MACHINE LEARNING FOR MODULI SPACE OF GENUS TWO CURVES AND AN APPLICATION TO POST-QUANTUM CRYPTOGRAPHY

ELIRA SHASKA AND TONY SHASKA

Abstract. We use machine learning to study the locus $\mathcal{L}_n$ of genus two curves with (n,n)-split Jacobian. More precisely we design a transformer model which given values for the Igusa invariants determines if the corresponding genus two curve is in the locus $\mathcal{L}_n$, for $n = 2, 3, 5, 7$. Such curves are important in isogeny based cryptography.

During this study we discover that there are no rational points $\mathfrak{p} \in \mathcal{L}_n$ with weighted moduli height ≤ 2 in any of $\mathcal{L}_2$, $\mathcal{L}_3$, and $\mathcal{L}_5$. This extends on previous work of the authors to use machine learning methods to study the moduli space of genus 2 algebraic curves.

1. Introduction

Genus two curves with (n,n)-split Jacobians have been the focus of study for a long time starting with Jacobi, Hermite, Goursat, Fricke, Brioschi, and in the last few decades by Frey, Kani, Shaska, Kumar, et al. For n odd, the locus of genus 2 curves with (n,n)-split Jacobian, denoted by Shaska in [17] by $\mathcal{L}_n$, is a 2-dimensional irreducible locus in the moduli space of genus 2 curves $\mathcal{M}_2$.

First Shaska [5, 11, 17] computed the equations of such loci for $n = 3, 5$ and then Kumar [4] confirmed such computational via a different approach. With the recent developments on Supersingular Isogeny key Encryption (SIKE) such curves have received new attention due to work of Castryck and Decru; see [3]. Cryptographic issues aside, there are still some computational questions that one could ask for the $\mathcal{L}_n$-locus in $\mathcal{M}_2$.

By a curve, we will always mean an irreducible algebraic curve defined over a field k of characteristic $\operatorname{char} k \neq 2$. When we consider the isomorphism classes of curves, these are always meant over the algebraic closure of k.

Given a genus 2 curve C, how easy it is to determine if C belongs to one of the loci $\mathcal{L}_n$? Since $\mathcal{L}_n$ are rational surfaces, there must be a way to generate rational points in $\mathcal{L}_n$ such that such points are fine moduli points (i.e, we can find a curve C defined over the field of moduli of points $\mathfrak{p} \in \mathcal{L}_n$)? Hence, even without knowing explicitly the equation of $\mathcal{L}_n$, we can create a training data and use such data to train some machine learning model.

To the best of our knowledge the only cases that are completely understood and explicitly given are for $n = 3$, and 5; see [11], [5] and some of the later refinements

Date: March 20, 2024.

1991 *Mathematics Subject Classification.* Primary: 68T07; Secondary: 68T20, 68Q32.

Key words and phrases. Stability, binary forms, weighted heights.

of such computations from authors. For example, for $n = 3$ a rational parametrization of $\mathcal{L}_3$ is given and the equation of genus 2 curve C is given in terms of such parameters (cf. Eq. (8)). Equations for $\mathcal{L}_n$ for $n > 3$ are very large and not very practical to use.

In this paper we use Machine Learning to design a model which helps determine if a curve C belongs to any of the loci $\mathcal{L}_n$. It builds on previous work of the authors on using neural networks to study the moduli space of genus 2 curves. We create a database of over one million isomorphism classes of genus two curves. Such classes are described by Igusa invariants J_2, J_4, J_6, J_{10}. We normalize each point $\mathfrak{p}$ in the weighted moduli space $\mathbb{WP}_{2,4,6,10}$ via the weighted greatest common divisor $\frac{1}{\text{wgcd}\,(\mathfrak{p})} \star \mathfrak{p}$. This assures that these moduli points are as small as possible. The data is stored in a python dictionary where the key is the set of absolute invariants (i_1, i_2, i_3) which are obtained via the Veronese embedding. This assures that there is no redundancy in our data.

We order the data points by their weighted moduli height (see [7]). For every point $\mathfrak{p} \in \mathbb{WP}_{2,4,6,10}$ we compute the weighted moduli height $\mathcal{S}_{\text{k}}(\mathfrak{p})$ of $\mathfrak{p}$, its automorphism group, whether the point is a fine moduli point (i.e, it is defined over its field of moduli), and if $\mathfrak{p}$ belongs to $\mathcal{L}_2$, $\mathcal{L}_3$, $\mathcal{L}_5$ and $\mathcal{L}_7$ even though the case $n = 7$ is much more involved computationally.

We use a transformer model to decide if a given moduli points $\mathfrak{p} = [J_2, J_4, J_6, J_{10}]$ is in $\mathcal{L}_n$. This gave an accuracy rate with 99% even though it failed to run using the whole database due to limitations in our computing capabilities. However, a sequential model would achieve the same accuracy for the whole database.

During our computations we noticed some interesting properties of spaces $\mathcal{L}_n$. For example, all $\mathcal{L}_n$, for $n = 2, 3, 5$ do not have rational points of weighted height ≤ 2. Using equations of $\mathcal{L}_n$ we were able to verify this. It provides an example how machine learning can be used effective as important tool in AI-assisted proofs.

2. Preliminaries

2.1. **Neural networks.** Let k be a field of characteristic $\text{char}\, k \neq 2$. A **neuron** is a function $f : k^n \to k$ such that for every $\mathbf{x} = (x_1, \ldots, x_n)$ we have

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i x_i + b,$$

where $b \in k$ is a constant called **bias**. We can generalize neurons to tuples of neurons via $F : \ k^n \to k^n$

$$\mathbf{x} \to (f_1(\mathbf{x}), \ldots, f_n(\mathbf{x}))$$

for any given set of weights $\mathfrak{w}_0, \ldots, \mathfrak{w}_n$. Then F is a function written as $F(\mathbf{x}) = W \cdot \mathbf{x} + \mathbf{b}$, for some $\mathbf{b} \in k^n$ and W an $n \times n$ matrix with integer entries. A (non-linear) function $g : k^n \to k^n$ is called an **activation function** while a **network layer** is a function $L : k^n \to k^n$, such that.

$$\mathbf{x} \to g\left(W \cdot \mathbf{x} + \mathbf{b}\right)$$

for some some activation function g. A **neural network** is the composition of many layers. The i-th layer

$$\cdots \longrightarrow k^n \xrightarrow{L_i} k^n \longrightarrow \cdots$$
$$\mathbf{x} \longrightarrow L_i(\mathbf{x}) = g_i\left(W^i \mathbf{x} + \mathbf{b}^i\right),$$

where g_i, W^i, and $\mathbf{b}^i$ are the activation, matrix, and bias corresponding to this layer.

