Numerical Approach to Real Algebraic Curves ...

Chapter 9

Barry H Dayton, (barryhdayton.space)

Proof of Harnack's Theorem for non-Singular Plane Curves

This proof got garbled in the original additions of this book, here is a correction for those with the fixed print (paperback and Kindle) editions.

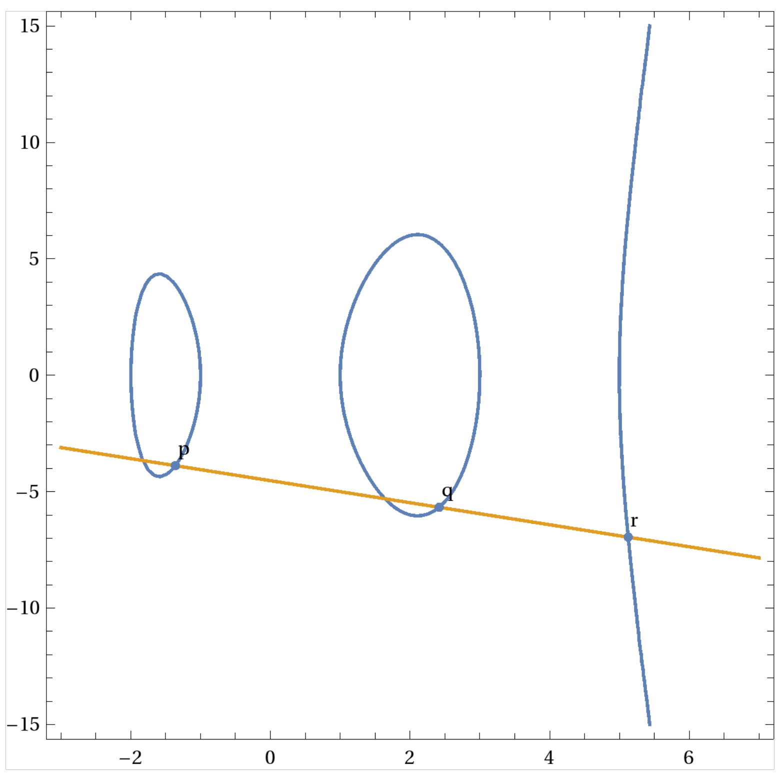
We will use three tools from earlier sections of the book:

1. At the end of Chapter 1, we showed that one can construct a curve of degree d through any affine collection of $\frac{(d+2)(d+1)}{2} - 1$ points.
2. Properties 1, 2, and 3 from Section 8.5.
3. Bézout's theorem from Chapter 5.

The following example will be a warm-up. In Chapter 7, we catalog all possible conics and cubics, but we will give a new argument that a nonsingular cubic can have only two topological components.

Suppose a cubic has three topological components: two ovals and one pseudo-line. We pick one point on each oval, say p and q. There is a unique line ℓ through p and q that, by Property 2 (Section 8.5), must meet the pseudo-line at a point r.

A plot might look like:



ℓ is the orange line. By a trivial application of Bézout's theorem, a line can only intersect a cubic in three or fewer real points, but this line also intersects the cubic in two other points on the ovals. Therefore, this configuration is impossible for a cubic.

Suppose $f = 0$ is a projective curve of degree $d > 3$, let

```
In[ ]:= Clear[d];
        m = (d - 1) (d - 2) / 2 + 1;
```

Suppose the associated projective curve has a simple closed path decomposition with m ovals $\Omega_1, \Omega_2, \ldots, \Omega_m$ and one more simple closed path Γ (which might be an oval if d is even, or pseudo-line if d is odd). In the second case, Γ is certainly infinite. We will assume our curve is not just a collection of isolated points and Γ is an oval (which is not an isolated point and, therefore, infinite in the first case). We pick a random point on each oval Ω_i but not a singular point (therefore not on any other Ω_i).

We now pick $\frac{d(d-1)}{2} - 1 - m$ random points on Γ; this number is positive since $d > 3$. Again, choosing randomly ensures that these points are not also on any of the Ω_i.

So we have $\frac{d(d-1)}{2} - 1$ points, and by our global interpolation method from Chapter 1, this is a curve $g = 0$ of degree $d - 2$ through these points. By our construction, $f = 0$ and $g = 0$ cannot share a common component, but by Property 1 (Section 8.5), the total intersection multiplicity of $g = 0$ with each oval $\Omega_1, \ldots, \Omega_m$ must be even, so $g = 0$ intersects each of these ovals in at least two points.

Altogether, by letting the Wolfram Language do the algebra, we have the following points of intersection of $f = 0$, $g = 0$:

```
In[ ]:= Expand[2 m + d (d - 1) / 2 - 1 - m]
Out[ ]= 1 - 2 d + d^2
```

By Bézout's theorem, we can have the following points:

```
In[ ]:= Expand[d (d - 2)]
Out[ ]= -2 d + d^2
```

We counted at least one more point than we were supposed to have, so our supposition above cannot be true.

