On the Complexity of the Crossbred Algorithm

João Diogo Duarte
joaodduarte@protonmail.com

Abstract

The Joux–Vitse Crossbred algorithm's aim is to efficiently solve a system of semi-regular multivariate polynomials. For random polynomials, this is thought to be at least hard for a classical computer to solve [9, p.194]. In this work, the algorithm is described in detail, a bivariate generating series to test the admissibility of parameters in $\mathbb{F}_2$ is investigated and from this, a new series for $\mathbb{F}_q$, $q > 2$ is presented. In addition, a complexity estimate is given for both $\mathbb{F}_2$ and $\mathbb{F}_q$, $q > 2$, which is compared to an estimate presented by Samardijska et al. [18]. Their estimate is shown to be an upper bound for the Crossbred algorithm. By obtaining optimal parameters using the previous results, the cost of theoretically applying Crossbred, FES and Hybrid-F_5 to polynomial systems of various sizes of various numbers of variables and q was plotted. Overall, it was determined that for larger fields, the Crossbred algorithm provides an improved asymptotic complexity over FES. However, it does not provide any improvement over Hybrid-F_5.

Keywords— Post-Quantum Cryptography, Multivariate Polynomial Systems, Gröbner Basis, Crossbred Algorithm, Complexity

1 Introduction

Currently, public key cryptosystems such as RSA rely on the difficulty of factoring a large number into two prime factors. However, as technology and algorithms become more sophisticated, key sizes must increase to harden security. Furthermore, with the development of Shor's Algorithm for quantum computers, the integer factorisation and discrete logarithm problems will be solvable in polynomial time, rather than exponential time with classical algorithms. Henceforth, when quantum computers become a feasible tool for computation, cryptosystems that rely on these aforementioned problems will be considered insecure.

Research into post-quantum cryptosystems has developed ideas which include lattice-based cryptography, hash-based cryptography and multivariate cryptography, all of which, at the time of writing, are considered secure against an attack by a quantum computer [19]. The multivariate approach uses simple mathematics, comparatively to the other post-quantum systems. Due to its simplicity, it can be used on low power devices such as smart cards. Furthermore, encryption and decryption are efficient and these systems often also have fast signature generation and verification [21].

Generally it is at least hard for classical computers to solve a set of random multivariate polynomials [9, p.194]. In addition, there are no known polynomial time algorithms that solve such a problem for quantum computers. There exist many classical algorithms that attempt to solve multivariate polynomial equations, such as F_5 and XL. However, since these algorithms have an exponential complexity, it is believed that there may exist more efficient algorithms for the task. This is why it is important to thoroughly study and understand new algorithms, such as the Joux–Vitse algorithm, as a better performance may the difference between breaking a cryptographic algorithm or not.

1.1 Introduction to the Joux–Vitse Crossbred Algorithm

The Crossbred algorithm was created by Joux and Vitse [15] in 2017. Their purpose was to produce a scalable algorithm that solves random systems of multivariate polynomial equations in $\mathbb{F}_2$ by combining ideas from the BooleanSolve algorithm [3] and the Kipnis-Patarin-Goubin algorithm [16], hence it being called 'Crossbred'. Specifically, given a multivariate system, it will produce an equivalent smaller system which is more easily solved. This algorithm was able to solve all Fukuoka Type I MQ challenges up to its last system of 148 quadratic equations in 74 variables, which beats the previous record which was held by an algorithm called Fast Exhaustive Search (FES).

However, some issues were found with the Joux–Vitse Crossbred algorithm, namely that, just like many of its contemporaries, despite its practical speed, upon performing some asymptotic analysis, Joux and Vitse determined

that it ultimately does not provide an improved asymptotic bound in relation to BooleanSolve and FXL. Due to this disappointing result and restrictive page limits, they did not fully expand on this analysis. The extent of the asymptotic analysis in the original paper was limited to presenting a bivariate generating series which determines if the parameters provided to the algorithm are admissible. However, due to the same issues, the series itself and its derivation are not explained. Hence, there is room to explain its derivation and why it works. Lastly, Joux and Vitse only worked with systems in $\mathbb{F}_2$, and stated that this algorithm should also work for small finite fields $\mathbb{F}_q$, where $q > 2$. Thus, there is room to investigate their algorithm's complexity for larger fields and investigate if it provides any asymptotic improvement for various ratios of m (the number of equations) and n (the number of variables) in relation to FES and Hybrid-F_5.

Furthermore, the method of testing the admissibility series was only defined for $\mathbb{F}_2$. Hence, there is space to analyse this series and fully explain what it means and why it works.

However, in the supporting documentation of a NIST Post-Quantum Cryptography Round 2 candidate, MQDSS, Samardijska et al. [18] provided a rough complexity estimate for the Crossbred algorithm for $\mathbb{F}_q$, $q > 2$, but they did not provide a clear explanation on how the estimate was obtained. This allows us to see if their complexity estimate matches with the one derived in this work and compare any differences.

2 Mathematical Recap

2.1 Regular and Semi-Regular Systems

Let $\mathbb{F}_q$ be a finite field of size q such that $\mathbb{F}_q[X = x_1 \dots x_n]$ is a polynomial ring over n variables.

Definition 1. (Regular Systems) A regular system is a sequence of homogeneous polynomials, $f_1, \dots, f_m \in \mathbb{F}_q[X]$, such that if $i \in \{1, \dots, m\}$ and $g \in \mathbb{F}_q[X]$:

$$gf_i \in \langle f_1, \dots, f_{i-1} \rangle \tag{1}$$

then, g is also in the ideal generated by $f_1, \dots, f_{i-1}$.

Regular systems, are at most, determined, meaning that they have, at most, $n = m$, whereby n is the number of variables in the system of polynomials and m is the number of equations [1].

Definition 2. (Semi-Regular Systems) A semi-regular system is a homogenous overdetermined system ($m > n$) in $\mathbb{F}_q[X]$ such that if $i \in \{1, \dots, m\}$ and $g \in \mathbb{F}_q[X]$:

$$gf_i \in \langle f_1, \dots, f_{i-1} \rangle \text{ and } deg(gf_i) < d_{reg} \tag{2}$$

then, g is also in the ideal generated by $f_1, \dots, f_{i-1}$ [1].

The term d_{reg} is the degree of regularity and will be discussed later.

2.2 Field Equations

In the original paper detailing the Crossbred Joux–Vitse algorithm [15], Joux and Vitse work with systems in $\mathbb{F}_2$, so they include field equations of the form $x_i^2 + x_i$. For $\mathbb{F}_2$, this does not affect computational cost and hence we remove squares as $x_i^2 = x_i$.

This means that every equation of the form f^2 will result in f (called the Frobenius Criterion *i.e.*. $f^2 = f$), which means that if $g_i = f$, then $g_i f = f$ [1]. Hence, one may initially assume that no system in $\mathbb{F}_2$ is semi-regular. Thus, let us only consider polynomials that are homogenous and we keep only the highest degree part of our field equations. This means that we are working over the quotient field $\mathbb{F}_2/\langle x_1^2, \dots, x_n^2 \rangle$, which will result in $f^2 = 0$. Hence, if $g_i = f$, then $g_i f = 0$. This means that in $\mathbb{F}_2$, we must work with homogenous polynomials so that we can obtain semi-regular systems.

2.3 Degree of Regularity

Given that q is a power of a prime, let I be an ideal in $\mathbb{F}_q[X = x_1 \dots x_n]$ generated by a homogeneous finite system of polynomials with n variables and m equations. Denote the set of all possible polynomials in a polynomial ring $\mathbb{F}_q[X]$ with a degree less or equal to an integer s as $\mathbb{F}_q[X]_{\leq s}$. We also denote an ideal of this polynomial ring, I, with a degree less or equal to an integer s as $I_{\leq s}$.

