Polynomial XL: A Variant of the XL Algorithm Using Macaulay Matrices over Polynomial Rings

Hiroki Furue[1] and Momonari Kudo[2]

[1] NTT Social Informatics Laboratories, Tokyo, Japan, hiroki.furue@ntt.com **
[2] Fukuoka Institute of Technology, Fukuoka, Japan, m-kudo@fit.ac.jp

Abstract. Solving a system of m multivariate quadratic equations in n variables over finite fields (the MQ problem) is one of the important problems in the theory of computer science. The XL algorithm (XL for short) is a major approach for solving the MQ problem with linearization over a coefficient field. Furthermore, the hybrid approach with XL (h-XL) is a variant of XL guessing some variables beforehand. In this paper, we present a variant of h-XL, which we call the *polynomial XL (PXL)*. In PXL, the whole n variables are divided into k variables to be fixed and the remaining $n-k$ variables as "main variables", and we generate a Macaulay matrix with respect to the $n-k$ main variables over a polynomial ring of the k (sub-)variables. By eliminating some columns of the Macaulay matrix over the polynomial ring before guessing k variables, the amount of operations required for each guessed value can be reduced compared with h-XL. Our complexity analysis of PXL (under some practical assumptions and heuristics) gives a new theoretical bound, and it indicates that PXL could be more efficient than other algorithms in theory on the random system with $n = m$, which is the case of general multivariate signatures. For example, on systems over the finite field with 2^8 elements with $n = m = 80$, the numbers of operations deduced from the theoretical bounds of the hybrid approaches with XL and Wiedemann XL, Crossbred, and PXL with optimal k are estimated as 2^{252}, 2^{234}, 2^{237}, and 2^{220}, respectively.

Keywords: MQ problem, MPKC, XL, hybrid approach, Macaulay matrices

1 Introduction

In the field of computer science, the problem of solving a multivariate polynomial system of degree ≥ 2 over a finite field (*the MP problem*) is one of the most important problems, where "solve" means to find (at least) one root of the system. The particular case where polynomials are all quadratic is called *the MQ problem*, and both the MP and MQ problems are known to be NP-hard [30]. Moreover, the hardness of the MQ problem is nowadays applied to constructing various cryptosystems (e.g., multivariate public key cryptosystems (MPKCs)

** This research was conducted while at the University of Tokyo.

such as UOV [37]). Therefore, the analysis even for the quadratic case is a very important task both in theory and in practice, and thus we mainly focus on solving the MQ problem in this paper.

A precise definition of the MQ problem is the following: Let n and m be positive integers, and let q be a power of a rational prime p. Given a sequence $F = (f_1, \ldots, f_m)$ of m quadratic polynomials $f_1, \ldots, f_m$ in n variables $x_1, \ldots, x_n$ over a finite field $\mathbb{F}_q$ of q elements, the MQ problem requires to find at least one $(a_1, \ldots, a_n) \in \mathbb{F}_q^n$ such that $f_i(a_1, \ldots, a_n) = 0$ for all i with $1 \leq i \leq m$. Throughout the rest of this paper, we deal with only the case of $n \leq m$ (*overdetermined* case). This is because algorithms solving the overdetermined MQ problem can be easily applied to the case of $n > m$, since, after the values of $n - m$ variables are randomly specified, the resulting system will have a solution in most cases. Furthermore, this paper evaluates the efficiency of algorithms solving the MQ problem by substituting specific parameters into the asymptotic complexity formula following the security evaluation for various multivariate signatures [8,9,27].

In the literature, there are various methods for solving the MQ problem such as Gröbner basis method, Linearization, resultant-based method [15, Chapter 3], and Wu's method [53]. In particular, Gröbner basis method is a generic method to solve the MQ problem. The most classical method to compute Gröbner bases is Buchberger's algorithm [11], and ones of the currently most efficient algorithms are Faugère's F_4 and F_5 algorithms [23,24]. When the ideal generated by F is zero-dimensional, namely the number of (affine) roots of F over an algebraic closure of $\mathbb{F}_q$ is finite, once a Gröbner basis for the input F is computed for a given monomial order (typically a graded reverse lexicographic order is chosen for practical efficiency) with the above algorithms, the FGLM conversion [25] enables us to obtain its lexicographical Gröbner basis, from which roots of F can be easily derived [16, Chapter 3].

As a linearization-based algorithm, Courtois et al. [14] proposed the *XL algorithm* at EUROCRYPT 2000, and this algorithm is an extension of Relinearization algorithm [38]. The main idea of XL, which is already used in [41,42] by Lazard in order to analyze Buchberger's algorithm, is: Linearize the given system by regarding each monomial as one variable, and then, similarly to F_4, use linear algebra to the coefficient matrix of the linearized system. More concretely, we first construct a shift $\mathscr{S}$ of F, that is, the set of polynomials of the form $t \cdot f_i$ for all $1 \leq i \leq m$ with monomials t up to given degree. By linearizing the system defined by $\mathscr{S}$, we then generate its coefficient matrix (this matrix is nothing but a *Macaulay matrix* of $\mathscr{S}$), and compute its reduced row echelon form (RREF) by the row reduction (Gaussian elimination). If the shift $\mathscr{S}$ is sufficiently large, then the number of linearly independent polynomials in $\mathscr{S}$ becomes close to the total number of monomials of degree up to the maximal degree of polynomials in $\mathscr{S}$, and hence a univariate equation would be obtained from RREF of the Macaulay matrix. We then solve the obtained univariate equation and repeat such processes with respect to the remaining variables. Note that XL is considered to be a redundant variant of F_4 algorithm (see [1,2] for details). Furthermore, Yang et al. [56] analyzed a variant of the XL algorithm called *Wiedemann XL (WXL)*,

which adopts Wiedemann's algorithm [52] instead of row reduction algorithms in the XL framework. WXL provides another complexity estimate that is used to evaluate the security of various MPKCs such as UOV [9].

One of the most effective improvements of XL is to apply the *hybrid approach* [7,55] (first proposed as FXL in [55] for XL, in which the "F" stands for "fix"), which is proposed as an approach applying an MQ solver such as F_4, F_5, or XL efficiently. This approach fixes the values of k among n variables (say $x_1, \ldots, x_k$), and then solves the remaining system in the $n-k$ variables $x_{k+1}, \ldots, x_n$ using an MQ solver. These processes are iterated until a solution is found. In the case of $n \approx m$, the hybrid approach may be effective, since the gain obtained by working on systems with less variables may overcome the loss due to the exhaustive search on the fixed variables. In this paper, we call the hybrid approach with XL (resp. WXL) *h-XL* (resp. *h-WXL*). Furthermore, Joux and Vitse proposed the Crossbred algorithm as a practical efficient algorithm for solving MQ systems over the binary field in 2017 [35]. This Crossbred is constructed based on h-XL by eliminating parts of Macaulay matrices before fixing the values of some variables. In this paper, we propose a new variant of XL following this direction to further reduce the time complexity.

Our contributions In this paper, we propose a new variant of the XL algorithm, which we call *polynomial XL* (PXL), as an improvement of h-XL. With notation same as in h-XL described above, the main idea of our improvement is the following: Before fixing the values of the variables $x_1, \ldots, x_k$, we partly perform Gaussian elimination on a Macaulay matrix *over the polynomial ring* $\mathbb{F}_q[x_1, \ldots, x_k]$, with keeping $x_1, \ldots, x_k$ as indeterminates. More specifically, for a given MQ system, namely a sequence $F = (f_1, \ldots, f_m) \in \mathbb{F}_q[x_1, \ldots, x_n]^m$ of m quadratic (not necessarily homogeneous) polynomials $f_1, \ldots, f_m$, we first regard each f_i as a polynomial in $(\mathbb{F}_q[x_1, \ldots, x_k])[x_{k+1}, \ldots, x_n]$, and construct a shift of F by multiplying all f_i's by monomials in $x_{k+1}, \ldots, x_n$ (up to some degree). We then generate the Macaulay matrix $\mathcal{PM}$ of the shift with respect to a *graded* monomial order in $x_{k+1}, \ldots, x_n$, where $\mathcal{PM}$ is a *polynomial* matrix with entries in the polynomial ring $\mathbb{F}_q[x_1, \ldots, x_k]$. Here, due to the gradedness of the monomial order, $\mathcal{PM}$ is *almost upper-block triangular*, and all of its (nearly-)diagonal blocks are matrices with entries in $\mathbb{F}_q$, *not* in $\mathbb{F}_q[x_1, \ldots, x_k]$. Thus we can execute row operations on these blocks efficiently, and as a result, we also obtain a partly-reduced matrix. Under some practical assumption and heuristic (Assumption 3 and Heuristic 1) such as the semi-regularity of a polynomial sequence, the size of the uneliminated part of this resulting matrix is expected to be much smaller than that of the original one (e.g., in the case where $n = m = 40$ and $k = 10$, the sizes of the original matrix and the uneliminated part are approximately 2^{30} and 2^{21}, respectively), so that the amount of manipulations for each guessed value can be reduced compared with h-XL. As we will see in Subsection 4.3 below, this enables us to solve the system with smaller complexity for some parameters.

We also discuss the time and space complexities of our PXL, and theoretically compare them with those of h-XL, h-WXL, and Crossbred. Comparing the

time complexities, we show that, under some practical assumptions and heuristic (Assumptions 2 and 3, and Heuristic 1 below) such as the affine semi-regularity of polynomial sequences, our PXL would be the most efficient in theory for the case of $n \approx m$, see Table 1 for details. For example, on the system over $\mathbb{F}_{2^8}$ with $n = m = 80$, the numbers of operations in $\mathbb{F}_q$ required for the execution of h-XL, h-WXL, Crossbred, and PXL are estimated as 2^{252}, 2^{234}, 2^{237}, and 2^{220}, respectively. On the other hand, in terms of the space complexity, PXL might be not well compared to h-WXL since the sparsity of the Macaulay matrix is not maintained through an execution of PXL. Therefore, the relationship between PXL and h-WXL can be seen as a trade-off between time and memory.

Organizations The rest of this paper is organized as follows: Section 2 reviews the XL algorithm and the hybrid approach. Section 3 is devoted to describing the proposed algorithm PXL. We estimate the time complexity, and theoretically compare it with those of h-XL, h-WXL, and Crossbred in Section 4, and Section 5 introduces experimental results obtained by our (unoptimized) implementation of PXL. Finally, Section 6 is devoted to the conclusion, where we summarize the key points and suggest possible future works. Also in Appendix A, we recall semi-regular polynomial sequences and their properties.

2 Preliminaries

In this section, we recall the definition of the XL algorithm [14], and discuss its complexity. We also explain the hybrid approach, which combines an exhaustive search with an MQ solver such as XL.

2.1 Notation and Macaulay matrices

We first fix the notations that are used throughout the rest of this paper. Let $X = \{x_1, \ldots, x_n\}$ be a set of n variables, and $\mathscr{T}(X)$ denote the set of monomials in $x_1, \ldots, x_n$. For each non-negative integer d, we also denote by $\mathscr{T}(X)_d$ (resp. $\mathscr{T}(X)_{\leq d}$) the set of all monomials in $x_1, \ldots, x_n$ of degree d (resp. less than or equal to d). Namely, we set

$$\begin{aligned}
\mathscr{T}(X) &:= \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} \mid (\alpha_1, \ldots, \alpha_n) \in (\mathbb{Z}_{\geq 0})^n\},\\
\mathscr{T}(X)_d &:= \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} \in \mathscr{T}(X) \mid \alpha_1 + \cdots + \alpha_n = d\},\\
\mathscr{T}(X)_{\leq d} &:= \mathscr{T}(X)_0 \cup \cdots \cup \mathscr{T}(X)_d = \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} \in \mathscr{T}(X) \mid \alpha_1 + \cdots + \alpha_n \leq d\}.
\end{aligned}$$

Once $X = \{x_1, \ldots, x_n\}$ is fixed, we may write $\mathscr{T}(X)$, $\mathscr{T}(X)_d$, and $\mathscr{T}(X)_{\leq d}$ as $\mathscr{T}$, $\mathscr{T}_d$, and $\mathscr{T}_{\leq d}$, respectively. For a commutative ring A of unity, we denote by $A[X] = A[x_1, \ldots, x_n]$ the polynomial ring with n variables $X = \{x_1, \ldots, x_n\}$ over A. The total degree of $f \in A[X]$ is denoted by $\deg(f)$, and for a monomial $t \in \mathscr{T}(X)$, let $\mathrm{coeff}(f, t)$ denote the coefficient of t in f. When F is a set or sequence of polynomials in $A[X]$, the ideal of $A[X]$ generated by F is denoted by $\langle F \rangle_{A[X]}$ or simply $\langle F \rangle$. In particular, when F is a finite set $\{f_1, \ldots, f_m\}$,

we denote it by $\langle f_1, \ldots, f_m \rangle_{A[X]}$ or $\langle f_1, \ldots, f_m \rangle$. For a subset or sequence F of polynomials in $A[X]$, and for a subset $T \subset \mathscr{T}(X)$, we set $T \cdot F = \{t \cdot f : t \in T,\ f \in F\}$, which is called the *shift* of F by T (we also call a union of shifts a shift). As a particular but important case, we define the following shifts:

$$\mathscr{S}_d(F) := \bigcup_{f \in F_{\leq d}} \mathscr{T}(X)_{d-\deg(f)} \cdot \{f\} = \{tf : f \in F_{\leq d},\ t \in \mathscr{T}(X)_{d-\deg(f)}\},$$
$$\mathscr{S}_{\leq d}(F) := \mathscr{S}_0(F) \cup \cdots \cup \mathscr{S}(X)_d = \{tf : f \in F_{\leq d},\ t \in \mathscr{T}(X)_{\leq d-\deg(f)}\}$$

with $F_{\leq d} := \{f \in F : \deg(f) \leq d\}$ for each non-negative integer d, where "$\mathscr{S}$" stands for "shift". In the case where $F_{\leq d}$ is empty, we set $\mathscr{S}_d(F) := \{0\}$ and $\mathscr{S}_{\leq d}(F) := \{0\}$. We may write $\mathscr{S}_d(F)$ and $\mathscr{S}_{\leq d}(F)$ simply by $\mathscr{S}_d$ and $\mathscr{S}_{\leq d}$ respectively, when F is fixed.

Here, we recall the definition of Macaulay matrices. Let $\prec$ be a monomial order on $\mathscr{T}(X)$. For a sequence $F = (f_1, \ldots, f_m) \in A[X]^m$ and an ordered subset $T = \{t_1, \ldots, t_\ell\} \subset \mathscr{T}(X)$ with $t_1 \succ \cdots \succ t_\ell$, we define the *Macaulay matrix* $\mathcal{M}_\prec(F, T)$ of F with respect to T as an $(m \times \ell)$-matrix over R whose (i, j)-entry is the coefficient of t_j in f_i, say

$$\mathcal{M}_\prec(F,T) := \begin{array}{c} \\ f_1 \\ \vdots \\ f_m \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & & \cdots & t_\ell \end{array} \\ \begin{pmatrix} \mathrm{coeff}(f_1, t_1) & \cdots & \mathrm{coeff}(f_1, t_\ell) \\ \vdots & & \vdots \\ \mathrm{coeff}(f_m, t_1) & \cdots & \mathrm{coeff}(f_m, t_\ell) \end{pmatrix} \end{array}.$$

When $\prec$ is clear from the context, we simply denote it by $\mathcal{M}(F, T)$.

Conversely, for an $(m \times \ell)$-matrix $M = (a_{i,j})$ over A and for T given as above, let $\mathcal{M}_\prec^{-1}(M, T)$ (or $\mathcal{M}^{-1}(M, T)$ simply) denote a unique list F' of polynomials in $A[X]$ such that $\mathcal{M}_\prec(F', T) = M$, namely, we set $g_i := \sum_{j=1}^{\ell} a_{i,j} t_j$ for $1 \leq i \leq m$, and $\mathcal{M}_\prec^{-1}(M, T) := (g_1, \ldots, g_m)$.

