Précision p-adique

Journées Nationales du Calcul Formel 2014

X.Caruso, P.Lairez, D.Roe, **T.Vaccon**

University of Rennes I, TU Berlin, University of Calgary, **University of Rennes I**

6 novembre 2014

Motivation for p-adic algorithm

Why should one work with p-adic numbers ?

- Going from $\mathbb{F}_p$ to $\mathbb{Z}_p$ and then back to $\mathbb{F}_p$ enables more computation ;

Motivation for p-adic algorithm

Why should one work with p-adic numbers ?

- Going from $\mathbb{F}_p$ to $\mathbb{Z}_p$ and then back to $\mathbb{F}_p$ enables more computation ;
- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;

Motivation for p-adic algorithm

Why should one work with p-adic numbers ?

- Going from $\mathbb{F}_p$ to $\mathbb{Z}_p$ and then back to $\mathbb{F}_p$ enables more computation ;
- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- Some questions or algorithms are p-adic by nature.

Motivation for p-adic algorithm

Why should one work with p-adic numbers ?

- Going from $\mathbb{F}_p$ to $\mathbb{Z}_p$ and then back to $\mathbb{F}_p$ enables more computation ;
- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- Some questions or algorithms are p-adic by nature.

Some examples of essentially p-adic algorithms

- Polynomial factorization with Hensel lemma ;

Motivation for p-adic algorithm

Why should one work with p-adic numbers ?

- Going from $\mathbb{F}_p$ to $\mathbb{Z}_p$ and then back to $\mathbb{F}_p$ enables more computation ;
- Working in $\mathbb{Q}_p$ instead of $\mathbb{Q}$, one can handle more efficiently the coefficients growth ;
- Some questions or algorithms are p-adic by nature.

Some examples of essentially p-adic algorithms

- Polynomial factorization with Hensel lemma ;
- Kedlaya's counting-point algorithm on hyperelliptic curves with p-adic cohomology ;

1. p-adic precision
   - p-adic algorithms and step-by-step analysis
   - The differential approach
   - Improvements

2. Precision in practice
   - Linear Differential Equation (cf BGPS)
   - About implementation
   - Lifting method

Table of contents

Definition of the precision

Finite-precision p-adics

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

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

Optimality

Step-by-step analysis is not optimal.

Let $f: \begin{array}{ccc} \mathbb{Q}_p^2 & \to & \mathbb{Q}_p^2 \\ (x, y) & \mapsto & (x + y, x - y). \end{array}$

Optimality

Step-by-step analysis is not optimal.

Let $f : \begin{array}{ccc} \mathbb{Q}_p^2 & \to & \mathbb{Q}_p^2 \\ (x,y) & \mapsto & (x+y, x-y). \end{array}$

We would like to compute $f \circ f(x,y)$ with

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

Optimality

Step-by-step analysis is not optimal.

Let $f : \begin{array}{ccc} \mathbb{Q}_p^2 & \to & \mathbb{Q}_p^2 \\ (x,y) & \mapsto & (x+y, x-y). \end{array}$

We would like to compute $f \circ f(x,y)$ with $(x,y) = (1 + \boxed{O(p^{10})}, 1 + O(p))$.

- If we apply f two times, we get :

$$f \circ f(x,y) = (2 + \boxed{O(p)}, 2 + O(p)).$$

Optimality

Step-by-step analysis is not optimal.

Let $f : \begin{array}{ccc} \mathbb{Q}_p^2 & \to & \mathbb{Q}_p^2 \\ (x,y) & \mapsto & (x+y, x-y). \end{array}$

We would like to compute $f \circ f(x,y)$ with $(x,y) = (1 + \boxed{O(p^{10})}, 1 + O(p))$.

- If we apply f two times, we get :

$$f \circ f(x,y) = (2 + \boxed{O(p)}, 2 + O(p)).$$

- If we remark that $f \circ f = 2Id$, we get :

$$f \circ f(x,y) = (2 + \boxed{O(p^{10})}, 2 + O(p)).$$

Optimality

Step-by-step analysis is not optimal.

Let $f : \begin{array}{ccc} \mathbb{Q}_p^2 & \rightarrow & \mathbb{Q}_p^2 \\ (x,y) & \mapsto & (x+y, x-y). \end{array}$

We would like to compute $f \circ f(x,y)$ with $(x,y) = (1 + \boxed{O(p^{10})}, 1 + O(p))$.

- If we apply f two times, we get :

$$f \circ f(x,y) = (2 + \boxed{O(p)}, 2 + O(p)).$$

- If we remark that $f \circ f = 2Id$, we get :

$$f \circ f(x,y) = (2 + \boxed{O(p^{10})}, 2 + O(p)).$$

Table of contents

The Main lemma of p-adic differential precision

Lemma

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

The Main lemma of p-adic differential precision

Lemma

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a ***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

*Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a **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

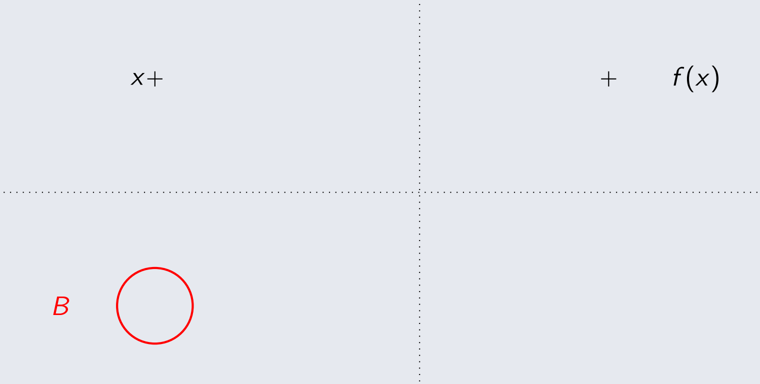
Lemma

*Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a **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**,*