This leads to Harnack's theorem: *let $f = 0$ be an algebraic curve of degree d, which is not a union of isolated points. The number of topological connected components of this curve in the real projective plane is less than or equal to $\frac{(d-1)(d-2)}{2} + 1$. Moreover, if singular $f = 0$ admits a decomposition into disjoint, except at the singularities, simple closed curves, satisfying Property 3 (Section 8.5), then the number of simple closed curves in this decomposition is also less than or equal to $\frac{(d-1)(d-2)}{2} + 1$.*

Our argument follows the argument in Bochnak's book (1998) except—as noted above—we can slightly perturb a singular curve to get our decomposition into ovals, satisfying our properties in Section 8.5, and the second part only requires these properties.

Other Issues

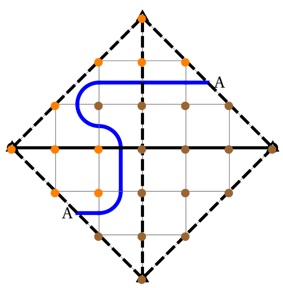
There are two other issues in this Chapter 9 which do not affect the fixed print editions but which will or have been corrected in the notebook versions on my website, but possibly not in the notebooks on wolfram.com. There is also a correction to the Chebyshev example in this chapter.

One is that the labeling primitives for infinite points of diamond diagrams no longer work in Mathematica 12. But it is better and easier just to use the symbolic graphics primitive `Text.`

The other is that some small changes have been made in the software for Chapter 9. Diamond diagrams take an association as input, these must include keys with values 0,1 for each pair $\{a, b\}$ with |a|+|b|≤ d where d is the size of the diagram. The function `dDiagram` now checks this condition and aborts with an error message if this condition is not satisfied. In particular Viro Diagrams work only with full polynomials, that is polynomials in which every monomial of degree less than or equal some fixed d has a non-zero coefficient. For sparse polynomials one uses instead a Gauss Diagram. The function `gaussDiagram` has been added to the Global Functions notebook on my website to work in the sparse case. Here is an example.

```
In[ ]:= f = y^3 - x^3 - 2 x + 3 x y - 2;
Show[gaussDiagram[f, {1, 2, 4}, x, y], Graphics[{Black, Text["A", {1.75, 1.5}], Text["A", {-1.75, -1.5}]}]]
```

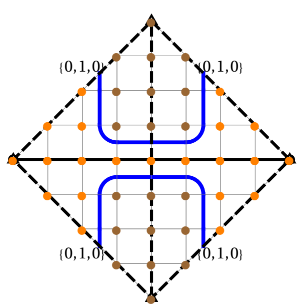
Out[ ]=



A second example, which should be avoided, is

```
Show[gaussDiagram[x^4 - y^2, {1, 2, 4, 8}, x, y],
 Graphics[{Black, Text["{0,1,0}", {2, 2.7}], Text["{0,1,0}", {-2, 2.7}], Text["{0,1,0}", {2, -2.7}],
   Text["{0,1,0}", {-2, -2.7}]}]]
```

Out[ ]=



Thus one should avoid curves with singularities or that contain the origin {0,0} or infinite points {1,0,0} or {0,1,0}.

Here is the correction to the Chebyshev example which is sparse and singular. To make it numerical we first rotate it.

```
In[252]:= {ar, br, cr} = {0.3898838058945685`, 0.48542069774603913`, 0.49793752820382864`};

In[253]:= Rrot = RotationMatrix[ar, {br, cr, 1}]

Out[253]= {{0.936873, -0.299821, 0.179935},
  {0.324275, 0.937495, -0.126286}, {-0.130825, 0.176663, 0.975538}}

In[ ]:= f = 1 - 8 x^2 + 8 x^4 + 3 y - 4 y^3
        fn = FLT[f, Rrot, x, y]
```

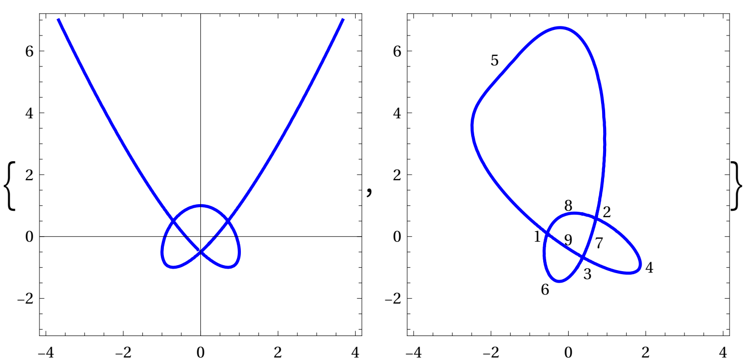
Out[ ]= $1 - 8 x^2 + 8 x^4 + 3 y - 4 y^3$

