MACAULAY2 IN IBADAN

TAYLOR BRYSIEWICZ

Contents

1. Macaulay2 Tips

1.1. **Random tips.** Here is an example of a Macaulay2 session.

```
i1 : QQ[x,y,z]
o1 = QQ[x, y, z]
o1 : PolynomialRing

i2 : f=(x+y+z)
o2 = x + y + z
o2 : QQ[x, y, z]

i3 : g=(x-y-z)
o3 = x - y - z
o3 : QQ[x, y, z]

i4 : h=(x+y-z)^2
      2           2                   2
o4 = x  + 2x*y + y  - 2x*z - 2y*z + z
o4 : QQ[x, y, z]

i5 : I=ideal(f,g,h)
                                    2           2                   2
o5 = ideal (x + y + z, x - y - z, x  + 2x*y + y  - 2x*z - 2y*z + z )
o5 : Ideal of QQ[x, y, z]

i6 : radical(I)
o6 = ideal (z, y, x)
o6 : Ideal of QQ[x, y, z]

i7 : J=oo
o7 = ideal (z, y, x)
o7 : Ideal of QQ[x, y, z]

i8 : J
o8 = ideal (z, y, x)
o8 : Ideal of QQ[x, y, z]
```

i1 Creates a ring in three variables over the rational numbers

i2,i3,i4 define polynomials in this ring. We give these polynomials names f, g, and h. After each definition, Macaulay2 tells us what we just wrote, and tells us where that object lives. So o4 tells us the polynomial h we wrote (expanded) and tells us the ring it lives in, $\mathbb{Q}[x, y, z]$.

i5 creates an ideal generated by f, g, and h and names this ideal I.

i6 takes the radical of I. But look, we forgot to name it!

i7 Names the last output J by using the oo command which refers to the last line outputted.

i8 Tells us what J is: the ideal generated by z,y, and x.

1.2. **Making a ring.** To make a ring, you need to choose a field. For us, this will usually be the rationals, denotedd

```
QQ
```

but it could also be the reals or complex numbers,

```
RR, or CC
```

Here is an example of making a ring

```
R=QQ[x,y]
```

What if I want to make a ring with many variables? I may not want to write $x_1, \ldots, x_{40}$ all by hand. Instead, I can write the following in Macaulay2:

```
R=QQ[x_1..x_40]
```

I can also have subscripts which are themselves lists.

For example, a ring over the complex numbers in the variables $x_{(0,0)}, x_{(0,1)}, x_{(1,0)}, x_{(1,1)}$ can be written quickly by

```
R=CC[x_(0,0)..x_(1,1)]
```

If we want the number of variables in the ring, we can type

```
numgens R
```

If we want to refer to the 1st variable we write

```
R_0
```

since Macaulay2 indexes beginning at zero.

This means that in order to get the 3rd variable of R we would write

```
R_2
```

1.3. **Making a ring: Exercises.**

(1) Make a ring in 4 variables over the rational numbers.

(2) Make a ring with variables $x_{(0,0,0)}, x_{(0,0,1)}, x_{(0,1,0)}, \ldots, x_{(1,1,1)}$.

(3) What is the 7th variable in the above ring?

(4) Create a ring over the complex numbers.

1.4. **Matrices.** Creating a matrix usually begins with creating a list of lists. The following command creates the matrix

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 8 & 12 \end{bmatrix}$$

```
M=matrix{{1,2,3},{4,8,12}}
```

If I want the list of lists $\{\{1,2,3\},\{4,8,12\}\}$ back again, I would say.

```
entries(M)
```

I can take transposes of matrices

```
N=transpose(M)
```

and I can multiply or add matrices together

```
S=N*M
T=M+matrix{{1,1,1},{2,2,2}}
```

I can take the determinant of a matrix

```
det(matrix{{3,5,6},{1,4,2},{2,3,4}})
```

and I can ask for the rank of a matrix

```
rank(M)
```

I can create an ideal of minors of a matrix

```
I=minors(2,matrix{{3,4,5,6},{1,5,2,1}})
```

Finally, we can use variables as entries of a matrix.

```
QQ[x_(0,0)..x_(1,1)]
genM=matrix{{x_(0,0),x_(0,1)},{x_(1,0),x_(1,1)}}
```

Here we have a generic 2×2 matrix. But there is a quicker way to do this!

```
R=QQ[x_(0,0)..x_(1,1)]
genM=genericMatrix(R,2,2)
```

where the arguments $2, 2$ indicate we want a generic 2 by 2 matrix in the variables of R.

