On computing the Jacobian of quotient modular curves of $X_0(N)$ and its $\mathbb{F}_{p^n}$-points

Francesc Bars*

1 Example on computing Jacobian decomposition and points over finite fields for a quotient modular curve $X_0(N)$.

Here, we introduce a Magma ad-hoc code in order to obtain the modular forms that could appear to $J_0(N)/W_N$ and an approach to determine m_{f_i} for each modular forms that appears when N is a product of three primes and $|W_N| = 4$, i.e. the Jacobian decomposition for $X_0(N)/W_N$. The m_{f_i} computed in the Magma programme is bellow is a upper bound for m_{f_i}, and after comparing with the genus of $X_0(N)/W_N$ we can decide if is the correct one or not. If we obtain a strict bound, thus, we study the situation by the action of Atkin-Lehner involutions when the modular form appears new and we count by Lema 2.2 and Proposition 2.2 (i)(ii) of [BG20] the correct power for such factors.

We show this with the example of $N = 308$ and $|W_N| = 4$.

Moreover to compute the possible Jacobian for $X_0(N)/W_N$, choice $p \nmid N$ and count the points $X_0(N)/W_N(\mathbb{F}_{p^n})|$ under the assumption that the Jacobian is well-computed, and introduce a_p the p-th Fourier coeficient for a modular form one dimensional over $\mathbb{Q}$, i.e. E an elliptic curve over $\mathbb{Q}$, in order to compute the points $2 \cdot |E(\mathbb{F}_{p^n})|$ to compare with the $\mathbb{F}_{p^n}$-points of $X_0(N)/W_N$.

Finally appears the dimension of $J_0(N)/W_N$ and if coincides with the genus of $X_0(N)/W_N$ all is fine and we obtained the modular forms that appears in the decomposition of the Jacobian (and we can compare if the elliptic curve could be an elliptic quotient or not).

In the case that such last number is bigger than the genus, we need an ad-hoc modification, controlling the Atkin-Lehner involution in the modular forms presented in order to obtain the exact Jacobian decomposition before computing $\mathbb{F}_{p^n}$-points.

We present for $N = 308$ one example that when we run all is working good because the last number coincides with the genus, and a second example, where we need to modify ad-hoc the programme taking account of the Atkin-Lehner involution when the modular form appears new, in order to control m_f.

All computations done by Magma Online Computation, V2.26-1.

```
L:=[* *];F:=[* *];Level:=[* *];  N:=308; Factorization(N);N;

Nd:=Divisors(N);
```

