 : we introduce the unit as their distance.

Rods are represented by couples of natural numbers: by $r(i, j)$ we mean that a rod is introduced in the point P_i . However, we can consider many rods jointed in the same point, hence the number j distinguishes between them. E.g. let r be the first rod in our construction to be jointed in P_3 , s the second rod jointed in P_3 : in our language r is represented by $r(3, 0)$ and s by $r(3, 1)$. In $r(i, j)$ the ordered values (i, j) are named the *indices* of the rod.

There are three admissible instructions:

- **onRod(k, i, j)** with $k, i, j \in \mathbb{N}$. It imposes that the point P_j lies on the rod $r(k, i)$. Recalling the components, it implies to put a cart on a rod

(P_j can move along $r(k, i)$).

- **dist**(k, i, n, j) with $k, i, j \in \mathbb{N}, n \in \mathbb{Z}$. It imposes that the point P_j lies on the rod $r(k, i)$ at a distance n from P_k . We have to note that n can also be a negative integer, thus we need to define an orientation of the rod, as deepened below. This instruction implies the introduction of a joint.
- **wheel**(k, i) with $k, i \in \mathbb{N}$. It imposes a wheel on the point P_k oriented as the rod $r(k, i)$.

We have to spend few more words about the introduction of integer distances on rods by **dist**. The starting point is that, once introduced the unit, we can constrain any two points on a rod to be at the distance of a unit. Hence, we could consider the instruction **unitDist**(k, i, j) with $k, i, j \in \mathbb{N}$ that imposes P_j to stay on $r(k, i)$ at a distance of one unit from P_k . Without the more general **dist**, we could obtain the point P_l on $r(k, i)$ at a distance of 2 units from P_k by introducing a new rod $r(j, 0)$ with P_k lying on it, and then imposing **unitDist**($j, 0, l$). In this way, the point P_l not coincident with P_k is the sought point. Iterating this construction, we can consider any integer distance on a rod; however, that would require to impose conditions on coincidence of different points (e.g. $P_l \neq P_k$), thus we prefer to adopt **dist** with any integer distance and an orientation.

Furthermore, about the possibility of introducing an orientation, we have to precise that the specific orientation of a rod is not important, it is important that the orientation of the rod is coherent with all its points: e.g. given the instructions **dist**(0,0,2,2) and **dist**(0,0,-1,3), the distance between P_2 and P_3 has to be 3, not 1. As an example to practice with **dist**, consider in Fig. 4 the visual representation of the following code (commented on the right). Note that, using only **dist**, we get exactly Kempe's linkages with integer-length rods.

Example 1.

Code. *Considering the two given fixed points P_0, P_1 , let us introduce the points P_2, P_3 at unary distance to, respectively, P_0 and P_1 . Set the distance between P_2 and P_3 equal to 2 units, and consider their middle point P_4 .*

dist (0,0,1,2)	<i>consider P_2 on the rod $r(0,0)$ at unary distance from P_0</i>
dist (1,0,1,3)	<i>consider P_3 on the rod $r(1,0)$ at unary distance from P_1</i>
dist (2,0,2,3)	<i>set the distance between P_2 and P_3 equal to 2 units along $r(2,0)$</i>
dist (2,0,1,4)	<i>P_4 is the point on $r(2,0)$ distant 1 unit from P_2</i>

Note that, even though there are two point on $r(2,0)$ distant 1 unit from P_2 , P_4 is uniquely defined because it has to be in the ray $\overrightarrow{P_2P_3}$ (P_3 has distance +2 from P_2 , thus with the same sign of the distance of P_4). The other point one unit away from P_2 is P_5 defined by **dist**(2,0,-1,5). About indices, note that two points with different indices can coincide: adding **dist**(3,0,2,2) and **dist**(3,0,1,6) would imply that the point P_6 concides with P_4 in every configuration.

Once defined the instructions, we can consider their analytical conversion. That permits us to use computer algebra as a tool of automated reasoning on the behavior of our machines. First of all, when introducing a rod $r(k, i)$, besides P_k we have to consider an auxiliary point $Q_{k,i}$ to determine the orientation of the

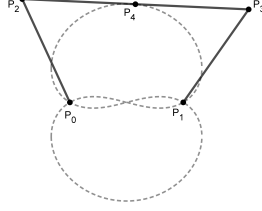


Figure 4: A simple machine without carts and wheels. The dotted line represents the locus defined by the point P_4 . See footnote 5 at page 19 for an analytical study.

rod. This point has to satisfy only the property of being one unit away from P_k (informally, $Q_{k,i}$ is the point P_j s.t. $\text{dist}(\mathbf{k}, \mathbf{i}, 1, \mathbf{j})$). Analytically, considering a system of Cartesian coordinates s.t. $P_0 = (0, 0)$ and $P_1 = (1, 0)$, and adopting the notation $x(P), y(P)$ respectively for the abscissa and the ordinate of the point P , we can introduce $Q_{k,i}$ by the equation:

$$(x(P_k) - x(Q_{k,i}))^2 + (y(P_k) - y(Q_{k,i}))^2 = 1.$$

With $Q_{k,i}$, it is easy to convert instructions in equations: `onRod(k, i, j)` becomes

$$(x(P_k) - x(Q_{k,i}))(y(P_j) - y(Q_{k,i})) = (x(P_j) - x(Q_{k,i}))(y(P_k) - y(Q_{k,i}))$$

and $\text{dist}(\mathbf{k}, \mathbf{i}, \mathbf{n}, \mathbf{j})$ is expressed by

$$\begin{cases} x(P_j) = x(P_k) + n(x(Q_{k,i}) - x(P_k)) \\ y(P_j) = y(P_k) + n(y(Q_{k,i}) - y(P_k)) \end{cases}.$$

For the wheel constraint, we have to consider a point not only as a couple of coordinates, but as a couple of functions. As typical in physics, consider $P_k = (x_k(t), y_k(t))$, i.e. consider the Cartesian coordinates of the point in function of the time. The instruction `wheel(k, i)` poses the condition that P_k cannot move perpendicularly to $r(k, i)$: so, considering $P'_k = (\frac{d}{dt}x_k, \frac{d}{dt}y_k)$, P'_k has to be parallel to $Q_{k,i} - P_k$. Thus, omitting the dependence on t and considering $P'_k = (x'(P_k), y'(P_k))$, the wheel constrain becomes

$$y'(P_k)(x(Q_{k,i}) - x(P_k)) = x'(P_k)(y(Q_{k,i}) - y(P_k)).$$

Thus, both wheel constraints and the other instructions are translatable in polynomials in the variables and their derivatives: as deepened in the algebraic part (section 3), such polynomials are named *differential polynomials* and constitute the basis for differential algebra.

In the following section we introduce subroutines to solve some problems about constructions. When using subroutines, we also need to define the instructions `return(i)` and `newPoint(i)` (with $i \in \mathbb{N}$). Note that, once fixed the input, each implementation can be obtained without subroutines and the last two instructions (they are not adding any primitive to the model). However, subroutines are useful to give general constructions in function of inputs.

About `return(a)`, it returns the natural number a to the program calling the subroutine.

The other instruction, `newPoint(i)`, returns a natural number not previously used to enumerate an already introduced point (i.e. first and third arguments of `onRod`, first and forth arguments of `dist`, first argument of `wheel`). To stay compact, the shorter notation `[i]` stands for `newPoint(i)`. This instruction comes in useful because, in a subroutine, we don't know which natural numbers are yet free to introduce new points. Let n be the highest value used as index of a point before the execution of the instruction: `newPoint` returns the integer $n + 1$. But later we may need to recall the newly introduced index, and we can use multiple `newPoint`: that's why it was necessary to add the index i (e.g. `onRod([0], 0, [1])` constrains the point $P_{[1]}$ to be on the rod $r([0], 0)$). Note that, to introduce a new rod in a given point P_i without knowing how many rods have already been joined in it, we can consider a new point $P_{[k]}$ (assuming that previously we have introduced $P_{[0]}, P_{[1]}, \dots, P_{[k-1]}$) coincident with P_i . The new rod $r([k], 0)$ can be defined by the instruction `dist([k], 0, 0, i)`.

To conclude, we have to remark that the use of `newPoint` can create some unwanted situations. For example, in a subroutine, it can happens that `newPoint` creates an index that is used in the following instructions as a fixed index of a point. To avoid that, it might be useful to have a set of indices reserved for these local construction points, but we leave such possible optimization (as a real software implementation) to future works.

2.4 Algebraic machines

It is also interesting to consider TMMs without wheels. In this case every constraint can be translated in algebraic polynomial equations, so we call the obtained machines “algebraic.” There are algebraic machines that cannot be considered as Kempe's linkages, because rods are not only used to constrain a fixed length between two junction points, but also to allow a point to move along a straight line (thanks to carts). With only Kempe's linkages to trace a straight line is not trivial at all, the problem was solved in an approximated way by Watt, and later exactly solved by Peaucellier (see, for example, [Demaine and O'Rourke, 2007, pp. 29-30]). Algebraic machines can be somehow considered as more adherent to Descartes' machines for geometry than to Kempe's linkages (Descartes used machines in which straight components were allowed to slide along other straight components). Furthermore, the introduction of “extensible” rods, allows to trace not only finite part of algebraic curves, but whole continuous branches of algebraic curves (as already observed, Kempe's linkages can construct only bounded portions of algebraic curves).

Before solving some problems to perform sum and product with algebraic machines, we need to remark that the components of algebraic machines are not part of Euclid's geometry: they are movable mechanical parts, not static traces on a plane, so constructions have to work dynamically. Thus, even though problems like the following ones can be easily solved with ruler and compass if we substitute “rod” with “segment,” we need new constructions for our mechanical setting. All the constructions dynamically work in every configuration of the inputs, and in general input objects are points and rods that can move.

Problem 1 (perpendicular). *Given a rod r and a point P , construct a rod s perpendicular to r passing through P .*

Code. *Let the rod r and P be in our language respectively $r(k, i)$ and P_j . The*

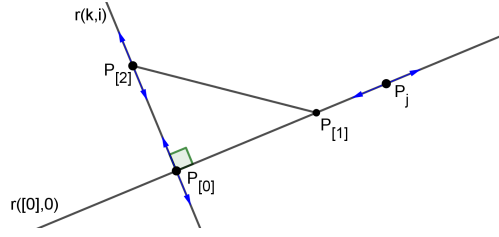


Figure 5: Construction of the rod $r([0], 0)$ perpendicular to $r(k, i)$ passing through P_j .

subroutine returns the number a such that $r(a, 0)$ is the sought rod. Note that we can guarantee that in the rod $r(a, 0)$ the second index is 0 because the point a is newly introduced by the subroutine.

perp(k, i, j)

onRod($k, i, [0]$)

dist($[0], 0, 4, [1]$)

dist($[0], 1, 3, [2]$)

dist($[2], 0, 5, [1]$)

onRod($[0], 0, j$)

onRod($k, i, [2]$)

return($[0]$)

signature of the subroutine

consider a new point $P_{[0]}$ on $r(k, i)$

consider a new point $P_{[1]}$ on $r([0], 0)$ distant 4 units from $P_{[0]}$

consider a new point $P_{[2]}$ distant 3 units from $P_{[0]}$

constrain $P_{[1]}$ to stay 5 units away from $P_{[2]}$

constrain P_j to stay on $r([0], 0)$

constrain $P_{[2]}$ to stay on $r(k, i)$

the sought rod is $r([0], 0)$, return $[0]$

Proof. Construct a right triangle by the junction of a Pythagorean triple as rod lengths (e.g. connect three rods respectively of length 3-4-5). Consider an infinitely extensible rod s for one of the catheti, and make the other cathetus slide on r with two carts on the vertices. Pose another cart on s in correspondence of P . As visible in Fig. 5, that solves the problem. $\square$

Problem 2 (parallel). Given a rod r and a point P , construct a rod parallel to r passing through P .

Code. Let the rod r and P be respectively $r(k, i)$ and P_j . The subroutine returns the number b such that $r(b, 0)$ is the sought rod.

parall(k, i, j)

return(**perp**(**perp**(k, i, j), 0, j))

signature of the subroutine

let's start from the inner subroutines: **perp**(k, i, j) returns the index a s.t. $r(a, 0)$ is a rod perpendicular to $r(k, i)$ passing through P_j . Then, **perp**($a, 0, j$) returns the index b s.t. $r(b, 0)$ is a rod perpendicular to $r(a, 0)$ passing through P_j . Such b is returned.