If a matrix is invertible, the command 'inverse' will return its inverse.

```
inverse(matrix{{1,4},{1,8}})
```

1.5. **Matrices: Exercises.**

(1) Create a 4 by 7 matrix. Multiply it by its transpose (on either side). Is there anything special about this new matrix?

(2) Is the matrix

$$M = \begin{bmatrix} 6 & 10 & 14 & 18 \\ 6 & 8 & 10 & 12 \\ 11 & 14 & 17 & 20 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

invertible? Check its determinant. Determine its rank.

(3) Write the polynomial, f, which is the determinant of a generic 4 by 4 matrix. Evaluate this polynomial at the matrix in the previous problem using the command

```
sub(f,flatten for i from 0 to 3 list for j from 0 to 3 list x_(i,j)=>(entries M)#i#j)
```

Make another 4 by 4 matrix of numbers and evaluate the determinant at this matrix.

(4) Explicitly show an example of the noncommutativity of matrix multiplication.

1.6. **Making an ideal and looking at generators.** Let's consider an ideal generated by two polynomials in two variables

```
R=QQ[x,y]
f=(x+y)^7
g=(x-y)^(20)
I=ideal(f,g);
```

If I want to get the first generator of this ideal, I could write

```
I_0
```

and if I wanted the second, I would write

```
I_1
```

If I want a list of all of the generators, I would write

```
L=flatten entries gens I
```

Let's unpack the above command.

```
gens I
```

gives you a matrix of the generators.

```
entries gens I
```

gives you a list of lists of the generators. But we don't want a list of lists, we want just one list. So we use

```
flatten entries gens I
```

We can ask questions about these ideals, such as what their dimensions and degrees are.

```
dim(I)
degree(I)
```

1.7. **Making an ideal and looking at the generators: Exercises.**

(1) Create an ideal generated by a polynomial of degree 3 and of degree 4 in 2 variables. What is its dimension? What is its degree?

1.8. **Creating lists.** A list in Macaulay2 is delimited by { and }. For example, if I wanted a list of the first 5 natural numbers, I could write

```
L={1,2,3,4,5}
```

I could also write it as

```
L=for i from 1 to 5 list i
```

or

```
L={1..5}
```

These all do the SAME thing: they define a list L of the first 5 natural numbers.

Now, let's say that I screwed up, and I really wanted the first 6 natural numbers. One solution is to use the command 'append'.

Append will add an element to the end of a list. The command

```
L=append(L,6)
```

has now redefined L to be the list consisting of the elements in L followed by the element 6.

Let's make another list, which I will call T.

```
T=for i from 1 to 5 list(for j from 1 to 3 list(10*i+j)
```

What is in T now?

Let's unpack this. First we loop through i from 1 to 5 and for each value of i, we list through j from 1 to 3.

This means we will have the list $\{\{11, 12, 13\}, \{21, 22, 23\}, \{31, 32, 33\}, \{41, 42, 43\}, \{51, 52, 53\}\}$.

It is a list of lists! If we write

```
T=flatten for i from 1 to 5 list(for j from 1 to 3 list(10*i+j)
```

then T becomes the list $\{11, 12, 13, 21, 22, 23, 31, 32, 33, 41, 42, 43, 51, 52, 53\}$.

If we want a list with all of the elements of L and all of the elements of T together, we could use the 'join' command.

```
S=join(L,T)
```

combines the two lists into the list

$$\{1, 2, 3, 4, 5, 6, 11, 12, 13, 21, 22, 23, 31, 32, 33, 41, 42, 43, 51, 52, 53\}$$

What if we want to know the 8-th element of this list? (the number 12)

Then we use the command

```
S#7
```

Why did we use 7 instead of 8? This is because Macaulay2 indexes its lists starting at 0. So

```
S#0
```

gives the first element of S, namely the number 1.

To get the first 13 elements, we would write

```
B=for i from 0 to 12 list L#i
```

Make a list of the first 100 Fibonacci numbers.

Make a list of the numbers from 1 to 100 followed by the numbers from 1000 to 1100.