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

Yasufumi Hashimoto *

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 give 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 much less than the exhaustive search.

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.

For this topic, there has 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

*Partially supported by JSPS Grant-in-Aid for Young Scientists (B) no. 20740027.
Keywords. under-defined multivariate quadratic equations

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 an algorithm to solve quadratic equations when $n \geq m(m+1)$ and the characteristic of the field is even. Note that the characteristic is odd, their algorithm requires the complexity $O(2^m \times (\text{polynomial}))$. Although Courtois et al. [1] modified it more effectively for odd characteristic cases, its modification requires much more variables.

In the present paper, we give two algorithms to solve quadratic equation 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 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 order 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.
$x = (x_1, \cdots, x_n)^t \in k^n$.
$\tilde{x} = (x_0, x_1, \cdots, x_n)^t \in k^{n+1}$.
$f_l(x) \in k$ $(1 \leq l \leq m)$: 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 $\tilde{f}_l(1, x_1, \cdots, x_n) = f_l(x_1, \cdots, x_n)$.
$e_i := (\underbrace{0, \cdots, 0}_{i-1}, 1, 0, \cdots, 0) \in k^n$ $(1 \leq i \leq n)$.
$\tilde{e}_i := (\underbrace{0, \cdots, 0}_{i}, 1, 0, \cdots, 0) \in k^{n+1}$ $(0 \leq i \leq n)$.
$a_i = (a_{1i}, \cdots, a_{ni})^t \in k^n$ $(1 \leq i \leq n)$: a vector with $a_{ii} \neq 0$.
$\tilde{a}_i = (\tilde{a}_{0i}, \cdots, \tilde{a}_{ni})^t \in k^{n+1}$ $(0 \leq i \leq n)$: a vector with $\tilde{a}_{ii} \neq 0$.
$U_i := (e_1, \cdots, e_{i-1}, a_i, e_{i+1}, \cdots, e_n)$: an invertible linear map such that $x_i \mapsto a_{1i}x_1 + \cdots + a_{ni}x_n$ and $x_j \mapsto x_j$ for $j \neq i$.
$\tilde{U}_i := (\tilde{e}_0, \cdots, \tilde{e}_{i-1}, \tilde{a}_i, \tilde{e}_{i+1}, \cdots, \tilde{e}_n)$: an invertible linear map such that $\tilde{x}_i \mapsto \tilde{a}_{0i}\tilde{x}_0 + \tilde{a}_{1i}\tilde{x}_1 + \cdots + \tilde{a}_{ni}\tilde{x}_n$ 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

For convenience, we prepare the following elementary facts in the undergraduate linear algebra.

Fact 1. Let $g(x) := \sum_{1\leq i,j\leq n} g_{ij}x_ix_j$ be a homogeneous quadratic form of $x = (x_1, \cdots, x_n)^t$ over k ($g_{ij} \in k$) 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 $g(x) = x^tGx$.

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

Fact 1 and 2 yield the following fact.

Fact 3. Let $g(x), G$ be as in Fact 1 and U_i as in Section 2.1. Denote by $g(U_lx) = \sum_{1\leq i,j\leq x} g_{ij}^{(l)}x_ix_j$. Then we have $g_{ll}^{(l)} = a_l^tGa_l = g(u_l)$, $g_{il}^{(l)} = e_i^tGa_l + a_l^tGe_i$ ($i \neq l$) 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 give an algorithm proposed by Kipnis-Patarin-Goubin [6] to solve m quadratic equations with n variables for $n > m(m+1)$.

Step 1. Find U_2 such that the coefficients of x_1x_2 in $f_1(U_2x), \cdots, f_m(U_2x)$ are zero. According to Fact 3, we see that this requires to solve m homogeneous linear equations with n variables.

Step 2. Put $f_l^{(2)}(x) := f_l(U_2x)$. Find U_3 such that the coefficients of x_1x_3, x_2x_3 in $f_1^{(2)}(U_3x), \cdots, f_m^{(2)}(U_3x)$ are zero. Similarly, this requires to solve $2m$ homogeneous linear equations with n variables.

