Submitted exclusively to the London Mathematical Society
doi:10.1112/0000/000000

Tracking p-adic precision

Xavier Caruso, David Roe and Tristan Vaccon

Abstract

We present a new method to propagate p-adic precision in computations, which also applies to other ultrametric fields. We illustrate it with many examples and give a toy application to the stable computation of the SOMOS 4 sequence.

Contents

1. *Introduction*

The last two decades have seen a rise in the popularity of p-adic methods in computational algebra. For example,

- Bostan et al. [4] used Newton sums for polynomials over $\mathbb{Z}_p$ to compute composed products for polynomials over $\mathbb{F}_p$;
- Gaudry et al. [8] used p-adic lifting methods to generate genus 2 CM hyperelliptic curves;
- Kedlaya [10], Lauder [11] and many followers used p-adic cohomology to count points on hyperelliptic curves over finite fields;
- Lercier and Sirvent [12] computed isogenies between elliptic curves over finite fields using p-adic differential equations.

Like real numbers, most p-adic numbers cannot be represented exactly, but instead must be stored with some finite precision. In this paper we focus on methods for handling p-adic precision that apply across many different algorithms.

Two sources of inspiration arise when studying p-adic algorithms. The first relates $\mathbb{Z}_p$ to its quotients $\mathbb{Z}/p^n\mathbb{Z}$. The preimage in $\mathbb{Z}_p$ of an element $a \in \mathbb{Z}/p^n\mathbb{Z}$ is a ball, and these balls cover $\mathbb{Z}_p$ for any fixed n. Since the projection $\mathbb{Z}_p \to \mathbb{Z}/p^n\mathbb{Z}$ is a homomorphism, given unknown elements in two such balls we can locate the balls in which their sum and product lie. Working on a computer we must find a way to write elements using only a finite amount of data. By lumping elements together into these balls of radius p^{-n}, we may model arithmetic in $\mathbb{Z}_p$ using the finite ring $\mathbb{Z}/p^n\mathbb{Z}$. In this representation, all p-adic elements have constant *absolute precision* n.

The second source draws upon parallels between $\mathbb{Q}_p$ and $\mathbb{R}$. Both occur as completions of $\mathbb{Q}$ and we represent elements of both in terms of a set of distinguished rational numbers. In $\mathbb{R}$, floating point arithmetic provides approximate operations $\oplus$ and $\odot$ on a subset $S_{\infty,h} \subset \mathbb{Z}[\frac{1}{2}]$

2000 *Mathematics Subject Classification* 00000.

that model $+$ and $\cdot$ in $\mathbb{R}$ up to a given relative precision h:

$$\left|\frac{x \circledast y}{x * y} - 1\right| \leq 2^{-h}$$

for $* \in \{+, \cdot\}$ and all $x, y \in S_{\infty,h}$ with $x * y \neq 0$. The p-adic analogue defines floating point operations on $S_{p,h} \subset \mathbb{Z}[\frac{1}{p}]$ with

$$\left|\frac{x \circledast y}{x * y} - 1\right|_p \leq p^{-h}.$$

When using floating point arithmetic, elements are represented with a constant *relative precision* h.

In both of these models, precision (absolute or relative) is constant across all elements. Since some operations lose precision, it can be useful to attach a precision to each element. Over the reals, such interval arithmetic is unwieldy, since arithmetic operations always increase the lengths of the inputs. As a consequence, most computations in the real numbers rely on statistical cancelation and external estimates of precision loss, rather than attempting to track known precision at each step. This tendency is strengthened by the ubiquity of floating point arithmetic in scientific applications, where Gaussian distributions are more common than intervals anyway.

In the p-adic world, precision tracking using intervals is much more feasible. Even a long sequence of operations with such elements may not sacrifice any precision. Intervals allow number theorists to provably determine a result modulo a given power of p, and the Gaussian distributions of measurement error over $\mathbb{R}$ have no direct analogue over $\mathbb{Q}_p$ anyway. As a consequence, interval arithmetic is ubiquitous in implementations of p-adic numbers. The mathematical software packages Sage [18], PARI [1] and Magma [3] all include p-adic elements that track precision in this way.

The approach of propagating precision with each arithmetic operation works well, but does sometimes underestimate the known precision of a result, as we will discuss in Section 2.1. Moreover, elements of $\mathbb{Q}_p$ provide building blocks for generic implementations of polynomials, vector spaces, matrices and power series. The practice of storing the precision within each entry is not flexible enough for all applications. Sometimes only a rough accounting of precision is needed, in which case storing and computing the precision of each entry in a large matrix needlessly consumes space and time. Conversely, recording the precision of each entry does not allow a constraint such as specifying the precision of $f(0)$, $f(1)$ and $f(2)$ for a quadratic polynomial f.

For a vector space V over $\mathbb{Q}_p$, we propose that the fundamental object used to store the precision of an element should be a $\mathbb{Z}_p$-lattice $H \subset V$. By using general lattices one can eliminate needless loss of precision. Moreover, specifying the precision of each entry or recording a fixed precision for all entries can both be interpreted in terms of lattices. In Section 2 we detail our proposal for how to represent the precision of an element of a vector space.

In Section 3, we develop the mathematical background on which our proposal is based. The most notable result of this section is Lemma 3.4 which describes how lattices transform under non-linear maps and allows us to propagate precision using differentials. More specifically, it describes a class of first order lattices, whose image under a map of Banach spaces is obtained by applying the differential of that map. In Section 3.2 we make the conditions of Lemma 3.4 more explicit in the case of locally analytic functions.

In Section 4 we propose methods for tracking precision in practice. Section 4.1 includes a discussion of two models of precision tracking: one-pass tracking, where the precision lattice is propagated at each step of the algorithm, and two-pass tracking, where an initial pass computing rough approximations is used in computing the precision lattices. We introduce precision types in Section 4.2, which allow a tradeoff between flexibility, space and time in computing with precision. In Section 4.3, we give an application of these ideas to an algorithm for computing terms of the SOMOS sequence.

In Appendix A, we extend the results of Section 3 to p-adic manifolds, describing how to specify precisions for points on elliptic curves and Grassmannians. Finally, Appendix B describes how to compute the derivative of some common operations on matrices as an example, with an eye toward applying Lemma 3.4.

2. *Precision proposals*

2.1. *Problems in precision*

The usual way to track p-adic precision consists in replacing p-adic numbers by approximate elements of the form $a + O(p^N)$ and performing all usual arithmetical operations on these approximations. We offer below three examples that illustrate cases where this way to track precision does not yield optimal results.

A linear map. Consider the function $f : \mathbb{Q}_p^2 \to \mathbb{Q}_p^2$ mapping (x, y) to $(x + y, x - y)$ and the problem of computing $f \circ f(a + O(p^n), b + O(p^m))$. Applying f twice, computing precision with each step, yields $\big(2a + O(p^{\min(m,n)}), 2b + O(p^{\min(m,n)})\big)$. On the other hand, $f \circ f(x, y) = (2x, 2y)$, so one may compute the result more accurately as $(2a + O(p^n), 2b + O(p^m))$, with even more precision when $p = 2$.

SOMOS 4. The SOMOS 4 sequence [17] is defined by the recurrence

$$u_n = \frac{u_{n-3}u_{n-1} + u_{n-2}^2}{u_{n-4}}.$$

We shall consider the case where the initial terms u_0, u_1, u_2 and u_3 lie in $\mathbb{Z}_p^\times$ and have precision $O(p^N)$. Let us first examine how the absolute precision of u_n varies with n if it is computed from the precision of $u_{n-4}, \dots, u_{n-1}$ using the recurrence. The computation of u_n involves a division by u_{n-4} and hence, roughly speaking, decreases the precision by a factor $p^{\mathrm{val}(u_{n-4})}$. Hence the step-by-step computation returns the value of u_n with precision

$$O(p^{N-v_n}) \quad \text{with} \quad v_n = \mathrm{val}(u_0) + \cdots + \mathrm{val}(u_{n-4}). \tag{2.1}$$

On the other hand, one can prove that the SOMOS 4 sequence exhibits the *Laurent phenomenon* [7]: for all integer n, there exists a polynomial P_n in $\mathbb{Z}[X^{\pm 1}, Y^{\pm 1}, Z^{\pm 1}, T^{\pm 1}]$ such that $u_n = P_n(u_0, u_1, u_2, u_3)$. From the latter formula and our assumption that u_0, u_1, u_2 and u_3 are units known up to precision $O(p^N)$, it follows that all u_n's are known with the same precision. Thus, the term v_n that appears in (2.1) does not reflect an intrinsic loss of precision but some numerical instability related to the algorithm used to compute u_n.

Remark 2.1. *From the above discussion, one can easily derive a numerically stable algorithm that computes the SOMOS 4 sequence:*
(i) *compute the polynomials P_n using the recurrence in the ring $\mathbb{Z}[X^{\pm 1}, Y^{\pm 1}, Z^{\pm 1}, T^{\pm 1}]$*
(ii) *evaluate P_n at the point (u_0, u_1, u_2, u_3).*
However, computing the P_n's is very time-consuming since it requires division in a polynomial ring with 4 variables and the size of the coefficients of P_n explodes as n grows.

In Section 4.3, we shall design an algorithm computing the SOMOS 4 sequence which turns out to be, at the same time, efficient and numerically stable.

LU factorization. Let us first recall that a square matrix M admits a LU factorization if it can be written as a product LU where L and U are lower triangular and upper triangular respectively. The computation of a LU factorization appears as an important tool to tackle many classical questions about matrices or linear systems, and is discussed further in Appendix B. When computing the entries of L and U from a $d \times d$ matrix over $\mathbb{Z}_p$ with entries of precision $O(p^N)$, one has a choice of algorithms:

- using usual Gaussian elimination and tracking p-adic precision step-by-step, the smallest precision on an entry of $L(M)$ is about $O(p^{N-\frac{2d}{p-1}})$ on average;
- computing $L(M)$ by evaluating Cramer-type formulae yields a result whose every entry is known up to precision $O(p^{N-2\log_p d})$ [5].

