Algorithms and Metaheuristics for Combinatorial Matrices

Ilias S. Kotsireas

Contents

Abstract

Combinatorial matrices are matrices that satisfy certain combinatorial properties and typically give rise to extremely challenging search problems with thousands of variables. In this chapter we present a survey of some recent algorithms to search for some kinds of combinatorial matrices, with an emphasis to algorithms within the realm of optimization and metaheuristics. It is to be noted that for most kinds of combinatorial matrices there are several known infinite classes

I.S. Kotsireas
Department of Physics and Computer Science, Wilfrid Laurier University, Waterloo, ON, Canada
e-mail: ikotsire@wlu.ca

in the literature, but these infinite classes do not suffice to cover the entire spectra of possible orders of these matrices, therefore it is necessary to resort to computational and meta-heuristic algorithms.

1 Introduction

The search for combinatorial matrices and their associated designs has been a fertile testing ground for algorithm designers for decades. These extremely challenging combinatorial problems have been tackled with algorithms using concepts and techniques from discrete mathematics, number theory, linear algebra, group theory, optimization, and metaheuristics. This chapter will survey some recent algorithms to search for combinatorial matrices, with an emphasis to algorithms within the realm of optimization and metaheuristics. The hope is that this chapter will arouse the interest of algorithm designers from various areas in order to invent new formalisms and new algorithms to search efficiently for combinatorial matrices. There are several open problems in the broad area of combinatorial matrices, and it is clear that new ideas are required to solve them. The existing algorithms continue to yield new results, especially in conjunction with the use of parallel programming techniques, but will still eventually reach a point of saturation, beyond which they will probably not be of much use. Therefore, cross-fertilization of research areas is necessary for producing new results in the search for new combinatorial matrices, at both the theoretical and the practical level.

Combinatorial matrices are defined as matrices that possess some combinatorial properties, such as prescribed row/column sums and special structure described in terms of circulant matrices. The research area of combinatorial matrices has been developed in a systematic manner over the last 50 years in the four books [10, 11, 48, 88]. It is worthwhile to point out that the unifying concept of combinatorial matrices includes well-known and widely studied categories of matrices, such as adjacency and incidence matrices of graphs, Hadamard matrices, and doubly stochastic matrices.

In addition, there is a number of more recent books that contain useful information and new developments in the area of combinatorial matrices and the closely related area of design theory. The book [52] (see also the update [53]) sheds considerable light in the cocyclic (i.e., group cohomological) aspects of certain kinds of combinatorial matrices and is the only systematic book-size treatment of this topic. The book [94] features a very readable exposition of design theory with emphasis on bent functions and coding theory aspects. The book [19] offers an alternative algebraic foundation of design theory. The book [54] places its emphasis on symmetric designs and their properties.

2 Preliminaries and Notations

A wide class of combinatorial matrices (or the relevant sequences) can be defined via the concepts of the periodic and aperiodic (or nonperiodic) autocorrelation functions associated to a finite sequence. These two concepts have been invented and studied widely within the engineering community, and their importance has been recognized in the combinatorics community as well, especially in connection with several kinds of combinatorial matrices.

All sequences in this chapter will be finite and will have elements either from $\{-1, +1\}$ (binary sequences) or from $\{-1, 0, +1\}$ (ternary sequences).

Definition 1 The periodic autocorrelation function associated to a finite sequence $A = [a_1, \ldots, a_n]$ of length n is defined as

$$P_A(i) = \sum_{k=1}^{n} a_k a_{k+i}, \; i = 0, \ldots, n-1,$$

where $k + i$ is taken modulo n, when $k + i > n$.

Definition 2 The aperiodic (or nonperiodic) autocorrelation function of a sequence $[a_1, \ldots, a_n]$ of length n is defined as

$$N_A(i) = \sum_{k=1}^{n-i} a_k a_{k+i}, \; i = 0, \ldots, n-1,$$

where $k + i$ is taken modulo n, when $k + i > n$.

Definitions 1 and 2 are best clarified with an example.