We now define the Hilbert Function of an ideal I as

Definition 3. (Hilbert Function)

$$HF_I(s) = \dim(\mathbb{F}_q[X]_{\leq s}) - \dim(I_{\leq s}) \tag{3}$$

[7, p.487]

This shows that the Hilbert Function is the dimension of polynomials in the polynomial ring that are not in the ideal. This means that when the dimension of the set of polynomials with degree less or equal to s in ideal is equal to the dimension of the set of polynomials with degree less or equal to s in the polynomial ring, the Hilbert Function will output 0. The smallest s such that the Hilbert Function results in a non-negative number is defined as the index of regularity and if the ideal is homogeneous and zero-dimensional (i.e has a finite number of solutions), it will be equal to the degree of regularity.

Let us now define the Hilbert Series for an ideal I with n variables as:

Definition 4. (Hilbert Series)

$$HS_I(t) = \sum_{i=0}^{\infty} HF_I(i)t^i \tag{4}$$

whereby HS_I is a formal power series in variable t.

As we can see, this power series is simply a Taylor Series of the Hilbert Function. The Hilbert Series of a semi-regular system with m equations and n variables is [2]:

$$S_{m,n} = \frac{\prod_{i=1}^{m}(1-t^{d_i})}{(1-t)^n} \tag{5}$$

For the rest of the paper, we define the degree of regularity of a non-boolean (i.e in $\mathbb{F}_q$, $q > 2$) semi-regular system as the power of the first non-negative coefficient of the series in Equation 5. According to [14], despite experimental evidence that such sequences are common, the problem lies on assessing the existence of such sequences.

Bardet [2, p.68] stated that the Hilbert Series of a semi-regular system in $\mathbb{F}_2$ is:

$$S_{m,n} = \frac{(1+t)^n}{\prod_{i=1}^{m}(1+t^{d_i})} \tag{6}$$

For the rest of the paper, we define the degree of regularity of a boolean semi-regular system (i.e in $\mathbb{F}_2$), as the power of the first non-negative coefficient of the series in Equation 6.

2.4 Macaulay Matrix

Consider an ideal $I = \langle f_1, \ldots, f_m \rangle \in \mathbb{F}_q[X]$. Let $F \in \mathbb{F}_q[X]$ be a polynomial system whereby the degree of each $f_i \in F$ is denoted as d_i.

Definition 5 (Macaulay Matrix of degree D)**.** We define a Macaulay Matrix, Mac_D, of degree D as a coefficient matrix of the set $\{u \cdot f_i \,|\, \deg(u) \leq D - d_i\}$ whereby $i \in \{1, \ldots, m\}$ and u is a monomial in $\mathbb{F}_q[X]$ and $f_i \in F$.

Another way of putting it is that a Macaulay Matrix of degree D is a coefficient matrix whose columns lists all monomials with degrees equal or less than D from largest till smallest, as per some fixed ordering. We then multiply each f_i by all monomials of degree equal or less than $D - d_i$, whereby d_i is the degree of f_i. Each row of the Macaulay Matrix is indexed by the result of these multiplications.

Lazard [17] proved that there exists a positive integer D whereby the rows of a row reduced Macaulay Matrix of degree D are a Gröbner Basis for the ideal $\langle f_1, \ldots, f_m \rangle$. Hence, we can use Macaulay Matrices to aid us to solve polynomial systems.

2.4.1 Relationship with the Degree of Regularity

According to Bardet [2, p.65-66], we can make a 1-to-1 correspondence between the Hilbert Function and a Macaulay Matrix for homogeneous semi-regular systems. This is done by setting $\dim(\mathbb{F}_q[X]_{\leq s})$ as the number of columns of the Macaulay Matrix and $\dim(I_{\leq s})$ as the number of linearly independent rows (rank) of the matrix. Hence, a definition for the degree of regularity of a polynomial system with n variables and m equations is the smallest degree whereby its Macaulay Matrix of this degree has as many (or more) linearly independent rows as columns.

Hence, it is possible to obtain linear equations (or low degree equations) from computing this Macaulay Matrix's row echelon form. Lastly, it was firstly noted by Lazard [17] that Gaussian Elimination of a degree $D = d_{reg}$ Macaulay Matrix is equivalent to performing the Buchberger Algorithm.

Lastly, there exists a very close relationship between the corank of a Macaulay Matrix and the Hilbert Function for homogeneous semi-regular systems. If we recall, we define the Hilbert Function as:

$$HF_I(s) = \dim(\mathbb{F}_q[X]_{\leq s}) - \dim(I_{\leq s})$$

which is the number of columns of the Macaulay Matrix minus its linearly independent rows (its rank). This is, by definition, the corank of a Macaulay Matrix.

3 State of the Art for Solving Multivariate Polynomials

3.1 Exhaustive Search

This is the most basic way of solving a system of polynomials. If q is a power of a prime and n is the number of variables, we iterate over $\mathbb{F}_q^n$ and test if they produce a valid solution for our polynomials. Since there are q^n values to test, the complexity of this algorithm is $\mathcal{O}(mq^n)$ for m polynomials.

In $\mathbb{F}_2[X]$, Fast Exhaustive Search (FES) was proposed by Boulliaguet et al. [6] and it efficiently enumerates over the search space of 2^n such that the complexity of exhaustive search is (hopefully) less than $\mathcal{O}(mq^n)$. In $\mathbb{F}_2$, the complexity of FES is $\mathcal{O}(\log_2(n) \cdot 2^n)$ and is independent of the number of polynomials, m [6]. We can expand FES for larger fields using $q-$ary Gray codes codes with a complexity of $\log_q(n)q^n$.

3.2 Under and Overdetermined Systems

We say that an underdetermined system has n variables in m polynomials whereby $n > m$ [20]. Kipnis et al. [16] managed to formulate an algorithm that solves underdetermined systems whereby $n \geq m(m+1)$ in polynomial time. Another more naïve way of solving these equations is merely a case of specialising $n - m$ variables to obtain $n = m$. Manipulating and choosing these specialisations wisely are the basis of the algorithm conceived by Thomae and Wolf [20]. This algorithm generalises the one presented by Kipnis et al. [16] to allow for other kinds of underdetermined systems to be solved in complexities ranging from polynomial to exponential.

An overdetermined system is one whereby $m > n$. According to Thomae and Wolf [20], the complexity of solving overdetermined systems via Gröbner Basis algorithms decreases as m approaches $n(n-1)/2$. When $m \geq n(n-1)$, these systems can be broken in polynomial time [16].

3.3 Hybrid-F_5

F_4 was created by Faugère [10] and it outputs a Gröbner basis for a given set of polynomials. An improved version called F_5 was later developed by Faugère [12]. The details of these algorithms will not be discussed in this paper.

Hybrid-F_5 [4] combines exhaustive search and F_5 by specialising k variables and running the F_5 algorithm over the remaining $n - k$ variables. The most costly part in the complexity of F_5 is the row reduction of a matrix of size $\binom{n+d_{reg}}{d_{reg}}$ for $q > 2$ and n, d_reg hence once can simplify its complexity estimate to:

$$C_{F_5} = \mathcal{O}\left(\binom{n + d_{reg}}{d_{reg}}^{\omega}\right) \tag{7}$$

whereby $2 \leq \omega \leq 3$ is the exponent of matrix multiplication. We set $\omega = 2$. By combining the above complexity estimate and exhaustive search, we obtain the complexity estimate for Hybrid-F_5 [5]:

$$C_{Hybrid} = q^k \cdot \mathcal{O}\left(\binom{n - k + d_{reg}(n-k)}{d_{reg}(n-k)}^2\right) \tag{8}$$

whereby $d_{reg}(n-k)$ is the degree of regularity of the system after evaluating k variables and hence, having $n - k$ variables left.

3.4 FXL/BooleanSolve

The BooleanSolve algorithm was developed by Bardet et al. [3] before the Crossbred algorithm. This algorithm specialises the last $n - k$ variables in the polynomials by iterating it through $\mathbb{F}_2^{n-k}$. It then tests the consistency of the system via Macaulay Matrices of $d_{reg}(k)$, whereby $d_{reg}(k)$ is the degree of regularity of the system after specialisation. If this system is not consistent, iterate to the next value. Otherwise, exhaustive search is conducted over the first k variables [11]. The complexity of this algorithm is $\mathcal{O}(2^{0.841n})$ and a probabilistic variant called the Las Vegas variant has a conditional complexity of only $\mathcal{O}(2^{0.792n})$.

