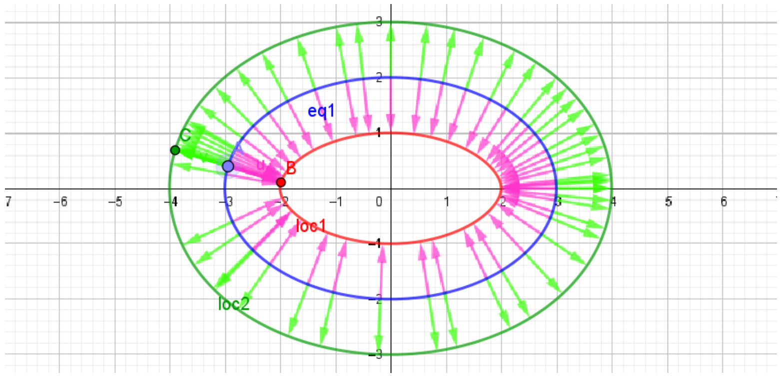
בס"ד

The Roland and Astrid Dana-Picard Chair for Mathematics, Education and Judaism

JERUSALEM COLLEGE OF TECHNOLOGY

LEV ACADEMIC CENTER



Singular points of envelopes and offsets: mathematical and educational challenges

THIERRY (NOAH) DANA-PICARD

JERUSALEM COLLEGE OF TECHNOLOGY AND JERUSALEM MICHLALA COLLEGE

ADG 2025 CONFERENCE – STUTTGART, GERMANY

AUGUST 1, 2025

Main issues I met with students

Mathematical issues

- Algebraic translation of geometry (e.g., Cartesian coordinates, polynomials)
- Symbolic computation (CAS)
- Constraint-based reasoning

Educational challenges

- Undergraduates: a weak CAS-DGS literacy (if any)
- Graduate student: some CAS-DGS literacy but generally no background to understand the algebraic methods provided by the CAS
- Which software is available: general policy of the institution (see Artigue 2002)

Educational frames

- Teacher training college:
  - One term course for undergraduates
  - One term course for graduate students
  - Of course, the mathematical contents are different, with a non empty intersection
  - Includes some creative work (in a STEAM approach)

- Excellency program at JCT:
  - A student works as a collaborator of a researcher
  - A large part of self-teaching (maths and technology)
  - A paper already published

A special case – a student entering research

- The framework: JCT excellency program
  - The student's background:
  - Started 2nd undergraduate year in engineering (CS)
  - Some literacy with mathematical software
  - A strong interest in learning and experimenting
  - Able to show initiatives
- The achievements:
  - New knowledge in maths: Polynomial rings, Gröbner bases, Envelopes, Offsets
  - Technological literacy: GGB, GGB-D and Maple
  - Developed a dialog with researchers (among them Zoltan K.)

What I can describe

- Three “experiments” at different paces:
- The “excellent student” worked at his own pace (partly imposed by JCT, I am not so demanding)
- He learnt, and then “produced novelties”
- Two classes of in-service teachers learning towards a 2nd degree M.Ed.
- In 2024-2025, two classes in a teacher training college:
  - Fall term – 13 weeks. Each week a 2-hour online meeting (2 of them were face-2-face in class): 22 ladies – strong motivation and achievements
  - Spring term: Concentrated in 3 weeks, only once face-2-face in class, the main part online: 5 men. Strong motivation, weaker achievements.

Envelopes of 1-parameter families of plane curves

Envelopes - Synthetic-limit definition

Let $\{\mathcal{C}_k\}$ be a family of plane curves. The characteristic $\mathcal{M}_k$ is the limit of the intersection of $\mathcal{C}_{k+h}$ with $\mathcal{C}_k$ when h ->0 . The union of the characteristics is called an envelope of the family $\{\mathcal{C}_k\}$.

Not easy to apply
Implies the analytic definition

Envelopes – Impredicative definition

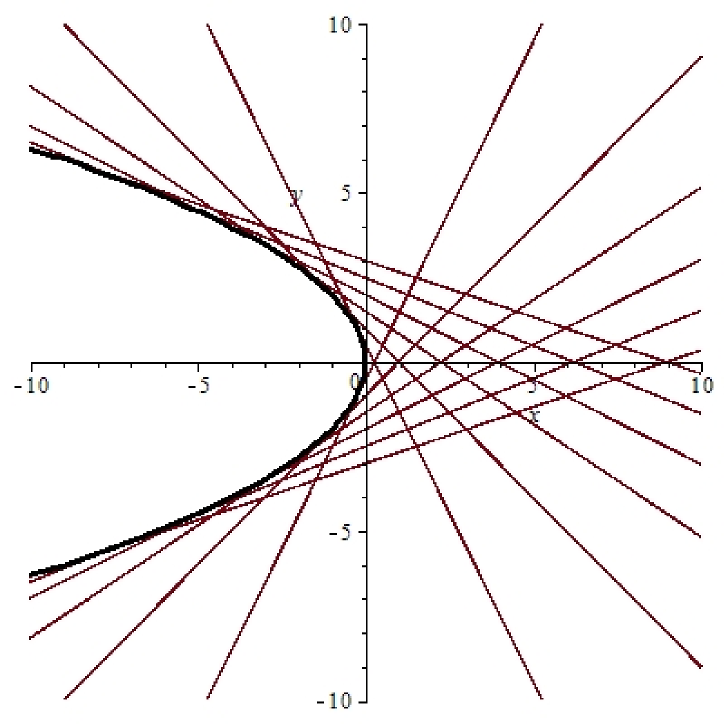
$\mathcal{E}$ is a curve with the property that at each of its points, it is tangent to a unique curve from the family $\{\mathcal{C}_k\}$. (Also, $\mathcal{E}$ should touch each $\mathcal{C}_k$)

