A computational framework for principally polarized abelian varieties and applications

Maria Corte-Real Santos, Etienne Piasecki, and Benjamin Wesolowski

ENS de Lyon, CNRS, UMPA, UMR 5669, Lyon, France

Abstract. We construct a new framework for cryptographers to work with principally polarized abelian varieties (PPAVs). This framework offers a computational approach to abelian varieties agnostic to the choice of a coordinate system, culminating in the definition of an *efficient model for principally polarised abelian varieties*. We exhibit an instantiation of our framework by means of the theta model — thereby streamlining the documented capacities of the model, and extending them with new fundamental algorithms, like the computation of automorphism groups. Our framework focuses on *what can be done* with these objects, computationally, while relegating low-level considerations to the background, like the specific choice of a coordinate system (and thus the necessity to rely on Mumford's theory of theta coordinates). We illustrate the utility of our framework by proving that we can interpolate polarised isogenies in any dimension, generalizing to higher dimensions the most disruptive algorithm for elliptic curves in recent years. We prove that this interpolation offers a *universal, canonical, and compact* way to represent isogenies.

Keywords: Isogeny-based cryptography · Principally polarized abelian varieties · Efficient representations · Interpolation

1 Introduction

In recent years, the substantial investment into building large scale quantum computers has resulted in increased interest in post-quantum cryptography. Isogeny-based cryptography is a promising candidate for constructing protocols that are secure in the presence of classical and quantum adversaries. Since its inception, the field has primarily focused on isogenies between elliptic curves. Indeed, though some protocols were constructed using the higher dimensional generalisation of elliptic curves (namely, principally polarised abelian varieties, or PPAVs) [7, 14, 17], they were mostly deemed to be of limited practical interest.

This changed in 2022 when a wave of results [6, 22, 30] broke one of the most well-established isogeny-based protocols in the field at that time: SIDH/SIKE [13], a key exchange mechanism. Remarkably, these attacks brought new life to the field. At a high-level, the attacks showed how to efficiently *interpolate* isogenies:

given two elliptic curves, the degree of the connecting isogeny, and how it acts on a sufficiently large torsion basis, one can recover the connecting isogeny.

This has proven to be a powerful constructive tool as it allows us to embed an isogeny of elliptic curves of large degree into an isogeny of smoother degree between higher dimensional abelian varieties. As a consequence of this breakthrough result, we now have a new way to represent isogenies; it is the first representation that can describe non-smooth isogenies between any two elliptic curves. With impressive strides towards optimising the computation of higher dimensional isogenies [9, 11], this new representation is efficient.

Higher dimensional isogenies are now an indispensable tool in constructing isogeny-based cryptosystems. Using this new representation of isogenies, existing protocols have been dramatically improved. Most notably, the recent two-dimensional variants of SQIsign, the only isogeny-based candidate in the NIST post-quantum cryptography standardisation process, are faster, smaller, and rely on more well-understood security assumptions. New protocols, which rely on this interpolation result, have also been constructed (see, for example, [2, 3, 4, 18]).

Despite their new-found importance, higher dimensional objects remain elusive to a wider audience. Moving beyond dimension one introduces new complexities, requiring a strong background in algebraic geometry. The current state of affairs is unwelcoming, focused on low-level coordinate systems, and impractical for the design higher level algorithms.

The algorithmic theory of PPAVs is built on Mumford's theory of theta coordinates. This approach, originating from [32], has proved extremely fruitful, and perfectly suited for the development of low-level algorithms and implementations. However, its cumbersome formalism is ill-suited for any higher-level applications. Consider, for example, computing a (polarised) isogeny of degree ℓ given its kernel; this task is ubiquitous is isogeny-based crytography. In dimension 1, it is clear that this can be achieved in $O(\ell)$ using Vélu's formulas [34], or $\widetilde{O}(\sqrt{\ell})$ using $\sqrt{\text{élu}}$ [5], independent of how we represent our elliptic curves. In dimension $g > 2$, the situation is far murkier, and the cryptographer who wants to justify an isogeny computation needs to jump through a series of hoops. Algorithms for computing isogenies between PPAVs of general dimension g were developed primarily by Cosset, Lubicz, and Robert using the theory of theta functions [8, 19, 20, 21, 31, 32]. These algorithms assume that the varieties are represented in the theta model of level n. This model is crucial for the design of the algorithm, but makes its subsequent use difficult: even parsing the input and output requires heavy Mumford-machinery, and the task itself is generally only presented as a 'change-of-level' algorithm, to be paired with Mumford's isogeny theorem to obtain the desired algorithm. In addition to this, though the work by Lubicz and Robert in 2023 [21] is the current state-of-the-art and is usually the cited reference, they assume that the degree of the isogeny is coprime to n. Usually we take even level n, thus ruling out computations of even degree isogenies. One must trace back the citations to find an algorithm in [20] that work for all degrees (coprime to the characteristic of the field).

Instead of jumping through these hoops, it is tempting to use PPAVs as a black box, but the state-of-the-art is missing the appropriate interface.

Contributions. The goal of this paper is to construct a new framework for cryptographers to work in higher dimensions. We offer a computational approach to abelian varieties agnostic to the choice of a coordinate system, culminating in the definition of an *efficient model for principally polarised abelian varieties*. This definition focuses on *what can be done* with these objects, computationally, in simple yet unambiguous language, without relying on highly abstract notions like Mumford's theory of theta coordinates. We then prove that in this framework, one can generically construct algorithms for more advanced functionalities, like computing the characteristic polynomial of an endomorphism.

Furthermore, we exhibit an instantiation of our framework by demonstrating that the level-4 theta model is such an efficient representation. This is done by reframing classical results on the theta model within our framework, and by providing new algorithms expanding the documented capabilities of this model. In particular, we prove that in this theta model, one can efficiently compute the automorphism group of a PPAV.

Finally, to show the utility of our framework, we use it to prove that we can interpolate (polarised) isogenies in any dimension, generalizing to higher dimensions the most disruptive algorithm for elliptic curves in recent years [6, 22, 30]. This generalisation was suspected soon after the case of dimension 1 [33], and the general structure of the algorithm (Algorithm 6) is indeed very similar. Yet, the lack of a systematic notion of an *efficient model* for the objects at play have impeded early attempts at proving this result. It was not clear whether some of the key ingredients were available in higher dimension (in particular, the computation of automorphism groups). Thanks to our framework, the interpolation algorithm Algorithm 6 can be expressed in clear and precise language, and it applies not only to the level-4 theta model, but also to any future instantiation of our framework.

We furthermore show that isogeny interpolation provides a *universal, canonical, and compact* way to represent isogenies, in the following sense: given any efficient representation of an isogeny, one can compute interpolation data for this isogeny that is compact (i.e., of length poly-logarithmic in its degree), canonical (i.e., only depends on φ and not on how it is encoded), and this interpolation data itself constitutes an efficient representation of φ (e.g., it allows one to evaluate φ on any input).

Related work. The theta model has been the standard way to computationally manipulate abelian varieties since Robert's thesis [32]. This seminal work demonstrated how Mumford's theory of theta functions provides an explicit coordinate system for PPAVs, where points are effectively described by projective coordinates and isogenies by rational maps. Subsequent literature expanded the library of efficient algorithms for this theta model, and applications emerged,

always with one major downside which the present work aims to address: its understanding requires a mastery of advanced notions of algebraic geometry.

The most noteworthy attempt at cataloguing the capabilities of the theta model has certainly been the lecture notes [16, Section 2.1]. In particular, some of the properties of the theta model listed in [16, Theorem 2.1.1] are axioms required in our definition of an efficient model — this has proved valuable in our proof that the theta model is indeed an efficient model.

One can see [16] as a first step towards making the computational theory of PPAVs more accessible, although it is still focused on one particular coordinate system: the theta model (unlike our framework which is agnostic to the underlying machinery — the theta model is only one possible instantiation). In the same vein, the article [12] presents an alternative perspective on the theta model from a more combinatorial point of view, meant to be more accessible.

Structure of the paper. The paper is structured as follows:

- Section 2 presents some background material, including a gentle introduction to PPAVs. We emphasize that the framework developed in this article is meant to be usable without extensive background in algebraic geometry. The material introduced in Section 2 is useful to understand the semantics of the model or for actually instantiating it (e.g., with the theta model).
- In Section 3, we develop the core of the new framework: the notion of efficient models for principally polarized abelian varieties. Starting with Definition 3.1, the rest of the section presents the list of computational axioms.
- In Section 4, we prove some results that hold generically in any instantiation of the model. They help in designing more advanced algorithms (like the interpolation algorithm). In particular, we prove how to compute the characteristic polynomial of an endomorphism, or the norm of an isogeny.
- In Section 5, we introduce the theta model, and in Section 6 prove that our framework can be instantiated by this model. More precisely, we prove that the theta model satisfies all the properties of an efficient models for PPAVs, except one: the *canonical and compact form for N-isogenies*. We call such a model a *proto-efficient model*. Note that while our framework is designed to be usable without extensive background in algebraic geometry, instantiating it does require familiarity with the underlying theory; in particular, these two sections require much more technical background.
- In Section 7, we prove the isogeny interpolation theorem (Theorem 7.1): in any proto-efficient model for PPAVs, given an integer N and the images of enough points through an N-isogeny, one can recover the isogeny in polynomial time. As an application, we deduce that any proto-efficient model can be enhanced into an efficient model for PPAVs (Theorem 7.5). In particular, the theta model is an efficient model (Corollary 7.7).

Notation. Let us introduce some basic notation used throughout the article.

Rings and fields. We write $\mathbb{Z}$ for the ring of integers, and $\mathbb{F}_q$ for the finite field with q elements. Throughout, p denotes the characteristic of the finite field $\mathbb{F}_q$ (i.e., q is a power of the prime number p), and we write $\overline{\mathbb{F}}_p$ its algebraic closure.

Strings. We write $\{0,1\}^*$ for the set of binary strings, and $\mathtt{1}^n = 1 \dots 1$ the binary string of length n. We use the `typewriter` font family to denote strings, with the implicit assumption that these are unambiguously encoded as binary strings. Given two strings x and y, we write $x \,|||\, y$ for their concatenation, separated by a special character (i.e., $\mathtt{xx} \,|||\, \mathtt{x}$ is distinct from $\mathtt{x} \,|||\, \mathtt{xx}$).

Asymptotic notation. We write $f = O(g)$ for the classic big O notation, and use the polynomial growth notation $\mathrm{poly}(f_1, \dots, f_n) = (f_1 + \cdots + f_n)^{O(1)}$.

Algorithms. We will define algorithms that have several subroutines. In such cases, we might say that $\mathscr{A}$ is an algorithm, and write $\mathscr{A}.\mathtt{xx}$ for its subroutine `xx`. Formally, the string `xx` can be interpreted as a first input to $\mathscr{A}$ which dictates what the algorithm is going to do, i.e., $\mathscr{A}.\mathtt{xx}(-) := \mathscr{A}(\mathtt{xx}, -)$. In particular, when we say that $\mathscr{A}$ is a probabilistic polynomial time algorithm, all of its subroutines $\mathscr{A}.\mathtt{xx}$ are probabilistic polynomial time algorithms (as a function of the length of the input).

Acknowledgements. This work was supported by the European Research Council under grant No. 101116169 (AGATHA CRYPTY), and by the PEPR quantique France 2030 programme (ANR-22-PETQ-0008).

2 Background

The aim of this section is to give a brief introduction to abelian varieties in any dimension, focusing on the aspects that are relevant to our work. We emphasize that the goal of this article is to give the reader the tools to work with abelian varieties for cryptography without needing an extensive background in algebraic geometry. As such, we do not claim to give a full treatment of the theory of abelian varieties; for a more in-depth introduction we refer a reader to [24]. In fact, our computational framework introduced in Section 3 can be used without even understanding much of the algebraic geometry discussed in the present section. The present section is useful to understand the *semantics* of the model (what are binary strings meant to represent?), or for actually instantiating the framework (as in Section 6 where we prove that the theta model is an efficient model).

2.1 The geometry of abelian varieties

An *abelian variety* X of dimension $\dim(X) = g$ is a smooth projective algebraic variety of dimension g which is also an algebraic group, namely it has a group law $+ : X \times X \to X$, such that $+$ and its inverse $-$ are morphisms of varieties. We denote by 0_X the identity element of X. We will only consider abelian varieties defined over $\overline{\mathbb{F}}_p$.

Key to isogeny-based cryptography is the notion of an isogeny between abelian varieties. Luckily in this case the definition of an isogeny between elliptic curves generalises naturally to abelian varieties of any dimension. Namely, a morphism $\varphi : X \to Y$ is an *isogeny of abelian varieties* if it is surjective and has finite kernel.[1] We say that X and Y are *isogenous* if there exists an isogeny between them. For almost all isogenies we consider in this paper,[2] the *degree* $\deg(\varphi)$ of φ is given precisely by the size of this kernel. Notably, this is the case for isogenies of degree coprime to p.

We can also consider (Cartesian) product abelian varieties $X_1 \times X_2$. These naturally come equipped with projection maps $\mathsf{pr}_i : X_1 \times X_2 \to X_i$ and inclusions maps $\mathsf{inc}_i : X_i \to X_1 \times X_2$. For morphisms $\varphi : X_1 \times X_2 \to Y_1 \times Y_2$, we define the *matrix representation* of φ to be $\begin{pmatrix} f_1 & f_2 \\ f_3 & f_4 \end{pmatrix}$, where

$$\varphi : P \mapsto \mathsf{inc}_1^Y[f_1(\mathsf{pr}_1^X(P)) + f_3(\mathsf{pr}_2^X(P))] + \mathsf{inc}_2^Y[f_2(\mathsf{pr}_1^X(P)) + f_4(\mathsf{pr}_2^X(P))].$$

2.1.1 Endomorphism rings. Let X and Y be two abelian varieties, we define by $\mathrm{Hom}(X, Y)$ the set of morphisms of abelian varieties between X and Y (not necessarily isogenies). Define the *endomorphism ring* $\mathrm{End}(X)$ of X to be $\mathrm{Hom}(X, X)$. An important invariant of endormorphism is their characteristic polynomial.

Definition 2.1. *Let $\alpha \in \mathrm{End}(X)$. Let $F_\alpha \in \mathbb{Z}[x]$ be the unique monic polynomial of degree $2g$ such that $F_\alpha(N) = \deg(\alpha - N)$ for all integers N. We call F_α the* characteristic polynomial *of α. Writing $F_\alpha(x) = x^{2g} + c_{2g-1}x^{2g-1} + \cdots + c_0$, we have that c_0 is the* degree $\deg(\alpha)$ *of α, and $-c_{2g-1}$ is the* trace $\mathrm{Tr}(\alpha)$ *of α.*

Remark 2.2. The trace form of an endomorphism is linear and $\mathrm{Tr}([1]) = 2g$.

Example 2.3. Common examples of endomorphisms are: the multiplication-by-N map for $N \in \mathbb{Z}$, which has degree N^{2g} [24, Theorem 7.2] for $N \neq p$; the Frobenius endomorphism π; and the Verschiebung $\widehat{\pi}$. Note that $\pi \circ \widehat{\pi} = [p]$.

2.2 Principally polarised abelian varieties

In cryptography, we often take for granted that we have equations defining our elliptic curves in projective space. For instance, every elliptic curve can be represented by a Weierstrass equation. In contrast, for higher dimensional abelian varieties, we no longer have nice models to work with in general. Hidden under the surface is the fact that the existence of equations for elliptic curves stems from the existence of a canonical *principal polarisation*. This fails in higher dimension: some abelian varieties have no principal polarisation, and others have more than one (so we need to be careful with our choice). To alleviate this issue, we consider only *principally polarised abelian varieties*, a notion which we explain in this section.

[1] The finiteness of the kernel ensures that $\dim(X) = \dim(Y)$.
[2] Namely, separable isogenies.

Divisors. The first step towards defining a principal polarisation is to define *divisors*. The reader may recall this notion from the case of elliptic curves, where a divisor is a formal sum of points on the curve. To generalise this notion in higher dimension, we view divisors as formal sums of codimension one subvarieties.

More formally, we define a *prime divisor* on an abelian variety X to be an irreducible subvariety Z of X of codimension one. A *divisor* on X is then given by a linear combination of prime divisors: $D = \sum n_Z[Z] \in \mathrm{Div}(X)$. If $n_Z \geq 0$ for all Z, we say that D is an *effective divisor*.

For f a regular map, we define its divisor $\mathrm{div}(f)$ to be a formal sum $\sum \mathrm{ord}_Z(f)Z$ of prime divisors, where $\mathrm{ord}_Z : \mathbb{F}_q(X)^\times \to \mathbb{Z}$ is the order of vanishing of f along Z. Note that $\mathrm{div}(f)$ is well-defined since $\mathrm{ord}_Z(f) = 0$ for all but finitely many Z. A divisor of the form $\mathrm{div}(f)$ is said to be *principal*. Two divisors are said to be *linearly equivalent* if they differ by a principal divisor. For any abelian variety X, we define the *Picard group* $\mathrm{Pic}(X)$ to be the group of divisors up to this linear equivalence. The subgroup $\mathrm{Pic}^0(X)$ consists of the equivalence classes of divisors D such that $\tau_Q^*(D) \simeq D$ for any $Q \in X$, with τ_Q being the translation map by a point Q.

Embedding into projective space. Let D an effective divisor on an abelian variety X. The space of regular functions on X associated to D is defined as:

$$L(D) := \{f \in \mathbb{F}_q(X)^\times \mid \mathrm{div}(f) + D \geq 0\} \cup \{0\}.$$

A choice of basis $(f_0, \dots f_m)$ of $L(D)$ defines a rational map $\iota_D : X \to \mathbb{P}^m$, given by $P \mapsto [f_0(P) : \cdots : f_m(P)]$. A divisor D is said to be *very ample* if the corresponding map ι_D is an embedding. We say that a divisor D is *ample* if there exists $n > 0$ such that nD is very ample. Mumford [25, Theorem p. 163] showed that for any ample divisor D on an abelian variety, and any $n \geq 3$, the divisor nD is very ample.

The dual variety. Let X an abelian variety. The dual variety $\hat{X}$ is an abelian variety such that its points are $\mathrm{Pic}^0(X)$ (see [24] for a detailed construction). Associated to dual varieties, are dual morphisms: for a morphism $\varphi : X \to Y$, there exists a dual morphism $\hat{\varphi} : \hat{Y} \to \hat{X}$ defined as $D \mapsto \varphi^*(D)$. If φ is an isogeny, then $\hat{\varphi}$ is also an isogeny, and we have $\ker(\hat{\varphi}) = \widehat{\ker(\varphi)}$ [24, Theorem 9.1]. This gives a functorial property to the dual.

Consider the map $\lambda__ : \mathrm{Pic}(X) \to \mathrm{Hom}(X, \hat{X})$ given by $D \mapsto (\lambda_D : Q \mapsto \tau_Q^* D - D)$. It is a group morphism with kernel $\mathrm{Pic}^0(X)$, therefore inducing an embedding $\mathrm{Pic}(X)/\mathrm{Pic}^0(X) \hookrightarrow \mathrm{Hom}(X, \hat{X})$. The quotient $\mathrm{Pic}(X)/\mathrm{Pic}^0(X)$ is called the *Néron-Severi group*. We say that two divisors D_1 and D_2 are *algebraically equivalent* if they define the same element in the Néron-Severi group.

We remark that the map λ_D is not necessarily an isogeny. It will be if its kernel is finite, i.e., if D is an ample divisor.

Polarisations. In the context of elliptic curves, we usually define the dual of an isogeny $\varphi : E \to E'$ to be an isogeny $E' \to E$. Indeed, an elliptic curve E has canonical isomorphism $E \cong \hat{E}$. Such an isomorphism is called a *principal polarisation*. Polarisations give us a way to connect an abelian variety to its dual, allowing us to recover such a 'dual' isogeny.

Let X an abelian variety. An ample divisor D (defined over $\overline{\mathbb{F}}_p$) gives rise to an isogeny $\lambda_D : X \to \hat{X}$ given by $Q \mapsto \tau_Q^*(D) - D$. If λ_D is an isomorphism, we say it is a *principal polarisation*. An ample divisor is said to be *separable* if the corresponding polarisation is a separable isogeny. In this article, we will only consider separable ample divisors.

Remark 2.4. Two algebraically equivalent divisors (i.e., the same element in the Néron-Severi group) lead to the same polarisation. When working with polarisations, we often implictly fix a representative in the algebraic class.

Remark 2.5. Note that if λ_D is a principal polarisation, then $\ker(\lambda_D) = \{\mathcal{O}\}$ and for every $n \in \mathbb{N}^*$, $\ker(\lambda_{nD}) = X[n]$.

