Fast Jacobian arithmetic for hyperelliptic curves of genus 3

Andrew V. Sutherland[1]

Massachusetts Institute of Technology

ANTS XIII — July 18, 2018

[1]Supported by NSF grant DMS-1522526 and Simons Foundation grant 550033.

Background

Let X be a nice (smooth, projective, geom. irred.) curve of genus g over a field k. Its Jacobian $\mathrm{Jac}(X)$ is an abelian variety of dimension g.

Suppose $X(k) \neq \emptyset$. Then there is a natural isomorphism

$$\mathrm{Jac}(X) \simeq \mathrm{Pic}^0(X),$$

where $\mathrm{Pic}^0(X) := \mathrm{Div}^0(X)/\,\mathrm{Princ}(X)$, and for any $O \in X(k)$ the map

$$\begin{aligned} X &\to \mathrm{Pic}^0(X) \\ P &\mapsto [P - O] \end{aligned}$$

is an injective morphism (an isomorphism when $g = 1$).

- When k is a number field $\mathrm{Jac}(X)$ is finitely generated.
- When k is a finite field $\mathrm{Jac}(X)$ is a finite abelian group.

Top ten reasons to care about $\mathrm{Jac}(X)$

1. Computing L-functions!
2. Computing zeta functions.
3. BSD conjecture for abelian varieties.
4. Galois representations.
5. Finding rational points with the Mordell-Weil sieve.
6. Cohen-Lenstra for function fields.
7. Lang-Trotter type questions.
8. Torsion subgroups.
9. Cryptographic applications.
10. Groups are more interesting than sets.

Computing L-functions.

Let $X/\mathbb{Q}$ be a nice curve of genus g.

$$L(X,s) := \prod_p L_p(p^{-s})^{-1},$$

For primes p of good reduction, $L_p \in \mathbb{Z}[T]$ is defined by

$$Z(X,T) := \exp\left(\sum_{n\geq 1} \#X(\mathbb{F}_{p^n})\frac{T^n}{n}\right) = \frac{L_p(T)}{(1-T)(1-pT)}.$$

For hyperelliptic X one can compute $L_p(T) \bmod p$ for all $p \leq B$ in $O(g^3 B(\log B)^{3+o(1)})$ time [Harvey 14, Harvey-S 14, Harvey-S 16].

For $g = 3$, one can lift $L_p(T) \bmod p$ to $L_p(T)$ in $O(p^{1/4+o(1)})$ time using computations in $\mathrm{Jac}(X)(\mathbb{F}_p)$ and $\mathrm{Jac}(\tilde{X})(\mathbb{F}_p)$ (assume $p \gg 1$).

For feasible B this is negligible, **provided Jacobian arithmetic is fast**.

Hyperelliptic curves

A hyperelliptic curve is a nice curve X/k of genus $g \geq 2$ that admits a degree-2 map $\phi\colon X \to \mathbf{P}^1$ (which we shall assume is defined over k). The hyperelliptic involution $P \mapsto \bar{P}$ interchanges points in each fiber.

Assume k is a perfect field of characteristic not 2. Then X has an affine model $y^2 = f(x)$, where $f \in k[x]$ is squarefree of degree $2g+2$ with roots corresponding to the Weierstrass points of X.

If X has a rational Weierstrass point P then by moving P to infinity we can obtain a model $y^2 = f(x)$ with f monic of degree $2g+1$.

This is typically not possible, in which case we are stuck with an even degree model $y^2 = f(x)$ which has either 0 or 2 points at infinity.

If X has a rational non-Weierstrass point, moving it to infinity will ensure that we are in the latter case (2 points at infinity).

Uniquely representing elements of $\mathrm{Pic}^0(X)$

A divisor is a finite formal sum $D := \sum n_P P$ of points $P \in X(\bar{k})$.
It is rational if it is fixed by $\mathrm{Gal}(\bar{k}/k)$ and effective if $n_P \geq 0$ for all P.
We may write effective divisors as $P_1 + \cdots + P_n$ (multiplicity allowed).