4 Joux–Vitse Crossbred Algorithm

Essentially, this algorithm involves the specialisation of variables and then solving the remaining ones via linearisation (a special case of F_5 where we treat each term as an individual variable). The advantage of this technique is that we can avoid solving the initial or the specialised system via, for example, the general version of F_5.

The main differentiating factor between this algorithm and BooleanSolve/FXL is that the manipulation of the Macaulay Matrix is done before specialising any variables. This is an advantage since linear algebra in a Macaulay Matrix is the most costly step of the BooleanSolve algorithm since it is performed 2^{n-k} times [15]. The Crossbred algorithm attempts to limit the size of the Macaulay Matrix to speed up the computation of a polynomial system's Gröbner Basis. Note that this algorithm assumes semi-regularity.

The algorithm accepts four arguments:

1. F, a system of m equations over n variables in $\mathbb{F}_q$.
2. D, the degree of the Macaulay Matrix of F. $D \geq 2$ and must be at least the degree of regularity of a system with k variables over m equations in $\mathbb{F}_q$.
3. d, the desired degree of the system after we specialise the last $n-k$ variables. $1 \leq d < D$.
4. k, the number of variables we want our specialised system to have. $1 \leq k \leq n$.

The total degree in the first k variables of a polynomial p is labelled as $deg_k p$.

If we wish for our system to reduce down to a linear system, then we set $d = 1$. However, to extend this to larger values of d, we must construct new equations with $deg_k p \leq d$, which is equivalent of saying that the total degree in the first k variables of our new equations must be at most d. According to Joux and Vitse [15], this allows us to select smaller values of D since we do not need to produce a Macaulay Matrix with a lot of polynomials in order to 'break them down' to a system of degree d. This is desirable since D must be large enough for any reduced equations to exist but also small enough to make the Macaulay Matrix manageable.

The main difficulty of this algorithm is selecting D, d and k.

4.1 Description of the Algorithm

To better understand the algorithm, let us divide it into two main steps, the pre-processing and then the actual algorithm. The pre-processing goes as following:

1. Construct the Macaulay Matrix of degree D of polynomial system F with its columns sorted in reverse graded lex.
2. Let $Mac^k_{D,d}(F)$ be a submatrix of Mac_D whereby each row $u_{ij} f_i$ represents a polynomial with the property that $deg_k u_{ij} \geq d - 1$.
3. Let $M^k_{D,d}(F)$ be a submatrix of $Mac^k_{D,d}$, whereby each column i represents the monomial M_i whereby $deg_k M_i > d$.

The actual algorithm does the following:

1. Construct the left kernel of $M^k_{D,d}$ and multiply this left kernel by $Mac^k_{D,d}(F)$. This forms a system of polynomials, P, whereby they have a total degree at most D and at most d in $x_1, \ldots, x_k$.
2. For all $a = \{a_{k+1}, \ldots, a_n\} \in \mathbb{F}_2^{n-k}$:
   (a) Partially evaluate the last $n-k$ variables of F at a. Let F^* represent this new system.
   (b) Construct a new Macaulay Matrix of degree d of F^*. Now, let $Mac_d(F^*)$ represent this new matrix.
   (c) Partially evaluate the last $n-k$ variables of polynomial system P at a. Let P^* represent this new system as a coefficient matrix.
   (d) Append $Mac_d(F^*)$ to P^*. Let PM^* represent this new system of polynomials.
   (e) Check if this system is consistent using dense linear algebra. If it is, extract variables $x_1 \ldots x_k$ and test the solution

By constructing $Mac^k_{D,d}(F)$, we obtain polynomials of the form $u_{i.j} \cdot f_i$ whereby the total degree of $u_{i.j}$ in the first k variables is at least $d-1$. For example, for $\mathbb{F}_2$, consider $n = 5$, $D = 4$, $d = 2$ and $k = 3$. That means that the first k variables are x_1, x_2, x_3 and the last $n-k$ are x_4 and x_5. Let $f = x_1 x_2 + 1$. Consider the following row in Mac_D:

$$x_4 x_5 \cdot f = x_1 x_2 x_4 x_5 + x_4 x_5$$

Clearly, since $Mac^k_{D,d}$ only contains rows whereby the multiplier has at least total degree $d-1=1$ in the first k variables, we would not include this row. This is because upon specialisation, such as $x_4 = x_3 = 1$, we would obtain:

$$1 \cdot f = x_1x_2 + 1$$

which is simply f. If we set $x_4 = x_3 = 0$ or any variation whereby at least one of the variables is 0, we would obtain:

$$0 \cdot f = 0$$

Hence, if we include these rows, we would simply obtain 0 or our initial f. None of these add any new information to our system because if we obtain f, it will result in a linearly dependent row and thus, produce the trivial solution $0 = 0$. If we obtain 0, it produces a row of zeroes, which also leads to the trivial solution $0 = 0$.

Furthermore, since multiplying a matrix by its kernel is equal to 0, multiplying the kernel of $M^k_{D,d}(F)$ by $Mac^k_{D,d}(F)$, we simply 'remove' all monomials in $Mac^k_{D,d}(F)$ that do not have a degree at most d in the first k variables as we remove columns whereby their total degree in the first k variables is greater than d. This will allow us to achieve a system of degree at most d after specialising the last $n-k$ variables.

All of this boils down to the fact that we want to obtain an equivalent system of F that has a degree at most D and a degree of at most d in the first k variables. As mentioned before, we want this because after we specialise the last $n-k$ variables, we obtain a smaller system with a total degree of at most d.

5 Finding Parameters

For $d = 1$, $Mac^k_{D,d}(F) = Mac_D(F)$, hence, we have selected all of the equations to be used when constructing P. The reason we create $Mac_d(F^*)$ is to include more equations since it reduces the amount of consistent systems that are obtained and therefore, you have less systems to test whether they are also consistent with F. This may seem like a disadvantage but the consistent systems we have avoided would also not be consistent with F and hence, we evaluate less systems that are bound to be incorrect Clearly, when $d = 1$, we are not producing any new information since all its rows would be linearly dependent with P. Henceforth, we consider $Mac_d(F^*)$ to be empty when $d = 1$.

According to Joux and Vitse [15], the number of equations of P must be at least the number of monomials in k variables of degree d, which is $\binom{k+d-1}{d}$ for $\mathbb{F}_q$ and $\sum_{d'=0}^{d}\binom{k}{d'}$ for $\mathbb{F}_2$. Thus, for $d = 1$ we need to simply check whether $|P| > k$. This is to ensure enough independent relations to finally solve the system via linearisation.

Note how we are using field equations for polynomial systems in $\mathbb{F}_2$. As mentioned before, they do not affect computational costs and remove squares. This is equivalent of working within a quotient ring of the form $\mathbb{F}_2/\langle x_1^2, \ldots, x_n^2\rangle$ [1]. However, multiplication in quotient rings involves reducing the polynomials by the field equations to ensure that they are in the quotient ring. This means that including field equations for large fields may not be computationally feasible.

5.1 Admissibility of Parameters

For $d > 1$, to determine the admissibility of the parameters, Joux and Vitse [15] derived a bivariate generating function. Firstly, let us define:

$$S^k_{D,d} = \frac{(1+X)^{n-k}}{(1-X)(1-Y)}\left(\frac{(1+XY)^k}{(1+X^2Y^2)^m} - \frac{(1+X)^k}{(1+X^2)^m}\right) \tag{9}$$

The coefficient of X^DY^d of $S^k_{D,d}$ represents the number of new independent polynomials after the reduction of $Mac^k_{D,d}$, which is equivalent to corank of $M^k_{D,d}$. The reason for this will be explained in the next section.

