MATHEMATICS OF COMPUTATION
Volume 86, Number 308, November 2017, Pages 3019–3029
http://dx.doi.org/10.1090/mcom/3200
Article electronically published on April 7, 2017

COMPUTING CANONICAL HEIGHTS ON THE PROJECTIVE LINE WITH NO FACTORIZATION

ELLIOT WELLS

Abstract. We give an algorithm which requires no integer factorization for computing the canonical height of a point in $\mathbb{P}^1(\mathbb{Q})$ relative to a morphism $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ of degree $d \geq 2$.

1. Introduction

Let $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ be a morphism of degree $d \geq 2$, and let h be the logarithmic height on $\mathbb{P}^1(\mathbb{Q})$. The *canonical height function relative to* ϕ (see [11]) is the function $\hat{h}_\phi : \mathbb{P}^1(\mathbb{Q}) \longrightarrow \mathbb{R}_{\geq 0}$ defined by

$$\hat{h}_\phi(P) = \lim_{n\to\infty} d^{-n} h(\phi^n(P)). \tag{1.1}$$

The canonical height of a point P relative to ϕ gives information about the behavior of P under the iteration of ϕ, and so canonical heights have numerous applications in arithmetic dynamics and arithmetic geometry. For example, $\hat{h}_\phi(P) = 0$ if and only if P is preperiodic under ϕ, and this fact provides a quick proof of a result of Northcott [8] that ϕ has finitely many preperiodic points in $\mathbb{P}^1(\mathbb{Q})$. Additionally, one of the quantities appearing in the Birch and Swinnerton-Dyer Conjecture—the regulator—is defined in terms of canonical heights on elliptic curves, which equivalently are canonical heights for Lattès maps on $\mathbb{P}^1$. There are also a number of related conjectures in arithmetic dynamics, such as the Dynamical Lehmer Conjecture, concerning lower bounds on canonical heights of non-preperiodic points (see [11, Conjecture 3.25]).

In this note, we give an algorithm for computing $\hat{h}_\phi(P)$ that is efficient even if every lift $\Phi = [F, G] : \mathbb{A}^2(\mathbb{Q}) \to \mathbb{A}^2(\mathbb{Q})$ of ϕ has large integer coefficients or if d is large. To appreciate why such an algorithm might be helpful, let us recall the usual methods for computing $\hat{h}_\phi(P)$. The limit definition (1.1) of $\hat{h}_\phi$ is generally unsuitable for computation, because the number of digits of the coordinates of $\phi^n(P)$ grows exponentially with n. As an alternative, one decomposes $\hat{h}_\phi(P)$ (with $P \neq \infty$) as a sum of local canonical heights corresponding to the different absolute values on $\mathbb{Q}$:

$$\hat{h}_\phi(P) = \sum_{v\in M_{\mathbb{Q}}} \hat{\lambda}_{\phi,v}(P) = \hat{\lambda}_{\phi,\infty}(P) + \sum_{p \text{ prime}} \hat{\lambda}_{\phi,p}(P); \tag{1.2}$$

see [11, Theorem 5.61]. Let $\Phi = [F, G] : \mathbb{A}^2(\mathbb{Q}) \to \mathbb{A}^2(\mathbb{Q})$ be a lift of ϕ such that F and G have relatively prime integer coefficients, and let $R := \mathrm{Res}(F, G)$ be the

Received by the editor March 3, 2016 and, in revised form, July 12, 2016.
2010 *Mathematics Subject Classification.* Primary 37P30; Secondary 11G50, 11Y16.

resultant of F and G. For all primes p such that $p \nmid R$, the local canonical height of $P = [x, y]$ at p is given by the simple formula

$$\hat{\lambda}_{\phi,p}(P) = \log\left(\frac{\max\{|x|_p, |y|_p\}}{|y|_p}\right),$$

which allows us to rewrite (1.2) as

$$\hat{h}_\phi(P) = h(P) + \tilde{\lambda}_{\phi,\infty}(P) + \sum_{p|R} \tilde{\lambda}_{\phi,p}(P), \tag{1.3}$$

where

$$\tilde{\lambda}_{\phi,v}(P) := \hat{\lambda}_{\phi,v}(P) - \log\left(\frac{\max\{|x|_v, |y|_v\}}{|y|_v}\right). \tag{1.4}$$

For any fixed $v \in M_{\mathbb{Q}}$, we can efficiently approximate the value of $\hat{\lambda}_{\phi,v}(P)$ (and hence also $\tilde{\lambda}_{\phi,v}(P)$) with the method described in [1, Section 5] or with the algorithm in [11, exercise 5.29]. So using decomposition (1.3), we can efficiently compute an approximation of $\hat{h}_\phi(P)$, *provided that we know the primes dividing* R. In practice, however, factoring R can be time-consuming. Indeed, R equals the determinant of a certain $2d \times 2d$ matrix whose entries are zeros and the coefficients of F and G (see [11, Proposition 2.13(d)]), so if d is of moderate size or if F and G have large coefficients, then we expect R to potentially be large and difficult to factor.