If X is an abelian variety and $\lambda_X : X \to \hat{X}$ a principal polarisation, we call the pair (X, λ_X) a *principally polarised abelian variety* (PPAV). Crucially, the embedding $\iota_D : X \hookrightarrow \mathbb{P}^m$ allows us to work with X as a projective variety. We often abuse notation and refer to X as a PPAV when the polarisation is clear by context.

We can now define the higher dimensional analogue of the 'dual' isogeny. Let (X, λ_X) and (Y, λ_Y) be PPAVs and $\varphi : X \to Y$ a morphism. We have a map $\mathrm{Hom}(X, Y) \to \mathrm{Hom}(Y, X)$ given by $\varphi \mapsto \varphi^\dagger := \lambda_X^{-1} \circ \hat{\varphi} \circ \lambda_Y$. We call $\varphi^\dagger$ the *adjoint isogeny*. Moreover, if $(X, \lambda_X) = (Y, \lambda_Y)$, we obtain an involution on $\mathrm{End}(X)$ called the *Rosatti involution*.

For $\varphi \in \mathrm{Hom}(X, Y)$, we note that $\varphi^\dagger \circ \varphi \in \mathrm{End}(X)$. One can therefore consider its characteristic polynomial and, in particular, its trace. This leads us to the definition of the *norm* of a morphism, which can be thought of as a Euclidean norm on $\mathrm{Hom}(X, Y)$.

Definition 2.6 (Norm). *Let (X, λ_X) and (Y, λ_Y) be two PPAV and $\varphi : X \to Y$ a morphism of abelian varieties. We define the* norm *of φ as $\|\varphi\| = \sqrt{\mathrm{Tr}(\varphi^\dagger \circ \varphi)}$.*

The Weil pairing. Let m be an integer prime to p. The *Weil pairing* for an abelian variety X is a non-degenerate bilinear map $e_m : X[m] \times \hat{X}[m] \to \mu_m$, where μ_m is the m-th roots of unity (see [24, §13] for a full construction). If X is principally polarised, we can identify X and $\hat{X}$ via λ_X, and obtain the pairing

$$\begin{aligned} e_m^\lambda : X[m] \times X[m] &\to \mu_m \\ (P, Q) &\mapsto e_m(P, \lambda_X(Q)) \end{aligned}$$

When the principal polarisation is clear by context, we will simply denote the Weil pairing by e_m.

Polarized isogenies. For a positive integer N, we define an *N-isogeny* between PPAVs X and Y to be an isogeny φ such that $\varphi^\dagger \circ \varphi = \varphi \circ \varphi^\dagger = [N]$. In particular, we have $\|\varphi\| = \sqrt{2gN}$ and $\deg(\varphi) = N^g$. Furthermore, φ is a *polarized isogeny*, in the sense that $\varphi^\dagger = \hat{\varphi} \circ \lambda_Y \circ \varphi$. We have the following nice characterisation of N-isogenies.

Definition 2.7. *Let X, Y be PPAVs and $\varphi : X \to Y$ an isogeny. We say that a subgroup G of $X[N]$ is* isotropic with respect to the N-Weil pairing *if $e_N(P, Q) = 1$ for all $P, Q \in G$, and* maximal isotropic *if it is not properly contained in any other isotropic subgroup.*

Proposition 2.8 (e.g., [15, Proposition 1.1]). *Let $\varphi : X \to Y$ be an isogeny. There exists a principal polarization on Y such that φ is an N-isogeny if and only if $\ker(\varphi)$ is a maximal isotropic subgroup of $X[N]$ with respect to the N-Weil pairing.*

3 Efficient models for principally polarized abelian varieties

In this section, we introduce the notion of an *efficient model* for PPAVs. We wish to represent PPAVs, points on PPAVs, and morphisms between PPAVs as binary strings. Such strings correspond to sets $S_{\text{ppav}}, S_{\text{pt}}, S_{\text{mrph}} \subset \{0,1\}^*$. The notation $\underline{X}, \underline{P}$ or $\underline{\varphi}$ will typically be used to denote a binary encoding of some underlying PPAV X, point P or morphism φ.

We then ask for algorithms to manipulate these encodings. For instance, an algorithm $\mathscr{A}.\texttt{isomorphic}$ would take as input the encodings $\underline{X}, \underline{X'} \in S_{\text{ppav}}$ of two PPAVs, and decide whether or not they are isomorphic. Another algorithm $\mathscr{A}.\texttt{torsion}$ would take as input the encoding $\underline{X} \in S_{\text{ppav}}$ of a PPAV and an integer, and return the encodings $\underline{P_i} \in S_{\text{pt}}$ of the N-torsion points of the variety. With appropriate bounds on the length of encodings and on the complexity of the family of algorithms $\mathscr{A}$, we say that $\mathscr{A}$ is an *efficient model for PPAVs*, and the elements of $S_{\text{ppav}}, S_{\text{pt}}$ or S_{mrph} are called *efficient ($\mathscr{A}$-)representations* of the underlying objects.

While the intuitive idea is rather simple, making it precise requires a level of formalism that can rapidly become cumbersome. In applications of this framework, one would typically conflate a PPAV X with its binary representation $\underline{X}$. In contrast, to ensure that we build on solid ground, the present section carefully distinguishes objects from their binary representation.

Definition 3.1 (Efficient model for PPAVs). *Let $\mathscr{A}$ be a (probabilistic) algorithm. Let $S_{\text{ppav}} \subset \{0,1\}^*$ be a set of binary strings, and let* ppav *be a map* $\text{ppav} : S_{\text{ppav}} \longrightarrow \{(q, X) \mid X \text{ is a PPAV over } \mathbb{F}_q\}$. *For any $\underline{X} \in S_{\text{ppav}}$ with $\text{ppav}(\underline{X}) = (q, X)$, let $S_{\text{point}}(\underline{X}) \subset \{0,1\}^*$ be a set of binary strings, and let $\text{pt}_{\underline{X}}$ be a map $\text{pt}_{\underline{X}} : S_{\text{point}}(\underline{X}) \to X(\overline{\mathbb{F}}_p)$. Let $S_{\text{mrph}} \subset \{0,1\}^*$ be a set of binary strings, and let* mrph *be a (not necessarily surjective) morphism* $\text{mrph} : S_{\text{mrph}} \longrightarrow \{\varphi \mid \varphi \text{ is a morphism between two PPAVs}\}$. *The tuple*

$(\mathscr{A}, \mathrm{ppav}, (\mathrm{pt}_{\underline{X}})_{\underline{X}}, \mathrm{mrph})$ *is a* proto-efficient model for PPAVs *if it satisfies the axioms A1, ..., A28. It is an* efficient model for PPAVs *if it also satisfies the axiom A29.*

The elements of $S_{\mathrm{ppav}}, S_{\mathrm{point}}(\underline{X})$ *and* S_{mrph} *are encodings of PPAVs, points, and morphisms respectively. More precisely,*

- *if* $\mathrm{ppav}(\underline{X}) = (q, X)$*, we say that the binary string* $\underline{X}$ *is an* $\mathscr{A}$-representation *of the PPAV* X *(which is defined over* $\mathbb{F}_q$*),*
- *if* $\mathrm{pt}_{\underline{X}}(\underline{P}) = P$*, we say that the binary string* $\underline{P}$ *is an* $\mathscr{A}$-representation *of the point* $P \in X$*, and*
- *if* $\mathrm{mrph}(\underline{\varphi}) = \varphi$*, we say that the binary string* $\underline{\varphi}$ *is an* $\mathscr{A}$-representation *of the morphism* φ*.*

Remark 3.2. The maps ppav, $\mathrm{pt}_{\underline{X}}$ and mrph are "semantic" maps: they are not algorithms, they only link binary strings to their abstract interpretation. The piece of data $\underline{z} \in \{0,1\}^*$ encodes the abstract concept z. Note that this encoding is not necessarily unique.

3.1 Axioms on efficiency and encodings

In the one dimensional case, elliptic curves are typically represented by some equation like the Weierstrass equation. The coefficients of the equation constitute an encoding of the curve. We then have efficient algorithms to, for instance, decide if two such equations define isomorphic elliptic curves. A point on a curve is a solution of the equation, and its coordinates constitute an encoding of the point. This encoding needs not be unique (for instance when using projective coordinates), but can always be normalized to a "canonical" form. The following definition captures these ideas and generalizes them to PPAVs.

A1. Efficiency. The algorithm $\mathscr{A}$ (i.e., each of its subroutines $\mathscr{A}.\mathtt{x}$) has expected complexity at most $\mathrm{length}(\mathtt{input})^{\mathrm{poly}(\dim(\mathtt{input}))}$.

> **Note:** Here, $\dim(\mathtt{input})$ denotes the maximum of the dimension of any of the data given as input. The dimension of $\underline{X}$ is $\dim(X)$, the dimension of $\underline{P}$ is the dimension of the PPAV containing P, the dimension of $\underline{\varphi}$ is the maximum of the dimension of its domain and codomain, and the dimension of any other piece of data is 0.
>
> **Note:** In particular, when restricting to a fixed dimension, the model runs in polynomial time.
>
> This axiom also imposes a limit on the scaling with respect to the dimension, although it is quite loose. In the theta model, most subroutines achieve the much stronger bound $\mathrm{poly}\left(\mathrm{length}(\mathtt{input}), 2^{\dim(\mathtt{input})}\right) = \mathrm{poly}(\mathrm{length}(\mathtt{input}))$ (the last equality following from the fact that in the theta model, the length of an encoding is of the order of $2^{\dim(\mathtt{input})}$).

A2. Type testing. For any $\underline{z} \in \{0,1\}^*$, we have that

$$\mathscr{A}.\texttt{type}(\underline{z}) \text{ outputs } \begin{cases} \texttt{ppav} & \text{if } \underline{z} \in S_{\mathrm{ppav}} \\ \texttt{point} & \text{if } \exists \underline{X} \in S_{\mathrm{ppav}} \text{ such that } \underline{z} \in S_{\mathrm{point}}(\underline{X}) \\ \texttt{morphism} & \text{if } \underline{z} \in S_{\mathrm{mrph}} \\ \bot & \text{otherwise.} \end{cases}$$

A3. Geometric surjectivity. For any PPAV $X/\mathbb{F}_q$, there exists an integer $k = 2^{\mathrm{poly}(\dim(X))}$ and $\underline{X'} \in S_{\mathrm{ppav}}$ such that $X \cong_{\overline{\mathbb{F}}_p} X'$, where $\mathrm{ppav}(\underline{X'}) = (q^k, X')$.

> **Note:** In other words, the model can represent any PPAV, up to isomorphism over the algebraic closure (and the base field of the model is not much larger than $\mathbb{F}_q$). This axiom could be strengthened to a "rational surjectivity", ensuring that every X over $\mathbb{F}_q$ is represented up $\mathbb{F}_q$-isomorphism, but we leave such rationality questions aside.

A4. Compacity of objects. For any $\underline{X} \in S_{\mathrm{ppav}}$, we have that $\mathrm{length}(\underline{X}) = (\log q)^{\mathrm{poly}(\dim(X))}$, where $\mathrm{ppav}(\underline{X}) = (q, X)$.

> **Note:** The theta model achieves the stronger bound $\mathrm{length}(\underline{X}) = \mathrm{poly}\left(\log q, 2^{\dim(X)}\right)$.

A5. Compacity of points. For any $\underline{P} \in S_{\mathrm{point}}(\underline{X})$, we have $\mathrm{length}(\underline{P}) = \mathrm{poly}(\mathrm{length}(\underline{X}), k)$ where k is the smallest integer such that $\mathrm{pt}_{\underline{X}}(\underline{P}) \in X(\mathbb{F}_{q^k})$.

A6. Canonical form for PPAVs. $\mathscr{A}.\texttt{canonical}(\underline{X})$ outputs an encoding $\underline{X_{\texttt{can}}} \in S_{\mathrm{ppav}}$ such that $\mathrm{ppav}(\underline{X}) = \mathrm{ppav}(\underline{X_{\texttt{can}}})$. Furthermore, for any $\underline{Y} \in S_{\mathrm{ppav}}$, we have $\mathrm{ppav}(\underline{X}) = \mathrm{ppav}(\underline{Y})$ if and only if $\mathscr{A}.\texttt{canonical}(\underline{Y}) = \underline{X_{\texttt{can}}}$.

> **Note:** The existence of this algorithm implies an equality testing algorithm $\mathscr{A}.\texttt{equal}(\underline{X}, \underline{X'})$ to decide whether $X = X'$.

A7. Canonical form for isomorphism classes. $\mathscr{A}.\texttt{canon_isom}(\underline{X})$ outputs an encoding $\underline{X_{\texttt{can}}} \in S_{\mathrm{ppav}}$ such that $\mathrm{ppav}(\underline{X}) \cong_{\overline{\mathbb{F}}_p} \mathrm{ppav}(\underline{X_{\texttt{can}}})$. Then for any $\underline{Y} \in S_{\mathrm{ppav}}$, we have $\mathrm{ppav}(\underline{X}) \cong_{\overline{\mathbb{F}}_p} \mathrm{ppav}(\underline{Y})$ if and only if $\mathscr{A}.\texttt{canon_isom}(\underline{Y}) = \underline{X_{\texttt{can}}}$.

> **Note:** The existence of this algorithm implies an isomorphism testing algorithm $\mathscr{A}.\texttt{isomorphic}(\underline{X}, \underline{X'})$ to decide whether $X \cong X'$.

A8. Canonical form for points. $\mathscr{A}.\texttt{canonical}(\underline{P})$ outputs $\underline{P_{\texttt{can}}} \in S_{\mathrm{point}}(\underline{X})$ such that $\mathrm{pt}_{\underline{X}}(\underline{P}) = \mathrm{pt}_{\underline{X}}(\underline{P_{\texttt{can}}})$. Furthermore, for any $\underline{Q} \in S_{\mathrm{point}}(\underline{X})$, we have $\mathrm{pt}_{\underline{X}}(\underline{P}) = \mathrm{pt}_{\underline{X}}(\underline{Q})$ if and only if $\mathscr{A}.\texttt{canonical}(\underline{Q}) = \underline{P_{\texttt{can}}}$.

> **Note:** The existence of this algorithm implies an equality testing algorithm $\mathscr{A}.\texttt{equal}(\underline{P}, \underline{Q})$ to decide whether $P = Q$.

Remark 3.3. To lighten the notation, and since the following axioms are not as encoding-focused as the former ones, we will start using implicitly that $\underline{z}$ represents the object z (with notations like X or Y for PPAVs, P or Q for points, and φ or ψ for morphisms). The maps ppav, $\mathrm{pt}_{\underline{X}}$ and mrph will be used sparingly, when their explicit use seems helpful.

3.2 Axioms for extracting metadata

The following axioms ensure that basic context can be extracted from the encodings, like the field of definition, or the domain and codomain of a morphism.

A9. Field of definition. $\mathscr{A}.\texttt{field}(\underline{z})$ outputs the field of definition $\mathbb{F}_q$ of the PPAV, point or morphism represented by $\underline{z}$.

A10. Dimension. For any $\underline{X} \in S_{\text{ppav}}$, $\mathscr{A}.\texttt{dimension}(\underline{X})$ outputs $\dim(X)$.

A11. Parent. For any $\underline{P} \in \{0,1\}^*$, $\mathscr{A}.\texttt{parent}(\underline{P})$ outputs $\underline{X}$ such that $\underline{P} \in S_{\text{point}}(\underline{X})$ (i.e., it represents a point $P \in X(\overline{\mathbb{F}}_p)$) if it exists, and $\perp$ otherwise.

A12. Domain and codomain. $\mathscr{A}.\texttt{domcodom}(\underline{\varphi})$ outputs $\mathscr{A}$-representations $\underline{X}$ and $\underline{Y}$ of the domain and codomain of $\varphi : X \to Y$.

A13. Bound on norm. Given $\underline{\varphi} \in S_{\text{mrph}}$, $\mathscr{A}.\texttt{bound_norm}(\underline{\varphi})$ outputs $B \in \mathbb{Z}$ such that $\|\varphi\| \leq B$.

> **Note:** We will prove that, thanks to this axiom, the exact norm $\|\varphi\|$ can be computed efficiently, see Corollary 4.11.

3.3 Axioms for point-arithmetic

The following axioms get us closer the nature of PPAVs as arithmetic-geometric objects: they are, in a sense, a structured collection of points, and we want to be able to compute with them. For instance, given two points P, Q on a PPAV, they can be added to form a point $P + Q$, and we want that to be efficiently computable. The following definition captures this idea.

A14. Zero. $\mathscr{A}.\texttt{zero}(\underline{X})$ outputs $\underline{P} \in S_{\text{point}}(\underline{X})$ such that $\text{pt}_{\underline{X}}(\underline{P}) = 0_X$.

A15. Group law. Given $\underline{P}, \underline{Q} \in S_{\text{point}}(\underline{X})$, $\mathscr{A}.\texttt{diff}(\underline{P}, \underline{Q})$ outputs $\underline{R} \in S_{\text{point}}(\underline{X})$ such that $\text{pt}_{\underline{X}}(\underline{R}) = P - Q$.

> **Note:** The existence of this algorithm to compute $P - Q$ is sufficient to construct a polynomial time algorithm $\mathscr{A}.\texttt{add}(\underline{P}, \underline{Q})$ to compute $P + Q$, since $P + Q = P - (0 - Q)$. It also allows one to construct a polynomial time algorithm $\mathscr{A}.\texttt{multiply}(n, \underline{P})$ to compute the scalar multiplication $[n]P$ (with $n \in \mathbb{Z}$), thanks to the double-and-add algorithm.

A16. Weil pairing. Let $\underline{P}, \underline{Q} \in S_{\text{point}}(\underline{X})$ and e_N be the N-Weil pairing. Then, $\mathscr{A}.\texttt{weil}(\underline{X}, N, \underline{P}, \underline{Q})$ outputs $e_N(P, Q)$ if $P, Q \in X[N]$, and $\perp$ otherwise.

A17. Torsion. For $N \in \mathbb{Z}_{>0}$, $\mathscr{A}.\texttt{torsion}(\underline{X}, 1^N)$ outputs $\underline{P_1}, \ldots, \underline{P_{N^{2g}}} \in S_{\text{point}}(\underline{X})$ such that $\{P_1, \ldots, P_{N^{2g}}\} = X[N]$ (with $g = \dim(X)$).

> **Note:** The input 1^N serves two purposes: it encodes N, and allows the complexity to scale as $\text{length}\left(1^N\right)^{\text{poly}(g)} = N^{\text{poly}(g)}$. This dependence is necessary since the output length itself scales as $N^{\text{poly}(g)}$.
>
> **Note:** This axiom implies that the maps $\text{pt}_{\underline{X}}$ are surjective.

3.4 Axioms for manipulating morphisms

The following axioms are concerned with the manipulation of morphisms between PPAVs. We follow the philosophy of *efficient representation of isogenies*, which has been fruitful in dimension 1: the core property of a representation $\underline{\varphi}$ is that it allows one to evaluate the underlying morphism $\varphi = \mathrm{mrph}(\underline{\varphi})$ on any input point.

A18. Evaluate. For $\underline{P} \in S_{\mathrm{point}}(\underline{X})$, $\mathscr{A}.\texttt{eval}(\underline{\varphi}, \underline{P})$ outputs $\underline{P'}$ encoding $\varphi(P)$.

A19. Composition. Given $\underline{\varphi}, \underline{\varphi'} \in S_{\mathrm{mrph}}$ encoding two morphisms $\varphi : X \to Y$ and $\varphi' : Y \to Z$, $\mathscr{A}.\texttt{composition}(\underline{\varphi}, \underline{\varphi'})$ outputs $\underline{\varphi''} \in S_{\mathrm{mrph}}$ such that $\mathrm{mrph}(\underline{\varphi''}) = \varphi' \circ \varphi$. Furthermore, the output length is bounded by $\mathrm{length}(\underline{\varphi}) + \mathrm{length}(\underline{\varphi'}) + O(1)$.

> **Note:** The constraint on output length is here to avoid an explosion of the length when iterating compositions. Imagine if instead we only had, for instance, $\mathrm{length}(\underline{\varphi' \circ \varphi}) \leq 2(\mathrm{length}(\underline{\varphi}) + \mathrm{length}(\underline{\varphi'}))$. Then, applying n successive compositions, we would get $\mathrm{length}(\underline{\varphi_n \circ \cdots \circ \varphi_1}) \leq 2^n \sum \mathrm{length}(\underline{\varphi_i})$, so we could see an exponential growth in n. This exponential growth could be avoided by simply storing the concatenation $(\underline{\varphi_n}, \ldots, \underline{\varphi_1})$. With the constraint $\mathrm{length}(\underline{\varphi' \circ \varphi}) \leq \mathrm{length}(\underline{\varphi}) + \mathrm{length}(\underline{\varphi'}) + O(1)$, we have the much nicer $\mathrm{length}(\underline{\varphi_n \circ \cdots \circ \varphi_1}) \leq \sum \mathrm{length}(\underline{\varphi_i}) + O(n)$.

