Using Gröbner Basis Computations to Teach Algebra

Michael Bulmer
Department of Mathematics
The University of Queensland
mrb@maths.uq.edu.au

Abstract

We give an example of an educational approach based on the philosophy that the best way to learn and understanding a topic is to teach it. Here we encourage students to think about algebraic representations and manipulations by teaching a computer how to simplify systems of polynomial equations. This is motivated and applied to the automated proof of geometry theorems.

Introduction

Students usually leave secondary school with skills in solving simultaneous equations, even systems with polynomial equations such as are needed for finding the intersections of a line with a circle. Examples like this give an understanding that a system of equations may have no solution, or may have more than one solution. However, it is rare that students will work with polynomial systems that are underdetermined, where they cannot actually "solve" the equations but must instead find some reduced form.

In early tertiary life students then learn about matrix algebra and vector spaces, seeing a general approach to solving systems of linear equations, including the subspaces generated by underdetermined systems. But little is then done about systems with higher degree polynomials, except perhaps the numerical solution of specific systems. This is partly due to the fact that a new language is required, replacing vector spaces with ideals and varieties.

However, there is a simple way to introduce these ideas. Computer algebra systems like Mathematica and Maple include functions for solving systems of polynomial equations. An excellent means of understanding something is to try and teach it to someone else, or something else, in the case of a computer. So how would we teach a computer to solve polynomial equations? This is an ideal reflective exercise for students to think about the algebraic skills they have come to take for granted. They quickly see that there are two steps involved. Firstly, the system is simplified as much as possible using symbol manipulation. Then rules like the quadratic formula or numerical methods are applied to the simpler form to get an actual solution.

For underdetermined systems, we can simply leave out the second step in this process and concentrate on the simplification. Again the need to teach a computer how to do this motivates a deeper understanding of the algebra involved, and leads to a description of Gröbner bases and Buchberger's algorithm. A natural application of these ideas, again drawing on a student's background, is in the automated proof of geometry theorems. This is naturally an underdetermined setting since we want to prove general theorems.

This approach has been used with tertiary students having a range of backgrounds and interests, including engineering, education, and finance. The next sections present the teaching sequence used with these students, starting with the problem of simplifying equations and then moving onto geometry theorems. The material for this development is drawn from Fraleigh (1999), Cox *et al.* (1992), and Chou (1987). We conclude with experiences and feedback on this approach.

Systems of Polynomial Equations

Suppose we want to find the points of intersection of a circle with a straight line, described by the following equations.

$$x^2 + y^2 = 3$$

$$x - y = 2$$

Here finding the points of intersection is equivalent to solving this system of equations. Go ahead and do this, but think as you do about what you are doing. This is likely the kind of problem that you've known how to solve for years, so it is sometimes difficult to step back and think about the fundamental processes involved. A useful way of approaching this is to think how you would teach a computer to do this task.

Essentially there are two main steps here. First we take the two equations and try to simplify them as much as possible. The second equation tells us that $y = x - 2$, which we can substitute into the first to get

$$2x^2 - 4x + 1 = 0$$

Note that you had to make a decision about whether to reduce x in terms of y or y in terms of x. This *ordering* on the variables will affect the outcome of the simplification step, so it really needs to be specified.

Once we have simplified as far as we can, the next step is to actually try and find values for the variables. In this case we can use the quadratic formula to find values for x. This will give exact values, $x = 1 \pm \text{Sqrt}(2)/2$, but in general this step is a *numerical* process, in comparison to the *symbolic* process of simplifying the equations. We will focus on the symbol processing and see how it can be described in general.

To start with, we can always write our equations as polynomials taken to be equal to 0. For example, the original system above can be written as the set of polynomials

$$F = \{ x^2 + y^2 - 3, x - y - 2 \}$$

The simplification process is then a function that takes this F and returns a new set G. This G should naturally be equivalent to F in that any common root of the polynomials in F should be a common root of the polynomials in G, and vice versa. But what criterion should we use to decide that a G is simpler? Think about possible choices and how you might teach a computer to find the corresponding simplest set. Before describing one particular criterion, we will first motivate it by looking at geometry theorems.

