A short-list of pairing-friendly curves resistant to the Special TNFS algorithm at the 192-bit security level

Diego F. Aranha, Georgios Fotiadis, Aurore Guillevic

To cite this version:

Diego F. Aranha, Georgios Fotiadis, Aurore Guillevic. A short-list of pairing-friendly curves resistant to the Special TNFS algorithm at the 192-bit security level. 2024. hal-04666521

HAL Id: hal-04666521

https://inria.hal.science/hal-04666521

Preprint submitted on 1 Aug 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
A short-list of pairing-friendly curves resistant to the Special TNFS algorithm at the 192-bit security level

Diego F. Aranha\textsuperscript{1}, Georgios Fotiadis\textsuperscript{2} and Aurore Guillevic\textsuperscript{1,3,4}

\textsuperscript{1} Aarhus University, Aarhus N, Denmark
\textsuperscript{2} Université du Luxembourg, Esch-sur-Alzette, Luxembourg
\textsuperscript{3} Université de Lorraine, CNRS, Inria, LORIA, Nancy, France
\textsuperscript{4} Univ Rennes, Inria, CNRS, IRISA, Rennes, France

Abstract. For more than two decades, pairings have been a fundamental tool for designing elegant cryptosystems, varying from digital signature schemes to more complex privacy-preserving constructions. However, the advancement of quantum computing threatens to undermine public-key cryptography. Concretely, it is widely accepted that a future large-scale quantum computer would be capable to break any public-key cryptosystem used today, rendering today's public-key cryptography obsolete and mandating the transition to quantum-safe cryptographic solutions. This necessity is enforced by numerous recognized government bodies around the world, including NIST which initiated the first open competition in standardizing post-quantum (PQ) cryptographic schemes, focusing primarily on digital signatures and key encapsulation/public-key encryption schemes. Despite the current efforts in standardizing PQ primitives, the landscape of complex, privacy-preserving cryptographic protocols, e.g., zkSNARKs/zkSTARKs, is at an early stage. Existing solutions suffer from various disadvantages in terms of efficiency and compactness and in addition, they need to undergo the required scrutiny to gain the necessary trust in the academic and industrial domains. Therefore, it is believed that the migration to purely quantum-safe cryptography would require an intermediate step where current classically secure protocols and quantum-safe solutions will co-exist. This is enforced by the report of the Commercial National Security Algorithm Suite version 2.0, mandating transition to quantum-safe cryptographic algorithms by 2033 and suggesting to incorporate ECC at 192-bit security in the meantime. To this end, the present paper aims at providing a comprehensive study on pairings at 192-bit security level. We start with an exhaustive review in the literature to search for all possible recommendations of such pairing constructions, from which we extract the most promising candidates in terms of efficiency and security, with respect to the advanced Special TNFS attacks. Our analysis is focused, not only on the pairing computation itself, but on additional operations that are relevant in pairing-based applications, such as hashing to pairing groups, cofactor clearing and subgroup membership testing. We implement all functionalities of the most promising candidates within the RELIC cryptographic toolkit in order to identify the most efficient pairing implementation at 192-bit security and provide extensive experimental results.

Keywords: pairing-friendly curves · SNARK · TNFS

E-mail: dfaranha@cs.au.dk (Diego F. Aranha), georgios.fotiadis@uni.lu (Georgios Fotiadis), aurore.guillevic@inria.fr (Aurore Guillevic)
1 Introduction

Pairings have been used as a black-box tool in the cryptographic landscape since the first constructive applications have been proposed in 2000 and 2001. Some of the early applications were identity-based encryption [BF01], short signatures [BLS01], and tripartite Diffie-Hellman key exchange [Jou04]. In the following decades, pairings increased in popularity following the explosion in research within the broad area of cryptographic computing, consisting in essential building blocks to modern digital signatures [PS16, TZ23], commitment schemes [KZG10], anonymous credentials [CL04], and zero-knowledge proof systems such as zkSNARKs [Gro16].

Since the practical constructions of cryptographic pairings over elliptic curves are not secure against quantum-computers, an active research area is developing quantum-safe alternatives for pairing-based cryptography. However, migrating to quantum-safe cryptography has disadvantages in terms of efficiency and compactness, and potentially loss of some functionality. Taking for example zkSNARKs, there are promising quantum-safe constructions, such as zkSTARKs [BBHR18] and LaBRADOR [BS23], but they may offer larger proof sizes and/or prover/verification times that are still linear in the size of the witness or statement. Despite the exciting work in progress to develop alternatives, such performance penalties can be prohibitive for many applications.

In this paper, we follow a nuanced approach and argue that practical applications of pairing-based cryptography will need more conservative parameter choices while quantum-safe alternatives are further developed. In order to offer security in the next decade, we envision that parameters at the 192- and 256-bit security level will be needed. For reference, the Commercial National Security Algorithm Suite version 2.0 mandates transition to quantum-safe algorithms by 2033, and establishes that elliptic curve cryptography (ECC) at the 192-bit security level should be used before transition, effectively deprecating ECC at 128-bit security. The rationale behind the latter decision is rather unclear and unlikely to be ever properly substantiated, but there are speculations around computing elliptic curve discrete logs in cube-root time [KM16], without as much storage cost as publicly-known algorithms [BL13].

1.1 Motivation

Concretely, pairings are bilinear maps that are instantiated on algebraic curves. Contrary to other domains of cryptography where the choice of parameters is limited, straightforward, and standardized, the picture is more elaborated with pairings. Genus two pairings do not offer any advantage compared to elliptic curve (genus one) pairings in terms of security, while at the same time the Jacobian arithmetic is by far more costly compared to elliptic curve point addition. Instances of efficient and secure pairings on genus two curves have been recently reported in [AFK24], however they are still less efficient than the well-known elliptic curve pairings. For genus one, Barreto-Naehrig (BN) curves over large characteristic fields [BN06] became the dominant choice at 128-bit security for many years, followed by Barreto-Lynn-Scott (BLS) curves with embedding degree 12 at the 192-bit level [AFK+13]. Supersingular curves over small-characteristic fields were also considered for smaller devices, but efficient algorithms and record computations for discrete logarithms over extension fields completely eliminated their viability.1 These attacks do not apply in large characteristic, but the Kim-Barbulescu variant of the TNFS attack [KB16] still forced larger parameters, and the widely deployed BN-254/256 curves became outdated. Current deployments of pairing-based schemes are tailored to the updated 128-bit security level, and practical deployments have employed the BLS12-381 elliptic curve designed by the ZCash project [Bow17].

1In 2012 happened the first large record computation of discrete logarithm computation https://dldb.loria.fr in GF(3^97) (923 bits), and a final milestone in 2019 with GF(2^{30750}) [HSST12, GKL+21].
Scaling the security of pairing-friendly curves under the new constraints is challenging. Some previously obvious choices became undefined: alternative curves are not of prime order, contrary to BN curves, or might not have $j$-invariant 0 like BN curves. While the choice of parameters at the 256-bit security level seems to be clearer [KIK+17, BMDFAF19], the story is more complicated for the 192-bit security level due to a broader range of candidate parameters. Hence, we focus our efforts exactly on finding and benchmarking parameter choices at 192-bit security, providing a detailed overview in terms of security and efficiency. We discuss which curves to choose, together with precise security estimates; and performance in terms of the cost of pairing computation and group operations including hashing, membership testing, scalar multiplication and exponentiation. There has been a long and dense bibliography on pairing implementation, from designing new curves (see the survey article [FST10]) to dedicated hardware optimizations, which we extend with a study dedicated to the 192-bit security level.

1.2 Our Contributions

We review pairing-friendly curves at the 192-bit security level and provide TNFS-secure short-listed curves according to popular criteria. Some pre-selected curves were sketched in [Gui20, §5]. We also consider Scott–Guillevic (SG), Fotiadis–Martindale (FM), and Gasnier–Guillevic (GG) curves [SG18, FM19, GG23]. Because the search space of pairing-friendly curve families is too large for a complete survey, we restrict the evaluated curves w.r.t. the following a-priori criteria:

- Embedding degrees from 15 to 28. We justify this choice in Section 2.5.
- curves with high degree twists: quartic twist: $4 | k$ and $j(E) = 1728$ ($k = 16, 20, 28$), sextic twist: $6 | k$ and $j(E) = 0$ ($k = 18, 24$), and also cubic twist: $3 | k$ odd and $j(E) = 0$ ($k = 15, 21, 27$).
- variable $\rho$-value from the lowest possible according to [FST10], to $\rho = 2$ thanks to [FK19, FM19]. We observe that at fixed $k$, a larger $\rho$ can provide better performance.

We provide a full comparison of curves in terms of pairing efficiency, and $G_1$, $G_2$, and $G_T$ group operations:

- Formulas for optimal ate pairing, fast $G_1$, $G_2$, $G_T$ cofactor clearing and subgroup membership testing.
- Formulas for hashing into $G_1$, $G_2$ with SwiftEC [CSRT22] and Koshelev’s generalization [Kos24]. A side-contribution is an implementation-friendly description of hashing, potentially impacting other parameters for 128-bit security (e.g. embedding degree 8 curves [GMT20]).
- SageMath and Magma prototype code. Automated estimated cost in terms of multiplications in the base field $GF(p)$.
- An optimized implementation within the RELIC library, and benchmarks of pairings and group operations for the most promising curves according to the SageMath estimates.

2 Preliminaries

Notations. Denote $E$ an ordinary elliptic curve defined over a prime field $F_p$. Denote $k$ the embedding degree such that the pairing embeds a pair of points of $E(F_p)$ and $E(F_{p^k})$ onto $F_{p^k}^*$. The three pairing groups of prime order $r$ are denoted as usual $G_1, G_2, G_T$. 
2.1 Pairings

The state-of-the-art implementations of pairings at 128-bit security are for the optimal ate pairing on different types of curves. Below we recall the definition of the optimal ate pairing and the formula to compute the Miller loop. For more information on this pairing type we refer to Vercauteren’s paper [Ver10]. Furthermore, we give a brief description of the final exponentiation. For a complete introduction on pairings we refer to Craig Costello’s guide *Pairings for beginners* [Cos12].

**Optimal ate pairing Miller loop.** This is defined as the bilinear, non-degenerate and efficiently computable map \( e: G_2 \times G_1 \rightarrow G_T \), with \( G_1 = E[r] \cap \ker(\pi_p - [1]) \) and \( G_2 = E[r] \cap \ker(\pi_p - [p]) \), where \( \pi_p: E \rightarrow E \) is the \( p \)-th-power Frobenius endomorphism.

Let \( L \) be the \( \varphi(k) \)-dimensional lattice defined as:

\[
L = \begin{pmatrix}
   r & 0 & 0 & \ldots & 0 \\
   -p & 1 & 0 & \ldots & 0 \\
   -p^2 & 0 & 1 & \ldots & 0 \\
   \vdots & \vdots & \vdots & \ddots & \vdots \\
   -p^{\varphi(k)-1} & 0 & 0 & \ldots & 1
\end{pmatrix}
\]

and let \( v = (c_0, c_1, \ldots, c_{\varphi(k)-1}) \) be the shortest vector of the lattice \( L \). Then the coordinates of the shortest vector \( v \) satisfy the following relation:

\[
\sum_{i=0}^{\varphi(k)-1} c_i p^i \equiv 0 \mod r, \quad \text{(1)}
\]

where the coordinates \( c_i \) are functions of the seed \( u \), used to instantiate the polynomials \((p(x), t(x), r(x))\). Based on this setup, the formula for computing the optimal ate pairing is [Ver10]:

\[
(Q, P) \rightarrow \prod_{i=0}^{\varphi(k)-1} f_{c_i, Q}(P) \cdot \prod_{i=0}^{\varphi(k)-2} \ell_{R_i, S_i}(P) \cdot v_{R_i+S_i}(P),
\]

where \( R_i = [s_{i+1}]Q, S_i = [c_ip^i]Q \) and the scalars \( s_i \) are defined as:

\[
s_i = \sum_{j=0}^{\varphi(k)-1} c_j p^j,
\]

for every \( i = 0, \ldots, \varphi(k) - 2 \). In addition, \( \ell_{R_i, S_i}(P) \) denotes the line passing through the points \( R_i \) and \( S_i \), which is evaluated at \( P \), and \( v_{R_i+S_i}(P) \) is the vertical line through the point \( R_i + S_i \), evaluated at \( P \).

The optimal ate pairing formula can be simplified using certain properties of the Miller loop. In particular:

- For every \( c_i < 0 \), we have:
  \[
  f_{c_i, Q}(P) = \frac{1}{f_{-c_i, Q}(P) \cdot v_{[c_i]Q}(P)}.
  \]

- By [BKLS02, Theorem 2] the following holds:
  \[
  f_{a+b, Q}(P) = f_{a, Q}(P) \cdot f_{b, Q}(P) \cdot \frac{\ell_{[a]Q, [b]Q}(P)}{v_{[a+b]Q}(P)}.
  \]
• By [ALH10, Lemma 1] the following relation holds:
  \[ f_{ab,Q}(P) = f_{b,Q}(P)^a \cdot f_{a,[b]Q}(P) = f_{a,Q}(P)^b \cdot f_{b,[a]Q}(P). \]

• We have: \( f_{0,Q}(P) = f_{1,Q}(P) = f_{-1,Q}(P) = 1 \).

• Every scalar multiplication \([p]Q\) can be efficiently computed by applying the \( p^i \)-power Frobenius endomorphism \( \pi^i \):
  \[ [p]Q = \pi^i(Q), \] for every \( i = 1, \ldots, \varphi(k) - 1 \).

**Final exponentiation.** The final exponentiation is the process of raising an element in \( \mathbb{F}_p^* \) to the exponent \( e = (p^k - 1)/r \). When working with pairing groups, this is accomplished by factorizing \( e \) as follows:
  \[ e = \frac{p^k - 1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}, \]
  where we refer to \( e_{\text{easy}} = (p^k - 1)/\Phi_k(p) \) as the *easy part* and \( e_{\text{hard}} = \Phi_k(p)/r \) as the *hard part*. Both parts can be significantly simplified, depending on weather the embedding degree \( k \) is even, composite odd, or prime. The formula for computing the easy part of the final exponentiation is the same for all pairing-friendly elliptic curves with the same embedding degree. On the contrary, the formula for computing the hard part of the final exponentiation is different for each family of curves. There are different methods for simplifying the hard part, which we explore in Section 3.

### 2.2 Constructing pairing-friendly curves and families

Plain ordinary elliptic curves are not pairing-friendly, in particular, their embedding degree \( k \) has the magnitude of \( r \) so the pairing is impracticable. Pairing-friendly curves should be designed specifically and there have been a long series of papers revealing new curves. The first ordinary curves were MNT \((k = 3, 4, 6)\), followed by BLS, Brezing–Weng, the popular BN curves, and the KSS curves [MNT01, BLS03, BW05, BN06, KSS08]. Freeman, Scott and Teske delivered a useful taxonomy [FST10]. The recent works are a generalization of the BN construction [SG18] and the KSS construction [GG23] though without finding new prime-order curves. With the deployment of SNARK and the need of cycles of prime-order pairing-friendly curves, impossibility results are also investigated [BMUS23, CCW19]. We also mention a very recent work on higher genus cycle constructions for SNARK [CRSCN24]. In this work we concentrate on complete families, parameterized by polynomials and of fixed small discriminant.

### 2.3 Pairings at 128-bit security

#### 2.3.1 Assessing the TNFS-security level of pairing-friendly curves.

In 2016 the Kim–Barbulescu attack downgraded the security level of many pairing-friendly curves [KB16, KJ17], in particular the curves with composite embedding degree such as 12, 16, 18, 24. Because choosing \( k = 2^i 3^j \) allows an implementation-friendly towering and efficient field arithmetic, such choices are very common. Menezes, Sarkar, and Singh [MSS16] were the first to generate TNFS-secure parameters, recommending BLS12 curves of about 384 bits. Since then, the curve BLS12-381 is replacing BN-254 curves almost everywhere. We can mention the standardization effort about pairing-friendly curves [SKSW22], [BGW+22] on BLS signatures, [TL23] on BLS-curve-based key representation. Apart from BLS12, there are other specific proposals in other contexts [GMT20, FK19, FM19, Fot21, CDS20].
Analyzing TNFS is a challenging task. Barbulescu and Duquesne made recommendations in a worst-case scenario with a powerful attacker [BD19] (with conjectured hypotheses in the polynomial selection step of the TNFS algorithm). Guillevic and Singh [GS21] refined the TNFS analysis with the computation of the estimator $\alpha$, and released an implementation of the simulator. There are very few record computations with TNFS. To the best of our knowledge, we are aware of Oisín Robinson’s work in $\text{GF}(p^4)$ [Rob22] and Gabriele De Micheli in $\text{GF}(p^6)$ [DGP21].

In parallel of a better understanding of the TNFS algorithm, new parameter sets are proposed [Gui20, CDS20]. A short-list at the 128-bit security level [Gui20] was published in 2020, it gave hints on possible curves at the 192-bit security level but missed the shortest $G_1$ constraint [CDS20] and lacked BLS24 at the 128-bit security level. More recently there is Guillevic’s blog post [Gui21].

Better pairing implementations continue to be an active area of research, with the long-term software development in C++ RELIC [AGM+] (see for example [APR21] for recent benchmarks) but also newer projects in Golang or Rust for SNARK (see the survey [AHG23]). Latest improvements about pairing computations include a faster final exponentiation for all BLS curves [HHT20] thanks to a decomposition pattern, and for KSS18 curves [CHZ22]; implementation of pairings and group operations on prime-embedding-degree-pairing-friendly curves BW13-310 and BW19-286 (with endomorphisms but without twists) [DZZ23a, DZZ23b, FAGA23, DZZ22] (Yu Dai’s github at [Dai23]); Cofactor clearing and subgroup membership testing on pairing-friendly curves [HGP22, DLZZ23]; pairings in arithmetic circuits in the context of SNARK [Hou23].

The recommendations at the 128-bit security level are the following [Gui21]. For a prime-order curve e.g. for a hybrid cycle in the context of SNARK, pick a BN curve of 384–448 bits (Freeman and MNT curves are also possible but less efficient). For the fastest pairing, choose a BLS12 curve of 384–448 bits. For smallest $G_1$, set a BLS24 curve of $\approx$320 bits, and modified Cocks–Pinch curves with $5 \leq k \leq 8$. Finally for small embedding degree $k$, there are curves with $k = 1$, supersingular curves with $k = 2$, and modified Cocks–Pinch curves with $5 \leq k \leq 8$.

The papers [KIK+17, BMDFAF19] consider the 256-bit security level with embedding degrees up to 48 (BLS48-581) and 54 (SG54-569). While [BD19] recommends a KSS18-1484 or a BLS24-1032 curve, BEG19 expects a BLS27-559 curve (however this looks undersized). We collected the data in Appendix C Table 22.

