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 $\text{Jac}(X)$ is an abelian variety of dimension $g$.

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

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

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

$$X \to \text{Pic}^0(X)$$

$$P \mapsto [P - O]$$

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

- When $k$ is a number field $\text{Jac}(X)$ is finitely generated.
- When $k$ is a finite field $\text{Jac}(X)$ is a finite abelian group.
Top ten reasons to care about $\text{Jac}(X)$

1. Computing $L$-functions!
2. Computing zeta functions.
3. BSD conjecture for abelian varieties.
5. Finding rational points with the Mordell-Weil sieve.
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

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

For hyperelliptic \(X\) one can compute \(L_p(T) \mod p\) 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 \(L_p(T) \mod p\) to \(L_p(T)\) in \(O(p^{1/4+o(1)})\) time using computations in \(\text{Jac}(X)(\mathbb{F}_p)\) and \(\text{Jac}(\tilde{X})(\mathbb{F}_p)\) (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 $\phi : X \to \mathbb{P}^1$ (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 $\text{Pic}^0(X)$

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 $\text{Gal}(\bar{k}/k)$ 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 $P_i \neq P_j$ 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 $D_\infty$ of degree $g$ supported on rational points at infinity. Each element of $\text{Pic}^0(X)$ can be written as $[D_0 - D_\infty]$, for a unique rational reduced divisor $D_0$ supported on affine points.

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

$$u(x) := \prod (x - x(P_i)), \quad u|(f - v^2), \quad \text{deg } v < \text{deg } u.$$
The balanced divisor approach

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

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

For \( 0 \leq n \leq g - \deg(u) \) define

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

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

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

**Compose.** Given \( D_1 := \text{div}[u_1, v_1, n_1] \) and \( D_2 := \text{div}[u_2, v_2, n_2] \):

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
   \[
   v_3 := (c_1u_1v_2 + c_2u_2v_1 + c_3(v_1v_2 + f))/w \mod u_3.
   \]

3. Output \( D_3 := \text{div}[u_3, v_3, n_3]^* \sim D_1 + D_2 \).

Note that \( D_3 \) is **not** the canonical representative for \( [D_1 + D_2] \).
Reducing and adjusting divisors

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

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

Adjust. Given \( \text{div}[u_1, v_1, n_1]^* \) 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 \( \text{div}[u_1, v_1, n_1 - \lceil \frac{g}{2} \rceil]^* \) and stop.
2. If \( n_1 < \lceil \frac{g}{2} \rceil \) let \( \delta = -1 \), otherwise, let \( \delta = +1 \).
3. Let \( \hat{v}_1 := v_1 + \delta(V - (V \mod u_1)) \) and \( u_2 := (f - \hat{v}_1^2)/u_1 \) made monic, and \( v_2 := -\hat{v}_1 \mod u_1 \) (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(\( \text{div}[u_2, v_2, n_2]^* \))
Addition and negation

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

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

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

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

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

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:

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

When \( g = 3, \) one call to \textbf{Reduce} and one nontrivial call to \textbf{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.

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

1. $w := (f - v_1^2)/u$ and $\tilde{s} := (v_2 - v_1)/u_1 \mod u_2$.
2. $c := 1/lc(\tilde{s})$ and $s = c\tilde{s}$ and $z := su_1$ (require $\deg(s) = 2$).
3. $u_4 := (s(z + 2cv_1) - c^2w)/u_2$ and $\tilde{v}_4 := v_1 + u_4 + (z \mod u_4)/c$.
4. $u_5 := (\tilde{v}_4^2 - f)/(2\tilde{v}_4u_4)$ and $v_5 := \tilde{v}_4 \mod u_5$ and $n_5 := 3 - \deg(u_5)$.

We then have $\text{div}[u_1, v_1, 0] + \text{div}[u_2, v_2, 0] \sim \text{div}[u_5, v_5, n_5]$.

$\text{div}[u_5, v_5, n_5]$ 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: \( I + 79M + 127A \) (vs \( 5I + 275M + 246A \)).
- Typical doubling: \( I + 82M + 127A \) (vs \( 5I + 285M + 258A \)).
- Typical negation: \( I + 14M + 24A \).

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. $z \leftarrow x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4 \quad (4M+3A)$
2. $z \leftarrow (((x^2)^2)^2)^2 \quad (4M, \text{ in fact } 4S)$

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

Which is faster?
Comparing operation counts (with caveats)

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

<table>
<thead>
<tr>
<th>Genus 2 odd degree</th>
<th>Addition</th>
<th>Doubling</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$I + 24M$</td>
<td>$I + 28M$</td>
<td>[Lange 05]</td>
</tr>
<tr>
<td>Genus 2 even degree</td>
<td>$I + 28M$</td>
<td>$I + 32M$</td>
<td>[GHM 08]</td>
</tr>
<tr>
<td>Genus 3 odd degree</td>
<td>$I + 67M$</td>
<td>$I + 68M$</td>
<td>[NMCT 06]</td>
</tr>
<tr>
<td>Genus 3 even degree</td>
<td>$I + 79M$</td>
<td>$I + 82M$</td>
<td>[this work]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Genus 3 even degree</th>
<th>Addition</th>
<th>Doubling</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$I + 75M$</td>
<td>$I + 86M$</td>
<td>[Rezai Rad 16]</td>
</tr>
</tbody>
</table>