In this note, we describe a practical algorithm *which requires no integer factorization* for computing an approximation of the nonarchimedean term in (1.3). Combining this with any already existing algorithm for approximating the value of the archimedean term (e.g., [1, Section 5] or [11, exercise 5.29]) yields an algorithm for approximating $\hat{h}_\phi(P)$. The inspiration behind our algorithm comes from an algorithm (requiring no integer factorization) in [6] for computing the canonical height of a point on an elliptic curve; we show how the ideas in [6]—which deal with morphisms of elliptic curves over $\mathbb{Q}$—generalize to morphisms of $\mathbb{P}^1$ over $\mathbb{Q}$.

In fact, it is instructive to compare our original problem of computing $\hat{h}_\phi(P)$ to that of computing the canonical height of a point on an elliptic curve $E/\mathbb{Q}$ (as defined, e.g., in [9]). One can show that for $P \in E(\mathbb{Q})$, the nonarchimedean term in (1.2) has the form

$$\sum_{p \text{ prime}} r_p \log(p),$$

where the r_p, which *a priori* are arbitrary real numbers, are in fact rational numbers of a very precise form; moreover, these rational numbers can often be determined, with very little factorization, by computing just a few multiples $[2]P, [3]P, \ldots$ of P. These observations are made in [10] to give a nearly factorization-free algorithm for computing canonical heights on elliptic curves, and [6] gives a completely factorization-free algorithm by exploiting the constraints on the values of the r_p's. For a morphism $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$, in contrast, it is unknown whether the r_p's must be rational, and in any case they cannot be determined by simply computing a small number of iterates of ϕ. This makes computing $\hat{h}_\phi(P)$ more difficult, and any algorithm for doing so will, in general, return an approximation of the r_p's, rather than an exact value as provided by the algorithms of [10] and [6].

2. Notation and preliminary results

The following is a summary of the notation and definitions used in this paper:

- $M_{\mathbb{Q}}$ - the set of absolute values on $\mathbb{Q}$, with the usual normalizations.
- $M_{\mathbb{Q}}^{\infty}$ - the set of archimedean absolute values on $\mathbb{Q}$.
- $M_{\mathbb{Q}}^{0}$ - the set of nonarchimedean absolute values on $\mathbb{Q}$.
- $\mathrm{ord}_p(a)$ - the greatest exponent e such that p^e divides $a \in \mathbb{Z}$, for p a prime.
- $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ - a morphism of degree $d \geq 2$.
- $\Phi = [F, G] : \mathbb{A}^2(\mathbb{Q}) \to \mathbb{A}^2(\mathbb{Q})$ - a lift of ϕ with integer coefficients.
- $\|F, G\|$ - the maximum of the (archimedean) absolute values of the coefficients of F and G.
- $\|\cdot\|_v$ - the sup norm associated to $v \in M_{\mathbb{Q}}$; i.e., $\|(x, y)\|_v = \max\{|x|_v, |y|_v\}$ for any $(x, y) \in \mathbb{A}^2(\mathbb{Q})$.
- $\mathrm{Res}(F, G)$ - the resultant of the homogeneous polynomials F and G.
- h - the logarithmic height on $\mathbb{P}^1(\mathbb{Q})$.
- $\hat{h}_\phi$ - the canonical height associated with the morphism ϕ.
- $\Lambda_{\Phi,v} : \mathbb{P}^1(\mathbb{Q}) \to \mathbb{R}$ - the function defined by

$$\Lambda_{\Phi,v} : [x, y] \mapsto \log(\|\Phi(x, y)\|_v^{-1}) - d \log(\|(x, y)\|_v^{-1})$$

  for $v \in M_{\mathbb{Q}}$ and a lift $\Phi = [F, G]$ of ϕ.
- $\Omega_{\Phi,s}, \mathcal{H}_{\Phi,s} : \mathbb{P}^1(\mathbb{Q}) \to \mathbb{R}$ - for $s \in \{0, \infty\}$, the functions defined by

$$\Omega_{\Phi,s} : P \mapsto \sum_{v \in M_{\mathbb{Q}}^s} \Lambda_{\Phi,v}(P)$$

  and

$$\mathcal{H}_{\Phi,s} : P \mapsto \sum_{n=0}^{\infty} \frac{\Omega_{\Phi,s}(\phi^n(P))}{d^{n+1}} \tag{2.1}$$

  for a lift $\Phi = [F, G]$ of ϕ.

