Appl. Math. and Comp. Sci., 1995, vol.5, No.1, 169–188

SPURIOUS GALOIS FIELDS[†]

CZESŁAW KOŚCIELNY[*]

An algebraic system consisting of a finite set of q elements (q – any integer ≥ 2) with two internal binary operations of addition and multiplication is studied. For the described system, which satisfies all the axioms of the fields except for the axiom of associativity of addition, which may, but need not, be assured, the name spurious Galois field and the symbol $SGF(q)$ are proposed. The spurious Galois fields constitute a class of algebraic systems, containing all the Galois fields as its small subclass; therefore, the presented problem can be considered as a generalization of finite fields. The way of forming the spurious Galois fields, the approach to computing in $SGF(q)$ as well as some possibilities of applications of this algebraic structure in cryptography are discussed.

1. Introduction

The task of the paper is to let the reader see the existence of algebraic systems very similar to the finite fields and to show how to construct them. The systems have been called spurious Galois fields, and the symbol $SGF(q)$ has been used to denote them.

This article is a considerably enlarged and modified version of the first communication on this subject (Kościelny, 1989). In comparison with this publication, the definition of $SGF(q)$ was essentially modified in the paper. Due to the this modification such algebraic systems are being studied, which to a great degree resemble the Galois fields.

$SGF(q)'s$ seem to be useful for cryptography, Doppler radar, coding theory, and algebraic theory of automata. The subject of spurious Galois fields may also attract the attention of those engineers and mathematicians who are interested in applications of latin squares, quasigroups, finite geometries and statistical designs. However, the paper has been written taking mainly into account the applications of spurious Galois fields in cryptography.

The paper is organized as follows: Section 2 contains the definition of a spurious Galois field, Section 3 gives the rules of computing in $SGF(q)$, Section 4 shows how to solve the problem of implementation of operations in $SGF(q)$ to obtain the powerful tool for applications, Section 5 discusses the question of isomorphisms of spurious Galois fields, Section 6 presents several practical remarks concerning problems of computing Zech's logarithms, while Section 7 should draw the reader's attention to the

[†] This work has been supported by the KBN (Committee for Scientific Research in Poland), grant No. PB 1230/P4/93/04.

[*] Technical University of Wrocław, Institute of Engineering Cybernetics, ul. Janiszewskiego 11–17, 50–372 Wrocław, Poland

method of constructing $SGF(q)$-based stream- and block-ciphers. Some important final remarks are presented in Conclusions.

The author follows strictly the notation commonly used in coding theory and cryptography (MacWilliams and Sloane, 1977; Peterson and Weldon, 1972), therefore, the paper is easy to read for specialists. Beginners, however, may have some difficulty in understanding this work but fortunately there exist an excellent and extensive algebraic introduction to finite fields (Lidl and Niederreiter, 1983) and a very good and actual guide concerning their applications as well (Menezes *et al.*, 1993).

2. Definition of a Spurious Galois Field

Consider an algebraic system

$$\langle SF,\ +,\ \cdot\rangle \tag{1}$$

consisting of a finite set of elements SF in which two internal binary operations, called addition and multiplication, respectively, are defined. Let $\mid SF \mid = q$ and let q denote an arbitrary integer ≥ 2.

The spurious Galois field, denoted by $SGF(q)$, is system (1), satisfying the axioms

A.1 $\langle SF,\ +\rangle$ is an abelian loop [1] with identity element denoted by 0.

A.2 $\langle SF^{\star},\ \cdot\rangle$ is an abelian cyclic group, where $SF^{\star} = SF \setminus \{0\}$. This implies that $SF = \{0,\ 1,\ \omega,\ \omega^2, \ldots, \omega^{q-2}\}$; ω is the generator of the multiplicative group, 1 denotes the identity element under multiplication, and $\omega^{q-1} = 1$.

A.3 $\exists\ -1 \in SF\ \ [-1 = 1$ if q is even $] \vee [-1 = \omega^{(q-1)/2}$ if q is odd$]$.

A.4 $\forall\, a \in SF\ \exists\ -a \in SF\ \ [-a = -1a] \wedge [a + (-a) = 0]$.

A.5 $\forall\, a,\ b,\ c \in SF\ \ [a(b+c) = ab + ac] \wedge [(b+c)a = ba + ca]$.

A.6 $\forall\, a \in SF\ \ 0a = 0$.

The above formulated axioms make it possible to form the algebraic structures preserving many properties of finite fields, even if q is not a power of prime.

3. The Rules of Performing Operations in $SGF(q)$

In discussing this question, the evident fact that the multiplication is associative and both operations are commutative will not be repeated.

Let

$$Q = \{0, 1, \ldots, q-2\} \tag{2}$$

[1] An algebraic system $\langle S,\ +\rangle$ is called a *quasigroup* if there is a binary operation $(+)$ defined in S and if, when any two elements $a, b \in S$ are given, equations $a + x = b$ and $y + a = b$, each, have exactly one solution. A *loop* $\langle L,\ +\rangle$ is a quasigroup with an identity element: that is, a quasigroup in which there exists an element $e \in L$ with the property that $e + x = x + e = x$ for every $x \in L$. A finite loop is *commutative* or *abelian* if, and only if, for an abritrary pair of loop elements a, b, $a + b = b + a$ holds.

From the properties of system $\langle SF^{\star},\ \cdot\rangle$ the following rules of performing the multiplicative operations result

$$\forall\, r,\ s \in Q \qquad \omega^r\omega^s = \omega^{r+s} \pmod{q-1} \tag{3}$$

$$\forall\, r \in Q \qquad \omega^{-r} = 1/\omega^r = \omega^{q-1-r} \tag{4}$$

$$\forall\, r \in Q \qquad \omega^{-r}\omega^r = \omega^0 = 1 \tag{5}$$

$$\forall\, r,\ k \in Q \qquad (\omega^r)^k = \omega^{rk} \pmod{q-1} \tag{6}$$

$$\forall\, r,\ s \in Q \qquad \omega^r/\omega^s = \omega^{r-s} \pmod{q-1} \tag{7}$$

$$\forall\, r \in Q \qquad [0\omega^r = 0] \wedge [1\omega^r = \omega^r] \tag{8}$$

The rules of addition in $SGF(q)$ are determined by the system $\langle SF,\ +\rangle$. Additive operations involving the identity element are as follows

$$\forall\, r \in Q \qquad \omega^r + 0 = \omega^r \tag{9}$$

$$\forall\, r \in Q \qquad \omega^r + (-\omega^r) = 0 \tag{10}$$

Let us introduce, as in the case of any $GF(q)$, the symbol $-\infty$ to denote 0 as $\omega^{-\infty}$. To perform addition in $SGF(q)$, the well known discrete implicit function, named Zech's logarithm (Huber, 1990; Imamura, 1980), can be applied. It is a special permutation, usually denoted by Z

$$Z : \{-\infty, 0, 1, 2, \ldots, q-2\} \mapsto \{-\infty, 0, 1, 2, \ldots, q-2\}$$

and defined by the equation

$$\omega^{Z(x)} = 1 + \omega^x \tag{11}$$

The properties of the system $\langle SF,\ +\rangle$ are fully described by the function $Z(x)$. It has been noted that the necessary conditions to satisfy axioms A.1 and A.3–A.6 are

$$Z(x) \neq x \qquad \text{for any } q \tag{12}$$

$$Z(-\infty) = 0 \qquad \text{for any } q \tag{13}$$

$$Z(0) = -\infty \qquad \text{for } q \text{ even} \tag{14}$$

$$Z\big((q-1)/2\big) = -\infty \qquad \text{for } q \text{ odd} \tag{15}$$

$$Z(x) \equiv Z(q-1-x) + x \pmod{q-1} \text{ for any } q \tag{16}$$

where

$$x \in \begin{cases} Q \setminus \{0\} & \text{for } q \text{ even,} \\ Q \setminus \{0,\ (q-1)/2\} & \text{for } q \text{ odd.} \end{cases}$$