2.2. **Weighted greatest common divisors.** Let $\mathbf{x} = (x_0, \dots x_n) \in \mathbb{Z}^{n+1}$ be a tuple of integers, not all equal to zero. Their greatest common divisor, denoted by $\gcd(x_0, \dots, x_n)$, is defined as the largest integer d such that $d|x_i$, for all $i = 0, \dots, n$. Let $q_0, \dots, q_n$ be positive integers. A set of weights is called the ordered tuple $\mathfrak{w} = (q_0, \dots, q_n)$. Denote by $r = \gcd(q_0, \dots, q_n)$ the greatest common divisor of $q_0, \dots, q_n$. A *weighted integer tuple* is a tuple $\mathbf{x} = (x_0, \dots, x_n) \in \mathbb{Z}^{n+1}$ such that to each coordinate x_i is assigned the weight q_i. We multiply weighted tuples by scalars $\lambda \in \mathbb{Q}$ via

$$\lambda \star (x_0, \dots, x_n) = (\lambda^{q_0} x_0, \dots, \lambda^{q_n} x_n)$$

For an ordered tuple of integers $\mathbf{x} = (x_0, \dots, x_n) \in \mathbb{Z}^{n+1}$, whose coordinates are not all zero, the **weighted greatest common divisor with respect to the set of weights $\mathfrak{w}$** is the largest integer d such that

$$d^{q_i} \mid x_i, \text{ for all } i = 0, \dots, n.$$

We will call a point $\mathfrak{p} \in \mathbb{P}^n_{\mathfrak{w}}(\mathbb{Q})$ **normalized** if $\operatorname{wgcd}(\mathfrak{p}) = 1$.

2.3. **Weighted projective spaces and moduli space of binary forms.** For any integer $m \geq 1$, let μ_m denote the group of m-th roots of unity generated by ξ_m, which is assumed to be contained in k. Consider the action of $k^* = k \setminus \{0\}$ on $\mathbb{A}^{n+1}_k \setminus \{(0, \cdots, 0)\}$ given by

$$\lambda \star (x_0, \dots, x_n) = (\lambda^{q_0} x_0, \dots, \lambda^{q_n} x_n), \text{ for } \lambda \in k^*. \tag{1}$$

Define the **weighted projective space**, denoted by $\mathbb{WP}^n_{\mathfrak{w}}(k)$, to be the quotient space $\mathbb{V}^{n+1}_k / k^*$ of this action, which is a geometric quotient since k^* is a reductive group. An element $\mathbf{x} \in \mathbb{WP}^n_{\mathfrak{w}}(k)$ is denoted by $\mathbf{x} = [x_0 : \cdots : x_n]$ and its i-th coordinate by $x_i(\mathbf{x})$.

2.4. **Heights on weighted projective spaces.** For any point $\mathfrak{p} = [x_0 : \cdots : x_n] \in \mathbb{P}^n_{\mathfrak{w},k}$ we can assume, without loss of generality, that $\mathfrak{p} = [x_0 : \cdots : x_n] \in \mathbb{P}^n_{\mathfrak{w},k}(\mathcal{O}_k)$. Let $\mathfrak{w} = (q_0, \dots, q_n)$ be a set of weights and $\mathbb{P}^n_{\mathfrak{w},k}$ the weighted projective space over a number field k. Let $\mathfrak{p} \in \mathbb{P}^n_{\mathfrak{w},k}$ a point such that $\mathfrak{p} = [x_0, \dots, x_n]$. We define the **weighted multiplicative height** of $\mathfrak{p}$ as

$$\mathcal{S}_{\mathrm{k}}(\mathfrak{p}) := \prod_{v \in M_k} \max \left\{ |x_0|_v^{\frac{n_v}{q_0}}, \dots, |x_n|_v^{\frac{n_v}{q_n}} \right\}. \tag{2}$$

Let $\mathbb{P}_{\mathfrak{w},k}$ be a well-formed weighted projective space and $\mathbf{x} = [x_0 : \cdots : x_n] \in \mathbb{P}_{\mathfrak{w},k}(k)$. Assume $\mathbf{x}$ normalized (i.e. $\operatorname{wgcd}_k(\mathbf{x}) = 1$). Clearly $\operatorname{wgcd}(\mathbf{x}) | \gcd(x_0, \dots, x_n)$ and therefore $\operatorname{wgcd}(\mathbf{x}) \leq \gcd(x_0, \dots, x_n)$.

Let $\mathbb{P}_{\mathfrak{w},k}$ be a well-formed weighted projective space and $\mathbf{x} = [x_0 : \cdots : x_n] \in \mathbb{P}_{\mathfrak{w},k}(k)$ such that $\mathbf{x}$ is absolutely normalized. Then $\gcd(x_0, \dots, x_n) = 1$.

If $\mathbf{x} = [x_0 : \dots, x_n]$ is a normalized point then by definition of the height

$$\mathcal{S}_{\mathrm{k}}(\mathbf{x}) = \max_{i=0}^{n} \{|x_i|^{\frac{1}{q_i}}\}$$

Assume now that $\mathbf{x} = [\lambda^{q_0} x_0 : \cdots : \lambda^{q_n} x_n]$ such that $\lambda = \text{wgcd}\,(\lambda^{q_0} x_0 : \cdots : \lambda^{q_n} x_n)$. Denote by s the index where $\min_j\{|\lambda^{q_i} x_j|^{\frac{1}{q_j}}\} = \lambda \min_j\{|x_j|^{\frac{1}{q_j}}\}$ is obtained. Then

$$\frac{1}{\lambda(x_s)^{1/q_s}} \star \mathbf{x} = \left[\frac{x_0}{x_s^{q_0/q_s}} : \cdots : 1 : \cdots : \frac{x_n}{x_s^{q_n/q_s}}\right] =: \mathbf{y} \tag{3}$$

where 1 is in the s position. Simplify all coordinates in Eq. (3). Multiplying $\mathbf{y}$ by $(x_s)^{\frac{1}{q_s}}$ we have

$$(x_s)^{\frac{1}{q_s}} \star \mathbf{y} = [x_0 : \cdots : x_n],$$

which is now a normalized point. Hence

$$\mathcal{S}_{\text{k}}(\mathbf{x}) = \frac{\max_{i=0}^n\{|\lambda^{q_i} x_i|^{\frac{1}{q_i}}\}}{\min_{i=0}^n\{|\lambda^{q_i} x_i|^{\frac{1}{q_i}}\}}$$