Example 1 For a finite sequence of length $n = 7$, $A = [a_1, \ldots, a_7]$, we have

$$\begin{aligned}
P_A(0) &= a_1^2 + a_2{}^2 + a_3{}^2 + a_4{}^2 + a_5{}^2 + a_6{}^2 + a_7{}^2\\
P_A(1) &= a_1a_2 + a_2a_3 + a_3a_4 + a_4a_5 + a_5a_6 + a_6a_7 + a_7a_1\\
P_A(2) &= a_1a_3 + a_2a_4 + a_3a_5 + a_4a_6 + a_5a_7 + a_6a_1 + a_7a_2\\
P_A(3) &= a_1a_4 + a_2a_5 + a_3a_6 + a_4a_7 + a_5a_1 + a_6a_2 + a_7a_3\\
P_A(4) &= a_1a_4 + a_2a_5 + a_3a_6 + a_4a_7 + a_5a_1 + a_6a_2 + a_7a_3\\
P_A(5) &= a_1a_3 + a_2a_4 + a_3a_5 + a_4a_6 + a_5a_7 + a_6a_1 + a_7a_2\\
P_A(6) &= a_1a_2 + a_2a_3 + a_3a_4 + a_4a_5 + a_5a_6 + a_6a_7 + a_7a_1
\end{aligned}$$

$$\begin{aligned}
N_A(0) &= a_1{}^2 + a_2{}^2 + a_3{}^2 + a_4{}^2 + a_5{}^2 + a_6{}^2 + a_7{}^2\\
N_A(1) &= a_1a_2 + a_2a_3 + a_3a_4 + a_4a_5 + a_5a_6 + a_6a_7\\
N_A(2) &= a_1a_3 + a_2a_4 + a_3a_5 + a_4a_6 + a_5a_7\\
N_A(3) &= a_1a_4 + a_2a_5 + a_3a_6 + a_4a_7\\
N_A(4) &= a_1a_5 + a_2a_6 + a_3a_7\\
N_A(5) &= a_1a_6 + a_2a_7\\
N_A(6) &= a_1a_7.
\end{aligned}$$

Definition 3 Two finite sequences $[a_1, \ldots, a_n]$ and $[b_1, \ldots, b_n]$ of length n each are said to have constant periodic autocorrelation if there is a constant c such that

$$P_A(i) + P_B(i) = c, \quad i = 1, \ldots, n-1.$$

Definition 4 Two sequences $[a_1, \ldots, a_n]$ and $[b_1, \ldots, b_n]$ of length n each are said to have constant aperiodic autocorrelation if there is a constant c such that

$$N_A(i) + N_B(i) = c, \quad i = 1, \ldots, n-1.$$

Note that the index $i = 0$ is omitted from Definitions 3 and 4, because of the property $P_A(0) = N_A(0) = \sum_{j=1}^{n} a_j^2$. Also note that when writing out binary and ternary sequences, it is customary in the combinatorial literature to denote -1 by $-$, zero by 0, and $+1$ by $+$, and this is the convention adopted in the current chapter. Also note that Definitions 3 and 4 can be extended to more than two sequences. Definitions 3 and 4 are best illustrated with an example.

Example 2 The following binary sequences with $n = 26$ have constant aperiodic autocorrelation with $c = 0$:

$$\begin{array}{c} + + + + - + + - - + - + - + - - + - + + + - - + + + \\ + + + + - + + - - + - + + + + + - + - - - + + - - - \end{array}$$

The following ternary sequences with $n = 30$ have constant aperiodic autocorrelation with $c = 0$:

$$\begin{array}{c} + + - - - - - + - - 0000 + 0 + - + + + + - - + - + - + + \\ + + - - - - - + - - + + - - 0 - 0000 - - + + - + - + - - \end{array}$$

The following properties of the periodic and aperiodic autocorrelation functions can be observed (and verified) in Examples 1 and 2:

Property 1 (Symmetry)

$$P_A(i) = P_A(n-i), i = 0, 1, \ldots, n.$$

Property 2 (Complementarity)

$$P_A(i) = N_A(i) + N_A(n-i), i = 0, 1, \ldots, n.$$

Property 3 (Second Elementary Symmetric Function)

$$P_A(1) + \cdots + P_A(n-1) = 2\, e_2(a_1, \ldots, a_n)$$

$$N_A(1) + \cdots + N_A(n-1) = e_2(a_1, \ldots, a_n)$$

where $e_2(a_1, \ldots, a_n) = \sum_{1 \le i < j \le n} a_i a_j$ is the second elementary symmetric function in the n variables $a_1, \ldots, a_n$.

