

Higher Order Differentiation over Finite Fields with Applications to Generalising the Cube Attack

Ana Sălăgean*

Matei Mandache-Sălăgean†

Richard Winter*

Raphael C.-W. Phan ‡

August 7, 2018

Abstract

Higher order differentiation was introduced in a cryptographic context by Lai. Several attacks can be viewed in the context of higher order differentiations, amongst them the cube attack and the AIDA attack. All of the above have been developed for the binary case.

We examine differentiation in larger fields, starting with the field $\text{GF}(p)$ of integers modulo a prime p . We prove a number of results on differentiating polynomials over such fields and then apply these techniques to generalising the cube attack to $\text{GF}(p)$. The crucial difference is that now the degree in each variable can be higher than one, and our proposed attack will differentiate several times with respect to each variable (unlike the classical cube attack and its larger field version described by Dinur and Shamir, both of which differentiate at most once with respect to each variable).

Finally we describe differentiation over finite fields $\text{GF}(p^m)$ with p^m elements and prove that it can be reduced to differentiation over $\text{GF}(p)$, so a cube attack over $\text{GF}(p^m)$ would be equivalent to cube attacks over $\text{GF}(p)$.

Keywords: Higher order differentiation, cube attack, higher order derivative.

1 Introduction

The original motivation for this work was to generalise the cube attack from the binary field to arbitrary finite fields. While doing so, we developed a number of tools and results for differentiation over finite fields which could have a broader applicability in cryptography.

Higher order differentiation was introduced in a cryptographic context by Lai in [14] (called there higher order derivative). This notion had already been used for a very long time, under the name of finite difference, in other areas of mathematics (notably for the numerical approximation of the derivative).

The finite difference for a function f is defined as the function $(\Delta_a f)(\mathbf{x}) = f(\mathbf{x} + \mathbf{a}) - f(\mathbf{x})$, for a fixed difference \mathbf{a} (the domain and codomain of f are commutative groups in additive notation). Usually f is a function of n variables, so $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{a} = (a_1, \dots, a_n)$. An important particular case is

*A. Sălăgean and R. Winter are with the Department of Computer Science, Loughborough University, Loughborough, UK, email: {A.M.Salagean, R.Winter}@lboro.ac.uk

†M. Mandache-Sălăgean is with Trinity College, University of Cambridge, UK, email: mfm41@cam.ac.uk

‡R. Phan is with the Faculty of Engineering, Multimedia University, Malaysia, email: raphael@mmu.edu.my

the finite difference with respect to one variable, namely $\mathbf{a} = h\mathbf{e}_i$ where the difference step h is a scalar constant (equal to 1 by default) and \mathbf{e}_i are the elementary vectors having a 1 in position i and zeroes elsewhere. Higher order differentiation means repeated application of the finite difference operator.

The functions we use here are functions in several variables over a finite field. Any such function can be represented as a polynomial function and after a sufficiently high number of applications of the finite difference operator the result is the identically zero function. However for certain choices of differences \mathbf{a} , this can happen prematurely, for example over the binary field $\text{GF}(2)$ differentiating twice using the same difference \mathbf{a} will always result in the zero function, regardless of the original function f . For our applications, we need to ensure that this does not happen prematurely.

A number of cryptographic attacks can be reformulated using higher order differentiation. Differential cryptanalysis (introduced by Biham and Shamir [3]) has been thus reformulated by Lai in [14]; the cube attack of Dinur and Shamir [5] and the related AIDA attack of Vielhaber [17] have been reformulated in Knellwolf and Meier [11], Duan and Lai [7].

Our main motivation came from the cube attack. In both the cube attack and the AIDA attack we have a “black box” function f in several public and secret variables and we select a set of indices of public variables $I = \{i_1, \dots, i_k\}$. Then f is evaluated at each point of a “cube” consisting of the vectors that have all the possible combinations of 0/1 values for the variables with index in I , whereas the remaining variables are left indeterminate; the resulting values are summed and the sum will be denoted f_I . The attacks hope that for suitable choices of subsets I of public variables, the resulting f_I is linear in the secret variables, for the cube attack (or equals to one secret variable or the product of several secret variables for the AIDA attack). This situation is particularly likely when the cardinality of I is just marginally lower than the total degree of the function. Such subsets I are found in a preprocessing phase, where the values of the keys can be chosen freely. In the online phase the key variables are unknown, and by computing the f_I for the sets I identified in the preprocessing phase, one obtains a system of linear equations in the key variables.

It was shown (see Knellwolf and Meier [11], Duan and Lai [7]) that choosing the variable indices $I = \{i_1, \dots, i_k\}$ and computing f_I (as described above) is equivalent to computing the k -order finite difference of f with respect to the elementary vectors $\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}$, i.e. by differentiating once with respect to x_{i_1} , then w.r.t. x_{i_2} and so on, finally differentiating w.r.t. x_{i_k} .

All the attacks above, as well as the higher order differentiation used in cryptography are over the binary field. While all cryptographic functions can be viewed as binary functions, there are a number of ciphers which make significant use of operations modulo a prime $p > 2$ in their internal processing, for example ZUC [9], IDEA [15, 16], MMB [4]. It may therefore be advantageous for such ciphers to also be viewed and analysed as functions over $\text{GF}(p)$, the field of integers modulo p . Unlike the binary case, a polynomial function can now have degree more than one in each variable, in fact it can have degree up to $p - 1$ in each variable. There are yet other ciphers which use operations over Galois fields of the form $\text{GF}(p^m)$, for example SNOW [8] and in such fields the degree of the polynomial functions can be up to $p^m - 1$ in each variable.

A first generalisation of the cube attack to $\text{GF}(p^m)$ was sketched by Dinur and Shamir in [5, page 284] and also developed more explicitly by Agnese and Pedicini [1]. We show that their approach can again be viewed as k -order differentiation, where we differentiate once with respect to each of the variables x_{i_1}, \dots, x_{i_k} . However we argue that their generalisation, while correct, has very low chances to lead to a successful attack because we don’t differentiate sufficiently many times. Namely, on one hand, like in the binary case, the best chances of success are when the function is differentiated a number of times just marginally lower than its total degree; on the other hand in their proposed scheme the number of times that the function is differentiated is upper bounded by the number of variables, which (unlike the binary

case) can be significantly lower than the degree of the function (see Remark 12).

Our proposed generalisation of the cube attack to $\text{GF}(p)$ improves the chances of success by differentiating several times with respect to each of the chosen variables. Thus there is no intrinsic limit on the number of differentiations and therefore this number can be as close as we want to the degree of the polynomial (only limited by the computing power available).

We first examine higher order differentiation in $\text{GF}(p)$ (Section 4.1). We show that for repeated differentiation with respect to the same variable, we can use any non-zero difference steps and the degree will decrease by exactly one for each differentiation. Choosing all the steps equal to one gives a compact and efficient formula for evaluating the higher order differentiation for a “black box” function.

We then show, in Section 4.2 that the main result of the classical cube attack, [5, Theorem 1], no longer holds when we differentiate repeatedly with respect to the same variable in $\text{GF}(p)$; Example 20 gives a counterexample. However, we show that a similar result does hold, see Theorem 21. Also, just like in the binary case, if the “black box” function has total degree d , differentiating $d - 1$ times with respect to public variables always results in a function which is either constant or is linear in the secret variables. The resulting algorithm is sketched in Section 4.3. Now we not only choose variables for the “cube” but we also choose the number of times we are going to differentiate with respect to each variable. For computational efficiency, choosing only one variable (or a small number of variables) and differentiating a large number of times with respect to that variable is preferable. In $\text{GF}(p)$ probabilistic linearity testing has a smaller expected number of tests than in $\text{GF}(2)$, see [10].

While this paper concentrates on generalising the cube attack, other attacks that use differentiation could also be generalised to $\text{GF}(p)$ using our technique, for example cube testers (see [2]) or differential cryptanalysis.

Finally, for completeness, we deal with generalisations to finite fields of the form $\text{GF}(p^m)$ in Section 5.2. Here, for functions such as x^d with $p \mid d$, differentiation with respect to x decreases the degree by more than one regardless of the difference step. We give a more precise expression of the decrease in degree for higher order differentiation depending on the representation of the degree in base p . Any function can be differentiated at most $m(p - 1)$ times before it becomes identically zero. Moreover, in order to avoid the result becoming identically zero even earlier, the difference steps will be chosen as follows: $p - 1$ steps equal to b_0 , $p - 1$ steps equal to b_1 and so on, where b_0, \dots, b_{m-1} is a base of $\text{GF}(p^m)$ when viewed as a vector space over $\text{GF}(p)$. We can thus differentiate $m(p - 1)$ times. Due to the fact that differentiation only uses the additive group of $\text{GF}(p^m)$, which is isomorphic to $\text{GF}(p)^m$, differentiation over $\text{GF}(p^m)$ can in fact be reduced to differentiation over $\text{GF}(p)$ in each component of the projection of the function f . Therefore, we feel that developing a cube attack in $\text{GF}(p^m)$, while possible, does not bring any additional advantages compared to a cube attack in $\text{GF}(p)$.

2 Preliminaries

Throughout this paper R denotes an arbitrary commutative ring with identity and $\text{GF}(p^m)$ denotes the finite field with p^m elements where p is prime. We denote by $\mathbf{e}_i = (0, \dots, 0, 1, 0, \dots, 0) \in R^n$ the vector which has a 1 in position i and zeroes elsewhere, i.e. $\mathbf{e}_1, \dots, \mathbf{e}_n$ is the canonical basis of the vector space R^n .

