C. Maistret, A. V. Sutherland *Res. Number Theory* (2025) 11:37
https://doi.org/10.1007/s40993-024-00605-7

Research in Number Theory

RESEARCH

# Computing Euler factors of genus 2 curves at odd primes of almost good reduction

Céline Maistret[1*] and Andrew V. Sutherland[2]

*Correspondence:
celine.maistret@bristol.ac.uk
[1]School of Mathematics, University of Bristol, Bristol BS8 1UG, UK
Full list of author information is available at the end of the article

**Abstract**

We present an efficient algorithm to compute the Euler factor of a genus 2 curve $C/\mathbb{Q}$ at an odd prime $p$ that is of bad reduction for $C$ but of good reduction for the Jacobian of $C$ (a prime of "almost good" reduction). Our approach is based on the theory of cluster pictures introduced by Dokchitser, Dokchitser, Maistret, and Morgan, which allows us to reduce the problem to a short, explicit computation over $\mathbb{Z}$ and $\mathbb{F}_p$, followed by a point-counting computation on two elliptic curves over $\mathbb{F}_p$, or a single elliptic curve over $\mathbb{F}_{p^2}$. A key feature of our approach is that we avoid the need to compute a regular model for $C$. This allows us to efficiently compute many examples that are infeasible to handle using the algorithms currently available in computer algebra systems such as Magma and Pari/GP.

## 1 Introduction

The $L$-functions and modular forms database (LMFDB) [21] currently contains a database of 66,158 genus 2 curves $C/\mathbb{Q}$ with absolute discriminant bounded by $10^6$ whose computation is described in [4]. The bound on the discriminant is motivated by the fact that it also serves as a bound on the conductor of the $L$-function of $C$. In the context of the LMFDB one naturally organizes objects according to the conductor of their $L$-function, since this invariant determines the level of the corresponding modular form predicted by the Langlands program (in the case of genus 2 curves, a Siegel modular form that is generically a paramodular form). One prioritizes examples of small conductor, because this makes it more feasible to compute their $L$-functions, and to identify other objects in the LMFDB (including modular forms) that have the same $L$-function.

But bounding the discriminant is overly restrictive: genus 2 curves of small conductor need not have small discriminant. This is also true of elliptic curves, but becomes much more pronounced in higher genus due to the possibility that the minimal discriminant of a curve may be divisible by primes $p$ that do not divide the conductor of its $L$-function, which cannot happen in genus 1. This necessarily happens when the Jacobian of $C$, reduces to a product of elliptic curves over $\mathbb{F}_p$, which forces the curve to have bad reduction. This is not an uncommon occurrence, and may happen even when the Jacobian is geometrically simple. We call a prime of bad reduction for $C$ that does not divide the conductor of its $L$-function a prime of almost good reduction for $C$.

© The Author(s) 2025. This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

There is work in progress to expand the genus 2 curve database in the LMFDB to include more than five million genus 2 curves over $\mathbb{Q}$ with conductor below $2^{20} \approx 10^6$, and more than half of these curves have almost good reduction at some prime, including almost half a million that have geometrically simple Jacobians [20]. The primes $p$ involved need not be small and may substantially exceed the conductor bound. For example, the curve

$$\begin{aligned} C : y^2 + (x^3 + x^2 + x)y = &-144061786290072x^6 - 23062462482396x^5 \\ &- 1266273619292236x^4 - 3052943051575761x^3 \\ &+ 3989955132045666x^2 + 50048078951052415x \\ &- 24854569174209566 \end{aligned}$$

has prime conductor 270761 and minimal discriminant $270761 \cdot 14556001^{22}$, with $p = 14{,}556{,}001$ a prime of almost good reduction, and there are examples with conductor less than $2^{20}$ that have primes of almost good reduction as large as 43,858,540,753.

None of the computer algebra systems Magma [2], Pari/GP [22], or SageMath [23] is currently able to compute the Euler factor at $p = 14{,}556{,}001$ for the $L$-function of the curve $C$ listed above, and the method for computing bad Euler factors using the approximate functional equation described in [4] cannot be feasibly applied here. In this example $p$ is so much larger than the conductor that the inability to compute the Euler factor is not critical (one can approximate special values and bound the analytic rank without it), but there are hundreds of thousands of cases with $p$ close to the square root of the conductor, a regime where it is difficult to accurately approximate the $L$-function without knowing the Euler factor at $p$ and also difficult to guess the correct Euler factor and heuristically check it using the functional equation. Moreover, a single genus 2 curve may have multiple primes of almost good reduction, which complicates matters further. There are several curves in the new dataset that have five or more primes of almost good reduction, all of which are small enough to have a major impact on the functional equation, and the number of possibilities for the bad Euler factors is far too large to make heuristic checking feasible. These almost good primes have proven to be a major obstacle to attempts to expand the database of genus 2 curves in the LMFDB, which is what motivated the work we present here. Our main result is the following.

**Theorem 1.1** *Let $C/\mathbb{Q}$ be a genus 2 curve $y^2 = f(x) = \sum_i f_i x^i \in \mathbb{Z}[x]$ with almost good reduction at an odd prime $p$. There is a deterministic algorithm that, given a nonsquare element of $\mathbb{F}_p^\times$, computes the $L$-polynomial $L_p(C, T)$ in time*

$$O(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p),$$

*where $\|f\| = \max_i \log |f_i|$. There is also a Las Vegas algorithm with the same expected running time that does not require a nonsquare element of $\mathbb{F}_p^\times$ to be provided as part of the input.*

For large $p$ the running time of this algorithm is dominated by the time to count points on elliptic curves using Schoof's algorithm. When $p$ is small and the power of $p$ dividing the discriminant is bounded, the running time is quasi-linear in $\|f\|$, which is the size of the input. The algorithm is described in detail in Sect. 4, and it is easy to implement. A simple implementation of the new algorithm in Magma is available in the GitHub

repository associated to this paper [13], and it is already a dramatic improvement over the EulerFactor function implemented in Magma.[1] It took roughly 242 CPU days to compute Euler factors at approximately 3.5 million primes of almost good reduction arising in our small conductor dataset using the EulerFactor function, except for 489 cases where the computation did not terminate within 8 hours, an average of about 6 seconds per Euler factor. By contrast, a simple Magma implementation of our new algorithm takes less than 1.3 CPU hours to compute every Euler factor, averaging close to one millisecond per Euler factor with a maximum time of less than 25 milliseconds, including the 489 examples we were not able to compute using the EulerFactor intrinsic. A low level C implementation of the new algorithm takes only 30 CPU seconds to accomplish the same task (about 8 microseconds per Euler factor). See Sect. 5 and Tables 2, 3, 4 and 5 for further details of our implementation and the tests we ran.

Our algorithm does not treat curves with almost good reduction at the prime $p = 2$, even though there are many such examples, notably including the modular curve $X_0(22)$ whose Jacobian has conductor $11^2$, the smallest conductor possible for an abelian surface. But our algorithm is still helpful in treating these cases because it makes it feasible to apply the methods of [4] that can efficiently use the functional equation to compute the Euler factor at 2, provided the Euler factors at all larger primes are known.

We expect that the cluster picture approach we use here can be extended to other types of bad reduction for genus 2 curves, at least in a semistable setting; this is an area for future work. We should note that while our algorithms could conceivably be extended to handle almost good primes of genus 3 hyperelliptic curves, our method does not scale well with the genus. There is recent work on new methods for computing regular models, notably that of Dokchitser [7] and Muselli [15] that is likely to handle higher genus curves better than our approach, and may also allow one to treat other reduction types, with similar efficiency.

## 2 Background

### 2.1 Euler factors

Let $C$ be a smooth projective curve of genus $g \geq 2$ over $\mathbb{Q}$. The $L$-function of $C$ is defined as the Euler product

$$L(C, s) = \prod_p L_p(C, p^{-s})^{-1},$$

where the $L$-polynomial $L_p \in \mathbb{Z}[T]$ is given by

$$L_p(C, T) \coloneqq \det\left(1 - T\,\mathrm{Frob}_p^{-1} | H^1_{\text{ét}}(C \otimes_{\mathbb{Q}} \mathbb{Q}^{\text{alg}}, \mathbb{Q}_\ell)^{I_p}\right),$$

and has degree $2g$ at primes of good reduction for the Jacobian of $C$, including primes of almost good reduction for $C$, and otherwise has degree strictly less than $2g$.

Here $\mathrm{Frob}_p$ denotes an arithmetic Frobenius element, $I_p$ is the inertia subgroup of a decomposition group $D_p \subset \mathrm{Gal}(\mathbb{Q}^{\text{alg}}/\mathbb{Q})$ and $\ell$ is a prime distinct from $p$.

[1] We compare our algorithm to the EulerFactor intrinsic in Magma because it is the only implementation we are aware of that gives correct results when the computation terminates without reporting an error; neither Pari/GP nor SageMath currently support the computation of Euler factors for genus 2 curves at primes of almost good reduction.

**Proposition 2.1** *Let $C/\mathbb{Q}$ be a genus 2 curve and let $p$ be a prime of almost good reduction for $C$. Let $\mathcal{C}$ be a $\mathbb{Z}_p$-model which is regular, and let $\overline{C}$ denote its special fiber. Then*

$$H^1_{\text{ét}}(C \otimes_{\mathbb{Q}} \mathbb{Q}^{\text{alg}}, \mathbb{Q}_\ell)^{I_p} \cong H^1_{\text{ét}}(\overline{C}_{\overline{\mathbb{F}}_p}, \mathbb{Q}_\ell).$$

*Proof* Note that all the components of $\overline{C}$ have multiplicity 1, since $C$ has semistable reduction at $p$. The proposition then follows directly from Proposition 2.8 in [3]. □

One can compute the Euler factor $L_p(C, T)$ via its connection to the zeta function of $\overline{C}/\mathbb{F}_p$, although we note that this is not the only possible approach; see Section 1.1 of [3].

**Definition 2.2** Let $X/\mathbb{F}_p$ be a curve. We define its zeta function by

$$Z(X/\mathbb{F}_p, T) = \exp\left(\sum_{n\geq 1} |X(\mathbb{F}_{p^n})| \frac{T^n}{n}\right).$$

**Theorem 2.3** *Let $\overline{C}/\mathbb{F}_p$ be the special fiber of a $\mathbb{Z}_p$-regular model of $C/\mathbb{Q}$. Then*

$$Z(\overline{C}, T) = \frac{P_1(T)}{P_0(T)P_2(T)},$$

*where $P_i(T) = \det(1 - T\text{Frob}_p^{-1} | H^i_{\text{ét}}(\overline{C}_{\overline{\mathbb{F}}_p}, \mathbb{Q}_\ell)$ for $i = 0, 1, 2$ and some prime $\ell \neq p$. In particular, $P_1(T)$ is the L-polynomial $L_p(C, T)$.*

*Proof* This is Theorem 13.1 in [14] with $X = \overline{C}$, combined with Proposition 2.1. □

It follows from Theorem 2.3 that one can compute $L_p(C, T)$ by computing the zeta function of the special fiber of a $\mathbb{Z}_p$-regular model of $C/\mathbb{Q}_p$. However, this may be computationally expensive; indeed, in the 489 cases mentioned in the introduction where Magma struggled to compute Euler factors at primes of almost good reduction, the difficulties arose while computing a regular model.