$$Lines: x + ky = k^2$$

$$Envelope: y^2 = 4x$$

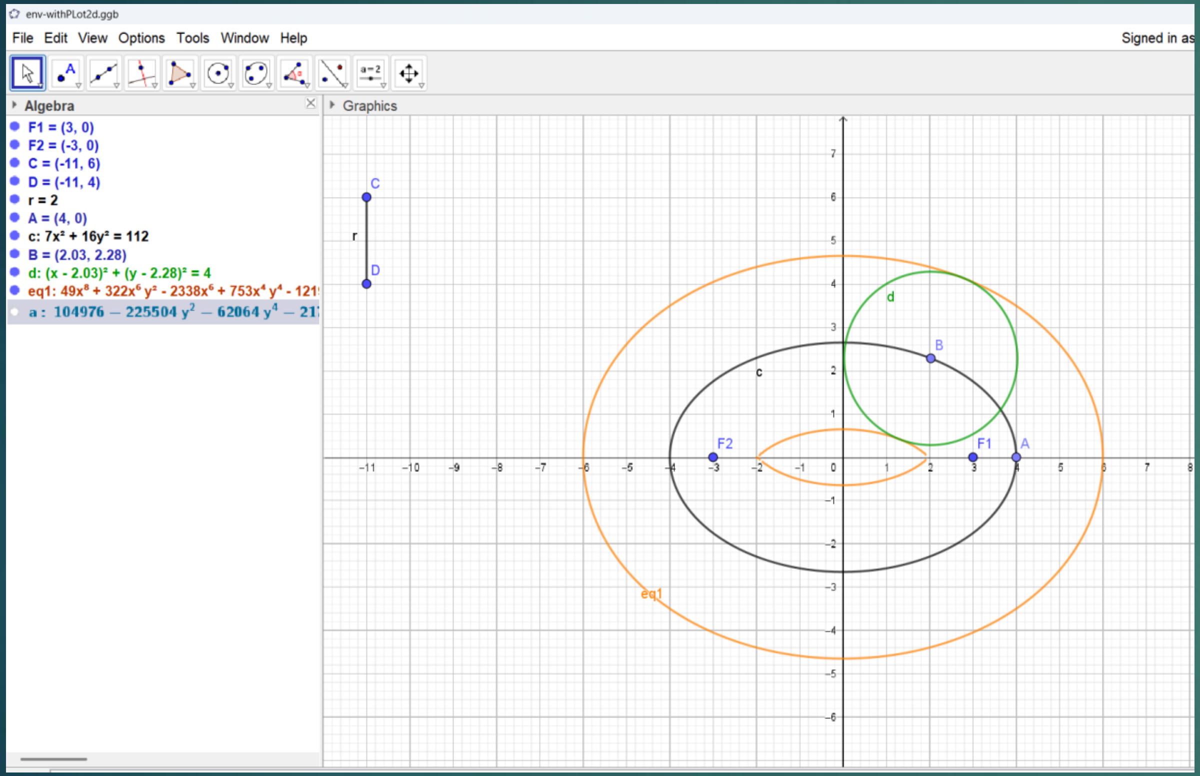
Plot obtained with Derive 6.1

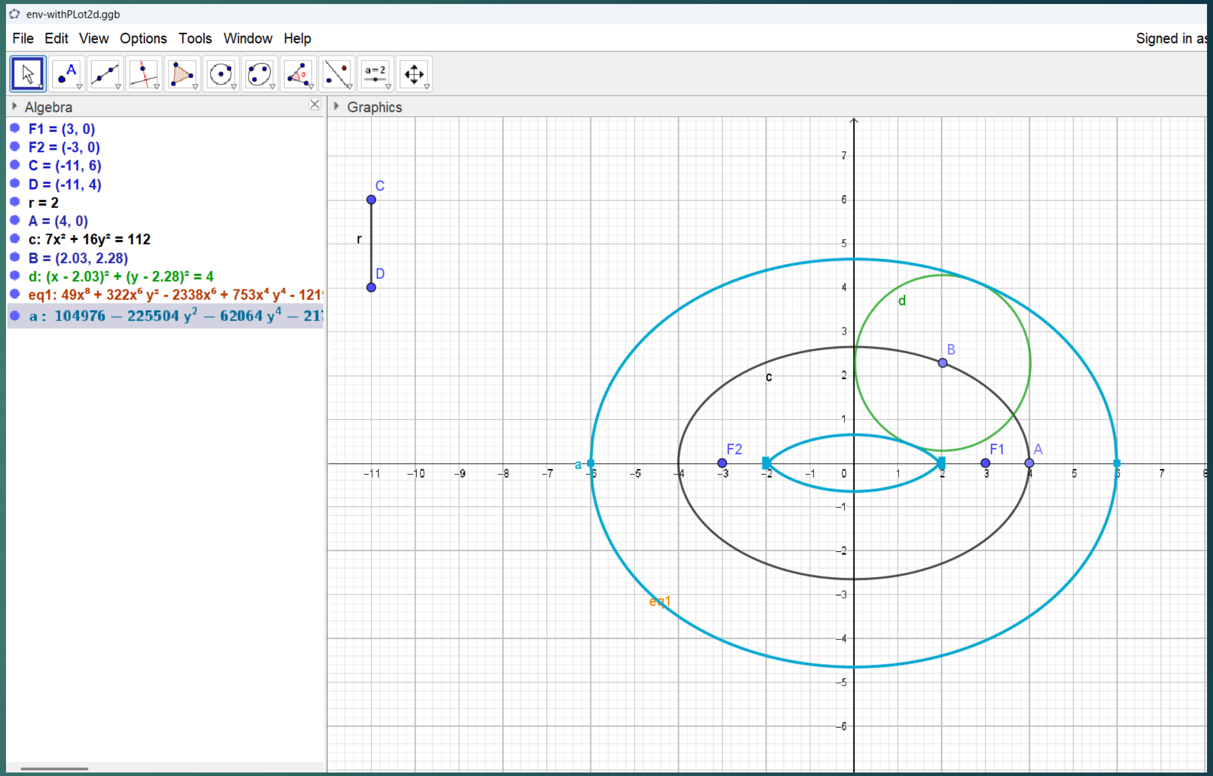
Can be done with GeoGebra (we suggest to define a slider increment enabling to view separate lines)



Th. DP and N. Zehavi (2016): Revival of a classical topic in Differential Geometry: the exploration of envelopes in a computerized environment, International Journal of Mathematical Education in Science and Technology 47(6), 938-959.

Interactive exploration – 1st examples





With the Envelope command

With Plot2D

GeoGebra's Envelope command

Suitable for "industrial" applications. How to translate into a computational frame?

Envelope Command

Envelope(<Path>, <Point>)

Creates the envelope equation of a set of output paths while the moving point is bound to another object.

An envelope is a curve that is tangent to each member of the family of the output paths at some point.

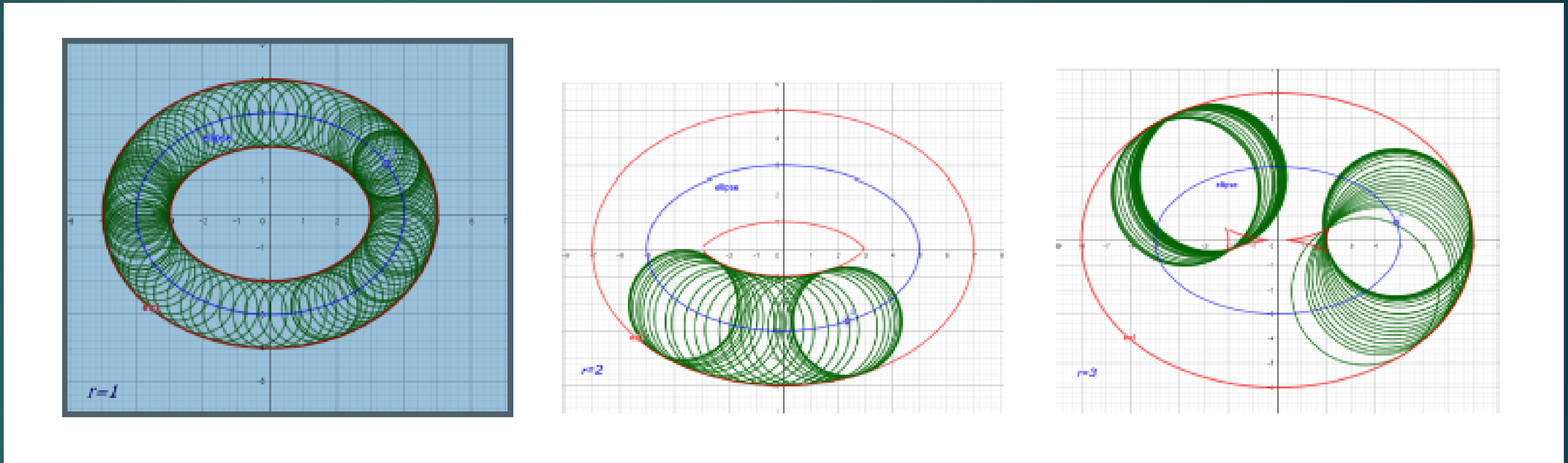
A ladder is leaning against the wall and sliding down.

The contour of its trace will be the envelope of the ladder. Strictly speaking, GeoGebra computes the envelope of the entire line containing the ladder as a segment. Only such envelopes can be com-puted where the appropriate construction leads to an algebraic equation system.

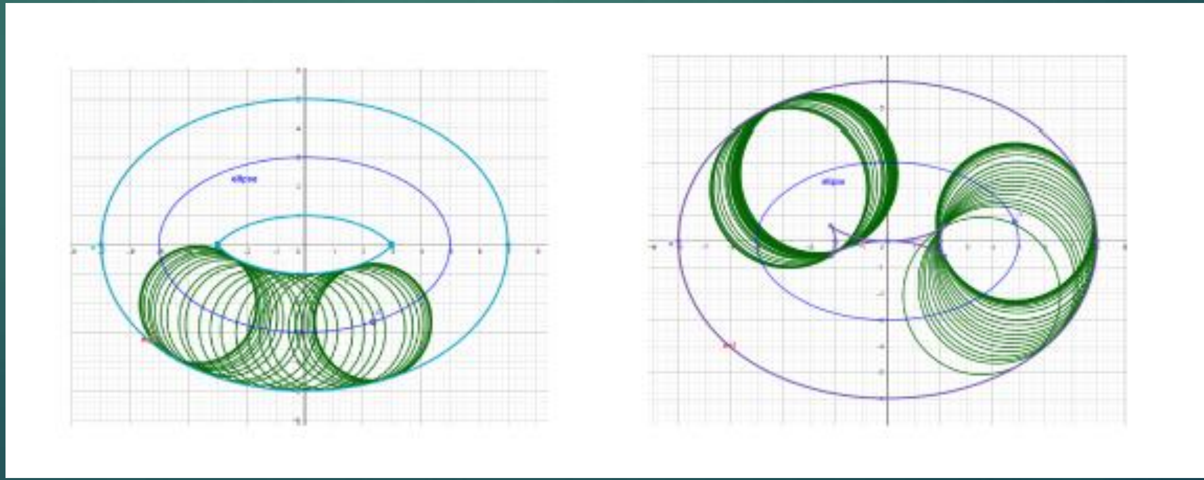
NOTE

See also Locus, LocusEquation commands and GeoGebra Automated Reasoning Tools: A Tutorial.