Notice that $\mathcal{S}_{\text{k}}(\mathbf{x})$ is given by

$$\mathcal{S}_{\text{k}}(\mathbf{x}) = \frac{\max_i |x_i|^{\frac{1}{q_i}}}{\min_j |x_j|^{\frac{1}{q_j}}}, \tag{4}$$

When $\mathbf{x} = [x_0 : \cdots : x_n] \in \mathbb{P}_{\mathfrak{w}}(\mathbb{Q})$ is an absolutely normalized point, then

$$\gcd(x_0, \ldots, x_n) \leq \mathcal{S}_{\text{k}}(\mathbf{x})$$

3. Moduli space $\mathcal{M}_2$ of genus 2 curves

The moduli space of genus 2 curves is isomorphic to $\mathbb{P}_{(1,2,3,5)}$. Moreover the invariant J_{10} of degree 10 is the discriminant of the sextic and therefore $J_{10} \neq 0$. Thus, the morphism $\mathbb{P}^3_{(2,4,6,10),k} \to \mathbb{P}^3_{(1,2,3,5),k}$, given by

$$[x_0 : x_1 : x_2 : x_3] \to [y_0 : y_1 : y_2 : y_3] = \left[x_0^2 : x_1^2 : x_2^2 : x_3^2\right] \tag{5}$$

is an isomorphism.

Since we want to design a model where the incoming features will be a genus two curve, then equivalently this means a point $[x_0 : x_1 : x_2 : x_3] \in \mathbb{P}_{(1,2,3,5)}$. Equivalently the input could be the equation of the curve, but this poses no issue for $g = 2$ since we can compute easily compute invariants.

There is also an issue to address when it comes to finding the "smallest" representatives for the equivalence class $[x_0 : x_1 : x_2 : x_3]$. Theoretically this is handled in [1], but that would require computing weighted greatest common divisors and that could be very costly for large coordinates $x_0, \ldots, x_3$.

Since $q = 1 \cdot 2 \cdot 3 \cdot 5 = 30$, the Veronese embedding is

$$[J_2 : J_4 : J_6 : J_{10}] \longrightarrow \left[J_2^{30} : J_4^{15} : J_6^{10} : J_{10}^6\right] = \left[\frac{J_2^{30}}{J_{10}^6} : \frac{J_4^{15}}{J_{10}^6} : \frac{J_6^{10}}{J_{10}^6} : 1\right] \tag{6}$$

So the triple

$$i_1 = \frac{J_2^{30}}{J_{10}^6}, \qquad i_2 = \frac{J_4^{15}}{J_{10}^6}, \qquad i_3 = \frac{J_6^{10}}{J_{10}^6} \tag{7}$$

uniquely determines the equivalence class. We create a dictionary with keys (i_1, i_2, i_3).

3.1. **Distribution of fine points in $\mathcal{M}_2$.** There are two types of points in $\mathbb{WP}_{(2,4,6,10)}$, namely *fine points* and *coarse points*. Fine points are those points such that their field of moduli is a field of definition, while the rest of points are called coarse points.

We also classify fine points in two classes, those with extra automorphisms and those which have automorphism group isomorphic to the cyclic group of order 2.

3.2. **Genus two curves with extra involutions.** The set of points with extra automorphisms is a 2-dimensional irreducible subvariety of the moduli space corresponding exactly to points $p \in \mathbb{WP}_{(2,4,6,10)}$ which satisfying $J_{30}(p) = 0$. This locus has two 1-dimensional loci corresponding to points with automorphism group D_4 and D_6; for details see [6].

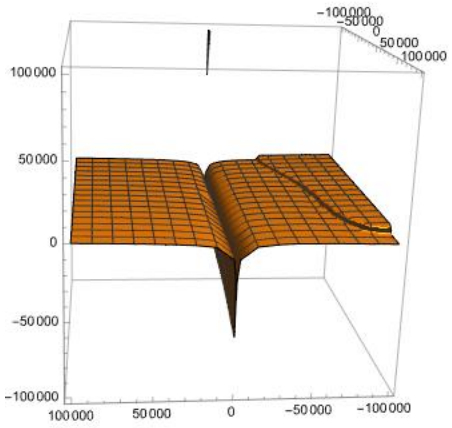

FIGURE 1. $\mathcal{L}_2$ surface

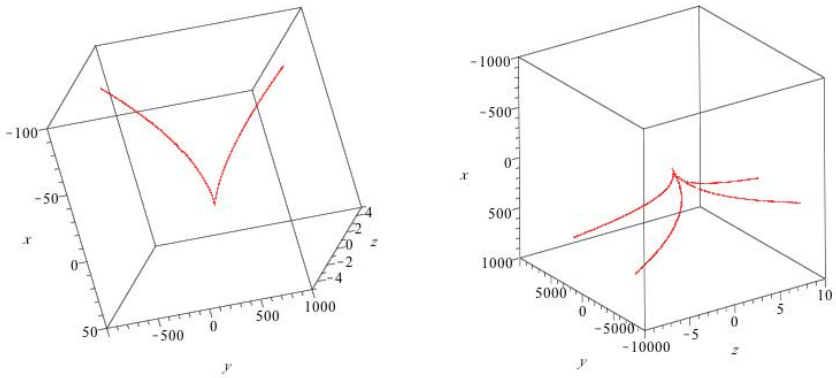

FIGURE 2. 1-dimensional subloci of $\mathcal{L}_2$: curves with automorphism group D_4 or D_6

4. Genus two curves with (n,n)-split Jacobians

Genus two curves with (n,n)-split Jacobians have been studied thoroughly during the last two decades and have been getting again some attention lately due to their use in isogeny based cryptography; see [5, 9, 11–14, 17, 18] among others. This extends the database from [2].

As in [17], we denote the locus of genus 2 curves with (n,n)-split Jacobian by $\mathcal{L}_n$. For n odd, it is a 2-dimensional irreducible locus in $\mathcal{M}_2$. Here we will describe how to create a database of points in $\mathcal{L}_n$ for $n = 5, 7, 11$.