Formulae (12) and (13) follow from the fact that there is only one identity element under addition and that it equals 0, while (14) and (15) result from A.3. Equation (16) is the most important condition, assuring the satisfaction of A.1 axiom. These equations also imply the following properties of Zech's logarithms

$$Z(0) \equiv \Big((q-1)(3q-5)\Big)/8 \equiv \Big((q-1)(5q-11)\Big)/8 \pmod 2 \text{ for } q \text{ odd} \tag{17}$$

$$2\sum_{x=1}^{(q-2)/2} Z(x) \equiv \Big((q-2)(5q-4)\Big)/8 \pmod{q-1} \qquad \text{for } q \text{ even} \tag{18}$$

$$2\sum_{x=1}^{(q-3)/2} Z(x) \equiv \Big((q-1)(5q-11)\Big)/8 - Z(0) \pmod{q-1} \text{ for } q \text{ odd} \tag{19}$$

Formulae (17)–(19), which can be easily proved, are of great importance in constructing effective algorithms for finding the values of function $Z(x)$.

It has also been observed by the author that for any even q there exists one particular $SGF(q)$ with the following values of Zech's logarithms

$$Z(-\infty) = 0, \quad Z(0) = -\infty$$

$$Z(2k-1) = q/2 + k - 1, \quad Z(2k) = k, \quad k = 1, 2, \ldots, (q-2)/2 \tag{20}$$

Using Zech's logarithms one can write down the general rule of performing addition in $SGF(q)$

$$\forall\, r, s \in Q \cup \{-\infty\}\ \omega^r + \omega^s = \begin{cases} \omega^{\min\{r,s\}+Z(|r-s|) \pmod{q-1}} \\ \quad \text{if } r,\ s \text{ and } Z(|r-s|) \neq -\infty, \\ \omega^r \text{ if } s = -\infty, \\ \omega^s \text{ if } r = -\infty, \\ 0 \quad \text{if } Z(|r-s|) = -\infty \vee r = s = -\infty. \end{cases} \tag{21}$$

Although addition in $SGF(q)$ may not be associative, the equation

$$a + x = b \tag{22}$$

has always a unique solution for any pair of $SGF(q)$ elements a, b. In order to solve equation (22), the notion of subtraction in $SGF(q)$ should be introduced. However, for the most cases subtraction in $SGF(q)$ cannot be performed as in $GF(q)$. About subtraction one can only say that if $a - b = c$, this means that $a = b + c$. Generally, to perform subtraction in $SGF(q)$ the following Dv and Dh implicit functions can be utilized

$$\omega^{Dh(x)} + \omega^x = 1 \tag{23}$$

$$\omega^{Dv(x)} + 1 = \omega^x \tag{24}$$

The formulae below follow immediately from (11), (23) and (24)

$$Z\Big(Dv(x)\Big) = Dv\Big(Z(x)\Big) \tag{25}$$

$$Z\Big(Dv(x)\Big) = x \tag{26}$$

$$Dh\Big(Dv(x) + q - 1 - x \pmod{q-1}\Big) = q - 1 - x \tag{27}$$

allowing to calculate the values of Dv and Dh functions provided that the values of Zech's logarithms in $SGF(q)$ are known. Using (23) and (24) one obtains

$$\forall\, r,\, s \in Q \cup \{-\infty\}\ \ \omega^r - \omega^s = \begin{cases} \omega^{s+Dv(r-s) \pmod{q-1}} \\ \qquad \text{if } r > s,\ \ r,\, s \neq -\infty, \\ \omega^{r+Dh(s-r) \pmod{q-1}} \\ \qquad \text{if } r < s \text{ and if } r,\, s \neq -\infty, \\ \omega^r \quad \text{if } s = -\infty, \\ -\omega^s \text{ if } r = -\infty, \\ 0 \qquad \text{if } r = s. \end{cases} \tag{28}$$

It should be noted that due to the presented approach, addition and subtraction over $SGF(q)$ may be applied in cryptograpy as mutually invertible transformations.

The numbers of all possible $SGF(q)'s$ versus $q \leq 25$ are given in Tab. 1. It was worked out after about 200 hours of computations on IBM PC 386. In Tab. 1, $ni(q)$ denotes the number of all $SGF(q)'s$ having various values of Zech's logarithms. As it can be seen later, some $SGF(q)'s$, having different $Z(x)$, may be isomorphic. Therefore, in Tab. 1, the value $ns(q)$ denoting the number of all non-isomorphic $SGF(q)'s$ is given. Unfortunately, if $q > 24$, the hardware and the algorithms, as used in computations were too failable, viz. unfit for calculating $ns(q)$ in a reasonable period of time.

Tab. 1. The Number of $SGF(q)'s$ Versus q.

q	$ni(q)$	$ns(q)$	q	$ni(q)$	$ns(q)$
2	1	1	3	1	1
4	1	1	5	2	1
6	1	1	7	4	2
8	3	2	9	12	4
10	9	3	11	56	14
12	25	5	13	224	56
14	133	14	15	960	160
16	631	87	17	5760	724
18	3857	242	19	42816	7136
20	25905	1453	21	320512	40064
22	188181	15714	23	2366080	??????
24	1515283	?????	25	20857088	???????

Among the $SGF(q)'s$ there is a subclass of spurious Galois fields satisfying

$$\forall a, b \in SF \quad a - b = a + (-1)b = a + (-b) \tag{29}$$

which are the most similar to fields. This subclass contains all the isomorphic Galois fields. The term *all the isomorphic Galois fields* means here all such finite fields with various values of $Z(x)$ function, whose elements are named in the same manner, resulting from the A.2 axiom. It can be seen that if $2 \le q \le 5$, then all $SGF(q)'s$ are the fields. The really spurious Galois fields can be constructed for $q \ge 6$.

There are very few $SGF(q)'s$ in which subtraction is performed according to (29). All such non-isomorphic $SGF(q)'s$ for $q \le 19$ are given in Tab. 2.