Variable radius: Singularities may appear



Singular points: Plot2D completes and emphasizes



Envelopes – Analytic definition

Let $\{\mathcal{C}_k\}$ be a family of plane curves given by the equation $F(x,y,k)=0$, where k is a real parameter. An envelope $\mathcal{E}$ of this family, if it exists, is determined by the solutions of the system of equations:

$$\begin{cases} F(x,y,k)=0 \\ \dfrac{\partial F}{\partial k}(x,y,k)=0 \end{cases}$$

Solving the system of equations with a CAS provides a parametric representation.

<u>Not always easy:</u> Implicitization is performed with a CAS (Maple's *PolynomilalIdeal package*, CoCoA, Groebner bases package in another CAS)

Offsets of a plane curve (Euler's Parallel curves)

Offsets

Let $\mathcal{C}$ be a plane curve and d be a positive real number. A **regular** point A of $\mathcal{C}$ is a point where $\mathcal{C}$ is smooth, i.e. tangents vectors and normal vectors are defined.

If A is a regular point, we denote by $\overrightarrow{N_A}$ a unit vector normal to $\mathcal{C}$ at A.

Let B be a point such that $\overrightarrow{AB} = d\,\overrightarrow{N_A}$ and C a point such that $\overrightarrow{AC} = -\,d\,\overrightarrow{N_A}$

The union of **the geometric loci** of B and C when A runs over $\mathcal{C}$ is called the **offset** of $\mathcal{C}$ at distance d.

The curve $\mathcal{C}$ is called the **progenitor** of the offsets.

Offsets

Let $\mathcal{C}$ be a plane curve and d be a positive real number. A **regular** point A of $\mathcal{C}$ is a point where $\mathcal{C}$ is smooth, i.e. tangents vectors and normal vectors are defined.

If A is a regular point, we denote by $\boxed{\overrightarrow{N_A}}$ a unit vector normal to $\mathcal{C}$ at A.

Let B be a point such that $\boxed{\overrightarrow{AB} = d\,\overrightarrow{N_A}}$ and C a point such that $\boxed{\overrightarrow{AC} = -\,d\,\overrightarrow{N_A}}$

The union of **the geometric loci** of B and C when A runs over $\mathcal{C}$ is called the **offset** of $\mathcal{C}$ at distance d.

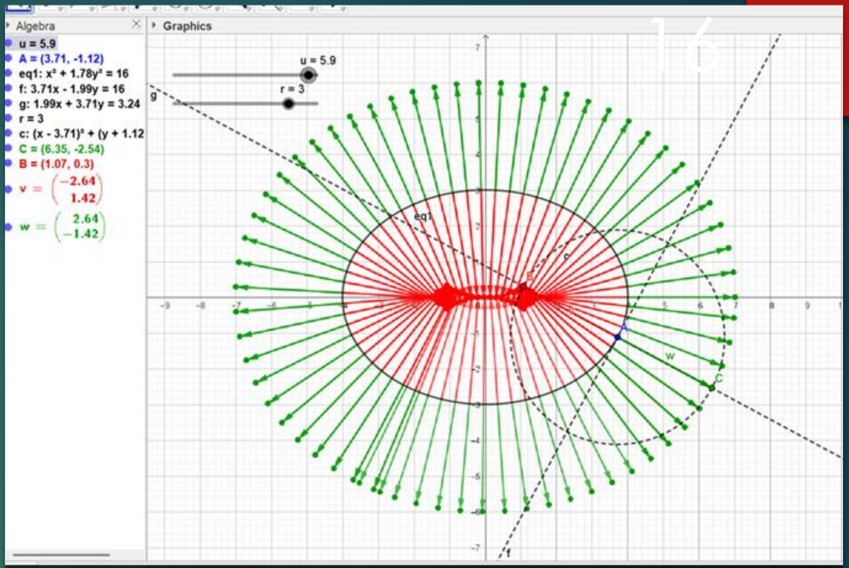
The curve $\mathcal{C}$ is called the **progenitor** of the offsets.

Interactive creation of an offset

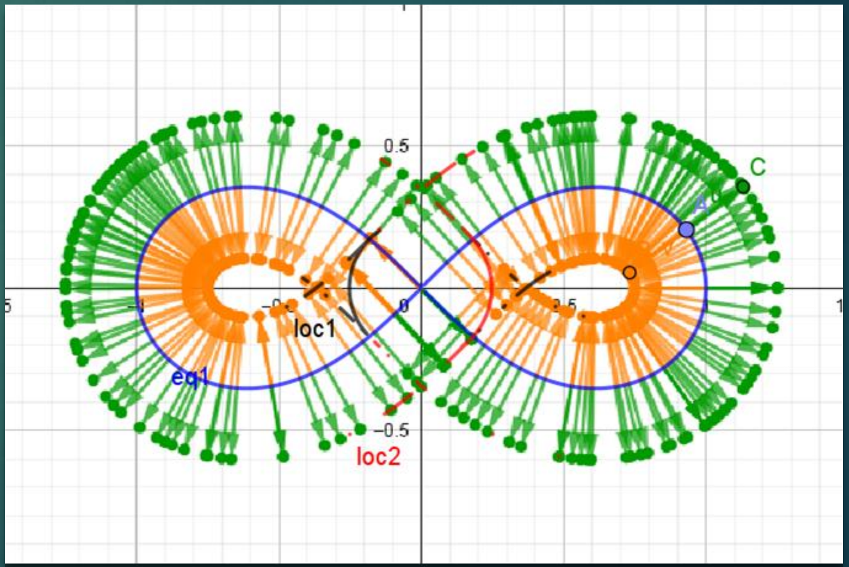
Used commands:

- Point on object
- Tangent
- Perpendicular line
- Circle with given radius
- Intersection
- Locus (Point, Point)
- Plot2D

Envelope of circles centered on an ellipse And offsets of this ellipse



Offsets of a Bernoulli lemniscate



Note that the locus plot is not complete

Offsets of a deltoid

- Parametric presentation of a deltoid: $\begin{cases} x = 2(\,2\cos t + \cos 2t\,) \\ y = 2(\,2\sin 2t - 2\sin t\,) \end{cases}$

- Polynomial equation of the envelope of unit circles centered on the deltoid:

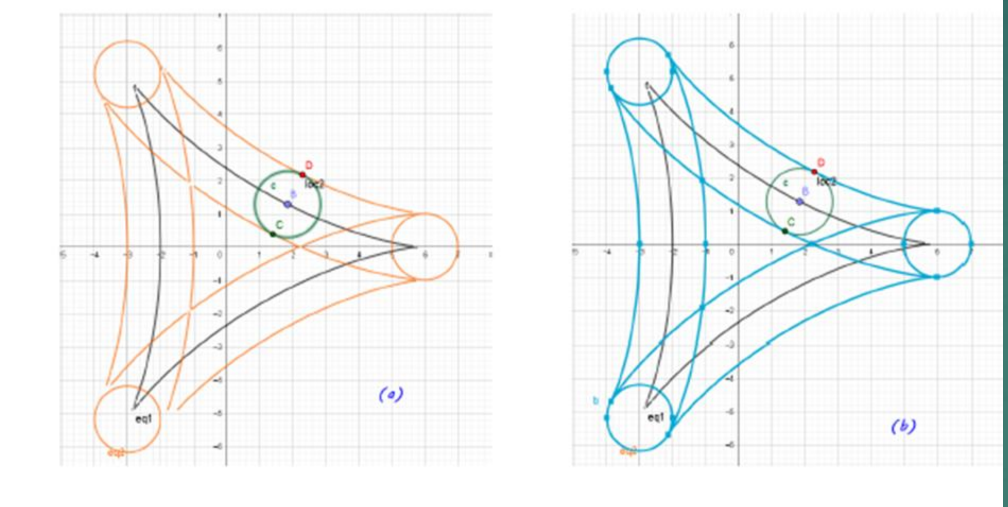
$$\begin{aligned} F(x, y) = {} & x^{14} - 32x^{13} + 7x^{12}y^2 + 396x^{12} - 64x^{11}y^2 - 2496x^{11} + 21x^{10}y^4 + 72x^{10}y^2 \\ & + 14790x^{10} + 160x^9y^4 - 2496x^9y^2 - 140172x^9 + 35x^8y^6 + 564x^8y^4 - 43554x^8y^2 \\ & + 866060x^8 + 640x^7y^6 + 14976x^7y^4 + 995328x^7y^2 - 2192044x^7 + 35x^6y^8 + 5360x^6y^6 \\ & - 8772x^6y^4 - 4691920x^6y^2 + 7473633x^6 + 800x^5y^8 + 34944x^5y^6 - 2808504x^5y^4 \\ & + 2192044x^5y^2 - 68503428x^5 + 21x^4y^{10} + 7476x^4y^8 + 174012x^4y^6 + 2477640x^4y^4 \\ & + 10557027x^4y^2 + 103434084x^4 + 448xy^{10} + 27456x^3y^8 + 3222624x^3y^6 + 10960220x^3y^4 \\ & + 137006856x^3y^2 + 924501276x^3 + 7x^2y^{12} + 3144x^2y^{10} + 126174x^2y^8 + 7995440x^2y^6 \\ & + 30330147x^2y^4 + 206868168x^2y^2 - 2633760360x^2 + 96xy^{12} + 7488xy^{10} + 88740xy^8 \\ & + 6576132xy^6 + 205510284xy^4 - 2773503828xy^2 + y^{14} + 140y^{12} + 1734y^{10} - 40180y^8 \\ & + 6155425y^6 + 103434084y^4 - 2633760360y^2 + 3375634500. \end{aligned}$$

Is this polynomial reducible?