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

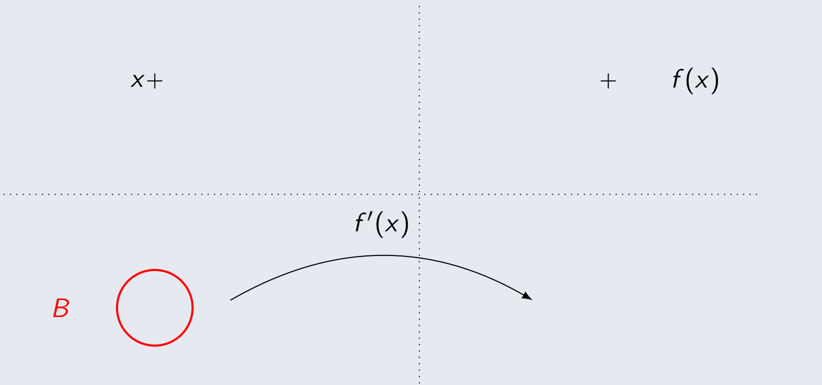
Geometrical meaning

Interpretation



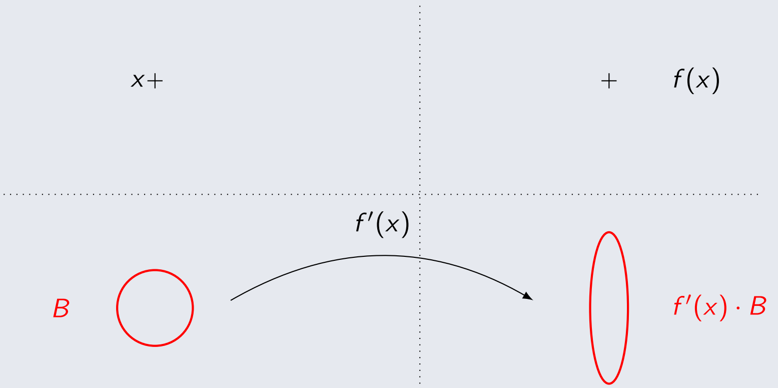
Geometrical meaning

Interpretation



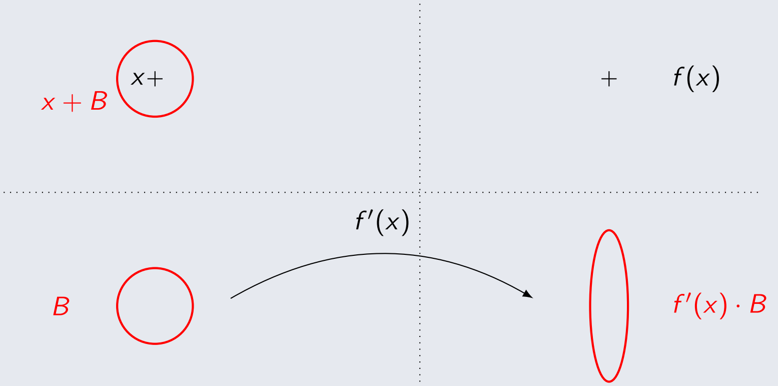
Geometrical meaning

Interpretation



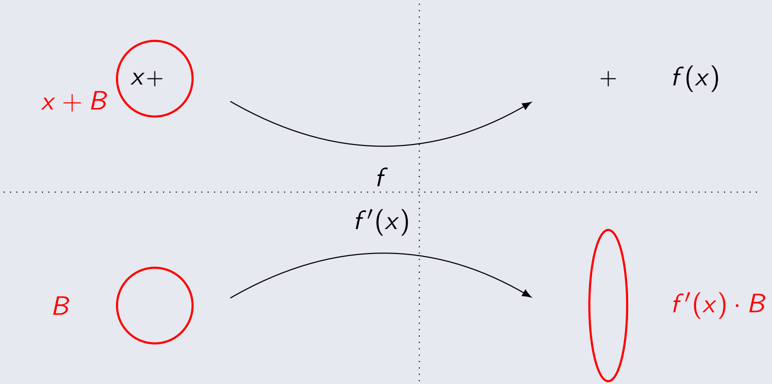
Geometrical meaning

Interpretation



Geometrical meaning

Interpretation



Geometrical meaning

Interpretation

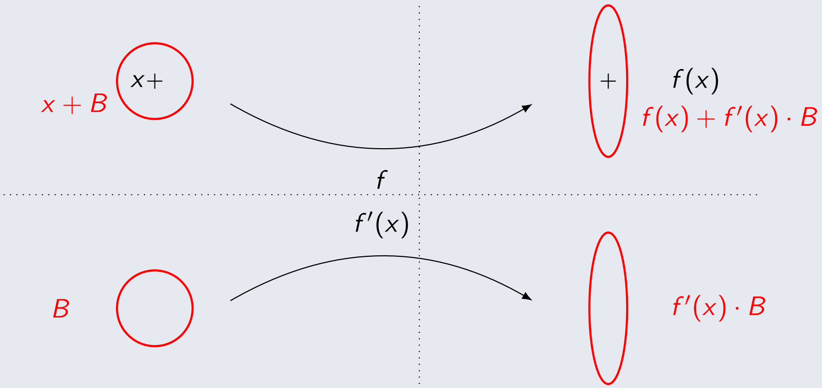


Table of contents

Lattices

Lattices

Lemma

Let $f : \mathbb{Q}_p^n \to \mathbb{Q}_p^m$ be a ***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 ***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 ***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 ***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.* $\mathbb{F}_p((X))^n$, $\mathbb{Q}((X))^m$).

