APPLICATIONS OF p-ADIC ANALYSIS IN CRYPTOGRAPHY

KATHERINE LI

Abstract. This paper will discuss the applications of p-adic analysis across a variety of cryptosystems.

1. Introduction

Cryptography is the science of securing information. Symmetric-key cryptography, for the purposes of our discussion, involves the generation of sequences to use as keys. One way of generating these keys, Feedback with Carry Shift Registers (FCSRs), uses the 2-adics to represent and generate pseudorandom sequences.

The fundamental problem of symmetric-key cryptography is that, in order to transmit information, two parties must first transmit the key through a non-encrypted channel, leaving it open to any malicious party. This is the advantage of key exchange protocols such as Diffie–Hellman.

Mostly, modern cryptography occurs over finite fields, but lifting into p-adics can provide essential cryptanalytic insights. The analysis of 2-adic numbers can break FCSR-based stream ciphers that are resistant to other known attacks, and the p-adic field can be used to break certain elliptic-curve cryptosystems as well as, recently, to propose a novel post-quantum isogeny-based cryptography.

2. Feedback with-carry shift registers

Definition 2.1. A *shift register* is a digital circuit that stores and moves data (as binary bits 0 and 1) sequentially. It consists of a series of flip-flops that are triggered by the same clock, such that at each signal the sequence of data is shifted one unit forward, a new bit is input, and the last bit is shifted out.

Definition 2.2. A *linear-feedback shift register (LFSR)* is a shift register whose input bit is determined by a linear function (composed of binary addition, i.e. XOR functions) of its current state.

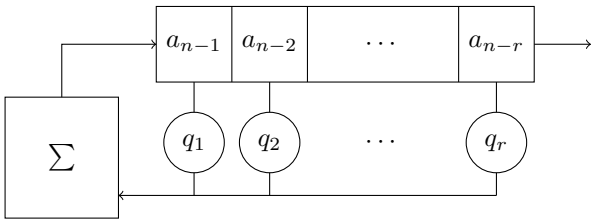

Figure 1. Linear-feedback shift register.

Definition 2.3. A *feedback with carry shift register (FCSR)* is an LFSR together with an additional piece of memory which stores the "carry" of the linear addition.

We can represent an FCSR as a finite-state machine such that state $n-1$ is the pair $(\mathbf{a}_{n-1}; m_{n-1})$, where $\mathbf{a}_{n-1}$ is an array of binary bits $(a_{n-1}, a_{n-2}, \cdots, a_{n-r})$ and m_{n-1} is the memory integer. The function of an FCSR is given by a 2-adic integer $q = -1 + q_1 2^1 + q_2 2^2 + \cdots + q_r 2^r$, called the *connection integer*, that stores the q_i.

Given an FCSR in the state $(\mathbf{a}_{n-1}, m_{n-1})$, computing the next state consists of finding both the input bit a_n and the next memory integer m_n. This is accomplished as follows:

S1. Compute the integer sum $\sigma_n = m_{n-1} + \sum_{i=1}^{r} q_i a_{n-i}$.
S2. Find $a_n \equiv \sigma_n \pmod 2$.
S3. Find $m_n = \lfloor \frac{\sigma_n}{2} \rfloor = \frac{\sigma_n - a_n}{2}$.

Date: June 7, 2026.

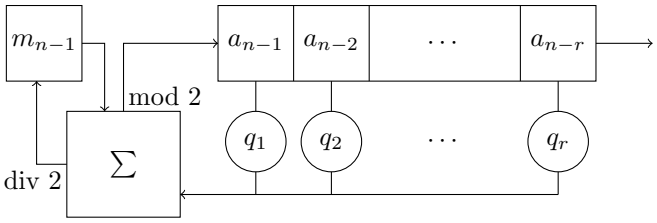

FIGURE 2. Feedback with-carry shift register.

Finally, the register is shifted one step to the right and a_{n-r} is shifted out; the state becomes

$$(a_n, a_{n-1}, \dots, a_{n-r+1}; m_n).$$

We define the infinite and eventually periodic sequence $\mathbf{a} = (a_0, a_1, \dots)$ as the *output* of the FCSR and store it in a rational 2-adic number $\alpha = \sum_{i=0}^{\infty} a_i 2^i$.

Definition 2.4. A *periodic state* of an FCSR is a state, a specification of $\mathbf{a}_n$ together with m_n, such that the FCSR will eventually return to that same state.