We begin by rewriting decomposition (1.3) in our notation.

Proposition 2.1. *Let $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ be a morphism of degree $d \geq 2$, and let Φ be a lift of ϕ. Then for any $P \in \mathbb{P}^1(\mathbb{Q})$, we have*

$$\hat{h}_\phi(P) = h(P) - \mathcal{H}_{\Phi,\infty}(P) - \mathcal{H}_{\Phi,0}(P).$$

Proof. This follows from [11, Theorem 5.59] and [11, exercise 5.29(c)]. □

Remark 2.2. Let us describe in detail the relationship between decomposition (1.3) and the decomposition of $\hat{h}_\phi(P)$ given in Proposition 2.1. By [11], we have

$$\mathcal{H}_{\Phi,\infty}(P) = -\tilde{\lambda}_{\phi,\infty}(P) \quad \text{and} \quad \mathcal{H}_{\Phi,0}(P) = -\sum_{p|R} \tilde{\lambda}_{\phi,p}(P)$$

for any $P \in \mathbb{P}^1(\mathbb{Q})$, where $R = \mathrm{Res}(F, G)$. Moreover, the second equality holds for each individual prime p, in the sense that

$$-\tilde{\lambda}_{\phi,p}(P) = \sum_{n=0}^{\infty} \frac{\Lambda_{\Phi,p}(\phi^n(P))}{d^{n+1}}. \tag{2.2}$$

The idea behind our algorithm is that rather than separately approximating the infinite series (2.2) for each prime p dividing R, we can consider all of these infinite series in aggregate by interchanging a summation; namely,

$$\mathcal{H}_{\Phi,0}(P) = -\sum_{p|R} \tilde{\lambda}_{\phi,p}(P) = \sum_{n=0}^{\infty} \frac{1}{d^{n+1}} \left(\sum_{p|R} \Lambda_{\Phi,p}(\phi^n(P)) \right).$$

Our algorithm iteratively computes the value of

$$\sum_{p|R} \Lambda_{\Phi,p}(\phi^n(P))$$

for $n = 0, 1, \ldots$ up to some bound N; crucially, this value is computed *without* separately computing $\Lambda_{\Phi,p}(\phi^n(P))$ for each individual p. This eliminates the need to factor R into primes.

Next, we establish some simple facts that are used in the following section to analyze our algorithm.

Lemma 2.3. *Let $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ be a morphism of degree $d \geq 2$, and let $\Phi = [F, G]$ be a lift of ϕ such that F and G have integer coefficients. Let $Q \in \mathbb{P}^1(\mathbb{Q})$, and write $Q = [x, y]$ with $(x, y) \in \mathbb{Z}^2$ and $\gcd(x, y) = 1$. Then*

$$\Omega_{\Phi,0}(Q) = \log(\gcd(F(x, y), G(x, y))).$$

Proof. Let $v \in M^0_{\mathbb{Q}}$ be the absolute value associated with the prime number p. Then

$$\Lambda_{\Phi,v}(Q) = \log(\|F(x, y), G(x, y)\|_v^{-1}) = \log p^{\min\{\mathrm{ord}_p(F(x,y)), \mathrm{ord}_p(G(x,y))\}},$$

and so

$$\Omega_{\Phi,0}(Q) = \sum_{p \text{ prime}} \log p^{\min\{\mathrm{ord}_p(F(x,y)), \mathrm{ord}_p(G(x,y))\}} = \log(\gcd(F(x, y), G(x, y))).$$

□

Lemma 2.4. *Let $F(X, Y), G(X, Y) \in \mathbb{Z}[X, Y]$ be homogeneous polynomials, and let $(a, b) \in \mathbb{Z}^2$ with $\gcd(a, b) = 1$. Then $\gcd(F(a, b), G(a, b))$ divides $\mathrm{Res}(F, G)$.*

Proof. See the proof of [2, Lemma 17.1]. □

Lemma 2.5. *Let $F(X, Y), G(X, Y) \in \mathbb{Z}[X, Y]$ be homogeneous polynomials, and let $(x, y) \in \mathbb{Z}^2$ with $\gcd(x, y) = 1$. Fix any $(x', y') \in \mathbb{Z}^2$ such that*

$$x' \equiv F(x, y) \pmod{\mathrm{Res}(F, G)},$$
$$y' \equiv G(x, y) \pmod{\mathrm{Res}(F, G)}.$$

