Fast $p$-adic arithmetic for (hyper)elliptic AGM point counting algorithms

R. Lercier

DGA & University of Rennes — France

email: reynald.lercier(at)m4x.org
www: http://perso.univ-rennes1.fr/reynald.lercier/

Counting Points: Theory, Algorithms and Practice

Montréal, April 2010
Motivation

- In 2000, Satoh and Mestre independently proposed very efficient $p$-adic methods for counting points on elliptic and hyperelliptic curves in $\mathbb{F}_{p^n}$.

- Numerous improvements finally made decrease the complexity in time from $O(n^{3+o(1)})$ to $O(n^{2+o(1)})$.

- We focus on the choice of good basis for $p$-adic unramified extensions, especially we consider $p$-adic analogues of the normal elliptic basis introduced by Couveignes and L. in 2009 for $\mathbb{F}_{p^n}$. 
Outline

1 Point counting over $\mathbb{F}_{p^n}$, $p$ small
   - Elliptic Curve
   - Hyperelliptic Curve

2 Fast Point Counting Algorithms
   - Notations
   - AGM
   - Fast canonical lift
   - Fields with Normal Basis
   - Fields without Normal Basis

3 $p$-adic Elliptic Periods
   - Normal basis
   - Multiplication Tensor
# Outline

1. **Point counting over $\mathbb{F}_{p^n}$, $p$ small**
   - Elliptic Curve
   - Hyperelliptic Curve

2. **Fast Point Counting Algorithms**
   - Notations
   - AGM
   - Fast canonical lift
   - Fields with Normal Basis
   - Fields without Normal Basis

3. **$p$-adic Elliptic Periods**
   - Normal basis
   - Multiplication Tensor
Elliptic curves

\(O(n^3+o(1))\) in time, \(O(n^3)\) in space:


\(O(n^3+o(1))\) in time, \(O(n^2)\) in space:


Elliptic curves

\(O(n^{2.5+\alpha(1)})\) time, \(O(n^2)\) in space:

Elliptic curves

$O(n^{2+o(1)})$ in time, $O(n^2)$ in space:

Hyperelliptic curves of small genus

Genus 2, $O(n^{3+o(1)})$ in time, $O(n^2)$ in space:


$O(n^{2+o(1)})$ in time, $O(n^2)$ in space:

Outline

1. Point counting over $\mathbb{F}_{p^n}$, $p$ small
   - Elliptic Curve
   - Hyperelliptic Curve

2. Fast Point Counting Algorithms
   - Notations
   - AGM
   - Fast canonical lift
   - Fields with Normal Basis
   - Fields without Normal Basis

3. $p$-adic Elliptic Periods
   - Normal basis
   - Multiplication Tensor
**Notations**

1. **$p$-adic numbers**
   - $p$-adic norm $|\cdot|_p$ of $r \in \mathbb{Q}^*$ is $|r|_p = p^{-\rho}$ ($r = p^\rho u/v$, $p \nmid u$, $p \nmid v$).
   
2. **Field of $p$-adic numbers** $\mathbb{Q}_p$ is the completion of $\mathbb{Q}$ w.r.t. $|\cdot|_p$,
   
   $$\sum_{i=\rho}^{\infty} a_i p^i, \quad a_i \in \{0, 1, \ldots, p-1\}, \quad \rho \in \mathbb{Z}.$$

3. **$p$-adic integers** $\mathbb{Z}_p$ is the ring with $|\cdot|_p \leq 1$ or $\rho \geq 0$.

4. $\mathbb{F}_p \cong \mathbb{Z}_p/M$ where $M$ is the unique maximal ideal
   
   $$M = \{x \in \mathbb{Q}_p \mid |x|_p < 1\} = p\mathbb{Z}_p.$$

**Def.** Let $\pi_m$ be the projection from $\mathbb{Z}/p^{m+1}\mathbb{Z}$ onto $\mathbb{Z}/p^m\mathbb{Z}$, then a **$p$-adic integer** is a sequence $x = (x_1, x_2, \ldots, x_m, \ldots)$ with $x_m \in \mathbb{Z}/p^m\mathbb{Z}$ and such that $\pi_m(x_{m+1}) = x_m$. 
$p$-adic field extensions

$K$ extension of $\mathbb{Q}_p$ of degree $n$ with valuation ring $\mathbb{Z}_q$ and maximal ideal $M_{\mathbb{Z}_q} = \{ x \in K \mid |x|_K < 1 \}$.