Step 3. Put $f_l^{(3)}(x) := f_l^{(2)}(U_3x)$. Find U_4 such that the coefficients of x_1x_4, x_2x_4, x_3x_4 in $f_1^{(3)}(U_4x), \cdots, f_m^{(3)}(U_4x)$ are zero. This requires to solve $3m$ homogeneous linear equations with n variables.

Continue similar computations.

Step $m-1$. $f_l^{(m-1)}(x) := f_l^{(m-2)}(U_{m-1}x)$. Find U_m such that the coefficients of $x_1x_m, x_2x_m, \cdots, x_{m-1}x_m$ in $f_1^{(m)}(U_mx), \cdots, f_m^{(m)}(U_mx)$ are zero. This requires to solve $m(m-1)$ homogeneous linear equations with n variables.

Note that, until Step $m-1$, we have to solve at most $m(m-1)$ homogeneous linear equations of n variables. Then we need $n > m(m-1)$ at this time.

Step m. Put $g_l(x) := f_l(U_2U_3\cdots U_mx)$. The coefficients of x_ix_j ($1 \leq i < j \leq m$) in $g_l(x)$ are zero. Substitute values into $x_{m+1}, \cdots, x_n$ such that $g_1(x), g_2(x), \cdots, g_m(x)$ are linear combinations of $x_1^2, \cdots, x_m^2$ and constants, without the monomials $x_1, \cdots, x_m$. To find such $x_{m+1}, \cdots, x_n$, we have to solve m^2 linear equations of $n-m$ variables. Then we need $n - m > m^2$, namely $n \geq m(m+1)$.

Step $m+1$. Since $g_l(x)$'s are linear combinations of $x_1^2, \cdots, x_m^2$ and constants, reduce the problem solving $g_l(x) = 0$ for $1 \leq l \leq m$ to the problem solving $x_1^2 = (\text{const}), \cdots, x_m^2 = (\text{const})$ by linear operations.

After finding the square roots of them, we can find a solution of given equations. When q is even, this algorithm will give a solution in polynomial time since any element of k has a square root. On the other hand when q is odd, this will require $2^m \times$ (polynomial) operations in average because almost half elements of k do not have square roots.

Note that Courtois et al. modified this algorithm for odd characteristic k. The complexity of the modified version is about $2^{40} times$(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 $n \geq$ (about) $m^2 - 2m^{3/2} + 2m$

In this section, we propose an algorithm to solve equations for $n \geq$ (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 $1 \leq l \leq n$, then $(u_{00}^{-1}u_{10}, \cdots, u_{00}^{-1}u_{n0})$ is a solution of $f_1(x) = 0, \cdots, f_m(x) = 0$.

This follows immediately from Fact 1 and 2. Then, instead 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 $x = (x_1, \cdots, x_n)^t \in k^n$. Find an invertible linear transform$U : k^n \to k^n$ such that the coefficients of $x_i x_j$ $(i + j \leq n)$ in $g(Ux)$ are zero.
Step 1. Find U_1 such that the coefficients of x_1^2 is zero. Due to Fact 3, we see that this requires to solve a quadratic homogeneous equation of $(a_{11}, \cdots, a_{n1})$.
Step 2. Put $g^{(1)}(x) := g(U_1 x)$. Find U_2 such that the coefficients of x_1x_2, x_2^2 in $g^{(1)}(U_2x)$ are zero. Similarly, this requires to solve a homogeneous linear equation of $(a_{22}, \cdots, a_{n2})$ and a homogeneous quadratic equation of $(a_{12}, \cdots, a_{n2})$.
Step 3. Put $g^{(3)}(x) := g^{(2)}(U_2x)$. Find U_3 such that the coefficients of x_1x_3, x_2x_3, x_3^2 in $g^{(2)}(U_3x)$ are zero. Similarly, this requires to solve two homogeneous linear equation of $(a_{33}, \cdots, a_{n3})$ and a homogeneous quadratic equation of $(a_{13}, \cdots, a_{n3})$.