Proof. According to Problem 1 we can construct a rod s perpendicular to r passing through P , and similarly we can construct a rod t perpendicular to s passing through P : t solves the problem. $\square$

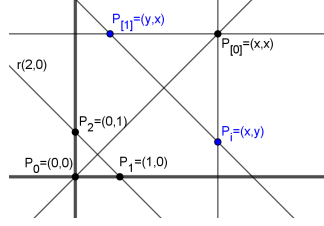


Figure 6: Schema of the construction of the point (y, x) given the point (x, y) using rods parallel and perpendicular to the x-axis and to $r(2, 0)$ (rod through $(1, 0)$ and $(0, 1)$).

In the following problems we adopt Cartesian coordinates to simplify the notation. We have already introduced the points P_0 and P_1 respectively of coordinates $(0, 0)$ and $(1, 0)$. Let's consider the rod $r(0, 0)$ as the usually oriented x-axis (by $\text{dist}(0, 0, 1, 1)$), and the point P_2 at unary distance from P_0 on the rod perpendicular to $r(0, 0)$ passing through P_0 (by $\text{dist}(\text{perp}(0, 0, 0), 0, 1, 2)$). Considering the x-axis oriented horizontally to the right, the possible positions of P_2 can be above or below the x-axis. However, our constructions work for both the possibilities, it only changes the orientation of the coordinate system. In the following figures we always consider the standard positive orientation (while the positive x-axis points right, the positive y-axis points up), but all the results remain with the negative orientation.

Problem 3 (inverse). *Given a point of Cartesian coordinates (x, y) , construct a point of coordinates (y, x) .*

Code. Assuming that P_0, P_1, P_2 are respectively $(0, 0), (1, 0), (0, 1)$, let $P_i = (x, y)$ be the input point. The subroutine returns the index of the point (y, x) .

<i>inv</i> (i)	<i>signature of the subroutine</i>
<code>onRod(2, 0, 1)</code>	the rod $r(2, 0)$ passes through P_1
<code>onRod(perp(0, 0, i), 0, [0])</code>	$P_{[0]}$ is on the rod through P_i parallel to the y-axis
<code>onRod(perp(2, 0, 0), 0, [0])</code>	$P_{[0]}$ also is constrained to lie on the bisector of the first and third quadrant
<code>onRod(parall(0, 0, [0]), 0, [1])</code>	$P_{[1]}$ is on the rod through $P_{[0]}$ parallel to the x-axis
<code>onRod(parall(2, 0, i), 0, [1])</code>	$P_{[1]}$ is also constrained to lie on the rod parallel to $r(2, 0)$ through P_i
<code>return([1])</code>	the sought point is $P_{[1]}$

Proof. As visible in Fig. 6, starting from the point (x, y) and calling $r(2, 0)$ the rod passing through $(1, 0)$ and $(0, 1)$, we can consider the points (x, x) and finally (y, x) by intersecting rods parallel and perpendicular to the x-axis and to $r(2, 0)$. $\square$

According to Problem 1 we can project a point on x- and y- axes and, according to the last problem, we can transpose a length from the ordinate to the abscissa. Indeed, given a point of coordinates (x_0, y_0) , we can construct the points of coordinates $(x_0, 0)$ and $(y_0, 0)$. Conversely, given the points of coordinates $(x_0, 0)$ and $(y_0, 0)$, we can construct the point (x_0, y_0) . Thus, to

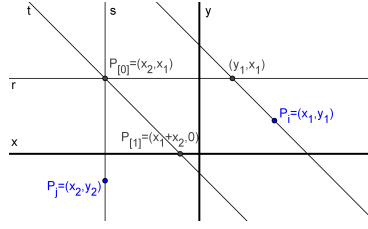


Figure 7: Construction of the point $(x_1 + x_2, 0)$ given the points (x_1, y_1) and (x_2, y_2) .

represent variables in algebraic machines, we can interpret them simply as points moving on the abscissa. Algebraically, such variables are real values. It's time to show how to perform the internal binary operations of sum, difference and multiplication for abscissas of points.

Problem 4 (sum). *Given two points of Cartesian coordinates (x_1, y_1) and (x_2, y_2) , construct a point of coordinates $(x_1 + x_2, 0)$.*

Code. *Consider the x-axis $r(0, 0)$, and let $(x_1, y_1), (x_2, y_1)$ respectively be P_i, P_j . The subroutine returns the index of the point $(x_1 + x_2, 0)$.*

sum(i, j)

onRod(parall(0, 0, inv(i)), 0, [0])

onRod(perp(0, 0, j), 0, [0])

onRod(inv(i), 0, i)

onRod(parall(inv(i), 0, [0]), 0, [1])

onRod(0, 0, [1])

return([1])

signature of the subroutine

a new point $P_{[0]}$ lies on $y = x_1$

$P_{[0]}$ has coordinates (x_2, x_1)

a new rod $r(inv(i), 0)$ passes through (x_1, y_1) and (y_1, x_1)

the new point $P_{[1]}$ lies on the rod parallel to $r(inv(i), 0)$ passing through $P_{[0]}$

the point $P_{[1]}$ is constrained to lie on the x-axis

the sought point $P_{[1]}$ has coordinates $(x_1 + x_2, 0)$

Proof. According to Problem 3, consider the point of coordinates (y_1, x_1) . Thanks to Problem 2 we can consider the rod r parallel to the x-axis passing through (y_1, x_1) and the rod s parallel to the y-axis passing through (x_2, y_2) . By carts we can identify the point (x_2, x_1) (the one lying on both r and s). Finally we can consider the rod t joined in (x_2, x_1) parallel to the rod passing through (x_1, y_1) and (y_1, x_1) : the point in the intersection of t and the x-axis has coordinates $(x_1 + x_2, 0)$ (cf. Fig. 7). $\square$

Problem 5 (difference). *Given two points of coordinates (x_1, y_1) and (x_2, y_2) , construct a point of coordinates $(x_1 - x_2, 0)$.*

Code. *Consider the x-axis $r(0, 0)$, and let $(x_1, y_1), (x_2, y_2)$ respectively be P_i, P_j . The subroutine return the index of the point $(x_1 - x_2, 0)$.*

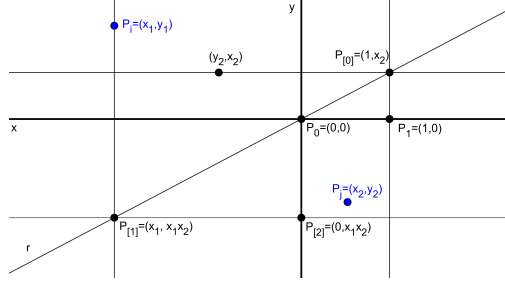


Figure 8: Construction for the multiplication of the abscissae of (x_1, y_1) and (x_2, y_2) . To obtain $(x_1 \cdot x_2, 0)$ we can invert the coordinates of $P_{[2]} = (0, x_1 \cdot x_2)$.

<i>diff</i> (<i>i</i> , <i>j</i>)	<i>signature of the subroutine</i>
<i>onRod</i> (0, 0, [0])	$P_{[0]}$ lies on the x-axis
<i>onRod</i> (0, 0, [1])	$P_{[1]}$ lies on the x-axis
<i>onRod</i> (<i>perp</i> (0, 0, <i>i</i>), 0, [1])	$P_{[1]}$ has coordinates $(x_1, 0)$
<i>dist</i> (<i>sum</i> ([0], <i>j</i>), 0, 0, [1])	the abscissa of $P_{[1]}$ has to be the sum of the abscissae of $P_{[0]}$ and P_j
<i>return</i> ([0])	the sought point $P_{[0]}$ has coordinates $(x_1 - x_2, 0)$

Proof. Consider $P_{[0]}, P_{[1]}$ on the x-axis, constrain $P_{[1]}$ to have the same abscissa of P_j (thus $P_{[1]} = (x_1, 0)$). Using Problem 4, we can constrain $P_{[1]}$ to have as abscissa the sum of the abscissae of $P_{[0]}$ and P_j : hence $P_{[0]} = (x_1 - x_2, 0)$. $\square$

Problem 6 (multiplication). *Given two points of Cartesian coordinates (x_1, y_1) and (x_2, y_2) , construct a point of coordinates $(x_1 \cdot x_2, 0)$.*

Code. Let $(0, 0), (1, 0), (x_1, y_1), (x_2, y_2)$ respectively be P_0, P_1, P_i, P_j , and let the x-axis be $r(0, 0)$. The subroutine return the index of the point $(x_1 \cdot x_2, 0)$.

<i>mult</i> (<i>i</i> , <i>j</i>)	<i>signature of the subroutine</i>
<i>onRod</i> (<i>parall</i> (0, 0, <i>inv</i> (<i>j</i>)), 0, [0])	a new point $P_{[0]}$ lies on the rod parallel to the x-axis passing through (y_2, x_2)
<i>onRod</i> (<i>perp</i> (0, 0, 1), 0, [0])	$P_{[0]}$ is constrained to have 1 as abscissa
<i>onRod</i> ([0], 0, 0)	the origin lies on the rod $r([0], 0)$
<i>onRod</i> (<i>perp</i> (0, 0, <i>i</i>), 0, [1])	a new point $P_{[1]}$ lies on a rod perpendicular to the x-axis passing through P_i
<i>onRod</i> ([0], 0, [1])	the point $P_{[1]}$ is constrained to lie on the rod passing through the origin and $(1, x_2)$
<i>onRod</i> (<i>parall</i> (0, 0, [1]), 0, [2])	$P_{[2]}$ is constrained on the rod parallel to the x-axis passing through $(x_1, x_1 \cdot x_2)$
<i>onRod</i> (<i>perp</i> (0, 0, 0), 0, [2])	$P_{[2]}$ has coordinates $(0, x_1 \cdot x_2)$
<i>return</i> (<i>inv</i> ([2]))	the sought point is the one obtained inverting abscissa and ordinate of $P_{[2]}$

Proof. As it happens in Descartes' interpretation of multiplication (the length $c = a \cdot b$ is given by the proportion $a : c = 1 : b$), we need to use the unit

length. According to Problem 3, construct the point (y_2, x_2) . Considering the intersection of the rod parallel to the x-axis through (y_2, x_2) and the rod parallel to the y-axis through $(1, 0)$, we obtain the point of coordinates $(1, x_2)$. We can introduce the rod r joined in $(1, x_2)$ passing through the origin $(0, 0)$. As visible in Fig. 8, the intersection of r with the rod parallel to the y-axis passing through (x_1, y_1) determines the point $(x_1, x_1 \cdot x_2)$. If we project it on the y-axis we obtain $(0, x_1 \cdot x_2)$, that for Problem 3 gives us the wanted $(x_1 \cdot x_2, 0)$. $\square$

The possibility of performing addition and multiplication with algebraic machines is useful to analytically define the behaviour of algebraic machines and general TMMs in section 4.

3 Differential algebra (symbolic computation)

In Cartesian geometry, polynomial algebra is used as finite tool for analysis. In the proposed differential extension, we substitute polynomials with differential polynomials, machines for algebraic constructions (algebraic machines) with TMMs, and algebraic curves with manifolds of zeros of differential polynomials. In this section, we delve deeper into the analytical counterpart of TMMs, the *differential algebra*, specifically *differential elimination*. The peculiarity of this approach is that it is algorithmically implementable (it is part of computer algebra): its finite symbolic manipulation does not need any reference to infinitary objects (as it happens in infinitesimal calculus). These algebraic tools allow answering some questions about TMMs in section 5.

Differential algebra started with Ritt [1932], where Ritt introduced suitable algebraic tools for differential equations. These results have been reformulated in more and more algebraic way in Ritt [1950] and later in Kolchin [1973]. Under both Ritt and Kolchin, basic differential algebra was developed from a constructive view point and the foundation they built has been advanced and extended to become applicable in symbolic computation, mainly thanks to the passage from old constructive methods (Ritt-Seidenberg algorithm of Seidenberg [1956]) to more recent computational complexity optimizations with Gröbner bases-like approach (firstly introduced in Carrà Ferro [1989])². For a brief introduction to these computational problems and the relative historical evolution see [Boulier, 2007, pp. 110–111].

The aim of differential algebra is to provide an algebraic theory for differential equations both ordinary or with partial derivatives. In particular, its tools and notations are an extension of commutative algebra. To give a short introduction to differential algebra, we recall [Boulier, 2007, pp. 112–116] because of

²Such ideas have been developed in Computer Algebra Systems, e.g. in the package *DifferentialAlgebra* (see <http://www.maplesoft.com/support/help/maple/view.aspx?path=DifferentialAlgebra>). This package is based on the software *BLAD* (standing for *Bibliothèques Lilloises d'Algèbre Différentielle*), developed in the *C* programming language by F. Boulier. While *DifferentialAlgebra* is a package for the commercial software *Maple*, the *BLAD* software is freely available online at <http://www.lifl.fr/~boulier/pmwiki/pmwiki.php?n=Main.BLAD>. Furthermore, even though yet under construction, a differential-algebra experimental package for the open software *SageMath* can be freely downloaded at the link <https://trac.sagemath.org/ticket/13268>. Another free alternative is *ApCoCoA*, available at apcocoa.org (for our purposes, we have to cite the package *diffalg*), a software package based on *CoCoA*, <http://cocoa.dima.unige.it>.

the clarity, the brevity, and the adherence with our aims (according to the kind of constraints obtained by TMMs, we are only interested in ordinary differential equations).

