SLVer Bullet: Straight-Line Verification for Bulletproofs

Brandon Goodell, Rigo Salazar, Freeman Slaughter, Luke Szramowski

Cypher Stack

June 28, 2025

Disclaimer

This paper is not yet completed; future updates to this document are forthcoming soon.

1 Introduction

In [Eag22], Eagen proposed a method for checking whether a sum of points in an elliptic curve group has been computed correctly. Eagen's method verifiably checks elliptic curve group operations only with linear combinations in the base field, allowing very general proofs to be encoded in inner product argument systems and rank-1 constraint systems (R1CS). We call this method "straight-lining," due to how it utilizes straight lines in the verification procedure. We present a method based on [Eag22] for straight-lining cryptographic protocols, offloading computational costs faced by verifiers to provers, which is useful when verification use light-weight devices or when verification must be performed repeatedly.

Although the iterative witness construction algorithm proposed in [Eag22] is correct, [Eag22] is rather informal, lacking precise protocol descriptions, claims, proofs, or efficiency analyses. In [Para], Parker used Eagen's method in a R1CS s to verify scalar multiplication of group elements, proposing a protocol based on Eagen's arguments. In [Bas], Bassa contributed towards formalizing Eagen's method, explicitly describing Parker's proof of scalar multiplication and sketching arguments towards proofs of soundness.

However, the soundness arguments in [Bas] are not without obstacles. In [Eag22], rational solutions to certain systems of polynomial equations are assumed to exist, however this assumption is not wholly accurate. Indeed, these equations do not admit rational solutions in general, so the verification equations proposed in [Eag22] and studied in [Bas] remain unjustified. Bassa lifted to the surface of pairs of elliptic curve group points, which partially resolves this problem.

Nonetheless, Bassa's amended approach results in alterations to the verification equations of [Eag22], which he does not substantiate. We formalize the correct verification equations, and consider the security of Eagen's approach. We present a corrected version of the general protocol,

and explicitly compute the completeness and soundness errors. We remark on how to complete the scheme, and our soundness error beats previous estimates in [Eag22] and [Bas].

We show how the corrected scheme can be used to verify scalar multiplication as in Parker's proposed protocol. We then apply this approach to zero-knowledge proof systems to illustrate the computational advantage offered by divisors. In particular, the prover can pre-compute division operations for the verifier, which are the computational bottlenecks on the verifier's side.

We transform the interactive Schnorr identification protocol [Sch91] and interactive Bulletproofs range-proving protocol [BBB$^+$17] in order to highlight the computational benefits of this corrected divisors procedure, then benchmark its efficiency. We choose Schnorr and Bulletproofs because these protocols are simple, secure, popular, and readily applicable in cryptocurrencies such as Monero or Salvium.

1.1 Change log

This document will be updated occasionally, especially when security-sensitive results come to light. We summarize such changes here.

- 17 June 2025. Add proof of Lemma 3.2.
- 11 June 2025. Initial preprint.

2 Notation and preliminary definitions

We begin assuming the reader has knowledge of basic algebra concepts related to groups, rings, and fields. Our notation and background primarily follows [Sil09], but we depart notationally in a few notable ways, especially with regard to divisors. Let $\mathbb{N} = \{1, 2, 3, \ldots\}$ and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. All sets herein are finite unless otherwise specified.

2.1 Polynomials and rational functions

For a set S, we use $x \xleftarrow{\$} S$ to mean that the element x is an independent uniformly-distributed sample from set S. Let $\mathbb{1}$ and $\mathbb{0}$ be the vectors of all 1's and 0's, respectively. For a ring R, we write $R^\times = R \setminus \{0\}$ to denote the multiplicative subgroup. Let p be a prime, K a finite field with $\mathrm{char}(K) = p$, algebraic closure $\overline{K}$, and multiplicative subgroup $K^\times = \{x \in K \mid x \neq 0\}$. Let X, Y, Z be indeterminates over $\overline{K}$. Finite linear combinations in products of X, Y, and Z are *polynomials*. We use the following usual notation for rings of polynomials over $\overline{K}$ and K.

- $\overline{K}[X,Y] = \left\{ \sum_{i=0}^{n} \sum_{j=0}^{m} a_{i,j} X^i Y^j \mid a_{i,j} \in \overline{K} \right\}$
- $K[X,Y] = \left\{ \sum_{i=0}^{n} \sum_{j=0}^{m} a_{i,j} X^i Y^j \mid a_{i,j} \in K \right\}$
- $\overline{K}[X,Y,Z] = \left\{ \sum_{i=0}^{n} \sum_{j=0}^{m} \sum_{k=0}^{l} a_{i,j,k} X^i Y^j Z^k \mid a_{i,j,k} \in \overline{K} \right\}$

- $K[X,Y,Z] = \left\{ \sum_{i=0}^{n} \sum_{j=0}^{m} \sum_{k=0}^{l} a_{i,j,k} X^i Y^j Z^k \mid a_{i,j,k} \in K \right\}$

For $f \in K[X,Y]$, say $f = \sum_i \sum_j a_{ij} X^i Y^j$, define $\deg(f) = \max\{i+j \mid a_{ij} \neq 0\}$ (and similarly for the other polynomial rings). Each of these rings (say R) admits a structure-preserving map $\deg : R \to \mathbb{N}_0 = \mathbb{N} \cup \{0\}$ such that $\deg(fg) = \deg(f) + \deg(g)$, $\deg(f+g) \leq \max\{\deg(f), \deg(g)\}$. Also, $\deg(f) = 0$ if and only if $f = 0$.

If $d \in \mathbb{N}_0$ and $f \in \overline{K}[X,Y]$, and there exists some $\lambda \in \overline{K}^\times$ such that $f(\lambda X, \lambda Y) = \lambda^d f(X,Y)$, then we say f is *homogeneous with degree* d; otherwise, *inhomogeneous*. Define an equivalence relation on the subset of homogeneous polynomials in $\overline{K}[X,Y,Z]$ such that every $f(X,Y,Z)$ relates to $f(\lambda X, \lambda Y, \lambda Z)$ for every $\lambda \in \overline{K}^\times$. Denote the equivalence class of f under this equivalence relation with $f(X:Y:Z)$, and the set of equivalence classes under this relation with $\overline{K}[X:Y:Z]$. Define $\deg : \overline{K}[X:Y:Z] \to \mathbb{N}_0$ by declaring $\deg(f(X:Y:Z)) = \deg(f(X,Y,Z))$. Similarly define $K[X:Y:Z]$ and its degree function. The rings $\overline{K}[X,Y]$, $K[X,Y]$, $\overline{K}[X,Y,Z]$, $K[X,Y,Z]$, $\overline{K}[X:Y:Z]$, and $K[X:Y:Z]$ are all integral domains.

We also define the respective field extensions of rational functions $\overline{K}(X,Y)$, $K(X,Y)$, $\overline{K}(X,Y,Z)$, and $K(X,Y,Z)$ as usual. We define the multiplicities of roots and poles of elements of $K(X,Y)$ as usual; multiplicity theory is a rich rabbit hole preceding Zariski, so we do not go into further detail. Given $P = (x,y) \in \overline{K}^2$, there exists a map $\mathrm{ord}_P : \overline{K}(X,Y) \to \mathbb{Z}$ called the *order of vanishing* at P defined as follows.

$$\mathrm{ord}_P(f) = \begin{cases} m > 0; & P \text{ is a root of } f \text{ with multiplicity } m \\ m < 0; & P \text{ is a pole of } f \text{ with multiplicity } |m| = -m \\ 0; & \text{otherwise} \end{cases} \tag{1}$$

We define $\mathrm{ord}_P(f)$ for subrings of $\overline{K}(X,Y)$ by restricting ord_P.

In the sequel, we use the distinguished family of polynomials parameterized by $\alpha, \beta \in K^2$ of the form $e = Y^2 - X^3 - \alpha X - \beta \in K[X,Y]$ such that $4\alpha^3 + 27\beta^2 \neq 0$; these are exactly the non-singular elliptic curves.

2.2 Affine and projective planes

Define $\mathbb{A}^2(\overline{K}) = \left\{(x,y) \in \overline{K}^2\right\}$, $\mathbb{A}^3(\overline{K}) = \left\{(x,y,z) \in \overline{K}^3\right\}$, $\mathbb{A}^2(K) = \left\{(x,y) \in K^2\right\}$, and $\mathbb{A}^3(K) = \left\{(x,y,z) \in K^3\right\}$. We say the elements of $\mathbb{A}^2(\overline{K})$ are *affine points* and the elements of $\mathbb{A}^2(K)$ are *K-rational* affine points. For any ideal $I \subseteq \overline{K}[X,Y]$, the *affine vanishing set* of I is $v_a(I) = \left\{(x,y) \in \mathbb{A}^2(\overline{K}) \mid f(x,y) = 0 \text{ for all } f \in I\right\}$. If $f(X,Y) \in \overline{K}[X,Y]$, we define the affine vanishing set for f as the affine vanishing set for the principally generated $I = (f)$, which we denote with $v_a(f)$.

Given affine vanishing set V, let $I(V) = \left\{f \in \overline{K}[X,Y] \mid f(x,y) = 0 \text{ for all } (x,y) \in V\right\}$ be the *ideal of* V. It is easy to check that $I(V)$ is an ideal of $\overline{K}[X,Y]$. We say an affine vanishing set V is *defined over* K, denoted with shorthand V/K, if $I(V)$ is generated as an ideal in $\overline{K}[X,Y]$ by a subset of $K[X,Y]$. For any affine vanishing set V, let $I(V/K) = I(V) \cap K[X,Y]$. It is easy to show

that $I(V/K)$ is an ideal. Moreover, V is defined over K if and only if $I(V)$ is generated as an ideal in $\overline{K}[X,Y]$ by $I(V/K)$, i.e. $I(V) = I(V/K)\overline{K}[X,Y]$.

We say V is an *affine variety* if $I(V)$ is a prime ideal in $\overline{K}[X,Y]$. The *affine coordinate ring* of an affine variety V/K is the quotient ring $K[V] = \frac{K[X,Y]}{I(V/K)}$. The *function field* of V is the field of fractions associated with $K[V]$, i.e. the set of all formal fractions $\frac{f}{g}$ where $f, g \in K[V]$ and $g \neq 0$, under an equivalence relation defined by relating $\frac{f_1}{g_1}$ to $\frac{f_2}{g_2}$ if and only if $f_1 g_2 - f_2 g_1 = 0$. We denote the function field $K(V)$.

Define an equivalence relation on $\mathbb{A}^3(\overline{K})\setminus\{\mathbb{0}\}$ by relating (x,y,z) to (x',y',z') if and only if there exists $\lambda \in \overline{K}^\times$ such that $(x',y',z') = (\lambda x, \lambda y, \lambda z)$. Denote the equivalence class of (x,y,z) with $(x:y:z)$. Use $\mathbb{P}^2(\overline{K})$ to denote the set of equivalence classes under this relation. Similarly define $\mathbb{P}^2(K)$ in the evident way. We say the elements of $\mathbb{P}^2(\overline{K})$ are *projective points* and the elements of $\mathbb{P}^2(K)$ are the *K-rational* projective points. For any homogeneous ideal $I \subseteq \overline{K}[X,Y,Z]$, the *projective vanishing set* of I is $v_p(I) = \big\{(x:y:z) \in \mathbb{P}^2(\overline{K}) \mid f(x,y,z) = 0 \text{ for all } f \in I\big\}$. Since the generating set for I contains only homogeneous polynomials, the presence of any representative in $(x:y:z)$ which kills all $f \in I$ implies all representatives kill all $f \in I$. For any $f \in \overline{K}[X,Y,Z]$, we define the projective vanishing set for f as the projective vanishing set for the ideal $I = (f)$, and we use the notation $v_p(f)$.

Given a projective vanishing set V, we overload notation and define $I(V)$ to be the homogeneous ideal of $\overline{K}[X:Y:Z]$ generated by the set $\big\{f \in \overline{K}[X:Y:Z] \mid f(x:y:z) = 0 \text{ for all } (x:y:z) \in V\big\}$. We say a projective vanishing set V is *defined over* K, denoted with shorthand V/K, if $I(V)$ is generated as an ideal in $\overline{K}[X:Y:Z]$ by a subset of $K[X:Y:Z]$. We say V is a *projective variety* if its homogeneous ideal $I(V)$ is a prime ideal in $\overline{K}[X,Y,Z]$. The *projective coordinate ring* of an projective variety V/K is the quotient ring $K[V] = \frac{K[XYZ]}{I(V/K)}$.

Affine points $P = (x,y) \in \mathbb{A}^2(\overline{K})$ (or $\mathbb{A}^2(K)$, respectively) naturally map to projective points $P^* = (x:y:1) \in \mathbb{P}^2(\overline{K})$ (or $\mathbb{P}^2(K)$, respectively). This map is injective, but $\mathbb{P}^2$ contains many copies of $\mathbb{A}^2$ and this map is not unique. Projective points $P = (x:y:z) \in \mathbb{P}^2(\overline{K})$ with $z \neq 0$ naturally map to affine points $P_* = (\frac{x}{z}, \frac{y}{z}) \in \mathbb{A}^2(\overline{K})$ in a similar way, but surjectively. The same statement as above holds with $\mathbb{A}^2(K)$ and $\mathbb{P}^2(K)$ replacing their counterparts everywhere.

There exists natural *homogenization* maps carrying $f \in K[X,Y]$ (or $f \in \overline{K}[X,Y]$) to a homogeneous $f^* \in K[X,Y,Z]$ (or $f^* \in \overline{K}[X,Y,Z]$, respectively) where $f^* = Z^{\deg(f)} f(\frac{X}{Z}, \frac{Y}{Z})$. This map is injective. We can also dehomogenize a homogeneous polynomial $g \in K[X:Y:Z]$ (or $\overline{K}[X,Y,Z]$) to a polynomial in the affine coordinate ring $g \mapsto g_*(X,Y) = g(X:Y:1) \in K[X,Y]$ (or $\overline{K}[X,Y]$, respectively). This map is surjective. The distinguished polynomial $e(X,Y) = Y^2 - X^3 - \alpha X - \beta$ has homogenization $e^*(X,Y) = Y^2Z - X^3 - \alpha XZ^2 - \beta Z^3$. The map $(x,y) \hookrightarrow (x:y:1)$ sends zeroes of $f \in K[X,Y]$ to zeroes of $f^* \in K[X:Y:Z]$.

Using these maps, we can translate $\mathbb{A}^2$ to $\mathbb{P}^2$ and back again. In this way, we define *projective closure* of an affine variety V as the projective vanishing set $\overline{V}$ such that $I(\overline{V}) = \{f^* \mid f \in I(V)\}$. We also define the function field for a projective variety $V \subseteq \mathbb{P}^2$ as the function field for the affine

restriction[1].