We will use the theory of cluster pictures to avoid computing a regular model. For a general exposition of the theory of cluster pictures, we refer the interested reader to [1], where explicit examples are used to illustrate theoretical points. In particular, [1, Section 6] provides examples of construction of special fibers from cluster pictures.

### 2.2 Cluster pictures

Let $K$ be a local field of odd residue characteristic $p$, let $v$ be a normalized valuation with respect to $K$, and let $C/K$ a hyperelliptic curve of genus $g \geq 2$ given by

$$y^2 = f(x) = c\prod_{r\in\mathcal{R}}(x - r),$$

where $f \in K[x]$ is separable with $\deg(f) = 2g + 1$ or $2g + 2$, and where $\mathcal{R}$ denotes the set of roots of $f(x)$ in $K^{\text{sep}}$.

**Definition 2.4** (Clusters and cluster pictures) A cluster is a non-empty subset $\mathfrak{s} \subseteq \mathcal{R}$ of the form $\mathfrak{s} = D \cap \mathcal{R}$ for some disc $D = \{x \in K^{\text{alg}} \mid v(x - z) \geq d\}$ for some $z \in K^{\text{alg}}$ and $d \in \mathbb{Q}$.

For a cluster $\mathfrak{s}$ with $|\mathfrak{s}| > 1$, its depth $d_\mathfrak{s}$ is the maximal $d$ for which $\mathfrak{s}$ is cut out by such a disc, that is $d_\mathfrak{s} = \min_{r,r'\in\mathfrak{s}} v(r - r')$. If moreover $\mathfrak{s} \neq \mathcal{R}$, we define the parent cluster $P(\mathfrak{s})$ of $\mathfrak{s}$ to be the smallest cluster with $\mathfrak{s} \subsetneq P(\mathfrak{s})$. Then the relative depth of $\mathfrak{s}$ is $\delta_\mathfrak{s} = d_\mathfrak{s} - d_{P(\mathfrak{s})}$.

We refer to this data as the cluster picture of $C$.

*Remark 2.5* The absolute Galois group of $K$ acts on clusters via its action on the roots. This action preserves depths and containments of clusters.

**Definition 2.6** If $\mathfrak{s}' \subsetneq \mathfrak{s}$ is a maximal subcluster, we refer to $\mathfrak{s}'$ as a child of $\mathfrak{s}$.

For two clusters (or roots) $\mathfrak{s}_1, \mathfrak{s}_2$ write $\mathfrak{s}_1 \wedge \mathfrak{s}_2$ for the smallest cluster containing them.

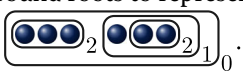
**Definition 2.7** A cluster $\mathfrak{s}$ is principal except when:

- $|\mathfrak{s}| \leq 2$, or
- $\mathfrak{s}$ has a child of size $2g$, or
- $\mathfrak{s} = \mathcal{R}$ is even and has exactly two children.

**Definition 2.8** For a cluster $\mathfrak{s}$ set

$$v_{\mathfrak{s}} = v(c) + \sum_{r \in \mathcal{R}} d_{r \wedge \mathfrak{s}}.$$

**Notation 2.9** We draw cluster pictures by drawing roots $r \in \mathcal{R}$ as ●, and we draw ovals around roots to represent clusters (of size $> 1$), such as:

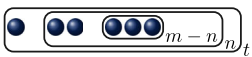


The subscript on the largest cluster $\mathcal{R}$ is its depth, while the subscripts on the other clusters are relative depths.

*Example 2.10* Consider the genus 2 curve $C : y^2 = x(x-p^2)(x-3p^2)(x-1)(x-1+p^4)(x-1-p^4)$ where $p \geq 5$ is a prime. Its cluster picture at $p$ is given by

## 3 Cluster pictures and special fibers

**Proposition 3.1** *Let $C/\mathbb{Q}$ be a genus 2 curve and $p$ an odd prime of almost good reduction for $C$. Then the possible cluster pictures for $C$ at $p$ are*

1) , 2) , 3) , 4)  ,

*for some integers $m, n, t$, with $m \geq n$ in case 2) and $m > n$ in case 4), such that the following congruence relations are satisfied:*
*$v_p(c)$ and $n$ are even in cases 1),*
*$v_p(c) \equiv n \equiv m \bmod 2$ in cases 2) and 4),*
*$v_p(c) \equiv t \bmod 2$ and $n$ is even in cases 3).*

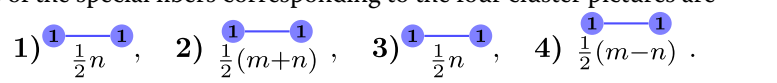
*Proof* This is Theorem 10.3 (5) in [6] which implies that clusters $\mathfrak{s} \neq \mathcal{R}$ in the picture must contain an odd number of roots. This gives the classification of pictures above. The parity of the valuation of the leading term and the depths follow from the fact that $v_{\mathfrak{s}} \in 2\mathbb{Z}$ for principal clusters. □

*Remark 3.2*

1. The assumption that $p$ is a prime of almost good reduction for $C$ is essential in Proposition 3.1, as it controls the parity of $v_p(c)$. Changing the parity of $v_p(c)$ amounts to taking a quadratic twist, which does not change the cluster picture but will force Jac $C$ to have bad reduction at $p$. Our algorithms will produce the same output given

$C$ or its quadratic twist by $p$, but in the latter case the input will not satisfy the necessary assumption that the Jacobian has good reduction at $p$; our algorithms cannot be used to compute the $L$-polynomial of the quadratic twist.

2. The assumption that $p$ is a prime of good reduction for Jac $C$ implies that the splitting field $L$ of $f$ is unramified (Theorem 10.3 (5) in [6]). It follows that $p$ is a uniformizer for $L$ and that the normalized valuation $v$ extends the valuation $v_p$ of $K = \mathbb{Q}_p$ with index 1.
3. The reduction type for $C$ in Proposition 3.1 corresponds to Namikawa-Ueno type $I_0 - I_0 - r$ listed in [6] Table 18.2. One can check using Theorem 8.6 in [6] that the dual graphs of the special fibers corresponding to the four cluster pictures are




   In the drawings above, vertices represent genus 1 components. These are linked by chains of $\frac{1}{2}n$, $\frac{1}{2}(m+n)$, $\frac{1}{2}n$, and $\frac{1}{2}(m-n)$ edges respectively, representing genus 0 components in the special fiber.
4. When $n = m$ in picture **2)** of Proposition 3.1, the Frobenius automorphism will permute the two clusters of size 3 if their centers are defined over the unramified quadratic extension of $\mathbb{Q}_p$ (and therefore are Galois conjugates). If this is the case we will draw a line between the two clusters in the picture as so:

   Note that in this case, the depths are necessarily equal.

*Remark 3.3*

1. We can assume without loss of generality that $C : y^2 = f(x)$ is defined by a polynomial $f \in \mathbb{Z}[x]$ of degree 6. This will ensure that the cluster picture of $C$ is never of type **3)**.
2. The depth $d_{\mathcal{R}} = t$ of the outer cluster in types **1)**, **2)**, **4)** of Proposition 3.1 can be set to $t = 0$. This is the first step of our algorithm (see Definition 3.4 and Corollary 3.5 below).
3. We can assume without loss of generality that $v_p(c)$ is 0 or 1, and in the latter case that every coefficient of $f \in \mathbb{Z}[x]$ is divisible by $p$.

We will give a constructive proof of the claims made in Remark 3.3 when we present our algorithms in the next section, but they motivate the following definition.

**Definition 3.4** Let $p$ be an odd prime. A squarefree sextic polynomial $f = \sum_i f_i x^i \in \mathbb{Z}[x]$ defining a genus 2 curve $C : y^2 = f(x)$ with almost good reduction at $p$ that satisfies $d_{\mathcal{R}} = t = 0$ and $v_p(f_6) = \min_i\{v_p(f_i)\} \leq 1$ is said to be *p*-normalized.

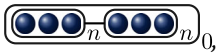
**Corollary 3.5** *Let $p$ be an odd prime and let $C : y^2 = f(x)$ be a genus 2 curve defined by a $p$-normalized polynomial $f \in \mathbb{Z}[x]$. Then the cluster picture for $C$ is one of the following:*

*Type **1**:=*

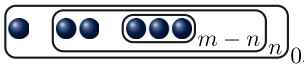
*where $v_p(c) = 0 \equiv n \bmod 2$,*

*Type **2a**:=*

*where $m \geq n$ and $v_p(c) \equiv m \equiv n \bmod 2$ with $v_p(c) \leq 1$,*

*Type **2b**:=* 

*where $v_p(c) \equiv n \bmod 2$ with $v_p(c) \leq 1$,*

*Type **4**:=* 

*where $m > n$ and $v_p(c) \equiv m \equiv n \bmod 2$ with $v_p(c) \leq 1$.*

**Proposition 3.6** *Let $p$ be an odd prime and let $C\colon y^2 = f(x)$ be a genus 2 curve defined by a $p$-normalized polynomial $f \in \mathbb{Z}[x]$, let $\tilde{f} = p^{-v_p(c)}f \in \mathbb{Z}[x]$, let $\bar{f} = \tilde{f} \bmod p \in \mathbb{F}_p[x]$, and let $\bar{c} = cp^{-v_p(c)} \bmod p \in \mathbb{F}_p$. Then depending on the type of the cluster picture of $C$ at $p$, exactly one of the following holds:*

***1)*** *$\bar{f} = \bar{c}\,(x-\bar{r})^3\bar{u}$ for some squarefree monic cubic $\bar{u} \in \mathbb{F}_p[x]$ and $\bar{r} \in \mathbb{F}_p$ with $\bar{u}(\bar{r}) \neq 0$.*
***2a)*** *$\bar{f} = \bar{c}\,(x-\bar{r})^3(x-\bar{s})^3$ for some distinct $\bar{r}, \bar{s} \in \mathbb{F}_p$.*
***2b)*** *$\bar{f} = \bar{c}\,\bar{u}^3$ for some irreducible monic quadratic $\bar{u} \in \mathbb{F}_p[x]$.*
***4)*** *$\bar{f} = \bar{c}\,(x-\bar{r})^5(x-\bar{s})$ for some distinct $\bar{r}, \bar{s} \in \mathbb{F}_p$.*

*Proof* Let $\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3$ denote the roots of $f(x)$ ordered left to right in the cluster picture.