Tab. 2. Zech's Logarithms for $SGF(q)'s$ Satisfying (29)

x	0	1	2	3	4	5	6	7	8	
$\star Z(x)$	$-\infty$									$q=2$
$\star Z(x)$	1									$q=3$
$\star Z(x)$	$-\infty$	2								$q=4$
$\star Z(x)$	1	3								$q=5$
$\star Z(x)$	2	4	1							$q=7$
$\star Z(x)$	$-\infty$	3	6	1						$q=8$
$\star Z(x)$	4	2	7	6						$q=9$
$Z(x)$	1	3	8	7	9					$q=11$
$\star Z(x)$	1	8	4	6	9					$q=11$
$\star Z(x)$	1	4	9	8	2	11				$q=13$
$Z(x)$	1	9	5	7	2	11				$q=13$
$Z(x)$	$-\infty$	4	7	12	1	11	8			$q=14$
$\star Z(x)$	$-\infty$	4	8	14	1	10	13	9		$q=16$
$Z(x)$	$-\infty$	4	9	14	1	10	8	13		$q=16$
$Z(x)$	2	5	3	12	11	15	14	13		$q=17$
$\star Z(x)$	2	5	15	12	11	6	14	10		$q=17$
$Z(x)$	2	6	11	10	1	4	14	3		$q=17$
$Z(x)$	2	11	7	9	1	4	14	3		$q=17$
$Z(x)$	1	6	4	13	12	16	3	14	17	$q=19$
$Z(x)$	1	6	16	13	12	7	3	11	17	$q=19$
$Z(x)$	1	13	4	8	10	16	3	14	17	$q=19$
$\star Z(x)$	1	13	16	8	10	7	3	11	17	$q=19$

In order to calculate all those values of Zech's logarithms which are not listed in Tab. 2, one ought to use (13)–(16). In Tab. 2, 12 Galois fields (marked by $\star$) and 10 spurious Galois fields, are presented.

An $SGF(q)$ becomes a $GF(q)$ if, and only if, q is a power of prime and if addition is an associative operation. Therefore, when $q = p^m$, p – prime, m – integer ≥ 1, Zech's logarithm has additional properties given by the formulae

$$Z\Big(Z\Big(\cdots \underbrace{Z(a\Big))\cdots\Big)}_{p} = a \quad \text{for any } p \text{ and any } a \in SF \tag{30}$$

$$Z\Big(\Big(Z(x)+(q-1)/2\Big) \pmod{q-1}\Big) \equiv \Big(x+(q-1)/2)\Big) \pmod{q-1} \text{ for } p>2 \tag{31}$$

$$Z\Big(\Big(Z(x)\Big) \equiv Z(0)+Z\Big(x+q-1-Z(0) \pmod{q-1}\Big)\Big) \pmod{q-1} \text{ for } p>2 \tag{32}$$

$$Z\Big(xp^i \pmod{q-1}\Big) \equiv p^i Z(x) \pmod{q-1} \tag{33}$$

$$Z\Big((q-1-x)p^i \pmod{q-1}\Big) \equiv p^i\Big(Z(x)+q-1-x \pmod{q-1}\Big) \pmod{q-1} \tag{34}$$

where

$$x \in \begin{cases} Q \setminus \{0\} & \text{if } p=2, \\ Q \setminus \{0,\ (q-1)/2\} & \text{if } p>2, \end{cases}$$

$$i = 1, 2 \ldots, y-1$$

and y is the least integer so that

$$xp^y \equiv x \pmod{q-1}$$

Equations (33) and (34) can be derived as a generalization of formula (18) given by Imamura (1980), an easy proof of (30), (31) and (32) is left to the reader.

The properties of Zech's logarithms expressed by means of (30)–(34) should be sufficient to distinguish a $GF(q)$ from $SGF(q)$.

4. Computing in $SGF(q)$

The easiest way to implement operations in $SGF(q)$ is to replace the abstract representation of elements of the $SGF(q)$ by the set of integers

$$F = \{0, 1, \ldots, q-1\} \tag{35}$$

and to apply the mapping

$$\sigma : \{0,\ 1,\ \omega,\ \cdots,\ \omega^{q-2}\} \mapsto F \tag{36}$$

defined by the function

$$\sigma(\omega^x) = \begin{cases} x+1 & \text{if } x \in \{0, 1, \ldots, q-2\}, \\ 0 & \text{if } x = -\infty. \end{cases} \tag{37}$$

Under the mapping σ any non-zero $SGF(q)$ element ω^x has its image in the set F equal to the ordinary sum $x+1$ while the image of the identity element for addition equals 0. Assuming that the mapping σ is an isomorphism, the relation

$$\forall\, a,\, b \in SGF(q)\ \Big[\sigma(ab) = \sigma(a) \otimes \sigma(b)\Big] \ \wedge\ \Big[\sigma(a+b) = \sigma(a) \oplus \sigma(b)\Big] \tag{38}$$

holds, where the symbols $\oplus$ and $\otimes$ denote the operations on elements of the set F, corresponding to addition and multiplication in $SGF(q)$, respectively. Therefore, for the mapping σ there exists an inverse mapping defined as follows

$$\sigma^{-1}(x) = \begin{cases} \omega^{x-1} & \text{if } x \in F \setminus \{0\}, \\ \omega^{-\infty} = 0 & \text{if } x = 0. \end{cases} \tag{39}$$

The mapping σ converts the operations in $SGF(q)$ onto ordinary operations on integers. For the needs of applications, it will be sufficient to define six functions, returning the product, the multiplicative inverse, the k-th power, the additive inverse, the sum and the difference of the images of $SGF(q)$ elements under the mapping σ. Denoting these functions by $P(x,y)$, $Mi(x)$, $Pwr(x,k)$, $Ai(x)$, $S(x,y)$ and $D(x,y)$, one can express them as follows

$$P(x,y) = \sigma\Big(\sigma^{-1}(x)\sigma^{-1}(y)\Big) \tag{40}$$

$$Mi(x) = \sigma\Big((1/\sigma^{-1}(x)\Big), \qquad x \neq 0 \tag{41}$$

$$Pwr(x,k) = \sigma\Big((\sigma^{-1}(x))^k\Big) \tag{42}$$

$$S(x,y) = \sigma\Big(\sigma^{-1}(x) + \sigma^{-1}(y)\Big) \tag{43}$$

$$Ai(x) = \sigma\Big(-\sigma^{-1}(x)\Big) \tag{44}$$

$$D(x,y) = \sigma\Big(\sigma^{-1}(x) - \sigma^{-1}(y)\Big) \tag{45}$$

Taking into account (4)–(16), (21), (23)–(28) and (35)–(45) one obtains:

$$P(x,y) = \begin{cases} 1 + \Big[x + y - 2 \pmod{q-1}\Big] & \text{if } x,\, y \in F \setminus \{0\} \\ 0 & \text{if } x = 0 \vee y = 0 \end{cases} \tag{46}$$

$$Mi(x) = \begin{cases} q + 1 - x & \text{if } x \in F \setminus \{0,\, 1\} \\ 1 & \text{if } x = 1 \end{cases} \tag{47}$$

$$Pwr(x,k) = \begin{cases} 1 + [(x-1)k \pmod{q-1}] & \text{if } x \in F \setminus \{0\} \\ 0 & \text{if } x = 0 \end{cases} \tag{48}$$

$$S(x,y) = \begin{cases} 1 + [\min\{x,y\} - 1 + Z(|x-y|) \pmod{q-1}] \\ \qquad \text{if } x,\ y \in F \setminus \{0\} \text{ and if } Z(|x-y|) \neq -\infty, \\ x+y \quad \text{if } x = 0 \vee y = 0, \\ 0 \qquad \text{if } Z(|x-y|) = -\infty, \end{cases} \tag{49}$$

$$Ai(x) = \begin{cases} 1 + [x - 1 + (q-1)/2 \pmod{q-1}] \\ \qquad \text{if } x \in F \setminus \{0\} \text{ and if } q \text{ is odd,} \\ x \quad \text{if } x \in F \setminus \{0\} \text{ and if } q \text{ is even,} \\ 0 \quad \text{if } x = 0, \end{cases} \tag{50}$$

$$D(x,y) = \begin{cases} 0 & \text{if } x = y, \\ x & \text{if } y = 0, \\ Ai(y) & \text{if } x = 0, \\ 1 + [(y - 1 + Dv(x-y)) \pmod{q-1}] \\ \qquad \text{if } x > y \text{ and if } x,\ y \in F \setminus \{0\}, \\ 1 + [(x - 1 + Dh(y-x)) \pmod{q-1}] \\ \qquad \text{if } x < y \text{ and if } x,\ y \in F \setminus \{0\}. \end{cases} \tag{51}$$

Formulae (46)–(51) can easily be written in any programming language. It is evident that the operations in $SGF(q)$ can also be implemented using the micro-programmed devices or the appropriate hardware.

