Module MAT-INFO
Année 2009–2010

Chris Peters

December 9, 2009

Contents

Introduction

This course is intended for third year students with some background in algebra; they need to know:

- what a group is and be familiar with the euclidean algorithm for the integers;
- how to do long division with polynomials in one variable;
- finite fields.

The main aim of the lectures is to familiarize the students with symbolic calculations in algebra as a research tool.

I have chosen to focus on two separate public domain packages: GAP and REDUCE. The first is particularly suitable for group calculations while the second is useful for calculations in polynomial rings.

The first chapter is devoted to computational group theory where Schreier's algorithm in its most primitive form is explained. Most symbolic calculations software for groups is based on this algorithm.

To pave the way for the theory of Gröbner bases we revise Euclid's algorithm for the polynomial ring of one variable, deduce the standard consequences about ideals in that ring, discuss a possible generalization to several variable polynomials and end with a discussion of the difficulties. For general polynomial rings computations are based on finite generation of ideals and so, in chapter 3 I start with a proof of Hilbert's basis theorem. Then I show how to solve the difficulties we met with at the end of Chapter 2 using the leading term ideal and the associated Gröbner basis.

The next chapter treats how to recognize such a basis using the S-polynomials and how this leads to Buchbergers algorithm which lies at the heart of all symbolic calculation packages dealing with polynomial ideals.

In Chapter 5 I give some applications of Gröbner bases: elimination theory on the one hand and automatic theorem proving in elementary geometry on the other hand. Only a few examples are given but they are chosen in such a way that some of the essential problems show up.

Chapter 6 is about factoring integer polynomials. I explain how one can do this by first reducing modulo a suitable prime and then apply Berlekamp's algorithm. This algorithm is explained in detail. This last chapter is the most advanced and most difficult chapter in that it combines several techniques from different branches of algebra.

I want to thank Hans Sterk from the Technical University Eindhoven for help in making sensible choices for topics of such an introductory course. His notes and suggestions have been most helpful.

Chapter 1

Computational group theory: Schreier trees and algorithmic applications

1.1 Permutation Groups

We suppose that the student knows the following concepts:

- Groups, subgroups, factor groups;
- Order of an element; order of a group;
- Homomorphisms of groups, normal subgroups;

In addition, we suppose that he/she knows :

Theorem ((First) isomorphism theorem)**.** *: if $f : G_1 \to G_2$ is a surjective homorphism of groups, then* $\ker(f)$ *is a normal subgroup in G_1 and there is a canonical isomomorphism $\bar{f} : G_1/\ker(f) \xrightarrow{\sim} G_2$.*

For a set S we let $\mathsf{Sym}(S)$ be the group of all its permutations; if $S = \{1, \ldots, n\}$ this is simply the *symmetric group on n letters*, S_n. The notation of a permutation is as follows: $(n_1, \ldots, n_k)$ is the cyclic permutation sending n_1 to n_2, n_2 to n_3 etc. A permutation can always be written as a product of disjoint cyclic permutations. The convention for composing two permutation is that one reads them *from left to right*:

$$(1, 2)(2, 3) = (1, 3, 2).$$

A *permutation group* is a subgroup of some $\mathsf{Sym}(S)$. We then have an *action* of G on the set S, in fact, an *effective* action.

Example 1.1.1. The group O of the symmetries of the cube with vertices labeled $\{1, 2, 3, 4, 5, 6, 7, 8\}$. Bottom: $\{1, 2, 3, 4\}$ (in circular order) and top $\{5, 6, 7, 8\}$. There are rotational symmetries such as $(1, 2, 3, 4)(5, 6, 7, 8)$ and reflections such as $(2, 5)(3, 8)$.

Lemma 1.1.2. *Every finite group G is a permutation group.*

Proof: Let $S = G$ and assign to $g \in G$ the right-multiplication $R_g : G \to G$ (sending h to hg. Check that this gives and injective homomorphism $G \to \mathsf{Sym}(G)$. □

To analyze permutations we need some further notation. If G acts on S we let s^g be the result of applying g to s. The *orbit* s^G of $s \in S$ is the set of all s^g when g varies over G. There is a complementary notion, the *stabilizer* G_s of s: the subgroup of elements of G that leave s fixed, i.e. $g \in G_s$ if and only if $s^g = s$. Here is how these notions relate:

Proposition 1.1.3. *If a finite group G acts on a finite set S, then for all $s \in S$ we have*

$$|G|/|G_s| = |s^G|.$$

Proof: Consider

$$f : G \to s^G, \quad g \mapsto s^g.$$

It is surjective and $f(g) = f(h)$ precisely when $hg^{-1} \in G_s$, i.e. if and only if the cosets $G_s g = f^{-1}(f(g))$ and $G_s h = f^{-1}(f(g))$ coincide. This shows that s^G is in bijection to the number of right cosets $G_s\backslash G$ which is exactly equal to $|G|/|G_s|$ as stated. □

Special case of the above: $S = H\backslash G$, the right cosets of H in G with action of G given by

$$R_g : H\backslash G \to H\backslash G, \quad R_g(Hh) = Hhg.$$

Since the stabilizer of $H1$ is just H, we get

Corollary 1.1.4 (Lagrange's Theorem)**.** *If G is a finite group, H a subgroup, then $|H|$ divides $|G|$. In particular, the order of $g \in G$ is a divisor on the order of the group G.*

1.2 Computations: orbits and stabilizers

A subset $X \subset G$ of a group G is called *a generating set* (and its elements *generators of* G) if every $g \in G$ can be written as a finite products of elements from X or their inverses. We can do without the inverses if G is finite. Why? We write $G = \langle X \rangle$.

Orbit of $s \in S$

Here is an algorithm

```
Input: X,s
Output: orbit
orbit:={s};
FOR  each x in X DO
 IF s^x not= s THEN  orbit:={s^x,orbit}
 ELSE orbit:=orbit
```

The end result will be the orbit s^G.

Stabilizer of s

The algorithm uses a *Schreier tree* for the generating set X and any $\alpha \in S$ (the *root* of the tree). To construct the tree:

1. place α at the bottom;
2. let the generators $a, b, c, \ldots$ act on α. This produces new vertices one level up connected by edges labeled $a, b, c, \ldots$.
3. Then let X again act on the new vertices. Of course, α is among then, but there may be new vertices "higher up".

Be careful *not to create cycles* in this process. At the end the vertices s give the orbit of α and from the tree one can find g written in terms of X for which $s = \alpha^g$: go down from s to the root α and write down the labels on the edges in the reverse order. This gives the permutation $t(s)$ with the property $\alpha^{t(s)} = s$

Using the tree we can find a subset $Y \subset G$ generating the stabilizer G_α as follows. Indeed, for every $h \in X$ the element $t(s)h[t(s^h)]^{-1}$ first maps α to s, then to s^h and then to α, i.e. this particular *Schreier element* is in the stabilizer G_α. Graphically, h "bridges" the two branches labeled by $t(s)$ and $t(s^h)$ but is not itself an edge. But these elements suffice:

Lemma 1.2.1 (Schreier)**.** *$G_\alpha = \langle Y \rangle$, where Y is the set of the Schreier elements $t(s)h[t(s^h)]^{-1}$, $s \in \alpha^G$, $h \in X$.*

Proof: We need to show that any $g \in G_\alpha$, written as a product of generators, say $g = b_1 \cdots b_r$, $b_i \in X$ can be rewritten as a product of Schreier generators, We may suppose that $r > 0$. Consider a path $\alpha, \alpha^{b_1}, \alpha^{b_1 b_2}, \ldots, \beta = \alpha^{b_1 b_2 \cdots b_j}$ in the tree with j maximal. Then $j < r$ (otherwise we would have a loop). Note also that since b_{j+1} is a bridge $t(\beta^{b_{j+1}})$ corresponds to a path in the tree starting from the root but ending a level lying not above β. Since $\beta = \alpha^{b_1 b_2 \cdots b_j}$ we have $t(\beta) = b_1 \cdots b_j$. Consider

$$\left[t(\beta)b_{j+1}[t(\beta^{b_{j+1}}]^{-1}\right]^{-1} g = t(\beta^{b_{j+1}})b_{j+2} \cdots b_r := g_1.$$

The left hand element stabilizes α, so does the right hand and we can replace g by g_1. Repeat the procedure with g_1 and recall that $t(\beta^{b_{j+1}})$ corresponds to a path in the tree starting from the root α. The maximal such path might also include some of the "tail-elements" b_i with $i \geq j+2$, but in any case the procedure gives a $g_2 \in G_\alpha$ with a new "tail" starting at some b_t with $t \geq j+3$. Continuing this way we construct $g_3, \ldots, g_s$ where each time the remaining tail shortens and we finish when no tail is left and then $g_s = 1$. So g_{s-1} is Schreier, and going up we find recursively that $g_{s-2}, \ldots, g$ are products of Scheier elements. □

This leads to the algorithm

```
Input: X, orbit
Output: stabilizer
stabilizer={emptyset};
  FOR each x in X and i in  orbit DO
   IF  [i,x,i^b] not an edge THEN
   stabilizer:={stabilizer, t(i)b[t(i^b)]^[-1]}
   ELSE stabilizer:=stabilizer
```

Order of G

Using the two algorithms and Proposition 1.1.3 we find an algorithm to compute the order of a group acting on $S = \{1, \ldots, n\}$ as follows: Consider the orbit of 1 and count. If $G_1 = 1$ we are done; if not we look at the orbit of 2 under G_1 and continue that way.

Example 1.2.2. Return to the example of the cube. Its symmetry group $G = \mathsf{O}$ acts transitively, so $|\mathsf{O}| = 8|G_1|$. Use $(2,5,4)(3,6,8)$ a rotation of 120^0 about the diagonal to see that the G_1 orbit contains 3 elements. Finally, the reflection $(45)(3,6)$ is the only non-trivial symmetry fixing 1 and 2 and it has an orbit of size 2. Clearly, if an isometry fixes $\{1,2,3\}$ point-wise it is the identity. It follows that $|G| = 8 \cdot 3 \cdot 2 = 48$.

Membership

Clearly, to test if some permutation $p \in \mathsf{S}_n$ belongs to the permutationgroup $G \subset \mathsf{S}_n$ it is sufficient to compare the orders of G and $\langle G, g\rangle$. This can be used

- To test if a group $H \subset \mathsf{S}_n$ is a subgroup of G: test membership of G for every $y \in H$;
- To test if H is a normal subgroup of G: test in addition membership of H of every conjugate $g^{-1}yg$, $g \in G$, $y \in H$.

Making things more efficient using a base

If G acts on a set S we say that an ordered set $B = \{s_1, \ldots, s_k\}$ is *base* for the action, if the stabiliser

$$G_B := \{g \in G \mid g(s_j) = (s_j), \ldots, j = 1, \ldots, k\}$$

is the identity. Such a base can be used to calculate the order of G by observing that by Prop 1.1.3 we have

$$|G| = |s_1^G| \cdot |G_{s_1}| = |s_1^G| \cdot |s_2^{G_{s_1,s_2}}| \cdot |G_{s_1.s_2,s_3}| = \cdots = |s_1^G| \cdot |s_2^{G_{s_1,s_2}}| \cdots |s_k^{G_{s_1,\cdots,s_{k-1}}}|.$$

Suppose $S = [1, \ldots, n]$. Then a base is produced at the same time as one calculates the orbits and the stabilizers: start with the algorithm for 1^G, the orbit of 1. Then compute G_1. Next pick an element not in the orbit of 1, say 2 and calculate 2^{G_1} and continue in this fashion. The algorithm stops when we have found a base $[1, 2, \ldots]$ for the action and it also finds $|G|$.

Exercise:

Write pseudo-code for the above algorithms for the order, membership, subgroup, base and normal subgro

Chapter 2

Euclid's algorithm for polynomials in one variable

2.1 Division with remainder

Fix a field k. A non-zero polynomial of degree d with coefficients in k can be written

$$f = a_0x^d + a_1x^{d-1} + \cdots + a_d, \quad a_j \in k, a_0 \neq 0.$$

We write $\mathsf{LT}(f) := a_0X^d$, the *leading term* of f. Note that

$$\deg(f) \leq \deg(g) \iff \mathsf{LT}(f) \text{ divides } \mathsf{LT}(g). \tag{2.1}$$

You learned in school how to perform division with rest:

Lemma 2.1.1. *Let g be a non-zero polynomial. Every polynomial f can be written*

$$f = qg + r, \quad r = 0 \text{ or } \deg(r) < \deg(f).$$

We say q is the quotient *of f by g, written* quotient*(f,g) and r the* remainder*, written* remainder*(f,g).*

There is an algorithm for this

```
Input: g,f
Output: q, r
q:=0, r:= f
WHILE r NOT zero AND LT(g) divides LT(r)  AND DO
 q:= q+ [LT(r)/LT(g)]
 r:= r- (LT(r))/LT(g))g.
```

One needs to see why it terminates. Look at the last step. Suppose that

$$r = a_0x^m + \cdots + a_m, \quad g = b_0x^k + \cdots + b_k.$$

Then the coefficient of x^m in $r - \dfrac{\mathsf{LT}(r)}{\mathsf{LT}(g)}g$ is equal to $a_0 - \dfrac{a_0}{b_0}b_0 = 0$ and so the degree of the new r has become strictly less.