$P_1 + \cdots + P_n$ is semi-reduced if $P_i \neq \overline{P}_j$ for $i \neq j$, and reduced if $n \leq g$.

Theorem (Paulus-Ruck 99)

Let X be a hyperelliptic curve of genus g with an effective divisor D_∞ of degree g supported on rational points at infinity. Each element of $\mathrm{Pic}^0(X)$ can be written as $[D_0 - D_\infty]$, for a unique rational reduced divisor D_0 supported on affine points.

The Mumford representation $\mathrm{div}[u, v]$ of a rational semi-reduced affine divisor $D := P_1 + \cdots + P_n$ is the unique pair $u, v \in k[x]$ satisfying

$$u(x) := \prod (x - x(P_i)), \quad u | (f - v^2), \quad \deg v < \deg u.$$

The balanced divisor approach

We now recall the method of [GHM, ANTS VIII].

Let $X\colon y^2 = f(x)$ be a hyperelliptic curve of genus g with rational points $P_\infty := (1:1:0)$, $\overline{P}_\infty := (1:-1:0)$ at infinity; f monic, degree $2g+2$.
Let $D_\infty := \lceil \frac{g}{2} \rceil P_\infty + \lfloor \frac{g}{2} \rfloor \overline{P}_\infty$.

For $0 \le n \le g - \deg(u)$ define

$$\operatorname{div}[u,v,n] := \operatorname{div}[u,v] + nP_\infty + (g - \deg(u) - n)\overline{P}_\infty - D_\infty.$$

Each divisor class in $\operatorname{Pic}^0(X)$ is uniquely represented by $\operatorname{div}[u,v,n]$ for some monic $u|(f-v^2)$ with $\deg(v) < \deg(u) \le G$ and $0 \le n \le g - \deg(u)$.
The trivial element of $\operatorname{Pic}^0(X)$ is represented by $\operatorname{div}[1,0,\lceil \frac{g}{2} \rceil] = 0$.

As shown by Mireles Morales, this representation yields efficient addition formulas when g is even, and in particular, when $g = 2$.

Composing balanced divisors

Define $\operatorname{div}[u,v,n]^* := \operatorname{div}[u,v] + nP_\infty + (2g - \deg(u) - n)\overline{P}_\infty - 2D_\infty$.

Compose. Given $D_1 := \operatorname{div}[u_1,v_1,n_1]$ and $D_2 := \operatorname{div}[u_2,v_2,n_2]$:

1. Use the Euclidean algorithm to compute $w, c_1, c_2, c_3 \in k[x]$ so that
$$w = c_1u_1 + c_2u_2 + c_3(v_1+v_2) = \gcd(u_1,u_2,v_1+v_2).$$
2. Compute $u_3 := u_1u_2/w^2$, $n_3 := n_1 + n_2 + \deg(w)$, and
$$v_3 := (c_1u_1v_2 + c_2u_2v_1 + c_3(v_1v_2+f))/w \bmod u_3.$$
3. Output $D_3 := \operatorname{div}[u_3,v_3,n_3]^* \sim D_1 + D_2$.

Note that D_3 is not the canonical representative for $[D_1+D_2]$.

Reducing and adjusting divisors

Reduce. Given $\mathrm{div}[u_1, v_1, n_1]^*$ with $\deg(u_1) > g + 1$:

1. Let $u_2 := (f - v_1^2)/u_1$ made monic and $v_2 := -v_1 \bmod u_2$.
2. If $\deg(v_1) = g + 1$ and $\mathrm{lc}(v_1) = \pm 1$ then let $\delta := \mp(g + 1 - \deg(u_2))$, otherwise let $\delta := (\deg(u_1) - \deg(u_2))/2$.
3. Output $\mathrm{div}[u_2, v_2, n_1 + \delta]^* \sim \mathrm{div}[u_1, v_1, n_1]^*$.