2.3 Plane curves

A *plane affine curve* $\mathcal{F}$ is just the affine vanishing set of some $f \in \overline{K}[X,Y]$, i.e. $\mathcal{F} = v_a(f)$. Define $\deg(\mathcal{F}) = \deg(f)$. Plane affine curves may have any of the properties of affine vanishing sets discussed above. A *plane projective curve* $\mathcal{F}$ is just the projective vanishing set of some homogeneous $f \in \overline{K}[X:Y:Z]$, i.e. $\mathcal{F} = v_p(f)$. Define $\deg(\mathcal{F}) = \deg(f)$. Plane projective curves may have any of the properties of projective vanishing sets discussed above.

Recall the distinguished family of polynomials $e = Y^2 - X^3 - \alpha X - \beta \in K[X,Y]$ such that $4\alpha^3 + 27\beta^2 \neq 0$. Define $\mathcal{E} = v_p(e)$ as the plane projective curve of the homogenization e^*. It is easy to show that the only $(x:y:z) \in \mathcal{E}$ with $z = 0$ is $(0:1:0)$. We call this the *point at infinity* on $\mathcal{E}$, and we denote $O = (0:1:0)$. It is also easy to show that $\mathcal{E}$ is a projective variety, $I(\mathcal{E}/K) = (e) \subseteq K[X:YZ]$ is the associated ideal over K, and the associated function field is as follows.

$$K(\mathcal{E}) = \left\{ \frac{f(X,Y) + I(\mathcal{E}/K)}{g(X,Y) + I(\mathcal{E}/K)} \mid f, g \in K[X,Y], g \notin I(\mathcal{E}/K) \right\} \tag{2}$$

Except in the case of confusion or for clarification, we often omit the ideal $I(\mathcal{E}/K)$ from our notation in the sequel, with the understanding that the numerator and denominator of every element in the function field is a coset modulo this ideal.

However, representing a function field element with coefficient representations of two arbitrary bivariate polynomials is expensive. Indeed if $m = \deg(f)$ and f is bivariate, then we can write $f = \sum_{i=0}^{m} \sum_{j=0}^{i} c_{i,j} X^{i-j} Y^j$, requiring $1 + 2 + 3 + \cdots + m + (m+1) = \frac{1}{2}(m+1)(m+2)$ coefficients from K. Thus, if $\deg(f), \deg(g) \leq m$, then we can represent $\frac{f}{g}$ with $(m+1)(m+2)$ coefficients from K. On the other hand, more efficient representations are available. Indeed, if $f \in K[X,Y]$,

[1] This restriction is more delicate in general than discussed here; see [Sil09].

then as $K[X,Y] = K[X][Y]$, we have the following.

$$
\begin{aligned}
f(X,Y) &= \sum_{j=0}^{m} b_j(X)Y^j \text{ where } b_j \in K[X] \text{ for } 0 \le j \le m && (3)\\
&= \sum_{j=0}^{\lceil m/2 \rceil} b_{2j}(X)Y^{2j} + b_{2j+1}(X)Y^{2j+1} \text{ where } b_{m+1} = 0 && (4)\\
&= \sum_{j} Y^{2j}\left(b_{2j}(X) + b_{2j+1}(X)Y\right) && (5)\\
&= \sum_{j} \left(X^3 + \alpha X + \beta\right)^j \left(b_{2j}(X) + b_{2j+1}(X)Y\right) && (6)\\
&= \underbrace{\left(\sum_{j} \left(X^3 + \alpha X + \beta\right)^j b_{2j}(X)\right)}_{a(X)} + \underbrace{\left(\sum_{j} \left(X^3 + \alpha X + \beta\right)^j b_{2j+1}(X)\right)}_{b(X)Y} Y && (7)\\
&= a(X) + b(X)Y && (8)
\end{aligned}
$$

where $a(X) = \sum_{j=0}^{\lceil m/2 \rceil} (X^3+\alpha X+\beta)^j b_{2j}(X) \in K[X]$ and $b(X) = \sum_{j=0}^{\lceil m/2 \rceil} (X^3+\alpha X+\beta)^j b_{2j+1}(X) \in K[X]$. Representing a polynomial f this way requires $\deg(a) + \deg(b) + 2$ elements of K. So, we can forget our above representation of elements of the function field $\frac{f}{g}$ with $f, g \in K[X,Y]$, and use $\frac{a_1(X)+b_1(X)Y}{a_2(X)+b_2(X)Y}$ for some $a_1, a_2, b_1, b_2 \in K[X]$, representing the function field element with only $\deg(a_1) + \deg(a_2) + \deg(b_1) + \deg(b_2) + 4$ elements of K instead.

We may go further, however. For each $a(X)+b(X)Y+(e)$, we have an element $a(X)-b(X)Y+(e) \in K[X,Y]/(e)$ satisfying the following due to the ideal $I(\mathcal{E}/K) = (e)$ and the multiplicative absorption property of ideals.

$$
\begin{aligned}
(a_1(X) + b_1(X)Y + (e))(a_1(X) - b_1(X)Y + (e)) &= a_1^2 - b_1^2Y^2 + (e) && (9)\\
&= a_1^2 - (X^3 + \alpha X + \beta)b_1^2 + (e) && (10)
\end{aligned}
$$

Hence, for any $\frac{a_1(X)+b_1(X)Y+(e)}{a_2(X)+b_2(X)Y+(e)} \in K(\mathcal{E})$, we can apply an identity map by multiplying and dividing by $(a_2(X) - b_2(X)Y + (e))$, yielding another representative whose denominator does not depend on X as follows

$$
\begin{aligned}
&\frac{a_1(X) + b_1(X)Y + (e)}{a_2(X) + b_2(X)Y + (e)} \cdot \frac{a_2(X) - b_2(X)Y + (e)}{a_2(X) - b_2(X)Y + (e)} && (11)\\
&= \frac{a_1(X)a_2(X) - a_1(X)b_2(X)Y + a_2(X)b_1(X)Y - b_1(X)b_2(X)Y^2 + (e)}{(a_2(X))^2 - (b_2(X))^2Y^2 + (e)} && (12)\\
&= \frac{f_1(X) + f_2(X)Y + (e)}{f_3(X) + (e)} && (13)
\end{aligned}
$$

where $f_1(X) = a_1(X)a_2(X) - (X^3 + \alpha X + \beta)b_1(X)b_2(X)$, $f_2(X) = a_2(X)b_1(X) - a_1(X)b_2(X)$, and $f_3(X) = (a_2(X))^2 - (X^3 + \alpha X + \beta)(b_2(X))^2$. This requires only $\deg(f_1(X)) + \deg(f_2(X)) + \deg(f_3(X)) + 3$ elements of K.

Thus, we write every element of $K(\mathcal{E})$ as $\frac{f_1(X)+f_2(X)Y+(e)}{f_3(X)+(e)}$ for some polynomials $f_0(X)$, $f_1(X)$, $f_2(X) \in K[X]$. Later, we outperform this slightly, as we practically only consider elements such that $\deg(f_3(X)+(e)) = 0$, i.e. $f_3(X)+(e) = 1+(e)$, requiring only $\deg(f_1)+\deg(f_2)+2$ coefficients to describe f.

2.4 Elliptic curve groups

Recall the distinguished polynomial $e(X,Y) = Y^2 - X^3 - \alpha X - \beta \in K[X,Y]$ where $4\alpha^3 + 27\beta^2 \neq 0$, and consider the plane affine curve $\mathcal{E} = v_a(e)$. Note $I(\mathcal{E}/K) = (e)$. This $\mathcal{E}$ is an elliptic curve with affine coordinate ring $K[\mathcal{E}] = \frac{K[X,Y]}{(e)}$ and function field $K(\mathcal{E}) = \mathrm{Frac}(K[\mathcal{E}])$. For a natural number $n \in \mathbb{N}$, the surface $\mathcal{E}^n$ is a variety with coordinate ring $K[\mathcal{E}^n] = K[X_1, Y_1, \ldots, X_n, Y_n]/I(\mathcal{E}^n/K)$, where $I(\mathcal{E}^n/K)$ is the ideal generated by the set $\{e(X_i, Y_i) \mid i \in [n]\}$, and function field $K(\mathcal{E}^n) = \mathrm{Frac}(K[X_1, Y_1, \ldots, X_n, Y_n]/I(\mathcal{E}^n/K))$.

In the following, we take Bézout's theorem for granted (for more details, see Theorem I.7.8 in [Har13]).

Theorem 2.4.1 (Bézout's Theorem)**.** Let $\mathcal{F}_1$, $\mathcal{F}_2$ be distinct curves in $\mathbb{P}^2(\overline{K})$, having degrees n_1, n_2, respectively. Let $\mathcal{F}_1 \cap \mathcal{F}_2 = \{P_1, \ldots, P_s\}$, and say, for each $1 \leq i \leq s$, that point P_i has multiplicity m_i. Then $\sum_{i=1}^{s} m_i = n_1 n_2$.

The projective variety corresponding to $\mathcal{E}$ is $\mathbb{G}(\mathcal{E}) = v_p(e)$, where e^* is the homogenization of e defined earlier. This $\mathbb{G}(\mathcal{E})$ admits an abelian group operation $+$ following the elliptic curve group law, which determines decompositions of the group identity in $\mathbb{G}(\mathcal{E})$ as follows. First, the point at infinity $O = (0:1:0)$ is distinguished as the group identity. Next, collinear points define decompositions of O. Bezout's theorem implies every line intersects $v_p(\mathcal{E})$ at 3 points, counting multiplicities. Hence, every set of collinear points is a triple $(P, Q, R) \in \mathbb{G}^3$ such that $P+Q+R = O$. We have three cases: P has multiplicity 3, 2, or 1.

- If $P = Q = R$, then $3P = O$, i.e. P has multiplicity 3. This is a contradiction for a non-singular elliptic curve $\mathcal{E}$.
- If $P = Q$, then $P + P + R = 2P + R = O$, so $2P = -R$. In this case, we sum points with themselves. However, this implies that the line ℓ interpolating P and R intersects $\mathcal{E}$ at P with multiplicity 2. However, $\mathcal{E}$ is non-singular, so it must be that ℓ is *tangent* to $\mathcal{E}$ at P.
- If P, Q, and R are all distinct, we have two sub-cases: one of these points is an inverse of another, or not.
  - If $P = -Q$, then $P + Q + R = R = O$, so $R = O$. In this case, the line interpolating P, Q, and R includes the point at infinity O. The lines passing through O and intersecting $\mathcal{E}$ at a non-identity point are precisely the *vertical lines*. Similarly for the case $P = -R$.
  - All other cases are when P, Q, and R are distinct, not inverses of each other, and correspond to lines which are neither vertical nor tangent.

That is to say, to find $-P$, we use the other root on the line passing through P which is *vertical*; to sum a point P with itself we use the other root on the line passing through P which is *tangent to* $\mathcal{E}$; to add two generic non-identity points P, Q we use the third root R of the line ℓ interpolating P, Q, setting $P + Q = -R$. These rules are sufficient for generating the elliptic curve group $\mathbb{G}(\mathcal{E})$. In the sequel, we denote this $\mathbb{G}$, as we use the same $\mathcal{E}$ throughout.

We emphasize that group points like P, Q, R are elements of $v_p(e)$, and thus are points from the projective plane, of the form $(x : y : z)$. All points (x, y) from $\mathbb{A}^2$ are embedded as $(x : y : 1)$, and the only solution to e^* in $\mathbb{P}^2$ with $z = 0$ is exactly $(0 : 1 : 0)$, so if P, Q, and R are non-identity, then these have $z \neq 0$. That is to say, each $(x : y : z)$ has a representative $(\frac{x}{z}, \frac{y}{z}, 1)$, and we may re-label our variables to assume the non-identity points can all be written $(x : y : 1) \in \mathbb{P}^2$. Such a point admits the dehomogenization, an affine point $(x, y) \in \mathbb{A}^2$ (which can, in turn, be re-embedded into $\mathbb{P}^2$). For this reason, we often conflate non-identity points like P, Q, $R \in v_p(e)$ with their dehomogenizations (x_P, y_P), (x_Q, y_Q), and (x_R, y_R), clarifying whether we are handling affine or projective points with context in the text.

In the sequel, we leave $\mathcal{E}$ implicit and denote this $\mathbb{G}$, as we use the same $\mathcal{E}$ throughout. Moreover, the group law as described here is equivalent to a group law defined for rational functions, not merely lines; the proof of this is beyond the scope of this text. See [Sil09] for details.

2.4.1 Group law and summing points in greater detail

We use lines to compute $P + Q$ explicitly as a rational function for a pair $(P, Q) \in (\mathbb{G} \setminus \{O\})^2$ such that $P \neq \pm Q$; this is useful in our derivations in Appendix A later. As both are not the identity and $P \neq \pm Q$, the line interpolating P and Q is neither vertical nor tangent to $\mathcal{E}$. By Bézout's theorem, the line must intersect $\mathcal{E}$ transversally at P and Q, as well as at a third distinct point R, and these intersection points all have multiplicity 1. The group law states $P + Q + R = O$. We compute (x_R, y_R) to represent R.

Note that, as elements of $\mathbb{G}$, we may write

$$P = (x_P : y_P : z_P) = \left\{(x, y, z) \mid \exists \lambda \in K^\times, (x, y, z) = (\lambda x_P, \lambda y_P, \lambda z_P)\right\}$$

and $Q = (x_Q : y_Q : z_Q)$ similarly. As non-identity elements, $P \neq O = (0 : 1 : 0)$ so $z_P \neq 0$, and the equivalence class $(x_P : y_P : z_P)$ contains a representation of the form $(x, y, 1)$ (and similarly for Q. Thus, we lose no generality by assuming $z_P = z_Q = 1$, in which case we can write $P = (x_P : y_P : 1)$ and $Q = (x_Q : y_Q : 1)$. Then P and Q may be dehomogenized to affine points (x_P, y_P), $(x_Q, y_Q) \in \mathcal{E}$; we identify $P = (x_P : y_P : 1)$ with (x_P, y_P), and similarly for Q, in our discussion below, keeping in mind that we are actually transferring between points on the projective plane to points on the affine plane via the dehomogenization and homogenization maps.