This has some well known consequences:

Corollary 2.1.2. *1. A non-zero polynomial of degree d has at most d roots in the field k.*

2. $k[X]$ is a principal ideal domain, i.e. every ideal is generated by one element.

Proof: 1) This is by induction on d. It is obviously true for $d = 0$ and if $x = a$ is some root in k perform division with remainder with $(f, x - a)$ to see that $f = (x - a)q$ with $\deg q = d - 1$ and apply the induction hypothesis to q.
2) If $I = (0)$ we are done, if not, there is some not zero $f \in I$ of minimal degree. If $g \in I$ is any other element, perform division with remainder on (f, g) to see that the remainder $r \in I$. But $\deg r < d$ and so by minimality $r = 0$, i.e. $g \in (f)$. □

2.2 Euclid's algorithm

Question: given $f, g \in k[X]$, find the polynomial $h \in k[X]$ for which $(f, g) = (h)$. Such h is called *greatest common divisor* $\mathrm{GCD}(f, g)$ because

1. h divides f and g;
2. Any polynomial p dividing both f and g must divide h.

So see the first, write $f = ah$, $g = bh$. For the second assertion, note that the assumption means that $f, g \in (p)$ and so $(p) \in (f, g) = (h)$ i.e. p divides h. Note that the GCD, as a generator of a principal ideal in $k[X]$ is unique up to a multiplication by a non-zero constant.

Here is Euclid's algorithm to compute $\mathrm{GCD}(f, g)$

```
Input: f,g
Output: GCD,s
GCD := f
s := g
WHILE s NOT 0 DO
   rem:= remainder(GCD,s)
   GCD: = s
   s: = rem
```

Let us explain why this works. So we need to do two things: at the end GCD should be equal to $\mathrm{GCD}(f, g)$, and we also need to see that the algorithm stops.

Note that at the start the pair (GCD,s) equals (f, g) and $s = g$. Write $f = qg + r$. Then clearly $(f, g) = (f - qg, g) = (r, g) = (g, r)$. Now, after the first step (GCD,s) $= (g, r)$ and $s = r$ which has degree strictly less than g. So the *ideal* (GCD,s) equals (f, g) during the entire algorithm while the polynomial s keeps on decreasing its degree until $s = 0$. At that moment (GCD, s) =(GCD)=(f, g) as claimed.

2.3 Extension to several polynomials

Suppose now that we want to find a generator h for the ideal $(f_1, \ldots, f_s)$. Such h is called *greatest common divisor* $\mathrm{GCD}(f_1, \ldots, f_s)$ for similar reasons as the case $s = 2$. To find it use the obvious

Lemma 2.3.1.

$$\mathrm{GCD}(f_1, \ldots, f_s) = \mathrm{GCD}(f_1, \mathrm{GCD}(f_2, \ldots, f_s)).$$

By induction this yields indeed an algorithm for finding $\mathrm{GCD}(f_1, \ldots, f_s)$.

Exercises

1. Show that $k[X, Y]$ is not a principal ideal domain by showing that (x, y) cannot be generated by a single polynomial.
2. Give pseudo-code for the algorithm that calculates $\mathrm{GCD}(f_1, \cdots, f_s)$ where s is some fixed number ≥ 3.
3. Use REDUCE to calculate $\mathrm{GCD}(x^4 + x^2 + 1, x^4 - x^2 - 2x - 1, x^3 - 1)$ and also $\mathrm{GCD}(x^3 + 2x^2 - x - 2, x^3 - 2x^2 - x - 2, x^3 - x^2 - 4x + 4)$.
4. Decide wether $x^2 - 4$ belongs to the ideal $(x^3 + x^2 - 4x - 4, x^3 - x^2 - 4x + 4, x^3 - 2x^2 - x - 2)$.

Chapter 3

Division with polynomials in several variables

3.1 Introduction

We now fix a field k and consider the ring $k[x_1, \ldots, x_n]$ in n variables where n is some natural number. Recall that an *ideal* I in a ring R consists of a subgroup under addition which is closed under multiplication by *any* element of R. So this is stronger than being a *subring* where we only demand closure under self-multiplication. We have seen that in $k[X]$ every ideal I is principal: $I = (f)$ but this no longer holds for 2 or more variables. We address the following questions in general:

1. Ideal description: can every ideal be generated by finitely many elements, i.e. is every ideal ofthe form $I = (f_1, \ldots, f_r)$, with $f_j \in k[x_1, \ldots, x_n]$;
2. Ideal membership: given I and f, decide wether $f \in I$;
3. Solving polynomial equations: find the common solutions in k^n of a system of polynomial equations $f_1 = \cdots = f_r = 0$, $f_j \in k[x_1, \ldots, x_n]$.

Example 3.1.1. 1) For $n = 1$ the membership question is easily solved using the fact that $I = (g)$ for some polynomial g: just divide f by g; then $f \in I$ if and only if the remainder is 0.
2) The easiest case of finding solutions is the case when all f are linear. You have seen this in your first year: you apply Gauss' method of row-reducing the matrix of the coefficients of the linear forms. This is indeed an algorithm.

3.2 The Hilbert Basis Theorem

We address the first problem. We have

Theorem 3.2.1 (Hilbert Basis Theorem)**.** *Every ideal I of $k[x_1, \ldots, x_n]$ is finitely generated: there exist $f_1, \ldots, f_s$ such that $I = (f_1, \ldots, f_s)$.*

The proof will be by induction on the number of variables. It is useful to introduce:

Definition 3.2.2. A ring is called *noetherian* if every ideal in it is finitely generated.

We have a useful criterion:

Lemma 3.2.3. *A ring is noetherian if and only if it satisfies the* ascending chain condition*: every increasing chain of ideals $I_1 \subset I_2 \subset I_3 \cdots$ must stabilise, i.e. $I_m = I_{m+1} = \cdots$.*

Proof: Let R be a ring and let $I_1 \subset I_2 \subset I_3 \cdots$ be an ascending chain of ideals. The set $I := \bigcup_{i=1}^{\infty} I_i$ is an ideal of R. If there exists a strictly increasing chain $I_1 \subsetneq I_2 \subsetneq I_3 \subsetneq \cdots$, then I cannot be finitely generated since the generators $g_1, \ldots, g_s$ all would belong to a common I_k, but I contains $I_{k+1} \supsetneq I_k$ so that $I \supsetneq g_1, \cdots, g_s)$. Conversely, if all ideals are finitely generated, also I must be finitely generated and so its generators $g_1, \ldots, g_s$ all would belong to a common I_k and so the chain stops at I_k. □

Now we can give the *proof of the Hilbert basis theorem.* The induction starts with k which is clearly noetherian. So it suffices to prove:

Proposition 3.2.4. *If R is noetherian, then so is $R[x]$.*

Proof: Let $I \subset R[x]$ be a non-zero ideal and pick $f_1 \in I$, $f_1 \neq 0$ of minimal degree d_1. If $I = (f_1)$ we are done. If not we can choose $f_2 \in I - (f_1)$ of minimal degree d_2 and keep continuing in this fashion. We want to show that this process terminates when we have reached a system of generators for I. Let a_i be the leading coefficient of f_i. The chain of ideals

$$(a_1) \subset (a_1, a_2) \subset (a_1, a_2, a_3) \subset \cdots$$

in R must terminate, say at stage m. Then $a_{m+1} = b_1 a_1 + \cdots b_m a_m$ for some $b_j \in R$. We claim that $I = (f_1, \ldots, f_m)$. Suppose that this is not the case so that we can choose $f_{m+1} \in I - (f_1, \ldots, f_m)$ of minimal degree d_{m+1}. Consider the polynomial

$$g := f_{m+1} - \sum b_j f_j x^{d_{m+1} - d_j}.$$

By construction, the coefficient of $x^{d_{m+1}}$ equals $a_{m+1} - \sum b_j a_j = 0$, i.e. $\deg(g) < d_{m+1}$ while $g \in I - (f_1, \ldots, f_m)$, a contradiction.

□

Example 3.2.5. Since we can perform euclidean division in $\mathbb{Z}$ this ring is also a principal ideal domain and in particular Noetherian. It follows that the rings $\mathbb{Z}[x_1, \ldots, x_n]$ are also Noetherian.

3.3 Monomial orderings

For one variable the degree orders the polynomials uniquely: we say $f < g$ if $\deg(f) < \deg(g)$. Now for more variable this is more complicated. First agree to write the monomials like this:

$$x^{\mathbf{a}} = x_1^{a_1} \ldots x_n^{a_n}, \quad \mathbf{a} = (a_1, \ldots, a_n).$$

The exponents belong to the semi-group $\mathbb{Z}^n_{\geq 0}$ and we want to order this semi-group. First of all note there is the *total degree* which for $\mathbf{a} = (a_1, \ldots, a_n)$ is given by $|\mathbf{a}| := a_1 + \cdots + a_n$, but is not a good order. Indeed we need more, as specified in the following definition:

Definition 3.3.1. A *monomial ordering* on $k[x_1, \ldots, x_n]$ is a total semi-group well-ordering $>$ on $\mathbb{Z}^n_{\geq 0}$, i.e

total ordering: for any couple $x, y \in \mathbb{Z}^n_{\geq 0}$ either $x > y$, $y > x$ or $x = y$;

semi-group ordering: if $x > y$ and $z \in \mathbb{Z}^n_{\geq 0}$, then $x + z > y + z$;

well-ordering: every non-empty subset of $\mathbb{Z}^n_{\geq 0}$ has a smallest element under $>$.

The well-ordering property can be checked fairly easily:

Lemma 3.3.2. *On order $>$ is a well-ordering iff every strictly decreasing sequence $\mathbf{a}_1 > \mathbf{a}_2 > \cdots$ has a smallest element.*

This is most easily seen by showing the contrapositive form: $>$ is *not* a well-ordering iff there is an infinite strictly decreasing sequence.

The following orders will be used:

Definition 3.3.3.

1. The *lexicographic order* or lex-order. We say $\mathbf{a} >_{\text{lex}} \mathbf{b}$ if in the vector difference $\mathbf{a} - \mathbf{b} \in \mathbb{Z}^n$ the left-most non-zero entry is positive;
2. The *graded lexicographic order* or grlex-order. We say that $\mathbf{a} >_{\text{grlex}} \mathbf{b}$ if either $|\mathbf{a}| > |\mathbf{b}|$, or $|\mathbf{a}| = |\mathbf{b}|$ and then $\mathbf{a} >_{\text{lex}} \mathbf{b}$.
3. The *graded reverse order* or grevlex-order. We say $\mathbf{a} >_{\text{grevlex}} \mathbf{b}$ if either $|\mathbf{a}| > |\mathbf{b}|$ or $|\mathbf{a}| = |\mathbf{b}|$ and then in the vector difference $\mathbf{a} - \mathbf{b} \in \mathbb{Z}^n$ the right-most non-zero entry is negative.

Of course one still needs to prove that the above examples do give monomial orders. For instance, the lexicographic order clearly is a total order and it is easy to see that it is a semi-group ordering. To see that it is well-ordered we argue as follows. Suppose that the lex-order is not well-ordered and let $\mathbf{a}_1 > \mathbf{a}_2 > \cdots$ and infinite strictly decreasing sequence. Look at the first entries: they are decreasing non-negative integers and thus must stabilize, say from $\mathbf{a}_k$ on. Next consider the second entries in the sequence $\mathbf{a}_k, \mathbf{a}_{k+1}, \cdots$. Continuing in this way we finally reach an element $\mathbf{a}_r$ with the property that in $\mathbf{a}_r, \mathbf{a}_{r+1}, \cdots$ all the n entries remain the same. This is a contradiction.

Remark.

1. Often we write $x^{\mathbf{a}} > x^{\mathbf{b}}$ for a given monomial order $>$ instead of $\mathbf{a} > \mathbf{b}$.
2. In the above we use the variables $x_1, \ldots, x_n$. In particular, these are ordered: for any of the three examples the order is $x_1 > x_2 > \cdots > x_n$. If you use other variables, such as (x, y, z) they must be viewed as ordered variables where conventionally the alphabetical order $x > y > z$ is used.
3. The lexicographic order is modeled upon the order of the words in a dictionary (with variables the letters of the alphabet) and 'annual' comes before 'annoys'. Note however that in a true dictionary the words can have any length.

4. Both grlex and grevlex first order according to total degree but next the order is completely different. Grlex looks at the *leftmost* variable and prefers the *higher* power, while grevlex looks at the *rightmost* variable and prefers the *smaller* power. For instance $x^5y^2z >_{\text{grlex}} x^4y^2z^2$ since $x^5 > x^4$, but $x^5y^2z >_{\text{grevlex}} x^4y^2z^2$ since $z >_{\text{grevlex}} z^2$.