**Def.** The Teichmuller Lift is the map $\omega : \mathbb{F}_q \rightarrow \mathbb{Z}_q$ defined by $\omega(0) = 0$ and for $x \neq 0$, $\omega(x)$ is the unique $q-1$-th root of one in $\mathbb{Z}_q$ such that $\pi(\omega(x)) = x$ with $\pi$ the canonical projection of $\mathbb{Z}_q$ to $\mathbb{F}_q$.

**Def.** The semi-Witt decomposition of $x \in \mathbb{Z}_q$ is the unique sequence $(x_i)_{i \geq 0}$ of $\mathbb{F}_q$ such that $x = \sum_{i \geq 0} \omega(x_i)p^i$.

The Galois group of (unramified) $K/\mathbb{Q}_p$ is cyclic with generator Frobenius substitution $\sigma$ and $\sigma$ modulo $M_{\mathbb{Z}_q}$ equals to the small Frobenius on $\mathbb{F}_q$.

**Prop.** Let $(x_i)_{i \geq 0}$ be the semi-Witt decomposition of a $p$-adic $x$, then $x^{\sigma} = \sum_{i \geq 0} \omega(x_i)p^i$. 

Bibliography

Notations

**Basis**

**Polynomial Basis.** Let $\mathbb{F}_q \cong \mathbb{F}_p[t]/(\overline{F}(t))$, let $F(t)$ be any lift of $\overline{F}(t)$ to $\mathbb{Z}_p[t]$, then $K$ can be constructed as

$$K \cong \mathbb{Q}_p[t]/(F(t)).$$

Such a choice yields a basis $\{1, t, \ldots, t^{n-1}\}$.

Multiplication, at precision $m$, costs $T_{m,n} = O((nm)^{1+o(1)})$.

**Gaussian Normal Basis (GNB).** For cyclic Galois extension $K/\mathbb{Q}_p$, there exists elements $\alpha$ which yields basis of the form $\{\alpha, \alpha^\sigma, \ldots, \alpha^{\sigma^{n-1}}\}$.

**Def.** For some $r$ such that $\exists$ a primitive $r$-th root of unity $\gamma$ in $\mathbb{Z}/(nr + 1)\mathbb{Z}$ and such that $\alpha = \sum_{i=0}^{r-1} \zeta^i$ (where $\zeta^{nr+1} = 1$) generates a gaussian normal basis over $\mathbb{Q}_p$ of type $r$.

In this case, $T_{m,n} = O((r nm)^{1+o(1)})$. 
**$O(n^{3+o(1)})$ time complexity**

A first algorithm by Satoh, improved by Vercauteran to obtain a $O(n^2)$ in space. Another algorithm by Mestre for $\mathbb{F}_{2^n}$, based on AGM.

---

**Algorithm 1: AGM**

**input**: An (ordinary) elliptic curve $E/\mathbb{F}_{2^n} : y^2 + xy = x^3 + \alpha$

**output**: The trace $c$ of $E$

// Lift phase
1 $a := 1 + 8\alpha \in \mathbb{Z}_q$; $b := 1 \in \mathbb{Z}_q$;
2 for $i := 1$ to $\lceil \frac{n}{2} \rceil + 2$ do
3 \[ a, b := \frac{a+b}{2}, \sqrt{ab} \]

// Norm phase
4 $A := a$; $B := b$;
5 for $i := 1$ to $n$ do
6 \[ a, b := \frac{a+b}{2}, \sqrt{ab} \]
7 return $\frac{A}{a} \mod 2^n$ as a signed integer in $[-2\sqrt{2^n}, 2\sqrt{2^n}]$. 


