Computing modular polynomials modulo a generic prime

Joint work with Steven Galbraith while on a visit at the University of Auckland

Laurane Marco - March 11th 2025

EPFL LASEC

Modular polynomials

ℓ^{th} modular polynomial $\Phi_\ell(X, Y)$:
- bivariate polynomial
- integer polynomial
- degree $\ell + 1$ in both variables
- $\Phi_\ell(X, Y) = \Phi_\ell(Y, X)$

$$\Phi_\ell(j, j') = 0 \Leftrightarrow \exists \phi : E \to E' \text{ with } j(E) = j, j(E') = j', \deg(\phi) = \ell$$

What do they look like?

$$\Phi_2(X, Y) = X^3 - X^2Y^2 + 1488X^2Y - 162000X^2 + 1488XY^2 + 40773375XY + 8748000000X$$
$$+ Y^3 - 162000Y^2 + 8748000000Y - 157464000000000$$ [1]

Total size of ϕ_ℓ is $\ell^3 \log \ell$

[1] https://math.mit.edu/~drew/ClassicalModPolys.html

Laurane Marco

EPFL LASEC

Applications

Main application: Point counting on elliptic curves

But also: They pop up in many isogeny-based protocols (CRS key exchange, OSIDH)

Also used recently by Steven Galbraith in his paper *Climbing and descending tall volcanoes* (ANTS 2024)

Challenges

There are 4 main tasks when it comes to dealing with modular polynomials:

1. Computing $\Phi_\ell(X, Y)$ over the integers → best hope $O(\ell^3 \log \ell)$

2. Computing $\Phi_\ell(X, Y)$ modulo a chosen prime p

3. Computing $\Phi_\ell(X, Y)$ modulo a generic prime p → $O(\ell^2 \log p)$

4. Evaluating $\Phi_\ell(X, Y)$ → best hope $O(\ell \log p)$

Computing modular polynomials

Recall: They are big!

$\rightarrow \Phi_\ell$ is about $\ell^3 \log \ell$

Algorithm by Broker, Lauter and Sutherland to compute them for prime levels ℓ using **isogeny volcanoes.**

Blueprint:

1. Find a set of suitable primes p
2. Compute all $\Phi_\ell \mod p$
3. Apply the CRT to recover Φ_ℓ over $\mathbb{Z}$

Blueprint re-used in most (all?) subsequent works to compute modular polynomials

Modular polynomials via isogeny volcanoes, Broker R., Lauter K., Sutherland A.

Modular polynomials via isogeny volcanoes

Very high level: Map out some well-chosen ℓ-isogeny volcanoes for some finite set of prime p and collect the ℓ-isogeneous j-invariants. Interpolate.

Slightly less high level:

1. Choose an order $\mathcal{O}$ with big class number (to have a big enough surface on the volcano) and a *special* prime p to ensure depth 1.
2. Starting from a $j-$invariant j_0 on the surface, enumerate all of them using the action of $Cl(\mathcal{O})$
3. Compute the descending isogenies, and collect all the j-invariants on the floor using $Cl(\mathcal{O}')$
4. Once all the volcanoes that make up $Ell_{\mathcal{O}}(\mathbb{F}_p) \cup Ell'_{\mathcal{O}}(\mathbb{F}_p)$ have been mapped, interpolate $\phi_\ell(X, Y) \mod p$ using the j-invariants
5. Repeat for enough well-chosen p's and recover $\phi_\ell(X, Y)$ over $\mathbb{Z}$ using CRT

Modular polynomials via isogeny volcanoes, Broker R., Lauter K., Sutherland A.

Switching to supersingular elliptic curves

Leroux: Isogeny-based crypto has lots of tools for supersingular curves!

Idea:

1. Choose any prime p to have at least $\ell + 2$ supersingular curves over $\mathbb{F}_{p^2}$.
2. **Gather a « good » set of maximal orders:** Compute a set of $\ell + 1$ orders in $\mathcal{B}_{p,\infty}$. From each of these compute $\ell + 1$ left ideals of norm ℓ
→ Get $(\ell + 1)^2$ orders of different types
3. Start from curve E_0 with $End(E_0) \cong \mathcal{O}_0$. For each order $\mathcal{O}$ collected:
Compute $I(\mathcal{O}_0, \mathcal{O})$, compute an equivalent smooth ideal and translate it into an isogeny.
Collect the j-invariant of the image curve.
4. Interpolate $\phi_\ell(X, Y)$ modulo p
5. Repeat for enough well-chosen p's and recover $\phi_\ell(X, Y)$ over $\mathbb{Z}$ using CRT

Computation of Hilbert class polynomials and modular polynomials from supersingular elliptic curves, Leroux A.