Define $\lambda = \frac{y_P - y_Q}{x_P - x_Q}$ and $\mu = y_P - \lambda x_P$ as functions of P and Q. These λ, μ parameterize the interpolating line $\ell(X, Y) = Y - \lambda X - \mu \in K[X, Y]$, and therefore $y_T = \lambda x_T + \mu$ for each $T \in \{P, Q, R\}$. Also, since $e(x_T, y_T) = 0$, we have $(\lambda x_T + \mu)^2 = x_T^3 + \alpha x_T + \beta$. Thus, the points P, Q, R have x coordinates which are distinct roots of the third degree polynomial $X^3 - \lambda^2 X^2 +$

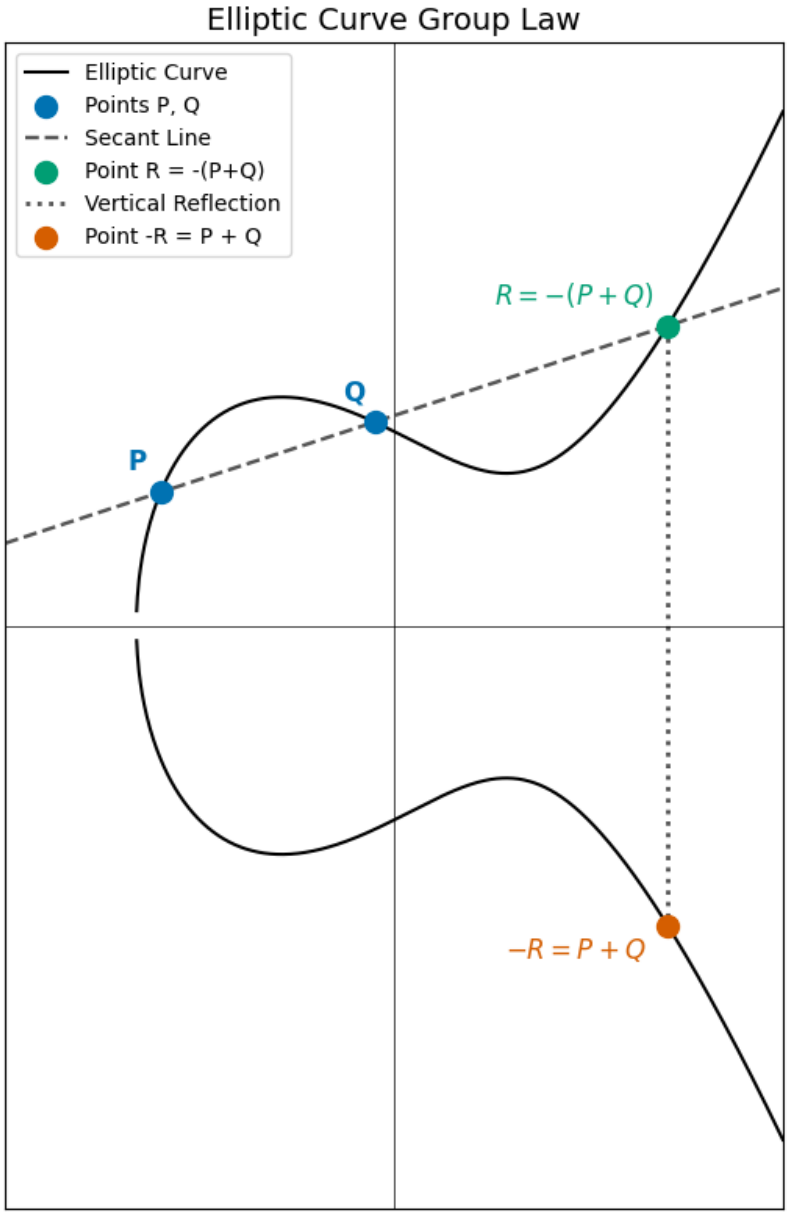

Figure 1: The points P, Q, R are collinear, so their sum is $P + Q + R = O$, where O is the group identity. In particular, $R = -(P + Q)$ and its reflection across the X-axis is $-R = P + Q$.

$(\alpha - 2\lambda\mu)X + (\beta - \mu^2)$ in X. All the roots are accounted for, so the polynomial splits into linear factors.

$$(X - x_P)(X - x_Q)(X - x_R) = X^3 - \lambda^2 X^2 + (\alpha - 2\lambda\mu)X + (\beta - \mu^2) \tag{14}$$

The linear independence of $\{1, X, X^2, X^3\}$ implies we may match coefficients, so $-\lambda^2 = -x_P - x_Q - x_R$, where x_P and x_Q are already given. Thus we have the following, where we write x_R and y_R as rational functions in the coordinates of P and Q:

$$x_R = \lambda^2 - x_P - x_Q \tag{15}$$

$$y_R = \lambda x_R + \mu \tag{16}$$

Recalling λ and μ defined above are rational functions, it is clear that these are both rational in (x_P, x_Q), as promised. Thus, the group law demands the following system of equations are satisfied.

$$\begin{cases} \lambda = & \frac{y_Q - y_P}{x_Q - x_P} \\ \mu = & y_P - \lambda x_P \\ y_P = & \lambda x_P + \mu \\ y_Q = & \lambda x_Q + \mu \\ y_R = & \lambda x_R + \mu \\ y_P^2 = & x_P^3 + \alpha x_P + \beta \\ y_Q^2 = & x_Q^3 + \alpha x_Q + \beta \\ y_R^2 = & x_R^3 + \alpha x_R + \beta \end{cases} \tag{17}$$

Note that if x_P, y_P, x_Q, y_Q are variables, then these are all rational functions.

2.5 Divisors over $\mathcal{E}$

For each $T = (x : y : z) \in \mathbb{G}$, let $[T]$ be a formal symbol corresponding to T. A *divisor* is a finite sum of these formal symbols. These sums can be summed, so divisors generate an additive group $\mathrm{Div}_K(\mathcal{E})$. We use functions $\nu : \mathbb{G} \to \mathbb{Z}$ with finite supports to represent a divisor $[\nu] = \sum_{T \in \mathbb{G}} \nu(T)\,[T]$. The *degree* of $[\nu]$ is $\deg([\nu]) = \sum_{T \in \mathbb{G}} \nu(T)$. Let $\mathrm{Div}_K^0(\mathcal{E})$ denote the set of degree-zero divisors. Note $\mathrm{Div}_K^0(\mathcal{E}) \subseteq \mathrm{Div}_K(\mathcal{E})$ and is closed under addition, i.e. is a subgroup.

Given any $f \in K(\mathcal{E})^\times$, we can define a map $\mathcal{E} \to \mathbb{Z}$ by mapping $T \mapsto \mathrm{ord}_T(f)$, and this has with finite support as the numerator and denominator of f both have finite degree (and hence a f has a finite number of roots and poles, even counting multiplicity). Denote $\mathrm{div}(f) = \sum_{T \in \mathcal{E}} \mathrm{ord}_T(f)[T]$. Define $\mathrm{Prin}(\mathcal{E}) = \{\mathrm{div}(f) \mid f \in K(\mathcal{E})\}$. Note we have the following chain of subgroups: $\mathrm{Prin}(\mathcal{E}) \subseteq \mathrm{Div}_K^0(\mathcal{E}) \subseteq \mathrm{Div}_K(\mathcal{E})$. In the sequel, define the following relation.

$$\mathcal{R}_{\mathrm{div}} = \left\{(f, [\nu]) \mid f \in K(\mathcal{E}), [\nu] \in \mathrm{Div}_K^0(\mathcal{E}), \text{ and } \mathrm{div}(f) = [\nu]\right\} \tag{18}$$

Later, we are particularly interested in the following sub-relation.

$$\mathcal{R}'_{\text{div}} = \left\{(f,[\nu]) \mid f \in K(\mathcal{E}), [\nu] \in \text{Div}^0_K(\mathcal{E}), |\text{supp}(\nu)| \geq 3, \text{ and } \text{div}(f) = [\nu]\right\} \tag{19}$$

We take the following for granted.

Theorem 2.5.1. Let $f \in K(\mathcal{E})$, $v_a(f) \setminus \{O\} = \{P_1, \ldots, P_n\}$, and $v_a(f^{-1}) \setminus \{O\} = \{Q_1, \ldots, Q_s\}$ with $n + s \geq 1$. If each P_i has multiplicity $m_i \in \mathbb{N}$ and each Q_j has multiplicity $t_j \in \mathbb{N}$, and $\sum_i m_i - \sum_j t_j \neq 0$, then:

(a) $\sum_i m_i - \sum_j t_j > 0$ implies f has a pole at O with multiplicity $\sum_i m_i - \sum_j t_j$,

(b) $\sum_i m_i - \sum_j t_j < 0$ implies f has a root at O with multiplicity $\sum_j t_j - \sum_i m_i$, and

(c) $\sum_i m_i P_i - \sum_j t_j Q_j = O$.

In particular, $\text{div}(f) = \sum_i m_i[P_i] - \sum_j t_j[Q_j] - (\sum_i m_i - \sum_j t_j)[O]$, $\deg(\text{div}(f)) = 0$, and $\text{div}(f)$ represents a nontrivial decomposition of $O \in \mathbb{G}$, namely $\sum_i m_i P_i - \sum_i t_j Q_j = O$.

If $f_1, f_2 \in K(\mathcal{E})$ have $(v_p(f_1^*) \cup v_p(1/f_1^*)) \cap (v_p(f_2^*) \cup v_p(1/f_2^*)) = \emptyset$, we say f_1 and f_2 have *disjoint roots and poles*. If f_1 and f_2 have disjoint roots and poles, it is easy to check that $v_p(f_1^* f_2^*) = v_p(f_1^*) \cup v_p(f_2^*)$ and $v_p(1/(f_1^* f_2^*)) = v_p(1/f_1^*) \cup v_p(1/f_2^*)$. Thus, for such an f_1, f_2 pair, we have that $\text{div}(f_1 f_2) = \text{div}(f_1) + \text{div}(f_2)$. More generally, div is a group homomorphism from the multiplicative subgroup of the function field $K(\mathcal{E})^\times$ to the additive (sub)group $\text{Prin}(\mathcal{E})$.

Note that if f_1 and f_2 do not have disjoint roots and poles, then the common roots and poles may partially or completely cancel in the product $f_1 f_2$. This expresses itself as terms canceling in the sum $\text{div}(f_1) + \text{div}(f_2)$.

2.6 Weil's reciprocity

Every $f \in K(\mathcal{E})$ induces a homomorphism. Indeed, let $[\eta] = \sum_{T \in \mathbb{G}} \eta(T)[T]$ be any divisor such that the support of η is disjoint from the roots and poles of f. Then let $\widehat{f}([\eta]) = \prod_{T \in \mathbb{G}} f^*(T)^{\eta(T)}$; this is well-defined because the points of $[\eta]$ are disjoint from the roots and poles of f.

Theorem 2.6.1. For each $f \in K(\mathcal{E})$, if $[\eta_1]$ $[\eta_2]$ have disjoint supports also disjoint from from the roots and poles of f, then $\widehat{f}([\eta_1 + \eta_2]) = \prod_{T \in \mathbb{G}} f^*(T)^{\eta_1(T) + \eta_2(T)} = \widehat{f}([\eta_1])\widehat{f}([\eta_2])$.

Theorem 2.6.2 (Weil's Reciprocity Law)**.** Let $f, g \in K(\mathcal{E})$, with $\text{div}(f) = \sum_T \nu(T)[T]$ and $\text{div}(g) = \sum_T \eta(T)[T]$. Then

$$\widehat{f}(\text{div}(g)) = \prod_{T \in \mathbb{G}} f^*(T)^{\eta(T)} = \prod_{S \in \mathbb{G}} g^*(S)^{\nu(S)} = \widehat{g}(\text{div}(f)).$$

In particular, if g has a degree 1 polynomial representing its numerator $aX + bY + c + (e)$ and $1 + (e)$ as the denominator, then $\widehat{g}([\nu]) = \prod_T (ax_T + by_T + c)^{\nu(T)}$.

2.7 Derivations

Polynomials admit formal derivatives and partial derivatives as usual: if $f = \sum_i a_i X^i$, define $f' = \sum_i i a_i X^{i-1}$, and if $f = \sum_{i,j} a_{i,j} X^i Y^j$, define $\frac{\partial}{\partial X} f = \sum_{i,j} i a_{i,j} X^{i-1} Y^j$ and $\frac{\partial}{\partial Y} f = \sum_{i,j} j a_{i,j} X^i Y^{j-1}$.

More generally, given any field extension $K \subseteq L$, an additive group homomorphism $d : L \to L$ is said to be a *derivation over* K if $d(K) = \{0\}$ and d satisfies Leibniz' rule: $d(ab) = d(a)b + ad(b)$. Given an intermediate field, $K \subseteq F \subseteq L$, such that $d(F) \subseteq F$, the restriction of d over K on L to F yields a derivation over K on F. In the case $L = \overline{K}(X, Y)$, derivations $\frac{\partial}{\partial X}$ and $\frac{\partial}{\partial Y}$ are defined as usual (and $\frac{\partial}{\partial Z}$ is defined over $\overline{K}(X, Y, Z)$ similarly). All derivations are linear combinations of these derivations $\frac{\partial}{\partial X}$, $\frac{\partial}{\partial Y}$, and $\frac{\partial}{\partial Z}$. Indeed,

(i) for any $a, b \in L$, every linear combination $a(X, Y)\frac{\partial}{\partial X} + b(X, Y)\frac{\partial}{\partial Y}$ is a derivation over K on L, and

(ii) every derivation over K on L is of the form $a(X, Y)\frac{\partial}{\partial X} + b(X, Y)\frac{\partial}{\partial Y}$, for some $a, b \in \overline{K}(X, Y)$.

For $F \subseteq \overline{K}(X, Y)$, the derivations satisfying $d(F) \subseteq F$ are precisely the derivations with corresponding $(a, b) \in F^2$.

Similarly, $\overline{K}(X, Y, Z)$ has derivations of the form $d = a\frac{\partial}{\partial X} + b\frac{\partial}{\partial Y} + c\frac{\partial}{\partial Z}$ for some $a, b, c \in \overline{K}(X, Y, Z)$, and if all a, b, c are in an intermediate field $K \subseteq F \subseteq \overline{K}(X, Y, Z)$, then d is a derivation on F.

Given any derivation $d : L \to L$ over K, there is a natural map $\delta : L \to L$ mapping $f \mapsto \frac{df}{f}$. Note that given any $f, g \in L$, we have that

$$\delta(fg) = \frac{d(fg)}{fg} = \frac{d(f)g + fd(g)}{fg} = \frac{d(f)}{f} + \frac{d(g)}{g} = \delta(f) + \delta(g).$$

In particular, δ is a group homomorphism from the multiplicative subgroup of L to the additive group of L.

2.8 Schwartz-Zippel

We take the following for granted.

Lemma 2.8.1. Let $0 \neq f \in \overline{K}[Z_1, Z_2, \ldots, Z_n]$ with $\deg(f) = d$, and let S be a finite subset of $\overline{K}$. If $s_1, s_2, \ldots, s_n$ are sampled independently and uniformly from S, then the probability that $f(s_1, \ldots, s_n) = 0$ is at most $\frac{d}{|S|}$.

3 Method

Suppose that a prover and verifier wish to implement a proving scheme in an elliptic curve group setting. In most cases, verification requires satisfying one or more linear constraints coming from the group law of $\mathbb{G}$. That is to say, a proof includes some non-identity group elements $P_1, \ldots, P_n \in$

$\mathbb{G} \setminus \{O\}$ and integer weights $m_1, \ldots, m_n \in \mathbb{Z}$, then the verification procedure requires checking the $\sum_{i=1}^{n} m_i P_i = O$.

The prover and verifier recall that every principal divisor $\sum_{i=1}^{n} m_i[P_i] - (\sum_{i=1}^{n} m_i)[O]$ corresponds to a decomposition of O, so they can merely check whether some divisor is principal. The map *div* is a surjective group homomorphism. Therefore, there exists an element $f \in K(\mathcal{E})$ such that $\mathrm{div}(f) = \sum_i m_i[P_i] - (\sum_i m_i)\,[O]$, which can be used as a witness of principality.