Theorem 2.5. *Given the initial state (initial loading) $(a_{r-1}, a_{r-2}, \dots, a_0; m_{r-1})$ of an FCSR, we define*

$$p = -m_{r-1}2^r + \sum_{i=0}^{r-1}\sum_{j=0}^{i} q_j a_{i-j} 2^i$$

and $q = -1 + \sum_{i=0}^{r} q_i 2^i$. Then $\alpha = p/q$.

Proof. Suppose we are calculating with memory m_n. We have (S1)

$$\sigma_n = m_{n-1} + \sum_{i=1}^{r} q_i a_{n-i} = 2m_n + a_n,$$

so

$$a_n = \sigma_n - 2m_n = m_{n-1} - 2m_n + \sum_{i=1}^{r} q_i a_{n-i}.$$

Thus

$$\begin{aligned}\alpha &= a_0 + a_1 2^1 + \cdots + a_{r-1}2^{r-1} + \sum_{n=r}^{\infty} a_n 2^n \\ &= \mathbf{a}_{r-1} + \sum_{n=r}^{\infty}\left(\sum_{i=1}^{r} q_i a_{n-i}\right) 2^n + \sum_{n=r}^{\infty}(m_{n-1} - 2m_n)2^n.\end{aligned}$$

The second summation telescopes to $m_{r-1}2^r$, so

$$\begin{aligned}\alpha &= \mathbf{a}_{r-1} + m_{r-1}2^r + \sum_{n=r}^{\infty}\sum_{i=1}^{r} q_i 2^i a_{n-i} 2^{n-i} \\ &= \mathbf{a}_{r-1} + m_{r-1}2^r + \sum_{i=1}^{r} q_i 2^i \left(\sum_{n=r}^{\infty} a_{n-i} 2^{n-i}\right) \\ &= \mathbf{a}_{r-1} + m_{r-1}2^r + \sum_{i=1}^{r} q_i 2^i \left(\alpha - \sum_{j=0}^{r-i-1} a_j 2^j\right) \\ &= \mathbf{a}_{r-1} + m_{r-1}2^r + \alpha\sum_{i=1}^{r} q_i 2^i - \sum_{i=1}^{r-1}\sum_{j=0}^{r-i-1} q_i 2^i a_j 2^j \\ &= m_{r-1}2^r + \alpha\sum_{i=1}^{r} q_i 2^i - \sum_{i=0}^{r-1}\sum_{j=0}^{r-i-1} q_i 2^i a_j 2^j.\end{aligned}$$

Finally, solving for α yields

$$\begin{aligned}\alpha &= \frac{\sum_{i=0}^{r-1}\sum_{j=0}^{r-i-1} q_i a_j 2^{i+j} - m_{r-1}2^r}{q} \\ &= \frac{\sum_{k=0}^{r-1}\sum_{i=0}^{k} q_i a_{k-i} 2^k - m_{r-1}2^r}{q} \\ &= \frac{p}{q},\end{aligned}$$

as desired. □

This means that we can compute, arithmetically, the 2-adic number of the output of any FCSR directly from its initial state.

Definition 2.6. The *size* of an FCSR with connection integer q, register size r, and initial memory m is given by

$$\lambda = r + \max\left(\lfloor \log_2(\mathrm{wt}(q+1))\rfloor + 1, \lfloor \log_2 |m|\rfloor + 1\right) + 1,$$

where $\mathrm{wt}(q+1)$ is the Hamming weight of $q+1 = q_1 2^1 + q_2 2^2 + \cdots + q_r 2^r$, i.e. the number of q_i that are 1. This sums the number of bits in the register and the total number of bits allotted to the carry, which must be larger than both the number of bits in the current memory $\lceil \log_2 |m|\rceil$ and the potential for growth $\lceil \log_2(\mathrm{wt}(q+1))\rceil$.

Definition 2.7. The 2-*adic span* of a $\mathbf{a} = (a_0, a_1, \dots)$ is denoted $\lambda_2(a)$ and is the smallest λ which occurs across all FCSRs that generate $\mathbf{a}$.

Definition 2.8. The *2-adic complexity* of a $\mathbf{a} = (a_0, a_1, \dots)$ is $\varphi_2(a) = \log_2(\Phi(p,q))$, where $\Phi(p,q) = \max(|p|, |q|)$.