Continue such operations until the coefficient o $x_i x_j$ for $1 \leq i, j \leq \lfloor m/2 \rfloor$ are reduced to be zero. Note that, to do so, we need to solve at most $\lfloor m/2 \rfloor$ homogeneous linear equations of $m + 1 - \lfloor m/2 \rfloor$ variables and a homogeneous quadratic equation. Then we see that the complexity of this algorithm until Step $\lfloor m/2 \rfloor$ is less than $\lfloor m/2 \rfloor \Omega(\lfloor m/2 \rfloor)$.

Put $V : k^n \to k^n$ be the linear transform such that the coefficients of $x_i x_j$ $(i, j \leq \lfloor m/2 \rfloor)$ in $g(Vx)$ are zero, and denote by $g^{(\lfloor m/2 \rfloor)}(x) := g(Vx)$.
Step $\lfloor m/2 \rfloor$**+1.** Find $U_{\lfloor m/2 \rfloor + 1}$ such that the coefficients of $x_1 x_{\lfloor m/2 \rfloor + 1}, \cdots, x_{\lfloor m/2 \rfloor - 1} x_{\lfloor m/2 \rfloor + 1}$ in $g^{(\lfloor m/2 \rfloor)}(U_{\lfloor m/2 \rfloor + 1}x)$ are zero. This requires to solve $\lfloor m/2 \rfloor - 1$ homogeneous linear equations of $(a_{\lfloor m/2 \rfloor + 1, \lfloor m/2 \rfloor + 1}, \cdots, a_{m, \lfloor m/2 \rfloor + 1})$.
Step $\lfloor m/2 \rfloor + 2$**.** Put $g^{(\lfloor m/2 \rfloor + 1)}(x) := g^{(\lfloor m/2 \rfloor)}(x)(U_{\lfloor m/2 \rfloor + 1}x)$. Find $U_{\lfloor m/2 \rfloor + 2}$ such that the coefficients of $x_1 x_{\lfloor m/2 \rfloor + 2}, \cdots, x_{\lfloor m/2 \rfloor - 2} x_{\lfloor m/2 \rfloor + 2}$ in $g^{(\lfloor m/2 \rfloor + 1)}(U_{\lfloor m/2 \rfloor + 2}x)$ are zero. This re-

quires to solve $\lfloor m/2 \rfloor - 2$ homogeneous linear equations of $(a_{\lfloor m/2 \rfloor+2,\lfloor m/2 \rfloor+2}, \cdots, a_{m,\lfloor m/2 \rfloor+2})$.

Continuing similar operations, we can find a linear transform U as in Aim. After Step $\lfloor m/2 \rfloor + 1$, we need to solve at most $\lfloor m/2 \rfloor - 1$ linear equations. Then the complexity after Step $\lfloor m/2 \rfloor + 1$ is less than $\lfloor m/2 \rfloor \Omega(\lfloor m/2 \rfloor)$. Therefore the total complexity of this algorithm is less than $m\Omega(\lfloor m/2 \rfloor)$.

Algorithm B.

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

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

and $g_1(x), \cdots, g_M(x)$ quadratic forms of $x = (x_1, \cdots, x_n)^t$. Find an invertible linear transform $U : k^n \to k^n$ such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1(x), \cdots, g_M(x)$ are zero.

Step1. Find an invertible linear transform $V_1 : k^n \to k^n$ such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1(V_1 x)$ are zero. This can be done by Algorithm A. Put $g_l^{(1)} := g_l(V_1 x)$.

