AWS_1_Magma_Basics

March 5, 2026

0.1 Question 0: The basics

The purpose of this question is to familarise you with some of the basic functionality of Magma. If you have used Magma before, feel free to skip this question!

The main object you will use in Magma is the *sequence*. Informally, sequences contain elements all from the "same place". For example, `[1..10]` returns the sequence of integers from 1 through 10.

[0]:
```
for i in [1..10] do
    print i;
end for;
```

There are some useful pieces of syntax for creating sequences. Using the pipe `|` allows us to pick out specific elements of a sequence. For example, if we want the primes up to 100 that are 1 (mod 4), we can write the following:

[0]:
```
[p : p in PrimesUpTo(100) | p mod 4 eq 1];
```

We can use sequences to define matrices, in the format `Matrix(R,a,b,seq)`, where `R` is a ring, `a` is the number of rows, `b` is the number of columns, and `seq` is a sequence giving the entries from left to right, top to bottom. For example:

[0]:
```
Z := Integers();
M := Matrix(Z,2,3,[1,2,3,4,5,6]);
M;
Transpose(M) eq Matrix(Z,3,2,[1,4,2,5,3,6]);
```

The fact that sequences contain elements all from the same place mean that strange things can sometimes happen. Can you figure out why this sequence behaves the way it does?

[0]:
```
n := 5;
M := Matrix(Z,2,2,[1,4,-2,3]);

[n, M];
```

If you need to store elements that are not from the same place, you will use a *List*, which is created by using `[* … *]`.

[0]:
```
n := 5;
M := Matrix(Z,2,2,[1,4,-2,3]);
```

```
[*n, M*];
```

Every object in Magma has a type, which is a string encoding the kind of object it is. The command `Type` returns this string. For eg, if P is a point on an elliptic curve, you can call `Type(P)` and it will return the string `PtEll`.

Sometimes, it is helpful to know all things that can be computed for a certain type, i.e., all intrinsics whose input is a certain type. Similarly, all intrinsics whose output is a certain type. This can be obtained using`ListSignatures`. The output is usually quite long.

For eg., which Magma intrinsics take an object of type `PtEll` as input? Or which Magma intrinsics produce an object of type `"PtEll"` as output? Make note of the intrinsics you find interesting, and try to use them.

[0]:
```
ListSignatures(PtEll : Search := "Arguments", Isa := false);
ListSignatures(PtEll : Search := "ReturnValues", Isa := false);
```

Most Magma intrinsics have natural names that you would expect. Some examples.

1. To create a number field, elliptic curve or a hyperelliptic curve, the respective commands are `NumberField`, `EllipticCurve`, `HyperellipticCurve`.
2. To access the signature, ring of integers, class number of a number field, the respective commands are `Signature`, `RingOfIntegers`, `ClassNumber`.
3. To access the conductor, bad primes, Mordell-Weil group of an elliptic curve, the respective commands are `Conductor`, `BadPrimes`, `MordellWeilGroup`.

Note the Camel case convention in intrinsic names.

To know more information about an intrinsic (its arguments, i.e, input(s), return values, i.e., output(s), and what it does), you can just enter the intrinsic name as the command. This will return brief info about all intrinsics with that name. There could be multiple.

For eg., to find out about the intrinsic `WeilPairing`, we do:

[0]:
```
WeilPairing;
```

To get more detailed information from the handbook (directly in your terminal), include a `?` at the start.

[0]:
```
?WeilPairing
```

To access the code behind the intrinsic, you will need the location of the file in which the intrinsic is defined. This can be found by appending `:Maximal;` at the end.

[0]:
```
WeilPairing:Maximal;
```

0.2 Question 1: Partitions

Write $p(n)$ for the number of partitions of the positive integer n. Using `NumberOfPartitions`, verify Ramanujan's congruence: $p(5k+4) \equiv 0 \pmod 5$ for $k = 0, \ldots, 20$.

[0]:
```

```

Ramanujan also proved two more congruences of this kind, modulo 7 and 11. Experiment with `NumberOfPartitions` to figure out what these might be.