We recall the definition of differentiation, which was introduced in the cryptographic context by Lai in [14]. This notion was used long before, under the name finite difference, in other areas of mathematics, notably for approximating the derivative.

Definition 1. Let $f : R^n \rightarrow R^s$ be a function in n variables x_1, \dots, x_n . Let $\mathbf{a} = (a_1, \dots, a_n) \in R^n \setminus \{\mathbf{0}\}$. The finite difference operator (or differentiation operator) with respect to \mathbf{a} associates to each function f the function $\Delta_{\mathbf{a}} f : R^n \rightarrow R^s$ defined as

$$\Delta_{\mathbf{a}} f(x_1, \dots, x_n) = f(x_1 + a_1, \dots, x_n + a_n) - f(x_1, \dots, x_n).$$

Denoting $\mathbf{x} = (x_1, \dots, x_n)$ we can also write $\Delta_{\mathbf{a}} f(\mathbf{x}) = f(\mathbf{x} + \mathbf{a}) - f(\mathbf{x})$.

For the particular case of $a = h\mathbf{e}_i$ for some $1 \leq i \leq n$ and $h \in R \setminus \{0\}$, we will call $\Delta_{h\mathbf{e}_i}$ the finite difference operator (or differentiation) with respect to the variable x_i with step h , or simply the finite difference operator with respect to the variable x_i if $h = 1$ or if h is clear from the context. We will use the abbreviation “w.r.t. x_i ” for “with respect to x_i ”.

Remark 2. Note that in the cryptographic literature this operator (and the resulting function) is usually called the derivative or differential (see [14, 12]). We will avoid the term derivative because of the risk of confusion with the well established mathematical notion of *formal derivative* of a polynomial. For a polynomial $\sum_{i=0}^d c_i x^i \in R[x]$ the formal derivative w.r.t. x is defined as $\sum_{i=1}^d c_i i x^{i-1}$. It can easily be seen that the formal derivative operator w.r.t. x_i coincides with the finite difference operator w.r.t. x_i only for polynomials which have degree at most one in x_i . Polynomial functions over GF(2) have degree at most one in each variable, so in this case these notions coincide. Hence the use of the term “derivative” for $\Delta_{h\mathbf{e}_i} f(x_1, \dots, x_n)$ is justified for polynomials over GF(2), but not for polynomials over other rings/fields.

Remark 3. For defining the finite difference operator, we do not actually need to work over a ring R , a commutative group (using additive notation for convenience) is sufficient. Here we used a ring due to our application to finite fields, and also due to some of the techniques involving polynomials.

The finite difference operator is a linear operator; it is commutative and associative. Repeated application of the operator (also called higher order differentiation or higher order derivative in [14]) will be denoted by

$$\Delta_{\mathbf{a}_1, \dots, \mathbf{a}_k}^{(k)} f = \Delta_{\mathbf{a}_1} \Delta_{\mathbf{a}_2} \dots \Delta_{\mathbf{a}_k} f$$

where $\mathbf{a}_1, \dots, \mathbf{a}_k \in R^n$ are not necessarily distinct. An explicit formula can be obtained easily from Definition 1 by induction:

Proposition 4. Let $f : R^n \rightarrow R^s$ be a function in n variables x_1, \dots, x_n . Let $\mathbf{a}_1, \dots, \mathbf{a}_k \in R^n \setminus \{\mathbf{0}\}$ not necessarily distinct. Then

$$\Delta_{\mathbf{a}_1, \dots, \mathbf{a}_k}^{(k)} f(\mathbf{x}) = \sum_{j=0}^k (-1)^{k-j} \sum_{\{i_1, \dots, i_j\} \subseteq \{1, \dots, k\}} f(\mathbf{x} + \mathbf{a}_{i_1} + \dots + \mathbf{a}_{i_j}).$$

Depending of the values of the $\mathbf{a}_1, \dots, \mathbf{a}_k$ and the characteristic of the ring, $\Delta_{\mathbf{a}_1, \dots, \mathbf{a}_k}^{(k)} f$ could collapse, becoming the identical zero function regardless of the function f . (This happens, for example, if the ring is GF(2) and $\mathbf{a}_1, \dots, \mathbf{a}_k$ are not linearly independent.) When differentiating w.r.t. one variable we need to choose the difference steps so that that this does not happen. Details will be given in Section 4.1 for finite fields of the form GF(p) and in Section 5.2 for finite fields of the form GF(p^m).

While the finite difference operator can be defined for any function, in the sequel we will concentrate on polynomial functions. We will denote by $\deg_{x_i}(f)$ the degree of f in the variable x_i . The total degree will be denoted $\deg(f)$. The following three results are well known and straightforward, but will be needed later. The first result states that differentiating with respect to one variable decreases the degree in that variable by at least one. The other propositions deal with the results of the differentiation in a few simple cases.

Proposition 5. Let $f : R^n \rightarrow R$ be a polynomial function. Let $h, h_1, \dots, h_k \in R \setminus \{0\}$ and $i \in \{1, \dots, n\}$. If $\deg_{x_i}(f) = 0$ then $\Delta_{h\mathbf{e}_i} f(x_1, \dots, x_n) \equiv 0$. If $\deg_{x_i}(f) > 0$ then $\deg_{x_i}(\Delta_{h\mathbf{e}_i} f) \leq \deg_{x_i}(f) - 1$. Consequently $\deg_{x_i}(\Delta_{h_1\mathbf{e}_i, \dots, h_k\mathbf{e}_i}^{(k)} f) \leq \deg_{x_i}(f) - k$ if $k \leq \deg_{x_i}(f)$, and $\Delta_{h_1\mathbf{e}_i, \dots, h_k\mathbf{e}_i}^{(k)} f$ is identically zero if $k > \deg_{x_i}(f)$.

Proposition 6. Let $h \in R$, $h \neq 0$ and $i \in \{1, \dots, n\}$. Let $f : R^n \rightarrow R$ be a polynomial function with $\deg_{x_i}(f) = 1$ i.e. $f(x_1, \dots, x_n) = x_i g_1(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) + g_2(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$ (g_1 and g_2 are polynomial functions that do not depend on x_i). Then

$$\Delta_{h\mathbf{e}_i} f(x_1, \dots, x_n) = h g_1(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n).$$

Proposition 7. Let $f : R^n \rightarrow R$ be a polynomial function. Let $h \in R$, $h \neq 0$ and $i \in \{1, \dots, n\}$. Factoring out x_i we write $f(x_1, \dots, x_n) = x_i g_1(x_1, \dots, x_n) + g_2(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$ (g_2 is a polynomial function that does not depend on x_i , but g_1 may depend on x_i). Then

$$(\Delta_{h\mathbf{e}_i} f)(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) = h g_1(x_1, \dots, x_{i-1}, h, x_{i+1}, \dots, x_n).$$

Recall that for integers d, k_1, k_2, \dots, k_s such that $\sum_{i=1}^s k_i = d$ and $k_i \geq 0$ the multinomial is defined as:

$$\binom{d}{k_1, k_2, \dots, k_s} = \frac{d!}{k_1! k_2! \cdots k_s!}.$$

One combinatorial interpretation is the number of ways that we can distribute n objects into s (labeled) boxes, so that the first box has k_1 elements, the second k_2 elements e.t.c. Multinomials are generalisations of the usual binomial coefficients, with

$$\binom{d}{k} = \binom{d}{k, d-k}.$$

Next we examine the effect of higher order differentiation on univariate monomials; the general formula for univariate polynomials can be obtained using the linearity of the Δ operator.

Theorem 8. Let $f : R \rightarrow R$ defined by $f(x) = x^d$. Let $h_1, \dots, h_k \in R \setminus \{0\}$

$$\Delta_{h_1\mathbf{e}_1, \dots, h_k\mathbf{e}_1}^{(k)} x^d = \sum_{j=k}^d \left(\sum_{\substack{(i_1, \dots, i_k) \in \{1, 2, \dots, j-k+1\}^k \\ i_1 + \dots + i_k = j}} \binom{d}{i_1, \dots, i_k, d-j} h_1^{i_1} \cdots h_k^{i_k} \right) x^{d-j}$$

Proof. Induction on k . For $k = 1$ we have $\Delta_{h_1\mathbf{e}_1} x^d = (x+h_1)^d - x^d = \sum_{j=1}^d \binom{d}{j} h_1^j x^{d-j} = \sum_{j=1}^d \binom{d}{j, d-j} h_1^j x^{d-j}$ and the statement is verified.