Similarly to classic algebraic geometry, we consider *differential polynomials*. In this case, out of the binary operations of sum and multiplication, we have to introduce the unitary operation of *derivation*. The derivation must be distributive over addition (for every $a, b \in R$ it holds $D(a + b) = D(a) + D(b)$) and must obey the product rule (also called *Leibniz rule*, $D(ab) = D(a)b + aD(b)$). Adopting the standard notation for ordinary derivatives, from now on we write a' instead of $D(a)$.

For our purposes, the coefficients of such differential polynomials are rational numbers. Specifically, given a finite set U of variables, named *differential indeterminates*, a differential polynomial on U is a polynomial on U and the relative derivatives ΘU (if $U = \{x_1, x_2\}$, $\Theta U = \{x_1, x_1', x_1'', \dots, x_2, x_2', x_2'', \dots\}$ and an example of differential polynomial is $\frac{1}{3}x_2'^2 - 5x_1^3x_1''^2x_2 + \frac{2}{13}x_1''^2x_1'''x_2x_2'^4$). In this paper, differential indeterminates can be considered as real functions depending on the single independent variable t , which we may think as the time. We also refer to differential indeterminates as *dependent variables*. Considering by $\mathbb{Q}\{U\}$ the set of all the differential polynomials with rational coefficients on the variables U , it is a *ring* (i.e. a mathematical structure equipped with sum and multiplication and satisfying certain properties: for an introduction to algebraic topics in the non-differential case see Lang [2005]) with a derivation, thus a *differential ring*.

The set of all the polynomials solving some polynomial conditions is captured by a structure named *ideal*. Given an ideal I of polynomials: I contains the null polynomial; the sum of two polynomials in I belongs to I ; the multiplication of a polynomial in I with another polynomial (not necessarily in I) still belongs to I . In algebraic geometry, the set of polynomials satisfied by a given polynomial system forms an ideal that is also radical: an ideal I is said to be *radical* if $a \in I$ whenever there exists some $p \in \mathbb{N}$ so that $a^p \in I$. In the differential case, an ideal I is a *differential ideal* if it is stable under derivation, which is $a' \in I$, for all $a \in I$. Besides, exactly as in non-differential case, a differential ideal I is *radical* if $a^p \in I$ implies $a \in I$ for any integer $p > 0$.

The set of all the “differential and algebraic consequences” of the differential polynomials in a system Σ is the radical differential ideal generated by Σ . To observe how radical differential ideal are related to the study of the solution of a system of differential polynomials, consider $x'^2 - 4x = 0$ (x is the only dependent variable). The analytical solutions are the zero function $x(t) = 0$ and the family of parabolas $x(t) = (t + c)^2$ where c is an arbitrary constant. These are also solutions of all the derivatives of $x'^2 - 4x = 0$ (i.e. $2x'(x'' - 2) = 0$, $2x'x'' + 2x''(x'' - 2) = 0, \dots$) and of every differential polynomial a power of which is a finite linear combination of the derivatives with arbitrary differential polynomials as coefficients, i.e. every element of the radical of the differential ideal generated by $x'^2 - 4x$.

When we are interested in a restriction of all the variables, we take a *projection* of an ideal, and the operation is called *elimination*. The projection of an ideal is still an ideal. In the elimination process for both purely algebraic and differential systems, we need as input a system of (differential) polynomials and an order defining the priority of the variables to be eliminated, so called *ranking*. The output is a system (or a family of systems when splitting is necessary) that

is equivalent to the input system restricted on some variables. Even if in practice the worst case complexity of the algorithms makes problems untreatable, in principle elimination is always possible.

For the differential elimination (and in general to decide the membership of a differential polynomial in a radical differential ideal) the key algorithm is *Rosenfeld-Gröbner* one. As readable in the description of the command in Maple³, given a system Σ containing differential-polynomial equations and inequations, the algorithm splits the given system into other systems defined by certain equations and inequations (the result and the number of cases depend on the ranking of the variables). The solutions of Σ are given by the union of the general solutions of each of the returned systems. Every system is an algebraic structure named *differential regular chain*, and the radical differential ideal generated by Σ is the intersection of all the obtained differential regular chains.

About the differential ranking, if U is a finite set of dependent variables, a *ranking* over U is a total ordering over the set ΘU of all the derivatives of the elements of U which satisfies, for all $a, b \in \Theta U$, $a' > a$ and $a > b \Rightarrow a' > b'$. When $U = \{a\}$ (there is a unique dependent variable), there exists only one ranking: $\dots > a'' > a' > a$. The choice of the ranking is non-trivial when we have more dependent variables. For our purposes, we have to introduce the *orderly* and *eliminating* rankings.

A ranking is said to be *orderly* if, for every $a, b \in U$ and for every positive integer value of i and j , $i > j \Rightarrow a^{(i)} > b^{(j)}$.

If U and V are two finite sets of differential variables, one denotes $U \gg V$ every ranking so that any derivative of any element of U is greater than any derivative of any element of V . Such rankings are said to *eliminate* U with respect to V .

Fixed a ranking, the *leader* is the highest ranking derivative appearing in a differential polynomial. Thus, given $\frac{1}{3}x_2'^2 - 5x_1^3x_1''^2x_2 + \frac{2}{13}x_1''^2x_1'''x_2x_2'^4$, with any orderly ranking the leader is x_1''' (there are no x_2 with derivative more than 1). We have the same leader with the ranking eliminating x_1 . On the contrary, with the ranking eliminating x_2 the leader is x_2' .

To sum up, recalling [Hubert, 2003, pp. 41–42], given a system of differential polynomials Σ in the dependent variables $x_1, \dots, x_n$, with an appropriate choice of the ranking we can:

- check whether a differential polynomial is a solution of $\Sigma = 0$;
- find the differential polynomials satisfied by the solutions of $\Sigma = 0$ in a subset of the dependent variables (we can obtain the equations governing the behavior of the components $x_1, \dots, x_m$, with $m < n$);
- find the lower order differential polynomials satisfied by the solutions of $\Sigma = 0$ (in particular, we can inquire whether the solutions of the system are constrained by purely algebraic equations).

³Cf. <https://www.maplesoft.com/support/help/Maple/view.aspx?path=DifferentialAlgebra/RosenfeldGroebner>

Even though very powerful, the introduced methods do not provide the answer to all the interesting questions of differential algebra. With regard to initial value problems from a computational symbolic perspective,⁴ a lot left to do. Even though Pritchard and Sit [2007] and the approach proposed by Markus Rosenkranz with regard to symbolic methods for (linear) boundary problems (e.g. Rosenkranz et al. [2012]), at my knowledge the symbolic solution of general initial value problems is far away from being solved.

4 Geometry (machines behavior)

To describe the behavior defined by TMMs, we adopt the *behavioral approach* of mathematical models [Polderman and Willems, 1998, pp. 1–8]. The main difference between the behavioral approach and the input/output one is that in the first one we consider all the variables without the need of distinguishing them between input and output. The advantage of missing this distinction comes from the fact that considering interconnection between components (the so-called *feedback*), it is generally hard or impossible to understand which variables are inputs and which ones are outputs. TMMs were firstly introduced adopting the input/output approach Milici [2012a], while in this paper we use the behavioral approach to analytically study the machines with differential algebra instead of classical infinitesimal calculus.

A mathematical model posits that some things can happen, while others cannot. We can formalize this idea by stating that a mathematical model selects a certain subset from a *universum* of possibilities. This subset consists of occurrences that the model allows, that it declares possible. We can refer to the subset in question as the *behavior* of the mathematical model. Such exclusion laws are usually expressed in terms of equations in some variables. The behavior obtained considering all the variables is called *total*. If we want to restrict only to the some variables, we speak of *restricted behavior* (or simply behavior, restricted may be implicit). With this approach two machines are equivalent if they have the same behavior. Before deepening the exploration of TMMs, we begin to explore the behavior of algebraic machines.

Proposition 1. *The total behavior of algebraic machines with n points and m rods is a real algebraic set with integer coefficients in $2(n + m)$ variables.*

Proof. First of all, for algebraic machines we can consider as behavior the set of the configurations allowed by the constraints of the machine, i.e. the possible contemporary positions of the various points. Being on a plane, each of the two coordinates of any point has to be a real value. Given the possibility of translating `onRod` and `dist` in algebraic polynomials with integer coefficients (cf. section 2.3), we can consider as variables the coordinates of the n points P_i . At them we have to add, for each of the m rods, the point $Q_{j,k}$. Therefore, in the

⁴For example, we are interested in the following problem: given two TMMs with their relative initial configurations, are their behaviors equivalent? Analytically, the question arises: given two systems of differential equations with the relative initial conditions, are the systems equivalent? We are looking for an algorithm to symbolically solve this problem. Differential algebra language does not permit even to express this problem because we need to explicitly state the relation between the dependent variables and the independent one (to pose the initial condition).

universum $\mathbb{R}^{2(n+m)}$, the behavior (the possible configurations) is the solution of a system of polynomials with integer coefficients on $2(n+m)$ variables. $\square$

Note that, from an algebraic perspective, it is better to consider rational coefficients instead of integer ones to allow the existence of the inverse of every non-zero coefficient (to define a so called *polynomial ring*). However, multiplying by the minimum common multiple of all the denominators, every polynomial with rational coefficients is equivalent to another one with only integer coefficients, hence we continue considering only integer coefficients.

Proposition 2. *Given any polynomial p with integer coefficients in n real variables, we can consider an algebraic machine having as restricted behavior exactly the zero set of p .*

Proof. Consider p on $x_1, \dots, x_n$ with integer coefficients, let such coefficients be $c_1, \dots, c_k$. Introduce the n moving points $P_i = (x_i, 0)$ by `onRod(0,0,i)` (keep in mind that the x-axis is $r(0,0)$) and the k fixed points $P_{n+j} = (c_j, 0)$ by `dist(0,0,c_j,n+j)`. The polynomial p is made up by sums and multiplications of x_i and c_j . Thus, using the code of Problems 4 and 6, we can construct the point $P_h = (p, 0)$ (with a certain index h) in function of $x_1, \dots, x_n$. To conclude, we can impose p to be constantly equal to 0 by `dist(0,0,0,h)`. Restricting the behavior to the abscissae of $P_1, \dots, P_n$, the proposed machine provides exactly the sought solution of the polynomial p . $\square$

Note that, once physically posed the constraints, not every configuration is always reachable given a certain initial condition. That happens because real algebraic sets (i.e. the solution of systems of algebraic polynomials on real variables) are not always made up by connected components (consider the hyperbola $xy - 1 = 0$: it is made up by two unconnected branches). However, every connected branch of an algebraic set can be continuously traveled by a machine given suitable initial conditions.

Now we have to manage the passage from total to restricted behavior. The projection of any real algebraic set (i.e. the set obtained eliminating some of the original real variables) is named semi-algebraic set (and every real semi-algebraic one can be obtained as the projection of a real algebraic set) Basu et al. [2006]. Any semi-algebraic set can be represented as finite union of sets each one defined by some polynomial equations and inequalities.

Projection of real algebraic sets can be performed by computer algebra software like Maple (that can be used also to analyse the behavior of general TMMs)⁵.

We complete the section with the characterization of the behavior of algebraic machines.

Theorem 1 (Algebraic universality). *Considering n variables, the behavior of algebraic machines coincides with any semi-algebraic set with integer coefficients.*

⁵To perform such projection one can use the *RegularChains* package <https://www.maplesoft.com/support/help/maple/view.aspx?path=RegularChains> (out of clarifying how to use the package, a mathematical definition of regular chains is provided), with also the subpackage *SemiAlgebraicSetTools* for the function *Projection* <https://www.maplesoft.com/support/help/maple/view.aspx?path=RegularChains%2fConstructibleSetTools%2fProjection>. To clarify the ideas, it follows the code to get the behavior of the point P_4 in the example 1 at page 7 (see Fig. 4 for the machine).

Proof. We split the theorem in two: 1. any machine behavior is a semi-algebraic set; 2. for any semi-algebraic set, we can consider a machine with this behavior.

1. By Prop. 1, the total behavior is a real algebraic set. Thus, restricting to n variables, the behavior has to be a semi-algebraic set.

