

**MACAULAY2 WORKSHEET FOR CHARACTERISTIC p
COMMUTATIVE ALGEBRA
JANUARY 23RD, 2017**

DUE FEBRUARY 6TH, 2017

You are encouraged to work in groups of up to 3, only one assignment is due for each group, but everyone needs to work together. Your solution must be LaTeX'd on this assignment.

There are lots of commands below. You are expected to play around with them to figure out how they work on your own. You can always use `help commandname` or possibly `viewhelp commandname`, or look at the documentation on Macaulay2's website.

1. RUNNING MACAULAY2 FOR THE FIRST TIME

It is easiest to run Macaulay2 from within emacs, so we will explain how to do that. If you are already running Macaulay2 in emacs regularly, you can probably skip this step. Once you have Macaulay2 installed, to set it up from your account for the first time open up a terminal and execute

`M2`

It should show you something like:

`Macaulay2, version 1.8.2`

`with packages: ConwayPolynomials, Elimination, IntegralClosure, LLLBases,
PrimaryDecomposition, ReesAlgebra, TangentCone`

`i1 :`

From here, execute the command

`i1 : setup()`

which should probably ask you some questions, answer yes to all of them.

Now exit out of the command line Macaulay2 by running

`exit`

Once you are back in the regular terminal, run

`emacs &`

Once emacs starts, hit the F12 key, this should start Macaulay2 (I hope). You may want to hide the existing emacs buffer by hitting `ctrl+X` followed by `1`. Now you are running emacs in a happy little window with what most people find is a nicer interface than the command line (ie, there is reasonable Macaulay2 syntax highlighting, autocompletion of commands works).

2. MAKING A RING AND A MAP OF RINGS

From within Macaulay2 execute the following commands.

```

i1 : R = ZZ/5[x,y,z]
i2 : S = ZZ/5[a,b]
i3 : f = map(S, R, {a, a*b, b^2})
i4 : I = ker f
i5 : T = R/I

```

Commands `i1` and `i2` simply create two rings R and S . Command `i3` creates a ring map $f : R \rightarrow S$ (note the first entry is the source, the second is the target). Indeed, to specify such a ring map one must say where the variables $\{x, y, z\}$ go, this is done by the list $\{a, a * b, b^2\}$ which says that $x \mapsto a, y \mapsto ab, z \mapsto b^2$.

Next we compute the kernel of f and form the quotient ring $T = R/I$ so that $T \cong \text{Image}(f)$ by construction.

3. FROBENIUS

Exercise 1. Define the Frobenius maps $F_R : R \rightarrow R$ and $F_T : T \rightarrow T$ in your Macaulay2 session. I called my maps `FR` and `FT` respectively.

Hint: If you are having trouble convincing Macaulay2 which ring your variables live in, you can always use the commands `use R` and `use T` respectively.

We want to view R as an R -module via Frobenius, to do that we need the `PushForward` package, which we start by running.

```
i7 : loadPackage "PushForward"
```

Once the package is loaded, we can pushforward modules. For example, first I'm going to create a rank-1 free R -module M and then push it forward by Frobenius (for R).

```

i13 : M = R^1
i14 : FM = pushFwd(M, FR)

```

Verify that Macaulay2 gives you something reasonable.

Exercise 2. Next, make N , a free T -module of rank 1 and push it forward by F_T to obtain a module I am calling `FN`. Verify that it is not locally free via the command `pdim` which computes the projective dimension. What answer should it give you if it was free. *Aside:* Note this only computes the projective dimension up to a certain limit, in this case 4. If you want to compute a longer (partial) projective resolution, you can use the command `resolution(FN, LengthLimit=>10)` and observe that it will compute a longer partial resolution.

Warning #1: I suspect that Macaulay2 will not always give the right answer for projective dimension if R is not a graded ring and M is not a graded module. You have been warned! Note in this case, if you set $k = T/\text{ideal}(x, y, z)$ and compute

```
dim Ext^1(FN, k)
```

you'll get something > -1 which pretty well proves that `FN` is not projective.

