# WORKED EXAMPLE FOR MY PAPER: COMPUTING INTEGRAL POINTS ON GENUS 2 HYPERELLIPTIC CURVES ESTIMATING HYPERELLIPTIC LOGARITHMS

HOMERO R. GALLEGOS–RUIZ

ABSTRACT. We completely determine the set of integral points on the rank 5, genus 2 curve $y^2 = x^5 + 105x^4 + 4405x^3 + 92295x^2 + 965794x + 4038280$ by estimating hyperelliptic logarithms.

This article explains the MAGMA commands and computations related to the example in my paper *Computing integral points on genus 2 hyperelliptic curves estimating hyperelliptic logarithms*.

Let $C$ be the hyperelliptic curve defined by

$$f(x) = x^5 + 105x^4 + 4405x^3 + 92295x^2 + 965794x + 4038280.$$

This polynomial equals $(x + 19)(x + 20)(x + 21)(x + 22)(x + 23) + (5 \cdot 4)^2$, which according to [6] should have rank at least 4.

In this document we perform all computations to find the integral points on $C$ with expanded commentaries.

We define the curve in MAGMA [1]:

```
> Q<x>:=PolynomialRing(Rationals());
> f:=x^5 + 105*x^4 + 4405*x^3 + 92295*x^2 + 965794*x + 4038280;
> a5:=LeadingCoefficient(f);
> C:=HyperellipticCurve(f);
```

We now define the Jacobian for the curve and compute bounds for its rank (unconditionally):

```
> J:=Jacobian(C);
> r:=RankBounds(J);
> r;
5
```

The output is a lower bound for the rank, followed by an upper bound. In our case they coincide, so we know the rank is 5. Since the rank is 5 and the genus is 2, Chabauty's method does not apply and the variant of the Mordell–Weil sieve described in [2] would be computationally expensive.

We can compute the torsion subgroup of the Jacobian.

```
> T:=TorsionSubgroup(J);
> T;
Abelian Group of order 1
> t:=Order(T);
```

We conclude that the Jacobian is torsion free.

We now look for points on the Jacobian generating a rank 5 subgroup of the Mordell–Weil group.

```
> bas := ReducedBasis(Points(J : Bound :=1000));
> bas;
[ (x + 19, 20, 1), (x + 23, 20, 1), (x + 21, 20, 1),
   (x + 22, -20, 1), (x^2 - 9*x - 580, -360*x - 7220, 2) ]
```

The command ReducedBasis returns an LLL-reduced basis for the subgroup of $J(C)(\mathbb{Q})$ generated by a sequence of points. Here the output consists of 5 divisors, so we now know a full rank subgroup of $J(C)(\mathbb{Q})$. We will prove in the last section that this is a set of generators for the full group $J(C)(\mathbb{Q})$.

The notation for divisors follows Mumford's representation. It is a sequence of two polynomials and an integer. The roots of the first polynomial are the $x$ coordinates of the points on the divisor. The $y$ coordinates for the points are obtained from the second polynomial in terms of $x$. The integer gives the degree of the divisor.

## 1. Upper bound for the size of the integral points

We now want to compute an upper bound for the size of the integral points. We compute it from Theorem 8.1 in [5]. We may need to scale the model of our curve to obtain a model of the form

$$ay^2 = x^5 + b_4x^4 + \cdots + b_0.$$

In our case we only need to set $a = 1$.

```
> a:=1;
```

We need a full set of representatives for $J(C)(\mathbb{Q})/2J(C)(\mathbb{Q})$. In order to compute them, we represent all numbers from 0 to $2^r - 1$ in base 2 as sequences of $r$ digits, filling with zeroes if necessary. We then will add the divisors corresponding to significative digits.

The command Intseq(n,2) takes as input a decimal number and outputs its binary digits as a sequence. In order to fill with zeroes we compute the digits of $n + 2^r$ and then delete the last digit.

```
> binarycoefficients:=[];
> for i in [0..31] do binarycoefficients:=Append(binarycoefficients,
>      Prune(Intseq(i+32,2))); end for;
```

For an element $l$ of the list we want to compute the sum of the elements on the basis indicated by the digits of $l$. We store the representatives in the list fullSetReps.

```
> fullSetReps:=[];
> for i in [1..32] do
for>   fullSetReps:=Append(fullSetReps,binarycoefficients[i][1]*bas[1]
for>     + binarycoefficients[i][2]*bas[2] + binarycoefficients[i][3]*bas[3]
for>     + binarycoefficients[i][4]*bas[4] + binarycoefficients[i][5]*bas[5]);
for> end for;
```

From this list we will compute the set $\mathcal{K}$ as in [5, Lemma 2.1].