Not let us assume the statement holds for a given k and we prove it for $k + 1$.

$$\begin{aligned}
& \Delta_{h_1 \mathbf{e}_1, \dots, h_{k+1} \mathbf{e}_1}^{(k+1)} x^d = \Delta_{h_{k+1} \mathbf{e}_1} \Delta_{h_1 \mathbf{e}_1, \dots, h_k \mathbf{e}_1}^{(k)} x^d \\
&= \sum_{j=k}^d \left(\sum_{\substack{(i_1, \dots, i_k) \in \{1, 2, \dots, j-k+1\}^k \\ i_1 + \dots + i_k = j}} \binom{d}{i_1, \dots, i_k, d-j} h_1^{i_1} \cdots h_k^{i_k} \right) ((x + h_{k+1})^{d-j} - x^{d-j}) \\
&= \sum_{j=k}^d \left(\sum_{\substack{(i_1, \dots, i_k) \in \{1, 2, \dots, j-k+1\}^k \\ i_1 + \dots + i_k = j}} \binom{d}{i_1, \dots, i_k, d-j} h_1^{i_1} \cdots h_k^{i_k} \right) \left(\sum_{i_{k+1}=1}^{d-j} \binom{d-j}{i_{k+1}} h_{k+1}^{i_{k+1}} x^{d-j-i_{k+1}} \right) \\
&= \sum_{j'=k+1}^d \left(\sum_{\substack{(i_1, \dots, i_{k+1}) \in \{1, 2, \dots, j'-k\}^{k+1} \\ i_1 + \dots + i_{k+1} = j'}} \binom{d}{i_1, \dots, i_{k+1}, d-j'} h_1^{i_1} \cdots h_{k+1}^{i_{k+1}} \right) x^{d-j'}.
\end{aligned}$$

The last line uses the identity:

$$\binom{d}{i_1, \dots, i_k, d-j} \binom{d-j}{i_{k+1}} = \binom{d}{i_1, \dots, i_{k+1}, d-j'}$$

where $j = i_1 + \dots + i_k$ and $j' = j + i_{k+1}$. \square

Recall that in a finite field $\text{GF}(p^m)$ we have $a^{p^m} = a$ for all elements a . Hence, while (formal) polynomials over $\text{GF}(p^m)$ could have any degree in each variable, when we talk about the associated *polynomial function*, there will always be a unique polynomial of degree at most $p^m - 1$ in each variable which defines the same polynomial function. In other words we are working in the quotient ring $\text{GF}(p^m)[x_1, \dots, x_n]/\langle x_1^{p^m} - x_1, \dots, x_n^{p^m} - x_n \rangle$, and we use as representative of each class the unique polynomial which has degree at most $p^m - 1$ in each variable.

Moreover, all functions in n variables over a finite field can be written as polynomial functions of n variables. The polynomial can be obtained by interpolation from the values of the function at each point in its (finite) domain. (This is obviously not the case for infinite fields). To summarise, each function in n variables over $\text{GF}(p^m)$ can be uniquely expressed as a polynomial function defined by a polynomial in $\text{GF}(p^m)[x_1, \dots, x_n]$ of degree at most $p^m - 1$ in each variable.

3 Classical cube attack and differentiation

In this section we first recall the classical cube attack from [5], and its interpretation in the framework of higher order differentials (see [11, 7]). We then recall a first generalisation to higher fields sketched in [5] (see also [1]).

In the cube attack ([5]), one has a “black box” polynomial function $f : \text{GF}(2)^n \rightarrow \text{GF}(2)$ in n variables x_1, \dots, x_n . Recall that polynomial functions over $\text{GF}(2)$ have degree at most one in each variable. (Note that the function is named p in the cube attack papers, but we had to rename it f as later we will work in fields of characteristic other than 2, and we felt p was a well-established notation for the characteristic.)

The next definitions are taken from [5]: “Any subset I of size k defines a k dimensional Boolean cube C_I of 2^k vectors in which we assign all the possible combinations of 0/1 values to variables in I and leave all the other variables undetermined. Any vector $\mathbf{v} \in C_I$ defines a new derived polynomial $f_{|\mathbf{v}}$ with $n - k$ variables (whose degree may be the same or lower than the degree of the original polynomial). Summing these derived polynomials over all the 2^k possible vectors in C_I we end up with a new polynomial which is denoted by $f_I = \sum_{\mathbf{v} \in C_I} f_{|\mathbf{v}}$.” Note that the computation of f_I requires 2^k calls to the “black box” function f . On the other hand denoting by t_I the product of the variables with indices in $I = \{i_1, \dots, i_k\}$, i.e. $t_I = x_{i_1} \cdots x_{i_k}$, we can factor the common subterm t_I out of some of the terms in f and write f as

$$f(x_1, \dots, x_n) = t_I f_{S(I)} + r(x_1, \dots, x_n).$$

where each of the terms of $r(x_1, \dots, x_n)$ misses at least one of the variables with index in I . Note that $f_{S(I)}$ is a polynomial in the variables with indices in $\{1, 2, \dots, n\} \setminus I$.

The cube attack is based on the following main result:

Theorem 9. ([5, Theorem 1]) *For any polynomial f and subset of variables I , $f_I \equiv f_{S(I)} \pmod{2}$.*

This result was reformulated using higher order differentials by several authors ([11, 7]). We present such a reformulation using our notations:

Theorem 10. *For any polynomial f and subset of variables $I = \{i_1, \dots, i_k\}$, we have $\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f = f_I = f_{S(I)}$.*

Proof. To show that $f_I = \Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f$ we use Proposition 4. We have

$$\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f(\mathbf{x}) = \sum_{(b_1, \dots, b_k) \in GF(2)^k} f(x_1, \dots, x_{i_1-1}, x_{i_1} + b_1, x_{i_1+1}, \dots, x_{i_k} + b_k, \dots, x_n)$$

Note that evaluating the expression above for any fixed constant values of x_{i_1}, \dots, x_{i_k} yields f_I . Hence $\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f$ does not depend on x_{i_1}, \dots, x_{i_k} and is equal to f_I . By Theorem 9, $f_I = f_{S(I)}$. \square

For the cube attack we are particularly interested in the situation when $f_{S(I)}$ (and therefore f_I) has degree exactly one, i.e. it is linear but not constant (the corresponding term t_I is then called maxterm in [5]). Let d be the total degree of f . Then I having $d - 1$ elements is a sufficient (but not necessary) condition for $f_{S(I)}$ to have degree at most one, i.e. to be linear or constant.

Generalising the cube attack from the binary field to $GF(p^m)$ was sketched in [5]: “Over a general field $GF(p^m)$ with $p > 2$, the correct way to apply cube attacks is to alternately add and subtract the outputs of the master polynomial with public inputs that range only over the two values 0 and 1 (and not over all their possible values of $0, 1, \dots, p$), where the sign is determined by the sum (modulo 2) of the vector of assigned values.”

We make this idea more precise; this was also done in [1] but we will follow a simpler approach for the proof of the main result. Let f be again a function of n variables x_1, \dots, x_n , but this time over an arbitrary finite field $GF(p^m)$. Note that now f can have degree up to $p^m - 1$ in each variable.

As before, we select a subset of k indices $I = \{i_1, \dots, i_k\} \subseteq \{1, 2, \dots, n\}$ and consider a “cube” C_I consisting of the n -tuples which have all combinations of the values 0/1 for the variables with indices in I , while the other variables remain indeterminate. The function f is evaluated at the points in the cube and these values are summed with alternating + and - signs obtaining a value

$$f_I = \sum_{\mathbf{v} \in C_I} (-1)^{k-w(\mathbf{v})} f_{|\mathbf{v}}$$

where $w(\mathbf{v})$ denotes the Hamming weight of \mathbf{v} ignoring the variables what have remained indeterminate.

On the other hand denoting by t_I the product of the variables with indices in I , we can factor the common subterm t_I out of some of the terms in f and write f as

$$f(x_1, \dots, x_n) = t_I f_{S(I)}(x_1, \dots, x_n) + r(x_1, \dots, x_n).$$

where each of the terms of $r(x_1, \dots, x_n)$ misses at least one of the variables with index in I . Note that, unlike the binary case, now $f_{S(I)}$ can contain variables with indices in I .

Now we can prove an analogue of Theorems 9 and 10. (A similar theorem appears in [1, Theorem 6], but both the statement and the proof are more complicated, involving a term $t = x_{i_1}^{r_1} \cdots x_{i_k}^{r_k}$ instead of $x_{i_1} \cdots x_{i_k}$ and consequently when factoring out t and writing $f = tf_{S(t)} + q$, having to treat separately the terms of q which contain some variables with indices in I , and the terms of q which do not.)

Theorem 11. *Let $f : GF(p^m)^n \rightarrow GF(p^m)$ be a polynomial function and I a subset of variable indices. Denote by \mathbf{v} the n -tuple having values of 1 in the positions with indices in I and indeterminates in the other positions, and by \mathbf{u} the n -tuple having values of 0 in the positions in I and indeterminates in the other positions. Then:*

$$f_I = (\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f)(\mathbf{u}) = f_{S(I)}(\mathbf{v})$$

Proof. The fact that $f_I = (\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f)(\mathbf{u})$ follows from Proposition 4 in the same way as in the proof of Theorem 10.

It suffices to show $(\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f)(\mathbf{u}) = f_{S(I)}(\mathbf{v})$ for the case when f is a monomial. The rest follows from the linearity of the operators, as $(f+g)_{S(I)} = f_{S(I)} + g_{S(I)}$ and Δ is a linear operator.

If f is a monomial not divisible by t_I , then both $f_{S(I)}$ and $\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f$ are identically zero, the latter using Proposition 5.

Now assume $f = t_I f_{S(I)}$ for some monomial $f_{S(I)}$. Like in the proof of [5, Theorem 1], we note that in the sum in the definition of f_I (or, equivalently in the sum given by Proposition 4 for $(\Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(k)} f)(\mathbf{u})$) only one term is non-zero; namely t_I evaluates to a non-zero value iff $x_{i_1} = \dots = x_{i_k} = 1$; hence $f_I = f_{S(I)}(\mathbf{v})$. \square

Remark 12. A cube attack based on Theorem 11 above would again search for sets I for which f_I is linear in the variables whose indices are not in I . If the total degree of f is d , the total degree of f_I can be, in the worst case, $d - k$. If $k = d - 1$ we can guarantee that f_I is linear or a constant. More generally, the closer k gets to $d - 1$ (while still having $k \leq d - 1$), the higher the chances of linearity.

