PREMUR 2007 Seminar
Week 2 Computer Laboratory Exercises

Background and Goals

This week, we will begin to work with some of the commands related to monomial orders and computing Gröbner bases in Mathematica. Note: this time the lab problems you are to hand in are "sprinkled through" the discussion, marked *Exercise 1*, *Exercise 2*, and so forth.

Days 1,2: Lists, Polynomial Division, Defining Monomial Orders

Before we start in with the division algorithm, we need to take a bit of time and look at some of the details of *lists* in Mathematica.

Lists and Vectors

To represent points in affine space, we use coordinate vectors. The same idea is also useful to represent any *ordered collection* of information. In Mathematica, these ordered collections are called *lists*, and the elements of a list can be *anything.* A list is indicated by a pair of *braces* ({,}) enclosing the items in the list, separated by commas. For instance, the input line

```
list1 = {a,b,c,d,e}
```

creates a list with five items (the letters a, b, c, d, e, treated as symbolic variables) and assigns the list to the name `list1`. We have said that lists are *ordered*. What this means, for instance, is that `alist` above is *different from* the list defined by `list2 = {a,d,c,e,b}`. You can test this by entering the definition of `list2`, then the command

```
list1 === list2
```

(The === is the comparison operator – not assignment; it returns a True or False value indicating whether the two lists are identical.)

A list can have any number of items, including *no items* at all – the empty list is written as {}. The same item can also occur at several places in a list (unlike the case for a *set*, which is an unordered collection of information with no repetitions. Note that the Mathematica notation for lists is the same as the mathematical notation for sets, but means something else(!)).

The items in a list *do not* all have to be items of the same *type.* For instance, another perfectly OK Mathematica list is the following:

```
list3 = {{a,b,c},d,e}
```

where there are three items in all. (The first list item in `list3` is the list `{a,b,c}`, and the second and third items are the symbolic variables d, e.)

Some useful list operations

- The built-in `Length` function returns the number of items in a list. (If one of the items is a list itself, it is counted as a single item.)
- Two builtin commands `First` and `Last` select the first and last elements in a list.
- To pick out the ith item in a named list (like `list1`), you can use the notation `list1[[i]]`.
- You can also pick out a consecutive sublist – the items in the slots numbered `first` to `last` – from a list by using the format `Take[list1,{first,last}]`. This gives a list as output.
- The `Append` command adds a new element *at the end of a list.* Try

  ```
  list3 = Append[list3,{g,h}]
  ```

  for instance.
- The `Reverse` command does what its name implies(!) (But, if one of the items in a list is a list itself, those elements are not reversed – the reversal only happens at the "top level.")
- The `Flatten` command takes a list that may contain other lists as elements, and generates one big list by removing the inner braces. Try `Flatten[list3]` to see the idea.

Exercise 1

Explain, and give examples showing:

a) how to insert a new item at the *start* of a list
b) how to create a list containing the entries of a given list, followed by a second copy of that same list
c) how to insert a new item as the third entry from the start in a list (assuming it has at least two entries to begin with) Be careful to test your idea both on lists with two and with more than two entries!

Note: There are other builtin list commands not discussed above. Some of these tasks may be covered by one of them, but they can also be done using the commands above.

Matrices

If we think of a list as a row vector, then it is natural to think of a list of lists (all of the same size) as a way to represent a *matrix*. We can do this in Mathematica as well, as a basic way to deal with square or more general rectangular matrices. For example,

```
mat = {{x,y},{u,v}}
```

creates a list of two row vectors, or a 2×2 matrix. If you want to see this presented in matrix format, use `MatrixForm[mat]`.

To select the entry in row i and column j from a matrix defined this way, we could use, for instance `mat[[i,j]]` (or `mat[[i]][[j]]` – in this second way, `mat[[i]]` is the ith row, so `mat[[i]][[j]]` is the jth entry in that row).

Polynomial Division

The items in a list can be any legal Mathematica objects – constants, variables, polynomials, expressions, other lists, ... For example, in many of our calculations, we will be dealing with lists of polynomials in a ring $k[x_1, \ldots, x_n]$. We will think of these as particular lists of generators for ideals $I \subset k[x_1, \ldots, x_n]$. For example,

```
Id = {x^3 - y^2 + 2 y, x y - y - 4 z, z^4 - 1}
```

would be one way to represent the ideal

$$Id = \langle x^3 - y^2 + 2y, xy - y - 4z, z^4 - 1\rangle.$$

(Recall that the ideal contains all the polynomials

$$g_1 \cdot (x^3 - y^2 + 2y) + g_2 \cdot (xy - y - 4z) + g_3 \cdot (z^4 - 1)$$

for $g_1, g_2, g_3 \in k[x, y, z]$. The list notation will parallel our notation from class for the ideal generated by a given set: $I = \langle f_1, \ldots, f_s\rangle$.