Then

$$\gcd(F(x, y), G(x, y)) = \gcd(x', y', \mathrm{Res}(F, G)).$$

Proof. In general, if $(a, b) \in \mathbb{Z}^2$ and $R \in \mathbb{Z}$ such that $\gcd(a, b)$ divides R, then

$$\gcd(a, b) = \gcd(a + cR, b + dR, R)$$

for any $c, d \in \mathbb{Z}$. By Lemma 2.4, we can apply this to the case $a = F(x, y), b = G(x, y)$, and $R = \mathrm{Res}(F, G)$, which gives the desired result. □

Lemma 2.6. *Let $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ be a morphism of degree $d \geq 2$, and let $\Phi = [F, G]$ be a lift of ϕ such that F and G have integer coefficients. Then for any $P \in \mathbb{P}^1(\mathbb{Q})$, we have $|\Omega_{\Phi,0}(P)| \leq \log |\mathrm{Res}(F, G)|$.*

Proof. Fix $P \in \mathbb{P}^1(\mathbb{Q})$, and write $P = [x, y]$ with $(x, y) \in \mathbb{Z}^2$ and $\gcd(x, y) = 1$. By Lemma 2.3, we know that $\Omega_{\Phi,0}(P) = \log(\gcd(F(x, y), G(x, y)))$, where $\gcd(F(x, y), G(x, y))$ divides $\mathrm{Res}(F, G)$ by Lemma 2.4. The desired inequality clearly follows. □

3. The algorithm

By Proposition 2.1, to efficiently compute $\hat{h}_\phi(P)$, it suffices to efficiently compute the three quantities $h(P), \mathcal{H}_{\Phi,\infty}(P)$, and $\mathcal{H}_{\Phi,0}(P)$. Of course, computing $h(P)$ is easy, and there are efficient algorithms for computing $\mathcal{H}_{\Phi,\infty}(P)$, e.g., the method described in [1, Section 5] or [11, exercise 5.29].

We give an algorithm that efficiently approximates the value of $\mathcal{H}_{\Phi,0}(P)$ by computing the first N terms in the infinite series definition (2.1) of $\mathcal{H}_{\Phi,0}(P)$; notably, our algorithm does not require the prime factorization of $\mathrm{Res}(F, G)$.

Our algorithm was inspired by [6], which develops a factorization-free algorithm for computing canonical heights on elliptic curves. In [7], the same authors provide such an algorithm for computing canonical heights on genus two Jacobians; in particular, see [7, Proposition 14.1] for an analogous algorithm in that context.

Algorithm 3.1. Input: A morphism $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ of degree $d \geq 2$ with lift $\Phi = [F, G]$ such that F and G have integer coefficients, a point $P \in \mathbb{P}^1(\mathbb{Q})$, and a number N of terms in the sum to compute.

Output: An approximation of the value of $\mathcal{H}_{\Phi,0}(P)$, accurate to within $O(d^{-N})$. More precisely, the output $\mathcal{H}$ satisfies

$$|\mathcal{H}_{\Phi,0}(P) - \mathcal{H}| \leq \frac{\log|\mathrm{Res}(F, G)|}{(d-1)d^N},$$

and is computed by working with numbers of size at most $O(\mathrm{Res}(F, G)^N)$.

(1) Write $P = [x_0, y_0]$ with $(x_0, y_0) \in \mathbb{Z}^2$ and $\gcd(x_0, y_0) = 1$.
(2) Set $\mathcal{H} = 0$ and $R = \mathrm{Res}(F, G)$.
(3) For $i = 0, 1, \ldots, N-1$:
   (a) $x'_{i+1} = F(x_i, y_i) \pmod{R^{N-i}}$,
   $y'_{i+1} = G(x_i, y_i) \pmod{R^{N-i}}$.
   (b) $g_i = \gcd(x'_{i+1}, y'_{i+1}, R)$.
   (c) $\mathcal{H} = \mathcal{H} + \dfrac{\log(g_i)}{d^{i+1}}$.
   (d) $(x_{i+1}, y_{i+1}) = (x'_{i+1}/g_i, y'_{i+1}/g_i)$.
(4) Return $\mathcal{H}$.