Proposition 2.9. *The* 2*-adic span and complexity are related by*

$$|\lambda_2(\mathbf{a}) - 2 - \varphi_2(\mathbf{a})| \le \log_2(\varphi_2(\mathbf{a})).$$

Based on this proposition, we have "quasi-triangle inequalities" for both the span and the complexity.

Theorem 2.10. *Suppose* $\mathbf{a}$ *and* $\mathbf{b}$ *are periodic binary sequences. Then,*

$$\varphi_2(\mathbf{a}+\mathbf{b}) \le \varphi_2(\mathbf{a}) + \varphi_2(\mathbf{b}) + 1$$

and

$$\lambda_2(\mathbf{a}+\mathbf{b}) \le \lambda_2(\mathbf{a}) + \lambda_2(\mathbf{b}) + 2\log_2(\lambda_2(\mathbf{a})) + 2\log_2(\lambda_2(\mathbf{b})) + 2.$$

Ultimately, 2-adic analysis of FCSRs is used in cryptanalysis to break the summation cipher, a type of stream cipher with a pseudorandom key yielded by a summation generator.

Definition 2.11. A *stream cipher* is a symmetric key cipher in which plaintext digits are combined with a pseudorandom keystream to form the ciphertext.

Definition 2.12. A *summation generator* generates a pseudorandom sequence by adding, with carry, the sequences $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_k$ of several LFSRs with maximal period $T = 2^r - 1$.

If the sequences have periods T_i that are coprime, the resulting sequence has linear span (smallest possible size of a generating LFSR) which is close to its period $L = T_1 \cdot T_2 \cdots T_k$. This span quickly becomes very large, as the with-carry addition obscures the linear structure. This makes the summation cipher resistant to attacks based on the Berlekamp–Massey algorithm, which finds the shortest LFSR for a given output sequence.

However, by theorem 2.10, we find that the 2-adic span is much smaller: it is only around $S = \sum T_i + \log_2(k) + \log_2(\sum T_i + \log_2(k))$. This makes sense because the with-carry addition is natural on FCSR sequences. Thus, if we can determine enough (namely $\propto 2 \cdot S$, much less than the actual period) consecutive bits of the pseudorandom sequence, we can find the unique FCSR that matches it, which we then know will generate the rest of the sequence.

We can determine the necessary reference bits by standard cryptographic methods, e.g. a known-plaintext attack provided with a crib (a short, known sequence of plaintext).

Theorem 2.13. *The* rational approximation algorithm*, as developed by Klapper and Goresky, outputs $g = (p, q)$ with smallest Φ such that the digit expansion of the 2-adic number $\alpha = p/q$ generates* **a***. We present it here, but the full proof is given in [KlaGor97].*

input a_i's until the first nonzero a_{k-1} is found
$\alpha = a_{k-1} \cdot 2^{k-1}$
$f = (0, 2)$
$g = (2^{k-1}, 1)$
while there are more bits **do**
  input a new bit a_k
  $\alpha = \alpha + a_k 2^k$
  if $\alpha \cdot g_2 - g_1 \equiv 0 \pmod{2^{k+1}}$ **then**
    $f = 2f$
  else if $\Phi(g) < \Phi(f)$ **then**
    Let d be odd and minimize $\Phi(f + dg)$
    $\langle g, f\rangle = \langle f + dg, 2g\rangle$
  else
    Let d be odd and minimize $\Phi(g + df)$
    $\langle g, f\rangle = \langle g + df, 2f\rangle$
  end
  $k = k + 1$
end
return g

Algorithm 1: Rational Approximation

As long as at least $\lceil 2\varphi_2(\mathbf{a})\rceil + 2$ bits are used, the binary expansion of $q + 1$ gives the desired FCSR, and $\alpha = p/q$ is the desired digit stream.

Crucially, this takes up much less space than Berlekamp–Massey on the summation cipher, and is analogous to Berlekamp–Massey on ordinary stream ciphers.

3. Elliptic-curve cryptography

Modern cryptography, roughly, involves the setting of computationally hard (high time-complexity) problems in order to ensure security. Historically, most ciphers have depended upon the prerequisite communication of a shared key in order to encrypt and decrypt messages. In the computer age, however, this is much more complicated—how do we facilitate secure key exchange?

Suppose Alice wants to send a secret message to Bob that can be intercepted by the nefarious Eve. Instead of passing a shared key through an insecure channel, they can use the *Diffie–Hellman key exchange* method to generate a symmetric key through the public channel such that the information passed through the channel alone is not enough to generate the key.