2. Any semi-algebraic set can be represented as finite union of sets each one defined by some polynomial equations and inequalities on n variables. For every inequality $q_i(x_1, \dots, x_n) > 0$, we can introduce a new variable t_i and the polynomial $\tilde{q}_i(x_1, \dots, x_n, t_i) = q_i \cdot t_i^2 - 1$. We can note that, posing $\tilde{q}_i = 0$, we have that $q_i \neq 0$ (q_i has to divide 1) and $q_i \geq 0$ (q_i multiplied by a non-negative value must give a positive value): that means that, thanks to the introduction of new variables, we can obtain the desired inequalities as projection using the new polynomials $\tilde{q}_i$.

We can also consider that the system of real polynomials $p_1 = \dots = p_l = 0$ is equivalent to a single polynomial $(p_1)^2 + \dots + (p_l)^2 = 0$. Hence, a semi-algebraic set can be considered as the projection of a finite union of sets each one defined by a polynomial in $x_1, \dots, x_n, t_1, \dots, t_m$: let such polynomials be $P_1, \dots, P_k$. Finally, the union of such sets has to satisfy the polynomial $P = P_1 \cdot \dots \cdot P_k$, and the semi-algebraic set has to be the projection of $P = 0$ on the variables $x_1, \dots, x_n$. By Prop. 2, the zero-set of P can be constructed by an algebraic machines. Thus, restricting on the variables $x_1, \dots, x_n$, we got a machine with the sought behavior. $\square$

4.1 Universum of TMMs

Differently from the case of algebraic machines, whose behavior is a subset of $\mathbb{R}^n$, the introduction of the wheel makes the previous representation ineffective. To evince it, we propose the example of a TMM: the set of its reachable points

<code>with(RegularChains):</code>	<code>with(SemiAlgebraicSetTools)</code>	load the packages
<code>R:= PolynomialRing([x2, y2, x3, y3, x4, y4])</code>		define the variables of our polynomials with a given order
<code>eq1:= x2^2+y2^2=1</code>		
<code>eq2:= (x3-1)^2+y3^2=1</code>		
<code>eq3:= x2-2*x4+x3=0</code>		define the various equations
<code>eq4:= y2-2*y4+y3=0</code>		
<code>eq5:= (x2-x3)^2+(y2-y3)^2=4</code>		
<code>proj:= Projection([eq1,eq2,eq3,eq4,eq5],2,R)</code>		compute the projection of the equations on the last two variables considering the given order
<code>Display(proj,R)</code>		show the various components of the projection

Thus, non considering the imaginary results and merging the singular solutions in the general one, we have that the locus of the point P_4 is given by the equation $4x^6 - 12x^5 + (13 + 12y^2)x^4 + (-6 - 24y^2)x^3 + (18y^2 + 12y^4 + 1)x^2 + (-12y^4 - 6y^2)x - 3y^2 + 5y^4 + 4y^6 = 0$.

Neglecting practical computer limitation, eliminations are always theoretically computable. For example one could prove by computer algebra that the algebraic machine introduced for the sum in problem 4 works properly without any geometrical consideration. One should translate all the instructions in algebraic equations and then consider the projection on three variables: the two addends (the abscissae of P_i and P_j , i.e. x_1 and x_2) and the variable that should give the result of the operation (the abscissa of $P_{[1]}$, denote it by t). The projection has to provide an equation equivalent to $x_1 + x_2 - t = 0$. Obviously, similar reasonings are obtainable for the operations of difference and multiplication, and, suitably interpreting the geometrical properties in analytic terms, also for all the other problems solved in the section 2.4.

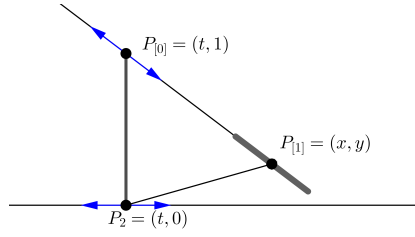


Figure 9: A simple TMM (the point P_2 moves along a line, $P_{[1]}$ rotates around).

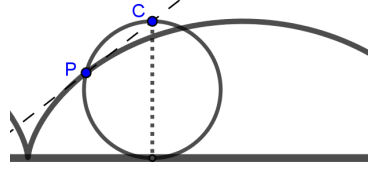


Figure 10: A cycloid can be traced by a point P fixed on a rolling disk. Calling C the top of the disk, the tangent at P has to pass through C .

is a real semi-algebraic set, so it can be obtained with an algebraic machine, but we will intuitively see why its behavior is substantially different from the ones of algebraic machines.

Example 2. Given $P_2 = (t, 0)$ moving on the abscissa and $P_{[0]} = (t, 1)$, consider $P_{[1]}$ s.t. $\overline{P_2 P_{[1]}} = 1$. In $P_{[1]}$ we can place a wheel so that the tangent to the curve traced by $P_{[1]}$ is always in direction of $P_{[0]}$ (see Fig. 9).

Code. First example of a TMM that is not an algebraic machine. Consider $P_0, P_1, r(0, 0)$ as usual.

`onRod(0, 0, 2)`

`dist(perp(0, 0, 2), 0, 1, [0])`

`dist(2, 0, 1, [1])`

`onRod([1], 0, [0])`

`wheel([1], 0)`

consider P_2 on $r(0, 0)$

$P_{[0]}$ is constrained to stay on the perpendicular to $r(0, 0)$ passing through P_2 , one unit away from P_2

a new point $P_{[1]}$ is put at distance 1 from P_2

the point $P_{[0]}$ is constrained to lie on the rod $r([1], 0)$

pose a wheel in $r([1], 0)$ at $P_{[1]}$

Note that, although by construction $P_{[0]} = (t, \pm 1)$, to avoid useless complications we consider only $P_{[0]} = (t, 1)$. While one moves P_2 along the abscissa, if the ordinate of $P_{[1]}$ is strictly less than 1, it has to describe an arc of cycloid, because of the geometrical property shown in Fig. 10 (for a precise analytic proof see the section 5.1, page 28). Remind that the cycloid is a transcendental curve, thus it cannot be traced with 1 degree of freedom by algebraic machines. On the contrary, when $P_{[1]}$ assumes coordinates $(t, 1)$, it can move not only along a cycloid but also in a purely horizontal way, losing the uniqueness (a similar machine was introduced in [Milici, 2015, pp. 10-12]).

the parameter can be considered as the time. Being an idealization of physical machines, we consider these functions to be C^∞ , i.e. *smooth* functions.

With reference to the example of the machine in Fig. 9, we need to consider as universum something like manifolds of curves. However, curves can be defined as classes of equivalence over vector-valued functions. So, to mathematically simplify the definition, we suggest to consider a “manifold of C^∞ functions” as universum for TMMs. In particular, considering n variables, these functions have to be $\mathbb{R} \rightarrow \mathbb{R}^n$.

Algebraic machines are a restriction of TMMs, so we can observe how the interpretation of the universum/behavior as real semi-algebraic set is reformulated as manifold of functions. From the point of view of paths, algebraic machines allow any path moving inside the defined semi-algebraic set $S \subset \mathbb{R}^n$, so the manifold of functions has to be made up by all the functions of class C^∞ having their image inside S .

4.2 Total behavior as solution of differential polynomial systems

We have just defined a manifold of smooth functions as universum of a TMM. Variables are coordinates of specific points, and are considered as functions. As introduced in section 2.3, both wheel constraints and algebraic conditions are translatable in polynomials in the variables and their derivatives: as seen in the algebraic part (section 3), such polynomials are named *differential polynomials* and constitute the basis for differential algebra. More formally:

Proposition 3. *The total behavior of a TMM with n points and m rods is the manifold of all the smooth real functions $\mathbb{R} \rightarrow \mathbb{R}^{2(n+m)}$ satisfying a system Σ of differential polynomial equations with integer coefficients.*

Proof. We just have to translate all the instructions defining the machine in differential equations (purely algebraic equations if the instruction is not `wheel`), taking care of introducing the auxiliary points $Q_{i,j}$ when introducing a rod. By construction, all the coefficients has to be integers. Renaming $x_1, \dots, x_{2(n+m)}$ all the coordinates in function of the time t , and considering $p_1, \dots, p_l$ the obtained differential polynomials on $x_1, \dots, x_{2(n+m)}$, the total behavior is $\{(x_1, \dots, x_{2(n+m)}) | x_i : \mathbb{R} \rightarrow \mathbb{R}, x_i \in C^\infty, p_1 = \dots = p_l = 0\}$. $\square$

Note that we found an analytical form only to the total behavior: for the restricted behavior, in general we have to *eliminate* the unwanted variables.

Proposition 4. *Given a system Σ of differential polynomial equations with integer coefficients, we can construct a machine having as restricted behavior the manifold of the solutions of Σ .*

Proof. First of all, we can convert Σ in an equivalent system involving more variables but with only first derivatives. Let $y_1, \dots, y_m$ be the variables of Σ , and let k_i be the maximum derivative of y_i present in Σ (i.e. $y_i^{(k_i)}$ appears, but $y_i^{(k_i+1)}$ doesn't): for every variable we introduce new auxiliary variables $y_{i,j}$ (for $j = 1, \dots, k_i$) and the differential polynomials

$$y_{i,0} - y_i, y_{i,1} - y'_{i,0}, \dots, y_{i,k_i} - y'_{i,k_i-1}. \quad (1)$$

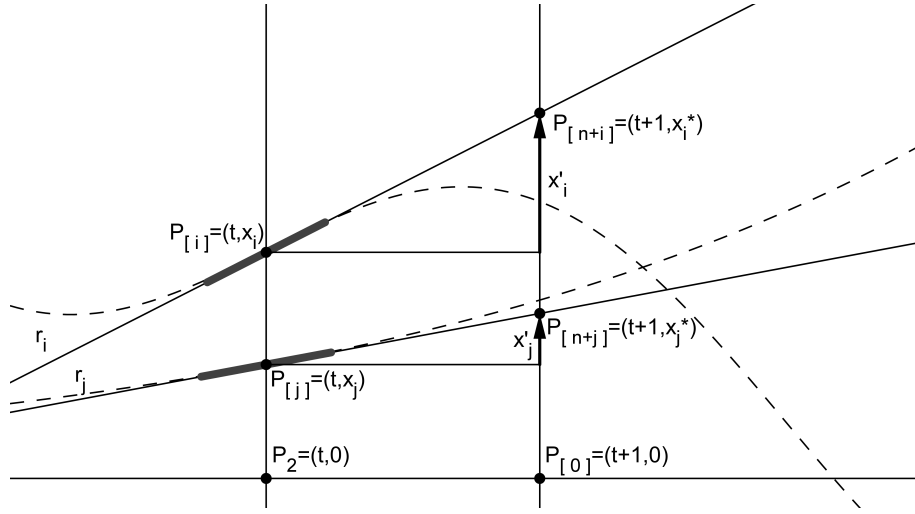


Figure 12: Construction of the derivative of the variables x_i, x_j .

Adding such new polynomials and modifying the system by substituting y_i^k with $y_{i,k}$, we get an equivalent system involving only purely algebraic polynomials and the first order derivatives of (1). To simplify the notation and use a single index (still involving only first order derivatives), denote the various variables $y_{i,k}$ (for all i and k) by $x_1, \dots, x_n$.

Working on real values, we can convert the system Σ given by $p_1 = \dots = p_l = 0$ in a single differential polynomial $p = (p_1)^2 + \dots + (p_l)^2 = 0$. This differential polynomial is a polynomial on $x_1, \dots, x_n$ and their first derivatives. As seen in Prop. 2, we can solve by algebraic machines any polynomial, hence we just have to show how to construct the derivatives of the variables $x_1, \dots, x_n$ (this construction was firstly expressed in Milici [2012a] to solve polynomial Cauchy problems). For this purpose we introduce the following code.

Code. *Given the fixed points P_0, P_1 and $r(0,0)$ as abscissa, in this code we introduce the following points: $P_2 = (t, 0)$, $P_{[0]} = (t+1, 0)$, $P_{[i]} = (t, x_i)$, $P_{[i+n]} = (t+1, x_i + x'_i)$ (for $i = 1, \dots, n$).*

$onRod(0,0,2)$	consider P_2 on $r(0,0)$
$dist([0],0,0,sum(1,2))$	denoting $P_2 = (t,0)$, $P_{[0]}$ is constrained to stay in $(t+1,0)$
$onRod(perp(0,0,2),0,[1])$	
$\dots$	each $P_{[i]}$ is constrained to lie on the rod perpendicular to $r(0,0)$ passing through P_2
$onRod(perp(0,0,2),0,[n])$	
$onRod(perp(0,0,[0]),0,[n+1])$	
$\dots$	each $P_{[n+i]}$ has to lie on the rod perpendicular to $r(0,0)$ passing through $P_{[0]}$
$onRod(perp(0,0,[0]),0,[2n])$	
$onRod([1],0,[n+1])$	
$\dots$	each rod $r([i],0)$ has to pass through $P_{[n+i]}$
$onRod([n],0,[2n])$	
$wheel([1],0)$	
$\dots$	in these lines we introduce wheels in
$wheel([n],0)$	$r([i],0)$

As Fig. 12 illustrates, consider the point $P_2 = (t,0)$ by a cart on the abscissa (t can assume any real value). Note that t is arbitrary, the important thing is that all the various x_i are considered in correspondence of the same t : t can be viewed as the *independent variable* in function of which the various functions (dependent variables) are computed. Then, consider the points $P_{[1]} = (t, x_1), \dots, P_{[n]} = (t, x_n)$. On these points, we can put n rods: call $r_i = r([i],0)$ the rod joined in $P_{[i]}$. Put also a wheel on every r_i in correspondence of $P_{[i]}$. We can construct the rod of equation $x = t+1$: call x_i^* the ordinate of the point $P_{[n+i]}$ in the intersection of $x = t+1$ and r_i .

