IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 3, 27 pages.

https://doi.org/10.62056/ayojbhey6b

Check for updates

Finding Practical Parameters for Isogeny-based Cryptography

Maria Corte-Real Santos[1], Jonathan Komada Eriksen[2], Michael Meyer[3] and Francisco Rodríguez-Henríquez[4]

[1] University College London, London, UK
[2] Norwegian University of Science and Technology, Trondheim, Norway
[3] University of Regensburg, Regensburg, Germany
[4] Cryptography Research Center, Technology Innovation Institute, Abu Dhabi, United Arab Emirates

Abstract. Isogeny-based schemes often come with special requirements on the field of definition of the involved elliptic curves. For instance, the efficiency of SQIsign, a promising candidate in the NIST signature standardisation process, requires a large power of two and a large smooth integer T to divide $p^2 - 1$ for its prime parameter p. We present two new methods that combine previous techniques for finding suitable primes: sieve-and-boost and XGCD-and-boost. We use these methods to find primes for the NIST submission of SQIsign. Furthermore, we show that our methods are flexible and can be adapted to find suitable parameters for other isogeny-based schemes such as AprèsSQI or POKE. For all three schemes, the parameters we present offer the best performance among all parameters proposed in the literature.

Keywords: Post-quantum cryptography · isogenies · parameter search · SQIsign · AprèsSQI · POKE

1 Introduction

Research has shown that large-scale quantum computers can break most widely deployed cryptographic schemes today, such as those based on the discrete logarithm problem (e.g., ECC) and the integer factorization problem (e.g., RSA). With increased investment in building large-scale quantum computers, there has been a significant focus on *post-quantum cryptography* in recent years. The aim of this area of cryptography is to construct cryptographic schemes whose security is based on alternative mathematical problems that are conjectured to be hard against adversaries with access to classical and quantum computers. In 2016, NIST began an effort to standardise post-quantum key encapsulation mechanisms and digital signature schemes.

This work focuses on a specific area of post-quantum cryptography known as isogeny-based cryptography. Here, we rely on the difficulty of finding an *isogeny* (a specific type of map) between two elliptic curves defined over a finite field. While isogeny-based cryptography offers promising schemes with small private/public keys and signatures, they have proved to be considerably slower compared to other candidates. Notably, SQIsign, the only isogeny-based signature scheme in Round 1 of NIST's alternate call for

Author list in alphabetical order; see https://www.ams.org/profession/leaders/CultureStatement04.pdf. This work has been supported by UK EPSRC grant EP/S022503/1 and by the German Federal Ministry of Education and Research (BMBF) under the project 6G-RIC (ID 16KISK033).

E-mail: maria.santos.20@ucl.ac.uk (Maria Corte-Real Santos), jonathan.k.eriksen@ntnu.no (Jonathan Komada Eriksen), michael@random-oracles.org (Michael Meyer), francisco.rodriguez@tii.ae (Francisco Rodríguez-Henríquez)

Received: 2024-07-09 Accepted: 2024-09-02

signatures [NIS23], boasts the smallest combined signature and public key sizes among all submissions. However, signing with SQIsign is orders of magnitude slower than lattice-based alternatives, while verification remains relatively fast in the range of milliseconds.

Increasingly many isogeny-based schemes have come with special requirements on parameters to ensure that the resulting schemes are efficient, e.g., B-SIDH [Cos20], SQIsign [DKL+20, DLLW23], or POKE [Bas24]. These schemes work with supersingular elliptic curves defined over a finite field $\mathbb{F}_{p^2}$ with prime p. The efficiency of computing isogenies relies on the choice of p. To make a *good* choice, we follow two rules of thumb:

1. For some $N \in \mathbb{N}$ we have $N \mid (p-1)(p+1)$: this ensures that all points of order N, resp. the N torsion, is defined over $\mathbb{F}_{p^2}$, thus leading to the efficient computation of isogenies of degree N.

2. Small prime divisors $\ell \mid N$: an isogeny of composite degree N with prime factorization $\prod_i \ell_i^{e_i}$ can more efficiently be computed as a composition of e_i isogenies of degree ℓ_i, i.e., a total of $\sum_i e_i$ prime-degree isogenies. Computing an ℓ-isogeny of prime degree has complexity $O(\ell)$ using Vélu's formulas [Vél71], and asymptotic complexity $\widetilde{O}(\sqrt{\ell})$ using $\sqrt{\text{élu}}$ formulas [BDLS20]. Thus, only isogenies of small degrees ℓ can be computed efficiently.

Hence, we can efficiently compute isogenies of degree N if $N \mid p^2 - 1$ and all prime factors of N are below a certain bound B, i.e., N is *B-smooth*.

The exact requirements on the size of p and factors of $p^2 - 1$ depend on the respective scheme. While SIDH exclusively used isogeny degrees N with $N \mid p + 1$, B-SIDH was the first scheme to explicitly use quadratic twists of the involved elliptic curves, hence allowing to also use factors of $p - 1$ as isogeny degrees. In particular, for NIST-I security level, ideal B-SIDH primes were required to be of size $p \approx 2^{256}$ and to have a B-smooth value $p^2 - 1$ for B as small as possible. However, finding such primes p is a difficult task that spurred a new line of research [Cos20, CMN21, BCC+23]. The best known smoothness bound for the mentioned NIST-I requirements is $B \approx 2^{15}$ [CMN21].

After SIDH – and simultaneously also B-SIDH – was broken [CD23, MMP+23, Rob23], variants of this problem have remained of high interest. We highlight in particular the signature scheme SQIsign [DKL+20, DLLW23]. For efficiency, SQIsign requires $2^f T \mid p^2 - 1$, where T is an odd smooth integer satisfying $T > p^{5/4}$, and f as large as possible. The signer then computes a number (depending on f) of T-isogenies, while the verifier computes a chain of 2^f-isogenies. As a result, signing times benefit from both larger values of f and better smoothness of T. Verification times mainly benefit from larger f, as detailed in [CEMR24, Fig. 2].

Prior to this work, De Feo, Leroux, Longa and Wesolowski [DLLW23] introduced the prime p_{3923}, which was used for NIST-I security. It has

$$\begin{aligned} f &= 66, \\ B &= 3923, \\ T &= 3^{65} \cdot 5^2 7 \cdot \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 29^2 \cdot 37^2 \cdot 43 \cdot 47 \cdot 79 \cdot 157 \cdot 197 \cdot 239 \cdot 263 \cdot 271 \cdot 281 \\ &\quad \cdot 283 \cdot 307 \cdot 461 \cdot 521 \cdot 563 \cdot 599 \cdot 607 \cdot 619 \cdot 743 \cdot 827 \cdot 941 \cdot 2357 \cdot 3923. \end{aligned}$$

Larger parameters were only considered in [BCC+23], but without large enough powers of 2 dividing $p \pm 1$.

In this work, we present two new methods to find SQIsign-friendly primes. While both techniques are simple, they seem to produce the best primes for SQIsign. These new methods combine sieving [CMN21] or the extended Euclidean algorithm techniques [Cos20, DKL+20] with the boosting method introduced in [BCC+23]. Using these techniques, we find new primes targeting NIST-I, -III and -V security. We highlight in particular a new NIST-I

prime p_{1973} with

$$\begin{aligned} f &= 75, \\ B &= 1973, \\ T &= 3^{36} \cdot 7^4 \cdot 11 \cdot 13 \cdot 23^2 \cdot 37 \cdot 59^2 \cdot 89 \cdot 97 \cdot 101^2 \cdot 107 \cdot 109^2 \cdot 131 \cdot 137 \cdot 197^2 \cdot 223 \cdot 239 \\ &\quad \cdot 383 \cdot 389 \cdot 491^2 \cdot 499 \cdot 607 \cdot 743^2 \cdot 1033 \cdot 1049 \cdot 1193 \cdot 1913^2 \cdot 1973. \end{aligned}$$

Comparing with the previous state-of-the-art prime, we achieve both a lower smoothness bound B and a *larger* power-of-2 f, leading to faster signing and verification.

Though we have thus far only discussed finding parameters for SQIsign, small adaptations of our techniques also apply to finding primes for AprèsSQI [CEMR24], a variant of SQIsign focussed on fast verification, and POKE, a new PKE scheme introduced by Basso [Bas24], showing the wide applicability of our methods. In particular, we give several primes for AprèsSQI which make both signing and verification faster than in the original SQIsign scheme.

Our contributions. In this article, we present two new techniques for finding primes p satisfying the requirements imposed by various isogeny-based schemes, as detailed above. The first of these methods is sieve-and-boost (see Section 3), where we sieve for B-smooth numbers r in a suitable range, and boost this to give cryptographically sized primes by evaluating polynomials of the form $p_n(x) = 2x^n - 1$ at $x = 2^{f_1}3^{f_2}r$. The second method is XGCD-and-boost (see Section 4). Here, rather than sieving, we construct pairs $(r, r \pm 1)$ of twin smooth numbers using the XGCD method and use them as inputs to $p_n(x)$.

We analyse the probability of obtaining suitable primes p with both our methods to determine the degree n and smoothness bound B for which to run our searches. Using this, we tackle three schemes in particular, where according to our cost metric, our parameters are the best in the literature:

- SQIsign. We found suitable primes for NIST security levels I, III, and V (cf. Table 5) that permit faster signing verification. These primes outperform those proposed in [DLLW23, BCC+23] and have been adopted in [CCD+23].
- AprèsSQI. We found suitable primes (cf. Table 6) that strictly outperform the primes selected in [CCD+23] for NIST-I security level, in both signature generation and verification.
- POKE. We present the first suitable primes of minimal size that permit an efficient computation of the PKE scheme constructed with the POKE framework [Bas24, §4], hence reducing key and ciphertext sizes compared to the original proposal.

Remark 1. Very recent work introduced 2-dimensional variants of SQIsign [BDD+24, DF24, NO24]. These variants improve upon the 4- and 8-dimensional SQIsignHD schemes [DLRW24] and feature a very promising performance profile. As they use an isogeny representation that differs from the discussion above, they have different, relatively mild conditions on the prime parameters. Nevertheless, due to the lack of optimised implementations of AprèsSQI, it is not clear if these schemes can offer a faster verification than SQIsign or AprèsSQI. Although our work does not apply to these new schemes, it is thus still important to find optimal SQIsign and AprèsSQI parameters. Furthermore, the applicability to POKE shows that our search techniques are of high interest in isogeny-based cryptography even beyond variants of SQIsign. Indeed, research on these prime finding methods first started within the context of SIDH, before later being applied to SQIsign. Our techniques are flexible, so protocol designers can simply adapt our search implementations to their needs.

Remark 2. A short description of the methods used in this work and the related search for SQIsign primes is included (but not formally published) in the NIST specification of SQIsign [CCD+23]. The fact that the primes found in our work are the parameters of choice of the SQIsign NIST submission emphasises the relevance of our work.

Roadmap. We begin by discussing the preliminaries in Section 2, including a description of the precise parameter requirements for the schemes we target in this article. Next, we describe our two methods: sieve-and-boost in Section 3 and XGCD-and-boost in Section 4. In Section 5, we analyse the smoothness probabilities related to each method. Finally, in Section 6, we detail the searches we perform and the best parameters we obtain for each scheme.

Acknowledgements. We thank Luca De Feo, Lorenz Panny, and the SQIsign NIST submission team for discussions, comments, and suggestions on our SQIsign prime searches. We thank Andrea Basso for pointing out the problem of finding POKE primes to us. We thank the authors of [AAA+24a] for sharing their preliminary results, which enabled us to define our cost metric using their prime scoring script for computing isogenies in field extensions. We thank the anonymous reviewers for their constructive feedback.

