J. Appl. Math. & Informatics Vol. **29***(2011), No. 3 - 4, pp. 547 - 562*
Website: http://www.kcam.biz

ON THE COMPUTATION OF THE NON-PERIODIC AUTOCORRELATION FUNCTION OF TWO TERNARY SEQUENCES AND ITS RELATED COMPLEXITY ANALYSIS

CHRISTOS KOUKOUVINOS* AND DIMITRIS E. SIMOS

Abstract. We establish a new formalism of the non-periodic autocorrelation function (NPAF) of two sequences, which is suitable for the computation of the NPAF of any two sequences. It is shown, that this encoding of NPAF is efficient for sequences of small weight. In particular, the check for two sequences of length n having weight w to have zero NPAF can be decided in $\mathcal{O}(n + w^2 \log w)$. For $n > w^2 \log w$, the complexity is $\mathcal{O}(n)$ thus we cannot expect asymptotically faster algorithms.

AMS Mathematics Subject Classification : 05B20, 68Q25, 68W40.
Key words and phrases : Sequences, non-periodic autocorrelation function, complexity, algorithms, worst-case analysis, weighing matrices.

1. Introduction

In this paper, we establish a new formalism that exhibits the cross-fertilization of Combinatorics with Theoretical Computer Science, in the study of sequences with zero non-periodic autocorrelation function. We begin with the definition of the Non-Periodic Autocorrelation Function, NPAF, from [18]:

Definition 1. *Let* $A = [a_1, a_2, \ldots, a_n]$ *be a sequence of length* n*. The* non-periodic autocorrelation function, NPAF*,* $N_A(s)$ *is defined as:*

$$N_A(s) = \sum_{i=1}^{n-s} a_i a_{i+s} \ , s = 0, 1, \ldots, n-1.$$

Definition 2. *Two sequences,* $A = [a_1, \cdots, a_n]$ *and* $B = [b_1, \cdots, b_n]$*, of length* n *are said to have* zero NPAF*, if* $N_A(s) + N_B(s) = 0$ *for* $s = 1, \ldots, n-1$*.*

Received June 27, 2010. Revised October 19, 2010. Accepted October 28, 2010.
*Corresponding author.

We define the NPAF vectors of the sequences A and B as:

$$NPAF(A) = [N_A(1), \ldots, N_A(n-1)]\ NPAF(B) = [N_B(1), \ldots, N_B(n-1)] \quad (1)$$

Therefore, we can decide if the sequences A and B have zero NPAF from its equivalent vector form $NPAF(A) + NPAF(B) = 0$, where with 0 we mean the zero vector of length $n-1$. We note that such pair of sequences have zero NPAF even though it is the sum of their autocorrelations that is zero.

Definition 3. *The support of a sequence $A = [a_1, \ldots, a_n]$ of length n is the set of positions where its entries are nonzero. Thus, the support of a sequence, A, is defined as the set $SUP(A) = \{\pm i : i, a_i > 0 \ \vee \ -i, a_i < 0 \mid i = 1, \ldots, n\}$.*

The support of a sequence and the indices of the sequence, can be made completely separable in the following sets. Clearly, for any sequence A we have $SUP(A) = POS(A) \bigcup \overline{NEG(A)}$, where with $\overline{S}$ we denote the negated set of S.

Definition 4. *The positive and negative support of a sequence $A = [a_1, \ldots, a_n]$ of length n is the set of position where its entries are positive and negative, respectively. Thus, the positive and negative support of a sequence, A, are defined as the sets $POS(A) = \{i : a_i > 0 \mid i = 1, \ldots, n\}$ and $NEG(A) = \{j : a_j < 0 \mid j = 1, \ldots, n\}$, respectively.*

A weighing matrix $W = W(n, w)$ is a square matrix with entries $0, \pm 1$ having w non-zero entries per row and column and inner product of distinct rows equal to zero. Therefore, W satisfies $WW^T = wI_n$ and the number w is called the weight of W. Weighing matrices have been studied extensively, see [4], [5] and [20] and references therein, for detailed information on known and unknown weighing matrices. A well-known necessary condition for the existence of $W(n, w)$ matrices is given in the following lemma taken from [9].

Lemma 1. *It there is a $W(n, w)$ and $n \equiv 2 \pmod 4$, then w is a sum of two squares. Moreover, $w < n$ unless $n = 2$.*

In this paper, we are focusing on $W(2n, w)$ constructed from two circulants. Two sequences of length n, are said to be of type $(0, \pm 1)$ and weight w if they have a total of w non-zero elements and may be used as the first rows of two circulant matrices to obtain a $W(2n, w)$. These pairs of $(0, \pm 1)$ sequences, are also called ternary complementary pairs (TCPs) when they have zero NPAF and are denoted by $TCP(n, w)$. For some details regarding the theory of TCPs, we refer to [6], while for their wide use in applications see Section 2. The two circulants construction for weighing matrices is described in the theorem below, taken from [9].

Theorem 1. *If there exist two circulant matrices A_1, A_2 of order n, with $0, \pm 1$ elements, satisfying $A_1 A_1^T + A_2 A_2^T = wI_n$ and w is an integer, then there exists a $W(2n, w)$, given as*

$$W(2n, w) = \begin{pmatrix} A_1 & A_2 \\ -A_2^T & A_1^T \end{pmatrix} \ \text{or} \ W(2n, w) = \begin{pmatrix} A_1 & A_2 R \\ -A_2 R & A_1 \end{pmatrix}$$

where R is the square matrix of order n with $r_{ij} = 1$ if $i + j - 1 = n$ and 0 otherwise.

2. Applications of ternary complementary pairs

We give some references to works describing applications of $TCP(n, w)$. We do not aim to provide a comprehensive, or by all means complete, treatment of the subject, as this is not the purpose of the present paper. We are merely interested in giving a flavor of the many different application areas involved, in order to exhibit that sequences with zero non-periodic autocorrelation function play a pivotal role in the theory of sequences, and that their applications is of broader interest.

