Precision models for arithmetic in local fields

David Roe
(joint work with Xavier Caruso)

Department of Mathematics
University of Calgary

February 29, 2012

Outline

Notation

- $\mathcal{O}_K$ – complete discrete valuation ring. We will occasionally suppose that $\mathcal{O}_K$ has finite residue field.
- π – a uniformizer of $\mathcal{O}_K$.
- K – the field of fractions of $\mathcal{O}_K$.
- v – the valuation on K, normalized so that $v(\pi) = 1$.

In practice, K will contain a dense subring k whose elements are simpler to represent.

- If $K = \mathbb{Q}_p$ then we may choose $k = \mathbb{Q}$.
- If $K = \mathbb{F}_q[\![t]\!]$ then $k = \mathbb{F}_q(t)$ is dense in K.

Precision

Suppose that $a \in k$ is an approximation to $x \in K$. We say that a has *absolute precision* n if

$$v(x - a) \geq n.$$

We say that a has *relative precision* n if

$$v(x/a - 1) \geq n.$$

- If a is an approximation of x with absolute precision n and b of y with absolute precision m then $a + b$ is an approximation of $x + y$ with absolute precision $\min(n, m)$.
- If a is an approximation of x with relative precision n and b of y with relative precision m then ab is an approximation of xy with relative precision $\min(n, m)$.

An *approximate element* of K is a pair (a, m), where $a \in k$ and $m \in \mathbb{Z}$. We think of (a, m) as representing all possible elements $x \in K$ with $v(x - a) \geq m$, and write

$$a + O(\pi^m).$$

Geometrically, this approximate element is a ball of radius p^{-m} around a. While there is no distinguished center of such a ball, in practice we can fix for each precision m a set of distinguished elements of k that are inequivalent modulo π^m. For example, if $K = \mathbb{Q}_p$ and $k = \mathbb{Q}$, then we may choose

$$\{a/p^k : k \geq 0 \text{ and } 0 \leq a < p^{k+m}\}.$$

Vector Precision

Suppose now that we want to specify an element of K^m. We could represent such a vector as

- A list of n approximate elements of K,
- a ball of radius p^{-m} around an element of k^n,
- an element of k^n together with an $\mathcal{O}_K$-lattice $P \subset K^n$.

Each of the first two options is a special case of the third, where the lattice P is constrained to a certain shape.
It's worth noting that we can always choose a basis for P consisting of vectors in k^n, and thus P is determined exactly, without need for approximation.

Polynomial Precision

The space of polynomials of degree at most n is just K^{n+1}, so the notions of precision for vectors apply to polynomials as well. There are some specific lattice shapes that make sense for polynomials:

- Newton polygons. If we consider a polynomial as a function from K to K, then having lots of extra precision in an "interior" coefficient does not add to the precision of any evaluation. Moreover, one can determine the Newton polygon of a product easily from the slopes of the input polygons, simplifying computation of the precision.
- Lagrange precision. We can give the precision of $f(a_i)$ for some fixed set of $a_i \in K$.

Matrix Precision

Similarly, the space of $m \times n$ matrices is isomorphic to K^{mn}, so all the vector precisions apply.

- If we consider a matrix as representing a linear map $K^n \to K^m$ then the image vectors will be defined with some precision lattice in K^m. This yields a "column precision" on our matrix, where each column has the same precision.
- Similarly, we can consider a "row precision," where the rows of a matrix all have the same precision.

Separation of Precision from Approximation

The conceptual separation of precision from approximation is key:

- Suppose we want to use an algorithm like Karatsuba for multiplying polynomials, which uses the identity

$$(a_0 + a_1 x^n)(b_0 + b_1 x^n) =$$
$$(a_0 b_0) + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1)x^n + a_1 b_1 x^{2n}.$$

  If the precision of the coefficients vary, the result obtained by Karatsuba may have less precision than the result obtained by naive multiplication. If we compute an approximation and precision separately, we can use fast algorithms and still retain the desired level of precision.
- This separation also allows for a more modular implementation, allowing many different ways of handling precision without lots of code duplication.

Applying Functions to Precisions

Theorem

Suppose $x \in K^n$ and $f : K^n \to K^m$ is differentiable at x with surjective differential df_x. For any O_K-submodule $P \subset K^n$ there exists $r \in \mathbb{Z}$ with the following property. If $a \in K$ with $v(a) > r$ then

$$f(x + aP) = f(x) + a\, df_x(P).$$

Determinants

As an application, we determine the precision of the determinant of a matrix. Suppose A is an approximation matrix with precision lattice dA. The differential of the determinant at the identity is the trace function, so as long as dA is small enough, the precision of $\det(A)$ is given by

$$\text{trace}(\det(A)A^{-1}dA).$$

We can therefore compute the determinant of a p-adic matrix by computing the determinant of an appropriate approximation, and then computing the precision separately.

LU Decomposition

We can compute precisions for the LU decomposition similarly. Let $f : M_{n\times m}(K) \to M_n(K) \times M_m(K)$ map A to its LU decomposition (L, U). Since

$$A + dA = (L + dL)(U + dU),$$

we have

$$dA = dL \cdot U + L \cdot dU + \text{higher order terms.}$$

Since dL is lower triangular and dU is upper triangular, we can solve for them from dA, L and U.

Further applications

One can apply similar reasoning to determine precisions for evaluation of polynomials, Euclidean division, root finding and factorization, images and kernels, inverse matrices and characteristic polynomials.

Precision Tradeoffs

There are a spectrum of precision types to choose from for polynomials and matrices.

- On one end lies the "flat precision," specified by a single integer. Computations with flat precision tend to be quite simple, but one may have to sacrifice precision in a long computation.
- Conversely, working directly with $\mathcal{O}_K$-lattices offers the greatest flexibility and preservation of precision, but at the cost of expensive Hermite form computations, with running times on the order of $O(n^{2.3})$.
- In between lie other precision types such as Newton polygons for polynomials, which offer a compromise between speed and flexibility.

Questions?