For what has been observed about the role of the wheel, r_i has to be tangent to the graph of (t, x_i) , hence x_i^* will be $x_i + x'_i$. Obviously, it was not strictly necessary to construct the rod of equation $x = t+1$: in the case of a rod of equation $x = t+a$ (for any constant $a \neq 0$), the intersection of r_i with the new rod is $(t+a, x_i + ax'_i)$ (in other words, $x_i^* = x_i + ax'_i$). It means that we can construct the point $(x'_i, 0)$ that can be used as a new variable, and so the differential polynomial can be considered as a purely algebraic polynomial on $x_1, \dots, x_n, x'_1, \dots, x'_n$. Specifically, the points $(x'_i, 0)$ are obtainable by $diff(inv([n+i]), inv([i]))$ (with $i = 1, \dots, n$).

So, the possibility of solving polynomials with algebraic machines assures that, for every system of differential polynomial equations Σ , we can consider a TMM having as restricted behavior (restricted to the original $y_1, \dots, y_m$) the solution of Σ . $\square$

4.3 Note on “independentization”

As a first example of passage from differential equation to TMM, we can consider the problem $y' = y$. To construct a machine solving it we can start considering a cart $(t,0)$ on a fixed rod (that we consider as abscissa), a rod perpendicular to the abscissa and translating according to the value of t , and on this rod the point (t,y) . As already observed, instead of the rod of equation $x = t+1$, we can consider any other form $x = t+a$. In particular, it is simpler if we adopt $a = -1$. Thus, $y^* = y + ay' = 0$ (for the problem is $y' = y$). Therefore, we have to introduce the rod r passing through (t,y) and $(t-1,0)$, and to put a wheel on

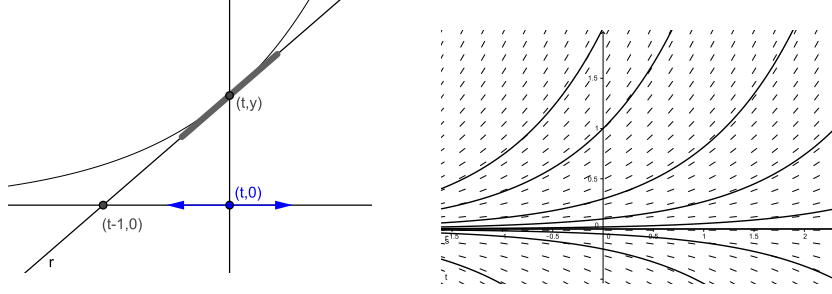


Figure 13: A machine solving the differential equation $y' = y$ (left) and the relative slope field (right).

it in correspondence of (t, y) , obtaining the machine of Fig. 13. Conceptually, this machine is constructively using the property of the exponential curve of having a fixed-length subtangent (i.e. the segment connecting $(t - 1, 0)$ and $(t, 0)$).

Until now, we passed from differential equation to TMM: conversely now we convert the machine in differential polynomials. Given the machine of Fig. 13, what is the differential polynomial system defining its behavior according to the imposed constraints? First of all, let's define the machine more precisely.

Example 3. Consider the machine defined by $P_2 = (t, y)$ and the constraint that the direction of the point (t, y) has to be the line passing through $(t - 1, 0)$.

Code. Consider the fixed points P_0, P_1 .

`onRod(2, 0, diff(2, 1))`

`wheel(2, 0)`

consider $P_2 = (t, y)$, thus $(t - 1, 0)$ is constrained to lie on the rod $r(2, 0)$

pose a wheel in $r(2, 0)$ at P_2

Because of the wheel, (t', y') has to be parallel to $r(2, 0)$, therefore $t'y - y' = 0$. Note that the obtained differential equation is different from the original one ($y' - y = 0$). The difference is given by the implicit assumption that $t' = 1$. When we construct a machine solving a system of differential equations with the method seen in Prop. 4, we implicitly assume that t is the independent variable, so everything is obtained in function of its value. The introduction of a new variable (with constant derivative 1) for the independent one is a standard method to pass from a differential polynomial involving also the independent variable to an equivalent polynomial not depending directly on the independent variable.

In summary, given a system of differential polynomials Σ , with the method of Prop. 4 we can construct a machine solving it, but the system Σ^* obtained analyzing this machine is slightly different from the original Σ . If we want to obtain Σ from Σ^* , we have to add the condition $x'_i = 1$ for the variable x_i that represents the abscissa of the independent point $(t, 0)$. We can call such additional condition the “independentization of a variable” (because from $x'_i = 1$ it follows $x_i = t + k$, i.e. x_i is exactly the independent variable eventually translated of a constant k).

4.4 Note on initial conditions

The total behavior of TMMs can be analytically defined by a system of differential polynomials. However, when a machine is considered to work on a plane, the initial position of its components can be considered implementing the initial conditions. In this section, we focus on how to apply these initial conditions.

Physical realizations of TMMs are devices that can be lifted and downed on the plane. While the device is not yet downed on the plane, there are fewer working constraints (because of the lack of wheel friction), so we can move some points that lose some degrees of freedom when wheels touch the plane. Therefore, if we consider TMMs as physical devices, their assembly and use can be distinguished in two different steps:

1. composition: the various parts are assembled in order to construct the machine;
2. friction on the plane: the machine is “put on the plane,” so wheels avoid lateral motions.

The difference between these two steps is the role of the wheel. In the first case the machine is constructed but, considering it lifted from the plane, the wheel constraints do not work, so on the machine only the holonomic constraints are active (the ones of algebraic machines). When we ideally put the constructed machine on the plane, wheels begin to have friction on the plane, and consequently the related nonholonomic constraints begin to work.

While the composed machine is already defining differential polynomial equations, the activation of the friction is related to the posing of initial conditions. In fact, in the instant when the constructed machine touches the plane (and the wheel friction begins), all the points have a certain position: the values of the variables relative to these positions can be viewed analytically as the initial conditions. Therefore, to pose an initial condition to some variables, we have to suitably move the points (the position of which is related to the wanted variables) when the device is lifted. The downing of the device assures that the variables solve the Cauchy problem.

To clarify these ideas, as an example, we propose a machine solving the differential equation $-y''(t) = y(t)$. According to different initial conditions, the same machine can generate the sine (posing $y(0) = 0, y'(0) = 1$) and the cosine function (with initial conditions $y(0) = 1, y'(0) = 0$).

As seen in Fig. 12, once introduced the point $(t, y(t))$, we can construct the point $(t + 1, y(t) + y'(t))$. Reporting the length $-y'(t)$ as represented in the Fig. 14 (parallel dotted lines represent the translation of lengths, without visualizing all the necessary components), we can construct $(t, -y'(t))$. Then, once constructed $(t + 1, -y'(t) - y''(t))$, it is possible to impose $y(t) = -y''(t)$ by reporting the length $-y''(t)$.

Now it is time to impose initial conditions. For cosine requires totally similar steps, let's consider only the sine function, thus we have to impose $y(0) = 0, y'(0) = 1$. Physically, this condition has to be posed after the construction of the machine, and before the “activation” of the friction of the wheels. First, we move the cart in $(t, 0)$ until it reaches the position $(0, 0)$, then we move the carts $(t, y(t))$ and $(t, -y'(t))$ until they (respectively) reach the positions $(0, 0)$ and $(0, -1)$. Once posed these conditions avoiding the nonholonomic constraints

3. solve the problem with differential algebra algorithms,
4. when requested, after the simplification, find the specific solution with diagrammatic construction of TMMs.

Regarding the third step, we suggest to manipulate equations with the *DifferentialAlgebra* package of the computer algebra system Maple, of which we include commands in footnotes.

5.1 The example of the cycloid

As a first example, by automated reasoning we prove what has been informally observed in the section 4.1 about the behavior of the machine of Ex. 2 (page 20).

Consider $P_2 = (t, 0)$ and $P_{[1]} = (x, y)$ moving around P_2 at unitary distance, so

$$(x - t)^2 + y^2 = 1. \quad (2)$$

The wheel in $P_{[1]}$ has as direction $P_{[1]} - P_{[0]}$, i.e. $P'_{[1]} = (x', y')$ has to be parallel to $P_{[1]} - P_{[0]} = (x - t, y - 1)$:

$$y'(x - t) = x'(y - 1). \quad (3)$$

Thus, we have two equations (the first purely algebraic and the second differential) in t, x, y . If we are interested in the curve traced by $P_{[1]}$, we can use differential elimination to eliminate t . We can proceed with the following steps:

1. consider the differential ring R with rational coefficients having as variables t, x, y , and adopt a ranking eliminating t ;
2. consider the ideal I in R generated by the two differential polynomials;
3. consider in I the *differential regular chains* eliminating t .

We can translate these steps in commands for computer algebra software⁷. In particular we obtain that the *differential regular chains* (for the ideal generated

⁷In Maple we can perform these operations with the following code lines (commented on the right):

<code>with(DifferentialAlgebra)</code>	load the package
<code>R := DifferentialRing(blocks=[t,x,y], derivations=[a])</code>	construct the differential ring with as independent variable a , and dependent ones t, x, y with the ranking $t \gg x \gg y$
<code>p := (x(a)-t(a))^2+y(a)^2 = 1</code>	p is an algebraic equation
<code>q := (diff(y(a), a))*(x(a)-t(a)) = (diff(x(a), a))*(y(a)-1)</code>	q is a differential equation (<code>diff(f(a), a)</code> stands for the derivative df/da)
<code>ideal := RosenfeldGroebner([p, q], R)</code>	<code>ideal</code> is the radical differential ideal generated by p, q
<code>Equations(ideal)</code>	returns the equations of <code>ideal</code>
<code>Inequations(ideal)</code>	returns the inequations of <code>ideal</code>

Note that the commands `Equations(ideal)` and `Inequations(ideal)` show the differential regular chains for the ideal in t, x, y .

Once obtained the differential regular chains reduced with respect to a certain ranking, the elimination of the greater depending variable only consists in taking all and only the equations and inequalities of the differential regular chains where the variable and its derivatives do not occur. Using Maple, it can be achieved with the command: `Equations(ideal, leader < t(a))`.

by the two equations characterizing the TMMs) reduced with the ranking $t \gg x \gg y$ are:

$$C_1 = \{ty' + x'y - x' - xy' = 0, x'^2y - x'^2 + y'^2y + y'^2 = 0, y' \neq 0, x'y - x' \neq 0, y - 1 \neq 0\};$$

$$C_2 = \{t - x = 0, y - 1 = 0\};$$

$$C_3 = \{t^2 - 2xt + x^2 + y^2 - 1 = 0; x' = 0, y' = 0, t - x \neq 0\};$$

$$C_4 = \{t - x = 0, x' = 0, y^2 - 1 = 0, y \neq 0\}.$$

But, as said, we are not interested in the behavior of t , so, if we eliminate it (considering the given ranking of the variables, we can just skip all the equations with t in C_1, C_2, C_3, C_4), we obtain

$$C_1^* = \{x'^2y - x'^2 + y'^2y + y'^2 = 0, y' \neq 0, x'y - x' \neq 0, y - 1 \neq 0\};$$

$$C_2^* = \{y - 1 = 0\};$$

$$C_3^* = \{x' = 0, y' = 0\};$$

$$C_4^* = \{x' = 0, y^2 - 1 = 0, y \neq 0\}.$$

Even though it is possible to do some simplifications, we adopted the given form (that is exactly the one given by the Maple code) to evince the fact that any reasoning can be conducted in a purely formal way without considering the semantic meaning. We can observe that C_3^* and C_4^* does not provide us anything interesting but single points.

On the contrary, we can observe that C_1^* contains as equation the general solution which, rewritten as an ODE, becomes the differential equation of the cycloid:

$$\left(\frac{dy}{dx}\right)^2 = \frac{1-y}{1+y}.$$

Another solution out of arcs of cycloids is made up by the line $y = 1$, that was excluded in C_1^* because of its inequalities.