5. Isomorphisms of Spurious Galois Fields

Let the element ω be a generator of the multiplicative group of $SGF(q)$, and let $Z(x)$ for $x = -\infty, \ 0, 1, \ldots, q-2$ be a set of all values of Zech's logarithms in $SGF(q)$. To find a complete set of values of this function in the isomorphic spurious Galois field, whose elements are denoted according to A.2 axiom, the mapping:

$$\tau : \omega \mapsto \omega^i \tag{52}$$

where

$$\gcd(i, q-1) = 1 \tag{53}$$

ought to be applied.

Further, let $\xi = \omega^i$, i satisfying (53), and let $Z'(x)$ denote now the values of Zech's logarithms in the spurious Galois field for which ξ is the generator of the multiplicative group of $SGF(q)$. Therefore

$$1 + \xi^x = \xi^{Z'(x)} \tag{54}$$

Since

$$1 + \xi^x = 1 + (\omega^i)^x = \omega^{Z(ix)} = \xi^{Z'(x)} = \omega^{iZ'(x)} \tag{55}$$

then

$$iZ'(x) \equiv Z\Big(ix \pmod{q-1}\Big) \qquad \pmod{q-1} \tag{56}$$

and finally

$$Z'(x) \equiv \mathrm{inv}(i)\, Z\Big(ix \pmod{q-1}\Big) \pmod{q-1} \tag{57}$$

where $inv(i)$ denotes the multiplicative inverse of i modulo $q-1$. The congruence equation (56) has the simplest solution if $\xi = 1/\omega$, viz. if $i = q-2$. It can be easily found that in this case

$$Z'(x) \equiv q - 1 + x - Z(x) \pmod{q-1} \tag{58}$$

If, applying mapping (52) for which (53) holds, the equation $Z(x) = Z'(x)$ is fulfilled, one can say that this mapping is an automorphism.

Let us consider an example. In Tab. 3 the values of $Z(x)$ for all non-isomorphic $SGF(11)'s$ are given. The remaining 42 sets of values of $Z(x)$ for isomorphic $SGF(11)$ can be found by means of (57) which ought to be applied for any of 14 outfits of values of this function for $i = 3,\ 7$ and 9. The reader can verify that when applying the above method to outfits No 2 and No 9 one obtains Tab. 4, where Zech's logarithms for the special $SGF(11)'s$ satisfying (29) are listed.

The four sets of values in Tab. 3, marked by $\star$, are the values of Zech's logarithms in the isomorphic $GF(11)'s$ while the remaining entries can be employed to construct the same $SGF(11)$ with an accuracy up to isomorphism.

Tab. 3. Zech's Logarithms for $SGF(11)$.

No	x	$-\infty$	0	1	2	3	4	5	6	7	8	9
1	$Z(x)$	0	1	3	7	9	8	$-\infty$	4	6	5	2
2	$Z(x)$	0	1	3	8	7	9	$-\infty$	5	4	6	2
3	$Z(x)$	0	1	4	7	9	2	$-\infty$	8	6	5	3
4	$Z(x)$	0	1	4	9	8	6	$-\infty$	2	5	7	3
5	$Z(x)$	0	1	5	8	2	7	$-\infty$	3	9	6	4
6	$Z(x)$	0	1	5	9	6	2	$-\infty$	8	3	7	4
7	$Z(x)$	0	1	7	4	8	3	$-\infty$	9	5	2	6
8	$Z(x)$	0	1	7	5	2	8	$-\infty$	4	9	3	6
9	$\star Z(x)$	0	1	8	4	6	9	$-\infty$	5	3	2	7
10	$Z(x)$	0	1	8	6	5	3	$-\infty$	9	2	4	7
11	$Z(x)$	0	1	9	5	7	6	$-\infty$	2	4	3	8
12	$Z(x)$	0	1	9	6	5	7	$-\infty$	3	2	4	8
13	$Z(x)$	0	5	2	8	7	3	$-\infty$	9	4	6	1
14	$Z(x)$	0	5	4	8	2	1	$-\infty$	7	9	6	3

Tab. 4. Zech's Logarithms for All Isomorphic $SGF(11)$ satisfying (29).

No	x	$-\infty$	0	1	2	3	4	5	6	7	8	9
2	$Z(x)$	0	1	3	8	7	9	$-\infty$	5	4	6	2
2→	$Z(x)$	0	3	2	7	9	8	$-\infty$	4	6	5	1
2→	$Z(x)$	0	7	9	5	4	6	$-\infty$	2	1	3	8
2→	$Z(x)$	0	9	8	4	6	5	$-\infty$	1	3	2	7
9	$\star Z(x)$	0	1	8	4	6	9	$-\infty$	5	3	2	7
9→	$\star Z(x)$	0	3	9	7	4	6	$-\infty$	2	1	5	8
9→	$\star Z(x)$	0	7	2	5	9	8	$-\infty$	4	6	3	1
9→	$\star Z(x)$	0	9	3	8	7	5	$-\infty$	1	4	6	2

6. Remarks Concerning Calculation of Zech's Logarithms

The problem of constructing the spurious Galois fields resolves itself into calculating Zech's logarithms, satisfying (12)–(16). However, no successful method has yet been discovered to determine directly Zech's logarithms for all possible $SGF(q)'s$ of preassigned q, nor does one know in advance how many such spurious fields there exist. Therefore, in order to estimate how quickly the number of $SGF(q)'s$ increases versus q, Tab. 1 was calculated using a systematic search. The algorithm as applied to this purpose utilized two essential software tools: the generator of of r-combinations of n distinct numbers from the set P,

$$C^{(i)}(n,r) = c_1^i, c_2^i, \ldots, c_r^i$$

$$c_k^i \in P, \quad |P| = n, \quad r \le n/2, \quad i = 1, 2, \ldots, \binom{n}{r}$$

and the generator of all permutations of a chosen combination

$$\pi^{(j)} : C^{(i)}(n,r) \mapsto C^{(i)}(n,r) = a_1^j, a_2^j, \ldots, a_r^j,$$

where

$$\pi^{(j)}(c_k^i) = a_k^j \qquad j = 1, 2, \ldots, rr, \quad rr = r! - \Delta rr$$

satisfying $a_k^j \ne k$, where Δrr, denotes the number of permutations, for which $a_k^j = k$.