2.4 Previous work on 192-bit security

The situation at the 192-bit security level is unclear, for at least two reasons. First there are more parameters to tune in TNFS for higher extension degrees. Second, there are more curve families to investigate (the embedding degree range to consider is larger for example). A nice short-list of best curves at the 192-bit security level is missing. Not only pairing computation but $G_1$, $G_2$, $G_T$ operations are important and need to be assessed. The size of $G_1$ and $G_2$ is an important criterion in certain usecases. Barbulescu and Duquesne focused on KSS18 and BLS24 (see Table 1). In 2019, Barbulescu, El Mrabet and Ghammam posted a preliminary report on many curves [BEG19]. Guillevic and Singh [GS21] and Guillevic [Gui20] proposed seeds and keysizes at the 192-bit security level for some families of curves (Tables 2 and 3). We aim at consolidating the knowledge for pairing-friendly curves at the 192-bit security level.

2.5 Justification of our a priori criteria

We listed in Section 1.2 our a priori criteria to restrict the considered curves. First we set boundaries on the embedding degree: $15 \leq k \leq 28$. The earlier work stated large bounds [Gui20, Eq. (8)]: $7168 \leq 384pk \leq 14336$. Because we target high degree twist curves, the
Table 1: Curves at the 192-bit security level from [BD19, Tab. 9, §7.6].

<table>
<thead>
<tr>
<th>$k$</th>
<th>Curve</th>
<th>NFS variant</th>
<th>seed</th>
<th>$p$, bits</th>
<th>$p^h$, bits</th>
<th>$\mathbb{F}_{p^h}$ sec, bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>KSS</td>
<td>STNFS</td>
<td>85 bits</td>
<td>678</td>
<td>12999</td>
<td>192</td>
</tr>
<tr>
<td>24</td>
<td>BLS</td>
<td>STNFS</td>
<td>56 bits</td>
<td>555</td>
<td>13300</td>
<td>192</td>
</tr>
</tbody>
</table>

18 $k$ KSS STNFS $\kappa = 1$ $\tau = -2^{254} - 2^{11} - 2^{26} + 2^6$ 678 12168 204
24 $k$ BLS STNFS $\kappa = 1$ $\tau = -2^{56} - 2^{43} + 2^9 - 2^6$ 559 13416 204

Table 2: Curves at the 192-bit security level from [Gui20, Tab. 10, §5].

<table>
<thead>
<tr>
<th>$k$</th>
<th>Curve</th>
<th>seed, bits</th>
<th>$r$, bits</th>
<th>$p$, bits</th>
<th>$p^h$, bits</th>
<th>$\mathbb{F}_{p^h}$ sec, bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>FST 6.3</td>
<td>52</td>
<td>620</td>
<td>928</td>
<td>12979-12992</td>
<td>194</td>
</tr>
<tr>
<td>15</td>
<td>BLS</td>
<td>78</td>
<td>620</td>
<td>928</td>
<td>13906-13920</td>
<td>193</td>
</tr>
<tr>
<td>20</td>
<td>FST 6.4</td>
<td>56</td>
<td>448</td>
<td>670</td>
<td>13371-13400</td>
<td>192</td>
</tr>
<tr>
<td>21</td>
<td>BLS</td>
<td>32</td>
<td>384</td>
<td>511</td>
<td>10691-10719</td>
<td>195</td>
</tr>
<tr>
<td>27</td>
<td>BLS</td>
<td>22</td>
<td>384</td>
<td>427</td>
<td>11496-11524</td>
<td>212</td>
</tr>
<tr>
<td>28</td>
<td>FST 6.4</td>
<td>32</td>
<td>384</td>
<td>510</td>
<td>14243-14280</td>
<td>208</td>
</tr>
</tbody>
</table>

2.6 Security Analysis: TNFS and its variants

In [BD19], Barbulescu and Duquesne further investigated the consequences of the extended Tower NFS algorithm of Kim et al. [KB16], after a first analysis in [MSS16]. In particular, they simulated worst-case scenarios from a cryptographer perspective to obtain key-size

we scan the curve families of the taxonomy paper plus the newer ones listed in 1.2 (Fotiadis–Martindale, Scott–Guillevic) and obtain these bounds. We provide a Sagemath script to reproduce this result. For RSA keysizes at the 192-bit security level, a modulus of 7680 bits is required according to NIST, while Lenstra Updated corresponds to 12548 bits. For $k = 15$, the work [Gui20] already considered BLS15 and stated that $p$ of 928 bits offers 192 bits of security. Such $p$ is very large and we do not include it in our analysis. However we can take $k = 15$ and $\rho = 2$ as in [FK19]. The parameters are $p$ of 768 bits, $r$ of 384 bits and the security level is about 192 bits, for $p^h$ of about 11520 bits. The paper [Gui20] already established that there is no curve family whose embedding degree is larger than 28 but whose $\rho$-value is small enough to satisfy $384\rho p \leq 14336$, hence $k \leq 28$. 

Table 3: Seeds at the 192-bit security level from [GS21, Tables 3, 4, 7].

<table>
<thead>
<tr>
<th>$k$</th>
<th>curve</th>
<th>$r$, bits</th>
<th>$p$, bits</th>
<th>$p^h$, bits</th>
<th>seed $\tau$</th>
<th>$\mathbb{F}_{p^h}$ sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>BN</td>
<td>1022</td>
<td>1022</td>
<td>12255</td>
<td>$-2^{254} + 2^{24} + 2^6$</td>
<td>191</td>
</tr>
<tr>
<td>12</td>
<td>BLS12</td>
<td>768</td>
<td>1150</td>
<td>13799</td>
<td>$-2^{192} + 2^{188} - 2^{17} - 2^{10} - 2^{4} - 1$</td>
<td>193</td>
</tr>
<tr>
<td>16</td>
<td>KSS16</td>
<td>605</td>
<td>766</td>
<td>12255</td>
<td>$2^{78} - 2^{76} - 2^{26} + 2^{14} + 2^1$</td>
<td>194</td>
</tr>
<tr>
<td>18</td>
<td>KSS18</td>
<td>474</td>
<td>638</td>
<td>11477</td>
<td>$2^{60} + 2^{77} + 2^{76} - 2^{61} - 2^{53} - 2^{14}$</td>
<td>193</td>
</tr>
<tr>
<td>24</td>
<td>BLS24</td>
<td>409</td>
<td>509</td>
<td>12202</td>
<td>$-2^{31} - 2^{28} + 2^{11} - 1$ [CLN11]</td>
<td>193</td>
</tr>
</tbody>
</table>
A short-list of pairing-friendly curves at the 192-bit security level

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_{\text{sieve}}$</td>
<td>cost of sieving, per relation</td>
<td>1</td>
</tr>
<tr>
<td>$c_{\text{lin. alg.}}$</td>
<td>factor of lin. algebra</td>
<td>1/4</td>
</tr>
<tr>
<td>$c_{\text{filter}}$</td>
<td>filtering factor</td>
<td>$\log_2 B$</td>
</tr>
<tr>
<td>$c_{\text{density}}$</td>
<td>matrix' weight per row</td>
<td>128</td>
</tr>
<tr>
<td>$\mathcal{A}$</td>
<td>automorphisms</td>
<td>$k / \gcd(\deg h, k / \deg h)$</td>
</tr>
</tbody>
</table>

Table 4: Choices of parameters in [BD19, GS21].

We recommend the following formula [BD19, Eq. (2)]:

$$
\text{cost} = c_{\text{sieve}} \frac{1}{A} \cdot \text{smoothness probability} + c_{\text{lin. alg.}} c_{\text{density}} \frac{\# \text{ required relations}}{A} \cdot \frac{\text{filtering ratio}}{\rho_D \left( \frac{\text{avg. log}_2 N_f}{\log_2 B} \right)^{-1} \cdot \rho_D \left( \frac{\text{avg. log}_2 N_g}{\log_2 B} \right)^{-1} + 4^7 \left( \frac{2B}{A \log B} \right)^2}
$$

where $\rho_D$ is Dickman’s $\rho$ function. Inside the relation collection model of cost, [BD19] assumes that the duplicates due to the roots of unity can be avoided for free in the relation collection, hence a number of obtained relations in a sieving space of dimension $2 \deg h$ and samples whose coefficients are bounded by $A$ of

$$
\frac{(2A + 1)^{2\deg h}}{2w} \rho_D \left( \frac{\text{avg. log}_2 N_f}{\log_2 B} \right) \rho_D \left( \frac{\text{avg. log}_2 N_g}{\log_2 B} \right) \geq \frac{2B}{\log B}.
$$

Guillevic and Singh propose a slightly different model of cost in [GS21] where they evaluate the smoothness bias of the norms in TNFS. For that they need practical curve parameters and polynomials. Their experiments are given as a Sagemath implementation. One requires

$$
\frac{(2A + 1)^{2\deg h}}{2w} \rho_D \left( \frac{\log N_f + \alpha (f, h)}{\log B} \right) \rho_D \left( \frac{\log N_g + \alpha (g, h)}{\log B} \right) \geq 2 \log \text{Integral}(B)
$$

and

$$
\text{cost} = \frac{(2A + 1)^{2\deg h}}{2A} c_{\text{sieve}} + c_{\text{lin. alg.}} c_{\text{density}} \left( \frac{2 \log \text{Integral}(B)}{c_{\text{filter}}} \right)^2
$$

We summarize in Table 2.6 the different choices. In [GS21], the cost of linear algebra is modeled as the number of 64-bit machine word limbs of $\ell$: $[\log_2 \ell / 64]$ times the weight per row, times the matrix size squared.

We incorporated the assumptions of [BD19] in the Sagemath simulator of [GS21] so that we can give the two estimates with the same simulation tool.

3 Pairings at 192-bit security

3.1 Selection of pairing-friendly curves

Our selection of pairing-friendly curves for 192-bit security follows the same strategy as the case of 128-bit security (e.g., see the recent study in [Gui20]). More concretely, the choice of elliptic curves that will be included in our analysis is based on the following criteria:

C1. Use elliptic curves that admit high degree twists $d \in \{3, 4, 6\}$.
C2. Focus on pairing-friendly elliptic curves with embedding degree $15 \leq k \leq 28$.
C3. The selected curves should be TNFS-resistant.
C4. Consider pairing-friendly curves with $\rho = \log p / \log r$ up to 2.
C5. Apply and implement the optimal ate pairing.
C6. Identify efficient candidate curves, based on additional pairing-related operations, beyond the pairing computation itself.

We choose to implement the optimal ate pairing for our selected curves, since the state-of-the-art for 128-bit security suggests that this is the most efficient pairing type. The curves that we consider admit high degree twists i.e., they have $j$-invariant 1728 $(4 \mid k)$ or $j$-invariant 0 $(3 \text{ or } 6 \mid k)$. The use of high-degree twists is crucial for ensuring fast optimal ate pairing instances [CLN10]. We note that extending the fastest curve options in 128-bit security (e.g., BLS12 and BN curves) to the 192-bit setting is not the optimal choice, since this would require the size of the base field prime $p$ to be larger than 1000 bits [Gui21]. The security of the selected curves should be evaluated taking into consideration the progress on the TNFS attacks targeting the DLP in extension fields of composite degree [KB16]. In particular, the security of our selected curves is evaluated using the STNFS software simulator\(^3\) of Guillevic and Singh [GS21].

Furthermore, although the initial requirement for pairing-friendly curves was to look for $\rho \approx 1$ [FST10], sometimes curves with larger $\rho$ can be beneficial. In particular, given a pairing-friendly family of curves with fixed and composite embedding degree $k$, increasing the size of the base field prime $p$ to counter the TNFS attacks results in the increase of the size of the prime $r$ defining the prime order subgroup of $E(\mathbb{F}_p)$ and hence an increase of the length of the Miller loop. Alternatively, one can choose a pairing-friendly family of curves for the same $k$ with larger $\rho$, so that the size of $p$ is increased without affecting the size of $r$. Therefore, in our study we include curves with $\rho$-value up to 2. Such curve instances at 192-bit security with $\rho = 2$ are studied in [FK19]. In addition, we point out that our results on optimal pairing-friendly curves for 192-bit security depend not only on the fast pairing implementation, but also on efficient exponentiation in the three pairing-groups $G_1, G_2, G_T$, hashing to $G_1, G_2$, cofactor clearing and subgroup membership testing in $G_1, G_2, G_T$. Hence, in our quest for the most efficient pairing candidates at 192-bit security, we include an analysis on the efficiency of these additional pairing functionalities.

Based on these criteria we identified the most promising candidate families of pairing-friendly elliptic curves in the literature which can result in efficient pairing-related operations at 192-bit security. In particular, we have concluded to the following families:

- Kachisa–Schaefer–Scott (KSS) [KSS08]. Families KSS16 and KSS18 for embedding degrees 16 and 18 respectively.
- Freeman–Scott–Teske (FST) [FST10]. We follow Construction 6.4 in [FST10] for families with $k = 20$ and $k = 28$ and we refer to these families as FST20 and FST28.
- Barreto–Lynn–Scott [BLS03]. Families BLS12, BLS15, BLS21, BLS24, BLS27, for embedding degrees 12, 15, 21, 24 and 27. Family BLS12 does not comply with the embedding degree selection criteria (C2). However, we include this family to justify our earlier claim that although BLS12 is the most efficient choice for 128-bit security, increasing the size of $p$ to reach 192-bit security is not the optimal strategy to follow. In addition, we include BLS15 to compare to FM15.
- Fotiadis–Martindale [FM19]. Families FM15, FM16, FM18 and FM20, for embedding degrees 15, 16, 18 and 20. These correspond to the families with number #21, #23, #25 and #27 in [FM19, Table 2]. We note that family FM20 coincides with FST20.
- Scott–Guillevic [SG18]. Families SG18 and SG20 for embedding degrees 18 and 20. These types of families are also known as Aurifeuillean and they are constructed following the method of Scott and Guillevic [SG18].
- Gasnier–Guillevic [GG23]. Families GG20b and GG28 for $k = 20$ and $k = 28$. As mentioned in [GG23], although they don’t improve the $\rho$-value compared to FST10

\(^{3}\)Simulation tool in SageMath under MIT license: https://gitlab.inria.fr/tnfs-alpha/alpha
Table 5: Curves of embedding degree 15 to 28 with a high degree twist $d \in \{3, 4, 6\}$ and such that the curve $E(F_p)$ offers at least 192 bits of security.

<table>
<thead>
<tr>
<th>$k$</th>
<th>curve</th>
<th>$\langle a, b \rangle$</th>
<th>$p \mod k$</th>
<th>$\log p$</th>
<th>$\log r$</th>
<th>$\log p^2$</th>
<th>secu/ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>BLS12</td>
<td>$-12x^{17} + x^{16} + x^{15} + x^{14} + x^{13} + 1$</td>
<td>(0, 1)</td>
<td>7</td>
<td>1170</td>
<td>706</td>
<td>1.50</td>
</tr>
<tr>
<td>15</td>
<td>BLS15</td>
<td>$2^{18} + 2^{16} + 2^{14} + 2^{13} + 2^{12} + 1$</td>
<td>(0, 1)</td>
<td>1</td>
<td>894</td>
<td>597</td>
<td>1.50</td>
</tr>
<tr>
<td>16</td>
<td>KSS16</td>
<td>$2^{4} + 2^{3} + x^{2} + x + 1$</td>
<td>(1, 0)</td>
<td>13</td>
<td>706</td>
<td>605</td>
<td>1.50</td>
</tr>
<tr>
<td>18</td>
<td>FST16</td>
<td>$2^{28} + 2^{26} + 2^{25} + 2^{24} + 2^{23} + 2^{22} + 2^{21} + 2^{20} + 2^{19} + 2^{18} + 2^{17} + 2^{16} + 2^{15} + 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>765</td>
<td>384</td>
<td>2.00</td>
</tr>
<tr>
<td>18</td>
<td>SG18</td>
<td>$2^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^{9} + x^{8} + x^{7} + x^{6} + x^{5} + x^{4} + x^{3} + x^{2} + x + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>765</td>
<td>384</td>
<td>2.00</td>
</tr>
<tr>
<td>18</td>
<td>SG18</td>
<td>$2^{15} - 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(0, 1)</td>
<td>1</td>
<td>1147</td>
<td>401</td>
<td>1.66</td>
</tr>
<tr>
<td>18</td>
<td>SG18</td>
<td>$2^{15} - 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>765</td>
<td>384</td>
<td>2.00</td>
</tr>
<tr>
<td>18</td>
<td>SG18</td>
<td>$-2^{15} + 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(0, 1)</td>
<td>1</td>
<td>765</td>
<td>384</td>
<td>2.00</td>
</tr>
<tr>
<td>18</td>
<td>SG18</td>
<td>$-2^{15} + 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(0, 1)</td>
<td>1</td>
<td>765</td>
<td>384</td>
<td>2.00</td>
</tr>
<tr>
<td>20</td>
<td>BLS20</td>
<td>$-2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>1999</td>
<td>767</td>
<td>1.52</td>
</tr>
<tr>
<td>20</td>
<td>BLS20</td>
<td>$-2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>765</td>
<td>384</td>
<td>2.00</td>
</tr>
<tr>
<td>20</td>
<td>BLS20</td>
<td>$-2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>509</td>
<td>200</td>
<td>1.37</td>
</tr>
<tr>
<td>20</td>
<td>BLS20</td>
<td>$-2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>509</td>
<td>200</td>
<td>1.37</td>
</tr>
<tr>
<td>21</td>
<td>BLS21</td>
<td>$2^{17} + 2^{16} + 2^{15} + 2^{14} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(0, 1)</td>
<td>1</td>
<td>511</td>
<td>384</td>
<td>1.34</td>
</tr>
<tr>
<td>27</td>
<td>BLS27</td>
<td>$2^{10} + 2^{9} + 2^{8} + 2^{7} + 2^{6} + 2^{5} + 2^{4} + 2^{3} + 2^{2} + 2 + 1$</td>
<td>(1, 0)</td>
<td>1</td>
<td>14276</td>
<td>594</td>
<td>1.31</td>
</tr>
</tbody>
</table>

and FST28, they are more resistant to the TNFS attack.

