JSIAM Letters Vol.2 (2010) pp.17–20 ©2010 Japan Society for Industrial and Applied Mathematics JSIAM Letters

Improvements in the computation of the Hasse-Witt matrix

Hiroki Komoto[1], Shunji Kozaki[1] and Kazuto Matsuo[1]

Institute of Information Security, 2-14-1, Tsuruya-cho Kanagawa-ku, Yokohama 221-0835, Japan[1]

E-mail *dgs074103@iisec.ac.jp*

Received November 17, 2009, Accepted January 8, 2010

Abstract

The Hasse-Witt matrix of a hyperelliptic curve gives partial information for the order of the Jacobian of the curve, therefore the Hasse-Witt matrices can be used for point counting of hyperelliptic curves. Bostan, Gaudry and Schost improved the Chudnovsky-Chudnovsky algorithm and computed the Hasse-Witt matrices by using their improved algorithm for constructing hyperelliptic cryptosystems. The both algorithms need p-adic integers with finite precision as the base operations. This paper shows improvements in the computation of the Hasse-Witt matrix that reduces the required precision of the p-adic integers.

Keywords hyperelliptic curves, hyperelliptic curve cryptosystems, point counting, Hasse-Witt matrices

Research Activity Group Algorithmic Number Theory and Its Applications

1. Introduction

Hyperelliptic curve cryptosystems are constructed on rational point groups of the Jacobians of hyperelliptic curves defined over finite fields. Their security depends on the difficulty of the discrete logarithm problems on the rational point groups. The complexity of the problems is strongly affected by the group orders. Therefore, in order to construct secure hyperelliptic curve cryptosystems, one needs to know the group orders. The order can be obtained from the characteristic polynomial of the Frobenius map on the Jacobian. The residues of the coefficients in the characteristic polynomial modulo p can be derived from the Hasse-Witt matrix, where p is the characteristic of the field over which the curve is defined. Therefore, the Hasse-Witt matrices can be used for computing the orders of the Jacobians of hyperelliptic curves and more general curves [1–5].

The Hasse-Witt matrix consists of coefficients of a power of a polynomial defining the curve. These coefficients can be computed by using the Chudnovsky-Chudnovsky algorithm [6] for a linear recurrence with the polynomial coefficients. Bostan, Gaudry and Schost [3, 5] improved the Chudnovsky-Chudnovsky algorithm and computed the Hasse-Witt matrices for constructing hyperelliptic curve cryptosystems over finite fields of relatively large characteristics. These algorithms essentially need p-adic integers with finite precision.

This paper shows a method to speed up the computation of the Hasse-Witt matrix. The proposed method reduces the required precision of the p-adic integers by using the reversals of polynomials.

This paper is organized as follows. Section 2 defines the Hasse-Witt matrix and describes the computation of the Hasse-Witt matrix. Section 3 shows improvements in the computation of the Hasse-Witt matrix and Section 4 shows experimental results for the improvements. Finally, Section 5 concludes this paper.

2. Computation of the Hasse-Witt matrix using a linear recurrence

This section defines the Hasse-Witt matrix of a hyperelliptic curve and summarizes the computation of the Hasse-Witt matrix using a linear recurrence.

Let p be an odd prime and $\mathbb{F}_p$ be a finite field of order p. A hyperelliptic curve C over $\mathbb{F}_p$ of genus g is defined by

$$C : Y^2 = F(X), \quad F(X) = \sum_{i=0}^{2g+1} f_i X^i \in \mathbb{F}_p[X], \tag{1}$$

where $F(X)$ is a monic (i.e. $f_{2g+1} = 1$) square-free polynomial. For simplicity, $g \ll p$ and $f_0 \neq 0$ are assumed in the following.

Definition 1 (Hasse-Witt matrix) *Let h_k denote the coefficient of X^k in the polynomial $(F(X))^{\frac{p-1}{2}}$ for C. The Hasse-Witt matrix of C is defined by a $g \times g$ matrix over $\mathbb{F}_p$ whose (i, j)-th component is h_{jp-i}:*

$$H = \begin{pmatrix} h_{p-1} & h_{2p-1} & \cdots & h_{gp-1} \\ h_{p-2} & h_{2p-2} & \cdots & h_{gp-2} \\ \vdots & \vdots & \ddots & \vdots \\ h_{p-g} & h_{2p-g} & \cdots & h_{gp-g} \end{pmatrix}.$$

The following theorem is known for the Hasse-Witt matrix.

Theorem 2 (Manin [7]) *Let $\chi_p(X)$ denote the characteristic polynomial of the p-power Frobenius map on the Jacobian of C and H denote the Hasse-Witt matrix of C, then*

$$\chi_p(X) \equiv (-1)^g X^g \det(H - XI) \mod p,$$

where I is a $g \times g$ unit matrix.

Therefore, the residues of the coefficients in the characteristic polynomial $\chi_p(X)$ modulo p can be computed from the Hasse-Witt matrix H by using Theorem 2.

An usual polynomial powering, such as the binary method, can compute all the coefficients of $(F(X))^{\frac{p-1}{2}}$ within $O(\mathtt{M}(gp)\log p)$ operations in $\mathbb{F}_p$, where $\mathtt{M}(n)$ denotes the cost for a multiplication of polynomials of degree less than n in $\mathbb{F}_p[X]$. On the other hand, the Chudnovsky-Chudnovsky algorithm in [6, Section 6] can compute the Hasse-Witt matrix within $O(g^{\omega+1}\mathtt{M}(\sqrt{p})$ $+g^3\mathtt{M}(\sqrt{p})\log p)$ operations, where $O(g^\omega)$ is the complexity of a $g \times g$ matrix multiplication. Bostan, Gaudry and Schost [3, 5] improved the Chudnovsky-Chudnovsky algorithm and showed that, by using their improved algorithm, the Hasse-Witt matrix can be computed within $O(g^{\omega+1}\sqrt{p} + g^3\mathtt{M}(\sqrt{p}))$ operations. Therefore, these algorithms are asymptotically faster than the algorithms using polynomial powering. In the following, we describe the computation of the Hasse-Witt matrix in [3, 5], restricting ourselves to what we need for our improvements in the later section.

Let rational functions $r_i(X)$ with a variable X for $1 \leq i \leq 2g+1$ be

$$r_i(X) = \frac{f_i\left(i\dfrac{p+1}{2} - X\right)}{f_0 X}, \tag{2}$$

and a $(2g+1) \times (2g+1)$ matrix $A(X)$ be

$$A(X) = \begin{pmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \\ r_{2g+1}(X) & r_{2g}(X) & \cdots & r_1(X) \end{pmatrix}. \tag{3}$$

For any positive integer k, $A(k)$ denotes the matrix which is obtained by substituting $X = k$. Let h_k be as in Definition 1 and a $(2g+1)$-dimensional column vector U_k be

$$U_k = {}^t\begin{pmatrix} h_{k-2g} & \cdots & h_{k-1} & h_k \end{pmatrix}, \tag{4}$$

where $h_{-2g} = \cdots = h_{-1} = 0$. Then one can obtain a linear recurrence given by

$$\begin{aligned} U_k &= A(k)U_{k-1} \\ &= A(k)A(k-1)\cdots A(1)U_0 \end{aligned} \tag{5}$$

from [8, Chapter IV] (see also [9, Problem 4] and [3, p. 52]). Therefore, one can compute

$$U_{jp-1} = {}^t\begin{pmatrix} h_{jp-2g-1} & \cdots & h_{jp-2} & h_{jp-1} \end{pmatrix}$$

for $1 \leq j \leq g$ by using the linear recurrence (5) starting from

$$U_0 = {}^t\begin{pmatrix} 0 & \cdots & 0 & f_0^{\frac{p-1}{2}} \end{pmatrix},$$

which can be obtained from the constant term $h_0 = f_0^{\frac{p-1}{2}}$ in $(F(X))^{\frac{p-1}{2}}$. Then the Hasse-Witt matrix H can be obtained from the components of the vectors U_{jp-1} for $1 \leq j \leq g$.

However, since the components in $A(X)$ contain the rational functions (2), divisions by p are involved in computing U_{jp-1}. Therefore, those algorithms cannot be executed over $\mathbb{F}_p$. The algorithm in [3, 5] used p-adic integers $\mathbb{Z}_p$ for computing the Hasse-Witt matrix, because p-adic numbers that contain $\mathbb{Z}_p$ permit divisions by p and the residues of elements in $\mathbb{Z}_p$ modulo p give elements in $\mathbb{F}_p$. The Hasse-Witt matrix can be obtained using $\mathbb{Z}_p$ as follows. First, one lifts the coefficients of $F(X)$ to $\mathbb{Z}_p$ (i.e. one considers $F(X)$ in $\mathbb{Z}_p[X]$) and computes all the vectors U_{p-1}, U_{2p-1}, $\cdots$, U_{gp-1} over $\mathbb{Z}_p$. Then, by reducing the components of these vectors modulo p, one can obtain the Hasse-Witt matrix over $\mathbb{F}_p$.

[3, 5] computed those vectors over $\mathbb{Z}_p$ with finite precision as follows. Let a matrix $B(k) := f_0 k A(k)$ over $\mathbb{Z}_p$, then the linear recurrence (5) implies

$$U_{jp-1} = \frac{1}{f_0^{jp-1}(jp-1)!}B(jp-1)\cdots B(1)U_0$$

for $1 \leq j \leq g$. By the assumption $g \ll p$, the factorial term $(jp-1)!$ in the denominator is exactly divided by p^{j-1}. So, U_{gp-1} for the case $j = g$ requires the division by the highest power of p, i.e. p^{g-1}. Therefore, in order to obtain the Hasse-Witt matrix over $\mathbb{F}_p$, it is enough to compute the vectors $U_{p-1}, U_{2p-1}, \cdots, U_{gp-1}$ over $\mathbb{Z}_p$ with the finite precision up to g, whose arithmetic can be done in $\mathbb{Z}/p^g\mathbb{Z}$.

The next section shows improvements in the computation of the Hasse-Witt matrix by reducing the required precision.

3. Improvements

This section shows that the precision of the p-adic integers in the algorithms using the linear recurrence can be reduced. Moreover, one can further reduce the precision by using the linear recurrence with the coefficients of the reversal of a polynomial.

3.1 Reducing the precision of the p-adic integers

In the algorithms using the linear recurrence (5), the divisions by p occur at the computation in

$$U_{jp} = A(jp)U_{jp-1}, \quad 1 \leq j < g.$$

The $(2g+1)$-th component of U_{jp} is

$$h_{jp} = \frac{w_j}{p},$$

where

$$\begin{aligned} w_j = {}& \frac{f_{2g+1}\left((2g+1)\dfrac{p+1}{2} - jp\right)}{f_0 j}h_{jp-2g-1} \\ & + \cdots + \frac{f_1\left(\dfrac{p+1}{2} - jp\right)}{f_0 j}h_{jp-1} \in \mathbb{Z}_p. \end{aligned}$$

If w_j is computed over $\mathbb{Z}/p^g\mathbb{Z}$, then the significant precision of h_{jp} is reduced from g to $g-1$, i.e. h_{jp} should be in $\mathbb{Z}/p^{g-1}\mathbb{Z}$. Note that h_{jp} is in $\mathbb{Z}_p$, because h_{jp} is the coefficient in $(F(X))^{\frac{p-1}{2}}$. Therefore, if U_{jp} is given, then U_{kp} for $k \geq j$ can be computed with the same precision as U_{jp}, whose precision is lower than U_{jp-1}. On the other hand, the components in the Hasse-Witt matrix can be obtained from the components of

$$U_{jp} = {}^t\begin{pmatrix} h_{jp-2g} & \cdots & h_{jp-1} & h_{jp} \end{pmatrix}$$

for $1 \le j < g$. Therefore, one can reduce the precision by computing $U_p, U_{2p}, \ldots, U_{(g-1)p}$ and U_{gp-1} as follows.

In order to obtain the Hasse-Witt matrix over $\mathbb{F}_p$, it is enough to compute

$$U_{gp-1} = A(gp-1)\cdots A\left((g-1)p+1\right)U_{(g-1)p}$$

over $\mathbb{Z}/p\mathbb{Z}$. Moreover, since the division by p occurs at the computation in $U_{(g-1)p} = A((g-1)p)U_{(g-1)p-1}$, it is enough to compute

$$U_{(g-1)p} = A\left((g-1)p\right)\cdots A\left((g-2)p+1\right)U_{(g-2)p}$$

over $\mathbb{Z}/p^2\mathbb{Z}$. Similarly, it is enough to compute

$$U_{jp} = A\left(jp\right)\cdots A\left((j-1)p+1\right)U_{(j-1)p}$$

over $\mathbb{Z}/p^{g-j+1}\mathbb{Z}$ for $1 \le j < g$. Consequently, one can reduce the precision by computing the vectors such that

$$U_p = A(p)\cdots A(1)U_0 \text{ over } \mathbb{Z}/p^g\mathbb{Z},$$

$$U_{2p} = A(2p)\cdots A(p+1)U_p \text{ over } \mathbb{Z}/p^{g-1}\mathbb{Z},$$

$$\vdots$$

$$U_{gp-1} = A(gp-1)\cdots A\left((g-1)p+1\right)U_{(g-1)p} \text{ over } \mathbb{Z}/p\mathbb{Z}$$

for obtaining the Hasse-Witt matrix.

3.2 Using the reversal of $F(X)$

The precision of the p-adic integers in Section 3.1 can be further reduced by using the reversal of $F(X)$. This section shows a method to compute the Hasse-Witt matrix using the reversal and estimates the efficiency of the method.

Let the reversal [10, p. 254] of $F(X)$ denoted by

$$\mathrm{rev}(F(X)) := X^{\deg F}F(1/X).$$

Then, we can see that

$$\mathrm{rev}((F(X))^{\frac{p-1}{2}}) = (\mathrm{rev}(F(X)))^{\frac{p-1}{2}}. \qquad (6)$$

Therefore, the coefficients of the higher degree in $(F(X))^{\frac{p-1}{2}}$ can be obtained by computing the coefficients of the lower degree in $(\mathrm{rev}(F(X)))^{\frac{p-1}{2}}$ using a linear recurrence similar to (5).

Let $\widehat{A}(X)$ denote a $(2g+1)\times(2g+1)$ matrix with the components defined by (2) for $(\mathrm{rev}(F(X)))^{\frac{p-1}{2}}$ similar to (3). Let $\hat{h}_k$ denote the coefficient of X^k in $(\mathrm{rev}(F(X)))^{\frac{p-1}{2}}$ and a $(2g+1)$-dimensional column vector $\widehat{U}_k$ be

$$\widehat{U}_k = {}^t\begin{pmatrix} \hat{h}_{k-2g} & \cdots & \hat{h}_{k-1} & \hat{h}_k \end{pmatrix}$$

similar to (4). Since (6) implies

$$h_{\frac{(2g+1)(p-1)}{2}-i} = \hat{h}_i$$

for $0 \le i \le (2g+1)(p-1)/2$, one can obtain the g-th column components $h_{gp-1}, h_{gp-2}, \cdots$ and h_{gp-g} in the Hasse-Witt matrix by computing

$$\widehat{U}_{\frac{p-1}{2}} = {}^t\begin{pmatrix} \hat{h}_{\frac{p-1}{2}-2g} & \cdots & \hat{h}_{\frac{p-1}{2}-1} & \hat{h}_{\frac{p-1}{2}} \end{pmatrix}$$

$$= {}^t\begin{pmatrix} h_{gp+g} & \cdots & h_{gp-g+1} & h_{gp-g} \end{pmatrix}$$

from $\widehat{U}_0 = {}^t\begin{pmatrix} 0 & \cdots & 0 & f_{2g+1}^{\frac{p-1}{2}} \end{pmatrix}$ using the linear recurrence. Similarly, one can obtain the other components in the Hasse-Witt matrix by computing $\widehat{U}_{\frac{p-1}{2}+ip}$ for $0 < i < \lfloor g/2 \rfloor$. Consequently, one can obtain the Hasse-Witt matrix by computing

$$U_p, U_{2p}, \cdots, U_{\lceil \frac{g}{2} \rceil p-1}$$

$$\text{and} \quad \widehat{U}_{\frac{p-1}{2}}, \widehat{U}_{\frac{p-1}{2}+p}, \cdots, \widehat{U}_{\lfloor \frac{g}{2} \rfloor p-\frac{p+1}{2}},$$

instead of computing U_p, $\cdots$, $U_{(g-1)p}$, U_{gp-1}. Applying the result in Section 3.1 to these vectors, one can reduce the precision for obtaining the Hasse-Witt matrix. That is, one can compute

$$U_p \text{ over } \mathbb{Z}/p^{\lceil \frac{g}{2} \rceil}\mathbb{Z}, \cdots, U_{\lceil \frac{g}{2} \rceil p-1} \text{ over } \mathbb{Z}/p\mathbb{Z}$$

for $F(X)$ and

$$\widehat{U}_{\frac{p-1}{2}} \text{ over } \mathbb{Z}/p^{\lfloor \frac{g}{2} \rfloor}\mathbb{Z}, \cdots, \widehat{U}_{\lfloor \frac{g}{2} \rfloor p-\frac{p+1}{2}} \text{ over } \mathbb{Z}/p\mathbb{Z}$$

for $\mathrm{rev}(F(X))$.

In the following, we roughly estimate the efficiency of the proposed method for $g = 2$ and 3. Let $S(j)$ denote the cost in bit operations of computing U_{k+p} from U_k over $\mathbb{Z}/p^j\mathbb{Z}$ for any integer $k \ge 0$. In the following discussion, we assume that a multiplication in $\mathbb{Z}/p^j\mathbb{Z}$ costs mj^α for a constant m and a real number α with $1 < \alpha \le 2$.

In the case of $g = 2$, the proposed method computes U_{p-1} from U_0 over $\mathbb{Z}/p\mathbb{Z}$ within $S(1)$ bit operations and $\widehat{U}_{\frac{p-1}{2}}$ from $\widehat{U}_0$ over $\mathbb{Z}/p\mathbb{Z}$ within $S(1)$ bit operations. So, the proposed method needs about $2S(1)$ bit operations. On the other hand, the previous method in [3, 5] computes U_{p-1} from U_0 over $\mathbb{Z}/p^2\mathbb{Z}$ within $S(2)$ bit operations and U_{2p-1} from U_{p-1} over $\mathbb{Z}/p^2\mathbb{Z}$ within $S(2)$ bit operations. So, the previous method needs about $2S(2)$ bit operations in total. Consequently, the proposed method can compute the Hasse-Witt matrix $S(2)/S(1)$ times faster than the previous method. Assuming that $S(j)$ is dominated by multiplications in $\mathbb{Z}/p^j\mathbb{Z}$, we have $S(2)/S(1) = 2^\alpha/1^\alpha = 2^\alpha$. Therefore, we can expect that the proposed method is about 2 to 4 times faster than the previous method.

In the case of $g = 3$, the proposed method computes U_p from U_0 over $\mathbb{Z}/p^2\mathbb{Z}$ within $S(2)$ bit operations, U_{2p-1} from U_p over $\mathbb{Z}/p\mathbb{Z}$ within $S(1)$ bit operations and $\widehat{U}_{\frac{p-1}{2}}$ from $\widehat{U}_0$ over $\mathbb{Z}/p\mathbb{Z}$ within $S(1)$ bit operations. So, the proposed method needs about $S(2) + 2S(1)$ bit operations. On the other hand, the previous method computes U_{p-1} from U_0 over $\mathbb{Z}/p^3\mathbb{Z}$ within $S(3)$ bit operations, U_{2p-1} from U_{p-1} over $\mathbb{Z}/p^3\mathbb{Z}$ within $S(3)$ bit operations and U_{3p-1} from U_{2p-1} over $\mathbb{Z}/p^3\mathbb{Z}$ within $S(3)$ bit operations. So, the previous method needs about $3S(3)$ bit operations. Consequently, the proposed method can compute the Hasse-Witt matrix $3S(3)/(S(2) + 2S(1))$ times faster than the previous method. Assuming that $S(j)$ is dominated by multiplications in $\mathbb{Z}/p^j\mathbb{Z}$, we have $3S(3)/(S(2)+2S(1)) = 3\cdot 3^\alpha/(2^\alpha + 2\cdot 1^\alpha) = 3^{\alpha+1}/(2^\alpha + 2)$. Therefore, we can expect that the proposed method is about 2 to 4.5 times faster than the previous method.

4. Experimental results

This section shows experimental results of the computation of the Hasse-Witt matrices of hyperelliptic curves of genus 2 and 3 by using the proposed method in Section 3.2.

We implemented the algorithm shown in [3] using Magma V2.15-10 [11]. We computed the Hasse-Witt matrices of hyperelliptic curves defined over $\mathbb{F}_p$ for $16 \leq \log_2 p \leq 32$ by using the proposed method. We also computed the Hasse-Witt matrices with the fixed precision shown in [3] for comparison. These experiments were run on an AMD Opteron 246 2.0GHz.

Fig. 1 shows the result for hyperelliptic curves of genus 2. The vertical axis denotes time in seconds to compute the Hasse-Witt matrix, and the horizontal axis denotes bit length of p. In Fig. 1, "Original" denotes the time to compute the Hasse-Witt matrix by using the method in [3], and "This work" denotes the time to compute the Hasse-Witt matrix by using the proposed method in Section 3.2. Similarly, Fig. 2 shows the result for hyperelliptic curves of genus 3.

The result for genus 2 hyperelliptic curves shows that the proposed method can compute the Hasse-Witt matrix about 1.5 to 2.2 times faster than the previous method. The result for genus 3 hyperelliptic curves shows that the proposed method can compute the Hasse-Witt matrix about 1.6 to 2.0 times faster than the previous method. These results show that the proposed method more efficiently compute the Hasse-Witt matrices than the previous method.

However, the ratios of the proposed method to the previous method in the results are smaller than the estimations in Section 3.2. One of the reasons for the difference is that the cost of multiplications in $\mathbb{Z}/p^j\mathbb{Z}$ is strongly affected by word operations rather than bit operations, because the bit length of p is less than the size of a word on the CPU so that a multiplication in $\mathbb{Z}/p^j\mathbb{Z}$ is executed in a few words for $1 \leq j \leq 3$.

5. Conclusion

This paper proposes improvements in the computation of the Hasse-Witt matrix of a hyperelliptic curve using a linear recurrence. The proposed method uses the reversal of a polynomial in order to reduce the precision of the p-adic integers for computing the Hasse-Witt matrix, so that the proposed method can speed up the computation of the Hasse-Witt matrix. The experimental results show that the proposed method can compute the Hasse-Witt matrices of hyperelliptic curves of genus 2 about 1.5 to 2.2 times faster than the previous method [3] and the proposed method can compute the Hasse-Witt matrices of hyperelliptic curves of genus 3 about 1.6 to 2.0 times faster than the previous method.

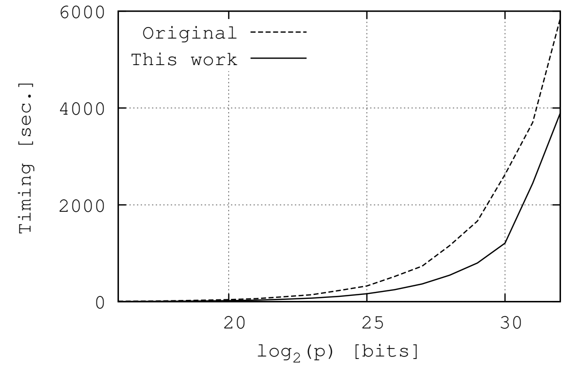

Fig. 1. Time for computing the Hasse-Witt matrices of hyperelliptic curves of genus 2 using Magma on AMD Opteron 246 2GHz.

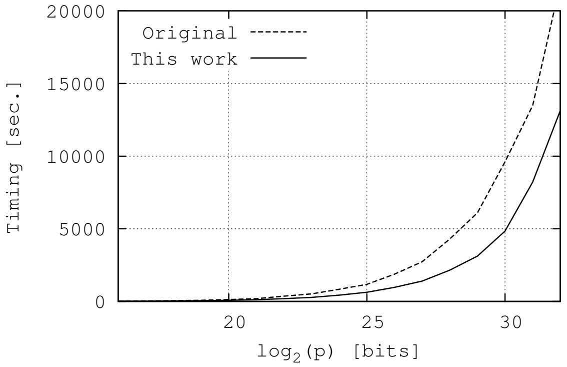

Fig. 2. Time for computing the Hasse-Witt matrices of hyperelliptic curves of genus 3 using Magma on AMD Opteron 246 2GHz.

References

[1] P. Gaudry and R. Harley, Counting points on hyperelliptic curves over finite fields, in: Proc. of the 4th Int. Symposium on Algorithmic Number Theory, W. Bosma ed., Lect. Notes Comput. Sci., Vol. 1838, pp. 313–332, Springer-Verlag, Berlin, 2000.

[2] K. Matsuo, J. Chao and S. Tsujii, Baby step giant step algorithms in point counting of hyperelliptic curves, IEICE Trans., **E86-A** (2003), 1127–1134.

[3] A. Bostan, P. Gaudry and É. Schost, Linear recurrences with polynomial coefficients and computation of the Cartier-Manin operator on hyperelliptic curves, in: Proc. of Finite Fields and Applications: 7th Int. Conf., Fq7, G. L. Mullen, A. Poli, and H. Stichtenoth, eds., Lect. Notes Comput. Sci., Vol. 2948, pp. 40–58, Springer-Verlag, Berlin, 2004.

[4] M. Bauer, E. Teske and A. Weng, Point counting on Picard curves in large characteristic, Math. Comp., **74** (2005), 1983–2005.

[5] A. Bostan, P. Gaudry and É. Schost, Linear recurrences with polynomial coefficients and application to integer factorization and Cartier-Manin operator, SIAM J. Comput., **36** (2007), 1777–1806.

[6] D. V. Chudnovsky and G. V. Chudnovsky, Approximations and complex multiplication according to Ramanujan, in: Ramanujan Revisited, pp. 375–472, Academic Press, Boston, 1988.

[7] J. I. Manin, The Hasse-Witt matrix of an algebraic curve, Trans. AMS, **45** (1965), 245–264.

[8] L. Euler, Introduction to Analysis of the Infinite, Book I (translation by J. D. Blanton), Springer-Verlag, New York, 1988.

[9] P. Flajolet and B. Salvy, The SIGSAM challenges: symbolic asymptotics in practice, ACM SIGSAM Bull., **31** (1997) 36–47.

[10] J. von zur Gathen and J. Gerhard, Modern Computer Algebra, 2nd Edition, Cambridge Univ. Press, Cambridge, 2003.

[11] The Magma computational algebra system, http://magma.maths.usyd.edu.au/magma/.