Computing envelopes in dynamic geometry environments

Francisco Botana,
Universidad de Vigo, Campus A Xunqueira, 36005 Pontevedra, Spain
fbotana@uvigo.es

Tomas Recio,
Universidad de Cantabria, Avda. Los Castros, 39071 Santander, Spain
tomas.recio@unican.es

September 29, 2015

Abstract

We review the behavior of standard dynamic geometry software when computing envelopes, relating these approaches with the various definitions of envelope. Special attention is given to the recently released version of GeoGebra 5.0, that implements a recent parametric polynomial solving algorithm, allowing sound computations of envelopes of families of plane curves. Specific details on this novel approach are provided in this paper.

Keywords: automated deduction in geometry, envelope computation, dynamic geometry

1 Introduction

Loci determination is unanimously accepted as a key characteristic of dynamic geometry (DG) systems. DG deals with geometric constructions that can be changed dynamically [1]. The locus ability is the second most present feature in the Wikipedia list of DG software[1]. Less attention has been paid both from DG developers and users to the computation of a particular kind of loci, namely, the envelopes of families of curves. Thus, from the developers side we observe the dominance of a simple strategy for envelope computation, consisting on tracing –for a sufficiently large number of instances– an element of the given family of curves, as the main tool for graphically suggesting the corresponding envelope. On their side, users seem to be comfortable with already existing, envelope computation related, tools in DG programs. At least we have not detected significant requests to enhance such tools, as it is the case concerning locus

[1] http://en.wikipedia.org/wiki/List_of_interactive_geometry_software

computation facilities[2]. Nevertheless, envelopes, apart from being a classical topic, involve different mathematical fields and can be used fruitfully in science teaching.

Throughout this paper we will only consider envelopes of families of plane curves.

The aim of this note is, first, to perform a systematic discussion of the construction of envelopes in the top four most widely distributed DG environments, namely Cabri[3], Cinderella[4], GeoGebra[5] and The Geometer's Sketchpad[6] (GSP). To this end, we recall basic notions about envelopes and refer to different reputed sources, some classical and some very modern, for further details on the elusive concept of envelope. Through selected examples and cases, we review how the above mentioned DG systems perform when computing envelopes. Then, we focus on describing the new command for envelope computation featured in GeoGebra 5.0. The algorithmic approach behind this command is presented, as well as some examples of its performance, along with a rough description of some essential limitations of the proposed method.

2 Formal definition of envelope

Usually, informal definitions of envelope highlight the idea of contact. For instance, an envelope is defined as *a curve which touches every member of a family of curves or lines*[7], or *a curve C such that C is tangent to every member of the set [of curves]*[8]. Even a collaborative source as Wikipedia begins its article on the concept of envelope defining it as *a curve that is tangent to each member of the family at some point*[9]. But soon, the encyclopedia introduces other alternative definitions, and mentions different set containments between them. Although someone could doubt on the soundness of the Wikipedia, in the envelope case its description faithfully follows the main ideas developed in more reputed sources as [2]. There, the *envelope* E_1 or *discriminant* of a family of curves in the real (x, y)-plane $F(x, y, t) = 0$, parametrized by $t \in \mathbb{R}$, is defined as the set[10]

$$E_1 = \{(x, y) \in \mathbb{R}^2 : \exists t \in \mathbb{R}, F(x, y, t) = \frac{\partial F}{\partial t}(x, y, t) = 0\}. \tag{1}$$

In the same reference, three other definitions of envelope are also introduced. For instance, a second notion, that of an envelope E_2 considered as the curve tangent to $F_t(x, y) = F(x, y, t)$, for each t. A new idea of envelope E_3 is also

[2] http://www.dynageo.de/discus/messages/14/232.html?1076183239

[3] http://www.cabri.com

[4] http://cinderella.de

[5] http://geogebra.org

[6] http://www.dynamicgeometry.com

[7] http://www-history.mcs.st-andrews.ac.uk/history/Curves/Definitions2.html#Envelope

[8] http://xahlee.info/SpecialPlaneCurves_dir/Envelope_dir/envelope.html

[9] https://en.wikipedia.org/wiki/Envelope_mathematics

[10] Note that we restrict the definition to families of plane curves. Furthermore, the notation here used is slightly modified with respect to [2].

presented as the limit of intersection points of nearby curves F_t. Finally, the notion of envelope E_4 is outlined as the boundary of the region filled by curves F_t. Furthermore, it is shown that $E_i \subset E_1$ for $i = 2, 3, 4$.

In all these cases, the given family of curves we would like to compute its envelope, is presented as depending on a single parameter t. Now, if the family of curves, as it usually happens in DG, depends on a point moving on some constructed path, users will deal not anymore with a uniparametric family, but with one involving two parameters, namely, the two coordinates of the mover point. In this case the family will be described by $F(x, y, t_1, t_2) = 0$, where parameters t_1 and t_2 are constrained by the restriction of point (t_1, t_2) to move along a one-dimensional path in the plane, that is, by adding an extra equation $g(t_1, t_2) = 0$. In this case the defining condition in envelope E_1 is to be replaced by

$$E_1 = \{(x, y) \in \mathbb{R}^2 : \exists t_1, t_2 \in \mathbb{R}, F(x, y, t_1, t_2) = g(t_1, t_2) = \frac{\partial F}{\partial t_1}\frac{\partial g}{\partial t_2} - \frac{\partial F}{\partial t_2}\frac{\partial g}{\partial t_1} = 0\}. \tag{2}$$

Finally, let us remark that, in general, more than two parameters could be involved in the construction of the moving path for the parametric point describing the family of curves. In such a case, given a n-parametric family $F(x, y, t_1, \ldots, t_n) = 0$, there will be, for sound DG constructions, exactly $n - 1$ constraints $g_1, \ldots, g_{n-1}$, and the envelope E_1 is the set of real solutions x, y of the system consisting of $F(x, y, t_1, \ldots, t_n) = 0, g_i(t_1, \ldots, t_n) = 0$, for $i = 1, \ldots, n - 1$, and the Jacobi determinant

$$\begin{vmatrix} \partial F/\partial t_1 & \ldots & \partial F/\partial t_n \\ \partial g_1/\partial t_1 & \ldots & \partial g_1/\partial t_n \\ \partial g_{n-1}/\partial t_1 & \ldots & \partial g_{n-1}/\partial t_n \end{vmatrix} = 0$$

for real values of $t_1, \ldots, t_n$.