Kunzweiler & Robert approach

Kunzweiler & Robert: How about the new HD toolbox?

Idea: use the HD representation and then use « higher order deformation » to compute $\phi_\ell(X, Y) \mod p$ for special p and then CRT to compute it over the integers

They also sketch a method to compute $\phi_\ell(X, Y) \mod p$ for **generic prime** p **!!!!**

Start from E_0 with known endomorphism ring

Use Clapotis to compute the $\ell + 1$ ℓ-isogenies (removes GRH heuristics) in $\tilde{O}((\log \ell + \log p)^{\mathfrak{O}})$ for some contant $\mathfrak{O}$ independent from p, ℓ

? Use deformation method again to recover the remaining j-invariants

Computing modular polynomials by deformation, Kunzweiler S., Robert D.

Using generic primes

In some applications we do not have any control over the prime.
What is the cost we pay on complexity?

Task: Compute $\Phi_\ell \mod p$ for any primes ℓ, p coprime to each other

Best we can hope for $\rightarrow O(\ell^2 \log p)$

Blueprint:

Start from some special chosen curve E_0

Compute $\ell + 1$ isogenies of degree ℓ and collect the j-invariants

Push the curves around to collect the remaining j-invariants

Interpolate to recover $\Phi_\ell \mod p$

Computing isogenies from the starting curve

Goal: From a given starting curve compute all outgoing ℓ-isogenies

Previously:
Compute a connecting ideal and then ideal-to-isogeny computation (Leroux)
Use Clapotis (Kunzweiler and Robert)

Idea:

Start from an elliptic curve E_0 of known endomorphism ring.

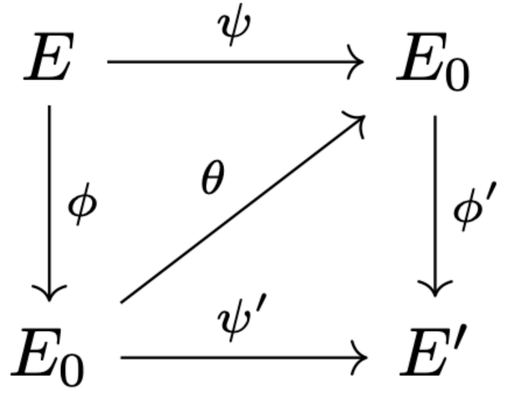
Use the technique introduced in QFESTA by Nakagawa and Onuki to compute the ℓ -isogenies by « factoring » endomorphism

Reminder: the QFESTA technique

E_0 of known endomorphism ring $\mathcal{O}_0$
p prime, D smooth that divides $p-1$ and $\ell < D$ an integer

Goal: Compute an isogeny $\phi : E_0 \to E$ of degree ℓ

1. Sample an endomorphism $\theta \in End(E_0)$ of norm $\ell(D-\ell)$ using FullRepresentInteger
2. Write $\theta = \psi \circ \phi : E_0 \to E_0$, with $\phi : E_0 \to E$ of degree ℓ and $\psi : E \to E_0$ of degree $(D-\ell)$
3. Construct a 2-dim isogeny Φ of degree D with $\ker(\Phi) = \{[\ell]P, \theta(P), P \in E_0(D)\}$
4. Evaluate ϕ as $\Phi(P,0)$



Using the QFESTA technique

In our setting:

1. Use QFESTA technique to compute endomorphisms of degree $\ell(D - \ell)$
2. Compute the codomain E of the ℓ-isogeny ϕ, and its j-invariant
3. Repeat until $\ell + 1$ different invariants have been collected

🙂 computing HD isogenies is pretty efficient these days
☹️ no real control over the endomorphisms generated
potentially a lot of redundancy

An observation

Let $\theta \in End(E_0)$, and $\mathcal{O}_0 = \langle \alpha_1, \alpha_2, \alpha_3, \alpha_4 \rangle$
Write $\theta = \lambda_1\alpha_1 + \lambda_2\alpha_2 + \lambda_3\alpha_3 + \lambda_4\alpha_4$.
Let $P, Q = E_0[\ell]$
Write A_i for the matrix of the action of α_i on P, Q

Let K be a kernel generator of ϕ.Then

$$\sum_{i=1}^{4} \lambda_i A_i(K) \equiv 0 \mod \ell$$

Can we just generate endomorphisms θ that verify this equation?