Adjust. Given $\mathrm{div}[u_1, v_1, n_1]^*$ with $\deg(u_1) \leq g + 1$:

1. If $\lceil \frac{g}{2} \rceil \leq n_1 \leq \lceil \frac{3g}{2} \rceil - \deg(u_1)$ output $\mathrm{div}[u_1, v_1, n_1 - \lceil \frac{g}{2} \rceil]$ and stop.
2. If $n_1 < \lceil \frac{g}{2} \rceil$ let $\delta = -1$, otherwise, let $\delta = +1$.
3. Let $\hat{v}_1 := v_1 + \delta(V - (V \bmod u_1))$ and $u_2 := (f - \hat{v}_1^2)/u_1$ made monic, and $v_2 := -\hat{v}_1 \bmod u_s$ (using precomputed V with $\deg(f - V^2) \leq g$).
4. Let $n_2 := n_1 + \delta(\deg(u_i) - (g + 1))$, where $i = (3 - \delta)/2$.
5. Output **Adjust**($\mathrm{div}[u_2, v_2, n_2]^*$)

Addition and negation

Addition. Given $D_1 := \mathrm{div}[u_1, v_1, n_1]$ and $D_2 := \mathrm{div}[u_2, v_2, n_2]$:

1. Set $\mathrm{div}[u, v, n]^* \leftarrow$ **Compose**$(\mathrm{div}[u_1, v_1, n_1], \mathrm{div}[u_2, v_2, n_2])$.
2. While $\deg(u) > g + 1$ set $[u, v, n]^* \leftarrow$ **Reduce**$(\mathrm{div}[u, v, n]^*)$.
3. Output $D_3 :=$ **Adjust**$(\mathrm{div}[u, v, n]^*) \sim D_1 + D_2$.

The output divisor D_3 is the canonical representative for $[D_1 + D_2]$.

Negation. Given $D_1 := \mathrm{div}[u_1, v_1, n_1]$:

1. If g is even output $\mathrm{div}[u_1, -v_1, g - \deg(u_1) - n_1]$ and stop.
2. If $n_1 > 0$ output $\mathrm{div}[u_1, -v_1, g - \deg(u_1) - n_1 + 1]$ and stop.
3. Output $D_2 :=$**Adjust**$(\mathrm{div}[u_1, -v_1, \lceil \frac{3g}{2} \rceil - \deg(u_1) + 1]^*) \sim -D_1$.

The output divisor D_2 is the canonical representative for $[-D_1]$.

For even g this is essentially Cantor's algorithm, except $\deg(f) = 2g + 2$.

Addition in the typical case.

Generically, we expect the following to hold when adding divisors:

- $\deg(u_1) = \deg(u_2) = g$, $\deg(v_1) = \deg(v_2) = g - 1$, and $n_1 = n_2 = 0$;
- After **Compose**, $\deg(u) = 2g$, $\deg(v) = 2g - 1$, and $n = 0$.
- Each call to **Reduce** decreases $\deg(u)$ by 2 and increases n by 1.
  When g is even we will have $\deg(u) = g$ after $g/2$ calls to **Reduce**.
  When g is odd we will have $\deg(u) = g + 1$ after $(g - 1)/2$ calls.
- When g is even **Adjust** simply sets $n = 0$ and returns.
  When g is odd, **Adjust** first makes $\deg(u) = g$ and $n = (g + 1)/2$,
  then simply sets $n = 0$ and returns.

When $g = 3$, one call to **Reduce** and one nontrivial call to **Adjust**.

Straight-line program for the typical case

Standard optimizations (following [Gaudry-Harley, Harley 00]):

- Use the CRT to avoid computing GCDs (for $u_1 \perp u_2$ or $u_1 \perp v_1$).
- Combine composition and one reduction into a single step.