If d is large compared to p, the second precision is much more accurate than the first one. On the other hand, the Cramer-type formulae in the second algorithm yield a substantially longer running time than the first.

2.2. *Lattices*

In order to make our proposals for tracking precision clear, we need some definitions from ultrametric analysis. See Schneider [16] for a more complete exposition.

Let K be a field with absolute value $|\cdot| : K \to \mathbb{R}_{\geq 0}$. We assume that the induced metric is an ultrametric (*i.e.* $|x+y| \leq \max(|x|,|y|)$) and that K is complete with respect to it. For example, we may take $K = \mathbb{Q}_p$ with the p-adic absolute value or $K = k((t))$ with the t-adic absolute value. Write $\mathcal{O}_K$ for the ring $\{x \in K : |x| \leq 1\}$ and assume that K contains a computable dense subring $R \subset K$ [14]. This assumption holds for $K = \mathbb{Q}_p$ and $K = \mathbb{F}_p((t))$ by setting $R = \mathbb{Z}[\frac{1}{p}]$ or $R = \mathbb{Q}$ in the case of $\mathbb{Q}_p$ and $R = \mathbb{F}_p[t,t^{-1}]$ or $R = \mathbb{F}_p(t)$ in the case of $\mathbb{F}_p((t))$.

If E is a K-vector space, possibly of infinite dimension, then an *ultrametric norm* on E is a map $\|\cdot\| : E \to \mathbb{R}^+$ satisfying:

(i) $\|x\| = 0$ if and only if $x = 0$;
(ii) $\|\lambda x\| = |\lambda| \cdot \|x\|$;
(iii) $\|x+y\| \leq \max(\|x\|,\|y\|)$.

A *K-Banach space* is a complete normed K-vector space. Note that any finite-dimensional normed K-vector space is automatically complete and all norms over such a space are equivalent. A *lattice* in a K-Banach space E is an open bounded sub-$\mathcal{O}_K$-module of E. We underline that any lattice H in E is also closed since its complement is the union of all cosets $a + H$ (with $a \notin H$) which are all open. For a K-Banach space E and $r \in \mathbb{R}_{\geq 0}$, write

$$B_E(r) = \{x \in E : \|x\| \leq r\}, \quad B_E^-(r) = \{x \in E : \|x\| < r\}.$$

Note that $B_E(r)$ and $B_E^-(r)$ are both lattices. We will also set $B_E(\infty) = B_E^-(\infty) = E$.

Suppose E is a K-Banach space and I a set. A family $(x_i)_{i\in I} \subset E$ is a *Banach basis* for E if every element $x \in E$ can be written $x = \sum_{i\in I} \alpha_i x_i$ for scalars $\alpha_i \in K$ with $\alpha_i \to 0$ (according to the filter of cofinite subsets), and $\|x\| = \sup_{i\in I}|\alpha_i|$. Note that if E is finite dimensional then the condition $\alpha_i \to 0$ is vacuous.

Given a basis $(x_i)_{i\in I}$ and a sequence $(r_i)_{i\in I}$ with $r_i \in \mathbb{R}_{>0}$, the sets

$$B_E((x_i),(r_i)) = \Big\{\sum_{i\in I} \alpha_i x_i : |\alpha_i| \leq r_i\Big\},$$
$$B_E^-((x_i),(r_i)) = \Big\{\sum_{i\in I} \alpha_i x_i : |\alpha_i| < r_i\Big\}$$

are lattices precisely when the r_i are bounded. If we have equipped E with a distinguished basis then we may drop (x_i) from the notation for $B_E^{(-)}((x_i),(r_i))$.

Approximate elements. Suppose that E is a K-Banach space with basis $(x_i)_{i\in I}$.

Definition 2.2.

- *An element $x \in E$ is* exact *if there is a finite subset $J \subseteq I$ and scalars $\alpha_j \in R$ with*
$$x = \sum_{j \in J} \alpha_j x_j. \tag{2.2}$$
- *An* approximate element *is a pair (x, H) where $x \in E$ is an exact element and H is a lattice in E.*

The pair (x, H) represents an undetermined element of the coset $x + H$. We will frequently write $x + O(H)$ to emphasize the fact that H represents the uncertainty in the value of the approximate element. In the special case that $E = K = \mathbb{Q}_p$, we recover the standard notation $a + O(p^n)$ for an approximate p-adic element. Note that the set of exact elements is dense in E, so every element of E can be approximated.

Lattices and computers. Suppose that $E \simeq K^d$ is finite dimensional. Then if $H \subset E$ is a lattice then there exist $a, b \in \mathbb{Q}_{>0}$ with
$$B_K(a)^d \subset H \subset B_K(b)^d. \tag{2.3}$$
Set $r = \frac{a}{b}$ and $R_r = \mathcal{O}_K / B_K(r)$. Then a lattice H satisfying (2.3) is uniquely determined by its image in the quotient $B_K(b)^d / B_K(a)^d \simeq R_r^d$. Since $R \cap \mathcal{O}_K$ is dense in $\mathcal{O}_K$, elements of R_r may be represented exactly. Thus H may be encoded as a $(d \times d)$ matrix with coefficients in R_r. For example, when $K = \mathbb{Q}_p$ the ring R_r is just $(\mathbb{Z}/p^n\mathbb{Z})$ for $n = \lfloor -\log_p r \rfloor$.

2.3. *Separating precision from approximation*

Definition 2.2 encapsulates the two main practical suggestions of this paper with regards to representing vector spaces, matrices, polynomials and power series over K:

(1) one should **separate** the approximation from the precision,
(2) the appropriate object to represent precision is a **lattice**.

In the rest of this section we discuss some of the benefits made possible these choices.

Note first that using an arbitrary lattice to represent the precision of an approximate element can reduce precision loss when compared to storing the precision of each coefficient α_i in (2.2) separately. Recall the map $f : (x, y) \mapsto (x + y, x - y)$ from the beginning of the section, and write (e_1, e_2) for the standard basis of $E = \mathbb{Q}_p^2$. Since f is linear, the image of the approximation $\big((a, b), B_E\big((e_1, e_2), (p^{-n}, p^{-m})\big)\big)$ is $\big((a + b, a - b), B_E\big((e_1 + e_2, e_1 - e_2), (p^{-n}, p^{-m})\big)\big)$. For $p \neq 2$, applying f again yields $\big((2a, 2b), B_E\big((e_1, e_2), (p^{-n}, p^{-m})\big)\big)$. By using lattices one eliminates the loss of precision seen previously. We shall see in the next section that a similar phenomenon occurs for non-linear mappings as well.

In addition to allowing for a more flexible representation of the precision of an element, the separation of precision from approximation has other benefits as well. If the precision is encoded with the approximation, certain algorithms become unusable because of their numerical instability. For example, the Karatsuba algorithm for polynomial multiplication [9] can needlessly lose precision when operating on polynomials with inexact coefficients. However, it works perfectly well on exact approximations, leaving the question of the precision of the product to be solved separately. By separating the precision, more algorithms become available.

3. *Lattices and differentials*

Our theory of p-adic precision rests upon a lemma in p-adic analysis: Lemma 3.4. This section develops the theory surrounding this result; we proceed to practical consequences in Section 4.

3.1. *Images of lattices under differentiable functions*

Our goal in this section is to relate the image of a lattice under a differentiable map to its image under the derivative.

Definition 3.1. *Let E and F be two K-Banach spaces, let U be an open subset of E and let $f : U \to F$ be a map. Then f is called* differentiable *at $v_0 \in U$ if there exists a continuous linear map $f'(v_0) : U \to W$ such that for any $\varepsilon > 0$, there exists an open neighborhood $U_\varepsilon \subset U$ containing v_0 with*

$$\|f(v) - f(w) - f'(v_0)\,(v - w)\,\| \leq \varepsilon\|v - w\|.$$

for all $v, w \in U_\varepsilon$. The linear map $f'(v_0)$ is called the differential *of f at v_0.*

Remark 3.2. *This notion of differentiability is sometimes called* strict differentiability*; it implies that the function $x \mapsto f'(x)$ is continuous on U.*

Definition 3.3. *Let E and F be two K-Banach spaces, $f : U \to F$ be a function defined on an open subset U of E and v_0 be a point in U. A lattice H in E is called a* first order lattice *for f at v_0 if $v_0 + H \subset U$ and the following equality holds:*

$$f(v_0 + H) = f(v_0) + f'(v_0)(H). \tag{3.1}$$

We emphasize that we require an equality in (3.1), and not just an inclusion! With this definition in hand, we are able to state our main lemma.

Lemma 3.4. *Let E and F be two K-Banach spaces and $f : U \to F$ be a function defined on an open subset U of E. We assume that f is differentiable at some point $v_0 \in U$ and that the differential $f'(v_0)$ is surjective.*

Then, for all $\rho \in (0, 1]$, there exists a positive real number δ such that, for all $r \in (0, \delta)$, any lattice H such that $B^-_E(\rho r) \subset H \subset B_E(r)$ is a first order lattice for f at v_0.

Proof. Without loss of generality, $v_0 = 0$ and $f(0) = 0$. Since $f'(0)$ is surjective, the open mapping theorem provides a $C > 0$ such that $B_F(1) \subset f'(0)(B_E(C))$. Let $\varepsilon > 0$ be such that $\varepsilon C < \rho$, and choose $U_\varepsilon \subset E$ as in Definition 3.1. We may assume $U_\varepsilon = B_E(\delta)$ for some $\delta > 0$.