A degree n covering $C \to E$, where C is a genus two curve and E an elliptic curve, induces a degree n covering $\phi : \mathbb{P}^1 \to \mathbb{P}^1$ with ramification

$$\left(2^{\frac{n-1}{2}}, 2^{\frac{n-1}{2}}, 2^{\frac{n-1}{2}}, 2^{\frac{n-3}{2}}, 2\right)$$

The unramified points in the fibers of the first four branch points are the Weierstrass points of the genus 2 curve.

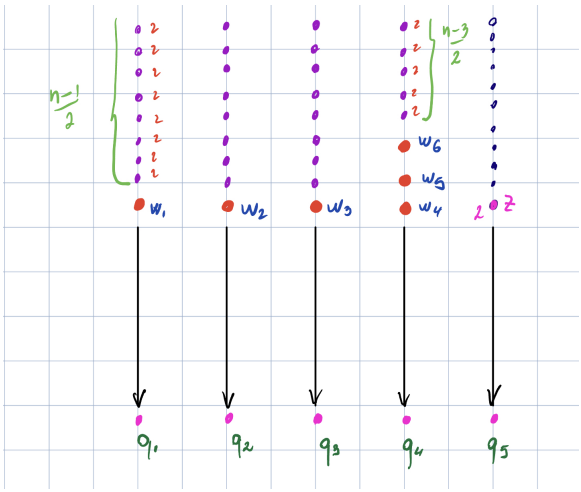

Figure 3. Ramification type and Weierstrass points

Denote by $F_i(x)$, $i = 1, \dots 4$ the polynomial over the branch point q_i, which has as roots points of ramification index 2. Hence, $\deg F_1 = \deg F_2 = \deg F_2 = \frac{n-1}{2}$, and $\deg F_4 = \frac{n-3}{2}$. We fix a coordinate on the lower $\mathbb{P}^1$ by letting $q_1 = 0$, $q_2 = \infty$, and $q_3 = 1$ and on the upper $\mathbb{P}^1$ by $w_1 = 0$, $w_1 = 1$, and $w_3 = \infty$. Then

$$\phi(x) = x \left(\frac{F_1(x)}{F_2(x)}\right)^2, \quad \phi(x) - 1 = (x-1) \left(\frac{F_3(x)}{F_2(x)}\right)^2,$$

Cases when $n = 3$ and $n = 5$ are special cases. When $n = 3$ the fibers of q_1, q_2, q_3, and q_5 are identical, so we have extra symmetries permuting these branch points. When $n = 5$ then the fibers of q_4 and q_5 are identical so we have an extra involution permuting q_4 and q_5. The first general case is when $n = 7$ which is somewhat simpler since we don't have to worry about such extra symmetries, but of course everything becomes computational more challenging when n gets bigger.

4.1. $(3,3)$**-split:** This case was studied in [17] and summarized in [11]. Let C be a genus 2 curve with $(3,3)$ split Jacobian. Then from [14, Theorem 4] C has equation

$$y^2 = \left(\mathfrak{v}^2x^3 + \mathfrak{u}\mathfrak{v}x^2 + \mathfrak{v}x + 1\right)\left(4\mathfrak{v}^2x^3 + \mathfrak{v}^2x^2 + 2\mathfrak{v}x + 1\right) \tag{8}$$

for $\Delta = v\,(v-27)\left(4u^3 - u^2v - 18uv + 4v^2 + 27v\right) \neq 0$. Its moduli point is

$$\begin{aligned}\mathfrak{p} &= \left[2v^4\alpha, 4v^7\beta, 4v^{10}\gamma, -16v^{17}(v-27)\delta^3\right] \\ &= \left[2v\alpha, 4v\beta, 4v\gamma, -16v^2(v-27)\delta^3\right]\end{aligned}$$

since $v \neq 0$ and $\alpha, \beta, \gamma, \delta$ are

$$\begin{aligned}\alpha &= 4u^2 - 12uv + 3v^2 + 252u - 54v - 405 \\ \beta &= u^4v - 24u^4 - 66u^3v + 9u^2v^2 + 1188u^3 + 297u^2v + 138u\,v^2 - 36v^3 - 8424uv \\ &\quad + 945v^2 + 14580v \\ \gamma &= 2u^6v^2 - 8u^5v^3 + 2u^4v^4 - 40u^6v + 106u^5v^2 + 495u^4v^3 - 204u^3v^4 + 18u^2v^5 - 144u^6 \\ &\quad + 1476u^5v - 18756u^4v^2 + 4280u^3v^3 - 1038u^2v^4 + 564u\,v^5 - 72v^6 + 160704u^4v \\ &\quad + 4464u^3v^2 + 75024u^2v^3 - 33480u\,v^4 + 3186v^5 - 104004u^3v - 1353996u^2v^2 + 315252u\,v^3 \\ &\quad - 4032v^4 + 3669786u\,v^2 - 622323v^3 - 2821230v^2 \\ \delta &= 4u^3 - u^2v - 18uv + 4v^2 + 27v\end{aligned}$$

Notice that for rational values of $\mathfrak{u}, \mathfrak{v}$ we get rational points $\mathfrak{p} \in \mathcal{L}_3$. This provides an easy way to generate a database of points fine points in $\mathcal{L}_3$. However, if we

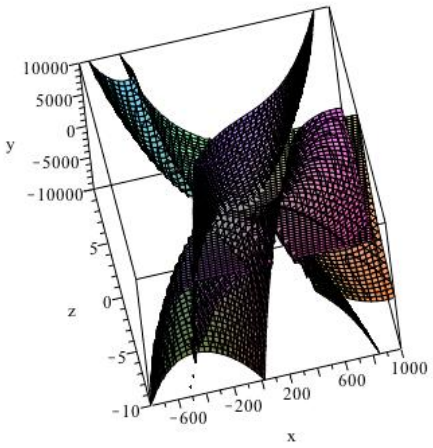

FIGURE 4. $\mathcal{L}_3$ surface

simply want to generalize rational points in $\mathcal{L}_3$, not necessarily fine moduli points, we have to use parameters r_1, r_2 which provide a birational parametrization of $\mathcal{L}_3$, but not necessarily genus 2 curves defined over $\mathbb{Q}$; see [11] for details.