2 Preliminaries

We start by covering the necessary preliminaries. Through the rest of this paper, let p denote a prime $p > 3$.

Before describing the precise parameter requirements for SQIsign, AprèsSQI, and POKE, we briefly recall required definitions on smooth integers. A discussion on relevant facts about smoothness probabilities required for analysing our search methods is provided in Section 5.

Definition 1. A positive integer r is called *B-smooth* if all of its prime divisors q_i satisfy $q_i \leq B$. A pair $(r, r+1)$ of B-smooth integers is called *B-smooth twins*. If all prime divisors q_i of a positive integer r satisfy $q_i > B$, r is said to be *B-rough*.

2.1 SQIsign prime requirements

The most expensive computational task in SQIsign is the computation of isogenies. We work with supersingular elliptic curves E defined over $\mathbb{F}_{p^2}$, whose group of $\mathbb{F}_{p^2}$-rational points has cardinality $(p+1)^2$. This ensures that for any point $P \in E$ of order $N \mid p+1$, we have that $P \in E(\mathbb{F}_{p^2})$. Similarly, for any point $P' \in E$ of order $N' \mid p-1$, we have that $P' \in E^t(\mathbb{F}_{p^2})$, where E^t denotes an arbitrary quadratic twist of E over $\mathbb{F}_{p^2}$.

Using Vélu's formulas [Vél71] or $\sqrt{\text{élu}}$ [BDLS20], together with x-only arithmetic, we can thus compute isogenies of any degree $\ell \mid (p+1)$ or $\ell \mid (p-1)$ in $O(\ell)$ or $\widetilde{O}(\sqrt{\ell})$, respectively, without moving to field extensions of $\mathbb{F}_{p^2}$. Hence, the efficiency of SQIsign hinges on choosing a prime p of suitable size, such that $\text{lcm}(p-1, p+1) = (p^2-1)/2$ contains sufficiently many smooth prime factors.

SQIsign-friendly primes are required to be of size $\log_2(p) \approx 2\lambda$, where λ is the prescribed security parameter (i.e., $\lambda \in \{128, 192, 256\}$ for NIST-I, -III, resp. -V security). We note that due to its performance profile, SQIsign naturally prioritises fast verification, while maintaining reasonable signing performance.

The verifier recomputes a response isogeny of degree $\approx p^{15/4}$ and a challenge isogeny of degree $\approx 2^\lambda$. For efficiency, the degrees of these isogenies are chosen to be as smooth as possible. In particular, we fix the response degree to be 2^e, and the challenge degree to

$2^f 3^{f'}$. The response isogeny of degree $2^e \approx p^{15/4}$ is computed as a composition of $\lceil e/f \rceil$ isogenies of degree 2^f, where 2^f is the maximal power of two dividing $p+1$.

In the signing procedure of SQIsign, the bottleneck is to compute $2\lceil e/f \rceil$ odd degree T-isogenies, where T is a smooth positive integer of size approximately $p^{5/4}$. Since T has to be coprime to the degree of the response isogeny, this yields the requirement $2^f \cdot T \mid (p^2-1)/2$, where T is odd and $3^{f'} \mid T$.

In summary, we get the following requirements:

1. Primes p of roughly 256, 384, resp. 512 bits for NIST-I, -III, resp. -V. Note that slightly smaller sizes by a few bits allow for more efficient field arithmetic implementations.

2. Smooth $T \mid (p^2-1)/2$: For the efficient computation of T-isogenies during signing, we require the odd factor $T \approx p^{5/4}$ to be B-smooth for B as small as possible. Since values of T close to $p^{5/4}$ may lead to frequent failures, we require $T > p^{1.27}$ for NIST-I and -III, and $T > p^{1.26}$ for NIST-V security to give a failure margin (see [CCD+23]).

3. Large f: Since a smaller number of steps $\lceil e/f \rceil$ in the response isogeny recomputation improves verification performance (see [CEMR24]), we require f to be as large as possible, such that $2^f \mid p+1$.

4. Power of 3: For an efficient challenge isogeny computation, we require $2^f 3^{f'} \geq 2^\lambda$ with $2^f 3^{f'} \mid p+1$, and $3^{f'} \mid T$.

There is an obvious conflict between requirements (2) and (3): larger values of f leave a smaller factor of $(p^2-1)/2$ (of $\log_2(2p-f-1)$ bits), from which we can pick smooth factors for T. As mentioned above, SQIsign aims at fast verification, thus prioritising (3) over (2).

We note that all current implementations of SQIsign restrict to primes $p \equiv 3 \pmod 4$, since this less general case significantly simplifies implementations and is beneficial for fast field arithmetic over $\mathbb{F}_{p^2}$. Our search methods follow this restriction.

2.2 AprèsSQI prime requirements

AprèsSQI [CEMR24] is a variant of SQIsign focusing on faster verification. Recall that SQIsign requires $T \mid (p^2-1)/2$ to ensure that all isogeny computations take place over $\mathbb{F}_{p^2}$. However, the fact that $T > p^{5/4}$ limits the size of f to a theoretical maximum of $2^f < p^{3/4}$. In practice, reaching this maximum is infeasible, and we often only reach $2^f \approx p^{1/4}$ for acceptable smoothness bounds B of T (see, for example, [DLLW23]).

AprèsSQI relaxes this condition on T by allowing isogeny computations over small extension fields $\mathbb{F}_{p^{2k}}$. This means that T can be composed of prime power factors of $p^k \pm 1$ for small k, easing the requirement on T. Hence, the theoretical maximum on the size of f in AprèsSQI is $2^f \approx p$, which allows for improved verification performance at the cost of potentially more expensive signing due to isogeny computations over extension fields.

All prime requirements from SQIsign carry over to AprèsSQI with the exception that AprèsSQI only requires $T \mid N_k(p)$ with

$$N_k(p) = \prod_{d=1}^{k} \Phi_d(p^2)/2,$$

for small values of k (see [CEMR24, Thm. 1]) instead of requiring $T \mid (p^2-1)/2$. If $f > \lambda$, we can set the challenge isogeny degree to 2^λ, and no additional power of 3 dividing $p+1$ is required (see requirement (4) above).

2.3 POKE prime requirements

POKE [Bas24] is a framework introduced by Basso to construct efficient PKEs, Split KEMs and OPRFs. We focus on the parameters used for the PKE [Bas24, §4]. In the set-up of the encryption scheme, a prime of the form $p = 2^a 3^b c - 1$ is fixed, where c is a cofactor to ensure primality. Due to security, we require $2^a \approx 2^\lambda$, $3^b \approx 2^{2\lambda}$ and $p - 1$ has a divisor x such that $x \approx 2^{\lambda/2}$ (where x does not need to be smooth). As a result, we have that $p \approx 2^{3\lambda}$. Although not strictly required, the security analysis of POKE is simplified by choosing $x \approx 2^{\lambda/2}$ to be prime.

The parameter finding methods presented in this paper allow us to identify prime numbers p close to $2^{2\lambda}$ such that $2^a T \mid p^2 - 1$ (for some smooth odd integer T), with which we can instantiate the PKE scheme. Ideally, the efficiency benefits of using a smaller prime outweighs the performance penalty incurred by employing smooth T-torsion compared to 3^b-torsion. Additionally, this strategy leads to reduced public key and ciphertext sizes. This idea has been mentioned by Basso [Bas24], but no corresponding parameters were provided.

2.4 Prior search methods

Prior methods of finding friendly primes for isogeny-based schemes focused on first finding B-smooth twins $(r, r+1)$ resp. pairs $(r, r+1)$ satisfying the relevant requirements. Then they define $p = 2r + 1$ and check if p is prime. In this case, $p^2 - 1 = 4r(r+1)$ inherits the relevant properties from $(r, r+1)$.

There are several approaches for finding suitable pairs $(r, r+1)$: constructive approaches using Pell equations or the Conrey-Holmstrom-McLaughlin algorithm to find fully smooth pairs $(r, r+1)$, and search methods based on polynomials or the extended Euclidean algorithm, potentially allowing for rough factors in $r(r+1)$.

Pell equations. Given $B > 0$ and a list of primes $\{2, 3, \ldots, q\}$ smaller than B of cardinality $\pi(B)$, a method to find all B-smooth twins is given by Størmer [Stø97] and further work by Lehmer [Leh64]. Assuming that $(r, r+1)$ are B-smooth twins, we set $x = 2r + 1$, so $x - 1$ and $x + 1$ are smooth. Defining D as the squarefree part of their product, we get $x^2 - 1 = Dy^2$ for some $y \in \mathbb{Z}$ with Dy^2 being B-smooth. For all $2^{\pi(B)}$ possible values of $D = 2^{\alpha_2} \cdot 3^{\alpha_3} \cdot \cdots \cdot q^{\alpha_q}$ with $\alpha_i \in \{0, 1\}$, Størmer proposes to solve the Pell equation $x^2 - Dy^2 = 1$, hence finding all B-smooth values of y, and therefore *all* B-smooth twins.

In principle this method could be used to find parameters with optimal smoothness bounds, but solving $2^{\pi(B)}$ Pell equations quickly becomes infeasible for large enough B. Indeed, the current record solves all 2^{30} Pell equations for $B = 113$, where the largest pair of twin smooths is of size roughly 2^{74}, and therefore too small for cryptographic purposes. A follow-up work shows that this method can be adapted to increase the chances of finding cryptographically sized instances of twin smooths [BHL+22]. However, there seems to be no method to enforce special requirements such as large powers of two dividing $r(r+1)$, making this method unsuitable for schemes like SQIsign or POKE.

The Conrey-Holmstrom-McLaughlin algorithm. Given a smoothness bound B, the Conrey-Holmstrom-McLaughlin (CHM) algorithm [CHM13] tries to construct B-smooth twins based on a simple observation: Given B-smooth twins $(r, r+1)$ and $(s, s+1)$ with $r < s$, computing

$$\frac{r}{r+1} \cdot \frac{s+1}{s} = \frac{t}{t'}$$

often yields $t' = t + 1$ when writing t/t' in lowest terms, hence producing a new pair $(t, t+1)$ of B-smooth twins.

The CHM algorithm defines a starting set $S = \{1, \ldots, B-1\}$ of integers r defining twins smooths $(r, r+1)$, and iterates over all pairs $S \times S$ to generate new twin smooths by the above equation. After adding new solutions to S, this procedure is repeated until no more new solutions can be found.

Although this algorithm finds *almost all* B-smooth twins for small values of B, it quickly becomes infeasible to run due to the large number of twins. The current record of running an optimised variant of CHM sets $B = 547$, and finds a set of 82,026,426 twin smooths [BCC+23], the largest of which is a 122-bit pair. Finding large enough twin smooths for cryptographic applications using only the CHM algorithm seems infeasible, and it is not known how to enforce further conditions such as a large power of 2 dividing $r(r+1)$, making it unsuitable for schemes like SQIsign or POKE.

The extended Euclidean algorithm (XGCD). This method was first proposed by Costello to find twin smooth integers to create B-SIDH-friendly primes [Cos20], and later adapted to SQIsign [DKL+20, DLLW23]. To generate a pair $(r, r+1)$ of 2λ bits, we choose two coprime B-smooth numbers a and b of size roughly 2^λ. Since $\gcd(a, b) = 1$, the XGCD algorithm returns integers s, t with $|s| < |b/2|$ and $|t| < |a/2|$ that satisfy $as + bt = 1$. If s and t have large enough smooth factors, we can use pairs $\{r, r+1\} := \{|as|, |bt|\}$ to generate primes of size roughly $2^{2\lambda}$ for isogeny applications.

