Extended Algorithm for Solving Underdefined Multivariate Quadratic Equations

Hiroyuki Miura[1], Yasufumi Hashimoto[2], and Tsuyoshi Takagi[3]

[1] Graduate School of Mathematics, Kyushu University, 744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan
[2] Department of Mathematical Sciences, University of the Ryukyus, 1, Senbaru, Nishihara, Okinawa 903-0213, Japan
[3] Institute of Mathematics for Industry, Kyushu University, 744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan

Abstract. It is well known that solving randomly chosen Multivariate Quadratic equations over a finite field (MQ-Problem) is NP-hard, and the security of Multivariate Public Key Cryptosystems (MPKCs) is based on the MQ-Problem. However, this problem can be solved efficiently when the number of unknowns n is sufficiently greater than that of equations m (This is called "Underdefined"). Indeed, the algorithm by Kipnis et al. (Eurocrypt'99) can solve the MQ-Problem over a finite field of even characteristic in a polynomial-time of n when $n \geq m(m+1)$. Therefore, it is important to estimate the hardness of the MQ-Problem to evaluate the security of Multivariate Public Key Cryptosystems. We propose an algorithm in this paper that can solve the MQ-Problem in a polynomial-time of n when $n \geq m(m+3)/2$, which has a wider applicable range than that by Kipnis et al. We will also compare our proposed algorithm with other known algorithms. Moreover, we implemented this algorithm with Magma and solved the MQ-Problem of $m = 28$ and $n = 504$, and it takes 78.7 seconds on a common PC.

Keywords: Multivariate Public Key Cryptosystems (MPKCs), Multivariate Quadratic Equations, MQ-Problem.

1 Introduction

Multivariate Public Key Cryptosystems (MPKCs) are cryptosystems whose security depends on the hardness of solving Multivariate Quadratic equations over a finite field (MQ-Problem). It is known that the MQ-Problem over a finite field is NP-hard [13] when the coefficients are randomly chosen, and no quantum algorithm efficiently solving the MQ-Problem has been presented. Therefore, MPKCs are one of candidates for post quantum cryptographies. For example, the Matsumoto-Imai cryptosystem [16], Hidden Field Equation (HFE) [18], Unbalanced Oil and Vinegar (UOV) [15], and Rainbow [7] are MPKCs. However, the MQ-Problem is efficiently solved under special n and m conditions. In particular, the algorithm by Kipnis et al. [15] can solve the MQ-Problem over a finite field of even characteristic in a polynomial-time of n when $n \geq m(m+1)$. It is also

P. Gaborit (Ed.): PQCrypto 2013, LNCS 7932, pp. 118–135, 2013.

known that the Gröbner basis algorithms [5,10,11] solve the MQ-Problem, and these algorithms are more effective in the Overdefined ($n \ll m$) MQ-Problem [1,2]. Thus, estimating the hardness of the MQ-Problem is important for the security of MPKCs.

The approach by Kipnis et al. diagonalizes the upper left $m \times m$ part of the coefficient matrices, solves linear equations, and reduces the MQ-Problem to find square roots over a finite field. Courtois et al. [6] and Hashimoto [14] modified this algorithm. Although the algorithm by Courtois et al. [6] has a much smaller applicable range, it can solve MQ-Problems over all the finite fields in polynomial-time. Hashimoto's algorithm presented a polynomial-time algorithm that solves those over all finite fields when $n \geq m^2 - 2m^{3/2} + 2m$, which extended the applicable range of that of Kipnis et al. [15]. However, we point out that Hashimoto's algorithm doesn't work efficiently due to some unsolved multivariate equations arisen from the linear transformation (See Appendix A). Recently, Thomae et al. [20] made n smaller than the algorithm by Kipnis et al. by using the Gröbner basis. This algorithm can be used when $n > m$, but it is an exponential-time algorithm.

We will present an algorithm in this paper solves the Underdefined ($n \gg m$) MQ-Problem in a polynomial-time when $n \geq m(m+3)/2$, which is wider than $n \geq m(m+1)$. Moreover, we implemented this algorithm on Magma [4] and solved an MQ-Problem with (n, m) which the algorithm by Kipnis et al. can't be used. We will compare these results with the algorithm by Kipnis et al. [15] and that by Courtois et al. [6].

2 MQ-Problem and Its Known Solutions

In this section we introduce the MQ-Problem and explain some algorithms to solve the Underdefined MQ-Problems.

2.1 MQ-Problem

Let q be a power of prime and k be a finite field of order q. For integers $n, m \geq 1$, denoted by $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ quadratic polynomials of $\boldsymbol{x} = {}^t(x_1, x_2, \ldots, x_n)$ over k.

$$
\begin{aligned}
f_1(x_1, ..., x_n) &= \sum_{1 \leq i \leq j \leq n} a_{1,i,j} x_i x_j + \sum_{1 \leq i \leq n} b_{1,i} x_i + c_1 \\
f_2(x_1, ..., x_n) &= \sum_{1 \leq i \leq j \leq n} a_{2,i,j} x_i x_j + \sum_{1 \leq i \leq n} b_{2,i} x_i + c_2 \\
&\vdots \\
f_m(x_1, ..., x_n) &= \sum_{1 \leq i \leq j \leq n} a_{m,i,j} x_i x_j + \sum_{1 \leq i \leq n} b_{m,i} x_i + c_m,
\end{aligned}
$$

where $a_{l,i,j}, b_{l,i}, c_l \in k; l = 1, ..., m$. We call it "the MQ-Problem of m equations and n unknowns over finite field k", that the problem tries to find one solution $(x_1, \ldots, x_n) \in k^n$ such that $f_i(x_1, ..., x_n) = 0$ for all $i = 1, \ldots, m$ among the many ones that exist.

2.2 Kipnis-Patarin-Goubin's Algorithm

We explain Kipnis-Patarin-Goubin's Algorithm [15].

Let $n, m \geq 1$ be integers with $n \geq m(m+1)$ and $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ be the quadratic polynomials of $\boldsymbol{x} = {}^t(x_1, x_2, \ldots, x_n)$ over k. Our goal is to find a solution $x_1, x_2, \ldots, x_n$ such that $f_1(\boldsymbol{x}) = 0, f_2(\boldsymbol{x}) = 0, \ldots, f_m(\boldsymbol{x}) = 0$. For $i = 1, \ldots, n$ the polynomials $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ are denoted by

$$f_i(x_1, x_2, \ldots, x_n) = {}^t\boldsymbol{x} F_i \boldsymbol{x} + (\text{linear.})$$

where $F_1, \ldots, F_m$ are $n \times n$ matrices over k.

We also use an $n \times n$ matrix T_t over k to transform all the unknowns, and T_t has the following form.

$$T_t = \begin{pmatrix} 1 & 0 & \cdots & 0 & a_{1,t} & 0 & \cdots & \cdots & 0 \\ 0 & 1 & \ddots & \vdots & a_{2,t} & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & \vdots & & & \vdots \\ \vdots & & \ddots & 1 & a_{t-1,t} & \vdots & & & \vdots \\ \vdots & & & 0 & 1 & 0 & & & \vdots \\ \vdots & & & \vdots & a_{t+1,t} & 1 & \ddots & & \vdots \\ \vdots & & & \vdots & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & a_{n,t} & 0 & \cdots & 0 & 1 \end{pmatrix} \tag{1}$$

where $a_{1,t}, \ldots, a_{t-1,t}, a_{t+1,t}, \ldots, a_{n,t} \in k$.

We want to obtain quadratic equations of the following form.

$$\begin{cases} \sum_{i=1}^{m} \beta_{1,i} x_i^2 - \lambda_1 = 0 \\ \qquad \vdots \\ \sum_{i=1}^{m} \beta_{m,i} x_i^2 - \lambda_m = 0, \end{cases} \tag{2}$$

where $\beta_{l,i}$ and $\lambda_l \in k$ $(l = 1, \ldots, m)$.

Step 1. Transform $\boldsymbol{x} \mapsto T_2 \boldsymbol{x}$ so that the coefficients of $x_1 x_2$ in f_j $(j = 1, \ldots, m)$ are zero.

$$F_j \mapsto \left(\begin{array}{cc|c} * & 0 & \\ \hline 0 & * & \\ \hline & & * \end{array}\right) \quad (j = 1, \ldots, m)$$

Step 2. Transform $\boldsymbol{x} \mapsto T_3\boldsymbol{x}$ so that the coefficients of x_1x_3, x_2x_3 in f_j $(j = 1,\ldots,m)$ are zero.

$$\left(\begin{array}{cc|c} * & 0 & \\ 0 & * & \\ \hline & & * \end{array}\right) \mapsto \left(\begin{array}{ccc|c} * & 0 & 0 & \\ 0 & * & 0 & \\ 0 & 0 & * & \\ \hline & & & * \end{array}\right)$$

$$\vdots$$

(We continue similar operations to "**Step $\boldsymbol{m-1}$.**".)

From "**Step 1.**" to "**Step $\boldsymbol{m-1}$.**", we require the condition $n-1 \geq m(m-1)$, i.e., $n \geq m^2 - m + 1$.

Then we can obtain the coefficient matrices of the form

$$\left(\begin{array}{ccc|c} * & & 0 & \\ & \ddots & & \\ 0 & & * & \\ \hline & & & * \end{array}\right)$$

for each $i = 1,\ldots,m$, and the following quadratic equations.

$$\begin{cases} \sum_{i=1}^{m} \beta_{1,i}x_i^2 + \sum_{i=1}^{m} x_iL_{1,i}(x_{m+1},\ldots,x_n) + Q_1(x_{m+1},\ldots,x_n) = 0 \\ \vdots \\ \sum_{i=1}^{m} \beta_{m,i}x_i^2 + \sum_{i=1}^{m} x_iL_{m,i}(x_{m+1},\ldots,x_n) + Q_m(x_{m+1},\ldots,x_n) = 0 \end{cases} \tag{3}$$

where L's are linear polynomials and Q's are quadratic polynomials in these variables.

Step $\boldsymbol{m}$. Solve linear equations $\{L_{i,j}(x_{m+1},\ldots,x_n) = 0\}$ for $i = 1,\ldots,m$, and $j = 1,\ldots,m$, and substitute the solutions $x_{m+1},\ldots,x_n$ into (3). This system of linear equations has $n-m$ unknowns and m^2 equations, so we can solve if n and m satisfy $n - m \geq m^2$ i.e. $n \geq m(m+1)$. Finally, we obtain quadratic equations of the form (2). Then we can compute the $x_1^2,\ldots,x_m^2$ values easily. The complexity of this algorithm is

$$\begin{cases} O(n^w m(\log q)^2) & (\text{char } k \text{ is } 2) \\ O(2^m n^w m(\log q)^2) & (\text{char } k \text{ is odd}), \end{cases}$$

where $2 \leq w \leq 3$ is the exponent of the Gaussian elimination. This is because this algorithm computes $n \times n$ matrices over finite field $k = \mathrm{GF}(q)$ and solves linear equations to obtain the $x_1^2,\ldots,x_m^2$ values. The complexity of these operations is $O(n^w(\log q)^2)$. When the characteristic of k is odd, the probability of existence of square roots is approximately $1/2$, and we can find a solution with probability of 2^{-m}. Therefore, when the characteristic of k is odd, the complexity of this algorithm is $O(2^m n^w(\log q)^2)$.

2.3 Courtois et al.'s Algorithm

Courtois et al. proposed an algorithm [6] which extend Kipnis-Patarin-Goubin's algorithm when char k is odd, and this algorithm can be applied when the number of equations m and the number of unknowns n satisfy $n \geq 2^{\frac{m}{7}}(m+1)$. This algorithm and Kipnis-Patarin-Goubin's algorithm are very similar until obtain quadratic equations of the form (2). Main idea of this algorithm is to reduce the number of equations and unknowns after they obtain the quadratic equations of the form (2). This algorithm can solve the MQ-Problem of m equations and n unknowns over k in time about $2^{40}(40 + 40/\log q)^{m/40}$.

2.4 Thomae et al.'s Algorithm

Thomae et al. proposed an algorithm [20] which extend Kipnis-Patarin-Goubin's algorithm, and this algorithm can be applied when the number of equations m and the number of unknowns n satisfy $n > m$. Main idea of this algorithm is to make more zero part by using more linear transformations than Kipnis-Patarin-Goubin's algorithm in order to reduce the number of equations and unknowns. This algorithm reduces the MQ-Problem of m equations and n unknowns over finite field k into the MQ-Problem of $(m - \lfloor n/m \rfloor)$ equations and $(m - \lfloor n/m \rfloor)$ unknowns over finite field k. Then this algorithm uses Gröbner basis algorithm, so the complexty of this algorithm exponential-time. Thomae et al. [20] claimed that the MQ-Problem of 28 equations and 84 unknowns over $\mathrm{GF}(2^8)$ has 80-bit security.

3 Proposed Algorithm

We propose an algorithm in this section that solves the MQ-Problem with $n \geq m(m+3)/2$, and explain the analysis of this algorithm.

3.1 Proposed Algorithm

Let $n, m \geq 1$ be integers with $n \geq m(m+3)/2$ and $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ be the quadratic polynomials of $\boldsymbol{x} = {}^t(x_1, x_2, \ldots, x_n)$ over k. Our goal is to find a solution $x_1, x_2, \ldots, x_n$ such that $f_1(\boldsymbol{x}) = 0, f_2(\boldsymbol{x}) = 0, \ldots, f_m(\boldsymbol{x}) = 0$. For $i = 1, \ldots, n$ the polynomials $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ are denoted by

$$f_i(x_1, x_2, \ldots, x_n) = {}^t\boldsymbol{x} F_i \boldsymbol{x} + (\text{linear.})$$

where $F_1, \ldots, F_m$ are $n \times n$ matrices over kD We also use an $n \times n$ matrix T_t over k of the form (1) to transform all the unknowns in "**Step *t*.**" $(t = 2, \ldots, m)$.

Step 1. Choose $c_i^{(1)} \in k$ $(i = 1, \ldots, m-1)$ so that the $(1,1)$-elements of $F_i - c_i^{(1)} F_m$ are zero, and replace F_i with $F_i - c_i^{(1)} F_m$. If the $(1,1)$-element of F_m is zero, exchange F_m for one of $F_1, \ldots, F_{m-1}$ that satisfies the $(1,1)$-element is not zero.

$$F_1, F_2, \ldots, F_m \mapsto \underbrace{\begin{pmatrix} 0 & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} 0 & \\ & \ast \end{pmatrix}}_{m-1}, \begin{pmatrix} \ast & \\ & \ast \end{pmatrix}$$

Step 2. (i) Transform $\boldsymbol{x}$ to $T_2\boldsymbol{x}$ so that the coefficients of x_1x_2 in $f_1, f_2, \ldots, f_m$ are zero.

$$\underbrace{\begin{pmatrix} 0 & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} 0 & \\ & \ast \end{pmatrix}}_{m-1}, \begin{pmatrix} \ast & \\ & \ast \end{pmatrix} \mapsto \underbrace{\begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}}_{m-1}, \begin{pmatrix} \begin{array}{c|c} \ast & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}$$

After the linear transformation $\boldsymbol{x} \mapsto T_2\boldsymbol{x}$, the coefficient matrices are denoted as

$${}^tT_2F_iT_2 \ (i = 1, 2, \ldots, m).$$

We determine the $a_{1,2}, a_{3,2}, \ldots, a_{n,2}$ values in T_2 by solving the linear equations of coefficients we want to make zero. Note that (1,2)-elements and (2,1)-elements of F_i are not always equal to zero. The picture means that sum of (1,2)-element and (2,1)-element of F_i is equal to zero for each $i = 1, \ldots, m$.

(ii) Choose $c_i^{(2)} \in k$ $(i = 1, \ldots, m-2)$ so that the $(2,2)$-elements of $F_i - c_i^{(2)}F_{m-1}$ are zero, and replace F_i with $F_i - c_i^{(2)}F_{m-1}$. If the $(2,2)$-element of F_{m-1} is zero, exchange F_{m-1} for one of $F_1, \ldots, F_{m-2}$ that satisfies the $(2,2)$-element is not zero.

$$\underbrace{\begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}}_{m-1}, \begin{pmatrix} \begin{array}{c|c} \ast & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}$$

$$\mapsto \underbrace{\begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array} & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array} & \\ & \ast \end{pmatrix}}_{m-2}, \begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}, \begin{pmatrix} \begin{array}{c|c} \ast & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}$$

Step 3. (i) Transform $\boldsymbol{x}$ to $T_3\boldsymbol{x}$ so that the coefficients of x_1x_3 and x_2x_3 in $f_1, f_2, \ldots, f_{m-1}$ and the coefficient of x_1x_3 in f_m are zero.

$$\underbrace{\begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array} & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & 0 \end{array} & \\ & \ast \end{pmatrix}}_{m-2}, \begin{pmatrix} \begin{array}{c|c} 0 & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}, \begin{pmatrix} \begin{array}{c|c} \ast & 0 \\ \hline 0 & \ast \end{array} & \\ & \ast \end{pmatrix}$$

$$\mapsto \underbrace{\begin{pmatrix} \begin{array}{c|c|c} 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & \ast \end{array} & \\ & \ast \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c|c} 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & \ast \end{array} & \\ & \ast \end{pmatrix}}_{m-2}, \begin{pmatrix} \begin{array}{c|c|c} 0 & 0 & 0 \\ \hline 0 & \ast & 0 \\ \hline 0 & 0 & \ast \end{array} & \\ & \ast \end{pmatrix}, \begin{pmatrix} \begin{array}{c|c|c} \ast & 0 & 0 \\ \hline 0 & \ast & \ast \\ \hline 0 & \ast & \ast \end{array} & \\ & \ast \end{pmatrix}$$

(ii) Choose $c_i^{(3)} \in k$ $(i = 1, \ldots, m-3)$ so that the $(3,3)$-elements of $F_i - c_i^{(3)} F_{m-2}$ are zero, and replace F_i with $F_i - c_i^{(3)} F_{m-2}$. If the $(3,3)$-element of F_{m-2} is zero, exchange F_{m-2} for one of $F_1, \ldots, F_{m-3}$ that satisfies the $(3,3)$-element is not zero.

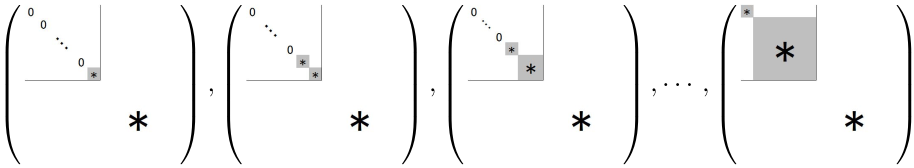
$$\underbrace{\begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&0&0\\ \hline 0&0&*\\ \hline \end{array} & \\ & * \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&0&0\\ \hline 0&0&*\\ \hline \end{array} & \\ & * \end{pmatrix}}_{m-2}, \begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&*&0\\ \hline 0&0&*\\ \hline \end{array} & \\ & * \end{pmatrix}, \begin{pmatrix} \begin{array}{c|c|c|} *&0&0\\ \hline 0&*&*\\ \hline 0&*&*\\ \hline \end{array} & \\ & * \end{pmatrix} \mapsto$$

$$\underbrace{\begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&0&0\\ \hline 0&0&0\\ \hline \end{array} & \\ & * \end{pmatrix}, \ldots, \begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&0&0\\ \hline 0&0&0\\ \hline \end{array} & \\ & * \end{pmatrix}}_{m-3}, \begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&0&0\\ \hline 0&0&*\\ \hline \end{array} & \\ & * \end{pmatrix}, \begin{pmatrix} \begin{array}{c|c|c|} 0&0&0\\ \hline 0&*&0\\ \hline 0&0&*\\ \hline \end{array} & \\ & * \end{pmatrix}, \begin{pmatrix} \begin{array}{c|c|c|} *&0&0\\ \hline 0&*&*\\ \hline 0&*&*\\ \hline \end{array} & \\ & * \end{pmatrix}$$

$$\vdots$$

(We continue similar operations to "**Step *m*.**".)

Then we can obtain the coefficient matrices of the form



for each $i = 1, \ldots, m$, and the following quadratic equations.

$$\begin{cases} x_m^2 + \sum_{1 \le i \le m} x_i L_{1,i}(x_{m+1}, \ldots, x_n) + Q_{1,2}(x_{m+1}, \ldots, x_n) = 0 \\ x_{m-1}^2 + x_m^2 + \sum_{1 \le i \le m} x_i L_{2,i}(x_{m+1}, \ldots, x_n) + Q_{2,2}(x_{m+1}, \ldots, x_n) = 0 \\ x_{m-2}^2 + Q_{3,1}(x_{m-1}, x_m) + \sum_{1 \le i \le m} x_i L_{3,i}(x_{m+1}, \ldots, x_n) + Q_{3,2}(x_{m+1}, \ldots, x_n) = 0 \\ \vdots \\ x_1^2 + Q_{m,1}(x_2, \ldots, x_m) + \sum_{1 \le i \le m} x_i L_{m,i}(x_{m+1}, \ldots, x_n) + Q_{m,2}(x_{m+1}, \ldots, x_n) = 0 \end{cases} \tag{4}$$

where L's are linear polynomials and Q's are quadratic polynomials in these variables.

Step $m+1$. Solve linear equations $\{L_{i,j}(x_{m+1}, \ldots, x_n) = 0\}$ of $x_{m+1}, \ldots, x_n$ for $i = 1, \ldots, m$ and $j = 1, \ldots, m-i+1$, and substitute the solutions $x_{m+1}, \ldots, x_n$ into (4). If there exists $t = 1, \ldots, m$ such that the (t,t)-elements of $F_1, \ldots, F_{m-t+1}$ are zero, remove $L_{m-t+1,t} = 0$ from the linear systems and choose the $x_{m+1}, \ldots, x_n$ that satisfies $L_{m-t+1,t} \neq 0$.

Finally, we obtain the following quadratic equations.

$$\begin{cases} x_m^2 - \lambda_1 = 0 \\ x_{m-1}^2 + \tilde{Q}_2(x_m) - \lambda_2 = 0 \\ x_{m-2}^2 + \tilde{Q}_3(x_{m-1}, x_m) - \lambda_3 = 0 \\ \qquad\qquad\vdots \\ x_2^2 + \tilde{Q}_{m-1}(x_3, \dots, x_m) - \lambda_{m-1} = 0 \\ x_1^2 + \tilde{Q}_m(x_2, \dots, x_m) - \lambda_m = 0 \end{cases}$$

where $\lambda_1, \dots, \lambda_m \in k$ and $\tilde{Q}$'s are quadratic polynomials in these variables.

We can find a solution for the quadratic equations in the following way. First, we solve the first equation and substitute the solution x_m into the others. Next, we solve the second equation and substitute the solution x_{m-1} into the remaining equations $\cdots$. If there exists $t = 1, \dots, m$ such that (t,t)-elements of $F_1, \dots, F_{m-t+1}$ are zero, the $(m-t+1)$-th equation takes the form of $x_t + Q(x_{t+1}, \dots, x_m) - \lambda_{m-t+1} = 0$.

3.2 Analysis of Proposed Algorithm

We will explain the required conditions and complexity of the proposed algorithm in this section.

Theorem 3.1. The proposed algorithm works when $n \geq m(m+3)/2$.

Proof. Our algorithm works if we can solve the linear equations.

In "**Step *t*.**" $(t = 2, \dots, m)$, the number of linear equations to be solved is

$$(m-t+1)(t-1) + \sum_{i=1}^{t-1} i = -\frac{1}{2}\left\{t - \left(m + \frac{3}{2}\right)\right\}^2 + \frac{1}{2}m^2 + \frac{1}{2}m + \frac{1}{8},$$

and the number of unknowns is $n-1$. Thus, we require $n \geq m(m+1)/2$ until "**Step *m*.**".

In "**Step *m* + 1.**", the number of linear equations to be solved is

$$\sum_{t=1}^{m}(m-t+1) = \frac{1}{2}m(m+1),$$

and the number of unknowns is $n-m$. Thus, we require $n \geq m(m+3)/2$.

For these reasons, we found that the proposed algorithm can be applied when $n \geq m(m+3)/2$. □

Lemma 3.2. For $n = m(m+3)/2$, the proposed algorithm succeeds in finding a solution of the MQ-Problem of m equations, n unknowns with probability of approximately

$$\begin{cases} 1 - q^{-1} & (\text{char } k \text{ is } 2) \\ 2^{-m}(1 - q^{-1}) & (\text{char } k \text{ is odd}). \end{cases}$$

Proof. When $n = m(m+3)/2$, we must solve the linear equations that are not underdefined in "**Step $\boldsymbol{m+1}$.**". Then, we fail to solve linear equations with probability of q^{-1}. When the characteristic of k is odd, the probability of existence of square roots over k is approximately 1/2. Therefore, the success probability of this algorithm is approximately $2^{-m}(1-q^{-1})$ when the characteristic of k is odd. □

Moreover, the proposed algorithm uses only $n \times n$ matrix operations and the calculation of square roots over finite field k, so we obtain the following result concerning the complexity of the proposed algorithm.

Theorem 3.3. The complexity of the proposed algorithm is

$$\begin{cases} O(n^w m (\log q)^2) & (\text{char } k \text{ is } 2) \\ O(2^m n^w m (\log q)^2) & (\text{char } k \text{ is odd}), \end{cases}$$

where $2 \leq w \leq 3$ is the exponent of the Gaussian elimination.

Proof. In this algorithm, we calculate $n \times n$ matrices over finite field $k = \mathrm{GF}(q)$ for about m times. The complexity of this operation is $O(n^w (\log q)^2)$. When the characteristic of k is odd, the probability of existence of square roots is approximately 1/2, and we can find a solution with probability of 2^{-m}. Therefore, when the characteristic of k is odd, the complexity of this algorithm is $O(2^m n^w m (\log q)^2)$. □

4 Implementations

We implemented the proposed algorithm using Magma [4], and compare the proposed algorithm and other known algorithms in this section. The results depend on the characteristic of k, and we will explain two cases, when the characteristic of k is 2 and an odd prime.

4.1 Parameters and Computational Environments

We chose the n and m parameters in which other algorithms can't be applied, and used homogeneous quadratic polynomials to experiment. We also chose $m = 28$ the same as Thomae et al. [20], and n so that the proposed algorithm can apply. The computer specification and software are listed in Table 1.

Table 1. Computer specifications

OS	CPU	RAM	Software
Windows 7 (64bit)	Intel Core i3 (1.33GHz)	4.00 GB	Magma V2.17-9

4.2 When char k Is 2

These algorithms have the same complexity $O(n^w m(\log q)^2)$, but the proposed algorithm has a wider applicable range than the others. The applicable ranges of the algorithms are drawn in Fig. 1.

Table 2. Applicable ranges of the proposed algorithm and other known algorithms (char k is 2)

	Applicable range	Complexity
Proposed	$n \geq m(m+3)/2$	(poly.)
Kipnis et al. [15]	$n \geq m(m+1)$	(poly.)
Courtois et al. [6]	$n \geq m(m+1)$	(poly.)

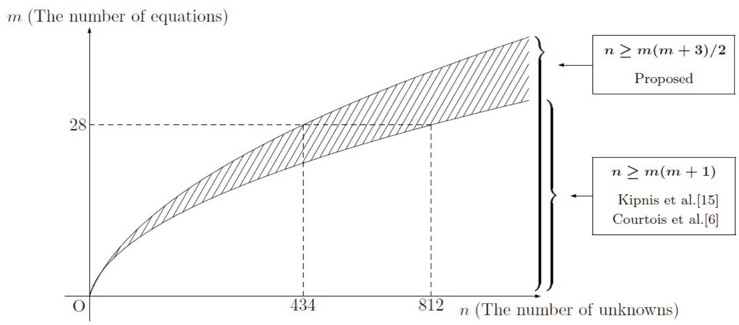


Fig. 1. Applicable range of proposed algorithm and other known algorithms

When $m = 28$, we can reduce the number of unknowns n from 812 to 434. The experimental results in our implementation are in Table. 3.

Table 3. Experimental results (char k is 2)

Field	n	m	Time / a try	Success probability
GF(2^8)	16	4	8.76 (msec.)	99.99 %
	84	11	506.83 (msec.)	100.0 %
	504	28	78.71 (sec.)	100.0 %

4.3 When char k Is Odd

We consider the algorithms by Courtois et al. [6]. Although the former one is polynomial-time of n, but it is not practical because the applicable range is too small. Thus, we compare the proposed algorithm and the latter one by Courtois et al. These algorithms are exponential-time. The applicable ranges of the algorithms are drawn in Fig. 2.

Table 4. Applicable ranges of the proposed algorithm and other known algorithms (char k is odd)

	Applicable range	Complexity
Proposed	$n \geq m(m+3)/2$	(exp.)
Courtois et al. [6]	$n \geq 2^{\frac{m}{7}} m(m+1)$	(poly.)
	$n \geq 2^{\frac{m}{7}} (m+1)$	(exp.)

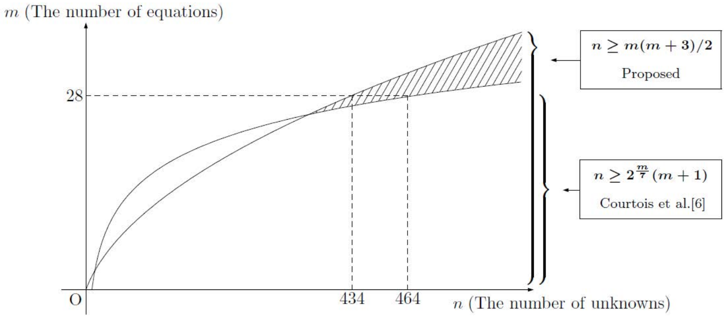


Fig. 2. Applicable range of proposed algorithm and algorithm by Courtois et al.

If $m \geq 27$, we can reduce the number of unknowns n to smaller than that of the algorithm by Courtois et al. The experimental results in our implementation are in Table. 5.

Table 5. Experimental results (char k is odd)

Field	n	m	Time / a try	Success probability
GF(7)	16	4	3.99 (msec.)	11.83 %
	84	11	259.28 (msec.)	0.22 %
	434	28	39.99 (sec.)	0.00 %

The proposed algorithm succeeds in solving the MQ-Problem with probability of 11.83% when $n = 16$ and $m = 4$, and 0.22% when $n = 84$ and $m = 11$. These results follow our success probability estimation, and we can get a similar result when $n = 434$ and $m = 28$ which can't use the algorithm by Courtois et al., and then, the success probability is $(4/7)^{28} \times (6/7) \approx 10^{-6.87} \approx 2^{-22.83}$. We estimate that it takes 1-core PC 9.44 years to solve the MQ-Problem of $n = 434$ and $m = 28$.

5 Conclusion

We presented an algorithm in this paper that can solve the MQ-Problem when $n \geq m(m+3)/2$, where n is the number of unknowns and m is the number of equations. This algorithm makes the range of solvable MQ-Problems wider than that by Kipnis et al. Moreover, we compared this algorithm and other known algorithms, and found that the proposed algorithm is easier to use than the others. In order to demonstrate the effectiveness of the proposed algorithm we implemented it using Magma on a PC. We were able to solve the MQ-Problem of $m = 28$ and n = 504 in 78.7 seconds.

Two open problems remain. The first is to make the applicable range wider and the second is to apply the proposed algorithm to the algorithm developed by Thomae et al. [20].

References

1. Bardet, M., Faugère, J.-C., Salvy, B., Yang, B.-Y.: Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems, MEGA 2005 (2005), http://www-polsys.lip6.fr/~jcf/Papers/BFS05b.pdf
2. Bettale, L., Faugère, J.-C., Perret, L.: Hybrid Approach for Solving Multivariate Systems over Finite Fields. Journal of Mathematical Cryptology 3, 177–197 (2009)
3. Braeken, A., Wolf, C., Preneel, B.: A Study of the Security of Unbalanced Oil and Vinegar Signature Schemes. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 29–43. Springer, Heidelberg (2005)
4. Computational Algebra Group, University of Sydney. The MAGMA Computational Algebra System for Algebra, Number Theory, and Geometry
5. Courtois, N.T., Klimov, A.B., Patarin, J., Shamir, A.: Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 392–407. Springer, Heidelberg (2000), http://www.minrank.org/xlfull.pdf
6. Courtois, N.T., Goubin, L., Meier, W., Tacier, J.-D.: Solving Underdefined Systems of Multivariate Quadratic Equations. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 211–227. Springer, Heidelberg (2002)
7. Ding, J., Schmidt, D.: Rainbow, a New Multivariate Polynomial Signature Scheme. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 164–175. Springer, Heidelberg (2005)
8. Ding, J., Gower, J.E., Schmidt, D.S.: Multivariate Public Key Cryptosystems. Springer (2006)

9. Ding, J., Hodges, T.J.: Inverting HFE Systems Is Quasi-Polynomial for All Fields. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 724–742. Springer, Heidelberg (2011)
10. Faugère, J.-C.: "A New Efficient Algorithm for Computing Gröbner bases (F_4)". Journal of Pure and Applied Algebra 139, 61–88 (1999)
11. Faugère, J.-C.: A New Efficient Algorithm for Computing Gröbner bases without reduction to zero (F_5). In: Proceedings of ISSAC 2002, pp. 75–83. ACM Press (2002)
12. Faugère, J.-C., Perret, L.: On the Security of UOV. In: Proceedings of SCC 2008, pp. 103–109 (2008)
13. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H.Freeman (1979)
14. Hashimoto, Y.: Algorithms to Solve Massively Under-Defined Systems of Multivariate Quadratic Equations. IEICE Trans. Fundamentals E94-A(6), 1257–1262 (2011)
15. Kipnis, A., Patarin, J., Goubin, L.: Unbalanced Oil and Vinegar Signature Schemes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 206–222. Springer, Heidelberg (1999)
16. Matsumoto, T., Imai, H.: Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 419–453. Springer, Heidelberg (1988)
17. Patarin, J.: Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt '88. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 248–261. Springer, Heidelberg (1995)
18. Patarin, J.: Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 33–48. Springer, Heidelberg (1996)
19. Shor, P.W.: Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In: Proceedings of 35th Annual Symposium on Foundations of Computer Science, pp. 124–134. IEEE Computer Society Press (1994)
20. Thomae, E., Wolf, C.: Solving Underdetermined Systems of Multivariate Quadratic Equations Revisited. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 156–171. Springer, Heidelberg (2012)

Appendix A: Hashimoto's Algorithm

In this appendix we explain Hashimoto's algorithm [14], which claimed that the MQ-Problem of $n \geq m^2 - 2m^{3/2} + 2m$ over all finite fields can be solved in a polynomial-time. The applicable range of Hashimoto's algorithm is wider than that of the algorithm by Kipnis et al. [15]. However, we point out that Hashimoto's algorithm doesn't work efficiently due to some unsolved multivariate equations arisen from the linear transformation.

A.1 Outline

In the following we describe Hashimoto's algorithm which consists of Algorithm A and Algorithm B.

Algorithm A
Let $g(\boldsymbol{x})$ be a quadratic form of unknowns $\boldsymbol{x} = {}^t(x_1, \ldots, x_n)$ over finite field k. We transform $\boldsymbol{x}$ by a linear matrix $U \in k^{n \times n}$. For $a_{2,1}, a_{3,1}, a_{3,2}, \ldots, a_{n,n-1} \in k$ we define U as follows :

$$U = \begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 \\ a_{2,1} & 1 & 0 & & & \vdots \\ a_{3,1} & a_{3,2} & 1 & \ddots & & \vdots \\ 0 & 0 & a_{4,3} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 1 & 0 \\ 0 & 0 & \cdots & 0 & a_{n,n-1} & 1 \end{pmatrix}.$$

We determine the linear transformation U such that the coefficients of x_1^2, x_1x_2, $x_1x_3, \ldots, x_1x_{n-1}$ in $g(U\boldsymbol{x})$ are all zero in the following way.
Step 1. Calculate $a_{2,1}, a_{3,1}$ such that the coefficient of x_1^2 in $g(U\boldsymbol{x})$ is zero.
Step 2. Calculate $a_{3,2}$ such that the coefficient of x_1x_2 in $g(U\boldsymbol{x})$ is zero.
Step 3. Calculate $a_{4,3}$ such that the coefficient of x_1x_3 in $g(U\boldsymbol{x})$ is zero.

$$\vdots$$

Step $n-1$. Calculate $a_{n,n-1}$ such that the coefficient of x_1x_{n-1} in $g(U\boldsymbol{x})$ is zero.

Algorithm B
Let $n, L, M \geq 1$ be integers that satisfy the following condition :

$$n \geq \begin{cases} 2L & (M = 1) \\ ML - M + L & (1 < M < L) \\ L^2 + 1 & (M = L) \end{cases}. \tag{5}$$

Let $g_1(\boldsymbol{x}), \ldots, g_M(\boldsymbol{x})$ be quadratic forms of $\boldsymbol{x}$ over k such that the coefficients of x_ix_j $(1 \leq i, j \leq L)$ in $g_1(\boldsymbol{x}), \ldots, g_{M-1}(\boldsymbol{x})$ are all zero. Then we can find an invertible linear transformation U such that the coefficients of $x_i x_j$ $(1 \le i, j \le L)$ in $g_1(U\boldsymbol{x}), \ldots, g_M(U\boldsymbol{x})$ are all zero.

$$\underbrace{{}^t\boldsymbol{x}\begin{pmatrix} O_L & * \\ * & * \end{pmatrix}\boldsymbol{x}, \ldots, {}^t\boldsymbol{x}\begin{pmatrix} O_L & * \\ * & * \end{pmatrix}\boldsymbol{x}}_{M-1}, {}^t\boldsymbol{x}\begin{pmatrix} * & * \\ * & * \end{pmatrix}\boldsymbol{x} \mapsto \underbrace{{}^t\boldsymbol{x}\begin{pmatrix} O_L & * \\ * & * \end{pmatrix}\boldsymbol{x}, \ldots, {}^t\boldsymbol{x}\begin{pmatrix} O_L & * \\ * & * \end{pmatrix}\boldsymbol{x}}_{M}$$

where O_L is $L \times L$ zero matrix. **Step 1.** (i) Using Algorithm A, find a transformation $T_{1,1}$ such that the coefficients of $x_1 x_j$ $(j = 1, \ldots, L-1)$ in $g_M(\boldsymbol{x})$ are zero, and transform $\boldsymbol{x} \mapsto T_{1,1}\boldsymbol{x}$.
(ii) Transform $\boldsymbol{x} \mapsto T_{2,1}\boldsymbol{x}$ such that the coefficients of $x_1 x_L$ in $g_M(\boldsymbol{x})$ and $x_i x_L$ $(i = 1, \ldots, L)$ in $g_1(\boldsymbol{x}), \ldots, g_{M-1}(\boldsymbol{x})$ are all zero.
Step 2. (i) Using Algorithm A, find a transformation $T_{1,2}$ such that the coefficients of $x_2 x_j$ $(j = 2, \ldots, L-1)$ in $g_M(\boldsymbol{x})$ are all zero, and transform $\boldsymbol{x} \mapsto T_{1,2}\boldsymbol{x}$.
(ii) Transform $\boldsymbol{x} \mapsto T_{2,2}\boldsymbol{x}$ such that the coefficients of $x_2 x_L$ in $g_M(\boldsymbol{x})$ and $x_i x_L$ $(i = 2, \ldots, L)$ in $g_1(\boldsymbol{x}), \ldots, g_{M-1}(\boldsymbol{x})$ are all zero.

$$\vdots$$

(We continue similar operations to "**Step $\boldsymbol{L-1}$.**".)

In "**Step $\boldsymbol{t}$.**-(i), (ii)" $(t = 1, \ldots, L-1)$, we use $n \times n$ matrices $T_{1,t}, T_{2,t}$ which have the following form :

$$T_{1,t} = \begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ a^{(t)}_{2,1} & 1 & \ddots & & & \vdots \\ a^{(t)}_{3,1} & a^{(t)}_{3,2} & 1 & \ddots & & \vdots \\ 0 & 0 & a^{(t)}_{4,3} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 1 & 0 \\ 0 & 0 & \cdots & 0 & a^{(t)}_{n,n-1} & 1 \end{pmatrix}, \; T_{2,t} = \begin{pmatrix} 1 & 0 & \cdots & 0 & b^{(t)}_{1,L} & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots & b^{(t)}_{2,L} & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & \vdots & & & \vdots \\ \vdots & & \ddots & 1 & b^{(t)}_{L-1,L} & \vdots & & & \vdots \\ \vdots & & & 0 & 1 & 0 & & & \vdots \\ \vdots & & & \vdots & b^{(t)}_{L+1,L} & 1 & \ddots & & \vdots \\ \vdots & & & \vdots & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & b^{(t)}_{n,L} & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

Step $\boldsymbol{L}$. Transform $\boldsymbol{x} \mapsto T_L\boldsymbol{x}$ such that the coefficients of $x_i x_L$ $(i = 1, \ldots, L)$ in $g_1(\boldsymbol{x}), \ldots, g_M(\boldsymbol{x})$ are all zero, where

$$T_L = \begin{pmatrix} 1 & 0 & \cdots & 0 & a_{1,L}^{(L)} & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots & a_{2,L}^{(L)} & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & \vdots & & & \vdots \\ \vdots & & \ddots & 1 & a_{L-1,L}^{(L)} & \vdots & & & \vdots \\ \vdots & & & 0 & a_{L,L} & 0 & & & \vdots \\ \vdots & & & \vdots & a_{L+1,L}^{(L)} & 1 & \ddots & & \vdots \\ \vdots & & & \vdots & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & a_{n,L}^{(L)} & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

If there is no such transformation, then go back to "**Step $\boldsymbol{L-1}$.**".
Step $\boldsymbol{L+1}$. Return $U = T_L T_{2,L-1} T_{1,L-1} \cdots T_{2,1} T_{1,1}$.

A.2 Analysis of Algorithm B

We find the following facts about Algorithm B.

Lemma A.1. Suppose $L \geq 3$. In "**Step $\boldsymbol{t}$.**" $t = 1, \ldots, L-2$) of Algorithm B,

$${}^tT_{1,t} \begin{pmatrix} O_{L,L} & * \\ * & * \end{pmatrix} T_{1,t} = \begin{pmatrix} O_{L,L} & * \\ * & * \end{pmatrix}.$$

This lemma shows that the $L \times L$ upper left part of $g_1(\boldsymbol{x}), \ldots, g_{M-1}(\boldsymbol{x})$ remains zero by linear transformation $T_{1,t}$.

Lemma A.2. In "**Step $\boldsymbol{t}$.**-(ii)" $(t = 1, \ldots, L-1)$, the coefficient of x_L^2 in $g_i(\boldsymbol{x})$ $(i = 1, \ldots, M-1)$ is

$$\sum_{1 \leq j \leq L-1} a_{j,L} L_{i,j}(a_{L+1,L}^{(t)}, \ldots, a_{n,L}^{(t)}) + Q_i(a_{L+1,L}^{(t)}, \ldots, a_{n,L}^{(t)}).$$

Theorem A.3. In "**Step $\boldsymbol{t}$.**-(ii)" $(t = 1, \ldots, L-1)$, the coefficient of $x_j x_L$ in $g_i(\boldsymbol{x})$ is equal to $L_{i,j}(a_{L+1,L}^{(t)}, \ldots, a_{n,L}^{(t)})$ $(i = 1, \ldots, M-1; j = 1, \ldots, L-1)$.

Observation A.4. In "**Step $\boldsymbol{t}$.**-(ii)" $(t = 1, \ldots, L-1)$, we must solve equations

$$\begin{cases} (\text{The coefficient of } x_1 x_L \text{ in } g_1(\boldsymbol{x})) = 0 \\ \qquad\vdots \\ (\text{The coefficient of } x_{L-1} x_L \text{ in } g_1(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_L^2 \text{ in } g_1(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_1 x_L \text{ in } g_2(\boldsymbol{x})) = 0 \\ \qquad\vdots \\ (\text{The coefficient of } x_L^2 \text{ in } g_{M-1}(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_t x_L \text{ in } g_M(\boldsymbol{x})) = 0, \end{cases}$$

i.e.,

$$\begin{cases} L_{1,1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ \qquad\qquad\vdots \\ L_{1,L-1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ \sum_{1\le j\le L-1} a_{j,L}L_{i,j}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) + Q_i(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ L_{2,1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ \qquad\qquad\vdots \\ L_{M-1,L-1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ (\text{The coefficient of } x_t x_L \text{ in } g_M(\boldsymbol{x})) = 0 \end{cases}$$

Note that we can solve linear equations without the L-th equation

$$\begin{cases} L_{1,1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ \qquad\qquad\vdots \\ L_{1,L-1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ L_{2,1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ \qquad\qquad\vdots \\ L_{M-1,L-1}(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) = 0 \\ (\text{The coefficient of } x_t x_L \text{ in } g_M(\boldsymbol{x})) = 0 \end{cases} \tag{6}$$

under the condition (5). However, $Q_i(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L})$ is not equal to zero in general for the solution of equations (6). It means that **Step *t*.**-(ii) fails in the case of $Q_i(a^{(t)}_{L+1,L},\dots,a^{(t)}_{n,L}) \neq 0$.

A.3 Example of Algorithm B

Let $k = \mathrm{GF}(7), n = 7, M = 2, L = 3$. We consider quadratic forms represented by the following matrices.

$$G_1 = \begin{pmatrix} 0&0&0&5&2&5&1 \\ 0&0&0&2&3&1&2 \\ 0&0&0&4&1&6&2 \\ 6&5&5&4&1&6&1 \\ 1&5&6&5&2&1&3 \\ 2&3&5&1&5&3&1 \\ 1&4&4&1&4&1&5 \end{pmatrix}, G_2 = \begin{pmatrix} 1&3&4&1&5&2&3 \\ 6&5&2&1&4&3&0 \\ 4&6&4&1&5&0&2 \\ 6&1&0&0&3&2&4 \\ 4&2&6&6&0&1&3 \\ 1&5&4&6&6&3&4 \\ 2&5&2&4&6&3&2 \end{pmatrix}.$$

Step 1.-(i) Using Algorithm A, we solve the equations

$$\begin{cases} (\text{The coefficient of } x_1^2 \text{ in } g_2(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_1 x_2 \text{ in } g_2(\boldsymbol{x})) = 0, \end{cases}$$

i.e.,

$$\begin{cases} 1 + 2a_{2,1}^{(1)} + a_{3,1}^{(1)} + 5{a_{2,1}^{(1)}}^2 + a_{2,1}^{(1)}a_{3,1}^{(1)} + 4{a_{3,1}^{(1)}}^2 = 0 \\ 6 + a_{3,2}^{(1)} = 0 \end{cases}$$

From these equations, we obtain $(a_{2,1}^{(1)}, a_{3,1}^{(1)}, a_{3,2}^{(1)}) = (2, 5, 1)$. Then,

$$G_1 \mapsto \begin{pmatrix} 0 & 0 & 0 & 1 & 6 & 2 & 1 \\ 0 & 0 & 0 & 6 & 4 & 0 & 4 \\ 0 & 0 & 0 & 4 & 1 & 6 & 2 \\ 6 & 3 & 5 & 4 & 1 & 6 & 1 \\ 6 & 4 & 6 & 5 & 2 & 1 & 3 \\ 5 & 1 & 5 & 1 & 5 & 3 & 1 \\ 1 & 1 & 4 & 1 & 4 & 1 & 5 \end{pmatrix}, G_2 \mapsto \begin{pmatrix} 0 & 1 & 0 & 1 & 3 & 1 & 6 \\ 6 & 3 & 6 & 2 & 2 & 3 & 2 \\ 1 & 3 & 4 & 1 & 5 & 0 & 2 \\ 1 & 1 & 0 & 0 & 3 & 2 & 4 \\ 3 & 1 & 6 & 6 & 0 & 1 & 3 \\ 3 & 2 & 4 & 6 & 6 & 3 & 4 \\ 1 & 0 & 2 & 4 & 6 & 3 & 2 \end{pmatrix}.$$

Step 1.-(ii)

$$\begin{cases} (\text{The coefficient of } x_1x_3 \text{ in } g_1(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_2x_3 \text{ in } g_1(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_1x_3 \text{ in } g_2(\boldsymbol{x})) = 0 \\ (\text{The coefficient of } x_3^2 \text{ in } g_1(\boldsymbol{x})) = 0, \end{cases}$$

i.e.,

$$\begin{cases} 5b_{5,3}^{(1)} + 2b_{7,3}^{(1)} = 0 \\ 2b_{4,3}^{(1)} + b_{5,3}^{(1)} + b_{6,3}^{(1)} + 5b_{7,3}^{(1)} = 0 \\ 1 + 2b_{4,3}^{(1)} + 6b_{5,3}^{(1)} + 4b_{6,3}^{(1)} = 0 \\ b_{1,3}^{(1)}(5b_{5,3}^{(1)} + 2b_{7,3}^{(1)}) + b_{2,3}^{(1)}(2b_{4,3}^{(1)} + b_{5,3}^{(1)} + b_{6,3}^{(1)} + 5b_{7,3}^{(1)}) + 4{b_{4,3}^{(1)}}^2 \\ \quad +6b_{4,3}^{(1)}b_{5,3}^{(1)} + 2b_{4,3}^{(1)}b_{7,3}^{(1)} + 2{b_{5,3}^{(1)}}^2 + 6b_{5,3}^{(1)}b_{6,3}^{(1)} + 3{b_{6,3}^{(1)}}^2 + 2b_{6,3}^{(1)}b_{7,3}^{(1)} + 5{b_{7,3}^{(1)}}^2 = 0 \end{cases}$$

These multivariate equations are hard to solve.