Lemma 1. *There are no rational points $p \in \mathcal{L}_3$ with weighted height $\mathcal{S}_{\mathrm{k}}(\mathfrak{p}) < 2$.*

The proof is a brute force approach. We check all points $\mathfrak{p} \in \mathbb{WP}_{(2,4,6,10)}$ of weighted height $\mathcal{S}_{\rm k}(\mathfrak{p}) \leq 3$. There are, however, many rational points for weighted height $2 < \mathcal{S}_{\rm k}(\mathfrak{p}) \leq 3$. $\mathcal{L}_3$ has at least these rational points

#	$\mathfrak{p}$	#	$\mathfrak{p}$	#	$\mathfrak{p}$
1	[6, 18, 27, 2]	2	[-6, -18, 45, 2]	3	[3, 18, 0, 4]
4	[-3, -18, 36, 4]	5	[5, -26, 56, 4]	6	[-3, 27, 315, 4]
7	[-5, 58, -76, 4]	8	[-5, 31, -49, 4]	9	[5, 29, -9, 4]
10	[-2, -18, 39, 6]	11	[-2, -18, 165, 6]	12	[2, 18, -15, 6]
13	[-8, -80, 429, 8]	14	[-8, 49, -101, 8]	15	[8, -47, -59, 8]
16	[-1, -18, 60, 12]	17	[8, 36, 69, 12]	18	[-1, -45, 105, 12]
19	[-8, -36, 123, 12]	20	[-5, 67, -55, 12]	21	[1, 18, -48, 12]
22	[1, 63, -15, 12]	23	[5, -65, -15, 12]	24	[6, 36, 36, 16]
25	[-6, -36, 108, 16]	26	[8, -63, 3, 24]	27	[-4, -36, 102, 24]
28	[-8, 81, -171, 24]	29	[4, 36, -6, 24]	30	[5, -59, 16, 32]
31	[5, -68, 100, 32]	32	[-3, -36, 108, 32]	33	[-3, -27, 504, 32]
34	[-5, 61, -132, 32]	35	[3, 36, -36, 32]	36	[9, 54, 108, 36]
37	[-9, -54, 216, 36]	38	[-2, -36, 132, 48]	39	[-2, -72, 336, 48]
40	[3, 18, 0, 4]	41	[-3, -18, 36, 4]	42	[-2, -18, 39, 6]
43	[2, 18, -15, 6]	44	[-1, -18, 60, 12]	45	

TABLE 1. Some rational points of height ≤ 3 in $\mathcal{L}_3$

Remark 1. *In* [10] *was studied the intersection* $\mathcal{L}_2 \cap \mathcal{L}_3$. *Such points* $\mathfrak{p} \in \mathcal{L}_2 \cap \mathcal{L}_3$ *have Jacobians which are* $(2,2)$*-split and* $(3,3)$*-split and rational points in such genus 2 curves were determined. In* [14] *are given other examples of genus 2 curves with many elliptic subcovers (Jacobian splits in more than one way).*

4.2. $(5,5)$**-split:** This case was studied in detail in [5]. The parametric family of genus two curves in the locus $\mathcal{L}_5$ is given by

$$C: \qquad y^2 = x(x-1)\left(a_3x^3 + a_2x^2 + a_1x + a_0\right) \tag{9}$$

where

$$\begin{aligned}
a_0 =& -b^4(2b^3a + 4b^3 - 2zab^2 + 7b^2a^2 + 8zb^2 + 4b^2 + 16ab^2 + 16zba + 6a^3b + 8ba \\
&+ 2za^2b + 12zb + 16ba^2 + 13za^2 + za^4 + 6za^3 + 4z + 12ya) \\
a_1 =& -b^2(12b^3 + 12b^4a + 32zba - 6a^4b^2 + 44b^2a^3 + 6ba^2 + 24ab^2 + 10a^3b + 44b^3a^2 + 2ba \\
&+ 52b^3a + 61b^2a^2 - 12ba^5 - 7za^2 - 2za + 12zb - 4a^6 + 12b^4 - a^4 - 40za^3b^2 - 16zb^3a^2 \\
&- 12za^5 + 36zb^2 - 18za^3 - 26za^4 + 56zab^2 + 4azb^3 + 2za^2b^2 - 20za^3b + 28za^2b \\
&+ 2za^6 + 24zb^3 + 4zba^5 - 4a^5 - 32za^4b) \\
a_2 =& 5b^2a^6 + 20b^2a^5 + 8ba^6 - 61b^4a^2 - 18b^5a - 56b^4a + 4zba + 5a^4b^2 - 18b^2a^3 - 24zb^4 \\
&- 14zb^4a - 4ab^2 + 8b^3a^4 + 2b^3a^5 - 54b^3a^3 - 70b^3a^2 - 24b^3a - 14b^2a^2 + 4a^4b + 10ba^5 \\
&- 6za^7 + 64za^3b^3 + 38za^4b^2 + 54za^3b^2 + 12zb^3a^2 - 14za^6b - 10zb^2a^5 - 4za^7b - 4a^6zb^2 \\
&+ 32a^2b^4z + 2a^7b - za^8 - 36zb^3 - 12za^5 - 12zb^2 - 4za^4 - 28zab^2 - 64azb^3 - 5za^2b^2 \\
&+ 16za^2b + 28za^4b - 4zba^5 - 13za^6 - 12b^5 - 12b^4 + 34za^3b \\
a_3 =& (2a+1)(za^4 - 2a^3b + 4za^3 + 6za^3b - 4ba^2 + 12za^2b^2 + 10za^2b - 9b^2a^2 + 5za^2 \\
&- 2ba + 2za - 8ab^2 - 12b^3a + 8azb^3 - 4b^3 - 4zb - 4b^4 - 12zb^2 - 8zb^3)
\end{aligned}$$

As mentioned above, there is an involution permuting branch points q_4 and q_5. Moreover, a, b, and z satisfy the equation

$$(1+2a)\,z^2 + \left(-a^2 - 2ab - 2a + 2b\right) z + 2ab + b^2,$$

see [5, Thm. 2] for details. This is a cubic surface and by a result of Clebsch it is rational. Hence $a, b \in k(t,s)$ for some parameters t and s. Thus invariants $J_2, J_4, J_6, J_{10} \in k(t,s)$. Giving random values to t and s generates rational points in $\mathcal{L}_5$.