Types **1**: Here $\beta_1, \beta_2, \beta_3$ denote the roots in the cluster of depth $n$. Since $v(\beta_i - \beta_j) = n$ for $i \neq j = 1, 2, 3$, we have that $\beta_1 \equiv \beta_2 \equiv \beta_3 \bmod p$, which gives the factor $(x-\bar{r})^3$. Now $v(\alpha_i - \alpha_j) = 0$ so that $\alpha_i \not\equiv \alpha_j \bmod p$ for $i \neq j = 1, 2, 3$. Moreover, $v(\alpha_i - \beta_j) = 0$ for $i, j = 1, 2, 3$, therefore $\alpha_i \not\equiv \beta_j$ for $i, j = 1, 2, 3$. It follows that $\bar{u} = (x-\overline{\alpha_1})(x-\overline{\alpha_2})(x-\overline{\alpha_3})$ is a square free cubic in $\mathbb{F}_p[x]$ with $\bar{u}(\bar{r}) \neq 0$. Types **2a** and **4** follow from a similar argument.

Type **2b**: Here the Frobenius automorphism permutes the $\alpha$s and $\beta$s pairwise. In particular, it has order 2 so that $\tilde{f}$ splits into two cubic polynomials $f_1, f_2$ over the unramified quadratic extension $F$ of $\mathbb{Q}_p$.

From the cluster picture we have that $\alpha_1 \equiv \alpha_2 \equiv \alpha_3 \bmod p\mathcal{O}_F$ and similarly for the $\beta$s. It follows that $\tilde{f}(x) \equiv (x-\lambda)^3(x+\lambda)^3 \bmod p^2$ for some $\lambda \in \mathbb{F}_{p^2}$. Let $\Lambda \in \mathbb{F}_p$ with $\lambda^2 = -\Lambda$ so that $\tilde{f}(x) \equiv (x^2+\Lambda)^3 \bmod p$. □

**Corollary 3.7** *Let $C/\mathbb{Q}$ be a genus 2 curve and $p$ an odd prime of almost good reduction for $C$. Then the special fiber $\overline{C}/\mathbb{F}_p$ of the minimal regular model of $C/\mathbb{Q}_p$ consists of a union of two elliptic curves linked by a chain of $\mathbb{P}^1$s.*

*Proof* This follows directly from the list of possible cluster pictures given in Proposition 3.1 and [6, Definition 8.5 and Theorem 8.6], where explicit models for the components of the special fibers are constructed. One can then check that the components associated to principal clusters are indeed elliptic curves (see the proof of Corollary 3.10 for an example of such a construction). Alternatively, since Jac $C$ has good reduction, it follows from the work of Raynaud ( [16], [8, Section 9]) that its Néron model is an abelian scheme. Since the identity component of the special fiber of the Néron model is $\mathrm{Pic}^0\overline{C}$, this forces $\overline{C}$ to be a genus 2 curve or a union of two elliptic curves. The former is a contradiction to the prime $p$ being of bad reduction for $C$. □

*Remark 3.8* For cluster pictures of type **2b** in Corollary 3.5, the elliptic curves are defined over $\mathbb{F}_{p^2}$ and permuted by Frobenius, hence have the same $L$-polynomial.

Let $E_1$ and $E_2$ be the two elliptic curves in $\overline{C}$ given by Corollary 3.7. We now show how to compute $L_p(C, T)$ from $L_p(E_1, T)$ and $L_p(E_2, T)$.

**Proposition 3.9** *Let $C/\mathbb{Q}$ be a genus 2 curve, $p$ an odd prime of almost good reduction for $C$, and $r \geq 0$ an integer. Let $E_1$ and $E_2$ be the two elliptic curves linked by a chain of $\mathbb{P}^1$s of length $r$, whose union forms the special fiber of the minimal regular model of $C/\mathbb{Q}_p$ as given by Corollary 3.7. Then*

*(1)* $L_p(C, T) = L_p(E_1, T)L_p(E_2, T)$ *if both* $E_1$ *and* $E_2$ *are defined over* $\mathbb{F}_p$.
*(2)* $L_p(C, T) = L_p(E_1/\mathbb{F}_{p^2}, T^2) = L_p(E_2/\mathbb{F}_{p^2}, T^2)$ *if both* $E_1$ *and* $E_2$ *are defined over* $\mathbb{F}_{p^2}$.

*Proof* (1) Suppose that $E_1$ and $E_2$ are defined over $\mathbb{F}_p$. It follows that each individual $\mathbb{P}^1$ in the chain linking $E_1$ to $E_2$ is also defined over $\mathbb{F}_p$. Therefore $P_2(T) = (1 - pT)^{2+r}$, by definition, since each component of $\overline{C}$ contributes a factor $(1 - pT)$.

Let $\alpha_1, \beta_1, \alpha_2, \beta_2$ be algebraic numbers such that $L_p(E_i, T) = (1 - \alpha_i T)(1 - \beta_i T)$ for $i = 1, 2$. In particular, for all $n \in \mathbb{Z}_{>0}$, $|E_i(\mathbb{F}_{p^n})| = p^n + 1 - \alpha_i^n - \beta_i^n$. It follows that

$$|\overline{C}(\mathbb{F}_{p^n})| = |E_1(\mathbb{F}_{p^n})| + |E_2(\mathbb{F}_{p^n})| + r|\mathbb{P}_1(\mathbb{F}_{p^n})| - (r + 1),$$

where we removed $r + 1$ intersection points that were counted twice. Therefore

$$\begin{aligned}|\overline{C}(\mathbb{F}_{p^n})| &= (2 + r)(p^n + 1) - \alpha_1^n - \beta_1^n - \alpha_2^n - \beta_2^n - (r + 1)\\ &= (2 + r)p^n - \alpha_1^n - \beta_1^n - \alpha_2^n - \beta_2^n + 1,\end{aligned}$$

and we have

$$\begin{aligned}Z(\overline{C}/\mathbb{F}_p, T) &= \exp\left(\sum_{n\geq 1}((2 + r)p^n + 1 - \alpha_1^n - \beta_1^n - \alpha_2^n - \beta_2^n)\frac{T^n}{n}\right)\\ &= \frac{(1 - \alpha_1 T)(1 - \beta_1 T)(1 - \alpha_2 T)(1 - \beta_2 T)}{(1 - T)(1 - pT)^{2+r}} = \frac{L_p(E_1, T)L_p(E_2, T)}{(1 - T)(1 - pT)^{2+r}}.\end{aligned}$$

The result then follows from Theorem 2.3.
(2) Suppose that $E_1$ and $E_2$ are defined over $\mathbb{F}_{p^2}$. We have two cases.
(i) If $r$ is even, then all $\mathbb{P}^1$s in the chain linking $E_1$ and $E_2$ are defined over $\mathbb{F}_{p^2}$ and are permuted by Frobenius. Therefore
$P_2(T) = (1 - pT)^{\frac{2+r}{2}}(1 + pT)^{\frac{2+r}{2}}$.
Let $\alpha, \beta$ be algebraic numbers such that $L_p(E_1, T) = (1 - \alpha T)(1 - \beta T)$. In particular,

$$|\overline{C}(\mathbb{F}_{p^n})| = \begin{cases}1, & \text{if } n \text{ is odd,}\\ (2 + r)p^n + 1 - 2\alpha^n - 2\beta^n, & \text{if } n \text{ is even.}\end{cases}$$

If we put $m = (2 + r)/2$, we then have

$$\begin{aligned}Z(\overline{C}/\mathbb{F}_p, T) &= \exp\left(\sum_{n\geq 1}\left(mp^n + (-1)^n mp^n + 1 - \alpha^n - (-\alpha)^n - \beta^n - (-\beta)^n\right)\frac{T^n}{n}\right)\\ &= \frac{(1 - \alpha T)(1 + \alpha T)(1 - \beta T)(1 + \beta T)}{(1 - T)(1 - pT)^m(1 + pT)^m}\\ &= \frac{L_p(E_1/\mathbb{F}_{p^2}, T^2)}{(1 - T)(1 - pT)^m(1 + pT)^m},\end{aligned}$$

and the result follows from Theorem 2.3.

(ii) If $r$ is odd, say $r = 2k + 1$, then the $2k$ $\mathbb{P}^1$s in the chain linking $E_1$ and $E_2$ are defined over $\mathbb{F}_{p^2}$ and permuted pairwise, while the "central" $\mathbb{P}^1$ is defined over $\mathbb{F}_p$.

It follows that we have $P_2(T) = (1 - pT)^{\frac{2+2k}{2}}(1 + pT)^{\frac{2+2k}{2}}(1 - pT)$.

As in (i), let $\alpha$, $\beta$ be algebraic numbers such that $L_p(E_1, T) = (1-\alpha T)(1-\beta T)$. It follows that

$$|\overline{C}(\mathbb{F}_{p^n})| = \begin{cases} p^n + 1, & \text{if } n \text{ is odd,} \\ (2k+2)p^n + p^n - 2\alpha^n - 2\beta^n - (2k+2), & \text{if } n \text{ is even.} \end{cases}$$

Therefore

$$\begin{aligned} Z(\overline{C}/\mathbb{F}_p, T) &= \exp\left(\sum_{n\geq 1}((k+1)p^n + (-1)^n(k+1)p^n + p^n \right. \\ &\qquad \left. +1 - \alpha^n - (-\alpha)^n - \beta^n - (-\beta)^n)\frac{T^n}{n}\right) \\ &= \frac{(1-\alpha T)(1+\alpha T)(1-\beta T)(1+\beta T)}{(1-T)(1-pT)^{k+1}(1+pT)^{k+1}(1-pT)} \\ &= \frac{L_p(E_1/\mathbb{F}_{p^2}, T^2)}{(1-T)(1-pT)^{k+1}(1+pT)^{k+1}(1-pT)}, \end{aligned}$$

and the result follows from Theorem 2.3. □

**Corollary 3.10** *Let $C: y^2 = f(x)$ and $p$ be as in Proposition 3.6 with type **1**, **2a**, **4**, and let $\tilde{f} = p^{-v_p(c)}f \in \mathbb{Z}[x]$. Let $L$ be the splitting field of $f$ over $\mathbb{Q}_p$, let $r_1 \in \mathcal{O}_L$ be a root in the cluster of depth $n$, and for types **2a**, **4** let $r_2 \in \mathcal{O}_L$ be a root in the cluster of depth $m$. Fix $s_1 \in \mathbb{Z}$ with $r_1 \equiv s_1 \bmod p^n\mathcal{O}_L$, and for types **2a**, **4**, $s_2 \in \mathbb{Z}$ with $r_2 \equiv s_2 \bmod p^m\mathcal{O}_L$. The elliptic curves $E_1/\mathbb{F}_p$ and $E_2/\mathbb{F}_p$ of Proposition 3.9 may be explicitly computed as follows:*

