Motivation
○○○○○○○○○

Precision types
○○○○○○○

Working with Precision
○○○○○

Numerical Methods
○○○○○

Numerical Methods in *p*-adic Linear Algebra

David Roe
(precision with Xavier Caruso & Tristan Vaccon)
(numerical methods with Kiran Kedlaya & Jen Balakrishnan)

Department of Mathematics
University of Calgary

PIMS Number Theory and Combinatorics Seminar
Lethbridge

Outline

Non-archimedian local fields

Let K be a finite extension of $\mathbb{Q}_p$ or $k((t))$ for some finite field k.
Let v be the valuation, $v(\pi) = 1$ and $O_K = \{x \in K : v(x) \geq 0\}$.
Let R be a dense subring of K, e.g. $\mathbb{Z}[\frac{1}{p}]$ or $k[t, t^{-1}]$.

We seek to compute with algebraic structures over K:

K^n – vectors of dimension n over K,

$M_n(K)$ – $n \times n$ matrices over K,

$K[x]$ – polynomials over K,

$K[\![x]\!]$ – power series over K,

as well as geometric ones, like points on varieties over K.

- We must perform computational tasks *quickly* and *stably*,
- and need to determine the precision of the result.

Precision for numbers

- A number $p^a u + O(p^N)$ has *absolute precision* N and *relative precision* $N - a$,
- analogous to the distinction between *fixed point* and *floating point* precision over $\mathbb{R}$.
- Over $\mathbb{R}$, precision is not normally tracked using intervals;
- over $\mathbb{Q}_p$ it is common to track precision on each operation.

$$\left(p^a u + O(p^N)\right) + \left(p^b v + O(p^M)\right) = p^a u + p^b v + O(p^{\min(N,M)})$$
$$\left(p^a u + O(p^N)\right) \cdot \left(p^b v + O(p^M)\right) = p^{a+b}\left(uv + O(p^{\min(N-a,M-b)})\right)$$

Precision loss in basic arithmetic

Precision loss occurs when addition and multiplication are mixed for elements with different valuations. For example, for odd p

$$\frac{\left(\left(1 + p^{99} + O(p^{100})\right) + \left(-1 + p^{99} + O(p^{100})\right)\right)}{p^{99} + O(p^{1000})} = 2 + O(p).$$

Unfortunately, mixing addition and multiplication is necessary for solving most problems.

Motivation
○○○●○○○○○

Precision types
○○○○○○○

Working with Precision
○○○○○

Numerical Methods
○○○○○

Precision for compound structures

Basic implementations of vectors, matrices, polynomials and
power series use a list of entries, each with its own precision.
I advocate a different representation. For example, for vectors:

Separate precision from approximation

A vector $v \in K^n$ is represented as pair $(\tilde{v}, P)$, where $\tilde{v} \in R^n$ and
$P \subset K^n$ is an O_K-submodule so that $v = \tilde{v} + h$ for some $h \in P$.

Motivation
0000●0000

Precision types
0000000

Working with Precision
00000

Numerical Methods
00000

Karatsuba

Consider the Karatsuba algorithm for multiplying polynomials. Given input polynomials of degree $2n$

$$F = F_0 + x^n F_1$$
$$G = G_0 + x^n G_1,$$

we set

$$H_0 = F_0 \cdot G_0$$
$$H_2 = F_1 \cdot G_1$$
$$H_1 = (F_0 + F_1) \cdot (G_0 + G_1) - H_0 - H_2.$$

Then

$$F \cdot G = H_0 + H_1 x^n + H_2 x^{2n}$$

but we have only used three polynomial multiplications of degree n rather than four, at the cost of two additions.

Numerical stability

Now suppose $F, G \in K[x]$ and we track precision for each operation. For example, if

$$F = (1 + O(p^4)) + (p^4 + O(p^8)) \cdot x,$$
$$G = (1 + O(p^4)) - (p^4 + O(p^8)) \cdot x,$$

we would set

$$H_1 = \big(1 + O(p^4) + p^4 + O(p^8)\big)\big(1 + O(p^4) - p^4 - O(p^8)\big)$$
$$- \big(1 + O(p^4)\big) + \big(p^8 + O(p^{12})\big)$$
$$= O(p^4).$$

But the actual coefficient of x in the product has more precision:

$$\big(1 + O(p^4)\big) \cdot \big(p^4 + O(p^8)\big) - \big(1 + O(p^4)\big) \cdot \big(p^4 + O(p^8)\big) = O(p^8).$$

Precision versus speed

We're thus presented with a choice:

- Use faster algorithms, but sacrifice precision on some (or all) inputs,
- Use naive algorithms in order to retain as much precision as possible, but sacrifice speed, making computations with large inputs infeasible.