4.2.1. *Equation of $\mathcal{L}_5$.* Computing the equation of $\mathcal{L}_5$ sounds as an easy exercise in elimination theory: compute $i_1(t,s), i_2(t,s), i_3(t,s)$ and eliminate t and s. This will give an affine equation $f(i_1, i_2, i_3) = 0$. The get the projective equation, substitute i_1, i_2, i_3 in terms of J_2, J_4, J_6, J_{10}.

The main problem with the above approach is that the degree of rational functions i_1, i_2, i_3 in terms of t and s are very large, which makes the elimination of t and s practically impossible.

In [5] was given the following approach in the computation of $\mathcal{L}_5$. Let

$$\mathfrak{u} = \frac{2a\ (ab + b^2 + b + a + 1)}{b\ (a + b + 1)}, \qquad \mathfrak{v} = \frac{a^3}{b\ (a + b + 1)}, \quad w = \frac{(z^2 - z + 1)^3}{z^2 (z-1)^2}$$

They are invariants of a group action on $k(a,b,z)$. Since the modular invariants J_2, J_4, J_6, J_{10} are invariants of any permutation of fibers, they can be expressed in terms of u, v, w. Moreover,

$$k(\mathcal{L}_5) = k(u, v, w),$$

where the equation of w in terms of u, v is

$$c_2 w^2 + c_1 w + c_0 = 0 \tag{10}$$

with c_0, c_1, c_2 as follows:

$$\begin{aligned} c_2 =& 64v^2(u - 4v + 1)^2 \\ c_1 =& -4v(-272v^2u - 20vu^2 + 2592v^3 - 4672v^2 + 4u^3 + 16v^3u^2 - 15vu^4 \\ & - 96v^2u^2 + 24v^2u^3 + 2u^5 - 12u^4 + 92vu^3 + 576vu - 128v^4 - 288v^3u) \\ c_1 =& (u^2 + 4vu + 4v^2 - 48v)^3 \end{aligned} \tag{11}$$

Notice that the surface in u, v, w is a septic surface and more difficult to get a parametrization of it. However, expressing $i_1(u,v,w), i_2(u,v,w), i_3(u,v,w)$ together with the (10) gives us a system of equations which is easier to handle and possible to eliminate u, v, and w.

Remark 2. *The equation of $\mathcal{L}_5$ was computed in [5]. Since it is too long it was not displayed in the paper, but in a webpage that no longer is available. Due to regular and repeated requests for this equation, we intend to display it in [16], even though it is exactly its impracticality of usage due to its length which motivates this paper.*

4.3. **Cases for $n \geq 7$.** Both cases $n = 3$ and $n = 5$ are special cases due to their ramification structure. For example, for $n = 3$ the fibers $\phi^{-1}(q_i)$, for $i = 1, 2, 3$ and $\phi^{-1}(q_5)$ are the same and for $n = 5$ the fibers $\phi^{-1}(q_4)$ and $\phi^{-1}(q_5)$ are the same. This fact induces extra symmetries and therefore a group action as shown in the computation of these spaces.

The first case which is a general case (i.e. the ramification structure is the same as large n) is the case $n = 7$. Its computation is much more involved for such a small paper.

Remark 3. *There are other ways how to generate rational points in $\mathcal{L}_n$, even though not general points. The general ramification for $n > 7$ has four cases (called degenerate cases in [5, 11]). These cases give 4 curves in $\mathcal{L}_n$ and two of these curves are genus zero curves. Hence, a rational parametrization of these curves would provide rational points in $\mathcal{L}_n$. However, any model based only on these points would be suited only for this curve and not the whole $\mathcal{L}_n$ space.*

5. A machine learning approach

Now that we know how to generate rational points in $\mathcal{L}_n$, we would like to see if we can generate some random data in $\mathcal{M}_2$ and train a model that answer arithmetic properties of $\mathfrak{p} \in \mathcal{M}_2$, including whether $\mathfrak{p} \in \mathcal{L}_n$. Our data will have points from $\mathcal{L}_n$, for $n = 2, 3, 5, 7$ and all points of weighted moduli height $\mathcal{S}_{\rm k} \leq 3$.

Notice that for a fixed $h \in \mathbb{R}^{\geq 0}$ the number of points in $\mathbb{WP}_{(2,4,6,10)}$ with bounded height h is finite, by Northcott's theorem for weighted heights (see [1, Theorem 1]. More precisely, such number is bounded by

$$(2h^2 + 1)(2h^4 + 1)(2h^6 + 1)2h^{10}$$

This bound is just a combinatorial bound and counts all tuples without considering their equivalence in the weighted projective space. For more detailed version of such counting issues see [8].

5.1. **Distribution of fine points in the moduli space.** Let $\mathbf{x} \in \mathbb{P}^n_{\mathfrak{w}}(\mathbb{Q})$. Can we find a binary form $f \in V_d$, defined over $\mathbb{Q}$, such that $\mathbf{x} = [f]$. The answer is in general negative. Points for which we can answer positively the above question are called **fine moduli points**, otherwise $\mathbf{x}$ has a non-trivial obstruction and we will call it a **coarse point**. The problem of determining which points are fine points is referred to as field of moduli versus field of definition problem in arithmetic of moduli spaces of curves; see [15] among many others.

To determine some distribution of fine moduli points using any machine learning techniques one can attempt support vector machines which we will discuss briefly later. Before doing that we want to focus on a special locus where the field of moduli is always a field of definition.

An entry in the dictionary looks like:

$$(x, y, z) : (\mathfrak{p}, \mathcal{S}_{\rm k}(\mathfrak{p}), \text{Fine}, \text{Aut}(\mathfrak{p}), \mathfrak{p} \in \mathcal{L}_3, \mathfrak{p} \in \mathcal{L}_5, \mathfrak{p} \in \mathcal{L}_7)$$

where

$$\begin{aligned}
\mathfrak{p} &= [J_2, J_4, J_6, J_{10}] \text{ weighted moduli point} \\
\mathcal{S}_{\rm k} &= \textit{ weighted height} \\
\text{Fine} &= \textit{ True/False} \\
\text{Aut} &= \text{Automorphism group } \text{Aut}(\mathfrak{p}) \\
flag &= (3,3) - \text{split Jacobian} \\
flag &= (5,5) - \text{split Jacobian} \\
flag &= (7,7) - \text{split Jacobian}
\end{aligned}$$

where the key (x, y, z) is the triple of absolute invariants (i_1, i_2, i_3). Notice that the automorphism group basically determines if the corresponding point is in the $\mathcal{L}_2$ locus or not. Other then the curve $y^2 = x(x^5 - 1)$, all curves with $|\mathrm{Aut}(\mathfrak{p})| > 2$ are in the $\mathcal{L}_2$ locus and they are fine moduli points.