The proof of the three properties stated above is a straightforward application of Definitions 1 and 2 and is left to the reader.

The complementarity property of periodic and aperiodic autocorrelation functions implies that:

Property 4 If two sequences have constant aperiodic autocorrelation, then they must also have constant periodic autocorrelation.

Therefore, the two pairs of sequences in Example 2 also have constant periodic autocorrelation with $c = 0$.

The converse of the above property does not hold, and counterexamples may be easily found.

The concept of a circulant matrix is also important in connection with several kinds of combinatorial matrices and sequences.

Definition 5 A $n \times n$ matrix $C(A)$ is called circulant if every row (except the first) is obtained by the previous row by a right cyclic shift by one. In particular,

$$C(A) = \begin{bmatrix} a_1 & a_2 & \ldots & a_{n-1} & a_n \\ a_n & a_1 & \ldots & a_{n-2} & a_{n-1} \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ a_3 & a_4 & \ldots & a_1 & a_2 \\ a_2 & a_3 & \ldots & a_n & a_1 . \end{bmatrix}$$

The relationship between circulant matrices and periodic autocorrelation is described by the following property, whose proof is also a straightforward application of Definitions 1 and 2.

Property 5 Consider a finite sequence $A = [a_1, \ldots, a_n]$ of length n and the circulant matrix $C(A)$ whose first row is equal to A. Then $P_A(i)$ is the inner product of the first row of $C(A)$ and the $i + 1$ row of $C(A)$.

It turns out that the concepts of periodic and aperiodic autocorrelation can serve as the unifying concepts needed to define several kinds of combinatorial matrices and sequences simultaneously. We summarize some of these kinds of combinatorial matrices and sequences in the next table.

Note that in Table 1, TCP stands for “ternary complementary pairs” and PCS stands for “periodic complementary sequences.” We refer the reader to the papers listed in Table 1 for the complete definitions of these combinatorial objects.

Table 1 Combinatorial matrices and sequences

Number/type of sequences	Defining property	Name	References
2 binary	aper. autoc. 0	Golay sequences	[8, 22, 36, 40, 41]
2 binary	per. autoc. 0	Hadamard matrices	[60]
2 binary	per. autoc. 2	D-optimal matrices	[29, 61]
2 binary	per. autoc. −2	Hadamard matrices	[63]
2 ternary	aper. autoc. 0	TCP	[18, 43]
2 ternary	per. autoc. 0	Weighing matrices	[64, 65]
3 binary	aper. autoc. const.	Normal sequences	[27]
4 binary	aper. autoc. 0	Base sequences	[26]
4 binary	aper. autoc. 0	Turyn-type sequences	[57]
4 ternary	aper. autoc. 0	T-sequences	[57]
2 . . . 12 binary	per. autoc. 0	PCS	[6, 28, 66]

In the remainder of this chapter, we focus on various kinds of algorithms that can be used to search efficiently for combinatorial matrices and sequences listed in Table 1, as well as any other such combinatorial objects that can be defined via periodic and aperiodic autocorrelation. In fact, we will concentrate on general algorithmic schemes that are applicable to all such combinatorial objects and will provide high-level descriptions for these algorithmic schemes.

3 Cyclotomy Algorithms

Cyclotomy algorithms to search for combinatorial matrices and sequences are based on the premise that certain combinatorial concepts called "supplementary difference sets" (abbreviated as SDS) can be constructed by taking unions of cyclotomic classes (or cosets) or group action orbits of a suitable subgroup of the group of invertible elements of the ring of integers modulo n, where n is a parameter referring to the length of the sequences. This powerful theme has been exploited by several authors at the theoretical and practical level over more than two decades.

The crucial concept of an SDS has been introduced by Jennifer Seberry 40 years ago [101, 102].