Example 1. Consider the following three quadratic polynomials (over $R = \mathbb{Z}$) in two variables x_1 and x_2:

$$f_1 = 5x_1^2 + 6x_1x_2 + 4x_1 + 5x_2 + 3,$$
$$f_2 = 4x_1^2 + 5x_1x_2 + 3x_2^2 + 6x_1 + 2x_2 + 2,$$
$$f_3 = 2x_1^2 + 4x_1x_2 + 2x_2^2 + 6x_1 + x_2 + 2.$$

When we put $F := (f_1, f_2, f_3)$, we construct a Macaulay matrix of the shift $\mathscr{S}_3 = \mathscr{S}_3(F) = \mathscr{T}_1 \cdot F = \{x_i f_j : 1 \leq i \leq 2,\ 1 \leq j \leq 3\}$, where $\mathscr{T}_1$ is the set of monomials in x_1 and x_2 of degree one. We order elements of $\mathscr{S}_3$ as follows: $\mathscr{S}_3 = \{x_1f_1, x_1f_2, x_1f_3, x_2f_1, x_2f_2, x_2f_3\}$. Let $\prec_{\mathrm{glex}}$ be the graded lexicographic order on the monomials in x_1 and x_2 with $x_1 \succ x_2$, that is, $x_1^{\alpha_1}x_2^{\alpha_2} \prec_{\mathrm{glex}} x_1^{\beta_1}x_2^{\beta_2}$ if $\alpha_1 + \alpha_2 < \beta_1 + \beta_2$, or $\alpha_1 + \alpha_2 = \beta_1 + \beta_2$ and $x_1^{\beta_1}x_2^{\beta_2}$ is greater than $x_1^{\alpha_1}x_2^{\alpha_2}$ with respect to the lexicographical order with $x_1 \succ x_2$. When we order elements of $\mathscr{T}_{\leq 3}$ (which is the set of monomials in $X = \{x_1, x_2\}$ of degree ≤ 3) by $\prec_{\mathrm{glex}}$,

the Macaulay matrix $\mathcal{M}_{\prec_{\mathrm{glex}}}(\mathscr{S}_3, \mathscr{T}_{\leq 3})$ of $\mathscr{S}_3$ with respect to $\mathscr{T}_{\leq 3}$ is given as follows:

$$\mathcal{M}_{\prec_{\mathrm{glex}}}(\mathscr{S}_3, \mathscr{T}_{\leq 3}) = \begin{array}{c} \\ x_1f_1 \\ x_1f_2 \\ x_1f_3 \\ x_2f_1 \\ x_2f_2 \\ x_2f_3 \end{array} \begin{array}{c} \begin{array}{cccccccccc} x_1^3 & x_1^2x_2 & x_1x_2^2 & x_2^3 & x_1^2 & x_1x_2 & x_2^2 & x_1 & x_2 & 1 \end{array} \\ \begin{pmatrix} 5 & 6 & 0 & 0 & 4 & 5 & 0 & 3 & 0 & 0 \\ 4 & 5 & 3 & 0 & 6 & 2 & 0 & 2 & 0 & 0 \\ 2 & 4 & 2 & 0 & 6 & 1 & 0 & 2 & 0 & 0 \\ 0 & 5 & 6 & 0 & 0 & 4 & 5 & 0 & 3 & 0 \\ 0 & 4 & 5 & 3 & 0 & 6 & 2 & 0 & 2 & 0 \\ 0 & 2 & 4 & 2 & 0 & 6 & 1 & 0 & 2 & 0 \end{pmatrix} \end{array}.$$

In the XL algorithm in Subsection 2.2, the reduced row echelon form of a Macaulay matrix of a shift of F is computed, with R a finite field $\mathbb{F}_q$ of order q, where q is a power of a prime. This corresponds to computing a basis G of the $\mathbb{F}_q$-vector space generated by the shift, and clearly the computed basis also generates the ideal $\langle F \rangle_{\mathbb{F}_q[X]}$, i.e., $\langle G \rangle_{\mathbb{F}_q[X]} = \langle F \rangle_{\mathbb{F}_q[X]}$. In general, G computed as above is not necessarily a Gröbner basis of $\langle F \rangle_{\mathbb{F}_q[X]}$, but we will review in Subsection 2.3 below that for sufficiently large shifts, G becomes a Gröbner basis.

2.2 XL algorithm

This subsection briefly reviews *the XL algorithm* (which stands for eXtended Linearizations), which is proposed in [14] by Courtois et al. to find a solution to a system of multivariate polynomials over finite fields. We write down the XL algorithm in Algorithm 1 below, where the notations are the same as in the previous subsections. We also suppose that the input system is zero-dimensional, namely, the input system has only finite (affine) roots over an algebraically closed field. Note also that the input polynomials are assumed to be all quadratic as in the original paper [14], but in fact, their idea is applicable to a general multivariate system of higher degree.

Algorithm 1 (XL, [14, Section 3, Definition 1]).

Input: A sequence $F = (f_1, \ldots, f_m) \in \mathbb{F}_q[x_1, \ldots, x_n]^m$ of (not necessarily homogeneous) quadratic polynomials, and a natural number D with $D \geq 2$.
Output: A solution over $\mathbb{F}_q$ to $f_i(x_1, \ldots, x_n) = 0$ for $1 \leq i \leq m$.

(1) ***Multiply:*** *Computing all the products $t \cdot f_i$ with $t \in \mathscr{T}_{\leq D-2}$, construct the shift $\mathscr{S}_{\leq D} := \mathscr{S}_{\leq D}(F) = \mathscr{T}_{\leq D-2} \cdot F$, which is the shift of F by $\mathscr{T}_{\leq D-2}$.*
(2) ***Linearize:*** *Make the Macaulay matrix $M := \mathcal{M}_{\prec}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D})$ with respect to some elimination monomial order $\prec$ such that all the terms containing one variable (say x_n) are eliminated last. Compute the reduced row echelon form B of M, and put $G := \mathcal{M}_{\prec}^{-1}(B, \mathscr{T}_{\leq D})$. A univariate polynomial $g(x_n)$ in x_n of degree at most D is surely contained in G when D is sufficiently large.*
(3) ***Solve:*** *Compute the roots in $\mathbb{F}_q$ of g by e.g., combining square-free, distinct-degree and equal-degree factorization algorithms such as* [58], [36] *and* [31] *respectively.*

(4) ***Repeat:*** *Substitute a root into x_n, simplify the equations of G, and then find the values of the other variables.*

Note that in the generation of $\mathcal{M}_{\prec}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D})$, one can sort elements in $\mathscr{S}_{\leq D}$ arbitrarily. We also note that, in XL, it suffices to obtain a univariate polynomial in Step (3) to continue the procedures, whence we do not need to compute a Gröbner basis. On the other hand, XL can be described as a redundant variant of F_4, supposing an assumption that the input system F has only one solution over a finite field, see [2] for details. Moreover, we remark that we can use any other monomial order (e.g., a graded monomial order), if we execute only Steps (1) and (2) to obtain a Gröbner basis of $\langle F \rangle$ (in this case, the computation can be viewed as a special case of Lazard's algorithm [41,42]). Even in this case, we can obtain a root easily from the computed Gröbner basis, under an assumption similar to [2], see Remark 1 below for details.

The condition of the natural number D for XL to continue the procedures is discussed in the next subsection.

2.3 Degree bounds for the success of XL

Algorithm 1 has an input parameter D called a *degree bound*, and it is known that the algorithm surely finds a zero of $\langle F \rangle$ for sufficiently large D. This subsection reviews bounds on such D both in theory and in practice. Let $R := K[x_1, \ldots, x_n]$ be the polynomial ring of n variables over a field K, and $F = (f_1, \ldots, f_m)$ be a sequence of *not necessarily homogeneous* polynomials in R of positive degrees $d_1, \ldots d_m$, respectively. We denote by f^{top} the maximal homogeneous part of $f \in R \setminus \{0\}$, and put $F^{\mathrm{top}} := (f_1^{\mathrm{top}}, \ldots, f_m^{\mathrm{top}})$. Put $R' = R[y]$ for an extra variable y for homogenization. We also denote by f^h the homogenization of $f \in R \setminus \{0\}$ by y, say $f^h = y^{\deg(f)} f(x_1/y, \ldots, x_n/y)$, and put $F^h := (f_1^h, \ldots, f_m^h) \in (R')^m$. For each $d \in \mathbb{Z}$, let I_d denote the degree-d homogeneous component of a homogeneous ideal I of R (resp. R'), namely $I_d = I \cap R_d$ (resp. $I_d = I \cap (R')_d$). We put $I_{\leq d} := I \cap R_{\leq d}$ with $R_{\leq d} := \bigoplus_{i=0}^{d} R_i$ for a (not necessarily homogeneous) ideal I of R, and this kind of notation is applied to R' and its arbitrary ideal.

A well-known (theoretical) upper bound is *Dubé's degree bound* [21] given by $D(n, d) := 2\left(\left(d^2/2\right) + d\right)^{2^{n-1}}$ with $d := \max\{\deg(f_i) : 1 \leq i \leq m\}$. For any degree D larger than or equal to the Dubé's bound, the reduced row echelon form of $\mathcal{M}_{\prec}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D})$ with $\mathscr{S}_{\leq D} = \mathscr{S}_{\leq D}(F)$ and $\mathscr{T}_{\leq D} = \mathscr{T}(X)_{\leq D}$ yields a Gröbner basis of $\langle F \rangle$ with respect to an elimination order $\prec$. Hence, for such a D one can obtain a root of F with Algorithm 1.

However, Dubé's degree bound would be impractical under the cryptographic setting, and we here recall quite smaller bounds under the following assumption:

Assumption 1. *The input sequence $F = (f_1, \ldots, f_m)$ is affine semi-regular, namely $F^{\mathrm{top}} = (f_1^{\mathrm{top}}, \ldots, f_m^{\mathrm{top}})$ is semi-regular.*

See Definition 4 in Appendix A below for the definition of affine semi-regular sequences. Semi-regular sequences are important in the theory of solving polynomial systems (cf. [3], [5]), and often (e.g., [33, Section 4.3]) the security of

multivariate cryptosystems is evaluated under Assumption 1. Under Assumption 1, a bound for the success of XL is obtained by considering the rank of the Macaulay matrix $\mathcal{M}_{\prec}(\mathscr{S}_{\leq d}, \mathscr{T}_{\leq d})$, denoted by $\mathrm{rank}(\mathcal{M}_{\prec}(\mathscr{S}_{\leq d}, \mathscr{T}_{\leq d}))$, where $\mathscr{S}_{\leq d} = \mathscr{S}_{\leq d}(F)$ and $\mathscr{T}_{\leq d} = \mathscr{T}(X)_{\leq d}$ with $X = \{x_1, \ldots, x_n\}$. This rank is clearly equal to the dimension $\dim_K(\langle \mathscr{S}_{\leq d}(F)\rangle_K)$ of the K-vector space $\langle \mathscr{S}_{\leq d}(F)\rangle_K$ generated by $\mathscr{S}_{\leq d}(F)$, and it does not depend on the order of the monomials in $\mathscr{T}_{\leq d}$. Thus, we need to investigate $\dim_K(\langle \mathscr{S}_{\leq d}(F)\rangle_K)$. For this, let us first recall the following theorem, whose mathematically rigorous and correct proof is given in [40] (or [39]) by Kudo-Yokoyama:

Theorem 1 ([40, Theorem 1 & 7, Corollary 1], [39, Theorem 1]). *With notation as above, assume that the sequence $F = (f_1, \ldots, f_m)$ of not necessarily homogeneous polynomials satisfies Assumption 1. Let $d_{\mathrm{reg}}(F^{\mathrm{top}})$ denote the degree of regularity for the homogeneous ideal $\langle F^{\mathrm{top}}\rangle_R$, defined as in Definition 2. Then, for any non-negative integer d with $d < d_{\mathrm{reg}}(F^{\mathrm{top}})$, we have*

$$\dim_K(R')_d/\langle F^h\rangle_d = \sum_{i=0}^{d} \dim_K R_d/\langle F^{\mathrm{top}}\rangle_d$$

with $F^h := (f_1^h, \ldots, f_m^h)$. Hence, the Hilbert series $\mathrm{HS}_{R'/\langle F^h\rangle}(z)$ of $R'/\langle F^h\rangle$ satisfies

$$\mathrm{HS}_{R'/\langle F^h\rangle}(z) \equiv \frac{\prod_{j=1}^{m}(1-z^{d_j})}{(1-z)^{n+1}} \pmod{z^D}$$

for $d_j := \deg(f_j)$ and $D := d_{\mathrm{reg}}(F^{\mathrm{top}})$, so that F^h is $d_{\mathrm{reg}}(F^{\mathrm{top}})$-regular. Moreover, if $d_{\mathrm{reg}}(F^{\mathrm{top}}) < \infty$ (which is equivalent to $m \geq n$ under Assumption 1), then the number of projective zeros of $\langle F^h\rangle_{R'}$ is finite at most, whence $\langle F\rangle_R$ is zero-dimensional.

Note that $d_{\mathrm{reg}}(F^{\mathrm{top}})$ in Theorem 1 is easily computed from the Hilbert series given in (A.3), and in fact it does not depend on F^{top} but is determined only by n, m, and $d_1, \ldots, d_m$. From this, for fixed m and $d_1, \ldots, d_m$, we set

$$D_{\mathrm{reg}}^{(n)} := d_{\mathrm{reg}}(F^{\mathrm{top}}) = \min\left\{ d \;\middle|\; \mathrm{coeff}\left(\frac{\prod_{j=1}^{m}(1-z^{d_j})}{(1-z)^n}, t^d\right) \leq 0 \right\},$$

which we interpret as ∞ if $m < n$. In particular, if $d_1 = \cdots = d_m = 2$, we have

$$D_{\mathrm{reg}}^{(n)} = \min\left\{ d \;\middle|\; \mathrm{coeff}\left((1-z)^{m-n}(1+z)^m, t^d\right) \leq 0 \right\}.$$

Here, even if we do not suppose the affine semi-regularity of F, we have

$$\langle F^h\rangle_d = \langle \mathscr{S}_d(F^h)\rangle_K \cong \langle \mathscr{S}_{\leq d}(F)\rangle_K \subset \langle F\rangle_{\leq d}$$

as K-vector spaces, where a K-isomorphism is given by the dehomogenization map $\langle \mathscr{S}_d(F^h)\rangle_K \ni h \longmapsto h|_{y=1} \in \langle \mathscr{S}_{\leq d}(F)\rangle_K$ (see e.g., [17, Section 4] for details), and therefore

$$\dim_K\langle F^h\rangle_d = \dim_K\langle \mathscr{S}_d(F^h)\rangle_K = \dim_K\langle \mathscr{S}_{\leq d}(F)\rangle_K \leq \dim_K\langle F\rangle_{\leq d}.$$

Moreover, it follows that $\dim_K(R')_d = |\mathscr{T}(X \cup \{y\})_d| = \dim_K R_{\leq d} = |\mathscr{T}_{\leq d}|$. Hence, as a corollary of Theorem 1, we obtain the following:

Corollary 1 (cf. [54, Proposition 1]). *Under the same setting and assumptions as in Theorem 1, for any d with $d < D_{\mathrm{reg}}^{(n)} = d_{\mathrm{reg}}(F^{\mathrm{top}})$, we have*

$$|\mathscr{T}_{\leq d}| - \dim_K(\langle \mathscr{S}_{\leq d}(F) \rangle_K) = \mathrm{coeff}\left(\frac{\prod_{j=1}^{m}(1 - z^{d_j})}{(1-z)^{n+1}}, z^d \right).$$

In particular, if the elements of F are all quadratic, then we have

$$|\mathscr{T}_{\leq d}| - \dim_K(\langle \mathscr{S}_{\leq d}(F) \rangle_K) = \mathrm{coeff}\left((1-z)^{m-n-1} (1+z)^m, z^d \right)$$

for any d with $d < D_{\mathrm{reg}}^{(n)}$.