Equations for Geometry

We would like to use polynomial simplification to prove a theorem like the following:

Theorem (Parallel Pappus) Suppose points A, B, and C are collinear and points P, Q, and R are collinear. Furthermore, suppose AQ is parallel to BR and BP is parallel to CQ. Then AP is parallel to CR.

As an exercise, draw a picture for the Pappus theorem and try to prove it in some way. Proving geometry theorems has always been a classical application of mathematical reasoning, and this is one of the few areas that computers have been able to achieve the same results as human intelligence.

In the previous section we looked at finding the points of intersection of a line and a circle, in fact the points of intersection of a particular line and a particular circle. It is because we had a particular line and circle in mind that we were able to find solutions. However, for geometry proofs we want to talk about *all* configurations of points that satisfy a theorem's hypotheses and so we cannot work with actual numbers. This is the reason why we are focussing on simplifying equations rather than on finding numerical solutions. It illustrates well the basic role of algebra in capturing relationships between quantities. Here we are forced to think about these relationships rather than leaping straight in to find a solution.

To turn the Pappus theorem into a polynomial problem, consider the general geometric statement that the line AB is parallel to the line CD. Algebraically, this means that the slope of AB is the same as the slope of CD, so that

$$(y_B - y_A)/(x_B - x_A) = (y_D - y_C)/(x_D - x_C),$$

or, as a polynomial equal to 0,

$$(y_B - y_A)(x_D - x_C) - (y_D - y_C)(x_B - x_A).$$

In this way we can represent the hypotheses of a geometry theorem by giving polynomials which say what conditions the points must satisfy. To prove a theorem is true all we need to show is that the set of points which satisfy all of the hypotheses polynomials then also satisfy the conclusion polynomial. It turns out we can do this by showing that the conclusion polynomial can be shown to be equal to 0 using the hypotheses polynomials. (See Cox *et al.* (1992) for details of the link between these algebraic *varieties* and *ideals*.) A choice of a "simple" set of polynomials is one that makes the task of checking if a polynomial is 0 easy.

Again the actual G we obtain will depend on the ordering we specify. In the circle and line example we only needed to decide on an ordering of x and y, but in general we must specify an ordering for all

monomials. There are a number of ways of doing this based on an ordering of x and y. One is a *total degree* ordering, where a monomial is bigger than another if its degree is bigger. For example, xy^2 is bigger than xy since its degree is 3 compared to 2. If the monomials have the same degree, then we look at them lexicographically in relation to the variable ordering. If we order y above x, then we would order y^3 above xy^2 because in a dictionary with y before x, y^3 would appear before xy^2. Once we have an ordering, we can identify the *leading term* of a polynomial f as the monomial in f of the highest order.

We are now ready to describe how we'd get a computer to simplify polynomials. Consider the set of polynomials

$$F = \{y^3 - 2xy - 2, xy^2 + y - 2x^2\}.$$

To decide if some polynomial g is 0 according to the members of F we need to be able to substitute and rearrange g to try and get 0. However, here we could work by replacing y by $2x^2 - xy^2$, or by replacing xy^2 with $2x^2 - y$, or a range of other possibilities. It is a decision that requires some judgement and a sense for what will get us closer to 0. This is too hard to teach a computer. Instead we would like the computer to use our ordering and simply try to replace the leading term of each polynomial wherever it finds it. That is, we would always replace y^3 by $2xy + 2$ and xy^2 by $2x^2 – y$. This is then an *algorithm*, a procedure the computer can perform mechanically.

So why isn't F above already "simple" in this sense? The problem is that restricting ourselves to just replacing in one direction doesn't allow us to reduce some polynomials to 0 even though they are equivalent to 0 under F. For example, consider the polynomial

$$x(y^3 - 2xy - 2) - y(xy^2 + y - 2x^2) \qquad (1)$$

This polynomial is 0 since the polynomials in brackets are both in F. (Think about why we chose this and why we multiplied by x and y, respectively.) However, when we expand out the terms we /R9 Td□(F)Tj

Gröbner bases have two important uses for us. Firstly, they give an algorithm for deciding whether a given polynomial is 0 given that a set of polynomials are all 0. In the above example, G tells us that all we have to do is replace all x^3 by -1/8 and all y by $4x^2$. If the result is 0 then the answer is "yes", otherwise the answer is "no". This is exactly what we want for proving geometry theorems.