Note that in order to complete the computation of the bound for the height of the integral points we do not need a set of generators for the full Mordell–Weil group, but only for the quotient $J(C)(\mathbb{Q})/2J(C)(\mathbb{Q})$. But for the reduction of the bound we do need a set of generators for $J(C)(\mathbb{Q})$.

Now we proceed to compute the set $\mathcal{K}$. We have to express every element in the sequence fullSetReps in the form indicated in Section 2.1 of [5].

We think on the divisors of $J(C)(\mathbb{Q})$ as degree two effective divisors. Suppose one of our divisors is given in the form

$$(x^2 + ax + b, cx + d, 2), \qquad a, b, c, d \in \mathbb{Q}.$$

The divisor $P_1+P_2$ is stable by the action of Galois. The roots of the first polynomial are the $x$ coordinates of the points on the support of the divisor. If $J(C)(\mathbb{Q})$ has no elements of order 2 defined over $\mathbb{Q}$, the $y$ coordinates for our points have $y(P) \neq 0$. Then all the required conditions for the divisor in Section 2.1 of [5] are satisfied. Suppose we are able to express the $x$ coordinates in the form $\gamma_i/d^2$, with $\gamma_i$ an algebraic integer. Then

$$x^2 + ax + b = (x - \gamma_1/d^2)(x - \gamma_2/d^2) = \frac{(d^2x - \gamma_1)(d^2x - \gamma_2)}{d^4}.$$

Hence, the quantity we should associate to the divisor is equal to $d^4(\alpha^2 + a\alpha + b)$, which modulo squares equals $(\alpha^2 + a\alpha + b)$. In other words, we evaluate the first polynomial on Mumford's representation on $\alpha$ and multiply by an appropriate integer. Note that when MAGMA computes elements in the number field $\mathbb{Q}(\alpha)$, it represents the results on the form

$$1/m \cdot (\text{integer linear combination of powers of } \alpha),$$

with $m$ an integer. Since $\alpha$ is an algebraic integer, the expression inside the parenthesis is an algebraic integer. We want an algebraic integer which modulo squares equals this quantity. MAGMA can compute a squarefree factorization for $m = m_1 m_2^2$. Since $(m_1 m_2)^2/m = m_1^2 m_2^2/m_1 m_2^2 = m_1$, then the following is equivalent modulo squares to the number we want:

$$m_1 \cdot (\text{integer linear combination of powers of } \alpha),$$

which is an algebraic integer. So we multiply $(\alpha^2 + a\alpha + b)$ by $m_1^2 m_2^2$ to obtain the algebraic integer in our set $\mathcal{K}$. We have to repeat the process with every element in our full set of representatives.

```
> K<alpha>:=NumberField(f);
> kappaset:=[];
> for i in fullSetReps do
for>       kappa:=Evaluate(i[1], alpha);
for>       d1,d2:=SquarefreeFactorization(Denominator(kappa));
for>       d:=d1*d2;
for>       kappa:=d^2*i[1];
for>       if i[3] eq 1 then kappa:=a*kappa; end if;
for>       kappaset:=Append(kappaset,kappa);
for> end for;
```

Once we have the set $\mathcal{K}$, for every $\kappa \in \mathcal{K}$ we have to compute the upper bounds for all nonzero integers satisfying $x - \alpha = \kappa\xi^2$ for some algebraic number $\xi$. This is done as in Theorem 8.1 of [5].

```
> load "upperbounds.m";
> S:=[];  // We are dealing with integral points
> Bounds:=[];
> for k in kappaset do;
for>   Bounds:= Append(Bounds,heightxBoundx(f,k,S));
for> end for;
```

```
> upperbound:= Max(Bounds);
```

After this computation we conclude that if $(x, y)$ is an integral point on $C$, then its height is at most:

```
> upperbound;
1.16434883654793772107557036153E546
```

Hence, if $(x, y)$ is an integral point on $C$,

$$|x| \leq \exp(1.17 \times 10^{546}).$$

## 2. Reduction of the bound

Now that we have an upper bound for the size of the integral points, we want to reduce it to manageable proportions according to Section 6 of [4]