However, to test admissibility, if the coefficient of X^DY^d is non-negative in the following subtraction, then the parameters (D, d, k) for the algorithm are admissible:

$$A^k_{D,d} = S^k_{D,d} - \frac{(1+Y)^k}{(1-X)(1-Y)(1+Y^2)^m} \tag{10}$$

The reason why will be explained further down this section.

5.2 Analysing the Generating Function

Let us now break down the various parts of the bivariate generating function. By expanding out the multiplication that occurs in $S^k_{D,d}$ and ignoring the $\frac{1}{1-X}$ and $\frac{1}{1-Y}$ parts as they will be explained later, we obtain:

$$\frac{(1+XY)^k(1+X)^{n-k}}{(1+X^2Y^2)^m} - \frac{(1+X)^n}{(1+X^2)^m} \tag{11}$$

The leftmost term represents the formal power series of the corank of $M^k_{D,d}$ and the rightmost term represents the formal power series of the corank of Mac_D. This allows for us to obtain the corank of these matrices for various values of parameters.

Proof. Let us prove that the rightmost term represents the formal power series of the corank of Mac_D.

As stated before, Bardet proved that there is a 1-to-1 correspondence between the corank of a Macaulay Matrix and the Hilbert Function for homogeneous semi-regular systems with n variables. Let us label one of these systems as F. Let F have an arbitrary degree. Specifically, for a Macaulay Matrix with degree D, we can write:

$$corank(Mac_D(F)) = HF_I(D) = \dim(\mathbb{F}_q[X]_{\leq D}) - \dim(I_{\leq D}) \tag{12}$$

Hence, the formal power series in variable t can be expressed as:

$$HS_I(t) = \sum_{i=0}^{\infty} HF_I(i)t^i = \sum_{i=0}^{\infty} corank(Mac_i(F))t^i \tag{13}$$

Therefore, the degree of regularity can be interpreted as the first non-negative coefficient of the above series, which is when the Macaulay Matrix has as many (or more) linearly independent rows as columns.

For $\mathbb{F}_2$, as mentioned before, Bardet proved that the above equation can be written as

$$HS_I(t) = \frac{(1+X)^n}{(1+X^2)^m}$$

This means that the degree of regularity is the first non-negative coefficient of the above series. Hence, given that F only contains quadratic polynomials:

$$HS_I(t) = \frac{(1+X)^n}{(1+X^2)^m} = \sum_{i=0}^{\infty} corank(Mac_i(F))t^i \tag{14}$$

Hence, the rightmost term of of Equation 11 represents the formal power series of the corank of $Mac_D(F)$. □

Proof. Let us prove that the leftmost term represents the formal power series of the corank of $M^k_{D,d}$. Let X measure the total degree of the polynomials involved in the Gröbner computation and Y focus on the first k variables, such that after guessing the first $n-k$ variables, the formal power series of the corank of the resulting system's Macaulay Matrix would be:

$$HS_I(t) = \frac{(1+Y)^k}{(1+Y^2)^m} \tag{15}$$

This clearly holds as Y focuses on the first k variables, which are the only ones remaining after specialising the last $n-k$ variables. Hence, we can omit X.

Since we are multiplying $Mac^k_{D,d}$ by the left kernel of $M^k_{D,d}$ to obtain P, P will have the same number of rows of the left kernel due to the rules of matrix multiplication. Hence, if we find the degree of regularity of P, we will obtain the corank of $M^k_{D,d}$. If we use the aforementioned X and Y, we can represent the Hilbert Series of P as:

$$HS_I(t) = \frac{(1+XY)^k(1+X)^{n-k}}{(1+X^2Y^2)^m} \tag{16}$$

and since this will also give us the formal power series of the corank of $M^k_{D,d}$, the following equality is satisfied:

$$HS_I(t,r) = \frac{(1+XY)^k(1+X)^{n-k}}{(1+X^2Y^2)^m} = \sum_{i=0}^{\infty}\sum_{j=0}^{D-1} corank(M^k_{i,j}(F))t^i r^j \tag{17}$$

This holds due to Equation 14. Hence, the leftmost term of Equation 11 is represents the formal power series of the corank of $M^k_{D,d}$. This proof holds for any $\mathbb{F}_q$, given that the appropriate Hilbert Series is used. □

As stated before, finding the dimension of the left kernel of $M^k_{D,d}$ will tell us the number of polynomials produced in P. Furthermore, we need to know how many of these polynomials are new in relation to our initial system, F, as we are going to be including our initial equations alongside P since $d > 1$. This is why we subtract the Hilbert Series of our initial Macaulay Matrix of degree D, which is the rightmost term of Equation 11.

Consider the subtraction that occurs on the left hand side of the admissibility series:

$$A^k_{D,d} = S^k_{D,d} - \frac{(1+Y)^k}{(1-X)(1-Y)(1+Y^2)^m} \tag{18}$$

This subtracts the corank of the new polynomials after evaluation, which removes the number of polynomials that reduce to 0 after evaluation (recall Equation 15). Let us refer to this term of pure Y as S_0^k.

Let $S'^k_{D,d}$ and S'^k_0 refer to $S^k_{D,d}$ and S^k_0 without the $\frac{1}{1-X}$ and $\frac{1}{1-Y}$ parts, respectively. Hence:

$$S'^k_{D,d} = (1+X)^{n-k}\left(\frac{(1+XY)^k}{(1+X^2Y^2)^m} - \frac{(1+X)^k}{(1+X^2)^m}\right)$$
$$S'^k_0 = \frac{(1+Y)^k}{(1+Y^2)^m}$$

The use of $\frac{1}{1-X}$ and $\frac{1}{1-Y}$ is to copy $S'^k_{D,d}$ and S'^k_0 to all degrees of X and Y. What this means is that since the expansion of $\frac{1}{1-X} = 1 + X + X^2 + \ldots$, we obtain:

$$\frac{(S^k_{D,d} - S^k_0)}{(1-X)(1-Y)} = (S^k_{D,d} - S^k_0) + XY(S^k_{D,d} - S^k_0) + X^2Y^2(S^k_{D,d} - S^k_0)\ldots$$

Hence, all possible combinations of $X^D Y^d$ are included in the series.

5.2.1 Example for d = 2

Now, consider the following polynomial system in $\mathbb{F}_2$ with $n = 3$:

$$F = \begin{cases} x_1x_3 + x_2x_3 + x_1 + x_3 \\ x_1x_2 + x_1 \\ x_1x_3 + x_2x_3 + x_3 \\ x_1x_3 + x_2x_3 + x_3 \end{cases}$$

The admissibility series presented in Equation 10 with $k = 2$ produces the following power series:

$$A^k_{D,d=2} = -1 - \ldots + 2X^3Y^2 + \ldots$$

Hence, we choose $D = 3$ and after evaluation, the expected number of independent polynomials in our finalised system is 2. $S^2_{D,d=2}$ produces the following admissibility series:

$$S^2_{D,d=2} = -2X - \ldots + 4X^3Y^2 + \ldots$$

Thus, we expect our finalised system to contain 4 independent polynomials before evaluation. By appending P and $Mac_2(F^*)$ and extracting only the independent polynomials, we get:

$$PM^* = \begin{cases} x_1x_2x_3 \\ x_1x_2 \\ x_1x_3 + x_2x_3 + x_3 \\ x_1 \end{cases}$$

Note how we have exactly 4 polynomials, which is what the admissibility series predicted.

By evaluating the last $n - k$ variables (*i.e.* x_3) to 0 and extracting the independent polynomials, we obtain 2 polynomials. However, if we set $x_3 = 1$, we obtain:

$$PM^*_{x_3=1} = \begin{cases} x_1x_2 \\ x_1 + x_2 + 1 \\ x_1 \end{cases}$$

This contradicts the expected result. Let us consider the columns of $M^k_{D,d}$, whose columns contain monomials whereby their total degree in the first k variables is larger than d. In this case, we have $k = 2$, meaning that the first k variables are x_1 and x_2. In $\mathbb{F}_2$, the maximum degree obtainable with 2 variables is with the monomial x_1x_2 as we cannot have squares. Henceforth, $M^k_{D,d}$ is empty as it is impossible for any monomials in the first k variables to be larger than 2. Since it is an empty matrix, it does not have a cokernel. Thus, the whole admissibility series does not correctly predict the size of the finalised system as it assumes that $M^k_{D,d}$ is not empty. However, some programming languages, such as SageMath, return an identity matrix for the left kernel of an empty matrix, which is why the above example 'worked'. Hence, for $\mathbb{F}_2$, it can be concluded that the admissibility series only correctly predicts the size of the finalised system if $\sum_{d'=0}^{d} \binom{k}{d'} > d$.