A20. Addition. Given $\underline{\varphi}, \underline{\varphi'} \in S_{\mathrm{mrph}}$ encoding two morphisms $\varphi, \varphi' : X \to Y$, $\mathscr{A}.\texttt{addition}(\underline{\varphi}, \underline{\varphi'})$ outputs $\underline{\varphi''} \in S_{\mathrm{mrph}}$ such that $\mathrm{mrph}(\underline{\varphi''}) = \varphi' + \varphi$. Furthermore, the output length is bounded by $\mathrm{length}(\underline{\varphi}) + \mathrm{length}(\underline{\varphi'}) + O(1)$.

A21. Adjoint. Given $\underline{\varphi} \in S_{\mathrm{mrph}}$ encoding $\varphi : X \to Y$, $\mathscr{A}.\texttt{adjoint}(\underline{\varphi})$ output $\underline{\varphi'} \in S_{\mathrm{mrph}}$ such that $\mathrm{mrph}(\underline{\varphi'}) = \varphi^\dagger := \lambda_X^{-1} \circ \varphi \circ \lambda_Y^{-1}$.

3.5 Axioms for creating morphisms

We now present axioms allowing one to create morphisms between PPAVs.

A22. Isomorphism. for any $\underline{X}$ and $\underline{Y}$, $\mathscr{A}.\texttt{isomorphism}(\underline{X}, \underline{Y})$ outputs $\underline{\varphi} \in S_{\mathrm{mrph}}$ such that $\mathrm{mrph}(\underline{\varphi}) = \varphi : X \to Y$ is an isomorphism over $\overline{\mathbb{F}}_p$ if it exists, and $\perp$ otherwise.

A23. Scalar. for any $N \in \mathbb{Z}$, and any $\underline{X}$ representing X, $\mathscr{A}.\texttt{scalar}(\underline{X}, N)$ outputs an $\mathscr{A}$-representation of the scalar endomorphism $[N] \in \mathrm{End}(X)$.

A24. Frobenius. $\mathscr{A}.\texttt{frobenius}(\underline{X}, k)$ outputs an $\mathscr{A}$-representation of the p^k-Frobenius isogeny with domain X.

A25. Automorphisms. For any $\underline{X}$, we have that $\mathscr{A}.\texttt{automorphisms}(\underline{X})$ outputs a list of $\mathscr{A}$-representations of all automorphisms of X over $\overline{\mathbb{F}}_p$.

A26. Quotient. $\mathscr{A}.\texttt{quotient}(\underline{X}, \underline{P_1}, \ldots, \underline{P_{N^g}})$ outputs an $\mathscr{A}$-representation of the isogeny of kernel $G = \{P_1, \ldots, P_{N^g}\}$ if G is a maximal isotropic subgroup of $X[N]$, and $\perp$ otherwise. The field of definition of the output is $\mathbb{F}_{q^d}$ where d is the smallest integer such that $\pi^d(G) = G$, where π is the Frobenius endomorphism of $X/\mathbb{F}_q$.

> **Note:** The codomain of this isogeny (computable using $\mathscr{A}.\texttt{domcodom}(\underline{\varphi})$) is the quotient PPAV X/G.

A27. Product. If $\underline{X}$ and $\underline{X'}$ are $\mathscr{A}$-representations of X and X' respectively, $\mathscr{A}.\texttt{product}(\underline{X}, \underline{X'})$ outputs an $\mathscr{A}$-representation of $X \times X'$, and an efficient $\mathscr{A}$-representation of the two inclusion maps $\mathsf{inc} : X \to X \times X'$ and $\mathsf{inc}' : X' \to X \times X'$, and the two projections: $\mathsf{pr} : X \times X' \to X$ and $\mathsf{pr}' : X \times X' \to X'$.

A28. Decomposition. If $\underline{X}$ is an $\mathscr{A}$-representation of X, $\mathscr{A}.\texttt{decompose}(\underline{X})$ outputs $(\underline{X_i}, \underline{\mathsf{inc}_i}, \underline{\mathsf{pr}_i})_{i=1}^n$ such that each $\underline{X_i}$ encodes an irreducible PPAV such that $X \cong X_1 \times \cdots \times X_n$, each $\underline{\mathsf{inc}_i}$ encodes the embedding $\mathsf{inc}_i : X_i \to X$, and each $\underline{\mathsf{pr}_i}$ encodes the projection $\mathsf{pr}_i : X \to X_i$.

3.6 Consequences

Other important properties can be expected from an efficient model, but do not need to be assumed as axioms: they are consequences of other axioms. We will prove that any (proto-)efficient model also satisfies the following properties. This will be proved in Section 4.

P1. Equality testing for morphisms. For any $\underline{\varphi}, \underline{\varphi'} \in S_{\text{mrph}}$, $\mathscr{A}.\texttt{equal}(\underline{\varphi}, \underline{\varphi'})$ outputs `true` if and only if $\text{mrph}(\underline{\varphi}) = \text{mrph}(\underline{\varphi'})$.

P2. Characteristic polynomial for endomorphisms. Given $\underline{\alpha} \in S_{\text{mrph}}$ such that $\text{mrph}(\underline{\alpha}) \in \text{Hom}(X, X)$ is an endomorphism of X, $\mathscr{A}.\texttt{char_poly}(\underline{\alpha})$ outputs $F \in \mathbb{Z}[x]$, the characteristic polynomial of α.

P3. Norm. Given $\underline{\varphi} \in S_{\text{mrph}}$, $\mathscr{A}.\texttt{norm}(\underline{\varphi})$ outputs $\|\varphi\| = \sqrt{\text{tr}(\varphi^\dagger \circ \varphi)}$.

P4. Testing polarized isogeny. Given $\underline{\varphi} \in S_{\text{mrph}}$, $\mathscr{A}.\texttt{is_polarized}(\underline{\varphi})$ output $N \in \mathbb{Z}$ such that $\varphi = \text{mrph}(\underline{\varphi})$ is an N-isogeny if it exists, and $\perp$ otherwise.

3.7 From proto-efficient to efficient

With the above axioms, one only gets a *proto*-efficient model. To get an efficient model, one needs to further satisfy the following axiom.

A29. Canonical and compact form for N-isogenies. For any integer N and any $\underline{\varphi} \in S_{\text{mrph}}$ such that $\text{mrph}(\underline{\varphi}) : X \to Y$ is an N-isogeny, $\mathscr{A}.\texttt{canonical}(\underline{\varphi})$ outputs $\underline{\varphi_{\texttt{can}}} \in S_{\text{mrph}}$ such that $\text{mrph}(\underline{\varphi}) = \text{mrph}(\underline{\varphi_{\texttt{can}}})$, and $\text{length}(\underline{\varphi_{\texttt{can}}}) = (\log(q) + \log(N))^{\text{poly}(\dim(X))}$. Furthermore, for any $\underline{\varphi'} \in S_{\text{mrph}}$, the following two statements are equivalent:

– $\mathscr{A}.\texttt{canonical}(\underline{\varphi'}) = \underline{\varphi_{\texttt{can}}}$;
– $\mathrm{mrph}(\underline{\varphi'}) = \mathrm{mrph}(\underline{\varphi_{\texttt{can}}})$ and $\mathscr{A}.\texttt{domcodom}(\underline{\varphi'}) = \mathscr{A}.\texttt{domcodom}(\underline{\varphi_{\texttt{can}}})$.

Note: One can efficiently test if $\varphi = \mathrm{mrph}(\underline{\varphi})$ is an N-isogeny, see Corollary 4.12.

4 Generic results for efficient models

We show that we can extend any proto-efficient model $\mathscr{A}$ with the additional subroutines P1 to P4. This will help proving that certain instantiations are indeed proto-efficient (illustrated with the theta model), or designing more advanced algorithms (as we will show with the interpolation algorithm).

4.1 The CRT representation of points

In the definition of an efficient representation of points, our algorithms depend polynomially on the degree of the extension field in which our points are defined. This leads to issues when manipulating points defined over several different extensions. For instance if $P \in X(\mathbb{F}_{q^k})$ and $Q \in X(\mathbb{F}_{q^{k'}})$, then $P + Q$ might only be defined over $\mathbb{F}_{q^{kk'}}$. Therefore, representing $P + Q \in X(\mathbb{F}_{q^{kk'}})$ scales polynomially in kk', while the pair (P, Q) has a length that scales only in $k + k'$. It can therefore be advantagous to represent certains points formally as tuples, like (P, Q) for the sum $P + Q$. This idea extends several axioms to a "CRT version" as follows.

As such, if our points are defined over exponentially-sized extension fields, our algorithms will not be efficient. If, however, our point is of smooth order, we can instead use a *CRT representation*.

Definition 4.1 (CRT representation of points). *Let X be a PPAV, and N a positive integer such that $N = q_1 \dots q_n$ with the q_i pairwise coprime. For a point $P \in X[N]$, we say that $(P_1, \dots, P_n)$ is a* $(q_1, \dots, q_n)$-CRT decomposition *of P if $P = \sum_{i=1}^{n} P_i$, and for $1 \leq i \leq n$, we have $P_i \in X[q_i]$. If each P_i is given as an $\mathscr{A}$-representation, the tuple is a* $(q_1, \dots, q_n)$-CRT $\mathscr{A}$-representation.

Given any factorization $N = q_1 \dots q_n$ with the q_i pairwise coprime, the subroutines $\mathscr{A}.\texttt{diff}$, $\mathscr{A}.\texttt{add}$ or $\mathscr{A}.\texttt{eval}$ can trivially be extended to takes as input points in $(q_1, \dots, q_n)$-CRT $\mathscr{A}$-representation and produce the output in $(q_1, \dots, q_n)$-CRT $\mathscr{A}$-representation, in polynomial time, by performing the operation on each component independently. Similarly, two points in $(q_1, \dots, q_n)$-CRT $\mathscr{A}$-representation can be tested for equality component-wise.

Remark 4.2. Note that given two positive integers k_1, k_2, there is an embedding $X[k_i] \hookrightarrow X[\mathrm{lcm}(k_1, k_2)]$. So, given points in distinct CRT representations, we can efficiently compute a common CRT representation.

This CRT representation is particularly helpful to compute "isogeny paths", i.e., N-isogenies where N is a smooth integer, as follows.

Proposition 4.3. *For any (proto-)efficient model $\mathscr{A}$, one can define a new polynomial time subroutine $\mathscr{A}.\texttt{quotient_CRT}$ such that the following holds. For any $\underline{X}$ with $\mathrm{ppav}(\underline{X}) = (q, X)$, with X a PPAV of dimension g, any integer $N = q_1^{e_1} \dots q_n^{e_n}$ (with pairwise coprime integers q_i) and any sequence of points $(P_i)_{i=1}^k$ each in $(q_1^{e_1}, \dots, q_n^{e_n})$-CRT $\mathscr{A}$-representation generating a maximal isotropic subgroup $G \subset X[N]$, and an integer d such that $G^{(q^d)} = G$ (i.e., the isogeny of kernel G is defined over $\mathbb{F}_{q^d}$), the algorithm $\mathscr{A}.\texttt{quotient_CRT}(\underline{X}, \mathtt{1}^d, (\mathtt{1}^{q_i^g})_{i=1}^n, (\underline{P_i})_{i=1}^k)$ outputs an $\mathscr{A}$-representation of an N-isogeny of kernel G (defined over $\mathbb{F}_{q^d}$).*

Algorithm 1 $\mathscr{A}.\texttt{quotient_CRT}$

Require: $(\underline{X}, \mathtt{1}^d, (\mathtt{1}^{q_i^g})_{i=1}^n, (\underline{P_i})_{i=1}^k)$ as in Proposition 4.3.
Ensure: An $\mathscr{A}$-representation of an N-isogeny of kernel $G = \langle P_1, \dots, P_k \rangle$.

Parse each CRT point P_i as $(P_i^j)_{j=1}^n$, where $P_i^j \in X[q_j^{e_j}]$ and $P_i = \sum_{j=1}^n P_i^j$.
$(Q_i^j)_{j=1}^n \leftarrow (P_i^j)_{j=1}^n$.
$\varphi \leftarrow [1]_X = \mathscr{A}.\texttt{scalar}(X, 1)$ the identity map on the domain X.
$Y \leftarrow X$ the codomain of φ.
for $j = 1, \dots, n$ **do**
  for $t = 1, \dots, e_j$ **do**
    for $i = 1, \dots, k$ **do**
      $R_i \leftarrow [q_j^{e_j - t}] Q_i^j$. ▷ Using $\mathscr{A}.\{\texttt{scalar}, \texttt{eval}\}$
    end for
    $H \leftarrow \langle R_1, \dots R_k \rangle \subset Y[q_j]$. ▷ Enumeration of all $\mathbb{Z}/q_i\mathbb{Z}$-linear combinations
    $\psi \leftarrow \mathscr{A}.\texttt{quotient}(Y, H)$ a q_i-isogeny with kernel H.
    $(Q_i^j)_{i,j} \leftarrow (\psi(Q_i^j))_{i,j} = (\mathscr{A}.\texttt{eval}(\psi, Q_i^j))_{i,j}$.
    $\varphi \leftarrow \psi \circ \varphi = \mathscr{A}.\texttt{compose}(\psi, \varphi)$.
    $Y \leftarrow$ the codomain of φ. ▷ With $\mathscr{A}.\texttt{domcodom}$
  end for
end for
return φ.

Proof. The algorithm, detailed in Algorithm 1, is a classical chained isogeny computation. At each (j,t)-iteration, write $H_{j,t-1}, \psi_{j,t-1}, \varphi_{j,t-1}$ for the state at the start of the iteration, and $H_{j,t}, \psi_{j,t}, \varphi_{j,t}$ for the state at the end. The standard argument for correctness adapts straightforwardly by observing inductively that for each (j,t), we have:

- $H_{j,t} = \varphi_{j,t-1}(G \cap X[q_j^t])$, hence $\ker \varphi_{j,t} \subseteq G$;
- if $t = e_j$, then for each i we have $R_i = Q_i^j = \varphi_{j,t-1}(P_i^j)$, hence $P_i^j \in \ker(\varphi_{j,t})$.

From the first item, we deduce that the kernel of the final φ is a subgroup of G, and from the second item, we deduce that all generators P_i^j (hence all elements of G) are in the kernel of the final φ.

Let us prove that it runs in polynomial time in the length of the input. The number of steps and iterations is clearly polynomially bounded in the length of

the input, and each step is polynomial in the length of its own input. Therefore, it only remains to show that the length of intermediate date remains bounded in the length of the input.

Since $H_{j,t} = \varphi_{j,t-1}(G \cap X[q_j^t])$, and G and $X[q_j^t]$ are both defined over $\mathbb{F}_{q^d}$, we deduce by induction that each $\varphi_{j,t-1}$ and each $H_{j,t}$ is defined over $\mathbb{F}_{q^d}$. In particular, the points $R_i = \varphi_{j,t-1}([q_j^{e_j - t}]P_i^j)$ are defined over the compositum of $\mathbb{F}_{q^d}$ (polynomially bounded in length($\mathtt{1}^d$)) and the field of definition of P_i^j (polynomially bounded in length(P_i^j)), so each R_i is polynomially bounded in the length of the input. We deduce that the length of each R_i, and in fact all points in H, is polynomially bounded. The length of H itself is polynomial since it contains q_i^g many points (which is polynomial in length($\mathtt{1}^{q_i^g}$)).

This bound on length(H) and the polynomial running time of $\mathscr{A}$.`quotient` and $\mathscr{A}$.`eval` imply the boundedness of ψ and $(Q_i^j)_{i,j}$. Bounding the length of $\varphi_{j,t}$ requires a slightly more subtle argument. Indeed, if we only reason from the fact that $\mathscr{A}$.`compose` runs in polynomial time, we get that there exists a constant c such that $\text{length}(\varphi_{j,t}) \leq (\text{length}(\psi_{j,t}) + \text{length}(\varphi_{j,t-1}))^c$, and an induction might explode superpolynomially. We therefore use the stronger fact that $\text{length}(\varphi_{j,t}) \leq \text{length}(\psi_{j,t}) + \text{length}(\varphi_{j,t-1}) + O(1)$, and the induction yields

$$\text{length}(\psi_{j,t}) = \text{poly}(\texttt{input length}) + O\left(\sum_{j'=1}^{j-1} e_{j'} + t\right) = \text{poly}(\texttt{input length}).$$

□

4.2 Equality testing for morphisms

We now show that in any proto-efficient model, one can efficiently test equality of morphisms. We first show that two morphisms of bounded norm that agree on a sufficiently large torsion subgroup are necessarily equal.

Lemma 4.4. *Let $N \in \mathbb{N}$ be a positive integer coprime to p, and φ a non-zero morphism of PPAVs such that* $\ker([N]) \subseteq \ker(\varphi)$. *We have $N \leq \|\varphi\|$.*

Proof. Let X be the domain of φ. By hypothesis, $\ker([N]) \subseteq \ker(\varphi)$. By [25, pg. 111, Theorem 1A] applied to the underlying unpolarized abelian varieties, there exists φ' such that $\varphi = \varphi' \circ [N]$. As $\varphi' \neq 0$ we get

$$\|\varphi\|^2 = \|\varphi' \circ [N]\|^2 = \text{tr}(\varphi' \circ [N] \circ [N] \circ \varphi'^{\dagger}) = (N^2)\,\|\varphi'\|^2 \geq N^2,$$

since $\|\varphi'\|^2 \in \mathbb{Z}$. Taking square roots gives the result. □

Remark 4.5. If moreover φ' is an N-isogeny for some $N \in \mathbb{N}$, then we have the better bound $\|\varphi\| \leq N\sqrt{2g}$ (because in that case, $\text{Tr}(\varphi' \circ \varphi'^{\dagger}) \in 2g\mathbb{Z}$).

Proposition 4.6. *Let $\mathscr{A}$ be a proto-efficient model for PPAVs. There exists an algorithm that implements a subroutine $\mathscr{A}$.`equal` satisfying the property P1 (equality testing for morphisms).*

Algorithm 2 Equality testing for morphisms

Require: The input consists of two morphisms φ and ψ in $\mathscr{A}$-representation;
Ensure: Decide whether $\varphi = \psi$.

1: $(\underline{X}, \underline{Y}) \leftarrow \mathscr{A}.\texttt{domcodom}(\underline{\varphi})$ the domain and codomain in $\mathscr{A}$-representation
2: $(\underline{X}', \underline{Y}') \leftarrow \mathscr{A}.\texttt{domcodom}(\underline{\psi})$ the domain and codomain in $\mathscr{A}$-representation
3: **if** $\mathscr{A}.\texttt{equal}(\underline{X}, \underline{X}') = \texttt{false}$ or $\mathscr{A}.\texttt{equal}(\underline{Y}, \underline{Y}') = \texttt{false}$ **then**
4: **return** `false`.
5: **end if**
6: $B \leftarrow \mathscr{A}.\texttt{bound_norm}(\underline{\varphi})$
7: $B' \leftarrow \mathscr{A}.\texttt{bound_norm}(\underline{\psi})$
8: $(\ell_j)_{j=1}^k \leftarrow$ the list of smallest primes (in increasing order) coprime with p such that $\prod_{j=1}^k \ell_j > B + B'$.
9: **for** $j = 1, \ldots, k$ **do**
10: $(P_i^j)_{i=1}^{2g} \leftarrow$ a basis of $X[\ell_i]$ ▷ Using $\mathscr{A}.\texttt{torsion}$
11: $(Q_i^j)_{i=1}^{2g} \leftarrow (\varphi(P_i^j))_{i=1}^{2g}$ ▷ Using $\mathscr{A}.\texttt{eval}$
12: $(R_i^j)_{i=1}^{2g} \leftarrow (\psi(P_i^j))_{i=1}^{2g}$ ▷ Using $\mathscr{A}.\texttt{eval}$
13: **if** $\mathscr{A}.\texttt{equal}(\underline{Q_i^j}, \underline{R_i^j}) = \texttt{false}$ for some $i \in \{1, \ldots, 2g\}$ **then**
14: **return** `false`.
15: **end if**
16: **end for**
17: **return** `true`.

Proof. We claim that Algorithm 2 implements a subroutine $\mathscr{A}.\texttt{equal}$ satisfying the property P1.

