On some p-adic differential equations

(with separation of variables)

Pierre Lairez, **Tristan Vaccon**

TU Berlin, 立教大学 **(Rikkyo University)–JSPS Post-Doctoral Fellow**

March, 29th 2016

RIKKYO UNIVERSITY

Why should one work with p-adic numbers ?

p-adic methods

- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;

Why should one work with p-adic numbers ?

p-adic methods

- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- *e.g.* Dixon's method (used in F4), Polynomial factorization via Hensel's lemma.

Why should one work with p-adic numbers ?

p-adic methods

- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- *e.g.* Dixon's method (used in F4), Polynomial factorization via Hensel's lemma.

p-adic algorithms

- Going from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$ and then back to $\mathbb{Z}/p\mathbb{Z}$ enables more computation,

Why should one work with p-adic numbers ?

p-adic methods

- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- *e.g.* Dixon's method (used in F4), Polynomial factorization via Hensel's lemma.

p-adic algorithms

- Going from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$ and then back to $\mathbb{Z}/p\mathbb{Z}$ enables more computation, *e.g.* the algorithms of Bostan et al. and Lercier et al. using p-adic differential equations ;

Why should one work with p-adic numbers ?

p-adic methods

- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- *e.g.* Dixon's method (used in F4), Polynomial factorization via Hensel's lemma.

p-adic algorithms

- Going from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$ and then back to $\mathbb{Z}/p\mathbb{Z}$ enables more computation, *e.g.* the algorithms of Bostan et al. and Lercier et al. using p-adic differential equations ;
- Kedlaya's and Lauder's counting-point algorithms via p-adic cohomology ;

Why should one work with p-adic numbers ?

p-adic methods

- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- *e.g.* Dixon's method (used in F4), Polynomial factorization via Hensel's lemma.

p-adic algorithms

- Going from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$ and then back to $\mathbb{Z}/p\mathbb{Z}$ enables more computation, *e.g.* the algorithms of Bostan et al. and Lercier et al. using p-adic differential equations ;
- Kedlaya's and Lauder's counting-point algorithms via p-adic cohomology ;

My personal (long-term) motivation

Computing (some) moduli spaces of p-adic Galois representations.

Table of contents

Definition of the precision

Finite-precision p-adics

Elements of $\mathbb{Q}_p$ can be written $\sum_{i=k}^{+\infty} a_i p^i$, with $a_i \in [\![0, p-1]\!]$, $k \in \mathbb{Z}$ and p a prime number.
While working with a computer, we usually only can consider the beginning of this power serie expansion: we only consider elements of the following form $\boxed{\sum_{i=l}^{d-1} a_i p^i + O(p^d)}$, with $l \in \mathbb{Z}$.

Definition of the precision

Finite-precision p-adics

Elements of $\mathbb{Q}_p$ can be written $\sum_{i=k}^{+\infty} a_i p^i$, with $a_i \in [\![0, p-1]\!]$, $k \in \mathbb{Z}$ and p a prime number.
While working with a computer, we usually only can consider the beginning of this power serie expansion: we only consider elements of the following form $\boxed{\sum_{i=l}^{d-1} a_i p^i + O(p^d)}$, with $l \in \mathbb{Z}$.

Definition of the precision

Finite-precision p-adics

Elements of $\mathbb{Q}_p$ can be written $\sum_{i=k}^{+\infty} a_i p^i$, with $a_i \in [\![0, p-1]\!]$, $k \in \mathbb{Z}$ and p a prime number.

While working with a computer, we usually only can consider the beginning of this power serie expansion: we only consider elements of the following form $\boxed{\sum_{i=l}^{d-1} a_i p^i + O(p^d)}$, with $l \in \mathbb{Z}$.

Definition

The **order**, or the **absolute precision** of $\sum_{i=k}^{d-1} a_i p^i + O(p^d)$ is d.

Definition of the precision

Finite-precision p-adics

Elements of $\mathbb{Q}_p$ can be written $\sum_{i=k}^{+\infty} a_i p^i$, with $a_i \in [\![0, p-1]\!]$, $k \in \mathbb{Z}$ and p a prime number.
While working with a computer, we usually only can consider the beginning of this power serie expansion: we only consider elements of the following form $\boxed{\sum_{i=l}^{d-1} a_i p^i + O(p^d)}$, with $l \in \mathbb{Z}$.

Definition

The **order**, or the **absolute precision** of $\sum_{i=k}^{d-1} a_i p^i + O(p^d)$ is d.

Example

The order of $3 * 7^{-1} + 4 * 7^0 + 5 * 7^1 + 6 * 7^2 + O(7^3)$ is 3.

p-adic precion *vs* real precision

The quintessential idea of the step-by-step analysis is the following :

Proposition (p-adic errors don't add)

Indeed,

$$(a + O(p^k)) + (b + O(p^k)) = a + b + O(p^k).$$

That is to say, if a and b are known up to precision $O(p^k)$, then so is $a + b$.

p-adic precion *vs* real precision

The quintessential idea of the step-by-step analysis is the following :

Proposition (p-adic errors don't add)

Indeed,

$$(a + O(p^{k})) + (b + O(p^{k})) = a + b + O(p^{k}).$$

That is to say, if a and b are known up to precision $O(p^k)$, then so is $a + b$.

p-adic precion *vs* real precision

The quintessential idea of the step-by-step analysis is the following :

Proposition (p-adic errors don't add)

Indeed,

$$(a + O(p^{k})) + (b + O(p^{k})) = a + b + O(p^{k}).$$

That is to say, if a and b are known up to precision $O(p^k)$, then so is $a + b$.