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.

Some differentiable operations

Some more examples

We can apply our method to:

Some differentiable operations

Some more examples

We can apply our method to:

- On matrices: determinant, characteristic polynomial, LU factorization...

Some differentiable operations

Some more examples

We can apply our method to:

- On matrices: determinant, characteristic polynomial, LU factorization...
- On $\mathbb{Q}_p[X]$: evaluation, interpolation, GCD, factorization...

Some differentiable operations

Some more examples

We can apply our method to:

- On matrices: determinant, characteristic polynomial, LU factorization...
- On $\mathbb{Q}_p[X]$: evaluation, interpolation, GCD, factorization...
- On $\mathbb{Q}_p[X_1, \ldots, X_n]$: division, Gröbner bases.

Some differentiable operations

Some more examples

We can apply our method to:

- On matrices: determinant, characteristic polynomial, LU factorization...
- On $\mathbb{Q}_p[X]$: evaluation, interpolation, GCD, factorization...
- On $\mathbb{Q}_p[X_1, \ldots, X_n]$: division, Gröbner bases.

Table of contents

Newton series

Definition

Let $P \in k[X]$, $P = \prod_i (X - \alpha_i)$ (over $\overline{k}$). The **Newton series** of P is

$$f_P = \sum_n (\sum_i \alpha_i^n) t^n = -\frac{(P^*)'}{P^*},$$

with P^* the reciprocal polynomial of P.

Newton series

Definition

Let $P \in k[X]$, $P = \prod_i (X - \alpha_i)$ (over $\overline{k}$). The **Newton series** of P is

$$f_P = \sum_n \Big(\sum_i \alpha_i^n\Big) t^n = -\frac{(P^*)'}{P^*},$$

with P^* the reciprocal polynomial of P.

Remark

When $char(k) = 0$, by Newton's identities, we can recover P from f_P.

Computation over Newton series

Proposition

Let $P \times Q = \prod_i (X - \alpha_i) \prod_j (X - \beta_j)$ *and* $P \otimes Q = \prod_{i,j} (X - \alpha_i \beta_j)$.

Computation over Newton series

Proposition

Let $P \times Q = \prod_i (X - \alpha_i) \prod_j (X - \beta_j)$ *and* $P \otimes Q = \prod_{i,j} (X - \alpha_i \beta_j)$.
Then $f_{P \times Q} = f_P + f_Q$

Computation over Newton series

Proposition

Let $P \times Q = \prod_i (X - \alpha_i) \prod_j (X - \beta_j)$ *and* $P \otimes Q = \prod_{i,j} (X - \alpha_i \beta_j)$. *Then* $f_{P \times Q} = f_P + f_Q$ *and* $f_{P \otimes Q}$ *is obtained by the product coefficient by coefficient of* f_P *and* f_Q.

Computation over Newton series

Proposition

Let $P \times Q = \prod_i (X - \alpha_i) \prod_j (X - \beta_j)$ *and* $P \otimes Q = \prod_{i,j} (X - \alpha_i \beta_j)$. *Then* $f_{P \times Q} = f_P + f_Q$ *and* $f_{P \otimes Q}$ *is obtained by the product coefficient by coefficient of* f_P *and* f_Q.

Remark

How to do that on $\mathbb{Z}/p\mathbb{Z}$?

Computation over Newton series

Proposition

Let $P \times Q = \prod_i (X - \alpha_i) \prod_j (X - \beta_j)$ *and* $P \otimes Q = \prod_{i,j} (X - \alpha_i \beta_j)$. *Then* $f_{P \times Q} = f_P + f_Q$ *and* $f_{P \otimes Q}$ *is obtained by the product coefficient by coefficient of* f_P *and* f_Q.

Remark

How to do that on $\mathbb{Z}/p\mathbb{Z}$? We can lift $\mathbb{Z}/p\mathbb{Z}$ to $\mathbb{Z}_p$, and reduce mod p afterwards.

Idea of the algorithm

Proposition (How to recover P with f_P)

We notice that

$$(P^*)' = -f_P * P^*,$$

Idea of the algorithm

Proposition (How to recover P with f_P)

We notice that

$$(P^*)' = -f_P * P^*,$$

Therefore

$$y' = a(x) * y,$$

with $a(x) = -f_P$ *and* $y = P^*$.

Idea of the algorithm

Proposition (How to recover P with f_P)

We notice that

$$(P^*)' = -f_P * P^*,$$

Therefore

$$y' = a(x) * y,$$

with $a(x) = -f_P$ *and* $y = P^*$. *Thus, we can recover* P *with* f_P *by solving a linear differential equation (in* $\mathbb{Z}_p[[x]]$*).*

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

We differentiate the mapping sending a to the solution of $y' = a(t)y$. y_0 corresponds to P^*, and a to $-f_P$, in $\mathbb{Z}_p[[x]]$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

We differentiate the mapping sending a to the solution of $y' = a(t)y$.

$$(y_0 + \delta y)' = (a + \delta a)(y_0 + \delta y)$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

We differentiate the mapping sending a to the solution of $y' = a(t)y$.

$$(y_0 + \delta y)' = (a + \delta a)(y_0 + \delta y)$$
$$(\delta y)' = a\delta y + y_0 \delta a$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

We differentiate the mapping sending a to the solution of $y' = a(t)y$.

$$(\delta y)' = a\delta y + y_0 \delta a$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

We differentiate the mapping sending a to the solution of $y' = a(t)y$.

$$(\delta y)' = a\delta y + y_0 \delta a$$

By variation of parameters,

$$\delta y = y_0 \int y_0^{-1} \times y_0 \delta a \mathrm{d}t.$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

By variation of parameters,

$$\delta y = y_0 \int y_0^{-1} \times y_0 \delta a \mathrm{d}t.$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

By variation of parameters,

$$\delta y = y_0 \int y_0^{-1} \times y_0 \delta a \mathrm{d}t.$$

$$\delta y = y_0 \int \delta a \mathrm{d}t.$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

$$\delta y = y_0 \int \delta a \mathrm{d}t.$$

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\log_p(n))$.