FM15, FM16 and FM18 families have $p = 2$. In Section 3.4 we compare these families with others which have smaller $p$ for the same embedding degree. Furthermore, for completeness we also considered the case $k = 22$ and specifically the families of Freeman–Scott–Teske obtained by Constructions 6.3 and 6.4 (FST22 (6.3), FST22 (6.6)) and the family of Gasnier–Guillluc (GG22). Such examples of pairings do not follow the selection criterion (C1), since such elliptic curves have only quadratic twists, and they don’t seem to be competitive to pairings on elliptic curves that admit high degree twists. However, because we already have SageMath implementations of these curves and in order to justify that indeed they are not competitive, we include these examples in Appendix A (see Table 18). As we were finalizing our paper, Lin, Zhao and Zheng posted [LZZ24] on ePrint, on implementing $k = 22 \ D = 7$ GG curves, investigating the strategy of super-optimal pairings [FAGA23]. In addition to the aforementioned families of pairing-friendly elliptic curves, we also include the following family for embedding degree 16 and $p = 2$ in our analysis.

**Family 1** (Aranha–Fotiadis–Guillluc (AFG16)). The following polynomials $p(x), t(x), r(x) \in \mathbb{Q}[x]$, where:

\[
p(x) = (x^{16} + 2x^{13} + x^{10} + 5x^8 + 6x^5 + x^2 + 4)/4
\]

\[
t(x) = r(x) + x^5 + 1 = x^8 + x^5 + 2
\]

\[
r(x) = \Phi_{16}(x) = x^8 + 1
\]

define a polynomial family of pairing-friendly elliptic curves with embedding degree $k = 16$, CM discriminant $D = 1$ and $\rho = 2$. The order of the curve in parametric form is $\#E(F_p) = h(x)r(x)$, where $h(x) = (x(x^3 + 1))^2$. The family is integer-valued for all $u \in \mathbb{Z}$. The polynomial $r$ evaluated at integers of the form $2u$ is odd, and is even at integers of the form $2u + 1$, for $u \in \mathbb{Z}$.

The above family of elliptic curves is similar to the one proposed in [FK19, Table 5] for $k = 16$. The difference is that the cofactor $h(x)$ is a perfect square and $\sqrt{h(x)}$ divides
**Table 6:** Optimal ate Miller loop formulas. The Miller functions \( f_{u,Q} \) and lines \( \ell_{Q,R} \) are evaluated at the point \( P \in G_1 \).

<table>
<thead>
<tr>
<th>( k )</th>
<th>curve</th>
<th>Equation (1)</th>
<th>Optimal ate formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>BLS12</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>15</td>
<td>BLS15</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>16</td>
<td>KSS16</td>
<td>( 2 + u p^3 + p^4 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>17</td>
<td>FM16</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>18</td>
<td>AFG16</td>
<td>( u + p^2 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>19</td>
<td>KSS18</td>
<td>( 1 + u p^2 + 2p^6 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>20</td>
<td>FM18</td>
<td>( u + p^2 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>21</td>
<td>FM20/FST20</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>22</td>
<td>SG20</td>
<td>( 2u + p^2 + p^3 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>23</td>
<td>GG20b</td>
<td>( u - p - 2p^6 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>24</td>
<td>BLS24</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>25</td>
<td>BLS27</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>26</td>
<td>FST28</td>
<td>( u - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td>27</td>
<td>GG28</td>
<td>( u - p - 2p^8 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
</tbody>
</table>

The seeds that we have chosen for instantiating these families are given in Table 5. We searched for seed with minimal Hamming weight, which would give curves with either \( a = 0 \) or \( a = 1 \) (mod 3) for fast hashing, and a nice towering of extensions with only small non-residues. For some families multiple seeds are reported, however we choose the ones highlighted in grey for our theoretical comparison, as well as for our benchmarking in Section 5. The resulting curves offer security level of at least 192-bit with respect to the TNFS attacks, except for the ones highlighted in red which offer slightly less. The seed \( 2^{48} + 2^{28} + 2^{26} \) for FM16 [FM19] does not give machine-word aligned parameters: \( p \) is 767-bit long but \( r \) is 385-bit long, this overflows the 384-bit standard size. We replace it with \( 2^{45} - 2^{44} \cdot 2^{38} + 2^{31} \) which has heavier Hamming weight but whose \( r \) fits in 384 bits. The seed \( -(2^{64} + 2^{35} - 2^{11} + 1) \) for FM18 does not give machine-word aligned parameters, we replace it with \( -2^{64} + 2^{33} + 2^{30} + 2^{20} + 1 \). The seed for KSS16 from [GS21] (see Table 3) does not fulfill all criteria for fast hashing (see 4.2) and we replace it with a new one. For BLS24, we consider the seed given in [GS21] which dates back to [CLN11] and we give two new ones well-suited for SNARK, with a high 2-valuation of \( p - 1 \) and \( r - 1 \).

### 3.2 Pairing Computation: Miller Loop

As stated earlier, for all curves given in Table 5 we apply the optimal ate pairing, introduced by Vercauteren [Ver10], following our selection criterion (C5). In Table 6 we give the form of Equation (1), as well as the corresponding optimal ate pairing formulas, for all families of pairing-friendly elliptic curves considered in Table 5. These formulas are simplified using the properties described in Section 2. We note that for some curves, there exist alternative choices for the short vectors, which however do not offer any advantage in terms of efficiency compared to the ones that we have chosen.

**Remark 1** (Concerning BLS curves with odd embedding degrees \( k = 3 \mod 6 \), examples...
A short-list of pairing-friendly curves at the 192-bit security level

Table 7: Miller loop cost in Weierstrass model from [CSB05, CLN10].

<table>
<thead>
<tr>
<th>(k)</th>
<th>curve</th>
<th>Db1 step, tangent line</th>
<th>sparse-sparse-(a_k)</th>
<th>full-sparse-(a_k)</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>(y^2 = x^3 + ax + b) (\text{quadratic twist})</td>
<td>(5m_{1,k}/2 + 6s_{k/2} + km)</td>
<td>(m_k)</td>
<td>(m_k)</td>
<td>[CSB05]</td>
</tr>
<tr>
<td>2</td>
<td>(y^2 = x^3 - 3x + b) (\text{quadratic twist})</td>
<td>(6m_{1,k}/2 + 4s_{k/2} + km)</td>
<td>(m_k)</td>
<td>(m_k)</td>
<td>[CSB05]</td>
</tr>
<tr>
<td>2</td>
<td>(y^2 = x^3 + b) (\text{quadratic twist})</td>
<td>(2m_{1,k}/2 + 7s_{k/2} + km)</td>
<td>(m_k)</td>
<td>(m_k)</td>
<td>[CLN10, §5, Tab.3]</td>
</tr>
<tr>
<td>2</td>
<td>(y^2 = x^3 + ax) (\text{quadratic twist})</td>
<td>(3m_{1,k}/2 + 8s_{k/2} + km)</td>
<td>(m_k)</td>
<td>(m_k)</td>
<td>[CLN10, §4, Tab.3]</td>
</tr>
<tr>
<td>3</td>
<td>(y^2 = x^3 + b) (\text{cubic twist})</td>
<td>(6m_{1,k}/3 + 7s_{k/3} + km + m_k)</td>
<td>(m_k)</td>
<td>(m_k)</td>
<td>[CLN10, §6]</td>
</tr>
<tr>
<td>6</td>
<td>(y^2 = x^3 + b) (\text{sextic twist})</td>
<td>(2m_{1,k/6} + 7s_{k/6} + (k/3)m)</td>
<td>(6m_{k/6})</td>
<td>(13m_{k/6})</td>
<td>[CLN10, §5]</td>
</tr>
<tr>
<td>6</td>
<td>(y^2 = x^3 + b) (\text{sextic twist})</td>
<td>(3m_{1,k/6} + 6s_{k/6} + (k/3)m)</td>
<td>(6m_{k/6})</td>
<td>(13m_{k/6})</td>
<td>[AKL+11, §4.6]</td>
</tr>
<tr>
<td>4</td>
<td>(y^2 = x^3 + ax) (\text{quartic twist})</td>
<td>(2m_{1,k/4} + 8s_{k/4} + (k/2)m)</td>
<td>(6m_{k/4})</td>
<td>(8m_{k/4})</td>
<td>[CLN10, §4]</td>
</tr>
</tbody>
</table>

are BLS15, BLS21, and BLS27 in Table 6). For these curves, the verticals are not in a subfield and there is no easy denominator elimination as for even embedding degrees. Nevertheless, the optimal Miller loop simplifies to \(f_{u,Q}(P)\). One starts expanding the Miller loop of length \(u-p\): \(f_{u-p,Q}(P) = f_{u,Q}(P)f_{u-p,Q}(P)\). Now observe that \(f_{u-p,Q}(P) = f_{u-p,Q}(P)/v(p)Q\) with a vertical at \(v(p)Q\) evaluated at \(P\). Also the line through \([u]Q\) and \(-[p]Q\) is a vertical since \([u-p]Q = O\). Finally, \(1/v(p)Q\) and \(\ell([u]Q,-[p]Q)\) cancel themselves, and \(f_{p,Q}(P)\) is a pairing and can be taken out. The final formula is indeed \(f_{u,Q}(P)\) like for BLS curves of even embedding degrees.

In Table 7 we report the cost for the different elliptic curve operations that are relevant to the pairing computation and especially the Miller loop. Specifically, we include the cost for computing the point addition and point doubling (column 3), as well as the cost of the line computation (column 3) and result accumulation (column 4) that occur in the Miller loop. The elliptic curve model that we have considered is the Weierstrass model, where points are represented in the projective coordinate system, following the works of Chatterjee, Sarkar, Barua [CSB05] and Costello, Lange, Naehrig [CLN10]. In addition, these costs refer to elliptic curves admitting degree 2, 3, 4 and 6 twists. In Table 10 we denote by \(\mathbf{m}_i\) and \(\mathbf{s}_i\) the multiplication and squaring, respectively, in the finite field \(\mathbb{F}_{p^i}\), for \(i > 1\) and by \(\mathbf{m}\) we denote multiplication over the base field \(\mathbb{F}_p\).

In column 4 of Table 7 we report the cost in terms of \(\mathbb{F}_p\)-multiplications for computing a product of two elements in \(\mathbb{F}_{q^d}\), where \(q = p^{k/d}\) and \(d \in \{2, 3, 4, 6\}\) is the degree of the twist. More concretely, we write \(a, b \in \mathbb{F}_{q^d}\), such that:

\[
\begin{align*}
    a &= \sum_{i=0}^{d-1} a_i w^i \\
    b &= \sum_{i=0}^{d-1} b_i w^i,
\end{align*}
\]

where \(a_i, b_i \in \mathbb{F}_q\) and \(w\) is the generator of the degree \(d\) extension \(\mathbb{F}_{q^d}\) of \(\mathbb{F}_q\). The term \(\text{sparse-sparse-}a_k\) refers to the case where in the representations of \(a\) and \(b\), some of the coefficients \(a_i\) and \(b_i\) are zero (sparse representation), while on the other hand, \(\text{full-sparse-}a_k\) is the case where one of the two representations is sparse. Such types of \(\mathbb{F}_p\)-multiplications appear when updating the Miller function after the doubling and addition steps in the Miller loop when sextic or quartic twists are applied. Note that when quadratic or cubic twists are employed, the \(\text{sparse-sparse-}a_k\) and \(\text{full-sparse-}a_k\) multiplications are equivalent to an \(a_k\) multiplication, as reported in Table 7.
### 3.3 Pairing Computation: Final Exponentiation

Table 8 shows the formulas for the easy and hard parts of the final exponentiation for each embedding degree we consider in this paper. The computation of the easy part is straightforward, while for the hard part we follow the state of the art approaches to obtain an optimized formula for the final exponentiation in the examples that we present here. That is, according to [HHT20], there are three distinct techniques for the efficient computation of the hard part: 1. a base-p expansion to exploit cheap Frobenius powers, 2. a lattice-reduction technique on the exponent in base $p$ to reduce further the size of the coefficients of the base-$p$ expansion, and 3. a cyclotomic decomposition of the exponent. The latter technique is better suited when the trace of Frobenius of the curve is represented as a polynomial of degree one, which is the case for all BLS curves.

In the following we give the formulas for the optimized hard part of the final exponentiation for all families of Table 5. In each case we present the theoretical cost in terms of the number of required operations. Section 3.3.1 decomposes the hard part exponent $\Phi_k(p)/r$ with the techniques in [HHT20] while Section 3.3.2 considers lattice reduction [FKR12] before a fine tuning, resulting in a final exponentiation by a multiple of the hard part exponent $\Phi_k(p)/r$.

#### 3.3.1 With Cyclotomic Decomposition

**BLS12.** The optimal formula [HHT20, §5 p.14] is:

$$3 \frac{\Phi_{12}(p)}{r} = (u - 1)^2(u + p)(p^2 + u^2 - 1) + 3$$

with cost: $2 \exp(|u - 1|) + 3 \exp(|u|) + 5 m_{12} + 1 s_{12} + 2 f_{12} + 1 f_5$.

**BLS15.** The formula in [HHT20, §5 p.14] is:

$$3 \frac{\Phi_{15}(p)}{r} = (u - 1)^2(u^2 + u + 1) \left( \sum_{i=0}^{7} \lambda_i p^i \right) + 3,$$

where $\lambda_7 = 1$ and the rest of the $\lambda_i$ are defined as follows:

$$\lambda_0 = u \lambda_7 - 1, \quad \lambda_5 = u \lambda_6, \quad \lambda_4 = u \lambda_5 + 1, \quad \lambda_3 = u \lambda_4 - 1, \quad \lambda_2 = u \lambda_3 + 1, \quad \lambda_1 = u \lambda_2 + 1, \quad \lambda_0 = u \lambda_1 - 1.$$

The cost is: $1 \exp(|u - 1|) + 1 \exp(|u^2 - 1|) + 7 \exp(|u|) + 14 m_{15} + 1 s_{15} + 7 f_{15} + 1 \text{inv-cyclo}_{15}$, where $\text{inv-cyclo}_{15}$ corresponds to cyclotomic inversion in $\mathbb{F}_{p^{15}}$ and is equivalent to $f^{-1} = f^{\frac{1}{15}} - f^{\frac{1}{45}}$ with cost $\text{inv-cyclo}_{15} = f_{10} + f_5 + m_{15}$.

### Table 8: Exponents for the final exponentiation, easy and hard parts.

<table>
<thead>
<tr>
<th>$k$</th>
<th>final exp. easy part</th>
<th>final exp. hard part</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>$(p^6 - 1)(p^5 + 1)$</td>
<td>$(p^4 - p^2 + 1)/r$</td>
</tr>
<tr>
<td>15</td>
<td>$(p^6 - 1)(p^5 + p + 1)$</td>
<td>$(p^5 - p^3 + p^2 - p^3 + p^5 - p + 1)/r$</td>
</tr>
<tr>
<td>16</td>
<td>$p^5 - 1$</td>
<td>$(p^5 + 1)/r$</td>
</tr>
<tr>
<td>18</td>
<td>$(p^6 - 1)(p^5 + 1)$</td>
<td>$(p^5 - p^4 + 1)/r$</td>
</tr>
<tr>
<td>20</td>
<td>$(p^4 - 1)(p^4 + 1)$</td>
<td>$(p^5 - p^6 + p^4 - p^2 + 1)/r$</td>
</tr>
<tr>
<td>21</td>
<td>$(p^4 - 1)(p^5 + p + 1)$</td>
<td>$(p^{12} - p^{11} + p^9 - p^8 + p^6 - p^4 + p^3 - p + 1)/r$</td>
</tr>
<tr>
<td>24</td>
<td>$(p^{12} - 1)(p^4 + 1)$</td>
<td>$(p^5 - p^4 + 1)/r$</td>
</tr>
<tr>
<td>27</td>
<td>$p^6 - 1$</td>
<td>$(p^{11} + p^7 + 1)/r$</td>
</tr>
<tr>
<td>28</td>
<td>$(p^{12} - 1)(p^4 + 1)$</td>
<td>$(p^{12} - p^{10} + p^9 - p^8 + p^4 - p^2 + 1)/r$</td>
</tr>
</tbody>
</table>
We note also that a short-list of pairing-friendly curves at the 192-bit security level includes:

\[
\frac{\Phi_{16}(p)}{r} = \left[\frac{u^2}{4}(u^6 + 1) + 1\right](p + u)(p^2 + u^2)(p^4 + u^4) + 1
\]

with cost: \(13 \exp(|u|) + 2 \exp(|u/2|) + 6m_{16} + 3f_{16} + c_{16}\). We note that the seed \(u\) for this family is always even, in order to produce primes \(p\) and \(r\) [FM19]. In addition, our implementation works for negative seeds as well, in which case one conjugation is needed in the operation count (\(c_{16}\)).

**AFG16.** We compute this formula for Family 1 for \(k = 16\):

\[
\frac{\Phi_{16}(p)}{r} = \left[\frac{u^2}{4}(u^6 + 1)^2 + 1\right]\left(p^5 + u(u^3p(1 + up^3) - 1)\right) + up(1 + up^3) + 1
\]

with cost: \(2 \exp(u/2) + 13 \exp(u) + 9m_{16} + 4f_{16}\).

**SG18.** For the hard part of SG18 we apply the formula of Scott–Guillevic [SG18], namely:

\[
\frac{\Phi_{18}(p)}{r} = (3u^2 - 1)^2\left[p^2 + 3pu^2 + 9u^4 + 3u(1 - 3up + p^3) - 3p^2\right] + (1 - 3up + p^3)
\]

with total cost: \(9 \exp(|u|) + 14m_{18} + 5s_{18} + 5f_{18} + 3c_{j_{18}} + c_{j_{18}}\). The additional conjugation is added to include the case of negative seeds.

**FM18.** Our formula for the hard part of the final exponentiation is the following:

\[
\frac{\Phi_{18}(p)}{r} = (p - u)\left[u^6 + \frac{(u - 1)^2}{3} + 1\right]\left(p^4 + p^3u - p - u^3(p^3u - p - u)\right)
\]

\[+ u(p^3u - p - u) + 1
\]

with cost: \(7 \exp(|u|) + \exp(|u - 1|) + \exp(|u - 1|/3) + 2 \exp(|u + 1|) + 13m_{18} + 2s_{18} + 4f_{18} + 4c_{j_{18}}\). We note also that \((u - 1)^2 \equiv 0 \mod 3\). Alternatively, one can multiply by 3 the exponent to avoid a possibly non-sparse \((u - 1)/3\):

\[
3\frac{\Phi_{18}(p)}{r} = (p - u)\left[2(2(u^6 + 1) - u) + u^2 - u^6\right]\left(p^4 + p^3u - p - u^3(p^3u - p - u)\right)
\]

\[+ 3u(p^3u - p - u) + 3
\]

with cost: \(11 \exp(|u|) + 12m_{18} + 3s_{18} + 4f_{18} + 6c_{j_{18}}\).