AGM iterations

- An AGM step is an isogeny of degree 2 between elliptic curves.

- Repeatedly, we get the following sequence

\[ J_{K_q}^1 \xrightarrow{\sigma^1} \cdots \xrightarrow{\sigma^{m-1}} J_{K_q}^m \xrightarrow{\sigma^m} \cdots \xrightarrow{\sigma^{m+n-1}} J_{K_q}^{m+n} \].

- Then, \((J_{K_q}^{m+i})_i\) converges to \(J_{\text{can}}^m\), the canonical lift of \(J_0^m\).
Fast canonical lift

\[ O(n^{2+o(1)}) \] time complexity

**Lift phase.** First,

\[
\begin{align*}
    a_{i+1} &= \frac{a_i + b_i}{2}, \\
    b_{i+1} &= \sqrt{a_i b_i},
\end{align*}
\]

can be replaced via \( c_i = \frac{a_i}{b_i} \) by \( c_{i+1} = \frac{2 + c_i}{2 \sqrt{c_i}} \).

Second,

\[ c_{i+1} = c_i^\sigma. \]

Consequently, one must solve at precision \( n/2 + O(1) \),

\[ 4x(x^\sigma)^2 = (1 + x)^2. \]

This equation is an equation of the form \( \phi(x, x^\sigma) \) where \( \phi(x, y) \) is a polynomial.

**Norm phase.** We simply have,

\[ c = N_{\mathbb{Z}_{2^n}/\mathbb{Z}_2} \left( \frac{2c_{\lceil n/2 \rceil} + 3}{1 + c_{\lceil n/2 \rceil} + 3} \right). \]
Fast “lift” and “norm” algorithms

\[ \mathbb{Z}_q : \left[ t^{n-1} + \ldots + t^0 \right] \quad \text{Lift} \]

\[ \mathbb{F}_q : \left[ \bigcirc t^{n-1} + \ldots + \bigcirc t^0 \right] \]

\[ \text{Norm} \quad \mathbb{Z}_p : \left[ \right] \]
Newton iteration

To compute the root of a polynomial $f(x)$ from

$$f(x + p^w \delta) = f(x) + p^w \delta \frac{\partial f}{\partial x}(x) + O(p^{2w}).$$

Algorithm 2: Newton

```
input : $x_0$ s.t. $f(x_0) \equiv 0 \mod p^{2k+1}$ where $k = \nu(\partial f/\partial x(x_0))$ and $m \in \mathbb{N}.$
output: $x$ a solution of $f(x) \mod p^m.$

1 if $m \leq 2k + 1$ then
2     return $x_0$
3 $x := \text{Newton}(x_0, \lceil \frac{m}{2} \rceil + k);$
4 $V := f(x) \mod p^m;$ $\Delta_x := \partial f/\partial x(x) \mod p^{w-k};$
5 return $x - V/\Delta_x$
```

Remark. Very fast in practice. For polynomials with $O(1)$ terms of degree $O(1)$, time complexity is $O(T_{m,n})$. 
Generalized Newton iterations

One generalizes Newton alg. to eq. of the form $\phi(x, x^\sigma) = 0$. Based on

$$\phi(x + p^w \delta, (x + p^w \delta)^\sigma) = \phi(x, x^\sigma) + p^w \delta \frac{\partial \phi}{\partial x} (x, x^\sigma) + p^w \delta^\sigma \frac{\partial \phi}{\partial y} (x, x^\sigma) + O(p^{2w}).$$

**Algorithm 3: NewtonLift**

input : $x_0$ s.t. $\phi(x_0, x_0^\sigma) \equiv 0 \mod p^{2k+1}$ where $k = v(\partial \phi/\partial y(x_0))$ and $m \in \mathbb{N}$.

output: $x$ a solution of $\phi(x, x^\sigma) \mod p^m$.