Taking the above into account and making good use of formulae (12)–(19), one can immediately construct the following algorithm for computing all different sets of Zech's logarithms for a given q

```
begin
   if $q$ $MOD$ $2 = 0$ then
   begin
      $i := 0$;  $n := q - 2$;  $r := (q - 2)/2$;  $P := \{1, 2, \ldots, q - 2\}$;
      repeat
         $i := i + 1$; generate $C_i^{(i)}(n,r)$;
```

```
        $t1 := 2\sum_{s=1}^{r} c_s^{(i)}\ MOD\ (q-1)$;
        $t2 := ((q-2)(5q-4))/8\ MOD\ (q-1)$;
        if $t1 = t2$ then
        for $j := 1$ to $rr$ do
        begin
            generate $\pi^{(j)} = a_1^j,\ a_2^j, \ldots, a_r^j$;
            for $s := 1$ to $r$ do
            begin
                $Z^{(j)}(s) := a_s^j$;  $Z^{(j)}(q-1-s) := (a_s^j - s + q - 1)\ MOD\ (q-1)$
            end;
            if for $s := 1$ to $q-2$  $Z^{(j)}(s)$ are all different then
            begin
                $Z^{(j)}(-\infty) := 0$;  $Z^{(j)}(0) := -\infty$;
                for $s := -\infty$ to $q-2$ do write($Z^{(j)}(s)$)
            end
        end
    until $i = \binom{n}{r}$
end
else begin
    $ii := -2$;  $n := q-3$;  $r := (q-3)/2$;
    $Z^{(i)}(0) := ((q-1)(5q-11))/8\ MOD\ 2$;
    if $Z^{(i)}(0) = 0$ then $ii := 0$ else $ii := -1$;  $P := \{1, 2, \ldots, q-2\}$;
    repeat
        $ii := ii + 2$;  $Z^{(i)}(0) := ii$;  $P := P \setminus \{Z^{(i)}(0)\}$;
        repeat
            $i := i + 1$; generate $C_i^{(i)}(n, r)$;
            $t1 := 2\sum_{s=1}^{r} c_s^{(i)}\ MOD\ (q-1)$
            $t2 := (((q-1)(5q-11))/8 - Z^{(i)}(0))\ MOD\ (q-1)$;
            if $t1 = t2$ then
            for $j := 1$ to $rr$ do
            begin
                generate $\pi^{(j)} = a_1^j,\ a_2^j, \ldots, a_r^j$;
                for $s := 1$ to $r$ do
                begin
                    $Z^{(j)}(s) := a_s^j$;  $Z^{(j)}(q-1-s) := (a_s^j - s + q - 1)\ MOD\ (q-1)$
                end;
                if for s:=1 to q-2  $Z^{(j)}(s)$ are all different then
                begin
                    $Z^{(j)}(-\infty) := 0$;  $Z^{(j)}(0) := Z^{(i)}(0)$;  $Z^{(j)}((q-1)/2) := -\infty$;
                    for $s := -\infty$ to $q-2$ do write($Z^{(j)}(s)$)
                end
            end
        until $i = \binom{n}{r}$;
        $P := P \cup \{Z^{(i)}(0)\}$
    until $ii \geq q-3$
end
end.
```

This algorithm may be the ancestor of many modified and much more effective algorithms for calculating Zech's logarithms. It should be underlined that for clarity, the notation of the algorithm is purposely redundant.

The use of algorithm for $q > 20$ has allowed the author to observe that in the space of all permutations $\pi^{(j)}$, those satisfying the conditions of Zech's logarithms are very irregularly distributed. There exist regions of conglomeration of permutations being Zech's logarithms and these regions are very far away from each other. This phenomenon may be of some importance in constructing algorithms for calculating Zech's logarithms for greater q. However, the difficult problem of determining Zech's logaritms for $q > 256$ is still waiting to be resolved.

As to the algorithm applied by the author for sieving Zech's logarithms for isomorphic $SGF(q)$, one must say that it is very inefficient and does not deserve publication.

7. Examples of $SGF(q)$-Based Ciphers

In this Section, the ciphers over $SGF(32)$ are considered. It is assumed that the values of Zech's logarithms, applied in the example, correspond to permutation

$$\begin{array}{l}(-\infty,\ 0)\ (2,\ 24,\ 21,\ 10,\ 20,\ 26,\ 11,\ 6,\ 23,\ 5,\ 16,\ 18,\ 14,\ 15)\\ (1,\ 4,\ 29,\ 22,\ 30,\ 3,\ 12,\ 19,\ 7,\ 28,\ 9,\ 8,\ 13,\ 27,\ 25,\ 17)\end{array}$$

which is represented as three disjont cycles. For the reader's convenience, Tabs. 5, 6 and 7, presenting the rules of addition, subtraction and multiplication, respectively, were computed by means of (13)–(16), (23)–(27), (35), (40)–(51) and by using the permutation

$$\left(\begin{array}{cccccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15\\ \star & A & B & C & D & E & F & G & H & I & J & K & L & M & N & O \end{array}\right.$$
$$\left.\begin{array}{cccccccccccccccc} 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 & 29 & 30 & 31\\ P & Q & R & S & T & U & V & W & X & Y & Z & . & ? & ! & \# & \% \end{array}\right)$$

The Tables will be useful during the verification of enciphering and deciphering procedures concerning the example ciphers.

Let the sequences of $SGF(q)$ elements

$$M = m_1\ m_2\ \cdots\ m_i\ \cdots,$$
$$K = k_1\ k_2\ \cdots\ k_i\ \cdots,$$
$$C = c_1\ c_2\ \cdots\ c_i\ \cdots,$$

denote the stream of characters of the plain-text, the stream of characters of the key and the stream of characters of the cryptogram, respectively. The stream-cipher is constructed by means of an enciphering function f_e as follows

$$c_i = f_e(m_i, k_i)$$

Tab. 5. Addition Table in $SGF(32)$.