In the context of the above discussion, we here list the following two kinds of bounds on D for which Algorithm 1 finds a solution:

Heuristic but practical bound from Yang-Chen, Ars et al., and Diem's studies. Assuming that F is an affine semi-regular sequence of quadratic polynomials, we consider a sufficient condition that a univariate polynomial in x_n is obtained in Step (2) of Algorithm 1, when we use an elimination order such that $x_n^D, x_n^{D-1}, \ldots, x_n, 1$ are listed at the end. It is straightforward that the last non-zero row vector of the reduced row echelon form of $\mathcal{M}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D})$ yields a univariate equation of x_n if $\mathrm{rank}(\mathcal{M}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D}))$ is larger than the number of columns minus $D+1$, i.e.,

$$\mathrm{rank}(\mathcal{M}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D})) \geq |\mathscr{T}_{\leq D}| - D,$$

equivalently

$$\chi(D) := |\mathscr{T}_{\leq D}| - \dim_K(\langle \mathscr{S}_{\leq D}(F) \rangle_K) \leq D, \tag{2.1}$$

which is used in [17] and [43]. Thus, it follows from Corollary 1 that the minimum D, denoted by D_{XL} here, required for the success of Step (2) of Algorithm 1 is upper-bounded by

$$D_{\mathrm{XL}} \leq D_0 := \min\left\{ d \;\middle|\; \mathrm{coeff}\left((1-z)^{m-n-1} (1+z)^m, z^d \right) \leq d \right\} \tag{2.2}$$

if $D_{\mathrm{XL}} < D_{\mathrm{reg}}^{(n)}$. The condition $\mathrm{coeff}((1-z)^{m-n-1}(1+z)^m, z^d) \leq d$ is equivalent to that the z^d-coefficient of $(1-z)^{m-n-1}(1+z)^m - (1-z)^{-2}$ is negative (cf. [2, Section 5.1]). Note that, even when $D_{\mathrm{XL}} \geq D_{\mathrm{reg}}^{(n)}$, it would be possible that Step (2) of Algorithm 1 produces a univariate polynomial at the degree equal to this upper-bound: See [54, Section 4], where the authors of [54] say "the minimum D required for the reliable termination of XL is given by D_0". From this, we may estimate $D_{\mathrm{XL}} \approx D_0$. Assuming the *Maximum Rank Conjecture* (which is equivalent over an infinite field to Fröberg conjecture [26], see [47] for a proof of

the equivalency), Diem also proved in [17, Theorem 1] that D_0 is a lower bound for (2.1) to be satisfied. One can easily confirm that D_0 tends to be much smaller than Dubé's degree bound (e.g., the value of D_0 on systems with $n = 10$ and $m = 11$ is 11, whereas Dubé's degree bound on the same system is approximately 10^{309}).

Remark 1. In the case where we use a graded monomial order as noted in Subsection 2.2, we consider the inequality $\chi(D) \leq 1$ instead of (2.1) as a sufficient condition for XL to compute a solution, supposing the following (i) and (ii):

(i) F has at most one root (counted with multiplicity) over an algebraic closure $\overline{K}$ of K (cf. [2, Condition 1] for a similar condition).
(ii) F^{top} has no root other than $(0, \ldots, 0)$.

Under these assumptions, there exists a sufficiently large integer d such that the above inequality definitely holds for any D with $D \geq d$. Indeed, it follows from (i) and (ii) that the number of projective zeros over $\overline{K}$ of F^h is also finite (in fact one at most), whence there exists $d > 0$ such that for any D with $D \geq d$, the value of the Hilbert function $\mathrm{HF}_{R'/\langle F^h \rangle}(D) = \dim_K (R'/\langle F^h \rangle)_D = \chi(D)$ is equal to the number of roots (counted with multiplicity) over $\overline{K}$ of F, see e.g., [13, Proposition 3.3.6] or [54, Corollary 10] for a proof (see also [39, Lemma 2.2.2]). In this case, we remark that the reduced Gröbner basis of $\langle F \rangle$ is $\{x_1 - a_1, \ldots, x_n - a_n\}$, where $(a_1, \ldots, a_n)$ is the unique root of F. For $m > n$, we estimate

$$D_{\mathrm{XL}} \approx D_1 := \min \left\{ d \geq 2 \;\middle|\; \mathrm{coeff}\left((1-z)^{m-n-1} (1+z)^m, t^d \right) \leq 1 \right\}, \qquad (2.3)$$

by a discussion following [54, Section 4], similarly to the case of elimination order. Note that the cases $d = 0$ and $d = 1$ are removed in (2.3), since $\chi(0) = 1$ and $\chi(1) = n + 1$. We also note that $D_1 \geq D_0$. We experimentally confirmed that, in most cases, XL for $D = D_1$ computes a Gröbner basis of the input system: In our experiments, we randomly generated sequences $H = (h_1, \ldots, h_m)$ of quadratic non-homogeneous polynomials over $\mathbb{F}_{31}$ with no constant term for several small n and for all m with $n < m \leq 2n$. For each generated sequence H, we choose $(a_1, \ldots, a_n) \in \mathbb{F}_{31}^n$ at random, and then put $f_i := h_i(x_1, \ldots, x_n) - h_i(a_1, \ldots, a_n)$ for $1 \leq i \leq m$ and $F := (f_1, \ldots, f_m)$. Then each sequence F constructed as above would satisfy the above assumptions (i) and (ii) (in fact, F^{top} would be semi-regular) with its unique root $(a_1, \ldots, a_n)$, in most cases. This construction of H and F may correspond to the general construction of multivariate public key encryption (see e.g., [19, Section 2.2], [33, Section 4.3]). Therefore, our experiments would be meaningful.

Expected theoretical bound from Semaev-Tenti and Kudo-Yokoyama's results. We also note that, as a theoretical upper-bound on D_{XL}, we may apply the following upper-bound on the *solving degree* of Gröbner basis computation:

Theorem 2 ([40, Lemma 4], [39, Theorem 3]). *Let $F = (f_1, \ldots, f_m)$ be a (not necessarily semi-regular) sequence of polynomials in $K[x_1, \ldots, x_n]$, and*

$\prec$ be a graded reverse lexicographic order on the monomials in $x_1, \ldots, x_n$. If $d_{\mathrm{reg}}(F^{\mathrm{top}}) < \infty$, then there constructively exists a Buchberger-like algorithm $\mathcal{A}$ for computing a Gröbner basis for F with respect to $\prec$ such that the degree of critical S-pairs (resp. S-polynomials) appearing in the execution of $\mathcal{A}$ is upper-bounded by $2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 1$ (resp. $2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 2$).

These upper-bounds had been proved by Tenti in his PhD thesis [51, Theorem 3.65] (see also [49, Theorem 2.1] by Semaev-Tenti) under some constraints (e.g., F contains field equations $x_i^q - x_i$ for $1 \leq i \leq n$), and Kudo-Yokoyama extended his result to a general case in [40, Section 5] (see also [39, Section 4] for algorithmic details). Since we can interpret the Gröbner basis computation as repeating to execute row reductions on Macaulay matrices as in F_4 [23] and (matrix-)F_5 [24], we *expect* that $D_{\mathrm{XL}} \leq 2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 1$. As for the magnitude relation between D_1 and $2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 1$, they are not equal to each other in general, and both $D_1 < 2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 1$ and $D_1 > 2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 1$ occur depending on parameters; the former case tend to hold as m is larger than n.

Salizzoni also proved in [48] that the solving degree of *mutant algorithms* (tamed in [28]) such as MutantXL [12] and MXL2 [44] is upper-bounded by $d_{\mathrm{reg}}(F^{\mathrm{top}}) + 1$, but this is not the case that we consider in this paper, since we will construct our algorithm based on the original XL [14], not on mutant algorithms.

2.4 Complexity

In this subsection, we estimate the time complexity of (plain) XL together with that of its variant Wiedemann XL (WXL). Here WXL uses Wiedemann's algorithm [52] instead of Gaussian elimination in the XL framework, which was first analyzed in [56]. Wiedemann's algorithm generally solves sparse linear systems more efficiently than Gaussian elimination.

Complexity of XL. We first consider plain XL (Algorithm 1), where the **Linearize** step is clearly dominant in terms of the time complexity. Recall from Subsection 2.3 that XL could output a solution of the input system for D equal to or larger than D_0 given in (2.2), and here we assume to take D to be this bound D_0. In the **Linearize** step, one uses linear algebra to obtain the reduced row echelon form of a Macaulay matrix with $m \cdot \binom{n+D-2}{D-2}$ rows and $\binom{n+D}{D}$ columns. However, in fact, the cost of this step can be estimated as that of Gaussian elimination on a matrix with $\binom{n+D}{D}$ rows and columns, assuming the following practical heuristic as in [45]:

Heuristic 1. *In XL, if we pick rows in $\mathcal{M}(\mathscr{S}_{\leq D}, \mathscr{T}_{\leq D})$ at random under the constraint that we have enough equations at each degree $d \leq D$, then usually we have a linearly independent set.*

From this heuristic, the complexity of XL is roughly estimated as

$$O\left(\binom{n+D}{D}^{\omega}\right), \tag{2.4}$$

where $2 \leq \omega < 3$ is the exponent of matrix multiplication.

Complexity of WXL. According to [9], the complexity of WXL is estimated as

$$O\left(\binom{n}{2} \cdot \binom{n+D}{D}^2\right), \tag{2.5}$$

where D can be taken to be D_0 given in (2.2). (We remove the constant part from the complexity in [9], since we focus on asymptotic complexity.) WXL consumes less memory than the plain XL, since it can deal with the Macaulay matrix as a sparse matrix, and its memory consumption is estimated as $O\left(\binom{n}{2} \cdot \binom{n+D}{D}\right)$, see [52] for details.

2.5 Improving XL via hybrid approach

One of the most effective improvements of XL (Algorithm 1) is to apply the *hybrid approach* [7,55], which is the best known technique for solving the MQ problem. The hybrid approach combines an exhaustive search with an MQ solver, and it was proposed in [7] (resp. [55]) for Gröbner basis algorithms such as F_4 and F_5 (resp. XL). Specifically, given an MQ system of m equations in n variables, the values for k ($0 \leq k \leq n$) variables are randomly guessed and fixed before an MQ solver is applied to the system in the remaining $n-k$ variables; this is repeated until a solution is obtained. The hybrid approach for XL presented in [55] is called FXL, where "F" stands for "fix", and it is constructed by adding the first and last steps below into Algorithm 1:

Algorithm 2 (Hybrid approach with XL (h-XL)).

Input: A sequence $F = (f_1, \ldots, f_m) \in \mathbb{F}_q[x_1, \ldots, x_n]^m$ of (not necessarily homogeneous) quadratic polynomials, the number k of guessed variables, and a degree bound D.

Output: A solution over $\mathbb{F}_q$ to $f_i(x_1, \ldots, x_n) = 0$ for $1 \leq i \leq m$.

(1) ***Fix:*** *Fix the values $a_1, \ldots, a_k \in \mathbb{F}_q$ for the k variables $x_1, \ldots, x_k$ randomly. In the following two steps, we set $f_i^{(\mathbf{a})} := f_i(a_1, \ldots, a_k, x_{k+1}, \ldots, x_n)$ and $F^{(\mathbf{a})} := (f_1^{(\mathbf{a})}, \ldots, f_m^{(\mathbf{a})})$ with $\mathbf{a} = (a_1, \ldots, a_k)$.*

(2) ***Multiply:*** *Construct the shift $\mathscr{S}_{\leq D}^{(k)}(F^{(\mathbf{a})}) := \mathscr{T}_{\leq D-2}^{(k)} \cdot F^{(\mathbf{a})}$, where we set $\mathscr{T}_{\leq D-2}^{(k)} := \mathscr{T}(X^{(k)})_{\leq D-2}$ with $X^{(k)} = \{x_{k+1}, \ldots, x_n\}$.*

(3) ***Linearize:*** *Compute the reduced row echelon form of $\mathcal{M}(\mathscr{S}_{\leq D}^{(k)}(F^{(\mathbf{a})}), \mathscr{T}_{\leq D}^{(k)})$, where we set $\mathscr{T}_{\leq D}^{(k)} := \mathscr{T}(X^{(k)})_{\leq D}$.*

(4) ***Solve:*** *Compute the root of a univariate polynomial obtained in* ***Linearize****.*

(5) ***Repeat:*** *Find the values of the other variables.*

(6) If there exists no solution, return to (1) ***Fix****.*

The complexities of the hybrid approaches using the plain XL and WXL as MQ solvers are estimated as

$$O\left(q^k \cdot \binom{n-k+D}{D}^{\omega}\right), \tag{2.6}$$

$$O\left(q^k \cdot \binom{n-k}{2} \cdot \binom{n-k+D}{D}^2\right), \tag{2.7}$$

respectively, by using the estimations (2.4) and (2.5). Here D can be taken as

$$D_0^{(n-k)} := \min\left\{ d \;\middle|\; \mathrm{coeff}\left((1-t)^{m-(n-k)-1}(1+t)^m, t^d\right) \leq d \right\} \tag{2.8}$$

from (2.2). In the use of the hybrid approach, the number k of guessed variables is chosen such that the function inside brackets in (2.6) or (2.7) takes the minimum value.

2.6 Crossbred Algorithm

This subsection recalls the Crossbred algorithm proposed by Joux and Vitse, which is a practical efficient algorithm for solving MQ systems over the binary field [35]. Our proposed algorithm described in Section 3 follows a framework similar to the Crossbred algorithm. Note that we here change the notation of Crossbred such that it fixes the values of k variables randomly for consistency with the description of our proposed algorithm.

We here roughly describe the Crossbred algorithm. The Crossbred algorithm takes the number k of guessed variables and the degrees d and D with $d \leq D$ as parameters. In this Crossbred algorithm, we perform some linear algebra operations on Macaulay matrices before fixing the values of the k variables as in h-XL. More specifically, for a given MQ system $F \in \mathbb{F}_{2^r}[x_1, \ldots, x_n]^m$, the Crossbred algorithm can be described by the following two steps: The first step generates the Macaulay matrix of the shift of F with degree $\leq D$, and then by linear algebra on the Macaulay matrix obtains a sequence $P = (p_1, \ldots, p_r)$ of some polynomials whose degrees in the remaining $n-k$ variables are lower than or equal to d. The second step then performs linear algebra on the Macaulay matrix of the shift of the polynomials obtained by fixing the value of k variables in F and P with degree $\leq d$. If the second step obtains a univariate polynomial, then one can find a solution as in the plain XL algorithm. This second step is iterated $O(q^k)$ times until one solution is found.

In Subsection 4.3 below, we estimate the complexity of the Crossbred algorithm by Multivariate Quadratic Estimator by the Technology Innovation Institute [22,34]. We refer to [6,20,46] for details on the complexity of the Crossbred algorithm.

3 Main Algorithm

In this section, we propose a new variant of the XL algorithm for solving the MQ problem of m equations in n variables over $\mathbb{F}_q$, in the case where $n \leq m$. We first discuss Macaulay matrices over polynomial rings, and second describe the outline of our proposed algorithm "polynomial XL (PXL)". After that, details of the most technical step will be described in Subsection 3.3, and degree bounds for the success of PXL will be discussed in Subsection 3.4. Furthermore, Subsection 3.5 explains the relationship of PXL with FXL and Crossbred, and Subsection 3.6 gives a toy example. Throughout this section, let

$F = (f_1, \ldots, f_m) \in \mathbb{F}_q[x_1, \ldots, x_n]^m$ be a sequence of m quadratic (and not necessarily homogeneous) polynomials in n variables $x_1, \ldots, x_n$ over $\mathbb{F}_q$, where q is a power of a prime.

3.1 Macaulay matrices over polynomial rings

In this subsection, we fix the notations that are used in the rest of this section. In particular, we construct a Macaulay matrix *over the polynomial ring* $\mathbb{F}_q[x_1, \ldots, x_k]$ with respect to $x_{k+1}, \ldots, x_n$ for $1 \leq k \leq n$, where each entry belongs to $\mathbb{F}_q[x_1, \ldots, x_k]$. Namely, a Macaulay matrix whose coefficient ring is $\mathbb{F}_q[x_1, \ldots, x_k]$ will be constructed. Such a Macaulay matrix, together with our construction, plays a key role in the main algorithm in Subsection 3.2 below. Note that most of the notations given below are similar to those defined in Subsection 2.1 for the case where the coefficient ring is a general ring.