The prover and verifier also recall Weil's reciprocity provides $\widehat{f}(\mathrm{div}(g)) = \widehat{g}(\mathrm{div}(f))$ for every pair $f, g \in K(\mathcal{E}) \subseteq K(\mathcal{E})$. For a fixed f, Weil's reciprocity implies an equality of rational functions in the coefficients of g and the coordinates of the points of $\mathrm{div}(g)$; clearing denominators, we obtain a polynomial expression. Evaluating this polynomial at a given $(g, \mathrm{div}(g))$ always yields equality.

Thus, if $f \in K(\mathcal{E})$ and some random $(g, \mathrm{div}(g))$ is a root of the polynomial obtained by rationalizing $\widehat{f}(\mathrm{div}(g)) - \widehat{g}(\mathrm{div}(f))$, then the Schwarz-Zippel Lemma implies that the polynomial is the zero polynomial exactly, except with a probability inversely proportional to the cardinality of the set from which we sample g.

We show that checking whether $\widehat{f}(\mathrm{div}(g)) = \widehat{g}([\nu])$ for a random linear g is sufficient to determine if $\mathrm{div}(f) = [\nu]$, except perhaps with a probability given by the Schwarz-Zippel Lemma. Thus, the prover and verifier can conclude that $\sum_{T \in \mathbb{G}} \nu(T) T = O$.

To map expensive multiplication operations to cheaper addition, we exploit logarithmic derivatives, reducing computational costs.

3.1 Problems with previous approaches

The system of 8 equations with 6 unknowns in Equation (17) influences many computations with derivations in the sequel, in turn impacting the application of the chain rule.

This is precisely the complication left unaddressed by Eagen in [Eag22], and sketched by Bassa in [Bas]. We think the work in [Eag22] intended to characterize solutions to this system as dependent upon λ and μ, the only free variables, exactly determining solutions to this system. Had this been correct, the resulting verification equations would have reduced Eagen's claims. This would simplify verification by allowing the chain rule to be computed with respect to the line $Y - \lambda X - \mu = 0$.

While solutions for this system of equations in terms of λ and μ are easily obtainable using Cardano's formula, the solutions are not guaranteed to be rational functions in λ and μ. In fact, generally these solutions are not rational, which complicates the application of derivatives beyond the computations in [Eag22].

On the other hand, by picking P and Q, we obtain x_P, y_P, x_Q, y_Q, instead over-determining the solution to this system; as Bassa noted, $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ form a pair on the surface $(P, Q) \in \mathcal{E}^2$. With this, solutions are rational in x_P, y_P, x_Q, y_Q. However, the y coordinates are *almost* uniquely determined by their corresponding x coordinate. Indeed, for each $T \in \{P, Q, R\}$, only two elements of K satisfy $y_T^2 = x_T^3 + \alpha x_T + \beta$. Distinguishing which y_T corresponds to the point T simply requires a convention for interpreting a parity/sign bit. Thus, representing (x_T, y_T) requires only one more bit than representing x_T alone. Thus, we really have five unknown field

elements $(x_P, x_Q, x_R, \lambda, \mu)$ and 3 sign bits (one each to uniquely determine y_P, y_Q, y_R). Above, we pick P and Q, i.e. two field elements and two bits, and this is sufficient to uniquely determine R, λ, and μ. By over-determining our solutions, we risk that with small probability a solution may not exist, however we only slightly over-determine our solutions by 2 bits. For this price, we then obtain rational solutions.

3.2 Method description

Let K be a finite field with characteristic p, and let $\mathcal{E}$ be a non-singular elliptic curve over $K[X, Y]$ such that $|\mathbb{G}(\mathcal{E})| = q > 3$ for some prime q. Let $H : \{0,1\}^* \to (\mathbb{G} \setminus \{O\})^2$ be a cryptographic hash function. Define the tuple of algorithms $\Pi = (\texttt{Setup}, \texttt{Prove}, \texttt{Verify})$ as follows.

- $\texttt{Setup}(\lambda) \to \texttt{params}$. Input security parameter and output setup parameters $\texttt{params}$.
- $\texttt{Prove}(\texttt{params}, [\nu]) \to \pi$. The prover does the following:
  1. Compute
  $$\texttt{out} \leftarrow \texttt{Wit}(\texttt{params}, \underbrace{P_0, \ldots, P_0}_{\nu(P_0) \text{ times}}, \underbrace{P_1, \ldots, P_1}_{\nu(P_1) \text{ times}}, \cdots, \underbrace{P_{n-1}, \ldots, P_{n-1}}_{\nu(P_{n-1}) \text{ times}})$$
  with Algorithm 2; if $\texttt{out} = \perp$, output $\perp$ and terminate.
  2. Compute $(P, Q) \leftarrow H(\texttt{params}, f, [\nu])$. If $P = \pm Q$, output $\perp$ and terminate.
  3. Compute $R = -(P + Q)$. If $P = \pm R$ or $Q = \pm R$ or $\{P, Q, R\} \cap \{P_i\}_{i=1}^n \neq \emptyset$, then output $\perp$ and terminate.
  4. Parse the following, if possible; otherwise, output $\perp$ and terminate.

$$(f_1(X), f_2(X)) = f \leftarrow \texttt{out} \tag{20}$$
$$\sum_{i=1}^{n} m_i[P_i] \leftarrow [\nu] \tag{21}$$
$$(x_i, y_i) \leftarrow P_i \text{ for each } 1 \leq i \leq n \tag{22}$$
$$(x_P, y_P) \leftarrow P \tag{23}$$
$$(x_Q, y_Q) \leftarrow Q \tag{24}$$
$$(x_R, y_R) \leftarrow R \tag{25}$$

5. Compute/set the following:

$$\Delta X = x_Q - x_P \tag{26}$$
$$\Delta Y = y_Q - y_P \tag{27}$$
$$\mu = y_P - \lambda x_P \tag{28}$$
$$f_X = f_1'(X) + Y f_2'(X) \tag{29}$$
$$f_Y = f_2(X) \tag{30}$$

6. Compute $\lambda = \frac{\Delta Y}{\Delta X}$ and $y'_T = \frac{3x_T^2+\alpha}{2y_T}$ for each $T \in \{P, Q, R\}$. If $y'_T = \lambda$ or $f(T) = 0$ for any $T \in \{P, Q, R\}$, output $\perp$ and terminate.
7. Compute the following:

$$A_i = (y_i - \lambda x_i - \mu)^{-1} \text{ for each } 1 \le i \le n \tag{31}$$
$$B_1 = \frac{2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P)}{2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P)} \tag{32}$$
$$B_2 = -\frac{2\lambda y_P (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_P^2 + \alpha - 2y_P(\lambda + \Delta X))}{2y_P y_R f(R)(3x_P^2 + \alpha - 2y_P\lambda)\Delta X} \tag{33}$$
$$B_3 = \frac{2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q)}{2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q)} \tag{34}$$
$$B_4 = \frac{2\lambda y_Q (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_Q^2 + \alpha - 2y_Q(\lambda + \Delta X))}{2y_Q y_R f(R)(3x_Q^2 + \alpha - 2\lambda y_Q)\Delta X} \tag{35}$$

8. Output $\pi = (f, \lambda, y'_P, y'_Q, y'_R, A_1, \ldots, A_n, B_1, B_2, B_3, B_4)$.

- `Verify(params,` $[\nu], \pi) \to b \in \{0, 1\}$.

1. Parse $(f, \lambda, y'_P, y'_Q, y'_R, A_1, \ldots, A_n, B_1, B_2, B_3, B_4) \leftarrow \pi$ if possible; otherwise, output 0 and terminate.
2. Execute steps 2-5 from `Prove`. If any fail, output 0 and terminate.
3. Compute the following.

$$b_1 = 2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P) \tag{36}$$
$$c_1 = 2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P) \tag{37}$$
$$b_2 = 2y_P y_R f(R)(3x_P^2 + \alpha - 2y_P\lambda)\Delta X \tag{38}$$
$$c_2 = -2\lambda y_P (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_P^2 + \alpha - 2y_P(\lambda + \Delta X)) \tag{39}$$
$$b_3 = 2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q) \tag{40}$$
$$c_3 = 2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q) \tag{41}$$
$$b_4 = 2y_Q y_R f(R)(3x_Q^2 + \alpha - 2\lambda y_Q)\Delta X \tag{42}$$
$$c_4 = 2\lambda y_Q (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_Q^2 + \alpha - 2y_Q(\lambda + \Delta X)) \tag{43}$$

4. If any of the following hold, output 0 and terminate.
   - $\lambda \Delta X \neq \Delta Y$,
   - $2y_T y'_T \neq 3x_T^2 + \alpha$ for any $T \in \{P, Q, R\}$,
   - $(y_i - \lambda x_i - \mu)A_i \neq 1$ for some $1 \leq i \leq n$,
   - $b_i B_i \neq c_i$ for some $1 \leq i \leq 4$, or
   - $B_1 + B_2 + B_3 + B_4 \neq \sum_i m_i A_i$.
5. Otherwise, output 1 and terminate.

$\texttt{line}(P, Q) \to \texttt{out} \in \{\perp\} \cup K(\mathcal{E})$
If $P \notin \mathcal{E}$ or $Q \notin \mathcal{E}$, return $\perp$.
Elif $P = Q = O$ return $1 \in K$.
Elif $P = O$ and $Q \neq O$, return $\texttt{line}(Q, Q)$.
Elif $P \neq O$ and $Q = O$, return $\texttt{line}(P, P)$.
Elif $P = Q$:
Set $\lambda = \frac{2y_P}{3x_P^2 + \alpha}$
Elif $P = -Q$:
Set $\lambda = 0$
Else:
Compute:
$\Delta y = y_Q - y_P$
$\Delta x = x_Q - x_P$
$\lambda = \frac{\Delta y}{\Delta x}$
Return (P, λ).

Algorithm 1: Input points $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$, and outputs the point-slope representation of a degree 1 polynomial. If $P = Q$, the line passes through P and is tangent to $\mathcal{E}$; if $P = -Q$, the line is vertical and passes through both P and Q; otherwise, the line interpolates P and Q.

$\texttt{Wit}(P_0, P_1, P_2, \ldots, P_{n-1}) \to \texttt{out} \in \{\perp\} \cup K(\mathcal{E})$

If $n = 0$ **then return** $\perp$.
If $(P_0 + P_1 + \cdots + P_{n-1}) \neq O$ **then return** $\perp$.
If $P_j = O$ for any j **then return** $\perp$.
Initialize $\texttt{queue} = \emptyset$.
Let $j = 0$.
While $j < n$:
  If $j + 1 < n$:
    Set pair $(A_{j/2}, B_{j/2}) = (P_j, P_{j+1})$.
    Set degree $m_j = 2$.
  Else:
    Set pair $(A_{j/2}, B_{j/2}) = (P_j, O)$.
    Set degree $m_j = 1$.
  Compute:
    $T_{j/2} = A_{j/2} + B_{j/2} \in \mathcal{E}$,
    $\texttt{out} \leftarrow \texttt{line}(A_{j/2}, B_{j/2})$.
    If $\texttt{out} = \perp$, output $\perp$ and terminate.
    Else:
      Parse $(U_j, \lambda_j) \leftarrow \texttt{out}$.
      $(w_j, z_j) \leftarrow U_j$.
      Form $\ell_j = (Y - z_j) - \lambda_j(X - w_j)$.
  Enqueue $\texttt{queue} \leftarrow (m_j, T_j, \ell_j)$.
  Increment j by 2.
While $\texttt{len}(\texttt{queue}) > 1$:
  Initialize $\texttt{nextQueue} = \emptyset$.
  If $\texttt{len(queue)}$ is odd:
    Dequeue $x \leftarrow \texttt{queue}$.
    Enqueue $\texttt{nextQueue} \leftarrow x$.
  While $\texttt{len(queue)} \geq 2$:
    Dequeue (m_a, T_a, f_a), $(m_b, T_b, f_b) \leftarrow \texttt{queue}$.
    Compute:
      $T_{a,b} = T_a + T_b$,
      $\ell_{a,b} = \texttt{line}(T_a, T_b)$,
      $\texttt{numerator} = f_a \cdot f_b \cdot \ell_{a,b} \mod e(X, Y)$,
      $\texttt{denominator} = \texttt{line}(T_a, -T_a) \cdot \texttt{line}(T_b, -T_b)$,
      $m_{a,b} = m_a + m_b$.
      $g = \frac{\texttt{numerator}}{\texttt{denominator}}$.
    Enqueue $\texttt{nextQueue} \leftarrow (m_{a,b}, T_{a,b}, g)$.
  Set $\texttt{queue} = \texttt{nextQueue}$.
Pop final element (m, T, f) from $\texttt{queue}$.
If $T \neq O$ or $m \neq n$, output $\perp$ and terminate.
Output f.

Algorithm 2: Constructing a rational function whose divisor interpolates a set of elliptic curve points. Based on [Para].

We note that as $|\mathbb{G}| > 3$ and prime order, $\mathbb{G}$ contains no elements of order 2, so it is not necessary

to check whether any points have $y_T = 0$ before performing divisions. Variations of this scheme can be deployed with groups with non-prime order presuming that all computations are checked to occur within a prime-order subgroup.

3.2.1 Usage

Augmenting the output of a "scalar multiplication" algorithm computing $P = aG$ from a and G with a proof from this scheme creates a verifiable scalar multiplication scheme. Indeed, to prove that a scalar multiplication has been performed correctly, say $P = aG$ for some $G \in \mathbb{G} \setminus \{O\}$ and some $1 \leq a < \text{ord}(\mathbb{G})$, the prover runs `Wit` with $[\nu] = a[G] + [-P]$, $[aG] + [-P]$, or $\sum_i s_i[2^i G] + [-P]$ where $\vec{s} \bmod p$ is a non-unique decomposition of aG into binary powers of G.

3.2.2 Efficiency

Note that, in `Wit`, we compute $\frac{n}{2}$ group operations during pre-computation, and then in the first inner loop we compute $\frac{n}{4}$ operations, then $\frac{n}{8}$, then $\frac{n}{16}$, and so on. In particular, `Wit` takes up to $\sum_i \frac{n}{2^i} \leq n$ group operations. For each of these, up to 3 queries are made to `line`, 3 products in $K(\mathcal{E})$ are computed, and one division in $K(\mathcal{E})$. Also, in the first inner loop, we compute $\frac{n}{4}$ sums of degrees, then $\frac{n}{8}$, and so on, so we have at most $\frac{n}{2}$ sums of degrees; we interpret these as sums in K.

We display some operation counts in Table 3.