Definition 6 Let n be a positive integer and let $S_1, \ldots, S_k$ be subsets of Z_n with cardinalities $n_1, \ldots, n_k$, respectively. Let $T_i = \{(s_1^i - s_2^i) \mod n, (s_2^i - s_i^i) \mod n, s_1^i, s_2^i \in S_i\}$ be the multiset of all pairwise differences (taken modulo n) of all elements in S_i, for $i = 1, \ldots, k$. If the multiset $T_1 \cup \cdots \cup T_k$ is equal to λ copies of $\{1, \ldots, n-1\}$, then $S_1, \ldots, S_k$ form an SDS with parameters $k - \{n; n_1, \ldots, n_k; \lambda\}$.

A simple counting argument shows that existence of an SDS with parameters $k - \{n; n_1, \ldots, n_k; \lambda\}$ implies the following condition:

$$\sum_{i=1}^{k} n_i(n_i - 1) = \lambda(n-1).$$

Here is a toy example, where the SDS property can be verified by hand.

Example 3 Let $n = 13$ and $t = 3$. Then there are 4 cyclotomic classes: $S_1 = [1,3,9]$, $S_2 = [2,5,6]$, $S_3 = [4,10,12]$ and $S_4 = [7,8,11]$. It turns out that S_1 and S_2 form an SDS with parameters $2 - \{13; 3, 3; 1\}$ because (mod 13) we have

$$T_1 \cup T_2 = \{1-3, 1-9, 3-9, 3-1, 9-1, 9-3\} \cup \{2-5, 2-6, 5-6, 5-2, 6-2, 6-5\},$$

i.e.,

$$T_1 \cup T_2 = \{1,2,3,4,5,6,7,8,9,10,11,12\}.$$

Here is a non-toy example, where the SDS property needs to be verified by a computer program.

Example 4 Let $n = 323$ and $t = 3$. Then there are 4 cyclotomic classes C_1, C_2, C_3, C_4 with cardinalities $n_1 = 144$, $n_2 = 144$, $n_3 = 18$ and $n_4 = 16$. Then the sets $S_1 = C_1 \cup C_3$ and $S_2 = S_1$ form an SDS with parameters $2 - \{323; 162, 162; 162\}$.

The paper [103] states a general theorem that indicates conditions under which suitable unions of cyclotomic classes form an SDS with specific parameters. The paper [38] contains recent theoretical results and in particular gives two constructions of skew Hadamard difference sets in the additive groups of suitable finite fields by using unions of cyclotomic classes. The paper [39] gives two constructions of strongly regular Cayley graphs on finite fields by using unions of cyclotomic classes. The paper [97] explores the idea of blending a mathematical programming formalism with a cyclotomy-based approach; the resulting algorithm is quite promising and deserves further investigation. Chapter 3 of the PhD thesis [47] is a virtual treasure trove of cyclotomy-based algorithms and applications of such algorithms to several different kinds of combinatorial matrices and sequences. Some of the most spectacular successes of cyclotomy algorithms include the discovery of new orders of Hadamard and skew Hadamard matrices [21, 23–25] as well as D-optimal matrices [29]. These types of methods are bound to continue to produce new results.

4 String Sorting Algorithms

String sorting algorithms to search for combinatorial matrices and sequences are based on certain restrictions that the properties of constant periodic or aperiodic autocorrelation imply. These restrictions possess the important advantage that they can be used to decouple the (quadratic) equations in Definitions 3 and 4. In addition, one needs to carefully distinguish between periodic and aperiodic autocorrelation, because the corresponding restrictions are of a different nature.

4.1 Periodic Autocorrelation

The restrictions imposed by the constancy of periodic autocorrelation are described via the concept of power spectral density (PSD) associated to a finite sequence. We illustrate the main idea in the context of Hadamard matrices with two circulant cores, following [42, 63].

Let n be an odd positive integer and suppose that two binary sequences $A = [a_1, \ldots, a_n]$ and $B = [b_1, \ldots, b_n]$, both of length n, have constant periodic autocorrelation function -2, i.e.,

$$P_A(s) + P_B(s) = -2, \text{ for } s = 1, \ldots, \frac{n-1}{2}. \tag{1}$$

Definition 7 Let $\omega = e^{\frac{2\pi i}{n}}$ be a primitive n-th root of unity.