1. if $m \leq 2k + 1$ then
2. \hspace{1em} return $x_0$
3. \hspace{1em} $w := \lceil m/2 \rceil + k$; $x := \text{NewtonLift}(x_0, w)$;
4. \hspace{1em} Lift $x$ to $\mathbb{Z}_q/p^m\mathbb{Z}_q$; $y := x^\sigma \mod p^m$;
5. \hspace{1em} $\Delta_x := \partial_x \phi(x, y) \mod p^{w-k}$; $\Delta_y := \partial_y \phi(x, y) \mod p^{w-k}$;
6. \hspace{1em} $V := \phi(x, y) \mod p^m$;
7. \hspace{1em} $a, b := \text{ArtinSchreierRoot}(-V/(p^{w-k} \Delta_y), -\Delta_x/\Delta_y, w - k, n)$;
8. \hspace{1em} return $x + p^{w-k} (1 - a)^{-1} b$

**Remark.** $\text{ArtinSchreierRoot}$ is a “black box” which solves equations of the form $x^\sigma = ax + b$, $a$ and $b$ in $\mathbb{Z}_q$. 
Artin-Schreier equations with Normal Basis

- For all \( k \in \mathbb{N} \), \( x^{\sigma^k} \equiv a_k x + b_k \mod p^w \).
- \( x^{\sigma^n} = x \), which means that \((1 - a_n)x = b_n\).
- A classical “square and multiply” composition formula, \( \forall k, k' \in \mathbb{Z}^2 \),
  \[ x^{\sigma^{k+k'}} = a_k^{\sigma^{k'}} a_{k'} x + a_k^{\sigma^k} b_{k'} + b_k^{\sigma^{k'}}. \]

Algorithm 4: ArtinSchreierRoot

<table>
<thead>
<tr>
<th>line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>if ( \nu = 1 ) then [ return ( a, b \mod p^m ) ]</td>
</tr>
<tr>
<td>2</td>
<td>[ return ( a, b \mod p^m ) ]</td>
</tr>
<tr>
<td>3</td>
<td>( w := \lfloor \nu/2 \rfloor ); ( A, B := ArtinSchreierRoot(a, b, w) );</td>
</tr>
<tr>
<td>4</td>
<td>( A, B := AA^{\sigma^w}, BA^{\sigma^w} + B^{\sigma^w} \mod p^m );</td>
</tr>
<tr>
<td>5</td>
<td>if ( \nu \equiv 1 \mod 2 ) then [ A, B := Aa^\sigma, bA^\sigma + B^\sigma \mod p^m ]</td>
</tr>
<tr>
<td>6</td>
<td>return ( A, B );</td>
</tr>
<tr>
<td>7</td>
<td>[ return ( A, B ); ]</td>
</tr>
</tbody>
</table>

Complexity is \( O(T_{m,n} \log n) \).
Norm computation with Normal Basis

A square and multiply approach suggested by Kedlaya.

Combine, from $a_0 = a$, quantities of the form

$$a_{i+1} := a_i^{\sigma^2} a_i \text{ for } i = 0, \ldots, \lfloor \log_2 n \rfloor.$$ 

---

**Algorithm 5: Norm**

**input**: $a$ in $\mathbb{Z}_q$ and a precision $m$ in $\mathbb{N}$.

**output**: $N_{K/\mathbb{Q}_p}(a) \mod p^m$.

1. $i := n$; $j := 0$, $r := 1$, $s := a$;
2. while $i > 0$ do
   3. if $i \equiv 1 \bmod 2$ then $r := s r^{\sigma^2 j}$;
   4. if $i > 1$ then $s := s s^{\sigma^2 j}$;
   5. $j := j + 1$; $i := \lfloor i/2 \rfloor$;
3. return $r$;

Complexity is $O(T_{m,n} \log n)$. 
Timings for counting points on elliptic curves defined over $\mathbb{F}_{2^n}$ (GNB)

On a 731 MHz Alpha EV6 CPU (2002 timings).