*First author is supported by MTM2016-75980-P and MDM-2014-0445.

```
N:=308;

m1:=4;m2:=7;m3:=11;

N1:=[*m1,m2,m1*m2*]; N2:=[*m1,m3,m1*m3*]; N3:=[*m2,m3,m2*m3*];
N4:=[*m1,m2*m3,m1*m2*m3*]; N5:=[*m2,m1*m3,m1*m2*m3*];
N6:=[*m3,m1*m2,m1*m2*m3*]; N7:=[*m1*m2,m2*m3,m1*m3*];




H:=[*N1,N2,N3,N4,N5,N6,N7*];



for subgroup in [1..1] do

  subgroup;

  Hh:=H[subgroup];

AtkinLehnerfix:=Hh; Involutions:=#AtkinLehnerfix;

countergenus:=0;

for j in Nd do

MS:=NewformDecomposition(CuspidalSubspace(ModularSymbols(j,2,1)));

      m:=#MS;

      M:=PrimeDivisors(j);

      Nr:=Numerator(N/j);

      divi:=GCD(j,Nr);

      jj:=Numerator(j/divi);

      Mm:=PrimeDivisors(jj);

      Nn:=Divisors(jj);

      mm:=#Mm;

      mn:=#Nn;
```

```
D:=Factorization(jj);

  for i in [1..m] do

     f:=Eigenform(MS[i],30);

     f2:=MS[i];

     K:=Parent(Coefficient(f,3)); d:=Dimension(MS[i]);

     X:=IdentityMatrix(Rationals(), d);

     u:=0;




         for jo in [1..Involutions] do

           dd:=GCD(j,AtkinLehnerfix[jo]);

          if dd eq AtkinLehnerfix[jo] then

            Y:=AtkinLehner(MS[i],dd);

             if Y eq X then
            u;
             else

             u:=1;

             end if;

          else

             if dd eq 1 then

             else

             end if;

          end if;

        end for;
```

```
          if u eq 0 then

             if GCD(m1,j) eq m1 then

         AtkinLehner(f2,m1);m1;

             end if;

             if GCD(m2,j) eq m2 then

          AtkinLehner(f2,m2);m2;

             end if;
 if GCD(m3,j) eq m3 then

          AtkinLehner(f2,m3);m3;

             end if;



          L:=Append(L,f);

          f;

          F:=Append(F,K);

uu:=#Basis(K);

countergenus:=countergenus+uu;

          Level:=Append(Level,j);

                else

                 end if;









      end for;

  end for;
```

```
L;F; p:=11;Level;
 felm:=# F;


C:=ComplexField(100); R<x>:=PolynomialRing(C); pj:=0*x+1; Roo:=[*

*]; for j in [1 .. felm] do

   if Degree(F[j]) eq 1 then

     cc:=Roots(x^2-Coefficient(L[j],p)*x+p,C);

     Roo:=Append(Roo,cc);

     pj:=pj*(x^2-Coefficient(L[j],p)*x+p);

   else

    dd:=Degree(F[j]);

    u:=Roots(DefiningPolynomial(F[j]),C); uu:= # u;

       for m in [1 .. uu] do

       f := hom< F[j] -> C | u[m][1]>;

       cc2:=Roots(x^2-f(Coefficient(L[j],p))*x+p,C);

       Roo:=Append(Roo,cc2);

       pj:=pj*(x^2-f(Coefficient(L[j],p))*x+p);

       end for;

   end if;

end for; pjdegree:=Degree(pj); pjdegree; PR:=[* *];

  d2:=Degree(pj);

long:= # Roo;
```

```
for nn in [1 .. 20] do s:=0;
   for i in [1 .. long] do
     for j in [1..2] do
      if Roo[i][j][2] gt 0 then
s:=s+(Roo[i][j][2])*(Roo[i][j][1])^(nn) ;
   else
   s:=s;
   end if;
end for; end for;
  a:=Round(1+p^(nn)-s); PR:=Append(PR,a); end for;
Jj:=[* *]; for aaa in [1..20] do
  ss:=0;
   adiv:=Divisors(aaa);
     for kk in adiv do
        vv:=aaa/kk;vv:=Numerator(vv);
       ss:=ss+(MoebiusMu(vv))*(PR[kk]);
      end for;

   vvv:=ss/aaa;
   Rr:=Integers(2); bb:=Rr!vvv;
   Jj:=Append(Jj,bb);
end for; jjel:=# Jj; ssum:=0; var:=0; for t in [1..jjel] do
   if Jj[t] eq 1 then
```

```
      tred:=Rr!t;
      if tred eq 1 then
      ssum:=ssum+t;
      var:=t;
      else
      ssum:=ssum;
      end if;
   else
      ssum:=ssum;
   end if;
end for;

PR2:=[* *]; a3:=0; cearrels:=Roots(x^2-a3*x+p,C);

for i in [1..20] do

b:=2*(p^i+1-Round(cearrels[1][1]^i+ p^i/cearrels[1][1]^i));
PR2:=Append(PR2,b); end for;

PR;PR2;

L:=[**]; F:=[**]; Level:=[**]; end for;
countergenus;
```