We use a transformer model for each $\mathcal{L}_n$ even though different models were tested with similar results. First we extract input features and labels from the dictionary `M_dict` containing moduli points $\mathfrak{p} = [J_2, J_4, J_6, J_{10}]$ and converts them into PyTorch tensors. We define a transformer-based neural network model using PyTorch's `nn.Module`. The model consists of an embedding layer, a transformer layer, global average pooling, and two fully connected layers with ReLU and sigmoid activation functions. The training is done by a binary cross-entropy loss and an Adam optimizer, which trains the model for a specified number of epochs using a loop. Within each epoch, it performs forward and backward passes, updates the model parameters, and prints the loss.

5.2. **Results.** Let us now see what is the distribution of fine points with automorphisms in out database (red points). From a computational point of view it is quite hard to do this simply by brute force for our database which has about 500 000 points. Instead we will use the above models to see what information we can gather and then prove our results (if any) via brute force computationally.

By taking a random sample and graphing all red points we get the following picture.

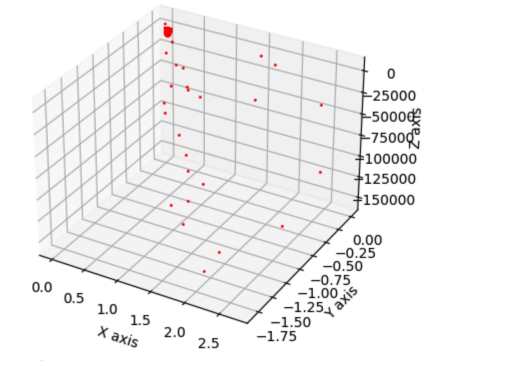

FIGURE 5. Distribution of points with extra automorphisms

Red points seem to be very scarce around the origin of the coordinate system and secondly there are no green points (fine points with trivial automorphism group). Somewhat to be expected by people who have extensive computational experience with the moduli space of genus 2 curves, but not any obvious theoretical reason for it.

5.2.1. *Coarse moduli points in $\mathcal{M}_2$.* We graph below all points for weighted height $\mathcal{S}_{\mathrm{k}}(p) \leq 3$.

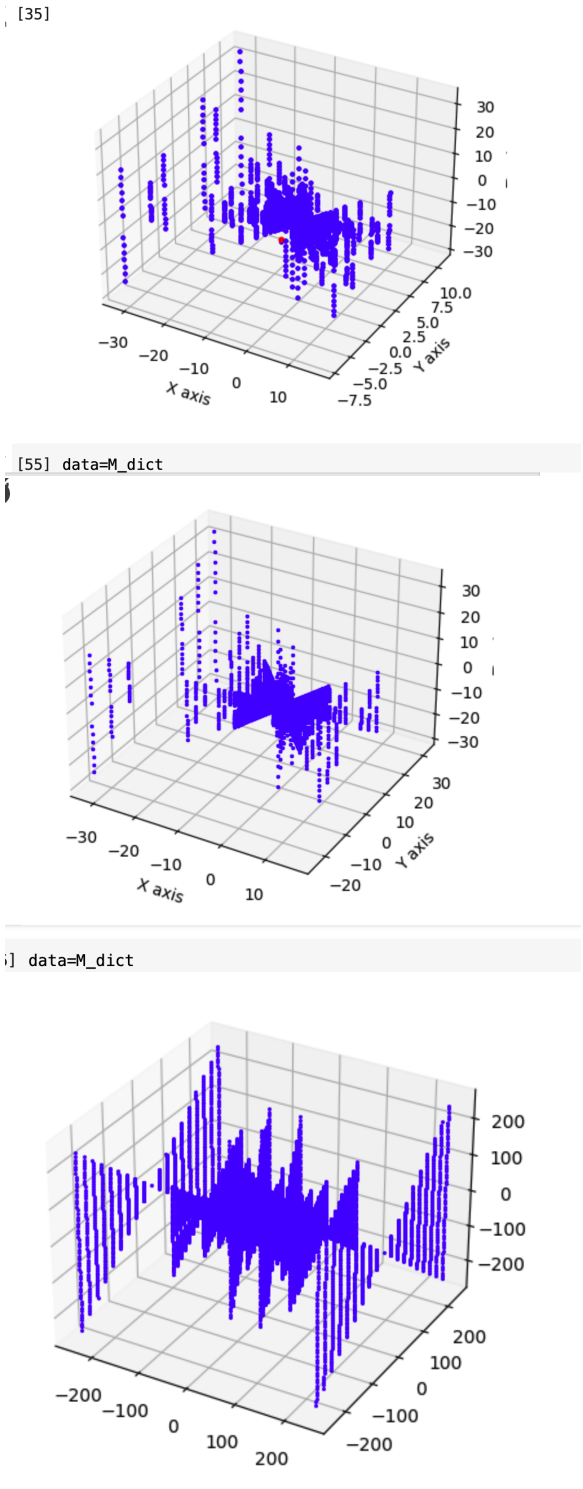

FIGURE 6. Graph of rational points of weighted height $\mathcal{S}_{\mathrm{k}}\, leq 2$

Surprisingly there is only one red dot in all these graphs. These graphs were obtained using the sequential method and the existing red dot was no surprise because that is a very special genus 2 curve and well known. However, these graphs were a strong enough reason for us to go through all the cases for $\mathcal{S}_{\mathrm{k}}(p) \leq 3$ and check computationally. Cases for $n = 3, 5$ generate similar pictures.

Remark 4. *There are no rational points with weighted moduli height $\mathcal{S}_{\mathrm{k}} < 2$ in the loci $\mathcal{L}_2, \mathcal{L}_3, \mathcal{L}_5$.*

Proof. Computationally compiling a list of all points with weighted height $\mathcal{S}_{\mathrm{k}}(p) \leq 3$ and checking each if they satisfy the equation of $\mathcal{L}_n$, for $n = 2, 3, 5$. □

6. Concluding remarks

With necessary adjustments, these results can be extended to characteristic 2. A further invariant of order eight, usually denoted by J_8, is needed in this case.