Secondly, a Gröbner basis is a simplified form of the original equations. We could now solve the above system since x is any cube root of -1/8. Indeed computer algebra systems like Maple and Mathematica implicitly use Gröbner basis in functions such as `solve`. However, they can also calculate them explicitly, and we will use this feature for working with geometry theorems.

In Maple you start by loading the Gröbner package:

```
> with(Groebner);
```

A total degree ordering can be specified in Maple using `tdeg(x,y)`. The Gröbner basis can then be computed using

```
> gbasis({x-y-2,x^2+y^2-3},tdeg(y,x));
```

This should give the polynomials equivalent to the simplified equations for the original circle and line problem.

We can then use `gbasis` to test whether the points that satisfy all the hypotheses in the Pappus theorem then also satisfy the conclusion.

In Maple we can define a function `parallel` to make entering theorems easier:

```
> parallel := (a,b,c,d) -> (y.b-y.a)*(x.d-x.c) - (y.d-y.c)*(x.b-x.a);
```

We can then define the collinear function by noting that A, B, and C lie on the same line if and only if AB is parallel to BC. Thus we can define

```
> collinear := (a,b,c) -> parallel(a,b,b,c);
```

We can then group all four hypotheses together and use `gbasis`

```
> pappusHyps := {collinear(A,B,C), collinear(P,Q,R),
    parallel(A,Q,B,R), parallel(B,P,C,Q)};
> pappusOrder := tdeg(xA,xB,xC,xP,xQ,xR,yA,yB,yC,yP,yQ,yR);
> pappusGB := gbasis(pappusHyps, pappusOrder);
```

It is interesting to expand `pappusHyps` and then compare it with `pappusGB`. The latter certainly doesn't look "simpler"; it takes up a couple of pages! However, it is simpler in the sense that it gives a straightforward algorithm for deciding whether another polynomial must be 0. We can prove the theorem by using the basis in this way to reduce the conclusion polynomial:

```
> normalf(parallel(A,P,C,R),pappusGB,pappusOrder);
```

This should give 0, indicating that the conclusion is always 0 if the hypotheses hold.

This is a simple approach to proving geometry theorems but there turns out to be a more powerful approach.

Proof by Contradiction

An alternative method of proof to direct reduction is proof by contradiction. As an example, consider the following theorem that caused a stir in Ancient Greece:

Theorem Sqrt(2) is irrational.

Proof Suppose Sqrt(2) is rational, so that Sqrt(2) = m/n for some integers m and n and suppose that m and n have no common factor (since otherwise we could simplify the fraction). Then $m^2/n^2 = 2$, so $m^2 = 2n^2$. Since m^2 is twice another number it must be even, so then m is also even (since the square of an odd number is odd). In that case, we can write $m = 2p$, for some integer p, so that $4p^2 = 2n^2$, which implies that $n^2 = 2p^2$. Thus n^2 is an even number, and so again n must be even. Hence both m and n are divisible by 2, contradicting our assumption that they have no common factor. We therefore cannot write Sqrt(2) as a fraction so it must be irrational.

This proof that Sqrt(2) is irrational involves assuming Sqrt is actually not irrational and then deducing something from that assumption that is obviously false. Then it cannot be not irrational and so it must be rational. This is the basic idea of proof by contradiction. (It is also used inside the proof - can you see where?)

In general, suppose we have some hypotheses H and we want to prove a conclusion c. (In the above example, H is the statement that Sqrt(2)$^2 = 2$ and c is the statement that Sqrt(2) is irrational.) Proof by contradiction involves taking H together with the negation of the conclusion c and trying to show that they are equivalent to something false. We will use this exact process with our polynomials, using `gbasis` to show that the set of polynomials are equivalent to the single polynomial 1 (that is, the equation 1 = 0, which is false).

