PREMUR 2007 Seminar
Week 3 Computer Laboratory Exercises

Background and Goals

This week, we will work through several applications of Buchberger's Algorithm for Gröbner bases, especially making use of the elimination properties of lex Groöbner bases. Then we will see how to compute some resultants using Mathematica as well.

Days 1,2 – A "sampler" of applications of Gröbner bases

Exercise 1 – Solving Systems of Equations

In this problem, you will find all of the points in

$$V = \mathbf{V}(x^2y - z^3, 2xy - 4z - x, z - y^2 + x - 1)$$

in $\mathbf{R}^3$, by following these steps:

a) First, compute a *lex* Gröbner basis (say with $x > y > z$), and call it H (or any other name you want!).

b) Identify a polynomial containing only z in your output; it should be the first polynomial in the list, or `H[[1]]`. Solve the equation obtained by setting that polynomial equal to zero using a command like

```
zroots = NSolve[H[[1]],z]
```

(The `NSolve` command uses an approximate *numerical method* to solve equations and finds approximations to all of the complex roots of the equation.) The output is a list of *substitutions* of the form

```
{z -> value}.
```

c) You can now substitute each real root back into the other equations to solve determine whether it is possible to solve for x, y and in how many different ways. The mathematics behind this is the statement of the Extension Theorem. The feature of Mathematica that will be most useful here is the /. ("apply rule") construction. For instance

```
H/.zroots[[1]]
```

will substitute the first approximate z-value into all the polynomials in the Gröbner basis H. You can then use the Extension Theorem to decide how many points (x, y, z) in V contain that z-value. (Note that the substitution may not give exactly zero in the first polynomial. However the value there should be a very small number in absolute value, so we will treat it as approximately zero.)

d) How many real solutions are there all together? (How are you counting?)

Exercise 2 – Implicitization of parametric curves and surfaces

Implicitization is the process of converting a parametric representation of a curve or surface into an implicit representation for a variety containing that curve or surface. This can also be done via Gröbner bases because of the elimination properties of the lex and other monomial orders. See the discussion in §8 of Chapter 2.

a) Problem 7, Chapter 2, §8 of "IVA".
b) Problem 8, Chapter 2, §8 of "IVA".
c) Problem 9, Chapter 2, §8 of "IVA".
d) Generate a composite graphics image showing the parametric curve from problem 9, and an implicit plot of the surface from problem 8. How does this relate to your answer to part c of Problem 9?

Exercise 3 – The envelope of a family of curves

A family of plane curves can be defined by a polynomial $F(x, y, t)$. For each real value $t = t_0$, the variety $\mathbf{V}(F(x, y, t_0))$ will be one of the curves in the family. The *envelope* of a family can be thought of in very intuitive terms as another curve tangent to each of the curves in the family (at some point). In many cases, the envelope will appear as something like the "boundary" of the region of the plane containing the curves in the family, although this will not always be very obvious. One way to define the envelope of the family is the following: It is the set of all $(x, y) \in \mathbf{R}^2$ such that

$$\begin{aligned} F(x, y, t) &= 0 \\ \frac{\partial}{\partial t} F(x, y, t) &= 0 \end{aligned}$$

for some t. This means we want to think of projecting the variety

$$\mathbf{V}\left(F, \frac{\partial F}{\partial t}\right)$$

to the x, y-plane, and the image will be contained in the variety of the elimination ideal

$$\left\langle F, \frac{\partial F}{\partial t} \right\rangle \cap \mathbf{R}[x, y]. \tag{1}$$

In this problem you will study the envelope of the family of circles of radius 1 defined by

$$F(x, y, t) = (x - t)^2 + (y - t^3)^2 - 1 = 0$$

a) Generate a plot showing the circles in the family for t increasing from $t = -2$ to $t = 2$ in steps of .25. (There are a number of ways to get Mathematica to draw these, parametric plotting of the circles is probably the best way to get good pictures. The

`Table` command may be useful as a way to build up lists of specifications of curves to feed to `ParametricPlot`.)

b) Compute the elimination ideal from (1) for this family using `GroebnerBasis`. You should find that it is generated by one polynomial $E(x, y)$.

c) Generate an implicit plot of the envelope equation, and then a plot showing the circles in the family together with the envelope equation.

d) If you look closely at the plot of the envelope curve, you might notice that there seem to be two small triangular "loops" on the curve where it does not have a nice tangent line. These are called *singular points*. They will be found at points where, simultaneously,

$$E(x,y) = 0, \qquad \frac{\partial E}{\partial x}(x,y) = 0, \qquad \frac{\partial E}{\partial y}(x,y) = 0.$$

By techniques like the ones we saw in Exercise 1 above, determine all singular points on the envelope curve $\mathbf{V}(E(x,y))$.