By the Prime Number Theorem, at Step 8, the length k and the entries ℓ_i are polynomially bounded in $\log B$ (which itself is polynomially bounded in length$(\underline{\varphi})$, because the binary representation of B was produced by a polynomial time algorithm on input $\underline{\varphi}$). In particular, and since all subroutines of $\mathscr{A}$ run in polynomial time, Algorithm 7 runs in polynomial time.

If the algorithm returns `false`, there is an index i and a point $P \in X[\ell_i]$ such that $\varphi(P) \neq \psi(P)$, so $\varphi \neq \psi$. So when the output is `false`, the algorithm is correct.

If the algorithm returns `true`, then $\varphi(P) = \psi(P)$ for all $P \in X[\ell_i]$, for all i. In other words, $\ker([N]) \subseteq \varphi - \psi$, where $N = \prod_i \ell_i$. If $\varphi \neq \psi$, by Lemma 4.4 and the triangle inequality, this implies $N \leq \|\varphi - \psi\| \leq \|\varphi\| + \|\psi\| = B + B'$, a contradiction (proving that $\varphi = \psi$). So when the output is `true`, the algorithm is correct, which finishes the proof. □

4.3 Computing the characteristic polynomial of endomorphisms

The goal of this section is to prove that in any proto-efficient model for PPAVs, we also have property P2: given an endomorphism $\alpha \in \mathrm{End}(X)$, one can efficiently compute its characteristic polynomial $F_\alpha(x) \in \mathbb{Z}[x]$. We prove this in Proposition 4.10, and then deduce that we also have the properties P3 and P4.

Our strategy consists of computing $F_\alpha(x) \mod \ell$ for several small prime numbers ℓ (by looking at the action of α on $X[\ell]$), and reconstructing $F_\alpha(x)$ using CRT. To do so, we first bound the coefficients of the polynomial $F_\alpha(x)$.

Lemma 4.7. *Let X be a PPAV of dimension g. For $\alpha \in \mathrm{End}(X)$, we have $\deg(\alpha) \leq \|\alpha\|^{2g}$.*

Proof. Let $\mathrm{End}^0(X) := \mathrm{End}(X) \otimes \mathbb{Q}$, and consider the bilinear form $\mathrm{End}^0(X) \times \mathrm{End}^0(X) \to \mathbb{Q}$, $\langle x, y\rangle \mapsto \mathrm{tr}(x^\dagger y)$. By [23, Theorem 17.3], it is positive definite. Furthermore, let $\gamma := \alpha^\dagger\alpha \in \mathrm{End}^0(X)$. Then, $\gamma^\dagger = \gamma$, and $\langle \gamma x, y\rangle = \langle x, \gamma y\rangle$. The matrix $\mathbf{M}_\gamma$ corresponding to the linear action of γ on $\mathrm{End}^0(X)$ given by $x \mapsto \gamma x$ is Hermitian, symmetric and positive definite, so it has $[\mathrm{End}^0(X) : \mathbb{Q}]$ real positive eigenvalues r_i. Applying the arithmetic mean-geometric mean inequality, we get

$$\frac{1}{f}\operatorname{Tr}(\mathbf{M}_\gamma) = \frac{1}{f}\sum r_i \geq \left(\prod r_i\right)^{1/f} = \det(\mathbf{M}_\gamma)^{1/f}. \tag{1}$$

To complete the proof, we now relate the trace and determinant of $\mathbf{M}_\gamma$ to the trace and norm of γ.[3] Suppose X is simple. Then $\mathrm{End}^0(X)$ is finite dimensional simple algebra over $\mathbb{Q}$, so [23, Lemma 12.11] states that $\operatorname{Tr}(\gamma) = \frac{2g}{f}\operatorname{Tr}(\mathbf{M}_\gamma)$ and $\deg(\gamma) = \det(\mathbf{M}_\gamma)^{2g/f}$. Combining this with (1), we get

$$\frac{1}{f}\operatorname{Tr}(\mathbf{M}_\gamma) = \frac{1}{2g}\operatorname{Tr}(\gamma) \geq \det(\mathbf{M}_\gamma)^{1/f} = \deg(\gamma)^{1/2g}.$$

Noting that $\operatorname{Tr}(\gamma) = \|\alpha\|^2$ and $\deg(\gamma) = \deg(\alpha)^2$, we obtain the required bound.

If X is not simple, then X is isogenous to a product of simple abelian varieties $X_1^{n_1} \times \cdots \times X_k^{n_k}$ pairwise non-isogenous (as an unpolarised PPAV). As the structure of the endomorphism algebra only depends on the isogeny class of X, it is sufficient to look at the case $X = X_1^{n_1} \times \cdots \times X_k^{n_k}$. Then, M_α the matrix representation of α is a block diagonal matrix, with diagonal blocks given by M_{α_i}, the matrix representation of $\alpha_i \in \mathrm{End}^0(X_i^{n_i})$. Write $\gamma = \alpha^\dagger\alpha$ as a tuple $(\gamma_1, \ldots, \gamma_k)$ such that $\gamma_i = \alpha_i^\dagger\alpha_i \in M_{n_i}(\mathrm{End}^0(X_i^{n_i}))$ is the image of γ through the projection on $X_i^{n_i}$. The characteristic polynomial $F_\gamma(x)$ of γ as the endomorphism of abelian varieties is the characteristic polynomial of the action of γ on $V_\ell X := T_\ell X \otimes \mathbb{Q}_\ell$. Noting that $V_\ell X = \prod V_\ell(X_i^{n_i})$, we have $F_\gamma(x) = \prod F_{\gamma_i}(x)$. Looking at the constant terms we have $\deg(\gamma) = \prod \deg(\gamma_i)$. As $M_{n_i}(\mathrm{End}^0(X_i))$ is simple (since X_i is), we have $\deg(\gamma_i) \leq \frac{\operatorname{Tr}(\gamma_i)^{2g_i}}{2g_i}$, where $g_i = n_i \dim(X_i)$. In conclusion,

$$\deg(\alpha) \leq \prod \frac{\|\alpha_i\|^{2g_i}}{2g_i} \leq \frac{\|\alpha\|^{\sum 2g_i}}{\prod 2g_i} \leq \|\alpha\|^{2g}.$$

□

[3] Namely, the trace and norm of the matrix corresponding to the action of γ on the Tate module $V_\ell X = T_\ell X \otimes \mathbb{Q}_\ell$ for some $\ell \neq p$ [23, Prop. 12.9].

Lemma 4.8. *Let $P(X) = a_d X^d + \cdots + a_1 X + a_0$ be a polynomial of degree d with coefficients $a_i \in \mathbb{Z}$. Then, for all $i = 0, \dots d$, we have $|a_i| \leq d^d \sum_{n=0}^{d} |P(n)|$.*

Proof. Applying Lagrange interpolation formula to pairs $\{(n, P(n))\}_{0 \leq n \leq d}$, we can write

$$P(X) = \sum_{n=0}^{d} P(n) \prod_{i=0, i \neq n}^{d} (X - i)/(n - i).$$

Letting c_k be the coefficient of X^k in $\prod_{\substack{i=0 \\ i \neq n}}^{d} X - i$, we have $|a_i| \leq |c_i| \sum_{n=0}^{d} |P(n)|$. Bounding the size of c_k by writing it in terms of elementary symmetric polynomials, we find

$$|a_i| \leq \binom{d}{d-i} d^{d-i} \sum_{n=0}^{d} |P(n)| \leq d^d \sum_{n=0}^{d} |P(n)|,$$

as required. □

Lemma 4.9. *Let X be a PPAV of dimension g, and $\alpha \in \mathrm{End}(X)$ with $\|\alpha\| \leq B$, for some $B \in \mathbb{Z}$. Let $F_\alpha(X) := X^{2g} + a_{2g-1} X^{2g-1} + \cdots + a_1 X + a_0$, be the characteristic polynomial of α, with $a_i \in \mathbb{Z}$. For all $i = 0, \dots, 2g-1$, we have*

$$|a_i| \leq (2g)^{2g} \sum_{n=0}^{2g} (B + n\sqrt{2g})^{2g}.$$

Proof. For any $N \in \mathbb{Z}$, we have $F_\alpha(N) = \deg(\alpha - N)$ [24, Theorem 10.9]. Combining this with Lemma 4.7, we get $|F_\alpha(N)| \leq \|\alpha - N\|^{2g}$. By the triangle equality, we obtain $|F_\alpha(N)| \leq (\|\alpha\| + \|[N]\|)^{2g} \leq (B + N\sqrt{2g})^{2g}$. Applying this bound to Lemma 4.8 with $d = 2g$, we obtain the result. □

To compute the characteristic polynomial of an endomorphism, we follow [1, Lemma 3.2], which is itself inspired by Schoof's point counting algorithm.

Proposition 4.10. *Let $\mathscr{A}$ be a proto-efficient model for PPAVs. There exists an algorithm that implements the subroutine $\mathscr{A}$.`char_poly` satisfying the property P2 (characteristic polynomial for endomorphisms).*

Proof. We claim that Algorithm 3 implements a subroutine $\mathscr{A}$.`char_poly` satisfying the property P2.

We first show that this algorithm is correct. The characteristic polynomial of α is the characteristic polynomial of the matrix describing the action of α on the Tate module $T_\ell(A)$ for any prime ℓ not equal to the characteristic of the underlying field. Therefore, the polynomials $F_\alpha^j(x) \in \mathbb{Z}/\ell_j\mathbb{Z}[x]$ computed in Step 11 are the characteristic polynomials of α modulo the primes ℓ_j. Applying CRT to $F_\alpha^{(1)}, \dots, F_\alpha^{(k)}$ we obtain a polynomial $F_\alpha(x) \in (\mathbb{Z}/L\mathbb{Z})[x]$, where $L := \prod_{j=1}^{k} \ell_j$. By Lemma 4.9, the coefficients of $F_\alpha(x)$ are bounded in size by N, and

Algorithm 3 Characteristic polynomial for endomorphisms

Require: The input consists of an endomorphism α in $\mathscr{A}$-representation;
Ensure: The characteristic polynomial $F_\alpha(x) \in \mathbb{Z}[x]$ of α.
1: $(\underline{X}, \underline{Y}) \leftarrow \mathscr{A}.\texttt{domcodom}(\alpha)$ the domain and codomain in $\mathscr{A}$-representation
2: **if** not $\mathscr{A}.\texttt{isomorphic}(\underline{X}, \underline{Y})$ **then**
3: **return** $\perp$ ▷ Input is not an endomorphism
4: **end if**
5: $B \leftarrow \mathscr{A}.\texttt{bound_norm}(\underline{\alpha})$
6: $N \leftarrow (2g)^{2g-1} \sum_{n=0}^{2g} (B + N\sqrt{2g})^{2g}$ ▷ c.f. Lemma 4.9.
7: $(\ell_j)_{j=1}^k \leftarrow$ the list of smallest primes (in increasing order) coprime with p such that $\prod_{j=1}^k \ell_j > 2N$.
8: **for** $j = 1, \ldots, k$ **do**
9: $(P_i^j)_{i=1}^{2g} \leftarrow$ a basis of $X[\ell_j]$ ▷ Using $\mathscr{A}.\texttt{torsion}$
10: $\mathbf{M} \leftarrow$ the matrix giving the action of α on $(P_i^j)_{i=1}^{2g}$ ▷ Using $\mathscr{A}.\texttt{eval}$
11: $F_\alpha^{(j)}(x) \leftarrow \det(xI - \mathbf{M})$ ▷ The characteristic polynomial of α modulo ℓ_j
12: **end for**
13: $F_\alpha(x) \leftarrow$ apply CRT to the coefficients of $F_\alpha^{(1)}(x), \ldots, F_\alpha^{(k)}(x)$
14: **return** $F_\alpha(x)$.

by construction $N < 2L$, so viewing $F_\alpha(x)$ as a polynomial over $\mathbb{Z}$ gives us the characteristic polynomial of α.

We now show that the algorithm is efficient, as per Axiom A1. By the Prime Number Theorem, at Line 7, the length k and the entries ℓ_i are polynomially bounded in $\log N$, which itself is polynomially bounded in $\log B$ by Lemma 4.9, and thus length$(\underline{\alpha})$. Therefore, since all subroutines of $\mathscr{A}$ called run in polynomial time, Algorithm 3 runs in polynomial time. □

As a consequence, we can efficiently compute the norm of a morphism, and test if an isogeny is polarised.

Corollary 4.11. *Let $\mathscr{A}$ be a proto-efficient model for PPAVs. Then, one can add a subroutine $\mathscr{A}.\texttt{norm}$ satisfying the property P3 (norm of morphisms).*

Algorithm 4 Computing the norm

Require: The input consists of a morphism $\varphi : X \to Y$ in $\mathscr{A}$-representation;
Ensure: The norm $\|\varphi\|$.
1: $\alpha \leftarrow \varphi^\dagger \circ \varphi$ ▷ Using $\mathscr{A}.\texttt{adjoint}$ and $\mathscr{A}.\texttt{compose}$
2: $F_\alpha(x) = \mathscr{A}.\texttt{char_poly}(\underline{\alpha})$ ▷ Using Proposition 4.10
3: Parse $F_\alpha(x) = x^{2g} + a_{2g-1}x^{2g-1} + \cdots + a_1 x + a_0$, where $g = \dim(X)$
4: **return** $\sqrt{-a_{2g-1}}$.

Proof. We claim that Algorithm 4 implements a subroutine $\mathscr{A}.\texttt{norm}$ satisfying the property P3. Indeed, the running time is clear from Proposition 4.10, and

correctness follows from the fact that $\|\varphi\|^2 = \mathrm{Tr}(\varphi^\dagger \circ \varphi) = -a_{2g-1}$, by definition. □

Corollary 4.12. *Let $\mathscr{A}$ be a proto-efficient model for PPAVs. Then, Algorithm 5 implements a subroutine $\mathscr{A}$.*`is_polarized` *satisfying the property P4 (testing polarized isogeny).*

Algorithm 5 Testing polarized isogeny

Proof. **Require:** The input consists of a morphism $\varphi : X \to Y$ in $\mathscr{A}$-representation;
Ensure: The integer N such that φ is an N-isogeny if it exists, and $\perp$ otherwise.

1: $(\underline{X}, \underline{Y}) \leftarrow \mathscr{A}.\texttt{domcodom}(\underline{\varphi})$ the domain and codomain in $\mathscr{A}$-representation
2: $g \leftarrow \mathscr{A}.\texttt{dimension}(\underline{X})$
3: **if** $\mathscr{A}.\texttt{dimension}(\underline{Y}) \neq g$ **then**
4: **return** $\perp$
5: **end if**
6: $N \leftarrow \|\varphi\|^2/2g$ ▷ Using $\mathscr{A}.\texttt{norm}$ from Corollary 4.11
7: $\alpha \leftarrow \varphi^\dagger \circ \varphi$ ▷ Using $\mathscr{A}.\texttt{adjoint}$, and $\mathscr{A}.\texttt{compose}$
8: **if** $N \notin \mathbb{Z}$ or $\alpha \neq [N]$ **then** ▷ Using $\mathscr{A}.\texttt{scalar}$, and $\mathscr{A}.\texttt{equal}$ from Proposition 4.6
9: **return** $\perp$
10: **end if**
11: **return** N.

Correctness of Algorithm 5 follows from the following facts: $\varphi : X \to Y$ is an N-isogeny if and only if $\dim(X) = \dim(Y)$ and $\varphi^\dagger \circ \varphi = [N]$; and if φ is an N-isogeny, then $\|\varphi\|^2 = 2\dim(X)N$. The runtime is clear from Corollary 4.11. □

5 A brief introduction to the theta model

To work with our definition of an efficient representation of PPAVs in practice, it is crucial to give an instantiation of our definitions. The goal of this section is to introduce our main example: the theta model.

5.1 The theta model

Let λ_D be a principal polarisation coming from an ample (separable) divisor D. Recall that for any $n \geq 3$, the divisor nD is very ample and that choosing a basis of $L(nD)$ leads to an embedding of X into projective space. The theory of *theta structures*, discussed at length in [32], [31], or [10], gives a canonical choice for such a basis. For simplicity, we restrict to the case of interest to us: the theta model of level $n = 4$ given by a symmetric theta structure on $4D$.

Remark 5.1. When using the theta model, we make the assumption that the 8-torsion of X is $\mathbb{F}_q$-rational to ensure that X can be represented in the theta model over $\mathbb{F}_q$. We can always extend the base field to ensure this: the degree of

the extension containing all $X[8]$ is the smallest k such that π^k acts trivially on $X[8]$. In particular, k is at most the largest order of an element in $\mathrm{Aut}(X[8])$, which is in $2^{O(\dim(X)^2)} = 2^{\mathrm{poly}(\dim(X))}$.

Theta structures. The *theta group* $G(D)$ of D on X is the set of couples (P, f) where $P \in X$ and f is a map such that $D = \tau_P^* D - \mathrm{div}(f)$. It has a group structure given by $(P, f) \cdot (Q, g) = (P + Q, \tau_P^* g \cdot f)$. We have a forgetful map $\mathrm{Ob} : (P, f) \mapsto P$ whose kernel is given by the constant non-zero maps, i.e., $\overline{\mathbb{F}}_p^\times$. The image of Ob is $X[N]$: the set of points P such that $\tau_P^* D - D$ is principal.

As the Weil pairing is non-degenerate, for every $n \in \mathbb{N}^*$, there exists a symplectic decomposition of $X[n] = G_1(n) \oplus G_2(n)$ [10, Lemma 5.1.9]. For $n \in \mathbb{N}$, we define $K_1(n) := (\mathbb{Z}/n\mathbb{Z})^g$, and $K_2(n) := \mathrm{Hom}(K_1(n), \overline{\mathbb{F}}_p^\times)$, and define a pairing $e_{K(n)}$ on $K(n) := K_1(n) \oplus K_2(n)$ by $((x_1, \chi_1), (x_2, \chi_2)) = \chi_2(x_1)\chi_1(x_2)^{-1} \in \overline{\mathbb{F}}_p^\times$. It follows that $(K(n), e_{K(n)})$ is symplectically isomorphic to $(X[n], e_n)$ for every n. Namely, there exists an isomorphism $\nu : K(n) \simeq X[N]$ such that for all $x, y \in K(n)$, $e_{K(n)}(x, y) = e_n(\nu(x), \nu(y))$.

We define the *Heisenberg group* $\mathcal{H}(n) := \overline{\mathbb{F}}_p^\times \times K(n)$ with the following group law $(\alpha_1, (x_1, \chi_1)) \cdot (\alpha_2, (x_2, \chi_2)) \mapsto (\alpha_1\alpha_2\chi_2(x_1), (x_1 + x_2, \chi_1\chi_2))$.

Definition 5.2. *A theta structure of level n is an isomorphism $\Theta_{nD} : \mathcal{H}(n) \to G(nD)$ trivial on the subgroup $\overline{\mathbb{F}}_p^\times$ and such that the induced morphism $\widetilde{\Theta_{nD}} : K(n) \to X[N]$ is a symplectic isomorphism. In other words, we have the following commutative diagram:*

$$\begin{array}{ccccccccc} 1 & \longrightarrow & \overline{\mathbb{F}}_p^\times & \longrightarrow & \mathcal{H}(n) & \xrightarrow{Ob} & K(n) & \longrightarrow & 1 \\ & & \| & & \Theta_{nD}\downarrow & & \widetilde{\Theta_{nD}}\downarrow & & \\ 1 & \longrightarrow & \overline{\mathbb{F}}_p^\times & \longrightarrow & G(nD) & \longrightarrow & X[N] & \longrightarrow & 1 \end{array}$$

Moreover, if ψ is an automorphism of $\mathcal{H}(n)$ such that its restriction to $\overline{\mathbb{F}}_p^\times$ is the identity and Θ_{nD} a theta structure, then $\Theta_{nD} \circ \psi$ is a theta structure. Conversely, every theta structure differs from another by such an automorphism (see [32, §3.5]). We denote by $\mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(n))$ this automorphism group.

Theta functions. We now explain how a theta structure Θ_{nD} determines a basis of $L(nD)$ when $n \geq 3$. The key is the fact that the Heisenberg group $\mathcal{H}(n)$ has a unique (up to isomorphism) irreducible representation on which $\overline{\mathbb{F}}_p^\times$ acts naturally [26, Proposition 3]. It is defined as $V(n) = \{f : K_1(n) \to \overline{\mathbb{F}}_p\}$, with the following action of $\mathcal{H}(n)$ on $V(n)$: $(\alpha, x, \chi) \cdot f = \left[y \mapsto \alpha\chi(y)^{-1} f(x - y)\right]$.