- ***1)*** *$E_1: y^2 = \bar{g}_1(x)$, where $\bar{g}_1 \in \mathbb{F}_p[x]$ is the squarefree part of $\tilde{f} \bmod p$, and*
  *$E_2: y^2 = \bar{g}_2(x)$, where $\bar{g}_2(x) = \tilde{f}(p^n x + s_1)/p^{3n} \bmod p \in \mathbb{F}_p[x]$.*
- ***2a)*** *$E_1: y^2 = \bar{g}_1(x)$, where $\bar{g}_1(x) = \tilde{f}(p^n x + s_1)/p^{3n} \bmod p \in \mathbb{F}_p[x]$, and*
  *$E_2: y^2 = \bar{g}_2(x)$, where $\bar{g}_2(x) = \tilde{f}(p^m x + s_2)/p^{3m} \bmod p \in \mathbb{F}_p[x]$.*
- ***4)*** *$E_1: y^2 = \bar{g}_1(x)$, where $\bar{g}_1 \in \mathbb{F}_p[x]$ is the squarefree part of $\tilde{f}(p^n x + s_1)/p^{5n} \bmod p$, and*
  *$E_2: y^2 = \bar{g}_2(x)$, where $\bar{g}_2(x) = \tilde{f}(p^m x + s_2)/p^{3m+2n} \bmod p$.*

*Proof* This follows from Definition 8.5 and Theorem 8.6 in [6], where explicit models for the components associated to principal clusters are constructed. We explicitly work out the case of Type **1**, the other types are similar. There are two principal clusters in the cluster picture of Type **1**: the full set of roots $\mathcal{R} = \{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3\}$ ordered from left to right in the cluster picture, and the cluster of depth $n$, say $\mathfrak{s}$. We now follow Definition 8.5 in [6], which associates a component to each principal cluster. Recall that since $C$ is semistable, $v_p(c) \in 2\mathbb{Z}$ (Definition 1.8 and Theorem 1.9 in [6]), and therefore $v_p(c) = 0$ by our assumption. For a root $r$, let $\bar{r}$ denote $r \bmod p\mathcal{O}_L$. We have

$$\Gamma_{\mathcal{R}} : Y^2 = \bar{c}(X - \bar{\alpha_1})(X - \bar{\alpha_2})(X - \bar{\alpha_3})(X - \bar{\beta_1}).$$

This gives $E_1/\mathbb{F}_p$ since $f \equiv \bar{c}(X-\bar{\alpha_1})(X-\bar{\alpha_2})(X-\bar{\alpha_3})(X-\bar{\beta_1})^3 \bmod p$. Note that $\bar{\beta}_1 \in \mathbb{F}_p$ but that $\bar{\alpha}_1, \bar{\alpha}_2, \bar{\alpha}_3$ may be permuted by Frobenius.

For $E_2$, we look at the component associated to $\mathfrak{s}$. Choose $z_{\mathfrak{s}} = r_1$. We have $v(r_1 - \alpha_i) = 0$ for $i = 1, 2, 3$, thus $c_{\mathfrak{s}} = \bar{c}(s_1 - \bar{\alpha_1})(s_1 - \bar{\alpha_2})(s_1 - \bar{\alpha_3})$. Therefore

$$\Gamma_{\mathfrak{s}} : Y^2 = \bar{c}\,(s_1 - \bar{\alpha}_1)(s_1 - \bar{\alpha}_2)(s_1 - \bar{\alpha}_3)\left(X - \frac{\overline{\beta_1 - r_1}}{p^n\mathcal{O}_L}\right)\left(X - \frac{\overline{\beta_2 - r_1}}{p^n\mathcal{O}_L}\right)$$

$$\times \left( X - \overline{\frac{\beta_3 - r_1}{p^n \mathcal{O}_L}} \right),$$

which is $f(p^n x + s_1)/p^{3n} \bmod p$ as claimed. □

**Corollary 3.11** *Let $C\colon y^2 = f(x)$ and $p$ be as in Proposition 3.6 with type **2b**, let $\tilde{f} = p^{-v_p(c)} f$, and let $L$ be the splitting field of $f$ over $\mathbb{Q}_p$. Let $u \in \mathbb{Z}[x]$ be any lift of the irreducible quadratic $\bar{u} \in \mathbb{F}_p[x]$ in case **2b** of Proposition 3.6, and define $F := \mathbb{Q}_p[z]/(u(z)) \subseteq L$, $\mathcal{O} := \mathbb{Z}[z]/(u(z)) \subseteq \mathcal{O}_L$, and $\kappa := \mathbb{F}_p[z]/(\bar{u}(z)) \simeq \mathbb{F}_{p^2}$. Let $r \in \mathcal{O}_L$ be a root of $f$, and let $s \in \mathcal{O}$ satisfy $r \equiv s \bmod p\mathcal{O}_L$. Let $\hat{f}$ denote the image of $f$ in $\mathcal{O}[x]$ via the inclusion $\mathbb{Z}[z] \subseteq \mathcal{O}[z]$ induced by $\mathbb{Z} \subseteq \mathcal{O}$, and let $\tilde{g} = \hat{f}(p^n x + s)/p^{3n} \bmod p\mathcal{O} \in \kappa[x] \simeq \mathbb{F}_{p^2}[x]$. Then $y^2 = \tilde{g}(x)$ and its $\mathrm{Gal}(\kappa/\mathbb{F}_p)$-conjugate are models for the elliptic curves $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$ of Proposition 3.9.*

*Proof* Denote $\mathfrak{s}_1$ and $\mathfrak{s}_2$ the two clusters of size 3 in the picture. As for Type **1** in the proof above, we closely follow Definition 8.5 and Theorem 8.6 in [6]. Both $\mathfrak{s}_1$ and $\mathfrak{s}_2$ are principal. Since the Frobenius automorphism permutes both clusters, the associated components ($E_1$ and $E_2$) must be Galois conjugate as mentioned in Remark 3.8. We construct $E_1$ explicitly. Denote $\alpha_1, \alpha_2, \alpha_3$ the roots in $\mathfrak{s}_1$ and $\beta_1, \beta_2, \beta_3$ the roots in $\mathfrak{s}_2$. Permuting indices if necessary, Frobenius permutes $\alpha_i$ and $\beta_i$ for $i = 1, 2, 3$. We choose $r = z_{\mathfrak{s}_1}$ a center for $\mathfrak{s}_1$. In particular, $r$ is one of the $\alpha$s. Since $v(r - \beta_i) = 0$ for $i = 1, 2, 3$, we have $c_{\mathfrak{s}_1} = \tilde{c}(s - \bar{\beta}_1)(s - \bar{\beta}_2)(s - \bar{\beta}_3)$, where for a root $\beta$, we have $\bar{\beta} = \beta \bmod p\mathcal{O}_L$. It follows that $E_1/\mathbb{F}_{p^2}$ is given by

$$\Gamma_{\mathfrak{s}_1} : Y^2 = c_{\mathfrak{s}_1} \left( X - \overline{\frac{r - \alpha_1}{p^n \mathcal{O}_L}} \right) \left( X - \overline{\frac{r - \alpha_2}{p^n \mathcal{O}_L}} \right) \left( X - \overline{\frac{r - \alpha_3}{p^n \mathcal{O}_L}} \right),$$

which is $\hat{f}(p^n x + s)/p^{3n} \bmod p\mathcal{O}$ since $v(r - \alpha_i) = n$ for $i = 1, 2, 3$. Now the construction of $\Gamma_{\mathfrak{s}_2}$, which defines $E_2/\mathbb{F}_{p^2}$, is the same as that of $\Gamma_{\mathfrak{s}_1}$ with the roots $\alpha$s and $\beta$s swapped. Since Frobenius permute them pairwise, it follows that $E_2$ is given by the $\mathrm{Gal}(\kappa/\mathbb{F}_p)$-conjugate of $\tilde{g}(x)$. □

## 4 Algorithms

In this section we describe our algorithm for computing $L_p(C, T)$ for a genus 2 curve $C/\mathbb{Q}$ at an odd prime $p$ of almost good reduction; so $p$ divides the minimal discriminant $\Delta(C) \in \mathbb{Z}$ of $C$ but does not divide the conductor $N(C) \in \mathbb{Z}$ of its Jacobian. Computing the set of primes that satisfy this assumption is at least as hard as factoring $\Delta(C)$, a problem for which no polynomial-time algorithm is known, but there are efficient algorithms to compute $\Delta(C)$ [12] and the $p$-adic valuation of $N(C)$ at a given odd prime [11], and these have been widely implemented in computer algebra systems such as Magma [2], Pari/GP [22], and SageMath [23]. We shall henceforth assume that any prime $p$ provided as an input to our algorithms is a prime of almost good reduction for $C$.

As we will be working exclusively in rings of odd or zero characteristic, we may assume that $C$ is specified by an integral model of the form $y^2 = f(x)$, where $f \in \mathbb{Z}[x]$ is a squarefree polynomial of degree 5 or 6 whose discriminant $\Delta(f)$ is divisible by $p$. We will state complexity bounds for our algorithms in terms of the logarithmic height

$$\|f\| := \log \max_i \{|f_i|\}$$

**Table 1** Asymptotic complexity bounds for arithmetic operations on ring elements used by our algorithms

| Operation | Complexity | Algorithm/references |
|---|---|---|
| Addition/subtraction | $O(b)$ | Schoolbook algorithm [24] |
| Multiplication | $O(b \log b)$ | Fast integer multiplication [9] |
| Reduction modulo $p$ | $O(b \log b)$ | Fast Euclidean division [24] |
| Greatest common divisor | $O(b \log^2 b)$ | Fast GCD [24] |
| Legendre symbol $(\frac{\cdot}{p})$ | $O(b \log^2 b)$ | Fast binary GCD [5] |
| Inversion in $\mathbb{F}_p^\times$ | $O(b \log^2 b)$ | Fast extended GCD [24] |
| Square roots in $\mathbb{F}_p^\times$ given $s \notin \mathbb{F}_p^{\times 2}$ | $O(b^2 \log^2 b / \log\log b)$ | Fast Tonelli-Shanks [19] |
| Computing $L_p(E, T)$ for $E/\mathbb{F}_p$ or $E/\mathbb{F}_{p^2}$ | $O(b^5)$ | Schoof's algorithm [17,18] |

Here $b$ denotes the number of bits used to represent the inputs, all of which we represent using $O(1)$ integers (as $\|f\|$ tends to infinity). We use the $O(b \log b)$ bound on integer multiplication throughout

of the input polynomial $f \in \mathbb{Z}[x]$, and the logarithm of the prime $p$. The discriminant $\Delta(f)$ can be expressed as a homogeneous polynomial in the coefficients $f_i$ of degree 10. It follows that $\log|\Delta(f)| = O(\|f\|)$ and $\log p = O(\|f\|)$.