Once we have an order we can order the individual terms in a polynomial. If we write the polynomial terms of f in this order, the first term is the *leading term*, $\mathsf{LT}(f)$. If $\mathsf{LT}(f) = cx^{\mathbf{a}}$ we call c the *leading coefficient* and $x^{\mathbf{a}}$ the *leading monomial*. Moreover, $\mathbf{a} = \deg(f)$, the *multidegree* of f. We reserve a special name for polynomials whose leading coefficient is 1:

Definition 3.3.4. A *monic* polynomial is a polynomial whose leading coefficient is 1.

Note that this depends on the chosen monomial order, in contrast to the one-variable situation.

Example 3.3.5. Let $f = 5x^4 - 7x^2y^2 - 8y^3z^3 + z^5$. With $>$ the lex-order we have

$$f = 5x^4 - 7x^2y^2 - 8y^3z^3 + z^5,\ \mathsf{LT}(f) = 5x^4\,, \deg f = (4,0,0).$$

but with respect to grlex (and also grevlex) we have

$$f = -8y^3z^3 + z^5 + 5x^4 - 7x^2y^2,\ \mathsf{LT}(f) = -8y^3z^3\,, \deg f = (0,3,3).$$

3.4 A division algorithm

Instead of division of f by a single polynomial g we need to be able to perform division by several polynomials $f_1, \ldots, f_s$ at once, i.e. we want to write

$$f = a_1f_1 + \cdots + a_sf_s + r$$

with quotients $a_1, \ldots, a_s$ and remainder r in $k[x_1, \ldots, x_n]$. We want to mimic the division with remainder for one variable observing that the order enters in a hidden way since we multiply a given polynomial in order to increase its degree in a maximal way. Let us use any polynomial order to find leading terms for the polynomials.

1. Start by looking if $\mathsf{LT}(f_1)$ divides $\mathsf{LT}(f)$ and if this is the case use division with remainder to get rid of the leading term.

2. If this is not the case, continue with f_2 and so on;

3. If at some intermediate step the leading term of the newly created remainder is not divisible by any of $\mathsf{LT}(f_j)$ we remember it to use it for r and move it away.

4. At the end all the terms moved away should be summed up to get r which then is a sum of monomials each of which is not divisible by any of $\mathsf{LT}(f_j)$.

Example 3.4.1. $f = x^2y + xy^2 + y^2$, $f_1 = xy - 1$, $f_2 = y^2 - 1$. Choose the lex-order. So $\mathsf{LT}(f) = x^2y$, $\mathsf{LT}(f_1) = xy$, $\mathsf{LT}(f_2) = y^2$ and after the first step we have $f = x(xy-1) + xy^2 - x + y^2$ and then, replacing f by $xy^2 - x + y^2$ we do the second division by f_1 which leaves $x + y^2 + y$. Its leading term is x so this is put apart for r. Then go on with $y^2 + y$ which can be divided by $y^2 - 1$ leaving $y + 1$ which also has to be put into r. So

$$x^2y + xy^2 + y^2 = (x + y)(xy - 1) + (y^2 - 1) + x + y + 1.$$

This leads to the following *division algorithm.*

```
Input:  f[1],....,f[s],f
Output: a[1],....,a[s],r

a[1]:=0;......;a[s]:=0;r:=0
p:=f
WHILE p NOT=0 DO
       i:=1
       divisionoccurred:=false
            WHILE  i <= s AND divisionoccurred=false DO
                 IF  LT(f[i]) divides LT(p) THEN
                       a[i]:=a[i]+ LT(p)/LT(f[i])
                       p:=p-(LT(p)/LT(f[i])*f[i]
                       divisionoccurred:=true
                 ELSE
                       i:=i+1
     IF divisionoccurred=false THEN
                       r:=r+LT(p)
                       p:=p-LT(p)
```

Some remarks are in order. While the algorithm gives a remainder, the result depends on the ordering of the f_i. This is visible in the above example when we interchange f_1 and f_2. We get

$$x^2y + xy^2 + y^2 = (x + 1)(y^2 - 1) + x(xy - 1) + 2x + 1.$$

So, unfortunately, this algorithm is not applicable to the ideal generated by f_1 and f_2 as it was in the 1-variable situation. This also makes clear that it cannot be used to test whether a given f belongs to (f_1, f_2).

Example 3.4.2. $f = xy^2 - x$, $f_1 = xy + 1$, $f_2 = y^2 - 1$. Then

$$xy^2 - x = y \cdot (xy + x)) + (-x - y) = x(y^2 - 1)$$

where the first identity comes from division by f_1, f_2 and the second identity from division by f_2, f_1. This last division shows that $f \in (f_1, f_2)$.

Exercises

1. Rewrite each of the following polynomials as ordered by the lex order, the grlex order and the grevlex order. Give also the total degree, the multidegree.

(a) $f(x, y, z) = 2x + 3y + z + 2x^2 - z^2 - x^3$.

(b) $f(x, y, z) = x^2y^8 + 3x^5yz^4 + xyz^3 - 7xy^4$

2. Do the same when we order the variables $z > y > z$.

3. Let $>$ be some monomial order on $k[x_1, \ldots, x_n]$.

   (a) Let f be a polynomial in $R = k[x_1, \ldots, x_n]$ and m a monomial in R. Prove that the leading term of $f \cdot m$ is the product $\mathsf{LT}(f) \cdot m$.

   (b) If f, g are two polynomials in R, is it true that $\mathsf{LT}(f \cdot g) = \mathsf{LT}(f) \cdot \mathsf{LT}(g)$?

   (c) What can one say about $\mathsf{LT}(f + g)$?

   (d) If $f_i, g_i \in R$, $i = 1, \ldots, k$. What can one say about $\mathsf{LT}(\sum f_i g_i)$?

4. Compute (by hand) remainder on division of f by the ordered set F. Next, change the order in a cyclic way. Use the grlex order and then the lex-order in each case. After you did the calculations verify them using REDUCE.

   (a) $f = x^7y^2 + x^3y^2 - y + 1$, $F = (xy^2 - x, x - y^3)$;

   (b) $f = xy^2z^2 + xy - yz$, $F = (x - y^2, y - z^3, z^2 - 1)$.

Chapter 4

Gröbner bases and Buchberger's algorithm

4.1 Monomial ideals and Gröbner bases

By definition, a *monomial ideal* is generated by monomials. These behave better from a computational point of view, mainly because a monomial ideal can always be generated by finitely many monomials (this is not a priori clear: we only know that we can choose finitely many polynomial generators). One sees this as follows. First we prove:

Lemma 4.1.1. *Let I be a monomial ideal. Then*
a) A monomial $x^{\mathbf{a}}$ belongs to I if and only there is some generating monomial of I which divides $x^{\mathbf{a}}$.
b) $f \in I$ if and only if every term of f belongs to I.

Proof:
a) Write $x^{\mathbf{a}} = \sum p_j x^{\mathbf{a}_j}$ where the $x^{\mathbf{a}_j}$ are amongst the generators of I. Expand the p_j as a sum of monomial terms $p_j = \sum_i a_{ij} x^{\mathbf{b}_{ij}}$. So the terms for which $\mathbf{b_{ij}} + \mathbf{a_i} = \mathbf{a}$ do not all get cancelled and in particular at least one such term must be present and then $x^{\mathbf{a}_i}$ divides $x^{\mathbf{a}}$.
b) Write f as a sum of monomial terms, $f = f_1 + \cdots + f_r$. We can also write f as a polynomial combination of generating monomials, say $f = \sum p_j x^{\mathbf{a}_j}$. Expand p_j as a sum of monomial terms, say $p_j^{(1)} + p_j^{(2)} \cdots +$. So each term f_i must be equal to a sum of terms $p_j^{(i)} x^{\mathbf{a}_j}$ of the same degree as f_i. In particular we have $f_i \in I$. The converse is clear. □

Corollary 4.1.2. *A monomial ideal is generated by finitely many of its generators.*

Proof: The monomial ideal I generated by a priori infinitely many monomial generators x^α, $\alpha \in A$. Write down some finite polynomial basis (this exists because of Hilbert's basis theorem) and consider for each member of the basis the monomials in the monomial expansion. By the lemma, part (b) these belong all to I and by construction they generate I. Now apply part (a) to replace each of the finite monomials found so far by some monomial generator x^α. These still generate I. □

Every ideal I has associated to it a canonical monomial ideal:

Definition 4.1.3. The *leading term ideal* of I, $\mathsf{LT}(I)$ is the ideal generated by the leading terms in I, i.e.

$$\mathsf{LT}(I) := \{\mathsf{LT}(f) \mid f \in I\}.$$

Since this ideal is monomial, application of Cor. 4.1.2 shows that $\mathsf{LT}(I) = (\mathsf{LT}(g_1), \ldots, \mathsf{LT}(g_s))$ for a finite set $\{g_1, \ldots, g_s\}$ of elements in I. Remarkably, the g_j themselves form a basis of I as we'll see shortly. Not only that, but division with remainder works much better using these g_j. To see this, let f any polynomial and perform division with remainder:

$$f = a_1 g_1 + \cdots a_s g_s + r$$

where either $r = 0$ or no term in r is divisible by any of the leading terms $\mathsf{LT}(g_j)$. Observe that $f - r \in I$.

Suppose next that $f \in I$. Then $r \in I$ hence its leading term belongs to $\mathsf{LT}(I) = (\mathsf{LT}(g_1), \ldots, \mathsf{LT}(g_s))$ and so by Lemma 4.1.1 a), if $\mathsf{LT}(r) \neq 0$, it must be divisible by one of its generators which is impossible by assumption. So $r = 0$ and f is a polynomial combination of the g_i, i.e. the g_i must form a basis for I.

Summarizing, $\{g_1, \ldots, g_s\}$ a basis of I and the ideal membership test we discussed previously in § 3.4, works for this basis. This merits a definition:

Definition 4.1.4. A finite collection $\{g_1, \ldots, g_s\}$ of elements of I is a *Gröbner basis* of I if its leading terms generate $\mathsf{LT}(I)$.

So, using this terminology, the preceding discussion shows:

Proposition 4.1.5. *1. A Gröbner basis is a basis;*

2. *Every non-zero ideal has a Gröbner basis;*
3. *For a Gröbner basis the ideal membership test works. More precisely, division by $g_1, \ldots, g_s$ leaves a unique remainder r independent of the order of the g_j; it is characterized as the unique polynomial r such that*
   - *(a) $r = 0$ or no term is divisible by any of the $\mathsf{LT}(g_j)$;*
   - *(b) $f - r \in I$.*

In fact, we have not yet proven the uniqueness of r but this is easy: if r' has the same properties, $r - r' \in I$ and if it is not zero, none of its terms is divisible by the $\mathsf{LT}(g_j)$ and as before this leads to a contradiction.

4.2 S-polynomials

We want to give a criterion to recognize whether a given basis is a Gröbner basis. This makes use of the S-polynomial $S(f, g)$ associated to a given pair $f, g \in k[x_1, \ldots, x_n]$:

Definition 4.2.1. Let $\deg f = \mathbf{a} = (a_1, \ldots, a_n)$, $\deg g = \mathbf{b} = (b_1, \ldots, b_n)$ and put $\mathbf{c} = (c_1, \ldots, c_n)$ where $c_i = \max(a_i, b_i)$. The S-polynomial of f and g is then

$$S(f, g) := \frac{x^{\mathbf{c}}}{\mathsf{LT}(f)} f - \frac{x^{\mathbf{c}}}{\mathsf{LT}(g)} g.$$

The monomial $x^{\mathbf{c}}$ is called the *least common multiple* of $\mathsf{LT}(f)$ and $\mathsf{LT}(g)$.

Note that the coefficient of $x^{\mathbf{c}}$ in $S(f,g)$ vanishes, i.e.

$$\deg S(f,g) < \mathbf{c}. \tag{4.1}$$

And here is the test:

Theorem 4.2.2. *A basis $G = \{g_1, \ldots, g_s\}$ for an ideal is a Gröbner basis iff the for all pairs $i \neq j$ the remainder on divison of $S(g_i, g_j)$ by G (in some order) is zero.*

Start of the proof: It is clear that if G is a Gröbner basis, then, since $S(g_i, g_j) \in I$ division by G gives zero remainder.
The proof of the converse is more involved. Let $f \in I$ be a non-zero polynomial. We need to show that its leading term is in the ideal of the leading terms of the $g_i \in G$ if the property as stated holds. Write

$$f = \sum h_i g_i. \tag{4.2}$$

Note that

$$\deg(f) \leq \max(\deg(h_i g_i)) \tag{4.3}$$

and that if equality occurs we must have $\deg(f) = \deg(h_i g_i)$ for some i. In that case $\mathsf{LT}(g)$ is divisible by $\mathsf{LT}(g_i)$ and we are done. So we need to see what happens when we have strict inequality, i.e. when some cancellation occurs. It is there where we use the property involving the S-polynomials as follows:

Lemma 4.2.3. *Suppose g is a sum of monomial products of the same multidegree $\mathbf{d}$ but $\deg g < \mathbf{d}$:*

$$g = \sum_{i=1}^{t} c_i x^{\mathbf{a_i}} g_i, \quad c_i \in k,\ \mathbf{a_i} + \deg g_i = \mathbf{d}, \quad \deg g < \mathbf{d}.$$

Then for some constants $d_{jk} \in k$ we have

$$\begin{aligned} g &= \sum d_{jk} x^{\mathbf{d}-\mathbf{c_{jk}}} S(g_j, g_k), \\ &\qquad \mathbf{c_{jk}} = \textit{ the least common multiple of } \mathsf{LT}(c_j) \textit{ and } \mathsf{LT}(c_k), \end{aligned}$$

and each term in this expression has multidegree $< \mathbf{d}$.