Proposition 3.2. *Algorithm* 3.1 *computes* $\mathcal{H}$*, which satisfies*

$$\mathcal{H} = \sum_{n=0}^{N-1} \frac{\Omega_{\Phi,0}(\phi^n(P))}{d^{n+1}} \tag{3.1}$$

and

$$|\mathcal{H}_{\Phi,0}(P) - \mathcal{H}| \leq \frac{\log|\mathrm{Res}(F, G)|}{(d-1)d^N}. \tag{3.2}$$

Proof. Let $x_j, y_j, x'_j, y'_j, g_j, R$, and $\mathcal{H}$ be as defined in Algorithm 3.1. Recursively define a sequence (a_j, b_j) by $(a_0, b_0) = (x_0, y_0)$ and

$$(a_{j+1}, b_{j+1}) = \left(\frac{F(a_j, b_j)}{\gcd(F(a_j, b_j), G(a_j, b_j))}, \frac{G(a_j, b_j)}{\gcd(F(a_j, b_j), G(a_j, b_j))} \right)$$

for $j = 0, \ldots, N-1$. It is clear by induction that $(a_j, b_j) \in \mathbb{Z}^2$ with $\gcd(a_j, b_j) = 1$ and that $\phi^j(P) = [a_j, b_j]$ for $j = 0, \ldots, N-1$. In particular, Lemma 2.3 implies that

$$\Omega_{\Phi,0}(\phi^j(P)) = \log(\gcd(F(a_j, b_j), G(a_j, b_j)))$$

for $j = 0, \ldots, N-1$, so to establish (3.1), it suffices to show that

$$g_j = \gcd(F(a_j, b_j), G(a_j, b_j))$$

for $j = 0, \ldots, N-1$. By using induction on j, we will show the stronger result that the following three statements hold for $j = 0, \ldots, N-1$:

(a) $x_j \equiv a_j \pmod{R^{N-j}}$; $\quad y_j \equiv b_j \pmod{R^{N-j}}$.
(b) $x'_{j+1} \equiv F(a_j, b_j) \pmod{R^{N-j}}$; $\quad y'_{j+1} \equiv G(a_j, b_j) \pmod{R^{N-j}}$.
(c) $g_j = \gcd(F(a_j, b_j), G(a_j, b_j))$.

For the base case ($j = 0$), we have that (a) and (b) hold because by definition, $(a_0, b_0) = (x_0, y_0)$ and

$$x'_1 \equiv F(x_0, y_0) \pmod{R^N}; \quad y'_1 \equiv G(x_0, y_0) \pmod{R^N}.$$

Moreover, Lemma 2.5 and (b) imply that

$$\gcd(F(a_0, b_0), G(a_0, b_0)) = \gcd(x'_1, y'_1, R),$$

which establishes (c).

Now assume that (a), (b), and (c) hold for $j = m-1$, for some fixed $m \leq N-1$. By (b), we have

$$x'_m \equiv F(a_{m-1}, b_{m-1}) \pmod{R^{N-(m-1)}},$$
$$y'_m \equiv G(a_{m-1}, b_{m-1}) \pmod{R^{N-(m-1)}}.$$

We know that g_{m-1} divides x'_m, y'_m, and R by definition, and (c) implies that

$$g_{m-1} = \gcd(F(a_{m-1}, b_{m-1}), G(a_{m-1}, b_{m-1}));$$

it follows that

$$\frac{x'_m}{g_{m-1}} \equiv \frac{F(a_{m-1}, b_{m-1})}{\gcd(F(a_{m-1}, b_{m-1}), G(a_{m-1}, b_{m-1}))} \pmod{R^{N-m}},$$
$$\frac{y'_m}{g_{m-1}} \equiv \frac{G(a_{m-1}, b_{m-1})}{\gcd(F(a_{m-1}, b_{m-1}), G(a_{m-1}, b_{m-1}))} \pmod{R^{N-m}},$$

which establishes (a) for $j = m$. Then (b) for $j = m$ follows immediately from (a) (for $j = m$) and from the definition of x'_{m+1}, y'_{m+1}. Finally, Lemma 2.5 applied to (b) when $j = m$ yields

$$\gcd(F(a_m, b_m), G(a_m, b_m)) = \gcd(x'_{m+1}, y'_{m+1}, R),$$

which shows that (c) holds for $j = m$.

By (3.1) and Lemma 2.6, we have

$$|\mathcal{H}_{\Phi,0}(P) - \mathcal{H}| \le \sum_{n=N}^{\infty} \frac{\log|\mathrm{Res}(F,G)|}{d^{n+1}} = \frac{\log|\mathrm{Res}(F,G)|}{(d-1)d^N},$$

which establishes (3.2). □

Remark 3.3. The modulus in Step (3) of Algorithm 3.1 decreases with each iteration of the loop. As a result, each successive iteration generally has a faster runtime than the previous one.