Algorithm 3: Operation Counts for Different Algorithms

Algorithm	$K, \frac{d}{dX}$	$K, +$	$K, \times$	$K, \div$	$\mathbb{G}, +$	$K(\mathcal{E}), \times$	$K(\mathcal{E}), \div$	Ext
`line`	0	2	3	1	0	0	0	-
`Wit`	0	$n/2$	0	0	n	$3n$	n	$3n$ (`line`)
`Prove`	2	$2n + 22$	$n + 72$	$n + 6$	1	0	0	1 (`Wit`, H)
`Verify`	2	$3n + 3$	$3n + 87$	0	1	0	0	1 (H)

Divisor proofs are impractical when divisor coefficients are too large. Consider an example where we wish to open a Pedersen commitment, $C = x_1 G_1 + x_2 G_2$. The most costly computations which a verifier must compute are field multiplications, so we only count those, for the sake of the example. The prover computes $\pi_1 \leftarrow \texttt{Prove}([x_1 G_1] + [x_2 G_2] + [-C])$, $\pi_2 \leftarrow \texttt{Prove}(x_1[G_1] + [-x_1 G_1])$, and $\pi_3 \leftarrow \texttt{Prove}(x_2[G_2] + [-x_2 G_2])$. For these, we have $n_1 = 3$, $n_2 = x_1 + 1$, and $n_3 = x_2 + 1$, so verifying these three proofs requires the verifier to compute 3 group operations and $3(n_1 + n_2 + n_3) + 261 = 3(5 + x_1 + x_2)$ field multiplications. At least one of these opening pieces of data x_1, x_2 is sampled uniformly from the scalar field $\mathbb{Z}/q\mathbb{Z}$ for $\mathbb{G}$, which typically has a cryptographically large q.

On the other hand, divisor proofs with bounded decompositions are practical. Consider the verifiable scalar multiplication gadget suggested in [Para]. That approach demonstrates scalar multiplication $P = aG$ by computing $s_0, s_1, \ldots, s_{t-1}$ such that $\sum_i s_i 2^i \equiv a (\text{mod } q)$. These $s_i \in \mathbb{Z}$ may be arbitrarily selected, and we use divisor $[\nu] = \sum_i s_i[2^i G] + [-P]$. Assuming q is some 256-bit prime, we have $n = 257$ elements in the support of this divisor. Thus, verifiers need to compute one group addition, two derivatives, one hash function, $3n + 3 = 774$ additions in K, $3n + 87 = 858$

multiplications in K, and no divisions. This compares favorably to, say, the Montgomery ladder in computing $P = aG$, which requires $O(2\log_2(a))$ divisions in K, cannot be placed within a circuit, and just recreates all the prover's work.

As most of the computational cost of verification addition and multiplication in K, without any division, proof verification can be done almost entirely within-circuit. Indeed, [Para] demonstrates methods for putting divisor proofs within rank-1 constraint systems.

3.2.3 Implementation risks

In the sequel, we omit `params` from the inputs of functions to be concise. In the generic divisors framework we present, the points P and Q may be selected non-interactively by utilizing the (strong) Fiat-Shamir heuristic [FS87]. This paradigm is often used to turn an interactive zero-knowledge proof into a non-interactive signature scheme. The Fiat-Shamir transform can implemented practically with a cryptographically secure hash function with codomain $(\mathbb{G} \setminus \{O\})^2$, which can model a PRNG (really, a cryptographically secure pseudo-random *elliptic curve point* generator, but this CSPRECPG may be modeled with a PRNG and a hash-to-point method), but care must be taken to avoid vulnerabilities with the weak Fiat-Shamir transform [NHB24, BPW16, DMWG23]. All public parameters and transcripted communications must be included in the hash input. To avoid vulnerabilities like the so-called Frozen Heart attack [Jon, Mil], the hash digest must have been generated from the commitments, previous randomness, user tags, counters, etc.

3.2.4 Extensions

Batch verification of multiple witnesses $f_1, \ldots, f_M$ for principal divisors $[\nu_1], \ldots, [\nu_M]$ may be possible by sampling coefficients $\gamma_1, \ldots, \gamma_M$ and testing the witness $\prod_j f_j^{\gamma_j}$ for the divisor $\sum \gamma_j[\nu_j]$. Furthermore, this can be simplified by asking the verifier to sample a single random value γ, then demonstrating the proof for $\sum_j \gamma^j[\nu_j]$.

However, see Section 3.2.2. Efficiency is impacted by the coefficients/multiplicities of a divisor. This should not be surprising, as scalar multiplication methods are more expensive as the scalar increases in bit length.

3.3 Security properties

It should be clear that, if $[\nu]$ is a nontrivial decomposition of O into a sum of non-identity points, then `MakePairs` certainly does not output $\perp$.

Lemma 3.3.1. Let $S = \{P_1, \ldots, P_n\}$ be a fixed subset of $\mathbb{G} \setminus \{O\}$. Let $\mathcal{C}$ be the set of 3-sets $\{P, Q, R\} \in \mathbb{G}^3$ such that P, Q, R are distinct, non-identity, and collinear. Then $|\mathcal{C}| = (|\mathbb{G}| - 1)(|K| - 2)$ and $|\{S \cap T = \emptyset \mid T \in \mathcal{C}\}| \geq |\mathcal{C}| - n(|K| - 2)$.

Proof. Each line in projective space which intersects $\mathcal{E}^*$ at three distinct non-identity points has a corresponding element in $\mathcal{C}$. Consider the point-slope representation of each such line. For any non-identity point $P = (x_P : y_P : 1) \in \mathbb{G} \setminus \{O\}$, every line passing through P can be represented

as $Y - \lambda(X - x_P) - y_P$, and each distinct choice of $\lambda \in K$ yields a distinct line. One of these lines is vertical, and one is tangent to $\mathcal{E}^*$. Thus, there are $|K| - 2$ non-vertical, non-tangent lines passing through any non-identity point P which intersect $\mathcal{E}$ at two other distinct non-identity points. Moreover, there are $|\mathbb{G}| - 1$ choices of non-identity P. Thus, we have $|\mathcal{C}| = (|\mathbb{G}| - 1)(|K| - 2)$.

Moreover, for each point $P_i \in S$, there are $|K|-2$ distinct non-vertical, non-tangent lines passing through P_i, so at most $n(|K| - 2)$ elements of $\mathcal{C}$ have interpolating lines with roots in S (generally fewer); thus, there are at least $|\mathcal{C}| - n(|K| - 2)$ triples disjoint from S (generally more). □

We obtain the following corollary immediately, but omit the proof. The proof is not illuminating and yet follows straightforwardly from using $v_p(\ell^*)$ as a choice of T in Lemma 3.3.1.

Corollary 3.3.2. Following Lemma 3.3.1, assuming $\ell \xleftarrow{\$} K[X, Y]$, the event that ℓ^* intersects $\mathcal{E}^*$ transversally at 3 distinct points is exactly $E = v_p(\ell^*) \in \mathcal{C} \wedge v_p(\ell^*) \cap S = \emptyset$. Moreover, $\Pr\left[E \mid \ell \xleftarrow{\$} K[X, Y]\right] \geq 1 - \frac{n}{|\mathbb{G}|-1}$.

Note that if $n = 1$ then calling `Wit` yields a failure symbol $\perp$, as a sum of a single point cannot be a nontrivial decomposition of $O \in \mathbb{G}$. If calling `Wit` with $n = 2$, we have two non-identity points P_0, P_1 such that $P_0 + P_1 = O$, but $P_i + P_j \neq O$ for any i, j.

Lemma 3.3.3. Let $n \geq 3$ and let $\{P_0, P_1, \ldots, P_{n-1}\} \subseteq \mathbb{G}$ be a multi-subset of non-identity points such that $\sum_i P_i = O$ and $P_i + P_j \neq O$ for any i, j. Then $\texttt{Wit}(P_0, P_1, P_2, \ldots, P_{n-1})$ does not fail, runs in time $t = O(n)$, and outputs some f such that there exists $m \in \mathbb{Z}$ satisfying $\mathrm{div}(f) = \sum_{i=0}^{n-1}[P_i] - m[O]$.

Proof. Pre-processing in `Wit` prepares the following $\lceil \frac{n}{2} \rceil$ pairs:

$$\begin{cases} (P_0, P_1), (P_2, P_3), \ldots, (P_{n-2}, P_{n-1}); & n \text{ even} \\ (P_0, P_1), (P_2, P_3), \ldots, (P_{n-1}, O); & n \text{ odd} \end{cases} \tag{44}$$

and computes the interpolating line $\ell_j = \texttt{line}(A_j, B_j)$ for each pair (A_j, B_j) with $0 \leq j \leq \lceil \frac{n}{2} \rceil$. Recall $\texttt{line}(P_{n-1}, O)$ is the line tangent to $\mathcal{E}$ at P_{n-1}. Then, the point $T_j = A_j + B_j$ is computed, the degree m_j is computed (which is 2 when $A_j \neq B_j$ are both non-identity and 1 when $A_j = O$ or $B_j = O$). Each triple (m_j, T_j, ℓ_j) is enqueued.

Once this queue is constructed, the primary loop begins. In the while-loop, `Wit` pops pairs of triples at a time and merges them. Note that for any one of these initial lines, $\mathrm{div}(\ell_j) = [A_j] + [B_j] + [-(A_j + B_j)]$ (disregarding the $[O]$ term to be concise).

We claim that all the triples which enter the queue are of the form (m, T, f) where $T = \sum_i \Omega_i$ for some non-identity points $\Omega_i \in \mathbb{G}(\mathcal{E})$, and such that $\mathrm{div}(f) = \sum_i [\Omega_i] + [-\sum_i \Omega_i]$. Certainly the initial lines satisfy this. Moreover, for each (m_f, T_f, f), (m_g, T_g, g) in the queue, say $T_f = \sum_i \Omega_i$ and $T_g = \sum_j \Theta_j$, write the non-identity terms of the divisors $\mathrm{div}(f) = \sum_i [\Omega_i] + [-T_f]$ and $\mathrm{div}(g) = \sum_j [\Theta_j] + [-T_g]$ and notice the merged function field element is $h = fg\frac{\texttt{line}(T_f, T_g)}{\texttt{line}(T_f, -T_f)\texttt{line}(T_g, -T_g)}$. The

new degree $m_f + m_g$ is computed, and $(m_f + m_g, T_f + T_g, h)$ is enqueued. For this h, we note that we have the following.

$$\begin{aligned}
\operatorname{div}(h) =& \operatorname{div}\left(fg \frac{\texttt{line}(T_f, T_g)}{\texttt{line}(T_f, -T_f)\texttt{line}(T_g, -T_g)} \right) && (45)\\
=& \operatorname{div}(f) + \operatorname{div}(g) + \operatorname{div}(\texttt{line}(T_f, T_g)) - \operatorname{div}(\texttt{line}(T_f, -T_f)) - \operatorname{div}(\texttt{line}(T_g, -T_g)) && (46)\\
=& \sum_i [\Omega_i] + [-T_f] + \sum_j [\Theta_j] + [-T_g] + [T_f] + [T_g] && (47)\\
&+ [-(T_f + T_g)] - [T_f] - [-T_f] - [T_g] - [-T_g] && (48)\\
=& \sum_i [\Omega_i] + \sum_j [\Theta_j] + [-(T_f + T_g)] && (49)\\
=& \sum_i [\Omega_i] + \sum_j [\Theta_j] + [-(\sum_i \Omega_i + \sum_j \Theta_j)] && (50)
\end{aligned}$$

Hence, every entry of `queue` is a triple $(m, T, f) \in \mathbb{N} \times \mathbb{G}(\mathcal{E}) \times K(\mathcal{E})$ such that $\operatorname{div}(f) = \sum_i [P_i] + [-\sum_i P_i] - m[O]$ as claimed.

Furthermore, by inspection, we see that, regardless of the order in which these merge operations take place, we eventually obtain a function field element h with divisor $\sum_i [P_i] + [-(\sum_i P_i)]$. If the input points $\{P_i\}_i$ are a nontrivial decomposition of O, then $[-(\sum_i P_i)] = [O]$, so $\operatorname{div}(h) = \sum_i [P_i]$ as desired. □

Note that in [Para], the line interpolating P with $-P$ is confused for the line tangent to $\mathcal{E}$ at P, introducing an erroneous point in the purported principal divisors.

Corollary 3.3.4. Under the assumptions of Lemma 3.3.3, the output f is not just a rational function, it is a polynomial (the degree of the denominator is 0).

Proof. Note that, while the fraction $\frac{\ell_{2i}\ell_{2i+1}\widehat{\ell_i}}{\widetilde{\ell_{2i}}\widetilde{\ell_{2i+1}}}$ is in the function field, the factorization described in the proof of Lemma 3.3.3 guarantees that $\frac{\ell_{2i}\ell_{2i+1}\widehat{\ell_i}}{\widetilde{\ell_{2i}}\widetilde{\ell_{2i+1}}}$ reduces to a polynomial. This is true for every iteration. □

The implementation in [Para] performs division, and if the remainder is non-zero, the algorithm fails and terminates; Corollary 3.3.3 demonstrates this check is unnecessary. In the sequel, we assume witnesses output from `Wit` have unit denominators, i.e. if $f = \frac{f_1(X) + f_2(X)Y + (e(X,Y))}{f_3(X) + (e(X,Y))}$, then $\deg(f) = 0$. Over a field, we can assume without losing generality that $f_3(X) + (e(X,Y)) = 1 + (e(X,Y))$, so we write witnesses as $f = f_1(X) + f_2(X)Y$ (leaving coset notation implicit).

Theorem 3.3.5. Let $\mathcal{E}$ be nonsingular with prime-order $\mathbb{G}(\mathcal{E})$ such that $|\mathbb{G}| > 3$. Π is a proving system for the relation $\mathcal{R}'_{\text{div}}$ from Equation 19 with completeness error $\kappa \leq \frac{n}{|\mathbb{G}|-1}$.

Proof. Presume that $f \in K(\mathcal{E})$, $[\nu] = \operatorname{div}(f)$, and the prover honestly executes `Prove`. Then the following occurs.

(a) The prover executes `Prove` step 1 to compute $\texttt{out} \leftarrow \texttt{Wit}(\texttt{params}, [\nu])$. Following Lemma 3.3.3, $\texttt{out} = f \in K(\mathcal{E})$ with certainty, where $\perp \neq f = \frac{f_1(X)+Yf_2(X)+(e(X,Y))}{f_3(X)+(e(X,Y))}$ and $f_3(X) = 1$.

(b) The prover executes `Prove` steps 2 to compute $(P, Q) \leftarrow H(\texttt{params}, f, [\nu])$, interpolates these points, and executes `Prove` step 3 to compute $R = -(P + Q)$. Note the constraints that $P, Q \neq O$ and $P \neq \pm Q$ admit an interpolating line ℓ which is not vertical and not tangent to $\mathcal{E}$, so $R \neq \pm P$ and $R \neq \pm Q$. Thus, the set $\{P, Q, R\}$ is sampled uniformly from the set of all 3-sets of collinear non-identity points on $\mathcal{E}$, i.e. the set $\mathcal{C}$ from Lemma 3.3.1. By Corollary 3.3.2, the probability of failure when computing these two steps is at most $\frac{n}{|\mathbb{G}|-1}$.

(c) Conditioned upon success until this point and an honest prover, parsing in `Prove` step 4 succeeds with certainty, and $f_3(X) = 1$, so the computations in `Prove` step 5 succeed with certainty. Since the prover has not yet failed, each $y'_T \neq \lambda$ and each $f(T) \neq 0$, so the prover succeeds at `Prove` steps 6 and 7 with certainty, outputting π in step 8.