However, unlike the binary case where $d \leq n$, now d can have any value up to $(p^m - 1)n$. Hence the (unknown) degree d of f could well be considerably higher than the number of variables n . In such a case, $k \leq n - 1$ is considerably lower than $d - 1$, and the chances of linearity are very small. In other words, since we differentiate at most once w.r.t. each variable, so a total of at most $n - 1$ times, the degree decreases by around $n - 1$ in general, and the resulting function can still have quite a high degree. Therefore, while a cube attack based on this result would be correct, it would have extremely low chances of success.

Our proposed generalisation of this attack would increase these chances by differentiating several times with respect to each variable. This will result in a greater decrease of the degree, thus improving the chances of reaching a linear result.

4 Generalisations to GF(p)

4.1 Differentiation in GF(p)

Differentiation with respect to a variable decreases the degree in that variable by at least one. In the binary case, the degree of a polynomial function in each variable is at most one, so we can only differentiate once w.r.t. each variable; a second differentiation will trivially produce the zero function. In GF(p) the degree in each variable is up to $p - 1$. We can therefore consider differentiating several times (and possibly using different difference steps) with respect to each variable. We first show that a monomial of degree d_i in a variable x_i can be differentiated m_i times w.r.t. x_i , for any $m_i \leq d_i$ (and using any collection of non-zero difference steps) and the degree decreases by exactly m_i . Hence we can differentiate d_i times without the result becoming identically zero.

Theorem 13. *Let $m_1 \leq d_1 \leq p - 1$ and $h_1, \dots, h_{m_1} \in \text{GF}(p) \setminus \{0\}$. Then*

$$\begin{aligned} \Delta_{h_1 \mathbf{e}_1, \dots, h_{m_1} \mathbf{e}_1}^{(m_1)} x_1^{d_1} &= \\ &= \sum_{j=m_1}^{d_1} \left(\sum_{\substack{(i_1, \dots, i_{m_1}) \in \{1, 2, \dots, j-m_1+1\}^{m_1} \\ i_1 + \dots + i_{m_1} = j}} \binom{d_1}{i_1, \dots, i_{m_1}, d_1 - j} h_1^{i_1} \cdots h_{m_1}^{i_{m_1}} \right) x_1^{d_1 - j} \end{aligned}$$

In the expression above the coefficient of $x_1^{d_1 - m_1}$ equals

$$\binom{d_1}{1, 1, \dots, 1, d_1 - m_1} h_1 \cdots h_{m_1} = \frac{d_1!}{(d_1 - m_1)!} h_1 \cdots h_{m_1} \neq 0$$

hence the degree in x_1 is exactly $d_1 - m_1$. For the particular case of $h_1 = \dots = h_{m_1} = 1$, the leading coefficient becomes

$$\binom{d_1}{1, 1, \dots, 1, d_1 - m_1} = \frac{d_1!}{(d_1 - m_1)!}$$

and the free term becomes

$$\sum_{\substack{(i_1, \dots, i_{m_1}) \in \{1, 2, \dots, d_1 - m_1 + 1\}^{m_1} \\ i_1 + \dots + i_{m_1} = d_1}} \binom{d_1}{i_1, \dots, i_{m_1}, 0}. \quad (1)$$

If $h_1 = \dots = h_{m_1} = 1$ and moreover $d_1 = m_1$ we have

$$\Delta_{\mathbf{e}_1, \dots, \mathbf{e}_1}^{(d_1)} x_1^{d_1} = \binom{d_1}{1, 1, \dots, 1, 0} \bmod p = d_1! \bmod p. \quad (2)$$

Proof. Use Theorem 8. Since $d_1 < p$, we have that $\frac{d_1!}{(d_1 - m_1)!}$ is not divisible by p . \square

Example 14. Let $f(x_1, x_2, x_3, x_4) = x_1^5x_2 + x_1^4x_3x_4 + x_1^6$ be a polynomial with coefficients in GF(31). We choose the variable x_1 and differentiate repeatedly w.r.t. x_1 , always with difference step equal to one. Differentiating once w.r.t. x_1 we obtain:

$$\begin{aligned}\Delta_{\mathbf{e}_1} f(x_1, x_2, x_3, x_4) &= \\ x_1^4(5x_2) + x_1^3(10x_2 + 4x_3x_4) + x_1^2(10x_2 + 6x_3x_4) + x_1(5x_2 + 4x_3x_4) + x_2 + x_3x_4\end{aligned}$$

Differentiating again w.r.t. x_1 we obtain:

$$\begin{aligned}\Delta_{\mathbf{e}_1, \mathbf{e}_1}^{(2)} f(x_1, x_2, x_3, x_4) &= \\ x_1^3(20x_2) + x_1^2(29x_2 + 12x_3x_4) + x_1(8x_2 + 24x_3x_4) + 30x_2 + 14x_3x_4\end{aligned}$$

Finally, if we differentiate a total of 5 times we obtain:

$$\Delta_{\mathbf{e}_1, \dots, \mathbf{e}_1}^{(5)} f(x_1, x_2, x_3, x_4) = 5!x_2 = 27x_2$$

as expected by (2).

For the remainder of this section, for simplicity we will always choose all the difference steps h_i equal to one. The case of arbitrary h_i can be treated similarly, but the formulae become more cumbersome. For convenience we will introduce some more notation. We pick a subset of k variable indices $I = \{i_1, \dots, i_k\}$ and we also pick multiplicities for each variable, m_1, \dots, m_k . Denote by t the term $t = x_{i_1}^{m_1} \cdots x_{i_k}^{m_k}$. We will apply the finite difference operator m_1 times w.r.t. the variable x_{i_1} , and m_2 times w.r.t. the variable x_{i_2} etc. always with difference step equal to one. More precisely we define:

$$f_t(x_1, \dots, x_n) = \Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_1}}^{(m_1)} \cdots \Delta_{\mathbf{e}_{i_k}, \dots, \mathbf{e}_{i_k}}^{(m_k)} f(x_1, \dots, x_n)$$

We now generalise Theorem 13 to the case when we differentiate w.r.t. several variables.

Theorem 15. Let $k \leq n$ and let m_1, \dots, m_k and d_1, \dots, d_k be integers such that $1 \leq m_\ell \leq d_\ell \leq p-1$ for $\ell = 1, \dots, k$. Let $\{i_1, \dots, i_k\} \subseteq \{1, \dots, n\}$ and let $f : \text{GF}(p)^n \rightarrow \text{GF}(p)$, $f(x_1, \dots, x_n) = x_{i_1}^{d_1} \cdots x_{i_k}^{d_k}$ and $t = x_{i_1}^{m_1} \cdots x_{i_k}^{m_k}$. We have

$$f_t = \sum_{j_1=m_1}^{d_1} \cdots \sum_{j_k=m_k}^{d_k} D(d_1, j_1, m_1) \cdots D(d_k, j_k, m_k) x_{i_1}^{d_1-j_1} \cdots x_{i_k}^{d_k-j_k}$$

where for any $1 \leq m \leq j \leq d$ we define

$$D(d, j, m) = \sum_{\substack{(i_1, \dots, i_m) \in \{1, 2, \dots, j-m+1\}^m \\ i_1 + \dots + i_m = j}} \binom{d}{i_1, \dots, i_m, d-j}.$$

The coefficient of $x_{i_1}^{d_1-m_1} \cdots x_{i_k}^{d_k-m_k}$ in f_t is equal to $\prod_{\ell=1}^k \frac{d_\ell!}{(d_\ell - m_\ell)!} \neq 0$ and the free term is equal to $\prod_{\ell=1}^k D(d_\ell, d_\ell, m_\ell)$. The total degree of f_t is $\sum_{\ell=1}^k (d_\ell - m_\ell)$.

For the particular case of $m_1 = d_1, \dots, m_k = d_k$, we have

$$f_t = d_1! \cdots d_k!.$$

Proof. Induction on k , applying Theorem 13. \square

When all the difference steps h_i are equal to one, the evaluation of the finite difference for a “black box” function f using Proposition 4 becomes simpler. We treat first the case when we differentiate w.r.t. a single variable:

Proposition 16. Let $f : R^n \rightarrow R$. Then

$$f_{x_1^{m_1}}(x_1, \dots, x_n) = \Delta_{\mathbf{e}_1, \dots, \mathbf{e}_1}^{(m_1)} f(x_1, \dots, x_n) = \sum_{i=0}^{m_1} (-1)^{m_1-i} \binom{m_1}{i} f(x_1 + i, x_2, \dots, x_n). \quad (3)$$

If R has characteristic p and $m_1 < p$ then all the coefficients $\binom{m_1}{i}$ in the sum above are non-zero. If f is a “black box” function, evaluating $f_{x_1^{m_1}}$ at one point in its domain requires $m_1 + 1$ evaluations of f .

Proof. The formula follows from Proposition 4. Since $0 \leq i \leq m_1 < p$ and p is prime, $\binom{m_1}{i}$ cannot be divisible by p , so it is non-zero in a field of characteristic p . \square

We now look at the situation where we differentiate w.r.t. several variables x_{i_1}, \dots, x_{i_k} .

Proposition 17. Let $f : R^n \rightarrow R$ and $t = \prod_{\ell=1}^k x_{i_\ell}^{m_\ell}$. Then

$$f_t(\mathbf{x}) = \sum_{j_1=0}^{m_1} \dots \sum_{j_k=0}^{m_k} (-1)^{\sum_{\ell=1}^k (m_\ell - j_\ell)} \binom{m_1}{j_1} \dots \binom{m_k}{j_k} f(\dots, x_{i_1} + j_1, \dots, x_{i_k} + j_k, \dots).$$

If R has characteristic p and all $m_\ell < p$, then all the coefficients $\binom{m_1}{j_1} \dots \binom{m_k}{j_k}$ in the sum above are non-zero. If f is a “black box” function, one evaluation of f_t needs $\prod_{\ell=1}^k (m_\ell + 1)$ evaluations of f . In particular evaluating f_t for $x_{i_\ell} = 0$, $\ell = 1, \dots, k$ we obtain:

$$\sum_{j_1=0}^{m_1} \dots \sum_{j_k=0}^{m_k} (-1)^{\sum_{\ell=1}^k (m_\ell - j_\ell)} \binom{m_1}{j_1} \dots \binom{m_k}{j_k} f(\dots, j_1, \dots, j_k, \dots) \quad (4)$$

with j_1, \dots, j_k in positions i_1, \dots, i_k respectively.