Proof.

$$\delta y = y_0 \int \delta a \mathrm{d}t.$$

Since $\int \sum_i a_i x^i = \sum_i a_i \frac{x^{i+1}}{i+1}$, it yields a logarithmic loss in precision.

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\boxed{\log_p(n)})$.

Proof.

$$\delta y = y_0 \int \delta a \mathrm{d}t.$$

In other words, if $\delta a = O(p^k)$, *i.e.* $\delta a = \sum_i u_i \boxed{p^k} t^i$, then $\delta y = \sum_i v_i \boxed{\frac{p^k}{i+1}} t^i$.

Computation of the differential

Theorem

The loss in precision to recover the n-th coefficient of P with f_P is in $O(\boxed{\log_p(n)})$.

Proof.

$$\delta y = y_0 \int \delta a \mathrm{d}t.$$

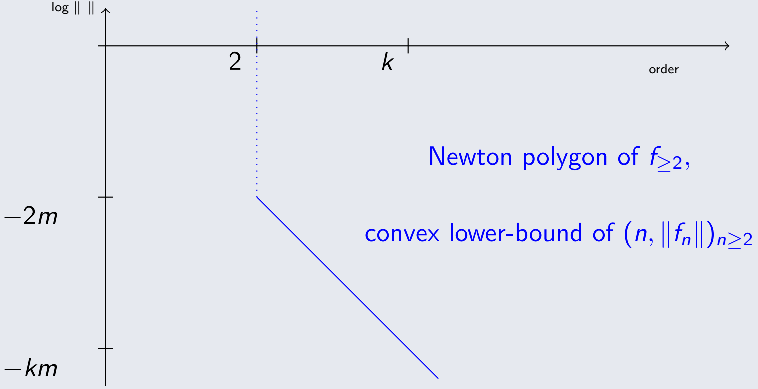
In other words, if $\delta a = O(p^k)$, *i.e.* $\delta a = \sum_i u_i \boxed{p^k} t^i$, then $\delta y = \sum_i v_i \boxed{\frac{p^k}{i+1}} t^i$.

Newton polygons technique

Newton and Legendre

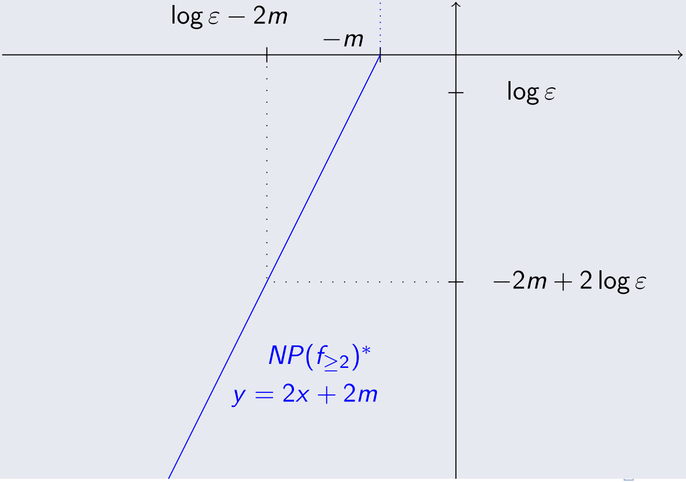
Newton polygons technique

Newton and Legendre



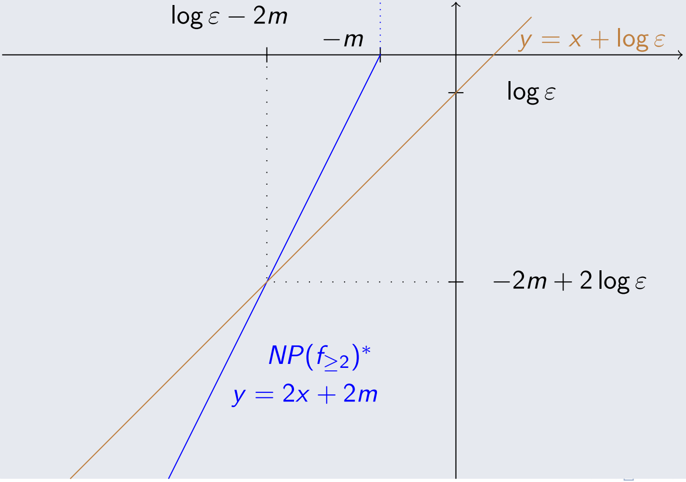
Newton polygons technique

Newton and Legendre



Newton polygons technique

Newton and Legendre



Newton polygons technique

Newton and Legendre

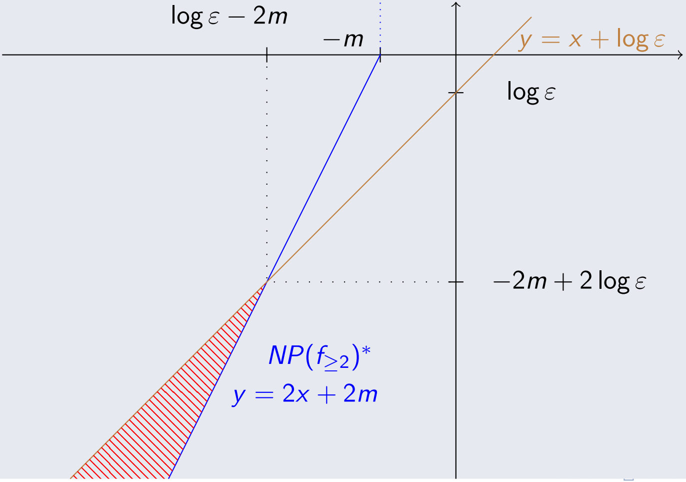


Table of contents

Newton scheme

Proposition

One can compute y_0 through some Newton scheme :

$$u_{l+1} \leftarrow u_l + \frac{u_l'}{a} \int (a \times u_l - u_l') \frac{a}{u_l'} + O(x^{2^{l+1}+1}).$$