In our complexity analyses we will always count bit operations. For ease of reference we list asymptotic bit-complexity bounds for various operations used by our algorithms in Table 1, in which the parameter $b$ bounds the bit-sizes of the inputs. We assume throughout that elements of the finite field $\mathbb{F}_p$ are uniquely represented as integers in $[0, p-1]$, so that reduction from $\mathbb{Z}$ to $\mathbb{F}_p$ amounts to computing a remainder modulo $p$. We assume that elements of $\mathbb{F}_{p^2} \simeq \mathbb{F}_p[z]/(\bar{g}(z))$ are explicitly represented as linear polynomials $z$ that have been reduced modulo an irreducible quadratic polynomial $\bar{g} \in \mathbb{F}_p[x]$ that will be chosen by our algorithms. The bounds in Table 1 apply to operations on elements of $\mathbb{F}_p$, $\mathbb{F}_{p^2}$, $\mathbb{Z}$, and also to polynomials over these rings that have bounded degree (we will only consider polynomials of degree at most 6), as well as elements and polynomials of bounded degree over the ring $\mathcal{O} \coloneqq \mathbb{Z}[z]/(g(z))$, with $g \in \mathbb{Z}[x]$ an irreducible monic quadratic that we will use to represent elements of the monogenic order $\mathcal{O}$ in the quadratic field $\mathbb{Q}[z]/(g(z))$.

In the descriptions of our algorithms that follow we will frequently need to reduce elements of characteristic zero rings modulo $p$, and also to lift elements of characteristic $p$ rings to characteristic zero. In our implementations all elements of characteristic $p$ rings are represented as integers in $[0, p-1]$ or lists of such integers, so there is no actual computation involved in lifting, but when describing our algorithms we will use the notation

$$a = \operatorname{lift}(\bar{a})$$

to indicate that $a$ is the lift of $\bar{a}$ from a characteristic $p$ ring (such as $\mathbb{F}_p$, $\mathbb{F}_p[x]$, $\mathbb{F}_{p^2} \simeq \mathbb{F}_p[z]/(\bar{g}(z))$, $\mathbb{F}_{p^2}[x]$) to the corresponding ring of characteristic zero ($\mathbb{Z}$, $\mathbb{Z}[x]$, $\mathcal{O} = \mathbb{Z}[z]/(g(z))$, $\mathcal{O}[x]$). For the sake of clarity we use the overline notation "$\bar{a}$" to indicate that $\bar{a}$ is an element of a characteristic $p$ ring, and the absence of an overline in the notation "$a$" indicates that $a$ lives in characteristic zero.

As can be seen from the cluster pictures in Proposition 3.1, while the polynomial $f \in \mathbb{Z}[x]$ that defines the curve $C\colon y^2 = f(x)$ must be squarefree, the reduction of $p^{-v_p(c)}f$ modulo $p$ will typically have repeated factors (with repeated roots defined over $\mathbb{F}_p$ or $\mathbb{F}_{p^2}$); these factors are made explicit in Proposition 3.6 in the case that $f$ is $p$-normalized. Efficiently

determining these repeated factors and their multiplicity plays a key role in our algorithms and motivates the following definition.

**Definition 4.1** For each positive integer $k$ and polynomial $f \in \mathbb{F}_p[x]$ we define

$$\gcd_k(f) := \prod_{g^k \mid f} g^{v_g(f)-k+1} \in \mathbb{F}_p[x],$$

where $g$ ranges over monic irreducible elements of $\mathbb{F}_p[x]$ and $v_g(f) = \max\{e \in \mathbb{Z} : g^e \mid f\}$.

When $p > \deg(f)$ we can efficiently compute $\gcd_k(f)$ via

$$\gcd_k(f) = \gcd\left(f, f^{(1)}, \ldots, f^{(k-1)}\right),$$

where the gcd on the right is understood to be monic. Assuming $\deg f = O(1)$ (we will always have $\deg f = 6$) this computation takes $O(b \log^2 b)$ time (we need $O(1)$ operations in $\mathbb{F}_p$, including an inversion needed to make the result monic). For $p \le \deg f = O(1)$ we can compute $\gcd_k(f)$ in $O(1)$ time by exhaustively testing $g^k \mid f$ for all monic $g$ of degree at most $\lfloor \deg f/k \rfloor$.

In order to apply the main results of the previous section, which assume we are in the setting of Corollary 3.5, we may need to adjust the model of $C\colon y^2 = f(x)$ to ensure that it is defined by a $p$-normalized polynomial $f$ (see Definition 3.4). This leads to our first algorithm.

**Algorithm 1** *Given a squarefree $f(x) = \sum_i f_i x^i \in \mathbb{Z}[x]$ of degree 5 or 6 and an odd prime $p$, construct a polynomial $g(x) = \sum_i g_i x^i \in \mathbb{Z}[x]$ with $v_p(g_6) = \min_i\{v_p(g_i)\} \le 1$ for which the genus 2 curve $y^2 = g(x)$ has an outer cluster of depth zero at $p$ and is $\mathbb{Q}$-isomorphic to $y^2 = f(x)$.*

1. If $\deg f = 5$, compute $f(a)$ for $a = 0, 1, 2, \ldots$ until $f(a) \ne 0$, replace $f(x)$ by $f(x + a)$, and then replace $f(x)$ by $x^6 f(1/x)$ so that $\deg f = 6$.
2. Let $v = v_p(f_6)$, and if $v > 1$ or $v \ne \min_i\{v_p(f_i)\}$ then do the following:
   a. Let $e = \max\{\lceil \frac{v - v_p(f_i))}{6-i} \rceil : 0 \le i \le 5\}$.
   b. Replace $f(x)$ by $p^{6e-w} f(x/p^e) \in \mathbb{Z}[x]$, where $w = 2\lfloor v/2 \rfloor$, and replace $v$ by $v_p(f_6)$.
3. Let $h = p^{-v} f$.
4. Repeat the following steps:
   a. Let $\bar{u} = \gcd_6(h \bmod p)$ and if $\deg \bar{u} = 0$ then go to step 5.
   b. Replace $h(x)$ by $p^{-6} h(px + \mathrm{lift}(\bar{a}))$ where $\bar{u} = x - \bar{a}$.
5. Return $g = p^v h$.

**Proposition 4.2** *Algorithm 1 is correct and runs in time $O(\|f\|^2 \log^2 \|f\| / \log p)$.*

*Proof* That $y^2 = f(x)$ and $y^2 = g(x)$ are $\mathbb{Q}$-isomorphic follows from the fact that the sextic forms $z^6 f(x/z)$ and $z^6 g(x/z)$ are related by an invertible linear transformation of $\mathbb{P}^1$ and multiplication by an even power of $p$, neither of which changes the $\mathbb{Q}$-isomorphic class of the corresponding genus 2 curves. After step 2 we have $v_p(f_6) = \min_i\{v_p(f_i)\} \le 1$, since step 2b ensures $v_p(f_6) = v - w \in \{0, 1\}$ and $v_p(g_i) = 6e - w - ie + v_p(f_i) \ge v - w = v_p(g_6)$ for $0 \le i \le 5$. Step 2 can increase the depth of the outer cluster by at most $e = O(\|f\| / \log p)$.

In step 4 we have $h(x) = \sum_i h_i x^i \in \mathbb{Z}[x]$ with $v_p(h_6) = 0$. The outer cluster of $y^2 = h(x)$ at $p$ will have nonzero depth if and only if the roots of $h \bmod p$ all coincide, equivalently, if and only if $\bar{u}$ has positive degree. If this happens then $\deg \bar{u} = 1$ (since $\deg(h \bmod p) = 6$), and $\bar{u} = x - \bar{a}$, where $\bar{a}$ has multiplicity 6 as a root of $h \bmod p$, and then replacing $h(x)$ by $p^{-6}h(px + \mathrm{lift}(\bar{a})) \in \mathbb{Z}[x]$ reduces the depth of the outer cluster by one. Step 4 will terminate after $O(\|f\|/\log p)$ iterations when the depth reaches zero, and the genus 2 curve $y^2 = g(x) = p^v h(x)$ has its outer cluster of depth zero.

For the time bound, let $b = \|f\|$. We have $a \le 6$ in step 1, which performs $O(1)$ ring operations in $\mathbb{Z}$, taking time $O(b \log b)$, yielding a new $f \in \mathbb{Z}[x]$ with $\|f\| = O(b)$. We can compute $v_p(f_i)$ in time $O(b \log b)$, which bounds the cost of steps 2, 3, and 5. Each iteration of step 4 takes $O(b \log^2 b)$ time, and there are $O(\|f\|/\log p)$ iterations, yielding a total running time of $O(\|f\|^2 \log^2 \|f\|/\log p)$. □

*Remark 4.3* For $p > 5$ steps 1 and 2 of Algorithm 1 can be replaced by the following: let $v = \min_i\{v_p(f_i)\}$, test $a = 0, 1, 2, \ldots$ until $f(a) \not\equiv 0 \bmod p^{v+1}$, replace $f(x)$ by $f(x + a)$, then replace $f(x)$ by $p^{-2\lfloor v/2 \rfloor} x^6 f(1/x)$. This has the virtue of not increasing the depth of the outer cluster, which potentially saves time in step 4, but this may not work when $p = 3, 5$.

*Remark 4.4* The loop in step 4 of Algorithm 1 which is used to decrease the depth of the outer cluster can be viewed as computing a common initial $p$-adic approximation to the roots of $h(x)$, which all coincide modulo a power of $p$ equal to the depth of the outer cluster. The same technique will be used in our algorithms below which work with the inner clusters. If $h \bmod p$ had a simple root modulo $p$, or more generally, if we had an integer $s$ for which $v_p(h(s)) > 2v_p(h'(s))$, we could use Hensel lifting to approximate a root to $O(\|h\|/\log p)$ digits of $p$-adic precision in quasi-linear time (as a function of $\|h\|$), rather than the quasi-quadratic time required by step 4, since we can double the $p$-adic precision of our approximation in each step, rather than simply incrementing it. But we are in precisely the situation where Hensel's lemma does not apply, and in general $h(x)$ need not have any $\mathbb{Q}_p$-rational roots, so there is no reason to expect that we can use Hensel lifting.

*Remark 4.5* Unlike all our remaining algorithms, Algorithm 1 makes no assumptions about $f(x)$ other than requiring it to be squarefree of degree 5 or 6, so that $C : y^2 = f(x)$ is a genus 2 curve. Its output is $p$-normalized when $C$ has almost good reduction at $p$, which will be true in the context of our main algorithm (see Algorithm 7 below) where it is used.

If $f \in \mathbb{Z}[x]$ is a $p$-normalized polynomial we shall refer to the type of $f$ as the type of the cluster picture of $y^2 = f(x)$ at $p$, one of **1**, **2a**, **2b**, **4**. Our next algorithm uses Proposition 3.6 to efficiently determine the type of $f$.

