Improvements to Satoh's point counting algorithm

2026/01/15 — Atelier Pari/GP, Bordeaux

Damien Robert

Équipe Canari, Inria Bordeaux Sud-Ouest

A commit

```
$ git show --stat abdf3b3b
commit abdf3b3bac1cb05eb462296ad2a163af1ec6bb59
Author: Bill Allombert <Bill.Allombert@math.u-bordeaux.fr>
Date:   Tue Feb 9 17:20:07 2021 +0100

    FlxqE_ellcard: Implement Damien Robert variant of Satoh

 src/basemath/FlxqE.c | 106 +++++++++------------------------------------------
 1 file changed, 17 insertions(+), 89 deletions(-)
```

Timings

q	Time (old)	Memory (old)	Time (new)	Memory (new)
11^{1008}	48.5s	512MB	4.5s	128MB
101^{102}	91s	1024MB	9s	128MB
101^{256}	633s	4096MB	26s	128MB
101^{310}	924s	8192MB	35s	256MB
101^{418}	1813s	16384MB	55s	256MB

Timings for point counting on an elliptic curve $E/\mathbb{F}_q$ with Satoh's algorithm.

Point counting on an elliptic curve

- $E/\mathbb{F}_q, q = p^n$, an ordinary elliptic curve
- **Goal**: compute the action of π_q on some cohomology group $H^1(E, G)$ to recover

$$\chi_\pi(X) = X^2 - tX + q = (X - \lambda)(X - q/\lambda),$$

  hence $\#E = q + 1 - t$

- Example: the Frobenius π_q acts on the global differentials $H^0(E, \Omega^1_{E/\mathbb{F}_q}) \subset H^1_{dR}(E/\mathbb{F}_q)$
- $\pi_q^*(\omega_E) = c\omega_E$
- But π_q is inseparable, so $c = 0$…
- Instead, use the action of the Verschiebung $\hat{\pi}_q$ on global differentials
- This time we get the invertible eigenvalue $\lambda \mod p$

☹ Computing $\hat{\pi}_q$ using Vélu's formulas is $\Omega(q)$

☺ If p is small, we may compute the action of the small Verschiebung $\hat{\pi}_p : E \to \sigma^{-1}(E)$ instead: if $\hat{\pi}_p^*(\sigma^{-1}(\omega_E)) = \lambda'\omega_E$, then $\lambda = N_{\mathbb{F}_{q^n}/\mathbb{F}_p}(\lambda')$

☹ But if p is small, $\lambda \mod p$ is not enough to recover t: $|t| \leq 2\sqrt{q}$, so we need $\approx 1/2 \log(q) = n/2 \cdot \log(p)$ bits of precision

☺ [Satoh 2000]: compute the action of a lift of $\hat{\pi}_p$ to $\hat{E}/\mathbb{Z}_q$ the canonical lift of $E/\mathbb{F}_q$.

Satoh's algorithm

1. Compute equations for the canonical lift $\hat{E}/\mathbb{Z}_q$ to p-adic precision $m = n/2 \cdot \log(p) + 5$
2. Compute the action of a suitable lift V of $\hat{\pi}_p$ on differentials:

$$V^*(\sigma^{-1}(\omega_{\hat{E}})) = \lambda' \omega_{\hat{E}}$$

   σ is a generator of $\mathrm{Gal}(\mathbb{F}_q/\mathbb{F}_p) = \mathrm{Gal}(\mathbb{Z}_q/\mathbb{Z}_p)$.

3. Return $t = N_{\mathbb{Z}_q/\mathbb{Z}_p}(\lambda') + q/N_{\mathbb{Z}_q/\mathbb{Z}_p}(\lambda')$ as an integer.

Satoh's algorithm: Step 1

Compute $\hat{E}/\mathbb{Z}_q$ to p-adic precision m:

- Use Newton iterations on the equation
$$\Phi_p(j(\hat{E}), \sigma(j(\hat{E})) = 0,$$
where Φ_p is the modular polynomial
- We use Newton iterations inside the Newton iterations (Harley) to solve the sesquilinear equation:
$$e \cdot \frac{\partial \Phi_p}{\partial X}(j(\hat{E}), \sigma(j(\hat{E}))) + \sigma(e) \cdot \frac{\partial \Phi_p}{\partial Y}(j(\hat{E}), \sigma(j(\hat{E}))) = c$$
- σ is computed by picking up a Teichmüller representative $T_n(X)$ for $\mathbb{Z}_q = \mathbb{Z}_p[X]/T_n(X)$.
- The first version of Satoh would instead lift simultaneously the system of equations:
$$\Phi_p(j(\hat{E}_i), j(\hat{E}_{i+1})) = 0, \quad j(\hat{E}_n) = j(\hat{E}_0),$$