The discrete Fourier transform (DFT) of A is defined by

$$\mathrm{DFT}_A(s) = \sum_{k=1}^{n} a_k \omega^{ks}, \text{ for } s = 0, \ldots, n-1.$$

The power spectral density (PSD) of A is defined by

$$\mathrm{PSD}_A(s) = |\mathrm{DFT}_A(s)|^2, \text{ for } s = 0, \ldots, n-1.$$

Therefore, the PSD values are the squared magnitudes of the DFT values, i.e., $\mathrm{PSD}_A(s) = Re(\mathrm{DFT}_A(s))^2 + Im(\mathrm{DFT}_A(s))^2$. Note that the well-known properties $\omega^n = 1$ and $\mathrm{DFT}_A(n-s) = \mathrm{DFT}_A(s), s = 0, 1, \ldots, n-1$ are routinely used in order to perform DFT/PSD calculations efficiently.

It can be proved (see [42]) that property (Eq. 1) implies that

$$\mathrm{PSD}_A(s) + \mathrm{PSD}_B(s) = 2n + 2, \text{ for } s = 1, \ldots, \frac{n-1}{2}. \tag{2}$$

Example 5 Let $n = 55$ and consider the two sequences from [13]:

$$-++-++++++--+--++-+-++-+-++--+---+-++++--+--+---+-++--$$
$$+--+++---++-+-+-++---+-+++++++-++++----+-+---+---+++--$$

The above sequences have constant periodic autocorrelation -2 and can be used to construct a Hadamard matrix of order $2 \cdot 55 + 2 = 112$. The following table records the $\frac{55-1}{2} = 27$ PSD values for the above sequences, as well as their sums, which is equal to 112 for $s = 1, \ldots, 27$ as expected.

Table 2 PSD values for binary sequences with $n = 55$

s	$\mathrm{PSD}_A(s)$	$\mathrm{PSD}_B(s)$	$\mathrm{PSD}_A(s)+$ $\mathrm{PSD}_B(s)$
1	29.80737844157036985336002	82.19262155456297731046190	112
2	30.28944472490585995185313	81.71055527763585566645278	112
3	55.43998376085268510929794	56.56001623465057222621275	112
4	102.0320272601899733257176	9.967972741391460830458954	112
5	89.65677565516284911337367	22.34322434507586735097245	112
6	15.47369185908972215139240	96.52630813774760368151000	112
7	11.47686739850811259463546	100.5231326000467370829829	112
8	76.73731376355725457366036	35.26268623768915584169775	112
9	46.11225525013170242777989	65.88774474677226578858688	112
10	0.4008926865885775543776042	111.5991073136016861161094	112
11	38.11145618000341879780940	73.88854381999376144822824	112
12	17.46064389518599154629647	94.53935610256366875094971	112
13	34.87931253417465350073346	77.12068746596972533277206	112
14	82.62901343793391797610934	29.37098656225313882607487	112
15	93.37954082614625749821953	18.62045917360481914328922	112
16	60.59338387747255107616868	51.40661612647071355873921	112
17	92.66303952890268414319827	19.33696047140536416932549	112
18	106.5590520047682870981779	5.440947992051451856194138	112
19	105.3670134147016013500540	6.632986586404617855942329	112
20	79.93243752088631246467129	32.06756247928573785046143	112
21	87.07144169101030219209510	24.92855831006362765566586	112
22	73.88854381999762010973329	38.11145617999893554527716	112
23	11.57785413372743006711576	100.4221458663961229278652	112
24	38.63848171187360617293295	73.36151828958980649610761	112
25	16.63035331130720400371166	95.36964668866435905703407	112
26	108.5508478995728704749974	3.449152098190036553335345	112
27	6.640953414706377747434240	105.3590465799776576376609	112

It is instructive to examine Table 2 in order to see how Eq. (2) are materialized for the particular solution of Example 5.

4.1.1 The PSD Criterion

Equation (2) can be used to derive the so-called PSD criterion by first remarking that since the PSD values are nonnegative, if for some $s \in \left\{1, \ldots, \frac{n-1}{2}\right\}$ it turns out that $\mathrm{PSD}_A(s) > 2n + 2$ or $\mathrm{PSD}_B(s) > 2n + 2$, then the corresponding A or B candidate sequence can be safely discarded from the search.