No :(Because we wouldn't have any control over their norms.

Can we control the endomorphisms better?

Can we incorporate both the equation

$$\sum_{i=1}^{4} \lambda_i A_i(K) \equiv 0 \mod \ell \quad (\star)$$

and the norm constraints i.e. $n(\theta) = \ell(D - \ell)$ together?

Generating endomorphisms of specific norm → RepresentInteger

What would it look like?
Start by sampling λ_3, λ_4 at random
Set $M = 4N - p(\lambda_3^2 + \lambda_4^2)$
~~Solve $x^2 + y^2 = M$ using Cornacchia's algorithm~~ → Incorporate ($\star$) and get a quartic form with **big coefficients**

→ hopeless :(

Can we control the endomorphisms better?

Can we use « simpler » endomorphisms and restrict ourselves to $\mathbb{Z}[i]$?

→ simplified form $\lambda_1 + i\lambda_2$, easier to control?

Choose D such that $\ell(D - \ell)$ is representable as a sum of two squares.
Compute all x, y such that $x^2 + y^2 = \ell(D - \ell)$

Observe $\ker(\phi) = E_0[\ell] \cap \ker(x + iy)$

We look at solutions of $x^2 + y^2 = 0 \mod \ell$ coming from the solutions of $x^2 + y^2 = \ell(D - \ell)$
We must have $(\frac{y}{x})^2 = -1 \mod \ell$
→ Only two possible values!
→ Independent of D

→ We can only ever hope two get two ℓ-isogenies ever

So, where do we stand?

Recall our overall strategy:

1. Compute endomorphisms of degree $\ell(D-\ell)$ that factor into an ℓ-isogeny
2. Compute the codomain E of the ℓ-isogeny ϕ, and its j-invariant
3. Repeat until $\ell+1$ different invariants have been collected

→ Use the condition $\sum_{i=1}^{4} \lambda_i A_i(K) \equiv 0 \mod \ell$ as fast check before execution Step **2.**

So, where do we stand?

1. Compute $\theta = \sum_{i=1}^{4} \lambda_i \alpha_i$ of degree $\ell(D - \ell)$
2. Compute $\ker(\sum_{i=1}^{4} \lambda_i A_i)$
3. Derive the kernel of the corresponding ℓ-isogeny.

4. If already seen, discard it, else compute the corresponding ℓ-isogeny, its image curve and the corresponding j-invariant

5. Repeat until $\ell + 1$ j-invariants have been collected

Collecting the remaining invariants

Current setting: A starting curve E_0 of j-invariant j_0, and a list of its $\ell + 1$ ℓ-isogeneous neighbours with associated j-invariants

→ Can interpolate $\Phi_\ell(j_0, Y)$ from its roots.

→ To recover $\Phi_\ell(X, Y)$ fully, need to repeat the collection step ℓ more times

Should we repeat our previous strategy?

☹ Need a curve of known endomorphism ring + some other costly stuff

Ideally we would want to re-use the work that's already been done

Collecting the remaining invariants

Strategy: « Push our information around »

Given E_0 and its ℓ-neighbours E_i.
Suppose we have $\langle P_0, Q_0 \rangle = E_0[d]$, for some d coprime to ℓ and P_i, Q_i its image on E_i through the ℓ-isogeny

1. From each of the E_i for $i = 0, \ell + 1$, compute all possible d-isogenies
2. Collect all the curves E_i' and their j-invariants

Repeat until all remaining j-invariants have been collected.

One can then get $\Phi_\ell(j_i, Y) \mod p$, for $i = 0, \ldots, \ell$ and then interpolate $\Phi_\ell(X, Y) \mod p$

EPFL LASEC

Summary of our algorithm

Given ℓ, p:

1. Pick a starting curve E_0 of known endomorphism ring $End(E_0) \cong \mathcal{O}_0$ and do the following:

a) Pick a powersmooth D and construct endomorphisms of degree $\ell(D - \ell)$ using FullrepresentInteger

b) Compute the corresponding kernels using our linear algebra trick.