We can also ask ourself which constraints can be added to construct exactly a cycloid. According to the property seen in Fig. 10 (page 20), we can impose a new tangent condition in another point of the circumference of the rolling disk. As such a point, we can consider the point symmetric to $P_{[1]}$ with respect to the center P_2 , as visible in Fig. 15. Such machine can be defined by appending few instructions to the code of the example 2:

Code. *We add a new point $P_{[2]}$ and pose a wheel on it.*

dist(2, 0, -1, [2])

onRod([2], 0, [0])

wheel([2], 0)

the point $P_{[2]}$ is introduced as the symmetric of $P_{[1]}$ with respect to P_2

the point $P_{[0]}$ is constrained to lie on the rod $r([2], 0)$

pose a wheel in $r([2], 0)$ at $P_{[2]}$

The wheel in $P_{[2]} = (2t - x, -y)$ has as direction $P_{[2]} - P_{[0]}$. That means that $P'_{[2]} = (2t' - x', -y')$ has to be parallel to $P_{[2]} - P_{[0]} = (t - x, -y - 1)$:

$$(2t' - x')(-y - 1) = -y'(t - x). \quad (4)$$

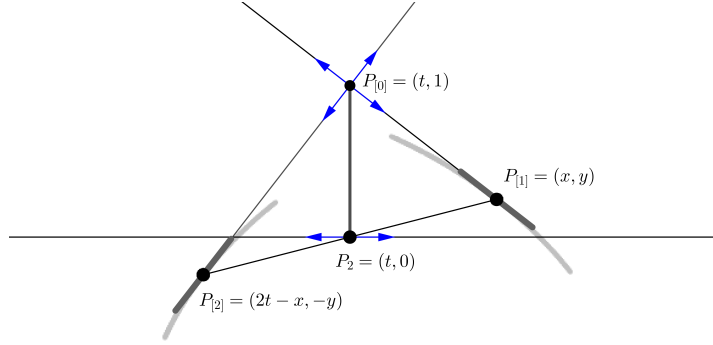


Figure 15: A machine for the cycloid. We introduce a new point $P_{[2]}$ symmetric of $P_{[1]}$ with respect to P_2 . Also in $P_{[2]}$ we pose a wheel on the rod passing through $P_{[0]}$.

If we consider the ideal generated by the three equations (2), (3) and (4), we can compute the relative differential regular chains eliminating t . We obtain that x and y have to satisfy the differential systems $C_1^{**}, C_2^{**}, C_3^{**}$ where $C_1^{**} = C_1^*, C_2^{**} = C_3^*, C_3^{**} = C_4^*$, i.e. there is no longer the solution $y = 1$ given by C_2^* . Thus, with the new conditions, the point (x, y) is always constrained to walk along a cycloid.

5.2 Differential universality theorem and constructible functions

It is time to give a characterization theorem for the behavior of TMMs, somehow the extension of Kempe's (algebraic) universality theorem to the differential case.

Theorem 2 (Differential universality). *Considering n variables, the behavior of a TMM coincides with the union of the solutions of a finite number of systems of differential polynomial equations and inequations with integer coefficients and with real functions as independent variables.*

Proof. We split the theorem in two: 1. the machine behavior is the union of the systems; 2. for any finite set of systems, we can consider a machine with this behavior.

1. By Prop. 3, the total behavior of any TMM is the solution of a differential polynomial system Σ in m variables ($m \geq n$). Thus, to restrict to n variables, we can use the Rosenfeld-Gröbner algorithm of section 3, hence the behavior can be given as a finite union of systems of differential-polynomial equations and inequations.

2. With certain modifications, we can reuse some ideas in the proof of theorem 1 (algebraic universality): for every inequation $q_i(x_1, \dots, x_n) \neq 0$ we can introduce a new variable t_i and the differential polynomial $\tilde{q}_i(x_1, \dots, x_n, t_i) = q_i \cdot t_i - 1$. Thus, even though with more variables, we got that every system can be considered as made up by only differential polynomial equations with integer coefficients. Then, we can use the peculiarity of the variables of TMMs to be real functions: any system of real differential polynomials $p_1 = \dots = p_l = 0$ is

equivalent to a single polynomial $(p_1)^2 + \dots + (p_l)^2 = 0$. Let $P_1, \dots, P_k$ be the differential polynomials identifying the k systems: the union of their zero sets has to satisfy the polynomial $P = P_1 \cdot \dots \cdot P_k$. By Prop. 4, the zero-set of P can be constructed by a TMM. Thus, restricting on the variables $x_1, \dots, x_n$, we got a machine with the sought behavior. $\square$

While theorems of algebraic universality and differential universality have many similarities, we have to highlight that in the algebraic case we have inequalities while in the differential only inequations. Remind that differential algebra does not distinguish between real or complex or other kind of indeterminates (Rosenfeld-Gröbner algorithms works for any differential ring), while semi-algebraic sets are defined specifically for the real case.

We can also define the nature of the functions that TMMs can generate. After the definition of *differentially algebraic functions* [Rubel, 1989, p. 777], we use it to give a classification of the constructible functions. This result is particularly interesting from an historical perspective: differently from the algebraic case, the classification of the curves traced by tractional motion was previously missing.

Definition 1. *A function y is differentially algebraic (shortly: D.A.) if it satisfies an algebraic differential equation, i.e. a differential equation in the form $P(t, y, y', \dots, y^{(n)})$ where P is a nontrivial polynomial in $n + 2$ variables.*

The non-triviality condition is essential because every function is solution of $0 = 0$.

Proposition 5 (constructible functions). *The curves generated by a TMM are all and only the image of D.A. functions.*

Proof. Given a certain TMM, consider a point (x, y) of the machine with one degree of freedom (if the point has two or more degrees of freedom it does not define a curve). The curves traced by this point are defined as all the value of x, y satisfying the restricted behavior, i.e. many systems of differential polynomial equations and inequations in x, y . Now we can be interested in interpreting a curve as the graph of a function (at least locally). So, we can consider the curve as a function $y = f(x)$ [respectively $x = g(y)$ in a neighborhood of vertical tangents; however, we will no longer consider this case because we only have to switch the role of x and y]. To achieve this aim, we can no longer consider x as a dependent variable, but as an independent one. Algebraically, this is translated (as seen in the section 4.3) by the “independentization condition” $x' = 1$. Indeed, if we add the new condition to the systems in x, y , we can again consider the elimination of x obtaining a family of differential regular chains only in y . Thus, we find that the curves $(t, y(t))$ are (locally) solutions of differential polynomials in y , so D.A.

Conversely, we have to recall that any D.A. function satisfies an algebraic differential equation *with integer coefficients* [Rubel, 1989, Th. 1, p. 778]. As observed in section 4.3, the introduction of a new variable with constant derivative 1 for the independent one allows the construction of an equivalent polynomial not depending directly on the independent variable. Once eliminated the new variable, the obtained differential polynomial in y can be solved by a TMM. Hence, the constructible functions are all and only the differentially algebraic ones. $\square$

This result is important because it means that TMMs generate a new dualism beyond algebraic/transcendental (and this time about functions, not curves or algebraic varieties as done with algebraic machines). Note, however, that a machine can construct functions that globally are not D.A., as visible since the first introduction of TMMs (Milici [2012a] concerned the construction of a machine tracing a cycloid, that globally is not D.A.), but locally each of these functions has to be D.A.

All the elementary functions are D.A., and even most of the transcendental functions that we find in analysis handbooks. Historically, the first example of non-D.A. function was the Γ of Euler, as proven in Hölder [1886]. Note that Γ function is not even locally D.A., that is why it cannot be constructed by TMMs.

As an example, we can continue with the cycloid, observing some differences when we “independentize” different variables.

Adding the constraint $x' = 1$ to the equations (2), (3) and (4), we can consider y in function of x . This time, with a ranking eliminating t and x , we obtain only one regular chain:

$$C_{\{x'=1\}} = \{y'^2 y + y'^2 + y - 1 = 0; y' y + y' \neq 0; y + 1 \neq 0\}.$$

This representation is not useful to identify the traced curve as the usual parametrization of a cycloid. This identification is more visible if we “independentize” another variable. Consider the additional constraint $t' = 1$ instead of $x' = 1$. Even in this case, we obtain only one regular chain that, upon eliminating t , becomes:

$$C_{\{t'=1\}} = \{x' - y - 1 = 0; y'^2 + y^2 - 1 = 0; y' \neq 0\}$$

Now we can observe that this representation is the one of

$$\begin{cases} x = t + \cos t \\ y = -\sin t \end{cases}$$

Indeed, instead of the trigonometric functions we can convert the system in a purely differential polynomial one:

$$\begin{cases} x' = 1 + y \\ y'^2 + y^2 = 1 \\ y'' = -y \end{cases}$$

Computationally, we can check that it has as regular chains exactly $C_{\{t'=1\}}$ (in both cases the computed regular chain is $\{y - x' + 1 = 0; x''^2 + x'^2 - 2x' = 0; x'' \neq 0\}$).

Obviously different machines can construct the same manifold of zeros. Remaining on the example of the cycloid, we can construct a TMM having a point of coordinate (x, y) satisfying the equations of $C_{\{t'=1\}}$ by the standard method seen in Prop. 4 and Fig. 12. This way, we consider separately the variables x and y by the introduction of the points (t, x) and (t, y) , impose algebraic and differential conditions on both dependent variables, and then we construct the point having as coordinate (x, y) . This method is general, but of course, does not provide the simplest machine (we do not intend to study the notion of simplicity, consider “simplest machine” in an intuitive sense).

5.3 Equivalence between TMMs

In the previous example, we have seen that two radical ideals were equivalent because they had the same representation. However, the opposite in general does not hold.

As seen in the section 4.2, the total behavior of TMMs is the solution of a system of differential polynomial equations, so the restricted behavior is the restriction to the relative manifold of solutions on some variables. Before checking the equality test between two machines on certain variables, we have to suppose that the variables of the restricted behavior are in the same number in the two manifolds.

Let $X = \{x_1, \dots, x_n\}$, $Y = \{y_1, \dots, y_m\}$, $Z = \{z_1, \dots, z_l\}$ be sets of dependent indeterminates. Consider the radical differential ideal A generated by the differential polynomials $p_1, \dots, p_i$ on $X \cup Y$, and consider B generated by $q_1, \dots, q_j$ on $X \cup Z$.

In general, there is no known method to check the equality of two radical differential ideals represented by regular differential chains (it is related with the so called *Ritt's open problem* Golubitsky et al. [2009]), but in this case we have much stronger hypothesis. In fact, we know the generators of the total behavior: with this condition the solution is easily achievable and computable.

To check the equality between two total behaviors (i.e. between radical differential ideals given by a finite set of generators), we can fix a certain ranking and compute the regular differential chains using the Rosenfeld-Gröbner algorithm, and then we can test whether all the generators of the first ideal belong to the second and vice-versa⁸.

But we are interested in behaviors obtained by eliminating some variables, that are in general represented by an intersection of families of regular chains, and there is no known algorithm to pass from a representation of families of regular chains to a list of generators.

Given the ranking $Z \gg Y \gg X$ (or $Y \gg Z \gg X$), compute the families of regular chains R_A and R_B representing respectively A and B (e.g using Rosenfeld-Gröbner's algorithm). Let R_A^* be R_A without the equations/inequations involving Y and, similarly, let R_B^* be R_B eliminating Z . We can verify the inclusion of A in B restricted to X by checking whether all the p_i belongs to R_B^* ; similarly for the opposite inclusion check whether q_i belongs to R_A^* . If both the inclusions are verified, the two systems are equivalent restricted to the variables X .

Note that we have treated TMMs without any reference to initial conditions. As far as my knowledge goes, the equality problem is still open if we introduce initial values (cf. note 4 at page 17). With regard to some positive results, we can consider Buchberger and Rosenkranz [2012], which provides an algorithm for the symbolic solution of linear boundary problems, passing from differential

⁸According to the notation of this section, in the case of total behaviors Y and Z are empty sets of variables. We can test whether the ideals A and B are equal using the Maple command `BelongsTo` of the *DifferentialAlgebra* package. Once given any ranking, and constructed with `RosenfeldGroebner` the ideals A and B by their generators, to check the equality we only have to test whether all the generators of A belongs to B and vice-versa. In Maple, the command `BelongsTo([p1, ..., pn], B)` produces as output a list of n true/false, the i -th of which indicates whether p_i belongs or not to B . Conversely `BelongsTo([q1, ..., qn], A)` can be used to check the belonging to A .

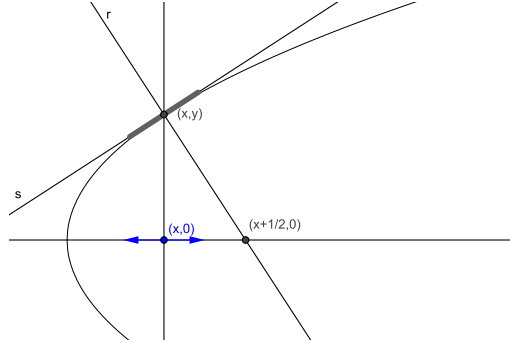


Figure 16: Sketch of a machine with the tangent in (x, y) perpendicular to the line passing through $(x + 1/2, 0)$. To simplify the visualization, only the cart for $(x, 0)$ is shown.

algebra to *integro-differential algebras* (Green’s operators).

Even though without any final answer about initial conditions problems, we want to underline that differential algebra permits to check the equivalence between TMMs. On the other side, even in the more concrete approach to calculus, *computable analysis* Weihrauch [2000], it is not possible to check the equality test between any general couple of generated objects (i.e. computable numbers). Considering intuitively “exactness” as the property of a computational frameworks (both in analytic or geometric paradigm) to be independent from non-finitary procedures (as unlimited approximations), we think that it will be interesting in future to deepen the relation between the computability of the equality test in a theory and the exactness of the theory.

5.4 Differential machines equivalent to algebraic ones

Consider a TMM defining the motion of a point P of coordinates (x, y) so that the tangent in P is perpendicular to the line passing through P and the point $(x + 1/2, 0)$ as the Fig. 16 shows. This machine is defined by the differential polynomial $x' - 2yy'$, which is the total derivative of $x - y^2$. Therefore, for every constant $c \in \mathbb{R}$, solutions are parabolas satisfying $x = y^2 + c$. That means that we are able to trace any of the solutions of this TMM with an algebraic one. Hence, the general question arises: can we characterize the TMMs having solution constructible with algebraic machines (by eventually adding a finite number of real constants of integration)?

Given a differential system or even its restriction on some variables, to find the algebraic constraints satisfied we can simply use the orderly ranking in the Rosenfeld-Gröbner algorithm. There are algebraic constraints if and only if in the obtained family of regular differential chains there are polynomial equations without any proper derivative (i.e. of order 0)⁹.

⁹In Maple, given the dependent variables $x_1, \dots, x_n$ and the independent variable t , one can construct a differential ring with the orderly ranking by the command `R := DifferentialRing(blocks = [[x1, ..., xn]], derivations = [t])` (the double square brackets [...] indicate the orderly ranking). After the usual construction of the ideal `ideal` with the Rosenfeld-Gröbner algorithm, the purely algebraic constraints are given by `Equations(ideal, order=0)`.

It is more complicated if we are interested not only in algebraic constraints, but also on first integrals given by algebraic constraints. Given a system Σ of ODEs, a *first integral* is a function $f(t)$ whose value is constant over the independent variable t along every solution of Σ . To find such first integral, we can use the algorithm *findAllFirstIntegrals* proposed in Boulier and Lemaire [2015]. The algorithm takes as input a family of systems of differential polynomial equations and inequations, and a set of monomials μ_i in the variables and their derivatives ($i = 1, \dots, n$); it returns as output the coefficients α_i s.t. the differential polynomials $\sum_{i=1}^n \alpha_i \mu_i$ are first integrals of Σ .

If in the algorithm we consider as input the behavior of a TMM (a family of regular chains) and as μ_i the various combination of the variables (not their derivatives), we can obtain all the algebraic first integrals of any fixed degree (e.g. given the variables x, y , to check all the algebraic first integrals up to second degree we can consider as μ_i the monomials x, y, x^2, xy, y^2). However, while for any degree we can suitably consider the monomials, at my knowledge there is no general method to check whether the behavior allows any algebraic first integral (of arbitrarily high degree).

5.5 3D TMMs

We considered TMMs working on a plane, but what about the extension of these machines beyond the planar behavior? Here we provide a sketch for the possible physical implementation of a 3D TMM and some shallow explanations about its behavior. Physically, we can introduce a cube of gelatinous material: on it we can hypothesize that thin rods can freely move, while a small disk of center C has to locally represent the tangent plane to the surface walked by C . Similarly to 2D TMMs, the main idea is to set some constraints to the tangent: in this case consider $u(x, y)$ (from now on we implicitly assume the dependence on x, y) as the function to be found and, as usual in partial differential equations, let $u_x = \frac{\partial u}{\partial x}$, $u_y = \frac{\partial u}{\partial y}$. Considering the point $C = (x, y, u)$, the disk centered in C has two perpendicular rods on the tangent plane passing respectively through $(x + 1, y, u + u_x)$ and $(x, y + 1, u + u_y)$, as visible in Fig. 17. That imposes the suitable partial derivative conditions u_x, u_y , and the lengths u_x, u_y can be used to set other conditions (as in 2D TMMs).

The study of such machines could be interesting as a future perspective. However, even if it is far away from the main aims of this work, we give an informal justification about the idea that 3D TMMs do not generate any function $\mathbb{R} \rightarrow \mathbb{R}$ out of the ones constructible by 2D TMMs (it is not a complete proof, we do not deal with coefficients or with a sufficiently deepened characterization of 3D TMMs).

Looking for a characterization of the behavior of such machines, considering as variables smooth $\mathbb{R}^2 \rightarrow \mathbb{R}$ functions, we can perform their sum, multiplication and derivation. Differently from the 2D case, this time the derivation is with respect to two different independent variables (x, y) . Even with these machines, the suitable analytical tool is differential algebra, but with partial derivatives: such development is present since Ritt's works [Ritt, 1950, Ch. IX]. The elimination is still available, so a constructible function $u : \mathbb{R}^2 \rightarrow \mathbb{R}$ defined by a 3D TMM has to be solution of non-trivial differential polynomials in u and a finite number of its partial derivatives.

To compare constructible functions between 3D and 2D TMMs, we have

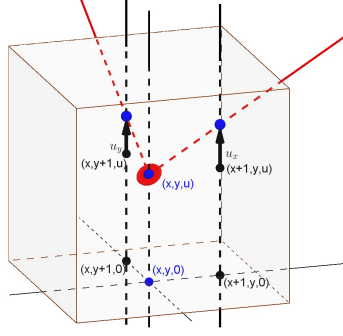


Figure 17: Sketch of 3D TMMs. To set partial derivative conditions on $u(x, y)$ we can consider a jelly cube. Inside, a small disk centered in (x, y, u) is constrained by two rods to lie on the plane passing through $(x + 1, y, u + u_x)$ and $(x, y + 1, u + u_y)$. That imposes u to satisfy certain values of u_x and u_y .

to restrict somehow the functions of the 3D case. Specifically, the graph of a unary function can be obtained intersecting the graph G of $(x, y, u(x, y))$ with a plane α perpendicular to OXY . The intersection between α and OXY is a line: let its natural parametrization be $x = \frac{a}{\sqrt{a^2+b^2}}t + a_0, y = \frac{b}{\sqrt{a^2+b^2}}t + b_0$ (as natural parametrization we mean that the derivative of the parametrization has unary length). Thus, called F the real function whose graph is defined by the intersection of α and G , $F(t) = u\left(\frac{a}{\sqrt{a^2+b^2}}t + a_0, \frac{b}{\sqrt{a^2+b^2}}t + b_0\right)$. We want to show that F is a D.A. function.

Consider the function $f(x, y) = u\left(\frac{a}{\sqrt{a^2+b^2}}x + a_0, \frac{b}{\sqrt{a^2+b^2}}x + b_0\right)$ (y is not used in its computation). We cannot convert such f directly with differential polynomials in partial differential algebra (differential algebra doesn't deal with arguments of functions), but we can compute the partial derivatives of f . Trivially $f_y = 0$, and, using the chain rule for the derivation of compositions involving multivariable functions, $f_x = u_x \frac{d}{dx} \left(\frac{a}{\sqrt{a^2+b^2}}x + a_0\right) + u_y \frac{d}{dx} \left(\frac{b}{\sqrt{a^2+b^2}}x + b_0\right) = \frac{au_x + bu_y}{\sqrt{a^2+b^2}}$. Thus, we can finally add the differential polynomial conditions for f_x, f_y to the polynomials defining the total behavior of the 3D TMM. By the elimination of all the dependent variables out of f , we get polynomials on f and its partial derivatives. But, being $f_y = 0$, we can simply rename $f(x, y) = F(x), f_x(x, y) = F'(x), f_{xx}(x, y) = F''(x), \dots$, thus obtaining a differential polynomial just on F and its ordinary derivatives, i.e. F has to be a D.A. function.

6 Conclusions and future perspectives

6.1 Extending Descartes' balance

The richness of Cartesian setting depends on the correspondence between objects of the analytical and the synthetic part. From this perspective, the role of suitable ideal machines was central. The balance between machines, algebra, and geometry as suggested by Descartes was historically broken by the increase in importance of the analytical part with respect to geometric constructions.

In particular, infinitesimal analysis introduced infinitary tools in the analytical part such as series or infinitesimal elements. However, even though with some centuries of delay, we can consider the finite approach to calculus objects of differential algebra as a legitimate descendant of polynomial algebra. Contrarily, the synthetic part can be managed with the proposed TMMs, which, as a well-defined model for tractional constructions, can be considered as an extension of Descartes' machines. The surprising result is that these heirs of Descartes' analytical and synthetic tools are still in balance (*differential universality theorem*), being the behavior of TMMs exactly the space of solutions definable with differential algebra (restricted to ordinary differential equations). Furthermore, to study the properties of the machines there is no need of geometric or analytic insights, because the algebraic part provides algorithms to compute them autonomously (automated reasoning).

In this paper, we have been able to define the behaviors of TMMs that have been introduced to formalize tractional constructions in a modern way. To my knowledge, it is the first clear definition of the limits of tractional motion. Such limits permit a distinction between objects that are constructible with TMMs and others that are not. To define the behavior of such machines, we used manifolds of functions: if Descartes' setting defined a dualism between algebraic and transcendental curves, our setting facilitates a new dualism between functions. As introduced in section 5.2, the obtainable functions are the *differential algebraic* ones (shortly: D.A.), i.e. solutions of algebraic differential equations¹⁰. As already mentioned, all elementary functions are D.A., and even most of the transcendental functions that we find in most of the analysis handbooks. Historically, the first example of non-D.A. function was the Γ of Euler, as proven in 1886 by Hölder. According to Rubel [1989], not-D.A. functions are named *transcendentally transcendental*. About functions in one variable, we can provide a finer distinction beyond the algebraic and transcendental dualism. Of course algebraic functions are also D.A., so, calling *algebraic-transcendental* the functions that are D.A. but not algebraic, we can divide functions in the cases of Table 1 (with some examples).

6.2 Beyond TMMs

About future extension beyond TMMs, a crucial role could be played by Euler Γ function. As we are going to examine, this function can play for D.A. functions the same role as the exponential curve played for algebraic curves.

Algebraic curves are defined as the zero set of polynomials, where a polynomial is an expression that involves only the operations of addition, subtraction, multiplication, and non-negative integer exponents. One can ask to relax the constraint of considering only non-negative integer exponents (e.g. we may be

¹⁰ Note that, even though without the geometrical interpretation, such dualism was introduced in analog computation by Shannon's *General Purpose Analog Computer* Shannon [1941] many centuries after the introduction of tractional constructions. As visible in the title of Shannon's paper, the GPAC was a theoretical model for the analog computer called *differential analyzer*.

Furthermore, we also need a note on an implicit assumption. We considered function locally smooth (i.e. of class C^∞ in a certain domain). In general, to be a solution of an algebraic differential equation of order n , we have to assume that the function is C^n , not necessary C^∞ . For not-smooth functions, most of the results of differential algebra fail. These cases are treated in Rubel [1983].

Table 1: Categorization of functions in one variable (taken from [Shannon, 1941, p. 501]).

Transcendental		Algebraic	
<i>Transcendentally transcendental</i>	<i>Algebraic-transcendental</i>	<i>Irrational algebraic</i>	<i>Rational</i>
Euler Γ , Riemann ζ	$e^x, \log(x)$; trigonometric, hyperbolic and inverses; Bessel, elliptic and probability functions	x^m (m a rational fraction); solutions of algebraic equations in terms of a parameter	polynomials, quotients of polynomials

interested in considering monomials in $x^{-\frac{3}{2}}$, or in $x^{\sqrt{5}}$): from this perspective, the exponential curve solves the problem of generic exponent. Concerning constructions, tractional motion justified the exponential curve with the introduction of loads subject to friction or with blades or wheels. Therefore, even though the extension from polynomials to formulas with any exponent is not enough to define analytically all the functions constructible by tractional motion, the construction of the exponential was important to focus on the role of the wheel for the expansion into the synthetic aspect.

D.A. functions are solutions of differential polynomials. Differential polynomials are polynomials in the variables and their derivatives, but these derivatives have to be of non-negative integer order. Negative integer-order derivatives can be considered integrals. However, what does it mean to consider derivatives of non-integer order? This question is older than three centuries and is at the core of *fractional calculus*.