Day 3 – Resultants

As we have seen, the resultant of two polynomials $f(x), g(x)$ also gives a way to test whether f and g have any common factors of positive degree in x. If the coefficients of f, g contain other variables besides x, so

$$f, g \in k[x, y_1, \ldots, y_n],$$

then the resultant also gives a definite *formula* for producing an element of the elimination ideal

$$\langle f, g\rangle \cap k[y_1, \ldots, y_n].$$

For simple elimination tasks, this often gives a more *efficient* method than computing a Gröbner basis, although the efficiency sometimes comes with a *cost* too – although the resultant is an element of the elimination ideal we want, it need not be a generator.

The Mathematic command for the basic resulant has the format

```
Resultant[f,g,var]
```

where f, g are two polynomials and `var` is a variable. You may wish to check the online documentation, but this one is quite straight-forward to use.

Exercise 4

Do problem 5 in Chapter 3, §5 of "IVA".

Exercise 5

We mentioned above that $\text{Res}(f, g, x)$ need not generate $\langle f, g\rangle \cap k[y_1, \ldots, y_n]$ when $f, g \in k[x, y_1, \ldots, y_n]$. Do problem 3 in Chapter 3, §6 of "IVA" to see an explicit example of this behavior.

Exercise 6

In the discussion today or tomorrow, we study the system of polynomials

$$\begin{aligned} f_1 &= x^4 - 2xy^2 + zw \\ f_2 &= wx^2 - w^2 z + y \\ f_3 &= x^3 + 3w \end{aligned}$$

and compute the generalized resultants of f_1, f_2, f_3 with respect to w, the polynomials h_α in the expansion

$$\text{Res}(f_1, u_1 f_2 + u_2 f_3, w) = \sum_\alpha h_\alpha(x, y, z) u^\alpha.$$

a) Use Mathematica to compute this (and collect all terms containing each fixed u^α to find the $h_\alpha(x, y, z)$.

b) Also show that the generalized resultants *do not* generate the elimination ideal

$$\langle f_1, f_2, f_3 \rangle \cap k[x, y, z]$$

in this case. You will need a Gröbner basis for the ideal membership test.

Day 4 – "Mini-Project"

You have just discovered the remnants of an ancient alien civilization in a subterranean complex. Near the entrance there are two large rooms connected by a short passageway 3 feet wide and 4 feet long. In a convenient set of coordinates, the relevant parts of the walls are formed by straight line segments

$$\begin{gathered} (-5, 3/2) \text{ to } (2, 3/2) \\ (2, 3/2) \text{ to } (2, 5) \\ (5, -3/2) \text{ to } (-2, -3/2) \\ (-2, -3/2) \text{ to } (-2, -5) \end{gathered}$$

(Note that there are right-angle corners at $(2, 3/2)$ and $(-2, -3/2)$.) In Room 2, at the point $A = (7/2, 7/2)$, you have discovered a functional, but very fragile, robot. It has a horizontal cross-section that is a circle of diameter 2 feet. The robot will accept movement commands, but will not let itself be moved by any other means. Your job is to figure out a way (any way) to maneuver the robot from point A to point $B = (-7/2, -7/2)$, subject to the following constraints:

1) The aliens were apparently very clever mathematicians, but the robot's control program is *very limited* – the way you must specify its motion is to give a single polynomial parametric curve $x = f(t), y = g(t)$ that traces the path of the *center* of the robot's circular cross-section. Only one curve is allowed – no piecewise polynomial paths.

2) The robot will be damaged beyond repair if it runs into one of the walls. If it does, you will have destroyed the most amazing archeological discovery of all time. *So be careful!*
3) Fortunately, you get to "simulate" possible motions using Mathematica as often as you like until you find one that works, and then you can give the information to the robot.

For this problem you will need to generate a Mathematica notebook showing the rooms and corridor, and a display showing that your path will keep the robot from hitting the walls at any point. And, *give a justification for why your path will not run the robot into the walls.* (By this I mean more evidence than the picture – for instance, if feasible, you could find the point on the envelope that is closest to the walls, and show that the distance at the closest point is strictly positive.)

Suggestions

1) A good first step will be to set up a plot showing the rooms, etc. You can also experiment and determine some paths from point A to point B to try.
2) The discussion of envelopes is relevant here! (Do you see why?)
3) *Important Note:* You will need to be somewhat clever in how you apply the techniques we have discussed – if you use a "brute force" method, to compute the envelope for some "obvious" choices of the path, then you can easily end up giving Mathematica a Gröbner basis calculation that will use up an hour of calculation and tie up the network server without producing any results(!)
4) Any mathematics we have discussed so far this summer is "fair game" here. There are several things you can experiment with:
   - If you need to do something that requires elimination of variables, Gröbner bases or resultants are possible.
   - You can try different monomial orders if you decide to use Gröbner bases.
   - You can reduce the number of variables by looking at one "slice" $x = const$ or $y = const$ at a time. This generally speeds things up *a lot*, but you will need to decide how to use these computations to make sure the robot follows an acceptable path.

   However you approach the problem, you will need *give a justification for why your path will not run the robot into the walls.*