In Step2, we want to find V_2 such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1^{(1)}(V_2 x)$ and $g_2^{(1)}(V_2 x)$ are zero, we want to find V_3 in Step 3 such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1^{(2)}(V_3 x), g_2^{(2)}(V_3 x), g_3^{(2)}(V_3 x)$ are zero and so on. To consider recursively, we assume that we can find V' such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1(V'x), \cdots, g_{N-1}(V'x)$ are zero until Step $N-1$. We will describe how to find an invertible V_N such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1^{(N-1)}(V_N x), \cdots, g_N^{(N-1)}(V_N x)$ are zero in Step N.

Step N.

Substep $N-1$. Using Algorithm A, find an invertible linear map $W_1 : k^L \to k^L$ such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L,\ i+j \leq L)$ in $g_N^{(N-1)}(\tilde{W}_1 x)$ are zero, where $\tilde{W}_1 := \begin{pmatrix} W_1 & \\ & I \end{pmatrix}$. Note that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $g_1^{(N-1)}(\tilde{W}_1 x), \cdots, g_{N-1}^{(N-1)}(\tilde{W}_1 x)$ are zero. and the complexity is less than $L\Omega(\lfloor L/2 \rfloor)$. Put $h_l^{(N-1)}(x) := g_l^{(N-1)}(\tilde{W}_1 x)$. Next, find U_L such that the coefficients of $x_i x_j$ $(1 \leq i, j \leq L)$ in $h_l^{(N-1,1)}(U_L x)$ for $1 \leq l \leq N-1$ and of $x_1 x_L$ in $h_N^{(N-1,1)}(U_L x)$ are zero. Due to Fact 3, we see that this requires to solve
(a) $(L-1)(N-1)$ homogeneous linear equations of $(a_{L+1,L}, \cdots, a_{n,L})$,
(b) 1 homogeneous linear equation of $(a_{L,L}, \cdots, a_{n,L})$,
(c) $N-1$ homogeneous quadratic equations of $(a_{1,L}, \cdots, a_{n,L})$ in the forms

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

In order to solve the equations (a),(b) and (c), first solve (a) and find $a_{L+1,L}, \cdots, a_{n,L}$. Then $a_{L,L}$ is automatically determined by (b). Substituting such values to (c), the quadratic

equations (c) become $N-1$ linear equation of $(a_{1,L},\cdots,a_{L-1,L})$. Thus we can claim that when $n-L>(L-1)(N-1)$ and $L\geq N$, this substep works with the complexity less than $L\Omega(\lfloor L/2\rfloor)+\Omega((L-1)(N-1))+\Omega(L-1)$. Put $g_l^{(N-1,1)}(x):=h_l^{(N-1)}(U_Lx)$.

Substep N**–2.** Using Algorithm A, find an invertible linear map $W_2:k^{L-1}\to k^{L-1}$ such that the coefficients of x_ix_j $(1\leq i,j\leq L,\ i+j\leq L+1)$ in $g_N^{(N-1,1)}(\tilde{W}_2x)$ are zero, where $\tilde{W}_2=\begin{pmatrix}1 & & \\ & W_2 & \\ & & I\end{pmatrix}$. This requires the complexity less than $(L-1)\Omega(\lfloor(L-1)/2\rfloor)$. Put $h_l^{(N-1,1)}(x):=g_l^{(N-1,1)}(\tilde{W}_2x)$. Next, find U_L such that the coefficients of x_ix_j $(1\leq i,j\leq L)$ in $g_l^{(N-1,2)}(U_Lx)$ for $1\leq l\leq N-1$ and of x_1x_L,x_2x_L in $g_N^{(N-1,2)}(U_Lx)$ are zero. Finding such U_L requires to solve (a) $(L-1)(N-1)+1$ homogeneous linear equations of $(a_{L+1,L},\cdots,a_{n,L})$,
(b) 1 homogeneous linear equation of $(a_{L,L},\cdots,a_{n,L})$,
(c) $N-1$ homogeneous quadratic equations of $(a_{1,L},\cdots,a_{n,L})$ in the forms

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

Similar to the previous substep, one can solve (a), (b) and (c) when $n-L>(L-1)(N-1)+1$ and $L\geq N$ and the complexity in this substep is less than $(L-1)\Omega(\lfloor(L-1)/2\rfloor)+\Omega((L-1)(N-1)+1)+\Omega(L-1)$. Put $g_l^{(N-1,2)}(x):=h_l^{(N-1,1)}(U_Lx)$.