<u>Tools:</u>

- GGB-CAS: Factor
- Maple: Evala(AFactor)

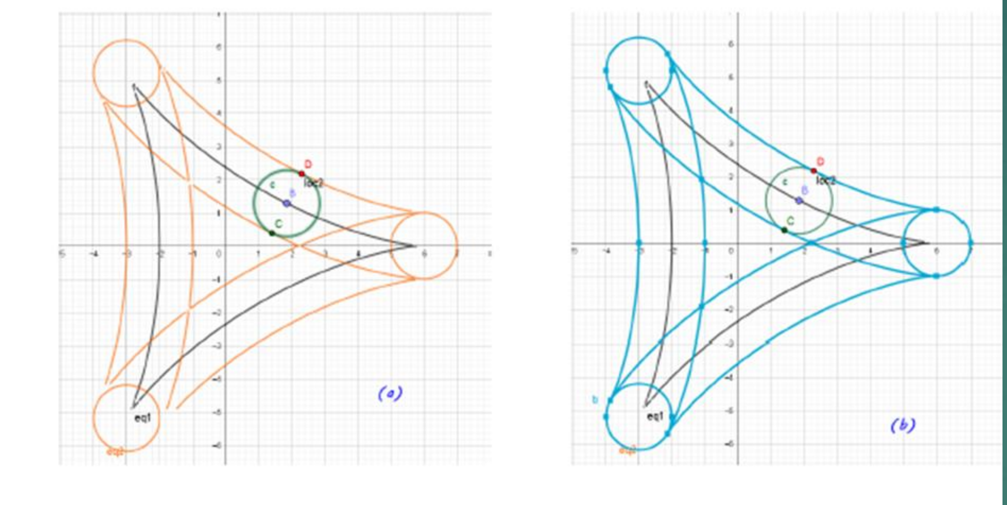
Envelope of unit circles centered on a deltoid



- It is incomplete, i.e. there are apparent singularities. This is not a surprise, as the progenitor has singularities, and we already mentioned that the offset has generally a more complicated topology than the progenitor.

Dana-Picard, Th., Kovács, Z. (2021): Networking of technologies: a dialog between CAS and DGS, The electronic Journal of Mathematics and Technology (eJMT), 15 (1), 43-59. https://php.radford.edu/~ejmt/deliveryBoy.php?paper=eJMT_v15n1p3.

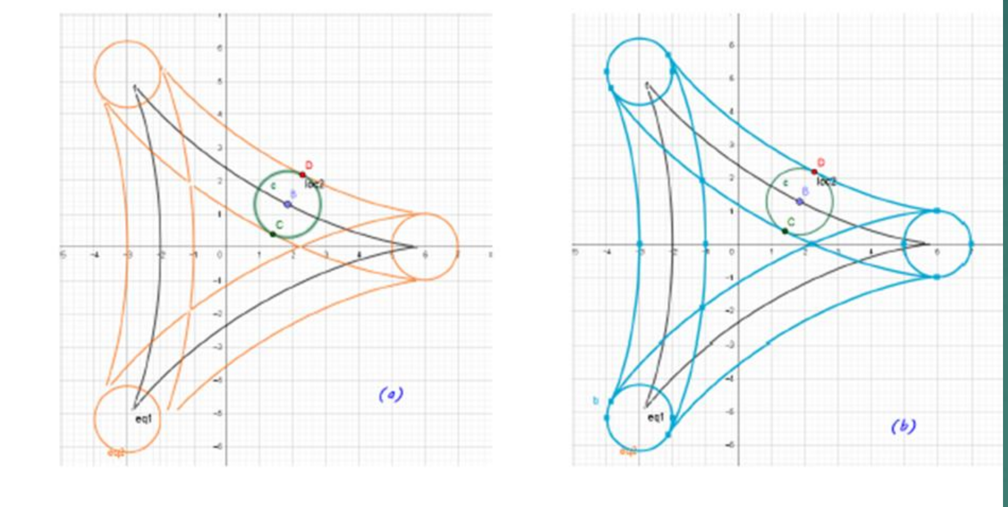
Envelope of unit circles centered on a deltoid



- (ii) There are 3 circles centered at the cusps of the deltoid who contribute arcs to the envelope (actually semi-circles, what has to be algebraically proven). These circles are not a part of curves defined by the polynomial F(x, y). The obtained envelope corresponds to another definition of envelopes; it fits what is needed for safety zones around industrial robots or entertainment parks.

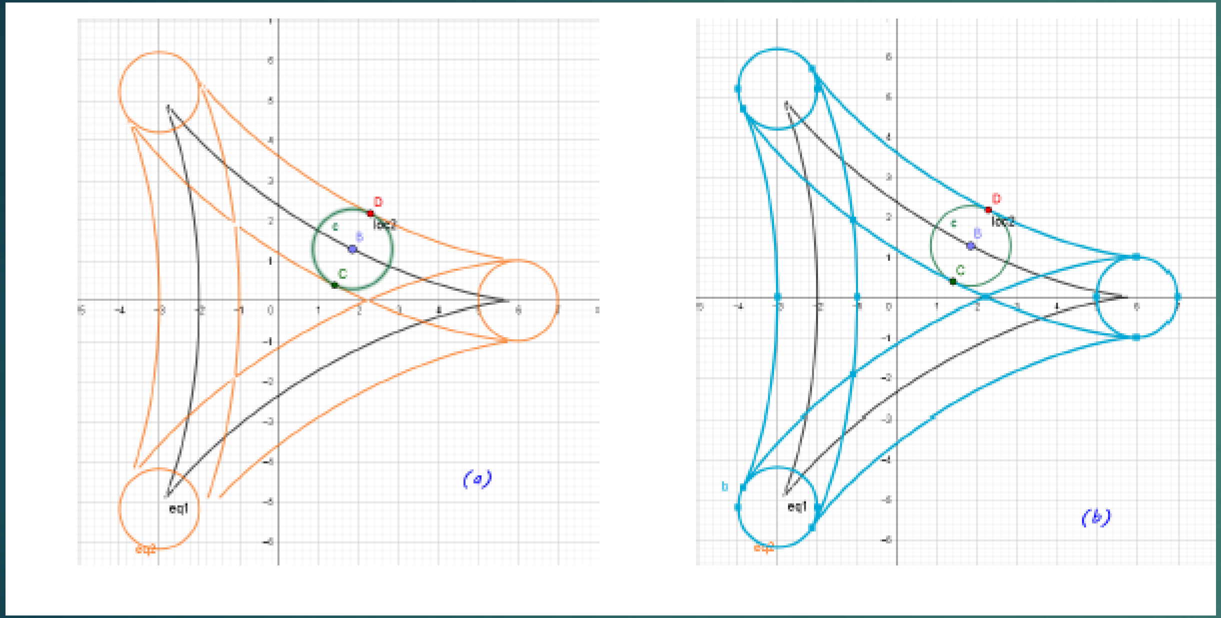
Dana-Picard, Th., Kovács, Z. (2021): Networking of technologies: a dialog between CAS and DGS, The electronic Journal of Mathematics and Technology (eJMT), 15 (1), 43-59. https://php.radford.edu/~ejmt/deliveryBoy.php?paper=eJMT_v15n1p3.