Proof: Note that the last assertion follows immediately from (4.1). Let us prove the other assertions. Suppose that

$$\mathsf{LT}(g_i) = d_i x^{\mathbf{b_i}}.$$

Then $\sum c_i x^{\mathbf{a_i}} \mathsf{LT}(g_i)$ vanishes, the degree of this sum being $< \mathbf{d}$ and so we have

$$\sum c_i d_i = 0. \tag{4.4}$$

On the other hand,

$$x^{\mathbf{d}-\mathbf{c_{jk}}} S(g_j, g_k) = \underbrace{x^{\mathbf{a_j}} \frac{g_j}{d_i}}_{p_j} - \underbrace{x^{\mathbf{a_k}} \frac{g_k}{d_k}}_{p_k}.$$

Now we re-arrange terms to make appear the differences $p_j - p_k$

$$\sum c_i x^{\mathbf{a_i}} g_i = \sum c_i d_i p_i = c_1 d_1 (p_1 - p_2) + (c_1 d_1 + c_2 d_2)(p_2 - p_3) + \cdots + (\sum_{j=1}^{t} c_j d_j) p_t.$$

Because of (4.4) the last term vanishes and so the above expression has the desired form. □

Now we can *complete the proof of Theorem 4.2.2*:
We consider all possible ways to write f as in (4.2). The multi-degree $\mathbf{d}$ of the right hand side might each time be different. Now we use the fact that we have a well-ordering: it is possible to choose an expression for which $\mathbf{d}$ is *minimal*. We shall use the property of the S-polynomials to show that then equality must hold in (4.1) which, as we saw at the start of the proof, implies the desired result. We argue by contradiction. So assume that $\deg(f) < \mathbf{d}$ and rewrite (4.2) so as to isolate the terms of multidegree $\mathbf{d}$. With $\deg(h_i g_i) = \mathbf{d_i}$ we have:

$$f = \sum_{\mathbf{d_i}=\mathbf{d}} \mathsf{LT}(h_i) g_i + \sum_{\mathbf{d_i}=\mathbf{d}} (h_i - \mathsf{LT}(h_i)) g_i + \sum_{\mathbf{d_i}<\mathbf{d}} h_i g_i. \tag{4.5}$$

Since the last two sums have multi-degree $< \mathbf{d}$ so must the first sum. This means that we can apply Lemma 4.2.3 and write it as

$$\sum_{\mathbf{d_i}=\mathbf{d}} \mathsf{LT}(h_i) g_i = \sum d_{jk} x^{\mathbf{d}-\mathbf{c_{jk}}} S(g_j, g_k). \tag{4.6}$$

Now use the hypothesis. By the division algorithm we have

$$S(g_j, g_k) = \sum_i a_{ijk} g_i, \quad \deg(a_{ijk} g_i) = \deg(S(g_j, g_k)) \tag{4.7}$$

Consider now the terms in the right hand side of (4.6). Each is a product of d_{jk} and the polynomial

$$x^{\mathbf{d}-\mathbf{c_{jk}}} S(g_j, g_k) = \sum_i \underbrace{a_{ijk} x^{\mathbf{d}-\mathbf{c_{jk}}}}_{b_{ijk}} g_i.$$

By Lemma 4.2.3 and (4.7) we have

$$\deg(b_{ijk} g_i) < \mathbf{d}$$

and so (4.6) can be rewritten

$$\sum_{\mathbf{d_i}=\mathbf{d}} \mathsf{LT}(h_i) g_i = \sum_i \left(\sum_{jk} d_{jk} b_{ijk} \right) g_i,$$

and each term has multi-degree $< \mathbf{d}$ since the d_{jk} are constants. But this gives a rewrite of first term in (4.5) as terms in the ideal $(g_1, \ldots, g_s)$ each of which has multi-degree $< \mathbf{d}$ and since the other terms also has this property we have reached a contradiction. □

4.3 Buchbergers's algorithm

This is the algorithm

```
Input F=(f1,...,fs) a list of poynomilas generating the ideal I
Output G=(g1,...,gt) a Groebner basis for I with G containing F

G:=F
REPEAT
        G':=G
        FOR each pair (p,q) in G' with p not=q DO
                      S:= remainder of S(p,q) on division by G'
                      IF S not=0 THEN G:=  (G,S)
UNTIL G=G'
```

It does the following. It starts with the set F and takes it to be G at the first step. Then it does the remainder test from Theorem 4.2.2 and stops if it is fulfilled. If not it adds the non-zero remainders of the S-polynomials to the "old" G. These remainders by construction all belong to I. The new collection G thus is still a basis of I. Again, it does the remainder test and so on. Why does this terminates? In the loop G' is the old G and the new G consists of G together with the remainders and $(\mathsf{LT}(G')) \subset (\mathsf{LT}(G))$ where the inequality is strict if $G' \neq G$. Indeed, let r be a non-zero remainder. Its leading term is not divisible by any of the $\mathsf{LT}(G')$ but $\mathsf{LT}(r) \in (\mathsf{LT}(G))$. Since an ascending chain of ideals must stabilize this shows that the algorithm terminates.

4.4 Reduced Bases

Unfortunately a Gröbner basis is not unique, but it becomes unique if we make some extra requirements:
Step 1. We can first of all assume that all of the g_i are monic (Def. 3.3.4). Then, if for some g_j we have that $\mathsf{LT}(g_j)$ is contained in the ideal J generated by $\mathsf{LT}(g_i)$, $i \neq j$, then we can leave out this g_j. Indeed, then the ideal J equals the ideal generated by all the $\mathsf{LT}(g_i)$, $i = 1, \ldots, s$ and so equals $\mathsf{LT}(I)$ which means that taking away g_j still gives a Gröbner basis. We end up with a *minimal basis* G, i.e. the $g \in G$ are monic and for all $p \in G$ its leading term is not in the ideal generated by $\mathsf{LT}(g)$, $g \in G - \{p\}$.
Step 2. Let $G = \{g, g_1, \ldots, g_r\}$ be any minimal Gröbner basis. We say that $g \in G$ is G-reduced if no term of g is in the ideal $(\mathsf{LT}(g_1), \ldots, \mathsf{LT}(g_r))$. We shall show how to replace g by g' so that the new $G' = (G - \{g\}) \cup \{g'\}$ is a minimal basis for which g' is G'-reduced. The g' we are after is found upon division with remainder:

$$g = \sum a_i g_i + g',\ g' \neq 0 \text{ and no term of } g' \text{ is divisible by the } \mathsf{LT}(g_i). \quad (4.8)$$

If you perform the division, you first test whether $\mathsf{LT}(f)$ is divisible by any of the $\mathsf{LT}(g_i)$ and since this is not the case by minimality of G, the algorithm puts $\mathsf{LT}(g)$ in the remainder. This is the largest term of the remainder g'. So $\mathsf{LT}(g) = \mathsf{LT}(g')$ and we have

$$\mathsf{LT}(G) = \mathsf{LT}(G'), \quad (4.9)$$

where the notation means the *set* consiting of the leading terms $\mathsf{LT}(g)$, $g \in G$ respectively $g \in G'$. This implies first that G' is a Gröbner basis and second that G' is minimal. Now the full non-divisibilty (4.8) of all terms of g' precisely says that g' is G'-reduced. Now repeat this procedure with G' and g'. We get a new minimal Gröbner basis $G'' = (G' - \{g'\}) \cup \{g''\}$ with g'' G''-reduced and where because of (4.9) the set $\mathsf{LT}(G'')$ has not changed. From the very definition of being G-reduced it follows that any element which is G'-reduced automatically is G''-reduced. So we end up with a minimal Gröbner basis in which *every* element is reduced with respect to this basis. Such a basis is called a *reduced Gröbner basis*.

Step 3. We show now that reduced Gröbner bases are unique. It is not hard to see that any two minimal Gröbner bases have the same set of leading terms. So it suffices to see that if $G, \tilde{G}$ are two reduced Göbner bases with $\mathsf{LT}(g) = \mathsf{LT}(\tilde{g})$ for some $g \in G$ and $\tilde{g} \in \tilde{G}$, then $g = \tilde{g}$, Consider $g - \tilde{g}$ which is of course in I and so, upon division by G gives zero remainder. On the other hand, $\mathsf{LT}(g) = \mathsf{LT}(\tilde{g})$ so in $g - \tilde{g}$ these terms cancel and since G as well $\tilde{G}$ are reduced, the remaining terms cannot be divisible by $\mathsf{LT}(G) = \mathsf{LT}(\tilde{G})$. So the remainder of $g - \tilde{g}$ upon division by G must be equal to $g - \tilde{g}$ and so, by the previous remark, equals zero.

Chapter 5

Applications of Gröbner bases

5.1 Ideal membership and equality of ideals

Once we have a Gröbner basis for I the membership test is easy: a polynomial f belongs to the ideal $I = (g_1, \ldots, g_t)$, where the $G := \{g_1, \ldots, g_t\}$ is a Gröbner basis exactly when the remainder of f upon division by G equals zero.

To test whether two ideals are equal, we have to compute two *reduced* Gröbner bases for them since these are unique. So $I = J$ if and only if the reduced Gröbner basis for I is the same as the reduced Gröbner basis for J.

5.2 Elimination: the case of linear equations

A system of linear equations can be solved using Gaussian elimination: first bring the system in upper-echelon form and then, starting from the last equation, eliminate successively the other variables. The equations correspond to generators of an ideal and one can show that Gaussian elimination corresponds to finding a Gröbner basis for this ideal. The row-reduced echelon from gives a reduced Gröbner basis.

Example 5.2.1.

$$\begin{array}{rcrcrcrcl} 3x & - & 6y & - & 2z & & & = & 0, \\ 2x & - & 4y & & & + & 4w & = & 0, \\ x & - & 2y & - & z & - & w & = & 0. \end{array}$$

The corresponding ideal is

$$I = (3x - 6y - 2z, 2x - 4y + 4w, x - 2y - z - w) \in k[x, y, z, w].$$

The echelon form and row reduced echelon forms are

$$\begin{pmatrix} 1 & -2 & -1 & -1 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & -2 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

One should think of the row reduced echelon form as giving a minimal set of equations without superfluous parameters: the variables x and z corresponding to the pivots can be expressed in the remaining *free* parameters y, w and one needs precisely 2 equations to describe the solution set which only depends on the ideal I and is customarily denoted $V(I)$. Since one needs 2 free parameters the dimension of this set is 2. Note also that these free parameters can not be eliminated further. Finally observe that the projection onto the (z, w)-plane gives a line whose equation is precisely $z + 3w = 0$, i.e. $I \cap k[z, w] = (z + 3w)$.

The corresponding bases for I are $G = \{g_1 = x - 2y - z - w, g_2 = z + 3w\}$ and $G' = \{g_1' = x - 2y + 2w, g_2' = z + 3w\}$. To see that the first is a Gröbner basis we compute the S-polynomial $S(g_1, g_2) = -(3xw + 2yz + z^2 + zw)$ and the remainder of $S(g_1, g_2)$ upon division by G. We find 0 since

$$\begin{aligned} -(3xw + 2yz + z^2 + zw) &= -3w(x - 2y - z - w) + (2y + z + w)(z + 3w) \\ &= -3wg_1 + (2y + z + w)g_2. \end{aligned}$$

It is easy to see that G and G' are minimal. That G' is reduced follows since the leading term ideal of g_2 is generated by z which is not in $\mathsf{LT}(g_1) = (x)$.

This works in general. To explain this, note that we may assume

$$g_1 = x_1 + \sum_{j \geq 2} a_j x_j, \quad g_2 = x_r + \sum_{j \geq r+1} b_j x_j,$$

and then

$$\begin{aligned} S(g_1, g_2) &= -\textstyle\sum_{j \geq r+1} b_j x_1 x_j + \sum_{j \geq 2} a_j x_j x_r \\ &= -(\textstyle\sum_{j \geq r+1} b_j x_j)(x_1 + \sum_{k \geq 2} a_k x_k) + (\sum_{k \geq 2} a_k x_k)(x_r + \sum_{j \geq r+1} b_j x_j) \\ &= -(\textstyle\sum_{j \geq r+1} b_j x_j)g_1 + (\sum_{k \geq 2} a_k x_k)g_2. \end{aligned}$$

For a row reduced matrix, the pivots 1 give monic leading monomials. The basis then is automatically minimal. For a row reduced matrix the matrix elements in the same column as the pivot 1 on the diagonal are all zero. Suppose that g_j is the polynomial corresponding to the j-th row. Its terms correspond to the non-zero elements in this row. The leading term ideal of the g_k, $k \neq j$ correspond to the rows containing the pivots. These rows have zeros in the j-th row and so no term of g_j is in the leading term ideal of the g_k, $k \neq j$, i.e. the basis is reduced.