Optimization specific to balanced divisor approach:

- Combine composition, reduction, adjustment into a single step.

TypicalAddition. Given $\mathrm{div}[u_i, v_i, 0]$, with $\deg(u_i) = 3$ and $u_1 \perp u_2$:

1. $w := (f - v_1^2)/u$ and $\tilde{s} := (v_2 - v_1)/u_1 \bmod u_2$.
2. $c := 1/\mathrm{lc}(\tilde{s})$ and $s = c\tilde{s}$ and $z := su_1$ (require $\deg(s) = 2$).
3. $u_4 := (s(z + 2cv_1) - c^2 w)/u_2$ and $\tilde{v}_4 := v_1 + u_4 + (z \bmod u_4)/c$.
4. $u_5 := (\tilde{v}_4^2 - f)/(2\tilde{v}_{43} u_4)$ and $v_5 := \tilde{v}_4 \bmod u_5$ and $n_5 := 3 - \deg(u_5)$.

We then have $\mathrm{div}[u_1, v_1, 0] + \mathrm{div}[u_2, v_2, 0] \sim \mathrm{div}[u_5, v_5, n_5]$.
$\mathrm{div}[u_5, v_5, n_5]$ is the canonical representative of its divisor class.

Optimizations and results

Standard tricks that can be used to optimize the algorithm:

1. Karatsuba and Toom style polynomial multiplication;
2. Fast algorithms for exact division of polynomials;
3. Bezout's matrix for computing resultants;
4. Montgomery's trick for combining field inversions;
5. Maximize parallelism and minimize modular reductions.

After applying these optimizations (and other minor tweaks):

- Typical addition: $\mathbf{I} + 79\mathbf{M} + 127\mathbf{A}$ (vs $5\mathbf{I} + 275\mathbf{M} + 246\mathbf{A}$).
- Typical doubling: $\mathbf{I} + 82\mathbf{M} + 127\mathbf{A}$ (vs $5\mathbf{I} + 285\mathbf{M} + 258\mathbf{A}$).
- Typical negation: $\mathbf{I} + 14\mathbf{M} + 24\mathbf{A}$.

Note that (5) has no impact on the field operation counts.

Caveat: field operation counts can be misleading

For an odd prime p, consider the following computations in $\mathbb{F}_p$:

1. $z \leftarrow x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4$ (4**M**+3**A**)
2. $z \leftarrow (((x^2)^2)^2)^2$ (4**M**, in fact 4**S**)

Which is faster?

In almost any implementation (1) will take much less time than (2).
For word-sized operands on a Haswell core, (2) is $4\times$ slower than (1).

How about

1. $z \leftarrow x_1y_1 + x_1y_2 + x_2y_1 + x_2y_2$ (4**M**+3**A**)
2. $z \leftarrow (x_1 + x_2)(y_1 + y_2)$ (1**M**+2**A**)

Which is faster?

Comparing operation counts (with caveats)

Operation counts for Jacobian arithmetic on hyperelliptic curves over fields of odd characteristic using affine coordinates:

	Addition	Doubling	Source
Genus 2 odd degree	$\mathbf{I} + 24\mathbf{M}$	$\mathbf{I} + 28\mathbf{M}$	[Lange 05]
Genus 2 even degree	$\mathbf{I} + 28\mathbf{M}$	$\mathbf{I} + 32\mathbf{M}$	[GHM 08]
Genus 3 odd degree	$\mathbf{I} + 67\mathbf{M}$	$\mathbf{I} + 68\mathbf{M}$	[NMCT 06]
Genus 3 even degree	$\mathbf{I} + 79\mathbf{M}$	$\mathbf{I} + 82\mathbf{M}$	[this work]

Genus 3 even degree	$\mathbf{I} + 75\mathbf{M}$	$\mathbf{I} + 86\mathbf{M}$	[Rezai Rad 16]