Similarly in Substep N–3, one reduces the coefficients of x_ix_j with $i+j=n+2$, $i,j\leq L$ and the coefficient of x_3x_L in $g_l^{(N-1,2)}(x)$ to be zero. Continue such operations and suppose that we can find a linear transform V until Substep $N-L-1$ such that the coefficients of x_ix_j $(i,j\leq L)$ in $g_1^{(N-1)}(Vx),\cdots,g_{N-1}^{(N-1)}(Vx)$ and the coefficients of x_ix_j $(i,j\leq L)$ except x_L^2 in $g_N^{(N-1)}(Vx)$ are zero Put $g_l^{(N-1,L-1)}(x):=g_l^{(N-1)}(Vx)$.

Substep $N-L$**.** Find U_L such that the coefficients of x_ix_j $(i,j\leq L)$ in $g_1^{(N-1,L-1)}(U_Lx)$, $\cdots,g_N^{(N-1,L-1)}(U_Lx)$ are zero. According to Fact 3, we see that this requires to solve
(a) $(L-1)N$ homogeneous linear equations of $(a_{L+1,L},\cdots,a_{n,L})$,
(b) $N-1$ homogeneous quadratic equations of $(a_{1,L},\cdots,a_{n,L})$ in the forms

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

(c) a homogeneous quadratic equation of $(a_{1,L},\cdots,a_{n,L})$ in the form

$$\begin{aligned}x_L^2+\sum_{i=1}^{L}a_{i,L}\times\big(&\text{linear form of }(a_{L+1,L},\cdots,a_{n,L})\big)\\+\big(&\text{quadratic form of }(a_{L+1,L},\cdots,a_{n,L})\big)=0,\end{aligned}$$

When $n-L>(L-1)N$, we can find a non-trivial solution of (a). Putting it into (b), we can reduce the quadratic equations (b) to linear equations. If $N\leq L$, express $a_{1,L},\cdots,a_{L-1,L}$ by linear combinations of $a_{L,L}$ and constants. Substitute them into (c)

and solve the quadratic equation of $a_{L,L}$. If it does not have solutions, change the choice of $(a_{L+1,L}, \cdots, a_{n,L})$ or go back to the previous substep and try again until the non-trivial $a_{L,L}$ is found. Then a solution of (a), (b) and (c) will be found. On the other hand, when $n - L = (L-1)N$, the solution of (a) is trivial, namely $a_{L+1,L} = \cdots = a_{n,L} = 0$. This means that (b) and (c) become homogeneous equations of $(a_{1,L}, \cdots, a_{L,L})$. Then $N < L$ is necessary. Thus, in this step, we need the condition that $N < (n-L)/(L-1)$, $N \leq L$ or $N = (n-L)/(L-1)$, $N < L$, namely

$$N \leq \begin{cases} \left\lfloor \dfrac{n-L}{L-1} \right\rfloor, & n \leq L^2 - L, \\ L-1, & L^2 - L + 1 \leq n \leq L^2, \\ L, & n \geq L^2 + 1. \end{cases}$$

Note that the complexity in this substep is $\Omega((L-1)N)+\Omega(N-1)$, and the total complexity in Step N is less than $L(L\Omega(\lfloor L/2 \rfloor)+\Omega((L-1)N)+\Omega(N-1))$. Summing this from $N=1$ to M, we see that the complexity of Algorithm B is less than

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

Since $\Omega(n) \ll n^3$, we can claim that this algorithm works in polynomial time.