<table>
<thead>
<tr>
<th>$n$</th>
<th>GNB type 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Lift</td>
</tr>
<tr>
<td>1018</td>
<td>2.5s</td>
</tr>
<tr>
<td>2052</td>
<td>10s</td>
</tr>
<tr>
<td>4098</td>
<td>1mn</td>
</tr>
<tr>
<td>8218</td>
<td>6mn 30</td>
</tr>
<tr>
<td>16420</td>
<td>34mn</td>
</tr>
<tr>
<td>32770</td>
<td>3h 17</td>
</tr>
<tr>
<td>65538</td>
<td>15h 45</td>
</tr>
<tr>
<td>100002</td>
<td>1d 18</td>
</tr>
</tbody>
</table>
Lifting the Frobenius at precision $m$ [Satoh-Harley]

Computing $x^\sigma$ in a polynomial basis is a costly task.

One lifts $\overline{F}(t)$ at precision $m$ to the minimal polynomial $F$ of $\omega(t)$ with

$$F(t^p) = \prod_{i=0}^{p-1} F(t\zeta^i) \text{ with } \zeta^p = 1.$$ 

This can be done by Newton iterations in $O(pT_{m,n} \log n)$.

It follows that $t^\sigma = t^p$ and

$$x^\sigma = \sum_{i=0}^{n-1} x_i t^{ip} = \sum_{j=0}^{p-1} \left( \sum_{0 \leq pk+j < n} x_{pk+j} t^k \right) C_j(t) \mod F(t).$$

With $C_j(t) = t^{jp} \mod F(t)$ precomputed, a $O(p T_{m,n})$ complexity.
A two-fold recursive algorithms to doubling the precision.

**Algorithm 6: ArtinSchreierRoot**

input : Eq. $x^\sigma = ax + b$ in $\mathbb{Z}_q/p^m\mathbb{Z}_q$ with $|b|_K < 1$, $m$ in $\mathbb{N}$.
output: A $x \in \mathbb{Z}_q$ s.t. $x^\sigma = ax + b \mod p^m$.

1. if $m = 1$ then
2. \hspace{1em} return $b^{\bar{\sigma}}$
3. $N := \lfloor m/2 \rfloor$; $M := m - N$;
4. $x_0 := \text{ArtinSchreierRoot}(a, b, N)$;
5. \hspace{1em} $\beta := (x_0^\sigma - ax_0 - b)/p^N \mod p^M$;
6. $x_1 := \text{ArtinSchreierRoot}(a, \beta, M)$;
7. return $x_0 + p^N x_1 \mod p^m$

Let $T(n)$ be the running time for precision $m$, then

$$T(m) \leq 2T(m/2) + (pnm)^{1+o(1)} \Rightarrow T(m) = O(pT_{m,n} \log m).$$
Norm computation without Normal Basis

For $\alpha \in \mathbb{Q}_p$,

$$N_{K/\mathbb{Q}_p}(\alpha) = p^{n \text{ord}_p(\alpha)} N_{K/\mathbb{Q}_p}(\alpha/p^{\text{ord}_p(\alpha)}) .$$

For $\alpha$ a unit, let $\alpha = \sum_{i=0}^{n-1} a_i t^i$, then

$$N_{K/\mathbb{Q}_p}(\alpha) = \text{Res}(F(t), \sum_{i=0}^{n-1} a_i t^i) .$$

The resultant $\text{Res}(F(t), \sum_{i=0}^{n-1} a_i t^i)$ can be computed in softly linear time using a variant of Moenck’s fast extended GCD algorithm.

Complexity is $O(T_{m,n} \log n)$, mostly due to multiplications of $2 \times 2$ matrices with (polynomial) coefficients in $\mathbb{Z}_p[t]$, at precision $m$. 
Harley’s timings

Measured on a 750 MHz Alpha EV6 (Nov. 2002, NMBRTHRY mailing list).