Definition 3.1 (Diffie–Hellman key exchange over $\mathbb{F}_p$)**.** Alice and Bob agree on a prime p and a base g, which is a primitive root modulo p. These are public. Alice then chooses a private key a and computes $g^a \pmod p$; likewise, Bob chooses a private key b and computes $g^b \pmod p$.

Alice sends g^a to Bob across the public channel and Bob sends g^b to Alice. With this information, each of them can find their shared key g^{ab} by raising g^b to the power of a and g^a to the power of b, respectively. However, nowhere in this process are either of their secret keys published.

This exchange is secure because the only pieces of information available to an interceptor, Eve, are p, g, g^a, and g^b. In order to find g^{ab}, Eve must compute a or b; that is, she must take $\log_g(g^a)$ in the cyclic group $\mathbb{F}_p^\times$. This is known to be much more computationally difficult than the process of exponentiating g.

Theorem 3.2 (Discrete logarithm problem)**.** *The generalization of the logarithm $\log_b(a) = x \iff b^x = a$ to cyclic groups is computationally intractable; that is, there is no known polynomial-time algorithm for solving it for x in general.*

One way of accomplishing this key exchange is by using the algebraic structure of elliptic curves.

Definition 3.3. An *elliptic curve* is the graph of an equation of the form $y^2 = x^3 + ax + b$, where a and b are such that $4a^3 + 27b^2 \neq 0$ (i.e. the RHS polynomial has no repeated roots) together with the point at infinity $\mathcal{O}$.

These curves are useful because we can construct an abelian group taking the points as elements under an operation called *addition*.

Definition 3.4 (Elliptic curve addition)**.** Consider points P and Q of an elliptic curve $E : x^3 + ax + b$. The line $\overline{PQ}$ intersects the elliptic curve at a third point. Then we define $P + Q$ as the reflection of this point over the x-axis. Note that we compute $2P = P + P$ by taking the tangent to the elliptic curve at P. We denote by $[n]P$ the addition $\underbrace{P + P + \cdots + P}_{n \text{ times}}$. Given $P = (x_p, y_p)$ and $Q = (x_q, y_q)$, we can use this method to find a formula for $P + Q = R = (x_r, y_r)$. Note that $\overline{PQ}$ has slope

$$m = \frac{y_p - y_q}{x_p - x_q}$$

and passes through P, so it has equation

$$y - y_p = m(x - x_p).$$

We substitute for y in the equation of E. The x-coordinates of the points on $\overline{PQ} \cap E$ are the solutions to

$$(mx - mx_p + y_p)^2 = x^3 + ax + b.$$

Rearranging yields

$$x^3 - m^2x^2 + (a + 2mx_p - 2y_p)x + b - y_p^2 = 0,$$

so $x_r = m^2 - x_p - x_q$ and $y_r = -(mx_r - mx_p + y_p)$.

Once we have defined a formula for addition, we can extend it to elliptic curves over any field.

Definition 3.5. An *elliptic curve over* $\mathbb{F}_p$ denoted $E(\mathbb{F}_p)$ is the set of points (x, y) which satisfy $y^2 \equiv x^3 + ax + b \pmod{p}$ for coefficients $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$.

Definition 3.6. An *elliptic curve over* $\mathbb{Q}_p$ denoted $E(\mathbb{Q}_p)$ is the set of points $(x, y) \in \mathbb{Q}_p^2$ such that $y^2 = x^3 + ax + b$ for $a, b \in \mathbb{Q}_p, 4a^3 + 27b^2 \neq 0$.

How is this relevant to cryptography? We can use the cyclic group over $E(\mathbb{F}_p)$ to construct a cryptosystem with Diffie–Hellman whose discrete logarithm is more difficult to solve than that over $\mathbb{F}_p$.

Definition 3.7 (Diffie–Hellman over $E(\mathbb{F}_p)$)**.** Alice and Bob pick a prime p, an elliptic curve E over $\mathbb{F}_p$, and a point P, publicly. Alice has a private key a and Bob has a private key b. Alice sends $[a]P$ to Bob and Bob sends $[b]P$ to Alice; then they can both compute their shared key $[ab]P$.

In the general case, the discrete logarithm over the group $E(\mathbb{F}_p)$ cannot be computed in polynomial time, so Eve cannot compute $[ab]P$ using the information passed through the public channel.