**FM20/FST20.** The hard part of the final exponentiation for FM20 [FM19, Family #27, Table 2] is equivalent to FST6.4 [FST10, Construction 6.4] for \(k = 20\). The formula we considered is:

\[
\frac{\Phi_{20}(p)}{r} = \frac{(u - 1)^2}{4}(u^2 + 1)(p + u)\left[(p^2 + u^2 - 1)(p^4 + u^4 + 1) - u^2p^2\right] + 1,
\]

with total cost: \(2 \exp(|u - 1|/2) + 9 \exp(|u|) + 8m_{20} + 1f_{20} + 2f_{2} + 1f_{1} + 2c_{j_{20}} + 1c_{j_{20}}\), where the one additional conjugation is required when the seed is negative.
For this type of curves, we apply [HHT20, Theorem 1] to obtain the following formula for the hard part of the final exponentiation:

\[ 3 \frac{\Phi_{21}(p)}{r} = (u - 1)(u^3 - 1) \left( \sum_{i=0}^{11} \lambda_i p^i \right) + 3, \]

where \( \lambda_{11} = 1, \lambda_i = u\lambda_{i+1} + c_i \) and \( c_i \) is defined as:

\[ c_i = \begin{cases} 
1, & \text{for } i = 2, 5, 8 \\
0, & \text{for } i = 1, 4, 6, 9 \\
-1, & \text{for } i = 0, 3, 7, 10 
\end{cases} \]

with cost: \( 15 \exp(|u|) + 15m_{21} + s_{21} + 11f_{21} + 2\text{inv-cyclo}_{21} \).

The hard part of the final exponentiation is written as [HHT20, §5 p.16]:

\[ 3 \frac{\Phi_{24}(p)}{r} = (u - 1)^2(u + p)(u^2 + p^2)(u^4 + p^4 - 1) + 3 \]

with cost: \( 2\exp(|u - 1|) + 7\exp(|u|) + 6m_{24} + s_{24} + 3f_{24} + c_{124} \).

For this type of curves, we apply [HHT20, Theorem 1] to obtain the following formula for the hard part of the final exponentiation:

\[ \frac{\Phi_{27}(p)}{r} = (u - 1)^2(p^2 + pu + u^2)(p^6 + p^3u^3 + u^6)(p^9 + u^9 + 1) + 3 \]

The total cost is: \( 2\exp(|u - 1|) + 17\exp(|u|) + 8m_{27} + s_{27} + f_{27} + f_{27} + f_{27} + f_{27} + f_{27} + f_{27} \)

\((+3(m + 2f) \text{ if } u < 0)\).

Our optimal formula for the hard part of the final exponentiation is:

\[ \frac{\Phi_{28}(p)}{r} = \frac{(u - 1)^2}{4}(u^2 + 1)(p + u)e + 1, \]

where the value \( e \) is determined as follows:

\[ e = \left( u^2(u^2 + p^2) + p^4 \right) \left( u^2(u^2 - 1) + 1 \right) + p^6 - 1 + \left( u^2 - 1 + p^2 \right). \]

The total cost is: \( 2\exp(|u - 1|/2) + 13\exp(|u|) + 12m_{28} + 1f_{28} + 2f_{2} + 1f_{1} + 2f_{6} + 2c_{128} + 1c_{128} \),

where the extra conjugation appears in the case where the seed is negative.

### 3.3.2 With Lattice Reduction

We use the following formula for the hard part in the case of FM15:

\[ 3u(u^3 - u^2 + 1) \frac{\Phi_{15}(p)}{r} = \sum_{i=0}^{7} \lambda_i p^i, \]

where the \( \lambda_i \) are defined by the following relations:

\[ \begin{align*}
\lambda_6 &= -3 - 3u^2 + 2u^4 + u^5 + 2u^6 - 3u^8, & \lambda_0 &= -(u^6 - u^4 + u^3)\lambda_6 + 3u^2 + 3u \\
\lambda_1 &= u^7\lambda_6 - (3u^3 + 3u^3 + 3u), & \lambda_2 &= u^3\lambda_6 \\
\lambda_3 &= -(u^7 + u^5 - 1)\lambda_6 + 3u^3 + 3u^2 + 3u, & \lambda_4 &= (u^7 - u^5 - 1)\lambda_6 - (3u^3 + 3u^2 - 3) \\
\lambda_5 &= u^4\lambda_6 - 3, & \lambda_7 &= -(u^3 - u^3 + u)\lambda_6 + 3 
\end{align*} \]

Then the total cost is: \( 15 \exp(|u|) + 37m_{15} + 6s_{15} + 11f_{15} + 2f_5 + 2\text{inv-cyclo}_{15} \) when \( u > 0 \),

while one less multiplication in \( \mathbb{F}_{p^{15}} \) is required for negative seeds.
**KSS18.** For the computation of the hard part of the final exponentiation, the most efficient method is presented in [CHZ22]. We write

\[ \frac{3u^2}{49} \cdot \Phi_{18}(p) \cdot \frac{5}{r} = \sum_{i=0}^{5} \lambda_i p^i \]

where the \( \lambda_i \) are defined in terms of the cofactor \( c = 3h(u)/49 = u^2 + 5u + 7 \):

\[
\begin{align*}
\lambda_5 &= u^2c + 3, \\
\lambda_4 &= -3u\lambda_5 - 49c, \\
\lambda_3 &= 2u^2\lambda_5 + 35uc, \\
\lambda_2 &= 2\lambda_4 + u\lambda_5, \\
\lambda_1 &= 2\lambda_4 + u\lambda_5, \\
\lambda_0 &= 2\lambda_3 + u\lambda_4, \\
\lambda_2 &= -u\lambda_3 + 2\lambda_5.
\end{align*}
\]

The cost for the hard part in this case is: \( 7\exp(u) + 24m_{18} + 11s_{18} + 7f_9 + 5f_{18} \).

**SG20.** For the hard part of the final exponentiation for the SG20 family we apply the lattice reduction to obtain the formula \( \Phi_{20}(p)/r = \sum_{i=0}^{5} \lambda_i p^i \). The \( \lambda_i \) in this representation are defined as \( \lambda_0 = uh + 1, \lambda_1 = uy \), with \( y = 2u\lambda_0 - 1, \lambda_0 = -uy', \) with \( y' = 2u\lambda_7, \lambda_1 = -uy', \) with \( y'' = -2u\lambda_0 \) and for the rest of the \( \lambda_i \) have:

\[
\begin{align*}
\lambda_4 &= \lambda_1 - \lambda_0 - y, \\
\lambda_3 &= -\lambda_1 - \lambda_0 + y'', \\
\lambda_2 &= -\lambda_3 + y' + \lambda_7 - h, \\
\lambda_5 &= -\lambda_3 + \lambda_0 - h.
\end{align*}
\]

The total cost is: \( 13\exp(|u|) + 19m_{20} + 6s_{20} + 7f_{20} + 7c_{j_{20}} (+c_{j_{20}}) \), where the additional conjugation is to account for the case of negative seeds.

### 3.3.3 Base-\( p \) Expansion

**KSS16.** For the hard part of the KSS16 family we obtained the following formula:

\[
\frac{2u^7 + 48u^4}{125} \cdot \Phi_{16}(p) \cdot \frac{5}{r} = (u^3c + 56)e_1 + ce_2 - 1540p^7,
\]

where \( c = 2h(u)/125 = u^2 + 2u + 5 \) and the values \( e_1, e_2 \) are defined as:

\[
\begin{align*}
e_1 &= u^4p^7 + u^3(1 - 2p^4) + u^2p^3(1 - 4p^4) + up^2(-11 + 2p^4) + p^3(7 + 24p^4), \\
e_2 &= 125p^3(2p^4 - 2) + 29up(3p^3 + 4) + 5u^2(-11p^4 + 2).
\end{align*}
\]

The total cost for computing the hard part for KSS16 is: \( 2\exp(u - 1) + 7\exp(u) + 29s_{16} + 30m_{16} + 10f_4 + 4f_{16} \), improving on the previous best count from [Gha16, §4.3 p.107, Eq.(4.9) p.114] (also at [GF16]) of \( 2\exp(u + 1) + 7\exp(u) + 37s_{16} + 35m_{16} + 4f_{16} + 2f_2 + f_1 \).

**GG20a and GG20b.** For the hard part of the final exponentiation we use the fact that \( p^8 \equiv (p^6 - p^5 + p^4 - 1) \mod r \) and we set the following values: \( c = u^2 - 2u + 5, \alpha_\pm = 2 \pm p^5, \beta_\pm = 4 \pm 3p^5, \gamma_\pm = 7 \pm 24p^5 \) and \( \delta_\pm = 11 \pm 2p^5 \). Then we obtain the following formula for the hard part of the GG20a family, where \( s \) is a scaling factor coprime to \( r \):

\[
\frac{\Phi_{20}(p)}{r} = (u^4c + 328)(-41p^2 + up\gamma_- + u^2p\delta_- + u^3p^4\beta_- + u^4p^3\alpha_+ + u^5p^7) + c(625pc_0 + 125u\beta_+ + 25u^2p^4\delta_+ + 5u^3p^3\gamma_+ + 38u^4p^7) + 6724p^7
\]

with a total cost: \( 2\exp(u - 1) + 9\exp(u) + 51s_{20} + 42m_{20} + 4f_6 + 9f_{20} \).

Using the same logic as for GG20a and changing only \( \alpha_\pm = \pm 2 + p^5 \) and \( \beta_\pm = \pm 4 + 3p^5 \), the GG20b final exponent is written as follows, with exactly the same cost as GG20a:

\[
\frac{\Phi_{20}(p)}{r} = (u^4c - 328)(-41p^2 + up\gamma_+ + u^2p\delta_+ + u^3p^4\beta_+ + u^4p^3\alpha_- + u^5p^7) + c(-625pc_0 + 125u\beta_- + 25u^2p^4\delta_- + 5u^3p^3\gamma_- - 38u^4p^7) + 6724p^7 .
\]
We present a theoretical analysis of the total cost for computing the different pairings Table 5, and we aim at presenting a theoretical comparison of the different pairings Table 5. More concretely, the usual practice in estimating the cost of a pairing is to express the Miller loop and the final exponentiation in terms of instances of Table 5. Such a theoretical comparison necessitates that all operations needed in the Miller loop and in the final exponentiation are translated to \( \mathbb{F}_p \)-multiplications. Then we set the following values:\n
\[
\begin{align*}
\delta &= 2 \pm 3 p^2, \\
gamma &= 11 p^2 + 2, \\
\beta &= 24 + 7 p^2, \\
\delta &= 24 p^2 + 7, \\
\epsilon &= -41 \pm 38 p^2, \\
\zeta &= -117 \pm 44 p^2.
\end{align*}
\]

Then we obtain the following formula for the hard part, where \( s \) is a scaling factor coprime to \( r \):

\[
\frac{s^{\Phi_{28}(p)}}{r} = (u^6 c - 232) e_1 + c e_2 + 3364 p^{11}
\]

where the values \( e_1, e_2 \) are defined as follows:

\[
\begin{align*}
e_1 &= -29 p^4 + u p^3 \zeta_+ + u^2 p^2 \epsilon_+ + u^3 p \delta_+ + u^4 \gamma + u^5 p^6 \beta_+ + u^6 p^5 \alpha_+ + u^7 p^{11}, \\
e_2 &= 5^5 p^4 \alpha_- + 5^5 u p^2 \beta_+ + 5^4 u^2 p \gamma + 5^3 u^3 \delta + 5^2 u^4 p^6 \epsilon_+ + 5 u^5 p^7 \zeta_+ + 278 u^6 p^{11}
\end{align*}
\]

The total cost is: \( 2 \exp(u - 1) + 13 \exp(u) + 68 s_{28} + 83 m_{28} + 12 f_r + 14 f_{28} \).

### 3.4 Pairing Computation: Theoretical Comparison

We present a theoretical analysis of the total cost for computing the different pairing instances of Table 5. More concretely, the usual practice in estimating the cost of a pairing is to express the Miller loop and the final exponentiation in terms of \( \mathbb{F}_p \)-multiplications. Then we aim at presenting a theoretical comparison of the different pairings Table 5, based on the total number of \( \mathbb{F}_p \)-multiplications required. This theoretical comparison is outlined in Table 10 and will serve as the baseline for selecting the most promising pairing candidates to be included in our benchmark experiments in Section 5.

Such a theoretical comparison necessitates that all operations needed in the Miller loop and in the final exponentiation are translated to \( \mathbb{F}_p \)-multiplications. This conversion is described in Table 9, following the usual estimates, e.g., [GMT20, Table 4] for a recent presentation. We denote \( m_i \), resp. \( s_i \), a multiplication, resp. squaring in \( \mathbb{F}_p \). Furthermore, we denote \( f_i \), \( s_i^{\text{cyco}} \) and \( i_i \), the Frobenius \( p \)-power, the cyclotomic squaring and inversion
in $\mathbb{F}_{p^k}$. When it is clear from the context, $f_j$ denotes a $p^j$-th power in $\mathbb{F}_{p^k}$. Based on the theoretical comparison in Table 10, we can extract the most promising curves to be benchmarked. We discuss the results of Table 10 and justify our selections of pairing-friendly curves to be benchmarked.

**Remark 2** (Inversions in extension fields). For inversions in $\mathbb{F}_{p^k}$, we use the classical formula:

$$x^{-1} = \frac{x^{p^2+p^3+\ldots+p^{k-1}}}{x^{1+p^0+p^1+p^2+\ldots+p^{k-1}}} ,$$

where the denominator is the norm $\text{Norm}_{\mathbb{F}_{p^k}/\mathbb{F}_p}(x)$ in $\mathbb{F}_p$. It costs the computation of the numerator with Frobenius powers and multiplications in $\mathbb{F}_{p^k}$, the denominator computation knowing that it is in $\mathbb{F}_p$, one inversion in $\mathbb{F}_p$, and a coefficient-wise multiplication. One can factor the exponent to minimize the multiplications. For $\mathbb{F}_{p^k}$: $p + p^2 + p^3 + p^4 = p(p+1)(p^2+1)$ and the numerator costs $3f_i + 2m_5 = 38m$. The denominator costs $5m$. The total cost is $48m+i$. For $\mathbb{F}_{p^i}$, see [Mas20, page ix]: $10m+i$. $p + p^2 + p^3 + p^4 + p^5 = p(p^2+1)(1+p+p^2)$ and the numerator costs $4f_i + 3m_7 = 90m$. The denominator costs $7m + i$. The result costs $7m$ more compared to FM15-762. However, we exclude both curves from our benchmark comparison, since both curves are not competitive to other embedding degrees.

In Table 10 we observe that of all curve instances, BLS12-1150 requires the smallest number of $\mathbb{F}_p$-multiplications, however the base field prime $p$ is 1150-bit long. At this point it is not clear how BLS12-1150 compares to the other candidate curves, however this will become apparent in Section 5 where we discuss the benchmarking results. We now compare the curves of same embedding degree. For $k = 15$, FM15-762 clearly offers a more efficient pairing computation than its competitor BLS15-894 as it requires much less $\mathbb{F}_p$-multiplications and operates over a much smaller prime field. Recall also that FM15-762 has $\rho = 2$, while BLS15-894 has $\rho = 1.5$ showing that in this case, a larger $\rho$ offers better performances. However, we exclude both curves from our benchmark comparison, since both curves are not competitive to other embedding degrees.

For $k = 16$, all three curves KSS16-766, FM16-765 and AFG16-766 are defined over a prime field of almost the same size. The difference in the three families is that FM16-765 and AFG16-766 have $\rho = 2$, while on the contrary, KSS16-766 has $\rho \approx 1.25$. Both FM16-765 and AFG16-766 require almost the same number of $\mathbb{F}_p$-multiplications, which is approximately 20% less than the case of KSS16-766. This justifies our earlier claim that for some embedding degrees, having a larger $\rho$ is beneficial, improving the total pairing computation. We include all three curves for $k = 16$ to verify that the 20% improvement of FM16-765 and AFG16-766 over KSS16-766 is also captured in the benchmarking results.

In the case of $k = 18$, the curves KSS18-638 and SG18-638 operate on a smaller prime field compared to FM18-768. In this case, the theoretical analysis shows that SG18-638 requires less $\mathbb{F}_p$-multiplications for the pairing computation than the two competitors KSS18-638 and FM18-768 and in particular, it requires 15% less $\mathbb{F}_p$-multiplications than KSS18-638. In Section 5 we benchmark all three curves to have more conclusive evidence on their actual performance.

In the case of $k = 20$ the theoretical analysis is less inconclusive as to which curve is the best option for this embedding degree. Note that FM20/FST20-670 and SG20-670 operate on a prime field of the same size and require comparable number of $\mathbb{F}_p$-multiplications, with FM20/FST20-670 having a slight advantage (approximately 3% less $\mathbb{F}_p$-multiplications). On the other hand, although GG20b-575 requires 587$m$ more compared to FM20/FST20-670, the prime field size is significantly smaller in the case of GG20b-575. Nevertheless, we do not provide benchmarks for these curves, since they are certainly not competitive to other candidates for different embedding degrees. Note that these curves admit degree 4 twists, forcing elliptic curve point operations in the Miller loop to be executed over $\mathbb{F}_{p^2}$.

The curves BLS21-511, BLS24-509, BLS27-426, FST28-510 and GG28-500 offer the smallest primes $p$, compared to the previous candidate curves. Of these five curves, BLS24-509 is the most promising since it admits degree 6 twists and hence point additions
Table 10: Optimal ate pairing and final exponentiation cost estimates in terms of finite field multiplications. The name of each curve is derived from the name of the corresponding family in Table 5, plus the size of the prime $p$.