5.3 Rewriting the Admissibility Test

Recall the definition of the Hilbert Series for a semi-regular system in $\mathbb{F}_q$ for a $q > 2$:

$$S_{m,n} = \frac{\prod_{i=1}^{m}(1-t^{d_i})}{(1-t)^n}$$

If we assume a homogeneous system of degree d_F (all polynomials in the system have degree d_F), we obtain:

$$S_{m,n} = \frac{(1-t^{d_F})^m}{(1-t)^n}$$

Therefore, since the admissibility series is just a collection of Hilbert Series in $\mathbb{F}_2$, we can use this to rewrite the admissibility series. All that needs to be done is to rewrite the Hilbert Series that are present in the admissibility series in terms of the Hilbert Series for $\mathbb{F}_q$, $q > 2$ instead of $\mathbb{F}_2$. For example, consider:

$$\frac{(1+X)^n}{(1+X^2)^m}$$

which is just the Hilbert Series for a quadratic homogeneous semi-regular system in $\mathbb{F}_2$. If we now assume our homogeneous semi-regular system is of degree d_F (all polynomials has degree d_F) in $\mathbb{F}_q$ with $q > 2$, we write:

$$\frac{(1-X^{d_F})^m}{(1-X)^n}$$

Hence, we will simply apply this method to all of the admissibility series in $\mathbb{F}_2$ to construct the admissibility series for $\mathbb{F}_q$, $q > 2$. Hence, if we let:

$$S = \frac{(1-X^{d_F}Y^{d_F})^m}{(1-XY)^k(1-X)^{n-k}(1-X)(1-Y)}$$

then:

$$S^k_{D,d} = S - \frac{(1-X^{d_F})^m}{(1-X)^n(1-X)(1-Y)} \tag{19}$$

And therefore, our admissibility series will be:

$$S^k_{D,d} - \frac{(1-Y^{d_F})^m}{(1-Y)^k(1-X)(1-Y)} \tag{20}$$

As per the explanation in the previous section, the admissibility series only correctly predicts the size of our finalised system if the number of monomials in k variables of degree d is larger than d. However, in this case, since we are not working in a quotient ring, we are allowed to have powers up to and including q, if q is the size of the field. Hence, this admissibility series only works if the number of degree d monomials over k is greater than d. This equivalent of saying that $\binom{k+d-1}{d} > d$.

5.4 Columns of $M^k_{D,d}$

Firstly, let us establish that the number of monomials in n from degree 0 up to d is given by

$$\binom{n+d}{d} = \sum_{d'=0}^{d}\binom{n+d'-1}{d'} \tag{21}$$

However, in $\mathbb{F}_2$, we are working in a quotient field with field equations $x_i^2 = 0$ as per our explanation in Section 2.2. This means we do not have any squares as, clearly, $x_i^2 = 0$. Hence, we adapt the above definition for $\mathbb{F}_2$:

$$\sum_{d'=0}^{d}\binom{n}{d'} \tag{22}$$

Since the columns of the Macaulay Matrix index monomials in n variables from degree 0 to d, we can use the above equations to calculate its number of columns. Let us continue with this example and let us construct $M^k_{D,d}$, which requires us to get rid of any monomials in the columns of $Mac^k_{D,d}$ whereby their degree in the first k variables is smaller or equal to d. If we use $d = 1$ and $k = 2$, all monomials that have a degree lesser or equal to d in the first k variables are removed. In this case, we have x_1x_2.

However, we would also include, for example, x_1x_2 multiplied by any monomial that is comprised of the last $n-k$ variables such as x_3, such that the result is $x_1x_2x_3$. This multiplication would have to result in a monomial whose total degree is at most D, hence, if we let d_k represent our initial monomial's degree in the first k variables (in this case, the initial monomial is x_1x_2), the degree of the monomial with variables from the last $n-k$ variables must be at most $D-d_k$. Henceforth, this is the same as saying that we also want all monomials comprised of the last $n-k$ variables of degree 0 till $D-d_k$. This leads us to the following equation for the number of columns for $\mathbb{F}_q$, $q>2$:

$$\sum_{d_k=d+1}^{D}\sum_{d'=0}^{D-d_k}\binom{k+d_k-1}{d_k}\binom{n-k+d'-1}{d'} \tag{23}$$

In $\mathbb{F}_2$, the number of columns would be:

$$\sum_{d_k=d+1}^{D}\sum_{d'=0}^{D-d_k}\binom{k}{d_k}\binom{n-k}{d'} \tag{24}$$

[15].

6 Complexity of the Crossbred Algorithm

The complexity of the algorithm for any $\mathbb{F}_q$ has the following form:

$$\mathcal{C}_{cross_q} = \mathcal{O}(\texttt{kernel}(M^k_{D,d})) + q^{n-k}\cdot\mathcal{O}((\texttt{solving}(P^*\cup Mac_d(F^*)))) \tag{25}$$

Recall that P^* is the system P upon evaluation of the last $n-k$ variables. Block Wiedemann or Lanczös algorithms can be used to calculate the kernel of a sparse matrix. The complexity of finding kernel vectors of a sparse matrix is:

$$\begin{aligned} C_{ker} &= \mathcal{O}(n_{cols}) + \mathcal{O}(n_{cols}^2\log n_{cols}\log\log n_{cols}) \\ &= \tilde{\mathcal{O}}(n_{cols}) + \tilde{\mathcal{O}}(n_{cols}^2) \end{aligned} \tag{26}$$

whereby n_{cols} the number of columns in our matrix [13]. Let us simplify this down to $\tilde{\mathcal{O}}(n_{cols}^2)$.

We can also use the block Wiedemann or Lanczös to probabilistically test the consistency of a set of polynomials, which in our case, is $P^*\cup Mac_d(F^*)$ for $d>1$ and just P for $d=1$. This has the same complexity as C_{ker}. The reason for this is because the number of columns of both $P^*\cup Mac_d(F^*)$ and P is equal to the number of monomials in k variables from degrees 0 to d.

If we assume that we then solve our resulting system via linearisation, then we can write the complexity estimate for the Crossbred Algorithm as:

$$\mathcal{C}_{cross_{q>2}} = \tilde{\mathcal{O}}\left(\left(\sum_{d_k=d+1}^{D}\sum_{d'=0}^{D-d_k}\binom{k+d_k-1}{d_k}\binom{n-k+d'-1}{d'}\right)^2\right) + q^{n-k}\cdot\tilde{\mathcal{O}}\left(\binom{k+d-1}{d}^{\omega}\right) \tag{27}$$

And in $\mathbb{F}_2$:

$$\mathcal{C}_{cross_{q=2}} = \tilde{\mathcal{O}}\left(\left(\sum_{d_k=d+1}^{D}\sum_{d'=0}^{D-d_k}\binom{k}{d_k}\binom{n-k}{d'}\right)^2\right) + q^{n-k}\cdot\tilde{\mathcal{O}}\left(\left(\sum_{i=0}^{d}\binom{k}{i}\right)^{\omega}\right) \tag{28}$$

Whereby ω is the exponent of matrix multiplication. For the rest of the paper, when discussing cases where the value of q could be ≥ 2 we refer to the complexity of the Crossbred algorithm as $\mathcal{C}_{cross_q}$, which could either be $\mathcal{C}_{cross_{q=2}}$ or $\mathcal{C}_{cross_{q>2}}$.