5.3 Elimination: the general case

We start with an example. Suppose that we have a parametrized curve in the plane given by

$$x = t^2, \quad y = t^3.$$

Of course, it is easy to eliminate t from this equation: we find $x^3 = y^2$. But we can show that it automatically comes out if we compute a Gröbner basis for the ideal

$$I := (f_1 = t^2 - x, f_2 := t^3 - y)$$

provided we use the lex-order for which $t > x > y$. Check that the Buchberger algorithm yields the basis

$$G := \{f_1, f_2, f_3 := tx - y, f_4 := ty - x^2, f_5 := x^3 - y^2\}.$$

This basis G is not minimal since $\mathsf{LT}(f_2) = t^3$ is divisible by $\mathsf{LT}(f_1) = t^2$ and we may leave out f_2. It can be seen that $G' := \{f_1, f_3, f_4, f_5\}$ is minimal but not reduced. Note that the last basis element f_5 gives the desired equation for the curve. This curve lives in the (x, y)-plane and is just $V(I \cap k[x, y]) = (x^3 - y^2)$: the curve has equation $x^3 - y^2 = 0$. The set $V(I)$ is a space curve in (t, x, y)-space and elimination corresponds geometrically to projection onto the (x, y)-plane.

The general result is:

Theorem 5.3.1 (Elimination)**.** *Let G be a Gröbner basis for the ideal I in $k[x_1, \ldots, x_n]$ with respect to the lex order (with $x_1 > x_2 > \cdots > x_n$). Then*

$$G_k = G \cap k[x_{k+1}, \ldots, x_n]$$

is a Gröbner basis for the ideal $I_k := I \cap k[x_{k+1}, \ldots, x_n]$.

Proof: Let $G = \{g_1, \ldots, g_m\}$. Fix some k between 0 and n and, relabelling if necessary, assume that $G_k = \{g_1, \ldots, g_r\}$, i.e the first r basis-elements belong to $k[x_{k+1}, \ldots, x_n]$. We shall show that G_k is a basis of I_k. Clearly, $(g_1, \ldots, g_r) \subset I_k$ and so we need only show that every $f \in I_k$ can be written as $f = h_1 g_1 + \cdots + h_r g_r$ where the h_k only involve the last $n - k$ variables. We first note that since G is a Gröbner basis for I, division of $f \in I_k \subset I$ by G gives zero remainder so that we do get an expression $f = h_1 g_1 + \cdots + h_r g_r + \cdots + h_m g_m$.

Since we are using the lex-order with $x_1 > x_2 > \cdots > x_n$, the leading terms $\mathsf{LT}(g_{r+1}), \ldots, \mathsf{LT}(g_m)$ must all involve at least one of the first k variables and so are all greater than every monomial in $f \in k[x_{k+1}, \ldots, x_n]$. Thus if we apply division of f by the g_j, the $g_{r+1}, \ldots, g_m$ are not present. This also implies that the division entirely takes place within the ring $k[x_{k+1}, \ldots, x_n]$ and so the h_i are also in this ring.

It remains to see that G_k is a Gröbner basis. By Theorem 4.2.2, it suffices to show that the remainder of $S(g_i, g_j)$ on division by G_k is zero. But this calculation also takes place within the ring $k[x_{k+1}, \ldots, x_n]$ and since the S-polynomials belong to I_k these remainders are zero. □

Now one can ask if the solution set $V(I_k)$ in $(x_{k+1}, \ldots, x_n)$-space is exactly the projection of $V(I)$ onto this subspace. This is a surprisingly subtle question and the answer is *no* in general. What is true however is that $V(I_k)$ contains this projection, but there may be extra points.

Example 5.3.2. Consider $I = (xy - 1, xz - 1)$ in $k[x, y, z]$. Then a Gröbner basis is $G = \{xz - 1, y - z\}$ so that $I_1 = (y - z)$ and $V(I_1)$ is the line $y = z$ in the (y, z)-plane. However the projection of $V(I)$ onto the (y, z)-plane is contained in this line but misses the origin since $xy = 1, xz = 1$ cannot have $(*, 0, 0)$ in its solution set! The points in the projection are all of the form (a, a) with $a \neq 0$.

Note that in the above example the missing point itself can be described by an ideal. This can also be generalized but only for special fields: those that are algebraically closed.

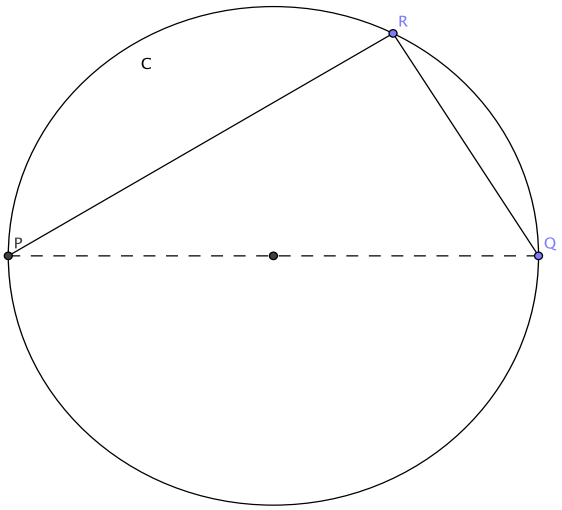
Example 5.3.3. Consider $I = (x^2 - y, x^2 - z)$ in $\mathbb{R}[x, y, z]$. Elimination of x gives the equation $y = z$ but the projection consists of the points (a, a^2, a^2), $a \in \mathbb{R}$ which forces $y, z \geq 0$ so that the missing points consist of a half-line which cannot be described as the set where some polynomial vanishes.

We are not going to discuss this further but refer to [C-L-O'S, Chapter4].

5.4 Geometry theorem proving

We give two examplesonly.

Example 5.4.1. Let C be a circle, $P, Q \in C$ two diametrically opposite points. Claim: for any $R \in C$ the segments RP and RQ are orthogonal. To show



this, one first has to choose coordinates (x, y). For instance such that the circle becomes $x^2 + y^2 - r^2 = 0$, $P = (-r, 0)$ and $Q = (r, 0)$. One has a point $R = (u, v)$ on the circle and $\overrightarrow{RP} = (u+r, v)$, $\overrightarrow{RQ} = (u-r, v)$ with inner product $(u + r) \cdot (u - r) + v^2$ which leads to:

```
Hypothesis: u^2+v^2-r^2   =0
Thesis:     (u+r).(u-r) +v.v =0
```

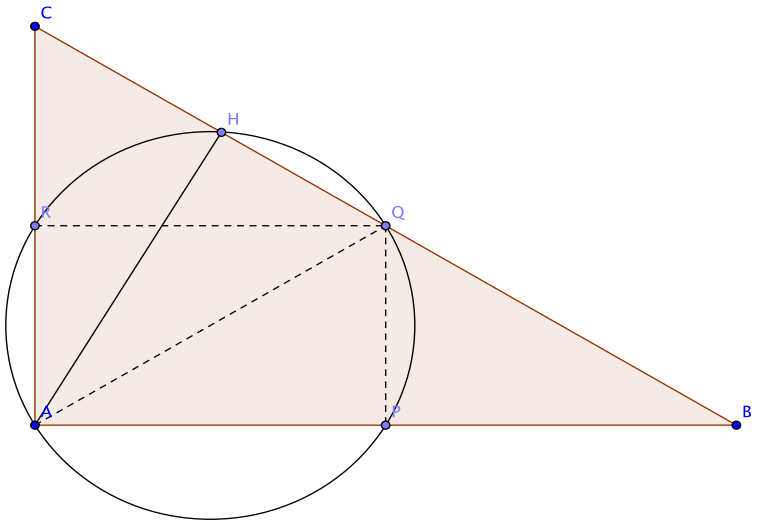
Clearly, the thesis follows directly from the hypothesis. How does one see this in terms of ideals? The hypothesis concerns the ideal $(u^2 + v^2 - r^2)$ in $\mathbb{R}[u, v, r]$ and the thesis states that $T := (u + r).(u - r) + v.v$ vanishes on $V(I)$. This follows if $T \in I$ which in this case is trivial, but in general such an assertion can be tested using the ideal membership test.

This can be generalized if the thesis can be seen as the condition that $R \in V(I)$ for some ideal I and if the hypothesis can be rephrased as a polynomial condition T stating that $T(p) = 0$ if $p \in V(I)$. Unfortunately, the resulting condition that $T \in I$ is in general only sufficient, but not necessary. For instance, one might have that $T \notin I$, but $T^2 \in I$ and then also T vanishes on $V(I)$.

There is yet another subtlety which has to do with the fact that one needs to ensure that a statement is not for some logical reason always trivially true.

Example 5.4.2. Consider a right triangle ABC with right angle at A. Let AH be the altitude line (with foot H). Claim: the midpoints of the sides and H lie on a circle (*circle theorem of Appolonius*).

The classical proof goes as follows: there is a circle passing through the vertices of the rectangle $APQR$. Then apply the previous example to A, Q, H.



To prove this analytically, choose coordinates so that $A = (0, 0)$, $B = (2x, 0)$, $C = (0, 2y)$. Then $P = (x, 0)$, $Q = (x, y)$ and $R = (0, y)$. The circle through P, Q, R has the equation $(X - \frac{1}{2}x)^2 + (Y - \frac{1}{2}y)^2 - \frac{1}{4}(x^2 + y^2) = 0$ which simplifies to

$$X^2 - Xx + Y^2 - Yy = 0,$$

and $H = (p, q)$ must satisfy

1. $(p, q) \perp (-x, y)$ which leads to $yq - px = 0$;
2. (p, q) is on the line BQ with equation $yX + xY - 2xy = 0$ so that $yp + xq - 2xy = 0$.

So we have

```
Hypothesis: I=(y.q-p.x,y.p+x.q-2.x.y)
Thesis T:=p^2-x.p+q^2-y.q belongs to I
```

Compute a Gröbner basis for I with respect to the lex-order with $x > y > p > q$ gives

$$G = \{g_1 = -yp - xq + 2xy, g_2 = -yq + px, g_3 = -yq^2 - yp^2 + 2y^2q\}$$

Unfortunately, doing division on T with G does not give zero remainder. We get:

$$-qg_2 - pg_1 = T \cdot y.$$

So there is the extraneous factor y. What is going on here? Apparently, one has to assume here that $y \neq 0$ which is equivalent to the fact that we have

an honest triangle with $C \neq A$. This has of course been assumed tacitly and corresponds to introducing an extra variable, say v and an extra equation $yv = 1$ which ensures $y \neq 0$. If we redo the calculations, assuming the lex-order with $x > y > p > q > v$ we now get

$$\begin{aligned} J &:= (yq - px, yp + xq - 2xy, 1 - yv), \\ G' &:= \{g_1 = -yp - xq + 2xy, g_2' = 2px - p^2 - q^2, g_3' = vxq - 2x + p, \\ &\quad g_4' = -q^2 + 2yq - p^2, g_5' = -1 + yv, g_6' = -2q + vq^2 + vp^2\} \end{aligned} \tag{5.1}$$

and this time $T \in I'$ since division by G' gives zero remainder.

Chapter 6

A polynomial factorization algorithm

6.1 Berlekamp's algorithm

6.1.1 Preliminaries

The problem is to factor a polynomial f over a finite field $k = \mathbb{F}_q$, $q = p^s$ where p is a prime number. The Berlekamp algorithm is an efficient way to achieve this. It uses the structure of the following set of polynomials

$$B(f) := \{v \in \mathbb{F}_q[x] \mid v^q \equiv v \bmod f\}.$$

To explain this, first note that $x^q - x = \prod_{a\in\mathbb{F}_q}(x-a)$ so that $v^q - v = \prod_{a\in\mathbb{F}_q}(v-a)$ where the factors are relatively prime. The basic observation is this:

$$f = \mathrm{GCD}(f, v^q - v) = \mathrm{GCD}(f, \prod_{a\in\mathbb{F}_q} \mathrm{GCD}(f, v-a)) = \prod_{a\in\mathbb{F}_q} \mathrm{GCD}(f, v-a). \quad (6.1)$$

If we are lucky a suitable polynomial v from $B(f)$ gives the full factorization at once by testing the common factors of f and $v + a$, $a \in \mathbb{F}_q$.