In the following, an integer k is fixed, unless otherwise noted. Similarly to the hybrid approach reviewed in Subsection 2.5, the main algorithm divides $x_1, \ldots, x_n$ into k variables $x_1, \ldots, x_k$ and the remaining $n - k$ variables $x_{k+1}, \ldots, x_n$, and then regards $f_1, \ldots, f_m$ as elements of the polynomial ring $(\mathbb{F}_q[x_1, \ldots, x_k])[x_{k+1}, \ldots, x_n]$. As in Subsection 2.1, we define subsets $\mathscr{T}_d^{(k)}$, $\mathscr{T}_{d';d}^{(k)}$, $\mathscr{T}_{\leq d}^{(k)}$, $\mathscr{S}_d^{(k)}$, $\mathscr{S}_{d';d}^{(k)}$, and $\mathscr{S}_{\leq d}^{(k)}$ of $(\mathbb{F}_q[x_1, \ldots, x_k])\,[x_{k+1}, \ldots, x_n]$ as follows: Putting $X^{(k)} = \{x_{k+1}, \ldots, x_n\}$, we set

$$\mathscr{T}_d^{(k)} := \mathscr{T}(X^{(k)})_d = \left\{ x_{k+1}^{\alpha_{k+1}} \cdots x_n^{\alpha_n} \in \mathscr{T}(X^{(k)}) : \sum_{i=k+1}^{n} \alpha_i = d \right\},$$

$$\mathscr{T}_{d';d}^{(k)} := \mathscr{T}_{d'}^{(k)} \cup \mathscr{T}_{d'+1}^{(k)} \cup \cdots \cup \mathscr{T}_d^{(k)}, \qquad \mathscr{T}_{\leq d}^{(k)} := \mathscr{T}_{0;d}^{(k)} = \mathscr{T}(X^{(k)})_{\leq d}$$

for $0 \leq d' \leq d$, and

$$\mathscr{S}_d^{(k)} := \bigcup_{1 \leq i \leq d} \mathscr{T}(X^{(k)})_{d-2} \cdot \{f_i\} = \{t f_i : 1 \leq i \leq m,\ t \in \mathscr{T}(X^{(k)})_{d-2}\}$$

for $2 \leq d$. We also set $\mathscr{S}_0^{(k)} := \{0\}$, $\mathscr{S}_1^{(k)} := \{0\}$, and

$$\mathscr{S}_{d';d}^{(k)} := \mathscr{S}_{d'}^{(k)} \cup \mathscr{S}_{d'+1}^{(k)} \cup \cdots \cup \mathscr{S}_d^{(k)}, \qquad \mathscr{S}_{\leq d}^{(k)} := \mathscr{S}_{0;d}$$

for $0 \leq d' \leq d$. In particular, $\mathscr{S}_{\leq d}^{(k)}$ is the shift of F by the set $\mathscr{T}_{\leq d-2}^{(k)}$ of monomials in $x_{k+1}, \ldots, x_n$ of degree $\leq d - 2$.

Here, we construct a Macaulay matrix of the shift $\mathscr{S}_{\leq D}^{(k)}$ with respect to $\mathscr{T}_{\leq D}^{(k)}$ for $D \geq 2$, as in the plain XL. For this, unlike the plain XL (mainly adopting an elimination order described in Section 2), we use a *graded* monomial order (e.g., graded lexicographic order), which is a monomial order first comparing the total degree of two monomials. Furthermore, as for the order of elements in $\mathscr{S}_{\leq D}^{(k)}$, we also use an order that first compares the degree of two polynomials.

To simplify the notation, once F, k, and D are fixed, we denote the Macaulay matrix $\mathcal{M}(\mathscr{S}_{\leq D}^{(k)}, \mathscr{T}_{\leq D}^{(k)})$ constructed as above by $\mathcal{PM}$ to emphasize that it is

a *polynomial matrix*, and call it a *Macaulay matrix of F at degree D over* $\mathbb{F}_q[x_1, \ldots, x_k]$. For two integers d_1 and d_2 with $2 \leq d_1 \leq D$ and $0 \leq d_2 \leq D$, we also denote by $\mathcal{PM}[\mathscr{S}_{d_1}^{(k)}, \mathscr{T}_{d_2}^{(k)}]$ the submatrix of $\mathcal{PM}$ whose rows (resp. columns) correspond to polynomials of $\mathscr{S}_{d_1}^{(k)}$ (resp. monomials of $\mathscr{T}_{d_2}^{(k)}$). Then, $\mathcal{PM}$ is divided by submatrices $\mathcal{PM}[\mathscr{S}_{d_1}^{(k)}, \mathscr{T}_{d_2}^{(k)}]$ for $2 \leq d_1 \leq D$ and $0 \leq d_2 \leq D$.

Thanks to our choice of a graded monomial order together with the quadraticity of F, the following lemma holds:

Lemma 1. *For a sequence $F = (f_1, \ldots, f_m)$ of quadratic and not necessarily homogeneous polynomials in $\mathbb{F}_q[x_1, \ldots, x_n]$ and for positive integers k and D with $1 \leq k \leq n$ and $D \geq 2$, let $\mathcal{PM}$ be a Macaulay matrix of F at degree D over $\mathbb{F}_q[x_1, \ldots, x_k]$. Then, for each integer d with $2 \leq d \leq D$, the submatrix $\mathcal{PM}[\mathscr{S}_{d}^{(k)}, \mathscr{T}_{d'}^{(k)}]$ with $d' \notin \{d, d-1, d-2\}$ is a zero matrix, and all elements of $\mathcal{PM}[\mathscr{S}_{d}^{(k)}, \mathscr{T}_{d}^{(k)}]$ belong to $\mathbb{F}_q$.*

Proof. Each f_i is written as

$$f_i = q_i(x_{k+1}, \ldots, x_n) + \sum_{j=k+1}^{n} \ell_{i,j}(x_1, \ldots, x_k) x_j + c_i(x_1, \ldots, x_k) \tag{3.1}$$

for a quadratic form $q_i(x_{k+1}, \ldots, x_n)$ in $\mathbb{F}_q[x_{k+1}, \ldots, x_n]$, linear polynomials $\ell_{i,j}(x_1, \ldots, x_k)$'s in $\mathbb{F}_q[x_1, \ldots, x_k]$, and a quadratic polynomial $c_i(x_1, \ldots, x_k)$ in $\mathbb{F}_q[x_1, \ldots, x_k]$. Therefore, multiplying it by a monomial $t \in \mathscr{T}_{d-2}^{(k)}$ in $x_{k+1}, \ldots, x_n$ of degree $d-2$, we have

$$tf_i = tq_i(x_{k+1}, \ldots, x_n) + \sum_{j=k+1}^{n} \ell_{i,j}(x_1, \ldots, x_k) t x_j + c_i(x_1, \ldots, x_k) t,$$

where tq_i is a form in $\mathbb{F}_q[x_{k+1}, \ldots, x_n]$ of degree d and where each tx_j is a monomial in $\mathbb{F}_q[x_{k+1}, \ldots, x_n]$ of degree $d-1$. This expression of the shift tf_i, which corresponds to a row of $\mathcal{PM}[\mathscr{S}_{d}^{(k)}, \mathscr{T}_{\leq d}^{(k)}]$ and vice versa, implies the assertions of the lemma. □

Due to this lemma, we can partly perform row reduction on $\mathcal{PM}$, which is a key operation of the proposed algorithm in the next subsection.

3.2 Outline of our algorithm PXL

This subsection describes the proposed algorithm polynomial XL (PXL). As in the h-XL described in Subsection 2.5, PXL first sets the first k variables $x_1, \ldots, x_k$ as guessed variables, whereas the main difference between our PXL and h-XL is the following: While h-XL performs row reduction after substituting actual k values to $x_1, \ldots, x_k$, PXL *partly* performs Gaussian elimination *before* fixing k variables. These manipulations are possible due to our construction of Macaulay matrices over $\mathbb{F}_q[x_1, \ldots, x_k]$ described in Lemma 1.

Here, we give the outline of PXL. The notations are same as those in Subsection 3.1.

Algorithm 3 (Polynomial XL).

Input: A sequence $F = (f_1, \ldots, f_m) \in \mathbb{F}_q[x_1, \ldots, x_n]^m$ *of not necessarily homogeneous polynomials of degree* 2*, the number* k *of guessed variables, and a degree bound* D*.*

Output: A solution over $\mathbb{F}_q$ *to* $f_i(x_1, \ldots, x_n) = 0$ *for* $1 \leq i \leq m$*.*

(1) ***Multiply****: Compute the set* $\mathscr{S}^{(k)}_{\leq D}$ *of all the products* $t \cdot f_i$ *with* $t \in \mathscr{T}^{(k)}_{\leq D-2}$*.*

(2) ***Linearize(1)****: Generate* $\mathcal{PM} := \mathcal{M}(\mathscr{S}^{(k)}_{\leq D}, \mathscr{T}^{(k)}_{\leq D})$*, which is the Macaulay matrix of* F *at degree* D *over* $\mathbb{F}_q[x_1, \ldots, x_k]$*, and partly perform Gaussian elimination on it. (The details will be described in Subsection 3.3 below.)*

(3) ***Fix****: Fix randomly the values for the* k *variables* $x_1, \ldots, x_k$ *in the resulting matrix of* **Linearize(1)**.

(4) ***Linearize(2)****: Compute the reduced row echelon form of the resulting matrix of step 3.*

(5) ***Solve****: If step 4 yields a univariate polynomial, compute its root.*

(6) ***Repeat****: Substitute the root, simplify the equations, and then repeat the process to find the values of the other variables.*

(7) If there exists no solution, return to (3) ***Fix****.*

Note that the definition of 'the resulting matrix of **Linearize(1)**' is given in the next paragraph.

Let us here describe only the first two steps, since the last four steps are executed similarly to h-XL. The **Multiply** step generates the shift $\mathscr{S}^{(k)}_{\leq D}$ of F by $\mathscr{T}^{(k)}_{\leq D-2}$, defined in Subsection 3.1, by regarding each polynomial as one in $(\mathbb{F}_q[x_1, \ldots, x_k])[x_{k+1}, \ldots, x_n]$. At the beginning of the **Linearize(1)** step, $\mathcal{PM}$ is a polynomial matrix with entries in the polynomial ring $\mathbb{F}_q[x_1, \ldots, x_k]$, but by Lemma 1 it is almost upper-block triangular, and all of its (nearly-)diagonal blocks are matrices with entries in $\mathbb{F}_q$. By utilizing this property, the **Linearize(1)** step repeats to transform such a block into the row echelon form and to eliminate entries of its upper blocks. After the **Linearize(1)** step, the resulting Macaulay matrix is supposed to be of the form $\begin{pmatrix} I & * \\ 0 & A \end{pmatrix}$, by interchanging rows (and columns). Here I is an identity matrix, and A is a matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$. Then, the last four steps deal with only the submatrix composed of rows and columns including no leading coefficient of the reduced part, which corresponds to A. We call this submatrix A the *resulting matrix of* **Linearize(1)**.

3.3 Details of Linearize(1) step

In this subsection, we describe the details of the **Linearize(1)** step in the proposed algorithm, and show that it works well as row operations on $\mathcal{PM}$. We use the same notations as in Subsection 3.1. In the following, we also denote by $\mathcal{PM}[\mathscr{S}^{(k)}_{d}, \mathscr{T}^{(k)}_{d'}]$ the same part even after $\mathcal{PM}$ is transformed.

The **Linearize(1)** step is mainly performed on each $\mathcal{PM}[\mathscr{S}^{(k)}_{d}, \mathscr{T}^{(k)}_{(d-2);d}]$, starting from $d = D$ down to 2. Each iteration d consists of the following three substeps:

(d)-1. Perform Gaussian elimination on $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$.

(d)-2. Perform the same row operations as those of (d)-1 on the submatrix $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_{(d-2);(d-1)}^{(k)}]$.

(d)-3. Using the *leading coefficients* of the resulting $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$ (namely the reduced row echelon form of the initial $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$), eliminate the corresponding columns of $\mathcal{PM}$. Here, a leading coefficient is the leftmost nonzero entry in each row of a row echelon form of a matrix.

Here, we show that the **Linearize(1)** step described above works well as row operations on $\mathcal{PM}$. Note that for any $3 \leq d \leq D$, the (d)-3 step does not affect the submatrix $\mathcal{PM}[\mathscr{S}_{\leq(d-1)}^{(k)}, \mathscr{T}_d^{(k)}]$, since $\mathcal{PM}[\mathscr{S}_{\leq(d-1)}^{(k)}, \mathscr{T}_d^{(k)}]$ is always a zero matrix by Lemma 1. This indicates that $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_{\leq D}^{(k)}]$ does not change from the original structure at the beginning of the (d)-1 step. Therefore, from Lemma 1, the manipulations in the (d)-1 and (d)-2 steps can be performed correctly and seen as row operations on $\mathcal{PM}$. Furthermore, the (d)-3 step can be also performed correctly, since the leading coefficients of the resulting $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$ belong to $\mathbb{F}_q$. As a result, we have that all the manipulations are practicable and regarded as row operations on the whole $\mathcal{PM}$.

After the **Linearize(1)** step, all manipulations are performed on the resulting matrix of **Linearize(1)** obtained by concatenating rows and columns including no leading coefficient of the row echelon form $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$ with $2 \leq d \leq D$.

Remark 2. As in the XL algorithm, in practice, PXL randomly chooses approximately $|\mathscr{T}_{\leq D}^{(k)}|$ independent rows from the Macaulay matrix with $|\mathscr{S}_{\leq D}^{(k)}|$ rows (namely we suppose a heuristic similar to Heuristic 1), and executes the **Linearize(1)** step on the submatrix composed of chosen row vectors. We then assume that the rank of the resulting matrix of **Linearize(1)** is large enough to yield a univariate equation, and we experimentally confirmed that this assumption is correct in most cases.

3.4 Degree bounds for the success of PXL

This subsection estimates the minimum value D_{PXL} where PXL with input $D = D_{\mathrm{PXL}}$ succeeds in finding a solution, under a practical assumption (Assumption 2 below), which requires conditions similar to (i) and (ii) in Remark 1. Note that the success of PXL means the following: For some evaluation of $\mathbf{a} = (a_1, \ldots, a_k) \in \mathbb{F}_q^k$ to $(x_1, \ldots, x_k)$ in the **Fix** step, the remaining steps finds a solution $(a_{k+1}, \ldots, a_n) \in \mathbb{F}_q^{n-k}$ to the multivariate system in $x_{k+1}, \ldots, x_n$ corresponding to the resulting matrix of the **Linearize(1)** step, and then $(a_1, \ldots, a_n)$ is exactly a solution to the original system.

To estimate the value of D_{PXL}, we discuss the rank of the resulting matrix of **Linearize(1)**. Recall from Subsection 3.2 that the **Linearize(1)** step

transforms the Macaulay matrix into a matrix of the form $\begin{pmatrix} I & * \\ 0 & A \end{pmatrix}$, by interchanging rows (and columns). Here I is an identity matrix, and A is a matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$. The resulting matrix of the **Linearize(1)** step is A, and let α be the number of columns of A. For $\mathbf{a} = (a_1, \ldots, a_k) \in \mathbb{F}_q^k$, we denote by $A^{(\mathbf{a})}$ (resp. $\mathcal{M}(\mathscr{S}_{\leq D}^{(k)}, \mathscr{T}_{\leq D}^{(k)})^{(\mathbf{a})}$) the matrix obtained by substituting $(a_1, \ldots, a_k)$ to $(x_1, \ldots, x_k)$ in A (resp. $\mathcal{M}(\mathscr{S}_{\leq D}^{(k)}, \mathscr{T}_{\leq D}^{(k)})$). Since an evaluation of $x_1, \ldots, x_k$ and elementary row operations over $\mathbb{F}_q[x_1, \ldots, x_k]$ (without multiplying rows by elements in $\mathbb{F}_q[x_1, \ldots, x_k]$ of degree ≥ 1) are commutative, we have the following:

Lemma 2. *With notation as above, we have*

$$\alpha - \mathrm{rank}(A^{(\mathbf{a})}) = |\mathscr{T}_{\leq D}^{(k)}| - \mathrm{rank}(\mathcal{M}(\mathscr{S}_{\leq D}^{(k)}, \mathscr{T}_{\leq D}^{(k)})^{(\mathbf{a})}).$$

Furthermore, we also suppose the following assumption, in order to estimate the value of D_{PXL}:

Assumption 2. *For any $\mathbf{a} = (a_1, \ldots, a_k) \in \mathbb{F}_q^k$, we have that the sequence $F^{(\mathbf{a})} := (f_1^{(\mathbf{a})}, \ldots, f_m^{(\mathbf{a})})$ with $f_i^{(\mathbf{a})} := f_i(a_1, \ldots, a_k, x_{k+1}, \ldots, x_n)$ satisfies the following conditions.*

(i) $F^{(\mathbf{a})}$ has at most one root (counted with multiplicity) over an algebraic closure $\overline{\mathbb{F}}_q$ of $\mathbb{F}_q$.
(ii) $(F^{(\mathbf{a})})^{\mathrm{top}}$ is semi-regular (hence it has no root other than $(0, \ldots, 0) \in \mathbb{F}_q^{n-k}$).