We also have an irreducible representation $L(nD)$ of $G(nD)$ given by, for any $(P, f) \in G(nD)$ and $s \in L(nD)$ by $(P, f) \cdot s = \tau_{-P}^*(s \cdot f)$. A theta structure $\Theta_n D : \mathcal{H}(n) \to G(nD)$ translates the action of $G(nD)$ on $L(nD)$ into an action of $\mathcal{H}(n)$. By unicity of the irreducible representation of $\mathcal{H}(n)$, Schur's lemma implies that there exists a unique (up to rescaling) isomorphism of representations $\beta_{\Theta_{nD}}$:

$V(n) \to L(nD)$. The image of the canonical basis $(\delta_i)_{i \in K_1(n)} = (\delta_i)_{i \in (\mathbb{Z}/4\mathbb{Z})^g}$ of $V(n)$ by the map $\beta_{\Theta_{nD}}$ is a basis $(\theta_i)_{i \in (\mathbb{Z}/4\mathbb{Z})^g}$ of $L(4D)$, called *theta functions*.

Symmetric theta structures. Instead of arbitrary theta structure, it is often convenient to restrict to *symmetric* theta structures (they notably simplify explicit formulas, e.g., addition). To define a symmetric theta structure, we need to start with a symmetric divisor. We say that D is symmetric if $[-1]^*D - D$ is principal. Fortunately, every divisor is algebraically equivalent to a symmetric one [10, Proposition 5.3.2]. Moreover, the principal polarisation $\lambda_D : X \to X^\vee$ depends only on the algebraic class of D, so we may always replace D by an algebraically equivalent symmetric divisor.

A theta structure Θ_{nD} is *symmetric* if $\Delta_{-1} \circ \Theta_{nD} = \Theta_{nD} \circ \nabla_{-1}$, where $\Delta_{-1} : G(nD) \to G(nD)$ and $\nabla_{-1} : \mathcal{H}(n) \to \mathcal{H}(n)$ are the morphisms defined in [10, Definition 5.3.9] and [10, Definition 5.3.12][4].

Theta null points and Riemann relations. Taking $n = 4$, theta functions provide a basis for the space $L(4D)$, inducing the embedding $\iota_\Theta : X \hookrightarrow \mathbb{P}^{4^g-1}$, given by $P \longmapsto (\theta_i(P))_{i \in (\mathbb{Z}/4\mathbb{Z})^g}$. As $4D$ is very ample, the map ι_Θ is an embedding and the θ_i define projective coordinates for X. The image of 0_X by this embedding is called the *theta null point*, and it fully determines the image $\iota_\Theta(X) \subseteq \mathbb{P}^{4^g-1}$. Indeed, the theta null point $\iota_\Theta(0_X)$ alone determines projective equations defining $\iota_\Theta(X)$, called *Riemann relations*. We refer the reader to [32, Theorem 4.4.6] for explicit formulas.

6 An instantiation of a proto-efficient representation of PPAVs

In this section, we define model $\mathscr{A}$ representing the symmetric theta model of level-4. The main result of this section is given by Theorem 6.25, which shows that the theta model is a proto-efficient model for PPAVs.

6.1 Encodings for the theta model

The first step is to define the encodings for the theta model.

Definition 6.1 (The theta model for PPAVs). *Let $\mathscr{A}$ be a probabilistic algorithm representing the symmetric theta model of level-4. We define the* encodings for PPAVs and points in the theta model *as follows:*

- *Let $S_{\text{ppav}} \subset \{0,1\}^*$ be a set of binary strings of the form*

$$\texttt{ppav} \,|||\, \underline{q} \,|||\, \texttt{1}^g \,|||\, \underline{t_0} \,|||\, \cdots \,|||\, \underline{t_d},$$

  where q is a prime power, g is a positive integer, $d = 4^g - 1$, and $t_0, \ldots, t_d \in \mathbb{F}_q$. For $D \in S_{\text{ppav}}$, we define $\text{ppav}(D) := (q, X)$ *where $X \subset \mathbb{P}^d$ is an abelian variety over $\mathbb{F}_q$ of dimension g with theta null point $(t_0 : \cdots : t_d) \in \mathbb{P}^d(\mathbb{F}_q)$.*

[4] ∇_{-1} is called D_{-1} in the reference. We denote it differently to avoid a notational conflict with divisors.

– *For any* $\underline{X} \in S_{\mathrm{ppav}}$ *with* $\mathrm{ppav}(\underline{X}) = (q, X)$ *of dimension* g*, let* $S_{\mathrm{point}}(\underline{X}) \subset \{0,1\}^*$ *be a set of binary strings of the form*

$$\mathtt{point} \,|||\, \underline{X} \,|||\, \mathtt{1}^k \,|||\, \underline{P_0} \,|||\, \cdots \,|||\, \underline{P_d},$$

where k *is a positive integer,* $P_i \in \mathbb{F}_{q^k}$ *and* $d = 4^g - 1$*. For* $D \in S_{\mathrm{point}}(\underline{X})$*, we define* $\mathrm{pt}_{\underline{X}}(D) := (P_0 : \cdots : P_d)$*, a point in* $X(\mathbb{F}_{q^k})$*.*

For morphisms, let $S_{\mathrm{mrph}} \subset \{0,1\}^*$ *be a set of binary strings of the form* `morphism` $|||$ `b` *where* `b` *is of the following possible forms:*

– `scalar` $|||\, \underline{X} \,|||\, \underline{N} \,|||\, \underline{B}$*: represents the multiplication-by-*N *map* $[N] : X \to X$ *for* $N \in \mathbb{Z}$ *and* $B = N\sqrt{2g}$*.*
– `isogeny` $|||\, \mathtt{1}^k \,|||\, \underline{X} \,|||\, \underline{X'} \,|||\, \underline{P_1} \,|||\, \cdots \,|||\, \underline{P_{N^g}} \,|||\, \underline{N} \,|||\, \underline{B}$*: represents a (non-scalar) polarised* N*-isogeny* $\varphi : X \to X'$ *defined over* $\mathbb{F}_{q^k}$ *with kernel* $\langle \underline{P_1}, \ldots, \underline{P_{N^g}} \rangle$*, where* N *is coprime to* p*, and* $B = \sqrt{2gN}$*.*
– `frobenius` $|||\, \underline{X} \,|||\, \underline{B}$*: represents the Frobenius endomorphism* π *on* X*, with* $\underline{X}$ *defined over* $\mathbb{F}_q$*, and* $B = \sqrt{2gq}$*.*
– `verschiebung` $|||\, \underline{X} \,|||\, \underline{B}$*: represents the Verschiebung endomorphism* $\widehat{\pi}$ *on* X*, with* $\underline{X}$ *defined over* $\mathbb{F}_q$*, and* $B = \sqrt{2gq}$*.*
– `proj` $|||\, \underline{Y} \,|||\, \underline{X_1} \,|||\, \cdots \,|||\, \underline{X_n} \,|||\, \underline{i} \,|||\, \underline{B}$*: represents the projection* $\mathsf{pr}_i : Y \to X_i$*, where* $Y = X_1 \times \cdots \times X_n$*,* $1 \le i \le n$*, and* $B = \sqrt{2g}$*.*
– `incl` $|||\, \underline{Y} \,|||\, \underline{X_1} \,|||\, \cdots \,|||\, \underline{X_n} \,|||\, \underline{i} \,|||\, \underline{B}$*: represents the inclusion* $\mathsf{inc}_i : X_i \to Y$*, where* $Y = X_1 \times \cdots \times X_n$*,* $1 \le i \le n$*, and* $B = \sqrt{2g}$*.*
– `matrix` $|||\, \mathtt{1}^k \,|||\, \underline{X} \,|||\, \underline{X'} \,|||\, \mathbf{M} \,|||\, \underline{B}$*: represents the isomorphism* $X \to X'$ *induced by linear map* $\mathbf{M} \in \mathrm{PGL}_d(\mathbb{F}_{q^k})$ *on* $\mathbb{P}^d$ *with* $d = 4^{\dim(X)} - 1$*, and* $B = \sqrt{2g}$*. If* $X = X'$*, this represents an automorphism.*

We call these atomic encodings for morphisms in the theta model*. We allow* S_{mrph} *to also contain binary strings of the following form:*

– `chain` $|||\, \mathtt{1}^k \,|||\, \underline{X} \,|||\, \underline{X'} \,|||\, \underline{\varphi_n} \,|||\, \ldots \,|||\, \underline{\varphi_1} \,|||\, \underline{B}$*: represents* $\varphi_n \circ \cdots \circ \varphi_1 : X \to X'$ *defined over* $\mathbb{F}_{q^k}$*, with* $\underline{\varphi_i}$ *an atomic encoding, and* $B = \prod_i \|\varphi_i\|$*.*
– `add` $|||\, \mathtt{1}^k \,|||\, \underline{X} \,|||\, \underline{X'} \,|||\, \underline{\varphi_1} \,|||\, \ldots \,|||\, \underline{\varphi_n} \,|||\, \underline{B}$*: represents* $\varphi_1 + \cdots + \varphi_n : X \to X'$ *defined over* $\mathbb{F}_{q^k}$*, where each* $\underline{\varphi_i}$ *is an atomic encoding, and* $B = \sum_i \|\varphi_i\|$*.*

We call the first part of `b` *(e.g.,* `scalar`*) the* morphism type*. For each possible* $\underline{\varphi} \in S_{\mathrm{mrph}}$*, we define the image* $\mathrm{mrph}(\underline{\varphi})$ *to be the corresponding morphism* φ*.*

For the rest of this section, we let $\mathscr{A}$ represent the theta model with encodings as defined in Definition 6.1. We prove a series of results that will culminate in showing $\mathscr{A}$ is a proto-efficient model.

Proposition 6.2. *$\mathscr{A}$ satisfies axioms for extracting metadata (A9 – A13). All the corresponding subroutines of $\mathscr{A}$ run in polynomial time.*

Proof. This follows almost immediately by construction, as we can extract the necessary information from the encodings allowing us to define the following algorithms:

- $\mathscr{A}.\texttt{field}(\underline{z}) = \begin{cases} q, \text{ if } \underline{z} \in S_{\text{ppav}} \\ q^k, \text{ if } \underline{z} \in S_{\text{pt}} \\ q^k, \text{ if } \underline{z} \in S_{\text{mrph}} \text{ of type } \texttt{isogeny}, \texttt{chain}, \texttt{add}, \texttt{matrix} \\ q, \text{ if } \underline{z} \in S_{\text{mrph}} \text{ of other type} \end{cases}$;
- $\mathscr{A}.\texttt{dimension}(\underline{X}) = g$;
- $\mathscr{A}.\texttt{parent}(\underline{P}) = \underline{X}$;
- $\mathscr{A}.\texttt{domcom}(\underline{\varphi}) = \begin{cases} \underline{X}, \underline{X}', \text{ if } \varphi \text{ is of type } \texttt{isogeny}, \texttt{chain}, \texttt{add}, \texttt{matrix} \\ \underline{X}, \underline{X}, \text{ if } \varphi \text{ is of type } \texttt{scalar}, \texttt{frobenius}, \texttt{verschiebung} \\ \underline{Y}, \underline{X}_i, \text{ if } \varphi \text{ is of type } \texttt{proj} \\ \underline{X}_i, \underline{Y}, \text{ if } \varphi \text{ is of type } \texttt{incl} \end{cases}$;
- $\mathscr{A}.\texttt{bound_norm}(\underline{\varphi}) = B$.

It is straightforward to see that $\mathscr{A}.\texttt{field}$, $\mathscr{A}.\texttt{dimension}$, $\mathscr{A}.\texttt{parent}$, $\mathscr{A}.\texttt{domcom}$ are correct. We are left to argue why $\mathscr{A}.\texttt{bound_norm}$ is correct.

For $\underline{\varphi} = [N]$ of type `scalar`, we have $\|[N]\| = N\sqrt{\mathrm{Tr}\,[1]} = N\sqrt{2g}$. If $\underline{\varphi}$ is of type `isogeny`, we have $\|\varphi\| \leq \sqrt{2gN}$ by Remark 4.5. For type `frobenius` and `verschiebung` we similarly deduce a bound of $\sqrt{2gq}$. For $\underline{\mathsf{pr}}_i$, we have $\|\mathsf{pr}_i\| = \sqrt{\mathrm{Tr}(\mathsf{inc}_i \circ \mathsf{pr}_i)} \leq \sqrt{\mathrm{Tr}(\mathrm{id})} \leq \sqrt{2g}$, where id is the identity. Similarly, $\|\mathsf{inc}_i\| = \sqrt{2g}$. Correctness follows for type `matrix`, by noting that the adjoint is $\mathbf{M}^\dagger$.

Suppose $\underline{\varphi}$ is of type `chain`. Then, $\varphi = \varphi_n \circ \cdots \circ \varphi_1$, with $\underline{\varphi}_i$ an atomic encoding. Noting that

$$\varphi_i^\dagger \circ \varphi_i = \begin{cases} \text{a forgetful map,} & \text{for } \underline{\varphi}_i \text{ of type } \texttt{proj} \\ \text{a scalar map,} & \text{otherwise} \end{cases},$$

the morphism $\varphi^\dagger\varphi$ is a composition of forgetful maps and the scalar map $[M]$, for some $M \in \mathbb{Z}$. Using this, we find that

$$\|\varphi_n \circ \cdots \circ \varphi_1\| \leq \sqrt{2gM} \leq \|\varphi_n\| \cdot \cdots \cdot \|\varphi_1\|,$$

as required. The bound for morphisms of type `add` follows from the triangle inequality. We conclude that for every $\underline{\varphi} \in S_{\text{mrph}}$, $\mathscr{A}.\texttt{bound_norm}(\underline{\varphi})$ outputs a correct bound.

It is clear that each subroutine runs in polynomial time. □

Proposition 6.3. *$\mathscr{A}$ satisfies axioms for encodings (A2 – A6, A8). The corresponding subroutines of $\mathscr{A}$ runs in polynomial time.*

Proof. We note that $\mathrm{length}(\underline{X}) = (\log q)\cdot 4^{\dim(X)} + \dim(X) + O(1)$ and $\mathrm{length}(\underline{P}) = k + \mathrm{length}(\underline{X}) + (\log q)\cdot(4^{\dim(X)} - 1) + O(1)$, so $\mathscr{A}$ satisfies axioms A4 and A5 on compacity.

We now show geometric surjectivity (A3). Suppose we are given a PPAV X defined over $\mathbb{F}_q$. Take k to be the minimal positive integer such that $X[8] \subset X(\mathbb{F}_{q^k})$, and define X' to be the base change of X to $\mathbb{F}_{q^k}$. Then, chosing a basis of $L(D)$ given by (symmetric) theta functions of level 4, gives us an embedding $X' \hookrightarrow \mathbb{P}^{4^{\dim(X)-1}}$. By noting that $k \leq 2^{\mathrm{poly}(\dim(X))}$, we have the result.

We now discuss type testing (A2). Given input $\underline{z}$, the algorithm $\mathscr{A}.\texttt{type}$ begins by parsing the first part of the encoding to obtain a string $\texttt{type} \in \{\texttt{ppav}, \texttt{point}, \texttt{morphism}\}$, or otherwise outputs $\perp$. It then proceeds as follows:

- If $\texttt{type} = \texttt{ppav}$, then $\underline{z} = \texttt{ppav} \,|||\, \underline{g} \,|||\, \texttt{1}^g \,|||\, \underline{t_0} \,|||\, \cdots \,|||\, \underline{t_d}$. Let $g = \mathscr{A}.\texttt{dimension}(\underline{z})$, and verify that $d = 4^g - 1$, otherwise output $\perp$. Then, verify whether the point $T := (t_0 \colon \cdots \colon t_d) \in \mathbb{P}^d(\mathbb{F}_q)$ satisfies the equations in [32, Eq. 4.3.1]. If so, T is a theta null point representing the polarised abelian variety defined in [32, Theorem 4.7.1], and as $d = 4^g - 1$, the polarisation is principal. Therefore, the algorithm outputs $\texttt{ppav}$, and otherwise $\perp$.
- If $\texttt{type} = \texttt{point}$, then $\underline{z} = \texttt{point} \,|||\, \underline{X} \,|||\, \texttt{1}^k \,|||\, \underline{P_0} \,|||\, \cdots \,|||\, \underline{P_d}$. The algorithm outputs $\texttt{point}$ if and only if $d = 4^g - 1$ and $P := (P_0 \colon \cdots \colon P_d) \in \mathbb{P}^d(\mathbb{F}_{q^k})$ verifies the Riemann relations on X. Indeed, in this case, $P \in X(\mathbb{F}_{q^k})$.
- If $\texttt{type} = \texttt{morphism}$, then the algorithm parses $\underline{\varphi}$ and extracts each atomic encoding $\underline{\varphi_i}$. If each atomic encoding is valid and has the correct domain and codomain, it outputs $\texttt{morphism}$. Otherwise, it outputs $\perp$.

Finally, we prove the canonical form axioms (A6, A8). On input $\underline{z}$, the algorithm $\mathscr{A}.\texttt{canonical}$ proceeds as follows:

- If $\mathscr{A}.\texttt{type}(\underline{z}) = \texttt{ppav}$: The algorithm extracts the theta null point $(t_0 \colon \cdots \colon t_d)$ from $\underline{z}$, and normalises by the first non-zero coordinate (e.g., $(1 : t_1/t_0 : \cdots : t_d/t_0)$ if $t_0 \neq 0$). The algorithm outputs an encoding of the PPAV $X_{\texttt{can}}$ corresponding to this normalised theta null point.
- If $\mathscr{A}.\texttt{type}(\underline{z}) = \texttt{point}$: The algorithm outputs an encoding of the point z normalised by the first non-zero coordinate, as above.
- Otherwise, outputs $\perp$. □

We delay Axiom A7 until the proof of the isomorphism axiom in Section 6.4.

6.2 Axioms for point arithmetic

We now turn to proving that $\mathscr{A}$ satisfies axioms for point arithmetic.

Proposition 6.4. *$\mathscr{A}$ satisfies the axioms for point-arithmetic (A14 – A17). Each of the corresponding subroutines of $\mathscr{A}$ is polynomial time.*

Proof. We define the algorithms as:

- $\mathscr{A}.\texttt{zero}(\underline{X})$ outputs $(t_0 \colon \cdots \colon t_d)$, the theta null point.
- $\mathscr{A}.\texttt{diff}(\underline{X}, \underline{P}, \underline{Q})$ uses the formulas from [31, Theorem 2.7.3] to, on input $\mathscr{A}.\texttt{zero}(\underline{X}) = T$, P, Q, compute $P - Q$ in polynomial time.
- For a positive integer N, $\mathscr{A}.\texttt{weil}(\underline{X}, N, \underline{P}, \underline{Q})$ computes the N-Weil pairing $e_N(P, Q)$ in polynomial time using an adaptation of Miller's algorithm to the theta model described in [31, §3.3.2].

– We follow [16, Theorem 2.1.1] to define $\mathscr{A}.\texttt{torsion}(\underline{X}, 1^{N^{2g}})$, for a positive integer N. The algorithm computes the polynomial formulas of degree $\mathrm{poly}(N)$ for the multiplication map $[N] : X \to X$ in the theta model, and computes its roots using the resultant method. The algorithm will output $\underline{P}_1, \ldots, \underline{P}_{N^{2g}}$, such that P_i are the aforementioned roots and indeed N-torsion points on $X(\overline{\mathbb{F}}_p)$. This algorithm terminates in $\mathrm{poly}(\mathrm{length}(\underline{X}), N)$ as required. □

6.3 Axioms for manipulating morphisms

In this section, we prove the axioms for manipulating morphisms. We split this proof into various parts, first proving the evaluation and adjoint axioms.

Lemma 6.5. *$\mathscr{A}$ satisfies the evaluate axiom (A18). The corresponding subroutine $\mathscr{A}.\texttt{eval}$ runs in polynomial time.*

Proof (of Lemma 6.5). We describe the algorithm $\mathscr{A}.\texttt{eval}$. On input $\varphi : X \to X'$ where $X/\mathbb{F}_q$ and a point $P = (P_0 : \cdots : P_d) \in \mathbb{P}^d(\mathbb{F}_{q^\ell})$ for some $\ell \geq 1$. We first check that $d = 4^{\dim(X)} - 1$ and $P \in X(\mathbb{F}_{q^\ell})$ using the Riemann relations. If not, we output $\perp$. We then proceed differently for each morphism type to compute $\varphi(P)$. For type `scalar`, we the double-and-add algorithm, and for type `isogeny`, we run [19, Algorithm 4.10]. For type `frobenius`, we compute the image as $(P_0^q : \cdots : P_d^q)$. For type `verschiebung`, we note that $\pi^\ell(P) = P$, so we can compute the image of P as $\pi^{\ell-1}([p]P)$. For type `matrix`, we simply output $\mathbf{M} \cdot P$.