Let $r \in (0, \delta)$. We suppose that H is a lattice with $B^-_E(\rho r) \subset H \subset B_E(r)$. We seek to show that f maps H surjectively onto $f'(0)(H)$. We first prove that $f(H) \subset f'(0)(H)$. Suppose $x \in H$. By differentiability at 0, $\|f(x) - f'(0)(x)\| \leq \varepsilon\|x\|$. Setting $y = f(x) - f'(0)(x)$, we have $\|y\| \leq \varepsilon r$. The definition of C implies that $B_F(\varepsilon r) \subset f'(0)(B_E(\varepsilon rC))$. Thus there exists $x' \in B_E(\varepsilon rC)$ such that $f'(0)(x') = y$. Since $\varepsilon C < \rho$, we get $x' \in B^-_E(\rho r) \subset H$ and then $f(x) = f'(0)(x - x') \in f'(0)(H)$.

We now prove surjectivity. Let $y \in f'(0)(H)$. Let $x_0 \in H$ be such that $y = f'(0)(x_0)$. We inductively define two sequences (x_n) and (z_n) as follows:

- z_n is an element of E satisfying $f'(0)(z_n) = y - f(x_n)$ and $\|z_n\| \leq C \cdot \|y - f(x_n)\|$ (such an element exists by definition of C), and
- $x_{n+1} = x_n + z_n$.

For convenience, let us also define $x_{-1} = 0$ and $z_{-1} = x_0$. We claim that the sequences (x_n) and (z_n) are well defined and take their values in H. We do so by induction, assuming that

x_{n-1} and x_n belong to H and showing that z_n and x_{n+1} do as well. Noticing that

$$\begin{aligned} y - f(x_n) &= f(x_{n-1}) + f'(0)(z_{n-1}) - f(x_n) \\ &= f(x_{n-1}) - f(x_n) - f'(0)(x_{n-1} - x_n) \end{aligned} \tag{3.2}$$

we deduce using differentiability that $\|y - f(x_n)\| \leq \varepsilon \cdot \|x_n - x_{n-1}\|$. Since we are assuming that x_{n-1} and x_n lie in $H \subset B_E(r)$, we find $\|y - f(x_n)\| \leq \varepsilon r$. Thus $\|z_n\| \leq C \cdot \varepsilon r < \rho r$ and then $z_n \in H$. From the relation $x_{n+1} = x_n + z_n$, we finally deduce $x_{n+1} \in H$.

Using (3.2) and differentiability at 0 once more, we get

$$\|y - f(x_n)\| \leq \varepsilon \cdot \|z_{n-1}\| \leq \varepsilon C \cdot \|y - f(x_{n-1})\|,$$

for all $n > 0$. Therefore, $\|y - f(x_n)\| = O(a^n)$ and $\|z_n\| = O(a^n)$ for $a = \varepsilon C < \rho \leq 1$. These conditions show that (x_n) is a Cauchy sequence, which converges since E is complete. Write x for the limit of the x_n; we have $x \in H$ because H is closed. Moreover, f is continuous on $H \subseteq U_\varepsilon$ since it is differentiable, and thus $y = f(x)$. □

We end this section with a remark on the surjectivity of $f'(v_0)$ assumed in Lemma 3.4. First, let us emphasize that this hypothesis is definitely necessary. Indeed, the lemma would otherwise imply that the image of f is locally contained in a proper sub-vector-space around each point where the differential of f is not surjective, which is certainly not true! Nevertheless, one can use Lemma 3.4 to prove a weaker result in the context that $f'(v_0)$ is not surjective. To do so, choose a closed sub-vector-space W of F such that $W + f'(v_0)(E) = F$. Denoting by pr_W the canonical projection of F onto F/W, the composite $\mathrm{pr}_W \circ f$ is differentiable at v_0 with surjective differential. For a given lattice H, there will be various choices of W to which Lemma 3.4 applies. For each such W,

$$f(v_0 + H) \subset f(v_0) + f'(v_0)(H) + W; \tag{3.3}$$

taking the intersection of the right hand side over many W yields an upper bound on $f(v_0 + H)$.

3.2. *The case of locally analytic functions*

In this section we make the constant δ in Lemma 3.4 explicit, under the additional assumption that f is locally analytic. We extend the definition of such functions from finite-dimensional K-vector spaces [16, §6] to K-Banach spaces.

Definition 3.5. *Let E and F be K-Banach spaces. Let U be an open subset of E and let $x \in U$. A function $f : U \to F$ is said to be* locally analytic *at x if there exists an open subset $U_x \subset E$ and continuous n-linear maps $L_n : E^n \to F$ for $n \geq 1$ such that*

$$f(x + h) = f(x) + \sum_{n \geq 1} L_n(h, \ldots, h)$$

for all h with $x + h \in U_x$.

Remark 3.6. *A function f which is locally analytic at x is* a fortiori *differentiable at x, with derivative given by L_1.*

For the rest of this section, we assume that K is algebraically closed. As in Definition 3.5, we consider two K-Banach spaces E and F and a family of continuous n-linear maps $L_n : E^n \to F$. For $n \geq 1$ and $h \in E$, we set $f_n(h) = L_n(h, \ldots, h)$ and

$$\|f_n\| = \sup_{h \in B_E(1)} \|f_n(h)\|.$$

When the series $\sum_n f_n(h)$ converges, we denote by $f(h)$ its sum; we shall write $f = \sum_{n\geq 0} f_n$. We assume that f is defined in a neighborhood of 0. Under this assumption, the datum of f uniquely determines the f_n's (a consequence of Proposition 3.9 below). To such a series f, we attach the function $\Lambda(f) : \mathbb{R} \cup \{+\infty\} \to \mathbb{R} \cup \{+\infty\}$ defined by:

$$\Lambda(f)(v) = \begin{cases} \log\Big(\sup_{h\in B_E^-(e^v)} \|f(h)\|\Big) & \text{if } f \text{ is defined on} B_E^-(e^v), \\ +\infty & \text{otherwise.} \end{cases}$$

The following lemma is easy and left to the reader.

Lemma 3.7. *Let $f = \sum_{n\geq 0} f_n$ and $g = \sum_{n\geq 0} g_n$ be two series as above. Then*

$$\Lambda(f+g) \leq \max(\Lambda(f), \Lambda(g)),$$
$$\Lambda(f\times g) \leq \Lambda(f) + \Lambda(g),$$
$$\Lambda(f\circ g) \leq \Lambda(f) \circ \Lambda(g).$$

Remark 3.8. *Using Lemma 3.7, one can easily derive an upper bound of $\Lambda(f)$ from a formula describing f.*

The function $\Lambda(f)$ we have just defined is closely related to the Newton polygon of f. Recall that the Newton polygon of f is the convex hull in $\mathbb{R}^2$ of the points $(n, -\log\|f_n\|)$ for $n \geq 0$, together with the extra point $(0, +\infty)$. We denote by $\mathrm{NP}(f) : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ the convex function whose epigraph is the Newton polygon of f.

We recall that the Legendre transform of a *convex* function $\varphi : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ is the function $\varphi^\star : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ defined by

$$\varphi^\star(v) = \sup_{u\in\mathbb{R}} \big(uv - \varphi(u)\big),$$

for $v \in \mathbb{R}$. One can check that the map $\varphi \mapsto \varphi^\star$ is an order-reversing involution: $(\varphi^\star)^\star = \varphi$ and $\varphi^\star \geq \psi^\star$ whenever $\varphi \leq \psi$. When necessary, we extend $\varphi^\star$ to $\mathbb{R} \cup \{+\infty\}$ by left continuity. We refer to [15] for a complete exposition on Legendre transforms.

Proposition 3.9. *Keeping the above notation, we have $\Lambda(f) = \mathrm{NP}(f)^\star$.*

Proof. Note that the functions $\Lambda(f)$ and $\mathrm{NP}(f)^\star$ are both left continuous. It is then enough to prove that they coincide expect possibly on the set of slopes of $\mathrm{NP}(f)$, a dense subset of $\mathbb{R}$.

Let $v \in \mathbb{R}$, not a slope of $\mathrm{NP}(f)$. We assume first that $\mathrm{NP}(f)^\star(v)$ is finite. We set $u = \mathrm{NP}(f)^*(v)$. The function $m \mapsto \mathrm{NP}(f)(m) - vm + u$ has the following properties:

(i) it is piecewise affine and everywhere nonnegative,
(ii) it does not admit 0 as a slope and
(iii) it vanishes at $x = n$ for some integer n and $u = vn + \log\|f_n\|$.

We deduce from these facts that there exists $c > 0$ such that

$$vm - u \leq -\log\|f_m\| - c\cdot|n-m|$$

for any $m \geq 0$. Since $vm - u = vm - vn - \log\|f_n\|$, we get

$$-vn - \log\|f_n\| + c\cdot|n-m| \leq -vm - \log\|f_m\|.$$

Therefore, for any $x \in B_E(e^v)$ and $m \geq 0$, we have

$$\|f_m(x)\| \leq e^{-c\cdot|n-m|}\cdot\|f_n\|\cdot e^{vn} \leq \|f_n\|\cdot e^{vn}.$$

Thus, the series $\sum_{m\geq 0} f_m(x)$ converges and $\|f(x)\| \leq \|f_n\| \cdot e^{vn}$. We then get

$$\Lambda(f)(v) \leq \log\big(\|f_n\| e^{vn}\big) = vn + \log\|f_n\| = u. \tag{3.4}$$

On the other hand, it follows from the definition of $\|f_n\|$ and the fact that $|K^\times|$ is dense in $\mathbb{R}$ (K is algebraically closed) that there exists a sequence $(x_i)_{i\geq 0}$ in $B_E^-(e^v)$ such that $\lim_{i\to\infty} \|f_n(x_i)\| = \|f_n\| \cdot e^{vn}$. Since $\|f_m(x_i)\| \leq e^{-c\cdot|n-m|} \cdot \|f_n\| \cdot e^{vn}$ for all m and i, we get $\|f_m(x_i)\| < \|f_n(x_i)\|$ for i large enough. For these i, we then have $\|f(x_i)\| = \|f_n(x_i)\|$. Passing to the limit on i, we find $\Lambda(f)(v) \geq u$. Comparing with (3.4), we get $\Lambda(f)(v) = u = \mathrm{NP}(f)^\star(v)$.