Remark 3.4. The error bound (3.2) involves the quantity $|\mathrm{Res}(F,G)|$, and the runtime of our algorithm is determined by the time needed to perform calculations with numbers of size $O(\mathrm{Res}(F,G)^N)$. So to understand the accuracy and efficiency of Algorithm 3.1, it is necessary to have a bound on the size of $\mathrm{Res}(F,G)$.

Recall (e.g., [11, Proposition 2.13(d)]) that the resultant of two homogeneous polynomials of degree d can be expressed as the determinant of a certain $2d \times 2d$ matrix in which each row consists of $d-1$ zeros and the $d+1$ coefficients of one of the polynomials. It follows from Hadamard's inequality that

$$|\mathrm{Res}(F,G)| \le (d+1)^d \, \|F,G\|^{2d}.$$

Remark 3.5. We can incorporate a factoring step in Algorithm 3.1, as follows. Choose some "small" bound B and factor $R = \mathrm{Res}(F,G)$ into primes as

$$R = p_1^{e_1} \cdots p_t^{e_t} \cdot \tilde{R},$$

with each $p_i \le B$. Then $\mathcal{H}_{\Phi,0}(P)$ is closely approximated by

$$\widetilde{\mathcal{H}} + \sum_{i=1}^{t} \tilde{\lambda}_{\phi,p_i}(P),$$

where the $\tilde{\lambda}_{\phi,p_i}(P)$'s are the modified local canonical heights defined by (1.4), and where $\widetilde{\mathcal{H}}$ denotes the output of Algorithm 3.1 with R replaced by $\tilde{R}$ in Step (3). As previously noted, there are efficient algorithms in the literature for computing $\tilde{\lambda}_{\phi,p_i}(P)$; alternatively, we can compute $\tilde{\lambda}_{\phi,p_i}(P)$ by running Algorithm 3.1 with R replaced by $p_i^{e_i}$ in Step (3).

More generally, if R factors into pairwise relatively prime integers as

$$R = \tilde{R}_1 \tilde{R}_2 \cdots \tilde{R}_t,$$

then $\mathcal{H}_{\Phi,0}(P)$ is approximated by the sum

$$\widetilde{\mathcal{H}}_1 + \widetilde{\mathcal{H}}_2 + \cdots + \widetilde{\mathcal{H}}_t,$$

where $\widetilde{\mathcal{H}}_i$ denotes the output of Algorithm 3.1 with R replaced by $\tilde{R}_i$ in Step (3).

4. Numerical examples

In this section, we give some examples illustrating the use of Algorithm 3.1. The most novel and useful aspect of our algorithm is that it does not require that we factor $R = \mathrm{Res}(F,G)$. Hence, Algorithm 3.1 is particularly advantageous when R is large, and by Remark 3.4 this can occur when either d is not very small, or when F and G have large coefficients. We give two examples demonstrating each of these scenarios.

We also note that the current implementation in Sage [3] for computing $\hat{h}_\phi(P)$ uses the decomposition (1.3) of $\hat{h}_\phi(P)$ into local canonical heights. In particular, one of the steps in this algorithm requires the factorization of R, rendering the algorithm completely impractical for morphisms ϕ for which R is very large. Algorithm 3.1 can be used to compute $\hat{h}_\phi(P)$ in these cases.

In the examples that follow, we continue to employ the same notation as in the previous sections, including the notation in Algorithm 3.1.

Example 4.1. Define "random" polynomials:

$$\tau_1(z) = 3z^{80} + z^{79} + 4z^{78} + z^{77} + 5z^{76} + \cdots + 9z^2 + 3z + 7,$$
$$\tau_2(z) = 2z^{80} + 7z^{79} + z^{78} + 8z^{77} + 2z^{76} + \cdots + 6z^2 + 9,$$

such that the coefficient of z^{81-i} in τ_1 (respectively, τ_2) equals the ith digit of π (respectively, e), and set

$$\sigma(z) = \frac{\tau_1(z)}{\tau_2(z)}.$$

Let $\phi : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ be the morphism of degree $d = 80$ induced by the rational map σ. We calculate the canonical height of the point $P = [-5, 1]$ relative to ϕ.