The key observation underlying this approach is that the product $s \cdot t$ of two numbers of size roughly 2^λ is much more likely to be smooth (or to contain a large enough smooth factor) than a random integer of size $2^{2\lambda}$. This can easily be deduced using the Dickman–de Bruijn function (see e.g. [CMN21]).

An advantage compared to the methods above is that we are free to choose any smooth numbers a and b, hence we can e.g. choose one of them to be divisible by a large power of two. Note however that we have to ensure a large enough search space, for instance, by fixing a suitable a and iterating over random smooth numbers b.

This method was used for SQIsign to find the 254-bit prime p_{3923}, featuring $f = 65$ and $B = 3923$ as a smoothness bound for the odd factor T [DLLW23]. Beyond the NIST-I security level, this method does not seem to scale well, see [CMN21, Table 3].

Searching with x^n polynomials. Another method proposed by Costello uses $(r, r+1) = (x^n - 1, x^n)$ for small values of n [Cos20]. For good choices of n, this has the advantage that $x^n - 1$ splits into smaller factors. For instance, when looking for a 2λ-bit pair $(r, r+1)$ using $n = 4$, we get that $x^4 - 1 = (x-1)(x+1)(x^2+1)$. Hence, we get one λ-bit and two $\lambda/2$-bit factors, which increases the chance for a large enough smooth factor of $x^4 - 1$, similar to the case of XGCD. Again, note that we require a large enough search space and so n has to be chosen relatively small.

This approach has been used for finding SQIsign primes for NIST-I security, choosing x to be a smooth number divisible by a large power of two [DLLW23]. However, none of the primes found by this method outperformed p_{3923} found via the XGCD approach. Similar to XGCD, this method does not scale well to higher security levels, see [CMN21, Table 3].

To overcome this issue, we can instead use this approach to *boost* twin smooth integers $(r, r+1)$ [BCC+23]. Continuing the example above, since $(x-1)$ divides $x^4 - 1$, we can set $x = r + 1$ for B-smooth twins $(r, r+1)$, ensuring that $x - 1 = r$ is smooth too. Setting our prime to be $p = 2x^4 - 1$ with $x = r + 1$, we have a guaranteed smooth factor of 1.25λ bits in $p^2 - 1 = 4x^4(x^4 - 1)$, improving the chances to find friendly primes for isogeny schemes. In [BCC+23] twin smooth integers found with the CHM approach were used as inputs, ensuring optimal smoothness bounds. Even though this produces somewhat SQIsign-friendly primes for the NIST security levels I, III, and V, the involved powers of 2 are too small to achieve efficient instantiations. We will reuse this boosting technique in a different context in the following sections.

Searching with PTE solutions. The main drawback of the approach using x^n polynomials is that $x^n - 1$ does not fully split in linear factors for $n > 2$, such that the non-linear factors hamper the probability of finding enough smooth divisors of $x^n - 1$. Instead, we can use $(r, r+1) = (f(x)/c, g(x)/c)$, where $f(x)$ and $g(x)$ are fully split polynomials of degree n that differ by a constant c [CMN21]. This means that the overall smoothness probability is boosted through r and $r+1$ splitting in n smaller factors, respectively. Such polynomials can be found through solutions to the Prouhet-Tarry-Escott (PTE) problem, and suitable pairs $(r, r+1)$ can be found using a sieving technique [CMN21].

However, in comparison to x^n polynomials, PTE polynomials do not allow for large exponents of linear factors of $f(x)$ and $g(x)$, hampering the probability of finding a large power of 2 dividing $r(r+1)$. To overcome this, Sterner proposed to use polynomials that *almost* split into linear factors, but allow for larger powers of linear factors [Ste23]. Nevertheless, although improving over [CMN21] for some target sizes of $(r, r+1)$, this method does not seem to produce appealing SQIsign-friendly primes due to their relatively modest power of two factor.

2.5 Cost metric for evaluating primes

To evaluate the relative merits of the primes found using our search methods, we define a detailed cost metric. This metric considers the efficiency of a low-level implementation for computing T-isogenies from kernels defined over $\mathbb{F}_{p^2}$ or $\mathbb{F}_{p^{2k}}$ and largely follows the framework presented in [AAA+24a, AAA+24b]. To assess the cost of computing a T-isogeny, where $T = \prod_{i=1}^{r} \ell_i^{e_i}$, we start by writing a list of tuples

$$L := [(\ell_i, e_i, k_i)]_{i=1}^{r},$$

where k_i denotes the smallest field-extension $\mathbb{F}_{p^{2k_i}}$, where $\ell_i^{e_i}$-torsion can be found. For each prime power, $\ell_i^{e_i}$, we estimate the cost of

1. Successively computing image curves by applying an $\ell_i^{e_i}$-isogeny mapping to the domain curve,

2. Pushing all the remaining kernel points necessary to compute the remaining prime power factors of T through this isogeny,

3. Drawing inspiration from methods used for isogeny computations in the CSIDH key exchange protocol [BBC+21], we require kernel points with torsion for all remaining prime factors ℓ_i within a given extension field k_i. This approach requires computing multiple scalar multiplications, which can be computed efficiently using Montgomery ladders and short differential addition chains for each ℓ_i, as described in [CCC+19, CCC+24].

We compute isogeny image curves and push points to them using one of three different methods for each prime power $\ell_i^{e_i}$: either using Vélu, or $\sqrt{\text{élu}}$, or using the approach from Deuring for the People [EPSV24]. Denoting each of these costs by $C_{\text{Vélu}}, C_{\sqrt{\text{élu}}}$ and C_{DftP}, we get the total cost as

$$C_{\text{isog},p}(T) = \sum_{(\ell_i, e_i, k_i) \in L} \min\{C_{\text{Vélu}}(\ell_i^{e_i}, k_i), C_{\sqrt{\text{élu}}}(\ell_i^{e_i}, k_i), C_{\text{DftP}}(\ell_i^{e_i}, k_i)\}. \tag{1}$$

If all the torsion in the T-group is rational (i.e., defined over the field $\mathbb{F}_{p^2}$), then C_{DftP} is never better than $\min\{C_{\text{Vélu}}, C_{\sqrt{\text{élu}}}\}$. For our case studies SQIsign and POKE, this property always holds true. However, the cost model for AprèsSQI becomes considerably more intricate. This complication arises from the large number of computational alternatives for processing prime factors defined over different field extensions.

We note that our cost model of Equation (1) carefully takes into consideration the computational expenses associated to perform scalar multiplications as mentioned in Item 3, which allows us to recover kernel points with the required torsion $\ell_i^{e_i}$. Furthermore, note that the order chosen for processing prime factors significantly impacts the total cost of Equation (1). Determining the optimal order and the optimal number of kernel points to minimise this cost presents a challenging optimisation problem. More on this complexity is explained below (cf. Section 6.2).

Remark 3. When assessing the quality of a prime for SQIsign and AprèsSQI, the total number of T-isogenies also depends on the largest f such that $2^f \mid p+1$. Thus, we can in this case use $C_{\text{isog},p}$ to define a new metric as

$$C_{\text{Sign},p}(T) = \lceil e/f \rceil \cdot C_{\text{isog},p}(T), \tag{2}$$

where 2^e is the length of the response isogeny, and $\lceil e/f \rceil$ accounts for the total cost of all T-isogenies computed during signing.

Remark 4. For optimised implementations, the efficiency of the field arithmetic over $\mathbb{F}_p$ and its extensions is an important measure for the quality of a prime. For instance, primes of the form $p = 2^f \cdot c - 1$ are considered to be *Montgomery-friendly*, resulting in particularly efficient operations over $\mathbb{F}_p$ when f is large, see [BD21] or [CCC+24] in the context of isogenies. The primes in our work increasingly benefit from these optimisations depending on the size of f. A more precise cost metric could thus use a factor $C_{\text{arith}} \in (0, 1]$ in Equation (2) to reflect the efficiency of field arithmetic implementations, where C_{arith} decreases with increasing size of f. However, it seems difficult to assess the exact values of C_{arith} from the literature without optimised implementations. Therefore, we ignore this factor in our cost metric.

Remark 5. Evaluating the exact cost of arithmetic operations in prime field extensions $\mathbb{F}_{p^{2k}}$ is a complex task that must consider several factors. For example, the presence of irreducible binomials of the form $x^k - \beta$ enable efficient reductions in the field. It is also crucial to develop formulas minimising the number of base field multiplications required for field multiplication and squaring in the extension field. A popular approach is to follow the Karatsuba-like field multiplication framework pioneered by Montgomery [Mon05] that has seen significant advancements in subsequent publications (see [Cen18] for a survey).

Selecting T when allowing extension-fields. Even when fixing a prime p, selecting the T-torsion that minimises $C_{\text{isog},p}(T)$ seems to be a difficult task. In protocols where we are solely working over $\mathbb{F}_{p^2}$, the following simple greedy algorithm works well: Start with $T = 1$ and multiply T by the smoothest prime power factors of $p^2 - 1$ until T is large enough, and then "shave off" smaller factors of T, if this keeps T above the size bound.

However, selecting T becomes much more complicated when allowing for extension fields. In this case, we also use a greedy algorithm, which first appeared in the software accompanying Deuring for the People [EPSV24], which we now briefly describe.

The idea is to start with a natural first approximation, namely setting T as power-smooth as possible, while keeping it above the size bound. Next, it sorts each prime power $\ell^e \mid T$, based on the cost of computing the ℓ^e-isogeny, divided by the size of ℓ.[1] From this point, it sequentially increases k, looking for new prime-power factors available in each extension $\mathbb{F}_{p^{2k}}$. In each extension, we compute the maximal $\ell_{\max}$ such that computing an $\ell_{\max}$-isogeny over $\mathbb{F}_{p^{2k}}$ is cheaper than the currently most expensive part of T.

The iterative process discussed above continues until until $\ell_{\max} < 3$. We can then iteratively remove any remaining small prime factors from T as long as it maintains its size

[1] This approach assumes that the cost of the ℓ^e factor is independent of the other factors of T. However, this is a first order simplification that should be used with care and that does not quite apply to the cost function $C_{\text{isog},p}$.

above the prescribed size bound. This approach achieves reasonable efficiency in practice and yields good choices of T. However, identifying methods to determine the absolute optimal choices of T for a given prime p remains an open problem as far as we know.

3 Sieve-and-boost

Our first prime search method, the sieve-and-boost method, builds upon the x^n-technique described in Section 2.4. However, it introduces a key modification: enforcing the presence of a large power of 2 in the prime factorisation.

This technique was outlined and used in [DLLW23] for $n = 4$ in search of NIST-I primes, but did not result in any good primes. In this work, we formalise the technique. For application to SQIsign, we also extend the search to other polynomials, and search for larger parameter sizes. This leads to the discovery of suitable parameter choices for NIST-III and NIST-V.

We additionally apply this technique to AprèsSQI and POKE for security level NIST-I. However, we will see that better parameters for these schemes can be found with the technique in Section 4. Details regarding the specific searches conducted and their results are presented in Section 6.

Description of technique. We search for primes of the form

$$p = 2(2^{f_n}3^{f'_n}x)^n - 1 \tag{3}$$

for a smooth number x, where the choice of (f_n, f'_n) depends on the specific prime requirements on $f \geq nf_n + 1$ and $f' \geq nf'_n$ for the protocols in use.