Envelope of unit circles centered on a deltoid

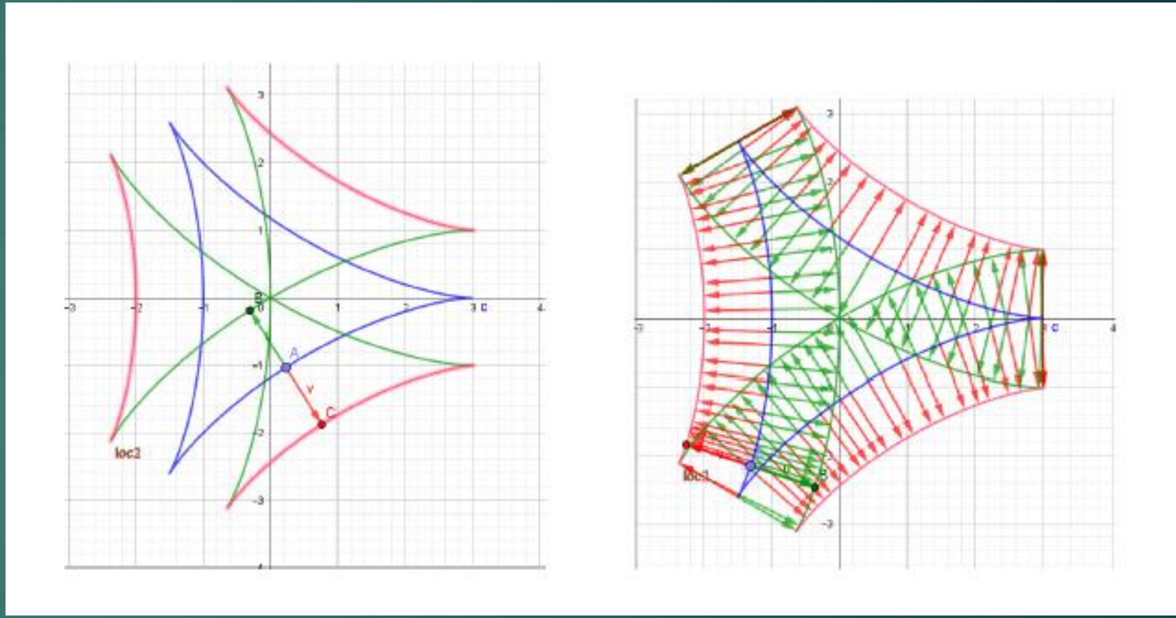


- (iii) The "holes" in the plot are larger than in the previous example. They are fllled (and the singular points are emphasized) when using Plot2D.

Envelopes of circles of constant radius and offsets – a comparison



Envelope of the family of unit circles centered on a deltoid



Offset at distance 1 of a deltoid

GeoGebra's loci (numerous) commands

Locus Command

Locus(<Point Creating Locus Line Q>, <Point P>)

Returns the locus curve of the point *Q*, which depends on the point *P*.

NOTE

Point *P* needs to be a point on an object (e. g. line, segment, circle).

Locus(<Point Creating Locus Line Q>, <Slider t>)

Returns the locus curve of the point *Q*, which depends on the values assumed by the slider *t*.

Locus(<Slopefield>, <Point>)

Returns the locus curve which equates to the slopefield at the given point.

Locus(<f(x, y)>, <Point>)

Returns the locus curve which equates to the solution of the differential equation $\frac{dy}{dx} = f(x,y)$ in the given point. The solution is calculated numerically.

Loci are specific object types, and appear as auxiliary objects. Besides Locus command, they are the result of some Discrete Math Commands and SolveODE Command. Loci are paths and can be used within path-related commands such as Point. Their properties depend on how they were obtained, see e.g. Perimeter Command and First Command.

NOTE

See also Locus tool.

LocusEquation Command

LocusEquation(<Locus>)

Calculates the equation of a Locus and plots this as an Implicit Curve.

LocusEquation(<Point Creating Locus Line Q>, <Point P>)

Calculates the equation of a Locus by using inputs tracer point *Q* and mover point *P*, and plots this as an Implicit Curve.

Let us construct a parabola as a locus: Create free Points *A* and *B*, and Line *d* lying through them (this will be the directrix of the parabola). Create free point *F* for the focus. Now create *P* on Line *d* (the mover point), then create line *p* as a perpendicular line to *d* through *P*. Also create line *b* as perpendicular bisector of Line Segment *FP*. Finally, point *Q* (the point creating locus line) is to be created as intersection of Lines *p* and *b*. Now `LocusEquation(Q,P)` will find and plot the exact equation of the locus.

LocusEquation(<Boolean Expression>, <Free Point>)

Calculates the locus of the free point such that the boolean condition is satisified.

`LocusEquation(AreCollinear(A, B, C), A)` for free points *A*, *B*, *C* calculates the set of positions of *A* that make *A*, *B* and *C* collinear—i.e. a line through *B* and *C*.

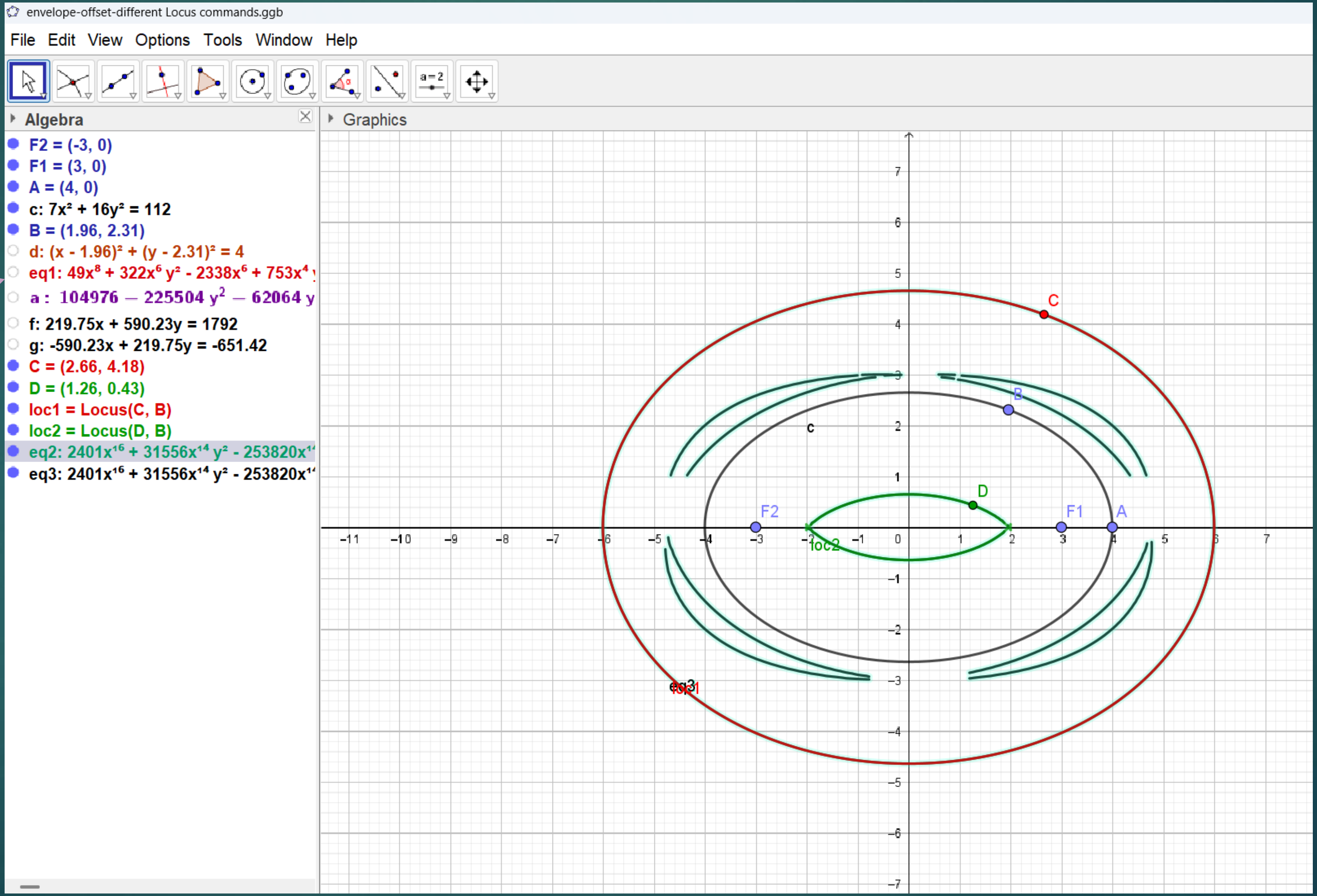
NOTE

- The Locus must be made from a Point (not from a Slider)
- Works only for a restricted set of geometric loci, i.e. using points, lines, circles, conics. (Rays and line segments will be treated as (infinite) lines.)