Remark

It is quite the opposite to when dealing with real numbers, because of **Round-off error** :

$$(1 + 5 * 10^{-2}) + (2 + 6 * 10^{-2}) = 3 + 1 * 10^{-1} + 1 * 10^{-2}.$$

That is to say, if a and b are known up to precision 10^{-n}, then $a + b$ is known up to $10^{(-n+1)}$.

p-adic precion *vs* real precision

The quintessential idea of the step-by-step analysis is the following :

Proposition (p-adic errors don't add)

Indeed,

$$(a + O(p^k)) + (b + O(p^k)) = a + b + O(p^k).$$

That is to say, if a and b are known up to precision $O(p^k)$, then so is $a + b$.

Remark

It is quite the opposite to when dealing with real numbers, because of **Round-off error** :

$$(1 + 5 * 10^{-2}) + (2 + 6 * 10^{-2}) = 3 + 1 * 10^{-1} + 1 * 10^{-2}.$$

That is to say, if a and b are known up to precision 10^{-n}, then $a + b$ is known up to $10^{(-n+1)}$.

Precision formulae

Proposition (addition)

$$(x_0 + O(p^{k_0})) + (x_1 + O(p^{k_1})) = x_0 + x_1 + O(p^{min(k_0,k_1)})$$

Precision formulae

Proposition (addition)

$$(x_0 + O(p^{k_0})) + (x_1 + O(p^{k_1})) = x_0 + x_1 + O(p^{min(k_0, k_1)})$$

Proposition (multiplication)

$$(x_0 + O(p^{k_0})) * (x_1 + O(p^{k_1})) = x_0 * x_1 + O(p^{min(k_0 + v_p(x_1), k_1 + v_p(x_0))})$$

Precision formulae

Proposition (addition)

$$(x_0 + O(p^{k_0})) + (x_1 + O(p^{k_1})) = x_0 + x_1 + O(p^{min(k_0,k_1)})$$

Proposition (multiplication)

$$(x_0 + O(p^{k_0})) * (x_1 + O(p^{k_1})) = x_0 * x_1 + O(p^{min(k_0+v_p(x_1),k_1+v_p(x_0))})$$

Proposition (division)

$$\frac{xp^a + O(p^b)}{yp^c + O(p^d)} = x * y^{-1}p^{a-c} + O(p^{min(d+a-2c,b-c)})$$

In particular,

$$\frac{1}{p^c y + O(p^d)} = y^{-1}p^{-c} + O(p^{d-2c})$$

Table of contents

A little warm-up on computing determinants : expansion

An example of determinant computation

$$\begin{bmatrix} p^5 + O(p^{10}) & 1 + O(p^{10}) & 1 + p^3 + O(p^{10}) \\ O(p^{10}) & 1 + O(p^{10}) & 1 + O(p^{10}) \\ 2p^6 + O(p^{10}) & 2p + O(p^{10}) & 2p + p^5 + O(p^{10}) \end{bmatrix}$$

A little warm-up on computing determinants : expansion

An example of determinant computation

$$\begin{bmatrix} p^5 + O(p^{10}) & 1 + O(p^{10}) & 1 + p^3 + O(p^{10}) \\ O(p^{10}) & 1 + O(p^{10}) & 1 + O(p^{10}) \\ 2p^6 + O(p^{10}) & 2p + O(p^{10}) & 2p + p^5 + O(p^{10}) \end{bmatrix}$$

Direct expansion

If we expand directly using the expression of the determinant in terms of the coefficients, we get:

A little warm-up on computing determinants : expansion

An example of determinant computation

$$\begin{bmatrix} p^5 + O(p^{10}) & \boxed{1 + O(p^{10})} & 1 + p^3 + O(p^{10}) \\ O(p^{10}) & 1 + O(p^{10}) & \boxed{1 + O(p^{10})} \\ \boxed{2p^6 + O(p^{10})} & 2p + O(p^{10}) & 2p + p^5 + O(p^{10}) \end{bmatrix}$$

Direct expansion

If we expand directly using the expression of the determinant in terms of the coefficients, we get:

$$-2p^9 + O(p^{10}),$$

because of $1 \times 1 \times O(p^{10})$.

A little warm-up on computing determinants : row-echelon form computation

An example of determinant computation

$$\begin{bmatrix} p^5 + O(p^{10}) & 1 + O(p^{10}) & 1 + p^3 + O(p^{10}) \\ O(p^{10}) & 1 + O(p^{10}) & 1 + O(p^{10}) \\ O(p^{10}) & O(p^{10}) & -2p^4 + p^5 + O(p^{10}) \end{bmatrix}$$

A little warm-up on computing determinants : row-echelon form computation

An example of determinant computation

$$\begin{bmatrix} p^5 + O(p^{10}) & 1 + O(p^{10}) & 1 + p^3 + O(p^{10}) \\ O(p^{10}) & 1 + O(p^{10}) & 1 + O(p^{10}) \\ O(p^{10}) & O(p^{10}) & -2p^4 + p^5 + O(p^{10}) \end{bmatrix}$$

Row-echelon form computation

If we compute **approximate** row-echelon form, we still get:

A little warm-up on computing determinants : row-echelon form computation

An example of determinant computation

$$\begin{bmatrix} p^5 + O(p^{10}) & \boxed{1 + O(p^{10})} & 1 + p^3 + O(p^{10}) \\ O(p^{10}) & 1 + O(p^{10}) & \boxed{1 + O(p^{10})} \\ \boxed{O(p^{10})} & O(p^{10}) & -2p^4 + p^5 + O(p^{10}) \end{bmatrix}$$

Row-echelon form computation

If we compute **approximate** row-echelon form, we still get:

$$-2p^9 + O(p^{10}),$$

because of $1 \times 1 \times O(p^{10})$.