And the result is given by

```
[ <2, 2>, <7, 1>, <11, 1> ] 308 1 [-1] 11 q - 2*q^2 - q^3 + 2*q^4 +
```

```
q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 - 2*q^12 +
    4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 + 2*q^21 - 2*q^22
    - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30)
[-1  0] [ 0 -1] 11 q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 -
2*q^9 - 2*q^10 + q^11 - 2*q^12 +
    4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 + 2*q^21 - 2*q^22
    - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30)
0 [1] 7 [1] 11 q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12
- 4*q^13 + 3*q^15 + 4*q^16
    + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 + 2*q^28 +
    10*q^29 + O(q^30)
0 [1] 7 [-1] 11 q + q^2 + 2*q^3 - q^4 - 2*q^5 + 2*q^6 - q^7 - 3*q^8
+ q^9 - 2*q^10 + q^11 -
    2*q^12 + 4*q^13 - q^14 - 4*q^15 - q^16 + 4*q^17 + q^18 + 2*q^20 - 2*q^21 +
    q^22 - 4*q^23 - 6*q^24 - q^25 + 4*q^26 - 4*q^27 + q^28 - 6*q^29 + O(q^30)
0 [1] 7 [1] 11 q - q^2 + q^4 - 4*q^5 - q^7 - q^8 - 3*q^9 + 4*q^10 -
q^11 + 2*q^13 + q^14 + q^16
    - 4*q^17 + 3*q^18 - 6*q^19 - 4*q^20 + q^22 + 4*q^23 + 11*q^25 - 2*q^26 -
    q^28 - 2*q^29 + O(q^30)
0 [1] 7 [-1] 11 q - q^2 + 2*q^3 + q^4 + 2*q^5 - 2*q^6 - q^7 - q^8 +
q^9 - 2*q^10 + q^11 + 2*q^12
    - 4*q^13 + q^14 + 4*q^15 + q^16 - q^18 + 4*q^19 + 2*q^20 - 2*q^21 - q^22 +
    4*q^23 - 2*q^24 - q^25 + 4*q^26 - 4*q^27 - q^28 + 2*q^29 + O(q^30)
0 [1] 7 [1] 11 q + q^2 + q^4 + 2*q^5 - q^7 + q^8 - 3*q^9 + 2*q^10 -
q^11 + 2*q^13 - q^14 + q^16
    + 2*q^17 - 3*q^18 + 2*q^20 - q^22 - 8*q^23 - q^25 + 2*q^26 - q^28 - 2*q^29 +
    O(q^30)
0 [1 0] [0 1] 7 [1 0] [0 1] 11 q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9
- q^11 + 6*q^12 - 4*q^13 + 3*q^15 + 4*q^16
    + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 + 2*q^28 +
    10*q^29 + O(q^30)
0 [1 0] [0 1] 7 [-1  0] [ 0 -1] 11 q + q^2 + 2*q^3 - q^4 - 2*q^5 +
2*q^6 - q^7 - 3*q^8 + q^9 - 2*q^10 + q^11 -
    2*q^12 + 4*q^13 - q^14 - 4*q^15 - q^16 + 4*q^17 + q^18 + 2*q^20 - 2*q^21 +
    q^22 - 4*q^23 - 6*q^24 - q^25 + 4*q^26 - 4*q^27 + q^28 - 6*q^29 + O(q^30)
1 1 1 1 1 1 1 1 [*
    q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 -
        2*q^12 + 4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 +
        2*q^21 - 2*q^22 - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30),
    q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 -
        2*q^12 + 4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 +
        2*q^21 - 2*q^22 - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30),
    q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12 - 4*q^13 + 3*q^15 +
        4*q^16 + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 +
        2*q^28 + 10*q^29 + O(q^30),
    q + q^2 + 2*q^3 - q^4 - 2*q^5 + 2*q^6 - q^7 - 3*q^8 + q^9 - 2*q^10 + q^11 -
        2*q^12 + 4*q^13 - q^14 - 4*q^15 - q^16 + 4*q^17 + q^18 + 2*q^20 - 2*q^21
        + q^22 - 4*q^23 - 6*q^24 - q^25 + 4*q^26 - 4*q^27 + q^28 - 6*q^29 +
        O(q^30),
```

```
    q - q^2 + q^4 - 4*q^5 - q^7 - q^8 - 3*q^9 + 4*q^10 - q^11 + 2*q^13 + q^14 +
        q^16 - 4*q^17 + 3*q^18 - 6*q^19 - 4*q^20 + q^22 + 4*q^23 + 11*q^25 -
        2*q^26 - q^28 - 2*q^29 + O(q^30),
    q - q^2 + 2*q^3 + q^4 + 2*q^5 - 2*q^6 - q^7 - q^8 + q^9 - 2*q^10 + q^11 +
        2*q^12 - 4*q^13 + q^14 + 4*q^15 + q^16 - q^18 + 4*q^19 + 2*q^20 - 2*q^21
        - q^22 + 4*q^23 - 2*q^24 - q^25 + 4*q^26 - 4*q^27 - q^28 + 2*q^29 +
        O(q^30),
    q + q^2 + q^4 + 2*q^5 - q^7 + q^8 - 3*q^9 + 2*q^10 - q^11 + 2*q^13 - q^14 +
        q^16 + 2*q^17 - 3*q^18 + 2*q^20 - q^22 - 8*q^23 - q^25 + 2*q^26 - q^28 -
        2*q^29 + O(q^30),
    q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12 - 4*q^13 + 3*q^15 +
        4*q^16 + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 +
        2*q^28 + 10*q^29 + O(q^30),
    q + q^2 + 2*q^3 - q^4 - 2*q^5 + 2*q^6 - q^7 - 3*q^8 + q^9 - 2*q^10 + q^11 -
        2*q^12 + 4*q^13 - q^14 - 4*q^15 - q^16 + 4*q^17 + q^18 + 2*q^20 - 2*q^21
        + q^22 - 4*q^23 - 6*q^24 - q^25 + 4*q^26 - 4*q^27 + q^28 - 6*q^29 +
        O(q^30)
*]
[*
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field
*]
[* 11, 22, 77, 77, 154, 154, 154, 154, 154 *]

18


[* 11, 311, 1364, 12851, 160501, 1786304, 19494871, 214266011,
2357852684, 25937591111, 285312734201, 3138436117424,
34522701304531, 379749650880911, 4177248268348804,
45949732763696171, 505447027733265101, 5559917274696609344,
61159090452530236991, 672749995396353069011 *] [* 24, 288, 2664,
28800, 322104, 3548448, 38974344, 428659200, 4715895384,
51875493408, 570623341224, 6276849667200, 69045424287864,
759499745115168, 8354496338831304, 91899458869708800,
1010894056998587544, 11119834636416253728, 122318180896829092584,
1345499989761370320000 *]

9
```