Dynamic exploration with GeoGebra commands: envelope - locus

Envelope

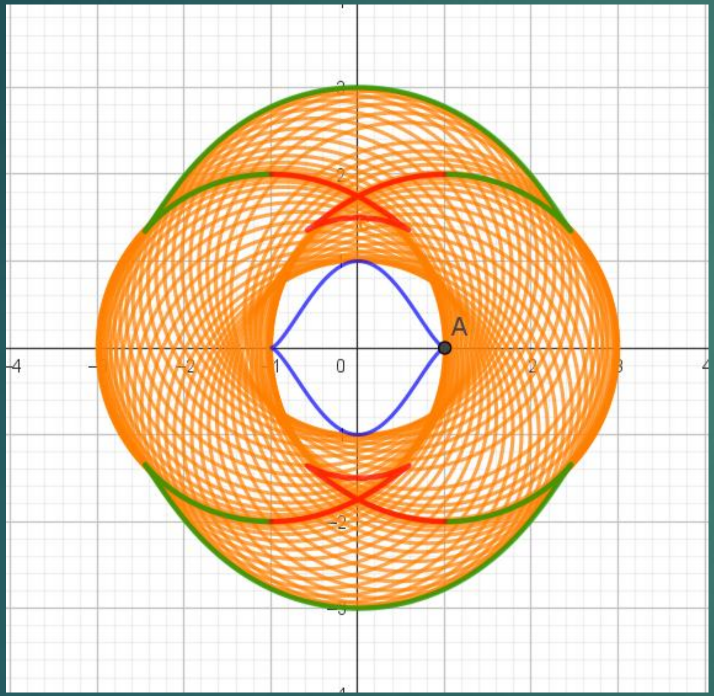
Plot2D

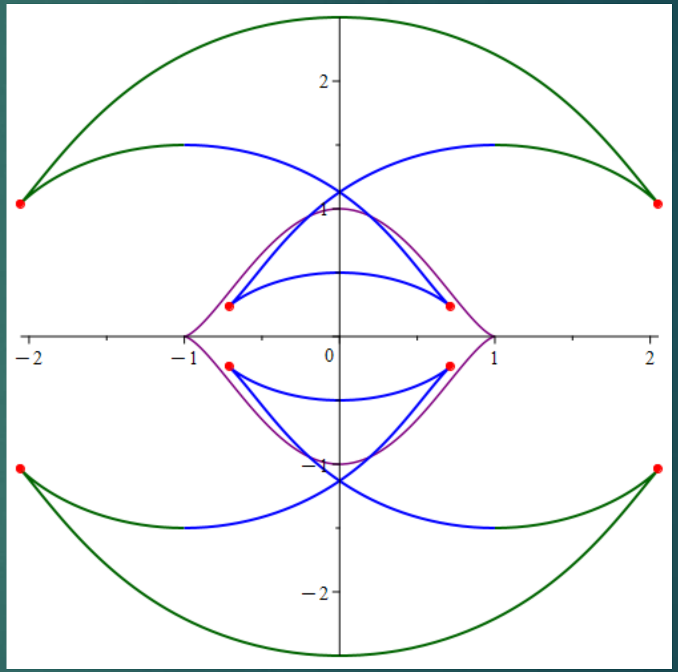


Irrelevant components may appear, for topological reasons; they cannot be separated by algebraic means, but with the dynamical geometry

Example: offsets of a kiss curve
(part of the work of the "excellent student")

$$\begin{cases} x = \cos u \\ y = \sin^3 u \end{cases}$$





Interactive comparison of an envelope and an offset (GeoGebra Discovery)

Plot with Maple (enhancing cusps)

Issues requiring strong algebraic abilities

- Solve the system of polynomial equations (determination of an envelope)

$$\begin{cases} F(x, y, k) = 0 \\ \dfrac{\partial F}{\partial k}(x, y, k) = 0 \end{cases}$$

- Solve equations – provides a set of parametric representations, i.e. the curve is presented as the union of several components. Continuity problems may appear in the plot
- Implicitization - elimination of a parameter for the determination of a geometric locus
- Determination of singular points of the obtained curve

Issues requiring strong algebraic abilities

- Solve the system of polynomial equations (determination of an envelope)

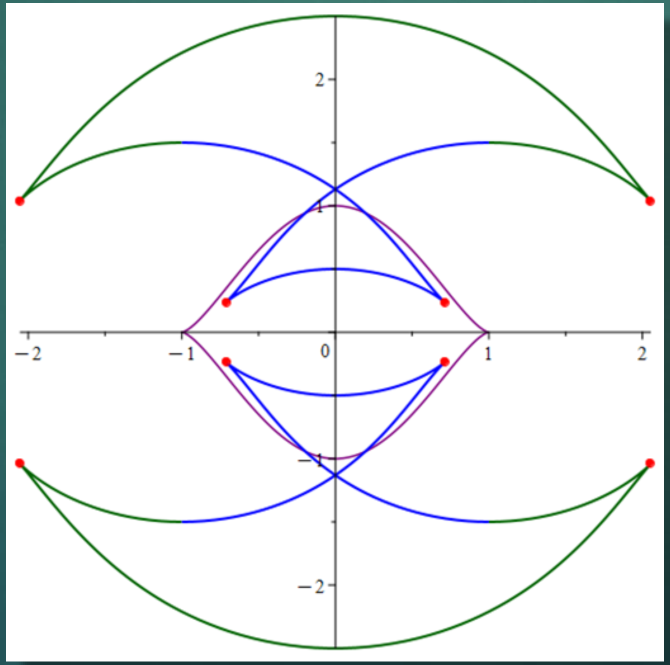
$$\begin{cases} F(x,y,k)=0 \\ \dfrac{\partial F}{\partial k}(x,y,k)=0 \end{cases}$$

- Solve equations – elimination of a parameter for the determination of a geometric locus
- Implicitization (with Elimination, or with Maple's **implicitize** command – requires an initial guess on the degree)
- Determination of singular points of the obtained curve

Looking for singular points

Their nature

- Self-intersections (crunodes)
- Cusps
- Etc…



The representations of the curves

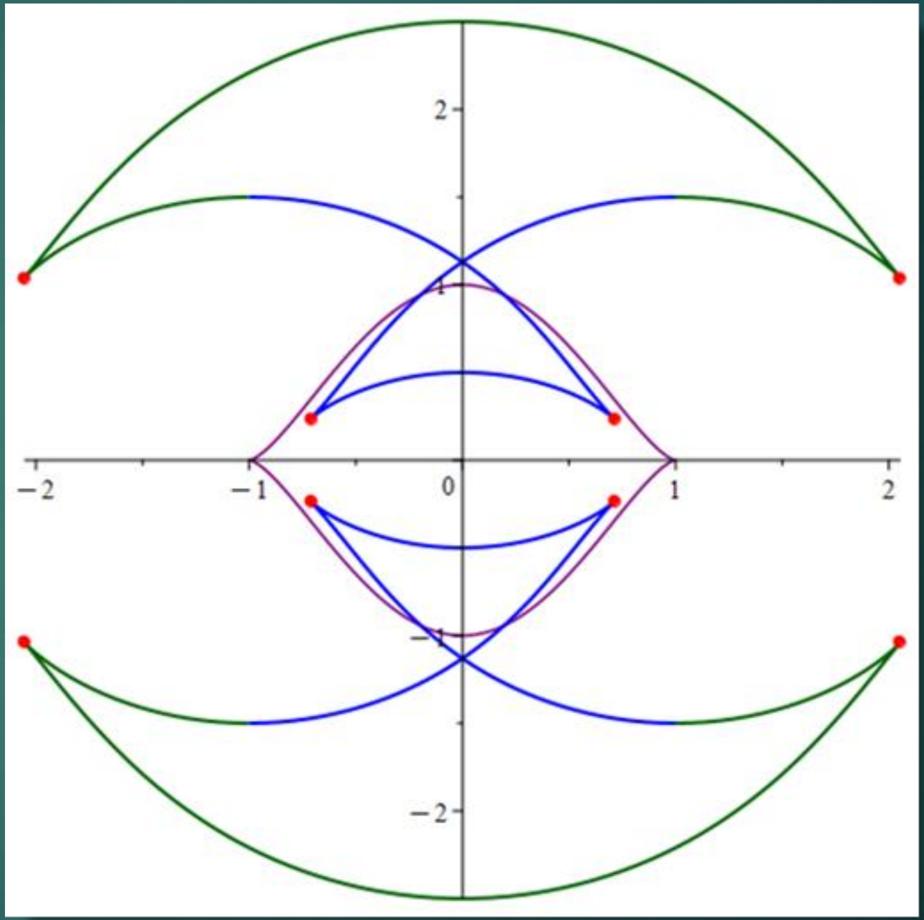
- Symbolic equation (polynomials)
- Parametric representation