So, `Prove` has accepting probability at least $1 - \frac{n}{|\mathbb{G}|-1}$. If the prover does not fail, then during verification the following occurs.

(a) Conditioned on an honest execution of `Prove`, parsing succeeds with certainty in `Verify` step 1, so the verifier does not output 0 in this step.

(b) Conditioned on an honest prover and a successful execution of `Prove`, the witness is valid $f = \frac{f_1(X)+f_2(Y)+(e)}{1+(e)} \in K(\mathcal{E})$, and the verifier obtains (P, Q) with $P \neq \pm Q$ without outputting 0 in `Verify` step 2.

(c) The verifier computes $R = -(P + Q)$ and checks for a collision. Conditioned on an honest prover and a successful execution of `Prove`, the points P, Q, R do not collide with the points $\{P_i\}$, so the verifier does not output 0 in `Verify` step 2.

(d) Conditioned on an honest prover, a successful execution of `Prove`, and the verifier not yet terminating, then all the computations in `Verify` step 3 succeed with certainty and the verifier obtains the following.

$$b_1 = 2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P) \tag{51}$$
$$c_1 = 2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P) \tag{52}$$
$$b_2 = 2y_P y_R f(R)(3x_P^2 + \alpha - 2y_P\lambda)\Delta X \tag{53}$$
$$c_2 = -2\lambda y_P(2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_P^2 + \alpha - 2y_P(\lambda + \Delta X)) \tag{54}$$
$$b_3 = 2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q) \tag{55}$$
$$c_3 = 2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q) \tag{56}$$
$$b_4 = 2y_Q y_R f(R)(3x_Q^2 + \alpha - 2\lambda y_Q)\Delta X \tag{57}$$
$$c_4 = 2\lambda y_Q(2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_Q^2 + \alpha - 2y_Q(\lambda + \Delta X)) \tag{58}$$

(e) Conditioned on an honest prover, a successful execution of `Prove`, and the verifier not yet terminating, all the following are satisfied with certainty.

$$A_i = (y_i - \lambda x_i - \mu)^{-1} \text{ for each } i \tag{59}$$

$$B_1 = \frac{2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P)}{2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P)} = \frac{c_1}{b_1} \tag{60}$$

$$B_2 = -\frac{2\lambda y_P (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_P^2 + \alpha - 2y_P(\lambda + \Delta X))}{2y_P y_R f(R)(3x_P^2 + \alpha - 2\lambda y_P)\Delta X} = \frac{c_2}{b_2} \tag{61}$$

$$B_3 = \frac{2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q)}{2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q)} = \frac{c_3}{b_3} \tag{62}$$

$$B_4 = \frac{2\lambda y_Q (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_Q^2 + \alpha - 2y_Q(\lambda + \Delta X))}{2y_Q y_R f(R)(3x_Q^2 + \alpha - 2\lambda y_Q)\Delta X} = \frac{c_4}{b_4} \tag{63}$$

Thus, each $(y_i - \lambda x_i - \mu)A_i = 1$ for each $1 \leq i \leq n$ and $b_i B_i = c_i$ for each $1 \leq i \leq 4$. So, the verifier does not output 0 in `Verify` step 4.

(f) In Equation 120 of the Appendix, we show $\delta(\widehat{f}(X_P, Y_P, X_Q, Y_Q)) \mid_{P,Q} = B_1 + B_2 + B_3 + B_4$. We also have $\widehat{f}(x_P, y_P, x_Q, y_Q) = f(P)f(Q)f(R)$, and by Weil's reciprocity, $f(P)f(Q)f(R) = \prod_{i=1}^{n}(y_i - \lambda x_i - \mu)^{m_i}$, where $Y - \lambda X - \mu$ is the line interpolating P, Q, and R. Hence, $\delta(\widehat{f}) \mid_{P,Q} = \sum_{i=1}^{n} \frac{m_i}{y_i - \lambda x_i - \mu}$. The verifier checked above that $(y_i - \lambda x_i - \mu)A_i = 1$, so $\delta(\widehat{f}) \mid_{P,Q} = \sum_i m_i A_i$, implying verification is passed; the verifier outputs 1 with certainty.

By inspection, it is clear that the only case of failure is in the case of a bad sample of (P, Q), which occurs with probability at most $\frac{n}{|G|-1}$. □

We note that the scheme can be completed with respect to $\mathcal{R}'_{\text{div}}$ by allowing provers to include random context strings $\texttt{ctx} \in \{0,1\}^*$ in their proofs for use as prefixes for the hash computing P, Q, resampling `ctx` until `Prove` succeeds. Since the completeness error above is $\frac{n}{|\mathbb{G}|-1}$, the prover can expect to resample once every $\frac{1}{n}(|\mathbb{G}| - 1)$ proofs, because the completeness error is $\frac{n}{|\mathbb{G}|-1}$. However, this completeness error is already negligible in the security parameter governing $|\mathbb{G}|$, so completing the scheme with respect to $\mathcal{R}'_{\text{div}}$ is not practically necessary.

Moreover, the scheme can be extended to all of $\mathcal{R}_{\texttt{div}}$ by modifying `Wit` to output vertical lines given input divisors with $|\text{supp}(\nu)| = 2$, because no single points on $\mathbb{G}(\mathcal{E})$ can be principal divisors for a non-singular $\mathcal{E}$ with $|\mathbb{G}(\mathcal{E})| > 3$. However, checking whether two points are inverses already only requires comparing two elements of K against each other, and comparing two sign/parity bits against each other. In this way, divisors-based proofs cannot improve the efficiency of checking whether a point inverse has been computed correctly; thus, extending to all of $\mathcal{R}_{\texttt{div}}$ is also not practically necessary.

Lemma 3.3.6. Let $\nu : \mathcal{E} \to \mathbb{Z}$ such that $|\text{supp}(\nu)| < \infty$, and let $f \in K(\mathcal{E})^\times$. If, for every $\ell \in K(\mathcal{E})$ with $\deg(\ell) = 1$, $\widehat{f}(\text{div}(\ell)) = \ell([\nu])$, then $[\nu] = \text{div}(f)$.

Proof. Define $\delta : \mathcal{E} \to \mathbb{Z}$ such that $\delta(T) = \nu(T) - \mathrm{ord}_T(f)$ for every $T \in \mathcal{E}$. Then $\mathrm{div}(f) = [\nu] + [\delta] \in \mathrm{Prin}(\mathcal{E})$. Thus, it is sufficient to show that $\delta = 0$.

Assume (in pursuit of a contradiction) that some $P_0 \in \mathrm{supp}(\delta)$. By Weil's reciprocity, $\widehat{f}(\mathrm{div}(\ell)) = \widehat{\ell}(\mathrm{div}(f)) = \widehat{\ell}([\nu] + [\delta])$. By Theorem 2.6.1, $\widehat{\ell}([\nu] + [\delta]) = \widehat{\ell}([\nu]) \cdot \widehat{\ell}([\delta])$. By assumption, $\widehat{f}(\mathrm{div}(\ell)) = \widehat{\ell}([\nu])$. Hence, $\widehat{\ell}([\nu]) = \widehat{\ell}([\nu])\widehat{\ell}([\delta])$, so $1 = \widehat{\ell}([\delta])$. However, this must hold for any line ℓ, even a line ℓ with a root in the support of δ, a contradiction. □

Theorem 3.3.7. Π is a proving system for the relation $\mathcal{R}_{\mathrm{div}}$ from Equation 19 with soundness error $\frac{n}{2|K|(|K|-1)}$.

Proof. We show that if $\nu : \mathbb{G} \setminus \{O\} \to \mathbb{Z}$ has $|\mathrm{supp}(\nu)| = n$ and $\pi \in \frac{K[X,Y]}{(e)} \times K^{n+4}$, say $\pi = (f, A_1, \ldots, A_n, B_1, \ldots, B_4)$, is any proof such that $\texttt{Verify}([\nu], \pi) = 1$, then $[\nu] = \mathrm{div}(f)$ with probability at least $1 - \frac{n}{2|K|(|K|-1)}$.

Assume $\texttt{Verify}([\nu], \pi) = 1$. Then $\pi = (f, A_1, \ldots, A_n, B_1, B_2, B_3, B_4)$ for some $f \in K(\mathcal{E})$ and some $A_1, \ldots, A_n, B_1, B_2, B_3, B_4 \in K$, otherwise the verifier would output 0 in the first step, a contradiction. Moreover, the verifier computes challenge points with the hash function H, say $(P, Q) \leftarrow H(\texttt{params}, f, [\nu])$, such that $P \neq \pm Q$; indeed, otherwise the verifier would output 0 in the second step, a contradiction. The verifier computes $R = -(P + Q)$, and finds that $P \neq \pm R$ and $Q \neq \pm R$ and $\{P, Q, R\} \cap \{P_i\}_{i=1}^n = \emptyset$, otherwise the verifier would output 0 in the third step, a contradiction. Thus, the verifier successfully computes ΔX, ΔY, λ, μ, R, and (b_i, c_i) for $i = 1$, 2, 3, and 4, yielding the following

$$\Delta X = x_Q - x_P \tag{64}$$
$$\Delta Y = y_Q - y_P \tag{65}$$
$$\lambda = \frac{\Delta Y}{\Delta X} \tag{66}$$
$$\mu = y_P - \lambda x_P \tag{67}$$
$$b_1 = 2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P) \tag{68}$$
$$c_1 = 2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P) \tag{69}$$
$$b_2 = 2y_P y_R f(R)(3x_P^2 + \alpha - 2y_P\lambda)\Delta X \tag{70}$$
$$c_2 = -2\lambda y_P(2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_P^2 + \alpha - 2y_P(\lambda + \Delta X)) \tag{71}$$
$$b_3 = 2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q) \tag{72}$$
$$c_3 = 2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q) \tag{73}$$
$$b_4 = \big(2y_Q y_R f(R)(3x_Q^2 + \alpha - 2\lambda y_Q)\Delta X\big) \tag{74}$$
$$c_4 = 2\lambda y_Q(2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_Q^2 + \alpha - 2y_Q(\lambda + \Delta X)) \tag{75}$$

At this point, the verifier finds $(y_i - \lambda x_i - \mu)A_i = 1$ for each $i = 1, 2, \ldots, n$, otherwise the verifier would output 0 after this check. Thus, the verifier is assured each $A_i = (y_i - \lambda x_i - \mu)^{-1}$ with certainty. Then the verifier finds that $b_i B_i = c_i$ for each $i = 1$, 2, 3, and 4, otherwise the verifier would output 0 after this check. Thus, the verifier is assured that each of these satisfy $B_i = c_i / b_i$

with certainty. Lastly, the verifier finds that $B_1 + B_2 + B_3 + B_4 = \sum_i m_i A_i$. Hence, the verifier is assured that the following is satisfied with certainty.

$$\frac{c_1}{b_1} + \frac{c_2}{b_2} + \frac{c_3}{b_3} + \frac{c_4}{b_4} = \sum_i \frac{m_i}{y_i - \lambda x_i \mu}$$

Moreover, since the verifier verified the numerators and denominators in this equation, the verifier concludes the following.

$$\begin{aligned}
&\frac{2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P)}{2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P)} - \frac{2\lambda y_P (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_P^2 + \alpha - 2y_P(\lambda + \Delta X))}{2y_P y_R f(R)(3x_P^2 + \alpha - 2y_P\lambda)\Delta X} \\
&+ \frac{2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q)}{2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q)} + \frac{2\lambda y_Q (2y_R f_X(R) + (3x_R^2 + \alpha) f_Y(R))(3x_Q^2 + \alpha - 2y_Q(\lambda + \Delta X))}{2y_Q y_R f(R)(3x_Q^2 + \alpha - 2\lambda y_Q)\Delta X} \\
&= \sum_i \frac{m_i}{y_i - \lambda x_i \mu}
\end{aligned}$$

By Equations 102 and 103 in the Appendix, the verifier knows the following.

$$\begin{aligned}
\frac{2y_P f_X(P) + (3x_P^2 + \alpha) f_Y(P)}{2y_P f(P)(3x_P^2 + \alpha - 2\lambda y_P)} &= \delta(f(X,Y)) \mid_{(X,Y)=(x_P, y_P)} \\
\frac{2y_Q f_X(Q) + (3x_Q^2 + \alpha) f_Y(Q)}{2y_Q f(Q)(3x_Q^2 + \alpha - 2\lambda y_Q)} &= \delta(f(X,Y)) \mid_{(X,Y)=(x_Q, y_Q)}
\end{aligned}$$

Also, the verifier knows the derivation d (and therefore the logarithmic derivative δ) on $K(\mathcal{E})$ over K extends to the function fields for the product variety $K(\mathcal{E}^2)$. Thus, $\delta(f(X_R, Y_R)) \mid_{(P,Q)}$ is as follows due to Equation 106.

$$\frac{2\lambda(f_X(R) + y'_R f_Y(R))}{(x_Q - x_P) f(R)} \left(\frac{y'_Q - \lambda - (x_Q - x_P)}{y'_Q - \lambda} - \frac{y'_P - \lambda + (x_Q - x_P)}{y'_P - \lambda} \right) = \delta\big(f(X,Y)\big) \mid_{(P,Q)} \quad (76)$$

where in this case, δ is the logarithmic derivative on on $K(\mathcal{E}^2)$. Since δ is a group homomorphism from the multiplicative subgroup of $K(\mathcal{E}^2)$ to the additive group of principal divisors, the verifier knows the sum of these three terms is certainly as follows.

$$\delta\big(f(P)f(Q)f(R)\big) = \delta\big(\widehat{f}(X_P, Y_P, X_Q, Y_Q)\big) \mid_{(X_P, Y_P, X_Q, Y_Q)=(x_P, y_P, x_Q, y_Q)}$$

Hence, the following holds.

$$\delta\big(\widehat{f}(X_P, Y_P, X_Q, Y_Q)\big) \mid_{(X_P, Y_P, X_Q, Y_Q)=(x_P, y_P, x_Q, y_Q)} = \sum_i \frac{m_i}{y_i - \lambda x_i \mu}$$

Of course, the verifier also knows that, for the line ℓ interpolating P and Q, the equality $\sum_i \frac{m_i}{y_i - \lambda x_i \mu} = \sum_i \delta\big(\ell(X_i, Y_i)\big) \mid_{(X_i, Y_i)=(x_i, y_i)}$ holds, where this time δ is the logarithmic derivative on

$K(\mathcal{E}^n)$. As before, this δ is a homomorphism, so the verifier concludes the following equality.

$$\delta\big(\widehat{f}(X_P, Y_P, X_Q, Y_Q)\big) \mid_{(X_P,Y_P,X_Q,Y_Q)=(x_P,y_P,x_Q,y_Q)} = \delta\left(\prod_{i=1}^{n} \ell(X_i, Y_i)^{m_i}\right) \mid_{\{(X_i,Y_i)\}_i=\{(x_i,y_i)\}_i} \quad (77)$$

The verifier recalls that, for a fixed f (and therefore fixed (P_i, m_i) pairs), both $f(P)f(Q)f(R)$ and $\prod_i \ell(X_i, Y_i)^{m_i}$ are rational functions in the variables (X_P, Y_P, X_Q, Y_Q), so (x_P, y_P, x_Q, y_Q) is a root of $f(P)f(Q)f(R) - \prod_i \ell(X_i, Y_i)^{m_i}$.