3 Envelopes in dynamic geometry software

3.1 Envelopes as boundaries of plane regions

Definition E_4 has been traditionally used in DG systems to suggest envelopes. Since tracing an element in any environment is a quite basic task for DG programs, given a family of curves, selecting a family member and activating its trace, a user can inspect the part of the plane swept by the curve and get a feeling of the envelope. Consider, for instance, a circle with center $(0, 0)$ and radius 4, the point $A(3, 0)$, and a point B moving on the circle. Draw the line AB and its perpendicular line through B. Activate the trace of this last line and drag B along the circle. The user will get, in any of the four DG systems enumerated in the Introduction, a screen similar to the one in Figure 1.

Nevertheless, stating that the graphically displayed envelope is, in fact, an ellipse, will usually be out of bounds for inexperienced users. With envelope

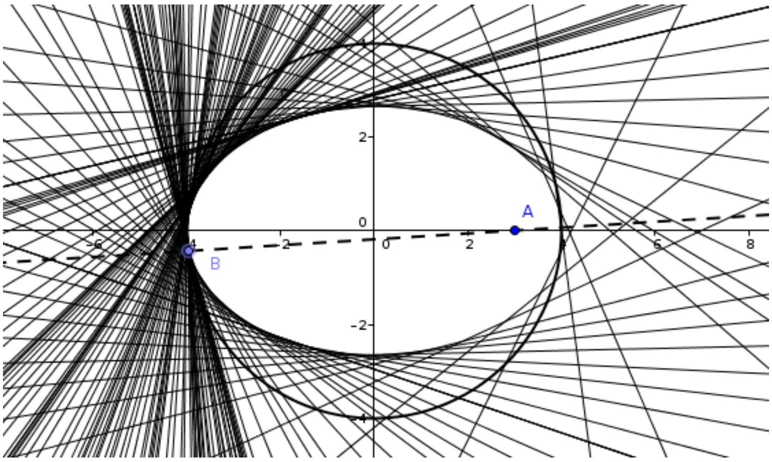


Figure 1: A family of straight lines enveloping an ellipse.

computation in DG systems mostly limited to this tracing approach, even teachers could ignore in some cases which is the precise curve that corresponds to the displayed envelope. Although this situation can be well suited for discovery tasks, the concern here is the automatic computation inside the system. And, since there is not a curve as result, this method for envelope computation sounds unpromising. Even worse, sometimes the boundary can be hard to visualize, ill–defined or non existent at all, adding difficulties for the user to draw any specific conclusion. As an illustration of this latter case, let us reconsider the construction above by designing A to lie in the circle, say $A(4,0)$, and then trace again the line AB. Now the trace is the whole plane (Figure 2), and thus –after some reflection– we should conclude that the envelope is empty, according to definition E_4. Moreover, the purely graphical approach does not provide an argument for the noticeable fact that the traced lines seem to concentrate on the circle point opposed to A, turning this point –for non expert users – into a potential candidate for the envelope. This fact is not fortuitous, as we will see below.

GeoGebra and GSP do not incorporate any refinement of this envelope-computation-by-tracing strategy. At least in these programs a user can manually sample the one-dimensional path of the parametric point and trace a family curve for each sample point, in order to get a more pleasant visualization of the envelope. A GeoGebra command for this task is **Sequence**[11], as it has been recommended in the GeoGebra forum, where a request for enhancing envelope computations has been posted[12]. On their side, both Cabri and Cinderella automate this process by using their **Locus** command. Although this command will be further discussed in a following subsection, let us here pay special attention

[11] http://wiki.geogebra.org/en/Sequence_Command

[12] http://www.geogebra.org/forum/viewtopic.php?f=2&t=20103

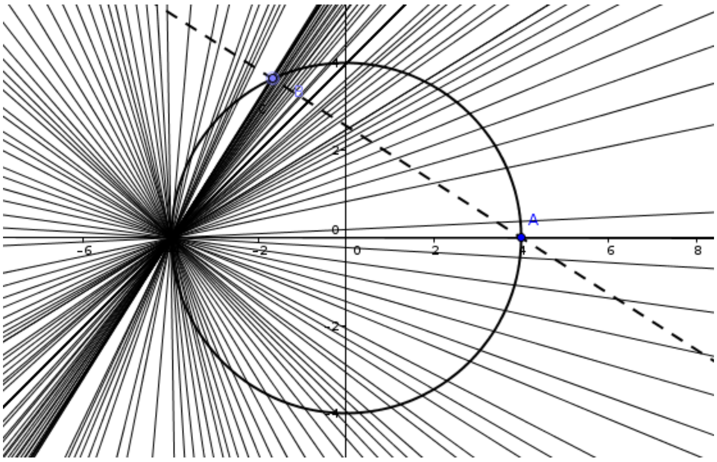


Figure 2: The lines sweep the whole plane.

to its behavior in the envelope computation context.

Thus, the standard DG approach to find the geometric locus of a (tracer) point depending somehow on another (mover) point consists of automatically sampling the path of the mover, and, for each sample, compute the actual position of the corresponding tracer point. The list of computed points is then plotted, showing the sought trajectory (see [3] for a full description of loci finding strategies in DG). Replacing the tracer point by a tracing curve, Cabri and Cinderella commands are then used to plot a collection of curves in order to suggest the envelope. The main difference with respect to the manual approach lies on the sampling mechanism, which can incorporate sophisticated methods to divide the parameter path. A direct translation of this idea is thus used by Cabri and Cinderella when dealing with envelopes. Facing the envelope of ellipses with foci $A(4, 0)$ and $B(0, t)$, where B is constrained to a segment such that $-3 \leq t \leq 3$, and major axis 5, Cinderella will output as result a 2–dimensional region (Figure 3, left). Cabri, if using a macro to construct ellipses from foci and an ellipse point, returns a similar region. Nevertheless, changing its Preferences in the Options menu by selecting Envelope and increasing the Number of objects in locus to 500, the border of the region appears (Figure 3, right), showing a more accurate result.