A little warm-up on computing determinants : SNF

An example of determinant computation

$$\begin{bmatrix} 1+O(p^{10}) & O(p^{10}) & O(p^{10}) \\ O(p^{10}) & p^3+O(p^{10}) & O(p^{10}) \\ O(p^{10}) & O(p^{10}) & -2p^6+O(p^{10}) \end{bmatrix}$$

A little warm-up on computing determinants : SNF

An example of determinant computation

$$\begin{bmatrix} 1 + O(p^{10}) & O(p^{10}) & O(p^{10}) \\ O(p^{10}) & p^3 + O(p^{10}) & O(p^{10}) \\ O(p^{10}) & O(p^{10}) & -2p^6 + O(p^{10}) \end{bmatrix}$$

Smith Normal Form (SNF) computation

If we compute **approximate** SNF, we now get:

A little warm-up on computing determinants : SNF

An example of determinant computation

$$\begin{bmatrix} \boxed{1+O(p^{10})} & O(p^{10}) & O(p^{10}) \\ O(p^{10}) & \boxed{p^3+O(p^{10})} & O(p^{10}) \\ O(p^{10}) & O(p^{10}) & \boxed{-2p^6+O(p^{10})} \end{bmatrix}$$

Smith Normal Form (SNF) computation

If we compute **approximate** SNF, we now get:

$$-2p^9 + O(p^{13}),$$

because of $1 \times p^3 \times O(p^{10}) = O(p^{13})$.

Summary: precision and p-adic computations

Direct method for precision

Summary: precision and p-adic computations

Direct method for precision

- Has often been enough to get a first view of the problem.

Summary: precision and p-adic computations

Direct method for precision

- Has often been enough to get a first view of the problem.
- Depends heavily on the algorithm chosen for the computation

Summary: precision and p-adic computations

Direct method for precision

- Has often been enough to get a first view of the problem.
- Depends heavily on the algorithm chosen for the computation
- No idea on what is **optimal**.

Table of contents

The Main lemma of p-adic differential precision

Lemma (CRV14)

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*

The Main lemma of p-adic differential precision

Lemma (CRV14)

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*

The Main lemma of p-adic differential precision

Lemma (CRV14)

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*
Then for any ball $B = B(0, r)$ ***small enough****,*

The Main lemma of p-adic differential precision

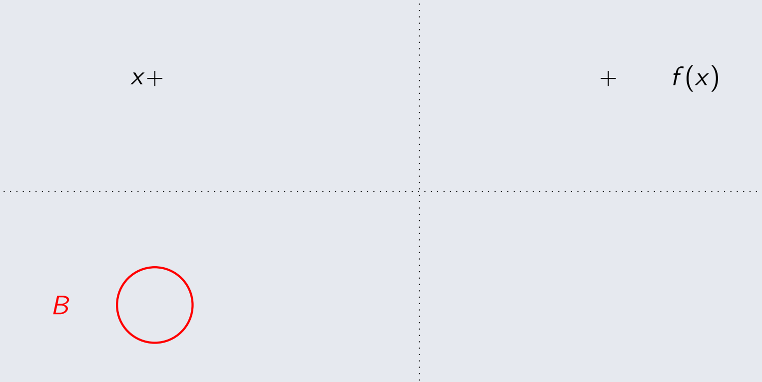
Lemma (CRV14)

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*
Then for any ball $B = B(0, r)$ ***small enough****,*