Now we know that genus of $X_0(308)/W1$ has genus 9, the programme compute all correctly, respect to the Jacobian which we list the modular forms appears (also we repeat them), the

field where aare defined, the level that appear (here be aware, for old forms repeated the level is a multiple of the newform and appears repeated. the 18 is to use to compute if we can use the criteria to discard no automorphism, and after the $\mathbb{F}_{p^n}$ points of $X_0(N)/W_N$ and 2 times the points of an elliptic curve with a_p (named a_p) the one choice inside the programme.

BUT CAN HAPPEN THAT NEED A MODIFICATION IN SOME SITUATIONS, CONSIDER $W7$ INSTEAD OF $W1$ then, the above programme is only modified in the first lines by

```
L:=[* *];F:=[* *];Level:=[* *];  N:=308; Factorization(N);N;

Nd:=Divisors(N);


N:=308;

m1:=4;m2:=7;m3:=11;

N1:=[*m1,m2,m1*m2*]; N2:=[*m1,m3,m1*m3*]; N3:=[*m2,m3,m2*m3*];
N4:=[*m1,m2*m3,m1*m2*m3*]; N5:=[*m2,m1*m3,m1*m2*m3*];
N6:=[*m3,m1*m2,m1*m2*m3*]; N7:=[*m1*m2,m2*m3,m1*m3*];




H:=[*N1,N2,N3,N4,N5,N6,N7*];



for subgroup in [7..7] do

NOW THE SAME CODE AS BEFORE, HERE WE STUDY THE QUOTIENT BY W7
```