4.1.2 A Subtle Point: Integer PSD Values

There is a subtle point regarding the computation of the PSD values that occur in string sorting algorithms, as well as other filtering-based algorithms to search for sequences using the PSD criterion. The subtlety comes from the fact that although in the vast majority of cases the PSD values are (positive) floating point numbers, there exist cases when the PSD values are (positive) integers. A particular case when this phenomenon occurs is described in the following result proved in [64].

Lemma 1 *Let n be an odd integer such that $n \equiv 0\,(mod\ 3)$ and let $m = \frac{n}{3}$. Let $\omega = e^{\frac{2\pi i}{n}} = \cos\left(\frac{2\pi}{n}\right) + i\sin\left(\frac{2\pi}{n}\right)$ the principal n-th root of unity. Let $[a_1, \ldots, a_n]$ be a sequence with elements from $\{-1, 0, +1\}$. Then we have that* $\mathrm{DFT}([a_1, \ldots, a_n], m)$ *can be evaluated explicitly in closed form and* $\mathrm{PSD}([a_1, \ldots, a_n], m)$ *is a nonnegative integer. The explicit evaluations are given by*

$$\mathrm{DFT}([a_1, \ldots, a_n], m) = \left(A_1 - \frac{1}{2}A_2 - \frac{1}{2}A_3\right) + \left(\frac{\sqrt{3}}{2}A_2 - \frac{\sqrt{3}}{2}A_3\right) i$$

$$\mathrm{PSD}([a_1, \ldots, a_n], m) = A_1^2 + A_2^2 + A_3^2 - A_1A_2 - A_1A_3 - A_2A_3$$

where

$$A_1 = \sum_{i=0}^{m-1} a_{3i+1}, \quad A_2 = \sum_{i=0}^{m-1} a_{3i+2}, \quad A_3 = \sum_{i=0}^{m-1} a_{3i+3}.$$

It is important to carefully account for the phenomenon described by Lemma 1 in any implementation of string sorting algorithms in order not to miss any solutions due to numerical round-off error. One way of doing this is to omit the $n/3$-th value from the construction of the string.

4.1.3 Bracelets and Necklaces

The symmetry property of periodic autocorrelation and PSD values under cyclic shifting must also be taken into account in order to avoid redundant computations for sequences that have the same periodic autocorrelation and PSD values. It turns out that the right combinatorial structures to deal with this symmetry property are bracelets and necklaces. The papers [89] and [90] and subsequent work by J. Sawada and his collaborators provide constant amortized time (CAT) algorithms to generate bracelets and necklaces, as well as extremely efficient C implementations of these algorithms.

4.2 Aperiodic Autocorrelation

The restrictions imposed by the constancy of periodic autocorrelation are described via the concept of a certain infinite class of functions associated to a finite sequence. We illustrate the main idea in the context of Turyn-type sequences, following [57].

Let n be an even positive integer and suppose that four binary sequences $X = [x_1, \ldots, x_n]$ and $Y = [y_1, \ldots, y_n]$, $Z = [z_1, \ldots, z_n]$ and $W = [w_1, \ldots, w_{n-1}]$ of lengths $n, n, n, n-1$ respectively, have constant aperiodic autocorrelation function in the sense that

$$N_X(s) + N_Y(s) + 2N_Z(s) + 2N_W(s) = 0, \text{ for } s = 1, \dots, n-1. \tag{3}$$

The sequences X, Y, Z, W satisfying (3) are called Turyn type.

We associate to a binary sequence X of length n, the periodic (of period 2π) nonnegative function

$$f_X(\theta) = N_X(0) + 2\sum_{j=1}^{n-1} N_X(j) \cos j\theta.$$

It can be proved (see [57]) that property (Eq. 3) implies that

$$f_X(\theta) + f_Y(\theta) + 2f_Z(\theta) + 2f_W(\theta) = 6n-2, \ \forall\theta. \tag{4}$$