This assumption is expected to hold since $F^{(\mathbf{a})}$ is highly overdetermined (see [33, Section 4.4] for arguments on (ii)). From the above lemma and assumption, we then obtain

$$\alpha - \mathrm{rank}(A^{(\mathbf{a})}) = \mathrm{coeff}\left((1-z)^{m-(n-k)-1}(1+z)^m, z^D\right),$$

if D is lower than $D_{\mathrm{reg}}^{(n-k)}$ as in Corollary 1, where

$$D_{\mathrm{reg}}^{(n-k)} = \min\left\{ d \;\middle|\; \mathrm{coeff}\left((1-z)^{m-(n-k)}(1+z)^m, z^d\right) \leq 0 \right\}. \tag{3.2}$$

Similarly to Remark 1, from Assumption 2, we obtain a *practical* estimation

$$D_{\mathrm{PXL}} \approx D_1^{(n-k)} := \min\left\{ d \geq 2 \;\middle|\; \mathrm{coeff}\left((1-z)^{m-(n-k)-1}(1+z)^m, z^d\right) \leq 1 \right\}. \tag{3.3}$$

Indeed, we experimentally confirmed that PXL finds a solution at $D = D_1^{(n-k)}$. Note that $D_1^{(n-k)} \geq D_0^{(n-k)}$ for the bound $D_0^{(n-k)}$ given in (2.8) for h-XL, but the equality holds in most cases. We also note that, as a *theoretical* upper-bound on D_{PXL} in the *worst* case, we expect from Theorem 2 that $D_{\mathrm{PXL}} \leq 2D_{\mathrm{reg}}^{(n-k)} - 1$.

3.5 Relationships with XFL and Crossbred

Remark 3 (Relationships with XFL [14,54]). We here briefly discuss the relationship between our algorithm PXL and XFL [14,54] proposed as a variant of h-XL. XFL is roughly described as follows: First, the k variables to be fixed are chosen and generate a shift of the given system by all monomials in the remaining $n-k$ variables up to some degree $D-2$. Second, construct a Macaulay matrix (over $\mathbb{F}_q$, but not over $\mathbb{F}_q[x_1, \ldots, x_k]$) of the shift with respect to all monomials in the whole n variables up to the degree D, and then eliminate only monomials of degree D including only the $n-k$ variables. Third, substitute actual values for the k variables, and execute XL for a system in $n-k$ variables obtained by the substitution.

The first step of XFL clearly coincides with the **Multiply** step of our PXL. The main difference of XFL from PXL is the second step: The second step of XFL eliminates monomials in the $n-k$ variables of degree D, and it corresponds to eliminating only $\mathcal{PM}[\mathscr{S}_D^{(k)}, \mathscr{T}_D^{(k)}]$ in the second step of our PXL (in fact, PXL eliminates every block $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$ with $2 \leq d \leq D$). Therefore, PXL can be regarded as an extension of XFL, and the size of the uneliminated part of the second step of XFL is larger than that of PXL.

Remark 4 (Relationships with Crossbred [35]). This remark explains the difference between our PXL and the Crossbred algorithm proposed by Joux and Vitse [35], from the following two points: (i) The parts of Macaulay matrices echelonized before the fixing step, and (ii) Our original structure of Macaulay matrices over the polynomial ring $\mathbb{F}_q[x_1, \ldots, x_k]$, where $x_1, \ldots, x_k$ are variables to be fixed.

First, the parts of Macaulay matrices echelonized before the fixing step for PXL are definitely different from those for Crossbred by the following reason: Crossbred eliminates monomials in which the degree of the remaining $n-k$ variables is larger than a given degree, whereas our algorithm PXL eliminates $\mathrm{rank}\left(\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]\right)$ monomials among degree d monomials in the $n-k$ variables for each $2 \leq d \leq D$. This could cause a difference in the estimations of the degrees D (for which a root is found) and the complexities.

Second, our Macaulay matrix is constructed over $\mathbb{F}_q[x_1, \ldots, x_k]$ by regarding each polynomial in $\mathbb{F}_q[x_1, \ldots, x_n]$ as an element of the polynomial ring in the $n-k$ variables over $\mathbb{F}_q[x_1, \ldots, x_k]$, unlike Crossbred, which uses a Macaulay matrix over the base field $\mathbb{F}_q$. In our Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$, row operations adding a multiple of one row with one variable x_i with $1 \leq i \leq k$ into another row can be realized. By contrast, such a row operation cannot be performed in the standard Macaulay matrix over $\mathbb{F}_q$ clearly. Therefore, row reductions performed in our PXL cannot be duplicated in the standard Macaulay matrix over $\mathbb{F}_q$, and thus row reductions of our PXL performed before fixing the values of k variables are different from those of Crossbred.

3.6 Toy Example

We here solve an MQ system $F = (f_1, f_2, f_3)$ in $n = 3$ variables (x_1, x_2, x_3) over $\mathbb{F}_7$ of $m = 3$ polynomials

$$
\begin{aligned}
f_1 &= 5x_1^2 + 6x_1x_2 + 4x_1x_3 + x_2x_3 + 5x_3^2 + 4x_1 + 5x_2 + 3,\\
f_2 &= 4x_1^2 + 5x_1x_2 + 4x_1x_3 + 3x_2^2 + 5x_2x_3 + x_3^2 + 6x_1 + 2x_2 + 3x_3 + 2,\\
f_3 &= 2x_1^2 + 4x_1x_2 + 2x_2^2 + 6x_3^2 + 6x_1 + x_2 + 3x_3 + 2,
\end{aligned}
$$

by our PXL with $k = 1$ and $D = 4$; in fact, we can take $D = 3$ by $D_1^{(n-k)} = 3$ from (3.3) (or $2d_{\mathrm{reg}}(F^{\mathrm{top}}) - 1 = 3$ in Theorem 2), but we take $D = 4$ for a demonstration.

Then the Macaulay matrix $\mathcal{PM}$ of F at degree D over $\mathbb{F}_7[x_1]$ is given as

	x_2^4	$x_2^3x_3$	$x_2^2x_3^2$	$x_2x_3^3$	x_3^4	x_2^3	$x_2^2x_3$	$x_2x_3^2$	x_3^3
$x_2^2f_1$		1	5			$6x_1+5$	$4x_1$		
$x_2^2f_2$	3	5	1			$5x_1+2$	$4x_1+3$		
$x_2^2f_3$	2		6			$4x_1+1$	3		
$x_2x_3f_1$			1	5			$6x_1+5$	$4x_1$	
$x_2x_3f_2$		3	5	1			$5x_1+2$	$4x_1+3$	
$x_2x_3f_3$		2		6			$4x_1+1$	3	
$x_3^2f_1$				1	5			$6x_1+5$	$4x_1$
$x_3^2f_2$			3	5	1			$5x_1+2$	$4x_1+3$
$x_3^2f_3$			2		6			$4x_1+1$	3
x_2f_1						1	5		
x_2f_2						3	5	1	
x_2f_3						2		6	
x_3f_1							1	5	
x_3f_2							3	5	1
x_3f_3							2		6
f_1									
f_2									
f_3									

	x_2^2	x_2x_3	x_3^2	x_2	x_3	1
$5x_1^2+4x_1+3$						
$4x_1^2+6x_1+2$						
$2x_1^2+6x_1+2$						
	$5x_1^2+4x_1+3$					
	$4x_1^2+6x_1+2$					
	$2x_1^2+6x_1+2$					
		$5x_1^2+4x_1+3$				
		$4x_1^2+6x_1+2$				
		$2x_1^2+6x_1+2$				
$6x_1+5$		$4x_1$		$5x_1^2+4x_1+3$		
$5x_1+2$		$4x_1+3$		$4x_1^2+6x_1+2$		
$4x_1+1$		3		$2x_1^2+6x_1+2$		
		$6x_1+5$	$4x_1$		$5x_1^2+4x_1+3$	
		$5x_1+2$	$4x_1+3$		$4x_1^2+6x_1+2$	
		$4x_1+1$	3		$2x_1^2+6x_1+2$	
		1	5	$6x_1+5$	$4x_1$	$5x_1^2+4x_1+3$
3		5	1	$5x_1+2$	$4x_1+3$	$4x_1^2+6x_1+2$
2			6	$4x_1+1$	3	$2x_1^2+6x_1+2$

,

and this can be regarded as a block matrix with the following form:

$\mathcal{PM}[\mathscr{T}_4^{(1)}, \mathscr{T}_4^{(1)}]$	$\mathcal{PM}[\mathscr{T}_4^{(1)}, \mathscr{T}_3^{(1)}]$	$\mathcal{PM}[\mathscr{T}_4^{(1)}, \mathscr{T}_2^{(1)}]$	$\mathcal{PM}[\mathscr{T}_4^{(1)}, \mathscr{T}_1^{(1)}]$	$\mathcal{PM}[\mathscr{T}_4^{(1)}, \mathscr{T}_0^{(1)}]$
$\mathcal{PM}[\mathscr{T}_3^{(1)}, \mathscr{T}_4^{(1)}]$	$\mathcal{PM}[\mathscr{T}_3^{(1)}, \mathscr{T}_3^{(1)}]$	$\mathcal{PM}[\mathscr{T}_3^{(1)}, \mathscr{T}_2^{(1)}]$	$\mathcal{PM}[\mathscr{T}_3^{(1)}, \mathscr{T}_1^{(1)}]$	$\mathcal{PM}[\mathscr{T}_3^{(1)}, \mathscr{T}_0^{(1)}]$
$\mathcal{PM}[\mathscr{T}_2^{(1)}, \mathscr{T}_4^{(1)}]$	$\mathcal{PM}[\mathscr{T}_2^{(1)}, \mathscr{T}_3^{(1)}]$	$\mathcal{PM}[\mathscr{T}_2^{(1)}, \mathscr{T}_2^{(1)}]$	$\mathcal{PM}[\mathscr{T}_2^{(1)}, \mathscr{T}_1^{(1)}]$	$\mathcal{PM}[\mathscr{T}_2^{(1)}, \mathscr{T}_0^{(1)}]$

.

In the **Linearize(1)** step, we first perform the Gaussian elimination on $\mathcal{PM}[\mathscr{S}_4^{(1)}, \mathscr{T}_4^{(1)}]$, and then $\mathcal{PM}[\mathscr{S}_4^{(1)}, \mathscr{T}_{2;4}^{(1)}]$ is changed into

$$\begin{array}{c}
\begin{array}{ccccc|ccc|c}
x_2^4 & x_2^3x_3 & x_2^2x_3^2 & x_2x_3^3 & x_3^4 & x_2^3 & x_2^2x_3 & x_2x_3^2 & x_3^3
\end{array} \\
\left(\begin{array}{ccccc|ccc|c}
1 & & & & & 4x_1+3 & 2 & 6x_1+3 & \\
 & 1 & & & & 5x_1+3 & 3x_1+2 & 5x_1+6 & \\
 & & 1 & & & 3x_1+6 & 3x_1+1 & 6x_1+3 & \\
 & & & 1 & & 5x_1+3 & 2x_1+5 & x_1+5 & \\
 & & & & 1 & 6x_1+5 & x_1+6 & x_1 & 5x_1 \\
 & & & & & 2x_1+5 & x_1 & 5x_1+6 & \\
 & & & & & 3x_1+6 & 3 & 4x_1+1 & \\
 & & & & & & 3x_1+6 & 3 & 4x_1+1 \\
 & & & & & 6x_1+5 & 3x_1+3 & 6x_1+2 & 4x_1+2
\end{array}\right.
\begin{array}{c|ccc}
 & x_2^2 & x_2x_3 & x_3^2 \\
\hline
 & 6x_1^2+2x_1+3 & 3x_1^2+x_1 & \\
 & 3x_1^2+x_1+6 & 6x_1^2+2x_1 & \\
 & 6x_1^2+2x_1+5 & 3x_1^2+x_1 & \\
 & 3x_1^2+x_1+6 & 6x_1^2+2x_1+2 & \\
 & 5x_1^2+4x_1+3 & 3x_1^2+x_1+1 & x_1^2+5x_1+2 \\
 & x_1^2+6x_1+5 & 6x_1^2+2x_1 & \\
 & 6x_1^2+2x_1+5 & x_1^2+6x_1+6 & \\
 & & 6x_1^2+2x_1+5 & x_1^2+6x_1+6 \\
 & 5x_1^2+4x_1+3 & 2 & 2x_1^2+3x_1
\end{array}
\end{array}.$$

Note that the first five rows of the above matrix can be ignored after this elimination. We then eliminate elements of $\mathcal{PM}[\mathscr{S}_3^{(1)}, \mathscr{T}_3^{(1)}]$, and then $\mathcal{PM}[\mathscr{S}_3^{(1)}, \mathscr{T}_{1;3}^{(1)}]$ is changed into

$$\begin{array}{c}
\begin{array}{ccc|ccc|cc}
x_2^3 & x_2^2x_3 & x_2x_3^2 & x_3^3 & x_2^2 & x_2x_3 & x_3^2 & x_2 & x_3
\end{array} \\
\left(\begin{array}{ccc|ccc|cc}
1 & & & 4x_1+3 & 2 & 6x_1+3 & 6x_1^2+2x_1+3 & 3x_1^2+x_1 \\
 & 1 & & 5x_1+3 & 3x_1+2 & 5x_1+6 & 3x_1^2+x_1+6 & 6x_1^2+2x_1 \\
 & & 1 & 3x_1+6 & 3x_1+1 & 6x_1+3 & 6x_1^2+2x_1+5 & 3x_1^2+x_1 \\
 & & & 5x_1+3 & 2x_1+5 & x_1+5 & 3x_1^2+x_1+6 & 6x_1^2+2x_1+2 \\
 & & & 3x_1+6 & 3 & 4x_1+1 & 6x_1^2+2x_1+5 & x_1^2+6x_1+6 \\
 & & & 4x_1+3 & 2x_1 & 3x_1+5 & 2x_1^2+5x_1+3 & 5x_1^2+4x_1
\end{array}\right)
\end{array}.$$

Then using the leading coefficient of this partly reduced $\mathcal{PM}[\mathscr{S}_3^{(1)}, \mathscr{T}_3^{(1)}]$, we eliminate nonzero elements of the last four rows of $\mathcal{PM}[\mathscr{S}_4^{(1)}, \mathscr{T}_3^{(1)}]$, and then the last four rows of $\mathcal{PM}[\mathscr{S}_4^{(1)}, \mathscr{T}_{1;2}^{(1)}]$ becomes the following form

$$\begin{array}{c}
\begin{array}{ccc|cc}
x_2^2 & x_2x_3 & x_3^2 & x_2 & x_3
\end{array} \\
\left(\begin{array}{ccc|cc}
x_1^2+6x_1+3 & 2x_1^2+x_1+5 & 2x_1^2+5x_1+2 & 4x_1^3+3x_1^2+4x_1+4 & x_1^3+3x_1 \\
3x_1^2+4x_1 & 3x_1^2+5x_1+1 & 6x_1+3 & 5x_1^2+3x_1+1 & 3x_1^2+x_1 \\
5x_1+3 & 3x_1^2+3x_1+6 & 3x_1^2+3x_1+5 & 3x_1^2+x_1+6 & 5x_1^2+3x_1+5 \\
5x_1^2+4x_1+3 & 2 & 2x_1^2+3x_1 & 5x_1^3+3x_1^2+3x_1+1 & 6x_1^3+3x_1+3
\end{array}\right)
\end{array}.$$

Similarly, we perform the (d)-1, (d)-2, and (d)-3 steps with $d = 2$, and then the resulting matrix of the **Linearize(1)** step is given as

$$\begin{array}{c}
\begin{array}{ccc}
x_2 & x_3 & 1
\end{array} \\
\left(\begin{array}{ccc}
5x_1^3+3x_1+2 & 3x_1^3+5x_1^2+2x_1+3 & 4x_1^4+3x_1^3+6x_1^2+3x_1+6 \\
3x_1^3+2x_1^2+2x_1+3 & 6x_1^3+3x_1^2+6x_1+2 & 2x_1^4+2x_1^3+5x_1^2+2 \\
6x_1^3+6x_1+6 & 4x_1^3+6x_1^2+3x_1+3 & 2x_1^4+3x_1^3+2x_1^2+4x_1+1 \\
3x_1^2+3x_1+1 & 3x_1^2+x_1+1 & x_1^2+2 \\
6x_1^2+2x_1+5 & 6x_1^2+6x_1+3 & 3x_1^3+x_1^2
\end{array}\right)
\end{array}.$$

In the **Fix** step, we here substitute $x_1 = 3$ and obtain the following matrix by the Gaussian elimination

$$\begin{array}{c}
\begin{array}{ccc}
x_2 & x_3 & 1
\end{array} \\
\left(\begin{array}{ccc}
1 & & 4 \\
 & 1 & 1
\end{array}\right).
\end{array}$$

Then, we can obtain two univariate equations $x_2 + 4 = 0$ and $x_3 + 1 = 0$, and thus a solution is $(x_1, x_2, x_3) = (3, 3, 6)$.