The Schwartz-Zippel lemma states that, if $g \in K[Z_1, \ldots, Z_n]$, $S \subseteq K^n$ is a finite set, $\vec{z} \xleftarrow{\$} S$ is sampled uniformly at random, and $g(\vec{z}) = 0$, then the probability that $g \neq 0 \in K[\{Z_i\}_i]$ is at most $\frac{n}{|S|}$. This extends to rational functions by finding common denominators.

In this case, the polynomial ring is the coordinate ring of the product variety $\mathcal{E}^2$, namely $K[X_P, Y_P, X_Q, Y_Q]/(e(X_P, Y_P), e(X_Q, Y_Q))$, the sampled evaluation quadruple is (x_P, y_P, x_Q, y_Q), and the rational function is $f(P)f(Q)f(R) - \prod_i \ell(X_i, Y_i)^{m_i}$. The hash function samples (x_P, y_P, x_Q, y_Q) such that $x_P \neq x_Q \in K$; from these, a single bit each is necessary to determine y_P and y_Q. A sample $x_P \neq x_Q$ without regard to order comes from a set with $\frac{1}{2}|K|(|K| - 1)$ elements, and a sample of two bits can be obtained from a set with 4 elements. Thus, $|S| = 2|K|(|K| - 1)$, so the probability that $f(P)f(Q)f(R) - \prod_i \ell(X_i, Y_i)^{m_i} \neq 0$ is at most $\frac{n}{2|K|(|K|-1)}$.

Now the conclusion of the proof is simply a corollary of Lemma 3.3.6. □

3.4 Computational refinements

3.4.1 Seed Trees

A standard improvement utilizes seed trees to effectively communicate the randomness used to generate the verifier's challenge points. Suppose that a prover has N different proofs, where the randomness $\texttt{seed}_i$ was used along with a CSPRNG to generate P_i and Q_i for $i = 1, \ldots, N$. The first improvement is obvious: instead of communicating $\{(P_i, Q_i)\}_{i=1}^N$, the prover can communicate $\{\texttt{seed}_i\}_{i=1}^N$, then the verifier may compute the points themselves from the CSPRNG. As a second improvement, one can go even farther: instead of the prover communicating each $\texttt{seed}_i$ to the verifier separately, they can utilize seed trees, wherein they place a master seed `master_seed` at the root of a Merkle tree, which is then used to generate each seed $\texttt{seed}_i$ at the leaves. In this manner, the prover can communicate the randomness used in their proofs in a way that is logarithmic in N, the total number of proofs. The benefit that this approach provides is most apparent when utilized non-interactively, as it permits for more compact proof sizes.

3.4.2 Fast Polynomial Multiplication

If the scheme only needs to handle divisors with some uniform bound on support cardinalities (i.e. only divisors $[\nu]$ with $|\mathrm{supp}(\nu)| \leq N$ for a fixed $N \in \texttt{params}$), then all witnesses will have bounded degrees. In this case, a careful choice of $\mathrm{char}(K) = p$ may allow number-theoretic transform (NTT) speedups to polynomial multiplication in $K[X, Y]$.

4 Applications

In this section, we highlight a number of practical use cases. By utilizing divisors, these applications benefit from a marked computational improvement to the verifier's checks, however they also enjoy the zero-knowledge property from their underlying zero-knowledge proof.

In this next section, we use $\star$ to denote the entrywise vector product, alternately referred to as the Hadamard or Schur product. We define $\vec{y} = (1, y, \ldots, y^{n-1})$ as the Vandermonde vector for a non-zero field element $y \in K^\times$, which permits the verifier to generate a test vector from a single field element that will be used to ensure the validity of the prover's computations. We refer to [MS25] for a proof that distinct values of y lead to linearly independent test vectors $\vec{y}$, so it is sufficient to query the prover for distinct values. Finally, for a cyclic group $\mathbb{G}$, we store group generators in vectors, namely $G = (G_1, \ldots, G_n)$ where $G_i \in \mathbb{G}$ for each i. Utilizing additive notation then, for $a \in K$, we write aG to denote the weighted sum $\sum_{i=1}^{n} a_i G_i$. This collapses n elliptic curve multiplications into a single point, so knowledge of multiple discrete logarithms may be demonstrated concisely.

4.1 Schnorr identification

To begin, we showcase the classic Schnorr protocol from [Sch91], which is a zero-knowledge proof system that is complete, sound, and zero-knowledge. In Algorithm 4 we describe how the Schnorr identification protocol is modified by the divisor approach presented above. The incorporation of divisors alters the soundness and completeness error negligibly, but accelerates the verifier's final check, and hence reduces the proof size. While it is unfounded to expect that the divisors approach is ever zero-knowledge, the Schnorr protocol enjoys this feature, and hence, this protocol can represent an efficient zero-knowledge proof system.

Improved Verification Schnorr Scheme		
Public: G, $P = a \cdot G$, $B_i = 2^i G$ for $i = 0, \ldots, k$		
Private: $s = (s_0, \ldots, s_k) \in K^{k+1}$ such that $a = s_0 + s_1 2 + \cdots + s_k 2^k$		
Prover		**Verifier**
Sample $r_i \xleftarrow{\$} K$ for $i = 0, \ldots, k$		
Set $\rho = r_0 + r_1 2 + \cdots + r_k 2^k$		
Set $R = \rho G$	$\xrightarrow{R}$	
	$\xleftarrow{c}$	Sample $c \xleftarrow{\$} K^\times$
Form $z = s + cr = (z_0, \ldots, z_k)$		
Set $\nu = \sum_{i=0}^{k} z_i \cdot [B_i] - [P] - c \cdot [R]$		
$\pi \leftarrow \texttt{Prove}(\nu)$	$\xrightarrow{\pi}$	
		Recompute ν
		Check: $\texttt{Verify}(\nu, \pi) = 1$

Algorithm 4: A computational improvement to the verification step in the Schnorr protocol.

In Algorithm 4, we present an improvement upon the traditional Schnorr scheme interactive protocol, leveraging the divisors framework. Utilizing divisors, the prover can replace the traditional last step communicating z with the exhibited communication of the proof π. Then the verifier's check in the above portion - which traditionally requires $k+2$ costly elliptic curve point multiplications - can be replaced with an efficiently computable divisor sum. So with a marginal increase in communication, the verifier can offload the bulk of their computation to the prover.

Theorem 4.1.1. Algorithm 4 is a zero-knowledge proof protocol demonstrating knowledge of a discrete logarithm.

Proof. Because the Schnorr scheme is zero-knowledge, this desirable property is carried over in our divisors approach. More precisely, if any party can extract information about the secret from the divisors, then they could have in the original Schnorr scheme as well. There will be a nominal change to the completeness and soundness error, due to the arguments presented in Section 3.3 (specifically Theorems 3.3.5 and 3.3.7), but this alteration is negligible in $|\mathbb{G}|$, so does not cause problems. □

The result is a 3-pass interactive protocol that preserves the zero-knowledge property of the Schnorr scheme (importantly, the divisor approach presented in [Bas] is not zero-knowledge) while drastically reducing the verifier's computations, with a minimal hit to communication. After applying the Fiat-Shamir heuristic to transform this zero-knowledge proof protocol into a signature scheme, where we reiterate that all the public parameters must be included in the hash input, the proof's verification time will therefore be heavily reduced at the cost of a negligible increase in the proof size.

Additional care will be necessary to account for attacks on adaptive security on threshold Schnorr signatures. Indeed, [CS25] recently highlighted how adaptive adversaries can threaten various schemes without appropriate modifications. That is unfortunately outside of the scope of this work, but given the novelty of the referred work, it was worth mentioning.

This foundation leads to more generic zero-knowledge proof systems in which the verifier may offload computation by avoiding expensive elliptic curve computations, speeding up verification time. We note that the above approach can be applied to the Chaum-Pedersen [CP93] or Okamoto [Oka93] schemes as well, with minimal changes. As such, this particularly pertains to cryptocurrencies like Lelantus Spark [JF21] or Salvium [Pro], or indeed, FCMP++ in Monero [Parb]. Divisors are of marked interest for cryptocurrencies, because transactions must be repeatedly verifiable over the life of the blockchain - a single proof will be verified a great many times, so it behooves the protocol designers to offload the verifier's computations onto the prover, who must only generate the proof once. Below, we highlight how this divisors framework can be used to improve the verification time for the Bulletproofs protocol [BBB$^+$17] specifically, which is implemented in a wide range of cryptocurrency settings. In future work, we hope to showcase how, through MuSig techniques, this can be optimized further such that the prover only needs to send a single batch proof.

4.2 Bulletproofs

We point out that these ideas may also be applied to the verification procedure in the range proof of Bulletproofs [BBB$^+$17]. Bulletproofs offers an astoundingly efficient way to demonstrate that a committed number $v \in K$, for $\mathrm{char}(K) = p$ large enough compared to n, is constrained to some interval $[0,\, 2^n - 1]$. Specifically, it provides a proof of the following relation:

$$\mathcal{R}_R = \{n \in \mathbb{N}, g, h \in \mathbb{G};\ v, \gamma \in K \mid V = vg + \gamma h \text{ and } v \in [0,\, 2^n - 1]\}.$$

The construction is quite clever; we recreate it below for completeness.

The main idea is to utilize the binary representation of a vector, were $a_L \in \{0,\, 1\}^n$ represents the bits of v in binary, so that $\langle a_L, \vec{2}\rangle = v$. If the prover can produce a certificate that a_L is binary, and genuinely represents the binary bits of v, then the verifier can be convinced that $0 \leq v \leq 2^n - 1$. To this end, the prover utilizes Pedersen commitments and inner product arguments to convince the verifier that they possess knowledge of an opening a_L such that

$$\langle a_L, \vec{2}\rangle = v \text{ and } a_R = \mathbb{1} - a_L \text{ and } a_L \star a_R = \mathbb{0}.$$

This first check ensures that a_L represents v, while the second and third guarantee that a_L is a binary vector. We present the setup protocol for Bulletproofs in Algorithm 5, which serves as a precomputation step to the main proof generation and verification protocols in Algorithm 6.

Bulletproofs Setup Protocol		
Public: $g, h \in \mathbb{G},\ \ G, H \in \mathbb{G}^n$ **Private**: $v \in K$		
PROVER		**VERIFIER**
Compute $a_L \in \{0,\, 1\}^n$ s.t. $\langle a_L, \vec{2}\rangle = v$ Set $a_R = \mathbb{1} - a_L$ Sample $\alpha, \rho, \gamma \xleftarrow{\$} K$ and $s_1, s_2 \xleftarrow{\$} K^n$ Set $A = \alpha h + a_L G + a_R H$ and $S = \rho h + s_L G + s_R H$	$\xrightarrow{A,\ S}$ $\xleftarrow{y,\ z}$	Sample $y, z \xleftarrow{\$} K^\times$

Algorithm 5: The setup algorithm for Bulletproofs.

Bulletproofs Proof Protocol		
Public: $g, h \in \mathbb{G}, \quad G, H \in \mathbb{G}^n$ **Private**: $v \in K$		
PROVER		**VERIFIER**
Define $l(x) = (a_L - z\mathbb{1}) + s_L x$, $r(x) = \vec{y} \star (a_R + z\mathbb{1} + s_R x) + z^2\vec{2}$, and set $\langle l(x),\, r(x)\rangle = t_0 + t_1 x + t_2 x^2$ Set $\delta = t_0 - z^2 v$		
For $i = 1, 2$: sample $\tau_i \xleftarrow{\$} K$ and set $T_i = t_i g + \tau_i h$ for $i = 1, 2$	$\xrightarrow{T_1,\, T_2}$	
	$\xleftarrow{x_0}$	Sample $x_0 \xleftarrow{\$} K^\times$
Compute $L = l(x_0)$, $R = r(x_0)$ Set $t = \langle L,\, R\rangle$ Define $\tau = \tau_2 x_0^2 + \tau_1 x_0 + z^2\gamma$ Set $\mu = \alpha + \rho x_0$ Define H' so that $H'_i = y^{-i+1} H_i$ Set $\delta = (z - z^2)\langle \mathbb{1},\, \vec{y}\rangle - z^3\langle \mathbb{1},\, \vec{2}\rangle$ Set $v_1 = [A] + x_0[S] - (z - L)[G]$ $\quad + (z\vec{y} + z^2\vec{2} - R)[H'] - \mu[h]$ Set $v_2 = z^2[V] + (\delta - t)[g] - \tau[h]$ $\quad + x_0[T_1] + x_0^2[T_2]$ Compute $\pi_1 \leftarrow \texttt{Prove}(v_1)$ Compute $\pi_2 \leftarrow \texttt{Prove}(v_2)$	$\xrightarrow{\pi_1, \pi_2}$	
		Parse $v_i \leftarrow \pi_i$ for $i = 1, 2$ Check $\texttt{Verify}(v_1,\, \pi_1) \stackrel{?}{=} 1$ Check $\texttt{Verify}(v_2,\, \pi_2) \stackrel{?}{=} 1$ Check $\langle L,\, R\rangle \stackrel{?}{=} t$

Algorithm 6: The proof generation and verification procedure for Bulletproofs.

Theorem 4.2.1. The Bulletproofs range proof presented in Algorithm 6 enjoys completeness, witness-extended emulation (a more robust notion of soundness), and is zero-knowledge.

For a proof of this claim, we refer to [BBB+17]. Traditionally, the Bulletproofs verification procedure involves three separate checks: the first ensures that $t = t_0 + t_1 x_0 + t_2 x_0^2$, the second that L and R are correct, and the third that t was computed correctly. The third is simply an inner product computation on witnesses that are contained in π_1 and π_2, so can be performed efficiently by the verifier - this is the last line on the right side of the protocol. The first and second checks however involve elliptic curve point multiplications, which are costly. Hence, the divisor framework may be effectively applied to improve the computational overhead inherent in these checks, with only a slight hit to the completeness and soundness error. In the original Bulletproofs paper, the verifier must compute $6n + 11$ elliptic curve point multiplications: $2n + 1$ to take products with S, 2 for T_1, T_2, and V, then n for G and H'. Utilizing divisors, these $6n + 11$ point multiplications can be reduced to divisor additions, which may be verified much more efficiently than computing

the elliptic curve multiplications.

A Deriving verification equations

Much of the narrative herein are used in supporting arguments for our completeness and soundness proofs above. All the derivations here are included in full for reference.

A.1 Logarithmic derivative of function field elements at the zeroes of a line

In this section, we formalize taking the logarithmic derivative of $f(P)f(Q)f(R)$ where $P+Q+R=O$ in the group law. Indeed, for any $T \in \mathcal{E}$, $f(T)$ is just an element of K, and so $dT = 0$. However, if P, Q, R are handled as variable points on the curve $\mathcal{E}$, say by introducing indeterminates X_P, Y_P, X_Q, Y_Q, X_R, and Y_R, then a derivation on $K(\mathcal{E})$ extends naturally to a derivation on $K(\mathcal{E}^3)$. However, given P and Q, there is a unique R satisfying $P+Q+R=O$, so we only need indeterminates X_P, Y_P, X_Q, and Y_Q.