Newton scheme

Proposition

One can compute y_0 through some Newton scheme :

$$u_{l+1} \leftarrow u_l + \frac{u_l'}{a} \int (a \times u_l - u_l') \frac{a}{u_l'} + O(x^{2^{l+1}+1}).$$

Remark

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

Newton scheme

Proposition

One can compute y_0 through some Newton scheme :

$$u_{l+1} \leftarrow u_l + \frac{u_l'}{a} \int (a \times u_l - u_l') \frac{a}{u_l'} + O(x^{2^{l+1}+1}).$$

Remark

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

One lose $E(\log_p(2^l))$ at each step. There are $E(\log_2(2^N))$ steps to reach x^{2^N}, it means a **naïve** loss in precision to compute the coefficient of x^{2^N} in $\left((\log 2^N)^2\right)$.

What happens in practice ?

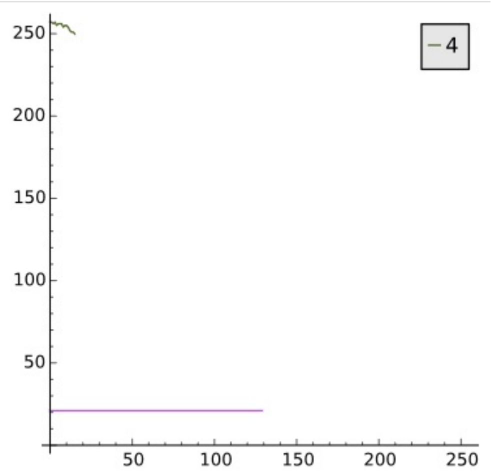


Figure: Precision over the output

What happens in practice ?

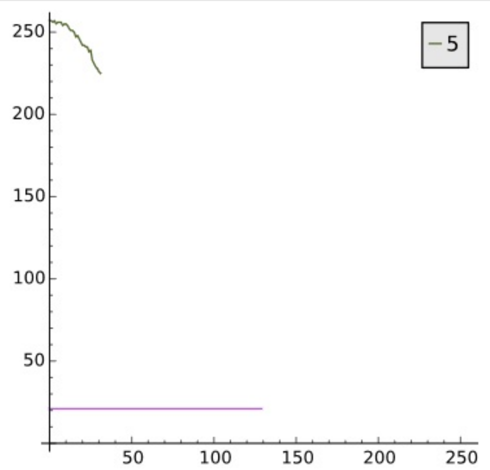


Figure: Precision over the output

What happens in practice ?

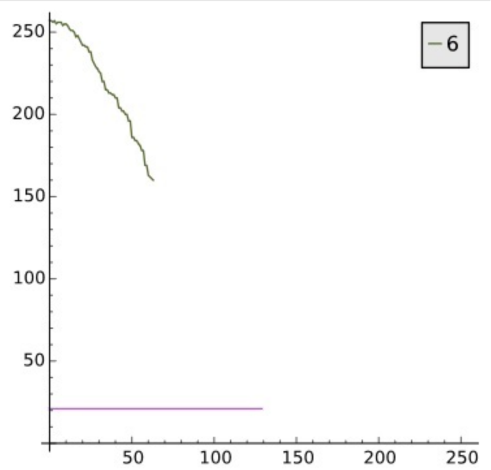


Figure: Precision over the output

What happens in practice ?