(+)	*	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%
*	*	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%
A	A	*	E	Y	M	#	Q	X	!	N	I	U	G	T	?	P	C	S	B	O	H	.	K	%	F	V	R	L	Z	J	W	D
B	B	E	*	F	Z	N	%	R	Y	#	O	J	V	H	U	!	Q	D	T	C	P	I	?	L	A	G	W	S	M	.	K	X
C	C	Y	F	*	G	.	O	A	S	Z	%	P	K	W	I	V	#	R	E	U	D	Q	J	!	M	B	H	X	T	N	?	L
D	D	M	Z	G	*	H	?	P	B	T	.	A	Q	L	X	J	W	%	S	F	V	E	R	K	#	N	C	I	Y	U	O	!
E	E	#	N	.	H	*	I	!	Q	C	U	?	B	R	M	Y	K	X	A	T	G	W	F	S	L	%	O	D	J	Z	V	P
F	F	Q	%	O	?	I	*	J	#	R	D	V	!	C	S	N	Z	L	Y	B	U	H	X	G	T	M	A	P	E	K	.	W
G	G	X	R	A	P	!	J	*	K	%	S	E	W	#	D	T	O	.	M	Z	C	V	I	Y	H	U	N	B	Q	F	L	?
H	H	!	Y	S	B	Q	#	K	*	L	A	T	F	X	%	E	U	P	?	N	.	D	W	J	Z	I	V	O	C	R	G	M
I	I	N	#	Z	T	C	R	%	L	*	M	B	U	G	Y	A	F	V	Q	!	O	?	E	X	K	.	J	W	P	D	S	H
J	J	I	O	%	.	U	D	S	A	M	*	N	C	V	H	Z	B	G	W	R	#	P	!	F	Y	L	?	K	X	Q	E	T
K	K	U	J	P	A	?	V	E	T	B	N	*	O	D	W	I	.	C	H	X	S	%	Q	#	G	Z	M	!	L	Y	R	F
L	L	G	V	K	Q	B	!	W	F	U	C	O	*	P	E	X	J	?	D	I	Y	T	A	R	%	H	.	N	#	M	Z	S
M	M	T	H	W	L	R	C	#	X	G	V	D	P	*	Q	F	Y	K	!	E	J	Z	U	B	S	A	I	?	O	%	N	.
N	N	?	U	I	X	M	S	D	%	Y	H	W	E	Q	*	R	G	Z	L	#	F	K	.	V	C	T	B	J	!	P	A	O
O	O	P	!	V	J	Y	N	T	E	A	Z	I	X	F	R	*	S	H	.	M	%	G	L	?	W	D	U	C	K	#	Q	B
P	P	C	Q	#	W	K	Z	O	U	F	B	.	J	Y	G	S	*	T	I	?	N	A	H	M	!	X	E	V	D	L	%	R
Q	Q	S	D	R	%	X	L	.	P	V	G	C	?	K	Z	H	T	*	U	J	!	O	B	I	N	#	Y	F	W	E	M	A
R	R	B	T	E	S	A	Y	M	?	Q	W	H	D	!	L	.	I	U	*	V	K	#	P	C	J	O	%	Z	G	X	F	N
S	S	O	C	U	F	T	B	Z	N	!	R	X	I	E	#	M	?	J	V	*	W	L	%	Q	D	K	P	A	.	H	Y	G
T	T	H	P	D	V	G	U	C	.	O	#	S	Y	J	F	%	N	!	K	W	*	X	M	A	R	E	L	Q	B	?	I	Z
U	U	.	I	Q	E	W	H	V	D	?	P	%	T	Z	K	G	A	O	#	L	X	*	Y	N	B	S	F	M	R	C	!	J
V	V	K	?	J	R	F	X	I	W	E	!	Q	A	U	.	L	H	B	P	%	M	Y	*	Z	O	C	T	G	N	S	D	#
W	W	%	L	!	K	S	G	Y	J	X	F	#	R	B	V	?	M	I	C	Q	A	N	Z	*	.	P	D	U	H	O	T	E
X	X	F	A	M	#	L	T	H	Z	K	Y	G	%	S	C	W	!	N	J	D	R	B	O	.	*	?	Q	E	V	I	P	U
Y	Y	V	G	B	N	%	M	U	I	.	L	Z	H	A	T	D	X	#	O	K	E	S	C	P	?	*	!	R	F	W	J	Q
Z	Z	R	W	H	C	O	A	N	V	J	?	M	.	I	B	U	E	Y	%	P	L	F	T	D	Q	!	*	#	S	G	X	K
.	.	L	S	X	I	D	P	B	O	W	K	!	N	?	J	C	V	F	Z	A	Q	M	G	U	E	R	#	*	%	T	H	Y
?	?	Z	M	T	Y	J	E	Q	C	P	X	L	#	O	!	K	D	W	G	.	B	R	N	H	V	F	S	%	*	A	U	I
!	!	J	.	N	U	Z	K	F	R	D	Q	Y	M	%	P	#	L	E	X	H	?	C	S	O	I	W	G	T	A	*	B	V
#	#	W	K	?	O	V	.	L	G	S	E	R	Z	N	A	Q	%	M	F	Y	I	!	D	T	P	J	X	H	U	B	*	C
%	%	D	X	L	!	P	W	?	M	H	T	F	S	.	O	B	R	A	N	G	Z	J	#	E	U	Q	K	Y	I	V	C	*

while during deciphering a function f_d is used

$$m_i = f_d(c_i, k_i)$$

In the simplest case one can assume that

$$\begin{aligned} f_e^{(1)} &= m_i + k_i, & f_d^{(1)} &= c_i - k_i \\ f_e^{(2)} &= m_i - k_i, & f_d^{(2)} &= c_i + k_i \\ f_e^{(3)} &= k_i - m_i, & f_d^{(3)} &= k_i - c_i \\ f_e^{(4)} &= -m_i - k_i, & f_d^{(4)} &= -c_i - k_i \end{aligned}$$

It is evident that taking the above into account and using the operations in an arbitrary $SGF(q)$ and the same sequences M and K, one can generate three

Tab. 6. Subtraction Table in $SGF(32)$.