We note that in terms of the time complexity of evaluating f_t , it is now not only the total degree of t that matters (as in the binary case), but also the exponents of each variable. For a given number m of differentiations (i.e. t of total degree m), the smallest time complexity is achieved when t contains only one variable, i.e. $t = x_{i_1}^m$. Among all t of total degree m that contain k variables, the best time complexity is achieved when t has degree one in each but one of its variables, e.g. $t = x_{i_1}^{m-k+1} x_{i_2} \dots x_{i_k}$.

Remark 18. We saw that in the binary case, differentiating once w.r.t. each of the variables x_{i_1}, \dots, x_{i_k} is equivalent to summing f evaluated over a “cube” consisting of all the 0/1 combinations for the variables x_{i_1}, \dots, x_{i_k} . According to Proposition 17, the analogue of the “cube” will now be a k -dimensional grid/mesh with sides of “length” m_1, m_2, \dots, m_k . Namely each variable x_{i_ℓ} w.r.t. which we differentiate will have increments of $0, 1, \dots, m_\ell$ and each term in the sum has alternating signs as well as being multiplied by binomial coefficients.

4.2 Fundamental theorem of the cube attack generalised to GF(p)

We will use the notation $f_t = \Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(m_1)} \dots \Delta_{\mathbf{e}_{i_k}, \dots, \mathbf{e}_{i_k}}^{(m_k)} f$ with $t = x_{i_1}^{m_1} \dots x_{i_k}^{m_k}$ as in the previous section. Factoring out t , we can write f as

$$f(x_1, \dots, x_n) = t f_{S(t)}(x_1, \dots, x_n) + r(x_1, \dots, x_n)$$

where $f_{S(t)}$ and r are uniquely determined such as none of the terms in r is divisible by t .

We can already give a bound on the degree of f_t :

Proposition 19. With the notations above, we have $\deg(f_t) \leq \deg(f_{S(t)})$. In particular, if $\deg(t) = \deg(f) - 1$ then f_t is linear or constant.

Proof. When f is a monomial divisible by t , we have $\deg(f_t) = \deg(f_{S(t)})$ by Theorem 15. If f is a monomial not divisible by t , then $f_t = 0$. Finally, for a general f we use the linearity of the Δ operator, the fact that $(f+g)_{S(t)} = f_{S(t)} + g_{S(t)}$ and the fact that $\deg(f+g) \leq \deg(f) + \deg(g)$, for any polynomials f and g . \square

The result above is already sufficient for a cube attack. However, we will give a more refined result shortly, in order to give an analogue of the main theorem of the classical cube attack (see Theorems 9, 10 and 11). At first sight we might expect Theorem 11 to hold here too, namely we might expect that $f_t(x_1, \dots, x_n)$ evaluated at $x_{i_j} = 0$ for $j = 1, \dots, k$ equals $f_{S(t)}(x_1, \dots, x_n)$ evaluated at $x_{i_j} = 1$ for $j = 1, \dots, k$. However this is not true in general, as the following counterexample shows:

Example 20. We continue Example 14 for $f(x_1, x_2, x_3, x_4) = x_1^5 x_2 + x_1^4 x_3 x_4 + x_4^6$.

We computed $f_{x_1}(x_1, x_2, x_3, x_4)$ in Example 14. Evaluating at $x_1 = 0$ we obtain $f_{x_1}(0, x_2, x_3, x_4) = x_2 + x_3 x_4$. On the other hand $f_{S(x_1)}(x_1, x_2, x_3, x_4) = x_1^4 x_2 + x_1^3 x_3 x_4$. Evaluating at $x_1 = 1$ we obtain $f_{S(x_1)}(1, x_2, x_3, x_4) = x_2 + x_3 x_4$. Hence we verified that $f_{x_1}(0, x_2, x_3, x_4) = f_{S(x_1)}(1, x_2, x_3, x_4)$ as expected by Theorem 11.

Differentiating again w.r.t. x_1 and evaluating $f_{x_1^2}(x_1, x_2, x_3, x_4)$ at $x_1 = 0$ gives $f_{x_1^2}(0, x_2, x_3, x_4) = 30x_2 + 14x_3 x_4$. On the other hand $f_{S(x_1^2)}(x_1, x_2, x_3, x_4) = x_1^3 x_2 + x_1^2 x_3 x_4$, which evaluated at $x_1 = 1$ gives $f_{S(x_1^2)}(1, x_2, x_3, x_4) = x_2 + x_3 x_4$.

Hence $f_{x_1^2}(0, x_2, x_3, x_4) \neq f_{S(x_1^2)}(1, x_2, x_3, x_4)$, so Theorem 11 cannot be extended in its current form to the case when we differentiate more than once w.r.t. one variable. However, note that the two quantities computed here do contain the same monomials.

Finally, if we differentiate 5 times with respect to x_1 we obtain: $f_{x_1^5}(x_1, x_2, x_3, x_4) = 27x_2$, whereas $f_{S(x_1^5)}(x_1, x_2, x_3, x_4) = x_2$ so again the two polynomials do not coincide; however they only differ by multiplication by a constant.

The correct generalisation of the main theorem of the classical cube attack is the following:

Theorem 21. Let $f : \text{GF}(p)^n \rightarrow \text{GF}(p)$ be a polynomial function and $t = \prod_{j=1}^k x_{i_j}^{m_j}$. Denote

$$f_t(x_1, \dots, x_n) = \Delta_{\mathbf{e}_{i_1}, \dots, \mathbf{e}_{i_k}}^{(m_1)} \dots \Delta_{\mathbf{e}_{i_k}, \dots, \mathbf{e}_{i_k}}^{(m_k)} f(x_1, \dots, x_n).$$

Write f as

$$f(\mathbf{x}) = t f_{S(t)}(\mathbf{x}) + r(\mathbf{x})$$

so that none of the monomials in r are divisible by t .

Denote by \mathbf{v} the n -tuple having values of 1 in the positions i_1, \dots, i_k and indeterminates elsewhere, and by \mathbf{u} the n -tuple having values of 0 in the positions i_1, \dots, i_k and indeterminates elsewhere.

Write $f_{S(t)} = t_1 g_1 + \dots + t_u g_u$, where g_i are polynomials that do not depend on any of the variables x_{i_1}, \dots, x_{i_k} and t_1, \dots, t_u are all the distinct terms in the variables x_{i_1}, \dots, x_{i_k} that appear in $f_{S(t)}$.

Then there are constants $c_1, \dots, c_u \in \text{GF}(p)$ such that $f_t(\mathbf{u})$ equals $c_1 g_1 + \dots + c_u g_u$ (which can also be viewed as $c_1 t_1 g_1 + \dots + c_u t_u g_u$ evaluated at \mathbf{v}). The exact values for the constants c_i can be determined as follows: if $t_i = x_{i_1}^{\ell_1} \dots x_{i_k}^{\ell_k}$, then $c_i = D(m_1 + \ell_1, m_1 + \ell_1, m_1) \dots D(m_k + \ell_k, m_k + \ell_k, m_k)$ where $D()$ is defined in Theorem 15.

In particular if $f_{S(t)}$ does not depend on any of the variables x_{i_1}, \dots, x_{i_k} then

$$f_t(\mathbf{x}) = m_1! \dots m_k! f_{S(t)}.$$

Proof. We first use Theorem 15 for individual monomials and then the linearity of the Δ operator. \square

Again, for the cube attack we are interested in the cases where $f_t(\mathbf{u})$ is linear:

Corollary 22. With the notations of Theorem 21,

$$\deg(f_t(\mathbf{u})) \leq \deg(f_{S(t)}(\mathbf{v})).$$

The latter is also equal to the total degree of $f_{S(t)}(\mathbf{x})$ in the variables $\{x_1, \dots, x_n\} \setminus \{x_{i_1}, \dots, x_{i_k}\}$. If $\deg(f_{S(t)}(\mathbf{v})) = 1$ then $f_t(\mathbf{u})$ is linear or constant.

4.3 Proposed Algorithm for the cube attack in $\text{GF}(p)$

In this section we give more details of the algorithm, drawing on the results from previous sections. The main idea of our proposed attack is that when the degree in one variable is higher than one, we can differentiate w.r.t. that variable repeatedly, unlike the cube attacks described in the Section 3, which use differentiation at most once for each variable.

We are given a cryptographic “black box” function $f(v_1, \dots, v_m, x_1, \dots, x_n)$ with v_i being public variables and x_i being secret variables.