To divide a polynomial f by a list of divisors $\{f_1, \ldots, f_s\}$ using a particular monomial order, the basic Mathematica command is `PolynomialReduce`, which has the format:

```
PolynomialReduce[f,polylist,varlist,monorder]
```

Mathematica has a completely general way to specify monomial orders that we will see shortly. It also has also several simplified predefined special cases for you to use. For example:

- The *lexicographic* order in $k[x_1, \ldots, x_n]$ with

  $$x_1 > x_2 > \cdots > x_n$$

  is *the default*. No `monorder` specification is necessary in this case. (Note: The ordering of the variables is determined by the order in which you list them in the `varlist`. Also, you cannot use dots in actual commands; you need to list out all the variables explicitly. For example, enter

  ```
  f = x^ 2 + x y - x
  divisors = {x + y^2 - 1, x y + 2 x + 1}
  PolynomialReduce[f,divisors,{x,y}]
  ```

  The output will be a list with two entries. The first is the list of quotients (one for each divisor), and the second is the *remainder* on division. Since we just specified the list of variables, the computation used the lex order with $x > y$.
- the *graded reverse lex order* in $k[x_1, \ldots, x_n]$ with

  $$x_1 > x_2 > \cdots > x_n$$

  is represented by giving the list of variables, ordered as above, `then`

```
MonomialOrder -> DegreeReverseLexicographic
```

(separated from the stuff before by a comma, inside the outer square brackets).

- The graded lex order is similarly indicated by

```
MonomialOrder -> DegreeLexicographic
```

Note: The basic Mathematica system does not include a direct command for determining leading terms of polynomials with respect to monomial orders. There is a package that we'll be using later that does this, and many other things besides.

Exercise 2

We did problems 1 and 2 from Chapter 2, §3 of "IVA" in discussion. Check your work using Mathematica.

Exercise 3

Do problems 5 and 6 in Chapter 2, §3 of "IVA".

Defining a Monomial Order by a Matrix

In discussion, we saw that we can generalize the weight orders $>_w$, to define monomial orders on $k[x_1, \ldots, x_n]$ starting from any $m \times n$ matrix M with

- $m \geq n$,
- $\operatorname{rank}(M) = n$,
- all entries non-negative integers.

Namely, suppose the rows of M are the vectors $w_1, \ldots, w_n$. Then we can compare monomials x^α and x^β by first comparing their w_1-weights, then breaking ties sucessively with the w_2-weights, w_3-weights, and so on until the w_m-weights. In symbols:

$$
\begin{aligned}
x^\alpha >_M x^\beta \Leftrightarrow\ & w_1 \cdot \alpha > w_1 \cdot \beta \\
& \text{or } [(w_1 \cdot \alpha = w_1 \cdot \beta) \text{ and } (w_2 \cdot \alpha > w_2 \cdot \beta)] \\
& \text{or } [(w_1 \cdot \alpha = w_1 \cdot \beta) \text{ and } (w_2 \cdot \alpha = w_2 \cdot \beta) \text{ and } (w_3 \cdot \alpha > w_3 \cdot \beta)] \\
& \text{or } \cdots \\
& \text{or } [(w_1 \cdot \alpha = w_1 \cdot \beta) \text{ and } \cdots \text{ and } (w_{m-1} \cdot \alpha = w_{m-1} \cdot \beta) \text{ and } (w_m \cdot \alpha > w_m \cdot \beta)]
\end{aligned}
$$

All monomial orders can be specified as $>_M$ orders for appropriate matrices M. In Mathematica, M must be a square matrix with rational entries.

To specify that we want to use the matrix order defined by a suitable square matrix `wmat` for polynomial division in Mathematica, we enter

```
MonomialOrder -> wmat
```

in the `PolynomialReduce` command. The matrix `wmat` itself is entered as a list of lists as above, either first in a command assigning the matrix to a name, or else directly in the `PolynomialReduce` command. The `varlist` indicates which components in the weight vectors (the rows of `wmat`) apply to which variables. For example, if for some reason we did not want to use the builtin lex order, we could define something equivalent on $\mathbf{Q}[x, y, z]$ using

```
MonomialOrder -> {{1,0,0},{0,1,0},{0,0,1}}
```

Exercise 4

What matrix would define the weight order on $\mathbf{Q}[x, y, z]$ where we compare exponents first with the weight vector $(1, 3, 7)$, then break ties using the lex order? Create an appropriate matrix term order in Mathematica, and repeat the division from Exercise 2a in Chapter 2, §3 from "IVA".

Additional Examples of Monomial Orders

Exercises 10 and 12 from Chapter 2, §5 of "IVA" contain two additional important examples of monomial orders – the *product* orders, and Bayer and Stillman's *elimination* orders.