$TCP(n, w)$ are used to construct sequences with desirable properties for radar applications, as described in [22]. Moreover, these sequences intervene in coded aperture imaging ([8]) and higher-dimensional signal processing applications such as time-frequency-coding ([10]) or spatial correlation, [12]. Furthermore, ternary correlated sequences are a way of forming hybrid absorber-diffusers that achieve better scattering performance without additional absorption as noted in [3].

Because of their many applications in communications, $TCP(n, w)$ have been extensively investigated. Recently, complementary sequences set is used in various areas of digital communication systems such as the user code of multi-carrier direct-sequence code division multiple access (DS-CDMA) systems [24], the precode of OFDM systems to reduce peak to average power ratio (PAPR) [2].

Cryptography [23, 28] and coding theory are often concerned with pseudo-random sequences and/or with sequences which can not be compressed any further. It turns out that pseudo-randomness and uncompressability are very close related with low autocorrelation.

Sequences of the above type are also of greatest importance, when constructing combinatorial designs, i.e. weighing matrices. Weighing matrices and optimal weighing designs have long been studied for their use in statistical experiments as first studied by Hotelling [13] and later by Raghavarao [21] and others [4, 19]. These weighing matrices may then used for Coding Theory purposes, i.e. to construct linear codes with desirable properties, see for example [1].

Last we would like to mention that such sequences are interesting objects to study for themselves [14, 18, 20].

3. A new formalism for the NPAF based on the support of two sequences

In this section, we present a new formalism for the NPAF of two sequences, based on their support. The driving force behind this interpretation which led us to formulate the NPAF on the support of the sequences, was to miscarry the unnecessary multiplications between possible zero elements of the sequences, which take place in the NPAF.

We are only concerned with the nonzero elements of the sequences, i.e. the support. Thus, we demonstrate that all the information required to compute the NPAF can be derived as a function of the weight of the two sequences (see also the complexity analysis, in next section).

The following notations and data structures appear to be handy in our effort to express the NPAF on the support of two sequences.

It is well known that two binary sequences with zero NPAF are equivalent to supplementary difference sets (SDS) (for more details see [9]). Our formalism can also be regarded as a generalization of SDS on three levels $\{0, \pm 1\}$. Following [25, 26] we shall be concerned with collections, (denoted by square brackets [ ]) defined on the fixed group $\mathbb{Z}_n$ of order n, in which repeated elements are counted multiply, rather than with sets (denoted by braces { }). The equivalent term in Computer Science is multiset or similar collection, for more details we refer to Knuth [15]. If T_1 and T_2 are two collections then T_1 and T_2 ($T_1 \uplus T_2$) will denote the result of adjoining the elements of T_1 to T_2 with total multiplicities retained. If the resulting list is sorted (with respect to lexicographical ordering or natural ordering for numerical sequences) it will be denoted as $T_1 \& T_2$. For example: $a_1 < a_2 < a_3 \in \mathbb{Z}_n$ and $T_1 = [a_1, a_3, a_2]$, $T_2 = [a_2, a_4, a_1]$ then

$$T_1 \uplus T_2 = [a_1, a_3, a_2, a_2, a_4, a_1] \text{ and } T_1 \& T_2 = [a_1, a_1, a_2, a_2, a_3, a_4] \tag{2}$$

If one evaluates the sum in the NPAF of a sequence A, the following summation is obtained:

$$N_A(s) = \sum_{i=1}^{n-s} a_i a_{i+s} = a_1 a_{1+s} + \ldots + a_{n-s} a_n \ , s = 0, 1, \ldots, n-1. \tag{3}$$

Henceforth, we are only concerned with candidate $TCP(n, w)$ sequences, thus their entries are taken from $\{0, \pm 1\}$. Thus, the pairs $(a_i, a_{i+s}) = a_i a_{i+s}$ have possible values from $\{0, \pm 1\}$. We are interested in finding how many pairs of elements in the support have distance s and how many such pairs will result to a positive or negative value in the NPAF of A. Only the following three cases can occur in the NPAF defined in the support of A:

(i) Let n_1 be the number of pairs (a_i, a_{i+s}) with $a_i = a_{i+s} = 1$.
(ii) Let n_2 be the number of pairs (a_i, a_{i+s}) with $a_i = a_{i+s} = -1$.
(iii) Let n_3 be the number of pairs (a_i, a_{i+s}) with $a_i a_{i+s} = -1$.

Then by a counting argument we derive that $N_A(s) = n_1 + n_2 - n_3, s = 0, 1, \ldots, n-1$.

One natural way to express the operations that occur in NPAF, when having a representation of the position of the elements in the sequence(s) (i.e. the support), is by signed differences. For each one of the above cases we define the following collections (multisets) of signed differences.

Notation. Let A be a sequence of length n as above, with entries from $\{0, \pm 1\}$.

(i) This case corresponds to the $POS(A)$. We define the signed differences in the positive support of A as $D^{+}_{A,1} = [x - y : x > y \ \wedge \ x, y \in POS(A)]$.

(ii) This case corresponds to the $NEG(A)$. We define the signed differences in the negative support of A as $D^{-}_{A,1} = [x-y : x > y \wedge x, y \in NEG(A)]$.

(iii) For $a_i a_{i+s} = -1$ to occur we have two cases. $a_i = 1$, $a_{i+s} = -1$ and vice versa. Thus, we have to define the cross differences between the positive and negative support of A as $D^{\pm}_{A,1} = [x - y : x > y \ \wedge \ x \in POS(A), y \in NEG(A)]$ and $D^{\mp}_{A,1} = [x - y : x > y \ \wedge \ x \in NEG(A), y \in POS(A)]$. Since, we count the totality of differences with repetitions in one way we define $C^{\rightrightarrows}_{A,1} = D^{\pm}_{A,1} \uplus D^{\mp}_{A,1}$.