4.2.1 Aperiodic Version of the PSD Criterion

Equation (4) can be used to derive the analog (an aperiodic version) of the PSD criterion in the aperiodic case. If for some value of θ it turns out that $f_X(\theta) > 6n-2$ or $f_Y(\theta) > 6n-2$, or $f_Z(\theta) > 3n-1$ or $f_W(\theta) > 3n-1$, then the corresponding X, Y, Z, W candidate sequence can be safely discarded from the search. In addition, if for some value of θ it turns out that $f_X(\theta) + f_Y(\theta) > 6n-2$, then the corresponding pair (X, Y) of candidates sequences can be safely discarded from the search. Similar conditions can be derived for other partial sums of the four values $f_X(\theta)$, $f_Y(\theta)$, $f_Z(\theta)$, $f_W(\theta)$ for arbitrary (but fixed) values of θ. The preprocessing of the sequences based on this aperiodic version of the PSD criterion is often carried out via a finite set of θ values, such as $\left\{\frac{j\pi}{500}, j = 1, \dots, 500\right\}$.

4.3 A General String Sorting Algorithmic Scheme

Based on the PSD criterion (periodic and aperiodic versions), a general string sorting algorithmic scheme can be defined as follows.

INPUT: lengths and types (binary/ternary) of two sequences A, B and the autocorrelation (periodic/aperiodic) property satisfied, value of PSD constant (or its aperiodic analog)

OUTPUT: sequences satisfying the input requirements (if any are found)

- Preprocess each one of the two sequences separately using the applicable version of the PSD criterion. In particular, discard all candidate sequences that violate the applicable version of the PSD criterion. If the set of all possible candidate sequences is too large to preprocess in its entirety, then use randomized methods to generate a representative part of this set.
- Encode candidate A-sequences by a string, namely, the concatenation of the integer parts of its PSD values, or the $f(\theta)$ values, for a small set of θ values.

Encode candidate B-sequences by a similar string, but taking the difference of the PSD (or $f(\theta)$ value) from the PSD constant (or its aperiodic analog).
- Sort the files containing the string encodings corresponding to each A-sequence and B-sequences separately. When the files containing the string encodings are too large to be sorted directly, this phase involves a distributed sorting algorithm.
- Look for identical strings in the two sorted files and output the corresponding solutions.

String sorting algorithms have been used in [?, 64] to compute several new weighing matrices of various weights constructed from two circulant cores, using the PSD criterion for periodic autocorrelation.

The ternary sequences with $n = 30$ in Example 2 have been computed by the author using a C implementation of a string sorting algorithm and the aperiodic version of the PSD criterion.

5 Genetic Algorithms

Genetic algorithms (GAs) form a powerful metaheuristic method that exploits algorithmic analogs of concepts from Darwin's theory of evolution to design efficient search methods. The theory of genetic algorithms is presented in a extremely readable way in the classical book [44]. Among the more recent systematic treatments of GAs, one can consult [87]. Genetic algorithms have been applied successfully to tackle a wide range of difficult problems across many application areas. A central concept in the theory of GAs is the "objective function." A judicious choice of an objective function is necessary in order to be able to apply GAs to solve a specific problem. Another important ingredient in the theory of GAs is the encoding of the solution space as a set of binary vectors. We will exemplify below how to formalize a problem of searching for combinatorial matrices as a GA problem, using binary sequences with constant periodic autocorrelation -2 as a case study, and following the exposition in [59].

Let n be an odd positive integer and suppose that two binary sequences $A = [a_1, \ldots, a_n]$ and $B = [b_1, \ldots, b_n]$, both of length n, have constant periodic autocorrelation function -2, as in (1). Since the $m = \frac{n-1}{2}$ Eq. (1) must be satisfied simultaneously, one can consider the following two types of objective functions arising naturally:

$$OF_1 = \mid P_A(1) + P_B(1) + 2 \mid + \ldots + \mid P_A(m) + P_B(m) + 2 \mid$$

and

$$OF_2 = (P_A(1) + P_B(1) + 2)^2 + \ldots + (P_A(m) + P_B(m) + 2)^2.$$

There are two potentially important differences between the objective functions OF_1 OF_2. First, OF_1 is not a continuous function, while OF_2 is. Second, OF_1 takes on smaller values than OF_2. The GA will seek to minimize OF_1 and OF_2, i.e., to find $2n$ $\{\pm 1\}$ values of the $2n$ variables $a_1, \ldots, a_n, b_1, \ldots, b_n$ such that OF_1 and OF_2

attain the value 0. Evidently, a set of $2n$ $\{\pm 1\}$ values with this property is also a solution of the original problem of finding two binary sequences with constant periodic autocorrelation function -2.