3.2 Envelopes as limit of intersections of nearby curves

Although no current DGS strictly features automatic envelope computation following this approach, it could be implemented by computing the intersection of a curve of the family with another (close) curve, and finding the locus of these intersection points, that will define the envelope. GSP, although not having a specific command to deal with envelopes, gives some related advice about

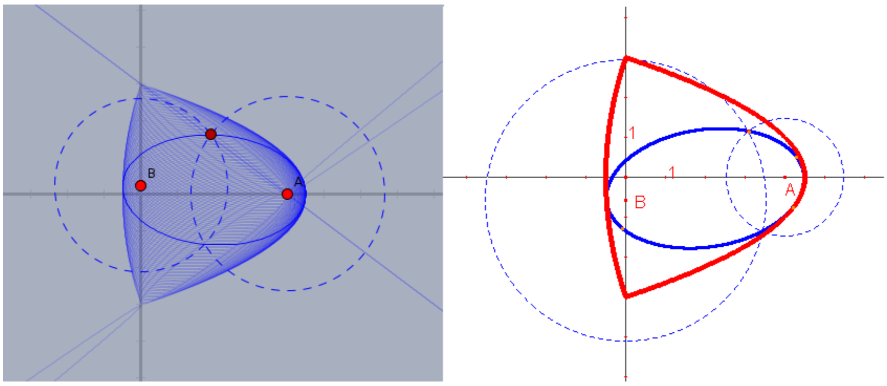


Figure 3: The envelope of a family of ellipses in Cinderella (left) and Cabri (right).

constructing envelopes[13]. There, the statement *a geometric envelope can be thought of as the limit or edge of the locus of a line or a circle* is succesfully applied to find some envelopes. This protocol catches the concept behind definition E_3, and it is a natural way to get the envelopes as curves. Yet there are some difficulties that we will summarily describe through an example. For instance, let us consider the construction from Figure 1, where the intersection of two close perpendicular lines will be a point whose locus, in turn, will be the sought envelope when both curves are infinitely close. Since taking such a limit is unfeasible in current DGs, we conclude that we can only find an approximate envelope through this strategy. Figure 4 shows a Cinderella construction where the intersection D of two family lines is controlled by dragging point C. A segment ending at C determines the distance between circle points B and B', and, thus, the selected pair of lines. The locus of D when B moves along the circle approaches the envelope as the segment length approaches to 0.

If we intend to go beyond this approximate construction and if we (wrongly) attempt to consider the limit case by selecting a 0-length segment, then no locus is plotted, although Cinderella reports that such locus exists in its Construction Text window. Since, in this construction, the locus has an identifying equation, $0 = 0$, we should understand that the computed locus is the whole plane! This answer, as a locus, is the correct one for the 0-length case, because we are asking about the position of any point in the perpendicular line while this line sweeps the plane. Obviously this is not the correct computation for the envelope, since the limit of the intersection of two lines, when the lines get arbitrarily close, is not the intersection of a single line with itself, as it happens here in the 0-length case. Thus, the discussed strategy is not suitable for envelope computation and can, at most, as a by-product of the approximate computation, reinforce the belief of the user on an ellipse being the envelope.

[13] http://www.dynamicgeometry.com/Technical_Support/FAQ/Constructions_and_Use/Envelope_Constructions.html

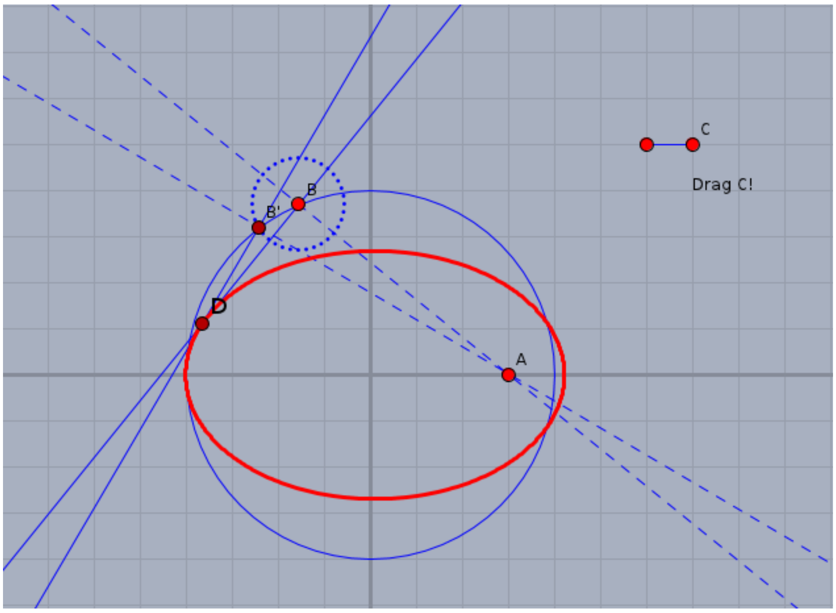


Figure 4: An approximate envelope computed with Cinderella.

As a side remark, let us notice that correctly computing the limit of intersection points, i.e. the *characteristics points*, requires considering, first, the family $F_t(x, y) = 0$ and, then, considering the distance of the points of one of the curves to a nearby one $F_{t+dt}(x, y) = 0$, such that this distance is greater than dt. That is, finding those F_t-ordinary points satisfying that

$$F(x, y, t) = 0, \frac{\partial F}{\partial t}(x, y, t) = 0. \tag{3}$$

In summary, the characteristic points of F_t are the ordinary points of F_t such that their distance to F_{t+dt} is an infinitesimal of higher order than dt [4, p. 37], which are the (x, y) values of the solutions of the above system.

Clearly, computing the loci of characteristic points returns correct envelopes, as it has been illustrated in [5]. But computing the characteristic points it is not an easy task. For instance, it should be noted that real characteristic points are not always limit of real intersection points. A canonical example in [4] illustrates this issue by considering a curve as the envelope of its osculating circles. These circles have not real intersection for infinitely close points. Their intersections are imaginary conjugated whereas their limit is a real point in the curve.

We do not discuss envelopes following definition E_2 because the concept of tangency is subsumed by the idea of studying intersections of nearby curves.