The modular curve $X_0(308)/W7$ has genus 11 and the answer is

```
[ <2, 2>, <7, 1>, <11, 1> ] 308 7 [-1] 11 q - 2*q^2 - q^3 + 2*q^4 +
q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 - 2*q^12 +
    4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 + 2*q^21 - 2*q^22
    - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30)
[-1] 7 q - q^2 - 2*q^3 + q^4 + 2*q^6 + q^7 - q^8 + q^9 - 2*q^12 -
4*q^13 - q^14 + q^16
    + 6*q^17 - q^18 + 2*q^19 - 2*q^21 + 2*q^24 - 5*q^25 + 4*q^26 + 4*q^27 + q^28
    - 6*q^29 + O(q^30)
[-1  0] [ 0 -1] 11 q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 -
2*q^9 - 2*q^10 + q^11 - 2*q^12 +
    4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 + 2*q^21 - 2*q^22
    - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30)
0 [1] 7 [1] 11 q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12
- 4*q^13 + 3*q^15 + 4*q^16
    + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 + 2*q^28 +
```

```
    10*q^29 + O(q^30)
0 [1] 7 [1] 11 q - q^2 + q^4 - 4*q^5 - q^7 - q^8 - 3*q^9 + 4*q^10 -
q^11 + 2*q^13 + q^14 + q^16
    - 4*q^17 + 3*q^18 - 6*q^19 - 4*q^20 + q^22 + 4*q^23 + 11*q^25 - 2*q^26 -
    q^28 - 2*q^29 + O(q^30)
0 [1] 7 [1] 11 q + q^2 + q^4 + 2*q^5 - q^7 + q^8 - 3*q^9 + 2*q^10 -
q^11 + 2*q^13 - q^14 + q^16
    + 2*q^17 - 3*q^18 + 2*q^20 - q^22 - 8*q^23 - q^25 + 2*q^26 - q^28 - 2*q^29 +
    O(q^30)
0 [-1  0] [ 0 -1] 7 [-1  0] [ 0 -1] 11 q + q^2 + a*q^3 + q^4 - a*q^5
+ a*q^6 + q^7 + q^8 + (-2*a + 1)*q^9 - a*q^10 +
    q^11 + a*q^12 + (-a - 2)*q^13 + q^14 + (2*a - 4)*q^15 + q^16 + 2*a*q^17 +
    (-2*a + 1)*q^18 + (-a - 6)*q^19 - a*q^20 + a*q^21 + q^22 + 4*q^23 + a*q^24 +
    (-2*a - 1)*q^25 + (-a - 2)*q^26 + (2*a - 8)*q^27 + q^28 + (2*a + 2)*q^29 +
    O(q^30)
0 [1 0] [0 1] 7 [1 0] [0 1] 11 q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9
- q^11 + 6*q^12 - 4*q^13 + 3*q^15 + 4*q^16
    + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 + 2*q^28 +
    10*q^29 + O(q^30)
1 1 0 0 0 [-1  0  0] [ 0 -1  0] [ 0  0 -1] 4 [-1  0  0] [ 0 -1  0] [
0  0 -1] 7 [-1  0  0] [ 0 -1  0] [ 0  0 -1] 11 q + a*q^3 + (-a^2 +
4)*q^5 + q^7 + (a^2 - 3)*q^9 + q^11 + (a^2 + a)*q^13 + (a^2
    - 2*a - 2)*q^15 + (-a^2 - 3*a + 4)*q^17 + 2*a*q^19 + a*q^21 + (a^2 + 2*a -
    6)*q^23 + (-a^2 - 4*a + 9)*q^25 + (-a^2 + 2)*q^27 + (-2*a^2 - 2*a + 10)*q^29
    + O(q^30)
1 1 1 1

[*
    q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 -
        2*q^12 + 4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 +
        2*q^21 - 2*q^22 - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30),
    q - q^2 - 2*q^3 + q^4 + 2*q^6 + q^7 - q^8 + q^9 - 2*q^12 - 4*q^13 - q^14 +
        q^16 + 6*q^17 - q^18 + 2*q^19 - 2*q^21 + 2*q^24 - 5*q^25 + 4*q^26 +
        4*q^27 + q^28 - 6*q^29 + O(q^30),
    q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 -
        2*q^12 + 4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + 2*q^20 +
        2*q^21 - 2*q^22 - q^23 - 4*q^25 - 8*q^26 + 5*q^27 - 4*q^28 + O(q^30),
    q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12 - 4*q^13 + 3*q^15 +
        4*q^16 + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 +
        2*q^28 + 10*q^29 + O(q^30),
    q - q^2 + q^4 - 4*q^5 - q^7 - q^8 - 3*q^9 + 4*q^10 - q^11 + 2*q^13 + q^14 +
        q^16 - 4*q^17 + 3*q^18 - 6*q^19 - 4*q^20 + q^22 + 4*q^23 + 11*q^25 -
        2*q^26 - q^28 - 2*q^29 + O(q^30),
    q + q^2 + q^4 + 2*q^5 - q^7 + q^8 - 3*q^9 + 2*q^10 - q^11 + 2*q^13 - q^14 +
        q^16 + 2*q^17 - 3*q^18 + 2*q^20 - q^22 - 8*q^23 - q^25 + 2*q^26 - q^28 -
        2*q^29 + O(q^30),
    q + q^2 + a*q^3 + q^4 - a*q^5 + a*q^6 + q^7 + q^8 + (-2*a + 1)*q^9 - a*q^10
        + q^11 + a*q^12 + (-a - 2)*q^13 + q^14 + (2*a - 4)*q^15 + q^16 +
        2*a*q^17 + (-2*a + 1)*q^18 + (-a - 6)*q^19 - a*q^20 + a*q^21 + q^22 +
```

```
        4*q^23 + a*q^24 + (-2*a - 1)*q^25 + (-a - 2)*q^26 + (2*a - 8)*q^27 +
        q^28 + (2*a + 2)*q^29 + O(q^30),
    q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12 - 4*q^13 + 3*q^15 +
        4*q^16 + 2*q^17 - 6*q^19 + 2*q^20 + 3*q^21 - 5*q^23 - 4*q^25 - 9*q^27 +
        2*q^28 + 10*q^29 + O(q^30),
    q + a*q^3 + (-a^2 + 4)*q^5 + q^7 + (a^2 - 3)*q^9 + q^11 + (a^2 + a)*q^13 +
        (a^2 - 2*a - 2)*q^15 + (-a^2 - 3*a + 4)*q^17 + 2*a*q^19 + a*q^21 + (a^2
        + 2*a - 6)*q^23 + (-a^2 - 4*a + 9)*q^25 + (-a^2 + 2)*q^27 + (-2*a^2 -
        2*a + 10)*q^29 + O(q^30)
*]
[*
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Rational Field,
    Number Field with defining polynomial x^2 + 2*x - 4 over the Rational Field,
    Rational Field,
    Number Field with defining polynomial x^3 + x^2 - 6*x - 2 over the Rational
    Field
*]
[* 11, 14, 22, 77, 154, 154, 154, 154, 308 *]


24


[* 9, 375, 1428, 12211, 159399, 1792242, 19510269, 214216091,
2357662668, 25937950215, 285314861379, 3138434294458,
34522679625729, 379749649254735, 4177248466215108,
45949732979450411, 505447026201207759, 5559917270791255362,
61159090460761618389, 672749995447543344211 *]


[* 24, 288, 2664, 28800, 322104, 3548448, 38974344, 428659200,
4715895384, 51875493408, 570623341224, 6276849667200,
69045424287864, 759499745115168, 8354496338831304,
91899458869708800, 1010894056998587544, 11119834636416253728,
122318180896829092584, 1345499989761370320000 *]


12
```