Preprocessing phase

1. Choose a term in the public variables, $t = v_{i_1}^{m_1} \dots v_{i_k}^{m_k}$, with $1 \leq m_i \leq p - 1$.
2. Using formula (4) in Proposition 17 we evaluate $f_t(\mathbf{0}, \mathbf{x})$ for several choices of the secret variables \mathbf{x} , in order to decide whether, with reasonably high probability, the total degree of $f_t(\mathbf{0}, \mathbf{x})$ in \mathbf{x} equals one. (For this one can use the textbook definition of linearity; namely, for various values of $a, b \in \text{GF}(p)$ and $\mathbf{y}, \mathbf{z} \in \text{GF}(p)^n$ test whether $a(f_t(\mathbf{0}, \mathbf{y}) - f_t(\mathbf{0}, \mathbf{0})) + b(f_t(\mathbf{0}, \mathbf{z}) - f_t(\mathbf{0}, \mathbf{0})) = f_t(\mathbf{0}, a\mathbf{y} + b\mathbf{z}) - f_t(\mathbf{0}, \mathbf{0})$; in $\text{GF}(p)$ with p large, we will need in general much fewer linearity tests than in the binary case, see [10]; one can at the same time check whether $f_t(\mathbf{0}, \mathbf{x})$ is non-constant).
3. If the decision above is “yes”, we determine $f_t(\mathbf{0}, \mathbf{x})$ explicitly, as $f_t(\mathbf{0}, \mathbf{x}) = c_0 + \sum_{i=1}^n c_i x_i$ where $c_0 = f_t(\mathbf{0}, \mathbf{0})$ and $c_i = f_t(\mathbf{0}, \mathbf{e}_i) - c_0$; we store $(t, c_0, c_1, \dots, c_n)$.
4. Repeat the steps above for different values of t until one obtains n linearly independent stored tuples (c_1, \dots, c_n) , or until one runs out of time/memory.

For the heuristic of choosing t one could take into account the computational cost for a term t , see Proposition 17 and the comment following it. However a full heuristic is beyond the scope of this paper. A number of optimisations have been proposed for the binary cube attack; many of them can be transferred to the modulo p case, but again, this is beyond the scope of this paper.

Online phase

1. For each $(t, c_0, c_1, \dots, c_n)$ stored in the preprocessing phase, compute $f_t(\mathbf{0}, \mathbf{x})$ (with \mathbf{x} being now unknown) using formula (4) in Proposition 17. Form the linear equation: $c_1x_1 + \dots + c_nx_n + c_0 = f_t(\mathbf{0}, \mathbf{x})$.
2. Solve the system of linear equations thus obtained, determining the secret variables x_1, \dots, x_n . If the preprocessing phase only produced $s < n$ equations, then not all the secret variables can be determined, we would need to do an exhaustive search for $n - s$ of them.

Remark 23. Let ℓ be the length of the binary representation of p . We can view each bit of an element in $\text{GF}(p)$ as one binary variable. If f is a function of n variables over $\text{GF}(p)$, we can also view it as ℓ binary functions in ℓn binary variables. We could therefore apply the classical (binary) cube attack on these functions. A rough estimate suggests that the running time for corresponding cubes will be approximately the same. (Differentiating $p - 1$ times with respect to one variable x_i in $\text{GF}(p)$ takes p evaluations of f ; differentiating once w.r.t. each of the binary variables that are components of x_i will take 2^ℓ evaluations of f ; we have $p \approx 2^\ell$.) The chances of success on a particular cube bear no easy relationship between the two approaches, because the degree of f and the degrees of the ℓ binary functions are not related in a simple way.

Hence we would argue that in general one cannot tell which of the attacks will work better, so one should try both. If the cipher has a structure that would suggest that the degree as polynomial over $\text{GF}(p)$ is relatively low, then a cube attack over $\text{GF}(p)$ should certainly be an approach to consider.

5 Generalisations to $\text{GF}(p^m)$

In this section we take our generalisation further, to arbitrary finite fields $\text{GF}(p^m)$. An important particular case would be $\text{GF}(2^m)$, as many cryptographic algorithms include operations over a field of this type.

5.1 Preliminaries

We need some known results regarding the values of binomial coefficients and multinomial coefficients in fields of finite characteristic.

Theorem 24. (*Kummer's Theorem*, [13, p. 115]) *Let $n \geq k \geq 0$ be integers and p a prime. Let j be the highest exponent for which $\binom{n}{k}$ is divisible by p^j . Then j equals the sum of carries when adding k and $n - k$ as numbers written in base p .*

Kummer's theorem has been generalised to multinomials by various authors (see for example [6] and citations therein).

Theorem 25. *Let d, k_1, k_2, \dots, k_s be integers such that $\sum_{i=1}^s k_i = d$ and $k_i \geq 0$ and let p be a prime. Let j be the highest exponent for which $\binom{d}{k_1, k_2, \dots, k_s}$ is divisible by p^j . Then j equals the sum of all the carries when adding all of k_1, k_2, \dots, k_s as numbers written in base p .*

We will be interested in the situations where the multinomial coefficients are not zero modulo p .

Corollary 26. Let p be a prime. The following are equivalent:

- (i) The multinomial coefficient $\binom{d}{k_1, k_2, \dots, k_s}$ is not zero modulo p .
- (ii) There are no carries when adding k_1, k_2, \dots, k_s as numbers written in base p .
- (iii) In base p , each digit of n equals to the sum of the digits of k_1, k_2, \dots, k_s in the corresponding position.

5.2 Differentiation in $\text{GF}(p^m)$

When moving from $\text{GF}(p)$ to $\text{GF}(p^m)$ several things work differently. For a start, differentiating once w.r.t. a variable x may decrease the degree in x by more than one, regardless of the difference step. For example let us differentiate x^d once. In $(x+h)^d - x^d$ the coefficient of x^{d-1} is dh , so when d is a multiple of p the degree is strictly less than $d-1$. To examine the general case we will use Theorem 8, so we introduce for convenience the following notation:

$$C_p(d, j, k) = \{(i_1, \dots, i_k) \mid 1 \leq i_j \leq d, i_1 + \dots + i_k = j, \text{ and } \binom{d}{i_1, \dots, i_k, n-j} \not\equiv 0 \pmod{p}\}.$$

Note that Corollary 26 gives a useful characterisation of this set. We have:

Theorem 27. Let $f : \text{GF}(p^m) \rightarrow \text{GF}(p^m)$, $f(x) = x^d$, $d < p^m$. Let $0 < k \leq m(p-1)$ and let $h_1, \dots, h_k \in \text{GF}(p^m) \setminus \{0\}$. The degree of $\Delta_{h_1 e_1, \dots, h_k e_1}^{(k)} x^d$ is less than or equal to the integer d' computed as follows: write d in base p as $d = d_u d_{u-1} \dots d_1 d_0$; let i be the highest integer for which $d_0 + d_1 + \dots + d_i \leq k$; define $d'_{i+1} = d_{i+1} - (k - (d_0 + d_1 + \dots + d_i))$; finally define d' as the number written in base p as $d' = d_u d_{u-1} \dots d_{i+2} d'_{i+1} 0 \dots 0$ (with $i+1$ zeroes at the end).

In particular, for $p=2$, the binary representation of the degree d' is obtained from the binary representation of d by replacing k of its ones by zeroes, starting from the least significant digit.

Proof. By Theorem 8 the degree d' will be less than or equal to $d-j$ where j is minimal such that $C_p(d, j, k) \neq \emptyset$. Using Corollary 26(iii) we see that the minimum value for j for given d and k is achieved by choosing $i_1, \dots, i_k \geq 1$ as small as possible while maintaining $\binom{d}{i_1, \dots, i_k, d-j}$ not equal to zero modulo p . This is achieved by choosing i_1, \dots, i_k as follows: d_0 of them will be equal to 1, d_1 will be equal to p (i.e. 10 in base p), d_2 will be equal to p^2 (i.e. 100 in base p), ..., d_i of them will be equal to p^i and finally $k - (d_0 + d_1 + \dots + d_i)$ will be equal to p^{i+1} . It can be verified that $d' = d - (i_1 + \dots + i_k)$ will then have the form described in the theorem statement. \square

Note that the sum of the digits of a number in base p plays an important role here. For any non-negative integer a we will introduce the notation $S_p(a)$ as being the sum of the digits in the base p representation of a . We define the digit-sum degree of a univariate polynomial f in a variable x_i as being $\max\{S_p(j) \mid c_j \neq 0\}$ where $f = \sum_{j=0}^d c_j x_i^j$ with c_j polynomials in the remaining variables. The previous theorem implies:

Corollary 28. Let f be a polynomial function and let s be the digit-sum degree of f in x_i . Then differentiating f a total of s times w.r.t. x_i will always produce a polynomial function which does not depend on x_i (possibly the identically zero function).

Corollary 29. Any function $f : \text{GF}(p^m) \rightarrow \text{GF}(p^m)$ can be differentiated at most $m(p-1)$ times w.r.t. a given variable before the result becomes the identically zero function.

Is the bound in Corollary 28 tight, in the sense that there are functions which are non-zero after a number of differentiations equal to their digit-sum degree? In particular, are there functions which are still non-zero after $m(p-1)$ differentiations? We will show that this indeed the case, but only if we choose the h_i carefully. First let us illustrate a choice of the steps h_i which we need to avoid. By Proposition 16, if we differentiate p times with all steps equal to 1 the result is identically zero regardless of the original function f :

$$\begin{aligned}\Delta_{\mathbf{e}_1, \dots, \mathbf{e}_1}^{(p)} f(x_1, \dots, x_n) &= \sum_{i=0}^p (-1)^{p-i} \binom{p}{i} f(x_1 + i, x_2, \dots, x_n) \\ &= f(x_1 + p, x_2, \dots, x_n) - f(x_1, x_2, \dots, x_n) \\ &= 0.\end{aligned}$$

because all the coefficients $\binom{p}{i}$ for $0 < i < p$ are divisible by p .

Denote by b_0, \dots, b_{m-1} a basis of $\text{GF}(p^m)$ viewed as a m -dimensional vector space over $\text{GF}(p)$. We choose the sequence $h_1, \dots, h_{(p-1)m}$ as follows: $p-1$ values of b_0 , followed by $p-1$ values of b_1 etc.