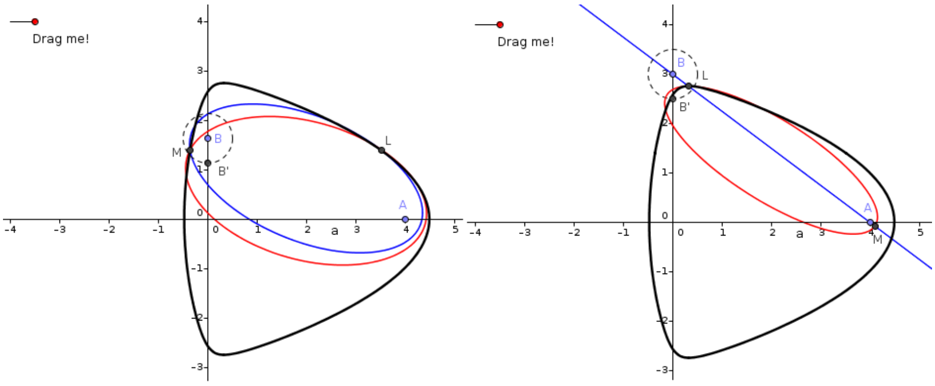


Figure 5: An approximate envelope computed through definition E_3 in GeoGebra.

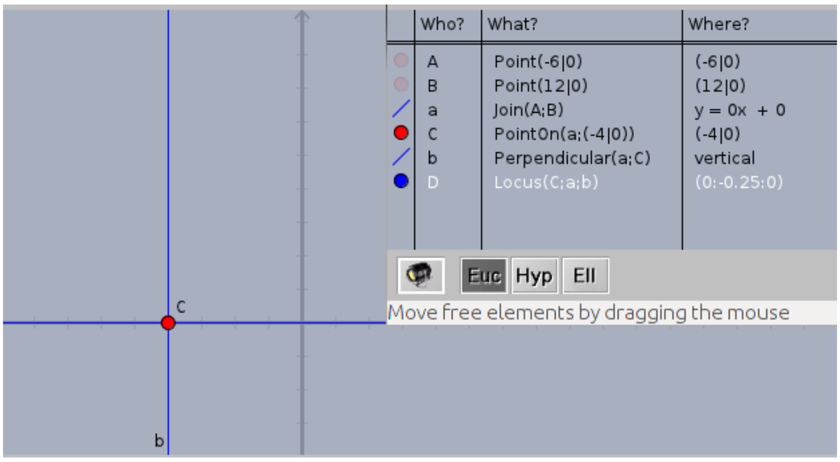
3.2.1 An illustration concerning differences between definitions E_3 and E_4

Let us reconsider the envelope of ellipses with foci $A(4,0), B(0,t), -3 \leq t \leq 3$, and major axis 5 (see Figure 3). The above approach of computing the loci of intersection points of nearby curves shows a result which is similar to the one obtained through the approach to the envelope computation as the boundary of some region traced by the curves of the given family. But there are some, albeit small, important differences. Figure 5, left, illustrates the approximate envelope when the distance between parametric points B, B' is set to 0.5. There, the envelope is computed as the loci of intersection points L, M of blue and red ellipses corresponding to positions B and B' of the variable focus. Note that if B is at an extremum position, say $(0,3)$, the blue ellipse degenerates into a double line (Figure 5, right). The intersections of this fixed (although degenerated) ellipse with nearby ones are two points: L, lying on the approximated envelope, and M, which is not part of it. Approaching B' to B (by dragging the red point in the construction), we get a sequence of points M, all lying in the degenerated ellipse, with limit A. In other words, A is an isolated characteristic point, and, as such, A should be also considered as part of the envelope.

3.2.2 Contradictory results in Cabri and Cinderella

Let $F(x,y,t) = x - t$ be a family of vertical lines in the plane. Its envelope, according to definition E_1, consists of the real points satisfying $\{x - t = 0, -1 = 0\}$, i.e. the empty set. Besides the mentioned set containment $E_i \subset E_1, i = 2,3,4$, it is clear that this family occupies the entire plane, so the E_4 envelope must be empty, as expected. On the other hand, the lines do not intersect, and, thus, the E_3 envelope is also empty.

The Envelope command in Cabri does not return any object for this family, elementary constructed with a semifree point moving along the x axis and the



	Who?	What?	Where?
○	A	Point(-6|0)	(-6|0)
○	B	Point(12|0)	(12|0)
/	a	Join(A;B)	y = 0x + 0
●	C	PointOn(a;(-4|0))	(-4|0)
/	b	Perpendicular(a;C)	vertical
●	D	Locus(C;a;b)	(0:-0.25:0)

Move free elements by dragging the mouse

Figure 6: The point at the infinity as envelope of vertical lines in Cinderella.

perpendicular lines through these points. Nevertheless, Cinderella, although not showing any new graphic element, computes a point as result of its Locus command (recall that the Cinderella command claims to return envelopes if selected a point on a linear path and a curve parametrically depending on the point). This new point C (see Figure 6) is a point at the infinity, so we guess that Cinderella follows, partially at least, definition E_3 for envelope searching. Note also that Cinderella provides native support for other geometries than Euclidian. So, considering the point at infinity as intersection of parallel lines seems to be natural inside this system.

Things change for Cabri if we restrict the values of the parameter. Instead of being unconstrained, we consider now a point gliding on the unit circle, and vertical lines passing through it. Now, the graphic envelope is correctly drawn. However, asking for the equation of the found envelope we get a wrong result (Figure 7). Cabri ability for returning loci equations, up to degree six, is not robust enough to deal with this simple envelope, which can be described by the conjunction of two equations. Cinderella, in turn, is not able to detect the new situation, returning again the point at the infinity.