While we use $\pm, \mp$ in the notation to denote the use of $POS(A), NEG(A)$, the index 1 in the multisets represents that we define the differences in one direction ($\rightrightarrows$) due to the non-periodic property of the autocorrelation function, corresponding to the index of the elements of the sequence A.

The problem of determining if two sequences form a $TCP(n, w)$, i.e. they have zero NPAF, is a decision problem. This statement, is illustrated in the following example:

Example 1. *Consider the following* $TCP(19, 10)$ *which gives rise to a weighing matrix* $W(2 \cdot 19, 10)$ *via Theorem 1 taken from* [7]*:*

```
A=00000+-00000-0000++
B=+000+0000+0+0-00000
```

We can check if the sequences A, B *have zero NPAF using the previous formalism. We use the support representation as follows.*

$$SUP(A) = \{6, -7, -13, 18, 19\} \text{ and } SUP(B) = \{1, 5, 10, 12, -14\}$$

The support sets of the two sequences can be made separable as $POS(A) = \{6, 18, 19\}, NEG(A) = \{7, 13\}$ *and* $POS(B) = \{1, 5, 10, 12\}, NEG(B) = \{14\}$. *The following six collections of differences can now be formed for the two sequences:*

$$\begin{array}{cc} D^{+}_{A,1} = [12, 13, 1] & D^{+}_{B,1} = [4, 9, 11, 5, 7, 2] \\ D^{-}_{A,1} = [6] & D^{-}_{B,2} = [\ ] \\ C^{\rightrightarrows}_{A,1} = [11, 5, 12, 6, 1, 7] & C^{\rightrightarrows}_{B,1} = [13, 9, 4, 2] \end{array}$$

By considering adjoining the elements of the previous collections, we have the following multisets:

$$\begin{array}{c} D^{+}_{A,1} \uplus D^{-}_{A,1} = [12, 13, 1, 6] \\ D^{+}_{B,1} \uplus D^{-}_{B,1} = [4, 9, 11, 5, 7, 2] \\ C^{\rightrightarrows}_{A,1} \uplus C^{\rightrightarrows}_{B,1} = [11, 5, 12, 6, 1, 7, 13, 9, 4, 2] \end{array}$$

It is easy to deduce that the problem of determining if the two sequences are $TCP(19, 10)$*, has boiled down to a comparison sorting problem; since the totality of the signed differences in the two sequences must appear exactly the same times as cross differences in them. This can be seen by forming:*

$$(D^{+}_{A,1} \uplus D^{-}_{A,1})\&(D^{+}_{B,1} \uplus D^{-}_{B,1}) = [1,2,4,5,6,7,9,11,12,13]$$
$$C^{\rightrightarrows}_{A,1}\&C^{\rightrightarrows}_{B,1} = [1,2,4,5,6,7,9,11,12,13]$$

It is clear that the sequences have zero NPAF if the element by element comparison of the previous collections results true for all positions. In other words, if the verification holds true for all positions, i.e. is a decision problem [27]. *We provide the NPAF vectors of A and B for independent verification, where "-" stands for -1:*

s	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$N_A(s)$	0	0	0	0	−	0	−	0	0	0	−	0	1	0	0	0	0	0
$N_B(s)$	0	0	0	0	1	0	1	0	0	0	1	0	−	0	0	0	0	0
$N_A(s) + N_B(s)$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

We write the following Lemma, which acts as a criterion to decide if any pair of two sequences has zero NPAF.

Lemma 2. *Let A, B be two sequences of length n and weight w with entries from $\{0, \pm 1\}$. Then the following are equivalent:*

(i) *A, B are $TCP(n, w)$, i.e. they have zero NPAF*
(ii) $(D^{+}_{A,1} \uplus D^{-}_{A,1})\&(D^{+}_{B,1} \uplus D^{-}_{B,1}) = (D^{\pm}_{A,1} \uplus D^{\mp}_{A,1})\&(D^{\pm}_{B,1} \uplus D^{\mp}_{B,1})$
(iii) $(D^{+}_{A,1} \uplus D^{-}_{A,1})\&(D^{+}_{B,1} \uplus D^{-}_{B,1}) = C^{\rightrightarrows}_{A,1}\&C^{\rightrightarrows}_{B,1}$

3.1. An extreme verification case of ternary complementary pairs. We give an example of an extreme case to decide if two sequences A, B have zero NPAF, which demonstrates the power of the new formalism in the computation of the NPAF derived from the support of the two sequences. The current limits of 32 or 64-bit architecture restricts the representation of an array in terms of memory to a machine. With the new formalism we are able to check if two sequences of length equal to 500 billions are complementary, which is currently entirely out of scope of a conventional machine.

Example 2. *Let A, B be two sequences of length $n = 500 \cdot 10^9$ and weight $w = 10$ with*

$$SUP(A) = 10^9 \cdot [125, -150, -300, 425, 450]$$
$$SUP(B) = 10^9 \cdot [10, 100, 225, 275, -325]$$

The support sets of the two sequences can be made separable as

$$POS(A) = 10^9 \cdot \{125, 425, 450\}, NEG(A) = 10^9 \cdot \{150, 300\}$$
$$POS(B) = 10^9 \cdot \{10, 100, 225, 275\}, NEG(B) = 10^9 \cdot \{325\}$$

The following six collections of differences can now be formed for the two sequences:

$$D^{+}_{A,1} = 10^9 \cdot [300, 325, 25] \qquad D^{+}_{B,1} = 10^9 \cdot [100, 225, 275, 125, 175, 50]$$
$$D^{-}_{A,1} = 10^9 \cdot [150] \qquad D^{-}_{B,1} = [\,]$$
$$C^{\rightrightarrows}_{A,1} = 10^9 \cdot [275, 125, 300, 150, 25, 175] \qquad C^{\rightrightarrows}_{B,1} = 10^9 \cdot [325, 225, 100, 50]$$