Warning #2: Note the command `isFree` only checks whether Macaulay2 is thinking of a module as locally free at the moment, it *does not check* whether the module is actually locally free.

We have just verified that T is not regular by checking that F_*T is not locally free. That's a ridiculous way to do this. The next section gives us another option.

4. FUNCTIONS

We are going to create a function which takes an ideal $I = \langle f_1, \dots, f_n \rangle$ and returns the ideal $I^{[p]} = \langle f_1^p, \dots, f_n^p \rangle$. First let's learn the necessary commands, like how to extract a list of generators from an ideal.

```
i38 : use T
i39 : J = ideal(x^4, x*y- z^5, y^6 - z^3)
i41 : L = first entries gens J
```

At this point L is a list a generators of J . Let's make a list of these generators to higher powers by running the command:

```
i43 : p = char T
i44 : Lp = apply(L, u -> u^p)
i45 : Jp = ideal(Lp)
```

The command `apply` applies a function to every element in the list. In our case, we made an inline ad-hoc function `u->u^p`.

More generally, the command `myFunction = u -> u = u+1; 2*u` should create a function that takes in a number, adds one to that, and then doubles the output. Indeed, try `myFunction(5)`.

Exercise 3. Create a function `frobeniusPower` which takes an ideal I and returns $I^{[p]}$.

Hint: If you are trying to grab the characteristic, note that `char ring I` should give you the characteristic of the ring.

After you've made this function, run the following commands.

```
i54 : use T
i55 : m = ideal(x,y,z)
i56 : mp = frobeniusPower(m)
i57 : degree(T^1/mp)
i58 : (char T)^(dim T)
```

Note `degree` computes the length of a module. Hopefully those two numbers you constructed are different. On the other hand, if T was regular of dimension 2 at the origin, by the completion argument we did earlier in class, $T/\mathfrak{m}^{[p]}$ should have length $p^{\dim T} = 5^2 = 25$. Indeed, this sounds like a good non-Macaulay2 exercise.

Exercise 4. Prove that if (R, \mathfrak{m}, k) is a regular local ring of dimension d , characteristic $p > 0$ and with a perfect residue field k , then

$$\text{length}_R(F_*R)/\mathfrak{m} = \text{length}_R F_*(R/\mathfrak{m}^{[p]}) = \text{length}_R(R/\mathfrak{m}^{[p]}) = p^d.$$

Make sure to explain why we can remove the F_* in the second to last equality.

5. THE OTHER WAY TO CHECK REGULARITY

There is one other obvious way to check regularity, via the Jacobian condition. I'll step you through this and then show you the easy way to get it.

```
i62 : use T
i63 : I = ideal T
i64 : jacobian I
i65 : minors(1, jacobian I)
```

```
i66 : jac = sub(minors(1, jacobian I), T)
i67 : jac == ideal(1_T)
```

Note the Jacobian of an ideal is a matrix (as it probably should be). We then compute the 1×1 minors of that matrix. Finally we substitute that matrix back into T and check whether we get the whole ring.

Another way to accomplish the same thing is via the command `singularLocus`.

Exercise 5. Play around with the `singularLocus` command and figure out how to use it to construct the ideal `jac` we constructed above. Turn in a sequence of commands that does it.

Finally, let's make a more interesting singular ring. Run the following commands.

```
i72 : restart
i1  : R = ZZ/5[a,b]
i2  : S = ZZ/5[x,y,z,u,v,w]
i3  : f = map(R, S, {a^5, a^4*b, a^3*(b^2 - b), a^2*b^3, a*b^4, b^5})
i4  : I = ker f
i5  : T = S/I
```

Feel free to make your own more singular ring T , the way I am constructing it, we will definitely get a domain though.

Exercise 6. Verify that the ring T (or your own more interesting ring) is singular. Compute the singular locus and use Macaulay2 to help you write it in as simple a way as possible. What is the dimension of T , what is the dimension of the singular locus of T ?

Hint: The command `minimalPrimes` should give you the minimal primes in a primary decomposition of an ideal.