3.3 Envelopes as discriminants

Neither if tracing some curves of a given family, nor when computing the locus of intersection points of nearby family curves, do DG environments provide, in general, further information about envelopes. For instance, a user gets no idea at all about the expression of the computed curve(s) in the envelope displayed of Figures 3 and 5. Even simple envelopes, such as the one in Figures 1 and 4, cannot be easily managed if ignoring their analytical expressions. Thus, a naive GeoGebra user, once convinced he/she is dealing with an ellipse, could attempt to construct five points on it and use the Conic command in order to

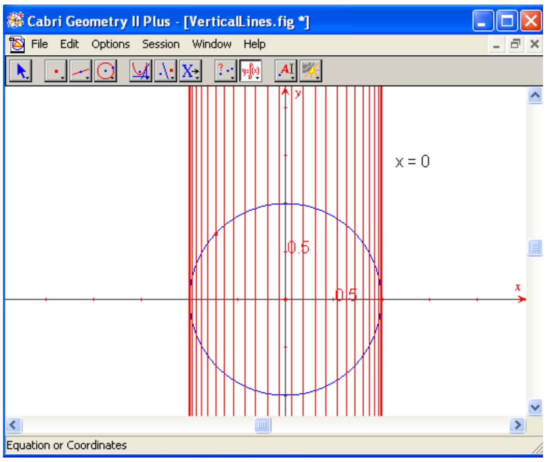

Cabri Geometry II Plus - [VerticalLines.fig *]
File Edit Options Session Window Help
x = 0
y
x
0.5
0.5
Equation or Coordinates

Figure 7: The correct graphic envelope in Cabri and its wrong equation.

obtain the corresponding analytic equation. But, as it is well–known, numerical inaccuracies will severely determine the final equation. Even worse, if there is no sound conjecture about the actual degree of the graphically presented curve, this interpolation procedure is just a non reliable guessing.

The reader will have observed that, for the construction we are discussing, Cinderella correctly labels as an ellipse the obtained locus, and that its equation is accessible in the Construction Text window. Unfortunately, this characteristic is not universal in Cinderella. When searching for loci, the system gets a list of locus points after sampling the path. This list is tested in order to see if their members are in a line, circle or general conic. This test is applied not only to the current instance of the construction, but to an extensive number of similar instances, in a probabilistic setting. If the tests are successfull, the equation of the line, circle or conic is returned for describing the object and for easier rendering. Concerning envelopes, it seems that Cinderella uses definition E_3, thus reducing envelopes to loci. Furthermore, intersections are only computed in the case of families of straight lines. If the curves of the family are different, the strategy is limited to a smart sampling of the parameter path and plotting some curves of the family.

No standard DG system is currently able to automatically compute equations of loci or envelopes in a sound manner, i.e. by using definition E_1 of envelopes as discriminants. In this direction let us mention that there has been some proposal about linking DG environments with Computer Algebra Systems (CAS) in order to enrich the former with symbolic capabilities. Also, academic prototypes have been developed showing the feasibility of these ideas. As a sequel of [6], JSXGraph, a Javascript library for DG computations, and GeoGebra, incorporated a symbolic approach allowing the knowledge of loci equations. A similar proposal related to envelopes was described in [7]. Both works were based on algebraic elimination via Gröbner bases, used special CAS packages, such as CoCoA and Singular, and implemented a protocol to communicate them with a DG system. The idea, for envelopes, is straightforward: construct in the DG system the family of curves depending on a point, translate the family and the point to an adequate algebraic description, and apply definition E_1 via polynomial elimination. Once the elimination is done, incorporate the result into the geometric environment as a new object whose equation is now known.

Let us note that this procedure is only defined for the purely algebraic setting (that is, limited to dealing with constructions that involve just equality constraints). So, no guarantee of success exists if applying it to semialgebraic constructions (i.e. if inequalities are involved). For instance, the above mentioned family of moving ellipses with a variable focus can be not considered as an algebraic one, since the focus is constrained to belong to a segment and, thus, one of the variables is subject to inequalities expressing it should be limited by the extremes of the segment. Lacking of an efficient proposal for dealing with the semialgebraic case, let us consider this family in a purely algebraic fashion, even if then its description includes some other, unexpected, curves. Thus, let us apply the standard construction for an ellipse of foci at points $A(4,0)$ and $B(0,t)$ and major axis of length 5, yielding that an ellipse of the family is described by

$$4y^2t^2-4yt^3-36x^2-100y^2+t^4-32xyt+16xt^2+164yt-82t^2+144x+81=0, \quad (4)$$

and the elimination of this polynomial and its partial derivative with respect to t is, once factorized,

$$(2x+y^2-9)(18x-y^2+9)(x^2-8x+y^2+16) \quad (5)$$

that is, two parabolas and the point $A(4,0)$. We observe that point A, also found following definition E_3, is, as expected, also included into the output for envelope E_1. Here, the new knowledge provided by this algebraic approach is about the borders of the region where the ellipses move, which are now clearly defined by the pair of parabolas. Note that these parabolas envelope not only the ellipses, but also do with the hyperbolas that the family equation defines when the variable focus B lies out of the segment, i.e. when $t<-3$ or $t>3$.

Besides the algebraic vs. semialgebraic setting, the elimination approach deals with a second difficulty, since it does not return projections, but their closure in a Zariski topology, i.e. it always returns algebraic sets and not just some parts of them. Yet, our definition of E_1 explicitly requires a projection (as expressed by the $\exists$ symbol included in the definition). Thus, when looking at a family of curves $F(x,y,t)=0$ as a surface, its envelope is the surface projection on the (x,y)–plane, and eliminating t we get an algebraic variety containing perhaps strictly the sought projection, since the algebraic variety can include some spurious points or sets that are not part of the envelope. Finally, degenerate instances of the construction can also introduce, through their projection, other spurious factors into the elimination result. Consider, for example, finding the deltoid as the envelope of Simson-Wallace lines. Given a triangle ABC and a point P moving on its circumcircle, construct the orthogonal projections of P on sides AB and AC, M and N, and find the envelope of lines MN when P describes the circle. As shown in [7, p. 8], the elimination result includes a quartic curve, the Steiner deltoid, and an extra linear factor which corresponds to a degeneracy. When P and A coincide, both points M and N are identical, and no line MN is defined. Thus, the case of $P = A$ should be excluded in order to get a sound envelope. An expert user could explicitly remove this degeneration when defining the polynomial system for elimination by using the trick of Rabinowitz, but currently there is no way to automatically detect them.