By considering adjoining the elements of the previous collections, we have the following multisets:

$$D^{+}_{A,1} \uplus D^{-}_{A,1} = 10^9 \cdot [300, 325, 25, 150]$$
$$D^{+}_{B,1} \uplus D^{-}_{B,1} = 10^9 \cdot [100, 225, 275, 125, 175, 50]$$
$$C^{\rightrightarrows}_{A,1} \uplus C^{\rightrightarrows}_{B,1} = 10^9 \cdot [275, 125, 300, 150, 25, 175, 325, 225, 100, 50]$$

By forming the required multisets as in Lemma 2 the resulting lists are equal element by element:

$$(D^{+}_{A,1} \uplus D^{-}_{A,1})\&(D^{+}_{B,1} \uplus D^{-}_{B,1}) = 10^9 \cdot [25, 50, 100, 125, 150, 175, 225, 275, 300, 325]$$
$$C^{\rightrightarrows}_{A,1}\&C^{\rightrightarrows}_{B,1} = 10^9 \cdot [25, 50, 100, 125, 150, 175, 225, 275, 300, 325]$$

we decide that A, B *are* $TCP(500 \cdot 10^9, 10)$ *since they have zero NPAF.*

4. Complexity analysis of the NPAF defined on the support of two sequences

In this section, we perform a complexity analysis of the problem to decide if two $\{0, \pm 1\}$ sequences have zero NPAF. In order to compare the efficiency of the proposed formalism we analyze the involved algorithms that are needed for the computation of the NPAF. There are three sequential subalgorithms that take place:

- Representation (Sequences or support)
- Computation (Summations or differences)
- Verification (NPAF vectors or sorting lists)

To analyze an algorithm is to determine the number of steps required for the algorithm to execute. We will need the basic tools of asymptotic analysis, big-$\mathcal{O}$ notation taken from [11]. That is, to understand the growth rate of the algorithm, as the input increases. Following [11, 17], we will try to express the running time of the algorithm as a function of the operations needed. Since our problem, involves two parameters for a candidate pair $TCP(n, w)$ of sequences, the time complexity will be given as a function of both length n and weight w which are the input parameters in order to obtain a refined analysis. Thus, the efficiency of the algorithms will be compared through the respective time complexities.

It is obvious, that in order to compare the efficiency of the algorithms we must compare them for the same representation of input. In our case, the given problem must either be two candidate $TCP(n, w)$ or support sets of two sequences. When given candidate $TCP(n, w)$ we will refer to our problem, as the *sequence problem* while if support sets are provided we state it as the *support problem*. We note that, we will not deal with matters of arithmetic or bit-size complexity and we shall count all comparisons or additive and multiplicative operators the same.

We give an outline of two algorithms which transform the representation input from one form to another, i.e. the sequence problem to support problem and vice-versa.

Algorithm 1 Sequence to Support Algorithm - `SEQ2SUP`

```
  function Seq2Sup(A)
Require: A is a {0, ±1} sequence of length n
    j ← 0
    k ← 0
    n ← |A|
    for i ← 1 to n do
      if A[i] = 1 then
        POS(A)[j] ← i
        j ← j + 1
      else if A[i] = −1 then
        NEG(A)[k] ← i
        k ← k + 1
      end if
    end for
  return (POS(A), NEG(A))
  end function
```

We derive an asymptotic estimation of the worst case algorithm 1 runs, thus we are interested in providing an upper bound for the worst possible input n (worst-case analysis, [11, 17]) .

Lemma 3. *The Sequence to Support algorithm uses* $T_{SEQ2SUP}(n) = 4n + \mathcal{O}(1)$ *operations and runs in* $\mathcal{O}(n)$ *time.*

Proof. For each step we can have at most 2 comparisons and 2 variable assignments. Thus, they may occur a total of $T_{SEQ2SUP}(n) = 3 + 4n$ operations. □

In a similar fashion, we give algorithm 2 to derive the sequences from given support sets.

Algorithm 2 Support to Sequence Algorithm - `SUP2SEQ`

```
  procedure Sup2Seq(SUP(A))
Require: SUP(A) is a set of integer values of length ≤ w
    w ← max |SUP(A)|
    for i ← 1 to w do
      if SUP(A)[i] < 0 then
        A[SUP(A)[i]] ← −1
      else if SUP(A)[i] > 0 then
        A[SUP(A)[i]] ← 1
      else
        A[i] ← 0
      end if
    end for
  return A
  end procedure
```

Lemma 4. *The Support to Sequence algorithm uses* $T_{SUP2SEQ}(w) = 3w + \mathcal{O}(1)$ *operations and runs in* $\mathcal{O}(w)$ *time.*

Proof. For each step we can have at most 2 comparisons and 1 variable assignment. Thus, they may occur a total of $T_{SUP2SEQ}(w) = 1 + 3w$ operations. We note that the running time is a function of the weight w of the sequence. □

Moving to the computation phase, we give algorithm 3 which computes the NPAF vector form of a sequence as given by relation (1).