For types `incl` and `proj`, suppose first that $n = 2$. We first show how to evaluate $\mathsf{inc}_1 : X_1 \to X_1 \times X_2$ and $\mathsf{inc}_2 : X_2 \to X_1 \times X_2$, using the Serge embedding $\mathbb{P}^{d_1} \times \mathbb{P}^{d_2} \hookrightarrow \mathbb{P}^{d_{12}}$, where $d_i = 4^{\dim(X_i)} - 1$ and $d_{12} = 4^{\dim(X_1)+\dim(X_2)} - 1$, defined as

$$\iota : ((P_0 : \cdots : P_{d_1}), (Q_0 : \cdots : Q_{d_2})) \mapsto (P_0Q_0 : P_0Q_1 : \cdots : P_iQ_j : \cdots : P_{d_1}Q_{d_2}) .$$

Let $T_2 := \mathscr{A}.\texttt{zero}(\underline{X}_2)$. For $P_1 \in X_1$, we have $\mathsf{inc}_1(P) = \iota(P_1, T_2)$. We can similarly evaluate points at inc_2.

To evaluate the projection pr_1 at $P = (P^{(1)}, P^{(2)}) \in X_1 \times X_2$, we find an index $0 \leq j \leq d_2$ such that $P_j^{(2)}$, and compute $\mathsf{pr}_1(P)$ as $(P_0^{(1)}P_j^{(2)} : \cdots : P_{d_1}^{(1)}P_j^{(2)})$. Note that such an index must exist as $P^{(2)} \in \mathbb{P}^{d_2}$. We similarly evaluate points at pr_2. For $n > 2$, we compute the evaluation by iterating this process.

We now extend $\mathscr{A}.\texttt{eval}$ to work with non-atomic encodings of type `chain` and `add` (namely, compositions and additions of isogenies). For each, given $\underline{\varphi}$, we extract the atomic encodings $\underline{\varphi}_1, \ldots, \underline{\varphi}_n$. For `chain`, the algorithm iterates through $i = 1, \ldots, n-1$, verifying that the codomain of φ_i is the domain of φ_{i+1}, and then evaluating $\varphi_i(P)$ at φ_{i+1} using the corresponding method above. For `add`, the algorithm verifies that the domain and codomain of each φ_i is equal, evaluates P at each φ_i to obtain Q_i, and outputs $Q_1 + \cdots + Q_n$ (using $\mathscr{A}.\texttt{add}$ derived from $\mathscr{A}.\texttt{diff}$). Every procedure runs in polynomial time in the length of the input. □

We now show that $\mathscr{A}$ satisfies the adjoint axiom. The following lemma states that projections and inclusions are adjoints of one another.

Lemma 6.6. *Let $n > 1$ be an integer. For $i = 1, \dots, n$, consider the projection and inclusion morphisms $\mathsf{pr}_i : X_1 \times \cdots \times X_n \to X_i$ and $\mathsf{incl}_i : X_i \to X_1 \times \cdots \times X_n$. For each i, we have $\mathsf{pr}_i^\dagger = \mathsf{inc}_i$ and $\mathsf{inc}_i^\dagger = \mathsf{pr}_i$.*

Proof. We prove this results for $n = 2$ for clarity, but the same argument applies to $n > 2$. Let λ_{D_1} and λ_{D_2} be the principal polarisations on X_1 and X_2 with assocaited ample divisors D_1 and D_2, respectively. We have the following commutative diagram:

$$\begin{array}{ccc} X_1 \times X_2 & \xrightarrow{\mathsf{pr}_1} & X_1 \\ {\scriptstyle \lambda_{\mathsf{pr}_1^* D_1 + \mathsf{pr}_2^* D_2}}\downarrow & & \downarrow{\scriptstyle \lambda_{D_1}} \\ (X_1 \times X_2)^\vee & \xleftarrow[\mathsf{pr}_1^*]{} & X_1^\vee \end{array}$$

For $P \in X_1$, we have

$$\mathsf{pr}_1^* \circ \lambda_{D_1}(P) = (\tau_P^* D_1 - D_1) \times X_2 \in (X_1 \times X_2)^\vee$$

On the other hand, we have:

$$\begin{aligned} \lambda_{\mathsf{pr}_1^* D_1 + \mathsf{pr}_2^* D_2} \circ \mathsf{inc}_1(P) = \ & \lambda_{\mathsf{pr}_1^* D_1 + \mathsf{pr}_2^* D_2}(P, 0) \\ & = \tau_{(P,0)}^*(\mathsf{pr}_1^* D_1 + \mathsf{pr}_2^* D_2) - \mathsf{pr}_1^* D_1 - \mathsf{pr}_2^* D_2 \\ & = (\mathsf{pr}_1 \tau_{(P,0)})^* D_1 - \mathsf{pr}_1^* D_1 + (\mathsf{pr}_2 \tau_{(P,0)})^* D_2 - \mathsf{pr}_2^* D_2 \\ & = (\tau_P \mathsf{pr}_1)^* D_1 - \mathsf{pr}_1^* D_1 + (\tau_0 \mathsf{pr}_2)^* - \mathsf{pr}_2^* D_2 \\ & = (\tau_P^* D_1) \times X_2 - D_1 \times X_2 + X_1 \times (\tau_0^* D_2) - X_1 \times D_2 \\ & = \mathsf{pr}_1^* \circ \lambda_{D_1}(P) + X_1 \times \underbrace{(\tau_0^* D_2 - D_2)}_{0} = \mathsf{pr}_1^* \circ \lambda_{D_1}(P). \end{aligned}$$

The equality in the third line follows from the fact that for two morphisms f, g we have $(fg)^* = g^* f^*$. In conclusion, $\mathsf{pr}_1^* \circ \lambda_{D_1} = \lambda_{\mathsf{pr}_1^* D_1 + \mathsf{pr}_2^* D_2} \circ \mathsf{inc}_1$, and so $\mathsf{pr}_1^\dagger = \mathtt{inc}_1$. Similarly, $\mathsf{pr}_2^\dagger = \mathtt{inc}_2$. □

Lemma 6.7. *$\mathscr{A}$ satifies the adjoint axiom (A21). On input $\underline{X}$, the corresponding subroutine $\mathscr{A}$.`adjoint` runs in expected time at most* $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

Proof. We first describe the algorithm $\mathscr{A}$.`adjoint` when the input $\underline{\varphi}$ is atomic.

- For type `scalar`, $\mathscr{A}.\mathtt{adjoint}(\underline{\varphi}) = \underline{\varphi}$.
- For type `isogeny`, let $\varphi : X \to X'$ be an N-isogeny with kernel $\langle P_1, \dots, P_{N^g} \rangle$ Compute $Q_i = \varphi(R_i)$, for every $R_i \in X[N]$ using $\mathscr{A}.\mathtt{torsion}$ and $\mathscr{A}.\mathtt{eval}$. Letting $\phi : X' \to X$ be the N-isogeny with kernel generated by the Q_i, we have that $\varphi^\dagger = \beta \circ \phi$ for some $\beta \in \mathrm{Aut}(X)$. The algorithm generates the list of automorphisms $\alpha_1, \dots, \alpha_m$ of X using $\mathscr{A}.\mathtt{automorphisms}$ (from Corollary 6.20), and then determines for which i we have $\alpha_i \circ \phi \circ \varphi = [N]$ using $\mathscr{A}.\mathtt{equality}$ (from Proposition 4.6) and $\mathscr{A}.\mathtt{composition}$. The algorithm outputs and encoding of the composition $\alpha_i \circ \phi$.

- For types `frobenius` and `verschiebung`, by definition we have $\pi^\dagger = \widehat{\pi}$. So, we define $\mathscr{A}$.`adjoint`$(\underline{\pi}) = \underline{\widehat{\pi}}$ and $\mathscr{A}$.`adjoint`$(\underline{\widehat{\pi}}) = \pi$.
- For types `incl` and `proj` we use by Lemma 6.6 and define $\mathscr{A}$.`adjoint`$(\mathsf{inc}_i) = \mathsf{pr}_i$ and $\mathscr{A}$.`adjoint`$(\mathsf{pr}_i) = \mathsf{inc}_i$.
- For type `matrix`, $\mathscr{A}$.`adjoint` outputs an encoding of the conjugate-transpose matrix.

For non-atomic encodings we note that $(\varphi_1 \circ \varphi_2)^\dagger = \varphi_2^\dagger \circ \varphi_1^\dagger$, and $(\varphi_1 + \varphi_2)^\dagger = \varphi_1^\dagger + \varphi_2^\dagger$. Therefore, $\mathscr{A}$.`adjoint` simply replaces each atomic encoding in $\underline{\varphi}$ by an encoding of its adjoint, computed using the methods above. In the case of `chain`, the algorithm also reverses the ordering of the atomic encodings.

All steps of this algorithm run in polynomial time in the length of the input, except for the invocation of $\mathscr{A}$.`automorphisms` for type `isogeny`. By Corollary 6.20 has complexity $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$. □

The composition and addition axioms (A19, A20) follow trivially by construction: we simply use the encoding types `chain` and `add`, respectively. Combining with the lemmas above, we get the following proposition.

Proposition 6.8. *$\mathscr{A}$ satisfies axioms (A18 – A21). The subroutines $\mathscr{A}$.`addition` and $\mathscr{A}$.`composition` run in polynomial time.*

6.4 Axioms for creating morphisms

The final group of axioms that we prove are those for creating morphisms. For this, we first develop theory to characterize isomorphisms in the theta model.

6.4.1 Isomorphisms and theta structures. In this section, we prove that isomorphisms in the theta model can all be obtained as a change of theta coordinates (Theorem 6.14). We start in the following lemma by proving that isomorphisms of PPAVs induce isomorphisms between their theta groups.

Lemma 6.9. *Let $\alpha : X \to X'$ be an isomorphism between PPAVs, with level-n theta groups $G(nD)$ and $G(nD')$ respectively. Suppose that $(\alpha^{-1})^*(nD) - nD'$ is principal, and let $h \in \mathbb{F}_q(X')^\times$ such that $\mathrm{div}(h) = (\alpha^{-1})^*(nD) - nD'$. We have an isomorphism*

$$\tilde{\alpha} : G(nD) \longrightarrow G(nD') : (P, f) \longmapsto \left(\alpha(P), \frac{(\alpha^{-1})^* f \cdot h}{\tau^*_{\alpha(P)} h}\right).$$

Proof. A direct computation proves that that it is well defined (i.e., for any $(P, f) \in G(nD)$, we have $\tilde{\alpha}(P, f) \in G(nD')$), it is a group homomorphism, and it is invertible with inverse $\widetilde{\alpha^{-1}}$. □

We now build an understanding on how these isomorphisms interact with the theta structures on X and X'.

Lemma 6.10. *Let $\alpha : X \to X'$ be an isomorphism between PPAVs, with level-n theta groups $G(nD)$ and $G(nD')$ respectively. Suppose that $(\alpha^{-1})^*D - D'$ is principal. Let $\Theta : \mathcal{H}(n) \to G(nD)$ be a theta structure on X. Then, $\tilde{\alpha} \circ \Theta$ is a level-n theta structure on X', and we have $\iota_\Theta = \iota_{\tilde{\alpha}\circ\Theta} \circ \alpha$.*

Proof. Let $h \in \mathbb{F}_q(X')^\times$ such that $\mathrm{div}(h) = (\alpha^{-1})^*D - D'$. The map $\gamma : L(nD) \to L(nD') : \theta \mapsto \frac{(\alpha^{-1})^*\theta}{h^n}$ is an isomorphism of representations of $\mathcal{H}(n)$ (via Θ and $\tilde{\alpha} \circ \Theta$ respectively). Let $\beta : V(n) \to L(nD)$ and $\beta' : V(n) \to L(nD')$ be the unique (up to scalar) isomorphisms with the representation $V(n)$ (see [26, Proposition 3]). By Schur's lemma, there is a scalar λ such that $\beta' = \lambda \cdot \gamma \circ \beta$. Write θ_i and $\tilde{\theta}_i$ the theta coordinates induced by Θ and $\tilde{\alpha} \circ \Theta$ respectively. We have

$$\tilde{\theta}_i = \beta'(\delta_i) = \lambda \cdot (\gamma \circ \beta)(\delta_i) = \lambda \cdot \gamma(\theta_i) = \frac{\lambda}{h^n}(\alpha^{-1})^*\theta_i.$$

In particular, for any $P \in X$, $\tilde{\theta}_i(\alpha(P)) = \frac{\lambda}{h^n(P)}(\alpha^{-1})^*\theta_i(\alpha(P)) = \frac{\lambda}{h^n(P)}\theta_i(P)$, hence $\iota_{\tilde{\alpha}\circ\Theta}(\alpha(P)) = \iota_\Theta(P)$. □

Lemma 6.11. *Let $\alpha : X \to X'$ be an isomorphism between PPAVs, with level-n theta groups $G(nD)$ and $G(nD')$ respectively. There exists a unique $Q \in X'$ such that $(\alpha^{-1})^*D - \tau_Q^*D'$ is principal. Let $h \in \mathbb{F}_q(X')^\times$ such that $\mathrm{div}(h) = (\alpha^{-1})^*D - \tau_Q^*D'$.*

Proof. Since α is an isomorphism, $(\alpha^{-1})^*D$ and D' are algebraically equivalent, i.e., $(\alpha^{-1})^*D - D' \in \mathrm{Pic}^0(X') = \hat{X}'(\overline{\mathbb{F}}_p)$. Since $\lambda_{D'} : X' \to \hat{X}'$ is an isomorphism, there exists a unique $Q \in X'$ such that $(\alpha^{-1})^*D - D' \simeq \lambda_{D'}(Q) \simeq \tau_Q^*D' - D'$, i.e., such that $(\alpha^{-1})^*D \simeq \tau_Q^*D'$, i.e., such that $(\alpha^{-1})^*D - \tau_Q^*D'$ is principal. □

Lemma 6.12. *Let X be PPAV with level-n theta structure $\Theta : \mathcal{H}(n) \to G(nD)$. Let $Q \in X$. Let $\tau_Q^* : G(nD) \to G(n(\tau_Q^*D)) : (P, f) \mapsto (P, \tau_Q^*f)$. We have $\iota_{\tau_Q^*\circ\Theta} = \iota_\Theta \circ \tau_Q$. Note that in particular, the codomains $\iota_\Theta(X)$ and $\iota_{\tau_Q^*\circ\Theta}(X)$ are the same projective variety (but their group structures differ by a translation).*

Proof. The map $\tau_Q^* : L(D) \to L(\tau_Q^*D) : \theta \mapsto \tau_Q^*\theta$ defines an isomorphism of representations of $\mathcal{H}(n)$ (via Θ and $\tau_Q^* \circ \Theta$ respectively), so there exists a scalar λ such that $\theta_i' = \lambda\tau_Q^*\theta_i$, where θ_i are the theta coordinated for Θ and θ_i' the theta coordinates for $\tau_Q^* \circ \Theta$. Therefore $\iota_{\tau_Q^*\circ\Theta} = \iota_\Theta \circ \tau_Q$. □

Definition 6.13. *Let X be a PPAV, and Θ a level-n theta structure on X with $n \geq 3$. For any $\psi \in \mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(n))$, the composition $\Theta \circ \psi$ is another theta structure on X. The isomorphism $\xi_\psi = \iota_{\Theta\circ\psi} \circ \iota_\Theta^{-1} : \iota_\Theta(X) \to \iota_{\Theta\circ\psi}(X)$ is called the* change-of-theta-coordinates *morphism.*

Theorem 6.14. *Consider two PPAVs X and X', with level-n theta structures Θ and Θ' (for divisors D and D' respectively), with $n \geq 3$. The set of isomorphisms from $\iota_\Theta(X)$ to $\iota_{\Theta'}(X')$ is equal to the set*

$$\left\{ \tau_P \circ \xi_\psi : \iota_\Theta(X) \to \iota_{\Theta'}(X') \;\middle|\; \begin{array}{l} \psi \in \mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(n)) \text{ such that } \iota_{\Theta'}(0_{X'}) \in \iota_{\Theta\circ\psi}(X), \\ P = \iota_{\Theta'}(0_{X'}) \text{ and } \tau_P \text{ is translation in } \iota_{\Theta\circ\psi}(X) \end{array} \right\}.$$

Proof. First note that every such $\nu = \tau_P \circ \xi_\psi$ is indeed an isomorphism from $\iota_\Theta(X)$ to $\iota_{\Theta'}(X')$. Indeed, consider a change-of-theta-coordinates isomorphism $\xi_\psi : \iota_\Theta(X) \to \iota_{\Theta\circ\psi}(X)$ such that $\iota_{\Theta'}(0_{X'}) \in \iota_{\Theta\circ\psi}(X)$. Now, let $P = \iota_{\Theta'}(0_{X'})$, and $Q = \iota^{-1}_{\Theta\circ\psi}(P) \in X$. By Lemma 6.12, we have $\tau_Q = \iota^{-1}_{\Theta\circ\psi} \circ \iota_{\tau_Q^*\circ\Theta\circ\psi}$, hence $\tau_P = \iota_{\tau_Q^*\circ\Theta\circ\psi} \circ \iota^{-1}_{\Theta\circ\psi}$. In particular, $\nu = \tau_P \circ \xi_\psi = \iota_{\tau_Q^*\circ\Theta\circ\psi} \circ \iota_\Theta^{-1}$ is indeed an isomorphism. It only remains to prove that it has the correct codomain. We have

$$\iota_{\tau_Q^*\circ\Theta\circ\psi}(0_X) = \nu(\iota_\Theta(0_X)) = (\tau_P \circ \xi_\psi)(\iota_\Theta(0_X)) = \tau_P(\iota_{\Theta\circ\psi}(0_X)) = P = \iota_{\Theta'}(0_{X'}).$$

Since $\iota_{\tau_Q^*\circ\Theta\circ\psi}(X)$ and $\iota_{\Theta'}(X')$ have the same theta null point, they define the same projective variety, so the codomain of $\tau_P \circ \xi_\psi$ is equal to $\iota_{\Theta'}(X')$.

Now, we prove that every isomorphism arises as such a map. Let $\beta : \iota_\Theta(X) \to \iota_{\Theta'}(X')$ be an isomorphism. Let $\alpha = \iota_\Theta^{-1} \circ \beta^{-1} \circ \iota_{\Theta'} : X' \to X$. Let $Q \in X$ such that $(\alpha^{-1})^* D' - \tau_Q^* D$ is principal. We obtain an isomorphism $\tilde{\alpha} : G(nD') \longrightarrow G(n\tau_Q^* D)$, and by Lemma 6.10, we have $\iota_{\Theta'} = \iota_{\tilde{\alpha}\circ\Theta'} \circ \alpha$. We deduce

$$\beta = \iota_{\Theta'} \circ \alpha^{-1} \circ \iota_\Theta^{-1} = \iota_{\Theta'} \circ (\iota_{\Theta'}^{-1} \circ \iota_{\tilde{\alpha}\circ\Theta'}) \circ \iota_\Theta^{-1} = \iota_{\tilde{\alpha}\circ\Theta'} \circ \iota_\Theta^{-1}.$$

Since Θ and $\tau_{-Q}^* \circ \tilde{\alpha} \circ \Theta'$ are two theta structures on X' for the divisor nD we can define $\psi = (\Theta)^{-1} \circ \tau_{-Q}^* \circ \tilde{\alpha} \circ \Theta'$, an automorphism of $\mathcal{H}(n)$. Applying Lemma 6.12, we get $\iota_{\tilde{\alpha}\circ\Theta'} = \iota_{\tau_{-Q}^*\circ\tilde{\alpha}\circ\Theta'} \circ \tau_Q = \iota_{\Theta\circ\psi} \circ \tau_Q$, therefore

$$\beta = \iota_{\tilde{\alpha}\circ\Theta'} \circ \iota_\Theta^{-1} = \iota_{\Theta\circ\psi} \circ \tau_Q \circ \iota_\Theta^{-1} = \tau_P \circ \iota_{\Theta\circ\psi} \circ \iota_\Theta^{-1} = \tau_P \circ \xi_\psi,$$

where $P = \iota_{\Theta\circ\psi}(Q)$. Finally, we have

$$\iota_{\Theta'}(0_{X'}) = \beta(\iota_\Theta(0_X)) = (\tau_P \circ \iota_{\Theta\circ\psi} \circ \iota_\Theta^{-1})(\iota_\Theta(0_X)) = \tau_P(\iota_{\Theta\circ\psi}(0_X)) = P.$$

□

6.4.2 Cycling through changes of theta coordinates. Using the results from the previous section, we now show how to compute all isomorphisms between two given PPAVs in the theta model.