This idea is adapted from [Cos20], which is based on the observation that when $p_n(x) := 2x^n - 1$, then

$$p_n(x)^2 - 1 = 4x^n \prod_{d|n} \Phi_d(x),$$

where Φ_d denotes the d-th cyclotomic polynomial (see [BCC+23]). Fixing values n, f_n, f'_n and a smoothness bound B, the search consists of two phases.

Sieving. Find all B-smooth numbers x in some suitable range $[L, R]$, with a sieving approach based on the classic sieve of Eratosthenes, see [CMN21].

Boosting. For all smooth x_i in $[L, R]$, compute $p_i = 2(2^{f_n}3^{f'_n}x_i)^n - 1$. For all primes p_i, compute the values $N_i = p_i^2 - 1$, and find the (odd) B-smooth part T_i of all the N_i, either using trial division or a more optimised remainder tree approach [Ber04]. Discard p_i if T_i is smaller than required.

Next, we describe a few specifics for applying the sieve-and-boost search to the three protocols we consider.

Sieving for SQIsign primes. Recall that for SQIsign, we have the requirement that $2^f 3^{f'} \geq 2^\lambda$ such that (a divisor of) $2^f 3^{f'}$ is the degree of the challenge isogeny, and $p \approx 2^{2\lambda}$, while we also want to push f to be as large as possible. Thus, when $2^f 3^{f'} \approx 2^\lambda$, and when searching for security levels around $\lambda = 128, 192, 256$ (corresponding to primes of sizes around $256, 384$ and 512 bits), the search space for x is roughly of size λ/n bits. For larger values of n, we can thus exhaust the full search space of potential smooth values of x. In other cases, we artificially reduce the size of x by increasing the size of f and f'. Specifically, if x is of bitsize m, and f_0 is the smallest value of f_n we accept, then the largest value of f_n becomes $\lfloor 2\lambda/n \rfloor - m$, and we can try all primes of the form

$$2(2^{f_n}3^{f'_n}x)^n - 1,$$

where $f_n \in [f_0, \lfloor 2\lambda/n \rfloor - m]$, and f'_n is set to be so the prime is of the right size. Specifically, we set $f'_n = \lfloor \log_3(2^{2\lambda/n - f_n - m}) \rfloor$.

Sieving for AprèsSQI primes. When searching for parameters for AprèsSQI, we have the same requirements as in SQIsign, except that in the boosting step, we allow for searching for the B-smooth part of

$$N_i := N_k(p_i) = \prod_{d=1}^{k} \Phi_d(p_i^2)/2 = \mathrm{lcm}(\{p_i^{2d} - 1\}_{d=1}^{k}),$$

instead of the usual $p_i^2 - 1$. Hence, we can run a straightforward adaptation of the SQIsign technique with this value of N_i. However, as one typically wants lower smoothness bounds over higher extension fields as computing these isogenies becomes more expensive, we can also adapt the sieve to find the smooth parts of each extension separately, with different smoothness bounds per extension. Specifically, we set a maximal extension $k_{\max}$, and for all $k \in \{1, \ldots, k_{\max}\}$, we choose a bound B_k, compute the B_k-smooth part of $p_i^{2k} - 1$, denoted T_k. Then finally, we compute T as

$$T = \mathrm{lcm}(\{T_k\}_{k=1}^{k_{\max}}).$$

Computing the T_k can again be done either simply with trial division, or in batches using remainder trees (with different trees for different bounds).

After identifying a prime candidate p through the sieve, we know from the search that there exists a T which is smooth enough over small extension. However, the sieve might not necessarily identify the optimal choice of T. Therefore, we post-process each prime p with the greedy algorithm from Section 2.5 to potentially find a more suitable value for T. This refined T is then used to evaluate the prime based on the computational cost of T-isogenies.

Sieving for POKE primes. From the requirements of POKE, we see that we can apply the same search as for SQIsign, except that we always fix $f = \lambda$, and relax the size requirement for T to only $2^{2\lambda}$ (down from $T > p^{5/4} \approx 2^{5\lambda/2}$ for SQIsign and AprèsSQI). Thus, in terms of smoothness of $p^2 - 1$, finding POKE primes is roughly as difficult as finding SQIsign primes with $f = \lambda/2$ for a given smoothness bound B.

4 XGCD-and-boost

We now describe the second technique that we call the XGCD-and-boost technique, which combines the XCGD approach for finding twin smooths from Section 2.4 with the boosting technique. Compared to sieve-and-boost, this technique is particularly useful for smaller parameter sets and smaller choices of n, where we cannot exhaust the full search interval. We apply this technique to SQIsign, AprèsSQI, and POKE. Details on the specific searches we ran and their results are presented in Section 6.

Description of technique. To generate primes of size 2λ, we aim to generate pairs $(r, r \pm 1)$ of twin smooth numbers of size $2^{2\lambda/n}$, and use them as inputs to $p_n(x)$ with $x = r$, similar to the technique of [BCC+23] described in Section 2.4. Since both x^n and $x \pm 1$ are divisors of $p_n(x)^2 - 1$, we are guaranteed a smooth factor T' of $\frac{n+1}{n}(\log_2(p) - 1) + 2$ bits in this case. However, unlike [BCC+23], we instead use the XGCD approach to generate the pair $(r, r \pm 1)$.

Thus, we again have a technique consisting of two phases. First, we fix a value $a = 2^{f_n} 3^{f'_n} \approx 2^{\lambda/n}$, depending on the requirements on $f \geq nf_n + 1$ and $f' \geq nf'_n$ of the

protocol. Further, we also fix a smoothness bound B, and a parameter M to control the size of the search space. The procedure can be summarized as follows:

XGCD. Sample M random B-smooth primes $\ell_j \notin \{2, 3\}$, and set $b_i = \ell_{\text{small}}^{k_i} \prod \ell_j$, where $\ell_{\text{small}} > 3$ is a small prime and k_i is chosen so that $b_i \approx 2^{\lambda/n}$. Then, compute s_i, t_i through XGCD such that $s_i a_i + t_i b_i = \gcd(a_i, b_i) = 1$, and store the pair $(r_i, r_i \pm 1) = (|s_i a_i|, |t_i b_i|)$ where $r_i \approx 2^{2\lambda/n}$. Collect all such pairs. The search space is of size $\binom{(\pi(B)-2)+M-1}{M}$.

Boost. For each pair $(r_i, r_i \pm 1)$, compute $p_i = 2(r_i)^n - 1$. For each prime p_i, compute the value $N_i = p_i^2 - 1$, and find the (odd) B-smooth part T_i of all the N_i either using trial division or a more optimised remainder tree approach [Ber04]. Discard p_i if T_i is smaller than required.

Notice that a priori, we do *not* need to require that the pair $(r_i, r_i \pm 1)$ is fully B-smooth, as a large enough smooth factor of $r_i(r_i \pm 1)$ could still give useful parameters, as long as a sufficiently big part of $N_i = p_i^2 - 1 = p_n(r_i)^2 - 1$ is B-smooth.

We can increase the search space by iterating through multiple XGCD solutions of appropriate size (see [Cos20]). Given a solution (s, t) for the input (a, b), there is an infinite number of solutions of the form $(s_j, t_j) = (s + jb, t - ja)$ with $j \in \mathbb{Z}$. However, only a limited number of these solutions is suitable for our purposes due to the size requirement of $r = |as_j| < 2^{2\lambda/n}$. On average, this increases the search space for our searches described in Section 6 by factor 4.

An *unbalanced* variant of this approach first samples M' small prime factors $\ell_j \notin \{2, 3, \ell_{\text{small}}\}$ for $1 \leq j \leq M'$, and sets

$$a = 2^{f_n} \cdot 3^{f'_n} \cdot \prod_{j=1}^{M'} \ell_j,$$

with (f_n, f'_n) as chosen above, increasing the size of a. In contrast, we decrease the size of b by sampling only $M - M'$ small prime factors $\ell_j \notin \{2, 3, \ell_1, \ldots, \ell_{M'}\}$ for $M' + 1 \leq j \leq M$ and setting

$$b = \ell_{\text{small}}^{k_i} \prod_{j=M'+1}^{M} \ell_j$$

with k_i such that $ab \approx 2^{2\lambda/n}$. Since $r = |as| \approx 2^{2\lambda/n}$, this reduces the size of the XGCD solution s, and thus increases the smoothness probability for r. This is beneficial, since r is the *boosted* factor, i.e., we have $r^n \mid (p_n(r)^2 - 1)$. In contrast, the smaller size of b and larger size of t reduce the smoothness probability of $r \pm 1 = |bt| \approx 2^{2\lambda/n}$. However, $r \pm 1$ does not necessarily need to be fully smooth, since other factors of $p_n(r) - 1$ are likely to contain smooth factors too, making this approach favorable overall. The search space is slightly reduced in this approach, due to the fact that a and b have to be coprime, yet this effect is negligible in practice.

XGCD-and-boost for SQIsign. To satisfy the SQIsign requirements, in the balanced approach we pick $a = 2^{f_n} \cdot 3^{f'_n} \approx 2^{\lambda/n}$ with (f_n, f'_n) such that potential primes $p = p_n(r) = p_n(|as|)$ satisfy

$$a^n = 2^{nf_n+1} \cdot 3^{nf'_n} \mid (p + 1),$$

i.e., $f \geq nf_n + 1$ and $f' \geq nf'_n$, and $2^{nf_n+1} 3^{nf'_n} \geq 2^\lambda$. We set M and B such that $\ell^M < 2^{\lambda/n}$ for the largest prime $\ell \leq B$ and compute $b \approx 2^{\lambda/n}$ accordingly. The unbalanced approach can be instantiated similarly with minor adaptations as described above. The smoothness bound B can be chosen such that the probability for finding twin smooths

$(|as|, |bt|)$ is larger than the inverse of the size of the search space, so it is likely to find twin smooths.

We particularly highlight the case $n = 2$ and aiming for $f' = \lambda/4 - 1$. In this case, when we find twin smooths $(r, r \pm 1)$ via XGCD, we are guaranteed that $p_n(r)^2 - 1$ contains a large enough odd smooth factor T to meet the SQIsign requirements, hence we only need $p_n(r)$ to be prime.

XGCD-and-boost for AprèsSQI. The adaptations to the above search to find AprèsSQI primes are completely analogous to the description in Section 3. In particular, we allow T to be a factor of

$$N_i = \prod_{d=1}^{k} \Phi_d(p_i^2)/2 = \mathrm{lcm}(\{p_i^{2d} - 1\}_{d=1}^{k}).$$

We note that the XGCD-and-boost approach limits the size of f compared to sieve-and-boost. Maximising f can be done by setting $a = 2^{f_n}$, but since we require a large enough search space, we cannot pick a too large, as this would decrease M. Therefore, in practice we set a limit of $a \approx 2^{\lambda/n}$, resulting in a maximum of $f \approx nf_n + 1 = \lambda + 1$.

XGCD-and-boost for POKE. Similar to the adaptations for sieve-and-boost in Section 3, finding POKE-friendly primes requires setting $a = 2^{f_n} = 2^{\lambda}$, such that $f \geq \lambda + 1$. In contrast, the size of the odd smooth factor T is only required to be of size $T > 2^{2\lambda}$. As for sieve-and-boost, finding POKE primes is roughly as difficult as finding SQIsign primes with $f = \lambda/2$ for a given smoothness bound B.

5 Smoothness probabilities

Unfortunately, it is not possible to know exactly what is the best prime we can obtain for each scheme and security level. The best we can do is determine how likely we are to find a prime p such that $p^2 - 1$ contains a large enough B-smooth factor using a certain method. In this section, we discuss how to compute these probabilities with respect to the methods we use.