6.1 Comparison with MQDSS's Estimate

In the supporting documentation for a NIST Post Quantum Cryptography Round 2 candidate, MQDSS, Samardijska et al. [18] provide a complexity for the Crossbred algorithm, for $q>2$, which has a very similar form:

$$C_{cross_{q>2}} = \mathcal{O}\left(\binom{n+D-1}{D}^2\right) + \log(n-k)\cdot q^{n-k}\cdot\binom{k+d-1}{d}^{\omega} \tag{29}$$

Note that Samardijska et al. included $\log(n-k)$. This is because $\log(n-k)$ is the amount of field operations necessary to specialise $n-k$ variables [18].

The $\mathcal{O}\left(\binom{n+D-1}{D}^2\right)$ part represents the complexity of finding kernel vectors in $M^k_{D,d}$ using, for example, the block Wiedemann algorithm. The only difference between the complexity estimate of this step in relation to the estimate provided in this work is the number of columns of $M^k_{D,d}$. Samardijska et al. [18] assumes that $M^k_{D,d}$ has $\binom{n+D-1}{D}$ columns, which is equivalent to the number of monomials in n variables of degree up to D. Clearly, $M^k_{D,d}$ will not have the same number of columns as the the initial Macaulay Matrix, making this estimate inaccurate. However, this may be interpreted as an upper bound.

The last $\binom{k+d-1}{d}^\omega$ represents solving an overdetermined system of multivariate polynomials with k variables of degree d via computing the Gaussian Elimination of a large matrix, which represents solving the system via linearisation. This is the same in our complexity estimate. Hence, the only real difference between the estimate provided in this paper is that the estimate presented by Samardijska et al. may be interpreted as an upper bound.

7 Methodology

To analyse the overall performance of the Crossbred Algorithm, n was incremented from 1 until 200 for $\mathbb{F}_2$, $\mathbb{F}_3$ and $\mathbb{F}_{256}$ We assume that all polynomials in this section are polynomials. Admissible parameters were obtained from the bivariate generating function such that they minimised the evaluation of the complexity function of the algorithm.

In terms of the field sizes, $\mathbb{F}_2$ was chosen as a baseline to compare the other results to. $\mathbb{F}_3$ was chosen since it represents a very small field, but still larger than $\mathbb{F}_2$ and $\mathbb{F}_{256}$ represents a very large field, hence, it represents a corner case. In fact, this field was taken from one of the parameter sets in the supporting documentation of a Round 2 NIST candidate, Rainbow [8, p.10]. $\mathbb{F}_7$ was also considered but it presented almost identical results to $\mathbb{F}_3$, hence, it is not included.

These complexity bounds are intended for semi-regular systems. Furthermore, to be clear, the results were obtained by finding the optimal admissible parameters and substituting them into the complexity estimate presented in the previous section. The optimal parameters were found by iterating through a triple-nested loop. The outer loops iterates over $k = 1, \ldots, n$. For each value of k, iterate over D from $2, \ldots, d_{reg}(n)$. This upper limit was chosen D is at most $d_{reg}(n)$ over m equations. Since we are assuming semi-regularity, if $\mathbb{F}_q$ with $q > 2$, we can calculate the degree of regularity by finding the first non-negative power of the Hilbert Series presented in Equation 5. If the system is in a $\mathbb{F}_2$, we instead use Equation 6. In the inner-most loop, we iterate over d from $1, \ldots, D-1$.

For each tuple (k, D, d), its admissibility is checked with the appropriate admissibility series. If it is admissible, we calculate the complexity by substituting in the values of the tuple and values of m and n into the appropriate complexity estimate equation presented in the previous section. After exiting the triple-nested loop, we output the minimum complexity found and the associated tuple.

8 Results

8.1 Results for m = 2n

It was assumed that $m = 2n$ and the results are presented in Figure 1 and Figure 2.

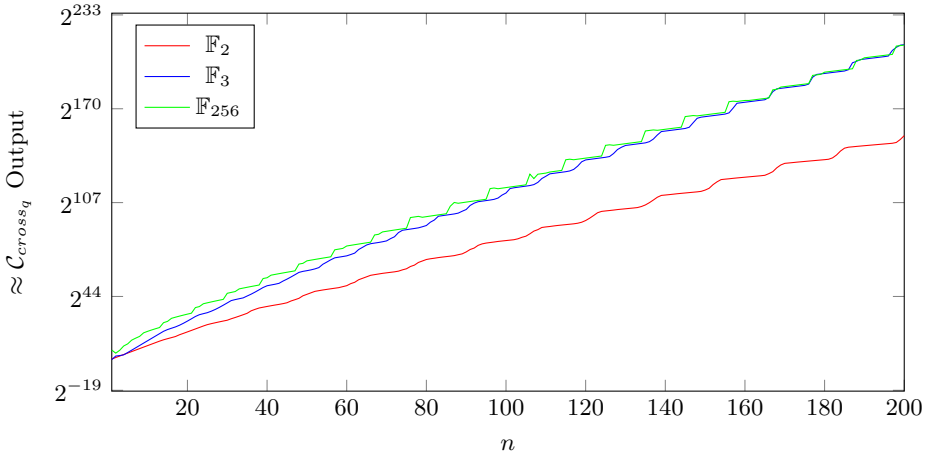

Figure 1: Optimal Cost of Crossbred as $m = 2n$ increases

The results shown in Figure 1 behave as expected, whereby as n increases, the complexity increases exponentially.

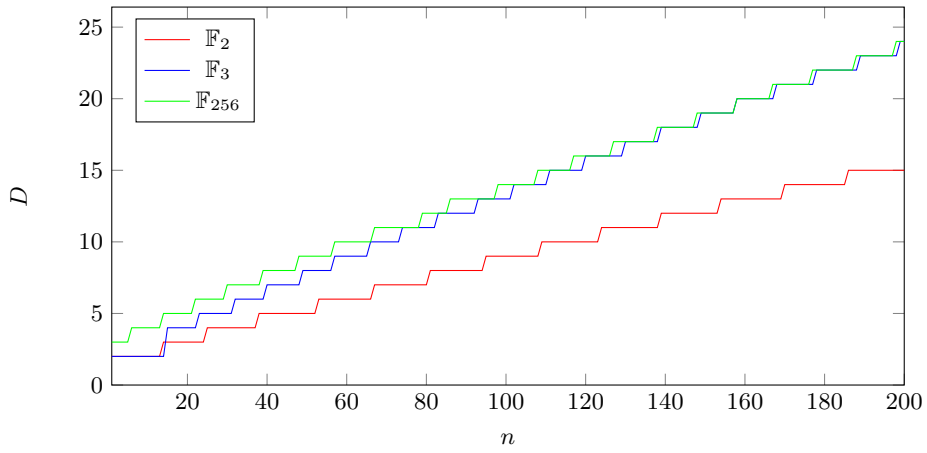

Figure 2: Growth of degree D as $m = 2n$ increases

One thing to note is that Crossbred clearly scales well to larger fields, as exemplified with $\mathbb{F}_3$ and $\mathbb{F}_{256}$.

8.1.1 Comparison to FES

To compare Crossbred and FES, n was enumerated from 1 till 200 and its complexity was calculated. Recall that for $\mathbb{F}_q$, the complexity of FES is $\log_q(n)q^n$. The results of this are plotted in Figure 3 and the results for the Crossbred algorithm are also included for ease of comparison.