Exercise 5

Let $R = \mathbf{Q}[x_1, x_2, x_3, y_1, y_2, y_3, y_4]$.

a) Using the matrix order setup in Mathematica, determine how to define a mixed order $>_{mixed}$ on R where

$$x^\alpha y^\beta >_{mixed} x^\gamma y^\delta \Leftrightarrow x^\alpha >_{lex} x^\gamma \text{ or } ((x^\alpha = x^\gamma) \text{ and } (y^\beta >_{grevlex} y^\delta)).$$

(Hint: Think of making the matrix from "blocks" like the ones we have seen for the orders separately.)

b) Using Mathematica's setup, determine how to define the elimination order $>_3$ from Exercise 12.

Days 3,4: S-polynomials and Gröbner Bases

Today, we will start to use some of the commands in the Mathematica package for "IVA" mentioned above. To load this package into your Mathematica session, use

```
<< path/groebner40.m
```

The first thing to learn is how to set monomial orders – this is somewhat different from what happens in the builtin Mathematica commands that use this.

The relevant package command is `MonOrder`. This command is used to set the monomial order used in all other commands in the package. The allowable monomial orders are denoted as follows:

- `Lex` (lexicographic order)
- `Grlex` (graded lexicographic order)
- `Grevlex` (graded reverse lexicographic order)
- {`k,n`} (elimination order, eliminating the first k of n variables)
- $\{\{a_{11}, ..., a_{1n}\}, ..., \{a_{n1}, ..., a_{nn}\}\}$ (matrix order for n variables).

Note the use of capital letters in the names of the first three of these orders. Thus, to change the monomial order to Grlex, you would issue the command

```
MonOrder[Grlex]
```

Furthemore, the command

```
MonOrder[]
```

will return the current monomial order. This is useful if you've forgotten what the order is. The default monomial order is Lex. Note that the list of variables is not specified at this point. That happens in commands for S-polynomials, etc. that use a monomial order.

S-polynomials and other calculations

The package command for computing an S-polynomial is called `SPoly`. The format is

```
SPoly[poly1,poly2,varlist]
```

This command returns the S-polynomial of `poly1` and `poly2`, and `varlist` determines the order of the variables used in the current monomial order.

Exercise 6

Do problem 5 from Chapter 2, §6 of "IVA" using the `SPoly` command and lex order with $x > y > z$, lex with $z > y > z$, and then graded lex with $x > y > z$. What is the answer to the following problem 6 based on the results?

Buchberger's Criterion

Recall that Buchberger's Criterion says $G = \{g_1, \ldots, g_t\}$ is a Gröbner basis for the ideal it generates if and only if for all pairs $i \neq j$,

$$\overline{S(g_i, g_j)}^G = 0,$$

where $\overline{f}^G$ is the remainder on division by G. Although we will be skipping the proof of this fact, it is very important – is the foundation for the algorithm for computing Gröbner bases that we will introduce.

Exercise 7

Using Buchberger's Criterion, do parts a and c of problem 9 in Chapter 2, §6 of "IVA." The remainders can be computed via the builtin command `PolynomialReduce` we discussed before, or via a command in the package called `PRemainder`. The format for this command is

`PRemainder[f,{f1,...,fs},varlist]`

where `f` is the polynomial to be divided and `{f1,...,fs}` is the list of polynomials to divide by. The output will be the remainder on division. The `varlist` is a list of variables. You should be aware that the order of the variables in varlist is important since it (together with what you specified in `MonOrder`) determines the monomial order.

Buchberger's Algorithm

Starting from any set of generators for an ideal I, Buchberger's Algorithm produces a Gröbner basis for I by building up a set of polynomials in the ideal for which Buchberger's Criterion is satisfied. This is done by adjoining any nonzero S-polynomials that are discovered in applying the criterion to the current set of polynomials. The pseudocode description is given on p. 87 of "IVA" and was discussed in class.

Exercise 8

Using Mathematica to "automate" the calculation of the S-polynomials, but controling the current list of polynomials "by hand" apply Buchberger's Algorithm to find a Gröbner basis of the ideal

$$\langle x^2 - xy + 1, xy - y + 2\rangle$$

using lex order with $x > y$, then lex with $y > x$, and finally the matrix order $>_M$ with

$$M = \begin{pmatrix} 2 & 5 \\ 3 & 1 \end{pmatrix}.$$

Exercise 9

There is a builtin `GroebnerBasis` command in Mathematica as well (this is not part of the IVA package). Look up the online documentation to see the format – it's very similar to `PolynomialReduce`. What happens if you apply it to the ideal from Exercise 8 with these monomial orders? Show that the bases that Mathematica finds generate the same ideal as your answers. (Recall the ideal membership test based on remainders.)