**Algorithm 2** (WhichType) *Given a $p$-normalized $f = \sum_i f_i x^i \in \mathbb{Z}[x]$, determine its type.*

1. *Compute $\tilde{f} = p^{-v_p(f_6)} f \in \mathbb{Z}[x]$ and $\bar{f} = \tilde{f} \bmod p \in \mathbb{F}_p[x]$.*
2. *Compute $\bar{g} = \gcd_3(\bar{f})$.*
3. *If $\deg \bar{g} = 1$ then return **1**, and if $\deg \bar{g} = 3$ then return **4**.*
4. *If $\left(\frac{\Delta(\bar{g})}{p}\right) = +1$ then return **2a**, and return **2b** otherwise.*

**Proposition 4.6** *Algorithm 2 is correct and runs in time $O(\|f\| \log^2 \|f\|)$.*

*Proof* By Proposition 3.6, we have a unique triple root if we are in case **1**, two triple roots if we are in case **2a**, the cube of an irreducible quadratic if we are in case **2b**, and a unique quintuple root if we are in case **4**, and these are mutually exclusive. The degree of $\bar{g}$ is thus 1, 2, or 3, depending on whether we are in case **1**, **2a**/**2b**, or **4**, respectively, and the cases **2a** and **2b** are distinguished by whether $\bar{g}$ has $\mathbb{F}_p$-rationals roots or not, equivalently, whether $\Delta(\bar{g})$ is a square or not.

Let $b = \|f\|$. From Table 1 we see that Step 1 takes $O(b \log b)$ time, the cost of step 3 is negligible, and steps 2 and 4 both take $O(b \log^2 b)$ time, which bounds the total complexity. □

Having determined the type of a $p$-normalized $f \in \mathbb{Z}[x]$, we can use Corollaries 3.10 and 3.11 to compute the $L$-polynomial of $C\colon y^2 = f(x)$ at $p$. To simplify the presentation we treat each of the four types separately.

**Algorithm 3** (Type **1**) *Given a $p$-normalized polynomial $f \in \mathbb{Z}[x]$ of type **1**, compute the $L$-polynomial of $C\colon y^2 = f(x)$ at $p$.*

1. Compute $\bar{f} \equiv f \bmod p \in \mathbb{F}_p[x]$ and $\gcd_3(\bar{f}) = x - \bar{r} \in \mathbb{F}_p[x]$ (per Proposition 3.6).
2. Let $\bar{g}_1 = \bar{f}(x + \bar{r})/x^2 \in \mathbb{F}_p[x]$ (a quartic) and let $E_1\colon y^2 = \bar{g}_1(x)$.
3. Let $r = \operatorname{lift}(\bar{r})$ and repeat the following steps:
   a. Replace $f$ with $f(px + r)/p^3 \in \mathbb{Z}[x]$ and let $\bar{g}_2 \equiv f \bmod p \in \mathbb{F}_p[x]$ (a cubic).
   b. If $\Delta(\bar{g}_2) \neq 0$, then let $E_2\colon y^2 = \bar{g}_2(x)$ and go to step 4.
   c. Compute $\gcd_3(\bar{g}_2) = x - \bar{r} \in \mathbb{F}_p[x]$ and replace $r$ with $\operatorname{lift}(\bar{r})$.
4. Return $L_p(C, T) = L_p(E_1, T)L_p(E_2, T) \in \mathbb{Z}[T]$.

*Remark 4.7* Step 3b only needs to be performed every second iteration, since the depth must be even, by Proposition 3.1.

**Proposition 4.8** *Algorithm 3 is correct and runs in time $O\big(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p\big)$.*

*Proof* Step 2 computes $\bar{h}$ as the squarefree part of $\bar{f}$, while step 3 iteratively computes the polynomial $\bar{g}_2(x) = f(p^n x + s_1)/p^{3n} \bmod p$, where $s_1 \equiv r_1 \bmod p^n$ for some root $r_1$ in the inner cluster of depth $n$; correctness follows from Corollary 3.10.

Let $b = \|f\|$. From Table 1 we see that step 1 takes $O(b \log^2 b)$ time, step 2 takes $O(b \log b)$ time, each iteration of step 3 takes $O(b \log^2 b)$ time, and step 4 takes $O(\log^5 p)$ time. There are $n = O(\|f\| / \log p)$ iterations of step 3, and this yields the desired complexity bound. □

Our algorithm for type **2a** is the one case where it is difficult to give an efficient deterministic algorithm because we need to compute the roots of a quadratic polynomial over $\mathbb{F}_p$. This can be done efficiently using a probabilistic algorithm, or by assuming the extended Riemann hypothesis, but in order to give an unconditional deterministic algorithm we will assume we are given a nonsquare $s \in \mathbb{F}_p^\times$. We can use $s$ to deterministically compute the square root of any element of $\mathbb{F}_p$ via the Tonelli-Shanks algorithm, which effectively computes square roots using a deterministic discrete logarithm computation in the 2-Sylow subgroup $H$ of $\mathbb{F}_p^\times$ with respect to a given generator of $H$, which we can take to be $s^m$ where $p = 2^e m + 1$ with $m$ odd.

*Remark 4.9* The nonsquare $s \in \mathbb{F}_p^\times$ can be precomputed in $O(\log p(\log\log p)^2)$ expected time by picking random $s \in \mathbb{F}_p^\times$ until one finds $(\frac{s}{p}) = -1$; this computation depends only on $p$, not $f$.

**Algorithm 4** *(*Type **2A***) Given a p-normalized polynomial $f = \sum_i f_i x^i \in \mathbb{Z}[x]$ of type **2a** and a nonsquare $s \in \mathbb{F}_p^\times$, compute the L-polynomial of $C\colon y^2 = f(x)$ at p.*

1. Compute $\tilde{f} = p^{-v_p(f_6)} f \in \mathbb{Z}[x]$ and $\bar{f} \equiv \tilde{f} \bmod p \in \mathbb{F}_p[x]$.
2. Compute $\bar{u} = \gcd_3(\bar{f}) = (x - \bar{r}_1)(x - \bar{r}_2) \in \mathbb{F}_p[x]$ (per Proposition 3.6).
3. Compute the roots $\bar{r}_1, \bar{r}_2 \in \mathbb{F}_p$ of $\bar{u}$ via the quadratic formula, using $s$ to compute $\sqrt{\Delta(\bar{u})}$.
4. Use $\bar{r}_1$ and $\bar{r}_2$ to compute elliptic curves $E_1$ and $E_2$ as in step 3 of Algorithm 3 (using $f = \tilde{f}$).
5. Return $L_p(C, T) = L_p(E_1, T)L_p(E_2, T) \in \mathbb{Z}[T]$.

**Proposition 4.10** *Algorithm 4 is correct and runs in time $O\big(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p\big)$.*

*Proof* Correctness of the algorithm follows from Corollary 3.10, and the complexity analysis is as in the proof of Proposition 4.8, once we note that step 3 takes $O(b^2 \log^2 b / \log\log b)$ time, with $b = \log p$ in Table 1, which is bounded by the cost of step 5. □

For type **2b** we work in the setting of Corollary 3.11. If we put $\tilde{f} \coloneqq p^{-v_p(f_6)} f \in \mathbb{Z}[x]$ then Proposition 3.6 implies that $\tilde{f} \bmod p$ has the form $\bar{c} \cdot \bar{u}^3 \in \mathbb{F}_p[x]$ with $\bar{c} \in \mathbb{F}_p^\times$ and $\bar{u}$ a monic quadratic. We let $u = \mathrm{lift}(\bar{u}) \in \mathbb{Z}[x]$ and define $\mathcal{O} \coloneqq \mathbb{Z}[z]/(u(z))$ (an order in the quadratic field $\mathbb{Q}[z]/(u(z))$) and "$\kappa \coloneqq \mathbb{F}_p[z]/(\bar{u}(z)) \simeq \mathcal{O}/p\mathcal{O} \simeq \mathbb{F}_{p^2}$. The reduction map $\pi\colon \mathcal{O} \to \mathcal{O}/p\mathcal{O} \xrightarrow{\sim} \kappa$ sends the image of $z$ in $\mathcal{O}$ to the image of $z$ in $\kappa$ and integers to their reductions modulo $p$, while the map $\bar{a} \mapsto \mathrm{lift}(\bar{a})$ is the unique section of $\pi$ that sends elements of $\mathbb{F}_p$ to integers in $[0, p-1]$. We use $\hat{f}$ to denote the image of $\tilde{f}$ under the embedding $\mathbb{Z}[x] \hookrightarrow \mathcal{O}[x]$ induced by $\mathbb{Z} \hookrightarrow \mathcal{O}$.

**Algorithm 5** *(*Type **2B***) Given a p-normalized polynomial $f \in \mathbb{Z}[x]$ of type **2b**, compute the L-polynomial of $C\colon y^2 = f(x)$ at p.*

1. Compute $\tilde{f} = p^{-v_p(f_6)} f \in \mathbb{Z}[x]$ and $\bar{f} \equiv \tilde{f} \bmod p \in \mathbb{F}_p[x]$.
2. Compute $\bar{u} = \gcd_3(\bar{f}) = x^2 + \bar{u}_1 x + \bar{u}_2 \in \mathbb{F}_p[x]$ (per Proposition 3.6).
3. Let $\mathcal{O} \coloneqq \mathbb{Z}[z]/(u(z))$ and $\kappa \coloneqq \mathbb{F}_p[z]/(\bar{u}(z))$, where $u = \mathrm{lift}(\bar{u})$, and $\pi\colon \mathcal{O} \to \kappa$ be as above.
4. Let $\hat{f}$ be the image of $\tilde{f}$ in $\mathcal{O}[x]$, let $r = z \in \mathcal{O}$, and repeat the following steps:
   a. Replace $\hat{f}$ with $\hat{f}(px + r)/p^3 \in \mathcal{O}[x]$ and let $\bar{g} = \pi(\hat{f}) \in \kappa[x] \simeq \mathbb{F}_{p^2}[x]$ (a cubic).
   b. If $\Delta(\bar{g}) \neq 0$, then let $E\colon y^2 = \bar{g}(x)$ and go to step 5.
   c. Compute $\gcd_3(\bar{g}) = x - \bar{r} \in \kappa[x] \simeq \mathbb{F}_{p^2}[x]$ and replace $r$ with $\mathrm{lift}(\bar{r})$.
4. Return $L_p(C, T) = L_p(E, T^2) \in \mathbb{Z}[T]$.