Why we track precision

Recording the precision at each step of an algorithm imposes an overhead to computations with *p*-adics. We track precision for two main reasons:

- As a convenience to the user, so that they know the accuracy of the output of the algorithm.
- So that the algorithm can perform inexact operations (such as inversion, *p*-adic logarithms and exponentials) to an appropriate working precision so that the final answer has sufficient accuracy.

Motivation
○○○○○○○○○●

Precision types
○○○○○○○

Working with Precision
○○○○○

Numerical Methods
○○○○○

Separation

Separate the approximation from the precision

By separating the precision of compound *p*-adic data types from the approximation, algorithms can break the dichotomy of precision vs speed by using fast algorithms to compute an approximation and computing the precision of the result separately.

This separation is possible because precision in the non-archimedian world behaves far better than in archimedian computations: the ultrametric gives us far better precision control than the triangle inequality.

Imprecise elements

An *approximate element* of K is a pair (a, m), where $a \in R$ 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 R that are inequivalent modulo π^m. For example, if $K = \mathbb{Q}_p$ and $R = \mathbb{Z}[\frac{1}{p}]$, then we may choose

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

Different precision models

- For $\mathbb{Z}_p$ and $\mathbb{Q}_p$, Sage currently supports three different precision models (capped relative, capped absolute and fixed modulus). But in reality these different models correspond more closely to the underlying data structure rather than the precision tracking, since there's only one shape of disc in $\mathbb{Q}_p$.

- For more complicated structures such as vectors, matrices, polynomials and power series, there are many different precision shapes possible. Different precision shapes have different tradeoffs between speed and accuracy.

Motivation
000000000

Precision types
000●0000

Working with Precision
00000

Numerical Methods
00000

Vector Precision

An element of K^n could be represented as

- a list of n approximate elements of K (together with the specification of a distinguished basis),
- a ball of radius p^{-m} around an element of R^n,
- an element of R^n together with an O_K-lattice $P \subset K^n$.

Note that the third option generalizes the first two, and that we can always choose a basis for P consisting of vectors in R^n. In fact, P is determined exactly, without need for approximation.

Motivation
000000000

Precision types
0000000

Working with Precision
00000

Numerical Methods
00000

Polynomial Precision

If we fix the degree n then $\{1, x, \ldots, x^n\}$ provides a distinguished basis for the space of polynomials of degree n as a K vector space. Some additional precision shapes have particular utility 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 $a_i \in K$.

- A mixture, involving the specification of various derivatives at various points.

Matrix Precision

Similarly, the space of $m \times n$ matrices is isomorphic to K^{mn}, so precision types for vectors apply to matrices as well. We have extra precision shapes as well.

- 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.

Power series

- The space $K[\![x]\!]$ is no longer finite dimensional over K, so additional complications arise: you need to truncate series both in the x and p "variables." Precision types similar to those for finite dimensional vector spaces make sense however.

- Some precision structures for power series may specify an infinite Newton polygon symbolically. When this Newton polygon has positive slope in the limit then it allows a rigorous computation of the precision of power series evaluation. Information about all coefficients is necessary in order to ensure convergence, and needs to be stored separately from the finite list of approximate coefficients. A precision structure gives a natural place to reason with this information.

Precision on varieties

If V is a variety defined over K then points on V are another type of inexact object we might want to work with. The Grassmanian $G(m, n)$ of m-dimensional subspaces of K^n and an elliptic curve E defined over K provide illustrative examples.

- The Grassmanian $G(m, n)$ is covered by affine charts, each isomorphic to $K^{m(n-m)}$. To specify a point we need to specify a chart, and then a point in the relevant vector space.

- An elliptic curve, on the other hand, is usually given as a subvariety of projective space, and we specify points in projective space that are supposed to lie on the curve. We can give the precision of such an approximate point as a lattice in the tangent space.