We now assume that $\mathrm{NP}(f)^\star(v) = +\infty$. The function $x \mapsto \mathrm{NP}(f)(x) - vx$ is then not bounded from below. Since it is convex, it goes to $-\infty$ when x goes to $+\infty$. By the definition of $\mathrm{NP}(f)$, the expression $vn + \log\|f_n\|$ goes to infinity as n grows. It is then enough to establish the following claim:

$$\forall n \in \mathbb{N}, \quad \Lambda(f)(v) \geq vn + \log\|f_n\| - \log 2. \tag{3.5}$$

Let n be a fixed integer. If $\|f_n\| = 0$, there is nothing to prove. Otherwise, we consider an element $x_n \in B_E^-(e^v)$ such that $\|f_n(x_n)\| \geq \frac{1}{2}\|f_n\| \cdot e^{vn}$. If the series $\sum_{m\geq 0} f_m(x_n)$ diverges, then $\Lambda(f)(v) = +\infty$ by definition and Eq. (3.5) holds. On the other hand, if it converges, the sequence $\|f_m(x_n)\|$ goes to 0 as m goes to infinity. Hence it takes its maximum value R a finite number of times; let us denote by $I \subset \mathbb{N}$ the set of the corresponding indices. For any $\lambda \in \mathcal{O}_K$, the series defining $f(\lambda x_n)$ converges and

$$f(\lambda x_n) \in B_F(R) \quad \text{and} \quad f(\lambda x_n) \equiv \sum_{m\in I} \lambda^m f_m(x_n) \pmod{B_F^-(R)}.$$

The quotient $B_F(R)/B_F^-(R)$ is a vector space over the residue field k of K. Since k is infinite, there must exist $\lambda \in \mathcal{O}_K$ such that $\sum_{m\in I} \lambda^m f_m(x_n)$ does not vanish in $B_F(R)/B_F^-(R)$. For such an element λ, we have $\|f(\lambda x_n)\| = R \geq \frac{1}{2}\|f_n\| \cdot e^{vn}$. The claim (3.5) follows. □

Remark 3.10. *It follows from Proposition 3.9 that $\Lambda(f)$ is a convex function.*

We now study the effect of truncation on series: given f as above and a nonnegative integer n_0, we set

$$f_{\geq n_0} = \sum_{n\geq n_0} f_n = f - (f_0 + f_1 + \cdots + f_{n_0-1}).$$

On the other hand, given a convex function $\varphi : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ and a real number v, we define $\varphi_{\geq v} : \mathbb{R} \to \mathbb{R} \cup \{\pm\infty\}$ by

$$\varphi_{\geq v}(x) = \inf_{y\geq 0} \big(\varphi(x+y) - vy\big).$$

The function $\varphi_{\geq v}$ is the maximum among all functions φ' with $\varphi' \leq \varphi$ and $x \mapsto \varphi'(x) - vx$ nondecreasing. When v is fixed, the construction $\varphi \mapsto \varphi_{\geq v}$ is nondecreasing: if φ and ψ are two convex functions such that $\varphi \leq \psi$ then $\varphi_{\geq v} \leq \psi_{\geq v}$.

Proposition 3.11. *With the above notations, we have $\Lambda(f_{\geq n_0}) \leq \Lambda(f)_{\geq n_0}$ for all $n_0 \in \mathbb{N}$.*

Proof. It follows easily from Proposition 3.9 and the fact that the slopes of the Legendre transform of a convex piecewise affine function f are exactly the abscissae of the points where f is not differentiable. □

We may now provide two sufficient conditions to effectively recognize first order lattices.

Proposition 3.12. *Let $f = \sum_{n \geq 0} f_n$ be a function as above. Let C be a positive real number satisfying $B_F(1) \subset f_1(B_E(C))$. Let $\rho \in (0,1]$ and ν be a real number such that*

$$\Lambda(f)_{\geq 2}(\nu) < \nu + \log\left(\frac{\rho}{C}\right). \tag{3.6}$$

Then the conclusion of Lemma 3.4 holds with $\delta = e^\nu$.

Remark 3.13. *On a neighborhood of $-\infty$, the function $x \mapsto \Lambda(f)_{\geq 2}(x) - x$ is affine with slope 1. This implies that, for all $\rho \in (0,1]$, there exists ν satisfying (3.6). Moreover, if ρ is close enough to 0, then one can take $\delta = e^\nu$ as a linear function of ρ.*

Remark 3.14. *In the statement of Proposition 3.12, one can of course replace the function $\Lambda(f)$ by any convex function φ with $\varphi \geq \Lambda(f)$. If f is given by some formula or some algorithm, such a function φ can be obtained using Remark 3.8.*

Proof. Pick ε in the interval $(e^{\Lambda(f)_{\geq 2}(\nu) - \nu}, \frac{\rho}{C})$. Going back to the proof of Lemma 3.4, we observe that it is enough to prove that

$$\|f_{\geq 2}(x)\| \leq \varepsilon \cdot \|x\|. \tag{3.7}$$

for all $x \in B_E(\delta)$. This inequality follows from Propositions 3.9 and 3.11 applied to the function $x \mapsto \frac{\Lambda_{\geq 2}(x)}{x}$. □

Remark 3.15. *It follows from the proof that Proposition 3.12 is still valid if K is not assumed to be algebraically closed. Indeed, the functions f_n — and then f also — extend to an algebraic closure $\bar{K}$ of K and (3.7) holds over $\bar{K}$, which is enough to conclude the result.*

Corollary 3.16. *We keep the notations of Proposition 3.12 and consider in addition a sequence $(M_n)_{n \geq 2}$ such that $\|f_n\| \leq M_n$ for all $n \geq 2$. Let $\mathrm{NP}(M_n)$ denote the convex function whose epigraph is the convex hull in $\mathbb{R}^2$ of the points of coordinates $(n, -\log M_n)$ for $n \geq 2$ together with the extra point $(0, +\infty)$.*

Let $\rho \in (0,1]$ and ν be a real number such that

$$NP(M_n)^\star(\nu) < \nu + \log\left(\frac{\rho}{C}\right).$$

Then the conclusion of Lemma 3.4 holds with $\delta = e^\nu$.

Remark 3.17. *If K has characteristic 0 and the vector spaces E and F are finite dimensional, then the M_n's defined by*

$$M_n = \frac{1}{|n!|} \cdot \sup_{\substack{1 \leq i \leq \dim E \\ |\underline{n}| = n}} \left\| \frac{\partial^n f_i}{\partial x^{\underline{n}}}(0) \right\|$$

do the job. Here f_i denotes the i-th coordinate of f, the notation $\underline{n}$ refers to a tuple of $(\dim F)$ nonnegative integers and $|\underline{n}|$ is the sum of the coordinates of $\underline{n}$.

4. *Precision in practice*

In this section we discuss applications of Lemma 3.4 and Proposition 3.12 to effective computations with p-adic numbers and power series.

4.1. *Optimal precision tracking*

We consider a function `f` (in the sense of computer science) that takes as input an approximate element lying in an open subset U of a K-Banach space E and outputs another approximate element lying in an open subset V of another K-Banach space F. In applications, this function models a continuous mathematical function $f : U \to V$: when `f` is called on the input $x + O(H)$, it outputs $x' + O(H')$ with $f(x + H) \subseteq x' + H'$. We say that `f` *preserves precision* if the above inclusion is an equality; it is often not the case as shown in Section 2.1.

Let us assume now that f is locally analytic on U and that $f'(x)$ is surjective. Proposition 3.12 then yields a rather simple sufficient condition to decide if a given lattice H is a first order lattice for f at x. For such a lattice, by definition, we have $f(x + H) = f(x) + f'(x)(H)$ and thus `f` must output $O(f'(x)(H))$ if it preserves precision. In this section we explain how, under the above hypothesis, one can implement the function `f` so that it always outputs the optimal precision.

One-pass computation. The execution of the function `f` yields a factorization:

$$f = f_n \circ f_{n-1} \circ \cdots \circ f_1$$

where the f_i's correspond to each individual basic step (like addition, multiplication or creation of variables); they are then "nice" (in particular locally analytic) functions. For all i, let U_i denote the codomain of f_i. Of course U_i must contains all possible values of all variables which are defined in the program after the execution of the i-th step. Mathematically, we assume that it is an open subset in some K-Banach space E_i. We have $U_n = V$ and the domain of f_i is U_{i-1} where, by convention, we have set $U_0 = U$. For all i, we set $g_i = f_i \circ \cdots \circ f_1$ and $x_i = g_i(x)$.

When we execute the function `f` on the input $x + O(H)$, we apply first f_1 to this input obtaining this way a first result $x_1 + O(H_1)$ and then go on with $f_2, \dots, f_n$. At each step, we obtain a new intermediate result that we denote by $x_i + O(H_i)$. A way to guarantee that precision is preserved is then to ensure $H_i = f'_i(x)(H_{i-1}) = g'_i(x)(H)$ at each step. This can be achieved by reimplementing all primitives (addition, multiplication, *etc.*) and make them compute at the same time the function f_i they implement together with its differential and apply the latter to the "current" lattice H_i.