Note that although these are all indeterminates over K, some Y is not indeterminate over $K[X]$. Indeed, Y is uniquely determined by X up to sign. Thus, we could instead use indeterminates X_P, S_P, X_Q, S_Q for indeterminates S_P and S_Q acting as variables for sign bits. However, this can significantly complicate the following formulae. Thus, despite that each Y is almost uniquely determined by each X, we still have four degrees of freedom in the choice of P and Q by using X_P, Y_P, X_Q, and Y_Q, and we instead treat X_R and Y_R as functions of P and Q.

We start with an arbitrary derivation d over K on the function field $K(\mathcal{E})$. Since $Y^2 - X^3 - \alpha X - \beta = 0$, we have $dY = \frac{3X^2+\alpha}{2Y} dX$. Write $Y' = \frac{3X^2+\alpha}{2Y}$ for short. Define the following as functions of $P = (X_P, Y_P)$ and $Q = (X_Q, Y_Q)$.

$$\Delta X = X_Q - X_P \tag{78}$$

$$\Delta Y = Y_Q - Y_P \tag{79}$$

$$\Lambda = \frac{\Delta Y}{\Delta X} \tag{80}$$

$$d\Lambda = \frac{\Delta X d\Delta Y - \Delta Y d\Delta X}{(\Delta X)^2} \tag{81}$$

$$= \frac{d\Delta Y - \Lambda d\Delta X}{\Delta X} \tag{82}$$

$$= \frac{(dY_Q - \Lambda dX_Q) - (dY_P - \Lambda dX_P)}{\Delta X} \tag{83}$$

$$= \frac{\left(Y'_Q - \Lambda\right) dX_Q - \left(Y'_P - \Lambda\right) dX_P}{\Delta X} \tag{84}$$

$$M = Y_P - \Lambda X_P \tag{85}$$

$$\begin{aligned}
dM =& dY_P - \Lambda dX_P - X_P d\Lambda && (86)\\
=& \left(Y'_P - \Lambda\right) dX_P - X_P d\Lambda && (87)\\
=& \left(Y'_P - \Lambda\right) dX_P - X_P \frac{\left(Y'_Q - \Lambda\right) dX_Q - \left(Y'_P - \Lambda\right) dX_P}{\Delta X} && (88)\\
=& \left(Y'_P - \Lambda\right) dX_P - \frac{X_P}{\Delta X}\left(Y'_Q - \Lambda\right) dX_Q + \frac{X_P}{\Delta X}\left(Y'_P - \Lambda\right) dX_P && (89)\\
=& \left(1 + \frac{X_P}{\Delta X}\right)\left(Y'_P - \Lambda\right) dX_P - \frac{X_P}{\Delta X}\left(Y'_Q - \Lambda\right) dX_Q && (90)\\
=& -\frac{X_P}{\Delta X}\left(Y'_Q - \Lambda\right) dX_Q + \frac{X_Q}{\Delta X}\left(Y'_P - \Lambda\right) dX_P && (91)
\end{aligned}$$

$$\begin{aligned}
X_R =& \Lambda^2 - X_P - X_Q && (92)\\
Y_R^2 =& X_R^3 + \alpha X_R + \beta && (93)\\
R =& (X_R, Y_R) && (94)
\end{aligned}$$

$$\begin{aligned}
dX_R =& 2\Lambda d\Lambda - dX_P - dX_Q && (95)\\
=& \frac{2\Lambda}{\Delta X}\Big(\left(Y'_Q - \Lambda\right) dX_Q - \left(Y'_P - \Lambda\right) dX_P\Big) - dX_P - dX_Q && (96)\\
=& \frac{2\Lambda}{\Delta X}\left(Y'_Q - \Lambda - \Delta X\right) dX_Q - \frac{2\Lambda}{\Delta X}\left(Y'_P - \Lambda + \Delta X\right) dX_P && (97)
\end{aligned}$$

$$\begin{aligned}
dY_R =& \frac{dY_R}{dX_R} dX_R && (98)\\
=& \frac{3X_R^2 + \alpha}{2Y_R} dX_R && (99)\\
=& Y'_R dX_R && (100)
\end{aligned}$$

Now we can compute $df(P)$, $df(Q)$, and $df(R)$ as follows. Note our abuse of notation; by $df(P)$, we mean $(d(f))(P)$. That is, the specific function of df evaluated specifically at P rather than the derivation of $f(P)$, which would obviously be 0.

$$df(P) = f_X(P)dX_P + f_Y(P)dY_P \tag{101}$$
$$= (f_X(P) + f_Y(P)Y'_P)dX_P \tag{102}$$
$$df(Q) = (f_X(Q) + f_Y(Q)Y'_Q)dX_Q \tag{103}$$

$$df(R) = f_X(R)dX_R + f_Y(R)dY_R \tag{104}$$
$$= (f_X(R) + Y'_R f_Y(R))dX_R \tag{105}$$
$$df(R) = \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))}{\Delta X}\Big(\left(Y'_Q - \Lambda - \Delta X\right) dX_Q - \left(Y'_P - \Lambda + \Delta X\right) dX_P\Big) \tag{106}$$

We can now compute $\delta(\widehat{f})$ only as a rational function of (X_P, Y_P, X_Q, Y_Q).

$$\delta(\widehat{f}) = \frac{df(P)}{f(P)} + \frac{df(Q)}{f(Q)} + \frac{df(R)}{f(R)} \tag{107}$$
$$= \frac{df(P)}{f(P)} - \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_P - \Lambda - \Delta X)}{f(R)\Delta X} dX_P \tag{108}$$
$$+ \frac{df(Q)}{f(Q)} + \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_Q - \Lambda - \Delta X)}{f(R)\Delta X} dX_Q \tag{109}$$
$$= \left(\frac{f_X(P) + Y'_P f_Y(P)}{f(P)} - \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_P - \Lambda - \Delta X)}{f(R)\Delta X} \right) dX_P \tag{110}$$
$$+ \left(\frac{f_X(Q) + Y'_Q f_X(Q)}{f(Q)} + \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_Q - \Lambda - \Delta X)}{f(R)\Delta X} \right) dX_Q \tag{111}$$

We are particularly interested in the case that $d(Y - \lambda^* X - \mu^*) = 1$ for some fixed $(\lambda^*, \mu^*) \in K^2$.

$$1 = dY - \lambda dX \tag{112}$$
$$1 = (Y' - \lambda)dX \tag{113}$$
$$dX = (Y' - \lambda)^{-1} \tag{114}$$

Note then that $d\Lambda = dM = 0$, and $\delta(\widehat{f})$ becomes the following.

$$\delta(\widehat{f}) = \left(\frac{f_X(P) + Y'_P f_Y(P)}{f(P)} - \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_P - \Lambda - \Delta X)}{f(R)\Delta X}\right) dX_P \tag{115}$$

$$+ \left(\frac{f_X(Q) + Y'_Q f_X(Q)}{f(Q)} + \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_Q - \Lambda - \Delta X)}{f(R)\Delta X}\right) dX_Q \tag{116}$$

$$= \left(\frac{f_X(P) + Y'_P f_Y(P)}{f(P)} - \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_P - \Lambda - \Delta X)}{f(R)\Delta X}\right) (Y'_P - \lambda^*)^{-1} \tag{117}$$

$$+ \left(\frac{f_X(Q) + Y'_Q f_X(Q)}{f(Q)} + \frac{2\Lambda(f_X(R) + Y'_R f_Y(R))(Y'_Q - \Lambda - \Delta X)}{f(R)\Delta X}\right) (Y'_Q - \lambda^*)^{-1} \tag{118}$$

This is the left-hand side of the verification equation. We can now evaluate at any $(X_P, Y_P, X_Q, Y_Q) = (x_P, y_P, x_Q, y_Q)$, setting $\lambda^* = \Lambda(x_P, y_P, x_Q, y_Q)$, $y'_P = \frac{3x_P^2+\alpha}{2y_P}$ and $y'_Q = \frac{3x_Q^2+\alpha}{2y_Q}$ to obtain the following.

$$\delta(\widehat{f}) \mid_{P,Q} = \frac{f_X(P) + y'_P f_Y(P)}{f(P)(y'_P - \lambda^*)} - \frac{2\lambda^*(f_X(R) + y'_R f_Y(R))(y'_P - \lambda - \Delta X)}{f(R)\Delta X(y'_P - \lambda)} \tag{119}$$

$$+ \frac{f_X(Q) + y'_Q f_X(Q)}{f(Q)(y'_Q - \lambda)} + \frac{2\lambda^*(f_X(R) + y'_R f_Y(R))(y'_Q - \lambda - \Delta X)}{f(R)\Delta X(y'_Q - \lambda)} \tag{120}$$

A.2 Logarithmic derivative of a line at the zeroes and poles of a function field element.

In this section, we consider the flip side of Weil reciprocity. Rather than looking at $f(P)f(Q)f(R)$ for a given f as a function of P and Q, we look at $\prod_T \ell(T)^{\nu(T)}$ as a function of coefficients of ℓ.

Indeed, the other side of Weil's reciprocity is $\widehat{\ell}(\mathrm{div}(f)) = \prod_T \ell(T)^{\nu(T)}$. As above, consider the function $\prod_T \ell(T)^{\nu(T)}$ as an evaluation of some $\widehat{\ell}(X_P, Y_P, X_Q, Y_Q) \in F$. Let $\widetilde{\ell} = Y - \Lambda X - M$, where Λ is as in Equation (80) and M is as in Equation (85). Say $\mathrm{div}(f)$ is the principal divisor $\sum_{i=1}^n \nu_i[T_i] - \nu_0[O]$ where $\nu_0 = \sum_{i=1}^n \nu_i$ and each $T_i = (x_i, y_i) \in \mathcal{E}$. Define $\widetilde{\ell}_i = \widetilde{\ell}(x_i, y_i, X_P, Y_P, X_Q, Y_Q) = y_i - \Lambda x_i - M$. Define $\underline{\ell} = \prod_{i=1}^n \widetilde{\ell}_i(T_i)^{\nu_i}$. Then $\ell(\mathrm{div}(f))$ is an evaluation of $\widehat{\ell}$ at $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$.

$$\underline{\ell} = \prod_{i=1}^{n} \widetilde{\ell}_i(T_i)^{\nu_i} \tag{121}$$

$$= \prod_{i=1}^{n} (y_i - \Lambda x_i - M)^{\nu_i} \tag{122}$$

$$\delta(\underline{\ell}) = \sum_{i=1}^{n} \nu_i \frac{d\,(y_i - \Lambda x_i - M)}{y_i - \Lambda x_i - M} \tag{123}$$

$$= \sum_{i} \nu_i \frac{dy_i - \Lambda dx_i - x_i d\Lambda - dM}{y_i - \Lambda x_i - M} \tag{124}$$

$$= \sum_{i} \nu_i \frac{(y_i' - \Lambda) dx_i - x_i d\Lambda - dM}{y_i - \Lambda x_i - M} \tag{125}$$

$$= \sum_{i} \nu_i \frac{1 - x_i d\Lambda - dM}{y_i - \Lambda x_i - M} \tag{126}$$

$$= \sum_{i} \nu_i (y_i - \Lambda x_i - M)^{-1} \tag{127}$$

where we refer to Equations (80), (85), (84), and (91) for Λ, M, $d\Lambda$, and dM, respectively. These x_i, y_i are fixed elements of K, and d is a derivation over K, so $dx_i = dy_i = 0$, explaining the last step.

References

[Bas] Alp Bassa. Soundness Proof for an Interactive Protocol for the Discrete Logarithm Relation.

[BBB+17] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short Proofs for Confidential Transactions and More. Cryptology ePrint Archive, Paper 2017/1066, 2017.

[BPW16] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. Cryptology ePrint Archive, Paper 2016/771, 2016. `https://eprint.iacr.org/2016/771`.

[CP93] David Chaum and Torben Pryds Pedersen. Wallet Databases with Observers. In Ernest F. Brickell, editor, *Advances in Cryptology — CRYPTO' 92*, pages 89–105, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[CS25] Elizabeth Crites and Alistair Stewart. A Plausible Attack on the Adaptive Security of Threshold Schnorr Signatures. Cryptology ePrint Archive, Paper 2025/1001, 2025.

[DMWG23] Quang Dao, Jim Miller, Opal Wright, and Paul Grubbs. Weak Fiat-Shamir Attacks on Modern Proof Systems. Cryptology ePrint Archive, Paper 2023/691, 2023. https://eprint.iacr.org/2023/691.

[Eag22] Liam Eagen. Zero Knowledge Proofs of Elliptic Curve Inner Products from Principal Divisors and Weil Reciprocity. Cryptology ePrint Archive, Paper 2022/596, 2022.

[FS87] Amos Fiat and Adi Shamir. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-47721-7_12.

[Har13] Robin Hartshorne. *Algebraic Geometry*, volume 52. Springer Science & Business Media, 2013.

[JF21] Aram Jivanyan and Aaron Feickert. Lelantus Spark: Secure and Flexible Private Transactions. Cryptology ePrint Archive, Paper 2021/1173, 2021.

[Jon] Marvin Jones. Vac 101: Transforming an Interactive Protocol to a Noninteractive Argument. https://vac.dev/rlog/vac101-fiat-shamir/. Accessed: 2024-10-15.

[Mil] Jim Miller. The Frozen Heart Vulnerability in Bulletproofs. https://blog.trailofbits.com/2022/04/15/the-frozen-heart-vulnerability-in-bulletproofs/. Accessed: 2024-10-15.

[MS25] Felice Manganiello and Freeman Slaughter. HammR: A ZKP Protocol for Fixed Hamming-Weight Restricted-Entry Vectors. Cryptology ePrint Archive, Paper 2025/475, 2025.

[NHB24] Hieu Nguyen, Uyen Ho, and Alex Biryukov. Fiat-Shamir in the Wild. Cryptology ePrint Archive, Paper 2024/1565, 2024. https://eprint.iacr.org/2024/1565.

[Oka93] Tatsuaki Okamoto. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In Ernest F. Brickell, editor, *Advances in Cryptology — CRYPTO' 92*, pages 31–53, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[Para] Luke "Kayaba" Parker. FCMP++. https://github.com/kayabaNerve/fcmp-plus-plus-paper. Accessed: 2025-05-25.

[Parb] Luke "Kayaba" Parker. Full-Chain Membership Proofs Development. https://www.getmonero.org/2024/04/27/fcmps.html. Accessed: 2025-06-11.

[Pro] Salvium Protocol. Salvium. https://salvium.io/. Accessed: 2025-06-03.

[Sch91] C. P. Schnorr. Efficient Signature Generation by Smart Cards. *J. Cryptol.*, 4(3):161–174, January 1991. https://doi.org/10.1007/BF00196725.

[Sil09] Joseph H Silverman. *The Arithmetic of Elliptic Curves*, volume 106. Springer, 2009.