Maple code for cusps


```
restart : with(plots) : setoptions(scaling = constrained) :
x := cos(t);
y := sin(t)^3;
d := 0.1;
dx := diff(x,t);
dy := diff(y,t);
denominator := sqrt(dx^2 + dy^2);
x_offset_pos := x + dy*d/denominator;
y_offset_pos := y - dx*d/denominator;
x_offset_neg := x - dy*d/denominator;
y_offset_neg := y + dx*d/denominator;
r := ⟨x,y⟩;
dr := diff(r,t);
ddr := diff(dr,t);
kO := (-ddr[1]*dr[2] + ddr[2]*dr[1])/(dr[1]^2 + dr[2]^2)^(3/2);
kI := (ddr[1]*dr[2] - ddr[2]*dr[1])/(dr[1]^2 + dr[2]^2)^(3/2);
sO := solve(kO = -1/d,t);
sI := solve(kI = -1/d,t);
t_values := [ ];
for i to nops([sO]) do
  if Im(evalf(sO[i])) = 0 then t_values := [op(t_values), evalf(sO[i])]; exp(1)*nd; end if;
end do;
t_values1 := [ ];
for i to nops([sI]) do
  if Im(evalf(sI[i])) = 0 then t_values1 := [op(t_values1), evalf(sI[i])]; exp(1)*nd; end if;
end do;
```

```
outer_singular_points := [seq([evalf(subs(t = t_values[i], x_offset_pos)),
        evalf(subs(t = t_values[i], y_offset_pos))], i = 1 .. nops(t_values))];
inner_singular_points := [seq([evalf(subs(t = t_values1[i], x_offset_neg)),
        evalf(subs(t = t_values1[i], y_offset_neg))], i = 1 .. nops(t_values1))];
curve_plot := plot([x, y, t = 0 .. 2*Pi], color = purple);
outer_singular_points_plot := pointplot(outer_singular_points, symbol = solidcircle,
        color = red, symbolsize = 8);
inner_singular_points_plot := pointplot(inner_singular_points, symbol = solidcircle,
        color = red, symbolsize = 8);
offset_pos_plot := plot([x_offset_pos, y_offset_pos, t = -Pi .. Pi],
        color = darkgreen, thickness = 2, discont = true);
offset_neg_plot := plot([x_offset_neg, y_offset_neg, t = -Pi .. Pi],
        color = blue, thickness = 2, discont = true);
display(curve_plot, offset_pos_plot, offset_neg_plot, inner_singular_points_plot,
        outer_singular_points_plot);
```

The output – determining cusps

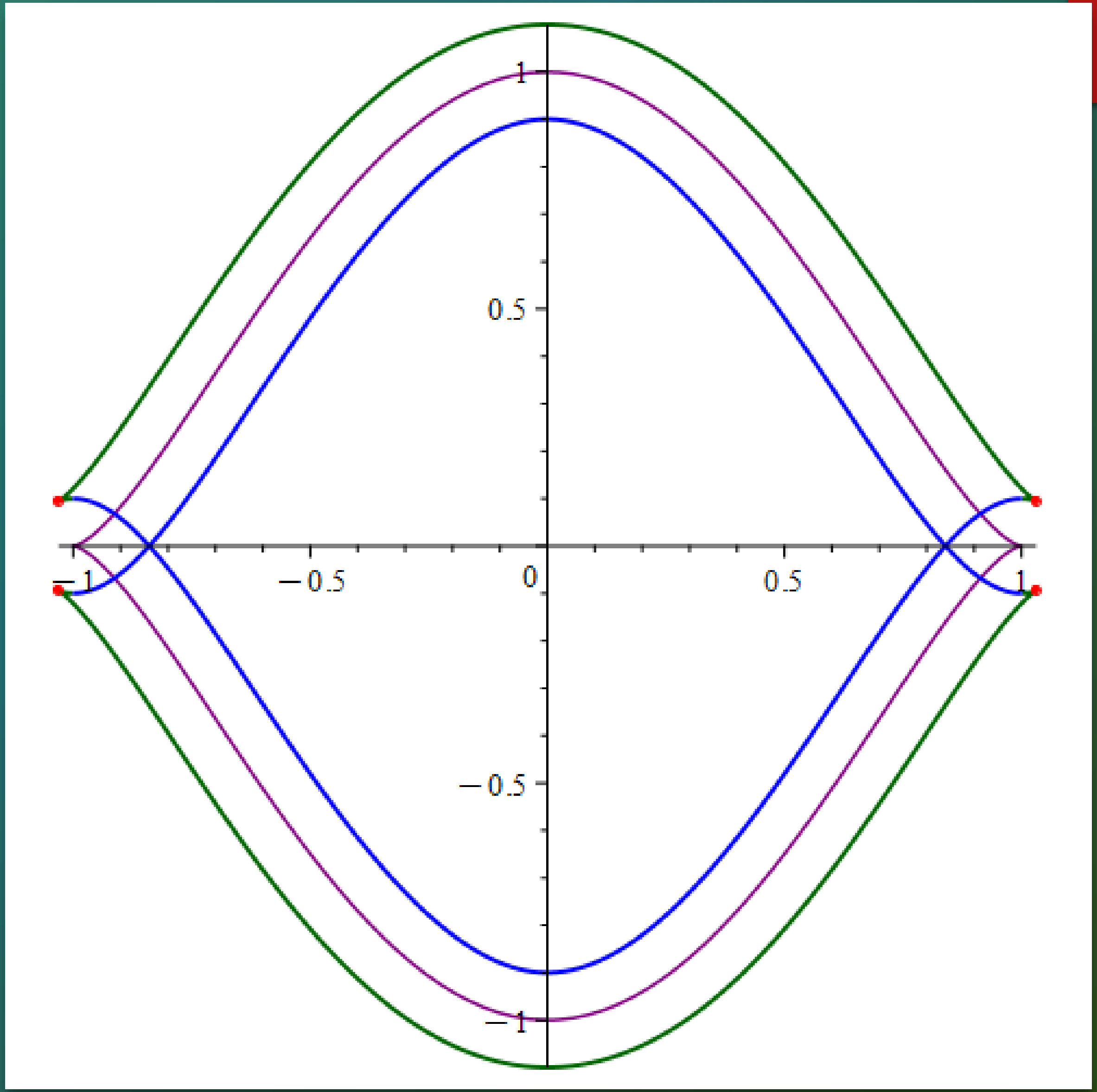


Maple curve – cusps (using curvature)

```
restart : with(plots) : setoptions(scaling = constrained) :
x := cos(t);
y := sin(t)^3;
d := 0.1;
dx := diff(x, t);
dy := diff(y, t);
denominator := sqrt(dx^2 + dy^2);
x_offset_pos := x + dy * d / denominator;
y_offset_pos := y − dx * d / denominator;
x_offset_neg := x − dy * d / denominator;
y_offset_neg := y + dx * d / denominator;
r := ⟨x, y⟩;
dr := diff(r, t);
ddr := diff(dr, t);
kO := (−ddr[1] * dr[2] + ddr[2] * dr[1]) / (dr[1]^2 + dr[2]^2)^(3/2);
kI := (ddr[1] * dr[2] − ddr[2] * dr[1]) / (dr[1]^2 + dr[2]^2)^(3/2);
sO := solve(kO = −1 / d, t);
sI := solve(kI = −1 / d, t);
t_values := [ ];
for i to nops([sO]) do
  if Im(evalf(sO[i])) = 0 then t_values := [op(t_values), evalf(sO[i])]; exp(1) * nd; end if;
end do;
t_values1 := [ ];
for i to nops([sI]) do
  if Im(evalf(sI[i])) = 0 then t_values1 := [op(t_values1), evalf(sI[i])]; exp(1) * nd; end if;
end do;
```

```
outer_singular_points := [seq([evalf(subs(t = t_values[i], x_offset_pos)),
        evalf(subs(t = t_values[i], y_offset_pos))], i = 1 .. nops(t_values))];
inner_singular_points := [seq([evalf(subs(t = t_values1[i], x_offset_neg)),
        evalf(subs(t = t_values1[i], y_offset_neg))], i = 1 .. nops(t_values1))];
curve_plot := plot([x, y, t = 0 .. 2 * Pi], color = purple);
outer_singular_points_plot := pointplot(outer_singular_points, symbol = solidcircle,
        color = red, symbolsize = 8);
inner_singular_points_plot := pointplot(inner_singular_points, symbol = solidcircle,
        color = red, symbolsize = 8);
offset_pos_plot := plot([x_offset_pos, y_offset_pos, t = −Pi .. Pi],
        color = darkgreen, thickness = 2, discont = true);
offset_neg_plot := plot([x_offset_neg, y_offset_neg, t = −Pi .. Pi],
        color = blue, thickness = 2, discont = true);
display(curve_plot, offset_pos_plot, offset_neg_plot, inner_singular_points_plot,
        outer_singular_points_plot);
```