4. Smart's attack

For special elliptic curves, we can use Hensel's lemma to lift points in $E(\mathbb{F}_p)$ into the p-adic field, allowing us to solve the discrete logarithm problem in linear time.

Definition 4.1. The *trace of Frobenius* of an elliptic curve $E(\mathbb{F}_p)$ is

$$a_p = p + 1 - \#E(\mathbb{F}_p).$$

We are interested in elliptic curves over $\mathbb{F}_p$ of trace of Frobenius one (called *anomalous* curves). Consider $E(\mathbb{F}_p) : y^2 = x^3 + ax + b$ such that there are p points on E. Given two points P' and Q' on the curve, we can solve the discrete logarithm problem

$$Q' = [m]P'.$$

using a method known as Smart's attack [Sma99]. First, we compute a lift of P' and Q' on $E(\mathbb{F}_p)$ to P and Q on $E(\mathbb{Q}_p)$ with the same equation. We take P to have the same x-coordinate as P' and compute the y-coordinate by Hensel lifting the y-coordinate of P' into a solution of $y^2 = x^3 + ax + b$ into the p-adics. Similarly, we can find Q.

Let's pause here to establish some machinery.

Definition 4.2. The group $E_1(\mathbb{Q}_p)$ is the kernel of the restriction of $E(\mathbb{Q}_p)$ to $E(\mathbb{F}_p)$; that is, it is the subgroup of elements in $E(\mathbb{Q}_p)$ that are mapped to the identity $0 \in E(\mathbb{F}_p)$. In general,

$$E_n(\mathbb{Q}_p) = \{(x, y) \in E(\mathbb{Q}_p) : v_p(-x/y) \geq n\}.$$

In other words, $E_n(\mathbb{Q}_p)$ is the set of points in $E(\mathbb{Q}_p)$ equivalent to 0 modulo p^n. As n increases, this specification becomes finer:

$$E(\mathbb{Q}_p) \supset E_1(\mathbb{Q}_p) \supset E_2(\mathbb{Q}_p) \supset \cdots \supset E_n(\mathbb{Q}_p) \supset \cdots .$$

Moreover,

$$E(\mathbb{Q}_p)/E_1(\mathbb{Q}_p) \cong E(\mathbb{F}_p),$$

and for $n \geq 1$,

$$E_n(\mathbb{Q}_p)/E_{n+1}(\mathbb{Q}_p) \cong \mathbb{F}_p^+.$$

Definition 4.3. The p-adic elliptic logarithm $\log_E$ is a mapping $E_1(\mathbb{Q}_p) \to p\mathbb{Z}_p$ such that $\log_E(P + Q) = \log_E(P) + \log_E(Q)$. This means that it is a group homomorphism. In general, it maps $E_n(\mathbb{Q}_p)$ to $p^n\mathbb{F}_p$.

Theorem 4.4 (Formal group logarithm **.** *Sil09] For $P = (x, y) \in E_1(\mathbb{Q}_p)$, the value $\log_E(P) = \ell(z(P))$ where $z(P) = -x/y$ may be computed by evaluating the convergent power series of $\ell(T)$, the formal logarithm of E, at P. Crucially, this can be done in polynomial time.*

Returning to our computation, we have $Q' \equiv [m]P \pmod P \implies Q \equiv [m]P \pmod p$, so

$$Q - [m]P = R \in E_1(\mathbb{Q}_p).$$

Note that Q, P are in $E(\mathbb{Q}_p)$ but R is in $E_1(\mathbb{Q}_p)$. Since $\#E(\mathbb{F}_p) = \#\mathbb{F}_p^+ = p$, we can scale both sides by p such that the summands of the LHS go to $E_1(\mathbb{Q}_p)$ and the RHS goes to $E_2(\mathbb{Q}_p)$. We get

$$[p]Q - [m][p]P = [p]R \in E_2(\mathbb{Q}_p).$$

Now, we can take $\log_E$ of both sides:

$$\log_E([p]Q) - m\log_E([p]P) = \log_E([p]R) \equiv 0 \pmod{p^2}.$$

Thus, since $\log_E([p]Q)$ and $m\log_E([p]P)$ are in $p\mathbb{Z}_p$ but not $p^2\mathbb{Z}_p$, we have

$$m \equiv \frac{\log_E([p]Q)}{\log_E([p]P)} \pmod p,$$

which, by theorem 4.4, is computable in polynomial time.