Algorithm 3 NPAF Vector Algorithm - `NPAFVEC`

```
  procedure NPafVec(A)
Require: A is finite sequence of length n
    for s ← 1 to n − 1 do
      NPAF(A)[s] ← NPafS(A, s)
    end for
  return NPAF(A)
  end procedure

  procedure NPafS(A,s)                    ▷ NPafS computes the NPAF of A in s
Require: A is finite sequence of length n
    N_A ← 0
    n ← |A|
    for i ← 1 to n − s do
      N_A ← N_A + A[i] ∗ A[i + s]
    end for
  return N_A
  end procedure
```

Lemma 5. *The NPAF Vector algorithm uses* $T_{NPAFVEC}(n) = 2n^2 + \mathcal{O}(n)$ *operations and runs in* $\mathcal{O}(n^2)$ *time.*

Proof. We first calculate the running time function $T_{N_A(s)}(n)$ of the NPAF of A in s. By counting 2 additions, 1 multiplication and 1 variable assignment we have that 4 operations occur in each one of the $n - s$ steps. Therefore $T_{N_A(s)}(n) = 4(n - s) + 2$ and depends on the shift s. Since $N_A(s)$ is called in the main body of the NPAFVEC algorithm, we need $\mathcal{O}(n)$ memory to read it as input since the input length of the sequence is n. We compute the running time of the entire algorithm to be $T_{NPAFVEC}(n) = \sum_{s=1}^{n-1} T_{N_A(s)}(n) + \mathcal{O}(n) = \sum_{s=1}^{n-1}(4(n-s)+2) + \mathcal{O}(n) = 2n^2 - 2 + \mathcal{O}(n) = 2n^2 + \mathcal{O}(1) + \mathcal{O}(n) = 2n^2 + \mathcal{O}(n)$.
□

As before, we give algorithm 4 to compute the NPAF defined on the support of a sequence A, defined by the signed differences $D^+_{A,1}, D^-_{A,1}$ and its cross differences $C^{\rightrightarrows}_{A,1} = D^{\pm}_{A,1} \uplus D^{\mp}_{A,1}$.

Algorithm 4 NPAF Support Algorithm - `NPAFSUP`

function NPafSup($POS(A), NEG(A), n$)
Require: $(POS(A), NEG(A))$ are sets of positive values of length $\leq w$
  $w^+ \leftarrow \max|POS(A)|$
  $w^- \leftarrow \max|NEG(A)|$
  for $x \leftarrow 1$ **to** $w^+ - 1$ **do** ▷ Construction of the multiset $D^+_{A,1}$
    for $y \leftarrow x+1$ **to** w^+ **do**
      if $POS(A)[x] > POS(A)[y]$ **then**
        $D^+_{A,1} \leftarrow POS(A)[x] - POS(A)[y]$
      end if
    end for
  end for
  for $x \leftarrow 1$ **to** $w^- - 1$ **do** ▷ Construction of the multiset $D^-_{A,1}$
    for $y \leftarrow x+1$ **to** w^- **do**
      if $NEG(A)[x] > NEG(A)[y]$ **then**
        $D^-_{A,1} \leftarrow NEG(A)[x] - NEG(A)[y]$
      end if
    end for
  end for
  for $x \leftarrow 1$ **to** w^+ **do** ▷ Construction of the multiset $D^{\pm}_{A,1}$
    for $y \leftarrow 1$ **to** w^- **do**
      if $POS(A)[x] > NEG(A)[y]$ **then**
        $D^{\pm}_{A,1} \leftarrow POS(A)[x] - NEG(A)[y]$
      end if
    end for
  end for
  for $x \leftarrow 1$ **to** w^- **do** ▷ Construction of the multiset $D^{\mp}_{A,1}$
    for $y \leftarrow 1$ **to** w^+ **do**
      if $NEG(A)[x] > POS(A)[y]$ **then**
        $D^{\mp}_{A,1} \leftarrow NEG(A)[x] - POS(A)[y]$
      end if
    end for
  end for
  $C^{\rightrightarrows}_{A,1} \leftarrow$ Join$(D^{\pm}_{A,1}, D^{\mp}_{A,1})$ ▷ Construction of the multiset $C^{\rightrightarrows}_{A,1}$
return $(D^+_{A,1}, D^-_{A,1}, C^{\rightrightarrows}_{A,1})$
end function

Lemma 6. *The NPAF Support algorithm uses* $T_{NPAFSUP}(w) = \frac{9}{2}w^2 + \mathcal{O}(w)$ *operations and runs in* $\mathcal{O}(w^2)$ *time.*

Proof. We note that $w^+, w^- \leq w$, i.e. the cardinalities of $POS(A), NEG(A)$ are bounded from above by the weight w of the sequence A, since $w^+ + w^- = w$. Thus, since we are interested in the worst-case we will consider the maximum number of loops for each one of the multisets constructed. We now calculate the running time function for each one of the multisets constructed by the NPAFSUP algorithm as follows:

- For each one loop in the construction of the multiset $D^+_{A,1}$ we have 1 comparison, 1 substraction and 1 variable assignment. Since the number

of loops are $\frac{w^+(w^+-1)}{2}$ the total number of operations that occur are $\frac{3w^+(w^+-1)}{2}$.

- By essentially the same argument for $\frac{w^-(w^--1)}{2}$ loops that are performed in the construction of the multiset $D^-_{A,1}$ we have another $\frac{3w^-(w^--1)}{2}$ operations.
- For the multisets $D^{\pm}_{A,1}, D^{\mp}_{A,1}$ we have $3w^+w^-$ operations each, since the same type of operations occur in each one of the w^+w^- loops.

We consider the adjoining of elements in the construction of $C^{\rightrightarrows}_{A,1}$ to consume $\mathcal{O}(w)$ memory. Therefore the total running time function of the PAFSUP algorithm is given by $T_{PAFSUP}(w) = \frac{3}{2}(w^+(w^+-1)+w^-(w^--1))+6w^+w^-+\mathcal{O}(w) = \frac{3}{2}(w^2-w)+3w^+w^-+\mathcal{O}(w) = \frac{9}{2}w^2-\frac{3w}{2}+\mathcal{O}(w) = \frac{9}{2}w^2+\mathcal{O}(w)$ operations. We note, that again as in the case of SUP2SEQ algorithm, the NPAFSUP algorithm depends only on the weight w of the sequence. Therefore, we have proven in addition a previous statement that our formalism depends only on the nonzero elements of a sequence. □

We now proceed to the last phase of verification, that is for given proper representation of sequences or support sets to be able to check (verify) that the sequences are complementary or in accordance with Lemma 2. We begin, with algorithm 5 that checks for zero NPAF in sequence form. We note that, this algorithm needs two sequences as its input.