The output

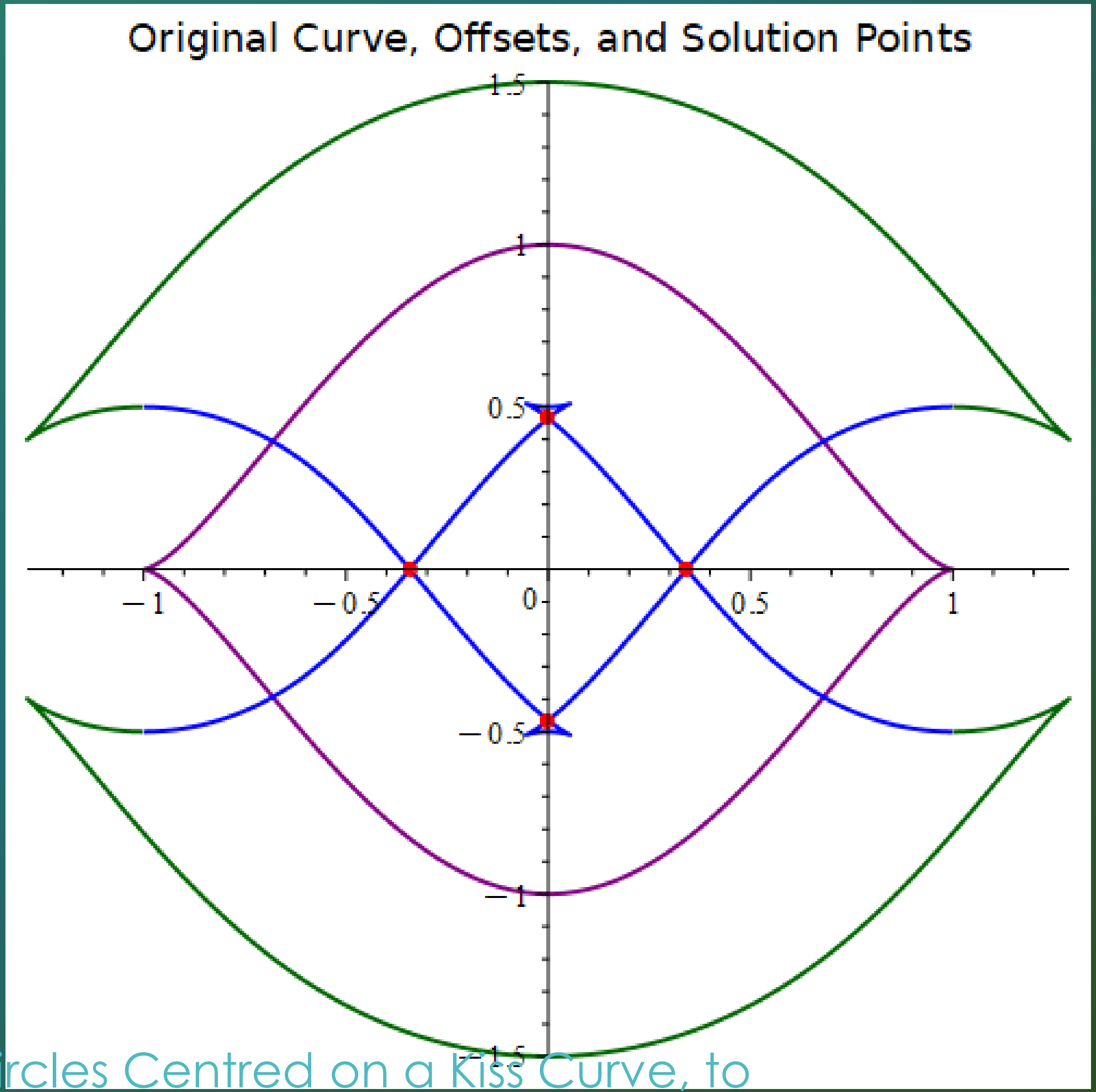


Maple code for self-intersections

```
with(plots) :

# Define the main functions
x := u → cos(u);
y := u → sin(u)^3;

# Symbolically differentiate x(u) and y(u) with respect to u
dx := D(x);
dy := D(y);

# Define the offset distance
 d := 0.5;

# Define the offset position functions using the symbolic derivatives
x_offset_pos := u → x(u) + dy(u) * d / sqrt(dx(u)^2 + dy(u)^2);
y_offset_pos := u → y(u) − dx(u) * d / sqrt(dx(u)^2 + dy(u)^2);
x_offset_neg := u → x(u) − dy(u) * d / sqrt(dx(u)^2 + dy(u)^2);
y_offset_neg := u → y(u) + dx(u) * d / sqrt(dx(u)^2 + dy(u)^2);

# Define the equations to solve for intersection points of the offset curves
eq1 := x_offset_neg(s) − x_offset_neg(t) = 0;
eq2 := y_offset_neg(s) − y_offset_neg(t) = 0;

# Initialize an empty list to store solutions
solutions := [ ];

 # Define a comprehensive set of ranges to find more solutions
ranges := [
   {s = −4 * Pi .. −3 * Pi, t = −4 * Pi .. −3 * Pi},
   {s = −7 * Pi / 2 .. −5 * Pi / 2, t = −7 * Pi / 2 .. −5 * Pi / 2},
   {s = −5 * Pi / 2 .. −3 * Pi / 2, t = −5 * Pi / 2 .. −3 * Pi / 2},

   {s = 3 * Pi .. 4 * Pi, t = 3 * Pi .. 4 * Pi}
 ];
```

```
# Find solutions within each specified range
for r in ranges do
  sol := fsolve({eq1, eq2}, r);
   if sol ≠ NULL and evalf(subs(sol, s)) ≠ evalf(subs(sol, t) ) then
    solutions := [op(solutions), sol];
  end if;
end do;

# Evaluate x and y positions for each solution and store them
points := [ ];
for sol in solutions do
  s_sol := evalf(subs(sol, s));
  t_sol := evalf(subs(sol, t));
  x_pos := evalf(x_offset_neg(s_sol));
  y_pos := evalf(y_offset_neg(s_sol));

  points := [op(points), [x_pos, y_pos] ];
 end do;


# Plot the main curve and offsets
plot1 := plot([x(u), y(u), u = −Pi .. Pi], color = purple, thickness = 2) :
plot2 := plot([x_offset_pos(u), y_offset_pos(u), u = −Pi .. Pi], color = darkgreen, thickness = 2) :
plot3 := plot([x_offset_neg(u), y_offset_neg(u), u = −Pi .. Pi], color = blue, thickness = 2) :

# Plot all solution points
points_plot := pointplot(points, symbol = solidcircle, color = red, symbolsize = 10) :

# Combine plots
combined_plot := display([plot1, plot2, plot3, points_plot], title = "Original Curve, Offsets, and Solution Points", titlefont = [Helvetica, 14]);
```

Self intersections – the output



Th. DP and D. Tsirkin (2025): Envelopes of Circles Centred on a Kiss Curve, to appear in Maple Transactions.

Experiencing with AI (not with the classes)

- Answers:
  - Gives generally an accurate description of the process
  - An accurate answer to a 1st question (the parabola obtained with the family of lines above – published by DMZ)
  - An accurate description and a wrong answer to a 2nd question (published in the same paper)
  - A wrong answer on an envelope of a family of circles of constant radius centered on an ellipse: claims that it is an envelope, and gives an equation.
  - Can help to improve code, not always

- The experimentation has been conducted a couple of months ago. We saw some improvement since a previous one. Maybe today the situation is again better.

Educational remarks

Regular students

- In the teacher training college (Jerusalem Michlala College): The in-service teachers told their dpt head that this was the most significant course in their degree
- They discovered both a new point of view and new topics in math and technology (actually an implementation of Artigue's claim, 2002)
- They experienced "exploration-conjecture-proof" sequences.
- They understand that the future of Math Educ is in a technology rich environment
- @ JCT: at the dpt head initiative, and with rector's authorization, a new such course will be opened for engineering undergraduates at JCT in Fall 2025.

A student as a research assistant

- Autonomous work
- The researcher is a **mentor**
- **The researcher does not provide scaffolding, but rather a collaborative atmosphere**
- **The student can propose directions and methods**
- **It is possible to observe his instrumental genesis**
- Direct contact with software developers has been developed

Thank you for your attention
Merci pour votre attention
Vielen Dank für Ihre Aufmerksamkeit
Хвала вам за вашу пажњу
Obrigado pela sua atenção
תודה על ההקשבה