The Frobenius π_p is étale in $X_0(p)$, but not the Verschiebung: by Kronecker's congruence
$$\Phi_p = (X^p - Y)(X - Y^p) \mod p,$$
so if $n > 2$,
$$\frac{\partial \Phi_p}{\partial X}(j(\hat{E}), \sigma(j(\hat{E}))) = 0, \frac{\partial \Phi_p}{\partial Y}(j(\hat{E}), \sigma(j(\hat{E}))) \neq 0 \mod p$$

If $q = p^2$, $\Phi_p(X, Y)$ is not a good model of $X_0(p)$

Satoh's algorithm: Step 2

Compute the action of a suitable lift V of $\hat{\pi}_p$ on differentials:

- $\mathrm{Ker}\,\hat{\pi}_p = E[p](\overline{\mathbb{F}}_q) = \{h_P(x) = 0\}$ with $\deg h_P = \frac{p-1}{2}$.
  Explicitely, $\Psi_p = h_p^p$, where Ψ_p is the p-division polynomial,

$\hat{E}[p]$ is not étale over $\mathbb{Z}_q$, so the points in $\mathrm{Ker}\,\hat{\pi}_p$ do not lift uniquely to $\hat{E}[p]$.

- Let F be the lift of π_p to $\hat{E}$, and V its dual (so that $FV = [p]$, $VF = [p]$). Then V is a "canonical" lift of $\hat{\pi}_p$: it is the unique lift whose kernel points live in the unramified extension $\mathbb{Z}_q^{un}$ of $\mathbb{Z}_q$.
- If $\widetilde{E}/(\mathbb{Z}_q/p^2\mathbb{Z}_q)$ is a deformation of E, the points in $E[p](\overline{\mathbb{F}}_q)$ can lift to $\widetilde{E}(\mathbb{Z}_q^{un}/p^2\mathbb{Z}_q^{un})$ iff

$$\widetilde{E} \equiv \hat{E} \mod p^2.$$

- Once we have lifted to p^2, Newton iterations converge
- In practice we lift h_p to H_p rather than the kernel points individually: using the equation

$$H_p(X) \mid \Psi_{p,\widehat{E}}(X).$$

- We recover the kernel of V.
- We can then use Vélu's formulas to compute the action of V on differentials.

Complexity

Complexity to work in p-adic precision m:

- Step 1 is dominated by the evaluations of the modular polynomial Φ_p: $\widetilde{O}(mnp^2)$
- For Step 2, lifting h_P is dominated by the evaluations of the division polynomial Ψ_p: $\widetilde{O}(mnp^2)$. Computing V via Vélu's formula is in $\widetilde{O}(mnp)$.
- For Step 3, the norm can be done by a resultant in $\widetilde{O}(mnp)$.

Total cost for point-counting, with $m = O(n)$: $\widetilde{O}(n^2p^2)$

In practice, for small p, Step 2 is dominating.

Modular forms

- Once we have lifted h_p to H_p, Vélu's formula give a rational function

$$(x, y) \mapsto (r(x), cyr'(x))$$

  describing the isogeny $V : \hat{E} \to \sigma^{-1}(\hat{E})$.
- The action on differentials is given by c.

- The literature (and the previous Pari code) had a slightly different approach: use Vélu's formula to only compute the normalized Weierstrass equations of the codomain of V:

$$\hat{E}' : y^2 = x^3 + a'x + b'$$

- Then we just need to find the isomorphism $\gamma : E' \simeq \sigma^{-1}(\hat{E})$:
  γ gives the action on differentials u (up to a sign).
- Concretely, if $\hat{E} : y^2 = x^3 + ax + b$:

$$u^4 = \left(a'/\sigma^{-1}(a')\right)^4, \quad u^6 = \left(b'/\sigma^{-1}(b)\right)^6.$$

- This is because in a short Weierstrass equation $y^2 = x^3 + a_4x + a_6$, the coefficients a_4, a_6 are modular forms of Weight 4 and 6 respectively.
- Modular forms are modular invariants that can keep track of the action on differentials (formally: sections of a power of the Hodge line bundle).

Bypassing step 2

- Step 1 already involves the modular function j via the modular equation:

$$\Phi_p(j(\hat{E}), \sigma(j(\hat{E}))) = 0$$

- Unfortunately j is of weight 0.
- But it's q-derivative j' is of weight 2:

$$j' = -18\frac{a_6}{a_4}j$$

- The modular equation $\Phi_p(j_1, j_2) = 0$ also gives a modular equation on j':

$$j_1' \frac{\partial \Phi_p}{\partial X}(j_1, j_2) + j_2' \frac{\partial \Phi_p}{\partial Y}(j_1, j_2) = 0$$

  Which is similar to the equation used for Newton lifting!

