$p$-Adic Arithmetic in SAGE

David Roe

Department of Mathematics
Harvard University

Sage Days 5
1 Motivating Goals
   - Mission
   - Applications and Objects of Interest

2 Mathematical tools needed

3 Current Status and Priorities
Motivating Goals

- Mission
- Applications and Objects of Interest

Mathematical tools needed

Current Status and Priorities
1 Motivating Goals
   • Mission
   • Applications and Objects of Interest

2 Mathematical tools needed

3 Current Status and Priorities
Why we want $p$-adics in Sage

To support the advancement of mathematical knowledge by providing the facility to compute with mathematically interesting objects relying on $\mathbb{Q}_p$. 
We want to compute with many mathematical objects that rely on $\mathbb{Q}_p$:

- Spaces of $p$-adic modular forms and overconvergent modular forms.
- $p$-adic and $\ell$-adic Galois representations as part of a more general framework for Galois representations.
- $p$-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- $p$-adic $L$-functions, zeta functions, etc.
- $p$-adic analogues of trace-formulae.
- ???
We want to compute with many mathematical objects that rely on $\mathbb{Q}_p$:

- **Spaces of $p$-adic modular forms and overconvergent modular forms.**
- $p$-adic and $\ell$-adic Galois representations as part of a more general framework for Galois representations.
- $p$-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- $p$-adic L-functions, zeta functions, etc.
- $p$-adic analogues of trace-formulae.
- ???
Mathematical Objects

We want to compute with many mathematical objects that rely on \( \mathbb{Q}_p \):

- Spaces of \( p \)-adic modular forms and overconvergent modular forms.
- \( p \)-adic and \( \ell \)-adic Galois representations as part of a more general framework for Galois representations.
- \( p \)-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- \( p \)-adic L-functions, zeta functions, etc.
- \( p \)-adic analogues of trace-formulae.
- ???
Mathematical Objects

We want to compute with many mathematical objects that rely on $\mathbb{Q}_p$:

- Spaces of $p$-adic modular forms and overconvergent modular forms.
- $p$-adic and $\ell$-adic Galois representations as part of a more general framework for Galois representations.
- $p$-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- $p$-adic L-functions, zeta functions, etc.
- $p$-adic analogues of trace-formulae.

???
We want to compute with many mathematical objects that rely on $\mathbb{Q}_p$:

- Spaces of $p$-adic modular forms and overconvergent modular forms.
- $p$-adic and $\ell$-adic Galois representations as part of a more general framework for Galois representations.
- $p$-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- $p$-adic L-functions, zeta functions, etc.
- $p$-adic analogues of trace-formulae.
- ???
Mathematical Objects

We want to compute with many mathematical objects that rely on $\mathbb{Q}_p$:

- Spaces of $p$-adic modular forms and overconvergent modular forms.
- $p$-adic and $\ell$-adic Galois representations as part of a more general framework for Galois representations.
- $p$-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- $p$-adic L-functions, zeta functions, etc.
- $p$-adic analogues of trace-formulae.

David Roe  
$p$-Adic Arithmetic in SAGE
We want to compute with many mathematical objects that rely on $\mathbb{Q}_p$:

- Spaces of $p$-adic modular forms and overconvergent modular forms.
- $p$-adic and $\ell$-adic Galois representations as part of a more general framework for Galois representations.
- $p$-adic cohomology theories (crystalline, étale, Monsky-Washnitzer) as part of a more general framework for cohomology in Sage.
- $p$-adic L-functions, zeta functions, etc.
- $p$-adic analogues of trace-formulae.
Similarly, we can apply $p$-adics to a host of problems . . .

- $p$-adic heights for points on elliptic curves.
- Applications of $p$-adic differential equations.
- Applications of $p$-adics to quadratic forms (e.g. the Hasse principle)
- Applications of $p$-adics to linear algebra over global fields (e.g. Dixon’s algorithm)
- ???
Similarly, we can apply $p$-adics to a host of problems . . .