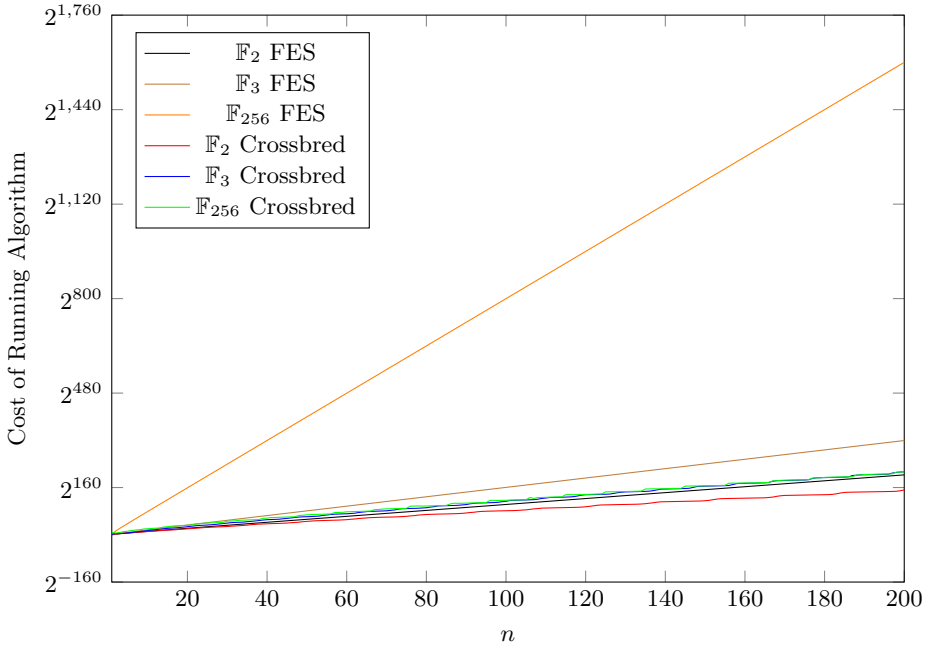

Figure 3: Optimal Cost of Running FES and Crossbred as $m = 2n$ increases

In $\mathbb{F}_2$, the difference between Crossbred and FES may be considered irrelevant as it may be due to small missing factors in either complexity estimates. Hence, in this case, despite practical speed, the Crossbred algorithm does not provide an improved complexity estimate. However, for larger fields, Crossbred clearly provides an improved complexity, which is in line with the practical findings of Joux and Vitse [15], even if their results were limited to $\mathbb{F}_2$.

8.1.2 Comparison to Hybrid-F_5

The methodology was used for comparison was the same used to compare the FES approach and Crossbred.

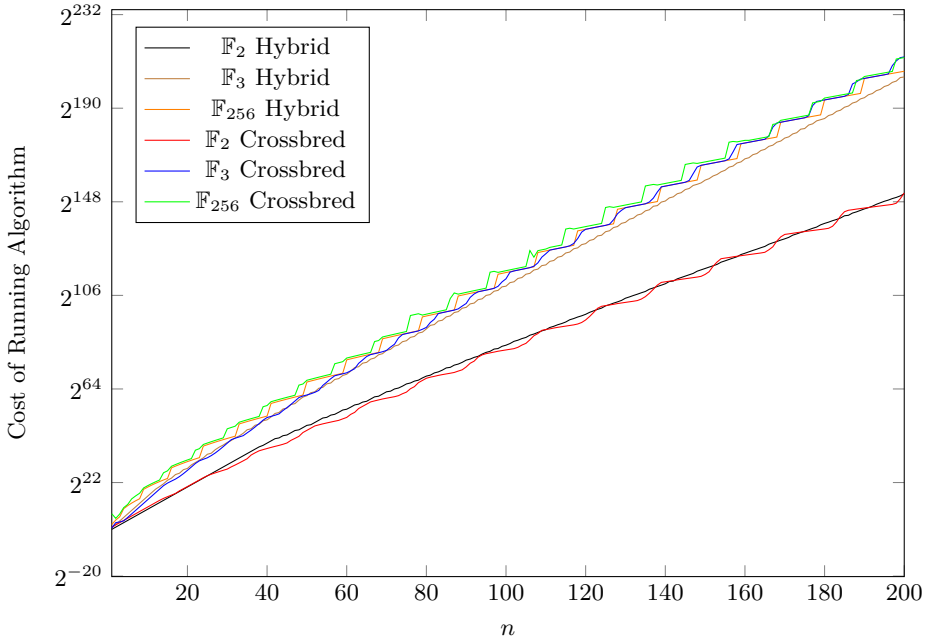

Figure 4: Optimal Cost of Running Hybrid-F_5 and Crossbred as $m = 2n$ increases

The complexities are very similar throughout for both the Crossbred and Hybrid approach. In general, Crossbred provided no asymptotic improvement over Hybrid-F_5.

An interesting note is that for $\mathbb{F}_{256}$, Hybrid-F_5 almost always sets the value of $k = 0$, indicating that for large fields, Hybrid-F_5 may degenerate down to F_5.

8.2 Results for m = n + 1

The methodology for this is identical to the previous section. The reason for selecting $m = n + 1$ is because it is the 'least' overdetermined system possible and hence, it can be considered a corner or boundary case. The results are presented in Figure 5.

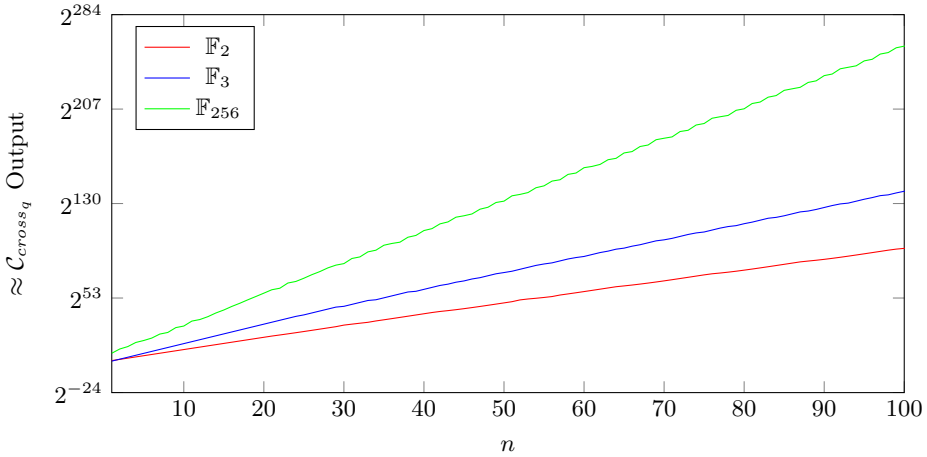

Figure 5: Optimal Cost of Crossbred as $m = n + 1$ increases

The results are very similar to the previous case and hence, there will not be much discussion surrounding them. However, it should be noted that the complexity is much larger in size, meaning that the Crossbred algorithm performs worse when $m = n + 1$. The increase in the complexity of the Crossbred algorithm is due to a larger degree of regularity as our ratio between equations and variables is almost $1 : 1$. Furthermore, there is a much larger separation between $\mathbb{F}_3$ and $\mathbb{F}_{256}$ in terms of their complexities.

8.2.1 Comparison to FES

The methodology used for comparison was the same used to compare the FES approach and Crossbred when $m = 2n$. The results can be found in Figure 6.

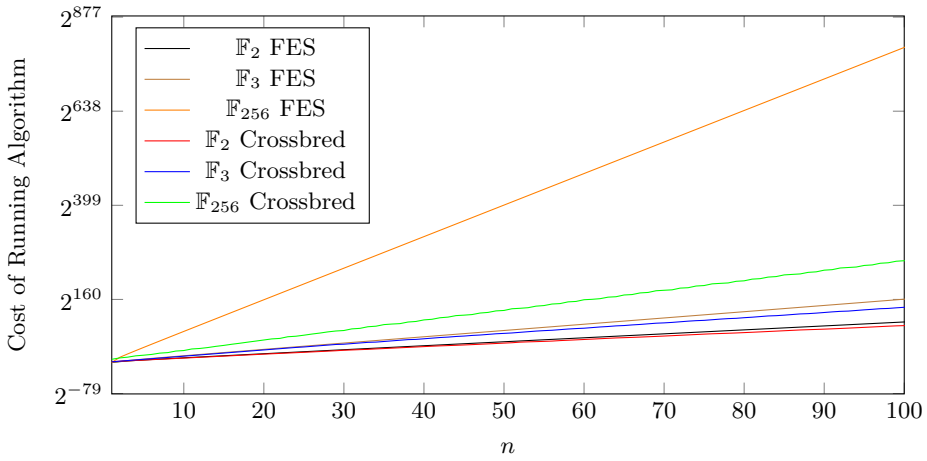

Figure 6: Optimal Cost of Running FES and Crossbred as $m = n+1$ increases

For $\mathbb{F}_2$ and $\mathbb{F}_3$, the Crossbred approach provides little to no asymptotic improvement over FES. However, for larger fields, it is clear that there is a significant improvement.