- So the modular polynomial Φ_p alone is already enough to recover the action on differentials! (This of course was known since [Elkies 1992] but curiously was not used in Satoh's algorithm).

$$u^2 = -p \cdot \frac{\frac{\partial \Phi_p}{\partial X}(\sigma(j(\hat{E})), j(\hat{E}))}{\frac{\partial \Phi_p}{\partial Y}(\sigma(j(\hat{E})), j(\hat{E}))}$$

- So we can directly recover the action on differentials at the end of the Newton iteration in Step 1!

Bypassing step 2: the GP code (simplified)

```
Phi = polmodular(p);
dPhiX=deriv(Phi,x);
dPhiY=deriv(Phi,y);

m=ffinit(p,deg,'t)
M=polteichmuller(m,p,prec);
u=ffgen(m,'u)
// [...]

FindTrace(J)={
  K=subst(J,'t,'t^p)) // K=σ(J)
  num=substvec(dPhiX,[x,y],[K,J]);
  den=substvec(dPhiY,[x,y],[K,J]);
  u2=-p*num/den; return(u2);

  /* The square of the trace can then be computed via:
  norm=polresultant(liftall(u2),M);
  squared_trace = norm+q^2/norm+2*q;
  */
}
```

Bypassing step 2: the Pari code

```
static GEN
get_trace_Robert(GEN J, GEN phi, GEN Xm, GEN T, GEN q, ulong p, long e)
{
  GEN K = ZpXQ_frob(J, Xm, T, q, p);
  GEN Jp = FpXQ_powers(J, n, T, q);
  GEN Kp = FpXQ_powers(K, n, T, q);
  GEN Jd = FpXC_powderiv(Jp, q);
  GEN Kd = FpXC_powderiv(Kp, q);
  GEN Dx = FpM_FpXQV_bilinear(phi, Kd, Jp, T, q);
  GEN Dy = FpM_FpXQV_bilinear(phi, Kp, Jd, T, q);
  GEN C = ZpXQ_inv(ZX_divuexact(Dy, p), T, utoi(p), e);
  return FpX_neg(FpXQ_mul(Dx, C, T, q), q);
}
```

Point counting algorithms

Étale cohomology:

- [Schoof 1985]: $\widetilde{O}(n^5 \log^5 p)$
- [SEA 1992]: $\widetilde{O}(n^4 \log^4 p)$ (Heuristic)

Crystalline cohomology:

- [Satoh 2000] (canonical lifts of ordinary curves): $\widetilde{O}(n^2 p^2)$

Monsky-Washnitzer / Rigid cohomology:

- [Kedlaya 2001]: $\widetilde{O}(n^3 p)$
- [Harvey 2007]: $\widetilde{O}(n^{3.5} p^{1/2} + n^5 \log p)$

Bonus: improving the dependency in p

- Bypassing Step 2 gives huge speed ups in practice, but does not change the asymptotic complexity.
- Can we bypass Step 1 instead?
- Yes! $\widetilde{E}$ is the canonical lift $\hat{E}$ iff when we lift h_P to $\widetilde{E}$ and compute the associated isogeny, the codomain is isomorphic to $\sigma^{-1}(\widetilde{E})$.
- This gives an implicit modular equation, to which we can apply Newton iterations
- As written above, Step 2 involves Ψ_p and costs $O(nmp^2)$.
- But we can recover h_P in quasi-linear time by interpolation on the equation $\hat{\pi}_p \circ \pi_p = [p]$.
- And during the lift of h_P, to check that $H_P \mid \Psi_p$, we can just evaluate $\Psi_p \mod H_P$ directly by the double and add algorithm. This is also quasi-linear.
- Total complexity: $\widetilde{O}(n^2 p)$ [Maiga-R. 2021]

- We can do even better using the "new" fancy HD representations of isogenies.
- The HD representation of $\hat{\pi}_p$ allows for its evaluation in time $\mathrm{polylog}(p)$.
- This is compatible with lifting.
- Total complexity: $\widetilde{O}(n^2 \log^8 p + n \log^{11} p)$ [R. 2022]

Point counting algorithms, revisited

- [Schoof 1985]: $\widetilde{O}(n^5 \log^5 p)$
- [SEA 1992]: $\widetilde{O}(n^4 \log^4 p)$

- [Satoh 2000]: $\widetilde{O}(n^2 p^2)$
- [Maiga – R. 2021]: $\widetilde{O}(n^2 p)$
- [R. 2022]: $\widetilde{O}(n^2 \log^8 p + n \log^{11} p)$

- [Kedlaya 2001]: $\widetilde{O}(n^3 p)$
- [Harvey 2007]: $\widetilde{O}(n^{3.5} p^{1/2} + n^5 \log p)$