We need to translate the bound to an upper bound for the coefficients of the expression

$$P - \infty = n_1 D_1 + \cdots + n_5 D_5 + Q, \tag{1}$$

where the $D_i$s are the elements in `bas`, the generators for the free part of the Mordell–Weil group, and $Q$ is a torsion point. According to Corollary 3.2 from the paper, the maximum for the coefficients on those expressions is given by

$$\sqrt{\mu_2^{-1} \left(\log|a_5| + 2B - \mu_1\right)}$$

where $\mu_1$ is a lower bound for the height difference (which we compute from Theorem 4 in [3]), $\mu_2$ is the least of the eigenvalues of the height pairing matrix, $a_5$ is the leading coefficient of $f$ and $B$ is an upper bound for the size of the integral points on $C$. Since in our case $a_5 = 1$, the term $\log|a_5|$ is 0.

```
> load "heightdifference.m";  // Code for the computation of the lower bound.
> mu1:=HeightDifferenceLowerBound(f);
> mu2:= Min([l[1] : l in Eigenvalues(HeightPairingMatrix(bas))]);
> N:= Sqrt(mu2^(-1)*(2*upperbound-mu1));
> N;
1.93467303142902221285342385764E273
```

We need to estimate how many digits of precision we need to compute the matrix $\mathcal{A}$. According to Proposition 6.2 $K$ must be of the order of $(\frac{1}{5}(48tN\sqrt{r} + 12\sqrt{r}N + 5N + 48))^{(r+4)/4}$, where $r$ is the rank and $t$ the order of the torsion subgroup.

```
> ((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48))^((r+4)/4);
1.39675625460522953100172935313E618
```

We need $K$ to be larger than this. So we need to compute the period matrix and the hyperelliptic logarithms with at least 620 digits of precision. We choose to compute with 1300 digits of precision.

```
> K:=10^970;
> F:=RealField(1300);
> SetDefaultRealField(F);
```

We now compute the analytic Jacobian, and the big period matrix using van Wamelen's implementation [8]. We turn the $2 \times 4$ matrix defined over $\mathbb{C}$ into a $4 \times 4$ matrix defined over $\mathbb{R}$.

```
> time AJ:=AnalyticJacobian(C);
> M:=BigPeriodMatrix(AJ);
> RealM:=Matrix(4, [Re(M[1,1]), Re(M[1,2]), Re(M[1,3]), Re(M[1,4]),
>    Im(M[1,1]), Im(M[1,2]), Im(M[1,3]), Im(M[1,4]),
>    Re(M[2,1]), Re(M[2,2]), Re(M[2,3]), Re(M[2,4]),
>    Im(M[2,1]), Im(M[2,2]), Im(M[2,3]), Im(M[2,4])]);
```

In the computer we were using, the computation took 1032.68 seconds. We prepare to find the matrix $\mathcal{A}_K$ from Proposition 6.2 in the paper. We have to compute the hyperelliptic logarithms of the divisors generating the Mordell–Weil group. The MAGMA function `ToAnalyticJacobian` takes as input a pair of complex numbers which are supposed to be the coordinates of a point on the curve, and outputs the integrals defining the hyperelliptic logarithm. When we have a degree 2 divisor $P_1 + P_2 - 2\infty$, we add the hyperelliptic logarithms of $P_1$ and $P_2$, and then reduce modulo $\Lambda$.

```
> R<x>:=PolynomialRing(ComplexField());
> analyticPts:=[];
> for pt in bas do;
for>    coords := [r[1] : r in Roots(Evaluate(pt[1],x))];
for>    coords := [<d1,Evaluate(pt[2],d1)> : d1 in coords];
for> analyticPts:=Append(analyticPts,coords);
for> end for;
```

Recall that MAGMA represents divisors by a pair of polynomials and an integer. With the previous commands we are computing the roots of the first polynomial to obtain the $x$ coordinates of the points on the support of the divisor, and subsituting them on the second polynomial gives us the $y$ coordinates. Once we have the corresponding points, we compute the logarithms and reduce modulo $\Lambda$. In order to perform the reduction modulo $\Lambda$, we multiply the image of $D_i$ by $M^{-1}$ to get $t_1, t_2, t_3, t_4$ such that