Taking $\Phi = [F, G]$ to be a lift of ϕ, where F and G are the respective degree 80 homogenizations of τ_1 and τ_2, we compute

$$\begin{aligned}\operatorname{Res}(F, G) &= 516438964415067184645 \cdots 3035414853764920593 92\\ &= 2^8 \cdot 3^2 \cdot R'\\ &\approx 2^{653}\end{aligned}$$

for some large R'. In particular, $\operatorname{Res}(F, G)$ is over 650 bits and so is time-consuming to factor into primes.

Using Algorithm 3.1 with $N = 50$, we compute

$$g_0 = 36, \quad g_1 = 2, \quad g_2 = 12,$$

and

$$g_i = \begin{cases} 2 & \text{if } i \equiv 1 \pmod 2,\\ 4 & \text{if } i \equiv 0 \pmod 2,\end{cases}$$

for $3 \le i \le 49$, so that

$$\mathcal{H}_{\Phi,0}(P) \approx \sum_{i=0}^{49} \frac{\log(g_i)}{d^{i+1}} \approx 0.044907161659276960113044136254.$$

By Proposition 3.2, the error in this approximation of $\mathcal{H}_{\Phi,0}(P)$ (prior to truncating the decimal expansion) is at most

$$\frac{\log|\operatorname{Res}(F, G)|}{(d-1)d^N} < 10^{-94}.$$

Note that for this computation, we must work with numbers modulo $\operatorname{Res}(F, G)^{50}$, which is approximately 32674 bits.

Using a close variant of the algorithm in [11, exercise 5.29] with 50 iterations, we compute

$$\mathcal{H}_{\Phi,\infty}(P) \approx -0.013757185585214127675440651473.$$

We also have

$$h(P) = \log(5) \approx 1.60943791243410037460075933 32.$$

So by Proposition 2.1, we have

$$\begin{aligned}\hat{h}_\phi(P) &= h(P) - \mathcal{H}_{\Phi,\infty}(P) - \mathcal{H}_{\Phi,0}(P)\\ &\approx 1.57828793636003754216315584 84.\end{aligned}$$

Example 4.2. This example is similar to Example 4.1, except in this case we use a point whose canonical height relative to our chosen morphism is very small.

Define "random" polynomials

$$\tau_1(z) = \sum_{i=0}^{65} a_i z^{65-i}, \quad \tau_2(z) = \sum_{i=0}^{65} b_i z^{65-i}$$

by

$$a_i = \begin{cases} -i & \text{if } i \text{ is prime,} \\ 1 & \text{if } i \text{ is not prime,} \end{cases} \qquad b_i = \begin{cases} 1 & \text{if } 0 \le i \le 33, \\ -1 & \text{if } 34 \le i \le 65, \end{cases}$$

and define σ, ϕ, and $\Phi = [F, G]$ in an analogous manner to Example 4.1. Note that $\mathrm{Res}(F, G) \approx 2^{433}$ is very large and thus difficult to factor.

Let $P = [0, 1]$. From Algorithm 3.1 with $N = 50$, which gives an approximation of $\mathcal{H}_{\Phi,0}(P)$ that is accurate to within 10^{-89}, we compute

$$g_0 = 1, g_2 = 513, g_2 = 1, g_3 = 1, \ldots, g_{46} = 19, g_{47} = 1, g_{48} = 1, g_{49} = 27;$$

each $g_i \in \{1, 19, 27, 513\}$, and the sequence of g_i's is periodic with period 20 (at least for the first 50 terms in the sequence). It follows that

$$\mathcal{H}_{\Phi,0}(P) \approx 0.00147698841002194309075886360 39$$

and

$$\hat{h}_\phi(P) \approx 0.00000034264800824399071146803578925.$$

Example 4.3. Consider the family of degree 2 morphisms $\phi_a : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$, where ϕ_a is induced by the rational map

$$z \mapsto \frac{z^2 + z + 1}{z^2 + az + 2}, \quad a \in \mathbb{Z}.$$

A lift $\Phi_a = [F_a, G_a]$ for ϕ_a is given by

$$F_a(X, Y) = X^2 + XY + Y^2, \quad G_a(X, Y) = X^2 + aXY + 2Y^2,$$

and the corresponding resultant is

$$\mathrm{Res}(F_a, G_a) = a^2 - 3a + 3,$$

which is hard to factor when a is large. For instance, if a equals the number formed by concatenating the first 201 digits of π, then $\mathrm{Res}(F_a, G_a) \approx 2^{1332}$. For this value of a and $P = [1, 1]$, we calculate $\hat{h}_{\phi_a}(P)$.

Using $N = 50$ terms in Algorithm 3.1, which allows for an approximation with error less than 10^{-12}, we compute

$$g_0 = 3, g_1 = 1, g_2 = 1, g_3 = 3, \ldots, g_{46} = 3, g_{47} = 1, g_{48} = 3, g_{49} = 1;$$

each $g_i \in \{1, 3\}$, and there is no apparent repeating pattern in the sequence of g_i's. This gives the approximation $\mathcal{H}_{\Phi_a,0}(P) \approx 0.62900702$, which yields

$$\hat{h}_{\phi_a}(P) \approx 307.43849177.$$

Example 4.4. Let $\phi_a : \mathbb{P}^1_{\mathbb{Q}} \to \mathbb{P}^1_{\mathbb{Q}}$ be the much-studied family of degree 2 morphisms induced by the rational map

$$z \mapsto az + \frac{1}{z}, \quad a \in \mathbb{Z}.$$

See for example [5]. We can lift ϕ_a to $\Phi_a = [F_a, G_a]$, where

$$F_a(X,Y) = aX^2 + Y^2, \quad G_a(X,Y) = XY,$$

and the resultant $\mathrm{Res}(F_a, G_a) = a$ is time-consuming to factor for large a.

As an example, we compute $\hat{h}_{\phi_a}(P)$ for $P = [a, 1]$, where a equals RSA-768—a 768-bit RSA modulus which took a team of researchers over 2 years to factor with the number field sieve (cf. [4]). Running Algorithm 3.1 to $N = 50$ terms, which gives an approximation with error bounded above by 10^{-12}, we get $g_0 = 1$, $g_1 = a$, and $g_i = 1$ for $2 \le i \le 49$. This provides the approximation

$$\mathcal{H}_{\Phi_a,0}(P) \approx 133.0260806,$$

which allows us to compute

$$\hat{h}_{\phi_a}(P) \approx 931.1825642.$$

Remark 4.5. We have implemented Algorithm 3.1 in Sage. Using this implementation, we computed the canonical heights in the previous four examples in $9.4, 6.6, 0.3$ and 0.1 seconds, respectively. These computations were performed on a Dell Inspiron N4010 laptop.

Acknowledgments

The author would like to thank his advisor, Joseph Silverman for his guidance and insightful discussions on this problem, as well as for his helpful suggestions on improving the drafts of this paper. The author also thanks Jan Steffen Müller and the referee for their helpful comments.

References

[1] G. S. Call and J. H. Silverman, *Canonical heights on varieties with morphisms*, Compositio Math. **89** (1993), no. 2, 163–205. MR1255693

[2] J. W. S. Cassels, *Lectures on elliptic curves*, London Mathematical Society Student Texts, vol. 24, Cambridge University Press, Cambridge, 1991. MR1144763

[3] The Sage Developers, *Sage Mathematics Software (Version 7.0)*, 2016, `http://www.sagemath.org`.

[4] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann, *Factorization of a 768-bit RSA modulus*, Advances in cryptology—CRYPTO 2010, Lecture Notes in Comput. Sci., vol. 6223, Springer, Berlin, 2010, pp. 333–350, DOI 10.1007/978-3-642-14623-7_18. MR2725602

[5] M. Manes, *$\mathbb{Q}$-rational cycles for degree-2 rational maps having an automorphism*, Proc. Lond. Math. Soc. (3) **96** (2008), no. 3, 669–696, DOI 10.1112/plms/pdm044. MR2407816

[6] J. Steffen Müller and Michael Stoll, *Computing canonical heights on elliptic curves in quasi-linear time*, arXiv:1509.08748v2, 2015.

[7] J. Steffen Müller and Michael Stoll, *Canonical heights on genus two jacobians*, arXiv:1603.00640v1, 2016.

[8] D. G. Northcott, *Periodic points on an algebraic variety*, Ann. of Math. (2) **51** (1950), 167–177. MR0034607

[9] J. H. Silverman, *Advanced topics in the arithmetic of elliptic curves*, Graduate Texts in Mathematics, vol. 151, Springer-Verlag, New York, 1994. MR1312368

[10] J. H. Silverman, *Computing canonical heights with little (or no) factorization*, Math. Comp. **66** (1997), no. 218, 787–805, DOI 10.1090/S0025-5718-97-00812-0. MR1388892

[11] J. H. Silverman, *The arithmetic of dynamical systems*, Graduate Texts in Mathematics, vol. 241, Springer, New York, 2007. MR2316407

Department of Mathematics, Brown University, Providence, Rhode Island 02912
E-mail address: ellwells@math.brown.edu