Motivation
00000000

Precision types
0000000

Working with Precision
●0000

Numerical Methods
00000

Applying Functions to Precisions

Theorem

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

$$f(z + \alpha P) = f(z) + \alpha \, df_z(P).$$

Motivation
○○○○○○○○○

Precision types
○○○○○○○

Working with Precision
○●○○○○

Numerical Methods
○○○○○

Example: Karatsuba

Consider the Karatsuba example: set $\tilde{F} = 1 + p^4 x$, $\tilde{G} = 1 - p^4 x$
and

$$f : K^4 \rightarrow K^3$$
$$(a, b, c, d) \mapsto (ac, ad + bc, bd).$$

Its differential at the point $z = (1, p^4, 1, -p^4)$ is given by

$$\begin{pmatrix} c & 0 & a & 0 \\ d & c & b & a \\ 0 & d & 0 & b \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ -p^4 & 1 & p^4 & 1 \\ 0 & -p^4 & 0 & p^4 \end{pmatrix}.$$

Let $P = \langle (p^4, 0, 0, 0), (0, p^8, 0, 0), (0, 0, p^4, 0), (0, 0, 0, p^8) \rangle$.
Then $df_z(P) = \langle (p^4, -p^8, 0), (0, p^8, -p^{12}), (p^4, p^8, 0), (0, p^8, p^{12}) \rangle$.
When $p \neq 2$, $df_z(P) = \langle (p^4, 0, 0), (0, p^8, 0), (0, 0, p^{12}) \rangle$.

Motivation
○○○○○○○○○

Precision types
○○○○○○○

Working with Precision
○○●○○

Numerical Methods
○○○○○

Example: Determinants

We have

$$\det(A + dA) = \det(A) \cdot \det(I + A^{-1} \cdot dA)$$
$$= \det(A) \cdot (1 + \text{Tr}(A^{-1} \cdot dA))$$
$$= \det(A) + \text{Tr}(\text{Com}(A) \cdot dA)$$

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.

Motivation
000000000

Precision types
0000000

Working with Precision
00080

Numerical Methods
00000

Example: LU Decomposition

We can compute precisions for the LU decomposition similarly.
Let $f\colon M_{n\times n}(K) \to M_n(K) \times M_n(K)$ map A to its LU
decomposition (L, U), normalized so that U is unipotent. Since

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

we have

$$L^{-1} \cdot dA \cdot U^{-1} = L^{-1} \cdot dL + dU \cdot U^{-1} + \text{higher order terms.}$$

Since dL is lower triangular and dU is upper triangular (with
zero diagonal), we can solve for them from dA, L and U.

Motivation
000000000

Precision types
0000000

Working with Precision
00000●

Numerical Methods
00000

Further examples

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.

Implementing Algorithms

1. For a given function $f \colon K^n \to K^m$ find a method for determining r from $P \subset K^n$.

2. Given an approximate element $(x, P) \in K^n$, compute $df_x(P)$ (possibly "rounding" back to a lattice of the same precision type).

3. Compute $f(x)$ using enough fixed-point or floating-point precision.

Analogies with $\mathbb{R}$

In searching for algorithms for actually computing $f(x)$, it pays off to consider the notions of numerical stability used over $\mathbb{R}$. For example, in computing LU-decompositions, the same pivoting methods work to reduce precision loss. Unlike $\mathbb{R}$, the squares in K (analogues of the positive reals) are not closed under addition, and thus there is no usable notion of positive definite matrix. This lack hamstrings some of the most useful algorithms over $\mathbb{R}$, and we are still searching for replacements.
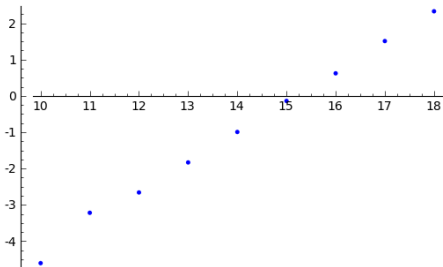
Characteristic polynomials in Magma

Disclaimer: I develop *p*-adics in Sage, an open source competitor to Magma. Moreover, I haven't run either of the following experiments personally, but rely on the experience of Kiran Kedlaya and Justin Walker. I give this example not to denigrate Magma but to show that handling precision appropriately is not easy.

Suppose you create a random 50×50 matrix *A* over $\mathbb{Z}_p$ in Magma 2.15 and ask for its characteristic polynomial. Even if the entries of *A* have precision 100, the resulting polynomial will have very few digits of precision remaining.

Motivation
○○○○○○○○○

Precision types
○○○○○○○

Working with Precision
○○○○○

Numerical Methods
○○○●○

Characteristic polynomials in Magma

In order to resolve this precision loss, Magma's characteristic polynomial algorithm appears to have changed in version 2.18. While the precision behavior improved dramatically, the runtime has regressed. In the following timing graph, the horizontal axis gives the size of the matrix and the vertical axis gives the log of the time to compute the characteristic polynomial.

Motivation
000000000

Precision types
0000000

Working with Precision
00000

Numerical Methods
0000●

Questions?