(-)	*	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%
*	*	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%
A	A	*	X	G	K	R	Z	C	J	O	H	D	V	Y	#	I	U	%	E	.	W	P	L	T	B	M	F	S	!	?	N	Q
B	B	R	*	Y	H	L	S	.	D	K	P	I	E	W	Z	%	J	V	A	F	?	X	Q	M	U	C	N	G	T	#	!	O
C	C	P	S	*	Z	I	M	T	?	E	L	Q	J	F	X	.	A	K	W	B	G	!	Y	R	N	V	D	O	H	U	%	#
D	D	%	Q	T	*	.	J	N	U	!	F	M	R	K	G	Y	?	B	L	X	C	H	#	Z	S	O	W	E	P	I	V	A
E	E	B	A	R	U	*	?	K	O	V	#	G	N	S	L	H	Z	!	C	M	Y	D	I	%	.	T	P	X	F	Q	J	W
F	F	X	C	B	S	V	*	!	L	P	W	%	H	O	T	M	I	.	#	D	N	Z	E	J	A	?	U	Q	Y	G	R	K
G	G	L	Y	D	C	T	W	*	#	M	Q	X	A	I	P	U	N	J	?	%	E	O	.	F	K	B	!	V	R	Z	H	S
H	H	T	M	Z	E	D	U	X	*	%	N	R	Y	B	J	Q	V	O	K	!	A	F	P	?	G	L	C	#	W	S	.	I
I	I	J	U	N	.	F	E	V	Y	*	A	O	S	Z	C	K	R	W	P	L	#	B	G	Q	!	H	M	D	%	X	T	?
J	J	!	K	V	O	?	G	F	W	Z	*	B	P	T	.	D	L	S	X	Q	M	%	C	H	R	#	I	N	E	A	Y	U
K	K	V	#	L	W	P	!	H	G	X	.	*	C	Q	U	?	E	M	T	Y	R	N	A	D	I	S	%	J	O	F	B	Z
L	L	.	W	%	M	X	Q	#	I	H	Y	?	*	D	R	V	!	F	N	U	Z	S	O	B	E	J	T	A	K	P	G	C
M	M	D	?	X	A	N	Y	R	%	J	I	Z	!	*	E	S	W	#	G	O	V	.	T	P	C	F	K	U	B	L	Q	H
N	N	I	E	!	Y	B	O	Z	S	A	K	J	.	#	*	F	T	X	%	H	P	W	?	U	Q	D	G	L	V	C	M	R
O	O	S	J	F	#	Z	C	P	.	T	B	L	K	?	%	*	G	U	Y	A	I	Q	X	!	V	R	E	H	M	W	D	N
P	P	O	T	K	G	%	.	D	Q	?	U	C	M	L	!	A	*	H	V	Z	B	J	R	Y	#	W	S	F	I	N	X	E
Q	Q	F	P	U	L	H	A	?	E	R	!	V	D	N	M	#	B	*	I	W	.	C	K	S	Z	%	X	T	G	J	O	Y
R	R	Z	G	Q	V	M	I	B	!	F	S	#	W	E	O	N	%	C	*	J	X	?	D	L	T	.	A	Y	U	H	K	P
S	S	Q	.	H	R	W	N	J	C	#	G	T	%	X	F	P	O	A	D	*	K	Y	!	E	M	U	?	B	Z	V	I	L
T	T	M	R	?	I	S	X	O	K	D	%	H	U	A	Y	G	Q	P	B	E	*	L	Z	#	F	N	V	!	C	.	W	J
U	U	K	N	S	!	J	T	Y	P	L	E	A	I	V	B	Z	H	R	Q	C	F	*	M	.	%	G	O	W	#	D	?	X
V	V	Y	L	O	T	#	K	U	Z	Q	M	F	B	J	W	C	.	I	S	R	D	G	*	N	?	A	H	P	X	%	E	!
W	W	#	Z	M	P	U	%	L	V	.	R	N	G	C	K	X	D	?	J	T	S	E	H	*	O	!	B	I	Q	Y	A	F
X	X	G	%	.	N	Q	V	A	M	W	?	S	O	H	D	L	Y	E	!	K	U	T	F	I	*	P	#	C	J	R	Z	B
Y	Y	C	H	A	?	O	R	W	B	N	X	!	T	P	I	E	M	Z	F	#	L	V	U	G	J	*	Q	%	D	K	S	.
Z	Z	?	D	I	B	!	P	S	X	C	O	Y	#	U	Q	J	F	N	.	G	%	M	W	V	H	K	*	R	A	E	L	T
.	.	U	!	E	J	C	#	Q	T	Y	D	P	Z	%	V	R	K	G	O	?	H	A	N	X	W	I	L	*	S	B	F	M
?	?	N	V	#	F	K	D	%	R	U	Z	E	Q	.	A	W	S	L	H	P	!	I	B	O	Y	X	J	M	*	T	C	G
!	!	H	O	W	%	G	L	E	A	S	V	.	F	R	?	B	X	T	M	I	Q	#	J	C	P	Z	Y	K	N	*	U	D
#	#	E	I	P	X	A	H	M	F	B	T	W	?	G	S	!	C	Y	U	N	J	R	%	K	D	Q	.	Z	L	O	*	V
%	%	W	F	J	Q	Y	B	I	N	G	C	U	X	!	H	T	#	D	Z	V	O	K	S	A	L	E	R	?	.	M	P	*

different cryptograms if q is even, and four various cryptograms if q is odd. This is a substantial progress in comparison with the stream-ciphers built over $GF(q)$ of characteristic 2.

The enciphering procedure using the stream-cipher over $SGF(32)$ in Tab. 8 is presented. It is left to the reader to verify that using the key K and the equations

$$m_i = c_i^{(1)} - k_i$$

$$m_i = c_i^{(2)} + k_i$$

$$m_i = k_i - c_i^{(3)}$$

respectively, one can obtain the same plain-text from the cryptograms $C^{(1)}, C^{(2)}$ and $C^{(3)}$.

Tab. 7. Multiplication Table in $SGF(32)$.

(x)	*	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
A	*	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%
B	*	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A
C	*	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B
D	*	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C
E	*	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D
F	*	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E
G	*	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F
H	*	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G
I	*	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H
J	*	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I
K	*	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J
L	*	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K
M	*	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L
N	*	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M
O	*	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	*	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	*	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	*	R	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	*	S	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	*	T	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	*	U	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	*	V	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	*	W	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	*	X	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	*	Y	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	*	Z	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
.	*	.	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
?	*	?	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.
!	*	!	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?
#	*	#	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!
%	*	%	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?	!	#

Tab. 8. Enciphering Procedure Using $SGF(32)$-based Stream-Cipher.

step	message	key	$C^{(1)}$	$C^{(2)}$	$C^{(3)}$
i	m_i	k_i	$c_i^{(1)} = m_i + k_i$	$c_i^{(2)} = m_i - k_i$	$c_i^{(3)} = k_i - m_i$
1	H	%	M	I	N
2	O	D	J	#	Y
3	M	D	L	A	K
4	O	?	K	M	W
5	$\star$	#	#	#	#
6	S	P	?	O	Z
7	U	A	.	K	P
8	M	L	P	!	D

Now the way of constructing block-ciphers by means of linear invertible transformations of 4-dimensional vector spaces over $SGF(q)$ will be presented.

One can verify that over an arbitrary $SGF(q)$ the matrix

$$E = \begin{bmatrix} 0 & 0 & a & 0 \\ b & c & d & 0 \\ 0 & 0 & 0 & e \\ 0 & f & 0 & 0 \end{bmatrix}$$

where a, b, c, d, e and f denote non-zero elements of $SGF(q)$, has the following inverse

$$D = \begin{bmatrix} -(d/(a*b)) & 1/b & 0 & -(c/(b*f)) \\ 0 & 0 & 0 & 1/f \\ 1/a & 0 & 0 & 0 \\ 0 & 0 & 1/e & 0 \end{bmatrix}$$

Let

$$M = \begin{bmatrix} m_1, & m_2, & m_3, & m_4 \end{bmatrix}$$

be a 4-character plain-text, thus multiplying it by the encrypting matrix one obtains the cryptogram

$$C = ME = \begin{bmatrix} m_1, & m_2, & m_3, & m_4 \end{bmatrix} \begin{bmatrix} 0 & 0 & a & 0 \\ b & c & d & 0 \\ 0 & 0 & 0 & e \\ 0 & f & 0 & 0 \end{bmatrix} = \begin{bmatrix} c_1, & c_2, & c_3, & c_4 \end{bmatrix}$$

Therefore

$$\begin{bmatrix} c_1, & c_2, & c_3, & c_4 \end{bmatrix} = \begin{bmatrix} b*m_2, & c*m_2+f*m_4, & a*m_1+d*m_2, & e*m_3 \end{bmatrix}$$

The decrypting process consists in calculating the product

$$M = CD = \begin{bmatrix} c_1, & c_2, & c_3, & c_4 \end{bmatrix} \begin{bmatrix} -(d/(a*b)) & 1/b & 0 & -(c/(b*f)) \\ 0 & 0 & 0 & 1/f \\ 1/a & 0 & 0 & 0 \\ 0 & 0 & 1/e & 0 \end{bmatrix}$$

$$= \begin{bmatrix} c_3/a-((c_1*d)/(a*b)), & c_1/b, & c_4/e, & c_2/f-((c*c_1)/(b*f)) \end{bmatrix}$$

$$= \begin{bmatrix} m_1, & m_2, & m_3, & m_4 \end{bmatrix}$$

giving finally the plain-text.