8.2.2 Comparison to Hybrid-F_5

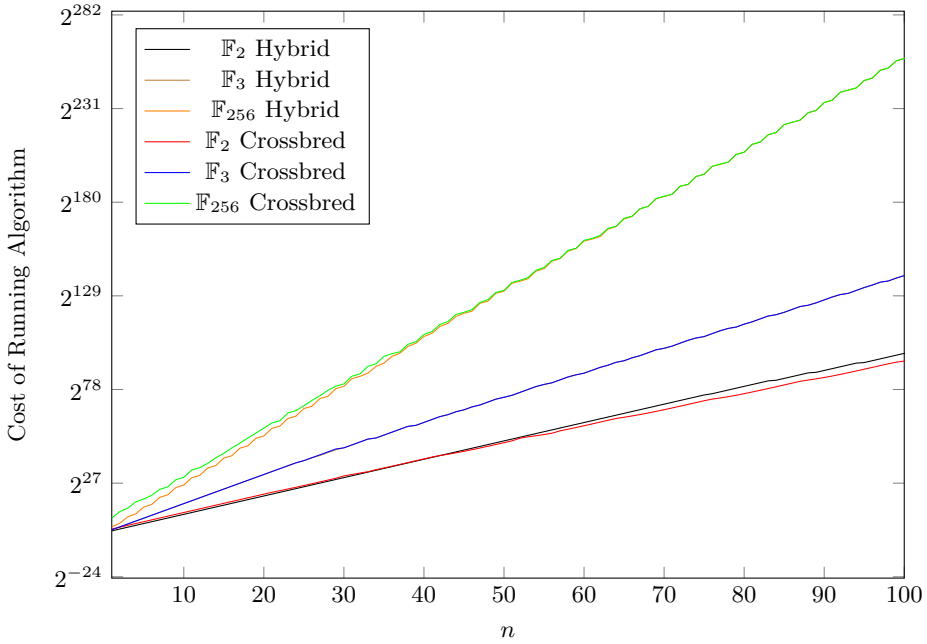

Figure 7: Optimal Cost of Running Hybrid-F_5 and Crossbred as $m = n + 1$ increases

Once again, Crossbred provides no asymptotic complexity improvement over the Hybrid approach. The reason the line representing $\mathbb{F}_3$ for Hybrid-F_5 is not visible is that the results of $\mathbb{F}_3$ for Crossbred are so similar that Crossbred covers Hybrid's results.

9 Conclusion

In conclusion, the bivariate generating function to test the admissibility of parameters to the algorithm in $\mathbb{F}_2$ was investigated and a new series for $\mathbb{F}_q$, $q > 2$ was presented. A complexity estimate was given for the Crossbred algorithm for both $\mathbb{F}_2$ and $\mathbb{F}_q$, $q > 2$. By plotting the best-case complexities of theoretically applying the Crossbred, FES and Hybrid-F_5 to polynomial systems of various sizes of n and q, it was determined that for larger fields, the Crossbred algorithm provides an improved asymptotic complexity over FES. However, it does not provide any improvement over Hybrid-F_5.

Whilst this question about whether the Crossbred attack poses any improvement over the state of the art algorithms, namely FES and Hybrid-F_5, is answered, there is still much research to be done. This is because the reason why this topic was investigated to begin with is because it is believed that there probably exists a better way of solving polynomial systems than F_5 or FES and by scrutinising these sort of algorithms, we are able to fully understand them. Hence, we are able to make an informed decision of whether an algorithm is an improvement on previous work and if so, how much of an improvement. In the end, even a slight improvement may be the difference between breaking an algorithm in the real-world or not.

Acknowledgements

This work is based on João Diogo Duarte's Master Thesis at Royal Holloway, University of London, supervised by Prof. Martin Albrecht. Special thanks to Dr. Simona Samardijska for reviewing the initial manuscript before submission.

References

[1] M Bardet, Jean-Charles Faugère, B Salvy, and Bo-Yin Yang. Asymptotic Behaviour of the Degree of Regularity of Semi-Regular Polynomial Systems. In *MEGA'05*, Sardinia, 2005.

[2] Magali Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* Theses, Université Pierre et Marie Curie - Paris VI, December 2004.

[3] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Pierre-Jean Spaenlehauer. On the complexity of solving quadratic boolean systems. *Journal of Complexity*, 29(1):53–75, 02 2013.

[4] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Hybrid Approach for Solving Multivariate Systems over Finite Fields. *Journal of Mathematical Cryptology*, 3:177–197, 2008.

[5] Ward Beullens, Bart Preneel, Alan Szepieniec, and Frederik Vercauteren. LUOV Signature Scheme Proposal for NIST PQC Project, 2019.

[6] Charles Bouillaguet, Chen-Mou Cheng, Tony (Tung) Chou, Ruben Niederhagen, Adi Shamir, and Bo-Yin Yang. Fast Exhaustive Search for Polynomial Systems in F_2. Cryptology ePrint Archive, Report 2010/313, 2010.

[7] David Cox, John Little, and Donal O'Shea. *Ideals, Varieties and Algorithms.* Springer International Publishing, 4 edition, 2012.

[8] Jintai Ding, Ming-Shing Chen, Albrecht Petzoldt, Dieter Schmidt, and Bo-Yin Yang. Rainbow Specification, 2019.

[9] Jintai Ding and Bo-Yin Yang. *Multivariate Public Key Cryptography*, pages 193–241. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[10] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1-3):61–88, 6 1999.

[11] Jean-Charles Faugère, Kelsey Horan, Delaram Kahrobaei, Marc Kaplan, Elham Kashefi, and Ludovic Perret. Fast Quantum Algorithm for Solving Multivariate Quadratic Equations. *CoRR*, abs/1712.07211, 2017.

[12] Jean Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). *Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC*, pages 75–83, 01 2002.

[13] M Giesbrecht, A Lobo, and B D Saunders. Certifying Inconsistency of Sparse Linear Systems. *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 113–119, 1998.

[14] T. J. Hodges, S. D. Molina, and J. Schlather. On the existence of semi-regular sequences, 2014.

[15] Antoine Joux and Vanessa Vitse. A Crossbred Algorithm for Solving Boolean Polynomial Systems. In Jerzy Kaczorowski, Josef Pieprzyk, and Jacek Pomykała, editors, *Number-Theoretic Methods in Cryptology*, pages 3–21. Springer International Publishing, 2017.

[16] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar Signature Schemes. *Advances in Cryptology — EUROCRYPT '99*, pages 206–222, 1999.

[17] Daniel Lazard. Gröbner-Bases, Gaussian Elimination and Resolution of Systems of Algebraic Equations. In *Proceedings of the European Computer Algebra Conference on Computer Algebra*, EUROCAL Ś83, pages 146–156, London, UK, 1983. Springer-Verlag.

[18] Simona Samardijska, Ming-Shing Chen, Andreas Hulsing, Joos Rijneveld, and Peter Schwabe. MQDSS Specifications, 2017.

[19] Dan Shumow, Josh Benaloh, Tolga Acar, and Craig Costello. Evaluating Post-Quantum Asymmetric Cryptographic Algorithm Candidates. In *Workshop on Cybersecurity in a Post-Quantum World.* NIST, 2015.

[20] Enrico Thomae and Christopher Wolf. Solving Underdetermined Systems of Multivariate Quadratic Equations Revisited. In *Public Key Cryptography – PKC 2012*, volume 7293 of *Lecture Notes in Computer Science*, page 156–171, 2012.

[21] Christopher Wolf. Using Multivariate Quadratic Polynomials in Public Key Cryptography. In *DIAMANT/EIDMA symposium 2005.* DIAMANT, 2005.