4 Complexity

In this section, we first estimate the size of the resulting matrix of **Linearize(1)**. After that, we estimate the time complexity of PXL and compare it with those of h-XL, h-WXL, and Crossbred. We take D to be $D_1^{(n-k)}$ so that PXL can find a solution (as described in Subsection 3.4).

4.1 Size of resulting matrix of Linearize(1)

Let α be the number of columns of the resulting matrix of **Linearize(1)**. In the following, we estimate the value of this α, and show that it can be quite smaller than the number of the columns of the original Macaulay matrix $\mathcal{PM}$. We also describe that the resulting matrix of **Linearize(1)** can be assumed to be an $\alpha \times \alpha$ matrix.

As in the proof of Lemma 1, we denote by q_i the sum of degree-2 terms with respect to $x_{k+1}, \ldots, x_n$ in f_i. Note that $q_i = f_i^{\mathrm{top}}(0, \ldots, 0, x_{k+1}, \ldots, x_n)$ for each i with $1 \leq i \leq m$. Then, by putting $F^{(\mathrm{top},k)} := (q_1, \ldots, q_m)$, it is straightforward that the elements in the shift $\mathscr{S}_d^{(k)}(F^{(\mathrm{top},k)})$, which is equal to $((\mathscr{S}_d^{(k)})^{\mathrm{top}})|_{(x_1,\ldots,x_k)=(0,\ldots,0)}$, correspond to the rows of $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$, for each non-negative integer d. We have that the number of columns eliminated in the step (d)-1 of **Linearize(1)** on $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$ is equal to the rank of $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_d^{(k)}]$, that is $\dim_{\mathbb{F}_q}\langle \mathscr{S}_d^{(k)}(F^{(\mathrm{top},k)})\rangle_{\mathbb{F}_q}$. Therefore, we have

$$\begin{aligned}
\alpha &= |\mathscr{T}_{\leq D}^{(k)}| - \sum_{d=0}^{D} \dim_{\mathbb{F}_q}\langle \mathscr{S}_d^{(k)}(F^{(\mathrm{top},k)})\rangle_{\mathbb{F}_q} \\
&= \sum_{d=0}^{D} \left(|\mathscr{T}_d^{(k)}| - \dim_{\mathbb{F}_q}\langle \mathscr{S}_d^{(k)}(F^{(\mathrm{top},k)})\rangle_{\mathbb{F}_q} \right) \\
&= \sum_{d=0}^{D} \left(\dim_{\mathbb{F}_q}\mathbb{F}_q[x_{k+1}, \ldots, x_n]_d - \dim_{\mathbb{F}_q}\langle F^{(\mathrm{top},k)}\rangle_d \right) \\
&= \sum_{d=0}^{D} \dim_{\mathbb{F}_q}\mathbb{F}_q[x_{k+1}, \ldots, x_n]_d / \langle F^{(\mathrm{top},k)}\rangle_d, \qquad (4.1)
\end{aligned}$$

where we used $\langle \mathscr{S}_d^{(k)}(F^{(\mathrm{top},k)})\rangle_{\mathbb{F}_q} = \langle F^{(\mathrm{top},k)}\rangle_d$ since all the elements in the sequence $F^{(\mathrm{top},k)} = (q_1, \ldots, q_m)$ are homogeneous. Here, we suppose the following:

Assumption 3. *The sequence $F^{(\mathrm{top},k)} = (q_1, \ldots, q_m)$ of homogeneous polynomials in $\mathbb{F}_q[x_{k+1}, \ldots, x_n]$ is semi-regular, where q_i is given in* (3.1) *of the proof of Lemma 1.*

Under this assumption, the value of (4.1) can be estimated as

$$\alpha = \sum_{d=0}^{D} \max\left\{\text{coeff}\left((1-t)^{m-(n-k)}(1+t)^m, t^d\right), 0\right\} \tag{4.2}$$

by Proposition 1. Note that this can be quite smaller than $\binom{n-k+D}{D}$, which is the number of the columns of the whole Macaulay matrix $\mathcal{PM}$. For example, when $n = m = 40$ and $k = 10$, the degree D for which PXL could succeed is estimated as 10 by (3.3), and then α and $\binom{n-k+D}{D}$ are approximately 2^{21} and 2^{30}, respectively.

Recall from Remark 2 that PXL randomly chooses approximately $|\mathscr{T}^{(k)}_{\leq D}|$ independent rows from the whole Macaulay matrix $\mathcal{PM}$. When $\tilde{\mathscr{S}}^{(k)}_d$ denotes the subset of $\mathscr{S}^{(k)}_d$ including polynomials corresponding to randomly chosen rows and $r^{(k)}_d$ denotes the rank of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_d, \mathscr{T}^{(k)}_d]$, the number of rows of the resulting matrix of **Linearize(1)** is equal to $\sum_{d=2}^{D}\left(|\tilde{\mathscr{S}}^{(k)}_d| - r^{(k)}_d\right)$, and we *suppose* the following approximation:

$$\sum_{d=2}^{D}\left(|\tilde{\mathscr{S}}^{(k)}_d| - r^{(k)}_d\right) \approx \alpha. \tag{4.3}$$

This can be realized by avoiding choosing too many rows from $\mathscr{S}^{(k)}_D$, and, by doing so, the size of the resulting matrix of **Linearize(1)** is approximately $\alpha \times \alpha$.

4.2 Time complexity

In this subsection, we estimate the time complexity of PXL. Here, $C_{(d)1}$ (resp. $C_{(d)2}$, $C_{(d)3}$) denotes the estimation of the sum of the number of operations in $\mathbb{F}_q$ required for $(d)-1$ (resp. $(d)-2$, $(d)-3$) in the **Linearize(1)** step for all d with $2 \leq d \leq D$. Furthermore, C_{fix} (resp. C_{li2}) denote the estimation of the number of operations in $\mathbb{F}_q$ required for the **fix** (resp. **Linearize(2)**) step. These estimations are determined from the number n of all variables, the number k of guessed variables, the degree bound D (which can be taken to be $D_1^{(n-k)}$ given in (3.3)), and the size α of the resulting matrix of **Linearize(1)**. After obtaining each of these five estimations, we give a practical estimation of total time complexity by (4.8) below.

Time Complexity of (d)*-1.* Recall that the (d)-1 step performs Gaussian elimination on $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_d, \mathscr{T}^{(k)}_d]$, and its complexity is given as $\max\{|\tilde{\mathscr{S}}^{(k)}_d|, |\mathscr{T}^{(k)}_d|\}^\omega$ for each d with $2 \leq d \leq D$. Since we have $\sum_{d=2}^{D}\max\{|\tilde{\mathscr{S}}^{(k)}_d| - |\mathscr{T}^{(k)}_d|, 0\} \leq \alpha$ from (4.3), an upper bound on the sum of the complexity estimation of the (d)-1

step for all $2 \leq d \leq D$ is given by

$$\sum_{d=2}^{D} \max\{|\tilde{\mathscr{S}}_d^{(k)}|, |\mathscr{T}_d^{(k)}|\}^\omega \leq \left(\sum_{d=2}^{D} \max\{|\tilde{\mathscr{S}}_d^{(k)}|, |\mathscr{T}_d^{(k)}|\}\right)^\omega \leq \left(|\mathscr{T}_{\leq D}^{(k)}| + \alpha\right)^\omega \leq (2 \cdot |\mathscr{T}_{\leq D}^{(k)}|)^\omega = O\left(\binom{n-k+D}{D}^\omega\right),$$

where we used the equality

$$\sum_{d=2}^{D} \max\{|\tilde{\mathscr{S}}_d^{(k)}|, |\mathscr{T}_d^{(k)}|\} = \sum_{d=2}^{D} \max\{|\tilde{\mathscr{S}}_d^{(k)}| - |\mathscr{T}_d^{(k)}|, 0\} + |\mathscr{T}_{\leq D}^{(k)}|.$$

Therefore, we set $C_{(d)1}$ to be $\binom{n-k+D}{D}^\omega$.

Time Complexity of (d)*-2.* In each (d)-2 step, the complexity of executing the same row operations as those in (d)-1 step is estimated as that of multiplying a square matrix over $\mathbb{F}_q$ of size $|\tilde{\mathscr{S}}_d^{(k)}| \times |\tilde{\mathscr{S}}_d^{(k)}|$ to the polynomial matrix $\mathcal{PM}[\tilde{\mathscr{S}}_d^{(k)}, \mathscr{T}_{(d-2);(d-1)}^{(k)}]$ from the left. Note that $\mathcal{PM}[\tilde{\mathscr{S}}_d^{(k)}, \mathscr{T}_{(d-2);(d-1)}^{(k)}]$ is a sparse matrix, since $\mathcal{PM}[\mathscr{S}_d^{(k)}, \mathscr{T}_{\leq (d-1)}^{(k)}]$ does not change from the original structure at the beginning of the (d)-2 step by the same discussion as in Subsection 3.3, where each row of it has at most $n-k+1$ non-zero entries. Thus, multiplying the two matrices are done in $O((n-k) \cdot |\tilde{\mathscr{S}}_d^{(k)}|^2)$ additions and scalar multiplications in $\mathbb{F}_q[x_1, \ldots, x_k]$. Since polynomials appearing in each addition or scalar multiplication have degree ≤ 2, its cost is bounded by $O\left(\binom{k+2}{2}\right)$ with naive approach. Considering above together, each (d)-2 step has complexity $O\left(\binom{k+2}{2} \cdot (n-k) \cdot |\tilde{\mathscr{S}}_d^{(k)}|^2\right)$, and hence the total complexity of (d)-2 for all $2 \leq d \leq D$ is given by

$$\sum_{d=2}^{D} \left(\binom{k+2}{2} \cdot (n-k) \cdot |\tilde{\mathscr{S}}_d^{(k)}|^2\right) \leq \binom{k+2}{2} \cdot (n-k) \cdot \left(\sum_{d=2}^{D} |\tilde{\mathscr{S}}_d^{(k)}|\right)^2 = \binom{k+2}{2} \cdot (n-k) \cdot |\mathscr{T}_{\leq D}^{(k)}|^2 = O\left(k^2 \cdot (n-k) \cdot \binom{n-k+D}{D}^2\right),$$

and thus $C_{(d)2}$ is set to be $k^2 \cdot (n-k) \cdot \binom{n-k+D}{D}^2$.

Time Complexity of (d)*-3.* To estimate the time complexity of (d)-3 for all d with $2 \leq d \leq D$, we use the following lemma:

Lemma 3. *At the time of executing the* (d)*-3 step with* $2 \leq d \leq D-1$*, the degree of every element of* $\mathcal{PM}[\tilde{\mathscr{S}}_{(d+1);D}^{(k)}, \mathscr{T}_d^{(k)}]$ *is lower than or equal to* $D-d$*.*

Proof. By the induction, we prove that, at the time of starting the (d)-3 step, the degree of every element of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d+1);D}, \mathscr{T}^{(k)}_{d}]$ and $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d+1);D}, \mathscr{T}^{(k)}_{d-1}]$ is lower than or equal to $D-d$ and $D-d+1$, respectively. In the case of $d = D-1$, the above statement clearly holds. In the following, we show that, if the statement holds when $d = d'$ with $3 \leq d' \leq D-1$, then it also holds when $d = d'-1$. Before executing the step (d')-3, it is clear that $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d'+1);D}, \mathscr{T}^{(k)}_{d'-2}]$ is a zero matrix. Then, the (d')-3 step adds row vectors, which are obtained by multiplying rows corresponding to $\tilde{\mathscr{S}}^{(k)}_{d'}$ by a polynomial with the degree $D-d'$, to rows corresponding to $\tilde{\mathscr{S}}^{(k)}_{(d'+1);D}$. Here, the degree of each entry of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d'}, \mathscr{T}^{(k)}_{d'-1}]$ and $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d'}, \mathscr{T}^{(k)}_{d'-2}]$ are at most 1 and 2, respectively. Hence, through (d')-3, the degree of each entry of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d'+1);D}, \mathscr{T}^{(k)}_{d'-2}]$ becomes at most $D-d'+2$ and that of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d'+1);D}, \mathscr{T}^{(k)}_{d'-1}]$ remains at most $D-d'+1$, Therefore, the statement holds in the case where $d = d'-1$, as desired. □

Each (d)-3 step eliminates the corresponding columns using the leading coefficients of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d}, \mathscr{T}^{(k)}_{d}]$. More concretely, for each i with $1 \leq i \leq r_d$, we conduct row operations to eliminate the non-zero entries in the column to which the leading coefficient of the i-th row of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d}, \mathscr{T}^{(k)}_{d}]$ (in reduced row echelon form) belong, where r_d is the rank of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d}, \mathscr{T}^{(k)}_{d}]$. Such the non-zero entries to be eliminated are ones of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d+1);D}, \mathscr{T}^{(k)}_{d}]$, and we suppose from (4.3) that the number of them is at most α for each i. In each elimination process, we multiply the i-th row of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d}, \mathscr{T}^{(k)}_{(d-2);d}]$ by a non-zero polynomial in $\mathbb{F}_q[x_1, \ldots, x_k]$ of degree at most $D-d$ (this degree bound comes from Lemma 3), and then add the multiple to a row of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{(d+1);D}, \mathscr{T}^{(k)}_{(d-2);d}]$. Since each entry of $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d}, \mathscr{T}^{(k)}_{(d-2);d}]$ is a polynomial in $\mathbb{F}_q[x_1, \ldots, x_k]$ of degree ≤ 2 at this point, and since $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{d}, \mathscr{T}^{(k)}_{(d-2);d}]$ has $|\mathscr{T}^{(k)}_{(d-2);d}| = O(|\mathscr{T}^{(k)}_{d}|)$ columns, each elimination process is done in $O\left(\binom{k+2}{2} \cdot \binom{k+D-d}{D-d} \cdot |\mathscr{T}^{(k)}_{d}|\right)$ with a naive approach. The total number of these elimination processes is upper-bounded by $r_d \cdot \alpha$, we estimate the complexity of the (d)-3 step as

$$O\left(\binom{k+D-d}{D-d} \cdot \binom{k+2}{2} \cdot \alpha \cdot r_d \cdot |\mathscr{T}^{(k)}_{d}|\right) \leq O\left(\binom{k+D-d}{D-d} \cdot \binom{k+2}{2} \cdot \alpha \cdot \binom{n-k+d-1}{d}^2\right).$$

Note that the (D)-3 step can be omitted since $\mathcal{PM}[\tilde{\mathscr{S}}^{(k)}_{\leq(D-1)}, \mathscr{T}^{(k)}_{D}]$ is a zero matrix. Consequently, the sum of the complexities of the (d)-3 step for all d with $2 \leq d \leq D-1$ is estimated by

$$\begin{aligned}
&\sum_{d=2}^{D-1} \left(\binom{k+D-d}{D-d} \cdot \binom{k+2}{2} \cdot \alpha \cdot \binom{n-k+d-1}{d}^2\right) \\
&\leq \binom{k+2}{2} \cdot \alpha \cdot \left(\sum_{d=2}^{D-1} \binom{n-k+d-1}{d}\right) \cdot \left(\sum_{d=2}^{D-1} \binom{k+D-d}{k} \cdot \binom{n-k+d-1}{n-k-1}\right). \qquad (4.4)
\end{aligned}$$

Putting $d' = k + D - d$, one has

$$\sum_{d=2}^{D-1} \binom{k+D-d}{k} \cdot \binom{n-k+d-1}{n-k-1} = \sum_{d'=k+1}^{k+D-2} \binom{d'}{k} \binom{(n+D-1)-d'}{(n-1)-k}$$
$$\leq \sum_{d'=0}^{n+D-1} \binom{d'}{k} \binom{(n+D-1)-d'}{(n-1)-k} = \binom{(n+D-1)+1}{(n-1)+1} = \binom{n+D}{D}$$

from a formula similar to Vandermonde's identity. Therefore, the right hand side of (4.4) is upper-bounded by

$$O\left(k^2 \cdot \alpha \cdot \binom{n-k+D}{D} \cdot \binom{n+D}{D}\right),$$

and thus we set $C_{(d)3}$ to be $k^2 \cdot \alpha \cdot \binom{n-k+D}{D} \cdot \binom{n+D}{D}$.