<table>
<thead>
<tr>
<th>Curve name</th>
<th>$r$ bits</th>
<th>Miller loop opt. ate (m)</th>
<th>final exp (m)</th>
<th>pairing</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLS12-1150</td>
<td>768</td>
<td>19288</td>
<td>245</td>
<td>14176</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>14421</td>
<td>33709</td>
</tr>
<tr>
<td>BLS15-894</td>
<td>597</td>
<td>25259</td>
<td>501</td>
<td>58014</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>58515</td>
<td>83774</td>
</tr>
<tr>
<td>FM15-762</td>
<td>381</td>
<td>16731</td>
<td>501</td>
<td>55321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>55822</td>
<td>72553</td>
</tr>
<tr>
<td>KSS16-766</td>
<td>605</td>
<td>16784</td>
<td>240</td>
<td>32836</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>33066</td>
<td>49850</td>
</tr>
<tr>
<td>FM16-765</td>
<td>384</td>
<td>10020</td>
<td>255</td>
<td>30024</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30279</td>
<td>40279</td>
</tr>
<tr>
<td>AFG16-766</td>
<td>384</td>
<td>10020</td>
<td>255</td>
<td>30282</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>30537</td>
<td>40557</td>
</tr>
<tr>
<td>KSS18-638</td>
<td>474</td>
<td>17433</td>
<td>480</td>
<td>27008</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>27488</td>
<td>44921</td>
</tr>
<tr>
<td>SG18-638</td>
<td>383</td>
<td>13351</td>
<td>480</td>
<td>24308</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24788</td>
<td>38139</td>
</tr>
<tr>
<td>FM18-768</td>
<td>384</td>
<td>13410</td>
<td>464</td>
<td>33184</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>33648</td>
<td>40299</td>
</tr>
<tr>
<td>KSS18-638</td>
<td>474</td>
<td>17433</td>
<td>480</td>
<td>27008</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>27488</td>
<td>44921</td>
</tr>
<tr>
<td>FM/FST20-670</td>
<td>448</td>
<td>18416</td>
<td>507</td>
<td>35276</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>35783</td>
<td>54199</td>
</tr>
<tr>
<td>SG20-670</td>
<td>383</td>
<td>16427</td>
<td>507</td>
<td>39152</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>39659</td>
<td>56086</td>
</tr>
<tr>
<td>FM20-575</td>
<td>379</td>
<td>17554</td>
<td>507</td>
<td>42017</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>42524</td>
<td>60078</td>
</tr>
<tr>
<td>BLS24-509</td>
<td>409</td>
<td>15345</td>
<td>658</td>
<td>24310</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24968</td>
<td>40313</td>
</tr>
<tr>
<td>BLS21-511</td>
<td>384</td>
<td>19321</td>
<td>717</td>
<td>62426</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>63143</td>
<td>82464</td>
</tr>
<tr>
<td>BLS27-426</td>
<td>384</td>
<td>22212</td>
<td>1185</td>
<td>88438</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>89907</td>
<td>112119</td>
</tr>
<tr>
<td>FST28-510</td>
<td>384</td>
<td>18940</td>
<td>859</td>
<td>66080</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>66939</td>
<td>75879</td>
</tr>
<tr>
<td>GG28-500</td>
<td>381</td>
<td>20326</td>
<td>859</td>
<td>78474</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>79339</td>
<td>99659</td>
</tr>
</tbody>
</table>

and doublings in the Miller loop are executed over $\mathbb{F}_p$. In BLS21-511, FST28-510 and GG28-500 additions and doublings are executed over $\mathbb{F}_{p^7}$, while in the case of BLS27-426 over $\mathbb{F}_{p^9}$. Furthermore, Table 10 shows that BLS21-511, FST28-510 and GG28-500 require almost 2× more $\mathbb{F}_p$-multiplications than BLS24-509 and BLS27-426 requires almost 3× more $\mathbb{F}_p$-multiplications than BLS24-509. Therefore, of these five curves we only benchmark BLS24-509.

To summarize, the curves that we have chosen to include in our optimized implementation step are, BLS12-1150, KSS16-766, FM16-765, AFG16-766, KSS18-638, SG18-638, FM18-768 and BLS24-509.

Normalized arithmetic cost. We compared the estimated cost of the Miller loop and final exponentiation in Table 10. However when the base field size is different, the comparison is not always obvious. For example, how do the $j = 1728$ curve families compare to each other in terms of Miller loop? Assuming an architecture with limbs of 64 bits, the fields need 12, 11, 9, and 8 machine-words for 768, 672, 576, and 512 bits resp. We apply the methodology from Aranha et al. [AFK$^{+}$13, Sect. 8]. Elements of $\mathbb{F}_p$ are assumed to be represented with $\ell = 1 + \lceil \log_2 p \rceil$ bits, packed in $w = \lfloor \ell / 64 \rfloor$ 64-bit machine-words. If the Montgomery representation is implemented, a multiplication with reduction in $\mathbb{F}_p$ has complexity $O(2w^2 + w)$. The authors deduce that one can estimate the ratio $m_{640} = 210/136 = 1.544m_{512}$ (for 10w). In the same way, we estimate $m_{576} = 171/136 = 1.257m_{512}$ (for 9w), $m_{768} = 253/136 = 1.860m_{512}$ (for 11w), $m_{768} = 300/136 = 2.205m_{512}$ (for 12 w). We finally obtain Table 11 and Figure 1.

4 Other pairing operations
In this section, we collect notes about how to implement other operations in pairing groups. We start by briefly surveying scalar multiplications and exponentiation algorithms, with
Table 11: Optimal ate pairing and final exponentiation cost estimates in terms of finite field multiplications, normalized w.r.t. a multiplication in $\mathbb{F}_p$ of 512 bits ($m = m_{512}$), with the ratios $m_{448} = 0.772m$, $m_{576} = 1.257m$, $m_{640} = 1.544m$, $m_{704} = 1.860m$, $m_{768} = 2.205m$, $m_{896} = 2.986m$, $m_{1152} = 4.897m$.

<table>
<thead>
<tr>
<th>Curve name</th>
<th>$r$ bits</th>
<th>Miller loop opt. ate ($m_{512}$)</th>
<th>final exp ($m_{512}$)</th>
<th>pairing total ($m_{512}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLS12-1150</td>
<td>768</td>
<td>94455</td>
<td>1200</td>
<td>69421</td>
</tr>
<tr>
<td>BLS15-894</td>
<td>597</td>
<td>75406</td>
<td>1496</td>
<td>173189</td>
</tr>
<tr>
<td>FM15-762</td>
<td>381</td>
<td>36907</td>
<td>1106</td>
<td>122032</td>
</tr>
<tr>
<td>KSS16-766</td>
<td>605</td>
<td>37024</td>
<td>530</td>
<td>72411</td>
</tr>
<tr>
<td>FM16-765</td>
<td>384</td>
<td>22103</td>
<td>563</td>
<td>66230</td>
</tr>
<tr>
<td>AFG16-766</td>
<td>384</td>
<td>22103</td>
<td>563</td>
<td>66799</td>
</tr>
<tr>
<td>KSS18-638</td>
<td>474</td>
<td>26919</td>
<td>742</td>
<td>41704</td>
</tr>
<tr>
<td>SG18-638</td>
<td>383</td>
<td>20616</td>
<td>742</td>
<td>37535</td>
</tr>
<tr>
<td>FM18-768</td>
<td>384</td>
<td>29581</td>
<td>1024</td>
<td>73200</td>
</tr>
<tr>
<td>FST20-670</td>
<td>448</td>
<td>34260</td>
<td>944</td>
<td>65624</td>
</tr>
<tr>
<td>SG20-670</td>
<td>383</td>
<td>30560</td>
<td>944</td>
<td>72835</td>
</tr>
<tr>
<td>GG20b-575</td>
<td>379</td>
<td>22072</td>
<td>638</td>
<td>52831</td>
</tr>
<tr>
<td>BLS21-511</td>
<td>384</td>
<td>19321</td>
<td>717</td>
<td>62426</td>
</tr>
<tr>
<td>BLS24-509</td>
<td>409</td>
<td>15345</td>
<td>658</td>
<td>24310</td>
</tr>
<tr>
<td>BLS27-426</td>
<td>383</td>
<td>17149</td>
<td>915</td>
<td>68280</td>
</tr>
<tr>
<td>FST28-510</td>
<td>384</td>
<td>18940</td>
<td>859</td>
<td>56080</td>
</tr>
<tr>
<td>GG28-500</td>
<td>381</td>
<td>20326</td>
<td>859</td>
<td>78474</td>
</tr>
</tbody>
</table>

Figure 1: Estimates of Miller loop in terms of normalized multiplications in $\mathbb{F}_p$ of 512 bits.
their constant-time counterparts. We then continue by giving closed formulas for hashing and cofactor-clearing according to the latest state-of-the-art, and finish by enumerating techniques for subgroup membership testing.

4.1 Scalar multiplication and exponentiation in pairing groups

Pairing-friendly curves are typically equipped with additional efficient endomorphisms, such that the widely known Gallant-Lambert-Vanstone (GLV) [GLV01] and Galbraith-Lin-Scott (GLS) [GLS11] scalar multiplication and exponentiation algorithms are the most efficient.

**Endomorphisms on \( G_1 \) and GLV.** All the curves we consider have either \( D = 1 \) and \( j \)-invariant 1728, or \( D = 3 \) and \( j \)-invariant 0. The curves of \( j \)-invariant 1728 (e.g. KSS16) have an endomorphism \( \phi \) of the form \((x, y) \mapsto (-x, iy)\) where \( i^2 = -1 \in \mathbb{F}_p \) as \( p \equiv 1 \mod 4 \), of characteristic polynomial \( \chi_{\phi}(X) = X^2 + 1 \). The curves of \( j \)-invariant 0 (e.g. KSS16, BLS) have an endomorphism \( \phi \) of the form \((x, y) \mapsto (\omega x, y)\) where \( \omega \) is a primitive third root of unity in \( \mathbb{F}_p \), \( \omega^2 + \omega + 1 = 0 \mod p \), as \( p \equiv 1 \mod 3 \), of characteristic polynomial \( \chi_{\phi}(X) = X^3 + X + 1 \). In the case of \( G_1 \), it is well-known that the GLV technique can be used together with the endomorphism \( \phi \) to decompose a scalar \( \ell \in \mathbb{Z}_r \) into half-sized subscalars \( (\ell_0, \ell_1) \) such that \( \ell \equiv \ell_0 + \ell_1 \lambda_p \mod r \), with \( \lambda_p \in \mathbb{Z}_r \) the eigenvalue of the \( \phi \) map. The scalar multiplication can then proceed through the formula \( \ell P = \ell_0 P + \ell_1 \phi(P) \), for \( P \in G_1 \). A widely used algorithm to compute scalar multiplication combines the \( w \)-NAF scalar recoding algorithm and interleaving of the two smaller scalar multiplications to save point doublings. Because applying \( \phi \) is very efficient in our pairing-friendly curves, we can compute a single precomputation table with \( 2^{w-2} \) points for \( P \) and apply \( \phi \) to points obtained from the table. Constant-time versions of the algorithm typically replace the recoding process with a regular \( w \)-NAF expansion [JT09], which increases the density of non-zero digits from \( \frac{1}{w+1} \) to \( \frac{1}{w-1} \) and force secure table lookups to prevent leaking what points from the table are being used in the main loop [OLAR13]. From the point of view of efficiency, this approach compares favorably to alternatives [FLS15], by computing fewer point additions at the same cost in precomputed storage.

**Endomorphisms on \( G_2 \) and \( G_T \), and GLS.** In the case of \( G_2 \) and \( G_T \), the GLS technique is used instead, assuming again that an efficient endomorphism is available. For \( G_2 \), the map \( \psi \) is constructed as \( (\tau^{-1} \circ \pi \circ \tau) \), where \( \tau \) is the twisting isomorphism \( \tau : E^2(\mathbb{F}_{p^k}) \rightarrow E(\mathbb{F}_{p^k}) \) and \( \pi \) is the Frobenius map over \( E(\mathbb{F}_{p^k}) \). This endomorphism \( \psi \) has characteristic polynomial \( X^2 - \ell X + p \). It is well-known that \( \psi \) has eigenvalue \( p \equiv t - 1 \mod r \) on \( G_2 \), by definition of \( G_2 = \ker(\pi - [p]) \) [HSV06, GS08, DLZZ23]. For \( G_T \), we choose \( \phi \) as a small multiple or power of the Frobenius \( \pi_p \in \mathbb{F}_{p^k} \). In both cases (\( G_2 \) and \( G_T \)), we can represent the scalar or exponent in base \( p \) (the eigenvalue) by decomposing it in \( \varphi(k) \) subscalars, where \( \varphi \) is Euler’s totient function, such that \( \ell \equiv \sum_{0 \leq i < \varphi(k)} \ell_i p^i \mod r \). We can then use the Frobenius map to evaluate a scalar multiplication in \( G_2 \) as \( [\ell]Q = \sum_{0 \leq i < \varphi(k)} [\ell_i]p^i(Q) \) in interleaved \( w \)-NAF fashion, or the analogue in \( G_T \) after translating to multiplicative notation. The endomorphisms \( \psi \) in \( G_2 \) and \( \pi_p \) in \( G_T \) are quite efficient to evaluate, but it is typical to precompute the various tables over \( \psi^i(Q) \) to avoid the cost of applying \( \psi^i \) dynamically in succession. The same notes about constant-time implementation apply, noting that the endomorphisms are more expensive to evaluate than in the case of \( G_1 \) and now there are \( \varphi(k) \) subscalars involved.

One simple implementation trick is useful when supporting multiple pairing-friendly curves simultaneously. Decomposing the scalar into base \( p \) can be performed by finding a suitable relationship between the prime subgroup order \( r \), the curve generation seed \( u \) and the prime modulus \( p \), such that we can exploit the fact that \( p \) and \( u \) are related modulo \( r \) to decompose in base \( u \) instead. After writing the seed \( u \) as an expression involving
a few powers of \( p \) for all curves, then we can decompose scalar \( \ell \) in base \( u \) instead, and perform the scalar recoding with successive (constant-time) integer divisions. Many suitable relationships come directly from the Optimal ate formulas in Table 6, but some require further algebraic manipulation. For AFG16 curves, we exploit \( u \equiv -p^2 \mod r \), to define \( \psi \) as the inverse \( p^2 \)-power Frobenius. For KSS16 curves, we exploit \( u \equiv (2p^5 - p) \mod r \) instead. For KSS18 curves, we exploit \( u = (p^3 - 3)p \mod r \), and for FM18 curves we exploit \( u \equiv (p^3 - p) \mod r \). For SG18 curves, we exploit \( -3u = (2p^2 - p^3) \mod r \) to recode in base \( 3u \) and define \( \psi \) as the \((p^5 - 2p^2)\)-power Frobenius, while noting that inversion in both \( \mathbb{G}_2 \) and \( \mathbb{G}_T \) can be efficiently computed.

4.2 Hashing to pairing source groups

Many cryptographic protocols involve hashing arbitrary bit strings to a group of points in an elliptic curve. The security requirements are generally the same for an ideal collision-resistant hash function, such that protocols can rely on the hash function to behave as closely as possible to a conventional random oracle. This notion is formalized as indifferentiability to a random oracle, or the infeasibility of differentiating the hash function from a random oracle [BCI+10] with non-trivial probability, given a polynomially-bounded number of queries. Given an indifferentiable hash function \( h \) bit strings as inputs/outputs, we can hash to curves by initially constructing a function to hash to a field \( \mathbb{F}_q \) and then an encoding function to map field elements to points in the curve, followed by a scalar multiplication by a cofactor to map the output to the right subgroup. From an implementation security standpoint, the full process should also be efficient when evaluated in constant time [WB19, AHST23].

The state-of-the-art approach for indifferentiable hashing to pairing-friendly elliptic curves is the SwiftEC algorithm [CSRT22] and its generalization [Kos24] to broader classes of elliptic curves. The former is restricted to curves with \( q \equiv 1 \mod 3 \), with either odd order or order divisible by 4, having \( a = 0 \) (or \( j \)-invariant 0) as a special simpler case. Koshelev's generalization expands the constructions to all curves with \( q \equiv 1 \mod 3 \) with non-zero \( j \)-invariant, which covers our remaining curves with \( b = 0 \) (or \( j \)-invariant 1728).

Let \( \ell = \lceil \log_2 q \rceil \) be the bit length required to encode an element from \( \mathbb{F}_q \). Function \( H : \{0,1\}^* \to E(\mathbb{F}_q) \) can be constructed by the composition of functions \( H = [c] \circ f \circ \eta \circ h \), where \( h : \{0,1\}^* \to \{0,1\}^{2\ell + 1} \) is a deterministic indifferentiable hash function, \( \eta : \{0,1\}^{2\ell + 1} \to \mathbb{F}_q^2 \times \{0,1\} \) is an encoding to the field (with an additional sign bit), \( f : \mathbb{F}_q^2 \times \{0,1\} \to E(\mathbb{F}_q) \) is an admissible encoding to the group of points (the additional bit is to choose the \( y \)-coordinate), and \([c]\) corresponds to scalar multiplication by the cofactor. We select \( h \) as the message expansion function XMD [FHSS+23] instantiated with SHA256 as the underlying hash function, and elaborate on the other choices below.

Given \((t_1, t_2) \in \mathbb{F}_q^2\) and a sign bit \( s \), define the constants \( \tau = \sqrt{-3} \in \mathbb{F}_q \) and \( \omega = \frac{\tau + 1}{2} \), and respectively the denominators \( d_j \) and numerators \( n_j \) for fractions \( X_j = \frac{n_j}{d_j} \) with \( j \in \{1, 2, 3\} \):

\[
\begin{align*}
\frac{d_1}{d_3} &= -2\tau \omega (t_0^6 + 2^33^3\tau t_1^3) + 2^6a^3 + 2^33^3\tau t_1^3t_2^2) t_1 \\
\frac{d_2}{d_3} &= -2^4 3^3 (t_1^2 - 2^2a) t_1^3 t_2^2 \\
n_1 &= t_0^6 + 2^2\omega a t_1^6 + 2^33^3\tau t_1^3 + 2^23^3\tau \omega^2 a t_1^3 + 2^6a^3 t_1^3 + 2^6\omega^2 a^4 + 2^33^3\tau (\omega^2 t_1^3 + 2^2a) t_1^3 t_2^2 \\
n_2 &= t_0^6 + 2^2\omega a t_1^6 + 2^33^3\tau t_1^3 + 2^23^3\tau \omega a t_1^3 + 2^6a^3 t_1^3 + 2^6\omega a^4 + 2^33^3\tau (\omega t_1^3 + 2^2a) t_1^3 t_2^2 \\
n_3 &= t_0^6 + 2^33^3\tau t_1^3 + 2^2(2^3a - 3^3b^3) t_1^3 + 2^43^3\tau b^3 t_1^3 + 2^3 a^6 \\
&= -2^33^3\tau (t_0^6 - 2^23a t_1^6 - 2^23^3\tau t_1^3 - 2^43^3\tau b^3 t_1^3 + 2^6a^3 - 2^33^3\tau t_1^3 t_2^2) t_1^3 t_2^2.
\end{align*}
\]
When all of the $X_j$ fractions are defined ($d_j \neq 0$), we select $X_j$ with highest $j$ as the $x$-coordinate, such that $g(X_j) = X_j^2 + aX_j + b$ is a quadratic residue, to compute $Y = \sqrt{g(X_j)}$. We then take the additional output bit of $\eta$ to select one of the square roots, and the output is $(X_j, y)$. Otherwise, we output the point at infinity $\mathcal{O}$, which happens only with negligible probability under the assumption that $h$ is indifferentiable from a random oracle. More formally,

$$f(t_1, t_2, s) = \begin{cases} 
\mathcal{O}, & \text{if any } d_j = 0, \text{ with } j \in \{1, 2, 3\}, \\
(X_i, (1 - 2s)Y), & \text{with } i = \max \left\{ j \bigg| \left( \frac{g(X_j)}{q} \right) = 1 \right\} \text{ otherwise.}
\end{cases}$$

The baseline cost for computing $X_j$ is one square-root extraction to compute $Y$, two tests for squaredness, one squaring and one multiplication in $\mathbb{F}_q$ to compute each $g(X_j)$ for pairing-friendly curves. Testing for squaredness in $\mathbb{F}_{p^k}$ costs 1 Legendre symbol computation, $(k - 1)$ Frobenius $p^k$-powers and $(k - 1)$ multiplications in $\mathbb{F}_{p^k}$ [AR14]. Multiplications by $\omega$ or $\tau$ consist in $k$ multiplications in $\mathbb{F}_p$, since the constants lie in the base field. Under the guarantee that at least one of the $g(X_j)$ is a square, we can initially assume that $g(X_1)$ is a square and update the choice of $X_j$ after testing the other two values with one symbol computation each. For curves with either $a = 0$ or $b = 0$, we can evaluate the curve equation with one squaring and one multiplication in $\mathbb{F}_q$. Below we discuss the various cases in more details and compute the total operation counts.