5. Isogeny-based cryptography

The development of (theoretical) quantum computers opens a new frontier of cryptography in that quantum computing algorithms break many of the most prevalent existing cryptosystems. In particular, Shor's algorithm can solve the discrete logarithm problem on any finite abelian group in polynomial time by reducing it to a period-finding problem, making elliptic curve cryptography insecure.

In this light, the current objective of cryptographers is to design new frameworks that will be resistant to quantum computing. One main approach relies on isogenies between elliptic curves.

Definition 5.1. An *isogeny* between two elliptic curves E and E' is a morphism that sends the identity of E to the identity of E' and has a finite kernel.

Theorem 5.2. *Every finite subgroup G of an elliptic curve E determines an isogeny $\phi : E \to E/G$ with kernel G.*

We will accept this theorem here intuitively, but it is proven in [Sil09, Ch. III.4]. There is an explicit set of formulae to compute, given G, the isogeny ϕ, developed by Vélu [Vél71]. For our purposes, it is only necessary to know that they exist.

Definition 5.3. The *j-invariant* of an elliptic curve E with Weierstrass equation

$$y^2 = x^3 + ax + b$$

is

$$j(E) = 1728\frac{4a^3}{4a^3 + 27b^2}.$$

Definition 5.4. The n-torsion group of an elliptic curve E over a field K is the set of points

$$E[n] \coloneqq \{P \in E(\overline{K}) \mid nP = \mathcal{O}\},$$

where $\mathcal{O}$ is the identity (point at infinity) and $\overline{K}$ is the algebraic closure of K. The elements of $E[n]$ are called n-torsion points.

Definition 5.5. The *characteristic* of a field K is the

Theorem 5.6. *If some prime p is the characteristic of a field K, then for an elliptic curve $E(K)$ we have*

$$E[p^i] \cong \begin{cases} \mathbb{Z}/p^i\mathbb{Z} & \text{for all } i \geq 0, \text{ or} \\ \{\mathcal{O}\} & \text{for all } i \geq 0 \end{cases}.$$

Definition 5.7. A *supersingular elliptic curve* is an elliptic curve E such that

$$E[p^i] \cong \{\mathcal{O}\}.$$

Roughly, the underlying hard problem of isogeny-based cryptosystems is to find, given two elliptic curves E and E', the isogeny $\phi : E \to E'$. This is called the supersingular isogeny problem (SIP).

Supersingular Isogeny Diffie–Hellman (SIDH) was a proposal for such a cryptosystem, a post-quantum analogue of Diffie–Hellman.

Definition 5.8 (SIDH)**.** We are given a prime of the form

$$p = p_A^{e_A} \cdot p_B^{e_B} \cdot f \pm 1,$$

where p_A and p_B are small primes and e_A and e_B are large exponents. Usually, in implementation, we work with $p_A = 2$, $p_B = 3$.

Then, take a supersingular elliptic curve $E(\mathbb{F}_{p^2})$ and choose some points P_A, Q_A with order $p_A^{e_A}$ and P_B, Q_B with order $p_B^{e_B}$ on E. Finally, Alice chooses a secret scalar m_A and Bob chooses a secret scalar m_B.

Alice computes the point $R_A = P_A + [m_A]Q_A$. Then, she generates from R_A the subgroup

$$\langle R_A \rangle = \{\mathcal{O}, R_A, 2R_A, 3R_A, \dots\}.$$

By theorem 5.2, Alice can find an isogeny $\phi_A : E \to E/\langle R_A \rangle$. Similarly, Bob computes $\phi_B : E \to E/\langle R_B \rangle$.

To establish a shared key, Alice applies ϕ_A to P_B and Q_B and publishes E_A, $\phi_A(P_B)$, and $\phi_A(Q_B)$. Bob applies ϕ_B to P_A and Q_A and publishes E_B, $\phi_B(P_A)$, and $\phi_B(Q_A)$. Finally, Alice computes an isogeny φ_{BA} from the kernel generated by

$$S_{BA} \coloneqq \phi_B(P_A) + [m_A]\phi_B(Q_A)$$

on the curve E_B and creates the elliptic curve $E/\langle S_{BA} \rangle$ isogenous to E; similarly, Bob's S_{AB} yields the elliptic curve $E/\langle S_{AB} \rangle$. The two curves curves are isomorphic (essentially, they can be thought of as the two isogenies applied twice in opposite order), and therefore have the same j-invariant K. This K is the shared secret key.