There is nevertheless an important issue with this approach: in order to be sure that Lemma 3.4 applies, we need *a priori* to compute the exact values of all x_i's, which is of course not possible! Assuming that $g'_i(x)$ is surjective for all i, we can fix it as follows. For each i, we fix a first order lattice $\tilde{H}_i$ for g_i at x. Under our assumption, such lattices always exist and can be computed dynamically using Proposition 3.12 and Lemma 3.7 (see also Remark 3.8). Now, the equality $g_i(x + \tilde{H}_i) = x_i + g'_i(x)(\tilde{H}_i)$ means that any perturbation of x_i by an element in $g'_i(x)(\tilde{H}_i)$ is induced by a perturbation of x by an element in $\tilde{H}_i \subset H$. Hence, we can freely compute x_i modulo $g'_i(x)(\tilde{H}_i)$ without changing the final result. Since $g'_i(x)(\tilde{H}_i)$ is a lattice in E_i, this remark makes possible the computation of x_i.

Remark 4.1. *In some cases, it is actually possible to determine suitable lattices $\tilde{H}_i$ together with their images under $g'_i(x)$ (or, at least, good approximations of them) before starting the computation by using mathematical arguments. If possible, this generally helps a lot. We shall present in §4.3 an example of this.*

Two-pass computation. The previous approach works only if the $g'_i(x)$'s are all surjective. Unfortunately, this assumption is in general not fulfilled. Indeed, remember that the dimension of E_i is roughly the number of used variables after the step i. If all $g'_i(x)$ were surjective, this would mean that the function `f` never initializes a new variable! In what follows, we propose another solution that does not assume the surjectivity of $g'_i(x)$.

For $i \in \{1, \dots, n\}$, define $h_i = f_n \circ \cdots \circ f_{i+1}$, so that we have $f = h_i \circ g_i$. On differentials, we have $f'(x) = h'_i(x_i) \circ g'_i(x)$. Since $f'(x)$ is surjective (by assumption), we deduce that $h'_i(x_i)$ is surjective for all i. Let H'_i be a lattice in E_i such that:
(a) H'_i is contained in $H_i + \ker h'_i(x_i) = h'_i(x_i)^{-1}\big(f'(x)(H)\big)$;
(b) H'_i is a first order lattice for h_i at x_i.
By definition, we have $h_i(x_i + H'_i) = x_n + h'_i(x_i)(H'_i) \subset x_n + f'(x)(H)$. Therefore, modifying the intermediate value x_i by an element of H'_i after the i-th step of the execution of `f` leaves the final result unchanged. In other words, it is enough to compute x_i modulo H'_i.

It is nevertheless not obvious to implement these ideas in practice because when we enter in the i-th step of the execution of `f`, we have not computed h_i yet and hence are *a priori* not able to determine a lattice H'_i satisfying the axioms (a) and (b) above. A possible solution to tackle this problem is to proceed in several stages as follows:
(1) for i from 1 to n, we compute x_i, $f'_i(x_{i-1})$ at small precision (but enough for the second step) together with an upper bound of the function $\Lambda(h \mapsto f_i(x_{i-1}+h) - f_i(x_{i-1}))$;
(2) for i from n to 1, we compute $h'_i(x_i)$ and determine a lattice H'_i satisfying (a) and (b);
(3) for i from 1 to n, we recompute x_i modulo H'_i and finally outputs $x_n + O\big(f'(x)(H)\big)$.
Using relaxed algorithms for computing with elements in K (*cf* [2, 20, 21]), we can reuse in Step (3) the computations already performed in Step (1). The two-pass method we have just presented is then probably not much more expensive than the one-pass method, although it is more difficult to implement.

We conclude this section by remarking that the two-pass method seems to be particularly well suited to computations with lazy p-adics. In this setting, a target precision is fixed and the software determines automatically the precision it needs on the input to achieve this output precision. To do this, it first builds the "skeleton" of the computation (*i.e.* it determines the functions f_i and eventually computes the x_i at small precision when branching points occur and it needs to decide which branch it follows) and then runs over this skeleton in the reverse direction in order to determine (an upper bound of) the needed precision at each step.

Non-surjectivity. From the beginning, we have assumed that $f'(x)$ is surjective. Let us discuss shortly what happens when this assumption is relaxed. As it is explained after the proof of Lemma 3.4, the first thing we can do is to project the result onto different quotients, *i.e.* to work with the composites $\mathrm{pr}_W \circ f$ for a sufficiently large family of closed sub-vector-spaces $W \subset F$ such that $W + f'(x)(E) = F$. If F has a natural system of coordinates, we may generally take the pr_W's as the projections on each coordinate. Doing this, we end up with a precision on each individual coordinate. Furthermore, we have the guarantee that each coordinate-wise precision is sharp, even if the lattice built from them is not.

Let us illustrate the above discussion by an example: suppose that we want to compute the function $f : (K^n)^n \to M_n(K)$ that takes a family of n vectors to its Gram matrix. The differential of f is clearly never surjective because f takes its values in the subspace consisting of symmetric matrices. Nevertheless, for all pairs $(i,j) \in \{1,\dots,n\}^2$, one can consider the composite $f_{ij} = \mathrm{pr}_{ij} \circ f$ where $\mathrm{pr}_{ij} : M_n(K) \to K$ takes a matrix M to its (i,j)-th entry. The maps f_{ij}'s are differentiable and their differentials are generically surjective. Let M be a matrix known at some finite precision such that $f'_{ij}(M) \neq 0$ for all (i,j). We can then apply a one- or two-pass computation and get $f_{ij}(M)$ together with its precision. Putting this together, we get the whole matrix $f(M)$ together with a sharp precision datum on each entry.

The study of this example actually suggests another solution to tackle the issue of non-surjectivity. Indeed, we remark that our f above did not have a surjective differential simply because its codomain was too large: replacing the codomain with the K-vector space $S_n(K)$ of $n \times n$ symmetric matrices over K makes the differential of f surjective. While the image of a general f is rarely a sub vector space of F, it is often a sub-K-manifold of F. We can then use the results of Appendix A to study $f : U \to f(U)$, which likely has surjective differential.

Quick comparison with floating point arithmetic. The two strategies described above share some similarities with standard floating point arithmetic over the reals. In each setting, we begin by choosing a large precision for all computations, and when we encounter an unconstrained digit we choose it "at random" or using good heuristics. However, in the ultrametric setting, mild hypotheses allow us to quantify the precision needed at each individual step in order to ensure a specified final precision.

4.2. *Precision Types*

Using an arbitrary lattice to record the precision of an approximate element has the benefit of allowing computations to proceed without unnecessary precision loss using Lemma 3.4. However, while recording a lattice exactly is possible it does require a lot of space. For example, the space required to store a lattice precision for a single $n \times n$ matrix with entries of size $O(p^N)$ is $O(Nn^4 \cdot \log p)$. Conversely, the space needed to record that every entry has precision $O(p^N)$ is just $O(\log N)$.

Definition 4.2. *Suppose that E is a K-Banach space, and write $\mathrm{Lat}(E)$ for the set of lattices in E. A* precision type *for a K-Banach space E is a set $\mathcal{T} \subseteq \mathrm{Lat}(E)$ together with a function* $\mathrm{round} : \mathrm{Lat}(E) \to \mathcal{T}$ *such that*

($*$) *For every lattice $H \in \mathrm{Lat}(E)$, the lattice $\mathrm{round}(H)$ is a least upper bound for H under the inclusion order: $H \subseteq \mathrm{round}(H)$ and if $T \in \mathcal{T}$ satisfies $T \subset \mathrm{round}(H)$ then $H \not\subseteq T$.*

Different precision types are appropriate for different problems. For example, the final step of Kedlaya's algorithm for computing zeta functions of hyperelliptic curves [10, §4: Step 3] involves taking the characteristic polynomial of the matrix of Frobenius acting on a p-adic cohomology space. Obtaining extra precision on the entries of the matrix requires a long computation, so it is advisable to work with a precision type that does not round too much.

The following list gives examples of useful precision types. A description of the round function has been omitted for brevity.

- The *lattice* precision type has $\mathcal{T} = \mathrm{Lat}(E)$.
- In the *jagged* precision type, $\mathcal{T}$ consists of lattices of the shape $B_E((e_i),(r_i))$ for a fixed Banach basis (e_i) of E.
- In the *flat* precision type, $\mathcal{T}$ consists of lattices $B_E(r)$. The flat precision type is useful since it takes so little space to store and it easy to compute with.
- If $E = K_{<d}[X]$ is the space of polynomials of degree less than d, the *Newton* precision type consists of lattices $B_E((X^i),(r_i))$ where $-\log r_i$ is a convex function of i. The Newton precision type is sensible if one thinks of polynomials as functions $K \to K$, since extra precision above the Newton polygon never increases the precision of an evaluation.
- If $E = M_{m\times n}(K)$, the *column* precision type consists of lattices with identical image under all projections $\mathrm{pr}_i : E \to K^m$ sending a matrix to its ith column. It is appropriate when considering linear maps where the image of each basis vector has the same lattice precision.
- If $E = \mathbb{Q}_p[\![X]\!]$, the *Pollack-Stevens* precision type consists of lattices of the form $H_N := B_E((X^i),(p^{\min(i-N,0)}))$ [13, §1.5]. These lattices are stable under certain Hecke operators, which is necessary for computing with overconvergent modular symbols.