<table>
<thead>
<tr>
<th>Bits</th>
<th>Point counting</th>
<th>Precomputation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Lift</td>
<td>Norm</td>
</tr>
<tr>
<td>197</td>
<td>0.04</td>
<td>0.04</td>
</tr>
<tr>
<td>409</td>
<td>0.26</td>
<td>0.25</td>
</tr>
<tr>
<td>571</td>
<td>0.76</td>
<td>0.61</td>
</tr>
<tr>
<td>1000</td>
<td>2.46</td>
<td>1.43</td>
</tr>
<tr>
<td>2003</td>
<td>15.2</td>
<td>7.71</td>
</tr>
<tr>
<td>4001</td>
<td>1m 33</td>
<td>52</td>
</tr>
<tr>
<td>8009</td>
<td>9m 30</td>
<td>6m 20</td>
</tr>
<tr>
<td>16001</td>
<td>59m</td>
<td>48m 56</td>
</tr>
<tr>
<td>32003</td>
<td>6h 9m</td>
<td>6h 41m</td>
</tr>
<tr>
<td>130020</td>
<td>?</td>
<td>67h 17m</td>
</tr>
</tbody>
</table>

**Remark.** Asymptotically fast lifts, but still a $O(n^{2+1/3} \log n \log \log n)$ norm computation (after Satoh).
Outline

1. Point counting over $\mathbb{F}_{p^n}$, $p$ small
   - Elliptic Curve
   - Hyperelliptic Curve

2. Fast Point Counting Algorithms
   - Notations
   - AGM
   - Fast canonical lift
   - Fields with Normal Basis
   - Fields without Normal Basis

3. $p$-adic Elliptic Periods
   - Normal basis
   - Multiplication Tensor
Some remarks

It is expected that normal basis (with fast multiplication tensors), even if it does not change the asymptotic complexity, yield faster point counting algorithms:

- it suppresses the computation of the lift $F$ in $\mathbb{Q}_p[t]$ of the definition polynomial $\overline{F}(t)$ for $\mathbb{F}_q$,
- it suppresses the $p$ factor in the complexity of some parts of the algorithm, especially the ArtinSchreierRoot routine,
- it is expected that $\mathbb{Z}_q/\mathbb{Z}_p$ norms can be computed faster.

Maybe more important, we may hope that memory requirements are slightly lowered too.

But, it is hopeless to expect that a Gaussian normal basis of small type $r$ exists for many degree $n$ : in general $r \approx n^3 \log^2(np)$ [Adleman-Lenstra 1986].
Elliptic Normal Basis (Finite Fields)

For $F_q$, we made use of torsion points on elliptic curves instead of roots of unity to obtain analogues of Gaussian normal basis.

**Theorem (Couveignes-L.)**

To every couple $(q, n)$ with $q$ a prime power and $n \geq 2$ an integer s.t. $n_q \leq \sqrt{q}$, one can associate a normal basis $\Theta(q, n)$ of the degree $n$ extension of $F_q$ such that the following holds:

- There exists an algorithm that multiplies two elements given in $\Theta(q, n)$ at the expense of $\tilde{O}(n \log q)$ elementary operations.

This can be easily extend to a result without any restriction on $q$ and $n$.

**Remark:** Here $n_q$ is such that

- $v_\ell(n_q) = v_\ell(n)$ if $\ell$ is prime to $q - 1$, $v_\ell(n_q) = 0$ if $v_\ell(n) = 0$,
- $v_\ell(n_q) = \max(2v_\ell(q - 1) + 1, 2v_\ell(n))$ if $\ell$ divides both $q - 1$ and $n$. 
A \textit{p}-adic generalisation

- Let $E/\mathbb{Q}_p$ be an elliptic curve given by
  \[ Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3. \]

- If $A$, $B$ and $C$ are three pairwise distinct points in $E(\mathbb{Q}_p)$, we define
  \[ \Gamma(A, B, C) = \frac{y(C - A) - y(A - B)}{x(C - A) - x(A - B)}. \]

- We define a function $u_{A, B} \in \mathbb{Q}_p(E)$ by $u_{A, B}(C) = \Gamma(A, B, C)$.
  It has degree two with two simple poles, at $A$ and $B$. 
Ingredient 1: Residue fields of divisors on elliptic curves

Let $E$ be an elliptic curve defined over $\mathbb{Q}_p$.