Example 6.1.1. Let $f = x^4 - 1 \in \mathbb{F}_5[x]$ and $v = x$. Then $x \in B(f)$. For the five elements $0, \pm 1, \pm 2$ we have $\mathrm{GCD}(f, x \pm 1) = x \pm 1$, $\mathrm{GCD}(f, x \pm 2) = (x \pm 2)$ so we get the full factorization.

In general, one could systematically try various polynomials v which shows that we should study $B(f)$ systematically.

To do this, let me recall the Chinese remainder theorem for polynomials:

Theorem 6.1.2 (Chinese Remainder Theorem)**.** *Let k be any field and $f = f_1^{e_1} \cdots f_r^{e_r}$ a factorization into irreducible polynomials f_j, $j = 1, \ldots, r$. The map*

$$\begin{aligned} \Phi : k[x] &\to k[x]/(f_1^{e_1}) \times \cdots \times k[x]/(f_r^{e_r}) \\ f &\mapsto (f \bmod f_1^{e_1}, \ldots, f \bmod f_r^{e_r}) \end{aligned} \quad (6.2)$$

induces an isomorphism of rings

$$k[x]/(f) \xrightarrow{\sim} k[x]/(f_1^{e_1}) \times \cdots \times k[x]/(f_r^{e_r})$$

Proof: Clearly, the map Φ is a homomorphism with kernel (f) and so by the isomorphism theorem for ring homomorphisms we only need to se that Φ is onto. We give an explicit inverse. Set $g_i = f_i^{e_i}$ and write $f = g_i \cdot h_i$ so that g_i and h_i are co-prime. So, one may write

$$a_i g_i + b_i h_i = 1, \quad a_i, b_i \in k[x].$$

Define

$$\psi(v_1, \ldots, v_r) := \sum v_i(b_i h_i).$$

Then ψ induces a well-defined homomorphism $\Psi : k[x]/(f_1^{e_1}) \times \cdots \times k[x]/(f_r^{e_r}) \to k[x]/(f)$ which is an inverse to Φ since $b_i h_i \equiv 0 \bmod f_j^{e_j}$ for $j \neq i$ while $b_i h_i \equiv 1 \bmod f_i^{e_i}$. □

6.1.2 The algorithm

Clearly, in view of (6.1), the set $B(f)$ has to be found first. This is just a linear algebra problem: Suppose $\deg(f) = n$ and use the basis $\{1, x, \ldots, x^{n-1}\}$ for $\mathbb{F}_q[x]/(f)$. The matrix Q for the map $v \mapsto v^q$ can be found by computing $x^{(i-1)q}$ modulo f and writing this in the i-th column, $i = 1, \ldots, n$. Then $B(f)$ is the kernel of $Q - I$. Now we need some way of finding the factors, starting from the set of polynomials $g + a$ where $g \in B(f)$ and $a \in \mathbb{F}_q$. This will be done by the Berlekamp algorithm. It is based on the following remark:

Proposition 6.1.3. *Via the map Φ above* (6.2) *the set $B(f)$ is isomorphic to the subring $\mathbb{F}_q^r$ of $\mathbb{F}_q[x]/(f_1^{e_1}) \times \cdots \times \mathbb{F}_q[x]/(f_r^{e_r})$; in particular, it is an $\mathbb{F}_q$-vector space of dimension r.*

Proof: Recall (6.1)

$$f = \prod_{a \in \mathbb{F}_q} \mathrm{GCD}(f, v - a).$$

Now any factor f_j of f divides exactly one of the factors in the right hand side, say $v - a$. But then f_j divides also $(v - a)^q - (v - a) = v^q - v$. But, since $v \in B(f)$ and so $f | v^q - v$ one deduces that $f_j^{e_j}$ divides $v - a$. So the class of v modulo $f_j^{e_j}$ corresponds to the constant a. If we have constants $(a_1, \ldots, a_r)$ with $v \equiv a_i \bmod f_i^{e_i}$, then $v^q - v \equiv 0 \bmod f = \prod_i f_i^{e_i}$, i.e. $v \in B(f)$. □

We explain now the Berlekamp algorithm which uses the set $B(f)$ to deduce a factorization of f into powers of irreducibles, the so-called *primary factors*. By the previous Proposition 6.1.3 $B(f)$ has dimension r. Let $B := \{v_1 = 1, v_2, \ldots, v_r\}$ be a basis. For any polynomial g and $a \in \mathbb{F}_q$ we let $g(a, j) := \mathrm{GCD}(g, v_j - a)$, $B(g, j) := \{a \in \mathbb{F}_q \mid g(a, j) \neq 1\}$ those a which give rise to the non-trivial factors $D(g, j) := \{g(a, j) \mid a \in B(g, j)\}$.

If $r = 1$ we must have $B = \{f\}$. If $r > 1$ start with v_2. If $B(f, 2)$ is not empty, set $B' := B(f, 2)$. Otherwise, set $B' = \{f\}$. Continue this process with each of the polynomials in the new B' and each of the basis-elements $v_j \in B$, $j \geq 2$ replacing $h \in B'$ by the polynomials in the set $B(h, j)$ in case $B(h, j) \neq \varnothing$. Consider what happens with B'. By construction, at each moment B' consists

of polynomials each of which is a product of one or more primary factors of f and each of these factors only appears once in a polynomial of B'. If B' changes, one or more of these polynomials are further broken up in the same way. Clearly the process terminates when in this way we can no longer further break up the factors. We must see that we then have found *all* of them. Since at the end of the construction no new gcd's are found, for every factor h we found, and for every j there must be some constant $s_j \in \mathbb{F}_q$ such that $v_j \equiv s_j \bmod h$. Since B is a basis of $B(f)$, for every $v \in B(f)$ there is thus a constant $s_v \in \mathbb{F}_q$ with $v \equiv s_v \bmod h$. If h would contain two distinct primary factors, say $f_i^{e_i}$, $i = 1, 2$, we would have $s_v \equiv v \bmod f_i^{e_i}$ and then the homomorphism Φ in Prop. 6.1.3 would not map $B(f)$ surjectively onto $\mathbb{F}_q^r$. This contradiction shows that we really found *all* primary factors, once the algorithm terminates.

Example 6.1.4. 1. $f = x^3 + 3x^2 + 6x + 4$ in $\mathbb{F}_7[x]$. One finds that the remainders of x^7, x^{14} upon division by f are x, x^2 respectively. So $Q - I = 0$ and the kernel is everything, i.e. $B(f) = \mathbb{F}_7 \oplus \mathbb{F}_7 x \oplus \mathbb{F}_7 x^2$ and one finds $f = (x + 1)(x + 3)(x + 6)$. Now consider the same polynomial in $\mathbb{F}_{49}$. We get the same answer since in the subfield $\mathbb{F}_7$ the polynomial is already fully factored.

2. $f = x^{12} - 1$ in $\mathbb{F}_5[x]$. One finds

$$Q - I = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{pmatrix}$$

and $B(f)$ has $\{1, x^3, x^6, x^9, x + x^5, x^2 + x^{10}, x^4 + x^8, x^7 + x^{11}\}$ as a basis. The vector $v_2 = x^3$ gives the factors $B' = \{x^3 + 4, x^3 + 3, x^3 + 2, x^3 + 1\}$. This set stays the same when we apply the Berlekamp algorithm with x^6 and x^9 but with $x^2 + x^{10}$ each of the factors in B' breaks up in two factors:

$$\begin{array}{llll} x^3 + 4 & \to & \{x^2 + x + 1, x + 4\}, & x^3 + 3 \to \{x^2 + 3x + 4, x + 2\}, \\ x^3 + 2 & \to & \{x^2 + 2x + 4, x + 3\}, & x^3 + 1 \to \{x^2 + 4x + 1, x + 1\} \end{array}$$

Since this yields 8 factors and $\dim B(f) = 8$ we are done:

$$\begin{aligned} x^{12} - 1 &= (x + 1) \cdot (x + 2) \cdot (x + 3) \cdot (x + 4) \cdot \\ &\quad (x^2 + x + 1) \cdot (x^2 + 3x + 4) \cdot (x^2 + 2x + 4) \cdot (x^2 + 4x + 1) \end{aligned}$$

What is still left to do is to find the prime factors h from the primary ones h^e. This is easy if e is not divisible by the characteristic p since then $h = \frac{1}{e} h^e / (h^e)'$. If $e = fp^\ell$ with f coprime to p we rewrite $h^e(x) = k^f(x^{p^\ell}) = [k^f(x)]^{p^\ell}$ and then compute $k(x)/k'(x)$ to find h.

Example 6.1.5. $q = 2$ and $h^{12} = x^{24} + x^{20} + x^{12} + x^4 + 1$ so that $e = 12 = 3 \cdot 4$. Rewrite $h^{12} = (x^4)^6 + (x^4)^5 + (x^4)^3 + x^4 + 1 = (x^6 + x^5 + x^3 + x + 1)^4$ and $k = x^6 + x^5 + x^3 + x + 1$. Then $k/k' = h = x^2 + x + 1$. Indeed $h^{12} = x^{24} + x^{20} + x^{12} + x^4 + 1$.

6.2 Polynomials with integer coefficients

For any field k the polynomial ring $k[x]$ in variable is a unique factorization domain: every polynomial factors as $f = cp_1^{n_1} \cdots p_r^{n_r}$ where the p_j are irreducible and monic, $n_j \in \mathbb{Z}_{>0}$, $c \in k$ and the irreducible factors and there exponents are unique. In particular this holds for $k = \mathbb{Q}$. So, if we have $f \in \mathbb{Z}[x]$, there is a unique factorization with $\mathbb{Q}$-coefficients. However, one can choose the irreducible factors in $\mathbb{Z}[x]$ as well thanks to the following result:

Proposition 6.2.1. *If $f \in \mathbb{Z}[x]$ and $f = gh$ is a factorization in $\mathbb{Q}[x]$, then for some $a, b \in \mathbb{Q}$ we have $ag, bh \in \mathbb{Z}[x]$ and $f = (ag \cdot (bg)$.*

Proof: Let $c(f)$, the *content of* f be the GCD of the coefficients of f. Content 1 means that reducing modulo any prime the polynomial never becomes the zero polynomial. In particular, since $\mathbb{F}_p[x]$ is an integral domain, products of two polynomials with content 1 must have content 1.

Note that $f = c(f)\tilde{f}$ with $c(\tilde{f}) = 1$. So, if $f = gh$, we have $c(f)\tilde{f} = c(g)c(h)\tilde{h}$ where $\tilde{f}, \tilde{g}$ and $\tilde{h}$ all have content 1. Clearly, $\tilde{f}$ is some multiple of $\tilde{g}\tilde{h}$ and since $\tilde{g}\tilde{h}$ has content 1 we must have $\tilde{f} = \pm\tilde{g}\tilde{h}$ so that

$$c(gh) = c(g)c(h).$$

To prove the proposition one may assume that $c(f) = 1$. Next, multiply g and h by rational numbers a, b so that $ag, bh \in \mathbb{Z}[x]$ and both have content 1. So $abf = (ag)(bh)$ has content 1 so that $ab = \pm 1$. Replacing a by $-a$ if necessarily we thus have $abf = f = (ag)(fh)$ as stated. □

The link with the Berlekamp algorithm is via reducing modulo p: every integer polynomial f can be considered modulo a prime and a factorization for f induces one modulo p. So, since one has an algorithm for factoring polynomials in $\mathbb{F}_p$ one should use this to find a factorization of f. Now this can be done, but one should realize that a factorization of f yield much more than a factorization modulo each prime p. In fact, one gets factorizations modulo each prime power p^m. Beware: this is **not** the same as a factorization in the field $\mathbb{F}_{p^m}$!

If we fix p and let m vary, the factorizations should be compatible in some precise sense.[1] Start with a factorization $f = gh$ where g and h are co-prime; if $f \equiv hg \bmod p^m$ and $f \equiv \tilde{h}\tilde{g} \bmod p^{m+1}$ one should have $\tilde{g} \equiv g \bmod p^m$ and likewise for h and $\tilde{h}$. Conversely, given g, h modulo p^m one can constructs such lifts:

Lemma 6.2.2 (Hensel)**.** *Let p be prime and $f, g, h \in \mathbb{Z}[x]$ monic (and not constant) such that $f \equiv gh \bmod p^m$ where the factors g, h are co-prime modulo p. Then there exist $\tilde{g}, \tilde{h} \in Z[x]$ such that $\tilde{g} \equiv g \bmod p^m$, $\tilde{h} \equiv h \bmod p^m$ such that $f \equiv \tilde{g}\tilde{h} \bmod p^{m+1}$. These lifts $\tilde{g}, \tilde{h}$ are unique modulo p^{m+1}.*

[1] For those who know: assembling all lifts for $m = 1, 2, 3, \ldots$ one gets a factorization in $\mathbb{Q}_p[x]$ where $\mathbb{Q}_p$ is the field of the p-adic numbers.