Definition 6.15. *For any $x = (j, \nu) \in K(n)$, we have an automorphism $\psi_x \in \mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(n))$ defined as $\psi_x(\alpha, i, \chi) = (\alpha\chi(j)\nu(i)^{-1}, i, \chi)$.*

Lemma 6.16. *Every automorphism $\psi \in \mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(n))$ can be written as $\psi = \psi' \circ \psi_x$, where $\psi' \in \mathrm{Aut}^0(\mathcal{H}(n))$ is a symmetric automorphism, and $x \in K(n)$.*

Proof. Write the forgetful map $\sigma : \mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(n)) \to \mathrm{Sp}(K(n)) : \varphi \to \sigma(\varphi)$. The restriction of this map to $\mathrm{Aut}^0(\mathcal{H}(n))$ is surjective (see [10, Lemma 6.2.7]), so there exists $\psi' \in \mathrm{Aut}^0(\mathcal{H}(n))$ such that $\sigma(\psi) = \sigma(\psi')$. Let $\tau = (\psi')^{-1}\psi \in \ker\sigma$. The exact sequence [10, Lemma 6.2.2] implies that $\ker\sigma = \{\psi_x \mid x \in K(n)\}$, so there exists $x \in K(n)$ such that $\psi_x = \tau$. By construction, $\psi = \psi' \circ \psi_x$. □

Lemma 6.17. *There exists an algorithm that, on input a PPAV X in $\mathscr{A}$-representation, computes all symmetric change-of-theta-coordinates morphisms from $\iota_\Theta(X)$. Furthermore, this can be extended to an algorithm that computes all change-of-theta-coordinates morphisms from $\iota_\Theta(X)$. Both algorithms run in expected time* $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

Proof. By listing all $\psi \in \mathrm{Aut}^0(\mathcal{H}(4))$ and computing $\xi_\psi : \iota_\Theta(X) \to \iota_{\Theta\circ\psi}(X)$ using the formulas in [10, Theorem 6.2.10], we obtain all symmetric change-of-theta-coordinate morphisms $\{\xi_\psi\}$. By [10, Lemma 6.2.7], we have

$$|\mathrm{Aut}^0(\mathcal{H}(4))| = |\mathrm{Sp}(K(4))||X[2]| = 2^{2\dim(X)} \cdot |\mathrm{Sp}_{2g}(\mathbb{Z}/4\mathbb{Z})|.$$

The runtime of the algorithm is therefore $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

If instead we want to compute all change-of-theta-coordinate morphisms, we compute $\xi_\psi : \iota_\Theta(X) \to \iota_{\Theta\circ\psi}(X)$ for all $\psi \in \mathrm{Aut}_{\overline{\mathbb{F}}_p^\times}(\mathcal{H}(4))$. By Lemma 6.16, this is equivalent to listing all $\psi' \in \mathrm{Aut}^0(\mathcal{H}(4))$ and $x \in K(4)$, and computing $\xi_{\psi'\circ\psi_x}$ for all possible pairs (ψ', ψ_x), where ψ_x as in Definition 6.15. Given (ψ', ψ_x), we compute $\xi_{\psi'\circ\psi_x}$, we apply the action of ψ_x on theta coordinates using the formulas in [10, Proposition 6.2.4(ii)], followed by the action of ψ' as above. As $|K(4)| = 4^{2\dim(X)}$, we get the stated runtime. □

Remark 6.18. The change-of-theta-coordinate morphisms computed in Lemma 6.17 are linear maps given by matrices $\mathbf{M} \in \mathrm{PGL}_d(\mathbb{F}_q)$, where $d = 4^{\dim(X)} - 1$ and $\underline{X}$ is defined over $\mathbb{F}_q$. So, we can encode them using the `matrix` morphism type.

6.4.3 Computing isomorphisms and automorphisms.

We use the results in the previous two sections to compute isomorphisms between PPAVs in the theta model, as well as their automophism group.

Proposition 6.19. *There exists an algorithm that on input two PPAVs X and X' in $\mathscr{A}$-representation, outputs a list of (encodings of) all isomorphisms between X and X' in expected time* $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

Proof. Using the algorithm from Lemma 6.17, we compute all change-of-theta-coordinates morphisms. By Remark 6.18, these are given by matrices $\mathbf{M} \in \mathrm{PGL}_d(\mathbb{F}_q)$, where $d = 4^{\dim(X)} - 1$ and $\underline{X}, \underline{X'}$ are defined over $\mathbb{F}_q$.

Let L be an empty list, and T, T' the output of $\mathscr{A}$.`zero` on $\underline{X}$ and $\underline{X'}$. For each $\mathbf{M}$, we let $X_\mathbf{M}$ be a PPAV (in $\mathscr{A}$-representation) defined by theta null point $\mathbf{M}\cdot T$. If T' lies on $X_\mathbf{M}$, which we can check using the Riemann relations, we add (and encoding of) $\mathbf{N}\cdot\mathbf{M}$ to L, where $\mathbf{N}$ is the translation-by-T' map on $X_\mathbf{M}$. The algorithm then outputs the list L. Correctness and runtime follow directly from Theorem 6.14 and Lemma 6.17. □

Corollary 6.20. *$\mathscr{A}$ satisfies axioms for computing isomorphisms (A22) and automorphisms (A25). The corresponding subroutines have an expected runtime of* $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

Proof. For axiom A22, we define the subroutine $\mathscr{A}.\texttt{isomorphism}(\underline{X}, \underline{X'})$ to run the algorithm from Proposition 6.19 on input $\underline{X}, \underline{X'}$ to obtain a list of isomorphisms, and outputs the first element of the list (or $\perp$ if empty). For axiom A25, we define $\mathscr{A}.\texttt{automorphisms}(\underline{X})$ to run the algorithm from Proposition 6.19 on input $\underline{X}, \underline{X'} = \underline{X}$, to obtain and output a list of automorphisms. □

Another consequence of Proposition 6.19 is an algorithm to compute the canonical form for isomorphism classes.

Lemma 6.21. *Let $n \geq 3$. $\mathrm{Aut}^0(\mathcal{H}(4))$ acts faithfully and transitively on the set of symmetric level-n theta structures Θ on X (for fixed divisor D) by right composition.*

Proof. Following [10, Proposition 6.2.1], the right action of $\mathrm{Aut}(\mathcal{H}(4))$ is faithful and transitive. As our case is a restriction to the symmetric case, only the transivity need to be proven. If Θ and Θ' are two symmetric theta structures, then as $\Theta = \Theta' \circ (\Theta'^{-1} \circ \Theta)$, we need to prove that $\Theta'^{-1} \circ \Theta \in \mathrm{Aut}^0(\mathcal{H}(4))$. Indeed:

$$\begin{aligned}
\Theta'^{-1} \circ \Theta \circ \nabla_{-1} &= \Theta'^{-1} \circ \Delta_{-1} \circ \Theta \\
&= (\Delta_{-1} \circ \Theta')^{-1} \circ \Theta \\
&= (\Theta' \circ \nabla_{-1})^{-1} \circ \Theta \\
&= \nabla_{-1} \circ \Theta'^{-1} \circ \Theta
\end{aligned}$$

□

Lemma 6.22. *$\mathscr{A}$ satisfies the axiom for the canonical form for isomorphism classes (A7). The corresponding subroutine of $\mathscr{A}$ has expected complexity at most* $\mathrm{length}(\texttt{input})^{\mathrm{poly}(\dim(\texttt{input}))}$.

Proof. Let X be a PPAV given in $\mathscr{A}$-representation of dimension g. On input $\underline{X}$, $\mathscr{A}.\texttt{canon_isom}$ works as follows.

First compute all possible symmetric change-of-theta-coordinates $\{\mathbf{M}_j\}$ using Lemma 6.17. Then, for every j compute $T_j = \mathbf{M}_j \cdot T$, where T is the theta null point of X (obtained using $\mathscr{A}.\texttt{zero}$). Defining X_j to be the PPAV in $\mathscr{A}$-representation defined by T_j, by Lemma 6.21 $\{X_j\}$ is a list of all theta-embeddings in the isomorphism class of X by a symmetric theta structure (for a fixed divisor).

Following Theorem 6.14, every isomorphism is a composition of a change-of-theta-coordinate and a translation by a point P. To preserve the symmetry (both of the theta structure [32, Proposition 4.2.24] and of the underlying divisor [10, Proposition 5.3.2(iii)]), P has to be of order 2.

For all j, compute the 2-torsion points $P_{ij} \in X_j[2]$ using $\mathscr{A}.\texttt{torsion}$ from Proposition 6.4 and compute $T_{ij} := \tau_{P_{ij}}(T_j)$ for all possible i. Defining X_{ij} to be the PPAV in $\mathscr{A}$-representation defined by T_{ij}, by the discussion above $\{X_{ij}\}$ is a list of all theta-embeddings in the isomorphism class of X by a symmetric theta structure. Now list the points T_{ij} in lexicographic order, and define X_{can} to be the PPAV corresponding to the first point in the list. Then $\mathscr{A}.\texttt{canon_isom}(\underline{X}) :=$

$\underline{X_{\mathrm{can}}}$. The runtime of this algorithm is $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$ as $\#\{X_{ij}\}$ is bounded by $|\mathrm{Sp}(K(4))||X[2]|^2 = 2^{4\dim(X)}|\mathrm{Sp}_{2g}(\mathbb{Z}/4\mathbb{Z})|$ following the proof of Lemma 6.17. □

6.4.4 Decomposition. We prove that $\mathscr{A}$ satisfies the decomposition axiom.

Lemma 6.23. *$\mathscr{A}$ satisfies the decomposition axiom (A28). The corresponding subroutine $\mathscr{A}$.*`decompose`$(\underline{X})$ *has an expected complexity of* $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

Proof. We first define an algorithm FindSplitting which, on input a PPAV X of dimension g in $\mathscr{A}$-representation, outputs tuples $(\underline{X_1}, \underline{\mathsf{inc}_1}, \underline{\mathsf{pr}_1})$ and $(\underline{X_2}, \underline{\mathsf{inc}_2}, \underline{\mathsf{pr}_2})$ such $X \cong X_1 \times X_2$. It works as follows. For every decomposition $g = g_1 + g_2$ with $g_1 \leq g_2$, For each possible change-of-theta-coordinates $\mathbf{M}$, we apply $\mathbf{M}$ to X using Lemma 6.17 to obtain a new PPAV X' (in $\mathscr{A}$-representation) with theta null point $T = (t_0 : \cdots : t_d)$. Then, for every decomposition $g = g_1 + g_2$ with $g_1 \leq g_2$, we do the following: Rewrite T as $(t_0^{(0)} : \cdots : t_{d_2}^{(0)} : \cdots : t_0^{(d_1)} : \cdots : t_{d_2}^{(d_1)})$. Let i be the first positive integer so that $T_1 := (t_0^{(i)} : \cdots : t_{d_2}^{(i)})$ is non-zero. Similarly, let j be the first positive integer so that $T_2 := (t_j^{(1)} : \cdots : t_j^{(d_1)})$ is non-zero. If T_1, T_2 are theta null points (checked with $\mathscr{A}$.`type`), we output the corresponding PPAVs along with (encodings of) inclusion and projection maps.

If we run through each $\mathbf{M}$, g_1, and g_2 without terminating, we output $\perp$. Correctness follows by noting that the theta null point of a product has a specific form given by the Serge embedding, and by cycling through change-of-theta-coordinates, we reach all possible isomorphic PPAVs (by Lemma 6.17). The runtime follows as from Lemma 6.17, and noting there are only $\lfloor g/2 \rfloor$ possible decompositions of g.

By recursively applying FindSplitting, we obtain $\mathscr{A}$.`decompose`. Correctness follows from correctness of FindSplitting. In this recursion, we apply FindSplitting at most $\dim(X) - 1$ times, so $\mathscr{A}$.`decompose` has the stated complexity. □

6.4.5 Creating morphisms. Putting all results together, we obtain the following proposition.

Proposition 6.24. *$\mathscr{A}$ satisfies the axioms for creating morphisms (A22 – A28). The corresponding subroutines terminate in polynomial time, except for the subroutines $\mathscr{A}$.*`isomorphism`*, $\mathscr{A}$.*`automorphisms` *and $\mathscr{A}$.*`decompose` *that have an expected complexity of at most* $\mathrm{length}(\underline{X})^{\mathrm{poly}(\dim(X))}$.

Proof. The axioms for creating scalar, Frobenius, and quotient morphisms (A23, A24, A26) are trivial by construction of our morphism encodings. Axioms A22, A25 and A28 follow from Corollary 6.20 and Lemma 6.23.

For (A27), we define $\mathscr{A}$.`product`$(\underline{X}, \underline{X'})$ by noting that the theta null point of $X \times X'$ (corresponding to the product theta structure) is given by product of the theta null points of X and X', and the Riemann relations are compatible

with this product theta structure [27, Lemma 1, p. 323]. The algorithm uses this to compute the theta null point of $X \times X'$ as

$$\left(T_0T'_0 : T_0T'_1 : \cdots : T_iT'_j : \cdots T_dT'_{d'}\right),$$

where T is the theta null point of X and $d = 4^{\dim(X)} - 1$, and similarly for T', d', and constructs $\underline{X_1 \times \cdots \times X_n}$. The algorithm outputs $\underline{X_1 \times \cdots \times X_n}$, and atomic encodings of the inclusion and projection maps. □

6.5 The theta model is proto-efficient

By combining the results in Sections 6.1 to 6.4, we obtain our main result.

Theorem 6.25. *The (level-4 symmetric) theta model $\mathscr{A}$ is proto-efficient.*

Remark 6.26. We remark that, though we only exhibit one instantiation of our framework, it is plausible that others exist. Many of our algorithms could be extended to work in level $n > 2$., and one could also consider other representations of PPAVs, such as the module representation (e.g., [28]).

7 Interpolating polarised isogenies

In this section, we prove that isogenies can be *interpolated*. This interpolation result is generic, in the sense that it does not depend on the way we represent PPAVs (it can be applied to the theta model, or to any other efficient model).

Theorem 7.1 (Interpolation). *Let $\mathscr{A}$ be a proto-efficient model of PPAVs. There is an algorithm (Algorithm 6) which, on input*

- *two PPAVs X_1 and X_2 over $\mathbb{F}_q$ of dimension g (in $\mathscr{A}$-representation);*
- *two coprime integers N and N', with $N < N'$ and not divisible by p;*
- *points $P_1, \ldots, P_{2g} \in X_1(\overline{\mathbb{F}}_q)$ (in CRT $\mathscr{A}$-representation) generating $X_1[N']$;*
- *points $Q_1, \ldots Q_{2g} \in X_2(\overline{\mathbb{F}}_q)$ (in CRT $\mathscr{A}$-representation) generating $X_2[N']$,*

outputs an N-isogeny $\varphi : X_1 \to X_2$ (in $\mathscr{A}$-representation) such that $\varphi(P_i) = Q_i$ for all $i \in \{1, \ldots, 2g\}$ if it exists, and $\perp$ otherwise. The algorithm runs in time $(x+\ell)^{\mathrm{poly}(g)}$ where x is the length of the input and ℓ is the largest prime factor of N'. Furthermore, these properties uniquely determine φ as soon as $N' > 2\sqrt{2gN}$.

Proof. The polynomial running time follows as each step has polynomial complexity, and the number of iterations of each loop is bounded by $2g$ or $\#\operatorname{Aut}(X_1)$.

It remains to focus on the correctness of the algorithm. Thanks to Step 20, it is sufficient to consider the case where there exists an N-isogeny φ such that $\varphi(P_j) = Q_j$ for all j. The matrix A defined in Step 4 satisfies $AA^\dagger = A^\dagger A = m\mathbf{I}$, where $A^\dagger$ is the conjugate transpose of A and $\mathbf{I}$ is the 4×4 identity matrix.

Algorithm 6 Interpolating isogenies with respect to a PPAV model $\mathscr{A}$

Require: The input consists of:
- two PPAVs X_1 and X_2 over $\mathbb{F}_q = \mathbb{F}_{p^k}$ (in $\mathscr{A}$-representation);
- two coprime integers N and N', with $N < N'$ and not divisible by p;
- points $P_1, \dots, P_{2g} \in X_1(\overline{\mathbb{F}}_q)$ (in CRT $\mathscr{A}$-representation) such that $X_1[N'] = \langle P_1, \dots, P_{2g} \rangle$;
- points $Q_1, \dots Q_{2g} \in X_2(\overline{\mathbb{F}}_q)$ (in CRT $\mathscr{A}$-representation) such that $X_2[N'] = \langle Q_1, \dots, Q_{2g} \rangle$;

Ensure: An N-isogeny $\varphi : X_1 \to X_2$ (in $\mathscr{A}$-representation) such that $\varphi(P_i) = Q_i$ for all $i \in \{1, \dots, 2g\}$ if it exists, and $\perp$ otherwise.

$m \leftarrow N' - N$
Decompose $m = m_1^2 + m_2^2 + m_3^2 + m_4^2$. ▷ Using the algorithm from [29]
Compute X_1^4 and X_2^4 together with the coefficient-wise embeddings $\imath_i : X_1 \to X_1^4$ and $\jmath_i : X_2 \to X_2^4$, for $i \in \{1, \dots, 4\}$ ▷ using $\mathscr{A}$.`product`
Let $\alpha \in \mathrm{End}(X_1)$ with matrix representation $A := \begin{pmatrix} m_1 & -m_2 & -m_3 & -m_4 \\ m_2 & m_1 & m_4 & -m_3 \\ m_3 & -m_4 & m_1 & m_2 \\ m_4 & m_3 & -m_2 & m_1 \end{pmatrix}$
for i from 1 to $2g$, and j from 1 to 4 **do**
    $P_{i,j} \leftarrow \imath_j(P_i) = (0, \dots, P_i, \dots, 0) \in X_1^4$ ▷ Non-zero coefficient at j-th position
    $Q_{i,j} \leftarrow \jmath_j(Q_i) \in X_2^4$
    $R_{i,j} \leftarrow \alpha^\dagger(P_{i,j}) \in X_1^4$. ▷ using $\mathscr{A}$.`adjoint`, and $\mathscr{A}$.`eval` in CRT-representation
end for
Compute $X = X_1^4 \times X_2^4$ together with the coefficient-wise embeddings $\mathsf{inc}_i^j : X_j \to X$ and projections $\mathsf{pr}_i^j : X \to X_j$, for $i \in \{1, \dots, 4\}$ and $j \in \{1, 2\}$
$G \leftarrow \{(R_{i,j}, Q_{i,j}) \mid i \in \{1, \dots, 2g\}, j \in \{1, 2, 3, 4\}\}$
$(\psi' : X \to Y) \leftarrow \mathscr{A}$.`quotient_CRT`$(X, G)$ ▷ Proposition 4.3
$\delta \leftarrow \mathscr{A}$.`isomorphism`$(Y, X)$
if $\delta = \perp$ **then**
    return $\perp$
end if
$(\alpha_i)_{i=1}^k \leftarrow \mathscr{A}$.`automorphisms`$(X)$
for i from 1 to k **do**
    $\varphi \leftarrow -\mathsf{pr}_1^2 \circ \alpha_i \circ \delta \circ \psi' \circ \mathsf{inc}_1^1$ ▷ Using $\mathscr{A}$.{`scalar`, `composition`}
    if φ is an N-isogeny and $\varphi(P_j) = Q_j$ for all j **then** ▷ Using Corollary 4.12
        return φ
    end if
end for
return $\perp$

Therefore α is an m-isogeny. Using this, we construct the following (N, m)-isogeny diamond between $4g$-dimensional PPAV:

$$\begin{array}{ccc} X_1^4 & \xrightarrow{\varphi\mathbf{I}} & X_2^4 \\ {\scriptstyle\alpha}\uparrow & & \uparrow{\scriptstyle\alpha} \\ X_1^4 & \xrightarrow[\varphi\mathbf{I}]{} & X_2^4 \end{array}$$

Consider the isogeny $\psi = \begin{pmatrix} \alpha & \varphi^\dagger \mathbf{I} \\ -\varphi\mathbf{I} & \alpha^\dagger \end{pmatrix} : X_1^4 \times X_2^4 \to X_1^4 \times X_2^4$. We have $\alpha\alpha^\dagger + \varphi^\dagger\mathbf{I}\varphi\mathbf{I} = (N+m)\mathbf{I} = N'\mathbf{I}$, so ψ is an N'-isogeny. Note that $m = N' - N$ is coprime to N (since the latter is coprime to N'), so

$$\begin{aligned}\ker(\psi) &= \{(P,Q) \in X_1^4 \times X_2^4 \mid \alpha(P) = -\varphi^\dagger\mathbf{I}(Q) \text{ and } \alpha^\dagger(Q) = \varphi\mathbf{I}(P)\} \\ &= \{(\alpha^\dagger(P), \varphi\mathbf{I}(P)) \mid P \in X_1^4[N']\} = \langle\{(\alpha^\dagger(P_{i,j}), Q_{i,j}) \mid i \le 2g, j \le 4\}\rangle.\end{aligned}$$

Therefore, the isogeny ψ' computed in Step 10 is an N'-isogeny with same kernel as ψ. In particular, for any isomorphism δ from the codomain of ψ' to the codomain of ψ, there exists an automorphism $\alpha \in \mathrm{Aut}(X)$ such that $\psi = \alpha\circ\delta\circ\psi'$. The last loop performs an exhaustive search over these automorphisms until a match is found. The returned N-isogeny φ satisfies $\varphi(P_j) = Q_j$ for all j. Since we assume the existence of such a φ, this exhaustive search will find it and return.