The idea of extending the meaning of $\frac{d^n y}{dx^n}$ to $n \notin \mathbb{N}$ appeared the first time in a letter of Leibniz to L'Hôpital (September 30, 1695), and later got many mathematicians interested in it: Euler, Fourier, Abel, Liouville, Riemann, Laurent, Hadamard, Schwartz (for more precise historical references see Ross [1975], Ross [1977]). Nowadays, fractional calculus finds use in many fields of science and engineering, including fluid flow, rheology, diffusive transport akin to diffusion, electrical networks, electromagnetic theory, and probability. There is not a unique definition of fractional integral, but the following (usually called *Riemann-Liouville fractional integral*) is probably the most used version. We are proposing it just to give a first shallow idea, for clarifications and further reading, see Miller and Ross [1993].

Starting from the Cauchy formula for repeated integration (it allows to compress n antidifferentiations of a function into a single integral)

$$D^{-n}f(x) = \frac{1}{(n-1)!} \int_a^x (x-t)^{n-1} f(t) dt,$$

we can generalize n to non-integer values and, since $n! = \Gamma(n+1)$, we get

$$D^{-v}f(x) = \frac{1}{\Gamma(v)} \int_a^x (x-t)^{v-1} f(t) dt.$$

This formula links Γ function and fractional calculus. The construction of Γ with idealized machines could be particularly important because a widely accepted geometric interpretation of fractional calculus is still missing (for some attempts,

see Adda [1997], Podlubny [2002], Tavassoli et al. [2013], Herrmann [2014]). Hence, from a historical/philosophical perspective, fractional calculus is now looking for a constructive-synthetic geometrical legitimation, as it happened in early modern period with transcendental curves. We hope that TMMs can constitute a solid step over which such extensions may come.

6.3 Further perspectives

Since Newton and Leibniz, the core concept of calculus is the constructive role of methods involving the infinity. On the contrary, the proposed mechanical setting and the differential algebra counterpart suggest that it is possible to consider calculus (at least the part dealing with differential polynomials) without the need of infinity, but with the less abstract idea that “the wheel direction is the tangent.” A pedagogical peculiarity of such an approach to infinitesimal analysis with mathematical machines is that students can manipulate concepts usually considered too abstract. Some very preliminary attempts to introduce tractional machines in math education have been proposed in a workshop Milici and Di Paola [2012] and in some papers Milici [2012b], Di Paola and Milici [2012], Salvi and Milici [2013]. A proposal for science museum activities can be also found in Milici and Dawson [2012].

Furthermore, the possibility of a restructuration of infinitesimal analysis in the light of TMMs and differential algebra should be interesting to be investigated from computational, instrumental, visual, algebraic, cognitive, and foundational viewpoints Milici [2017].

Finally, we want to conclude with a remark on the mutual evolution of analog and digital/symbolic computation in mathematics. An approach to break the Church-Turing thesis is to check whether some results beyond Turing computational limits may be reached somehow (the *hypercomputation* problem, e.g. Copeland [2002]). With regard to this question, it could be interesting to set the problem from a purely mathematical point of view. Instead of considering the physical limits of analog computing, one could look for an “exact” approach to analog computation through geometry because of its cognitive simplicity and richness. From this point of view, considering diagrammatic constructions and symbolic manipulations respectively as analog and digital computations, the evolution of mathematical foundational paradigms from the geometric/arithmetical perspectives (with their relative intercourses and extensions) can be considered an evolution of computational limits.

Considering the computational power of mathematical approaches, Pythagorean rational numbers (arithmetic perspective) were not sufficient to express some values generated by the arithmetic reinterpretation of ruler-and-compass geometric constructions (the length of the diagonal of a square with respect to the edge). On the contrary, later polynomial algebra introduced values not geometrically constructible by ruler and compass (the exactness problem in the early modern period). However, the unbalance between the powers of the different paradigms is not a constant. Descartes balanced their powers in analytical geometry, and this powerful paradigm became the hard core over which calculus evolved, generating a rich symbolism inspired by ideas derived from geometry and mechanics. Something new happened with regard to calculus: if the geometrical paradigm had already been abandoned in other periods, for the

first time there was the acceptance of entities generated by infinite processes (note that infinite procedures were also adopted by Archimedes, but only as an investigative tool to be later interpreted from a synthetic perspective). This acceptance of infinite processes made it difficult to re-interpret the obtained entities in everyday (finite) experience. Hence the claim of this paper: we reached part of infinitesimal calculus with suitable geometrical constructions (synthesis) and symbolic tools provided by a finite algebra (analysis).

Even if the balance introduced has no claim of constructing something beyond Turing limits, it is another case in which analog and digital constructive powers are balanced (as in Descartes' geometry). As differential calculus evolved on Cartesian geometry, we hope that in future it could be interesting to investigate whether the new balance proposed in this work could become a step for new computational paradigms and mathematical constructive approaches beyond the limits of today.

References

- F. B. Adda. Geometric interpretation of the fractional derivative. *Journal of Fractional Calculus*, 11:21–52, 1997.
- S. Basu, R. Pollack, and M. F. Roy. *Algorithms in real algebraic geometry*. Springer, 2006.
- V. Blasjo. *Transcendental curves in the Leibnizian Calculus*. Academic Press, 2017.
- H. J. M. Bos. Tractional motion and the legitimation of transcendental curves. *Centaurus*, 31(1):9–62, 1988.
- H. J. M. Bos. *Redefining geometrical exactness: Descartes' transformation of the early modern concept of construction*. Springer-Verlag, New York, 2001.
- F. Boulier. Differential elimination and biological modelling. *Gröbner bases in symbolic analysis*, 2:109–137, 2007.
- F. Boulier and F. Lemaire. Finding first integrals using normal forms modulo differential regular chains. In *International Workshop on Computer Algebra in Scientific Computing*, pages 101–118. Springer, 2015.
- B. Buchberger and M. Rosenkranz. Transforming problems from analysis to algebra: a case study in linear boundary problems. *Journal of Symbolic Computation*, 47(6):589–609, 2012.
- G. Carrà Ferro. Gröbner bases and differential algebra. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 129–140. Springer, 1989.
- B. J. Copeland. Hypercomputation. *Minds and machines*, 12(4):461–502, 2002.
- D. Crippa and P. Milici. A relationship between the tractrix and logarithmic curves with mechanical applications. *The Mathematical Intelligencer*, 2019. DOI: 10.1007/s00283-019-09895-7.

- E. D. Demaine and J. O'Rourke. *Geometric folding algorithms*. Cambridge university press Cambridge, 2007.
- B. Di Paola and P. Milici. Geometrical-mechanical artefacts mediating tangent meaning: the tangentograph. *Acta Didactica Universitatis Comenianae - Mathematics*, pages 1–13, 2012.
- M. P. Do Carmo. *Differential geometry of curves and surfaces*, volume 2. Prentice-Hall Englewood Cliffs, 1976.
- O. D. Golubitsky, M. V. Kondratieva, and A. I. Ovchinnikov. On the generalized Ritt problem as a computational problem. *Journal of Mathematical Sciences*, 163(5):515–522, 2009.
- R. Herrmann. Towards a geometric interpretation of generalized fractional integrals - Erdélyi-Kober type integrals on R^N , as an example. *Fractional Calculus & Applied Analysis*, 17(2):361, 2014.
- O. Hölder. Ueber die Eigenschaft der Gammafunction keiner algebraischen Differentialgleichung zu genügen. *Mathematische Annalen*, 28(1):1–13, 1886.
- E. Hubert. Notes on triangular sets and triangulation-decomposition algorithms II: Differential systems. *Symbolic and Numerical Scientific Computation*, pages 40–87, 2003.
- U. Hickenbeck. Euclidian geometry in terms of automata theory. *Theoretical computer science*, 68(1):71–87, 1989.
- M. Kapovich and J. J. Millson. Universality theorems for configuration spaces of planar linkages. *Topology*, 41(6):1051–1107, 2002.
- A. B. Kempe. On a general method of describing plane curves of the n th degree by linkwork. *Proceedings of the London Mathematical Society*, 7:213–216, 1876.
- E. R. Kolchin. *Differential algebra & algebraic groups*, volume 54 of *Pure and Applied Mathematics*. Academic Press, 1973.
- L. Kvasz. *Patterns of change: linguistic innovations in the development of classical mathematics*. Springer Science & Business Media, 2008.
- S. Lang. *Undergraduate algebra*. Springer Science & Business Media, 2005.
- P. Milici. Tractional motion machines extend GPAC-generable functions. *International Journal of Unconventional Computing*, 8(3):221–233, 2012a.
- P. Milici. Tractional motion machines: tangent-managing planar mechanisms as analog computers and educational artifacts. In *Unconventional Computation and Natural Computation*, volume 7445 of *LNCS*, pages 164–173. Springer, 2012b.
- P. Milici. A geometrical constructive approach to infinitesimal analysis: Epistemological potential and boundaries of tractional motion. In *From Logic to Practice*, pages 3–21. Springer, 2015.

- P. Milici. Tractional constructions as foundation of differential equations: ancient open issues, new results, possible fallouts. *Oberwolfach Reports*, 14(4): 3520–3523, 2017.
- P. Milici and R. Dawson. The equiangular compass. *The Mathematical Intelligencer*, 34:63–67, 2012.
- P. Milici and B. Di Paola. Geometrical-mechanical artefacts for managing tangent concept. *International Journal for Mathematics in Education, HMS i JME*, 4:486–492, 2012.
- K. S. Miller and B. Ross. *An introduction to the fractional calculus and fractional differential equations*. Wiley New York, 1993.
- O. Pedersen. Master John Perks and his mechanical curves. *Centaurus*, 8(1): 1–18, 1963.
- J. Perks. The construction and properties of a new quadratrix to the hyperbola. *Philosophical Transactions*, 25:2253–2262, 1706.
- J. Perks. An easy mechanical way to divide the nautical meridian line in Mercator’s projection, with an account of the relation of the same meridian line to the curva catenaria. *Philosophical Transactions*, 29(338-350):331–339, 1714.
- I. Podlubny. Geometric and physical interpretation of fractional integration and fractional differentiation. *Fractional calculus and applied analysis*, 5(4): 367–386, 2002.
- J. W. Polderman and J. C. Willems. *Introduction to mathematical systems theory: a behavioral approach*. Springer, 1998.
- J. Poleni. *Epistolarum mathematicarum fasciculus*. Typogr. Seminarii, 1729.
- F. L. Pritchard and W. Y. Sit. On initial value problems for ordinary differential–algebraic equations. *Gröbner Bases in Symbolic Analysis. Proceedings of the Special Semester on Gröbner Bases and Related Methods*. In: *Radon Series Comp. Appl. Math*, 2:283–340, 2007.
- J. F. Ritt. *Differential equations from the algebraic standpoint*, volume 14 of *Colloquium Publications*. American Mathematical Soc., 1932.
- J. F. Ritt. *Differential algebra*, volume 33 of *Colloquium Publications*. American Mathematical Soc., 1950.
- M. Rosenkranz, G. Regensburger, L. Tec, and B. Buchberger. Symbolic analysis for boundary problems: From rewriting to parametrized Gröbner bases. In U. Langer and P. Paule, editors, *Numerical and Symbolic Scientific Computing*, Texts & Monographs in Symbolic Computation, pages 273–331. Springer Vienna, 2012.
- B. Ross. A brief history and exposition of the fundamental theory of fractional calculus. In *Fractional calculus and its applications*, pages 1–36. Springer, 1975.

- B. Ross. The development of fractional calculus 1695–1900. *Historia Mathematica*, 4(1):75–89, 1977.
- L. A. Rubel. Solutions of algebraic differential equations. *Journal of differential equations*, 49(3):441–452, 1983.
- L. A. Rubel. A survey of transcendently transcendental functions. *American Mathematical Monthly*, 96(9):777–788, 1989.
- M. Salvi and P. Milici. Laboratorio di matematica in classe: due nuove macchine per problemi nel continuo e nel discreto. *Quaderni di Ricerca in Didattica (Mathematics)*, 23:15–24, 2013.
- A. Seidenberg. *An elimination theory for differential algebra*. University of California Press, 1956.
- C. E. Shannon. Mathematical theory of the differential analyzer. *Journal of Mathematics and Physics MIT*, 20:337–354, 1941.
- M. H. Tavassoli, A. Tavassoli, and M. R. Ostad Rahimi. The geometric and physical interpretation of fractional order derivatives of polynomial functions. *Differential Geometry–Dynamical Systems*, 15:93 – 104, 2013.
- D. Tournès. *La construction tractionnelle des équations différentielles*. Blanchard, Paris, 2009.
- K. Weihrauch. *Computable analysis: an introduction*. Springer Science & Business Media, 2000.