**The case $b = 0$.** We have two subcases in our curve selection. For $\mathbb{G}_2$, we have that multiplicative twists with coefficient $a' = \xi$, for some small non-residue $\xi$ in a subfield, allowing for multiplications by $a'$ to be cheap and evaluated with additions only. By substituting $b = 0$ in the expressions above, after optimizing for common subexpressions and using Montgomery’s simultaneous inversion technique, we can evaluate the composition $(f \circ \eta \circ h)$ at the cost of 1 square-root, 2 Legendre symbols and $5s_{k/d} + 19m_{k/d} + 5(k/d)m + 1_{k/d} + ((k/d) - 1)(f_{k/d} + m_{k/d})$ while adding the baseline costs to evaluating the formulas:

- $h_0 = t_1^2$, $h_1 = h_0^2$, $h_2 = 4a'$, $h_3 = 64a'^3$, $h_4 = h_0h_1 + h_3$, $h_5 = t_2^2$, $h_6 = \tau t_1$
- $h_7 = 24h_0h_5h_6$, $h_8 = h_0h_3$, $h_9 = h_2^2$, $h_{10} = \omega(4a' h_4 + h_0 h_7)$
- $d_1 = -2h_0\omega(h_4 + h_7)$, $d_2 = d_1\omega$, $d_3 = -432(h_0 - h_2)^2h_1h_5$
- $n_1 = h_9 + h_8 + 4a' h_7 + h_{10} \omega$, $n_2 = h_9 + h_8 + 4a' h_7 + h_{10}$
- $n_3 = h_1(h_9 + 2h_8) + 4096a'^6 - h_7(h_4 - 12a'(h_1 + 4a' h_0) - h_7)$.

By further specializing to the case $a = 1$ for $\mathbb{G}_1$, the formulas cost 1 square-root, 2 Legendre symbols and $5s + 23m + 1$:

- $h_0 = t_1^2$, $h_1 = h_0^2$, $h_2 = h_1h_0 + 64$, $h_5 = t_2^2$, $h_6 = \tau t_1$
- $h_7 = 24h_0h_5h_6$, $h_9 = h_2^2$, $h_{10} = \omega(4h_4 + h_0 h_7)$
- $d_1 = -2h_0\omega(h_4 + h_7)$, $d_2 = d_1\omega$, $d_3 = -432(h_0 - 4)^2h_1h_5$
- $n_1 = h_9 + 4(16h_0 + h_7) + h_{10} \omega$, $n_2 = h_9 + 4(16h_0 + h_7) + h_{10}$
- $n_3 = h_1(h_9 + 8(16h_0)) + 4096 - h_7(h_4 - 3(4h_1 + 16h_0) - h_7)$.

**The case $a = 0$.** For this case, we turn instead to the SwiftEC algorithm, which can be evaluated through 1 square-root extraction, 2 Legendre symbol computations and the
additional costs of $7s_{k/d} + 11m_{k/d} + (k/d)m + i_{k/d} + ((k/d) - 1)(f_{k/d} + m_{k/d})$ in $G_2$, and $7s + 12m + 1$ in $G_1$:

$$h_0 = t_1^2, \ h_1 = t_2^2, \ h_2 = h_0 + b - h_1, \ h_3 = 2h_1 + h_2, \ h_6 = \tau t_1, \ h_7 = h_0h_6, \ h_8 = 2h_6t_2$$

$$d_1 = 2h_3h_8, \ n_1 = h_8(h_7 - t_1h_3), \ n_2 = (2h_3)^2$$

$$X_1 = n_1/d_1, \ X_2 = -t_1 - n_1/d_1, \ X_3 = t_1 + (n_2/d_1)^2.$$  

### 4.2.1 Cofactor clearing for $G_1$.

As in Section 2, we denote $c_1$ the cofactor of the curve, that is, $\#E(\mathbb{F}_p) = c_1 \cdot r$, and define the **cofactor clearing** operation as

$$E(\mathbb{F}_p) \rightarrow \mathbb{G}_1 \quad P \mapsto [c_1]P$$

In Appendix B Table 21 we give the parameterized formulas for the cofactor $c_1$ of $G_1$. Because $c_1$ might be quite large, there are two strategies to speed-up the multiplication-by-$c_1$ map $[c_1]$. First, the curves admit an endomorphism for fast GLV scalar multiplication. Second, when the cofactor $c_1$ has some square factor $n_1^2$ satisfying the properties of Schoof’s [Sch87, Proposition 3.7] (also in [EHG22, Theorem 1]), the strategy of Wahby–Boneh applies [WB19] (multiplying by $n_1$ instead of $n_1^2$).

It is important to note that the two techniques (GLV and Wahby–Boneh) are *orthogonal* w.r.t. the $\mathbb{F}_p$-rational curve endomorphism $\phi$. In other words, we cannot *combine* the two techniques, we apply them respectively on disjoint subgroups. Following [HGP22], we aim at obtaining the structure of $E(\mathbb{F}_p)$ so as to identify the respective subgroups where to apply each technique:

$$E(\mathbb{F}_p) \simeq \mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_2\mathbb{Z}, \ n_1 \mid n_2 \quad (2)$$

so that $E[n_1] \subset E(\mathbb{F}_p)$, $n_1^2 \mid E(\mathbb{F}_p)$. In our context, we know that $r$ is prime, and $r^2$ does not divide the curve order over $\mathbb{F}_p$, so that $r$ divides $n_2$ in Eq. (2) that we rewrite as

$$E(\mathbb{F}_p) \simeq \mathbb{Z}/n_1\mathbb{Z} \oplus \mathbb{Z}/n_1c_1r\mathbb{Z}, \ n_1^2c_1 = c_1 \quad (3)$$

We apply the Wahby–Boneh technique to clear the square factor $n_1^2$, followed by the GLV technique to clear the factor $c_1' = \#E(\mathbb{F}_p)/(n_1^2r)$.

**Faster co-factor clearing with the GLV method.** To improve the cofactor clearing step, one would like to use the endomorphism $\phi$. For that one needs to identify the cyclic subgroup of $E(\mathbb{F}_p)$ stable under $\phi$, so that $\phi$ acts as a multiplication by an eigenvalue $\lambda_\phi$. With the notations above, this subgroup has order $c_1' = \#E(\mathbb{F}_p)/(n_1^2r)$. We compute in SageMath the parameters: eigenvalue $\lambda_\phi$ modulo $c_1'$, short scalars $(a_0, a_1)$ so that $a_0 + a_1\lambda_\phi$ is a multiple of $c_1'$. The multiplication-by-$c_1'$ map becomes

$$E(\mathbb{F}_p) \rightarrow E(\mathbb{F}_p) \quad P \mapsto [a_0]P + [a_1]\phi(P).$$

**Faster co-factor clearing with the Wahby–Boneh technique and Schoof’s theorem.** We apply the Wahby–Boneh technique to the appropriate subgroup of order $n_1$ (Eq. (2)), which satisfies the conditions of Theorem 1. We compute in SageMath the parameter $n_1$ for each curve.
Theorem 1 ([Sch87, Proposition 3.7]). Let $E$ be an elliptic curve over $F_p$ and $n_1 \in \mathbb{Z}_{\geq 1}$ with $p \nmid n_1$. Let $\pi_p$ denote the Frobenius endomorphism of $E$ and $t$ its trace. Then,

$$E[n_1] \subset E(F_p) \iff \begin{cases} n_1^2 \mid \#E(F_p), \\ n_1 \mid p-1 \text{ and} \\ \pi_p \in \mathbb{Z} \text{ or } \mathcal{O}(\frac{p-1}{n_1}) \subset \text{End}_{F_p}(E). \end{cases}$$

To conclude, we are looking for a decomposition of the curve order so that the Wahby–Boneh technique applies to $n_1$ and the GLV technique applies to $c_1'$:

$$\#E(F_p) = n_1^2 \cdot c_1' \cdot r . \quad (4)$$

4.2.2 Cofactor clearing for $G_2$

Recently Yu Dai et al. [DLZZ23] extended on finding efficient formulas and provided a Magma script [Dai23]. We rely on their work to obtain the formulas for our curves. As the formulas are quite long for $G_2$, we implemented them in SageMath to validate the equations. The general strategy is the same as for $G_1$, but with $\psi$ instead. Note that [FAG20] investigated SG curves.

4.3 Subgroup membership testing

Yu Dai et al. [DLZZ23] consider subgroup membership testing on pairing-friendly curves and generalize Scott’s technique [Sco21]. The strategy dates back to GLV idea and for $G_1$ it consists in finding a short vector $(a_0, a_1)$ such that $a_0 + a_1 \lambda_0 \equiv 0 \pmod{r}$, then testing if $[a_0]P + [a_1]\phi(P) = \mathcal{O}$. Dai et al. solve the problem for KSS16 and KSS18 curves where the textbook formula of short vector $(a_0, a_1) = (1, \lambda_0 \pmod{r})$, resp. $(1, \lambda_0 + 1 \pmod{r})$ induces a multiplication by a multiple of $r$ but which is not coprime to the cofactor $c_1$. Moreover they obtain a generic technique that can be applied to all curves for $G_1$ and $G_2$. We apply their criterion [DLZZ23, Theorem 1] to our curves in Table 12 and solve the problem of cofactors for $G_1$ in Table 15.

4.3.1 Subgroup membership testing in $G_1$

Theorem 2 ([DLZZ23, Theorem 3] $G_1$ for $j = 0$ or $j = 1728$ curves). Let $E$ be an ordinary elliptic curve defined over a finite field $F_q$ with $j$-invariant 0 or 1728, and $r$ a large prime such that $r \parallel \#E(F_q)$. Let $\phi$ be a GLV endomorphism on $E$, and act as multiplication by an integer $\lambda_0$ in $G_1$. Let $(a_0, a_1) \in \mathbb{Z}^2$ with $a_0 + a_1 \lambda_0 \equiv 0 \pmod{r}$. Assume

- $\gcd(a_0^2 - a_0 a_1 + a_1^2, \#E(F_q)) = r$, if $j(E) = 0$,
- $\gcd(a_0^2 + a_1^2, \#E(F_q)) = r$, if $j(E) = 1728$.

Given a non-zero point $P \in E(F_q)$, then $P \in G_1$ if and only if $[a_0]P + [a_1]\phi(P) = \mathcal{O}$.

For $G_1$ membership testing there is a problem for KSS-like curves (KSS16, KSS18, GG20, GG28). The naive vector is $(a_0, a_1) = (1, \lambda_0 \pmod{r})$ ($D = 1$), resp. $(a_0, a_1) = (1, \lambda_0 + 1 \pmod{r})$ ($D = 3$) or the half-gcd of $(r, \lambda_0)$. To apply the criterion of Theorem 2, one computes $a_0^2 + a_1^2 = 1 + \lambda_0^2$, resp. $a_0^2 - a_0 a_1 + a_1^2 = 1 + \lambda_0 + \lambda_0^2$ but this value is not coprime to the cofactor $c_1$ (see Table 12). In other words if there is a common divisor $c_1'$ of $\text{Res}(\chi_0(X), a_0 + a_1 X)$ and $c_1$, the test returns true for points of order a divisor $c_1'r$. 
Given a non-identity point \( \lambda \) be the twisting isomorphism, where the naive vector \((a_0, a_1)\) are two strategies to reduce the basis, one experimental involving LLL in [DLZZ23] and GG28 curves. It means that the naive vector \((a_0, a_1)\) is not as short as possible. There are two strategies to reduce the basis, one experimental involving LLL in [DLZZ23] and one theoretical involving shrinking a basis of a vector space in [Smi15]. We summarise our results in Table 15.

### 4.3.2 Subgroup membership testing for \( \mathbb{G}_2 \)

**Theorem 3** ([DLZZ23, Theorem 1]) \( \mathbb{G}_2 \) for curves with a twist. Let \( E \) be an ordinary elliptic curve over \( \mathbb{F}_q \) and \( t \) the trace of the Frobenius endomorphism \( \pi \). Let \( \phi : E' \rightarrow \mathbb{G}_2 \) be the twisting isomorphism, where \( E' \) is defined over \( \mathbb{F}_{p^r} \). Let \( r \) be a large prime such that \( \|E(\mathbb{F}_q)\| \approx \#E(\mathbb{F}_{p^r}) \). Define \( \psi = \phi^{-1} \circ \pi \circ \phi \) with the characteristic polynomial \( g(\psi) = \psi^t - t \cdot \psi + p \). Let \( \eta = \sum_{i=0}^{s} c_i \cdot p^i \) be a multiple of \( r \) and \( f(\psi) = \sum_{i=0}^{s} c_i \psi^i \) a polynomial with respect to \( \psi \). Denote by \( b_0 + b_1 \psi \) the remainder for \( f(\psi) \) divided by \( g(\psi) \), i.e.,

\[
b_0 + b_1 \psi = f(\psi) \mod g(\psi) \,.
\]

Assume

\[
gcd(b_0^2 + b_0b_1t + b_1^2p, \#E' \mathbb{F}_{p^r})) = r \,.
\]

Given a non-identity point \( Q \in E'(\mathbb{F}_{p^r}) \), then \( Q \in \mathbb{G}_2 = E'(\mathbb{F}_{p^r})[r] \) if and only if \( f(\psi)(Q) = \mathcal{O}_{E'} \).

As noted by Yu Dai et al. [DLZZ23], the formulas for optimal ate pairing computation and fast \( \mathbb{G}_2 \) membership testing are very similar. For the optimal ate pairing, the formula parameters should satisfy

\[
\sum_{i=0}^{\ell} c_i q^i = m \cdot r \text{ and } mkq_1^{-1} \neq \frac{q_1^{-1} - 1}{r} \sum_{i=0}^{\ell} ic_i q_1^{-1} \mod r
\]
Table 13: $G_1$ cofactor clearing, where $\lambda_0$ is the eigenvalue of the endomorphism $\phi$, $\lambda_0 \mod c'_1$ is given such that $\lambda_0 \mod r$ matches the data in Table 12, the short vector $(a_0, a_1)$ means evaluating $R \leftarrow [a_0]P + [a_1]\phi(P)$.

<table>
<thead>
<tr>
<th>Curve</th>
<th>$\lambda_0 \mod c'_1$</th>
<th>short vector $(a_0, a_1)$</th>
<th>criterion</th>
<th>cofactor clearing</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLS12</td>
<td>$x + 1$</td>
<td>$(1, -1)$</td>
<td>$c'_0 = a_0 a_1 + a_1 c'_1$</td>
<td>$Q = (x + 1)/P$; $R = c'_0 + (x + 1)/Q$</td>
</tr>
<tr>
<td>FM15</td>
<td>$(x^4 + 3x^3 + 2x^2 - 4x - 8)/7$</td>
<td>$(1x + 2^2 - 4x)/4$; $(4x^2 - 2x - 3)/3$</td>
<td>$c'_0 = a_0 a_1 + c'_1$</td>
<td>$Q = [a_1]P$, $R = [a_1]Q + [a_1](0, a_1)$</td>
</tr>
<tr>
<td>KSS16</td>
<td>$y(x + 1)/2$</td>
<td>$(1, -a_0 a_1 - a_1 c'_1)$</td>
<td>$Q = P + 3a_0 \mod c'_1/</td>
<td>P</td>
</tr>
<tr>
<td>FM16</td>
<td>$(1, a_0 a_1 + a_1 c'_1)$</td>
<td>$c'_0 + a_1 c'_1 = c'_1$</td>
<td>$Q = [a_1]P$, $R = [a_1]Q + [a_1](0, a_1)$</td>
<td></td>
</tr>
<tr>
<td>AFG16</td>
<td>$x + 2$</td>
<td>$(1, a_0 a_1 + a_1 c'_1)$</td>
<td>$Q = P + [a_1]P + a_1 \mod c'_1[</td>
<td>P</td>
</tr>
<tr>
<td>SG18</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FM18</td>
<td>$-(x^3 + 3x^2 + 3x + 1)/2$</td>
<td>$(2x - 1)/3$; $x^2 + (x - 1)/3$</td>
<td>$c'_0 = a_0 a_1 + a_1 c'_1$</td>
<td>$Q = [(x - 1)/3]P$, $R = ([x(x + 1)] + Q + P + ((x + 1))/Q(Q)$</td>
</tr>
<tr>
<td>FT20</td>
<td>$x$</td>
<td>$(1, -1)$</td>
<td>$c'_0 + a_1 c'_1 = c'_1$</td>
<td>$Q = [x(x - 1)/2]P$, $R = + (x - 1)/Q(Q)$</td>
</tr>
<tr>
<td>SG20</td>
<td>$-(a + x + a)$</td>
<td>$(2x - 1, x)$</td>
<td>$c'_0 + a_1 c'_1 = c'_1$</td>
<td>$Q = [x(x - 1)/2]P$, $R = + (x + 1)/Q(Q)$</td>
</tr>
<tr>
<td>GG20b</td>
<td>$x + 2$ mod $70$, $x^2 + 2x + 1$</td>
<td>$(1, 1)/2$</td>
<td>$c'_0 = a_0 a_1 + a_1 c'_1$</td>
<td>$Q = [(x-1)/2]P$, $R = + (x + 1)/Q(Q)$</td>
</tr>
<tr>
<td>BLS12</td>
<td>$x$</td>
<td>$(1, -x)$</td>
<td>$c'_0 + a_1 c'_1 = c'_1$</td>
<td>$Q = [x(x - 1)/2]P$, $R = + (x - 1)/Q(Q)$</td>
</tr>
<tr>
<td>FM16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 14: From [DLZZ23, Table 2]. Note that for KSS18 $G_2$, we translate the short vector $(2x/7, 1, 0, x/7, 0, 0)$ to $(0, 2x/7, 1, 0, x/7)$ otherwise $\sum a_i \mu_i$ produces 0, not a multiple of $r(x)$. For KSS16, we use the data from [DLZZ23, §5.2.2].