As in Section 4, we pick a set of variables and their multiplicities, defining the term $t = x_{i_1}^{m_1} \cdots x_{i_k}^{m_k}$. For a polynomial function f in n variables, we now define:

$$f_t(x_1, \dots, x_n) = \Delta_{h_1 \mathbf{e}_{i_1}, \dots, h_{m_1} \mathbf{e}_{i_1}}^{(m_1)} \cdots \Delta_{h_k \mathbf{e}_{i_k}, \dots, h_{m_k} \mathbf{e}_{i_k}}^{(m_k)} f(x_1, \dots, x_n)$$

where the sequence $h_1, \dots, h_{m(p-1)}$ has been fixed as above. We will concentrate on differentiating several times w.r.t. one variable, x_1 .

For these choices of h_i Proposition 4 becomes:

Proposition 30. Let $f : \text{GF}(p^m)^n \rightarrow \text{GF}(p^m)$ be a polynomial function and let $t = x_1^{m_1}$. Write $m_1 = q(p-1) + r$ with $0 < r \leq p-1$ (note the slight difference from the usual quotient and remainder, as here the remainder can be $p-1$ but cannot be 0). Then

$$f_t(\mathbf{x}) = \sum_{a_0, \dots, a_q} (-1)^{m_1 - \sum_{i=0}^q a_i} \binom{p-1}{a_0} \cdots \binom{p-1}{a_{q-1}} \binom{r}{a_q} f(x_1 + a_0 b_0 + \dots + a_q b_q, x_2, \dots, x_n)$$

where the sum is over all tuples $(a_0, \dots, a_q) \in \{0, \dots, p-1\}^q \times \{0, \dots, r\}$.

All the coefficients in the sum above are non-zero. If f is a “black box” function, one evaluation of f_t needs $p^q(r+1)$ evaluations of f .

Proof. Similar to the proof of Proposition 17. □

We show next that our choice of h_i is indeed a valid choice in the sense that there are functions which can be differentiated $m(p-1)$ times w.r.t. the same variable without becoming zero.

Proposition 31. For each $t = x_1^{m_1}$ with $0 \leq m_1 \leq m(p-1)$ there is at least a function $f : \text{GF}(p^m)^n \rightarrow \text{GF}(p^m)$ such that $f_t(\mathbf{x})$ is not the identical zero function. Moreover there is at least a polynomial function f with digit-sum degree in x_1 equal to m_1 such that $f_t(\mathbf{x})$ is a non-zero constant function.

Proof. We will construct a polynomial function f in one variable, x_1 . Write $m_1 = q(p-1) + r$ with $0 < r \leq p-1$ as in Proposition 30. In the formula in Proposition 30 all the terms in the sum have

a non-zero coefficient, and have distinct arguments for f . We will prescribe the values of f at these evaluation points and then interpolate f . Namely we will prescribe $f(a_0 b_0 + \dots + a_q b_q) = 0$ for all $(a_0, \dots, a_q) \in \{0, \dots, p-1\}^q \times \{0, \dots, r\}$ except $a_0 = \dots = a_q = 0$, where $f(0) \neq 0$. The polynomial f is interpolated as

$$f(x_1) = \prod (x_1 - (a_0 b_0 + \dots + a_q b_q))$$

where the product is over $(a_0, \dots, a_q) \in \{0, \dots, p-1\}^q \times \{0, \dots, r\}$ except $a_0 = \dots = a_q = 0$. It can be easily seen that $\deg_{x_1}(f) = (r+1)p^q - 1$, so the representation of its degree in base p consists of a digit of r followed by q digits of $p-1$. Hence $S_p(\deg_{x_1}(f)) = q(p-1) + r = m_1$.

On the other hand, we know from Theorem 27 that f_t is a constant (possibly zero). However $f_t(0) = f(0) \neq 0$ by Proposition 30 and our choice of interpolation points. Hence f_t is a non-zero constant. \square

Note that there are other possible valid choices of h_i , but we aimed to keep things simple computationally by using this particular choice.

Example 32. Consider $f(x_1) = x_1^5 \in \text{GF}(9)[x_1]$. We define $h_1 = h_2 = 1$ and $h_3 = h_4 = \alpha$, where α is a primitive element of $\text{GF}(9)$. We compute the third order differential $\Delta_{\mathbf{e}_1, \mathbf{e}_1, \alpha \mathbf{e}_1}^{(3)} f(x_1)$ using either Proposition 4 or Theorem 8 and obtain:

$$\Delta_{\mathbf{e}_1, \mathbf{e}_1, \alpha \mathbf{e}_1}^{(3)} f(x_1) = 2\alpha^3 + \alpha = 2\alpha(\alpha + 1)(\alpha + 2) \neq 0.$$

Generalising the results of this section to differentiation w.r.t. several variables is not difficult but the notation becomes cumbersome. Moreover, we will see in the next subsection that such a generalisation is not very useful for a practical attack, so we leave it as an exercise to the reader.

5.3 Reducing differentiation over $\text{GF}(p^m)$ to differentiation over $\text{GF}(p)$

Fix a basis $b_0, \dots, b_{m-1} \in \text{GF}(p^m)$ for $\text{GF}(p^m)$ viewed as an m -dimensional vector space over $\text{GF}(p)$. Any element $a \in \text{GF}(p^m)$ can be uniquely written as $a = a_0 b_0 + \dots + a_{m-1} b_{m-1}$ with $a_i \in \text{GF}(p)$. Denote by $\varphi : \text{GF}(p^m) \rightarrow \text{GF}(p)^m$ the vector space isomorphism defined by $\varphi(a) = (a_0, \dots, a_{m-1})$; this can be naturally extended to $\varphi : \text{GF}(p^m)^n \rightarrow \text{GF}(p)^{nm}$; denote by $\pi_j : \text{GF}(p^m) \rightarrow \text{GF}(p)$ the m the projection homomorphisms defined as $\pi_j(a) = a_j$.

Let $f : \text{GF}(p^m)^n \rightarrow \text{GF}(p^m)$ be a polynomial function in n variables x_1, \dots, x_n . By writing $x_i = x_{i0} b_0 + \dots + x_{i,m-1} b_{m-1}$ the function f can be alternatively viewed as a function $\bar{f} : \text{GF}(p)^{nm} \rightarrow f : \text{GF}(p)^m$ defined by $\bar{f} = \varphi^{-1} \circ f \circ \varphi$:

$$\begin{array}{ccc} \text{GF}(p^m)^n & \xrightarrow{f} & \text{GF}(p^m) \\ \varphi \downarrow & & \downarrow \varphi \\ \text{GF}(p)^{nm} & \xrightarrow{\bar{f}} & \text{GF}(p)^m \end{array}$$

Alternatively we can view f as m polynomial (projection) functions $\bar{f}_0, \dots, \bar{f}_{m-1} : \text{GF}(p)^{nm} \rightarrow \text{GF}(p)$, defined by $\bar{f}_i = \varphi^{-1} \circ f \circ \pi_i$ each in nm variables x_{ij} with $i = 1, \dots, n$ and $j = 0, \dots, m-1$.

Proposition 33. With the notations above, the total degree of \bar{f}_j in the variables $x_{i0}, \dots, x_{i,m-1}$ is at most the digit-sum degree of f in x_i .

Proof. It suffices to prove the statement for $f = x_1^d$. We do it by induction on the number of digits in the representation of d_1 in base p . For one digit, i.e. $d = 1, 2, \dots, p - 1$ it is trivially satisfied. For the induction step, write $d = pd' + d_0$, with $0 \leq d_0 < p$. Let $A = x_{10}b_0 + x_{11}b_1 + \dots + x_{1,m-1}b_{m-1}$. We have:

$$A^{pd'+d_0} = (A^{d'})^p A^{d_0}$$

In $\text{GF}(p)$ we have $x^p = x$, so computing the power p of any polynomial function over $\text{GF}(p)$ does not change the degree in each variable. Hence the degree in $x_{10}, \dots, x_{1,m-1}$ of A^d equals at most the degree of A^{d_0} plus the degree of $A^{d'}$, so by the induction hypothesis, it is at most equal to d_0 plus the sum of the digits of d' ; this equals the sum of the digits of d . \square

Note that in the proposition above $d_i \leq p^m - 1$, so the sum of the digits of d_i is at most $(p - 1)m$. On the other hand, \bar{f}_j are polynomial functions over $\text{GF}(p)$, so they have degree at most $p - 1$ in each variable. Therefore their total degree in the variables $x_{i0}, \dots, x_{i,m-1}$ is at most $(p - 1)m$, so these facts tie in.

In the next theorem, since we are differentiating functions in a different number of variables, we will use for elementary vectors the notation $\mathbf{e}_i^{(n)}$ instead of \mathbf{e}_i to clarify the length n of the elementary vector. Note that here the difference steps are elements in a basis of $\text{GF}(p^m)$, but they do not need to be in the order prescribed in the discussion at the beginning of Section 5.2.

Theorem 34. *Let $f : \text{GF}(p^m)^n \rightarrow \text{GF}(p^m)$ be a polynomial function and let r_0, \dots, r_{m-1} with $0 \leq r_i \leq p - 1$.*

$$\overline{\Delta_{b_0 \mathbf{e}_1^{(n)}, \dots, b_m \mathbf{e}_1^{(n)}}^{(r_0)} \cdots \Delta_{b_{m-1} \mathbf{e}_1^{(n)}, \dots, b_{m-1} \mathbf{e}_1^{(n)}}^{(r_{m-1})} f} = \Delta_{\mathbf{e}_1^{(mn)}, \dots, \mathbf{e}_1^{(mn)}}^{(r_0)} \cdots \Delta_{\mathbf{e}_{m-1}^{(mn)}, \dots, \mathbf{e}_{m-1}^{(mn)}}^{(r_{m-1})} \bar{f}$$

the latter being a differentiation r_0 times w.r.t x_{10} , r_1 times w.r.t x_{11} , ..., r_{m-1} times w.r.t. $x_{1,m-1}$.