Because of 12, we observe that the Jacobian is wrong computed, some m_f is wrong computed. This is the case that we need to add ad-hoc for the Jacobian in order to erase the modular curve that is repeated too much times in the decomposition, as we explained in detail

bellow:

2 An Example of computing the Jacobian of $X_0(N)/W_N$

Let us introduce an example and refer to a small Magma programme to compute such Jacobian decomposition (which is enough for all our levels of the paper under review, we are interested in levels that are a product of two or three primes): consider ($N = 308 = 2^2 \cdot 7 \cdot 11, \mathcal{W} := \langle w_{28}, w_{44} \rangle$) of genus 11. From the $\mathbb{Q}$-decomposition of $J_0(308)$ and the assumption $f \in \mathrm{New}_M$ with $M|N$ and for all $d||M$ with $w_d \in W_N$ $f \in \mathrm{New}_N^{w_d}$ we obtain to consider the following modular forms (that we can control Atkin-Lehner involutions for the power of primes that divides M):

```
f1=q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11 -
       2*q^12 + 4*q^13 + 4*q^14 - q^15 - 4*q^16 - 2*q^17 + 4*q^18 + O(q^20)
```

corresponds to $E11a$, now $w_{11} = -1$. We can lift to level 28, we need to leaf with $w_4 = -1$ because have to be fix by w_{44}, thus by the Proposition above, and there is only one possible lift, because we can lift by 2 with ± 1 but to obtain -1 is only a possibility. Now to lift from level 28 to 308 we need to lift by 7, and we need that lift with $w_7 = -1$ in order to obtain that w_{28} is fixed (because $w_4 = -1$ in level 28). Therefore $m_{f_1} = 1$.

```
f2:=q - q^2 - 2*q^3 + q^4 + 2*q^6 + q^7 - q^8 + q^9 - 2*q^12 -
4*q^13 - q^14 +  q^16 + 6*q^17 - q^18 + 2*q^19 + O(q^20),
```