Based on Algorithm B, we give 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 $f_1(x) = 0, \cdots, f_m(x) = 0$.
Step 1. Put $L_0 := n+1$ and choose $M_1 < \sqrt{L_0}$. Put

$$L_1 := \min\left(\left\lfloor \frac{L_0}{2} \right\rfloor, \left\lfloor \frac{n+M_1}{M_1+1} \right\rfloor\right).$$

Using Algorithm B, find an invertible linear map $V_1 : k^{L_0} \to k^{L_0}$ such that the coefficients of $x_i x_j$ $(0 \leq i, j \leq L_1 - 1)$ in $\tilde{f}_1(V_1\tilde{x}), \cdots, \tilde{f}_{M_1}(V_1\tilde{x})$ are zero. Put $\tilde{f}_l^{(1)}(\tilde{x}) := \tilde{f}_l(V_1\tilde{x})$.
Step 2. Choose $M_2 < \sqrt{L_1}$. Put

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

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

Continuing such operations until $L_t = 1$, we 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 $1 \leq l \leq M_1 + M_2 + \cdots + M_t$ are zero. From Fact 4, we see that this algorithm solves the equations when $m \leq M_1 + \cdots + M_t \sim$

$n^{1/2} + n^{1/4} + \cdots$, namely $n \geq$ about $m^2 - 2m^{3/2} + 2m$. The complexity of this algorithm is less than $O(n^4) + O(n^2) + O(n) + \cdots \sim O(n^4)$. Then Algorithm 1 works in polynomial time.

The following is the table of the number of variables required to solve m equations.

m	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	$\cdots$
n	1	3	4	9	12	16	20	25	36	49	64	81	90	100	121	144	156	$\cdots$

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

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

Algorithm 2.

Aim. Find a solution $x \in k^n$ of the equations $f_1(x) = 0, \cdots, f_m(x) = 0$.

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. This requires to solve a homogeneous quadratic equation of $n+1$ variables. Then this can be done when $n \geq 2$. 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. This requires to solve two homogeneous linear equations and a homogeneous quadratic equation of $n+1$ variables. Then we need $n \geq 2+2 = 4$. 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 $\tilde{f}_2^{(1,1)}(1, z_2, 0, \cdots, 0) = 0$, 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 until such $z_2 \in k$ appears. It is easy to see 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. Note that Step 1 and 2 solves 2 equations of at least 4 variables.

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. This requires to solve 3 homogeneous linear equations and 2 homogeneous quadratic equations of $n+1$ variables. When $n \geq 3+4$, this can be done by Step 1 and 2. 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 $\tilde{f}_3^{(2,1)}(1, z_3, 0, \cdots, 0) = 0$, 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 until such $z_3 \in k$ appears. It is easy to see 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. Note that Step 1 to 3 solves 3 equations of at least 7 variables.

To consider recursively, suppose that, until Step $N-1$, we can find an invertible linear map $U' : k^{n+1} \to k^{n+1}$ such that the coefficients of x_0^2, x_0x_1, x_1^2 in $\tilde{f}_1(U'\tilde{x}), \cdots, \tilde{f}_{N-2}(U'\tilde{x})$ and of x_0^2 in $\tilde{f}_{N-1}(U'\tilde{x})$ are zero when $n \geq N(N-1)/2+1$. This also means that Step 1 to $N-1$ solves $N-1$ quadratic equations of at least $n \geq N(N-1)/2+1$ variables. Put $\tilde{f}_l^{(N-1)}(\tilde{x}) := \tilde{f}_l(U'\tilde{x})$.