$$f(x + B) = f(x) + f'(x) \cdot B.$$

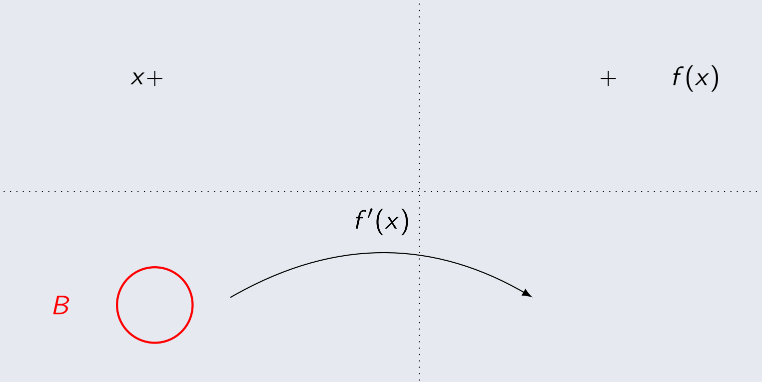
Geometrical meaning

Interpretation



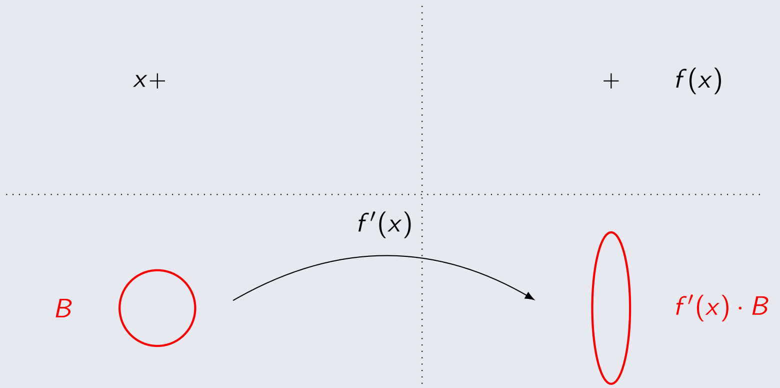
Geometrical meaning

Interpretation



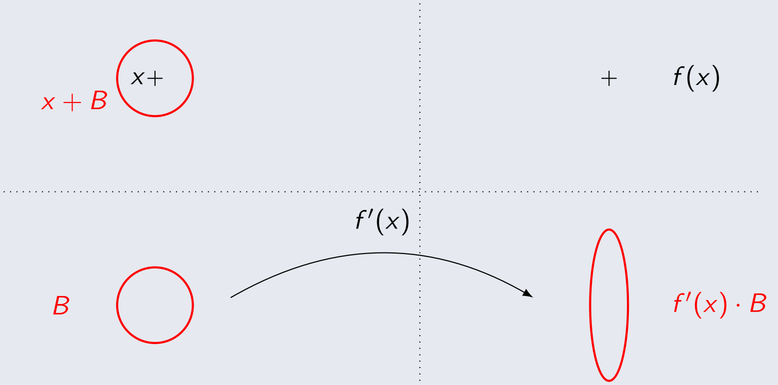
Geometrical meaning

Interpretation



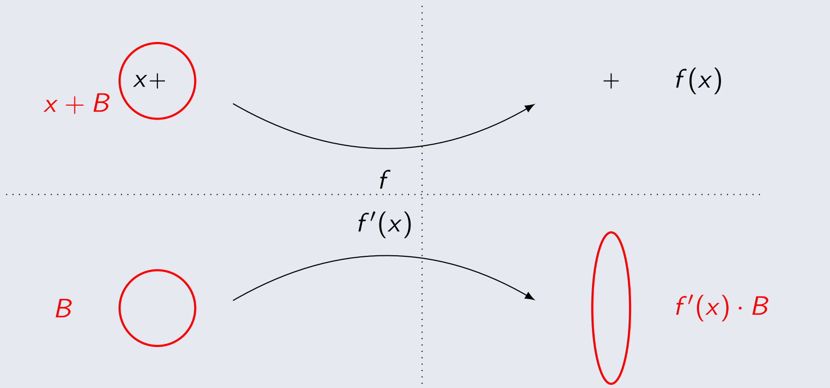
Geometrical meaning

Interpretation



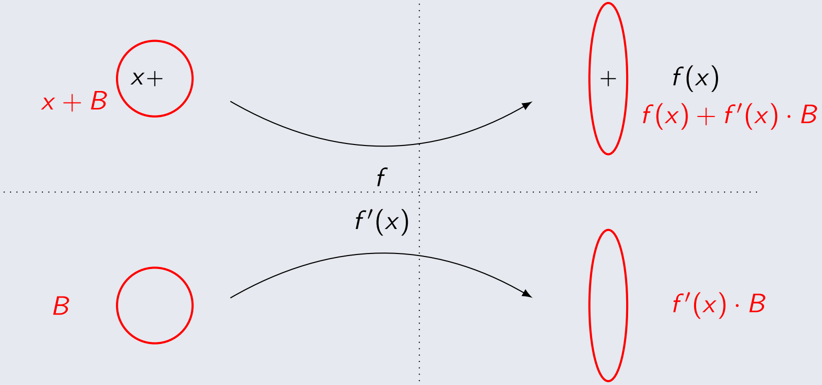
Geometrical meaning

Interpretation



Geometrical meaning

Interpretation



Lattices

Lattices

Lemma

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*
Then for any ball $B = B(0, r)$ ***small enough****,*

$$f(x + B) = f(x) + f'(x) \cdot B.$$

Lattices

Lemma

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*
Then for any ball $B = B(0, r)$ ***small enough****, for any open* ***lattice*** *$H \subset B$*

$$f(x + H) = f(x) + f'(x) \cdot H.$$

Lattices

Lemma

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*
Then for any ball $B = B(0, r)$ ***small enough****, for any open* ***lattice*** *$H \subset B$*

$$f(x + H) = f(x) + f'(x) \cdot H.$$

Remark

This allows more models of precision, like

$$(x, y) = (1 + O(p^{10}), 1 + O(p)).$$

Lattices

Lemma

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a (strictly) ***differentiable*** *mapping.*
Let $x \in \mathbb{Q}_p^n$. We assume that $f'(x)$ is ***surjective****.*
Then for any ball $B = B(0, r)$ ***small enough****, for any open* ***lattice*** *$H \subset B$*

$$f(x + H) = f(x) + f'(x) \cdot H.$$

Remark

This allows more models of precision, like

$$(x, y) = (1 + O(p^{10}), 1 + O(p)).$$

Remark

Our framework can be extended to **(complete) ultrametric K-vector spaces** (*e.g.* being $\mathbb{F}_p((X))^n$, $\mathbb{Q}((X))^m$, $\mathbb{R}((\varepsilon))^s$).

Higher differentials

Higher differentials

What is small enough ?

How can we determine when the lemma applies ?

Higher differentials

What is **small enough** ?

How can we determine when the lemma applies ?
When f is locally analytic, it essentially corresponds to

$$\sum_{k=2}^{+\infty} \frac{1}{k!} f^{(k)}(x) \cdot H^k \subset f'(x) \cdot H.$$

Higher differentials

What is **small enough** ?