**Proposition 4.11** *Algorithm 4 is correct and runs in time $O\big(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p\big)$.*

*Proof* Correctness follows from Corollary 3.11, and the complexity analysis is as in the proof of Proposition 4.8; the fact that we are working in $\mathcal{O}$ rather than $\mathbb{Z}$ and $\kappa \simeq \mathbb{F}_{p^2}$ rather than $\mathbb{F}_p$ changes the constant factors, but the asymptotic complexity bound is the same. □

**Algorithm 6** *(*Type **4***) Given a p-normalized polynomial $f \in \mathbb{Z}[x]$ of type **4**, compute the L-polynomial of $C : y^2 = f(x)$ at p.*

1. Compute $\tilde{f} = p^{-v_p(f_6)} f \in \mathbb{Z}[x]$ and $\bar{f} \equiv \tilde{f} \bmod p \in \mathbb{F}_p[x]$.
2. Compute $\gcd_5(\bar{f}) = x - \bar{r} \in \mathbb{F}_p[x]$ (per Proposition 3.6).
3. Let $r = \text{lift}(\bar{r})$ and repeat the following steps:
   a. Replace $\tilde{f}$ with $\tilde{f}(px + r)/p^5 \in \mathbb{Z}[x]$ and let $\bar{f} \equiv \tilde{f} \bmod p \in \mathbb{F}_p[x]$ (a quintic).
   b. If $\gcd_3(\bar{f}) = (x - \bar{s})^e$ has degree $e = 1$ then let $E_1 : y^2 = \bar{f}(x)/(x - \bar{s})^2$ and go to step 4.
   c. Compute $\gcd_5(\bar{f}) = x - \bar{r} \in \mathbb{F}_p[x]$ (per Proposition 3.6).
4. Let $r = \text{lift}(\bar{s})$ and repeat the following steps:
   a. Replace $\tilde{f}$ with $\tilde{f}(px + r)/p^3 \in \mathbb{Z}[x]$ and let $\bar{g} \equiv \tilde{f} \bmod p \in \mathbb{F}_p[x]$ (a cubic).
   b. If $\Delta(\bar{g}) \neq 0$, then let $E_2 : y^2 = \bar{g}(x)$ and go to step 5.
   c. Compute $\gcd_3(\bar{f}) = x - \bar{r}$ and replace $r$ by $\text{lift}(\bar{r})$.
5. Compute $L_p(C, T) = L_p(E_1, T)L_p(E_2, T)$.

**Proposition 4.12** *Algorithm 6 is correct and runs in time $O\big(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p\big)$.*

*Proof* Correctness follows from Corollary 3.10, and the complexity analysis is as in the proof of Proposition 4.8; steps 3 and 4 of Algorithm 6 have the same complexity as step 3 of Algorithm 3. □

We now present our main algorithm

**Algorithm 7** *(*Main*) Given a squarefree polynomial $f \in \mathbb{Z}[x]$ of degree 5 or 6 defining a genus 2 curve $C : y^2 = f(x)$ with almost good reduction at an odd prime p, and a nonsquare $s \in \mathbb{F}_p^\times$, compute the L-polynomial of C at p.*

1. Use Algorithm 1 to replace $f$ with a $p$-normalized polynomial $f$.
2. Use Algorithm 2 to determine the type of $f$ (**1**, **2a**, **2b**, or **4**).
3. Use whichever of Algorithms 3-6 matches the type to compute $L_p(C, T) \in \mathbb{Z}[T]$.

**Proposition 4.13** *Algorithm 7 is correct and runs in time $O(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p)$.*

*Proof* This follows immediately from Propositions 4.2, 4.6, 4.8, 4.10, 4.11 and 4.12. □

**Corollary 4.14** *There is a probabilistic implementation of Algorithm 7 that does not require the input $s \in \mathbb{F}_p^\times$ and runs in $O(\|f\|^2 \log^2 \|f\| / \log p + \log^5 p) \subseteq O(\|f\|^5)$ expected time.*

*Proof* As noted in Remark 4.9, we can compute a nonsquare $s \in \mathbb{F}_p^\times$ in $O(\log p (\log \log p)^2)$ expected time and then apply Algorithm 7. □

## 5 Implementation

A simple Magma implementation of Algorithms 1–7 is available in the Genus2Euler GitHub repository associated to this paper [13]. There is also a low-level C implementation based on the smalljac library [10] that is still being refined; we report preliminary timings here.

**Table 2** Timings for computing 3,454,506 Euler factors of genus 2 curves $C/\mathbb{Q}$ of small conductor at odd primes of almost good reduction

| Method | Total time | Average time | Median time | Maximum time |
|---|---|---|---|---|
| EulerFactor | 242 days | 6.1 seconds | 0.9 seconds | over 8 hours |
| Algorithm 7 (Magma) | 1.23 hours | 1.3 milliseconds | 1.2 milliseconds | 24 milliseconds |
| Algorithm 7 (C) | 27.1 seconds | 7.8 microseconds | 3.4 microseconds | 21 milliseconds |

We tested our algorithms on a dataset of approximately 2.5 million genus 2 curves $C/\mathbb{Q}$ of small conductor that have almost good reduction at some prime $p$. This dataset covers a conductor range comparable to the current database of genus 2 curves in the LMFDB ($2^{20}$ versus $10^6$), but spans a much larger discriminant range (the largest minimal discriminant is $|\Delta(C)| \approx 10^{217}$ versus $|\Delta(C)| \le 10^6$ in the LMFDB).

Many of these curves have almost good reduction at more than one odd prime, and the total number of $(C, p)$ pairs is 3 454 506. We attempted to compute each of the corresponding Euler factors in three ways: (1) using the EulerFactor intrinsic in Magma [2], (2) using the Magma implementation of Algorithm 7 available in [13], (3) using a smalljac-based C implementation of Algorithm 7. We ran our tests on a server equipped with dual 128-core AMD EPYC 9754 CPUs running at 2.25 GHz and 1.5 TB memory, with at most 256 tests running in parallel to avoid hyperthreading. All reported times are CPU times for a single core. We repeated each computation using our Magma and C implementations of Algorithm 7 for 100 and 10,000 iterations, respectively, to increase the accuracy of the timings (the algorithms are deterministic so there is very little variance in running times on the same input, but it is difficult to accurately time computations that take less than one millisecond), but we ran the computations using EulerFactor only once, due to the time involved; we terminated 489 of the tests involving EulerFactor that did not finish within 8 hours.

The total, average, median, and maximum running times to compute Euler factors for the 3 454 506 pairs $(C, p)$ are shown above, with the 489 cases not completed by EulerFactor capped at 8 hours (Table 2).

The maximum time for the C implementation of Algorithm 7 is larger than one might expect (relative to the Magma implementation of Algorithm 7) because in some of the **2b** cases we need to count points on elliptic curves over $\mathbb{F}_{p^2}$ with $p \approx 2^{30}$, a regime for which the smalljac library has not been optimized (it is much faster over prime fields); if one excludes the type **2b** cases the maximum running time for our C implementation drops to 34 microseconds, even with $p \approx 2^{35}$.

More detailed timing information broken out by type for small primes $p \le 2^{10}$ (about 90% of the test cases) can be found in Table 3. Finally, Tables 4 and 5 list ten examples of pairs $(C, p)$ where Magma's EulerFactor function struggled; Table 4 lists 10 examples where EulerFactor took roughly an hour, and Table 5 lists 10 of the 489 examples where EulerFactor failed to terminate within 8 hours. A complete list of these 489 cases can be found in [13].

**Table 3** Timings for Magma's EulerFactor function, and implementations of Algorithm 7 in Magma and C

| Prime(s) | Type | Count | EulerFactor | | Alg 7 Magma | | Alg 7 C | |
|---|---|---|---|---|---|---|---|---|
| | | | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ |
| 3 | **1** | 9413 | 1140 | 270 | 1.290 | 0.079 | 0.001872 | 0.000238 |
| | **2a** | 20,863 | 538 | 519 | 1.284 | 0.072 | 0.002346 | 0.000601 |
| | **2b** | 53,815 | 1151 | 270 | 0.942 | 0.080 | 0.002352 | 0.000699 |
| | **4** | 19,699 | 355 | 429 | 1.212 | 0.052 | 0.002185 | 0.000244 |
| 5 | **1** | 65,163 | 742 | 541 | 1.461 | 0.053 | 0.001979 | 0.000134 |
| | **2a** | 35,374 | 460 | 499 | 1.492 | 0.052 | 0.002248 | 0.000522 |
| | **2b** | 145,388 | 1163 | 270 | 1.092 | 0.057 | 0.002098 | 0.000604 |
| | **4** | 36,715 | 314 | 400 | 1.349 | 0.054 | 0.002115 | 0.000133 |
| 7 | **1** | 76,975 | 572 | 539 | 1.181 | 0.060 | 0.002157 | 0.000092 |
| | **2a** | 24,187 | 361 | 444 | 1.196 | 0.061 | 0.002488 | 0.000468 |
| | **2b** | 134,170 | 1158 | 269 | 0.828 | 0.054 | 0.002829 | 0.000803 |
| | **4** | 27,771 | 249 | 331 | 1.147 | 0.059 | 0.002295 | 0.000109 |
| $2^3 - 2^4$ | **1** | 149,872 | 412 | 470 | 1.283 | 0.064 | 0.002166 | 0.000102 |
| | **2a** | 21,027 | 293 | 380 | 1.317 | 0.076 | 0.002461 | 0.000395 |
| | **2b** | 188,755 | 1166 | 271 | 0.989 | 0.080 | 0.005278 | 0.002037 |
| | **4** | 25,475 | 204 | 258 | 1.192 | 0.076 | 0.002299 | 0.000078 |
| $2^4 - 2^5$ | **1** | 249,467 | 351 | 443 | 1.228 | 0.057 | 0.002245 | 0.000168 |
| | **2a** | 13,877 | 265 | 359 | 1.254 | 0.056 | 0.002544 | 0.000374 |
| | **2b** | 271,493 | 1198 | 271 | 1.089 | 0.143 | 0.006651 | 0.002138 |
| | **4** | 18,862 | 196 | 245 | 1.108 | 0.080 | 0.002380 | 0.000121 |
| $2^5 - 2^6$ | **1** | 191,028 | 313 | 407 | 1.252 | 0.053 | 0.002429 | 0.000178 |
| | **2a** | 3085 | 226 | 307 | 1.284 | 0.047 | 0.002788 | 0.000359 |
| | **2b** | 174,822 | 1217 | 284 | 1.697 | 0.577 | 0.008004 | 0.002155 |
| | **4** | 5141 | 185 | 220 | 1.098 | 0.069 | 0.002628 | 0.000074 |
| $2^6 - 2^7$ | **1** | 185005 | 290 | 390 | 1.287 | 0.051 | 0.002625 | 0.000178 |
| | **2a** | 851 | 233 | 313 | 1.324 | 0.061 | 0.003039 | 0.000359 |
| | **2b** | 161260 | 1379 | 602 | 1.189 | 0.867 | 0.009324 | 0.002153 |
| | **4** | 2087 | 188 | 230 | 1.105 | 0.067 | 0.002868 | 0.000101 |
| $2^7 - 2^8$ | **1** | 151,925 | 288 | 395 | 1.420 | 0.089 | 0.003010 | 0.000213 |
| | **2a** | 191 | 256 | 356 | 1.485 | 0.107 | 0.003422 | 0.000406 |
| | **2b** | 133,090 | 1858 | 2779 | 1.190 | 0.677 | 0.010872 | 0.002523 |
| | **4** | 684 | 229 | 304 | 1.169 | 0.103 | 0.003292 | 0.000193 |
| $2^8 - 2^9$ | **1** | 131,149 | 286 | 402 | 1.253 | 0.053 | 0.003851 | 0.000357 |
| | **2a** | 43 | 190 | 222 | 1.277 | 0.052 | 0.004163 | 0.000525 |
| | **2b** | 108,496 | 2623 | 7999 | 1.338 | 0.754 | 0.012742 | 0.002923 |
| | **4** | 203 | 324 | 460 | 1.083 | 0.067 | 0.004166 | 0.000336 |
| $2^9 - 2^{10}$ | **1** | 98,664 | 314 | 430 | 1.275 | 0.046 | 0.007900 | 0.001029 |
| | **2a** | 3 | 170 | 16 | 1.267 | 0.047 | 0.008911 | 0.000183 |
| | **2b** | 80,801 | 7051 | 46,480 | 1.687 | 1.003 | 0.015652 | 0.003182 |
| | **4** | 98 | 450 | 603 | 1.106 | 0.065 | 0.008300 | 0.000945 |