All we need now is way of negating the conclusion, in this case the polynomial for `parallel(A,P,C,R)`. Call this polynomial p and consider the new polynomial pz - 1, where z is a new variable we have introduced. Remember that these polynomials capture the equations $p = 0$ and $pz - 1 = 0$. Now if $p = 0$ then $pz = 0$, so the second equation becomes 1 = 0, which is false. Conversely, if $pz - 1 = 0$ then $p = 0$ must be false. So pz - 1 is the negative form of p we need for proof by contradiction. (In fact, this approach is related to the deeper result of Hilbert's Nullstellensatz; see Cox *et al.* (1992) for details.)

Now we can prove the Parallel Pappus theorem by contradiction. Try the following in Maple:

```
> pappusThm := {collinear(A,B,C), collinear(P,Q,R),
    parallel(A,Q,B,R), parallel(B,P,C,Q), parallel(A,P,C,R)*z-1};
> pappusOrder := tdeg(xA,xB,xC,xP,xQ,xR,yA,yB,yC,yP,yQ,yR,z);
> pappusGB := gbasis(pappusThm, pappusOrder);
```

If all is well you should get the single polynomial 1.

Finally, consider what appears to be a simpler theorem:

Theorem (Collinearity) Suppose points A, B, and C are collinear and points A, B, and D are collinear. Then B, C, and D are collinear.

Draw a picture of this configuration and think about whether the theorem is true. Trying to prove it by contradiction using Maple, we find that we don't get 1 but instead get a set of three polynomials including `xA - xB`. So the theorem isn't true, but why? We have formed the negation of the theorem, by adding the negated conclusion, and we are trying to show that it is equivalent to false. Thus the theorem would be true if the Gröbner basis was false. Now the basis is a collection of polynomials which we interpret as a conjunction of conditions. It will thus be false if any one of its members is false. That is, the basis will be false and so the theorem will be true if x_A - x_B was not 0. This means that the theorem will be true if x_A is not x_B. Geometrically, this is requiring that the points A and B be different. If you didn't see this earlier, draw a picture of the configuration with A and B the same point; it should be clear then why this theorem is not necessarily true.

The proof by contradiction approach, using the notion of simplifying systems of polynomials, is thus quite remarkable. It can be used to prove theorems but more significantly it can tell you what is wrong if the theorem isn't true. Further details of this and other approaches, as well as an encyclopaedic collection of geometry theorems that have been proved using the algebraic approach, are given by Chou (1987).

Conclusions

The above sequence has been used in an introductory scientific computation course, to compare with the numerical methods that typically dominate the material, and in an advanced operations research course, to motivate alternative ways of thinking about problems. In both settings the student feedback was very positive, especially after seeing the unexpected collinearity theorem. Students were keen to try proving other theorems using the approach and quickly realised the usefulness of the diagnostic feature of proof by contradiction. For example, suppose B and C are points on a circle, at each end of a diameter through the centre A. Students would often capture this by saying that B, A, and C were collinear and that the distance AB was the same as the distance AC. They would quickly discover that this was not sufficient; these conditions would be satisfied if B and C were the same point.

The task was also handled well by students with little background in symbolic algebra packages or in computers in general. Only a few Maple functions were used, and no programming structures like loops and conditionals were required. For this reason, the approach has also been presented to practicing secondary teachers.

In general terms, most student experience with packages like Maple involves calculus-based work, replacing or extending traditional teaching in differentiation and integration. In some sense this is rather wasteful since a lot of these methods could simply be done numerically and the extension of symbolic approaches is less useful for the majority of students (Bulmer, 2001). The exploration of Gröbner bases and geometry theorems puts the emphasis strongly on the representational nature of algebraic relations, the universality of polynomial equations, and the idea of what simplification means. These issues, and the notion that you might not actually want to get numerical solutions, give students a richer insight into the nature of algebraic reasoning.

References

Bulmer, M. (2001) Algebra in an age of numerical mathematics. The future of the teaching and learning of algebra. Twelfth ICMI study. University of Melbourne, December 10-14.

Chou, S.-C. (1987) *Mechanical Geometry Theorem Proving*. Holland: D. Reidel Publishing.

Cox, D., Little, J. & O'Shea, D. (1992) *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer-Verlag.

Fraleigh, J.B. (1999) *A first course in abstract algebra*. Sixth edition. Addison-Wesley.