Out[ ]= $1.24825 + 1.96209 x - 5.67008 x^2 - 5.83276 x^3 + 5.95115 x^4 + 2.26778 y - 1.84109 x y - 4.03446 x^2 y + 8.5237 x^3 y - 3.55517 y^2 + 1.88663 x y^2 + 5.1695 x^2 y^2 - 2.78224 y^3 + 0.015189 x y^3 + 0.485854 y^4$

Comparing *f* with *fn*, labeling points on *fn* using drawing tools:

```
In[ ]:= {ContourPlot[f == 0, {x, -4, 4}, {y, -3, 7}, ContourStyle → Blue, Axes → True],
   Show[ContourPlot[fn == 0, {x, -4, 4}, {y, -3, 7}, ContourStyle → Blue, Axes → False],
     Graphics[{Black, Text["1", {-.8, 0}], Text["2", {1, .8}], Text["8", {0, 1}],
        Text["5", {-1.9, 5.7}], Text["3", {.5, -1.2}], Text["6", {-.6, -1.7}],
        Text["9", {0, -.1}], Text["7", {.8, -.2}], Text["4", {2.1, -1}] }]]}
```

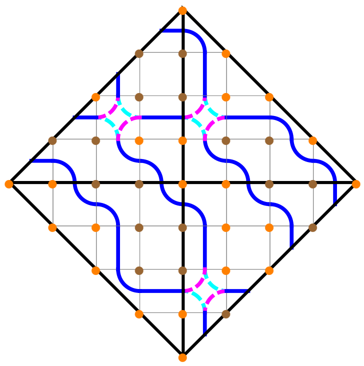
Out[ ]=



Now we find the Gauss diagram of the right-hand curve *fn*:

In[ ]:=
```
AC = coefficientSigns[fn, {x, y}];
DC = dvAssoc[AC];
T = dDiagram[DC]
```

Out[ ]=



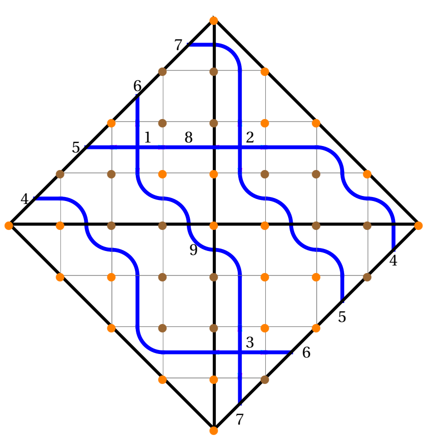
This doesn’t look much like the contour plot of *fn*, but the axis points are counted correctly and Descartes is OK with this (since the number of infinite points of *fn*, 0, is an even number less than the diagram).

Since we have a singular curve with normal crossings, we replace the ambiguous squares with singular squares.

Then label infinite points and the singular points to get a Gauss Diagram

In[ ]:=
```
Show[T, pickSingularity[-2, 1], pickSingularity[0, 1], pickSingularity[0, -3],
 Graphics[{Black, Text["4", {3.5, -.7}], Text["5", {2.5, -1.8}],
   Text["6", {1.8, -2.5}], Text["7", {.5, -3.8}],  Text["4", {-3.7, .5}],
   Text["5", {-2.7, 1.5}], Text["6", {-1.5, 2.7}], Text["7", {-.7, 3.5}],
   Text["2", {.7, 1.7}], Text["1", {-1.3, 1.7}], Text["3", {.7, -2.3}],
   Text["9", {-.4, -.5}], Text["8", {-.5, 1.7}]}]]
```

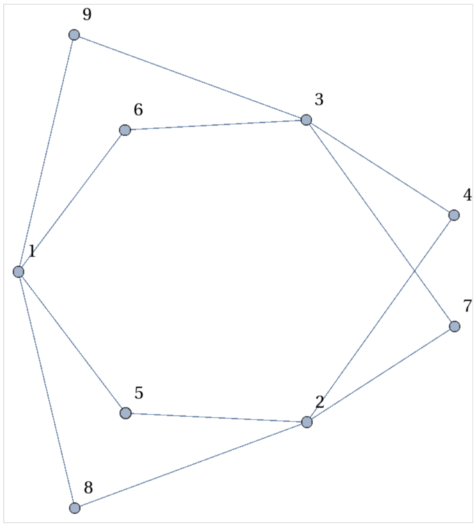
Out[ ]=



The surprise is that if we construct the projective Euler graph (Chapter 5) from the data from the contour plot or the data from the diamond diagram above, then it is the same graph:

In[ ]:=
```
Graph[{1 → 5, 5 → 2, 2 → 7, 7 → 3, 3 → 6, 6 → 1, 1 → 8, 8 → 2, 2 → 4, 4 → 3, 3 → 9, 9 → 1},
 VertexLabels → "Name", DirectedEdges → False]
```

Out[ ]=



The diamond diagram above has the same projective topology as the curve but not the same affine topology.