If K has not been visited before, collect the corresponding endomorphism.

c) Repeat until $\ell + 1$ endomorphisms have been collected, and compute the j-invariants corresponding to the image curve of their ℓ-isogeny part using the high-dimensional isogeny computation.

2. Push the curves and j-invariants around

3. Use the j-invariants collected to interpolate $\Phi_\ell(X, Y) \mod p$

Complexity?

Given ℓ, p:

1. Pick a starting curve E_0 of known endomorphism ring $End(E_0) \cong \mathcal{O}_0$ and do the following:

a) Pick a powersmooth D and construct endomorphisms of degree $\ell(D - \ell)$ using FullrepresentInteger

Nothing new: FullRepresentInteger runs in $O(\text{poly} \log N)$ where N is the input norm.

We have $N = O(p)$ so the complexity is $O(\text{poly} \log p)$

Complexity?

1. b) Compute the corresponding kernels using our linear algebra trick.
If K has not been visited before, collect the corresponding endomorphism.

Three subtasks:

1. Computing a basis $P, Q$of the ℓ-torsion on E_0.
2. Computing the matrices corresponding to the action of the basis of $End(E_0)$ on the ℓ-torsion.
3. Computing the kernel of $\sum_{i=0}^{3} \lambda_i A_i$.

1. and **2.** are the most costly but they can be done once and for all per choice of ℓ

Complexity?

Three subtasks:

1. Computing a basis $P, Q$of the ℓ-torsion on E_0.
2. Computing the matrices corresponding to the action of the basis of $End(E_0)$ on the ℓ-torsion.
3. Computing the kernel of $\sum_{i=0}^{3} \lambda_i A_i$.

1. **Dominating cost** → the ℓ-torsion basis will be defined over a field extension F of size at most $\ell - 1$.
   Overall complexity: $(\log p + \log \ell)M(F)$
2. Solve a double-discrete logarithm, folklore in isogeny-based protocols → kinda efficient
3. Negligible: $O(n^3)$ matrix operations with $n = 2$ in our case

Complexity?

1. c) Repeat until $\ell + 1$ endomorphisms have been collected, and compute the j-invariants corresponding to the image curve of their ℓ-isogeny part using the high-dimensional isogeny computation.

Expected number of repetitions is $O(\ell \log \ell)$ to collect $\ell + 1$different endomorphisms

Cost of computing the $\ell + 1$ HD isogenies:

We compute a D isogeny, with $D = \sum_{i=1}^{k} p_i^{e_i}$ powersmooth → Decompose it

(Robert) → can be done in polynomial time, the exact complexity depends on where the points on $E[p_i^{e_i}]$ and $E[p_i^{e_i} p_j^{e_j}]$, and the biggest p_i.

Complexity?

2. Push the curves and j-invariants around

For each small prime degree d_i, compute all $d_i + 1$ outgoing isogenies for each of the $\ell + 2$ curves using Vélu's formulae. Repeat until $\prod_i d_i + 1 \geq \ell$.

Let d_{max} be the biggest degree, then the total complexity will be $\tilde{O}(\ell^2\sqrt{d_{max}})$ and $d_{max} = O(\log \ell \log\log \ell)$ (roughly the $\log \ell^{th}$ prime)

So total complexity is $O(\ell^2\sqrt{\log \ell \log\log \ell})$

Complexity?

2. Use the j-invariants collected to interpolate $\Phi_\ell(X, Y) \mod p$

Standard step: $O(\ell^2 \log \ell^{2+\epsilon} \log p^{1+\epsilon})$ (from BLS algorithm)

Overall?

Expensive **one-time** pre computation of the ℓ-torsion basis $O((\log^2 p + \log \ell)M(F))$ for an extension F of degree $\leq \ell + 1$

(and the smooth torsion basis for pushing curves around → negligible in comparison)

Collecting the right endomorphisms: $O(\ell \log \ell \text{poly} \log p)$

HD computation: $\tilde{O}((\ell + 1) \sum e_i \log p^2 D_1 B_1^{2u}))$

Pushing curves around: $O(\ell^2 \sqrt{\log \ell \log \log \ell})$

and interpolate.

Discussion

We are far from rigorous approach, there are heuristics hiding everywhere.

Hard to compare with the Kunzweiler and Robert approach as it is only sketched.

Maybe marginally better than Leroux

Played around with the implementation, but we're lacking the tools to compute the HD isogenies

Arithmetic over extension fields :(

Any insights are welcome :)

Thanks!