For our cryptographic applications, we saw that it is enough to chose a prime p such that

$$N_k(p) = \prod_{d=1}^{k} \Phi_d(p^2)/2,$$

where Φ_d denoted the d-th cyclotomic polynomial, is sufficiently smooth. For example, in SQIsign we take $k = 1$ and require $2^f T \mid p^2 - 1$ with smooth odd $T \approx p^{5/4}$ and f as large as possible. As the search to find suitable primes is a computationally heavy task, it is essential to determine what parameters defining the searches are likely to produce good primes yielding sufficient smoothness.

We follow Banks and Shparlinski [BS07] to determine the probability of $N_k(p)$ being sufficiently smooth given some prime p. More precisely, given a smooth integer $r \mid N_k(p)$ or a pair $(r, r \pm 1)$ of twin smooths with $r(r \pm 1) \mid N_k(p)$, we want to calculate the probability that $N_k(p)$ has b bits of B-smoothness.

The first step is to determine the probability that the factor r or $r(r \pm 1)$ is fully smooth. To do so, we use the following counting function:

$$\Psi(X, B) = \#\{x \leq X : x \text{ is } B\text{-smooth}\}.$$

For a large range of X and B, it is known that

$$\Psi(X, B) \sim \rho(u)X,$$

Table 1: This table gives the probability of obtaining B-smooth integer r or B-smooth twins $(r, r \pm 1)$ of bitsize $\log_2(r)$, computed with the Dickman-de Bruijn function.

	Probability of smooth r for smoothness bound B			Probability of smooth twins $(r, r \pm 1)$ for bound B		
$\log_2(r)$	$B = 2^{11}$	$B = 2^{15}$	$B = 2^{18}$	$B = 2^{11}$	$B = 2^{15}$	$B = 2^{18}$
32	$2^{-4.1}$	$2^{-2.0}$	$2^{-1.2}$	$2^{-8.2}$	$2^{-4.0}$	$2^{-2.5}$
64	$2^{-14.9}$	$2^{-8.6}$	$2^{-6.1}$	$2^{-29.7}$	$2^{-17.3}$	$2^{-12.3}$
128	$2^{-44.0}$	$2^{-27.5}$	$2^{-20.6}$	$2^{-87.9}$	$2^{-55.0}$	$2^{-41.3}$
192	$2^{-79.0}$	$2^{-50.6}$	$2^{-38.6}$	$2^{-157.3}$	$2^{-101.1}$	$2^{-77.3}$
256	$2^{-116.9}$	$2^{-76.2}$	$2^{-58.9}$	$2^{-233.9}$	$2^{-152.5}$	$2^{-117.7}$

where $u = (\log X)/(\log B)$ and ρ is the Dickman–De Bruijn function [Dic30, dB66]. This function is implemented in various computer algebra software packages, such as SageMath, from which we can compute $\Psi(X, B)$ for various X and B.

Explicitly, we compute the probability of finding a B-smooth integer $r \leq 2^{b'}$ for some $b' \in \mathbb{Z}_{>0}$ as $\rho(\frac{b'}{\log B})$. To compute the probability of finding a B-smooth pair $r, r \pm 1 \leq 2^{b'}$, we assume that the probabilities of r being smooth and $r \pm 1$ being smooth are independent [CMN21, Heuristic 1]. This is quite a strong assumption but we will see from our results that it is sufficient for our estimates. Then, the probability is given by $\rho(\frac{b'}{\log B})^2$. In Table 1, we give these probabilities for a range of $\log_2(r)$. We find that, for fixed B we have a lower probability of finding a smooth integer r of larger bitsize b'. Similarly, for a fixed bitsize, the probability of finding r decreases as B decreases.

Next, we investigate the probability of $N_k(p)$ having enough smoothness. More precisely, given smooth integers r (where $r \pm 1$ is also smooth for the XGCD-and-boost method), we want to determine with what probability $N_k(p)$ has b bits of B-smoothness with $p = p_n(R)$, where $p_n(x) = 2x^n - 1$ and R is an integer depending on r, whose precise value depends on the method we use. For this will use another counting function

$$\Theta(X, B, D) = \#\{x \leq X : D < \text{ largest } B\text{-smooth divisor of } x\}.$$

The value $\Theta(X, B, D)$ will give the number of positive integers $\leq X$ for which there exists a B-smooth divisor $d > D$. The first two terms in the asymptotic expansion of $\Theta(X, B, D)$ for X, B, D varying over a large domain are given by Banks and Shparlinski [BS07, Theorem 1]. Using this, we can compute the probability $\Theta(X, B, D)/X$ that we are interested in up to a sufficient level of precision.

For simplicity, let us consider the case where $k = 1$ and $N_1(p) = (p^2 - 1)/2$. This is required for SQIsign and POKE-friendly primes; for AprèsSQI we can allow $k \geq 1$. We can readily see that $p_n(x) = 2x^n - 1 = 4x^n(x-1)g(x)$ for some polynomials $g(x) \in \mathbb{Z}[x]$. Write $g(x) = g_1(x) \cdots g_m(x)$, where each g_i is irreducible of degree $d_i = \deg(g_i)$ and $d = \deg(g)$. We will assume that the probability of $g(r)$ having at least b bits of B-smoothness is the product of the probabilities of each of its factors g_i having at least b_i bits of B-smoothness where $b = \sum_{i=1}^{m} b_i$. This assumption was also made in [BCC+23] and can be seen as an extension of [CMN21, Heuristic 1]. Furthermore, similarly to [BCC+23], we assume that smoothness is distributed evenly between g_i (weighted by the degree), i.e., $b_i = d_i \frac{b}{d}$. Note that, in reality, this only gives us a lower bound for the probability, but this will suffice for our purposes.

Sieve-and-boost. Under these assumptions, given a b'-bit B-smooth integer r, the probability that $p^2 - 1$ has b-bits of smoothness with $p = p_n(R)$ for $R = 2^{f_n}3^{f'_n}r$ is

$$\prod_{i=1}^{m} \frac{\Theta(R^{d_i}, B, 2^{d_i(b-b_g)/d})}{R^{d_i}}, \tag{4}$$

where $b_g = n(b' + f_n + f'_n \cdot \log_2 3) + 2$ is the guaranteed smoothness and $p_n(x) = 4x^n(x - 1)g_1(x) \cdots g_m(x)$ as above.

XGCD-and-boost. We are now given a pair $(r, r \pm 1)$ of B-smooth b'-bit integers. The probability that $p^2 - 1$ has b-bits of smoothness, where $p = p_n(R)$ where $R = r$ is again given by Equation (4), where now $b_g = \frac{n+1}{n}(\log_2(p) - 1) + 2$ is the amount of guaranteed smoothnes. We remark that this mirrors the way the probabilities are computed for the CHM-and-boost method in [BCC+23, Table 3].

5.1 Probabilities for SQIsign primes

In Table 2, we give the probabilities for constructing SQIsign parameters for NIST Level I, III and V security using sieve-and-boost. We observe that the probability of finding a SQIsign friendly prime given r is correlated to the degrees of the factors of $p_n(x)$; the probability is higher for larger degree n, especially when they factor into lots of small degree polynomials. Indeed, in this case, we only have to search for smoothness in many small integers. We remark, however, that for bigger n, the sieve interval for r is not large enough. For each NIST security level, we chose n that balance these two opposing constraints.

In Table 3 we show the probabilities for the XGCD-and-boost method. As n increases, the bitsize of r decreases and the amount of smoothness required to find increases. Therefore, the probability of finding SQIsign friendly primes p decreases with increasing n, however we can more easily find smooth twins of the required bitsize (see Table 1). As before, we must balance these two constraints when choosing appropriate n to run our searches.

The computation of probabilities for POKE can be done in an analogous way. Furthermore, by extending the probability formula for $N_k(p)$ with $k \geq 1$ we can analogously compute the probabilities associated to finding AprèsSQI primes. These will guide the parameters we choose to run the experiments in Section 6.

6 Results

In this section, we describe our implementations of the search approaches and the concrete search spaces we exhausted, and give our resulting primes for SQIsign, AprèsSQI, and POKE. In all cases, we collect a large list of primes satisfying the requirements for a suitable smoothness bound B, before scoring the primes with the cost metric described in Section 2.5 to determine the best primes.

We implemented our sieve-and-boost approach in Python, based on the implementation of PTE sieving [CMN21]. The involved sieving step, which is the performance bottleneck in this technique, uses the efficient C implementation by Costello, Meyer and Naehrig [CMN21].

Our implementation of the XGCD-and-boost approach is based on the XGCD prime search script by De Feo, Kohel, Leroux, Petit and Wesolowski [DKL+20], written in C++. All of our prime search were carried out on two AMD EPYC 7763 64-Core processors at 2.45 GHz, running up to 128 threads in parallel.

For evaluating primes, we use a SageMath script that chooses T and evaluates the cost of computing a T-isogeny, possibly over extensions of $\mathbb{F}_{p^2}$. It is based on the implementation

Table 2: This tables gives probabilities relating to the sieve-and-boost method. Let $R = 2^{f_n}3^{f'_n}r$ where r is B-smooth, $f_n = f/n$ and $f'_n = \log_3 2 \cdot \left(\frac{\log_2(p)}{2n} - f_n\right)$. This table presents the probability of $p = p_n(R)^2 - 1$ having a B-smooth divisor $2^f T \mid (p^2 - 1)$ with $T \approx p^{5/4}$.

		NIST-I $B = 2^{11}$ $\log_2(p) = 254$ $f = 64$		**NIST-III** $B = 2^{15}$ $\log_2(p) = 382$ $f = 96$		**NIST-V** $B = 2^{18}$ $\log_2(p) = 510$ $f = 128$	
		$\log_2(r)$	Probability	$\log_2(r)$	Probability	$\log_2(r)$	Probability
	2	63.5	$\approx 2^{-32.6}$	95.5	$\approx 2^{-38.2}$	127.5	$\approx 2^{-45.0}$
	3	42.3	$\approx 2^{-33.7}$	63.7	$\approx 2^{-39.5}$	85.0	$\approx 2^{-46.4}$
	4	31.8	$\approx 2^{-27.1}$	47.8	$\approx 2^{-32.3}$	63.8	$\approx 2^{-38.3}$
	6	21.2	$\approx 2^{-23.2}$	31.8	$\approx 2^{-27.2}$	42.5	$\approx 2^{-32.2}$
	8	15.9	$\approx 2^{-25.4}$	23.9	$\approx 2^{-30.0}$	31.9	$\approx 2^{-35.6}$
n	10	12.7	$\approx 2^{-26.4}$	19.1	$\approx 2^{-30.8}$	25.5	$\approx 2^{-36.2}$
	12	10.6	$\approx 2^{-20.3}$	15.9	$\approx 2^{-23.4}$	21.3	$\approx 2^{-27.2}$
	14	9.1	$\approx 2^{-27.8}$	13.6	$\approx 2^{-33.0}$	18.2	$\approx 2^{-39.2}$
	16	7.9	$\approx 2^{-17.1}$	11.9	$\approx 2^{-21.1}$	15.9	$\approx 2^{-25.8}$
	18	7.1	$\approx 2^{-21.7}$	10.6	$\approx 2^{-25.7}$	14.2	$\approx 2^{-30.6}$
	24	5.3	$\approx 2^{-15.3}$	8.0	$\approx 2^{-16.8}$	10.6	$\approx 2^{-18.0}$

Table 3: Let $R = r$ where $(r, r \pm 1)$ (resp. $(r, r - 1)$ for $n = 3$), is B-smooth. This table displays the probability of $p = p_n(R)^2 - 1$ having a B-smooth divisor $2^f T \mid (p^2 - 1)$ with $T \approx p^{5/4}$.

	n	$\log_2(r)$	Bits of smoothness required	Probability
NIST-I	2	127.0	0	1
$B = 2^{11}$	3	84.7	41.7	$\approx 2^{-8.7}$
$\log_2(p) = 254$	4	63.5	62.8	$\approx 2^{-11.7}$
$f = 64$	6	42.3	83.3	$\approx 2^{-14.2}$
	8	31.8	94.4	$\approx 2^{-18.8}$
NIST-III	2	191.0	0	1
$B = 2^{15}$	3	127.3	63.0	$\approx 2^{-10.4}$
$\log_2(p) = 382$	4	95.5	94.8	$\approx 2^{-13.9}$
$f = 96$	6	63.7	126.5	$\approx 2^{-16.9}$
	8	47.8	142.4	$\approx 2^{-22.2}$
NIST-V	2	255.0	0	1
$B = 2^{18}$	3	170.0	84.3	$\approx 2^{-12.3}$
$\log_2(p) = 510$	4	127.5	126.8	$\approx 2^{-16.5}$
$f = 128$	6	85.0	169.2	$\approx 2^{-20.2}$
	8	63.8	190.4	$\approx 2^{-26.3}$

Table 4: List of ranges that have been exhausted for searches using the sieve-and-boost approach described in Section 3. The degree n specifies the exponents of Equation (3) used in the prime search.