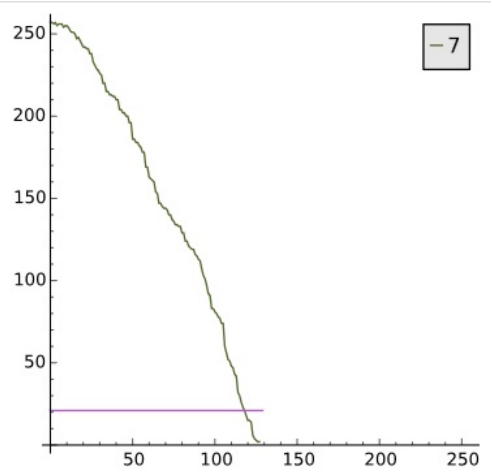


Figure: Precision over the output

Table of contents

More on the differential method

Differential tracking of precision

$x + O(p^N)$

$x+$

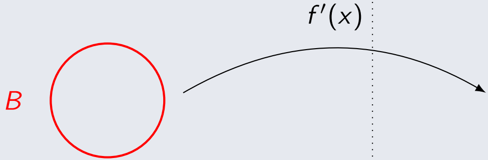
B

More on the differential method

Differential tracking of precision

$x + O(p^N)$

$x+$



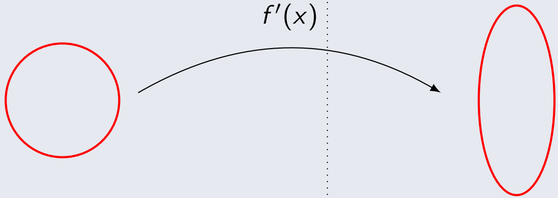
More on the differential method

Differential tracking of precision

$x + O(p^N)$

$x +$

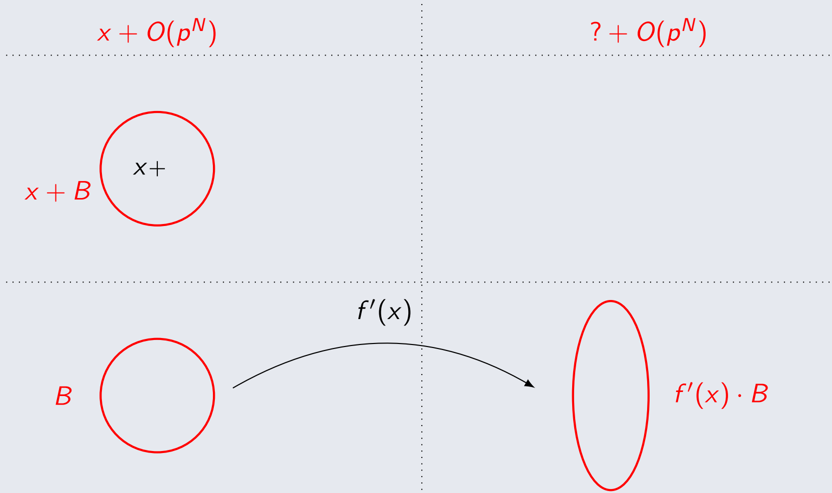
B



$f'(x) \cdot B$

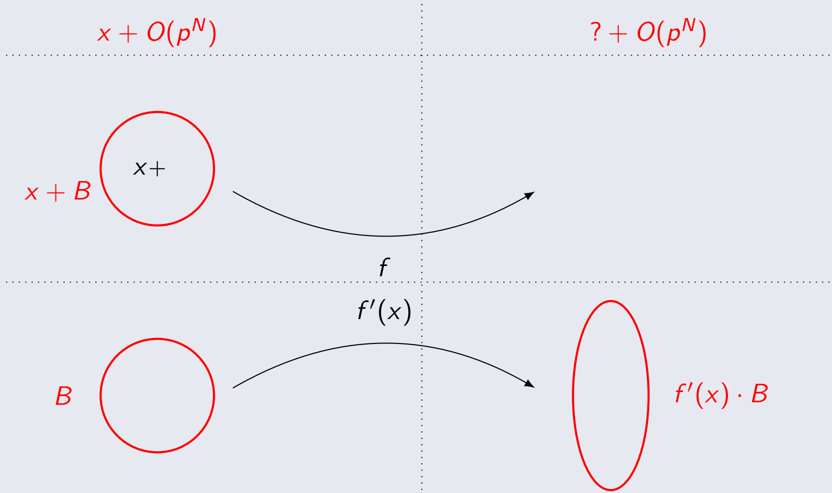
More on the differential method

Differential tracking of precision



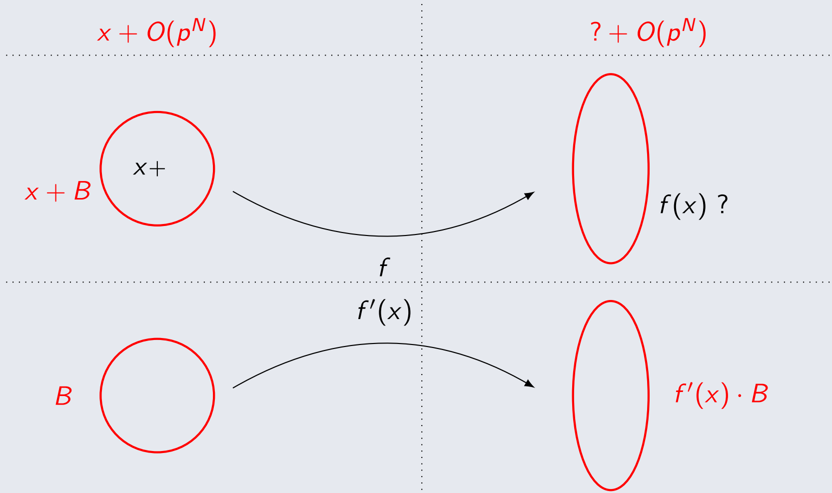
More on the differential method

Differential tracking of precision



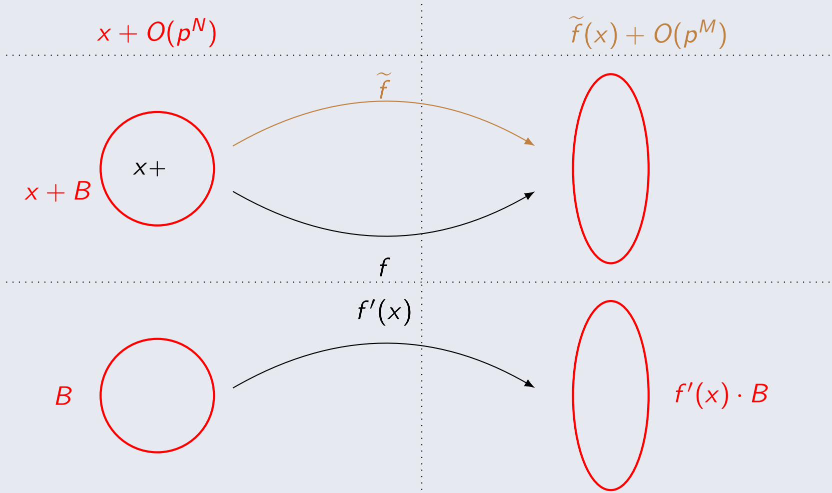
More on the differential method

Differential tracking of precision



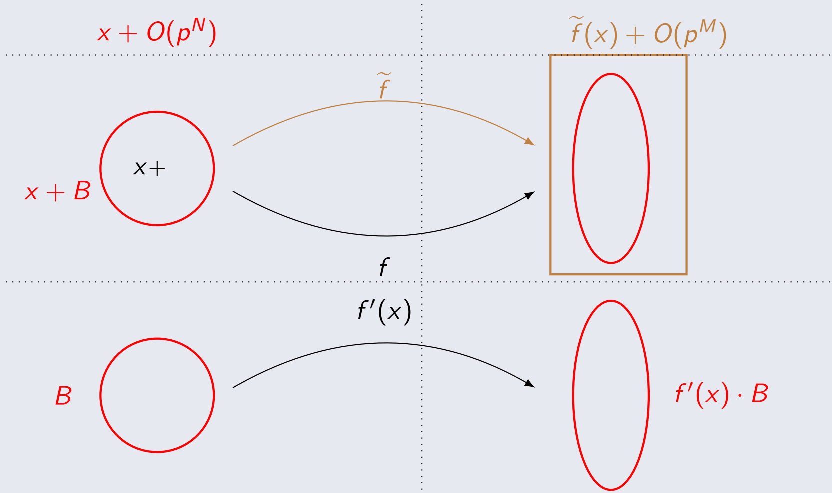
More on the differential method

Differential tracking of precision



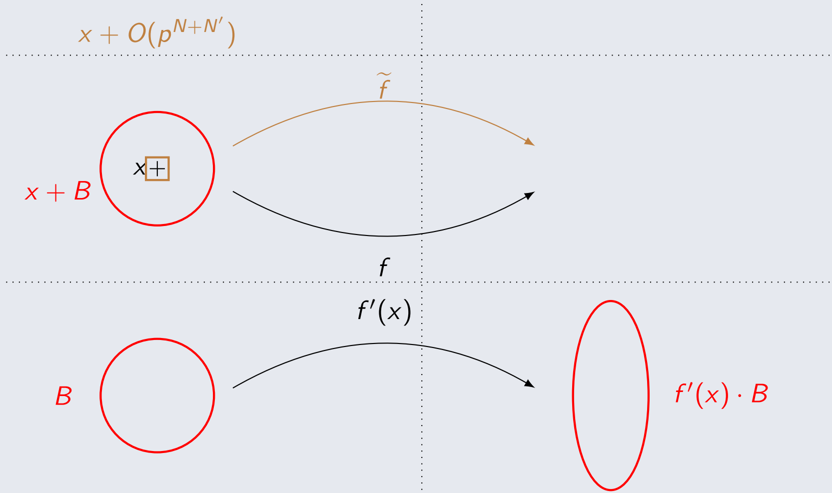
More on the differential method

Differential tracking of precision



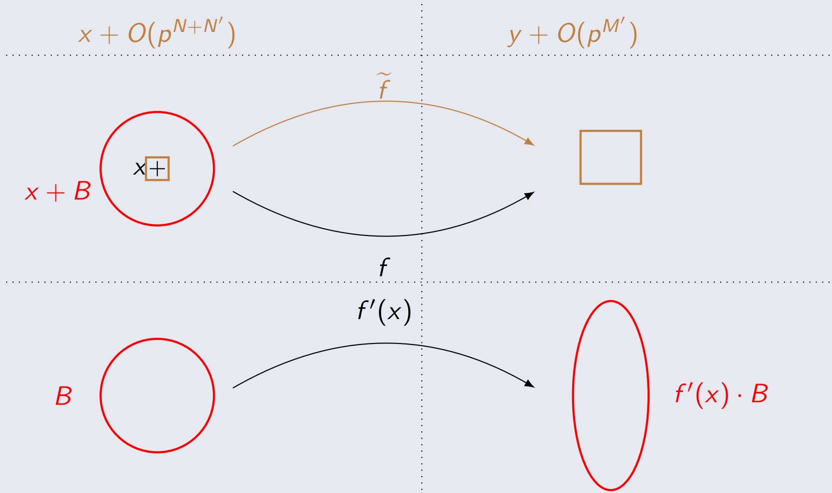
More on the differential method

Differential tracking of precision



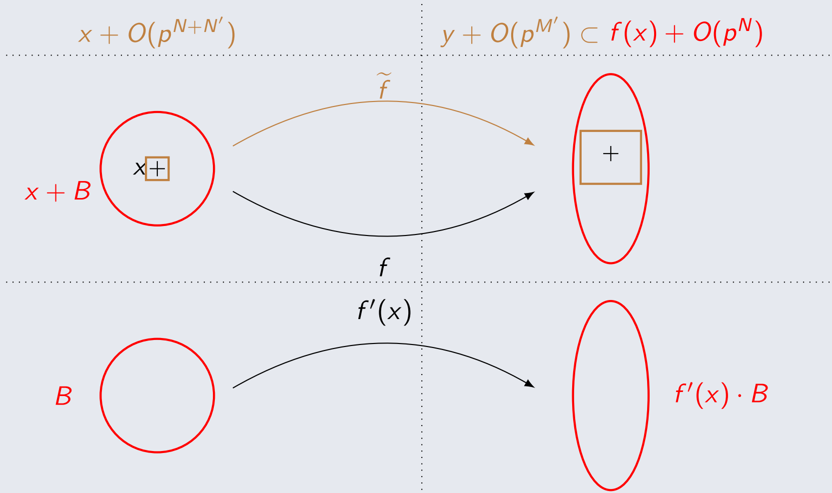
More on the differential method

Differential tracking of precision



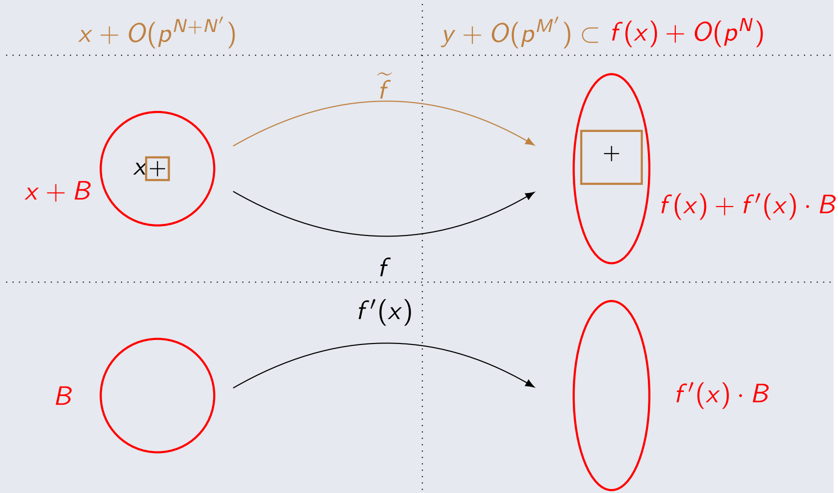
More on the differential method