Algorithm 5 NPAF Vector Verification Algorithm - `NPAFVECVER`

```
  procedure NPafVecVer(NPAF(A), NPAF(B))
Require: NPAF(A), NPAF(B) are integer arrays of length n − 1
    for i ← 1 to n − 1 do
      if NPAF(A)[i] = −NPAF(B)[i] then
        bool ← true
      else
        bool ← false
        break
      end if
    end for
  return (bool)
  end procedure
```

Lemma 7. *The NPAF Vector Verification algorithm uses* $T_{NPAFVECVER}(n) = 2n + \mathcal{O}(1)$ *operations and runs in* $\mathcal{O}(n)$ *time.*

Proof. There can be at most $n-1$ comparisons and $n-1$ boolean variable assignments. Thus, they may occur a total of $T_{NPAFVECVER}(n) = 2n-2 = 2n+\mathcal{O}(1)$ operations. □

Finally, we give algorithm 6 that is a direct implementation of Lemma 2. As already discussed in the previous Section, the verification for complementary

sequences in this case is reduced to a comparison sort problem. Though, there are a number of sorting and comparison sorting algorithms, see for example, [16], we will consider the cost of running a sorting routine as "black box" to be $n \log n$ operations, running time of $\mathcal{O}(n \log n)$ and $\mathcal{O}(n)$ memory is required in order to sort n records, i.e. an array of length n, in the worst-case. Such sorting routines are merge sort, heapsort, introsort and others [16].

Algorithm 6 NPAF Support Verification Algorithm - `NPAFSUPVER`

procedure NPafVecVer($D^+_{A,1}, D^-_{A,1}, C^{\rightrightarrows}_{A,1}, D^+_{B,1}, D^-_{B,1}, C^{\rightrightarrows}_{B,1}$)
Require: $D^+_{A,1}, D^-_{A,1}, C^{\rightrightarrows}_{A,1}, D^+_{B,1}, D^-_{B,1}, C^{\rightrightarrows}_{1,2}$ are arrays of length $\leq w^2$
  if $(|D^+_{A,1}| + |D^-_{A,1}| + |D^+_{B,1}| + |D^-_{B,1}|) \neq (|C^{\rightrightarrows}_{A,1}| + |C^{\rightrightarrows}_{B,1}|)$ **then**
    $bool \leftarrow false$ ▷ The lists that are going to be compared must be of equal length
  else
    $maxlen \leftarrow \max(|C^{\rightrightarrows}_{A,1}| + |C^{\rightrightarrows}_{B,1}|)$
    $lhs \leftarrow$ Sort($[D^+_{A,1}, D^-_{A,1}, D^+_{B,1}, D^-_{B,1}]$)
    $rhs \leftarrow$ Sort($[C^{\rightrightarrows}_{A,1}, D^{\rightrightarrows}_{B,1}]$)
    for $i \leftarrow 1$ **to** $maxlen$ **do**
      if $lhs[i] = rhs[i]$ **then**
        $bool \leftarrow true$
      else
        $bool \leftarrow false$
        break
      end if
    end for
  end if
return ($bool$)
end procedure

Lemma 8. *The NPAF Support Verification algorithm uses* $T_{NPAFSUPVER}(w) = 4w^2 \log w + \mathcal{O}(w^2)$ *operations and runs in* $\mathcal{O}(w^2 \log w)$ *time.*

Proof. The maximum size of the multisets is of $\mathcal{O}(w^2)$ where w is the cardinality of the support of the represented sequence, since the totality of the differences is bounded by w^2. It is clear that the multisets $(D^+_{A,1} \uplus D^-_{A,1})\&(D^+_{B,1} \uplus D^-_{B,1})$ and $C^{\rightrightarrows}_{A,1}\&C^{\rightrightarrows}_{B,1}$ must be of equal length in order to be sorted and compared as seen in Lemma 2. This check costs 1 comparison and 4 additions. Assuming all multisets to be of maximum length $\mathcal{O}(w^2)$, it is needed $6 \cdot \mathcal{O}(w^2) = \mathcal{O}(w^2)$ memory for the algorithm to adjoin the elements in the lists that are going to be sorted. We consider this memory reservation suffices, for input in the sort routines. The cost for sorting lists of length w^2 two times is $2w^2 \log w^2 = 4w^2 \log w$ operations. Finally we perform at most $2w^2$ comparisons and boolean variable assignments. Therefore, the total operations that may performed by the algorithm is $T_{NPAFSUPVER}(w) = 5 + \mathcal{O}(w^2) + 4w^2 \log w + 4w^2 = 4w^2 \log w + \mathcal{O}(w^2)$. Once again, we note that the NPAFSUPVER algorithm depends only on the weight w of the candidate $TCP(n, w)$ sequences. □

4.1. Worst-case analysis for the NPAF of two ternary sequences. At a first glance, we see that the SUP2SEQ representation uses less operations that its inverse algorithm SEQ2SUP even though their asymptotic growth rate is the same since $w < n$. This is a first indication that the support representation is more efficient to decide for candidate $TCP(n, w)$ to have zero NPAF. However, so far this is not the case met in the theory of complementary sequences. We revise the three phases needed for the computation of NPAF of two sequences in algorithmic notion:

- Sequence to Support and Support to Sequence algoritmhs
- NPAF Vector and NPAF Support algorithms
- NPAF Vector Verification and NPAF Support Verification algorithms

We now compare the performance of the algorithms given previously, for the two cases of the sequence and support problem. In the following Tables, an algorithmic procedure is given in order to decide if two sequences have zero NPAF, having as sequential steps the previous algorithms. We first begin, with the sequence problem, that is when candidate $TCP(n, w)$ are given. In this case, there is no need to transform the input as this is already in sequence form.