Sieve range	n for NIST-I	n for NIST-III	n for NIST-V
$[2^{45}, 2^{46}]$	-	4	-
$[2^{41}, 2^{42}]$	3	-	6
$[2^{31}, 2^{32}]$	4	6	8
$[2^{20}, 2^{21}]$	6	9	12
$[2^{15}, 2^{16}]$	8	12	16
$[2^{9}, 2^{10}]$	12	18	24

of [EPSV24] for choosing T, and relies on the data from the optimised implementation of [AAA+24a] for the cost of computing isogenies over extension fields using their Magma script [AAA+24b].

All the primes reported and discussed in this section are fully specified in Appendix A. Our SQIsign search implementations is available with the NIST Round 1 submission package of SQIsign.[2] Minor adaptations allow searching for primes with different requirements such as in AprèsSQI or POKE.

6.1 Results for SQIsign

For SQIsign, we ran a large search, using both sieve-and-boost and XGCD-and-boost, for security levels NIST-I, NIST-III, and NIST-V, i.e., primes of roughly 256, 384, resp. 512 bits. For estimating the performance of signing, we use our cost metric from Section 2.5 that approximates the performance bottleneck for computing all T-isogenies during signing, depending on both f and T. For estimating verification performance for NIST-I parameters, we use the data given in [CEMR24, Fig. 2], only depending on f.[3] Note that our searches were used to find the primes p^{I}_{1973}, p^{III}_{47441}, and p^{V}_{318233} for the NIST submission of SQIsign [CCD+23].

Sieve-and-boost search. For sieve-and-boost, we used polynomials $p_n(x)$, with $n \in \{3, 4, 6, 8, 9, 12, 16, 18, 24\}$. For different security levels, these polynomials naturally require different sieving intervals; we sieved a total of six different intervals, and applied them accordingly to find suitable parameters. An overview of which intervals gave which parameters can be found in Table 4. The smoothness bounds B we used depend on the size of f_n and vary between 2^{11} and 2^{13} for NIST-I, between 2^{15} and 2^{18} for NIST-III, and between 2^{16} and 2^{19} for NIST-V.

XGCD-and-boost search. For XGCD-and-boost, we are limited by being able to find (nearly) twin smooth integers $(r, r \pm 1)$, which requires large smoothness bounds for larger sizes such as $r \approx 2^{256}$ (see [CMN21, Table 3]). In contrast, for relatively small sizes of r the search space is small due to the small value of M. Referring to Section 5, the sweet spot for this approach thus is to find twin smooths of size $r \approx 2^{128}$, and use $n \in \{2, 3, 4\}$ to find parameters for the NIST-I, NIST-III, resp. NIST-V security level.

We ran this approach with various choices of $f_n \in [32, 40]$ and appropriate $f'_n \in [15, 20]$, such that $2^{2f_n} \cdot 3^{2f'_n} \approx 2^{64}$. The smoothness bound B was chosen as $B = 2^{11}$ for $f_n \leq 35$, $B = 2^{12}$ for $36 \leq f_n \leq 39$, and $B = 2^{13}$ for $f_n \geq 40$. representing the trade-off between smoothness of T and size of f. We used $\ell_{\mathrm{small}} \in \{5, 7\}$, $\ell_j < 2^{10}$, unbalancing by $0, 1$, or 2

[2] Available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.
[3] There is no similar data available for NIST-III and NIST-V parameters.

Table 5: Table of the best SQIsign primes and their signing costs. X+B refers to the XGCD-and-boost technique, while S+B refers to sieve-and-boost. All primes satisfy $2^f T \mid (p^2-1)/2$, where T is odd and smooth. We also report the largest factor $\ell_{\max}$ that divides T. The primes found are listed by security level, sorted by the most cost-effective signing. We also compare our results to primes from [DLLW23, BCC+23].

	Prime	$\log_2(p)$	f	$\ell_{\max} \mid T$	$C_{\text{Sign},p}(T)$	Method
NIST-I	p^{I}_{1223}	251.9	67	1223	1.97	X+B
	p^{I}_{1973}	253.7	75	1973	2.01	X+B
	p^{I}_{8011}	253.6	83	8011	2.22	X+B
	p_{3923} [DLLW23]	253.7	65	3923	2.16	XGCD
	p_{479} [BCC+23]	252.2	49	479	2.80	CHM + boost
NIST-III	p^{III}_{5563}	378.3	55	5563	6.05	S+B ($n=6$)
	p^{III}_{22741}	379.1	69	22741	5.64	S+B ($n=4$)
	p^{III}_{47441}	377.9	97	47441	5.67	S+B ($n=4$)
	p^{III}_{194581}	381.4	129	194581	6.96	S+B ($n=4$)
	p_{1097} [BCC+23]	361.1	39	1097	9.52	CHM + boost
	p_{10243} [BCC+23]	381.2	79	10243	6.08	CHM + boost
NIST-V	p^{V}_{40609}	504.2	73	40609	12.78	S+B ($n=6$)
	p^{V}_{66343}	505.7	91	66343	12.12	S+B ($n=6$)
	p^{V}_{141079}	499.9	115	141079	11.50	S+B ($n=6$)
	p^{V}_{318233}	501.2	145	318233	12.40	S+B ($n=6$)
	p_{150151} [BCC+23]	507.8	85	150151	17.05	CHM + boost

factors ℓ_j, and $M=6$, resulting in a search space of size $2^{35.1}$ per search configuration.[4] Due to the sampling of small primes, this approach is probabilistic. Given the relatively small search spaces combined with the high number of trials conducted, we estimate that we have exhaustively covered the entire search space for the specified parameters.

The choice of B can be motivated by Table 3. For example, when setting $n=2$ and $f_n=32$, finding B-twin smooth integers $(r, r\pm 1)$ with $B=2^{11}$ has a probability of $2^{-29.7}$. The probability of $p_n(r)$ to be prime can be estimated through the prime number theorem, so with the search space of size $2^{36.1}$ per search configuration, we can expect to find enough twin smooths to produce a SQIsign-friendly prime. Note that this is only a rough estimate, since $(r, r\pm 1)$ are not required to be fully smooth if $p_n(r)-1$ contains enough smooth factors not inherited from $r\pm 1$. For larger f_n, the probabilities for finding a large enough smooth T decreases, for which we account by increasing the smoothness bound B.

Best primes for SQIsign. We summarise the best primes we found for SQIsign in Table 5. As suggested by our probability analysis, XGCD-and-boost with $n=2$ is superior for finding NIST-I primes, where the unbalanced version found suitable primes. For NIST-III and NIST-V, sieve-and-boost gives the best results, using $n=4$ and $n=6$.

NIST-I. For NIST-I primes, targeting $p\approx 2^{256}$, we found several primes with odd B-smooth factor $T > p^{1.27}$ using $B=2^{11}$. For instance, the primes p^{I}_{1223} and p^{I}_{1973} feature powers of two of size $f=67$ and $f=75$. Therefore, they improve upon the the prime p_{3923} used in previous work [DLLW23] both in signing and verification time. In particular,

[4] Iterating over XGCD solutions as described in Section 4 further increases this by factor 4 on average.

we deem p^{I}_{1973} to be the most beneficial choice for efficient implementations, improving signing performance by 7% and verification cost by 7.5% over p_{3923}. The verification cost of p^{I}_{1223} only improves marginally upon p_{3923}.

In particular, p^{I}_{1973} is a 254-bit prime using

$$T = 3^{36} \cdot 7^4 \cdot 11 \cdot 13 \cdot 23^2 \cdot 37 \cdot 59^2 \cdot 89 \cdot 97 \cdot 101^2 \cdot 107 \cdot 109^2 \cdot 131 \cdot 137 \cdot 197^2 \cdot 223 \cdot 239 \cdot 383 \cdot 389 \cdot 491^2 \cdot 499 \cdot 607 \cdot 743^2 \cdot 1033 \cdot 1049 \cdot 1193 \cdot 1913^2 \cdot 1973.$$

Increasing the size of f in exchange for a larger smoothness bound B yields the prime p^{I}_{8011} with $B = 2^{13}$. The larger $f = 83$ improves verification performance by 3.5%, while signing performance is slower by 10% compared to p^{I}_{1973}, which makes practical application less appealing.

Note also that, somewhat surprisingly, the prime p_{479} found by CHM with boosting [BCC+23] has a much smaller smoothness bound $B = 479$ of T, but results in slower performance by roughly 40% for signing and 27% for verifying compared to p^{I}_{1973}, due to the relatively small size of f.

NIST-III. For the NIST-III security level, targeting $p \approx 2^{384}$, the only parameters proposed prior to our work used the CHM with boosting approach [BCC+23], but no implementation was available. The most promising prime defined there is p_{10243}, featuring $f = 79$ and $B = 10243$.

Resulting from our sieve-and-boost searches, the most promising prime is p^{III}_{47441} with $f = 97$, permitting good performance for both signing and verifying. It uses

$$T = 3^{68} \cdot 5 \cdot 7^{12} \cdot 11^4 \cdot 13 \cdot 47^4 \cdot 89 \cdot 113 \cdot 157^4 \cdot 173 \cdot 233 \cdot 239 \cdot 241 \cdot 443 \cdot 509^4 \cdot 569 \cdot 761^4 \cdot 1229 \cdot 2393 \cdot 3371 \cdot 4517 \cdot 5147 \cdot 5693 \cdot 5813 \cdot 9397 \cdot 26777 \cdot 39679 \cdot 47441.$$

Despite the larger smoothness bound compared to p_{10243}, the estimated signing cost is 7% lower, while also allowing for faster verification due to the larger value of f.

Similar to p_{479} for NIST-I, p_{1097} from [BCC+23] offers a much smaller smoothness bound, at the cost of the smaller choice $f = 39$. Again, this leads to both much slower signing and verification.

NIST-V. Similar so NIST-III, the only primes for NIST-V, targetting $p \approx 2^{512}$ have been found through CHM with boosting [BCC+23]. The most promising proposal regarding overall performance is p_{150151}, featuring $f = 85$ and $B = 150151$.

Our sieve-and-boost searches produced many primes that are superior both for signing and verification. The best overall performance can be reached when using p^{V}_{318233} with the much larger $f = 145$ and

$$T = 3^{72} \cdot 5 \cdot 7 \cdot 13^6 \cdot 17 \cdot 37 \cdot 41^6 \cdot 53 \cdot 67^6 \cdot 73 \cdot 103^6 \cdot 127 \cdot 151 \cdot 461^6 \cdot 643 \cdot 733 \cdot 739 \cdot 827^6 \cdot 1009 \cdot 2539 \cdot 4153 \cdot 5059 \cdot 7127 \cdot 10597 \cdot 13591 \cdot 14923 \cdot 15541 \cdot 15991 \cdot 18583 \cdot 23227 \cdot 48187 \cdot 63247 \cdot 65521 \cdot 318233.$$

Again, the larger smoothness bound compared to p_{150151} is mitigated by the larger value of f, resulting in an estimated signing cost reduction by 27%, and much faster verification. Thus, our search methods seem to scale better to larger parameter sizes than previous approaches.

6.2 Results for AprèsSQI

We ran a large search for AprèsSQI primes at security level NIST-I, again using both sieve-and-boost and XGCD-and-boost. We ran both techniques with different smoothness

Table 6: Table of the best AprèsSQI primes and their signing costs. In the table, $\ell_{\max}$ refers to the largest prime $\ell \mid T$, while $k_{\max}$ refers to the largest extension field we have to use.

Prime	$\log_2(p)$	f	$\ell_{\max} \mid T$	$k_{\max}$	$C_{\mathrm{Sign},p}(T)$	Method
$p_{3917}^{\mathrm{APRÈS}}$	253.7	99	3917	14	1.34	S+B
$p_{2791}^{\mathrm{APRÈS}}$	249.7	107	2791	15	1.34	X+B
$p_{3527}^{\mathrm{APRÈS}}$	250.1	119	3527	13	1.43	X+B
$p_{4441}^{\mathrm{APRÈS}}$	249.1	129	4441	11	1.60	X+B
$p_{12433}^{\mathrm{APRÈS}}$	253.9	161	12433	26	2.19	S+B
p_7 [CEMR24]	253.6	145	1663	23	2.26	S+B
p_{1973}^{I}	253.7	75	1913	9	1.74	X+B

bounds over the different extensions, using $k \leq 8$ and $2^8 \leq B \leq 2^{12}$ for different extensions for the initial choice of T. Further, for sieve-and-boost, we allow larger bounds B for larger values of $f > 128$. Unlike for SQIsign-primes, both methods seem to produce good primes for AprèsSQI at NIST-I, as seen in Table 6.

Interestingly, we can also apply the AprèsSQI-techniques to the "official" SQIsign-prime p_{1973}^{I}. This allows us to replace the two most expensive factors of p_{1973}^{I}, with cheap factors over small extension fields, lowering the total cost from 2.01 to 1.74. We use this as a baseline to compare to the AprèsSQI-specific primes.

The most promising AprèsSQI-prime we found, $p_{2791}^{\mathrm{APRÈS}}$, has a signing cost of 1.34. Compared to p_{1973}^{I}, this prime thus has both significantly faster verification by 15% and signing by 23%. Alternatively, $p_{12433}^{\mathrm{APRÈS}}$ is also an interesting candidate, which has only slightly slower signing than SQIsign with p_{1973}^{I}, but which has $f = 161$, yielding a verification speed-up by 30%.

We also point out that there is little correlation between $\ell_{\max}, k_{\max}$ and the total cost. This is likely due to the fact that the total cost depends much more on the whole distribution of T, not only the upper bounds. The full information on T for the different primes can be found in Appendix A.

Remark 6. Most AprèsSQI primes listed in Table 6 strictly outperform the SQIsign NIST-I prime p_{1973}^{I} submitted by [CCD+23], in the sense that they provide both faster signing and faster verification for SQIsign according to our cost metric. Hence, it seems beneficial in general to use the AprèsSQI approach of computing T-isogenies over field extensions. However, as a caveat, we recall that the cost estimates provided in Table 6 are highly dependent on the specific cost of performing arithmetic operations within different field extensions.

6.3 Results for POKE

We searched for POKE-friendly primes for the NIST-I security level. As described above, we can aim for the smoothness bounds B to be the same as in searches for SQIsign NIST-I primes with $f = 64$. Hence, we ran sieve-and-boost and XGCD-and-boost searches with $B = 2^{11}$. Similar to SQIsign, XGCD-and-boost is superior in this setting, and we found 19 primes of size $\approx 2^{256}$ satisfying the POKE requirements. We rated these primes according to the cost of computing a single T-isogeny given by $C_{\mathrm{isog},p}(T)$. The best results are shown in Table 7, indicating whether there are prime factors of $p^2 - 1$ that exceed 2^{64}.

Table 7: Table of the best POKE primes and the cost of computing a T-isogeny. The largest prime factor that divides T is denoted as $\ell_{\max}$.

Prime	$\log_2(p)$	$\ell_{\max} \mid T$	$C_{\text{isog},p}(T)$	Prime factor $> 2^{\lambda/2}$?
p_{1951}^{POKE}	253.9	1951	115.004	✗
p_{1879}^{POKE}	251.3	1879	115.349	✓
p_{1373}^{POKE}	246.8	1373	118.092	✗
p_{1693}^{POKE}	244.2	1693	119.646	✗
p_{1487}^{POKE}	249.0	1487	119.916	✓

The most promising prime appears to be p_{1879}^{POKE}, satisfying $2^f T \mid (p^2-1)/2$ with $f = 129$ and

$$T = 3^{10} \cdot 5 \cdot 7 \cdot 11^2 \cdot 13^2 \cdot 17 \cdot 19 \cdot 43^2 \cdot 47^2 \cdot 73^2 \cdot 79 \cdot 139^2 \cdot 233^2 \cdot 263 \cdot 317^2 \cdot 383^2 \cdot 401^2 \cdot 443^2 \cdot 599 \cdot 643 \cdot 1231 \cdot 1301 \cdot 1549 \cdot 1879.$$

This cost for computing a T-isogeny is almost minimised among the primes we found, and we have $q \mid (p^2-1)/2$ with a prime $q \approx 2^{65}$.

Similarly, we could run searches for the NIST-III and NIST-V security levels, searching for primes of roughly 384 resp. 512 bit. As for SQIsign, we can expect that sieve-and-boost outperforms XGCD-and-boost for these sizes. The equivalence to searching SQIsign parameters with $f \approx 2^{\lambda/2}$ further suggests that we can find primes for the smoothness bounds $B \approx 2^{15.5}$ resp. $B \approx 2^{18}$ in these cases.

References

[AAA+24a] Marius A. Aardal, Gora Adj, Arwa Alblooshi, Diego Aranha, Isaac A. Canales-Martínez, Jorge Chávez-Saab, Décio Luiz Gazzoni Filho, Krijn Reijnders, and Francisco Rodríguez-Henríquez. Optimized SQIsign 1D verification on Intel and Cortex-M4. Personal communication, manuscript in preparation, 2024.

[AAA+24b] Marius A. Aardal, Gora Adj, Arwa Alblooshi, Diego Aranha, Isaac A. Canales-Martínez, Jorge Chávez-Saab, Décio Luiz Gazzoni Filho, Krijn Reijnders, and Francisco Rodríguez-Henríquez. Scoring primes for computing isogenies in extension fields. Available at: http://delta.cs.cinvestav.mx/~francisco/codigo.html, 2024.

[Bas24] Andrea Basso. POKE: A Framework for Efficient PKEs, Split KEMs, and OPRFs from Higher-dimensional Isogenies. Cryptology ePrint Archive, Paper 2024/624, 2024. URL: https://eprint.iacr.org/2024/624.

[BBC+21] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. CTIDH: faster constant-time CSIDH. *IACR TCHES*, 2021(4):351–387, 2021. https://tches.iacr.org/index.php/TCHES/article/view/9069. doi:10.46586/tches.v2021.i4.351-387.

[BCC+23] Giacomo Bruno, Maria Corte-Real Santos, Craig Costello, Jonathan Komada Eriksen, Michael Meyer, Michael Naehrig, and Bruno Sterner. Cryptographic smooth neighbors. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 190–221. Springer, Singapore, December 2023. doi:10.1007/978-981-99-8739-9_7.

[BD21] Jean-Claude Bajard and Sylvain Duquesne. Montgomery-friendly primes and applications to cryptography. *Journal of Cryptographic Engineering*, 11(4):399–415, November 2021. doi:10.1007/s13389-021-00260-z.

[BDD+24] Andrea Basso, Luca De Feo, Pierrick Dartois, Antonin Leroux, Luciano Maino, Giacomo Pope, Damien Robert, and Benjamin Wesolowski. SQIsign2D-West: The Fast, the Small, and the Safer. Cryptology ePrint Archive, Paper 2024/760, 2024. URL: https://eprint.iacr.org/2024/760.

[BDLS20] Daniel J Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. *Open Book Series*, 4(1):39–55, 2020. doi:10.2140/obs.2020.4.39.

[Ber04] Daniel J. Bernstein. How to find smooth parts of integers, 2004. http://cr.yp.to/papers.html#smoothparts.

[BHL+22] Jan Buzek, Junaid Hasan, Jason Liu, Michael Naehrig, and Anthony Vigil. Finding twin smooth integers by solving Pell equations. *CoRR*, abs/2211.04315, 2022. doi:10.48550/ARXIV.2211.04315.

[BS07] William D. Banks and Igor E. Shparlinski. Integers with a large smooth divisor. *Integers. Electronic Journal of Combinatorial Number Theory*, 7:A17, 11, 2007. doi:10.5281/zenodo.8281131.

[CCC+19] Daniel Cervantes-Vázquez, Mathilde Chenu, Jesús-Javier Chi-Domínguez, Luca De Feo, Francisco Rodríguez-Henríquez, and Benjamin Smith. Stronger and faster side-channel protections for CSIDH. In Peter Schwabe and Nicolas Thériault, editors, *LATINCRYPT 2019*, volume 11774 of *LNCS*, pages 173–193. Springer, Cham, October 2019. doi:10.1007/978-3-030-30530-7_9.

[CCC+24] Fabio Campos, Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Michael Meyer, Krijn Reijnders, Francisco Rodríguez-Henríquez, Peter Schwabe, and Thom Wiggers. Optimizations and practicality of high-security CSIDH. *IACR Communications in Cryptology*, 1(1), 2024. doi:10.62056/ANJBKSDJA.

[CCD+23] Jorge Chavez-Saab, Maria Corte-Real Santos, Luca De Feo, Jonathan Komada Eriksen, Basil Hess, David Kohel, Antonin Leroux, Patrick Longa, Michael Meyer, Lorenz Panny, Sikhar Patranabis, Christophe Petit, Francisco Rodríguez-Henríquez, Sina Schaeffler, and Benjamin Wesolowski. SQIsign: Algorithm specifications and supporting documentation, 2023. National Institute of Standards and Technology. URL: https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/sqisign-spec-web.pdf.

[CD23] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 423–447. Springer, Cham, April 2023. doi:10.1007/978-3-031-30589-4_15.

[CEMR24] Maria Corte-Real Santos, Jonathan Komada Eriksen, Michael Meyer, and Krijn Reijnders. AprèsSQI: Extra Fast Verification for SQIsign Using Extension-Field Signing. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 63–93. Springer, 2024. doi:10.1007/978-3-031-58716-0_3.