[0]:
```
```

0.3 Question 2: A representation of A_4

In this question we will construct a rational representation of A_4. The **Hurwitz order** is a maximal order $\mathcal{O}_B$ of the quaternion algebra $B = \left(\frac{-1,-1}{\mathbb{Q}}\right)$. The algebra B is just the Hamiltonian quaternions $\mathbb{H}$, bu with rational coefficients, rather than reals:

$$B = \mathbb{Q} \oplus \mathbb{Q}\mathbf{i} \oplus \mathbb{Q}\mathbf{j} \oplus \mathbb{Q}\mathbf{k},$$

with $\mathbf{i}^2 = \mathbf{j}^2 + \mathbf{k}^2 = \mathbf{ijk} = -1$. We can define quaternion algebras in Magma:

[0]:
```
B := QuaternionAlgebra(Rationals(),-1,-1);
OB := MaximalOrder(B);
```

The unit group of $\mathcal{O}_B$ is finite and isomorphic to $\mathrm{SL}_2(\mathbb{F}_3)$, which we can check as follows:

[0]:
```
U, phi := UnitGroup(OB);
GroupName(U);
```

The algebra $\mathbb{H}$ acts on $\mathbb{R}^3$ by identifying $\mathbb{R}^3$ with the space of *purely imaginary quaternions:*

$$\mathfrak{I}(\mathbb{H}) = \{x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \mid x, y, z \in \mathbb{R}\},$$

then conjugating

$$(x, y, z) \cdot q = q^{-1}(x\mathbf{i} + y\mathbf{j} + z\mathbf{k})q.$$

This actions restricts to B. Its kernel is $ $\pm 1$$, giving a faithful representation of $A_4 \simeq \mathrm{SL}_2(\mathbb{F}_3)$.

First, we need a function that takes a quaternion `q` and returns a matrix giving its action on a vector `(x y z)`. Magma uses row vectors, so matrices act on the *right!* This is important to remember and can become confusing!

Magma has lots of ways to construct matrices. The most straightforward way is with the following command:

`M := Matrix(Rationals(), 3, 3, [1, 2, 3, 4, 5, 6, 7, 8, 9 ]);`

It is also possible to apply `Matrix` to a sequence of sequences with the appropriate sizes. If we have

`L := [ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ];`

then `Matrix(L);` will give the same matrix. This is useful because we can construct an operator on a basis of some space by going one vector at a time.

[0]:
```
ActionOfQuaternion := function(q)
    // your code here...
```

```
end function;
```

We can then define the representation of A_4 by assigning a matrix to each generator of the group `U`, using `ActionOfQuaternion`. Check the Magma documentation for G-modules to see how to do this.

[0]:

Extension: Is the A_4 module we defined irreducible over $\mathbb{C}$?

Extension: Are there any non-split central extensions of A_4 by this module?

[0]:

0.4 Question 3: Supersingular j-invariants

We can test whether a particular elliptic curve `E` defined over a finite field is supersingular with the command `IsSupersingular`, as well as create a polynomial with all supersingular j-invariants as its roots (aside from 0 and 1728) using the command `SupersingularPolynomial`. Which primes $p \leq 100$ have all supersingular j-invariants defined over $\mathbb{F}_p$ (as opposed to the generic case, $\mathbb{F}_{p^2}$)? What about the primes $p \leq 200$?

[0]:

Extension: Use the internet to find out more about the significance of these primes.

0.5 Question 4: Counting subgroups

Using `SmallGroups`,`Subgroups` and `GroupName`, answer the following question: out of all the groups of order 32, which has the most subgroups up to conjugacy? Which has the least?

[0]:

Out of all the groups of order 32, how many have 50 total subgroups (i.e. not up to conjugacy)?

[0]:

0.6 Question 5: L-polynomials

Let E be the elliptic curve

$$E\colon y^2 = x^3 - 27x + 8694$$

We are going to test the formula for the L-polynomial of E at $p = 2$, which states that

$$\exp\left(\sum_{n=1}^{\infty} \frac{\#E(\mathbb{F}_{p^n})}{n} t^n\right) = \frac{1 - a_p t + pt^2}{(1-t)(1-pt)}$$

where $a_p = 1 + p - \#E(\mathbb{F}_p)$, valid when p is coprime to the conductor of E. Define a `PowerSeriesRing` of precision 10, and create the series on the left hand side, up to precision. Compare this to the right hand side.

[0]:

Now let C be the hyperelliptic curve

$$C\colon y^2 = x^5 - x + 1$$

Repeat the process above for $p = 3$. (But don't make the power series precision too high! Point counting gets very costly over finite fields of large degree). What is the polynomial appearing in the numerator of the resulting rational function?

[0]:

Extension: Can you explain the quantities appearing in this polynomial in terms of the hyperelliptic curve, as in the elliptic curve case?

[0]:

Extension: By considering the Frobenius automorphism on E, prove the formula given in the question.

[0]: