

# Algorithms to solve massively under-defined systems of multivariate quadratic equations

Yasufumi Hashimoto \*

Institute of Systems & Information Technologies/Kyushu  
7F 2-1-22, Momochihama, Fukuoka 814-0001, JAPAN  
e-mail:hasimoto@isit.or.jp

**Abstract.** It is well known that the problem to solve a set of randomly chosen multivariate quadratic equations over a finite field is NP-hard. However, when the number of variables is much larger than the number of equations, it is not necessarily difficult to solve equations. In fact, when  $n \geq m(m+1)$  ( $n, m$  are the numbers of variables and equations respectively) and the field is of even characteristic, there is an algorithm to solve equations in polynomial time (see [Kipnis et al., Eurocrypt'99] and also [Courtois et al., PKC'02]). In the present paper, we propose two algorithms to solve quadratic equations; one is for the case of  $n \geq$  (about)  $m^2 - 2m^{3/2} + 2m$  and the other is for the case of  $n \geq m(m+1)/2 + 1$ . The first algorithm solves equations over any finite field in polynomial time. The second algorithm requires exponential time operations. However, the number of required variables is much smaller than that in the first one, and the complexity is essentially less than the exhaustive search.

**Keywords.** under-defined multivariate quadratic equations

## 1 Introduction

It is well known that the problem to solve a set of randomly chosen multivariate quadratic equations over a finite field is NP-hard. Then the cryptosystems based on multivariate quadratic equations (Matsumoto-Imai, HEF, UOV, STS, TTM and so on, see e.g. [5], [7] and their references) have been expected to be secure against the quantum attacks. However, not all quadratic equations are difficult to be solved while the problem itself is NP-hard. In fact, some of such cryptosystems were already broken and some others of them are weaker than expected when they were proposed. Thus it is important to study which quadratic equations are solved easily and how to characterize its difficulty for the practical use of quadratic equations in cryptology.

---

\* Partially supported by JST Strategic Japanese-Indian Cooperative Programme on multidisciplinary Research Field, which combines Information and Communications Technology with Other Fields, entitled "Analysis of cryptographic algorithms and evaluation on enhancing network security Based on Mathematical Science", and JSPS Grant-in-Aid for Young Scientists (B) no. 20740027.

For this topic, there have been several works in the view of the relation between the numbers of variables and equations. In fact, Courtois et al. ([2] and [3]) have studied how to solve the equations when  $m$  is much larger than  $n$  (where  $m, n$  are the numbers of equations and variables respectively). On the other hand, Kipnis et al. [6] studied the case when  $n$  is much larger than  $m$ . In fact, they found a polynomial time algorithm to solve quadratic equations when  $n \geq m(m+1)$  and the characteristic of the field is even. Note that, when the characteristic is odd, their algorithm requires  $O(2^m \times (\text{polynomial}))$  operations. Although Courtois et al. [1] modified it to be more effectively for odd characteristic cases, its modification requires much more variables.

In the present paper, we propose two algorithms to solve multivariate quadratic equations when  $n$  is sufficiently larger than  $m$ . The first algorithm solves equations over any finite field in polynomial time when  $n \geq$  (about)  $m^2 - 2m^{3/2} + 2m$ . The number of variables required in this algorithm is less than that in [6], and this algorithm works in polynomial time both for even and odd characteristic fields. The second algorithm solves equations for  $n \geq m(m+1)/2 + 1$ . The complexity of the second algorithm is roughly estimated by  $O(2^m)$  or  $O(3^m)$ . While it is in exponential time, it is much better than the exhaustive search, especially for large fields, and furthermore the number of required variables is much less than that in the first algorithm.

## 2 Preparations

### 2.1 Notations

Throughout this paper, we use the following notations.

$q$ : a power of prime.

$k$ : a finite field of order  $q$ .

$n, m \geq 1$ : integers.

$x = (x_1, \dots, x_n)^t \in k^n$ .

$\tilde{x} = (x_0, x_1, \dots, x_n)^t \in k^{n+1}$ .

$f_l(x) \in k$  ( $1 \leq l \leq m$ ): a quadratic form of  $x$ .

$\tilde{f}_l(\tilde{x}) \in k$  ( $1 \leq l \leq m$ ): the homogeneous quadratic form of  $\tilde{x}$  such that

$\tilde{f}_l(1, x_1, \dots, x_n) = f_l(x_1, \dots, x_n)$ .

$e_i := (\underbrace{0, \dots, 0}_{i-1}, 1, 0, \dots, 0)^t \in k^n$  ( $1 \leq i \leq n$ ).

$\tilde{e}_i := (\underbrace{0, \dots, 0}_i, 1, 0, \dots, 0)^t \in k^{n+1}$  ( $0 \leq i \leq n$ ).

$a_i = (a_{1i}, \dots, a_{ni})^t \in k^n$  ( $1 \leq i \leq n$ ): a vector with  $a_{ii} \neq 0$ .

$\tilde{a}_i = (\tilde{a}_{0i}, \dots, \tilde{a}_{ni})^t \in k^{n+1}$  ( $0 \leq i \leq n$ ): a vector with  $\tilde{a}_{ii} \neq 0$ .

$U_i := (e_1, \dots, e_{i-1}, a_i, e_{i+1}, \dots, e_n)$ : an invertible linear map such that  $x_i \mapsto a_{1i}x_1 + \dots + a_{ni}x_n$  and  $x_j \mapsto x_j$  for  $j \neq i$ .

$\tilde{U}_i := (\tilde{e}_0, \dots, \tilde{e}_{i-1}, \tilde{a}_i, \tilde{e}_{i+1}, \dots, \tilde{e}_n)$ : an invertible linear map such that  $\tilde{x}_i \mapsto \tilde{a}_{0i}\tilde{x}_0 + \tilde{a}_{1i}\tilde{x}_1 + \dots + \tilde{a}_{ni}\tilde{x}_n$  and  $\tilde{x}_j \mapsto \tilde{x}_j$  for  $j \neq i$ .

$\Omega(n)$ : the complexity of the Gaussian elimination to solve  $n$  linear equations.

## 2.2 Elementary facts

Remember the following elementary facts learned in the undergraduate linear algebra.

**Fact 1.** Let  $g(x) := \sum_{1 \leq i, j \leq n} g_{ij} x_i x_j$  be a homogeneous quadratic form of  $x = (x_1, \dots, x_n)^t \in k^n$  ( $g_{ij} \in k$ ) and  $G = (G_{ij})_{1 \leq i, j \leq n}$  an  $n \times n$  matrix over  $k$  ( $G_{ij} \in k$ ) with  $g_{ii} = G_{ii}$  and  $g_{ij} = G_{ij} + G_{ji}$  for  $i \neq j$ . Then  $g(x) = x^t G x$ .

**Fact 2.** Let  $G, U$  be  $n \times n$  matrices and  $u_1, \dots, u_n \in k^n$  the column vectors in  $U$ , namely  $U = (u_1, \dots, u_n)$ . Then the  $ij$ -entry of  $U^t G U$  is  $u_i^t G u_j$ .

**Fact 3.** Let  $g(x), G$  be as in Fact 1 and  $U_i$  be as in Section 2.1. Denote by  $g(U_l x) = \sum_{1 \leq i, j \leq n} g_{ij}^{(l)} x_i x_j$ . Then we have  $g_{ll}^{(l)} = a_l^t G a_l = g(u_l)$ ,  $g_{il}^{(l)} = e_i^t G a_l + a_l^t G e_i$  ( $i \neq l$ ) and  $g_{ij}^{(l)} = g_{ij}$  ( $i, j \neq l$ ).

## 3 Kipnis-Patarin-Goubin's algorithm for $n \geq m(m+1)$

In this section, we recall the algorithm proposed by Kipnis-Patarin-Goubin [6] to solve  $m$  quadratic equations with  $n$  variables for  $n \geq m(m+1)$ .

**Step 1.** Find  $U_2$  such that the coefficients of  $x_1 x_2$  in  $f_1(U_2 x), \dots, f_m(U_2 x)$  are zero.

**Step 2.** Put  $f_l^{(2)}(x) := f_l(U_2 x)$ . Find  $U_3$  such that the coefficients of  $x_1 x_3, x_2 x_3$  in  $f_1^{(2)}(U_3 x), \dots, f_m^{(2)}(U_3 x)$  are zero.

⋮

**Step  $m-1$ .** Put  $f_l^{(m-1)}(x) := f_l^{(m-2)}(U_{m-1} x)$ . Find  $U_m$  such that the coefficients of  $x_1 x_m, x_2 x_m, \dots, x_{m-1} x_m$  in  $f_1^{(m-1)}(U_m x), \dots, f_m^{(m-1)}(U_m x)$  are zero.

$$f_l(x) \mapsto x^t \left( \begin{array}{cc|c} * & 0 & * \\ 0 & * & * \\ * & * & * \end{array} \right) x \mapsto x^t \left( \begin{array}{ccc|c} * & 0 & 0 & * \\ 0 & * & 0 & * \\ 0 & 0 & * & * \\ * & * & * & * \end{array} \right) x \mapsto \dots \mapsto x^t \left( \begin{array}{ccc|c} * & & & O \\ & \ddots & & * \\ O & & * & * \\ * & & * & * \end{array} \right) x.$$

**Step  $m$ .** Put  $g_l(x) := f_l^{(m)}(U_m x)$ . Substitute values into  $x_{m+1}, \dots, x_n$  such that  $g_1(x), g_2(x), \dots, g_m(x)$  are linear combinations of  $x_1^2, \dots, x_m^2$  and constants.

**Step  $m+1$ .** Find a solution  $(x_1, \dots, x_m)$  of  $g_l(x) = 0$  for  $1 \leq l \leq m$ .

**Observation 1.** According to Fact 3, we see that Step  $t$  ( $1 \leq t \leq m-1$ ) requires to solve  $tm$  homogeneous linear equations with  $n$  variables. Then one needs the condition  $n > m(m-1)$  until Step  $m-1$ .

**Observation 2.** Remember that the coefficients of  $x_i x_j$  ( $1 \leq i < j \leq m$ ) in  $g_l(x)$  are zero. We see that Step  $m$  requires to solve  $m^2$  linear equations with  $n-m$  variables  $(x_{m+1}, \dots, x_n)$ . Then one needs the condition  $n \geq m(m+1)$  in Step  $m$ .

**Observation 3.** Since  $g_l(x)$ 's are linear combinations of  $x_1^2, \dots, x_m^2$  and constants, one can reduce the problem solving  $g_l(x) = 0$  for  $1 \leq l \leq m$  in Step

$m + 1$  to the problem solving  $x_1^2 = (\text{const}), \dots, x_m^2 = (\text{const})$  by linear operations. This requires to calculate the square roots. Thus this algorithm will work in polynomial time when  $q$  is even, and with  $2^m \times (\text{polynomial})$  operations in average when  $q$  is odd.

Note that Courtois et al. modified this algorithm for odd characteristic  $k$  with the complexity  $2^{40} \times (\text{polynomial})$ . However the number of required variables is  $n \geq 2^{m/7}(m + 1)$ . See [1] for the detail of its modification.

#### 4 Solving quadratic equations for $n \geq (\text{about})$ $m^2 - 2m^{3/2} + 2m$

In this section, we propose an algorithm to solve equations for  $n \geq (\text{about})$   $m^2 - 2m^{3/2} + 2m$ . For the algorithm, we first prepare the following elementary fact.

**Fact 4.** Let  $U = (u_{ij})_{0 \leq i, j \leq n}$  be an invertible matrix over  $k$ . If  $U$  satisfies that  $u_{00} \neq 0$  and the coefficient of  $x_0^2$  of  $\tilde{f}_l(U\tilde{x})$  are zero for  $1 \leq l \leq m$ , then  $(u_{00}^{-1}u_{10}, \dots, u_{00}^{-1}u_{n0})$  is a solution of  $f_1(x) = 0, \dots, f_m(x) = 0$ .

This follows from Fact 1 and 2 immediately. Then, instead of solving the equation, we will give an algorithm to find such  $U$  in this section.

Before it, we prepare the following two algorithms.

##### Algorithm A.

**Aim.** Let  $g(x)$  be a quadratic form  $x = (x_1, \dots, x_m)^t \in k^m$ . Find an invertible linear transform  $U : k^m \rightarrow k^m$  such that the coefficients of  $x_i x_j$  ( $i + j \leq m$ ) in  $g(Ux)$  are zero.

$$g(x) \mapsto x^t \begin{pmatrix} O & * \\ \cdot & \cdot \\ * & * \end{pmatrix} x$$

**Step 1.** Find  $U_1$  such that the coefficients of  $x_1^2$  is zero.

**Step 2.** Put  $g^{(1)}(x) := g(U_1x)$ . Find  $U_2$  such that the coefficients of  $x_1 x_2, x_2^2$  in  $g^{(1)}(U_2x)$  are zero.

$\vdots$

**Step  $\lfloor m/2 \rfloor$ .** Put  $g^{(\lfloor m/2 \rfloor - 1)}(x) := g^{(\lfloor m/2 \rfloor - 2)}(U_{\lfloor m/2 \rfloor - 1}x)$ . Find  $U_{\lfloor m/2 \rfloor}$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq \lfloor m/2 \rfloor$ ) in  $g^{(\lfloor m/2 \rfloor - 1)}(U_{\lfloor m/2 \rfloor}x)$  are zero.

**Step  $\lfloor m/2 \rfloor + 1$ .** Put  $g^{(\lfloor m/2 \rfloor)}(x) := g^{(\lfloor m/2 \rfloor - 1)}(U_{\lfloor m/2 \rfloor}x)$ . Find  $U_{\lfloor m/2 \rfloor + 1}$  such that the coefficients of  $x_i x_{\lfloor m/2 \rfloor + 1}$  ( $1 \leq i \leq \lfloor m/2 \rfloor - 1$ ) in  $g^{(\lfloor m/2 \rfloor)}(U_{\lfloor m/2 \rfloor + 1}x)$  are zero.

**Step  $\lfloor m/2 \rfloor + 2$ .** Put  $g^{(\lfloor m/2 \rfloor + 1)}(x) := g^{(\lfloor m/2 \rfloor)}(U_{\lfloor m/2 \rfloor + 1}x)$ . Find  $U_{\lfloor m/2 \rfloor + 2}$  such that the coefficients of  $x_i x_{\lfloor m/2 \rfloor + 2}$  ( $1 \leq i \leq \lfloor m/2 \rfloor - 2$ ) in  $g^{(\lfloor m/2 \rfloor + 1)}(U_{\lfloor m/2 \rfloor + 2}x)$  are zero.

$\vdots$

**Step  $m - 1$ .** Put  $g^{(m-2)}(x) := g^{(m-3)}(x)(U_{m-2}x)$ . Find  $U_{m-1}$  such that the coefficients of  $x_1x_{m-1}$  in  $g^{(m-1)}(U_{m-1}x)$  are zero.

**Observation.** Due to Fact 3, we see that Step  $t$  ( $1 \leq t \leq \lfloor m/2 \rfloor$ ) requires to solve a quadratic homogeneous equation and  $t - 1$  homogeneous linear equations of  $n$  variables. On the other hand, Step  $t$  ( $\lfloor m/2 \rfloor + 1 \leq m - 1$ ) requires to solve  $m - t$  homogeneous linear equations of  $n$  variables. Thus the complexity in this algorithm is less than  $m\Omega(\lfloor m/2 \rfloor)$ .

**Algorithm B.**

**Aim.** Let  $n, L, M \geq 1$  be integers with  $L \leq n/2$  and

$$M \leq \begin{cases} \left\lfloor \frac{n-L}{L-1} \right\rfloor, & (n \leq L^2 - L), \\ L-1, & (L^2 - L + 1 \leq n \leq L^2), \\ L, & (n \geq L^2 + 1), \end{cases}$$

and  $g_1(x), \dots, g_M(x)$  be quadratic forms of  $x = (x_1, \dots, x_n)^t$ . Suppose that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_1(x), \dots, g_{M-1}(x)$  are zero. Find an invertible linear transform  $U : k^n \rightarrow k^n$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_1(x), \dots, g_M(x)$  are zero.

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

**Step 1.** Using Algorithm A, find an invertible linear map  $W_1 : k^L \rightarrow k^L$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L, i + j \leq L$ ) in  $g_M(\tilde{W}_1 x)$  are zero, where  $\tilde{W}_1 := \begin{pmatrix} W_1 \\ I \end{pmatrix}$ . Find  $U_L$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_l(\tilde{W}_1 U_L x)$  for  $1 \leq l \leq M - 1$  and of  $x_1 x_L$  in  $g_M(\tilde{W}_1 U_L x)$  are zero.

$$g_M(x) \xrightarrow{\text{Alg. A}} x^t \left( \begin{array}{ccc|c} O & 0 & * & \\ & \ddots & * & * \\ & & \ddots & * \\ & \ddots & \ddots & \ddots \\ & 0 & * & \ddots \\ * & * & & * \\ \hline & & * & * \end{array} \right) x \xrightarrow{\text{change } x_L} x^t \left( \begin{array}{ccc|c} O & 0 & 0 & \\ & \ddots & * & * \\ & & \ddots & \ddots \\ & \ddots & \ddots & \ddots \\ & 0 & * & \ddots \\ 0 & * & & * \\ \hline & & * & * \end{array} \right) x$$

**Step 2.** Put  $g_l^{(1)}(x) := g_l(\tilde{W}_1 U_L x)$ . Using Algorithm A, find an invertible linear map  $W_2 : k^{L-1} \rightarrow k^{L-1}$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L, i + j \leq L + 1$ ) in  $g_M^{(1)}(\tilde{W}_2 x)$  are zero, where  $\tilde{W}_2 = \begin{pmatrix} 1 \\ W_2 \\ I \end{pmatrix}$ . Find  $U_L$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_l^{(1)}(\tilde{W}_2 U_L x)$  for  $1 \leq l \leq M - 1$  and of  $x_1 x_L, x_2 x_L$  in  $g_M^{(1)}(\tilde{W}_2 U_L x)$  are zero.

⋮

**Step  $L - 1$ .** Put  $g_i^{(L-2)}(x) := g_i^{(L-3)}(\tilde{W}_{L-2}U_Lx)$ . Using Algorithm A, find an invertible linear map  $W_{L-1} : k^2 \rightarrow k^2$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L, i+j \leq 2L-1$ ) in  $g_M^{(L-2)}(\tilde{W}_{L-1}x)$  are zero, where  $\tilde{W}_{L-1} = \begin{pmatrix} I \\ W_{L-1} \\ I \end{pmatrix}$ .

Find  $U_L$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_i^{(L-2)}(\tilde{W}_{L-1}U_Lx)$  for  $1 \leq l \leq M-1$  and of  $x_1 x_L, x_2 x_L, \dots, x_{L-1} x_L$  in  $g_M^{(L-2)}(\tilde{W}_{L-1}U_Lx)$  are zero.

**Step  $L$ .** Put  $g_i^{(L-1)}(x) := g_i^{(L-2)}(\tilde{W}_{L-1}U_Lx)$ . Find  $U_L$  such that the coefficients of  $x_i x_j$  ( $i, j \leq L$ ) in  $g_1^{(L-1)}(U_Lx), \dots, g_N^{(L-1)}(U_Lx)$  are zero.

**Observation 1.** According Fact 3, we see that Step  $t$  ( $1 \leq t \leq L - 1$ ) requires to solve

- (i)  $(L - 1)(M - 1) + t - 1$  homogeneous linear equations of  $(a_{L+1,L}, \dots, a_{n,L})$ ,
- (ii) 1 homogeneous linear equation of  $(a_{L,L}, \dots, a_{n,L})$ ,
- (iii)  $M - 1$  homogeneous quadratic equations of  $(a_{1,L}, \dots, a_{n,L})$  in the forms

$$\sum_{i=1}^L a_{i,L} \times (\text{linear form of } (a_{L+1,L}, \dots, a_{n,L})) \\ + (\text{quadratic form of } (a_{L+1,L}, \dots, a_{n,L})) = 0.$$

In order to solve the equations (i),(ii) and (iii), first solve (i) and find  $a_{L+1,L}, \dots, a_{n,L}$ . Then  $a_{L,L}$  is automatically determined by (ii). Substituting such values to (iii), we see that the quadratic equations (iii) become  $N - 1$  linear equations of  $(a_{1,L}, \dots, a_{L-1,L})$ . Thus one needs  $n - L > (L - 1)(M - 1) + L - 2$  and  $L \geq M$  until Step  $L - 1$ .

**Observation 2.** Similarly, Step  $L$  requires to solve

- (i)  $(L - 1)M$  homogeneous linear equations of  $(a_{L+1,L}, \dots, a_{n,L})$ ,
- (ii)  $M - 1$  homogeneous quadratic equations of  $(a_{1,L}, \dots, a_{n,L})$  in the same forms to (iii) in the previous observation.
- (iii) a homogeneous quadratic equation of  $(a_{1,L}, \dots, a_{n,L})$  in the form

$$a_{L,L}^2 + \sum_{i=1}^L a_{i,L} \times (\text{linear form of } (a_{L+1,L}, \dots, a_{n,L})) \\ + (\text{quadratic form of } (a_{L+1,L}, \dots, a_{n,L})) = 0.$$

When  $n - L > (L - 1)M$ , one can find a non-trivial solution of (i). Substitute it into (ii). The quadratic equations (ii) become to linear equations, and (iii) becomes a quadratic equation whose quadratic terms is only  $a_{L,L}^2$ . Then a solution will be found when  $M \leq L$ . On the other hand, when  $n - L = (L - 1)N$ , the solution of (i) is trivial, namely  $a_{L+1,L} = \dots = a_{n,L} = 0$ . This means that (ii) and (iii) become homogeneous equations of  $(a_{1,L}, \dots, a_{L,L})$ . Then  $M < L$  is necessary. Therefore, in Step  $L$ , one needs the condition that  $M < (n - L)/(L - 1)$ ,

$M \leq L$  or  $M = (n - L)/(L - 1)$ ,  $M < L$ , namely

$$M \leq \begin{cases} \lfloor \frac{n-L}{L-1} \rfloor, & n \leq L^2 - L, \\ L - 1, & L^2 - L + 1 \leq n \leq L^2, \\ L, & n \geq L^2 + 1. \end{cases}$$

**Observation 3.** Each step include Algorithm A with  $m \leq L$ , solving at most  $(L - 1)M$  linear equations and a quadratic equation. Thus we see that the complexity in Algorithm B is less than  $L(L\Omega(\lfloor L/2 \rfloor) + \Omega((L - 1)M) + \Omega(M - 1))$ . Since  $\Omega(n) \ll n^3$ , we can claim that this algorithm works in polynomial time.

**Algorithm C.**

**Aim.** Let  $n, L, M \geq 1$  be integers with the same conditions in Aim of Algorithm B, and  $g_1(x), \dots, g_M(x)$  be quadratic forms of  $x = (x_1, \dots, x_n)^t$ . Find an invertible linear transform  $U : k^n \rightarrow k^n$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_1(x), \dots, g_M(x)$  are zero.

$$\underbrace{x^t \begin{pmatrix} * & * \\ * & * \end{pmatrix} x, \dots, x^t \begin{pmatrix} * & * \\ * & * \end{pmatrix} x}_M \mapsto \underbrace{x^t \begin{pmatrix} O_L & * \\ * & * \end{pmatrix} x, \dots, x^t \begin{pmatrix} O_L & * \\ * & * \end{pmatrix} x}_M$$

**Step 1.** Find an invertible linear transform  $V_1 : k^n \rightarrow k^n$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_1(V_1 x)$  are zero by using Algorithm A.

**Step 2.** Put  $g_1^{(1)}(x) := g_1(V_1 x)$ . Using Algorithm B, find an invertible linear transform  $V_2 : k^n \rightarrow k^n$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_1^{(1)}(V_2 x), g_2^{(1)}(V_2 x)$  are zero.

$\vdots$

**Step  $M$ .** Put  $g_i^{(M-1)}(x) := g_i^{M-2}(V_{M-1} x)$ . Using Algorithm B, find an invertible linear transform  $V_M : k^n \rightarrow k^n$  such that the coefficients of  $x_i x_j$  ( $1 \leq i, j \leq L$ ) in  $g_1^{(M-1)}(V_M x), \dots, g_M^{(M-1)}(V_M x)$  are zero.

**Observation.** According to Observation 3 in Algorithm B, we see that Step  $t$  ( $1 \leq t \leq M$ ) requires the complexity  $L(L\Omega(\lfloor L/2 \rfloor) + \Omega((L - 1)t) + \Omega(t - 1))$ . Summing up this from  $t = 1$  to  $M$ , we can estimate the complexity in this algorithm by

$$\begin{aligned} & ML(L\Omega(\lfloor L/2 \rfloor) + \Omega((L - 1)M) + \Omega(M - 1)) \\ & < n(L\Omega(\lfloor L/2 \rfloor) + \Omega(n - M) + \Omega(M - 1)) = O(n\Omega(n)). \end{aligned}$$

Thus the Algorithm C works in polynomial time.

Based on Algorithm C, we propose an algorithm to solve quadratic equations for  $n \geq$  (about)  $m^2 - 2m^{3/2} + 2m$ .

**Algorithm 1.**

**Aim.** Find a solution  $x \in k^n$  of the equations  $f_1(x) = 0, \dots, f_m(x) = 0$  when  $n \geq$  (about)  $m^2 - 2m^{3/2} + 2m$ .

**Step 1.** Choose  $M_1 < \sqrt{n+1}$ . Put

$$L_1 := \min \left( \left\lfloor \frac{n+1}{2} \right\rfloor, \left\lfloor \frac{n+1+M_1}{M_1+1} \right\rfloor \right).$$

Using Algorithm C, find an invertible linear map  $V_1 : k^{n+1} \rightarrow k^{n+1}$  such that the coefficients of  $x_i x_j$  ( $0 \leq i, j \leq L_1 - 1$ ) in  $\tilde{f}_1(V_1 \tilde{x}), \dots, \tilde{f}_{M_1}(V_1 \tilde{x})$  are zero. Put  $\tilde{f}_i^{(1)}(\tilde{x}) := \tilde{f}_i(V_1 \tilde{x})$ .

**Step 2.** Choose  $M_2 < \sqrt{L_1}$ . Put

$$L_2 := \min \left( \left\lfloor \frac{L_1}{2} \right\rfloor, \left\lfloor \frac{L_1+M_2}{M_2+1} \right\rfloor \right).$$

Using Algorithm C, find an invertible linear map  $V_2 : k^{L_1} \rightarrow k^{L_1}$  such that the coefficients of  $x_i x_j$  ( $0 \leq i, j \leq L_2 - 1$ ) in  $\tilde{f}_{M_1+1}^{(1)}(\tilde{V}_2 \tilde{x}), \dots, \tilde{f}_{M_1+M_2}^{(1)}(\tilde{V}_2 \tilde{x})$  are zero, where  $\tilde{V}_2 := \begin{pmatrix} V_2 \\ I \end{pmatrix}$ . Put  $\tilde{f}_i^{(2)}(\tilde{x}) := \tilde{f}_i^{(1)}(\tilde{V}_2 \tilde{x})$ .

$\vdots$

Continue such operations until  $L_t = 1$ . Then one can get an invertible linear map  $U = (u_{ij})_{0 \leq i, j \leq n}$  such that the coefficients of  $x_0^2$  in  $\tilde{f}_l(U \tilde{x})$  for  $1 \leq l \leq M_1 + M_2 + \dots + M_t$  are zero.

$$\begin{aligned} \tilde{f}_1(\tilde{x}), \dots, \tilde{f}_m(\tilde{x}) &\xrightarrow{\text{Step 1}} \underbrace{\tilde{x}^t \begin{pmatrix} O_{L_1} & * \\ * & * \end{pmatrix} \tilde{x}, \dots, \tilde{x}^t \begin{pmatrix} O_{L_1} & * \\ * & * \end{pmatrix} \tilde{x}, \tilde{x}^t \begin{pmatrix} * & * \\ * & * \end{pmatrix} \tilde{x}, \dots, \tilde{x}^t \begin{pmatrix} * & * \\ * & * \end{pmatrix} \tilde{x}}_{M_1} \\ &\xrightarrow{\text{Step 2}} \underbrace{\tilde{x}^t \begin{pmatrix} O_{L_2} & * \\ * & * \end{pmatrix} \tilde{x}, \dots, \tilde{x}^t \begin{pmatrix} O_{L_2} & * \\ * & * \end{pmatrix} \tilde{x}, \tilde{x}^t \begin{pmatrix} * & * \\ * & * \end{pmatrix} \tilde{x}, \dots, \tilde{x}^t \begin{pmatrix} * & * \\ * & * \end{pmatrix} \tilde{x}}_{M_1+M_2} \\ &\mapsto \dots \mapsto \underbrace{\tilde{x}^t \begin{pmatrix} 0 & * \\ * & * \end{pmatrix} \tilde{x}, \dots, \tilde{x}^t \begin{pmatrix} 0 & * \\ * & * \end{pmatrix} \tilde{x}}_{M_1+M_2+\dots+M_t} \end{aligned}$$

Due to Fact 4, this algorithm finds a solution when  $m \leq M_1 + \dots + M_t \sim n^{1/2} + n^{1/4} + \dots$ , namely  $n \geq (\text{about}) m^2 - 2m^{3/2} + 2m$ . According to Observation in Algorithm C, we see that the complexity of this algorithm is  $O(n^4) + O(n^2) + O(n) + \dots = O(n^4)$ . Thus Algorithm 1 works in polynomial time.

The following is the table of the number of variables required to solve  $m$  equations in Algorithm 1 and the algorithm in [6].

$m$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	$\dots$
$n$ for Alg. 1	1	3	4	9	12	16	20	25	36	49	64	81	90	100	121	144	156	169	$\dots$
$n$ for [6]	2	6	12	20	30	42	56	72	90	110	132	156	182	210	240	272	306	342	$\dots$

## 5 Solving quadratic equations for $n \geq m(m+1)/2 + 1$

In this section, we propose an algorithm to solve equations for  $n \geq m(m+1)/2+1$ .

### Algorithm 2.

**Aim.** Find a solution  $x \in k^n$  of the equations  $f_1(x) = 0, \dots, f_m(x) = 0$  when  $n \geq m(m+1)/2 + 1$ .

**Step 1.** Find  $\tilde{U}_0$  such that the coefficient of  $x_0^2$  in  $\tilde{f}_1(\tilde{U}_0\tilde{x})$  is zero. Put  $\tilde{f}_l^{(1)}(x) := \tilde{f}_l(\tilde{U}_0\tilde{x})$ .

**Step 2.** Find  $\tilde{U}_1$  such that the coefficients of  $x_0x_1$  and  $x_1^2$  in  $\tilde{f}_1^{(1)}(\tilde{U}_1\tilde{x})$  and the coefficient of  $x_0x_1$  in  $\tilde{f}_2^{(1)}(\tilde{U}_1\tilde{x})$  are zero. Put  $\tilde{f}_l^{(1,1)}(\tilde{x}) := \tilde{f}_l^{(1)}(\tilde{U}_1\tilde{x})$ . If there is a solution  $z_2 \in k$  of  $\tilde{f}_2^{(1,1)}(1, z_2, 0, \dots, 0) = 0$ , denote by  $V_2 := \begin{pmatrix} 1 & 0 \\ z_2 & 1 \\ & & I \end{pmatrix}$ , and

put  $\tilde{f}_l^{(2)}(\tilde{x}) := \tilde{f}_l^{(1,1)}(V_2\tilde{x})$ . If there are no such  $z_2$ , take another  $\tilde{U}_1$  and repeat it until such  $z_2 \in k$  appears.

**Step 3.** Find  $\tilde{U}_1$  such that the coefficients of  $x_0x_1$  and  $x_1^2$  in  $\tilde{f}_1^{(2)}(\tilde{U}_1\tilde{x})$ ,  $\tilde{f}_2^{(2)}(\tilde{U}_1\tilde{x})$  and the coefficient of  $x_0x_1$  in  $\tilde{f}_3^{(2)}(\tilde{U}_1\tilde{x})$  are zero. Put  $\tilde{f}_l^{(2,1)}(\tilde{x}) := \tilde{f}_l^{(2)}(\tilde{U}_1\tilde{x})$ . If there is a solution  $z_3 \in k$  of  $\tilde{f}_3^{(2,1)}(1, z_3, 0, \dots, 0) = 0$ , denote by  $V_3 := \begin{pmatrix} 1 & 0 \\ z_3 & 1 \\ & & I \end{pmatrix}$ .

and put  $\tilde{f}_l^{(3)}(\tilde{x}) := \tilde{f}_l^{(2,1)}(V_3\tilde{x})$ . If there are no such  $z_3$ , take another  $\tilde{U}_1$  and repeat it until such  $z_3 \in k$  appears.

⋮

**Step  $m$ .** Find  $\tilde{U}_1$  such that the coefficients of  $x_0x_1$  and  $x_1^2$  in  $\tilde{f}_1^{(m-1)}(\tilde{U}_1\tilde{x}), \dots, \tilde{f}_{m-1}^{(m-1)}(\tilde{U}_1\tilde{x})$  and the coefficient of  $x_0x_1$  in  $\tilde{f}_m^{(m-1)}(\tilde{U}_1\tilde{x})$  are zero. Put  $\tilde{f}_l^{(m-1,1)}(\tilde{x}) := \tilde{f}_l^{(m-1)}(\tilde{U}_1\tilde{x})$ . If there is a solution  $z_m \in k$  of  $\tilde{f}_m^{(m-1,1)}(1, z_m, 0, \dots, 0) = 0$ ,

denote by  $V_m := \begin{pmatrix} 1 & 0 \\ z_m & 1 \\ & & I \end{pmatrix}$  and put  $\tilde{f}_l^{(m)}(\tilde{x}) := \tilde{f}_l^{(m-1,1)}(V_m\tilde{x})$ . If there are no

such  $z_m$ , take another  $\tilde{U}_1$  and repeat until such  $z_m \in k$  appears.

**Observation 1.** Step 1 requires to solve a homogeneous quadratic equation of  $n+1$  variables. Thus one needs  $n \geq 2$  in Step 1. According to Fact 4, we see that Step 1 solves a quadratic equation of at least 2 variables.

**Observation 2.** Step 2 requires to solve two homogeneous linear equations and a homogeneous quadratic equation of  $n+1$  variables. The linear equations is solved by the elimination and the quadratic equation is solved by Step 1. Since Step 1 requires at least 2 variables, Step 2 requires at least  $2+2=4$  variables. We also note that the coefficients of  $x_0^2, x_0x_1, x_1^2$  in  $\tilde{f}_1^{(2)}(\tilde{x})$  and  $x_0^2$  in  $\tilde{f}_2^{(2)}(\tilde{x})$  are zero. Thus Step 1 and 2 solve two quadratic equation of at least 4 variables.

**Observation 3.** Step 3 requires to solve 3 homogeneous linear equations and 2 homogeneous quadratic equations of  $n+1$  variables. If  $n \geq 3+4=7$ , this

can be done by Step 1 and 2 and linear operations. Note that the coefficients of  $x_0^2, x_0x_1, x_1^2$  in  $\tilde{f}_1^{(3)}(\tilde{x}), \tilde{f}_2^{(3)}(\tilde{x})$  and  $x_0^2$  in  $\tilde{f}_3^{(3)}(\tilde{x})$  are zero. Thus Step 1 to 3 solve three quadratic equation of at least 7 variables.

**Observation 4.** Suppose that Step 1 to Step  $t$  ( $1 \leq t \leq m$ ) solves  $t$  quadratic equations of at least  $r_t$  variables. Since Step  $t + 1$  requires to solve  $t + 1$  homogeneous linear equations and  $t$  homogeneous quadratic equations of  $n + 1$  variables. If  $n \geq r_t + t$ , this can be done by Step 1 to Step  $t$  and linear operations. Note that the coefficients of  $x_0^2, x_0x_1, x_1^2$  in  $\tilde{f}_1^{(t+1)}(\tilde{x}), \dots, \tilde{f}_t^{(t+1)}(\tilde{x})$  and  $x_0^2$  in  $\tilde{f}_{t+1}^{(t+1)}(\tilde{x})$  are zero. Thus Step 1 to  $t + 1$  solve  $t + 1$  quadratic equation of at least  $n \geq r_t + t$  variables. Since  $t_1 = 2$  and Step 1 and 2 solves two quadratic equations, we see that Step 1 to Step  $m$  solves  $m$  quadratic equations of at least  $t_m = m(m + 1)/2 + 1$  variables.

**Observation 5.** We now estimate the complexity of this algorithm. It is too difficult in general since we do not know how many times one computes  $\tilde{U}_1$  in each step. Then, for simplicity, we do it under the assumption that one computes  $\tilde{U}_1$  once when  $q$  is even and twice when  $q$  is odd in all steps, because the probability that univariate quadratic equation has a solution is almost 1 if  $q$  is even and  $1/2$  if  $q$  is odd. Let  $c_t$  be the complexity in Step  $t$ . Since Step  $t$  includes Step 1 to Step  $t - 1$  again and linear operations, we have

$$c_t = \begin{cases} c_1 + c_2 + \dots + c_{t-1} + (\text{polyn}), & (2 \mid q), \\ 2(c_1 + c_2 + \dots + c_{t-1}) + (\text{polyn}), & (2 \nmid q). \end{cases}$$

Thus it is easy to see that  $c_t = O(2^t)$  when  $q$  is even and  $c_t = O(3^t)$  when  $q$  is odd. This means that the complexity of Algorithm 2 is roughly estimated by  $c_1 + \dots + c_m = O(2^m)$  for even  $q$  and  $O(3^m)$  for odd  $q$ .

## 6 Solving equations over small fields

In Section 4 and 5, we propose algorithms to solve equations for general finite fields. When  $q$  is not very larger than  $n$  and  $m$ , one can solve equations effectively by inserting the exhaustive search in each step of Algorithm 1 little by little if  $n$  is (not very) smaller than as described in the table at the end of Section 4.

As examples, we describe how to solve quadratic equations with  $(q, m, n) = (16, 64, 16)$  and  $(16, 48, 16)$ , which are used for UOV suggested in [6]. For simplicity to estimate the complexity, suppose that the complexity of Algorithm C is  $n(n - M)^3/3$  since  $\Omega(n) \sim n^3/3$  for the classical Gaussian elimination. Of course, using the faster elimination algorithm, the complexity becomes smaller.

### 6.1 Solving equations of $(q, m, n) = (16, 64, 16)$ .

**Aim.** Find a solution  $x \in k^{64}$  of  $f_1(x) = 0, \dots, f_{16}(x) = 0$ .

**Step 1.** Use Algorithm C to find  $V_1 : k^{65} \rightarrow k^{65}$  such that the coefficients of  $x_i x_j$  ( $0 \leq i, j \leq 7$ ) in  $\tilde{f}_1(V_1 \tilde{x}), \dots, \tilde{f}_8(V_1 \tilde{x})$  are zero. Put  $x^{(1)} := (x_0, \dots, x_7)^t$  and  $f_l^{(1)}(x^{(1)}) := \tilde{f}_l(V_1(x_0, \dots, x_7, 0, \dots, 0)^t)$ .

**Step 2.** Use Algorithm C to find  $V_2 : k^8 \rightarrow k^8$  such that the coefficients of  $x_i x_j$  ( $0 \leq i, j \leq 2$ ) in  $f_9^{(1)}(V_2 x^{(1)})$ ,  $f_{10}^{(1)}(V_2 x^{(1)})$  are zero. Put  $x^{(2)} := (x_0, x_1, x_2)^t$  and  $f_l^{(2)}(x^{(2)}) := f_l^{(1)}(V_2(x_0, x_1, x_2, 0, \dots, 0)^t)$ .

**Step 3.** Find  $x^{(2)} = (1, x_1, x_2)^t$  such that  $f_{11}^{(2)}(x^{(2)}) = 0$ . Check whether  $f_{12}^{(2)}(x^{(2)}) = 0$  for the same  $x^{(2)}$ . If so, go to the next step, and if not, change  $x^{(2)}$  until  $f_{12}^{(2)}(x^{(2)}) = 0$ .

**Step 4.** Check whether  $f_{13}^{(2)}(x^{(2)}) = f_{14}^{(2)}(x^{(2)}) = f_{15}^{(2)}(x^{(2)}) = 0$ . If so, go to the next step, and if not, go back to Step 2.

**Step 5.** Check whether  $f_{16}^{(2)}(x^{(2)}) = 0$ . If so, finish this algorithm, and if not, go back to Step 1.

**Observation.** The complexity in Step 1 and 2 are respectively  $65 \times 57^3/3$  and  $8 \times 5^4/3$ . Step 3 requires to find a square root and the exhaustive search for  $f_{12}^{(2)}(x^{(2)}) = 0$ . Since the probability that  $f_{12}^{(2)}(x^{(2)}) = 0$  for randomly chosen  $x^{(2)}$  is about  $q^{-1}$ , the complexity in Step 3 is  $\log q \times q = 2^5$  in average. Similarly, since the probabilities that  $f_{13}^{(2)}(x^{(2)}) = f_{14}^{(2)}(x^{(2)}) = f_{15}^{(2)}(x^{(2)}) = 0$  and  $f_{16}^{(2)}(x^{(2)}) = 0$  are respectively  $q^{-3}$  and, one repeats Step 2 by  $q^3 = 2^{12}$  times and Step 1 by  $q = 2^4$  times on average. Thus the complexity of this approach is

$$2^4 \times (65 \times 57^3/3 + 2^{12} \times (8 \times 5^4/3 + 2^5)) \sim 2^{26.4}.$$

## 6.2 Solving equations of $(q, m, n) = (16, 48, 16)$ .

**Aim.** Find a solution  $x \in k^{48}$  of  $f_1(x) = 0, \dots, f_{16}(x) = 0$ .

**Step 1.** Use Algorithm C to find  $V_1 : k^{49} \rightarrow k^{49}$  such that the coefficients of  $x_i x_j$  ( $0 \leq i, j \leq 6$ ) in  $\tilde{f}_1(V_1 \tilde{x}), \dots, \tilde{f}_6(V_1 \tilde{x})$  are zero. Put  $x^{(1)} := (x_0, \dots, x_6)^t$  and  $f_l^{(1)}(x^{(1)}) := \tilde{f}_l(V_1(x_0, \dots, x_6, 0, \dots, 0)^t)$ .

**Step 2.** Use Algorithm C to find  $V_2 : k^7 \rightarrow k^7$  such that the coefficients of  $x_i x_j$  ( $0 \leq i, j \leq 2$ ) in  $f_7^{(1)}(V_2 x^{(1)})$ ,  $f_8^{(1)}(V_2 x^{(1)})$  are zero. Put  $x^{(2)} := (x_0, x_1, x_2)^t$  and  $f_l^{(2)}(x^{(2)}) := f_l^{(1)}(V_2(x_0, x_1, x_2, 0, \dots, 0)^t)$ .

**Step 3.** Find  $x^{(2)} = (1, x_1, x_2)^t$  such that  $f_9^{(2)}(x^{(2)}) = 0$ . Check whether  $f_{10}^{(2)}(x^{(2)}) = 0$  for the same  $x^{(2)}$ . If so, go to the next step, and if not, change  $x^{(2)}$  until  $f_{10}^{(2)}(x^{(2)}) = 0$ .

**Step 4.** Check whether  $f_{11}^{(2)}(x^{(2)}) = f_{12}^{(2)}(x^{(2)}) = 0$ . If so, go to the next step, and if not, go back to Step 2.

**Step 5.** Check whether  $f_{13}^{(2)}(x^{(2)}) = \dots = f_{16}^{(2)}(x^{(2)}) = 0$ . If so, finish this algorithm, and if not, go back to Step 1.

**Observation.** The complexity in Step 1 and 2 are respectively  $49 \times 42^3/3$  and  $7 \times 4^4/3$ . Step 3 requires to find a square root and the exhaustive search for  $f_{10}^{(2)}(x^{(2)}) = 0$ . Since the probability that  $f_{10}^{(2)}(x^{(2)}) = 0$  for randomly chosen  $x^{(2)}$  is about  $q^{-1}$ , the complexity in Step 3 is  $\log q \times q = 2^5$  in average. Similarly, since the probabilities that  $f_{11}^{(2)}(x^{(2)}) = f_{12}^{(2)}(x^{(2)}) = 0$  and  $f_{13}^{(2)}(x^{(2)}) = \dots = f_{16}^{(2)}(x^{(2)}) = 0$  are respectively  $q^{-2}$  and  $q^{-4}$ , one repeats Step 2 by  $q^2 = 2^8$  times

and Step 1 by  $q^4 = 2^{16}$  times on average. Thus the complexity of this approach is

$$2^{16} \times (49 \times 42^3/3 + 2^8 \times (7 \times 4^3/3 + 2^5)) \sim 2^{36.4}.$$

Remark that the complexity to solve the quadratic equations with  $(q, m, n) = (16, 64, 16)$  and  $(16, 48, 16)$  have been studied in [1] and [4] to analyze the security of UOV with such parameters proposed in [6]. We summarize the complexities of the attacks by [1], [4] and our approach in the following table.

$(q, n, m)$	$(16, 48, 16)$	$(16, 64, 16)$
exhaustive	$2^{64}$	$2^{64}$
Courtois et al. [1]	$2^{46}$	$2^{42}$
Faugère-Perret [4]	$2^{40.5}$	$2^{40.5}$
Our attack	$2^{36.4}$	$2^{26.4}$

## 7 Conclusion

In the present paper, we propose two algorithms to solve quadratic equations when  $n$  is much larger than  $m$ . Though we reduce the required  $n$  compared to the works in [6] and [1], it is still too large to break most cryptosystems based on multivariate quadratic equations. Then it is important to improve our algorithms and to study the lower bound of  $n$  such that  $m$  equations can be solved in polynomial (or effective) time.

## References

1. N. Courtois, L. Goubin, W. Meier and J. Tacier, *Solving underdefined systems of multivariate quadratic equations*, PKC'02, LNCS **2274**, pp.211–227.
2. N. Courtois, A. Klimov, J. Patarin and A. Shamir, *Efficient algorithms for solving overdefined systems of multivariate polynomial equations*, Eurocrypt'00, LNCS **1807**, pp.392–407.
3. N. Courtois and J. Pieprzyk, *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations*, Asiacrypt'02, LNCS **2501**, pp. 267–287.
4. J. Faugère and L. Perret, *On the security of UOV*, Proceedings of SCC'08, pp.103–109.
5. J. Ding, J. Gower and D. Schmidt, *Multivariate public key cryptosystems*, Advances in Information Security, Springer, 2006.
6. A. Kipnis, J. Patarin and L. Goubin, *Unbalanced Oil and Vinegar Signature Schemes*, Eurocrypt'99, LNCS **1592** (1999), pp. 206–222, extended in [citeseer/231623.html](http://citeseer.231623.html), 2003-06-11.
7. S. Tsujii, T. Kaneko, K. Tadaki and M. Gotaishi, *Design Policy of MPKC based on Piece in Hand Concept (in Japanese)*, IEICE Technical Report **108** (2008), pp.15–22.