Our models didn't always give us what we expected in terms of efficiency and at this point it is unclear to us if this is due to our limitations in computing power or limitations of the architectures chosen. This remains to be further investigated.

Our main goal was to test whether machine learning techniques can be used in theoretical areas of mathematics and we can definitely say that this note gives a positive answer to that question.

References

[1] L. Beshaj, J. Gutierrez, and T. Shaska, *Weighted greatest common divisors and weighted heights*, J. Number Theory **213** (2020), 319–346. MR4091944

[2] L. Beshaj, R. Hidalgo, S. Kruk, A. Malmendier, S. Quispe, and T. Shaska, *Rational points in the moduli space of genus two*, Higher genus curves in mathematical physics and arithmetic geometry, [2018] ©2018, pp. 83–115. MR3782461

[3] Wouter Castryck and Thomas Decru, *An efficient key recovery attack on SIDH*, Advances in cryptology—EUROCRYPT 2023. Part V, [2023] ©2023, pp. 423–447. MR4591003

[4] Abhinav Kumar, *Hilbert modular surfaces for square discriminants and elliptic subfields of genus 2 function fields*, Res. Math. Sci. **2** (2015), Art. 24, 46. MR3427148

[5] Kay Magaard, Tanush Shaska, and Helmut Völklein, *Genus 2 curves that admit a degree 5 map to an elliptic curve*, Forum Math. **21** (2009), no. 3, 547–566. MR2526800

[6] A. Malmendier and T. Shaska, *From hyperelliptic to superelliptic curves*, Albanian J. Math. **13** (2019), no. 1, 107–200. MR3978315

[7] Sajad Salami and Tony Shaska, *Local and global heights on weighted projective varieties and Vojta's conjecture*, arXiv preprint arXiv:2204.01624 (2022).

[8] E. Shaska and T. Shaska, *Deep learning and the moduli space of curves*, 2024. in progress.

[9] T. Shaska, *Curves of genus 2 with (N, N) decomposable Jacobians*, J. Symbolic Comput. **31** (2001), no. 5, 603–617. MR1828706

[10] ———, *Genus 2 curves with $(3, 3)$-split Jacobian and large automorphism group*, Algorithmic number theory (Sydney, 2002), 2002, pp. 205–218. MR2041085

[11] ———, *Genus 2 fields with degree 3 elliptic subfields*, Forum Math. **16** (2004), no. 2, 263–280. MR2039100

[12] ———, *Genus two curves covering elliptic curves: a computational approach*, Computational aspects of algebraic curves, 2005, pp. 206–231. MR2182041

[13] ———, *Genus two curves covering elliptic curves: a computational approach*, Computational aspects of algebraic curves, 2005, pp. 206–231. MR2182041 (2006g:14051)

[14] ———, *Genus two curves with many elliptic subcovers*, Comm. Algebra **44** (2016), no. 10, 4450–4466. MR3508311

[15] ———, *From hyperelliptic to superelliptic curves*, in preparation, 2020.

[16] ———, *https://github.com/tshaska*, 2020.

[17] Tanush Tony Shaska, *Curves of genus two covering elliptic curves*, ProQuest LLC, Ann Arbor, MI, 2001. Thesis (Ph.D.)–University of Florida. MR2701993

[18] Tony Shaska, *Genus two curves with many elliptic subcovers*, Comm. Algebra **44** (2016), no. 10, 4450–4466. MR3508311

Appendix A. Transformer model

```
import torch
import torch.nn as nn
import torch.optim as optim
# Extract input features and labels from the dictionary
coordinates = []
features = []
labels = []
for coords, (input_features, label, _) in M_dict.items():
    coordinates.append(coords)
    features.append(input_features)
    labels.append(int(label))  # Convert boolean to int
# Convert lists to PyTorch tensors
features = torch.tensor(features, dtype=torch.float32)
labels = torch.tensor(labels, dtype=torch.float32)
# Map the original features to indices
unique_features, inverse_indices = torch.unique(features, dim=0,
    return_inverse=True)
# Define the transformer model
class TransformerModel(nn.Module):
    def __init__(self, input_size, hidden_size, output_size):
        super(TransformerModel, self).__init__()
        self.embedding = nn.EmbeddingBag(num_embeddings=len(
    unique_features), embedding_dim=hidden_size)
        self.transformer = nn.Transformer(d_model=hidden_size, nhead
    =4, num_encoder_layers=3)
        self.global_pooling = nn.AdaptiveAvgPool1d(1)  # Global
    average pooling
        self.fc1 = nn.Linear(hidden_size, 64)
        self.fc2 = nn.Linear(64, output_size)
        self.relu = nn.ReLU()
        self.sigmoid = nn.Sigmoid()
    def forward(self, x):
        x = self.embedding(x, torch.zeros(x.size(0), dtype=torch.long)
    )  # Use zeros as offsets
        x = x.view(1, x.size(0), -1)  # Add batch dimension and
    rearrange features
        x = self.transformer(x, x)  # Use x as both source and target
    sequences
        x = self.global_pooling(x.permute(0, 2, 1))  # Global average
    pooling
        x = x.squeeze(dim=-1)
        x = self.relu(self.fc1(x))
        x = self.sigmoid(self.fc2(x))
        return x
# Instantiate the model
model = TransformerModel(input_size=len(unique_features), hidden_size
    =32, output_size=1)
# Define loss and optimizer
```

```
criterion = nn.BCELoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
# Ensure labels is a 1D tensor
labels = labels.view(-1)
# Train the model
num_epochs = 10
for epoch in range(num_epochs):
    optimizer.zero_grad()
    outputs = model(inverse_indices)
    outputs = outputs.squeeze(dim=-1)  # Remove singleton dimension
    loss = criterion(outputs, labels)
    loss.backward()
    optimizer.step()
    print(f'Epoch [{epoch + 1}/{num_epochs}],Loss: {loss.item():.4f}')
```

LISTING 1. TransformerModel

DEPARTMENT OF MATHEMATICS AND STATISTICS, OAKLAND UNIVERSITY, ROCHESTER, MI, 48309
Email address: elirashaska@oakland.edu

DEPARTMENT OF MATHEMATICS AND STATISTICS, OAKLAND UNIVERSITY, ROCHESTER, MI, 48309
Email address: shaska@risat.org