- Assume $E(\mathbb{Q}_p)$ contains a cyclic subgroup $T$ of order $n$
  (find such a curve mod $p$ and lift it, with $T$, to $\mathbb{Q}_p$).
- Let $I: E \to E'$ be the degree $n$ cyclic isogeny with kernel $T$.
- Take $a$ in $E'(\mathbb{Q}_p)$ s.t. $\hat{I}(a) \neq O_E$.
- Let $\mathcal{P}$ be the fibre $I^{-1}(a) = \sum_{t \in T} [b + t]$, a simple divisor over $\mathbb{Q}_p$.
- Then, $\phi(b) - b \in T$ (where $\phi$ is the Frobenius map).

Under some mild condition, $\phi(b) - b$ is a generator of $T$ and the $n$ geometric points above $a$ are defined on a degree $n$ extension $K$ of $\mathbb{Q}_p$ (and permuted by Galois action).

$K$ is the residue extension of $\mathbb{Q}_p(E)$ at $\mathcal{P}$. 
Coming back to the functions $u_{AB}$, we choose for $A$ and $B$ consecutive points in $\mathcal{T}$.

For $k \in \mathbb{Z}/d\mathbb{Z}$, we more precisely set

$$u_k = a u_{kt, (k+1)t} + b$$

($a$ and $b$, constants chosen such that $\sum u_k = 1$), and we evaluate the $u_k$'s at $b$.

Lemma (A normal basis)

The system $\Theta = (u_k(b))_{k \in \mathbb{Z}/d\mathbb{Z}}$ is a $\mathbb{Q}_p$ normal basis of $K$. 
Ingredient 2: Relations among elliptic functions

We can prove the following identities (with Taylor expansions at poles)

\[ \Gamma(A, B, C) = \Gamma(B, C, A) = -\Gamma(B, A, C) - a_1 = -\Gamma(-A, -B, -C) - a_1, \]

\[ u_{A,B} + u_{B,C} + u_{C,A} = \Gamma(A, B, C) - a_1, \]

and

\[ u_{A,B}u_{A,C} = x_A + \Gamma(A, B, C)u_{A,C} + \Gamma(A, C, B)u_{A,B} + a_2 + x_A(B) + x_A(C), \]

\[ u^2_{A,B} = x_A + x_B - a_1 u_{A,B} + x_A(B) + a_2, \]

where

- \( \tau_A : E \rightarrow E \) denotes the translation by \( A \),
- and in \( \mathbb{Q}_p(E) \), \( x_A = x \circ \tau_{-A} \) and \( y_A = y \circ \tau_{-A} \).
A fast multiplication algorithm

\[ u_{A,B} u_{A,C} = x_A + \Gamma(A, B, C) u_{A,C} + \Gamma(A, C, B) u_{A,B} + a_2 + x_A(B) + x_A(C), \]
\[ u_{A,B}^2 = x_A + x_B - a_1 u_{A,B} + x_A(B) + a_2. \]

This yields a multiplication tensor for \( \Theta \) with quasi-linear complexity,

\[ \bar{\alpha} \times \bar{\beta} = (a^2 \vec{i}) \star \left( (\bar{\alpha} - \sigma(\bar{\alpha})) \circ (\bar{\beta} - \sigma(\bar{\beta})) \right) + \]
\[ \overrightarrow{u_{R}}(-1) \star\left( (\overrightarrow{u_{R}} \star \bar{\alpha}) \circ (\overrightarrow{u_{R}} \star \bar{\beta}) - (a^2 \vec{x}_{R}) \star \left( (\bar{\alpha} - \sigma(\bar{\alpha})) \circ (\bar{\beta} - \sigma(\bar{\beta})) \right) \right). \]

Notations:

- \( \bar{\alpha} \star \bar{\beta} \), the convolution product \( (\bar{\alpha} \star_j \bar{\beta})_j \), with \( \bar{\alpha} \star_j \bar{\beta} = \sum_i \alpha_i \beta_{j-i} \).
- \( \sigma(\bar{\alpha}) = (\alpha_{i-1})_i \), the cyclic shift of \( \bar{\alpha} \).
- \( \bar{\alpha} \circ \bar{\beta} = (\alpha_i \beta_i)_i \), the component-wise product.
Evaluations/interpolations

It consists in evaluations and interpolations at $n$ points $r + kt$, where

$$r \in E(\mathbb{Q}_p) - E[d].$$

Constants are

$$\vec{l} = (\nu_i)_{0 \leq i \leq d - 1} \text{ s.t. } x(b) = \sum_{0 \leq k \leq d - 1} \nu_k \theta_k,$$

$$\vec{x}_R = (x(r + kt))_{0 \leq k \leq d - 1},$$

$$\vec{u}_R = (u_0(r + kt))_{0 \leq k \leq d - 1}.$$
Fast convolutions

Convolution and polynomial multiplication:

\[ F(X) = \sum_{i=0}^{n-1} f_i X^i, \quad G(X) = \sum_{i=0}^{n-1} g_i X^i \]

Then:

\[ \vec{h} = \vec{f} \ast \vec{g} \iff H(X) \equiv F(X)G(X) \mod (X^n - 1) \]

FFT’s speedup:

\[ \vec{f} \ast \vec{g} = \hat{f} \odot \hat{g}^{(-1)} \]
Application to normal elliptic basis

\[(a^2 \iota) \ast \left( (\tilde{\alpha} - \sigma(\tilde{\alpha})) \diamond (\tilde{\beta} - \sigma(\tilde{\beta})) \right) + \]
\[\overrightarrow{u}_R^{(-1)} \ast \left( (\tilde{u}_R \ast \tilde{\alpha}) \diamond (\tilde{u}_R \ast \tilde{\beta}) - (a^2 \bar{x}_R) \ast \left( (\tilde{\alpha} - \sigma(\tilde{\alpha})) \diamond (\tilde{\beta} - \sigma(\tilde{\beta})) \right) \right)\]

<table>
<thead>
<tr>
<th>Product</th>
<th>“Dense” Polynomial Basis</th>
<th>Normal Elliptic Basis</th>
<th>“Sparse” Polynomial Basis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4+3= 7 FFTs of lg. 2n</td>
<td>3+5= 8 FFTs of lg. n</td>
<td>2+1= 3 FFTs of lg. 2n</td>
</tr>
<tr>
<td></td>
<td>(\simeq 14) FFTs of lg. n</td>
<td></td>
<td>(\simeq 6) FFTs of lg. n</td>
</tr>
<tr>
<td>Squaring</td>
<td>3+3= 6 FFTs of lg. 2n</td>
<td>2+4= 6 FFTs of lg. n</td>
<td>1+1= 2 FFTs of lg. 2n</td>
</tr>
<tr>
<td></td>
<td>(\simeq 12) FFTs of lg. n</td>
<td></td>
<td>(\simeq 4) FFTs of lg. n</td>
</tr>
</tbody>
</table>

- Precompute FFTs for \(\iota, \overrightarrow{u}_R^{(-1)}, \tilde{u}_R\) \& \(\bar{x}_R\),
- 3 direct FFTs, for \(\tilde{\alpha}, \tilde{\beta}\) \& \((\tilde{\alpha} - \sigma(\tilde{\alpha})) \diamond (\tilde{\beta} - \sigma(\tilde{\beta}))\),
- 5 inverse FFTs.
To conclude

It is expected that elliptic normal basis yields faster practical implementations of Satoh/Mestre’s algorithms.

Especially, for $p$ large enough such that the Hasse’s bound $n \leq p + 1 + 2\sqrt{p}$ is satisfied.

For $p$ very small, typ. $p = 2$, it is not clear that the extra $\log n$ penalty to pay for the existence of an elliptic normal basis will be too large.