Columns $\mu$ and $\sigma$ are means and standard deviations in milliseconds, running on a single core of an AMD EPYC 9754 2.25 GHz processor

**Table 4** Examples that Magma's EulerFactor intrinsic took more than an hour to compute, with running times for Magma and C implementations of Algorithm 7

| Type | $p$ | $C$ | |
|---|---|---|---|
| **1** | 2,095,451 | $y^2 = 65366932x^6 + 46833852x^5 + 950560081x^4 + 1014328354x^3 - 714563571x^2 - 632448x + 750321408$ | 1.3 ms, 12.7 µs |
| **1** | 2,129,069 | $y^2 = -282619x^6 + 11424694x^5 - 66742653x^4 + 267785664x^3 + 783733439x^2 - 4055750250x - 6492528143$ | 1.3 ms, 14.5 µs |
| **1** | 2,141,299 | $y^2 = 35664905x^6 - 1683266x^5 + 81620201x^4 - 43104564x^3 - 550491952x^2 + 869809612x - 867569192$ | 1.4 ms, 12.3 µs |
| **1** | 2,192,653 | $y^2 = -7200195x^6 + 140298398x^5 + 82315425x^4 + 1863655712x^3 + 540423460x^2 - 272234940x - 11070656$ | 1.4 ms, 12.4 µs |
| **1** | 2,192,653 | $y^2 = -86525452x^6 + 301639692x^5 + 165992173x^4 - 631039776x^3 - 602207136x^2 + 199197628x + 586943224$ | 1.4 ms, 13.0 µs |
| **2b** | 2,192,653 | $y^2 = -7200195x^6 + 140298398x^5 + 82315425x^4 + 1863655712x^3 + 540423460x^2 - 272234940x - 11070656$ | 1.4 ms, 12.4 µs |
| **2b** | 2,192,653 | $y^2 = -86525452x^6 + 301639692x^5 + 165992173x^4 - 631039776x^3 - 602207136x^2 + 199197628x + 586943224$ | 1.4 ms, 13.0 µs |
| **2b** | 3,356,999 | $y^2 = -69840280x^6 - 88002004x^5 + 527168100x^4 + 947722520x^3 - 3244499x^2 - 2027697150x - 67561855$ | 1.4 ms, 12.6 µs |
| **2b** | 3,365,389 | $y^2 = 3365504x^6 + 423061824x^5 - 732552719x^4 + 490957150x^3 - 2239426319x^2 - 172708548x - 1287844864$ | 1.3 ms, 13.3 µs |
| **2b** | 3,520,511 | $y^2 = 87716080x^6 - 257007920x^5 - 436267519x^4 + 272523200x^3 - 1361866162x^2 + 320039284x - 994564803$ | 1.4 ms, 12.6 µs |

**Table 5** Examples that Magma's EulerFactor intrinsic was unable to compute in 8 hours, with running times for our Magma and C implementations of Algorithm 7

| Type | $p$ | $C$ | |
|---|---|---|---|
| **1** | 2,095,451 | $y^2 = 65366932x^6 + 46833852x^5 + 950560081x^4 + 1014328354x^3 - 714563571x^2 - 632448x + 750321408$ | 1.3 ms, 12.7 µs |
| **1** | 2,129,069 | $y^2 = -282619x^6 + 11424694x^5 - 66742653x^4 + 267785664x^3 + 783733439x^2 - 4055750250x - 6492528143$ | 1.3 ms, 14.5 µs |
| **1** | 2,141,299 | $y^2 = 35664905x^6 - 1683266x^5 + 81620201x^4 - 43104564x^3 - 550491952x^2 + 869809612x - 867569192$ | 1.4 ms, 12.3 µs |
| **1** | 2,192,653 | $y^2 = -7200195x^6 + 140298398x^5 + 82315425x^4 + 1863655712x^3 + 540423460x^2 - 272234940x - 11070656$ | 1.4 ms, 12.4 µs |
| **1** | 2,192,653 | $y^2 = -86525452x^6 + 301639692x^5 + 165992173x^4 - 631039776x^3 - 602207136x^2 + 199197628x + 586943224$ | 1.4 ms, 13.0 µs |
| **2b** | 2239 | $y^2 = 2720385x^6 + 58061748x^5 + 239277452x^4 - 714666410x^3 + 196933484x^2 - 986351148x + 596368845$ | 6.3 ms, 16.3 µs |
| **2b** | 2683 | $y^2 = -6613595x^6 - 33446278x^5 + 32788943x^4 + 45761248x^3 + 96912643x^2 - 3579181026x + 9931057425$ | 5.9 ms, 21.2 µs |
| **2b** | 2833 | $y^2 = 7977728x^6 + 73397364x^5 + 225744772x^4 + 359439708x^3 - 1032070399x^2 + 26023938x + 656386269$ | 5.7 ms, 21.5 µs |
| **2b** | 2957 | $y^2 = -2208879x^6 - 9568852x^5 + 45531886x^4 + 1710470736x^3 + 1769873909x^2 - 4037475972x - 1096633020$ | 5.7 ms, 22.2 µs |
| **2b** | 3079 | $y^2 = 28114349x^6 + 47422758x^5 - 91418589x^4 - 577694296x^3 + 735994923x^2 + 990569722x - 1007267139$ | 6.1 ms, 20.0 µs |

**Acknowledgements**
We are grateful to Matthew Bisatt and Tim Dokchitser for helpful conversations. Maistret was supported by a Royal Society Dorothy Hodgkins Fellowship, and Sutherland was supported by Simons Foundation Grant 550033.

**Data availability** A subset of the data used to test our implementation (including all the most difficult cases) is available in the Genus2Euler GitHub repository. The full dataset is available upon request.

**Author details**
[1]School of Mathematics, University of Bristol, Bristol BS8 1UG, UK, [2]Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

Received: 26 January 2024 Accepted: 9 September 2024 Published online: 20 February 2025

**References**

1. Best, A., Betts, A., Bisatt, M., van Bommel, R., Dokchitser, V., Faraggi, O., Kunzweiler, S., Maistret, C., Morgan, A., Muselli, S., Nowell, S.: A user's guide to the local arithmetic of hyperelliptic curves. Bull. Lond. Math. Soc. **54**, 825–867 (2022). https://doi.org/10.1112/blms.12604
2. Bosma, W., Cannon, J. J., Fieker, C., Steel, A. (eds.): Handbook of Magma Functions, Version 2.28-5 (2023). https://magma.maths.usyd.edu.au/magma/handbook/
3. Bouw, I., Wewers, S.: Computing L-functions and semistable reduction of superelliptic curves. Glasg. Math. J. **59**, 77–108 (2017). https://doi.org/10.1017/S0017089516000057
4. Booker, A., Sijsling, J., Sutherland, A.V., Voight, J., Yasaki, D.: A database of genus 2 curves over the rational numbers. Twelfth Algorithmic Number Theory Symposium (ANTS XII). LMS J. Comput. Math. **19**, 235–254 (2016). https://doi.org/10.1112/S146115701600019X
5. Brent, R.P., Zimmerman, P.: An $O(M(n \log n))$ algorithm for the Jacobi symbol. Ninth Algorithmic Number Theory Symposium (ANTS IX). LNCS **6197**, 83–95 (2010). https://doi.org/10.1007/978-3-642-14518-6_10
6. Dokchitser, T., Dokchitser, V., Maistret, C., Morgan, A.: Arithmetic of hyperelliptic curves over local fields. Math. Ann. **385**, 1213–1322 (2023). https://doi.org/10.1007/s00208-021-02319-y
7. Dokchitser, T.: Models of curves over discrete valuation rings. Duke Math. J. **170**, 2519–2574 (2021). https://doi.org/10.1215/00127094-2020-0079
8. Grothendieck, A.: Modèles de Néron et monodromie, LNM, vol. 288, Séminaire de Géométrie 7, Exposé IX. Springer (1973). https://doi.org/10.1007/BFb0068688
9. Harvey, D., van der Hoeven, J.: Integer multiplication in time $O(n \log n)$. Ann. Math. **193**, 563–617 (2021). https://doi.org/10.4007/annals.2021.193.2.4
10. Kedlaya, K.S., Sutherland, A.V.: Computing $L$-series of hyperelliptic curves. Eighth Algorithmic Number Theory Symposium (ANTS VIII). LNCS **5011**, 321–326 (2008). https://doi.org/10.1007/978-3-540-79456-1_21
11. Liu, Q.: Conducteur et discriminant minimal de courbes de genre 2. Compos. Math. **94**, 51–79 (1994)
12. Liu, Q.: Computing minimal Weierstrass equations. Res. Number Theory **9**, 76 (2023). https://doi.org/10.1007/s40993-023-00483-5
13. Maistret, C., Sutherland, A.V.: Genus2Euler, GitHub repository. https://github.com/AndrewVSutherland/Genus2Euler. Accessed 5 Feb 2024
14. Milne, J.S.: Étale Cohomology PMS, vol. 33. Princeton University Press (1980)
15. Muselli, S.: Regular models of hyperelliptic curves. Indag. Math. **35**, 646–697 (2024). https://doi.org/10.1016/j.indag.2023.12.001. arXiv:2206.10420
16. Raynaud, M.: Variétés abéliennes et géométrie rigide, Actes du congrès international de Nice 1970, tome 1, pp. 473–477. https://doi.org/10.1090/S0025-5718-10-02356-2
17. Schoof, R.: Elliptic curves over finite fields and the computation of square roots mod $p$. Math. Comput. **44**, 483–494 (1985)
18. Shparlinksi, I., Sutherland, A.V.: On the distribution of Atkin and Elkies primes for reductions of elliptic curves on average. LMS J. Comput. Math. **18**, 308–322 (2015). https://doi.org/10.1112/S1461157015000017
19. Sutherland, A.V.: Structure computation and discrete logarithms in finite abelian $p$-groups. Math. Comput. **80**, 477–500 (2011). https://doi.org/10.1090/S0025-5718-10-02356-2
20. Sutherland, A.V.: Genus 2 curves of small conductor **(in preparation)**
21. The LMFDB Collaboration: The $L$-functions and modular forms database. https://www.lmfdb.org. Accessed 22 Jan 2024
22. The PARI Group: PARI/GP version `2.15.4`, Univ. Bordeaux (2024). http://pari.math.u-bordeaux.fr/
23. The Sage Developers: SageMath, the Sage Mathematics Software System Version 10.1 (2024). https://www.sagemath.org
24. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra, 3rd edn. Cambridge University Press, Cham (2013)

## Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.