4 A symbolic approach to envelope computation in GeoGebra 5.0

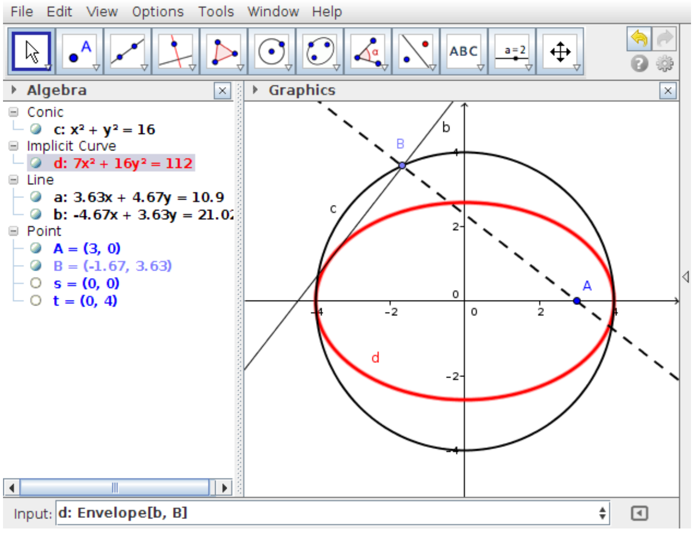
As shown in the previous section, there are different problems that need to be circumvented concerning the elimination-based approach to envelope computation. An improvement in this direction is the new **Envelope** command in the desktop version of GeoGebra 5.0[14]. Following some ideas by the authors of the present paper, among others [3], instead of performing standard ideal-theoretic elimination, the polynomial system is seen as a parametric one, where here we think that x, y are *parameters* and t or $t_1, \ldots, t_n$ are variables (note the change of the usually assigned meaning). Thus, parametric polynomial system solving can be applied to study the solutions for given parameter values. Being more specific, GeoGebra uses the recent GröbnerCover algorithm [8] to discuss and classify the system attending to its number of solutions (finite, infinite and no solution). The envelope of Figure 1, for $A(3, 0)$, is computed by this algorithm as the difference of two varieties

$$\mathbb{V}(7x^2 + 16y^2 - 112) \setminus \mathbb{V}(9y^2 + 49, 3x + 16). \tag{6}$$

Note that points $(-16/3, \pm 7i/3)$ should be excluded for the envelope output; nevertheless, being non-real points, just the ellipse $7x^2 + 16y^2 = 112$ is returned as an implicit curve (Figure 8) and GeoGebra shows a correct result. If the points to exclude were real, GeoGebra would currently return the complete ellipse without removing any point, because its data structure is not able to manage constructible sets, i.e. sets described as difference of varieties. Users must be aware of this fact, as well as that internal computations are performed in the complex field, not the real one, that is, what is actually computed is the *complex, algebraic* envelope, not the real or semialgebraic one.

Dragging A to $(4, 0)$ the algebraic computation of the envelope outputs now the equation $x^2+8x+y^2 = 16$ and no plot is shown. Note that the curve reduces now to the real point $(-4, 0)$, and that current plotting features of GeoGebra are not able of graphing such (degenerated to a point) implicit curves. For such an instance of this envelope construction, solving the associated parametric polynomial system we get two types of solutions:

[14] http://download.geogebra.org/installers/5.0



Conic

c: x² + y² = 16

Implicit Curve

d: 7x² + 16y² = 112

Line

a: 3.63x + 4.67y = 10.9

b: -4.67x + 3.63y = 21.0

Point

A = (3, 0)

B = (-1.67, 3.63)

s = (0, 0)

t = (0, 4)

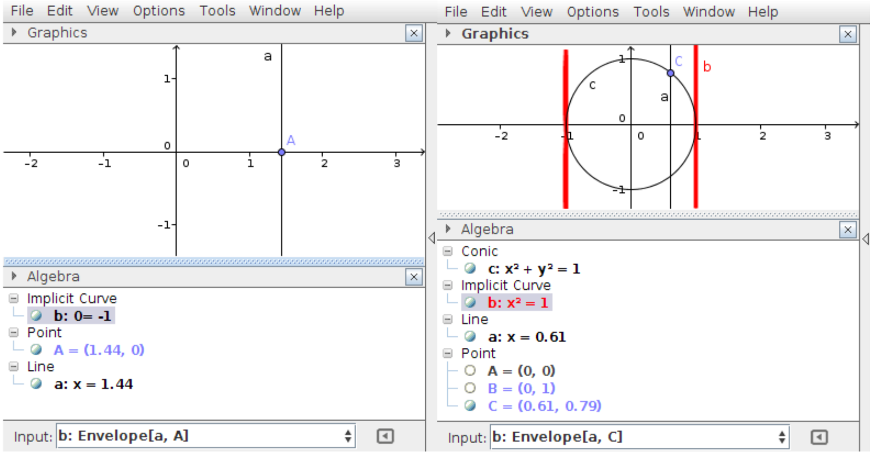
Input: d: Envelope[b, B]

Figure 8: An envelope correctly computed in GeoGebra5.0.

- $t_1 = 4$ and $t_2 = 0$ for parameter values in $\mathbb{V}(y) \setminus \mathbb{V}(y, x + 4)$, and
- $t_1^2 + t_2^2 = 16$ for parameter values in $\mathbb{V}(y, x + 4)$.

Note that the first one corresponds to a degeneration (when point $B(t_1, t_2)$ coincides with $(4, 0)$, being undefined the line AB and so its perpendicular). Since the line $y = 0$, a 1–dimensional object, corresponds to a 0–dimensional set of values (t_1, t_2), this solution is to be discarded. On the contrary, the second part exhibits the opposite behavior. For any other position of B in the circle than $(4, 0)$, we get $x = -4, y = 0$. Having in mind the importance of caustic curves for the historical development of the theory of envelopes (see [9]), one would say that the family lines *burned* the point $(-4, 0)$.