Decrypting the message would require solving SIP for either Alice or Bob.

Isogeny-based cryptosystems are considered promising because, as of yet, there is no known Shor-type algorithm that breaks their underlying hard problem. However, in 2022, SIDH (and consequently SIKE, the key encapsulation mechanism that incorporates SIDH into a public-key system) was broken by the devastating non-quantum Castryck–Decru attack [CasDec23]. The vulnerability in the SIDH architecture is that each party, say Alice, publishes not only E_A but also $\phi_A(P_B)$ and $\phi_A(Q_B)$ (auxiliary torsion images), revealing information about how the isogeny ϕ_A acts.

One cryptosystem proposed to avoid this attack technique was published very recently in 2025 [CheDey25]. It uses the same SIDH-style structure, but sets it in the characteristic-zero field $\mathbb{Q}_p((t))$ of Laurent series with p-adic coefficients.

Definition 5.9 (Laurent series over $\mathbb{Q}_p$)**.** The Laurent-series field is defined

$$\mathbb{Q}_p((t)) \coloneqq \left\{ f(t) = \sum_{n=-k}^{\infty} a_n t^n \,\middle|\, a_n \in \mathbb{Q}_p, k \in \mathbb{N} \right\}.$$

This encodes two dimensions of information: the t-variable parametrizes the series and the p-adic valuation specifies the arithmetic precision of the coefficients.

We then consider an elliptic curve whose coefficients in its Weierstrass equation are elements of $\mathbb{Q}_p((t))$.

Definition 5.10. An elliptic curve E over $\mathbb{Q}_p((t))$ is defined as

$$E : y^2 = x^3 + A(t)x + B(t); A(t), B(t) \in \mathbb{Z}_p[[t]] \subset \mathbb{Q}_p((t)),$$

where $\mathbb{Z}_p[[t]]$ is the ring in $\mathbb{Q}_p((t))$ of formal power series with coefficients in $\mathbb{Z}_p$; i.e.,

$$\mathbb{Z}_p[[t]] \coloneqq \left\{ f(t) = \sum_{n=0}^{\infty} a_n t^n \Big| a_n \in \mathbb{Z}_p \right\}.$$

In the new scheme, we first restrict the coefficients to the integers so that we can reduce the elliptic curve modulo p:

$$\widetilde{E} : y^2 = x^3 + \widetilde{A}(t)x + \widetilde{B}(t),$$

which is defined over the Laurent series field $\mathbb{F}_p((t))$. This field has characteristic p, so we can find torsion points as in SIDH.

For a simple example, let's take $p > 3$, $B(t) = 1 + pt$ so that $E : y^2 = x^3 + 1 + pt \implies \widetilde{E} : y^2 = x^3 + 1$ and find a 2-torsion point. Such a point T is specified by $2T = \mathcal{O}$; one solution occurs when $y = 0$ and the line tangent to the elliptic curve at T is vertical. Thus we want to solve $f(x) = x^3 + 1 + pt = 0$ or $x = (-(1 + pt))^{1/3}$. If $p \equiv 1 \pmod 3$, a cube root $\zeta_3 \in \mathbb{F}_p^\times$ exists, so the constant series $\zeta_3 \in \mathbb{F}_p((t))$ is a 2-torsion point in the reduction $\widetilde{E}$.

Then, we want to lift this root into $\mathbb{Q}_p((t))$ (since the characteristic of $\mathbb{Q}_p((t))$ is 0, we could not compute a torsion point directly). Let $a_0 = \zeta_3$; then $f(a_0) \equiv 0 \pmod p$ and $f'(a_0) \equiv 3\zeta_3^2 \not\equiv 0 \pmod p$. Thus, by Hensel's lemma, there exists a unique lift $\alpha = x(t) \in \mathbb{Q}_p((t))$ that we can compute as

$$x(t) = \zeta_3(1 + pt)^{1/3} = \zeta_3 \left(1 + \frac{pt}{3} - \frac{(pt)^2}{9} + \cdots\right).$$

Then $T = (x(t), 0)$ is our 2-torsion point.

What is necessary about the p-adic environment is that it allows us to encode $x(t)$ as a truncated Engel series with coefficients in $\mathbb{Z}_p[[t]]$.