How can we determine when the lemma applies ?
When f is locally analytic, it essentially corresponds to

$$\sum_{k=2}^{+\infty} \frac{1}{k!} f^{(k)}(x) \cdot H^k \subset f'(x) \cdot H.$$

This can be determined with **Newton-polygon** techniques.

Looking back to the case of the determinant

Differential of the determinant

It is well known:

$$\det{}'(M) : dM \mapsto \mathrm{Tr}(\mathrm{Com}(M) \cdot dM).$$

Looking back to the case of the determinant

Differential of the determinant

It is well known:

$$\det{}'(M) : dM \mapsto \mathrm{Tr}(\mathrm{Com}(M) \cdot dM).$$

Consequence on precision

- Loss in precision: coefficient of $\mathrm{Com}(M)$ with smallest valuation.

Looking back to the case of the determinant

Differential of the determinant

It is well known:

$$\det{}'(M) : dM \mapsto \mathrm{Tr}(\mathrm{Com}(M) \cdot dM).$$

Consequence on precision

- Loss in precision: coefficient of $\mathrm{Com}(M)$ with smallest valuation.
- Corresponds to the products of the $n-1$-first invariant factors.

Looking back to the case of the determinant

Differential of the determinant

It is well known:

$$\det{}'(M) : dM \mapsto \mathrm{Tr}(\mathrm{Com}(M) \cdot dM).$$

Consequence on precision

- Loss in precision: coefficient of $\mathrm{Com}(M)$ with smallest valuation.
- Corresponds to the products of the $n-1$-first invariant factors.
- **Approximate SNF is optimal.**

Table of contents

An example of p-adic algorithm

An example of p-adic algorithm

Isogeny and Differential equations (*cf* Schoof-Elkies-Atkin algorithm, Bostan-Morain-Salvy-Schost 08, Lercier-Sirvent 08, ...)

Let E and $\tilde{E}$ be two elliptic curves over $\mathbb{Z}/p\mathbb{Z}$:

$$E \; : \; y^2 = x^3 + Ax + B,$$
$$\tilde{E} \; : \; y^2 = x^3 + \tilde{A}x + \tilde{B}.$$

An example of p-adic algorithm

Isogeny and Differential equations (*cf* Schoof-Elkies-Atkin algorithm, Bostan-Morain-Salvy-Schost 08, Lercier-Sirvent 08, ...)

Let E and $\tilde{E}$ be two elliptic curves over $\mathbb{Z}/p\mathbb{Z}$:

$$\begin{aligned} E &: y^2 = x^3 + Ax + B, \\ \tilde{E} &: y^2 = x^3 + \tilde{A}x + \tilde{B}. \end{aligned}$$

Let us assume that there exists some normalized isogeny I between E and $\tilde{E}$. Then, for some rational fraction U,

$$I(x, y) = (U(x), yU'(x)),$$

An example of p-adic algorithm

Isogeny and Differential equations (*cf* Schoof-Elkies-Atkin algorithm, Bostan-Morain-Salvy-Schost 08, Lercier-Sirvent 08, ...)

Let E and $\tilde{E}$ be two elliptic curves over $\mathbb{Z}/p\mathbb{Z}$:

$$\begin{aligned} E &: y^2 = x^3 + Ax + B, \\ \tilde{E} &: y^2 = x^3 + \tilde{A}x + \tilde{B}. \end{aligned}$$

Let us assume that there exists some normalized isogeny I between E and $\tilde{E}$. Then, for some rational fraction U,

$$I(x, y) = (U(x), yU'(x)),$$

Writing $U = \frac{1}{S(\frac{1}{\sqrt{x}})^2}$, we get :

$$(Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6.$$

Change of variable and the differential equation

The differential equation

Let S be such that

$$U = \frac{1}{S(\frac{1}{\sqrt{x}})^2}.$$

Then if $A, B, \tilde{A}, \tilde{B}$ are in $\mathbb{Z}_p$,

$$S \in \mathbb{Z}_p[[t]]$$

We have the following differential equation for S :

$$(Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6.$$

A p-adic computation of a solution

Computing the isogeny

Given E and $\widetilde{E}$, the goal is to compute the isogeny I *via* the differential equation:

$$\begin{cases} S(0) = 0, \\ (Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6. \end{cases}$$

Going through $\mathbb{Z}_p$

Not easy to solve a differential equation in $\mathbb{Z}/p\mathbb{Z}$.

A p-adic computation of a solution

Computing the isogeny

Given E and $\widetilde{E}$, the goal is to compute the isogeny I *via* the differential equation:

$$\begin{cases} S(0) = 0, \\ (Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6. \end{cases}$$

Going through $\mathbb{Z}_p$

Not easy to solve a differential equation in $\mathbb{Z}/p\mathbb{Z}$. Consequently:

A p-adic computation of a solution

Computing the isogeny

Given E and $\widetilde{E}$, the goal is to compute the isogeny I *via* the differential equation:

$$\begin{cases} S(0) = 0, \\ (Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6. \end{cases}$$

Going through $\mathbb{Z}_p$

Not easy to solve a differential equation in $\mathbb{Z}/p\mathbb{Z}$. Consequently:

1. Lift (consistently) from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$.

A p-adic computation of a solution

Computing the isogeny

Given E and $\widetilde{E}$, the goal is to compute the isogeny I *via* the differential equation:

$$\begin{cases} S(0) = 0, \\ (Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6. \end{cases}$$

Going through $\mathbb{Z}_p$

Not easy to solve a differential equation in $\mathbb{Z}/p\mathbb{Z}$. Consequently:

1. Lift (consistently) from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$.
2. Solve the differential equation in $\mathbb{Z}_p$.