$$\varphi(D_i) = t_1\omega_1 + t_2\omega_2 + t_3\omega_3 + t_4\omega_4$$

and then compute the fractional part of the $t_i$s ($t_i - \lfloor t_i \rfloor$).

```
> hyperlogs:=[];
> time for pt in analyticPts do;
for>    hyperlogpt:= &+[ToAnalyticJacobian(d[1],d[2],AJ): d in pt];
for>    hyperlogpt:=[Re(hyperlogpt[1][1]), Im(hyperlogpt[1][1]),
for>    Re(hyperlogpt[2][1]), Im(hyperlogpt[2][1])];
for>    coordsInOmegas:=Eltseq(RealM^(-1)*Matrix(1,hyperlogpt));
for>    coordsReduced:=[];
for>    for i in coordsInOmegas do;
for|for>      coordsReduced:=Append(coordsReduced, i-Floor(i));
for|for>    end for;
for>    hyperlogpt:=RealM*Matrix(1,coordsReduced);
for>    hyperlogs:=Append(hyperlogs,hyperlogpt);
for> end for;
```

We write block matrices that will help us to produce the matrix $\mathcal{A}$.

```
> B1:=ScalarMatrix(r,1);
> B2:=ZeroMatrix(Integers(),r,4);
> B3:=HorizontalJoin(B1,B2);
> B4:=K*Transpose(Matrix(hyperlogs));
> B5:=K*RealM;
> B6:=HorizontalJoin(B4,B5);
```

The matrix B6 has real coordinates. We want to round its entries to the nearest integer.

```
> D:=Eltseq(B6);
> B7:=[];
> for i in D do; B7:=Append(B7,Round(i)); end for;
> B7:=Matrix(r+4,B7);
> A:=VerticalJoin(B3,B7);
```

We turn the matrix into a lattice in MAGMA, and compute the minimum.

```
> L:=Lattice(Transpose(A));
> l:=Minimum(L);
> assert(l gt ((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48)));
```

The program carries on, so the assertion is true. We compute our new bound.

```
> K1:=2^(7/2)*Exp(-mu1/4)/Sqrt(a5);
> K2:=mu2/4;
> N:=Sqrt((Log(4*K*K1)- Log(l-((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48))))/K2);
> N;
> 101.510616695925694147772537655
```

The previous value for $N$ was $1.94 \cdot 10^{273}$. The improvement is significant, but computing all expressions of the form

$$n_1 D_1 + \cdots + n_5 D_5$$

with $|n_i| \leq 102$ would take a very long time.

After this process we are in a similar situation to that of the beginning of the reduction: we now that if $P$ is an integral point on $C$ with $y(P) > 0$ and $x(P)$ large enough, then

$$|L_i(P)| \leq K_1 \exp(-K_2 M_P^2)$$

and $M_P \leq 102$. The values $K_1$ and $K_2$ remain the same. So we can try to repeat the reduction process and see whether we get a better bound.

```
> K:=10^13;
> B4:=K*Transpose(Matrix(hyperlogs));
> B5:=K*RealM/t;
> B6:=HorizontalJoin(B4,B5);
> D:=Eltseq(B6);
> B7:=[];
> for i in D do; B7:=Append(B7,Round(i)); end for;
> B7:=Matrix(r+4,B7);
> A:=VerticalJoin(B3,B7);
> L:=Lattice(Transpose(A));
> l:=Sqrt(Minimum(L));
> assert(l gt ((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48)));
> N:=Sqrt((Log(4*K*K1)- Log(l-((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48))))/K2);
> N;
13.6733152992874150989475483539
```

Though 13 is a much better value, we can try to reduce it further.

```
> K:=2*10^10;
> B4:=K*Transpose(Matrix(hyperlogs));
> B5:=K*RealM/t;
> B6:=HorizontalJoin(B4,B5);
> D:=Eltseq(B6);
> B7:=[];
> for i in D do; B7:=Append(B7,Round(i)); end for;
> B7:=Matrix(r+4,B7);
> A:=VerticalJoin(B3,B7);
> L:=Lattice(Transpose(A));
> l:=Sqrt(Minimum(L));
> assert(l gt ((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48)));
> N:=Sqrt((Log(4*K*K1)- Log(l-((1/5)*(48*Sqrt(r)*N*t + 12*Sqrt(r)*N + 5*N + 48))))/K2);
12.7253310643914283237744873673
```

After this, trying different values for $K$ did not result on an improvement on the bound.