Definition 5.11 (Engel expansion)**.** The *Engel expansion* of a number x is the unique non-decreasing sequence of integers $\{a_k\}$ such that

$$x = \frac{1}{a_1} + \frac{1}{a_1 a_2} + \frac{1}{a_1 a_2 a_3} + \cdots = \sum_{n=1}^{\infty} \frac{1}{\prod_{i=1}^{n} a_i}.$$

The greedy algorithm for constructing $\{a_i\}$ is given by

$$\begin{aligned} u_1 &= x, \\ u_{k+1} &= u_k a_k - 1, \\ a_k &= \left\lceil \frac{1}{u_k} \right\rceil. \end{aligned}$$

We use similar expansions in $\mathbb{Q}_p((t))$ to represent elements of $\mathbb{Q}_p((t))$ to specify degrees of accuracy.

Definition 5.12 (Engel expansions on $\mathbb{Q}_p(t)$)**.** For $f(t) \in \mathbb{Q}_p((t))$, an Engel expansion is a convergent series

$$f(t) = \sum_{n=1}^{\infty} \frac{1}{a_1(t) a_2(t) \cdots a_n(t)},$$

where

$$a_i(t) \in \mathbb{Z}_p[[t]], a_i(0) \equiv 1 \pmod p.$$

Just as in 5.12, we construct the sequence $a_1(t)$ by

$$\begin{aligned} a_1(t) &= \lfloor 1/f(t) \rfloor_p, \\ a_{k+1}(t) &= \left\lfloor \frac{1}{f(t)} - \sum_{i=1}^{k} \frac{1}{\prod_{j=1}^{i} a_j(t)} \right\rfloor, \end{aligned}$$

where $\lfloor \cdot \rfloor_p$ extracts the p-adic leading term (takes the series coefficient-wise $\pmod p$).

In order to perform a SIDH-style exchange, we write $x(t)$ of torsion point P as an Engel series and store the Engel expansion $\{a_i(t)\}$ truncated at a certain i. Instead of publishing the image of the torsion point under isogeny, we publish this Engel encoding. This adds a second hard problem, concerning the inversion of the Engel expansion, to the process of decryption.

Definition 5.13. The *Engel-expansion inversion problem (EEIP)* is the problem of recovering $x(t)$ from its Engel expansion $\{a_i(t)\}$. There is no apparent reduction of it to a period-finding algorithm, which is vulnerable to Shor's algorithm.

As per [CheDey25], there is no apparent reduction of EEIP to a period-finding algorithm. Moreover, while the Castryck–Decru attack exploited the algebraic structures exposed by the publication of the torsion images $\phi_B(P_A)$, $\phi_B(Q_A)$ etc., the new framework publishes only the truncated Engel coefficients associated with the torsion points (note that the torsion groups (and thereby the isogenies) associated with these points are the secret keys), so it is not directly vulnerable to the known attack. However, it is not yet known whether it can be considered unconditionally hard or secure.

The isogeny-based cryptography proposed by Cherkaoui et al. is relatively recent and yet to be subject to the cryptanalytic attack for which time will provide opportunity, as post-quantum cryptography remains an active area of research.

References

[CasDec23] Wouter Castryck and Thomas Decru. "An Efficient Key Recovery Attack on SIDH". In: *Advances in Cryptology – EUROCRYPT 2023*. Lecture Notes in Computer Science 14004 (2023), pp. 423–447.

[CheDey25] Ilias Cherkaoui and Indrakshi Dey. *Engel p-adic Isogeny-based Cryptography over Laurent Series: Foundations, Security, and an ESP32 Implementation*. 2025. arXiv: 2511.20533 [cs.CR].

[KlaGor97] Andrew Klapper and Mark Goresky. "Feedback Shift Registers, 2-Adic Span, and Combiners with Memory". In: *Journal of Cryptology* 10.2 (1997), pp. 111–147. DOI: 10.1007/s001459900022. URL: https://doi.org/10.1007/s001459900022.

[Sil09] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. 2nd ed. Vol. 106. Graduate Texts in Mathematics. Springer, 2009.

[Sma99] Nigel P. Smart. "The Discrete Logarithm Problem on Elliptic Curves of Trace One". In: *Journal of Cryptology* 12.3 (1999), pp. 193–196. DOI: 10.1007/s001459900052. URL: https://doi.org/10.1007/s001459900052.

[Vél71] Jacques Vélu. "Isogénies entre courbes elliptiques". In: *Comptes Rendus de l'Académie des Sciences de Paris* 273 (1971), pp. 238–241.