Note that sometimes the precision of a final result can be computed a priori (using the methods of Appendix B for example). Taking advantage of such knowledge can minimize artificial precision loss even when using rougher precision types such as flat or jagged.

```
Algorithm 1: SOMOS($a, b, c, d, n, N$)
Input: $a, b, c, d$ — four initial terms of a SOMOS 4 sequence $(u_n)_{n \geq 0}$
Input: $n, N$ — two integers
Assumption: $a$, $b$, $c$ and $d$ lie in $\mathbb{Z}_p^\times$ and are known at precision $O(p^N)$
Assumption: None of the $u_i$ $(0 \leq i \leq n)$ is divisible by $p^N$
Output: $u_n$ at precision $O(p^N)$

prec ← $N$;
for $i$ from 1 to $n - 3$ do
    prec ← prec + $v_p(bd + c^2)$;
    lift $b$, $c$ and $d$ arbitrarily to precision $O(p^{\mathrm{prec}})$;
    prec ← prec − 2 $v_p(a)$;
    $e \leftarrow \frac{bd+c^2}{a}$;        // $e$ is known at precision $O(p^{\mathrm{prec}})$
    $a, b, c, d \leftarrow b + O(p^{\mathrm{prec}}), c + O(p^{\mathrm{prec}}), d + O(p^{\mathrm{prec}}), e$;
return $d + O(p^N)$;
```

Separating precision from approximation also makes it much easier to implement algorithms capable of processing different precision types, since one can implement the arithmetic of the approximation separately from the logic handling the precision.

4.3. *Application to SOMOS sequence*

We illustrate the theory developed above by giving a simple toy application. Other applications will be discussed in subsequent articles. More precisely, we study the SOMOS 4 sequence introduced in §2.1. Making a crucial use of Lemma 3.4 and Proposition 3.12, we design a *stable* algorithm for computing it.

Recall from §2.1 that a SOMOS 4 sequence is a four-term inductive sequence defined by $u_n = \frac{u_{n-3}u_{n-1}+u_{n-2}^2}{u_{n-4}}$ exhibiting the Laurent phenomenon. We will focus on SOMOS sequences with values in $\mathbb{Q}_p$, and assume for simplicity that $u_0, u_1, u_2, u_3 \in \mathbb{Z}_p^\times$. By the Laurent phenomenon, $u_n \in \mathbb{Z}_p$ for all n, and if u_0, u_1, u_2, u_3 are known with finite precision $O(p^N)$ then all u_n are known with the same absolute precision. Algorithm 1 presented on page 14 performs this computation.

We now prove that it is correct. We introduce the function $f : \mathbb{Q}_p^\times \times \mathbb{Q}_p^3 \to \mathbb{Q}_p^4$ defined by $f(a, b, c, d) = (b, c, d, \frac{bd+c^2}{a})$. For all i, we have $(u_i, u_{i+1}, u_{i+2}, u_{i+3}) = f_i(u_0, u_1, u_2, u_3)$ where $f_i = f \circ \cdots \circ f$ (i times). Clearly, f is differentiable on $\mathbb{Q}_p^\times \times \mathbb{Q}_p^3$ and its differential in the canonical basis is given by the matrix:

$$D(a, b, c, d) = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{bd+c^2}{a^2} & \frac{d}{a} & \frac{2c}{a} & \frac{b}{a} \end{pmatrix}$$

whose determinant is $\frac{bd+c^2}{a^2}$. Thus, if the $(i+4)$-th term of the SOMOS sequence is defined, the mapping f_i is differentiable at (u_0, u_1, u_2, u_3) and its differential $\varphi_i = f'_i(u_0, u_1, u_2, u_3)$ is given by the matrix $D_i = D(u_{i-1}, u_i, u_{i+1}, u_{i+2}) \cdots D(u_1, u_2, u_3, u_4) \cdot D(u_0, u_1, u_2, u_3)$. Thanks to the Laurent phenomenon, we know that D_i has integral coefficients, *i.e.* φ_i stabilizes the lattice $\mathbb{Z}_p^4$. We are now going to prove by induction on i that, at the end of the i-th iteration of the loop, we have prec $= N + v_p(\det D_i)$ and

$$(a, b, c, d) \equiv (u_i, u_{i+1}, u_{i+2}, u_{i+3}) \pmod{p^N \varphi_i(\mathbb{Z}_p^4)}. \tag{4.1}$$

The first point is easy. Indeed, from $D_i = D(u_{i-1}, u_i, u_{i+1}, u_{i+2})D_{i-1}$, we deduce $\det D_i = \det D_{i-1} \cdot \frac{u_i}{u_{i-3}}$ and the assertion follows by taking determinants and using the induction hypothesis. Let us now establish (4.1). To avoid confusion, let us agree to denote by a', b', c', d' and prec' the values of a, b, c, d and prec respectively at the *beginning* of the i-th iteration of the loop. By induction hypothesis (or by initialization if $i = 1$), we have:

$$(a', b', c', d') \equiv (u_{i-1}, u_i, u_{i+1}, u_{i+2}) \pmod{p^N \varphi_{i-1}(\mathbb{Z}_p^4)}. \tag{4.2}$$

Moreover, we know that the determinant of φ_{i-1} has valuation prec'. Hence (4.2) remains true if a', b', c' and d' are replaced by other values which are congruent to them modulo $p^{\mathrm{prec}'}$. In particular it holds if a', b', c' and d' denotes the values of a, b, c and d after the execution of line 4. Applying Lemma 3.4 and Proposition 3.12 to φ_{i-1} and φ_i (at the point (u_0, u_1, u_2, u_3)), we get:

$$f\big((u_{i-1}, u_i, u_{i+1}, u_{i+2}) + p^N \varphi_{i-1}(\mathbb{Z}_p^4)\big) = (u_i, u_{i+1}, u_{i+2}, u_{i+3}) + p^N \varphi_i(\mathbb{Z}_p^4).$$

By the discussion above, this equation implies in particular that $f(a', b', c', d')$ belongs to $(u_i, u_{i+1}, u_{i+2}, u_{i+3}) + p^N \varphi_i(\mathbb{Z}_p^4)$. We conclude by remarking that $(a, b, c, d) \equiv f(a', b', c', d') \pmod{p^{\mathrm{prec}} \mathbb{Z}_p^4}$ by construction and that $p^{\mathrm{prec}} \mathbb{Z}_p^4 \subset p^N \varphi_i(\mathbb{Z}_p^4)$.

Finally (4.1) applied with $i = n - 3$ together with the fact that φ_i stabilizes $\mathbb{Z}_p^4$ imply that, when we exit the loop, the value of d is congruent to u_n modulo p^N. Hence, our algorithm returns the correct value.

We conclude this section by remarking that Algorithm 1 performs computations at precision at most $O(p^{N+v})$ where v is the maximum of the sum of the valuations of five consecutive terms among the first n terms of the SOMOS sequence we are considering. Experiments show that the value of v varies like $c \cdot \log n$ where c is some constant. Assuming that we are using a FFT-like algorithm to compute products of integers, the complexity of Algorithm 1 is then expected to be $\tilde{O}(Nn)$ where the notation $\tilde{O}$ means that we hide logarithmic factors.

We can compare this with the complexity of the more naive algorithm consisting of lifting the initial terms u_0, u_1, u_2, u_3 to enough precision and then doing the computation using a naive step-by-step tracking of precision. In this setting, the required original precision is $O(p^{N+v'})$ where v' is the sum of the valuation of the u_i's for i varying between 0 and n. Experiments show that v' is about $c' \cdot n \log n$ (where c' is a constant), which leads to a complexity in $\tilde{O}(Nn + n^2)$. Our approach is then interesting when n is large compared to N: under this hypothesis, it saves roughly a factor n.

Appendix A. *Generalization to manifolds*

Many natural p-adic objects do not lie in vector spaces: points in projective spaces or elliptic curves, subspaces of a fixed vector space (which lie in Grassmannians), classes of isomorphism of certain curves (which lie in various moduli spaces), *etc.* In this appendix we extend the formalism developed in Section 3 to a more general setting: we consider the quite general case of differentiable manifolds locally modeled on ultrametric Banach spaces. This covers all the aforementioned examples.

A.1. *Differentiable K-manifolds*

The theory of finite dimensional K-manifolds is presented for example in [16, Ch. 8-9]. In this section, we shall work with a slightly different notion of manifolds which allows also Banach vector spaces of infinite dimension. More precisely, for us, a *differentiable K-manifold* (or just *K-manifold* for short) is the data of a topological space V together with an open covering $V = \bigcup_{i \in I} V_i$ (where I is some set) and, for all $i \in I$, an homeomorphism $\varphi_i : V_i \to U_i$ where

U_i is an open subset of a K-Banach space E_i such that for all $i, j \in I$ for which $V_i \cap V_j$ is nonempty, the composite map

$$\psi_{ij} : \varphi_i(V_{ij}) \xrightarrow{\varphi_i^{-1}} V_{ij} \xrightarrow{\varphi_j} \varphi_j(V_{ij}) \quad (\text{with } V_{ij} = V_i \cap V_j) \tag{A.1}$$

is differentiable. We recall that the mappings φ_i above are the so-called *charts*. The ψ_{ij}'s are the transition maps. The collection of φ_i's and ψ_{ij}'s is called an *atlas* of V. In the sequel, we shall assume further that the open covering $V = \bigcup_{i \in I} V_i$ is locally finite, which means that every point $x \in V$ lies only in a finite number of V_i's. Trivial examples of K-manifolds are K-Banach spaces themselves.

If V is a K-manifold and x is a point of V, we define the *tangent space* $T_x V$ of V at x as the space E_i for some i such that $x \in V_i$. We note that if x belongs to V_i and V_j, the linear map $\psi'_{ij}(\varphi_i(x))$ defines an isomorphism between E_i and E_j. Furthermore these isomorphisms are compatible in an obvious way. This implies that the definition of $T_x V$ given above does not depend (up to some canonical isomorphism) on the index i such that $x \in V_i$ and then makes sense.