[Cen18] Murat Cenk. Karatsuba-like formulae and their associated techniques. *Journal of Cryptographic Engineering*, 8(3):259–269, September 2018. doi:10.1007/s13389-017-0155-8.

[CHM13] Brian Conrey, Mark Holmstrom, and Tara McLaughlin. Smooth neighbors. *Experimental Mathematics*, 22(2):195–202, 2013. doi:10.1080/10586458.2013.768483.

[CMN21] Craig Costello, Michael Meyer, and Michael Naehrig. Sieving for twin smooth integers with solutions to the prouhet-tarry-escott problem. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 272–301. Springer, Cham, October 2021. doi:10.1007/978-3-030-77870-5_10.

[Cos20] Craig Costello. B-SIDH: Supersingular isogeny Diffie-Hellman using twisted torsion. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 440–463. Springer, Cham, December 2020. doi:10.1007/978-3-030-64834-3_15.

[dB66] Nicolaas G. de Bruijn. On the number of positive integers $\leq$ x and free of prime factors $> y$, ii. *Indag. Math*, 38:239–247, 1966. doi:10.1016/S1385-7258(66)50029-4.

[DF24] Max Duparc and Tako Boris Fouotsa. Sqiprime: A dimension 2 variant of sqisignhd with non-smooth challenge isogenies. Cryptology ePrint Archive, Paper 2024/773, 2024. URL: https://eprint.iacr.org/2024/773.

[Dic30] Karl Dickman. On the frequency of numbers containing prime factors of a certain relative magnitude. *Arkiv for matematik, astronomi och fysik*, 22(10):A–10, 1930.

[DKL+20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 64–93. Springer, Cham, December 2020. doi:10.1007/978-3-030-64837-4_3.

[DLLW23] Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the deuring correspondence - towards practical and secure SQISign signatures. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 659–690. Springer, Cham, April 2023. doi:10.1007/978-3-031-30589-4_23.

[DLRW24] Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. Sqisignhd: New dimensions in cryptography. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2024. doi:10.1007/978-3-031-58716-0_1.

[EPSV24] Jonathan Komada Eriksen, Lorenz Panny, Jana Sotáková, and Mattia Veroni. Deuring for the people: Supersingular elliptic curves with prescribed endomorphism ring in general characteristic. In *LuCaNT: LMFDB, computation, and number theory*, volume 796 of *Contemporary Mathematics*, pages 339–373. Amer. Math. Soc., Providence, RI, 2024. doi:10.1090/conm/796/16008.

[Leh64] Derrick H. Lehmer. On a problem of Störmer. *Illinois Journal of Mathematics*, 8(1):57–79, 1964. doi:10.1215/ijm/1256067456.

[MMP+23] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 448–471. Springer, Cham, April 2023. doi:10.1007/978-3-031-30589-4_16.

[Mon05] Peter L. Montgomery. Five, Six, and Seven-Term Karatsuba-Like Formulae. *IEEE Trans. Computers*, 54(3):362–369, 2005. doi:10.1109/TC.2005.49.

[NIS23] NIST. Post-quantum cryptography: Digital signature schemes, 2023. Personal communication, 2023. Available at: https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

[NO24] Kohei Nakagawa and Hiroshi Onuki. SQIsign2D-East: A New Signature Scheme Using 2-dimensional Isogenies. Cryptology ePrint Archive, Paper 2024/771, 2024. URL: https://eprint.iacr.org/2024/771.

[Rob23] Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 472–503. Springer, Cham, April 2023. doi:10.1007/978-3-031-30589-4_17.

[Ste23] Bruno Sterner. Towards optimally small smoothness bounds for cryptographic-sized twin smooth integers and their isogeny-based applications. Cryptology ePrint Archive, Paper 2023/1576, 2023. URL: https://eprint.iacr.org/2023/1576.

[Stø97] Carl Størmer. Quelques théorèmes sur l'équation de Pell $x^2 - dy^2 = \pm 1$ et leurs applications. *Christiania Videnskabens Selskabs Skrifter, Math. Nat. Kl*, (2):48, 1897. doi:10.1215/ijm/1256067456.

[Vél71] Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes Rendus de l'Académie des Sciences de Paris, Séries A*, 273:238–241, 1971.

A List of primes

A.1 SQIsign

$$p^{\mathrm{I}}_{1223} = \texttt{0xea6a4dda9518e5c5d50ccdfbd97e4c49efe85e0e09039c7fffffffffffffff}$$
$$p^{\mathrm{I}}_{1973} = \texttt{0x34e29e286b95d98c33a6a86587407437252c9e49355147ffffffffffffffffff}$$
$$p^{\mathrm{I}}_{8011} = \texttt{0x31ebc32c245c72c40115748f25c4ba516cb58aaae247ffffffffffffffffffff}$$
$$p^{\mathrm{III}}_{5563} = \texttt{0x4cd95e35908847e31ac2953eb6d35610ccd37a339b81a09214ad43375dd1219f}$$
$$\texttt{9ed34f2a4ad05b5507fffffffffffff}$$
$$p^{\mathrm{III}}_{22741} = \texttt{0x851b4a8ba9ca5268304fcfea6b20d3641c5982a3e888543d00f3741c8764bdb}$$
$$\texttt{ef38bf6a1531aa1fffffffffffffffff}$$
$$p^{\mathrm{III}}_{47441} = \texttt{0x3df6eeeab0871a2c6ae604a45d10ad665bc2e0a90aeb751c722f669356ea468}$$
$$\texttt{4c6174c1fffffffffffffffffffffff}$$
$$p^{\mathrm{III}}_{194581} = \texttt{0x2a61eff6f5b99e8a6531a3dd016bce053791af1d1f4c95da3643c770c28ca9}$$
$$\texttt{e1fffffffffffffffffffffffffffff}$$
$$p^{\mathrm{V}}_{40609} = \texttt{0x1258a04d42f6b813d52a2b7a316c88f20e534878009f8262082fa9996b5bb08ed}$$
$$\texttt{14526e626a06c28e0388a0721ebd5514e072a9d10861fffffffffffffffff}$$
$$p^{\mathrm{V}}_{66343} = \texttt{0x353849c34fc94533c0d441876a7a45402ba8961efb78f3aef29a679fc7623bfcc}$$
$$\texttt{519c82a014ac7fa64b2f97fa47d154b20f99af07fffffffffffffffffffff}$$
$$p^{\mathrm{V}}_{141079} = \texttt{0xf18c5c8a0bafce13183dc4b177b859181420c5d9aa73483b708189cf1f67db4a}$$
$$\texttt{ec4488346a2cfe7fc41c079c2123aea07fffffffffffffffffffffffffff}$$
$$p^{\mathrm{V}}_{318233} = \texttt{0x255946a8869bc68c15b0036936e79202bdbe6326507d01fe3ac5904a0dea65fa}$$
$$\texttt{f0a29a781974ce994c68ada6e1fffffffffffffffffffffffffffffffffff}$$

A.2 AprèsSQI

$$p^{\mathrm{APRÈS}}_{3917} = \texttt{0x367db3d1610fd86993ad6d494796c777fecd0787fffffffffffffffffffffff}$$
$$p^{\mathrm{APRÈS}}_{2791} = \texttt{0x35115c96469bb569c2dd34314f43c0614f2c7fffffffffffffffffffffff}$$
$$p^{\mathrm{APRÈS}}_{3527} = \texttt{0x4513330eaf8964fda7d3c89a0be26929c7fffffffffffffffffffffffff}$$
$$p^{\mathrm{APRÈS}}_{4441} = \texttt{0x221664ddd26fb24cc84ae0ed4d41291fffffffffffffffffffffffffff}$$
$$p^{\mathrm{APRÈS}}_{12433} = \texttt{0x3e30773d028b14a993d5acd1fffffffffffffffffffffffffffffffff}$$

A.3 POKE

$$p^{\mathrm{POKE}}_{1951} = \texttt{0x3aff3eb7970fa4ceb7c8678eb8d6cc11fffffffffffffffffffffffffffff}$$
$$p^{\mathrm{POKE}}_{1879} = \texttt{0x99d82349ba3fd6e8bf5468b25b7a621fffffffffffffffffffffffffffff}$$
$$p^{\mathrm{POKE}}_{1373} = \texttt{0x7036364ab7f4f4dd8e06adbd7c231fffffffffffffffffffffffffffff}$$
$$p^{\mathrm{POKE}}_{1693} = \texttt{0x12128537506e5ac7cb81cbb7f737c7fffffffffffffffffffffffffffff}$$
$$p^{\mathrm{POKE}}_{1487} = \texttt{0x1faead661e4fec39df163a284d8bae1fffffffffffffffffffffffffffff}$$

Table 8: Choice of $T = \prod \ell_i^{e_i}$, and the minimal fields $\mathbb{F}_{p^{2k}}$ for the prime $p_{3917}^{\text{APRÈS}}$

k	$\ell_i^{e_i}$
1	$3^{63}, 7^3, 47, 163, 211, 233, 283^2, 397, 419, 523^2, 719^2,$ $1187, 1201, 1613, 2857, 3389, 3917$
2	$5, 373$
3	$19, 43$
5	11^2
6	13
8	17
9	37
11	23
14	29

Table 9: Choice of $T = \prod \ell_i^{e_i}$, and the minimal fields $\mathbb{F}_{p^{2k}}$ for the prime $p_{2791}^{\text{APRÈS}}$

k	$\ell_i^{e_i}$
1	$3^{14}, 5^6, 7^4, 23, 29^2, 61^2, 101, 151, 229^2, 251^2, 293, 331^2, 563^2, 587^2,$ $823, 863, 997, 1181, 1319, 1933, 2003, 2791$
2	$37, 109, 773$
3	$19, 79$
5	11
6	13
8	17
10	41
15	31

Table 10: Choice of $T = \prod \ell_i^{e_i}$, and the minimal fields $\mathbb{F}_{p^{2k}}$ for the prime $p_{3527}^{\text{APRÈS}}$

k	$\ell_i^{e_i}$
1	$3^9, 7^3, 13^2, 31^2, 43^2, 67^2, 71^4, 83^2, 163, 269^2, 349, 569^2,$ $613, 647, 673, 787, 797, 857, 1283, 1571, 3527$
2	$5, 137, 1861$
3	19
4	$17, 41, 337$
5	11
6	$61, 73$
7	29
9	37
11	23
13	53

Table 11: Choice of $T = \prod \ell_i^{e_i}$, and the minimal fields $\mathbb{F}_{p^{2k}}$ for the prime $p_{4441}^{\text{APRÈS}}$

k	$\ell_i^{e_i}$
1	$3^{17}, 5^5, 11^2, 17, 19, 31, 127^2, 293, 479^2, 521, 577, 659,$ $823^2, 857^2, 1409^2, 2339^2, 2843, 4441$
2	$13^2, 1277$
3	$7, 43, 67$
5	$101^2, 131$
6	$61, 73$
7	29
11	23
13	53
18	37

Table 12: Choice of $T = \prod \ell_i^{e_i}$, and the minimal fields $\mathbb{F}_{p^{2k}}$ for the prime $p_{12433}^{\text{APRÈS}}$

k	$\ell_i^{e_i}$
1	$3^{16}, 5, 7^6, 11, 17, 59, 79^2, 157, 193^2, 283, 419, 587, 953^2, 1301, 5657, 12433$
2	$13, 37^2, 349$
3	$19, 73, 487$
4	$89, 233, 313$
5	131
6	61
7	127
9	181
10	41
11	$23, 67$
12	97
14	29
15	31
18	109
21	43
23	47
26	53