Now we have to compute all points of the form

$$n_1 D_1 + \cdots + n_5 D_5$$

with $|n_i| \leq 12$. We compute the combinations with MAGMA, and we will keep only those that are represented by a degree 1 divisor, for those are equivalent to a divisor of the form $P - \infty$ with $P$ a rational point on $C$. We will store the rational points obtained in the list pts.

```
> N0:=Floor(N);
> pts:=[];
> print "Computing all points of the form n1 D1+ ... + n5 D5";
> print "with Euclidean norm at most N"; // Takes a long time
> // We only compute those with i1 ge 0, as the others come from negatives.
> time for i1:=0 to N0 do;
for>   for i2:=-N0 to N0 do;
for|for>     for i3:=-N0 to N0 do;
for|for|for>       for i4:=-N0 to N0 do;
for|for|for|for>         for i5:=-N0 to N0 do;
for|for|for|for|for>           if (i1^2 + i2^2 + i3^2 + i4^2 + i5^2 le N^2) then;
for|for|for|for|for|if>             pp:=i1*bas[1]+i2*bas[2]+i3*bas[3]+i4*bas[4]+i5*bas[5];
for|for|for|for|for|if>             if pp[3] eq 1 then;
for|for|for|for|for|if|if>               pts:=Append(pts,pp);
for|for|for|for|for|if|if>             end if;
for|for|for|for|for|if>           end if;
for|for|for|for|for>         end for;
for|for|for|for>       end for;
for|for|for>     end for;
for|for>   end for;
for> end for;
> pts;
[ (x + 23, -20, 1), (x + 221/9, -1580/243, 1), (x + 21,
    -20, 1), (x + 22, 20, 1), (x + 22, -20, 1), (x + 21, 20, 1), (x + 221/9,
    1580/243, 1), (x + 23, 20, 1), (x + 19, 20, 1), (x - 381, 3240100, 1), (x -
    29, 17660, 1), (x + 20, -20, 1), (x - 377, -3160100, 1), (x - 1411/9,
    -102391900/243, 1) ]
```

That means that the only integral points on $C$ in our search region are

$$(377, \pm 3160100), (-20, \pm 20), (29, \pm 17660), (381, \pm 3240100),$$
$$(-19, \pm 20), (-23, \pm 20), (-21, \pm 20), (-22, \pm 20)$$

Now we have to look for the integral points with $|x(P)| \leq \max\{1, \max_i\{2\|\alpha_i\|\}\}$.

```
> boundx:=2*Max([Modulus(i[1]) : i in Roots(Evaluate(f,x))]);
> boundx:=Max([boundx,1]);
> ptsBelowBound:=Points(C: Bound:=Ceiling(boundx));
> ptsBelowBound;
{@ (1 : 0 : 0), (-19 : -20 : 1), (-19 : 20 : 1), (-20 : -20 : 1),
 (-20 : 20 : 1), (-21 : -20 : 1), (-21 : 20 : 1), (-22 : -20 : 1),
(-22 : 20 : 1), (-23 : -20 : 1), (-23 : 20 : 1), (29 : -17660 : 1),
(29 : 17660 : 1) @}
```

We have completely determined the set of integral points on $C$.

The running time of the computations was

```
Total time: 7989.220 seconds, Total memory usage: 200.44MB
```

## References

[1] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[2] Yann Bugeaud, Maurice Mignotte, Samir Siksek, Michael Stoll, and Szabolcs Tengely. Integral points on hyperelliptic curves. *Algebra & Number Theory*, 2(8):859–885, 2008.

[3] E. V. Flynn and N. P. Smart. Canonical heights on the Jacobians of curves of genus 2 and the infinite descent. *Acta Arith.*, 79(4):333–352, 1997.

[4] Homero Gallegos-Ruiz. Computing integral points on genus 2 curves entimating hyperelliptic logarithms. *Acta Airth.*, 2019.

[5] Homero R. Gallegos-Ruiz. $S$-integral points on hyperelliptic curves. *Int. J. Number Theory*, 7(3):803–824, 2011.

[6] Kirti Joshi and Pavlos Tzermias. On the Coleman-Chabauty bound. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 329(6):459–463, 1999.

[7] Samir Siksek. Infinite descent on elliptic curves. *Rocky Mountain Journal of Mathematics*, 25(4):1501–1538, 1995.

[8] Paul van Wamelen. Computing with the jacobian of a genus 2 curve. Available at https://www.math.lsu.edu/∼wamelen/genus2.html.

CONACyT fellow, Unidad Académica de Matemáticas, Universidad Autónoma de Zacatecas, Mexico