Adapting these general considerations to $SGF(32)$ with the operation tables as in Tabs. 5–7, and posing $a = M$, $b = C$, $c = F$, $d = Q$, $e = Z$ and $f = S$ the following encrypting matrix can be obtained

$$E = \begin{bmatrix} \star & \star & M & \star \\ C & F & Q & \star \\ \star & \star & \star & Z \\ \star & S & \star & \star \end{bmatrix}$$

The decrypting matrix will then be

$$D = \begin{bmatrix} C & \# & \star & Q \\ \star & \star & \star & N \\ T & \star & \star & \star \\ \star & \star & G & \star \end{bmatrix}$$

Using the tables of addition, subtraction and multiplication in $SGF(32)$ one can easily verify that

$$\begin{bmatrix} \star & \star & M & \star \\ C & F & Q & \star \\ \star & \star & \star & Z \\ \star & S & \star & \star \end{bmatrix} \begin{bmatrix} C & \# & \star & Q \\ \star & \star & \star & N \\ T & \star & \star & \star \\ \star & \star & G & \star \end{bmatrix} = \begin{bmatrix} A & \star & \star & \star \\ \star & A & \star & \star \\ \star & \star & A & \star \\ \star & \star & \star & A \end{bmatrix}$$

and that

$$\begin{bmatrix} C & \# & \star & Q \\ \star & \star & \star & N \\ T & \star & \star & \star \\ \star & \star & G & \star \end{bmatrix} \begin{bmatrix} \star & \star & M & \star \\ C & F & Q & \star \\ \star & \star & \star & Z \\ \star & S & \star & \star \end{bmatrix} = \begin{bmatrix} A & \star & \star & \star \\ \star & A & \star & \star \\ \star & \star & A & \star \\ \star & \star & \star & A \end{bmatrix}$$

Let the plain-text be $HOMO\star SUM$. Splitting it into two 4-dimensional vectors one can generate the two blocks of cryptogram as follows

$$\begin{bmatrix} H & O & M & O \end{bmatrix} \begin{bmatrix} \star & \star & M & \star \\ C & F & Q & \star \\ \star & \star & \star & Z \\ \star & S & \star & \star \end{bmatrix} = \begin{bmatrix} Q & P & Z & G \end{bmatrix}$$

$$\begin{bmatrix} \star & S & U & M \end{bmatrix} \begin{bmatrix} \star & \star & M & \star \\ C & F & Q & \star \\ \star & \star & \star & Z \\ \star & S & \star & \star \end{bmatrix} = \begin{bmatrix} U & U & D & O \end{bmatrix}$$

Thus one will obtain the cryptogram $QPZGUUDO$, which can assist in verifying the decrypting process

$$\begin{bmatrix} Q & P & Z & G \end{bmatrix} \begin{bmatrix} C & \# & \star & Q \\ \star & \star & \star & N \\ T & \star & \star & \star \\ \star & \star & G & \star \end{bmatrix} = \begin{bmatrix} H & O & M & O \end{bmatrix}$$

$$\begin{bmatrix} U & U & D & O \end{bmatrix} \begin{bmatrix} C & \# & \star & Q \\ \star & \star & \star & N \\ T & \star & \star & \star \\ \star & \star & G & \star \end{bmatrix} = \begin{bmatrix} \star & S & U & M \end{bmatrix}$$

It is obvious that there are many other methods of constructing $SGF(q)$-based ciphers. Among them, the most interesting seem to be the methods which take advantage of the fact that the addition in $SGF(q)$ is not an associative operation. However, this question is beyond the scope of the paper.

8. Conclusions

The spurious Galois fields are suitable for constructing strong and easily implemented cryptosystems. If one constructs, for example, a universal stream cipher for the protection of messages written in the extended ASCII code using addition and subtraction over $SGF(256)$ as mutually invertible transformations, one can obtain an endomorphic cryptosystem of the order of at least 10^{100} (very rough, but credible extrapolation of data from Tab. 1), the degree of which is equal to 256^n, n denoting the length of the message. For cryptographic purposes one also can formally construct over $SGF(q)$ similar algebraic objects as over $GF(q)$, viz. linear recurrences, vector spaces, rings of polynomials, etc. Such constructions, revealing many unusual properties, may considerably extend the possibility of forming cryptosystems much more resistant to cryptanalysis than the stream-cipher. The strength of cryptosystems based on $SGF(q)'s$ results from the facts that there is a huge number od spurious Galois fields even for rather small order of q (say, several hundreds) and that addition over $SGF(q)$ is not associative. This very fact proves the superiority of $SGF(q)$ over $GF(q)$. It is well known that with an accuracy up to isomorphism, there exists only one $GF(q)$. That is why, in order to increase the strength of the cipher constructed over $GF(q)$, q should be equal to about 2^{250}. Evidently, the greater is q, the more difficult the cryptanalytics' work is, but at the same time, enciphering and deciphering procedures become more complicated and more time-consuming. An immense number of $SGF(q)$'s for $q > 100$ eliminates the necessity of increasing the order of q, and ensures sufficient security level. One can therefore expect that $SGF(2^{10})$-based ciphers will be more secure than those using operations in $GF(2^{250})$ with the former requiring much more lower costs of enciphering and deciphering. Thus spurious Galois fields are an advantageous alternative of the cryptographic tools exploited up to now, all the more so as it is possible to construct $SGF(q)$-based public key cryptosystems.

It is also important for cryptology that the knowledge about the spurious Galois fields is still very poor.

The applications of $SGF(q)$'s should not limit themselves to cryptography. The author has noticed that there exists a new kind of PN-sequences over $SGF(q)$. This suggests the existence of $(q^k - 1, k)$ "linear" codes over $SGF(q)$, which may be cyclic. It is also to be hoped that $SGF(q)$'s may be applied in the design of jamming-resistant Doppler radar systems (Schroeder, 1990 – p.267). To sum up, it seems that $SGF(q)$'s are worthy noticing by engineers and researchers.

Acknowledgement

The author thanks J. Irach, M.Sc., for the assistance in implementing the programs for computing Zech's logarithms.

References

Huber K. (1990): *Some comments on Zech's logarithms.* — IEEE Trans. Inform. Theory, v.IT-36, pp.946–950.

Imamura K. (1980): *A method for computing addition tables in $GF(p^n)$.* — IEEE Trans. Inform. Theory, v.IT-26, No.3, pp.367–369.

Kościelny C. (1989):*Spurious Galois fields.* — Proc. IEEE Pacific RIM Conf. *Commun., Comput. and Signal Process.*, Victoria, Canada, pp.416–418.

Kościelny C. and Mochnacki W. (1991): *Cryptographic keys for improving the reliability of ciphers.* — Computer Communications, v.14, No.9, pp.557–561.

Lidl R. and Niederreiter H. (1983): *Finite Fields.* — Addison, Massachusetts: Wesley Publishing Company, Reading.

MacWilliams F.J. and Sloane N.J.A. (1977): *The Theory of Error-Correcting Codes.* — Amsterdam: North Holland.

Menezes A.J. *et al.* (1993): *Applications of Finite Fields.* — Boston-Dordrecht-London: Kluwer Academic Publishers.

Peterson W.W. and Weldon E.J. (1992): *Error-Correcting Codes.* — Cambridge: MIT Press.

Schroeder M.R. (1990): *Number Theory in Science and Communication With Applications in Cryptography, Physics, Digital Information, Computing and Self-Similarity.* — Berlin: Springer-Verlag.

Received: October 1, 1994
Revised: January 20, 1995