As this example roughly shows, it is crucial for envelope computation to compare dimensions of solution sets for different pieces of the parameter space. Thus, the envelope protocol currently followed by GeoGebra performs such comparison, accepting as (part of the) envelope the parameter values that yield to solutions of dimension verifying certain conditions, involving sophisticated algebraic computations [10]. Think, for instance, that there are constructions where an envelope factor has dimension 1 and it comes from a finite set of t_i values, being not a degeneracy. Recall, for example, the case of the envelope of the family of vertical lines. We concluded in a preceding section that this family has no envelope at all, since the contradiction $-1 = 0$ is included into the system one has to solve to compute the envelope. GeoGebra answers as expected, returning the empty implicit curve $0 = -1$ (Figure 9, left). But if the family of vertical lines is now defined as the collection of vertical lines passing through a moving point in the unit circle, the envelope is not anymore empty


File Edit View Options Tools Window Help
Graphics
a
A
1
0
-1
-2
-1
0
1
2
3
Algebra
File Edit View Options Tools Window Help
Graphics
c
C
b
a
1
0
-1
-2
-1
0
1
2
3
Algebra

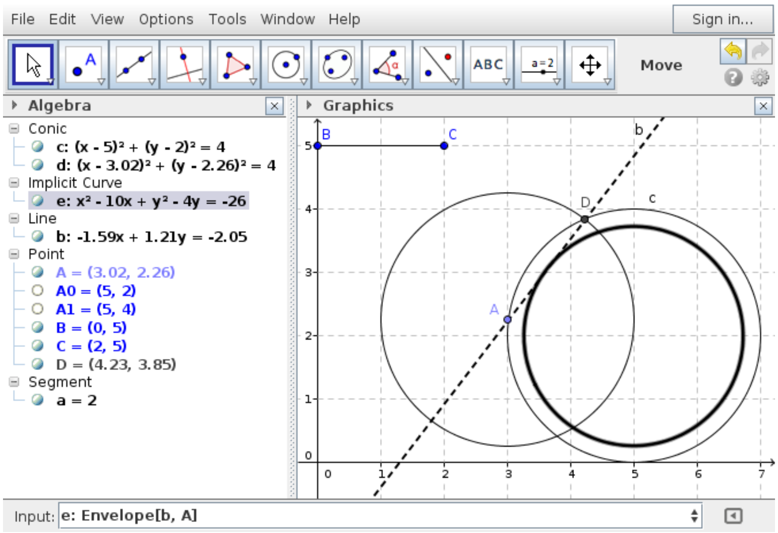
Implicit Curve
$b: 0 = -1$
Point
$A = (1.44, 0)$
Line
$a: x = 1.44$
Input: b: Envelope[a, A]

Conic
$c: x^2 + y^2 = 1$
Implicit Curve
$b: x^2 = 1$
Line
$a: x = 0.61$
Point
$A = (0, 0)$
$B = (0, 1)$
$C = (0.61, 0.79)$
Input: b: Envelope[a, C]

Figure 9: A family of lines without envelope (left), and the envelope of vertical lines passing through a point in the unit circle (right) computed with GeoGebra version 5.0.18.

but, at least, it contains the lines $x = 1$ and $x = -1$, following definition E_4. Computing in this case the envelope E_1 with the GröbnerCover algorithm (for parametric polynomial system elimination [3]), we get $t_1 = x$ and $t_2 = 0$ for parameter values in $\mathbb{V}(x-1) \cup \mathbb{V}(x+1)$. That is, the 1–dimensional set of lines $x = 1$ and $x = -1$ corresponds to $t_1 = \pm 1$ and $t_2 = 0$. According to the current protocol implemented in GeoGebra, both lines should be removed from the envelope, thus giving a wrong result. Note that none of the lines comes from a degeneration, but they are ordinary members of the family for $(\pm 1, 0)$ positions of the moving point in the unit circle. As a consequence, the criterium on the dimensions of variables and parameters spaces needs to be filtered by studying whether an envelope part emerges from degeneration. This filter is not available in current versions of GeoGebra 5.0. Depending on the selected version, users will get either such components (Figure 9, right), or nothing. We expect to find this new approach included and thoroughly documented in a forthcoming subversion of GeoGebra.

GeoGebra users have been advised about being aware of the complex (as opposite to real) character of the computed envelopes in the current version of GeoGebra. At the beginning of this Section a discussion involving the envelope of Example 1 shows that the final result lies in the complex field. The curve is an ellipse without two imaginary points, so being also correct in the real plane. Nevertheless, it is easy to find constructions where working within the complexes leads to cognitive difficulties for non experts users. Consider, for instance, the following construction: a point A sliding in a circle c is the center of another circle with radius BC, and D is a common point to both circles. The envelope of lines AD is the inner circle in Figure 10, with equation $x^2 - 10x + y^2 - 4y = -26$. The original circle and the one centered at A do not share real points



Conic
c: $(x - 5)^2 + (y - 2)^2 = 4$
d: $(x - 3.02)^2 + (y - 2.26)^2 = 4$
Implicit Curve
e: $x^2 - 10x + y^2 - 4y = -26$
Line
b: $-1.59x + 1.21y = -2.05$
Point
A = (3.02, 2.26)
A0 = (5, 2)
A1 = (5, 4)
B = (0, 5)
C = (2, 5)
D = (4.23, 3.85)
Segment
a = 2

Input: e: Envelope[b, A]

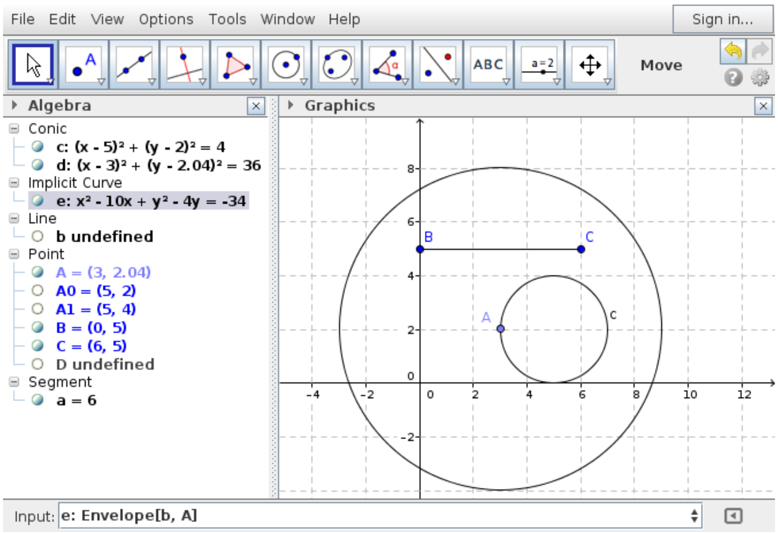
Figure 10: A circle as the envelope of lines AD when A moves along circle c.

whenever length of BC is greater than 4, but the Implicit Curve object denoting the envelope persists in the Algebra window of GeoGebra as a complex circle (Figure 11), despite the environment labels the intersection point D and the line AD as *undefined*. There is no way to remove such inconsistencies while using GröbnerCover to compute envelopes. Future work to deal only with real solutions will imply the use of techniques from real algebraic geometry, such as Cylindrical Algebraic Decomposition.

