Appendix D

Independent Projects

Unlike the rest of the book, this appendix is addressed to the instructor. We will discuss several ideas for computer projects or research papers based on topics introduced in the text.

§1 General Comments

Independent projects can be valuable for a variety of reasons:

- The projects get the students to actively understand and apply the ideas presented in the text.
- The projects expose students to the next steps in subjects discussed in the text.
- The projects give students more experience and sophistication as users of computer algebra systems.

Projects of this type are also excellent opportunities for small groups of two or three students to work together and learn collaboratively.

Some of the projects given below have a large computer component, whereas others are more theoretical. The list is in no way definitive or exhaustive, and users of the text are encouraged to contact the authors with comments or suggestions concerning these or other projects they have used.

The description we give for each project is rather brief. Although references are provided, some of the descriptions would need to be expanded a bit before being given to the student.

§2 Suggested Projects

1. **Implementing the Division Algorithm in** $k[x_1, \ldots, x_n]$. Many computer algebra systems (including REDUCE and Maple) have some sort of "normal form" or "reduce" command that performs a form of the division algorithm from Chapter 2. However, those commands usually display only the remainder. Furthermore, in some cases, only certain monomial orders are allowed. The assignment here would be for the students to implement the general division algorithm, with input a polynomial f, a list of divisors F, a list of variables X, and a monomial ordering.

The output would be the quotients and the remainder. This project would probably be done within a computer algebra system such as Maple or Mathematica.

2. **Implementing Buchberger's Algorithm**. Many computer algebra systems have commands that compute a reduced Groebner basis of an ideal $\langle f_1, \ldots, f_s \rangle$. This project would involve implementing the algorithm in a way that produces more information and (possibly) allows more monomial orderings to be used. Namely, given the input of a list of polynomials F, a list of variables X, and a monomial order in $k[x_1, \ldots, x_n]$, the program should produce a reduced Groebner basis G for the ideal generated by F, together with a matrix of polynomials A expressing the elements of the Groebner basis in terms of the original generators $G = AF$. As with the previous project, this would be done within a computer algebra system. The program could also give additional information, such as the number of remainders computed at each stage of the algorithm.
3. **The Complexity of the Ideal Membership Problem**. In §9 of Chapter 2, we briefly discussed some of the worst-case complexity results concerning the computation of Groebner bases and solving the ideal membership problem. The purpose of this project would be to have the students learn about the Mayr and Meyer examples, and understand the *double exponential* growth of degree bounds for the ideal membership problem. A suggested reference here is BAYER and STILLMAN (1988) which gives a nice exposition of these results. With some guidance, this paper is accessible reading for strong undergraduate students.
4. **Solving Polynomial Equations**. For students with some exposure to numerical techniques for solving polynomial equations, an excellent project would be to implement the criterion given in Theorem 6 of Chapter 5, §3 to determine whether a system of polynomial equations has only finitely many solutions over $\mathbb{C}$. If so, the program should determine all the solutions to some specified precision. This would be done by using numerical techniques to solve for one variable at a time from a lexicographic Groebner basis. A comparison between this method and more standard methods such as the multivariable Newton's Method could also be made. As of this writing, very little theoretical work comparing the complexity of these approaches has been done.
5. **Groebner Basis Conversion for Zero-Dimensional Ideals**. As in the previous project, to solve systems of equations, lexicographic Groebner bases are often the most useful bases because of their desirable elimination properties. However, lexicographic Groebner bases are often more difficult to compute than Groebner bases for other monomial orderings. For zero-dimensional ideals (i.e., $I \subset \mathbb{C}[x_1, \cdots, x_n]$ such that $\mathbf{V}(I)$ is a finite set), there are methods known for *converting* a Groebner basis with respect to some other order into a lexicographic Groebner basis. For this project, students would learn about these methods, and possibly implement them. There is a good introductory dicussion of these ideas in HOFFMANN (1989). The original reference is FAUGÈRE, GIANNI, LAZARD, and MORA (1993). See also Chapter 2 of COX, LITTLE and O'SHEA (1998).
6. **Curve Singularities**. A multitude of project topics can be derived from the general topic of curve singularities, which we mentioned briefly in the text. Implementing an algorithm for finding the singular points of a curve $\mathbf{V}(f(x, y)) \subset \mathbb{R}^2$ or $\mathbb{C}^2$ could

be a first part of such a project. The focus of the project would be for students to learn some of the theoretical tools needed for a more complete understanding of curve singularities: the Newton polygon, Puiseux expansions, resolutions by quadratic transformations, etc. A good general reference for this would be BRIESKORN and KNÖRRER (1986). There are numerous other treatments in texts on algebraic curves as well. Some of this material is also discussed from the practical point of view of "curve tracing" in HOFFMANN (1989).

7. **Surface Intersections**. The focus of this project would be algorithms for obtaining equations for plane projections of the intersection curve of two surfaces $\mathbf{V}(f_1(x, y, z))$, $\mathbf{V}(f_2(x, y, z))$ in $\mathbb{R}^3$. This is a very important topic in geometric modeling. One method, based on finding a "simple" surface in the pencil defined by the two given surfaces and which uses the projective closures of the two surfaces, is sketched in HOFFMANN (1989). Another method is discussed in GARRITY and WARREN (1989).
8. **Bézier Splines**. The Bézier cubics introduced in Chapter 1, §3 are typically used to describe shapes in geometric modeling as follows. To model a curved shape, we divide it into some number of smaller segments, then use a Bézier cubic to match each smaller segment as closely as possible. The result is a *piecewise* Bézier curve, or Bézier spline. For this project, the goal would be to implement a system that would allow a user to input some number of control points describing the shape of the curve desired and to see the corresponding Bézier spline curve displayed. Another interesting portion of this assignment would be to implement an algorithm to determine the *intersection points* of two Bézier splines. Some references can be found on p. xvi of FARIN (1990). We note that there has also been some recent theoretical work by BILLERA and ROSE (1989) that applies Groebner basis methods to the problem of determining the vector space dimension of multivariate polynomial splines of a given degree on a given polyhedral decomposition of a region in $\mathbb{R}^n$. See also Chapter 8 of COX, LITTLE and O'SHEA (1998).
9. **The General Version of Wu's Method**. In our discussion of Wu's method in geometric theorem proving in Chapter 6, we did not introduce the general algebraic techniques (characteristic sets, Ritt's decomposition algorithm) that are needed for a general theorem-prover. This project would involve researching and presenting these methods. Implementing them in a computer algebra system would also be a possibility. See CHOU (1988), MISHRA (1993), WANG (1994a) and (1994b), and WU (1983).
10. **Molien's Theorem**. An interesting project could be built around Molien's Theorem in invariant theory, which is mentioned in §3 of Chapter 7. The algorithm given in STURMFELS (1991) could be implemented to find a set of generators for $k[x_1, \ldots, x_n]^G$. This could be applied to find the invariants of some larger groups, such as the rotation group of the cube in $\mathbb{R}^3$. Molien's theorem is also discussed in Chapter 7 of BENSON and GROVE (1985).
11. **Groebner Bases over More General Fields**. For students who know some field theory, a good project would be to compute Groebner bases over fields other than $\mathbb{Q}$. For example, one can compute Groebner bases for polynomials with coefficients in $\mathbb{Q}(i)$ using the variable i and the equation $i^2 + 1 = 0$. More generally, if $\mathbb{Q}(\alpha)$ is

any finite extension of $\mathbb{Q}$, the same method works provided one knows the minimal polynomial of α over $\mathbb{Q}$. The needed field theory may be found in Sections 5.1, 5.3, and 5.5 of HERSTEIN (1975). The more advanced version of this project would discuss Groebner bases over finite extensions of $\mathbb{Q}(u_1, \ldots, u_m)$. In this way, one could compute Groebner bases over any finitely generated extension of $\mathbb{Q}$.

12. **Computer Graphics**. In §1 of Chapter 8, we used certain kinds of projections when we discussed how to draw a picture of a 3-dimensional object. These ideas are very important in computer graphics. The student could describe various projections that are commonly used in computer graphics and explain what they have to do with projective space. If you look at the formulas in Chapter 6 of FOLEY, VAN DAM, FEINER and HUGHES (1990), you will see certain 4×4 matrices. This is because points in $\mathbb{P}^3$ have four homogeneous coordinates!
13. **Implicitization via Resultants**. As mentioned in Chapter 3, §3, implicitization can be done using resultants rather than Groebner bases. A nice project would be to report on the papers ANDERSON, GOLDMAN and SEDERBERG (1984a) and (1984b), and MANOCHA (1992). The resultants used in these papers differ from the resultants discussed in Chapter 3, where we defined the resultant of two polynomials. For implicitization, one needs the resultant of three or more polynomials, often called *multipolynomial resultants*. These resultants are discussed in BAJAJ, GARRITY and WARREN (1988) and Cox, LITTLE and O'SHEA (1998).
14. **Optimal Variable Orderings**. There are situations where reordering the variables (but keeping the same type of term order) can have a strong effect on the Groebner basis produced. For example, in part (a) of Exercise 13 from Chapter 2, §9, you computed a rather complicated Groebner basis using lex order with $x > y > z$. However, switching the variables to $z > y > x$ (still with lex order) leads to a much simpler Groebner basis. A heuristic algorithm for picking an optimal ordering of the variables is described in BOEGE, GEBAUER and KREDEL (1986). A good project would be to implement a straightforward version of the Buchberger algorithm which incorporates variable optimization. This algorithm is implemented in the REDUCE Groebner basis package—see Appendix C, §4.
15. **Selection Strategies in the Buchberger's Algorithm**. In the discussion following the improved Buchberger algorithm (Theorem 11 in Chapter 2, §9), we mentioned the selection strategy of choosing a pair $(i, j) \in B$ in Theorem 11 such that $\mathrm{LCM}(\mathrm{LT}(f_i), \mathrm{LT}(f_j))$ is as small as possible. This is sometimes called the *normal selection strategy*. However, there are other selection strategies which are used in practice, and a nice project would be to describe (or implement) one of these strategies. Here are two that are of interest:
    a. The concept of *sugar* was introduced in GIOVINI, MORA, NIESI, ROBBIANO and TRAVERSO (1991). This paper explains why the normal selection strategy can cause problems with non-graded monomial orders (such as lex) and defines the concept of sugar to get around this problem. Sugar is implemented in the Groebner basis commands used by most of the computer algebra systems described in Appendix C.
    b. In the special case of lex order, some other heuristics for selecting pairs are discussed in CZAPOR (1991). Here, the basic idea is to pick a pair (i, j) such that the multidegree of the S-polynomial $S(f_i, f_j)$ is as small as possible.

16. **Other Types of Groebner Bases**. In Chapter 2, we defined Groebner bases for an ideal in a polynomial ring, assuming we knew the monomial order and the coefficient field. But there are other notions of what it means to be a Groebner basis, and a good project would be for a student to explore one of these. Here are some of the more interesting types of Groebner bases:

a. We have seen that different monomial orderings can lead to different Groebner bases. As you vary over all monomial orderings, it turns out that there are only finitely many distinct Groebner bases for a given ideal. These can be put together to form what is called a *universal Groebner basis*, which is a Groebner basis for *all possible* monomial orders. A good reference (including references to the literature) is pages 514–515 of BECKER and WEISPFENNING (1993).

b. Another phenomenon (mentioned in Chapter 6, §3) is that if the base field contains parameters, then a Groebner basis over this field may fail to be a Groebner basis when we specialize the parameters to specific values. However, it is possible to construct a Groebner basis which remains a Groebner basis under *all possible* specializations. This is called a *comprehensive Groebner basis.* For a description and references to the literature, see pages 515–518 of BECKER and WEISPFENNING (1993).

c. Besides doing Groebner bases over fields, it is sometimes possible to define and compute Groebner bases for ideals in a polynomial ring $R[x_1, \ldots, x_n]$, where R is a ring. The nicest case is where R is a principal ideal domain (PID), as defined in Chapter 1, §5. The basic theory of how to do this is described in Chapter 4 of ADAMS and LOUSTAUNAU (1994) and Section 10.1 of BECKER and WEISPFENNING (1993).

d. Finally, the notion of an ideal $I \subset k[x_1, \ldots, x_n]$ can be generalized to a *module* $M \subset k[x_1, \ldots, x_n]^r$, and there is a natural way to define term orders and Groebner bases for modules. Basic definitions and interesting applications can be found in ADAMS and LOUSTAUNAU (1994), BECKER and WEISPFENNING (1993), COX, LITTLE and O'SHEA (1998), and EISENBUD (1995).

Besides the projects listed above, there are other places where instructors can look for potential projects for students, including the following:

- COX, LITTLE and O'SHEA (1998) includes chapters on local rings, algebraic coding theory and integer programming which could serve as the basis for projects. Other chapters in the book may also be suitable, depending on the interests of the students.
- ADAMS and LOUSTAUNAU (1994) contains sections on minimal polynomials of field extensions, the 3-color problem and integer programming. These could form the basis for some interesting projects.
- EISENBUD (1995) has a list of seven projects in Section 15.12. These projects are more sophisticated and require more background in commutative algebra, but they also introduce the student to some topics of current interest in algebraic geometry.

If you find student projects different from the ones listed above, we would be interested in hearing about them. There are a *lot* of wonderful things one can do with Groebner bases and algebraic geometry, and the projects described in this appendix barely scratch the surface.