*Time Complexity of **Fix**.* The size of the resulting matrix of **Linearize(1)** is approximately $\alpha \times \alpha$ due to the discussion in Subsection 4.1, and the degree of every element in the matrix is lower than or equal to D from Lemma 3. Therefore, the time complexity of **Fix** is estimated as that of substituting k values to $x_1, \ldots, x_k$ in α^2 polynomials with degree D in $\mathbb{F}_q[x_1, \ldots, x_k]$. When we use a naive approach, the complexity of evaluation of a polynomial with degree d in k variables is estimated by $\binom{k+d}{d}$. Therefore, C_{fix} is given by

$$C_{\mathsf{fix}} = q^k \cdot \alpha^2 \cdot \binom{k+D}{D}, \tag{4.5}$$

since the **Fix** step is iterated for any values of $x_1, \ldots, x_k$.

*Time Complexity of **Linearize(2)**.* The **Linearize(2)** step performs Gaussian elimination on an $\alpha \times \alpha$ matrix over $\mathbb{F}_q$, and thus we estimate C_{li2} by

$$C_{\mathsf{li2}} = q^k \cdot \alpha^\omega, \tag{4.6}$$

considering q^k times iterations.

Rough Estimations of Time Complexity Here, we present a more compact formula for the time complexity of PXL. Comparing the estimations $C_{(d)2}$ and $C_{(d)3}$, we can easily confirm that the value of $C_{(d)3}$ is larger than that of $C_{(d)2}$. Furthermore, comparing the estimations $C_{(d)1}$ and $C_{(d)3}$, we experimentally confirmed that, for the case where $10 \leq n \leq 100$, $m = n$, $1.5n$, $2n$, and k is the value minimizing the sum of the above five estimations, the value of $C_{(d)3}$ is always much larger than that of $C_{(d)1}$ (e.g., $C_{(d)1}$ and $C_{(d)3}$ in the case where $n = m = 100$ with $q = 2^8$ is approximately 2^{210} and 2^{259}, respectively). These facts indicate that the complexity of the **Linearize(1)** step is dominated by $C_{(d)3}$ for practical cases, and it is estimated as follows:

$$O\left(k^2 \cdot \alpha \cdot \binom{n-k+D}{D} \cdot \binom{n+D}{D}\right). \tag{4.7}$$

Table 1. The number of field operations approximated by power of 2 between PXL (4.8), h-XL (2.6), h-WXL (2.7), and Crossbred [22], the optimal number k of guessed variables of PXL, the value of $D = D_1^{(n-k)}$ estimated in (3.3), and the estimated size α of the resulting matrix of **Linearize(1)** on the MQ system with $n = m = 20$, 40, 60, and 80 over $\mathbb{F}_{2^8}$ (above) and over $\mathbb{F}_{31}$ (below).

	$n = m$	20		40		60		80	
	ω	2.37	2.81	2.37	2.81	2.37	2.81	2.37	2.81
$\mathbb{F}_{2^8}$	h-XL	2^{75}	2^{85}	2^{134}	2^{153}	2^{194}	2^{221}	2^{252}	2^{287}
	h-WXL	2^{75}	2^{75}	2^{129}	2^{129}	2^{182}	2^{182}	2^{234}	2^{234}
	Crossbred	2^{65}	2^{74}	2^{123}	2^{137}	2^{180}	2^{201}	2^{237}	2^{265}
	PXL	$\mathbf{2^{62}}$	$\mathbf{2^{64}}$	$\mathbf{2^{117}}$	$\mathbf{2^{121}}$	$\mathbf{2^{169}}$	$\mathbf{2^{178}}$	$\mathbf{2^{220}}$	$\mathbf{2^{233}}$
	k	3	3	6	5	8	7	10	8
	D	9	9	14	15	19	20	24	27
	α	2^{14}	2^{14}	2^{27}	2^{29}	2^{42}	2^{44}	2^{56}	2^{60}

	$n = m$	20		40		60		80	
	ω	2.37	2.81	2.37	2.81	2.37	2.81	2.37	2.81
$\mathbb{F}_{31}$	h-XL	2^{66}	2^{73}	2^{119}	2^{131}	2^{170}	2^{191}	2^{221}	2^{246}
	h-WXL	2^{65}	2^{65}	2^{116}	2^{116}	2^{162}	2^{162}	2^{208}	2^{208}
	Crossbred	2^{57}	2^{62}	2^{109}	2^{117}	2^{158}	2^{170}	2^{208}	2^{224}
	PXL	$\mathbf{2^{57}}$	$\mathbf{2^{57}}$	$\mathbf{2^{105}}$	$\mathbf{2^{107}}$	$\mathbf{2^{152}}$	$\mathbf{2^{158}}$	$\mathbf{2^{197}}$	$\mathbf{2^{208}}$
	k	5	5	8	8	11	10	13	12
	D	7	7	12	12	16	17	21	22
	α	2^{11}	2^{11}	2^{24}	2^{24}	2^{37}	2^{38}	2^{51}	2^{53}

By using this estimation on $C_{(d)3}$, the time complexity of PXL is roughly estimated by $C_{(d)3} + C_{\mathsf{fix}} + C_{\mathsf{li2}}$, say

$$O\left(k^2 \cdot \alpha \cdot \binom{n-k+D}{D} \cdot \binom{n+D}{D} + q^k \cdot \left(\alpha^2 \cdot \binom{k+D}{D} + \alpha^\omega\right)\right). \tag{4.8}$$

4.3 Comparison

We compare the complexity of our PXL with those of h-XL, h-WXL, and Crossbred with our motivation towards contribution of PXL to evaluating the security of MPKCs. Following the security estimation of [9], we choose h-WXL among the XL family as a target for comparison. We also adopt the complexity of h-XL on which h-WXL is originally based (in fact, h-XL is the most basic method in the framework of the hybrid approaches with XL) and that of Crossbred recognized as the theoretical most efficient algorithm for some parameter sets in [6]. Recall that the complexities of h-XL, h-WXL, Crossbred, and PXL are estimated by (2.6), (2.7), [22] and (4.8), respectively, where the estimation (4.8) for our PXL

is obtained by supposing practical Assumptions 2 and 3, and Heuristic 1. Note that, for fixed n, m, and q, each of the four approaches chooses the number k of guessed variables (and D and d for Crossbred) so that its complexity estimation becomes the smallest value, and thus the value of k depends on each approach. Furthermore, we here take the exponent of matrix multiplication ω as 2.37 [29] and 2.81 [50]. As we will see below, PXL is theoretically more efficient than other algorithms in the case of $n = m$ (this is the case where hybrid approaches for the MQ problem work most efficiently).

Table 1 compares the bit complexities of PXL, h-XL, h-WXL, and Crossbred on the MQ system of m equations in n variable with $n = m$ over $\mathbb{F}_{2^8}$ and $\mathbb{F}_{31}$. These orders of the finite fields are chosen following the MQ challenge [57], and in particular, $q = 2^8 = 256$ is also suggested as a parameter of [9]. Note also that we do not choose $q = 2$ since exhaustive searches are known to be effective in this case. Specifically, Table 1 shows the bit complexities of the four approaches, the optimal k of PXL minimizing the value of (4.8), the value of $D = D_1^{(n-k)}$ estimated in (3.3), and the estimated size α of the resulting matrix of **Linearize(1)** obtained from (4.2) for the case where $n = m$ with $n \in \{20, 40, 60, 80\}$. For example, when $q = 2^8$, $n = m = 80$, and $\omega = 2.37$, the complexities of h-XL, h-WXL, Crossbred, and PXL are approximately estimated as 2^{252}, 2^{234}, 2^{237}, and 2^{220}, respectively. As a result, we expect that PXL has the less complexity than those of other algorithms especially in the case of $\omega = 2.37$; we also expect that similar results will be obtained in other finite fields from the form of the complexity estimation (4.8).

On the other hand, we confirmed that PXL is not efficient in highly overdetermined cases. This is because, in such overdetermined cases, k is set to be a very small value for efficiency.

Remark 5 (Space Complexity). The memory space consumed by PXL is upper-bounded by $O\left(\binom{k+D}{D} \cdot \binom{n-k+D}{D}^2\right)$, since the degree of every element of the Macaulay matrix and its transformed matrices in **Linearize(1)** is at most D through an execution of PXL from Lemma 3. This estimation cannot be directly compared with other algorithms, since the values of the following two parameters depend on one's choice of an algorithm: The degree bound D (for the success of the algorithm) and the number k of fixed values.

On the other hand, focusing on the sparsity/density of matrices, we predict that PXL is not efficient compared with h-WXL in terms of the space complexity for the following reason: Through the elimination process of Macaulay matrices, WXL can deal with a Macaulay matrix as a sparse matrix due to Wiedemann's algorithm, whereas PXL maintains some dense submatrices. Considering this together with the time complexities for practical parameters, we conclude that the relationship between PXL and h-WXL would be a trade-off between time and memory.

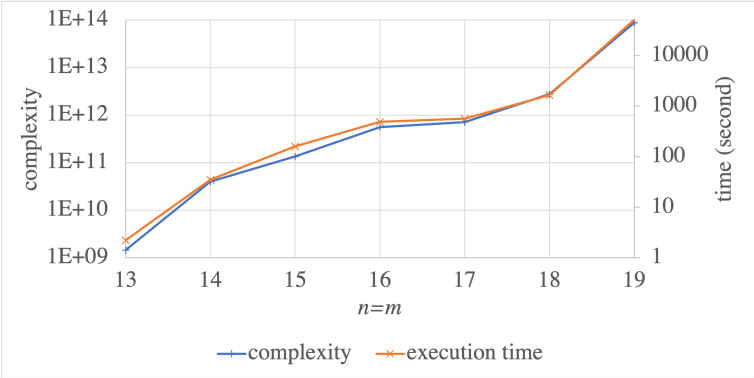

Fig. 1. Comparison between the estimation of complexity by (4.7) and the execution time of the **Linearize(1)** step on an MQ system with $n = m$ over $\mathbb{F}_{2^4}$.

5 Experimental Results

We implemented the proposed algorithm PXL in the Magma computer algebra system (V2.26-10) [10], in order to examine that it behaves as our complexity estimation provided in Section 4. (As it will be described below, note that our current implementation is not optimized one, see also Remark 6.) We also confirmed in our experiments that PXL outputs a solution correctly at $D = D_1^{(n-k)}$ as estimated in (3.3).

First, we confirmed that the **Linearize(1)** step behaves as in (4.7). The reason why we focus on the behavior of the **Linearize(1)** step is the following: In the estimation (4.8) of the total time complexity, only $C_{(d)3}$ is specific to our estimation in theory, while the later parts C_{fix} and C_{li2} for the **Fix** and **Linearize(2)** steps just come from known complexity estimations. Figure 1 compares the execution time of the **Linearize(1)** step and the bit complexity (4.7) on the system with $n = m$ from $n = 13$ to $n = 19$ over $\mathbb{F}_{2^4}$, and the number k of fixed variables is chosen so as to minimize the value of (4.8). As a result, Figure 1 shows that the execution time and our estimation (4.7) have almost the same behavior, which indicates that the estimation (4.7) would be reliable.

On the other hand, our current Magma implementation of the **Fix** and **Linearize(2)** steps does not show the similar behavior as our complexity estimation, due to the use of unoptimized implementation. For example, in the case of $n = m = 16$ with $k = 5$, **Linearize(1)**, **Fix**, and **Linearize(2)** took 10 min., 40 hr., and 30 min., respectively, whereas the estimated numbers of field operations of these three steps from (4.7), (4.5), and (4.6) are 2^{39}, 2^{44}, and 2^{39}, respectively. We observe that this inefficiency of the latter two steps (in particular **Fix** with a lot of for-loops) is due to the use of Magma's interpreter language. Using compiler languages such as C instead could be a solution to resolve this problem, but we must newly implement the arithmetic of matrices and polyno-

mials efficiently, which is not the topic of this paper. We leave such an efficient implementation with compiler languages to future work.

Remark 6. We remark that here we do not compare the execution time of our PXL with that of any other variant of XL, since the practical behavior deeply depends on how one implements the arithmetic of matrices (and polynomials) efficiently, which is not the topic of this paper. For a fair comparison, providing optimized implementations of several variants including PXL is required, and it is a very important task for practical cryptanalysis.

6 Conclusion

We presented a new variant of XL, which is a major approach for solving the MQ problem. Our proposed polynomial XL (PXL) eliminates the linearized monomials in polynomial rings to solve the system efficiently, and we estimated its complexities. Given an MQ system of m equations in n variables, the proposed algorithm first regards each polynomial in n variable as one in $n-k$ variables $x_{k+1}, \ldots, x_n$, whose coefficients belong to the polynomial ring $\mathbb{F}_q[x_1, \ldots, x_k]$. We then generate a Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$, and partly perform the row reduction (Gaussian elimination). Finally, random values are substituted for the k variables, and the remaining part of the (partly-reduced) Macaulay matrix is transformed into the reduced row echelon form. Partly reducing the (polynomial) Macaulay matrix is done mainly on submatrices over $\mathbb{F}_q$ (not over $\mathbb{F}_q[x_1, \ldots, x_k]$) with arithmetic of polynomials in $\mathbb{F}_q[x_1, \ldots, x_k]$ of bounded degree, and under some practical assumption and heuristic (Assumption 3 and Heuristic 1), the remaining part is expected to have size much smaller than the original one. This construction can reduce the amount of field operations for each guessed value, compared to h-XL. Supposing the above assumption and heuristic and additional but still practical one (Assumption 2), which assumes the affine semi-regularity of polynomial sequences, we gave an asymptotic estimation of the time complexity of PXL, which implies that PXL could solve the system faster in theory for the case of $n \approx m$ than h-XL, h-WXL, and Crossbred. On the other hand, PXL might be less efficient than h-WXL with respect to the space complexity.

This paper discusses only the quadratic case, but, as in the plain XL, the proposed algorithm can be also generalized to higher degree cases. Therefore, one considerable future work is to analyze the complexity of PXL on such higher degree systems. Furthermore, for a comparison of the practical time-efficiencies of our PXL and other XL variants, it is important to implement PXL (and the other variants) efficiently. In our experiments, we implemented PXL over Magma, but this can be more optimized by using an alternative (compiler) programming language, e.g., C. Note that there will be a drawback that the construction of our PXL over the polynomial ring prohibits the use of existing linear algebra libraries, which are often heavily optimized. Therefore, to provide such an optimized code for PXL will be a challenging task. Finally, we leave the analysis of the effect of PXL on the security of various multivariate signature schemes to a future work.

Acknowledgements

The authors thank the anonymous referees for helpful comments and suggestions. The authors also thank Tsuyoshi Takagi and Kazuhiro Yokoyama for helpful comments and suggestions. The authors are grateful to Kosuke Sakata for his advice on the implementation of our proposed algorithm.