A p-adic computation of a solution

Computing the isogeny

Given E and $\widetilde{E}$, the goal is to compute the isogeny I *via* the differential equation:

$$\begin{cases} S(0) = 0, \\ (Bx^6 + Ax^4 + 1)S'^2 = 1 + \tilde{A}S^4 + \tilde{B}S^6. \end{cases}$$

Going through $\mathbb{Z}_p$

Not easy to solve a differential equation in $\mathbb{Z}/p\mathbb{Z}$. Consequently:

1. Lift (consistently) from $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$.
2. Solve the differential equation in $\mathbb{Z}_p$.
3. Reduce mod p to get the solution in $\mathbb{Z}/p\mathbb{Z}$.

Table of contents

Change of equation

When $p \neq 2$, we can replace $y'^2 \times G = H(y)$ by $y' = g \times h(y)$ with $g, h \in \mathbb{Z}_p^\times$.

Change of equation

When $p \neq 2$, we can replace $y'^2 \times G = H(y)$ by $y' = g \times h(y)$ with $g, h \in \mathbb{Z}_p^\times$.

Direct analysis

Newton scheme to solve $y' = g \times h(y)$:

Change of equation

When $p \neq 2$, we can replace $y'^2 \times G = H(y)$ by $y' = g \times h(y)$ with $g, h \in \mathbb{Z}_p^\times$.

Direct analysis

Newton scheme to solve $y' = g \times h(y)$:

$$N_{g,h}(u) \leftarrow u - h(u) \int \left(\frac{u'}{h(u)} - g \right).$$

Change of equation

When $p \neq 2$, we can replace $y'^2 \times G = H(y)$ by $y' = g \times h(y)$ with $g, h \in \mathbb{Z}_p^\times$.

Direct analysis

Newton scheme to solve $y' = g \times h(y)$:

$$N_{g,h}(u) \leftarrow u - h(u) \int \left(\frac{u'}{h(u)} - g \right).$$

Remark

$$\int O(p^m) x^k = \frac{O(p^m)}{k+1} x^{k+1}.$$

Change of equation

When $p \neq 2$, we can replace $y'^2 \times G = H(y)$ by $y' = g \times h(y)$ with $g, h \in \mathbb{Z}_p^\times$.

Direct analysis

Newton scheme to solve $y' = g \times h(y)$:

$$N_{g,h}(u) \leftarrow u - h(u) \int \left(\frac{u'}{h(u)} - g \right).$$

Remark

$$\int O(p^m) x^k = \frac{O(p^m)}{k+1} x^{k+1}.$$

One loses $O(N)$ digits at each step, for N the order of truncation.

Change of equation

When $p \neq 2$, we can replace $y'^2 \times G = H(y)$ by $y' = g \times h(y)$ with $g, h \in \mathbb{Z}_p^\times$.

Direct analysis

Newton scheme to solve $y' = g \times h(y)$:

$$N_{g,h}(u) \leftarrow u - h(u) \int \left(\frac{u'}{h(u)} - g \right).$$

Remark

$$\int O(p^m) x^k = \frac{O(p^m)}{k+1} x^{k+1}.$$

One loses $O(N)$ digits at each step, for N the order of truncation.
To compute $y \mod x^{2^N+1}$, we need an initial precision of $O(N^2)$ digits.

Table of contents

Differential and differential equation

Theorem

Let $\Phi : (g, h) \mapsto y$ *such that* $y(0) = 0$ *and* $y' = gh(y)$*. Then,*

$$\Phi'(g, h) \cdot (\delta g, \delta h) = h(y) \int \delta g + \frac{g \delta h(y)}{h(y)}.$$

Differential and differential equation

Theorem

Let $\Phi : (g, h) \mapsto y$ *such that* $y(0) = 0$ *and* $y' = gh(y)$. *Then,*

$$\Phi'(g, h) \cdot (\delta g, \delta h) = h(y) \int \delta g + \frac{g \delta h(y)}{h(y)}.$$

Proposition

In our case, $p \neq 2$, $y, g, h \in \mathbb{Z}_p[\![x]\!]$, $g(0) = h(0) = 1$. *If* $\delta g = \delta h = O(p^k)$, *then*

Differential and differential equation

Theorem

Let $\Phi : (g, h) \mapsto y$ *such that* $y(0) = 0$ *and* $y' = gh(y)$. *Then,*

$$\Phi'(g, h) \cdot (\delta g, \delta h) = h(y) \int \delta g + \frac{g \delta h(y)}{h(y)}.$$

Proposition

In our case, $p \neq 2$, $y, g, h \in \mathbb{Z}_p[\![x]\!]$, $g(0) = h(0) = 1$. *If* $\delta g = \delta h = O(p^k)$, *then*

$$\Phi'(y) \cdot (\delta g, \delta h) \mod x^{2^N+1} \in \frac{O(p^k)}{p^N} \mathbb{Z}_p[\![x]\!].$$

First conclusion on the application of the lemma

Proposition

$\Phi(g, h) \mod (p, t^{2^n})$ *is determined by* $g, h \mod (p^{1+\log_p 2^n}, t^{2^n}))$. *In other words, we have a logarithmic loss in precision.*

What happens in practice ?