It remains to prove that when $N' > 2\sqrt{2gN}$, the isogeny φ is the unique N-isogeny satisfying $\varphi(P_j) = Q_j$. Denote by φ^* another such N-isogeny, we get $\|\varphi^* - \varphi\| \le \|\varphi^*\| + \|\varphi\| = 2\sqrt{2gN}$, since $\|\varphi\|^2 := \mathrm{tr}(\varphi\varphi^\dagger) = \mathrm{tr}([N]) = 2gN$.

Moreover, naming $f := \varphi^* - \varphi$, if $\varphi \neq \varphi^*$ then $f \neq 0$ and Lemma 4.4 gives the lower bound $N' \le \|\varphi^* - \varphi\| \le 2\sqrt{2gN}$. In particular, when $2\sqrt{2gN} < N'$, we get a contradiction, so $\varphi = \varphi^*$. □

Definition 7.2 (Interpolation representation of isogenies). *Let $\mathscr{A}$ be a proto-efficient model of PPAVs. Consider the same notation as Theorem 7.1, and suppose that the N-isogeny φ exists and that $N' > 2\sqrt{2gN}$. Let ℓ be the largest prime factor of N', and $\mathbb{F}_q$ the field of definition of φ. Then the tuple $(\underline{X_1}, \underline{X_2}, \mathbb{F}_q, N, N', \mathtt{1}^\ell, (\underline{P_i})_i, (\underline{Q_i})_i)$ (where each component is in $\mathscr{A}$-representation) is called an* interpolation $\mathscr{A}$-representation of φ.

Lemma 7.3 (Universality of the interpolation representation). *For any proto-efficient model for PPAVs $\mathscr{A}$, there is a polynomial time algorithm which, on input an $\mathscr{A}$-representation F of an N-isogeny φ, outputs an interpolation $\mathscr{A}$-representation of φ of length $(\log(q) + \log(N))^{\mathrm{poly}(\dim(X))}$. Furthermore, the output in completely determined by $\mathscr{A}.\mathtt{domcodom}(F)$ and by $\varphi = \mathrm{mrph}(F)$.*

Remark 7.4. With these properties, the resulting interpolation $\mathscr{A}$-representation is *compact* and *canonical* in the sense of axiom A29, *Canonical and compact form for N-isogenies.*

Proof. The algorithm is given in Algorithm 7. Let us prove that it runs in the claimed time and the output is indeed compact (i.e., of length $(\log(q) + \log(N))^{\dim(X)}$) and canonical (i.e., it is fully determined by $\varphi = \mathrm{mrph}(F)$ and by the output of $\mathscr{A}.\mathtt{domcodom}(F)$).

Let us first explain Step 7. Given the domain X and the prime ℓ_i, the algorithm $\mathscr{A}.\mathtt{torsion}$ outputs the list of all points $(R_j)_{j=1}^{\ell_i^{2g}} = X[\ell_i]$ (in polynomial time in the length of X and in ℓ_i^{2g}). We need to show that we can extract from this list a basis of $X[\ell_i]$, and that this choice of basis is uniquely determined by

Algorithm 7 Constructing a canonical interpolation $\mathscr{A}$-representation

Require: The input consists of an N-isogeny $\varphi : X \to Y$ in $\mathscr{A}$-representation;
Ensure: An interpolation $\mathscr{A}$-representation of φ.
1: $N \leftarrow \mathscr{A}.\texttt{is_polarized}(\varphi)$
2: $g \leftarrow \mathscr{A}.\texttt{dimension}(X)$
3: $(X, Y) \leftarrow \mathscr{A}.\texttt{domcodom}(\varphi)$ the domain and codomain in $\mathscr{A}$-representation
4: $(\ell_j)_{j=1}^k \leftarrow$ the list of smallest primes (in increasing order) coprime with pN such that $\prod_{j=1}^k \ell_j > \max(N, 2\sqrt{2gN})$.
5: $N' \leftarrow \prod_{j=1}^k \ell_j$.
6: **for** $j = 1, \ldots, k$ **do**
7: $\quad (P_i^j)_{i=1}^{2g} \leftarrow$ a deterministically chosen basis of $X[\ell_i]$ ▷ Using $\mathscr{A}.\texttt{torsion}$, and see the proof for a deterministic choice
8: $\quad (Q_i^j)_{i=1}^{2g} \leftarrow (\varphi(P_i^j))_{i=1}^{2g}$ in canonical form ▷ Using $\mathscr{A}.\texttt{eval}$ and $\mathscr{A}.\texttt{canonical}$
9: **end for**
10: **for** $j = 1, \ldots, 2g$ **do**
11: $\quad P_i \leftarrow (P_i^j)_{j=1}^k$ the CRT representation of $P_i = \sum_{j=1}^k P_i^j \in X[N']$.
12: **end for**
13: **return** $(X, Y, \mathbb{F}_{q^k}, N, N', \mathtt{1}^{\ell_k}, (P_i)_{i=1}^{2g}, (Q_i)_{i=1}^{2g})$

X and ℓ_i (even though $\mathscr{A}.\texttt{torsion}$ might be a randomized algorithm). Let us sketch a very naive (yet sufficient) approach. Thanks to $\mathscr{A}.\texttt{canonical}$, we can assume that the representation of all points R_j are in canonical form. We can then order the list $(R_j)_j$ in lexicographic order (as binary strings). Now, extract a basis by keeping only the points R_j such that $R_j \notin \langle R_1, \ldots, R_{j-1}\rangle$ (this can be done naively since we only need the running time to be polynomial in ℓ_i^{2g}).

This proves that the choice of bases $(P_i^j)_{i=1}^{2g}$ are determined by the (representation of the) domain X. At Step 8, the points Q_i^j are determined by P_i^j and φ (since they are in canonical form, they do not depend on the choice of an $\mathscr{A}$-representation of φ). The other steps have no randomness, and no dependence on the $\mathscr{A}$-representation of φ, so the output of the algorithm is canonical.

By the Prime Number Theorem, at Step 4, the length k and the entries ℓ_i are polynomially bounded in $\log N$, which implies the claimed time. To bound the output length, observe that

$$\begin{aligned}\operatorname{length}(P_i) &= O\left(\sum_{j=1}^{k} \operatorname{length}(P_i^j)\right) = O\left(\sum_{j=1}^{k} \operatorname{length}(\mathscr{A}.\texttt{torsion}(X, \mathtt{1}^{\ell_j}))\right) \\ &= (\operatorname{length}(X), \max_j \ell_j)^{\operatorname{poly}(\dim(X))} = (\log(q) + \log(N))^{\operatorname{poly}(\dim(X))},\end{aligned}$$

where the first equality is a property of the CRT representation, the second uses that the list $(P_i^j)_j$ is a sublist of the output of $\mathscr{A}.\texttt{torsion}(X, \mathtt{1}^{\ell_j})$, the third uses that the output of a polynomial time algorithm has length polynomially bounded in the length of the input, and the last uses the compacity of the $\mathscr{A}$-representation of X. It immediately follows that the output of Algorithm 7 has length $(\log(q) + \log(N))^{\operatorname{poly}(\dim(X))}$. □

Theorem 7.5. *For any proto-efficient model for PPAVs $\mathscr{A}$, there is an efficient model for PPAVs $\mathscr{A}'$ and a polynomial time algorithm transforming $\mathscr{A}$-representations to $\mathscr{A}'$ representations.*

Proof. Recall that an efficient model is simply a proto-efficient model that additionally satisfies axiom A29, *canonical and compact form for N-isogenies*. This canonical and compact form is given by interpolation representations. Therefore, we construct $\mathscr{A}'$ from $\mathscr{A}$ by: (1) *adding* a new type of morphism representations[5] (namely, interpolation $\mathscr{A}$-representations) to S_{mrph}; (2) defining the algorithm $\mathscr{A}'.\texttt{canonical}$ for morphisms as Algorithm 7; and (3) for each algorithm $\mathscr{A}.\texttt{x}$ that takes a morphism as input, the algorithm $\mathscr{A}'.\texttt{x}$ is the same algorithm, where, whenever the input is in interpolation representation, we first run the interpolation algorithm (Algorithm 6) to convert it to an $\mathscr{A}$-representation.

By Lemma 7.3, Algorithm 7 satisfies the *Efficiency* axiom A1, and the output is *compact* and *canonical* in the sense of axiom A29. The conversion from $\mathscr{A}$-representations to $\mathscr{A}'$-representations is the identity (the new type of morphism representation is only added, and does not *replace* the previous ones). □

Remark 7.6. Note that this also proves that even within the proto-efficient model $\mathscr{A}$, each morphism has a compact representation: given any $\underline{\varphi} \in S_{\text{mrph}}$, one can obtain its interpolation data with Algorithm 7 (which is compact, but is not in S_{mrph}), then turn it back to an $\mathscr{A}$-representation with Algorithm 6. Since Algorithm 6 runs in polynomial time, and the interpolation data is compact, the resulting $\mathscr{A}$-representation is compact (because polynomial time algorithms produce outputs of polynomial length).

Then, one might wonder why we need to extend the set S_{mrph} to a larger set S'_{mrph} (injecting interpolation representations into the model). The reason is for the "canonical form" property: the interpolation data is both compact *and* canonical. Once processed through Algorithm 6, it remains compact, but is no longer canonical because it depends on the choice of decomposition $m = m_1^2 + m_2^2 + m_3^2 + m_4^2$. There is no known polynomial time algorithm to decompose an integer into a sum of squares in a deterministic way.

Corollary 7.7. *The theta model, together with isogeny interpolation, is an efficient model for PPAVs.*

Proof. This follows immediately from Theorem 6.25 (the theta model is a proto-efficient model for PPAVs) and Theorem 7.5 (interpolation enhances any proto-efficient model to an efficient model).

[5] Formally, one might need to fiddle with binary representations, for instance setting $S'_{\text{mrph}} = (\texttt{0} + S_{\text{mrph}}) \sqcup (\texttt{1} + S''_{\text{mrph}})$ where $+$ is the binary concatenation, and S''_{mrph} is some encoding of interpolation $\mathscr{A}$-representations.

References

[1] Samuele Anni, Gaetan Bisson, Annamaria Iezzi, Elisa Lorenzo García, and Benjamin Wesolowski. *On the computation of endomorphism rings of abelian surfaces over finite fields*. 2025. arXiv: 2503.08925 [math.NT]. URL: https://arxiv.org/abs/2503.08925.

[2] Andrea Basso, Giacomo Borin, Wouter Castryck, Maria Corte-Real Santos, Riccardo Invernizzi, Antonin Leroux, Luciano Maino, Frederik Vercauteren, and Benjamin Wesolowski. "PRISM: Simple and Compact Identification and Signatures from Large Prime Degree Isogenies". In: *Public-Key Cryptography – PKC 2025*. Ed. by Tibor Jager and Jiaxin Pan. Cham: Springer Nature Switzerland, 2025, pp. 300–332. ISBN: 978-3-031-91826-1.

[3] Andrea Basso and Luciano Maino. "POKÉ: A Compact and Efficient PKE from Higher-Dimensional Isogenies". In: *Advances in Cryptology – EUROCRYPT 2025, Part II*. Ed. by Serge Fehr and Pierre-Alain Fouque. Vol. 15602. Lecture Notes in Computer Science. Madrid, Spain: Springer, Cham, Switzerland, 2025, pp. 94–123. DOI: 10.1007/978-3-031-91124-8_4.

[4] Andrea Basso, Luciano Maino, and Giacomo Pope. "FESTA: Fast Encryption from Supersingular Torsion Attacks". In: *Advances in Cryptology – ASIACRYPT 2023, Part VII*. Ed. by Jian Guo and Ron Steinfeld. Vol. 14444. Lecture Notes in Computer Science. Guangzhou, China: Springer, Singapore, Singapore, 2023, pp. 98–126. DOI: 10.1007/978-981-99-8739-9_4.

[5] Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. "Faster computation of isogenies of large prime degree". In: *ANTS XIV—Proceedings of the Fourteenth Algorithmic Number Theory Symposium*. Vol. 4. Open Book Series. Math. Sci. Publ., Berkeley, CA, 2020, pp. 39–55. ISBN: 978-1-935107-08-8; 978-1-935107-07-1. DOI: 10.2140/obs.2020.4.39.

[6] Wouter Castryck and Thomas Decru. "An Efficient Key Recovery Attack on SIDH". In: *Advances in Cryptology – EUROCRYPT 2023, Part V*. Ed. by Carmit Hazay and Martijn Stam. Vol. 14008. Lecture Notes in Computer Science. Lyon, France: Springer, Cham, Switzerland, 2023, pp. 423–447. DOI: 10.1007/978-3-031-30589-4_15.

[7] Wouter Castryck, Thomas Decru, and Benjamin Smith. "Hash functions from superspecial genus-2 curves using Richelot isogenies". In: *Journal of Mathematical Cryptology* 14.1 (2020), pp. 268–292. DOI: doi:10.1515/jmc-2019-0021.

[8] Romain Cosset. "Applications des fonctions thêta à la cryptographie sur courbes hyperelliptiques." PhD thesis. Université Henri Poincaré-Nancy I, 2011.

[9] Pierrick Dartois. "Fast computation of 2-isogenies in dimension 4 and cryptographic applications". In: *Journal of Algebra* 683 (2025), pp. 449–514. ISSN: 0021-8693. DOI: https://doi.org/10.1016/j.jalgebra.2025.06.033.

[10] Pierrick Dartois. "Fast computation of higher dimensional isogenies for cryptographic applications". PhD thesis. Université de Bordeaux, 2025.

[11] Pierrick Dartois, Luciano Maino, Giacomo Pope, and Damien Robert. "An Algorithmic Approach to (2, 2)-Isogenies in the Theta Model and Applications to Isogeny-Based Cryptography". In: *Advances in Cryptology – ASIACRYPT 2024, Part III*. Ed. by Kai-Min Chung and Yu Sasaki. Vol. 15486. Lecture Notes in Computer Science. Kolkata, India: Springer, Singapore, Singapore, 2024, pp. 304–338. DOI: 10.1007/978-981-96-0891-1_10.

[12] Max Duparc. *Superglue: Fast formulae for (2,2)-gluing isogenies*. Cryptology ePrint Archive, Paper 2025/736. 2025. URL: https://eprint.iacr.org/2025/736.

[13] Luca De Feo, David Jao, and Jérôme Plût. "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies". In: *Journal of Mathematical Cryptology* 8.3 (2014), pp. 209–247. DOI: doi:10.1515/jmc-2012-0015.

[14] E. Victor Flynn and Yan Bo Ti. "Genus Two Isogeny Cryptography". In: *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*. Ed. by Jintai Ding and Rainer Steinwandt. Chongqing, China: Springer, Cham, Switzerland, 2019, pp. 286–306. DOI: 10.1007/978-3-030-25510-7_16.

[15] Ernst Kani. "The number of curves of genus two with elliptic differentials". In: *Journal für die reine und angewandte Mathematik* 1997.485 (1997), pp. 93–122. DOI: doi:10.1515/crll.1997.485.93. URL: https://doi.org/10.1515/crll.1997.485.93.

[16] Jean Kieffer. "Isogeny graphs of abelian varieties over finite fields". In: (2024). URL: Isogenygraphsofabelianvarietiesoverfinitefields.

[17] Sabrina Kunzweiler, Yan Bo Ti, and Charlotte Weitkämper. "Secret Keys in Genus-2 SIDH". In: *SAC 2021: 28th Annual International Workshop on Selected Areas in Cryptography*. Ed. by Riham AlTawy and Andreas Hülsing. Vol. 13203. Lecture Notes in Computer Science. Virtual Event: Springer, Cham, Switzerland, 2022, pp. 483–507. DOI: 10.1007/978-3-030-99277-4_23.

[18] Antonin Leroux. "Verifiable Random Function from the Deuring Correspondence and Higher Dimensional Isogenies". In: *Advances in Cryptology – EUROCRYPT 2025, Part VII*. Ed. by Serge Fehr and Pierre-Alain Fouque. Vol. 15607. Lecture Notes in Computer Science. Madrid, Spain: Springer, Cham, Switzerland, 2025, pp. 167–194. DOI: 10.1007/978-3-031-91098-2_7.

[19] David Lubicz and Damien Robert. "Computing isogenies between abelian varieties". In: *Compositio Mathematica* 148.5 (2012), pp. 1483–1515.

[20] David Lubicz and Damien Robert. "Computing separable isogenies in quasi-optimal time". In: *LMS Journal of Computation and Mathematics* 18.1 (2015), pp. 198–216. DOI: 10.1112/S146115701400045X.

[21] David Lubicz and Damien Robert. "Fast change of level and applications to isogenies". In: *Research in Number Theory* 9.1 (2023), Paper No. 7, 28. ISSN: 2522-0160,2363-9555. DOI: 10.1007/s40993-022-00407-9.

[22] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. "A Direct Key Recovery Attack on SIDH". In: *Advances in Cryptology – EUROCRYPT 2023, Part V*. Ed. by Carmit Hazay and Martijn Stam. Vol. 14008. Lecture Notes in Computer Science. Lyon, France: Springer, Cham, Switzerland, 2023, pp. 448–471. DOI: 10.1007/978-3-031-30589-4_16.

[23] J. S. Milne. "Abelian Varieties". In: *Arithmetic Geometry*. Ed. by Gary Cornell and Joseph H. Silverman. New York, NY: Springer New York, 1986, pp. 103–150. ISBN: 978-1-4613-8655-1. DOI: 10.1007/978-1-4613-8655-1_5. URL: https://doi.org/10.1007/978-1-4613-8655-1_5.

[24] J. S. Milne. *Algebraic Geometry (v5.00)*. https://www.jmilne.org/math/CourseNotes/AV110.pdf. Course Notes, available online. 2020.

[25] David Mumford. *Abelian varieties*. Oxford university press, 1974.

[26] David Mumford. "On the equations defining abelian varieties. I". In: *Inventiones mathematicae* 1.4 (1966), pp. 287–354.

[27] David Mumford. "On the equations defining abelian varieties. I". In: *Inventiones Mathematicae* 1 (1966), pp. 287–354. DOI: 10.1007/BF01389737. URL: https://doi.org/10.1007/BF01389737.

[28] Aurel Page, Damien Robert, and Julien Soumier. *On the conversion of module representations for higher dimensional supersingular isogenies*. Cryptology ePrint Archive, Paper 2026/276. 2026. URL: https://eprint.iacr.org/2026/276.

[29] Michael O. Rabin and Jeffery O. Shallit. "Randomized algorithms in number theory". In: *Communications on Pure and Applied Mathematics* 39.S1 (1986), S239–S256. DOI: https://doi.org/10.1002/cpa.3160390713. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160390713. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160390713.

[30] Damien Robert. "Breaking SIDH in Polynomial Time". In: *Advances in Cryptology – EUROCRYPT 2023, Part V*. Ed. by Carmit Hazay and Martijn Stam. Vol. 14008. Lecture Notes in Computer Science. Lyon, France: Springer, Cham, Switzerland, 2023, pp. 472–503. DOI: 10.1007/978-3-031-30589-4_17.

[31] Damien Robert. "Efficient algorithms for abelian varieties and their moduli spaces". English. In: (2021). HDR, Institut de Mathématiques de Bordeaux (IMB). URL: https://oskar-bordeaux.fr/handle/20.500.12278/191315.

[32] Damien Robert. "Fonctions thêta et applications à la cryptographie". PhD thesis. Université Henri Poincaré-Nancy I, 2010.

[33] Damien Robert. "Some applications of higher dimensional isogenies to elliptic curves (overview of results)". In: (2023).

[34] Jacques Vélu. "Isogénies entre courbes elliptiques". In: *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences. Séries A et B* 273 (1971), A238–A241. ISSN: 0151-0509.