corresponds to $E14a$ with $w_2 = +1$ and $w_7 = -1$ at level 14. We need to lift to 308. Lift first to $14 * 11 = 154$, thus by previous lemma and Proposition we only need to lift by one modular form that we choice the one with $w_{11} = -1$ because we want be fixed by w_{77} at level 154. To reach level 308 need to leaf by 2 thus only one possibility to lift with negative at two (following Lemma) because we want $w_4 = -1$. Thus such lift satisfies $w_4 = -1$, $w_7 = -1$ and $w_{11} = -1$ to become fix by $\mathcal{W}$, thus $m_{f_2} = 1$.

```
f3:=q - 3*q^3 - 2*q^4 - q^5 - q^7 + 6*q^9 - q^11 + 6*q^12 - 4*q^13 +
3*q^15 + 4*q^16 + 2*q^17 - 6*q^19 + O(q^20),
```

corresponds to $E77a$ with $w_7 = 1$ and $w_{11} = 1$ thus to lift by 4 and reach level $77 \cdot 4 = 308$, we have two posibilities lift by + in step by step (at prime two) or lift by $-$ and $-$ in each step by step (at prime two). This gives two lifts of f_3 in level 308, thus $m_{f_3} = 2$.

```
f4:=q - q^2 + q^4 - 4*q^5 - q^7 - q^8 - 3*q^9 + 4*q^10 - q^11 +
2*q^13 + q^14 + q^16 - 4*q^17 + 3*q^18 - 6*q^19 + O(q^20);
```

is $E154a$ with $w_2 = w_7 = w_{11} = 1$ to lift by two to level 308, where we need $w_4 = 1$ we have only one possibility by act by $+1$ by previous Lemma and Proposition, thus $m_{f_4} = 1$.

```
f5:=q + q^2 + q^4 + 2*q^5 - q^7 + q^8 - 3*q^9 + 2*q^10 - q^11 +
2*q^13 - q^14 + q^16 + 2*q^17 - 3*q^18 +  O(q^20),
```

is $E154b$ with $w_2 = -1$ and $w_7 = w_{11} = 1$ to lift by two to level 308, where we need $w_4 = 1$ we have only one possibility by act by -1 at prime two to lift by previous Lemma and Proposition, thus $m_{f_5} = 1$.

```
f6:=q + q^2 + a*q^3 + q^4 - a*q^5 + a*q^6 + q^7 + q^8 + (-2*a +
1)*q^9 - a*q^10 + q^11 + a*q^12 + (-a - 2)*q^13 + q^14 + (2*a -
4)*q^15 + q^16 + 2*a*q^17 + (-2*a + 1)*q^18 + (-a - 6)*q^19
+O(q^20),
```

of level 154, with $w_7 = w_{11} = I_2$, thus we have to up to level 308 only one way that can be computed if we are interested by asking the action of AL-involution w_2 is this modular form. Here is unnecessary and $m_{f_6} = 1$. We observe that a is a root of $x^2 + 2x - 4$ thus A_{f_6} has dimension two. Finally,

```
f7:=q + a*q^3 + (-a^2 + 4)*q^5 + q^7 + (a^2 - 3)*q^9 + q^11 + (a^2 +
a)*q^13 + (a^2 - 2*a - 2)*q^15 + (-a^2 - 3*a + 4)*q^17 + 2*a*q^19 +
O(q^20)
```

of level 354 that is fix by $\mathcal{W}$, where a a root of $x^3 + x^2 - 6x - 2$, thus $dim(A_{f_7}) = 3$. Thus we obtain the Jacobian decomposition of dimension 11 corresponding to the genus.

And easily we make an ad-hoc modification in the Magma programme to compute well the number of $\mathbb{F}_{p^n}$-points.