Proof: Write

$$\tilde{g} = g + p^m u, \quad \tilde{h} = h + p^m v, \quad \deg(u) < \deg(g),\, \deg(v) < \deg(h),$$

and substitute in $f - \tilde{g}\tilde{h} = (f - gh) + p^m[(uh + vg) + p^m uv]$. We know that $f - gh = p^m r$ for some explicitly known r and we want that $r + (uh + vg)$ is divisible by p. This is possible as stated because g and h are co-prime in $\mathbb{F}_p[x]$. Unicity modulo p follows because of the degree restrictions. □

Example 6.2.3. Let us come back to Exemple 6.1.4 1), i.e. $f = x^3+3x^2+6x+4$ in $\mathbb{F}_7[x]$ with factorization $f = (x+1)(x+3)(x+6)$. Consider $f = (x+3)(x^2-1)$ and try to lift this particular factorization modulo 7^2. We find $r = x+1$ and we want $u = u_0$ to be a constant and $v = v_0 + v_1 x$ linear with coefficients $u_0, v_{0,1}$ to be determined so that

$$(x+1)+u(x^2-1)+(v_0+v_1x)(x+1) = (u+v_1)x^2+(v_1+v_0+1)x+(-u+1+v_0v_1)$$

is divisible by 7. This yields $u \equiv 2, v_0 \equiv -2, v_1 \equiv -2$ modulo 7 and the lifted factorization is

$$f \equiv [(x+1) + 7 \cdot 2] \cdot [x^2 - 1 + 7 \cdot (-2 - 2x)].$$

6.3 Factoring integer polynomials

6.3.1 Discriminant

We first make some general remarks about polynomials in one variable. Recall that the *discriminant* of a (real or complex) monic polynomial f is the product of the squares of all its *complex* roots. For non-monic polynomials one has to multiply with a certain power of the leading coefficient:

$$\mathrm{discr}(f) := a^{2n-2} \prod_{i<j} (\alpha_i - \alpha_j)^2, \quad f = a \prod_i (x - \alpha_i).$$

This discriminant vanishes if there is a repeated root.

There is a formula for the discriminant in terms of the coefficients of f. This works in *any* field k using a factorization of f in a suitable extension field. We refer to the algebra course where it is shown that such extensions always exist.

To give the formula for the discriminant, it we need to introduce the *resultant* of two polynomials, say

$$f(x) := a_n x^n + \cdots + a_0, \quad g(x) = b_m x^m + \cdots + b_0, \quad f, g \in k[x].$$

Then we define $R(f, g) \in k$ to be the following determinant of size $(n + m) \times (n + m)$:

$$R(f, g := \det \begin{pmatrix} a_n & a_{n-1} & \cdots & a_0 & 0 & \cdots & 0 \\ 0 & a_n & \cdots & a_1 & a_0 & 0 & 0 \\ \vdots & \ddots & \cdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & a_n & \cdots & & a_1 & a_0 \\ b_m & b_{m-1} & \cdots & b_0 & 0 & \cdots & 0 \\ 0 & b_m & \cdots & b_1 & b_0 & 0 & 0 \\ \vdots & \ddots & \cdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & b_m & \cdots & & b_1 & b_0 \end{pmatrix},$$

where the first m rows repeat the coefficients of f each time shifted one entry to the right and the last n rows do the same with the coefficients of g. The formula which relates the discriminant to the resultant will be given without proof:

$$\mathrm{discr}(f) = (-1)^{\frac{1}{2}n(n-1)}\frac{1}{a_n}R(f,f'). \tag{6.3}$$

However, the essential point is that $R(f,f') = 0$ as soon as f has a multiple zero which follows from the following Lemma whose (easy) proof we give. We can therefore take formula (6.3) as a definition of the discriminant. Indeed, this is how it will be used in what follows.

Lemma 6.3.1. *$R(f,g) = 0$ if f and g have a non-trival common factor.*

Proof: First, note that if f and g have h as a common factor, then $(f/h)g + (-g/h)g = 0$ furnishes an expression

$$Af + Bg = 0, \quad \deg(A) < \deg(g),\ \deg(A) < \deg(f).$$

The converse is also true: Suppose that it were false. Then $(f,g) = 1$ and hence, by the euclidean algorithm, $uf + vg = 1$ and hence $ufB + vgB = B = ufB - Afv = (uB - vA)f$ contradicting $\deg(B) < \deg(f)$.

Now the existence of A and B is a linear algebra problem in the finite dimensional vector space generated by $1, x, \ldots, x^{n+m-1}$ which when written out yields a system of equations whose coefficients are given by the above matrix. □

Example 6.3.2. Let $f = ax^2 + bx + c$. Then $f' = 2ax + b$ and

$$\mathrm{discr}(f) = -\frac{1}{a}\det\begin{pmatrix} a & b & c \\ 2a & b & 0 \\ 0 & 2a & b \end{pmatrix} = b^2 - 4ac.$$

6.3.2 An algorithm

We now come to the main topic: factorization of integer polynomials. The crucial remark here is that an integer polynomial f can be factored by a finite algorithm basically since the coefficients of any factor of f can be bounded in terms of those of f. The reason is that once we have such a bound, we can now pick a suitable prime power p^m which is so big that any potential factor has a unique representative modulo p^m with coefficients in the interval $[-c, c]$ with $c < \frac{1}{2}p^m$. This makes it possible to find these factors in a unique way if we know them already modulo p^m. We saw how to do this in previous sections.

Let us now explain the algorithm. So assume

for every coefficient c of every divisor of f we have $|c| \leq C = C(f) \in \mathbb{R}$.

We can make some preliminary simplifications:

1. We may suppose that f has content 1;
2. Reduce the computation to square free polynomials: if $f = h^2 g$ with g square-free, $\mathrm{GCD}(f, f') = h$ can be found from f and one can separately factor g and h. Note that h itself need not be square-free so one has to repeat the process of extracting a square-free factor from h;

3. We may ensure that f remains square-free upon reducing modulo p. For this it is sufficient that discriminant of f is not divisible by p;

4. Assure that p is chosen so as not to divide a so that the degree does not drop when we reduce modulo p^m.

We pick p satisfying the last two conditions and make m so big that $p^m > 2C \cdot a$ where $ax^n = \mathsf{LT}(f)$.

Next, write $f = a\hat{f}$ in $\mathbb{F}_p[x]$ which $\hat{f}$ monic and use Berlekamp's algorithm to find directly[2] a factorization modulo p for $\hat{f}$ and then Hensel's Lemma to successively lift it to $p^2, \ldots, p^m$. So we may assume that we have a factorization

$$f \equiv a g_1 \cdots g_r \bmod p^m$$

with pairwise distinct monic polynomials g_j. By construction only those partial products $h = a \prod_{j \in S} g_j$ with the property that *all* coefficients of h have absolute values $\leq C \cdot a$ need to be considered. Moreover one may assume that $\deg h \leq n/2$. Test whether such h divides $a \cdot f$ and assemble the divisors h obtained in this fashion.

In practice we do the calculation making use of the following sharp bound:

Theorem (Landau-Mignotte)**.** *If $g = g_0 + g_1 x + \cdots + g_m x^m$ is a degree m factor of the polynomial f, then*

$$|g_i| \leq \binom{m}{i} \|f\|$$

where $\|f\|$ is the ℓ_2-norm of f.

For an easy but clever proof of this theorem we refer to [C-C-S].

Example 6.3.3. $f = x^3 + 3x^2 + 6x + 4$ has $\|f\| = \sqrt{62}$ and since the true divisors have degree 1 or 2, the above bound is at most $C = 2\sqrt{62} = 15.748016$. One has $c(f) = 1$, and $f' = 3x^2 + 6x + 6$ is coprime to f. The discriminant is $108 = 2^2 \cdot 3^3$ and so we may choose $p = 7$ and $m = 2$. We have seen in example 6.1.4 1) that f already factors fully in $\mathbb{F}_7$ as

$$f \equiv (x-1)(x+1)(x+3) \bmod 7.$$

Only linear factors need to be considered and it turns out that only $(x+1)$ divides f over the integers so that

$$f = (x+1) \cdot (x^2 + 2x + 4).$$

Exercises

1. Using GAP verify the examples of § 1.

2. Using GAP verify that the discriminant of $f = x^3 + 3x^2 + 6x + 4$ equals -108 and that f factors as stated in $\mathbb{F}_7[x]$.

[2] by our assumptions $\hat{f}$remains square free in $\mathbb{F}_p[x]$ and so a primary factor really is an irreducible factor and not a power of it.

3. Find the factorization in $\mathbb{Z}[x]$ of the polynomial $g = x^3 - x^2 - x - 2$ using the Berlekamp algorithm as in the case of f. (Calculate the discriminant. Show that f is square free and that we can work modulo 5^2. Perform Berlekamps algorithm for $p = 5$ and test whether this already gives the full factorization. Next show that there are 3 factors in $\mathbb{F}_3[x]$ and determine these. Is f square free in $\mathbb{F}_3[x]$? Test whether the factors already give some linear factor of g. If not use Hensel's method to lift modulo 9.)

   Verify the steps with GAP.

Appendix A

Symbolic Computation: Lessons in GAP and REDUCE

A.1 GAP Lesson 1[1]

Make a special directory where you save your files. Change to that directory and start up GAP. Pause for discussion at the spaces between command groupings.
Arithmetic Operations: You learn: $+ - * /$*, terminating input with ;, last. Factors, mod, Int, Gcd, Lcm, Factorial.*

Try the following. Press the return key at the end of each line.

```
gap> LogTo("GAPlesson1");
This writes your work to the file GAPlesson1
gap> 5+7;
gap> 12/17;
gap> 2/3 + 3/4;
gap> 2^3*3^3;
gap> 3.14159;
What happens here?
gap> Factors(1111111);
Factorization of integers
gap> Factors(11111111111);
gap> Factors(2^32 + 1);
gap> Factors(216);
gap> last;
 What happens here?
gap> Collected(last);
There are also variables last2 and last3. Guess what their values are.

Exercise: Find the first number of the form 111..1 which is prime.
```

[1]Most GAP- lessons have been adapted slightly from the ones found on Peter Webb's homepage `http://www.math.umn.edu/ webb/`

```
Note that such a number must have a prime number of digits.
As well as Factors, another useful function is IsPrime.

gap> -5 mod 11;
gap> 6 mod -5;
gap> 6 mod 0;
gap> Int(295/7);
gap> Gcd(216,930);
gap> Lcm(216,930);
gap> Factorial(6);
```

Permutations
You learn: permutations are enclosed in (), operations have the same form as for integers. Conjugation is built in.

```
gap> (1,2,3)*(1,2);
gap> (1,2,3)^-1;
gap> (1,2,3)^(2,5);
This is conjugation - which one?
gap> 1^(1,2,3);
What does this mean?
```

Help
You learn: lines are not terminated with ; How to call up and browse sections of the manual.

```
gap> ?
gap> ?Help
gap> ?Factor
gap> ?A f s
gap> ?>
gap> ???
```

True and False; Assignment of Variables
You learn: The difference between = and :=, how to store things in memory.

```
gap> 216=2^3*3^3;
gap> 216=2^3;
gap> g=17;
gap> g;
gap> g:=17;
gap> g;
gap> g=17;
gap> g=21;
gap> g^2;
gap> 3<=2;
gap> 3>=2;
```

Exercise: What response do you expect from?

```
gap> true and false;
gap> true or false;
```

Permutation groups; groups from the library; operations on groups
You learn: how to input a group generated by permutations. The operations Size, Center, DerivedSubgroup, in, Elements, SymmetricGroup, AlternatingGroup, DihedralGroup.

```
gap> a:=Group((2,3,5)(6,7,8),(1,2,4,7)(3,6,8,5));
gap> Size(a);
gap> Center(a);
gap> Size(last);
gap> g:=SymmetricGroup(6);
gap> Size(g);
gap> h:=DerivedSubgroup(g);
gap> Size(h);
gap> (1,2) in h;
gap> (1,2,3) in h;
gap> d:=DihedralGroup(8);
gap> Size(d);
```

A.2 GAP Lesson 2

Change to the directory where you store your GAP files, and then start up GAP. In the session which follows, pause for discussion at the spaces between command groupings.