Phase	Algorithm	Operations / Memory	Running Time
Computation	NPAFVEC	$4n^2 + \mathcal{O}(n)$ / $\mathcal{O}(n)$	$\mathcal{O}(n^2)$
Verification	NPAFVECVER	$2n + O(1)$ / $\mathcal{O}(n)$	$\mathcal{O}(n)$
Total Cost	-	$4n^2 + \mathcal{O}(n)$ / $\mathcal{O}(n)$	$\mathcal{O}(n^2)$

We proceed, with the support problem, that is when given support sets for the candidate sequences. As before, there is no need to transform the input as this is already in set form.

Phase	Algorithm	Operations / Memory	Running Time
Computation	NPAFSUP	$9w^2 + \mathcal{O}(w)$ / $\mathcal{O}(w)$	$\mathcal{O}(w^2)$
Verification	NPAFSUPVER	$4w^2 \log w + \mathcal{O}(w^2)$ / $\mathcal{O}(w^2)$	$\mathcal{O}(w^2 \log w)$
Total Cost	-	$4w^2 \log w + \mathcal{O}(w^2)$ / $\mathcal{O}(w^2)$	$\mathcal{O}(w^2 \log w)$

We deduce, that for the first case the complexity, $T_{SEQ}(n)$, is quadratic with respect to the length n, while in the second case the complexity, $T_{SUP}(w)$, is logquadratic with respect to the weight w. We compare our formalism with the standard method of the computation of NPAF, in the same representation of input, by invoking the representation algorithms, and give the time functions in terms of n and w for a more refined asymptotic analysis [11].

(1) For the sequence problem we have, $T_{SEQ}(n) = 4n^2 + \mathcal{O}(n)$ and $2 \cdot T_{SEQ2SUP}(n) + T_{SUP}(w) = 8n + 4w^2 \log w + \mathcal{O}(w^2)$. Thus, our formalism performs $T(n, w) = 8n + 4w^2 \log w + \mathcal{O}(w^2)$ operations and runs in $\mathcal{O}(n + w^2 \log w)$ time.

(2) For the support problem we have, $2 \cdot T_{SUP2SEQ}(w) + T_{SEQ}(n) = 6w + 4n^2 + \mathcal{O}(n)$ and $T_{SUP}(w) = 4w^2 \log w + \mathcal{O}(w^2)$. Since w can be at most $2n$ we have $T_{SUP2SEQ}(w) + T_{SEQ}(n)$ is of $\mathcal{O}(n^2)$.

We are interested in how large can the weight w be with respect to the length n such as our formalism provides a faster algorithm to decide for two sequences to have zero NPAF.

Criterion. For candidate $TCP(n, w)$ sequences the following cases are efficient for the new formalism (when compared to $T_{SEQ}(n) = \mathcal{O}(n^2)$ time):

(1) If $n > w^2 \log w$, then the formalism can be executed in $\mathcal{O}(n)$ time, and thus is optimal.
(2) If $n > w^2$, then the formalism can be executed in $\mathcal{O}(n \log w)$ time.
(3) If $n > w \log w$, then the formalism can be executed in $\mathcal{O}(nw)$ time.

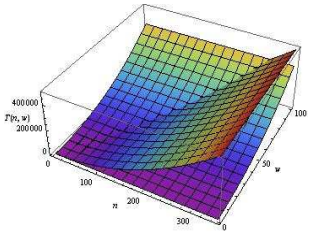
One might ask whether there are potential classes of $TCP(n, w)$ such as the previous criterion is applicable. In this section, we demonstrate that for sequences of small weight the new formalism is efficient.

However, in real-time computing, asymptotic analysis offers an approximation of how fast the running time grows, if the input increases. In practice, it is more convenient to know more than the leading terms in the running time functions [11]. Thus, we recalculate the number of operations that occur in algorithms 3, 5 and in algorithms 1, 4, 6 ignoring $\mathcal{O}(1)$ operations, since this is a result of an actual implementation and may varies from case to case.

Therefore, we have $T_{SEQ}(n) = 2 \cdot T_{NPAFVEC}(n) + T_{NPAFVECVER}(n) = 4n^2 + 2n$ operations in contrast with $T_{SUP}(n, w) = 2 \cdot T_{SEQ2SUP}(n) + 2 \cdot T_{NPAFSUP}(w) + T_{NPAFSUPVER}(w) = 8n + 9w^2 - 3w + 4w^2 + 4w^2 \log w = 8n + 13w^2 - 3w + 4w^2 \log w$. Thus, our formalism runs faster when,

$$T_{SEQ}(n) > T_{SUP}(n, w) \Leftrightarrow 4n^2 - 6n + w(4w \log w + 13w - 3) > 0 \qquad (4)$$

It is clear, that for fixed small w as the length n increases, our formalism runs faster. In particular when $w \lesssim n/3$ the implementation of the formalism is more efficient, i.e. when the totality of zeros which appears in two $\{0, \pm 1\}$ sequences are $2/3$ the length of the sequences. We note, that is in accordance with the theory of TCPs [6]; once a weight is established this way for some length, this length can be increased arbitrarily. Thus, weight is fundamental, and length is arbitrarily large. In the following graph we plot the time complexity functions of $T_{SEQ}(n)$ versus our formalism $T_{SUP}(n, w)$. The resulting plot justifies our previous results, and we propose the new formalism for ternary sequences when $w \lesssim n/3$.



Acknowledgments

The research of the second author was supported by a scholarship awarded by the Secretariat of the Research Committee of National Technical University of Athens.