Step N**.** Find $\tilde{U}_1$ such that the coefficients of x_0x_1 and x_1^2 in $\tilde{f}_1^{(N-1)}(\tilde{U}_1\tilde{x}), \cdots, \tilde{f}_{N-1}^{(N-1)}(\tilde{U}_1\tilde{x})$ and the coefficient of x_0x_1 in $\tilde{f}_N^{(N-1)}(\tilde{U}_1\tilde{x})$ are zero. This requires to solve N homogeneous linear equations and $N-1$ homogeneous quadratic equations of $n+1$ variables. If $n \geq N(N+1)/2+1$, this can be done by Step 1 to $N-1$. Put $\tilde{f}_l^{(N-1,1)}(\tilde{x}) := \tilde{f}_l^{(N-1)}(\tilde{U}_1\tilde{x})$. If there is a solution $z_N \in k$ of $\tilde{f}_N^{(N-1,1)}(1, z_N, 0, \cdots, 0) = 0$, denote by $V_N := \begin{pmatrix} 1 & 0 & \\ z_N & 1 & \\ & & I \end{pmatrix}$ and put $\tilde{f}_l^{(N)}(\tilde{x}) := \tilde{f}_l^{(N-1,1)}(V_N\tilde{x})$. If there are no such z_N, take another $\tilde{U}_1$ and repeat until such $z_N \in k$ appears. It is easy to see that the coefficients of x_0^2, x_0x_1, x_1^2 in $\tilde{f}_l^{(N)}(\tilde{x})$ for $1 \leq l \leq N-1$ and x_0^2 in $\tilde{f}_N^{(N)}(\tilde{x})$ are zero. Note that Step 1 to N solves N equations of at least $N(N+1)/2+1$ variables.

Thus we can claim that Algorithm 2 solves quadratic equations when $n \geq m(m+1)/2+1$.

We now estimate the complexity of this algorithm. Let c_N be the complexity in the N-th step. For simplicity, assume that one computes $\tilde{U}_1$ once if q is even and twice if 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. Since the N-th step requires to solve $N-1$ linear equations and $N-1$ quadratic equations, we have

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

Then $c_N = O(2^N)$ when q is even and $c_N = O(3^N)$ when q is odd. Since the complexity of this algorithm is $c_1 + \cdots + c_m$, we can roughly estimate the complexity by $O(2^m)$ when q is even and $O(3^m)$ when q is odd.

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 bigger than n and m, one can solve equations effectively by combining Algorithm B and the exhaustive search if n is 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 our convenience to estimate the complexities roughly, suppose that the complexity of Algorithm B is $n(n-M)^3/3$ since $\Omega(n) \sim n^3/3$ for the classical Gaussian elimination.

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

Step 1. Use Algorithm B to find $V_1 : k^{65} \to k^{65}$ such that the coefficients of x_ix_j $(0 \leq i, j \leq 7)$ in $\tilde{f}_1(V_1\tilde{x}), \cdots, \tilde{f}_8(V_1\tilde{x})$ are zero. The complexity in this step is $65 \times 57^3/3$. Put $x^{(1)} := (x_0, \cdots, x_7)^t$ and $f_l^{(1)}(x^{(1)}) := \tilde{f}_l\big(V_1(x_0, \cdots, x_7, 0, \cdots, 0)^t\big)$. By the choice of V_1, we see that $f_1^{(1)}(x^{(1)}) = \cdots = f_8^{(1)}(x^{(1)}) = 0$ for any $x^{(1)}$.

Step 2. Use Algorithm B to find $V_2 : k^8 \to k^8$ 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. The complexity in this step is $8 \times 5^4/3$. Put $x^{(2)} := (x_0, x_1, x_2)^t$ and $f_l^{(2)}(x^{(2)}) := f_l^{(1)}\big(V_2(x_0, x_1, x_2, 0, \cdots, 0)^t\big)$. By the choice of V_2, we see that $f_9^{(2)}(x^{(2)}) = f_{10}^{(2)}(x^{(2)}) = 0$ for any $x^{(2)}$.
Step 3. Find $x^{(2)} = (1, x_1, x_2)^t$ such that $f_{11}^{(2)}(x^{(2)}) = 0$. This can be done by the algorithm to find a square root. After that 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 $f_{12}^{(2)}(x^{(2)}) = 0$. Since the probability that $f_{12}^{(2)}(x^{(2)}) = 0$ for randomly chosen $x^{(2)}$ is about q^{-1}, the complexity in this step is roughly $\log q \times q = 2^5$.
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. Since the probability that $f_{13}^{(2)}(x^{(2)}) = f_{14}^{(2)}(x^{(2)}) = f_{15}^{(2)}(x^{(2)}) = 0$ is q^{-3}, one may repeat it $q^3 = 2^{12}$ times on average.
Step 5. Check whether $f_{16}^{(2)}(x^{(2)}) = 0$. If so, go to the next step, and if not, go back to Step 1. Since the probability that $f_{16}^{(2)}(x^{(2)}) = 0$ is q^{-1}, one may repeat it $q = 2^4$ times on average.