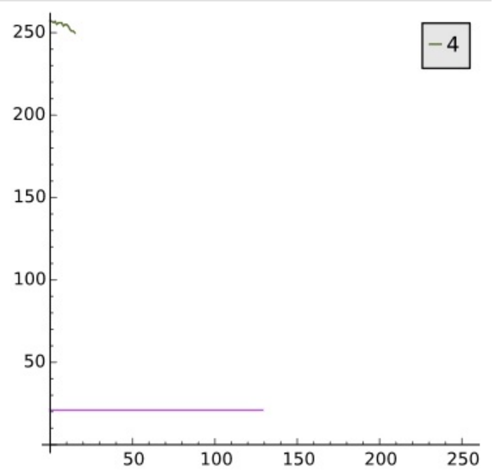


Figure: Precision over the output

What happens in practice ?

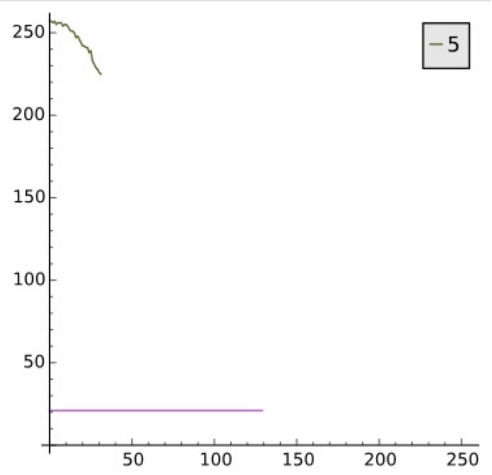


Figure: Precision over the output

What happens in practice ?

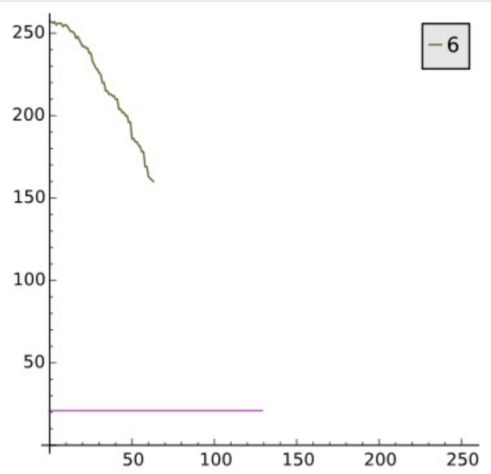


Figure: Precision over the output

What happens in practice ?

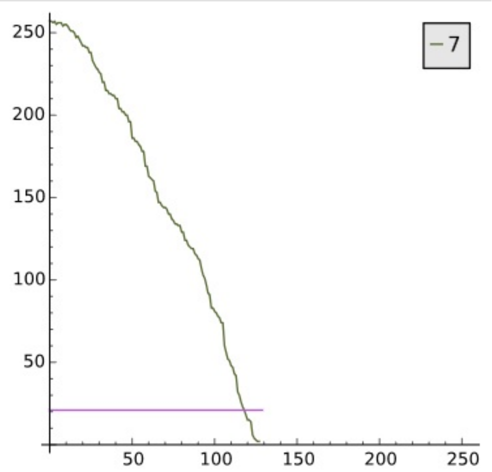


Figure: Precision over the output

Table of contents

Different way of representing the p-adics

Another take on the computation

Different way of representing the p-adics

Another take on the computation

- In the previous computation, we start with some given approximations of g, h, u_0 and try **to follow** the algorithm for the exact counterparts of g, h, u_0.

Different way of representing the p-adics

Another take on the computation

- In the previous computation, we start with some given approximations of g, h, u_0 and try **to follow** the algorithm for the exact counterparts of g, h, u_0. This is somehow **much stronger** than our desire: computing a good approximate solution.

Different way of representing the p-adics

Another take on the computation

- In the previous computation, we start with some given approximations of g, h, u_0 and try **to follow** the algorithm for the exact counterparts of g, h, u_0. This is somehow **much stronger** than our desire: computing a good approximate solution.
- Another way is then to modify the current g, h, u_0 **at each step**, in a consistent way, so as to keep on getting better approximate solutions.

Different way of representing the p-adics

Another take on the computation

- In the previous computation, we start with some given approximations of g, h, u_0 and try **to follow** the algorithm for the exact counterparts of g, h, u_0. This is somehow **much stronger** than our desire: computing a good approximate solution.
- Another way is then to modify the current g, h, u_0 **at each step**, in a consistent way, so as to keep on getting better approximate solutions.
- A third way here will be to work entirely in $\mathbb{Z}/p^{\kappa}\mathbb{Z}$.

New framework

In this new computation, we consider h as given, and not varying for the lemma.

Lemma

Let $Y : g \mapsto y$ such that $y(0) = 0$ and $y' = gh(y)$. Then,

$$Y'(g) \cdot (\delta g) = h(y) \int \delta g.$$

A consequence of the lemma

Corollary

Let $n > 0$ and $\kappa > 1$ be integers, and let $g \in \mathbb{Z}_p[\![t]\!]$ such that $Y(g) \pmod{t^{n+1}}$ has integer coefficients. For any $y \in \mathbb{Q}_p[\![t]\!]$ the following are equivalent:

1. $y = Y(\bar{g}) \pmod{t^{n+1}}$ *for some power series* $\bar{g} \in \mathbb{Z}_p[\![t]\!]$ *such that* $\int(\bar{g} - g) = 0 \pmod{p^\kappa}$;
2. $y = Y(g) \pmod{p^\kappa, t^{n+1}}$.