3 A Magma code for a product of two primes

For exactly two primes dividing N the Magma code is similar, we use

```
L:=[* *];F:=[* *];Level:=[* *];  N:=153; Factorization(N);N;

Nd:=Divisors(N);



m1:=9;m2:=17; N1:=[*m1*]; N2:=[*m2*];  N3:=[*m1*m2*];


H:=[*N1,N2,N3*];




for subgroup in [3..3] do

  subgroup;

  Hh:=H[subgroup];

AtkinLehnerfix:=Hh; Involutions:=#AtkinLehnerfix;

countergenus:=0;

for j in Nd do

MS:=NewformDecomposition(CuspidalSubspace(ModularSymbols(j,2,1)));

      m:=#MS;
```

```
M:=PrimeDivisors(j);

Nr:=Numerator(N/j);

divi:=GCD(j,Nr);

jj:=Numerator(j/divi);

Mm:=PrimeDivisors(jj);

Nn:=Divisors(jj);

mm:=#Mm;

mn:=#Nn;

D:=Factorization(jj);

  for i in [1..m] do

     f:=Eigenform(MS[i],30);

     f2:=MS[i];

     K:=Parent(Coefficient(f,3)); d:=Dimension(MS[i]);

     X:=IdentityMatrix(Rationals(), d);

     u:=0;




          for jo in [1..Involutions] do

            dd:=GCD(j,AtkinLehnerfix[jo]);

           if dd eq AtkinLehnerfix[jo] then

             Y:=AtkinLehner(MS[i],dd);

              if Y eq X then
             u;
              else

              u:=1;

              end if;
```

```
                else
                   if dd eq 1 then
                   else
                   end if;
                end if;
              end for;



          if u eq 0 then
             if GCD(m1,j) eq m1 then
         AtkinLehner(f2,m1);m1;
             end if;
             if GCD(m2,j) eq m2 then
          AtkinLehner(f2,m2);m2;
             end if;

          L:=Append(L,f);
          f;
          F:=Append(F,K);
uu:=#Basis(K);
countergenus:=countergenus+uu;
          Level:=Append(Level,j);
                else
                 end if;
```

```
      end for;
  end for;
```

```
L;F;Level; p:=7;
 felm:=# F;

C:=ComplexField(100); R<x>:=PolynomialRing(C); pj:=0*x+1; Roo:=[*
*]; for j in [1 .. felm] do
   if Degree(F[j]) eq 1 then
     cc:=Roots(x^2-Coefficient(L[j],p)*x+p,C);
     Roo:=Append(Roo,cc);
     pj:=pj*(x^2-Coefficient(L[j],p)*x+p);
   else
    dd:=Degree(F[j]);
    u:=Roots(DefiningPolynomial(F[j]),C); uu:= # u;
       for m in [1 .. uu] do
       f := hom< F[j] -> C | u[m][1]>;
       cc2:=Roots(x^2-f(Coefficient(L[j],p))*x+p,C);
       Roo:=Append(Roo,cc2);
       pj:=pj*(x^2-f(Coefficient(L[j],p))*x+p);
```

```
       end for;
   end if;
end for; pjdegree:=Degree(pj); pjdegree; PR:=[* *];
  d2:=Degree(pj);
long:= # Roo;



for nn in [1 .. 20] do s:=0;
   for i in [1 .. long] do
     for j in [1..2] do
      if Roo[i][j][2] gt 0 then
s:=s+(Roo[i][j][2])*(Roo[i][j][1])^(nn) ;
   else
   s:=s;
   end if;
end for; end for;
  a:=Round(1+p^(nn)-s); PR:=Append(PR,a); end for;
Jj:=[* *]; for aaa in [1..20] do
  ss:=0;
   adiv:=Divisors(aaa);
     for kk in adiv do
        vv:=aaa/kk;vv:=Numerator(vv);
       ss:=ss+(MoebiusMu(vv))*(PR[kk]);
      end for;
```

```
   vvv:=ss/aaa;
   Rr:=Integers(2); bb:=Rr!vvv;
   Jj:=Append(Jj,bb);
end for; jjel:=# Jj; ssum:=0; var:=0; for t in [1..jjel] do
   if Jj[t] eq 1 then
      tred:=Rr!t;
      if tred eq 1 then
      ssum:=ssum+t;
      var:=t;
      else
      ssum:=ssum;
      end if;
   else
      ssum:=ssum;
   end if;
end for;

PR2:=[* *]; a3:=4; cearrels:=Roots(x^2-a3*x+p,C);

for i in [1..20] do

b:=2*(p^i+1-Round(cearrels[1][1]^i+ p^i/cearrels[1][1]^i));
PR2:=Append(PR2,b); end for;
```

```
PR;PR2;



L:=[**]; F:=[**]; Level:=[**]; end for;

countergenus;
```

References

[BG20] Francesc Bars and Josep González. Bielliptic modular curves $X_0^*(N)$. *J. Algebra*, 559:726–759, 2020.

Francesc Bars Cortina
Departament Matemàtiques, Edif. C, Universitat Autònoma de Barcelona
08193 Bellaterra, Catalonia
francesc@mat.uab.cat