References

1. K. T. Arasu and T. A. Gulliver, *Self-dual codes over $\mathbb{F}_p$ and weighing matrices*, IEEE Trans. Inform. Theory **47** (2001), 2051–2055.
2. S. Boyd, *Multitone signals with low crest factor*, IEEE Trans. Circuits Systems **33** (1986), 1018–1022.
3. T. J. Cox, J. A. S. Angus and P. DAntonio, *Ternary and quadriphase sequence diffusers*, J. Acoust. Soc. Am. **119** (2006), 310–319.
4. R. Craigen, *Weighing matrices and conference matrices*, in The CRC Handbook of Combinatorial Designs, (Eds. C. J. Colbourn and J. H. Dinitz), CRC Press, Boca Raton, Fla., 1996, pp. 496–504.
5. R. Craigen and H. Kharaghani, *Orthogonal designs*, in Handbook of Combinatorial Designs, (Eds. C.J. Colbourn and J.H. Dinitz), 2nd ed. CRC Press, Boca Raton, Fla., 2006, pp. 290–306.
6. R. Craigen and C. Koukouvinos, *A theory of ternary complementary pairs*, J. Combin. Theory Ser. A **96** (2001), 358–375.
7. R. Craigen, S. Georgiou, W. Gibson and C. Koukouvinos, *Further explorations into ternary complementary pairs*, J. Combin. Theory Ser. A **113** (2006), 952–965.
8. E. Fenimore and T. Cannon, *Coded aperture imaging with uniformly redundant arrays*, Appl. Optics **17** (1978), 337–347.
9. A. V. Geramita and J. Seberry, *Orthogonal designs. Quadratic forms and Hadamard matrices*, Lecture Notes in Pure and Applied Mathematics, 45, Marcel Dekker Inc. New York, 1979.
10. S. Golomb and H. Taylor, *Two-dimensional synchronization patterns for minimum ambiguity*, IEEE Trans. Inform. Theory **28** (1982), 600–604.
11. D. H. Greene and D. E. Knuth, *Mathematics for the analysis of algorithms*, 3rd edition, Modern Birkhauser Classics, Birkhauser, Boston, 2008.
12. J. Hersheya and R. Yarlagadda, *Two-dimensional synchronisation*, Electron. Lett. **19** (1983), 801–803.
13. H. Hotelling, *Some improvements in weighing and other experimental techniques*, Ann. Math. Stat. **16** (1944), 294–300.
14. H. Kharaghani and C. Koukouvinos, *Complementary, Base and Turyn Sequences*, in Handbook of Combinatorial Designs, (Eds. C.J. Colbourn and J.H. Dinitz), 2nd ed. Chapman and Hall/CRC Press, Boca Raton, Fla., 2006, pp. 317-321.
15. D. E. Knuth, *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, 3rd edition, Addison- Wesley Series in Computer Science and Information Processing, Addison-Wesley Publishing Co., Mass.- London-Don Mills, 1998.
16. D. E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching*, 3rd edition, Addison- Wesley Series in Computer Science and Information Processing, Addison-Wesley Publishing Co., Mass.- London-Don Mills, 1998.
17. D. E. Knuth, *Selected Papers on Analysis of Algorithms*, Centre for the Study of Language and Information - CSLI Lecture Notes, no. 102, Stanford, California, 2000.
18. C. Koukouvinos, Sequences with Zero Autocorrelation, in *The CRC Handbook of Combinatorial Designs*, (Eds. C. J. Colbourn and J. H. Dinitz), CRC Press, 1996, pp. 452-456.
19. C. Koukouvinos and J. Seberry, *Weighing matrices and their applications*, J. Statist. Plann. Inference **62** (1997), 91–101.

20. C. Koukouvinos and J. Seberry, *New weighing matrices and orthogonal designs constructed using two sequences with zero autocorrelation function-a review*, J. Statist. Plann. Inference **81** (1999), 153–182.
21. D. Raghavarao, *Constructions and Combinatorial Problems in Design of Experiments*, Wiley Series in Probability and Statistics, John Wiley and Sons, New York-Sydney-London, 1971.
22. G. Weathers and E. M. Holiday, *Group-complementary array coding for radar clutter rejection*, IEEE Transaction on Aerospace and Electronic Systems **19** (1983), 369–379.
23. B. Schneier, *Applied Cryptography*, 2nd edition, John Wiley and Sons, New York, 1996.
24. J. R. Seberry, B. J. Wysocki and T. A. Wysocki, *On a use of Golay sequences for asynchronous DS CDMA applications*, In Advanced Digital Signal Processing for Communication Systems, chapter 9, pp. 182–196, Kluwer Academic Pulbishers, Boston, Dordrecht, London, 2002.
25. J. Seberry Wallis, *On supplementary difference sets*, Aequationes Math. **8** (1972), 242–257.
26. J. Seberry Wallis, *A note on supplementary difference sets*, Aequationes Math. **10** (1974), 46–49.
27. M. Sipser, *Introduction to the Theory of Computation*, 2nd edition, International edition, Thomson Learning Inc., Boston, Massachusetts, 2006.
28. D. R. Stinson, *Cryptography : Theory and Practice*, CRC Press on Discrete Mathematics and Its Applications, Boca Raton, 1995.

Christos Koukouvinos is a Professor at the National Technical University of Athens, Department of Mathematics. He holds a Bachelor in Mathematics (1983) and a PhD (1988) in Statistics both obtained from the University of Thessaloniki. He is the author of numerous papers in the field of Statistics and Combinatorics and on the Editorial board of six related journals. He was awarded the prestigious Hall Medal of the ICA in 1996. He is a Fellow of the ICA. His research interests include statistical experimental and optimal designs, combinatorial designs and coding theory.

Department of Mathematics, National Technical University of Athens, Zografou 15773, Athens, Greece.
e-mail: ckoukouv@math.ntua.gr

Dimitris E. Simos received his Bachelor in Mathematics (2006) from the University of Athens, and MSc in Applied Mathematical Sciences (2007) from the National Technical University of Athens. He is currently a PhD candidate at National Technical University of Athens since 2008. His research interests are combinatorial designs, coding theory and cryptography.

Department of Mathematics, National Technical University of Athens, Zografou 15773, Athens, Greece.
e-mail: dsimos@math.ntua.gr