We finally note that the complexity of this approach is about

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

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

Step 1. Use Algorithm B to find $V_1 : k^{49} \to k^{49}$ such that the coefficients of $x_i x_j$ $(0 \leq i, j \leq 6)$ in $\tilde{f}_1(V_1\tilde{x}), \cdots, \tilde{f}_6(V_1\tilde{x})$ are zero. The complexity in this step is $49 \times 42^3/3$. Put $x^{(1)} := (x_0, \cdots, x_6)^t$ and $f_l^{(1)}(x^{(1)}) := \tilde{f}_l\big(V_1(x_0, \cdots, x_6, 0, \cdots, 0)^t\big)$. By the choice of V_1, we see that $f_1^{(1)}(x^{(1)}) = \cdots = f_6^{(1)}(x^{(1)}) = 0$ for any $x^{(1)}$.
Step 2. Use Algorithm B to find $V_2 : k^7 \to k^7$ 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. The complexity in this step is $7 \times 4^4/3$. Put $x^{(2)} := (x_0, x_1, x_2)^t$ and $f_l^{(2)}(x^{(2)}) := f_l^{(1)}\big(V_2(x_0, x_1, x_2, 0, \cdots, 0)^t\big)$. By the choice of V_2, we see that $f_7^{(2)}(x^{(2)}) = f_8^{(2)}(x^{(2)}) = 0$ for any $x^{(2)}$.
Step 3. Find $x^{(2)} = (1, x_1, x_2)^t$ such that $f_9^{(2)}(x^{(2)}) = 0$. This can be done by the algorithm to find a square root. After that chichi 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 $f_{10}^{(2)}(x^{(2)}) = 0$. Since the probability that $f_{10}^{(2)}(x^{(2)}) = 0$ for randomly chosen $x^{(2)}$ is about q^{-1}, the complexity in this step is roughly $\log q \times q = 2^5$.
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. Since the probability that $f_{11}^{(2)}(x^{(2)}) = f_{12}^{(2)}(x^{(2)}) = 0$ is q^{-2}, one may repeat it $q^2 = 2^8$ times on average.
Step 5. Check whether $f_{13}^{(2)}(x^{(2)}) = \cdots = f_{16}^{(2)}(x^{(2)}) = 0$. If so, go to the next step, and if not, go back to Step 1. Since the probability that $f_{13}^{(2)}(x^{(2)}) = \cdots = f_{16}^{(2)}(x^{(2)}) = 0$ is q^{-4}, one may repeat it $q = 2^{16}$ times on average.

We finally note that the complexity of this approach is about

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

We note that the complexity to solve the 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. The following table summarizes the complexities of the attacks by [1], [4] and our approach.

(q, n, m)	$(16, 48, 16)$	$(16, 64, 16)$
exhaustive	2^{64}	2^{64}
Courtois et al. [1]	2^{46}	2^{42}
Faugére-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 attack against most cryptosystems based on multivariate quadratic equations. Then it is important to improve our algorithms and to study theoretically 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. Faugére 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, 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.

HASHIMOTO, Yasufumi
Institute of Systems, Information Technologies and Nanotechnologies,
7F 2-1-22, Momochihama, Fukuoka 814-0001, JAPAN
e-mail:hasimoto@isit.or.jp