Final take on the Newton scheme

As a consequence, we can prove that it is harmless to work in $\mathbb{Z}/p^k\mathbb{Z}$ for our computation.

Proposition

We can obtain the solution $\Phi(g,h) \mod (p, t^{n+1})$ *knowing* $g, h \mod (p^{\lfloor \log_p n \rfloor + 1}, t^{n+1})$ *and applying the following iteration:*

$$N_{g,h}(u) \leftarrow u - h(u) \int \left(\frac{u'}{h(u)} - g \right),$$

Final take on the Newton scheme

As a consequence, we can prove that it is harmless to work in $\mathbb{Z}/p^k\mathbb{Z}$ for our computation.

Proposition

We can obtain the solution $\Phi(g,h) \mod (p, t^{n+1})$ *knowing* $g, h \mod (p^{\lfloor \log_p n \rfloor + 1}, t^{n+1})$ *and applying the following iteration:*

$$N_{g,h}(u) \leftarrow u - h(u) \int \left(\frac{u'}{h(u)} - g \right),$$

modulo $p^{\lfloor \log_p n \rfloor + 1}$ *and growing order of truncation.*

Timings

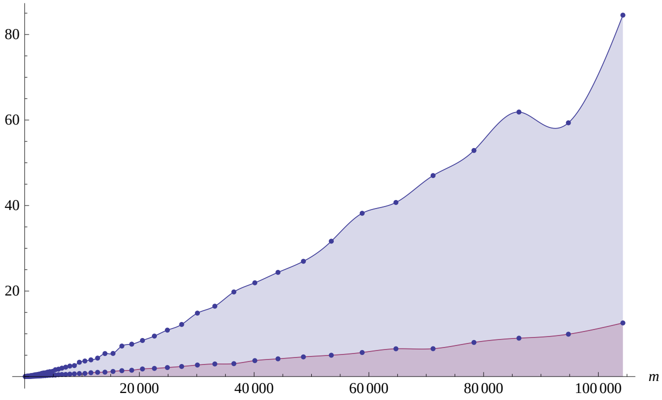


Figure: Timings in seconds, measured on a laptop, of our Algorithm run at precision λ_{old} (upper curve) and λ_{new} (lower curve) in order to compute an approximation modulo $(5, t^{4m+1})$ of some given m-isogenies.

Speedup

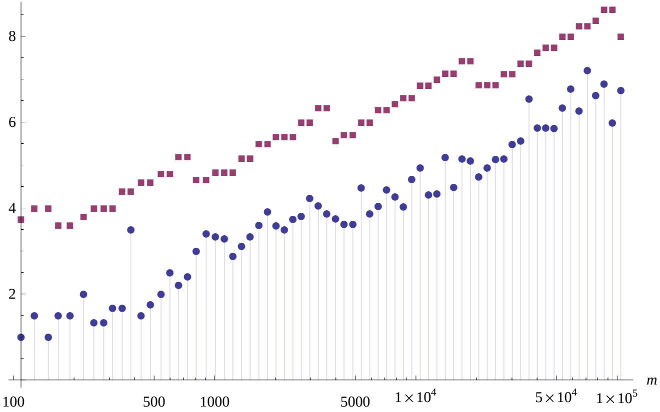


Figure: Practical speedup obtained with the new precision analysis compared with the theoretical improvement (m-axis in logarithmic scale). (■) is the ratio on precisions, (•) is the actual speedup.

To sum up

On p-adic precision

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.
- Differential calculus : **intrinsic** and can handle both **gain** and **loss**.

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.
- Differential calculus : **intrinsic** and can handle both **gain** and **loss**.
- Can stabilize and attain **optimal** precision, even though naïve computation lose too much precision.

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.
- Differential calculus : **intrinsic** and can handle both **gain** and **loss**.
- Can stabilize and attain **optimal** precision, even though naïve computation lose too much precision.

On differential equations

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.
- Differential calculus : **intrinsic** and can handle both **gain** and **loss**.
- Can stabilize and attain **optimal** precision, even though naïve computation lose too much precision.

On differential equations

- Can attain **optimal loss** in precision for differential equations with separation of variables.

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.
- Differential calculus : **intrinsic** and can handle both **gain** and **loss**.
- Can stabilize and attain **optimal** precision, even though naïve computation lose too much precision.

On differential equations

- Can attain **optimal loss** in precision for differential equations with separation of variables.
- Future works: higher order and $p = 2$.

To sum up

On p-adic precision

- Step-by-step analysis : as a first step. Can show differentiability and naïve loss in precision during the computation.
- Differential calculus : **intrinsic** and can handle both **gain** and **loss**.
- Can stabilize and attain **optimal** precision, even though naïve computation lose too much precision.

On differential equations

- Can attain **optimal loss** in precision for differential equations with separation of variables.
- Future works: higher order and $p = 2$.

References

Initial article

- XAVIER CARUSO, DAVID ROE AND TRISTAN VACCON Tracking p-adic precision, ANTS XI, 2014.

Linear Algebra

- XAVIER CARUSO, DAVID ROE AND TRISTAN VACCON p-adic stability in linear algebra, ISSAC 2015.

Differential equations

- PIERRE LAIREZ AND TRISTAN VACCON On p-adic differential equations with separation of variables, arXiv:1602.00244.

Thank you for your attention

Thanks