- $p$-adic heights for points on elliptic curves.
- Applications of $p$-adic differential equations.
- Applications of $p$-adics to quadratic forms (e.g. the Hasse principle)
- Applications of $p$-adics to linear algebra over global fields (e.g. Dixon’s algorithm)
- ???
Similarly, we can apply $p$-adics to a host of problems . . .

- $p$-adic heights for points on elliptic curves.
- Applications of $p$-adic differential equations.
- Applications of $p$-adics to quadratic forms (e.g. the Hasse principle)
- Applications of $p$-adics to linear algebra over global fields (e.g. Dixon’s algorithm)
- ???

David Roe

$p$-Adic Arithmetic in SAGE
Similarly, we can apply $p$-adics to a host of problems . . .

- $p$-adic heights for points on elliptic curves.
- Applications of $p$-adic differential equations.
- Applications of $p$-adics to quadratic forms (e.g. the Hasse principle)
  - Applications of $p$-adics to linear algebra over global fields (e.g. Dixon’s algorithm)
- ???
Similarly, we can apply \( p \)-adics to a host of problems . . .

- \( p \)-adic heights for points on elliptic curves.
- Applications of \( p \)-adic differential equations.
- Applications of \( p \)-adics to quadratic forms (e.g. the Hasse principle)
- Applications of \( p \)-adics to linear algebra over global fields (e.g. Dixon’s algorithm)
Similarly, we can apply $p$-adics to a host of problems . . .

- $p$-adic heights for points on elliptic curves.
- Applications of $p$-adic differential equations.
- Applications of $p$-adics to quadratic forms (e.g. the Hasse principle)
- Applications of $p$-adics to linear algebra over global fields (e.g. Dixon’s algorithm)
- ???
Currently, Sage does not support extensions of $\mathbb{Q}_p$. This will change soon.

- There will be special types for unramified extensions of $\mathbb{Q}_p$, eisenstein extensions of $\mathbb{Q}_p$, general absolute extensions and general relative extensions.
- A general extension will be converted transparently into a two step extension (first an unramified extension, then an eisenstein extension).
- Krasner’s lemma will be used to determine if the defining polynomial uniquely defines an extension.
- Eventually we will need a version of round 4 to split an arbitrary extension into unramified and totally ramified parts.
More on Extensions

- Underlying arithmetic will be done by the NTL classes ZZ_pE and ZZ_pEX.
- As with the base classes, different elements of the same ring or field can have different precision, necessitating casting during arithmetic.
- Elements of a general extension will cache their absolute and relative forms.
It should be trivial in Sage to obtain the completion of a number field $K$ at a given prime/place.

Extensions need to happen first.

There should be a type, “Completion of number field” that includes a local field and a map from the number field.
Polynomials

- There should be special classes to take advantage of fast NTL code.
- How should precisions of coefficients be restricted?
- One can write good algorithms in the mixed precision case.
- Need to support operations such as resultants that require linear algebra.
Linear algebra and modules

- We need to perform the standard linear algebra operations (determinants and traces, kernels, characteristic and minimal polynomials, etc). But these operations are harder in the $p$-adic case because of precision issues.
- As with polynomials, one can try to find an answer and find the precision separately (for some problems at least).
- As with polynomials, there is a question of how precision varies across a matrix, within a module or vector space.
- Algorithms need to be numerically stable.
Motivating Goals
Mathematical tools needed
Current Status and Priorities

Power series

• Much of the work on $p$-adics generalizes to power series.
• With power series over $p$-adics one can impose more complicated precision restrictions.
• It’s probably fast enough to keep implementing power series as a Sage polynomial and a precision.
• There is some demand for bidirectional power series, with conditions on the norms of the coefficients.
The base classes are generally in good shape, though the lazy classes need work.

Almost all the $p$-adic code suffers from lack of doctests.

The current code for the base classes is quite fast, comparable to Magma in arithmetic tests.
What is our plan for $p$-adics in Sage?