As usual, we can define the notion of differentiability (at some point) for a continuous mapping between two K-manifolds by viewing it through the charts. A differentiable map $f : V \to V'$ induces a linear map on tangent spaces $f'(x) : T_x V \to T_{f(x)} V'$ for all x in the domain V. It is called the *differential* of f at x.

A.2. *Precision data*

Returning to our problem of precision, given V a K-manifold as above, we would like to be able to deal with "approximations up to some precision" of elements in V, *i.e.* expressions of the form $x + O(H)$ where x belongs to a dense *computable* subset of V and H is a "precision datum". For now, we fix a K-manifold V and we use freely the notations I, V_i, φ_i, *etc.* introduced in §A.1.

Definition A.1. *Let $x \in V$. A* precision datum *at x is a lattice in the tangent space $T_x V$ such that for all indices i and j with $x \in U_i \cap U_j$, the image of $T_x V$ in E_i is a first order lattice for ψ_{ij} at $\varphi_i(x)$ (*cf *Definition 3.3).*

Remark A.2. *The definition of a precision datum at x depends not only on x and the manifold V where it lies but also on the chosen atlas that defines V.*

Lemma A.3. *Let $x \in V$ and H be a precision datum at x. The subset*

$$\varphi_i^{-1}\big(\varphi_i(x) + \varphi_i'(x)(H)\big) \subset V$$

does not depend on the index i such that $x \in V_i$.

Proof. Let i and j be two indices such that x belongs to V_i and V_j. Set $x_i = \varphi_i(x) \in E_i$ and $H_i = \varphi_i'(x)(H)$. The equality

$$\varphi_i^{-1}\big(\varphi_i(x) + \varphi_i'(x)(H)\big) = \varphi_j^{-1}\big(\varphi_j(x) + \varphi_j'(x)(H)\big)$$

is clearly equivalent to $\psi_{ij}(x_i + H_i) = \psi_{ij}(x_i) + \psi'_{ij}(x_i)(H_i)$ and the latter holds because H_i is a first order lattice for ψ_{ij} at x_i. □

We are now in position to define $x + O(H)$.

Definition A.4. *Let $x \in V$ and H be a precision datum at x. We set*

$$x + O(H) = \varphi_i^{-1}\big(\varphi_i(x) + \varphi_i'(x)(H)\big) \subset V$$

for some (equivalenty, all) i such that $x \in V_i$.

Change of base point. In order to restrict ourselves to elements x lying in a dense computable subset, we need to compare $x_0 + O(H_0)$ with varying $x + O(H)$ when x and x_0 are close enough. Let us first examine the situation in a fixed given chart: we fix some index $i \in I$ and pick two elements x_0 and x in V_i. We consider in addition a lattice $\tilde{H}_0$ in E_i — which should be think as $\varphi_i'(x_0)(H_0)$ — and we want to produce a lattice $\tilde{H}$ such that $\varphi_i(x_0) + \tilde{H}_0 = \varphi_i(x) + \tilde{H}$. Of course $\tilde{H} = \tilde{H}_0$ does the job as soon as $\varphi_i(x) - \varphi_i(x_0) \in \tilde{H}_0$. Now, we remark that the tangent spaces $T_{x_0}V$ and T_xV are both isomorphic to E_i via the maps $\varphi_i'(x_0)$ and $\varphi_i'(x)$ respectively. A natural candidate for H is then:

$$H = \left(\varphi_i'(x)^{-1} \circ \varphi_i'(x_0)\right)(H_0). \tag{A.2}$$

With this choice, $x + O(H) = x_0 + O(H_0)$ provided that x and x_0 are close enough in the following sense: the difference $\varphi_i(x) - \varphi_i(x_0)$ lies in the lattice $\varphi_i'(x_0)(H_0)$. We furthermore have a property of independence on i.

Proposition A.5. *Let $x_0 \in V$ and H_0 be a precision datum at x_0. Then, for all x sufficiently close to x_0,*
(i) the lattice H defined by (A.2) does not depend on i and is a precision datum at x, and
(ii) we have $x + O(H) = x_0 + O(H_0)$.

Proof. We first prove (i). For an index i such that $x, x_0 \in V_i$, let us denote by $f_i : T_{x_0}V \to T_xV$ the composite $\varphi_i'(x)^{-1} \circ \varphi_i'(x_0)$. Given an extra index j satisfying the same assumption, the difference $f_i - f_j$ goes to 0 when x converges to x_0 (see Remark 3.2). Since H_0 is open in $T_{x_0}V$, this implies that $(f_j - f_i)(H_0)$ contains $f_i(H_0)$ and $f_j(H_0)$ if x and x_0 are close enough. Now, pick $w \in f_j(H_0)$ and write it $w = f_j(v)$ with $v \in H_0$. Then w is equal to $f_i(v) + (f_j - f_i)(v)$ and thus belongs to $f_i(H_0)$ because each summand does. Therefore $f_j(H_0) \subset f_i(H_0)$. The inverse inclusion is proved in the same way. The fact that H is a precision datum at x is easy and left to the reader. Finally, if x is close enough to x_0, it is enough to check (ii) in the charts but this was already done. □

A.3. *Generalization of the main Lemma*

With above definitions, Lemma 3.4 extends to manifolds. To do so, we first need to define a norm on the tangent space T_xV (where V is some K-variety and x is a point in V). There is actually in general no canonical choice for this. Indeed, let us consider a K-manifold V covered by charts U_i's ($i \in I$) which are open subset of K-Banach spaces E_i's. If x is a point in V, the tangent space T_xV is by definition isomorphic to E_i for each index i such that $x \in V_i$. A natural norm on T_xV is then the one obtained by pulling back the norm on E_i. However, since the transition maps are not required to be isometries, this norm depends on the choice of i. They are nevertheless all equivalent because the transition maps are required to be continuous.

In the next lemma, we choose any of the above norms for T_xV.

Lemma A.6. *Let V and W be two K-manifolds. Suppose that we are given a differentiable function $f : V \to W$, together with a point $x \in V$ such that $f'(x) : T_xV \to T_{f(x)}W$ is surjective.*

Then, for all $\rho \in (0,1]$, there exists a positive real number δ such that, for all $r \in (0,\delta)$, any lattice H in T_xV such that $B^-_{T_xV}(\rho r) \subset H \subset B_{T_xV}(r)$ is a first order lattice for f at x.

Proof. Apply Lemma 3.4 in charts. □

Remark A.7. *The constant δ that appears in the lemma depends (up to some multiplicative constant) on the norm that we have chosen on T_xV. However, once this norm is fixed, and assuming further that V and W are locally analytic K-manifolds and the mapping f is locally analytic as well, the constant δ can be made explicit using the method of Section 3.2.*

A.4. *Examples*

We illustrate the theory developped above by some classical examples, namely elliptic curves and grassmannians.

Elliptic curves. In this example, we assume for simplicity that K does not have characteristic 2. Let a and b be two elements of K such that $4a^3 + 27b^2 \neq 0$ and let E be the subset of K^2 consisting of the pairs (x, y) satisfying the usual equation $y^2 = x^3 + ax + b$. Let $\mathrm{pr}_x : E \to K$ (resp. $\mathrm{pr}_y : E \to K$) denote the map that takes a pair (x, y) to x (resp. to y).

We first assume that a and b lie in the subring R of exact elements. For each point $P_0 = (x_0, y_0)$ on E except possibly a finite number of them, the map pr_x define a diffeomorphism from an open subset containing P_0 to an open subset of K; the same is true for pr_y. Moreover, around each $P_0 \in E$, at least one of these projections satisfies the above condition. Hence the maps pr_x and pr_y define together an atlas of E, giving E the structure of a K-manifold.

Let P_0 be a point in E around which pr_x and pr_y both define charts. Lemma A.3 then tells us that a precision datum on x determines a precision datum on y and vice versa. Indeed, in a neighborhood of P_0 we can write $y = \sqrt{x^3 + ax + b}$ (for some choice of square root) and find the precision on y from the precision on x using Lemma 3.4. We can go in the other direction as well by writing x locally as a function of y. A precision datum at P_0 is then nothing but a precision datum on the coordinate x or on the coordinate y, keeping in mind that each of them determines the other. Viewing a precision datum at P_0 as a lattice in the tangent space is a nice way to make it canonical but in practice we can just choose one coordinate and track precision only on this coordinate.

We conclude this example by showing a simple method to transform a precision datum on x to a precision datum on y and *vice versa*. Differentiating the equation of the elliptic curve, we get:

$$2y \cdot dy = (3x^2 + a) \cdot dx. \tag{A.3}$$

In the above dx and dy should be thought as a little perturbation of x and y respectively. Equation (A.3) then gives a linear relation between the precision on x (which is represented by dx) and those on y (which is represented by dy). This relation turns out to correspond exactly to the one which is given by Lemma 3.4.

Finally, consider the case where a and b are themselves given with finite precision and E is not fully determined. So we cannot consider it as a K-manifold and the above discussion does not apply readily. Nevertheless, we can always consider the submanifold of K^4 consisting of all tuples (a, b, x, y) satisfying $y^2 = x^3 + ax + b$. The projections on the hyperplanes $a = 0$, $b = 0$, $x = 0$ and $y = 0$ respectively define charts of this K-manifold. From this, we see that a precision datum on a point of the "not well determined" elliptic curve E is a precision datum on a tuple of three variables among a, b, x and y.