This work was supported by JST CREST Grant Number JPMJCR2113, Japan, JSPS KAKENHI Grant Number JP22KJ0554, Japan, and JSPS Grant-in-Aid for Young Scientists 20K14301 and 23K12949, Japan.

References

1. M.-R. Albrecht, C. Cid, J.-C. Faugère, and L. Perret. On the relation between the MXL family of algorithms and Gröbner basis algorithms. *J. Symb. Comput.*, 47(8):926–941, 2012.
2. G. Ars, J.-C. Faugère, H. Imai, M. Kawazoe, and M. Sugita. Comparison between XL and Gröbner basis algorithms. In *ASIACRYPT 2004*, pages 338–353. Springer, 2004.
3. M. Bardet. *Étude des systèms algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* PhD thesis, Université Pierre et Marie Curie-Paris VI, 2004.
4. M. Bardet, J.-C. Faugére, and B. Salvy. On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations (extended abstract). In *ICPSS 2004*, pages 71–74, 2004.
5. M. Bardet, J.-C. Faugére, B. Salvy, and B.-Y. Yang. Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In *MEGA 2005*, 2005.
6. E. Bellini, R. H. Makarim, C. Sanna, and J. A. Verbel. An estimator for the hardness of the MQ problem. In *AFRICACRYPT 2022*, pages 323–347. Springer, 2022.
7. L. Bettale, J.-C. Faugère, and L. Perret. Hybrid approach for solving multivariate systems over finite fields. *J. Math. Cryptol.*, 3:177–197, 2009.
8. W. Beullens, F. Campos, S. Celi, B. Hess, and M. J. Kannwischer. MAYO specification. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/mayo-spec-web.pdf`, 2023.
9. W. Beullens, M.-S. Chen, J. Ding, B. Gong, M. J. Kannwischer, J. Patarin, B.-Y. Peng, D. Schmidt, C.-J. Shih, C. Tao, and B.-Y. Yang. UOV: Unbalanced oil and vinegar algorithm specifications and supporting documentation version 1.0. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/UOV-spec-web.pdf`.
10. W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symb. Comput.*, 24(3-4):235–265, 1997.
11. B. Buchberger. *Ein algorithmus zum auffinden der basiselemente des restklassenringes nach einem nulldimensionalen polynomideal.* PhD thesis, Universität Innsbruck, 1965.
12. J. A. Buchmann, J. Ding, M. S. E. Mohamed, and W. S. A. E. Mohamed. MutantXL: Solving multivariate polynomial equations for cryptanalysis. In *Dagstuhl seminar proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2009.
13. J. G. Capaverde. *Gröbner Bases: Degree Bounds and Generic Ideals.* PhD thesis, Clemson University, 2014.

14. N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *EUROCRYPT 2000*, pages 392–407. Springer, 2000.
15. D.-A. Cox, J. Little, and D. O'Shea. *Using algebraic geometry*. Springer, second edition edition, 2005.
16. D.-A. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms*. Springer, fourth edition edition, 2015.
17. C. Diem. The XL-algorithm and a conjecture from commutative algebra. In *ASIACRYPT 2004*, pages 323–337. Springer, 2004.
18. C. Diem. Bounded regularity. *Journal of Algebra*, 423:1143–1160, 2015.
19. J. Ding, A. Petzoldt, and D. S. Schmidt. *Multivariate public key cryptosystems (Second edition)*. Advances in Information Security, 80, Springer, 2020.
20. J. D. Duarte. On the complexity and admissible parameters of the Crossbred algorithm in $\mathbb{F}_{q\geq 2}$. Cryptology ePrint Archive, Paper 2023/1664, 2023.
21. T.-W. Dubé. The structure of polynomial ideals and Gröbner bases. *SIAM J. Comput.*, 19(4):750–773, 1990.
22. A. Esser, J. Verbel, F. Zweydinger, and E. Bellini. `CryptographicEstimators`: A software library for cryptographic hardness estimation. Cryptology ePrint Archive, Paper 2023/589, 2023.
23. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases (F4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999.
24. J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *ISSAC 2002*, pages 75–83. ACM, 2002.
25. J.-C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *J. Symb. Comput.*, 16(4):329–344, 1993.
26. R. Fröberg. An inequality for Hilbert series of graded algebras. *Math. Scand*, 56:117–144, 1985.
27. H. Furue, Y. Ikematsu, F. Hoshino, T. Takagi, K. Yasuda, T. Miyazawa, T. Saito, and A. Nagai. QR-UOV specification document. `https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/qruov-spec-web.pdf`, 2023.
28. G. Gaggero and E. Gorla. The complexity of solving a random polynomial system. arxiv:2309.03855, 2023.
29. F. L. Gall. Powers of tensors and fast matrix multiplication. In *ISSAC 2014*, pages 296–303. ACM, 2014.
30. M.-R. Garey and D.-S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman, 1979.
31. J. v. z. Gathen and V. Shoup. Computing Frobenius maps and factoring polynomials. *Comput. Complexity*, 2(3):87–224, 1992.
32. G.-M. Greuerl and G. Pfister. *A Singular Introduction to Commutative Algebra (2nd Edition)*. Springer, 2007.
33. Y. Ikematsu, S. Nakamura, and T. Takagi. Recent progress in the security evaluation of multivariate public-key cryptography. *IET Information Security*, 17(2):210–226, 2023.
34. Technology Innovation Institute. Multivariate quadratic estimator. `https://estimators.crypto.tii.ae/configuration?id=MQEstimator`.
35. A. Joux and V. Vitse. A Crossbred algorithm for solving boolean polynomial systems. In *NuTMiC 2017*, pages 3–21. Springer, 2017.
36. E. Kaltofen and V. Shoup. Subquadratic-time factoring of polynomials over finite fields. *Math. Comp.*, 67(223):1179–1197, 1998.

37. A. Kipnis, J. Patarin, and L. Goubin. Unbalanced oil and vinegar signature schemes. In *EUROCRYPT 1999*, pages 206–222. Springer, 1999.
38. A. Kipnis and A. Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In *CRYPTO 1999*, pages 19–30. Springer, 1999.
39. M. Kudo and K. Yokoyama. The solving degrees for computing Gröbner bases of affine semi-regular polynomial sequences. arXiv:2404.03530., 2024.
40. M. Kudo and K. Yokoyama. On Hilbert-Poincaré series of affine semi-regular polynomial sequences and related Gröbner bases. In *Mathematical Foundations for Post-Quantum Cryptography*, page 26 pages. Springer, 2024, to appear (arXiv:2401.07768).
41. D. Lazard. Systems of algebraic equations. In *EUROSAM 1979*, pages 88–94. Springer, 1979.
42. D. Lazard. Gröbner bases, gaussian elimination and resolution of systems of algebraic equations. In *Computer algebra (London, 1983), LNCS, 162*, pages 146–156. Springer, Berlin, 1983.
43. G. McGuire and D. O'Hara. On the termination of the general XL algorithm and ordinary multinomials. *J. Symb. Comput.*, 104:90–104, 2021.
44. M. S. E. Mohamed, W. S. A. E. Mohamed, J. Ding, and J. Buchmann. MXL2: Solving polynomial equations over GF(2) using an improved mutant strategy. In *PQCrypto 2008*, pages 203–215. Springer, 2008.
45. W. S. A. Mohamed. *Improvements for the XL algorithm with applications to algebraic cryptanalysis*. PhD thesis, TU Darmstadt, 2011.
46. S. Nakamura. Admissible parameter sets and complexity estimation of Crossbred algorithm. Cryptology ePrint Archive, Paper 2023/1687, 2023.
47. K. Pardue. Generic sequences of polynomials. *Journal of Algebra*, 324.4:579–590, 2010.
48. F. Salizzoni. An upper bound for the solving degree in terms of the degree of regularity. arXiv:2304.13485, 2023.
49. I. Semaev and A. Tenti. Probabilistic analysis on Macaulay matrices over finite fields and complexity constructing Gröbner bases. *Journal of Algebra*, 565:651–674, 2021.
50. V. Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13(4):354–356, 1969.
51. A. Tenti. *Sufficiently overdetermined random polynomial systems behave like semiregular ones*. PhD thesis, University of Bergen, 2019.
52. D.H̃. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theor.*, 32(1):54–62, 1986.
53. W.-T. Wu. Basic principles of mechanical theorem proving in elementary geometries. *J. Autom. Reason.*, 2(3):221–252, 1986.
54. B.-Y. Yang and J.-M. Chen. All in the XL family: Theory and practice. In *ICISC 2004*, pages 67–86. Springer, 2004.
55. B.-Y. Yang, J.-M. Chen, and N. Courtois. On asymptotic security estimates in XL and Gröbner bases-related algebraic cryptanalysis. In *ICICS 2004*, pages 401–413. Springer, 2004.
56. B.-Y. Yang, O.C.-H. Chen, D.J. Bernstein, and J.-M. Chen. Analysis of QUAD. In *FSE 2007*, pages 290–308. Springer, 2007.
57. T. Yasuda, X. Dahan, Y.-J. Huang, T. Takagi, and K. Sakurai. MQ challenge: Hardness evaluation of solving multivariate quadratic problems, 2015. NIST Workshop on Cybersecurity in a Post-Quantum World.
58. D. Y.Ỹ. Yun. On square-free decomposition algorithm. In *ISSAC 1976*, pages 26–35. ACM, 1976.

A Semi-regular sequences

We here review the notion of *semi-regular* sequence, which is introduced first by Bardet et al. (e.g., [3], [4], [5]). Semi-regular sequences are formulated also by Diem [18] in terms of commutative and homological algebra. See also [40, Section 2] for a survey.

We use the following notation: Let $R = K[x_1, \ldots, x_n]$ be the polynomial ring of n variables $x_1, \ldots, x_n$ over a field K. For a finitely generated graded R-module $M = \bigoplus_{d \in \mathbb{Z}} M_d$ (namely M_d is the degree-d homogeneous component), we denote by HF_M its Hilbert function, namely $\mathrm{HF}_M(d) = \dim_K M_d$ for each integer d, and denote by HS_M the Hilbert series of M, say $\mathrm{HS}_M(z) = \sum_{d=0}^{\infty} \mathrm{HF}_M(d) z^d \in \mathbb{Z}[\![z]\!]$. For a sequence $(f_1, \ldots, f_m)$ of *homogeneous* polynomials in R of positive degrees, let $K_\bullet(f_1, \ldots, f_m)$ denote the Koszul complex on the sequence (see e.g., [32, Section 7.6] for its definition), and let $H_i(K_\bullet(f_1, \ldots, f_m))$ be its i-th homology group. In particular, the first homology group is a finitely generated graded R-module given by

$$H_1(K_\bullet(f_1, \ldots, f_m)) = \mathrm{syz}(f_1, \ldots, f_m)/\mathrm{tsyz}(f_1, \ldots, f_m), \tag{A.1}$$

the sum of whose homogeneous components of degree less than or equal to d is denoted by $H_1(K_\bullet(f_1, \ldots, f_m))_{\leq d}$ for each $d \in \mathbb{Z}$. Here, $\mathrm{syz}(f_1, \ldots, f_m)$ denotes the module of syzigies on $(f_1, \ldots, f_m)$, say

$$\mathrm{syz}(f_1, \ldots, f_m) = \left\{ (h_1, \ldots, h_m) \in \bigoplus_{j=1}^{m} R(-d_j)\mathbf{e}_j \right\},$$

where each $R(-d_j)$ is the shifted graded ring given by $R(-d_j)_d = R_{d-d_j}$ for $d \in \mathbb{Z}$, and where each $\mathbf{e}_j$ denotes a standard basis element. On the other hand, $\mathrm{tsyz}(f_1, \ldots, f_m)$ is defined as an R-submodule of $\mathrm{syz}(f_1, \ldots, f_m)$ given by

$$\mathrm{tsyz}(f_1, \ldots, f_m) := \langle \mathbf{t}_{i,j} := f_i \mathbf{e}_j - f_j \mathbf{e}_i : 1 \leq i < j \leq m \rangle_R,$$

which is called the module of trivial syzigies on $(f_1, \ldots, f_m)$.

We first recall the definiton of d-regular sequences:

Definition 1 ([4, Definition 3], [18, Definition 1]). *Let $f_1, \ldots, f_m \in R$ be homogeneous polynomials of positive degrees $d_1, \ldots, d_m$ respectively, and put $I = \langle f_1, \ldots, f_m \rangle_R$. For each integer d with $d \geq \max\{d_i : 1 \leq i \leq m\}$, we say that a sequence $(f_1, \ldots, f_m)$ is d-regular if it satisfies the following condition:*

- *For each i with $1 \leq i \leq m$, if a homogeneous polynomial $g \in R$ satisfies $gf_i \in \langle f_1, \ldots, f_{i-1} \rangle_R$ and $\deg(gf_i) < d$, then we have $g \in \langle f_1, \ldots, f_{i-1} \rangle_R$.*

The (truncated) Hilbert series of d-regular sequences was determined by Diem [18], as in the following proposition:

Theorem 3 (cf. [18, Theorem 1]). *We use the same notation as in Definition 1. Then, the following are equivalent for each d with $d \geq \max\{d_i : 1 \leq i \leq m\}$:*

(1) The sequence $(f_1, \ldots, f_m)$ of homogeneous polynomials is d-regular.
(2) We have

$$\mathrm{HS}_{R/\langle f_1,\ldots,f_m\rangle}(z) \equiv \frac{\prod_{j=1}^{m}(1-z^{d_j})}{(1-z)^n} \pmod{z^d}. \tag{A.2}$$

(3) $H_1(K_\bullet(f_1, \ldots, f_m))_{\leq d-1} = 0$.

Recall that a finitely generated graded R-module M is said to be *Artinian* if there exists a sufficiently large $D \in \mathbb{Z}$ such that $M_d = 0$ for all $d \geq D$.

Definition 2 ([4, Definition 4], [5, Definition 4]). *For a homogeneous ideal I of R, we define its degree of regularity $d_{\mathrm{reg}}(I)$ as follows: If the finitely generated graded R-module R/I is Artinian, we set $d_{\mathrm{reg}}(I) := \min\{d : R_d = I_d\}$ with $I_d = I \cap R_d$, and otherwise we set $d_{\mathrm{reg}}(I) := \infty$. We also denote $d_{\mathrm{reg}}(I)$ by $d_{\mathrm{reg}}(F)$ for a subset or a sequence F of homogeneous elements in R generating the homogeneous ideal I.*

Definition 3 ([4, Definition 5], [5, Definition 5]; see also [18, §2]). *A sequence $(f_1, \ldots, f_m) \in R^m$ of homogeneous polynomials of positive degrees is said to be semi-regular if it is $d_{\mathrm{reg}}(I)$-regular, where we set $I = \langle f_1, \ldots, f_m\rangle_R$.*

The semi-regularity is characterized by equivalent conditions in the following proposition:

Proposition 1 ([18, Proposition 1 (d)]; see also [5, Proposition 6]). *With the same notation as in Definition 1, we put $D = d_{\mathrm{reg}}(I)$. Then, the following are equivalent:*

(1) The sequence $(f_1, \ldots, f_m)$ of homogeneous polynomials is semi-regular.
(2) We have

$$\mathrm{HS}_{R/I}(z) = \left[\frac{\prod_{j=1}^{m}(1-z^{d_j})}{(1-z)^n}\right], \tag{A.3}$$

where $[\cdot]$ means truncating a formal power series over $\mathbb{Z}$ after the last consecutive positive coefficient.
(3) $H_1(K_\bullet(f_1, \ldots, f_m))_{\leq D-1} = 0$.

Note that, by Definition 3, if $(f_1, \ldots, f_m)$ is semi-regular, then the degree of regularity $d_{\mathrm{reg}}(I)$ coincides with $\deg(\mathrm{HS}_{R/I})+1$, where we set $I = \langle f_1, \ldots, f_m\rangle_R$.

Finally, we recall the definition of an affine semi-regular sequence:

Definition 4 ([5, Definition 5]). *A sequence $F = (f_1, \ldots, f_m) \in R^m$ of not necessarily homogeneous polynomials of positive degrees is said to be semi-regular if the sequence $F^{\mathrm{top}} = (f_1^{\mathrm{top}}, \ldots, f_m^{\mathrm{top}})$ is semi-regular . In this case, the sequence F is said to be affine semi-regular.*