<table>
<thead>
<tr>
<th>Curve</th>
<th>$G_1$</th>
<th>short vector $(a_i)$</th>
<th>criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>BN</td>
<td>$G_2$</td>
<td>$(x + 1, x, x, -2x)$</td>
<td></td>
</tr>
<tr>
<td>BLS12</td>
<td>$G_2$</td>
<td>$(x, -1, 0, 0)$</td>
<td></td>
</tr>
<tr>
<td>KSS16</td>
<td>$G_2$</td>
<td>$(31x^4 + 625)/5700, (12x^4 + 625)/5700$</td>
<td>$a_0^2 + a_1^2 = r$, $a_0 + a_1 (\lambda_0 \mod r) = -17r$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$= (31x^4)/5 + 1/14, -17(x(x)/5)^4 + 1/14$</td>
<td>$-3a_0 - 1 + 17a_0 (\lambda_0 \mod r) = -17^2r$</td>
</tr>
<tr>
<td>G2</td>
<td>$G_2$</td>
<td>$1/70, 11x + 5, 9x + 15, -3x - 5, -3x - 5, 13x - 25, -7x + 35, -x - 25, -11x + 5$</td>
<td>$\sum a_i \mu_i = 127r$</td>
</tr>
<tr>
<td>KSS16</td>
<td>$G_2$</td>
<td>$((x/7)^4 - 18(x/7)^3 - 1)$</td>
<td>$a_0^2 - a_0 a_1 + a_1^2 = r$, $a_0 + a_1 (\lambda_0 \mod r) = -18r$</td>
</tr>
<tr>
<td></td>
<td>$G_2$</td>
<td>$(0, 0, 2x/7, 1, 0, x/7)$</td>
<td>$a_0^2 a_2 + a_1^2 a_2 + a_0 a_1^2 = -18^2r$</td>
</tr>
</tbody>
</table>

Table 15: $G_1$ membership testing, solving the issues in Table 12. The endomorphism $\phi$ has eigenvalue $\lambda_0$, the short vector $(a_0, a_1)$ satisfies $a_0^2 + a_1^2 = r$ ($j = 1728$), resp. $a_0^2 - a_0 a_1 + a_1^2 = r$ ($j = 0$), where $a_0$, $a_1$ are integers. Note that for KSS16, $x \equiv 25, 45 \mod 70$ hence 5 $| x$; for KSS18, $x \equiv 14 \mod 21$ hence 7 $| x$; for GG20b, $x \equiv 1465, 1565 \mod 2050$ hence 5 $| x$.

<table>
<thead>
<tr>
<th>Curve</th>
<th>$\lambda_0 \mod r$</th>
<th>$(a_0, a_1)$ s.t. $a_0^2 + a_1^2 = r$, resp $a_0^2 - a_0 a_1 + a_1^2 = r$</th>
<th>observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSS16</td>
<td>$(x^4 + 24x)/7$</td>
<td>$(-443(x/9)^4 - 1114(x/9)^5 + 514)/4$</td>
<td>$a_0 = -443r + 157$</td>
</tr>
<tr>
<td>KSS18</td>
<td>$x^3 + 18$</td>
<td>$(x^3(x/7)^4 - 1, (x/7)^5)$</td>
<td>$a_0 = 19a_1 + 1$</td>
</tr>
<tr>
<td>GG18</td>
<td>$(x^3 + 1)/4$</td>
<td>$-(3x^2, 3x^2 + 1)$</td>
<td>$a_1 = -a_0 + 1$</td>
</tr>
<tr>
<td>GG20b</td>
<td>$(x^3 - 38)/41$</td>
<td>$(61(x/5)^4 - 54(x/5)^4 + 31(x/5)^4 - 14(x/5)^5 + 5)/41$, $(148(x/5)^4 - 47(x/5)^4 + 81(x/5)^5 - 3(x/5)^5 + 4)/41$</td>
<td>$u = 30 \mod 41$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$(92(x/5)^4 - 63(x/5)^4 + 32(x/5)^5 - 13(x/5)^5 + 4)/41$, $(131(x/5)^4 - 34(x/5)^5 - 6(x/5)^5 + 5)/41$</td>
<td>$u = 7 \mod 41$</td>
</tr>
<tr>
<td>BLS27</td>
<td>$x^9$</td>
<td>$(-4x^2 + 1)/3, (x^2 + 2)^3$</td>
<td>$a_1 = -a_0 + 1$</td>
</tr>
<tr>
<td>GG28</td>
<td>$(x^7 + 278)/29$</td>
<td>$((2x^4 + 9x^2 + 26x + 92x^2 + 106x^2 + 120x - 14)/29$, $(5x^8 + 8x^7 + 7x^6 - 12x^5 - 83x^2 - 272x - 673)/29$, $(5x^8 + 12x^7 + 23x^6 + 32x^5 + 13x^4 - 108x - 497)/29$, $(2x^4 - x^3 - 14x^2 - 51x^3 - 134x^5 - 281x^4 - 454x)/29$</td>
<td>$u = 14 \mod 29$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$x = 19 \mod 29$</td>
<td>$u = 19 \mod 29$</td>
</tr>
</tbody>
</table>
for some non-zero integer $m$ and short integers $c_i$ [Ver10, Theorem 1].

As explained in [Smi15, HGP22, DLZZ23], the points on the $d$-th twist $E'(\mathbb{F}_{p^d})$ containing $G_2$ have eigenvalue $\lambda_\psi$ under Galbraith–Scott endomorphism $\psi = \tau^{-1} \circ \pi \circ \tau$, where $\psi$ is a root of the characteristic polynomial $\chi_\psi(X) = X^2 - tX + p$, with the trace $t$ of $E(\mathbb{F}_p)$. A point in $G_2$ has eigenvalue $\lambda_\psi$ mod $r = p \mod r = t - 1 \mod r$ under $\psi$. But for subgroup membership testing, one cannot assume that the point being tested has eigenvalue $t - 1$, one relies on the generic formula of $\lambda_\psi$ the root of $\chi_\psi(X)$ modulo the curve order which maps to $t - 1$ when reduced modulo $r$. One checks that the resultant of $\chi_\psi(X)$ and the polynomial formula $f(X) = c_9 + c_1X + \ldots + c_2X^t$ (see eq. (7)) has no common factor with the cofactor $c_2$ of $G_2$, that is, $\text{Res}_\psi(\text{Res}_\lambda(\chi_\psi(X), f(X)), c_2(x)) \neq 0$ [DLZZ23, Theorem 1]. The result is a rational number. The final step is to check that $c_2(x)$ has no root modulo each prime divisor of the numerator of that result.

We start from the data in Table 6 and check if Yu Dai et al. Theorem 3 is satisfied. Yu Dai et al. already solved the case for KSS16 with $x = 45$ mod 70, and KSS18, see Table 14.

**BLS curves.** All BLS curves have a trace $t = u + 1$, so that the curve order is $p + 1 - t = p - u$, and this number is a multiple of the prime order $r$ of $G_2$. The optimal ate pairing formula is based on the equation $u - p = 0$ mod $r$. For BLS12 and BLS24 curves, this equation directly gives a $G_2$ membership test: as soon as $r = \Phi_4(u)$ is prime (it corresponds to $\gcd(c_1, c_2) = 1$), $Q \in G_2 \iff [u]Q - \psi(Q) = \mathcal{O}$ that is, $Q \in G_2 \iff [u]Q = \psi(Q)$. This is not the case for BLS15, BLS21 nor BLS27 curves.

**BLS12, BLS24 curves.** Solutions appear already in previous works [Sco21, HGP22, DLZZ23]. Assuming that the order $r$ of $G_1, G_2$ is prime and $\gcd(c_1, c_2) = 1$ (the cofactors of $G_1$ and $G_2$ respectively are coprime), the test is $[u]Q = \psi(Q)$.

**BLS15, BLS21 curves.** This case is different as the $\gcd$ of $c_1$ and $c_2$ is $x^2 + x + 1$. For BLS15, note that $\Phi_5(p) = p^8 - p^7 + p^5 - p^4 + p^3 + p^2 - p + 1$. Then $p^7(x - p) = xp^7 - p^8 = 1 - p + p^3 - p^4 + (x - 1)p^7 \mod \Phi_5(p)$ and this formula satisfies Theorem 3. That is, the $G_2$ membership test is $Q - \psi(Q) + \psi^3(Q) = \psi^4(Q) + \psi^5 + [u - 1]\psi^7(Q) = \mathcal{O}$. When $7 \mid h_3$, this test still does not distinguish points of order 7. For BLS21, note that $\Phi_5(p) = p^{11} - p^{10} + p^9 - p^8 + p^6 - p^4 + p^3 - p + 1$. Then $p^{11}(x - p) = xp^{11} - p^{12} = 1 - p + p^3 - p^4 + p^6 - p^8 + p^9 + (x - 1)p^{11} \mod \Phi_5(p)$ and this formula succeeds in Theorem 3 when $c_2$ is odd. The $G_2$ membership test is $Q - \psi(Q) + \psi^3(Q) + \psi^4(Q) + \psi^5 + \psi^6 - \psi^8(Q) + \psi^{11}(Q) + [u - 1]\psi^{13}(Q) = \mathcal{O}$.

**BLS27 curves.** The $\gcd$ of the resultant and $c_2$ is $3$ with the formula $(u, -1)$. The formula $p^{17}(u - p) = 1 - p^3 + up^{17} \mod \Phi_5(p)$ does not solve the problem of the cofactor 3. With Smith technique we obtain that $(1 + p + p^2 + \ldots + p^6)(1 - u)/3 + p^7$ is a multiple of $r$ and avoids the cofactor 3 issue. The $G_2$ membership test is $[1 - u/3](S - \psi^3(S)) + \psi^3(Q) = \mathcal{O}$ where $S \leftarrow Q + \psi(Q) + \psi^2(Q) + \ldots + \psi^8(Q)$.

**FM15 curves.** The formula for optimal ate pairing is $u - p^4$ (Table 6). The $\gcd$ in $\mathbb{Q}[u]$ of $\text{Res}_X(X^2 - t(u)X + p(u), u - X^4)$ and $\#E'(\mathbb{F}_{p^2}) = h_2(u)r(u)$ is $r(u)$. Following [HGP22], we then compute

$$\text{Res}_u(\text{Res}_X(X^2 - t(u)X + p(u), u - X^4)/r(u), h_2(u))$$

and obtain a very large number whose small factors are $2^4 \cdot 1372471$. We observe that $\text{Res}_X(u)$ and $h_2(u)$ share the irreducible factor $(u^2 + u + 1)$ modulo 2 but do not share a common root. Actually, the resultant Res$_X(u)$ cannot be even. This ensures that the test based on the optimal ate pairing formula: whether $[u]Q - \psi^4(Q) = \mathcal{O}$ is a valid $G_2$ subgroup membership test.
KSS16 curves. An optimal pairing formula can be \( x/5p(x) + p^2(x) + 2x/5p^3(x) \) but the derived subgroup membership testing formula does not distinguish points of order 2r. With the same technique as [DLZZ23], we obtain the following formula for membership testing when \( x = 25 \mod 70 \) ([DLZZ23] addresses \( x = 45 \mod 70 \)).

\[
\frac{1}{70}(11x + 5, 11x + 5, -19x - 15, 3x - 5, -3x + 5, -13x - 25, 7x - 35, x + 45) \quad (8)
\]

We suggest this formula with larger scalars that works for any \( x = 25, 45 \mod 70 \):

\[
(u_5 + 1, u_5, u_5, -3u_5 - 2, -3u_5 - 2, -3u_5 - 2, -3u_5 - 2), \quad u_5 = (u - 5)/10 \quad (9)
\]

FM16 curves. The formula \( u - p \) gives the equation \([u]Q - \psi(Q) = \mathcal{O}\) and satisfies Theorem 3.

AFG16 curves. The formulas \(-1 + up^3\) and \(u + p^5\) give the equations \(-Q + [u] \psi^3(Q) = \mathcal{O}\) and \([u]Q + \psi^3(Q) = \mathcal{O}\) that both satisfy Theorem 3.

KSS18 curves. For KSS18 curves, the usual formulas of optimal ate pairing are \([19, 28] \mod 70 \) do not satisfy membership \([u]Q - \psi(Q) = \mathcal{O}\) and satisfies Theorem 3.

SG18 curves. The optimal pairing formula \( u + p^2 + up^3 \) gives the equation \([u]Q + \psi^2(Q) + [u] \psi^3(Q) = \mathcal{O}\) and satisfies Theorem 3.

FM18 curves. The formulas \( 1 + up^2 \) and \( u + p - p^4 \) give the equations \( Q + [u] \psi^2(Q) = \mathcal{O} \) and \([u]Q + \psi(Q) - \psi^4(Q) = \mathcal{O}\) and satisfy Theorem 3.

FST20, FST28 curves. The optimal pairing formula \( u - p \) does not give a valid equation, the resultant of \( u - X \) and \( X^2 - t(u)X + p(u) \) is \((u^2 + 1)r(u)\). We apply the same trick as for BLS: \( p^{2(4-k)-(u-p)} \mod \Phi_k(p) \) gives \( 1 - p^2 + p^4 - p^6 + up^7 \) for \( k = 20 \) and \( 1 - p^2 + p^4 - p^6 + p^8 - p^{10} + up^7 \) for \( k = 28. \) The equations \( Q - \psi^2(Q) + \psi^4(Q) - \psi^6(Q) + [u] \psi^7(Q) = \mathcal{O}\) for \( k = 20 \) and \( Q - \psi^2(Q) + \psi^4(Q) - \psi^6(Q) + \psi^8(Q) - \psi^{10}(Q) + [u] \psi^{11}(Q) = \mathcal{O}\) satisfy Theorem 3.

SG20 curves. The optimal pairing formulas \( u - up^5 - p^7 \) and \( u + p^2 + up^5 \) give the equations \([u]Q - [u] \psi^5(Q) - \psi^7(Q) = \mathcal{O} \) and \([u]Q + \psi^2(Q) + [u] \psi^5(Q) = \mathcal{O} \) and satisfy Theorem 3.

GG20b curves. The optimal pairing formulas \( u - p - 2p^6 \) and \( 2 + up^4 - p^5 \) do not satisfy Theorem 3 as the gcd is \((u^2 + 4u + 5)r(u)\). We set \( p^{2(k-5)}(2 + up^4 - p^7) \mod \Phi_{20}(p) = 1 - p^2 + p^4 - p^6 + 2p^3 + up^7 \) and obtain a formula that satisfies Theorem 3: \((\text{Id} - \psi^2 + \psi^4 - \psi^6 + [2] \psi^3 + [u] \psi^7)(Q) = \mathcal{O}\).

GG28 curves. The optimal pairing formulas \( u - p - 2p^6 \) and \( 2 + up^4 - p^7 \) do not satisfy Theorem 3 as the gcd is \((u^2 - 4u + 5)r(u)\). We set \( p^{2(k-7)}(2 + up^6 - p^7) \mod \Phi_{28}(p) = 1 - p^2 + p^4 - p^6 + p^8 - p^{10} + 2p^3 + up^4 \) and obtain a formula that satisfies Theorem 3: \((\text{Id} - \psi^2 + \psi^4 - \psi^6 + \psi^8 - \psi^{10} + [2] \psi^3 + [u] \psi^{11})(Q) = \mathcal{O}\).
5 Experimental results

We implemented the most promising curves from Section 3.4 within the RELIC cryptographic toolkit [AGM+] due to its strong support to pairing-based cryptography. RELIC already implemented several pairing-friendly curves (such as BLS) with state-of-the-art performance, so extending the library to include new curve families is natural and favors a fair comparison between candidates. The library is implemented in the C programming language, with handwritten ASM acceleration for finite field arithmetic in multiple sizes of the prime moduli. For each of the new or previously supported curve families, we included group operations and pairing computation, by either speeding-up existing routines or including entirely new ones. The resulting source code can be found at our anonymized repository 4, with prebuilt binaries to facilitate reproduction of the timings. Our SageMath and MAGMA scripts can also be found at the same repository.

Benchmarking measurements were taken by computing the average latency of running the operation for $10^4$ consecutive executions on an Intel Kaby Lake Core i7-7700 CPU running at 3.60GHz. The main compiler used was GCC version 13.2.1, with optimization flags `-O3 -funroll-loops -march=native -mtune=native`; but we also verified that clang would exhibit similar performance. Following best practices5, the TurboBoost and HyperThreading features were disabled in the benchmarking machine for higher stability. Table 16 shows the cycle counts observed for executing scalar multiplications or exponentiation in pairing groups using a $w$-NAF algorithm with $w = 4$ and Jacobian coordinates in the Weierstrass model. The constant-time version for the source groups was implemented with almost-complete exception-free homogeneous projective coordinates [RCB16] using a regular $w$-NAF algorithm with $w = 5$ to compensate the performance loss. The table also shows timings for hashing to source groups, testing group elements for subgroup membership and pairing computation, split in the Miller loop and Final Exponentiation. From the table, it is clear that BLS24-509 has superior performance for all benchmarked operations, with speedups ranging from to 3.2% to 56.1%. We also highlight the fastest operations in curves with embedding degree 16 or 18, and show that AFG16 is a competitive candidate in comparison to other curves with the same embedding degree.

When checking for consistency between the timings for pairing computation and the normalized operation counts from Table 11, we can conclude that most figures are consistent, showing strong alignment between the implementation effort and the performance estimates. There are two exceptions, curve BLS12-1150 and FM16-765. The former deviates from the other implementations in the sense that it ends up more efficient than estimated, because it uses the low-level interface of the GMP library with Karatsuba splitting, leading to a slightly different cost model than the Schoolbook method assumed during normalization. For FM16-765, the observed performance is worse than AFG16 due to a less efficient choice of towering, that particular affects the timings for the final exponentiation. This is why the pairing latency looks closer to KSS16 than AFG16. Furthermore, we do not claim that all of our timings are the best possible for all parameters. Our emphasis was to fairly optimize all parameters with similar effort, such that meaningful comparisons could be made. For this reason, we decided to not implement techniques that could favor one parameter over the others [Lon23, BCN14] due to availability of more efficient code, although we leave to implement them as future work, as long as we feel confident that the same optimization level can be achieved across all parameters.

4https://github.com/cic-pairing192/suppl-material
5https://bench.cr.yp.to/supercop.html
### Table 16: Latency in $10^3$ clock cycles for performing scalar multiplication or exponentiation in $G_1, G_2$ and $G_T$, possibly with constant-time (CT) countermeasures.