Lists
You learn: notation for lists, how to access terms in a list, how lists are stored and duplicated, Sort, Add, Append, Length, Position, Flat, vector operations, List, ranges.

```
gap> LogTo("GAPlesson2");
gap> Primes;
gap> Primes[5];
gap> 15 in Primes;
gap> Position(Primes,31);
gap> Length(Primes);

gap> things:=["apple",true,,17];
gap> things[1];
gap> things[3];
gap> things[4]:=9;
gap> things[6]:=[1,2,3];
gap> things;

gap> newthings[1]:=72;
gap> newthings;
gap> newthings:=[];
gap> newthings;
gap> newthings[1]:=72;
gap> newthings;

gap> newthings:=things;
```

```
gap> newthings[1]:=23;
gap> newthings;
gap> things;
gap> things[1]:=14;
gap> things;
gap> newthings;

gap> newthings:=ShallowCopy(things);
gap> newthings[1]:=23;
gap> newthings;
gap> things;
gap> Add(things,16);
gap> things;
gap> Add(things,newthings);
gap> things;
gap> Append(things,newthings);
gap> things;
gap> Flat(things);
gap> ?Lists
gap> ?Flat

gap> r:=[1,2,3];
gap> s:=[3,1,4];
gap> r+s;
gap> 2*r;
gap> Sort(s);
gap> s;

gap> Append(r,s);
gap> r;
gap> List(r,x->x^2);
gap> List([1..10],IsPrime);
gap> [1..10];
gap> List([1..10],x->x);

gap> g:=Group((1,2,3),(1,2));
gap> els:=Elements(g);
gap> List(els,x->Order(x));

gap> els[3]:=14;
gap> els:=ShallowCopy(els);
gap> els[3]:=14;
gap> els;
```

Programming
You learn: do loops, how to initialize a list.

```
gap> g:=Group((1,2,3),(1,2));
gap> els:=Elements(g);

Discuss what would happen if you were to do the following
```

```
(don't try it unless you want to):

gap> for i in els do
> Order(i);
> od;

Instead, do this:

gap> orders:=[];
gap> for i in els do
> Add(orders,Order(i));
> od;
gap> orders;

The following is perhaps a less elegant way to program a list of orders
of the elements of g because the enumeration is done
over [1..Length(els)] instead of els itself.
However, sometimes we have to do this kind of inelegant thing.

gap> orders:=[];
gap> for i in [1..Length(els)] do
> orders[i]:=Order(els[i]);
> od;
gap> orders;
[ 1, 2, 2, 3, 3, 2 ]

At this point we note that there are other forms of do loop, namely
while ... do ... od;
and
repeat... until...;

Note the following:

gap> for i in [1..10] do
> Print(i);
> od;

gap> for i in [1..10] do
> Print(i,"\n");
> od;
```

A.3 GAP Lesson 3

Change to the directory where you store your GAP files, and then start up GAP. In the session which follows, pause for discussion at the spaces between command groupings.

Reading a file of code into GAP and using it.

On http://www.math.umn.edu/ webb there is a file called Conway. Create a copy of this file in your GAP file directory. Its contents are:

```
Conway:=function(seq)
local i, r, newseq;
newseq:=[];
i:=1;
while i<=Length(seq) do
r:=0;
repeat r:=r+1;
until i+r>Length(seq) or not seq[i]=seq[i+r];
Append(newseq,[r,seq[i]]);
i:=i+r;
od;
return(newseq);
end;
```

To read and make a print out of this file do:

```
gap> Read("Conway");
gap> Print(Conway);

gap> Conway([2]);
gap> Conway([1,2]);
gap> Conway(last);
gap> Conway(last);
gap> Conway(last);
```

Some manipulation with permutation groups.
You learn: Orbits, Action, Tuples, UnorderedTuples, Arrangements, Combinations. In an earlier version of GAP4 and also in GAP3 the command Action was Operation. It may be that in the implementation you use you need to type Operation.

```
gap> At:=KnownAttributesOfObject;
There is also KnownPropertiesOfObject.

gap> c:=Group((1,2,3,4,5,6,7),(2,3)(4,7));
gap> At(c);
gap> Size(c);
gap> GeneratorsOfGroup(c);
gap> c.1;
gap> c.2;
gap> At(c);
gap> s:=SylowSubgroup(c,2);
gap> At(c);
gap> At(s);
gap> Orbits(s,[1..7]);
gap> news:=Action(s,last[3]);
gap> Size(news);

How would it be different if you had done instead
gap> news:=Action(s,last[2]);
gap> Size(news);
```

```
What do you think would happen if you were to do
gap> Orbits(s,[1..8]);
or
gap> Orbits(s,[2..7]);
or
gap> Orbits(s,[3..7]);
?

gap> Tuples([1..3],2);
gap> UnorderedTuples([1..3],2);
gap> Arrangements([1..3],2);
gap> Combinations([1..3],2);

Exercise: Find out what GAP command returns a list of all partitions
of [1,2,3].

gap> newers:=Action(news,Tuples([1..4],2),OnTuples);
gap> Orbits(newers);

Other possibilities in Action instead of OnTuples: OnPairs, OnTuples,
OnSets, OnRight, OnLeft. Investigate by doing ?Basic Actions.

gap> Action(c,RightCosets(c,s)
```

A.4 Reduce Lesson 1

Change to the directory where you store your REDUCE files, and then start up REDUCE.

Polynomial factoring

You learn: factorize polyomials of one variable, computing the GCD for 2 or more polynomials, membership test.

```
1: factorize(x^63-1);
2:  f:=x^63-1;
3:  g:=x^4-x^2-2x-1;
4:  h:=x^4+x^2+1;
5: gcd:=GCD(f,g);
6: gcd;
7: gcd:=GCD(gcd,h);

# With the switch FACTOR on,  the polynomials are automatically
# factored.

8: on factor; f;

# in order to calculate a gcd of more than 2 polynomials REDUCE
# has to work with expanded polynomials.  Try for instance to calculate
#  gcd(f,g,h) with f= x^3-1 and g=x^6-1 and h=x-1.
# What do you get?
```

```
# In order to remedy the situation on needs to put  the switch EXP
#  on or the switch FACTOR  off.

10: on EXP;
11: GCD(GCD(x^3-1,x^6-1),x-1);

# How to find remainders

10: remainder(gcd, x^2-3); remainder(f,2*x^2+2*x+2);

# what happens?
# WARNING: reduce works with integers, one has to TELL that it has to
# work with a field, for instance with the rationals:

12: on rational; remainder(x^63-1,2*x^2+2*x+2);

13: gcd:=GCD(x^4+x^2+1,x^4-x^2-2x-1); GCD(gcd,x^3-1);
15: gcd:=GCD(x^3+2x^2-x-2,x^3-2x^2-x+2); newgcd:=GCD(gcd,x^3-x^2-4x+4);
17: remainder(x^2-4,newgcd);


# How to calculate in fields of finite characteristic

18: setmod 7; on modular;
20:  f;

# put the switch off if you want to go on
21: off modular;
```

Filehandling
You learn: how to work with files. This is highly system dependent. Here I suppose that you are working in a window which has a menu with an option to save a log-file at any moment. For other systems file handing is more cumbersome.

The first step is to tell reduce to produce output that can be read normally:

```
1: off nat$
```

If you now save to a file for example *redlesson1* the file `redlesson1.log` is created. Now suppose you want to use the file later as a reduce file. Then some further care has to be taken:

```
1: off echo$
2: off nat$
```

and just before you save, type:

```
12: write ";end"$
```

If you then want to read in the file `redlesson1.log` you can either read it in from the menu, or using the command

```
1:  in redlesson1.log;
```

A.5 Reduce Lesson 2

Change to the directory where you store your REDUCE files, and then start up REDUCE.
You learn: how to define monomial orders, S-polynomials, remainders on division and to compute Groebner bases.

```
1:  load groebner;

# the monomial order is automatically set to LEX.
# In REDUCE it is called TORDER
# other orders known: GRADLEX=grlex and
# REVGRADLEX=grevlex to change it.

2: torder gradlex;

# Note: this prints the PREVIOUS monomial order!

# REDUCE needs to know how the variables are ordered.
# The orders lex and grlex use
# the alphabetical order and so a>b>c...>x>y>z
# if you need something else, e.g. x>y>z>t, say

3:  torder {{x,y,z,t}, lex};

# you can check this by the command gsort

4: gsort(t+x+y+z);
```

```
# to find leading term of a polynomial and the rest:

5: gsplit(t+x+y+z):

# one gives lists of polynomials as follows:

6: gb:={x^2+y,x+2x*y};

# this gives a list of polynomials. Note the syntax!
# To calculate the remainder upon division we use

7:  preduce(x^3+3y^2+x*y,gb);

# to find the quotients one can modify things as follows

8: gb:={g1=x^2+y,g2=x+2x*y}; preducet(q=x^3+3y^2+x*y,gb);

#  Note that this expresses the remainder  r=a1*g1+b1*g2+q
# in terms of the quotients

10:  f:=x^3+3y^2+x*y; g:=x*y+x+y;

# to calculate  the S-polynomial of g and f;

12: gspoly(f,g);
```

Problems.

1. Use the above to compute S-polynomials and remainders and then test if you get a Groebner bases;
   (a) $x^3 - y, x^3 - z$ grlex order.
   (b) $x^2 - y, x^3 - z$ grevlex order.
   (c) $xy^2 - xz + y, xy - z^2, x - yz^4$ lex order.
2. Produce a Groebner basis, then a minimal basis and next a reduced bases in the following cases:
   (a) $x^3 - y, x^3 - z$ grlex order.
   (b) $x^2 - y, x^3 - z$ grevlex order.
   (c) $y - x^2, z - x^3$ lexorder with $x > y > z$.

You can verify your results using the command

```
groebner(list of polynomials);
```

A.6 Reduce Lesson 3

Change to the directory where you store your REDUCE files, and then start up REDUCE.

1. Check example 5.2.1 for $I = (3x - 6y - 2z, 2x - 4y + 4w, x - 2y - z - w)$ with lex-order $x > y > z > w$ by testing that $\{x - 2y - z - w, z + 3w\}$ is a Gröbner basis and that $\{x - 2y + 2w, z + 3w\}$ is a reduced Gröbner basis.

2. Check the example of the beginning of 5.3 for $I := (t^2 - x, t^3 - y)$ with lex-order $t > x > y$ by applying the Buchberger algorithm.

3. Check example 5.3.3. Here $I = (xy - 1, xz - 1)$. Show that $\{xz - 1, y - z\}$ is a Gröbner basis.

4. Check example 5.3.4. Here $I = (x^2 - y, x^2 - z)$. Give a Gröbner basis.

5. Solve example 5.4.2 analytically. Here $I = (yq - pxy, p + xq - 2xy)$.

   (a) Show that $\{g_1 := -yp - xq + 2xy, g_2 := -yq + px, g_3 := -yq^2 - yp^2 + 2y^2q\}$ is a Gröbner basis for I and test whether $T = p^2 - x \cdot p + q^2 - y \cdot q$ belongs to I.

   (b) Now $J = (yq - px, yp + xq - 2xy, 1 - yv)$; verify that G' (see (5.1)) is a Gröbner basis and test whether T belongs to J.

The solutions count for your note. Please join a print-out of the REDUCE calculations.

A.7 GAP Lesson 4

Change to the directory where you store your GAP files, and then start up GAP.

Calculating with polynomials in GAP

You first have to tell GAP that you work with $\mathbb{Q}[x]$ before defining polynomials:

```
gap> x:=Indeterminate(Rationals);
gap> f:=x^4+3*x^2-17*x+1;g:=x^3-2*x+9;
```

Then GAP has to be reminded that it works with x as the first variable which it calls x_1.

```
gap> Resultant(f,g,1);
gap> Discriminant(g,1);
```

If one wants to work with finite fields one uses the notation GF(q). One can do division with remainder in these fields either by saying f mod g or *EuclideanRemainder*(f, g).

The elements of a finite field are denoted in a special way. If q is a prime the group of invertible elements of the field is cyclic and these elements are denoted as $Z(q)^k$ where $Z(q)$ is the generator. For instance if $q = 5$ we can take $Z(5) = 2$ with successive powers $Z(5)^2 = 4, Z(5)^3 = 3, 1 = Z(5)^0$.

```
gap>  x:=Indeterminate(GF(8),1);
gap>  h:=x^2+1;
gap>  Factors(h);
gap>  k:=2*x-1;
gap>  h mod k;
gap>  EuclideanRemainder(x^12,h);
```

Calculating with matrices in GAP

A matrix is just a list of lists, every list being the elements of a row of the matrix. There are special command to calculate the kernel and the solution-spaces over the rationals or over the integers. There is a way to denote the identity matrix.

```
gap> M:=[[0, 1, 2],[2 ,3, 4],[2,4,6]];
[ [ 0, 1, 2 ], [ 2, 3, 4 ], [ 2, 4, 6 ] ]
gap> NullspaceIntMat(M);
[ [ 1, 1, -1 ] ]
gap> m:=IdentityMat(3); M-m;
gap> NullspaceIntMat(M-m);
gap> mat:=[[1,2,7],[4,5,6],[7,8,9],[10,11,19],[5,7,12]];;
gap> SolutionMat(mat,[95,115,182]);
[ 47/4, -17/2, 67/4, 0, 0 ]
gap> SolutionIntMat(mat,[95,115,182]);
[ 2285, -5854, 4888, -1299, 0 ]
```

Bibliography

[C-C-S] Cohen, A., H. Cuypers and H. Sterk:*Some tapas of computer algebra*, Algorithms and Comp. in Math. **4**, Springer-Verlag, New York, Berlin etc. (1999)

[C-L-O'S] Cox, D., J. Little and D. O'Shea: *Ideals, varieties and algorithms*, Undergrad. Texts in Math. Springer-Verlag, New York, Berlin etc. (1992)