A final issue related to GeoGebra computations deals with timing constraints. Since GröbnerCover uses sophisticated algebraic algorithms, some envelopes can require an amount of time which is clearly excessive for a true interactive system. The construction in Figure 12 illustrates such a case. The envelope of dotted lines when C moves along line AB cannot be determined within the system default time (5 seconds), returning *undefined* as result. A future GeoGebra version should employ concurrent tactics. Given an envelope to be computed, it should simultaneously perform classical elimination and GröbnerCover. If the slot time for GröbnerCover is not fulfilled, return the classical result, stating that spurious factors could perhaps be included in the output. Figure 12 includes both results: e is the unsuccessful output of the envelope computation through GröbnerCover, and f is the envelope computed through standard elimination.

5 Conclusion

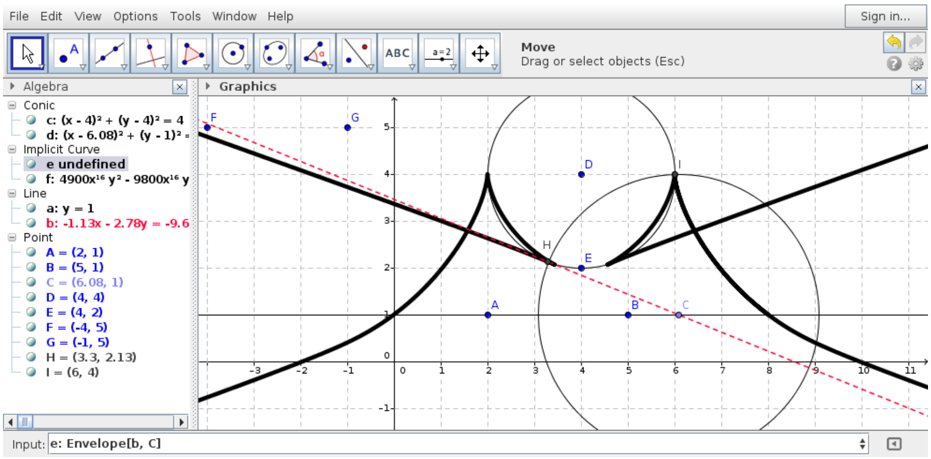
We have reviewed abilities and shortcomings of standard dynamic geometry environments such as Cabri, Cinderella, GeoGebra and The Geometer's Sketchpad



```
Conic
  c: (x - 5)² + (y - 2)² = 4
  d: (x - 3)² + (y - 2.04)² = 36
Implicit Curve
  e: x² - 10x + y² - 4y = -34
Line
  b undefined
Point
  A = (3, 2.04)
  A0 = (5, 2)
  A1 = (5, 4)
  B = (0, 5)
  C = (6, 5)
  D undefined
Segment
  a = 6
```

Input: e: Envelope[b, A]

Figure 11: The envelope is still a complex circle $x^2 - 10x + y^2 - 4y = -34$ despite there is no family of real lines.



Move
Drag or select objects (Esc)

```
Conic
  c: (x - 4)² + (y - 4)² = 4
  d: (x - 6.08)² + (y - 1)² =
Implicit Curve
  e undefined
  f: 4900x¹⁶ y² - 9800x¹⁶ y
Line
  a: y = 1
  b: -1.13x - 2.78y = -9.6
Point
  A = (2, 1)
  B = (5, 1)
  C = (6.08, 1)
  D = (4, 4)
  E = (4, 2)
  F = (-4, 5)
  G = (-1, 5)
  H = (3.3, 2.13)
  I = (6, 4)
```

Input: e: Envelope[b, C]

Figure 12: The envelope of lines b cannot be computed with GröbnerCover (e) but standard elimination returns an equation (f) containing it.

when dealing with envelopes of families of plane curves. These approaches are related to the different envelope definitions. Special attention has been given to describe the current efforts concerning the recent GeoGebra version where remote symbolic computations are performed to output more reliable envelopes and with more associated information, both in geometric as well as in analytic terms.

Acknowledgement

The authors have been partially supported by the Spanish “Ministerio de Economía y Competitividad” and the “European Regional Development Fund” (FEDER), under the projects MTM2011–25816–C02–02 and MTM2014–54141–P.

Conflict of interests

The authors declare that there is no conflict of interests regarding the publication of this article.

References

[1] Gao, X S. Automated geometry diagram construction and engineering geometry. Lect Notes Comp Sci, 1999, 1669: 241–267

[2] Bruce J W, Giblin P J. Curves and Singularities. Cambridge: Cambridge University Press, 1984

[3] Abánades M A, Botana F, Montes A, Recio T. An algebraic taxonomy for locus computation in dynamic geometry. Comp Aided Des, 2014, 56: 22–33

[4] Julia G. Éléments de Géométrie Infinitésimale. Paris: Gauthier–Villars, 1936

[5] Amir–Moéz A R. Synthetic approach to the theory of envelopes. Am Math Month, 1957, 64: 265–268

[6] Botana F, Valcarce J L. A dynamic–symbolic interface for geometric theorem discovery. Comput Educ, 2002, 38: 21–35

[7] Botana F, Valcarce J L. Automatic determination of envelopes and other derived curves within a graphic environment. Math Comp Simul, 2004, 67: 3–13

[8] Montes A, Wibmer M. Gröbner bases for polynomial systems with parameters. J Symb Comp, 2010, 45: 1391–1425

[9] Scarpello G M, Scimone A. The work of Tschirnhaus, La Hire and Leibniz on catacaustics and the birth of the envelopes of lines in the 17th century. Arch Hist Exact Sci, 2005, 59: 223–250

[10] Botana F, Recio, T. Some issues on the automatic computation of plane envelopes in interactive environments. Math Comp Simul, *in press*