<table>
<thead>
<tr>
<th>Curve Size of $p$</th>
<th>BLS12 1150</th>
<th>FM16 765</th>
<th>KSS16 766</th>
<th>AFG16 766</th>
<th>FM18 768</th>
<th>SG18 638</th>
<th>KSS18 638</th>
<th>BLS24 509</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[e]P$ in $G_1$</td>
<td>6803</td>
<td>2216</td>
<td>3282</td>
<td>2157</td>
<td>2079</td>
<td>1306</td>
<td>1718</td>
<td>1066</td>
</tr>
<tr>
<td>$[e]Q$ in $G_2$</td>
<td>15776</td>
<td>12193</td>
<td>18433</td>
<td>11227</td>
<td>13524</td>
<td>9622</td>
<td>11416</td>
<td>5199</td>
</tr>
<tr>
<td>$g'$ in $G_T$</td>
<td>23639</td>
<td>8867</td>
<td>13375</td>
<td>8451</td>
<td>12372</td>
<td>9437</td>
<td>11748</td>
<td>7129</td>
</tr>
<tr>
<td>CT $[e]P$ in $G_1$</td>
<td>9168</td>
<td>2805</td>
<td>4288</td>
<td>2802</td>
<td>2764</td>
<td>1956</td>
<td>2394</td>
<td>3275</td>
</tr>
<tr>
<td>CT $[e]Q$ in $G_2$</td>
<td>20332</td>
<td>19068</td>
<td>25795</td>
<td>17952</td>
<td>13524</td>
<td>9622</td>
<td>11416</td>
<td>7548</td>
</tr>
<tr>
<td>CT $g'$ in $G_T$</td>
<td>27754</td>
<td>12359</td>
<td>16485</td>
<td>11426</td>
<td>15760</td>
<td>12938</td>
<td>15085</td>
<td>11058</td>
</tr>
<tr>
<td>Hash to $G_1$</td>
<td>6280</td>
<td>2845</td>
<td>1759</td>
<td>2495</td>
<td>3970</td>
<td>1490</td>
<td>1115</td>
<td>489</td>
</tr>
<tr>
<td>Hash to $G_2$</td>
<td>18662</td>
<td>27573</td>
<td>22907</td>
<td>22952</td>
<td>27350</td>
<td>15130</td>
<td>8894</td>
<td>5788</td>
</tr>
<tr>
<td>Testing in $G_1$</td>
<td>4547</td>
<td>1652</td>
<td>3060</td>
<td>1639</td>
<td>1494</td>
<td>1018</td>
<td>1808</td>
<td>797</td>
</tr>
<tr>
<td>Testing in $G_2$</td>
<td>4935</td>
<td>3029</td>
<td>6547</td>
<td>2536</td>
<td>2057</td>
<td>1634</td>
<td>1927</td>
<td>1061</td>
</tr>
<tr>
<td>Testing in $G_T$</td>
<td>5351</td>
<td>2579</td>
<td>5895</td>
<td>2125</td>
<td>4225</td>
<td>9878</td>
<td>2359</td>
<td>1294</td>
</tr>
<tr>
<td>Miller Loop</td>
<td>28484</td>
<td>8188</td>
<td>11871</td>
<td>6950</td>
<td>9749</td>
<td>7335</td>
<td>9327</td>
<td>5429</td>
</tr>
<tr>
<td>Final Exp.</td>
<td>29317</td>
<td>31649</td>
<td>28533</td>
<td>25666</td>
<td>27176</td>
<td>13628</td>
<td>15607</td>
<td>9670</td>
</tr>
<tr>
<td>Pairing</td>
<td>57802</td>
<td>39882</td>
<td>40404</td>
<td>32617</td>
<td>36925</td>
<td>20763</td>
<td>24971</td>
<td>15100</td>
</tr>
</tbody>
</table>

### 6 Conclusion

We performed an extensive comparison of several candidate families of pairing-friendly curves at the 192-bit level, both from theoretical and practical perspectives. Our recommended choices of pairing-friendly curves are the following. For a prime-order curve, choose a BN curve of $\approx 1152$ bits. For the fastest pairing, take a BLS24 curve of $\approx 512$ bits. For smallest $G_1$, a BLS27 curve can have $p$ of $\approx 427$ bits. While we focus exclusively on the asymmetric pairings that offer the best performance, curves with embedding degree $k = 1$ and supersingular curves with $k = 2$ can be used to instantiate a symmetric pairing, at high performance penalty due to the very large base fields.

### References


[AHST23] Diego F. Aranha, Benjamin Salling Hvass, Bas Spitters, and Mehdi Tibouchi. Faster constant-time evaluation of the Kronecker symbol with application...


[CDS20] Rémi Clarisse, Sylvain Duquesne, and Olivier Sanders. Curves with fast computations in the first pairing group. In Stephan Krenn, Haya Shulman,


A short-list of pairing-friendly curves at the 192-bit security level


A short-list of pairing-friendly curves at the 192-bit security level


Diego F. Aranha, Georgios Fotiadis, Aurore Guillevic


A short-list of pairing-friendly curves at the 192-bit security level


A Discarded curves

In Table 18 we give the seeds for evaluating the families of pairing-friendly elliptic curves with embedding degree \( k = 22 \). We focus on Freeman–Scott–Teske (FST) families obtained by construction 6.3 (FST6.3) and construction 6.6 (FST6.6) in [FST10]. In addition we consider the family of curves of Gasnier–Guillevic [GG23], GG22 with CM-discriminant \( D = 7 \). Further, in Table 19 we present the formula of the optimal ate pairing for the three families of pairing-friendly curves with embedding degree 22. The formula of the easy part of the final exponentiation is given in Table 17.

<table>
<thead>
<tr>
<th>( k )</th>
<th>curve</th>
<th>seed</th>
<th>( (u, b) )</th>
<th>( p \mod r )</th>
<th>( \log p )</th>
<th>( \log r )</th>
<th>( p )</th>
<th>( \log p^2 )</th>
<th>secu/ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>FST6.3</td>
<td>( 2^{10} + 2^{13} + 2^{15} + 1 )</td>
<td>( (1,0) )</td>
<td>1</td>
<td>534</td>
<td>348</td>
<td>1.30</td>
<td>11965</td>
<td>194</td>
</tr>
<tr>
<td></td>
<td>GG22, ( D = 7 )</td>
<td>( -2^{19} + 2^{13} + 2^{11} - 2^{10} - 2^8 - 2^6 + 1 )</td>
<td>( (\frac{117}{2}, \frac{119}{2}) )</td>
<td>3</td>
<td>457</td>
<td>383</td>
<td>1.2</td>
<td>10052</td>
<td>220</td>
</tr>
</tbody>
</table>

Table 17: Exponents for the final exponentiation, easy and hard parts.

\[
\begin{align*}
final \ exp. \ easy \ part & = (p^{11} - 1)(p + 1) \\
final \ exp. \ hard \ part & = (p^{19} - p^7 + p^8 - p^6 - p^5 + p^4 - p^4 + p^2 - p + 1)/r
\end{align*}
\]

Table 18: Additional families of pairing-friendly elliptic curves and seeds at 192-bit security.

Table 19: Optimal ate Miller loop formulas. The Miller functions \( f_{u,Q} \) and lines \( \ell_{Q,R} \) are evaluated at the point \( P \in \mathbb{G}_1 \).

<table>
<thead>
<tr>
<th>( k )</th>
<th>curve</th>
<th>Equation (1)</th>
<th>Optimal ate formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>FST6.3</td>
<td>( u^r - p \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) )</td>
</tr>
<tr>
<td></td>
<td>FST6.6</td>
<td>( u^2 - up^3 + p^3 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) \cdot f_{u,Q}(P)^{-p} \cdot \ell_{u[2Q,-u][u,Q]}(P) )</td>
</tr>
<tr>
<td></td>
<td>GG22</td>
<td>( u^2 - up + 2p^2 \equiv 0 \mod r )</td>
<td>( f_{u,Q}(P) \cdot f_{u,Q}(P)^{-p} \cdot \ell_{u[2Q,-u][u,Q]}(P) \cdot \ell_{Q,Q}(P)^{3p^2} )</td>
</tr>
</tbody>
</table>

Table 20: Optimal ate pairing and final exponentiation cost estimates in terms of finite field multiplications. The name of each curve is derived from the name of the corresponding family in Table 5, plus the size of the prime \( p \).

<table>
<thead>
<tr>
<th>curve</th>
<th>( p ) bits</th>
<th>( r ) bits</th>
<th>Miller loop optimal ate</th>
<th>final exp.</th>
<th>pairing</th>
</tr>
</thead>
<tbody>
<tr>
<td>FST6.3-544, ( k = 22 )</td>
<td>544</td>
<td>420</td>
<td>39.797m</td>
<td>789m</td>
<td>65.604m</td>
</tr>
<tr>
<td>FST6.6-534, ( k = 22 )</td>
<td>534</td>
<td>383</td>
<td>33.955m</td>
<td>789m</td>
<td>64.200m</td>
</tr>
<tr>
<td>GG22-457, ( D = 7 )</td>
<td>457</td>
<td>383</td>
<td>41.154m</td>
<td>789m</td>
<td>72.352m</td>
</tr>
</tbody>
</table>

Table A shows the number of \( \mathbb{F}_p \)-multiplications required for the Miller loop and the final exponentiation for the three types of curves with \( k = 22 \), FST6.3, FST6.6 and GG22. We discarded these curves from our study because they do not seem to be competitive candidates compared to the ones presented in Section 3. We note that such curves admit quadratic twists and hence the point addition and doubling operations in the Miller loop are executed over \( \mathbb{F}_{p^4} \).

For group operations using GLV, the curve of \( j \)-invariant -3375 (discriminant \( D = 7 \)) has an endomorphism \( \phi \) of the form \( (x, y) \mapsto (\phi_x(x), y\phi_y(x)) \) with some \( s = \sqrt{-7} \in \mathbb{F}_p \), of characteristic polynomial \( \chi_\phi(X) = X^2 + X + 2 \).
FST 6.3 $k = 22$. Here is a formula for the hard part of the final exponentiation on input $m$:

\[
\begin{align*}
\bar{m} &\leftarrow m^{p_{11}}; \\
a &\leftarrow m^{u^2} \cdot \bar{m}; \quad f \leftarrow f^p \cdot a; \quad a \leftarrow a^{u^2} \cdot m; \quad f \leftarrow f^p \cdot a; \\
a &\leftarrow m^{u^2} \cdot \bar{m}; \quad f \leftarrow f^p \cdot a; \quad a \leftarrow a^{u^2} \cdot m; \quad f \leftarrow f^p \cdot a; \\
a &\leftarrow m^{u^2} \cdot \bar{m}; \quad f \leftarrow f^p \cdot a; \quad a \leftarrow a^{u^2} \cdot m; \quad f \leftarrow f^p \cdot a; \\
a &\leftarrow m^{u^2} \cdot \bar{m}; \quad f \leftarrow f^p \cdot a; \\
f &\leftarrow f^{u^2} \cdot f^{p_{11}}; \quad f \leftarrow f^{u^4} \cdot f^{p_{11}}; \quad f \leftarrow f \cdot m^4
\end{align*}
\]

which costs $24 \exp(u) + 2s_{22} + 21m_{22} + 9f_{22}$.

FST 6.6 $k = 22$. Here is a formula for the hard part of the final exponentiation:

\[
(u^3 - 1)^2((u^{10}(u + q^4) + uq^7 - 1)(u^3 + q)(u^6 + q^2) - u^{10}q^7) + 3(u + q^4)
\]

which costs $26 \exp(u) + 11m_{22} + s_{22} + 6f_{22}$.

GG22D7. We obtained a final exp hard formula which costs $22 \exp(u) + 20f_{22} + 41m_{22} + 50s_{22}$.

B Formulas and parameters for group operations

In Table 21 we give polynomial description for families of curves considered in this paper. Specifically, this table lists the CM-discriminant for each type of curves, the polynomial $r(x)$ dividing the order of the curve, the polynomial representation of the cofactor $c_1(x)$, such that $E(F_{p(x)}) = c_1(x)r(x)$, as well as the congruences that the seeds must satisfy in order for the polynomial family to produce integer values.

C Curves at the 256-bit security level

In Table 22 we give known instantiations in the literature for existing families of pairing-friendly elliptic curves aiming at 256-bit security level.
Table 21: Curve parameters, $k$ is the embedding degree, $r$ is the prime order of $G_1$, $c_1$ is the cofactor so that the curve over $\mathbb{F}_p$ has order $r \cdot c_1$, and $x$ is the seed.

<table>
<thead>
<tr>
<th>$k$</th>
<th>$D$</th>
<th>Curve</th>
<th>$r(x)$</th>
<th>$c_1(x)$</th>
<th>$x$ s.t. $r \cdot c_1$ are integers</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>3</td>
<td>BLS12</td>
<td>$x^2 - x^2 + 1$</td>
<td>$(x - 1)^2/3$</td>
<td>1 mod 3</td>
</tr>
<tr>
<td>15</td>
<td>3</td>
<td>BLS15</td>
<td>$x^2 - x^2 + x^2 - x^2 + x^2 - x + 1$</td>
<td>$(x - 1)^2/3 (x^3 + x + 1)$</td>
<td>1, 1 mod 15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FM18</td>
<td>$x^2 - x^2 + x^2 - x^2 + x^2 - x + 1$</td>
<td>$x^3 - 2x^2 - x^2 - 2x^2 + 3/3$</td>
<td>0, 3 mod 15</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>KSS16</td>
<td>$(x^5 + 4x^4 + 625)/61250$</td>
<td>$125/2 (x^5 + 2x + 5)$, 1250</td>
<td>25, 45 mod 70</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FM10</td>
<td>$\Phi_{16} = x + 1$</td>
<td>$x^2 / 4 (x^6 + 1)$</td>
<td>0 mod 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AFG16</td>
<td>$x^2 (x^3 + 1)^{1/4}$</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>KSS18</td>
<td>$(x^8 + 37x^4 + 333)/33$</td>
<td>$49/3 (x^8 + 5x + 7), 333 \mid c_1$</td>
<td>14 mod 21</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SG18</td>
<td>$2x^6 + 9x^5 + 1$</td>
<td>$(3x^2 - 1)^{2}$</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FM18</td>
<td>$\Phi_3 = x^3 - x^2 + 1$</td>
<td>$(3x^6 + x^2 - 2x + 1)/3$</td>
<td>1 mod 3</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>FM20</td>
<td>$\Phi_{20} = x^8 - x^6 + x^2 - x^2 + 1$</td>
<td>$(x - 1)^2 (x^2 + 1)/4$</td>
<td>1 mod 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SG20</td>
<td>$16x^8 + 16x^6 + 8x^2$</td>
<td>$-4x^4 + 2x^2 + 2x + 1$</td>
<td>any</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GG20a</td>
<td>$(x^3 + x^2 + 111x^6 + 21x^2 + 41x^4 + 120x^3 + 275x^2 + 500x + 625) / (51^2 - 411)$</td>
<td>$125x^2 - 4x + 5$</td>
<td>1715, 1815 mod 2050</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GG20b</td>
<td>$(x^3 + x^2 + 111x^6 + 21x^2 + 41x^4 + 120x^3 + 275x^2 + 500x + 625) / (51^2 - 411)$</td>
<td>$x^2 - 4x + 5$</td>
<td>1465, 1565 mod 2050</td>
</tr>
<tr>
<td>21</td>
<td>3</td>
<td>BLS21</td>
<td>$\Phi_{21} = x^2 - x^1 + x^2 - x^5 + x^6 - x^4 - x^3 - x - 1$</td>
<td>$(x - 1)^2/3</td>
<td>c_1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BLS24</td>
<td>$\Phi_{24} = x^2 - x^1 + 1$</td>
<td>$(x - 1)^2/3</td>
<td>3 \mid c_1$</td>
</tr>
<tr>
<td>27</td>
<td>3</td>
<td>BLS27</td>
<td>$\Phi_{27}/3 = (x^9 + x^1 + 1)/3$</td>
<td>$(x - 1)^2/3</td>
<td>c_1$</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
<td>FST28</td>
<td>$\Phi_{28} = x^4 - x^3 + x^2 + 1$</td>
<td>$(x - 1)^2/4 (x^2 + 1), 2</td>
<td>c_1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GG28</td>
<td>$(x^4 + 14x^5 + 11x^6 + 21x^7 + 41x^8 + 124x^9 + 200x^{10} + 1025x^{11} + 3000x^{12} + 6875x^{13} + 12500x + 15625) / 29$</td>
<td>$(x^2 - 4x + 5), (x^2 - 2x + 5) / 580$</td>
<td>309, 449, 1759, 1888 mod 2030</td>
</tr>
</tbody>
</table>

Table 22: Curves at 256-bit security level from [BD19], [KIK+17, Table 6], and [BMDFAF19]

<table>
<thead>
<tr>
<th>curve</th>
<th>seed</th>
<th>$r$ (bits)</th>
<th>$p$ (bits)</th>
<th>$p^k$ (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSS18</td>
<td>$2^{186} - 2^{78} - 2^{22} + 2$</td>
<td>1108</td>
<td>1484</td>
<td>26712</td>
</tr>
<tr>
<td>BLS24</td>
<td>$-2^{101} + 2^{101} + 2^{96} + 2^{90}$</td>
<td>827</td>
<td>1032</td>
<td>24768</td>
</tr>
<tr>
<td>BLS24</td>
<td>$2^{109} - 2^{77} + 2^{65} - 1$</td>
<td>872</td>
<td>1089</td>
<td>26122</td>
</tr>
<tr>
<td>KSS32</td>
<td>$2^{32} - 2^{10} + 2^{5} + 2^{4} - 2^{2} - 2^{1} - 2^{1} - 2^{2} + 1$</td>
<td>738</td>
<td>861</td>
<td>27536</td>
</tr>
<tr>
<td>KSS36</td>
<td>$-2^{98} + 2^{45} + 2^{20} + 2^{16} + 2^{3}$</td>
<td>669</td>
<td>798</td>
<td>28699</td>
</tr>
<tr>
<td>BLS42</td>
<td>$2^{13} - 2^{8} - 2^{4} - 2^{2}$</td>
<td>516</td>
<td>687</td>
<td>28830</td>
</tr>
<tr>
<td>BLS48</td>
<td>$-2^{13} - 2^{9} - 2^{6} - 2^{4} - 2^{3}$</td>
<td>518</td>
<td>581</td>
<td>27851</td>
</tr>
<tr>
<td>BLS48</td>
<td>$2^{24} - 2^{18} - 2^{10} - 2^{4}$</td>
<td>512</td>
<td>575</td>
<td>27600</td>
</tr>
<tr>
<td>SG54</td>
<td>$2^{24} + 2^{26} + 2^{22} + 2^{14} + 2^{16} + 2^{2}$</td>
<td>512</td>
<td>569</td>
<td>30726</td>
</tr>
</tbody>
</table>