Differential tracking of precision



More on the differential method

Differential tracking of precision



Buffer method

Proposition

One can use the following Newton scheme with lift:

- $\boxed{Lift}$ u_l to $O(p^{N+m})$,
- $a + O(x^{2^l}) \leftarrow u_l \times u_l'^{-1} + O(p^{N+m}, x^{2^l})$.
- $u_{l+1} \leftarrow u_l + \frac{u_l'}{a} \int (a \times u_l - u_l') \frac{a}{u_l'} + O(x^{2^{l+1}+1})$.

Numerical experiments

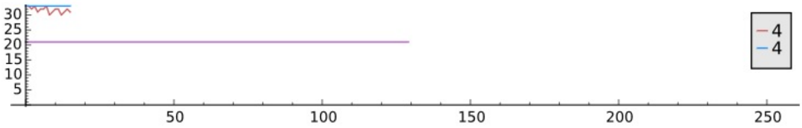


Figure: Precision over the output

Numerical experiments

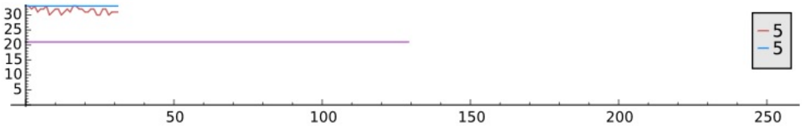


Figure: Precision over the output

Numerical experiments

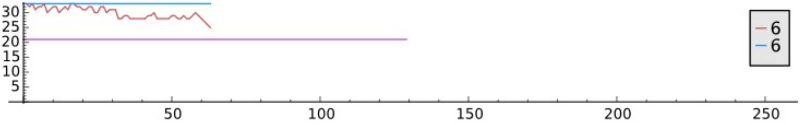


Figure: Precision over the output

Numerical experiments

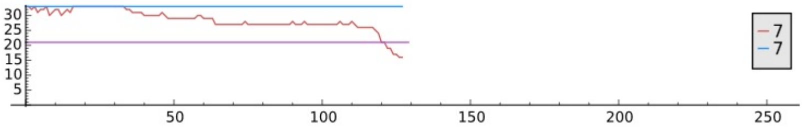


Figure: Precision over the output

Comparison of the methods

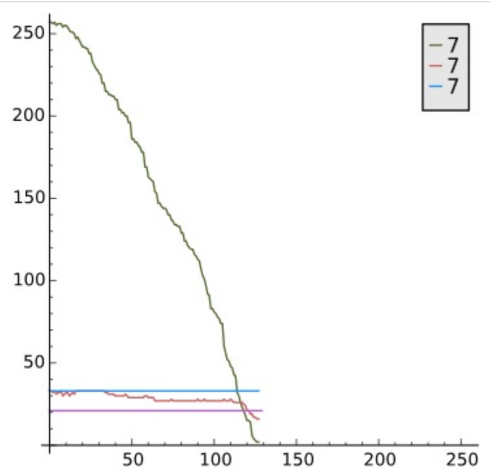


Figure: Naïve versus liftings

More differential equations

Generalization

This approach can be generalized to:

More differential equations

Generalization

This approach can be generalized to:

- Differential equation with separation of variables :

$$y' = g(x) \times h(y(x)).$$

More differential equations

Generalization

This approach can be generalized to:

- Differential equation with separation of variables :

$$y' = g(x) \times h(y(x)).$$

- Differential equations to compute normalized isogenies between elliptic curves :

$$y'^2 = g(x) \times h(y(x)).$$

  Cf Lercier, Sirvent On elkies subgroups of l-torsion points in elliptic curves defined over a finite field.

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.

Future works

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.

Future works

- Implement automatic computation of differentials, within 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**.
- Can stabilize and attain **optimal** precision, even though naïve computation lose too much precision.

Future works

- Implement automatic computation of differentials, within the computation.
- Analysis of the constants for classical operations (Work in progress).

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.

Future works

- Implement automatic computation of differentials, within the computation.
- Analysis of the constants for classical operations (Work in progress).
- More on differential equations.

References

Over real precision

- G.H.Golub, C.F.Van Loan Matrix Computation (John Hopkins University Press).

Over p-adic analysis

- P.Schneider p-adic Lie groups. (Springer).

Over these p-adic differential equations

- Bostan, González-Vega, Perdry, Schost From Newton sums to coefficients: complexity issues in characteristic p. (MEGA 05).

Over p-adic precision

- X.Caruso, D.Roe, T.Vaccon Tracking p-adic precision, ANTS XI.