Proof. Induction on $\sum_{i=0}^{p-1} r_i$. For the base case, we can assume without loss of generality that $r_0 = 1$ and $r_i = 0$ for $i \geq 1$. We have:

$$\begin{aligned} & \Delta_{b_0 \mathbf{e}_1} f(x_1, \dots, x_n) \\ &= f(x_1 + b_0, \dots, x_n) - f(x_1, \dots, x_n) \\ &= f(x_{10}b_0 + \dots + x_{1,m-1}b_{m-1} + b_0, \dots, x_n) - f(x_{10}b_0 + \dots + x_{1,m-1}b_{m-1}, \dots, x_n) \\ &= f((x_{10} + 1)b_0 + \dots + x_{1,m-1}b_{m-1}, \dots, x_n) - f(x_{10}b_0 + \dots + x_{1,m-1}b_{m-1}, \dots, x_n) \end{aligned}$$

Hence

$$\begin{aligned} & \overline{\Delta_{b_0 \mathbf{e}_1} f(x_1, \dots, x_n)} \\ &= \bar{f}(x_{10} + 1, x_{11}, \dots, x_{n,m-1}) - \bar{f}(x_{10}, x_{11}, \dots, x_{n,m-1}) \\ &= \Delta_{\mathbf{e}_1^{(mn)}} \bar{f}(x_{10}, x_{11}, \dots, x_{n,m-1}) \end{aligned}$$

The inductive step is similar, using the fact that φ is an isomorphism and π_j are homomorphisms. \square

5.4 Cube attack in $\text{GF}(p^m)$

The fundamental result of the cube attack can also be generalised to $\text{GF}(p^m)$. We will formulate it for differentiation w.r.t. one variable.

Theorem 35. Let $f : \text{GF}(p^m)^n \rightarrow \text{GF}(p^m)$ be a polynomial function of degree d_1 in x_1 . Write

$$f(x_1, \dots, x_n) = \sum_{i=0}^{d_1} g_i(x_2, \dots, x_n) x_1^i$$

Let $m_1 \geq 1$ and define $t = x_1^{m_1}$. Let j_1, \dots, j_r be all those integers between 0 and d_1 such that $S_p(j_w) \leq m_1$. Then

$$f_t(0, x_2, \dots, x_n) = \sum_{w=1}^r c_{j_w} g_{j_w}(x_2, \dots, x_n),$$

where c_{j_1}, \dots, c_{j_r} are constants which only depend on h_1, \dots, h_{m_1} (and do not depend on x_1, \dots, x_n) given by:

$$c_{j_w} = \sum_{(i_1, \dots, i_{m_1}) \in C_p(j_w, j_w, m_1)} \binom{j_w}{i_1, \dots, i_{m_1}, 0} h_1^{i_1} \cdots h_{m_1}^{i_{m_1}} \quad (5)$$

Proof. We apply Theorem 8 and the linearity of the Δ operator. \square

Corollary 36. Let $f : \text{GF}(p^m)^n \rightarrow \text{GF}(p^m)$ be a polynomial function and $d_1 = \deg_{x_1}(f)$. Write f as:

$$f(x_1, \dots, x_n) = \sum_{i=0}^{d_1} g_i(x_2, \dots, x_n) x_1^i.$$

Let m_1 be the digit-sum degree of f in x_1 . Let j_1, \dots, j_r be those integers between 0 and d_1 for which $S_p(j_w) = m_1$ and $g_{j_w} \neq 0$. Then m_1 is the highest number of times that we can differentiate f w.r.t. x_1 before it becomes identically zero. Moreover, for $t = x_1^{m_1}$ we have

$$f_t(x_1, x_2, \dots, x_n) = \sum_{w=1}^r c_{j_w} g_{j_w}(x_2, \dots, x_n),$$

where c_{j_w} are as defined in (5).

Note that in the Corollary above f_t can be evaluated at any point, not necessarily at $x_1 = 0$ like in Theorem 35, because in this case f_t does not depend on x_1 .

Remark 37. A cube attack for a polynomial function f over $\text{GF}(p^m)$ can be developed based on Theorem 35 (generalised to several variables). However, by using Theorem 34, such an attack can be reduced to an attack in $\text{GF}(p)$ on the polynomial functions $\bar{f}_0, \dots, \bar{f}_{m-1}$ simultaneously. In the cube attack we are looking to differentiate f so that the result is linear in the secret variables. A polynomial function f is linear iff all the polynomial functions $\bar{f}_0, \dots, \bar{f}_{m-1}$ are linear. However, if we mount an attack in $\text{GF}(p)$ on $\bar{f}_0, \dots, \bar{f}_{m-1}$ individually (rather than an attack translated from the attack in $\text{GF}(p^m)$) there are chances that some of the $\bar{f}_0, \dots, \bar{f}_{m-1}$ are linear, even if not all of them are linear. This suggests that for functions f over $\text{GF}(p^m)$ an attack in $\text{GF}(p)$ on each component independently is more promising than an attack in $\text{GF}(p^m)$ on the whole function f . Therefore, in Section 4.3 we only described the attack over $\text{GF}(p)$.

6 Conclusion

We examined higher order differentiation over integers modulo a prime p , as well as over general finite fields of p^m elements, proving a number of results applicable to cryptographic attacks, and in particular generalising the fundamental theorem on which the cube attack is based.

Using these results we proposed a generalisation of the cube attack to functions over the integers modulo p ; the main difference to the binary case is that we can differentiate several times with respect to the same variable. Such an attack would be particularly suited to ciphers that use operations modulo p in their internal structure.

We also show that a further generalisation to general finite fields $\text{GF}(p^m)$ is possible, but not as promising as the generalisation to $\text{GF}(p)$, due to the fact that differentiation in $\text{GF}(p^m)$ can be reduced to differentiation in $\text{GF}(p)$.

References

- [1] Andrea Agnesse and Marco Pedicini. Cube attack in finite fields of higher order. In *Proceedings of the Ninth Australasian Information Security Conference - Volume 116*, AISC '11, pages 9–14, 2011.
- [2] Jean-Philippe Aumasson, Itai Dinur, Willi Meier, and Adi Shamir. Cube testers and key recovery attacks on reduced-round MD6 and Trivium. In Orr Dunkelman, editor, *Fast Software Encryption*, volume 5665 of *Lecture Notes in Computer Science*, pages 1–22. Springer Verlag, 2009.
- [3] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.
- [4] Joan Daemen, Rene Govaerts, and Joos Vandewalle. Block ciphers based on modular arithmetic. In *Proc. of the 3rd Symposium on the State and Progress of Research in Cryptography, W. Wolfowicz, Ed., Fondazione Ugo Bordoni*, pages 80–89, 1993.
- [5] Itai Dinur and Adi Shamir. Cube attacks on tweakable black box polynomials. In *EUROCRYPT*, pages 278–299, 2009.
- [6] Fred Dodd and Rhodes Peele. Some counting problems involving the multinomial expansion. *Mathematics Magazine*, 64(2):115–122, 1991.
- [7] Ming Duan and Xuejia Lai. Higher order differential cryptanalysis framework and its applications. In *International Conference on Information Science and Technology (ICIST)*, pages 291–297, 2011.
- [8] P. Ekdahl and T. Johansson. SNOW-a new stream cipher. In *Proceedings of the first NESSIE Workshop, Heverlee, Belgium*, 2000.
- [9] ETSI/SAGE. Specification of the 3GPP confidentiality and integrity algorithms 128-EEA3 & 128-EIA3. document 2: ZUC specification. Technical Report 1.6, ETSI, 2011.
- [10] T. Kaufman and D. Ron. Testing polynomials over general fields. In *Proceedings. 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 413 – 422, 2004.
- [11] S. Knellwolf and W. Meier. High order differential attacks on stream ciphers. *Cryptography and Communications*, 4(3-4):203–215, 2012.

- [12] L. R. Knudsen. Truncated and higher order differentials. In B. Preneel, editor, *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 196–211. Springer Verlag, 1995.
- [13] E. E. Kummer. Über die Ergänzungssätze zu den allgemeinen Reciprocitatsgesetzen. *Journal für die reine und angewandte Mathematik*, 1852(44), 1852.
- [14] Xuejia Lai. Higher order derivatives and differential cryptanalysis. In Richard E. Blahut, Daniel J. Costello, Jr., Ueli Maurer, and Thomas Mittelholzer, editors, *Communications and Cryptography*, volume 276 of *The Springer International Series in Engineering and Computer Science*, pages 227–233. Springer Verlag, 1994.
- [15] Xuejia Lai and James L. Massey. A proposal for a new block encryption standard. In *Advances in Cryptology-EUROCRYPT’90*, pages 389–404. Springer Verlag, 1990.
- [16] Xuejia Lai, James L. Massey, and S. Murphy. Markov ciphers and differential cryptanalysis. In *EUROCRYPT*, pages 17–38, 1991.
- [17] M. Vielhaber. Breaking ONE.FIVIUM by AIDA an algebraic IV differential attack. Cryptology ePrint Archive, Report 2007/413, 2007. url`http://eprint.iacr.org/`.