Grassmannians. Let d and n be two nonnegative integers such that $d \leq n$. The Grassmannian $\mathrm{Grass}(d,n)$ is the set of all sub-vector spaces of K^n of dimension d. It defines an algebraic variety over K and hence *a fortiori* a K-manifold. Concretely, a vector space $V \subset K^n$ of dimension d is given by a rectangular matrix $M \in M_{d,n}(K)$ whose rows form a basis of V and two such matrices M and M' define the same vector space if there exists $P \in \mathrm{GL}_d(K)$ such that $M = PM'$. Performing row echelon, we find that we can always choose the above matrix M in the particular shape:

$$M = \begin{pmatrix} I_d & N \end{pmatrix} \cdot P \tag{A.4}$$

where I_d denotes the $(d \times d)$ identity matrix, $N \in M_{d,n-d}(K)$ and P is a permutation matrix of size n. Moreover two such expressions with the same P necessarily coincide. Hence each permutation matrix P defines a chart $U_P \subset \mathrm{Grass}(d,n)$ which is canonically diffeomorphic to $M_{d,n-d}(K) \simeq K^{d(n-d)}$.

In other words, if V is a subspace of K^n of dimension d, we represent it as a matrix M of the shape (A.4) (using row echelon) and a precision datum at V is nothing but a precision datum on the matrix N. If we choose another permutation matrix to represent V, say P', we end up with another matrix N'; the matrices N and N' are then related by a simple relation. Differentiating it, we find a formula for translating the precision datum expressed in the chart U_P to the same precision datum expressed in the chart $U_{P'}$. Of course, in practice, when we are doing computations on subspaces of K^n (like sum or intersection), we represent the spaces in charts as above and perform all the calculations in these charts.

Appendix B. *Example: Matrices*

We saw in the core of the article that the differential of an operation encodes the intrinsic loss/gain of precision when performing this operation. In this appendix we compute the differential of various common operations on matrices. Surprisingly, we observe that all differentials we will consider are rather easy to compute even if the underlying operation is quite involved.

In what follows, we use freely the "method of physicists" to compute differentials: given a function f differentiable at some point x, we consider a small perturbation dx of x and write $f(x+dx) = y + dy$ by expanding LHS and neglecting terms of order 2. The differential of f at x is then the linear mapping $dx \mapsto dy$.

Determinants and characteristic polynomials. We first outline the standard computation of the differential of the function $\det : M_n(K) \to K$. Suppose that $M \in \mathrm{GL}_n(K)$ and that $\mathrm{Com}(M) = \det(M) M^{-1}$. Then

$$\begin{aligned} \det(M + dM) &= \det(M) \cdot \det(I + M^{-1} \cdot dM) \\ &= \det(M) \cdot \big(1 + \mathrm{Tr}(M^{-1} \cdot dM)\big) \\ &= \det(M) + \mathrm{Tr}(\mathrm{Com}(M) \cdot dM). \end{aligned}$$

The differential of det at M is then $dM \mapsto \mathrm{Tr}(\mathrm{Com}(M) \cdot dM)$. It turns out that this formula is still valid when M is not invertible. The same computation extends readily to characteristic polynomials, since they are defined as determinants. More precisely, let us consider the function $\chi : M_n(K) \to K_n[X]$ taking a matrix M to its monic characteristic polynomial $\det(X - M)$. Then χ is differentiable at each point $M \in M_n(K)$ and its differential is given by $dM \mapsto \mathrm{Tr}(\mathrm{Com}(X - M) \cdot dM)$.

LU factorization. Define the LU factorization of a square matrix $M \in M_n(K)$ as a decomposition $M = LU$ where L is lower triangular and unipotent and U is upper triangular.

Such a decomposition exists and is unique provided that no principal minor of M vanishes. We can then consider the mapping $M \mapsto (L, U)$ defined over the Zariski-open set of matrices satisfying the above condition. In order to differentiate it, we differentiate the relation $M = LU$ and rewrite the result as

$$L^{-1}dM\,U^{-1} = L^{-1} \cdot dL + dU \cdot U^{-1}.$$

We remark that in the right hand side of the above formula, the first summand is lower triangular with zero diagonal whereas the second summand is upper triangular. Hence in order to compute dL and dU, one can proceed as follows:

(1) compute the product $dX = L^{-1}dM\,U^{-1}$,
(2) separate the lower and upper part of dX obtaining $L^{-1} \cdot dL$ and $dU \cdot U^{-1}$
(3) recover dL and dU by multiplying the above matrices by L on the left and U on the right respectively.

The above discussion extends almost *verbatim* to LUP factorization; the only difference is that LUP factorizations are not unique but they are on a small neighborhood of M if we fix the matrix P.

QR factorization. A QR factorization of a square matrix $M \in M_n(K)$ will be a decomposition $M = QR$ where R is unipotent upper triangular and Q is orthogonal in the sense that ${}^{\mathrm{t}}Q \cdot Q$ is diagonal. As before, such a decomposition exists and is unique on a Zariski-open subset of $M_n(K)$. The mapping $f : M \mapsto (Q, R)$ is then well defined on this subset. We would like to emphasize at this point that the orthogonality condition defines a sub-manifold of $M_n(K)$ which is *not* a vector space: it is defined by equations of degree 2. The codomain of f is then also a manifold; this example then fits into the setting of Appendix A but not to those of Section 3. We can differentiate f by following the method used for LU factorization. Differentiating the relation $M = QR$, we obtain

$${}^{\mathrm{t}}Q \cdot dM \cdot R^{-1} = {}^{\mathrm{t}}Q \cdot dQ + \Delta \cdot dR \cdot R^{-1} \tag{B.1}$$

where $\Delta = {}^{\mathrm{t}}Q \cdot Q$ is a diagonal matrix by definition. Moreover by differentiating ${}^{\mathrm{t}}Q \cdot Q = \Delta$, we find that ${}^{\mathrm{t}}Q \cdot dQ$ can be written as the sum of an antisymmetric matrix and a diagonal one. Since moreover $dR \cdot R^{-1}$ is upper triangular with all diagonal entries equal to 0, we see that (B.1) is enough to compute dQ and dR from Q, R and dM.

References

[1] Christian Batut, Karim Belabas, Dominique Benardi, Henri Cohen, and Michel Olivier, *User's guide to PARI-GP*, 1985-2013.

[2] Jérémy Berthomieu, Joris van der Hoeven, and Grégoire Lecerf, *Relaxed algorithms for p-adic numbers*, J. Théorie des Nombres des Bordeaux **23** (2011), no. 3, 541–577.

[3] Wieb Bosma, John Cannon, and Catherine Payoust, *The Magma algebra system. I. The user language.*, J. Symbolic Comput. **24** (1997), no. 3-4, 235–265.

[4] Alin Bostan, Laureano González-Vega, Hervé Perdry, and Éric Schost, *From Newton sums to coefficients: complexity issues in characteristic p*, MEGA'05, 2005.

[5] Xavier Caruso, *Random matrices over a DVR and LU factorization*, 2012. arXiv:1212.0308.

[6] Xavier Caruso and David Lubicz, *Linear algebra over Zp[[u]] and related rings*, 2014. to appear in LMS J. Comp. and Math.

[7] Sergey Fomin and Andrei Zelevinsky, *The Laurent phenomenon*, Advances in Applied Math. **28** (2002), no. 2, 119–144.

[8] Pierrick Gaudry, Thomas Houtmann, Annegret Weng, Christophe Ritzenthaler, and David Kohel, *The 2-adic CM method for genus 2 curves with application to cryptography*, Asiacrypt 2006, 2006, pp. 114–129.

[9] Anatolii Karatsuba and Yuri Ofman, *Multiplication of many-digital numbers by automatic computers*, Proceedings of the USSR Academy of Sciences **145** (1962), 293–294.

[10] Kiran S. Kedlaya, *Counting points on hyperelliptic curves using monsky–washnitzer cohomology*, J. Ramanujan Math. Soc. **16** (2001), 323–338.

[11] Alan Lauder, *Deformation theory and the computation of zeta functions*, Proc. London Math. Soc. (3) **88** (2004), no. 3, 565–602. MR2044050 (2005g:11166)

[12] Reynald Lercier and Thomas Sirvent, *On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field*, J. Théorie des Nombres des Bordeaux **20** (2008), 783–797.

[13] Rob Pollack and Glenn Stevens, *Overconvergent modular symbols and p-adic L-functions*, Annales scientifiques de l'ENS **44** (2011), no. 1, 1–42.

[14] Michael O. Rabin, *Computable algebra, general theory and theory of computable fields*, Transactions of the AMS **95** (1960), 341–360.

[15] R. Tyrell Rockafellar, *Variational analysis*, Grundlehren der Mathematischen Wissenschaften 317, Springer-Verlag, 1997.

[16] Peter Schneider, *p-Adic Lie groups*, Grundlehren der mathematischen Wissenschaften 344, Springer, Berlin, 2011.

[17] Michael Somos, *Problem 1470*, Crux Mathematicorum **15** (1989), 208.

[18] William Stein et al, *Sage Mathematics Software*, The Sage Development Team, 2005-2013.

[19] Tristan Vaccon, *Matrix-F5 algorithms over finite-precision complete discrete valuation fields*, 2014. available at http://hal.archives-ouvertes.fr/hal-00951954.

[20] Joris van der Hoeven, *Relax, but don't be too lazy*, J. Symbolic Comput. **34** (2002), no. 6, 479–542.

[21] ———, *New algorithms for relaxed multiplication*, J. Symbolic Comput. **42** (2007), no. 8, 792–802.

Xavier Caruso and Tristan Vaccon
Institut de recheche en mathématiques de Rennes (IRMAR)
Université Rennes 1
Campus de Beaulieu
35042 Rennes Cedex
France

David Roe
Department of Mathematics
University of Calgary
2500 University Dr NW
Calgary AB T2N 1N4
Canada