The GAs requirement of encoding the solution space as a set of binary vectors is inherent in the problem of searching for binary sequences with constant periodic/aperiodic autocorrelation function.

5.1 A General GAs Algorithmic Scheme

INPUT: k binary sequences $A_1, \ldots, A_k$, all of length n, and the autocorrelation (periodic/aperiodic) property satisfied

OUTPUT: k-tuples of sequences satisfying the input requirements (if any are found)

- Specify an initial randomly chosen population (first generation) of binary sequences of length n.
- Encode the required periodic/aperiodic autocorrelation property in the form of OF_1 or OF_2.
- Set current generation to be the first generation.
- Evolve the current generation to the next generation, using a set of genetic operators. Commonly used such operators include reproduction, crossover, and mutation.
- Examine the next generation to see whether it contains k binary sequences with the required periodic/aperiodic autocorrelation property. If yes, then output this solution and exit. If no, then set current generation to be the next generation and iterate the previous step.

Note that popular termination criteria for GAs include a predetermined number of generations to evolve or a predetermined amount of execution time.

There are several papers that use GAs to solve design-theoretic and combinatorial problems. In [1] the authors devise a GA to search for cocyclic Hadamard matrices. In [2] the author explains how to use genetic algorithms to solve three design-theoretic problems of graph-theoretic flavor. In [50] the authors apply GAs to construct D-optimal designs. In [68] the authors use so-called competent GAs to search efficiently for weighing matrices. In [17] the authors use SGA (simple genetic algorithm) to construct Hadamard matrices of various orders. The thesis [84] is concerned with the application of GAs to construct D-optimal experimental designs. The thesis [46] contains GAs to search for normal and near-Yang sequences.

6 Simulated Annealing

Simulated annealing (SA) was presented in [58] as a new approach to approximate solutions of difficult optimization problems. This approach is inspired from statistical mechanics and is motivated by an analogy to the behavior of physical systems in the presence of heat. In the words of Kirkpatrick et al., there is a deep and useful connection between statistical mechanics (the behavior of system with many degrees of freedom in thermal equilibrium at a finite temperature) and multivariate or combinatorial optimization (finding the minimum of a given function depending on many parameters).

In particular, they found that the Metropolis algorithm was well suited to approximate the observed behavior of solids when subjected to an annealing process. Recall that a local optimization method starts with some initial solution and, at each iteration, searches its neighbors for a better solution until no better solution can be found. A weakness of the local optimization method is that the program may get trapped at some local minima instead of detecting a global minimum. Simulated annealing addresses this issue by using randomization to improve on the local optimization search process, and in particular, occasional uphill moves (i.e., less than optimal solutions) are accepted with the hope that by doing so, a better solution will be obtainable later on. There are several parameters in the simulated annealing process which can significantly impact the actual performance, and there do not seem to be general rules on how to choose these parameters for the specific problem at hand.

In the annealing process, the physical system being optimized is first "melted" at a high temperature. The temperature is then decreased slowly until the system "freezes," and no further physical changes occur. When the system's structure is "frozen," this correspond to a minimum energy configuration. The physical analogy for simulated annealing is often described using the annealing method for growing a crystal. The rate at which the temperature is reduced is vitally important to the annealing process. If the cooling is done too quickly, widespread irregularities will form and the trapped energy level will be higher than what one would find in a perfectly structured crystal. It can be shown that the states of this physical system correspond to the solutions of an optimization problem. The energy of a state in the physical world corresponds to the cost of a solution in the optimization world. The minimum energy configuration that is obtained when a system is frozen corresponds to an optimal solution in an optimization problem.

6.1 A General SA Algorithmic Scheme

The basic ingredients needed to formulate a problem in a manner amenable to SA algorithms are:

1. A finite set S. This is the set of all possible solutions.
2. An objective function OF defined on S.
3. The set S^* is the set of global minima (i.e., the desired solutions) using the OF. It is assumed that $S^* \subset S$, a proper subset of S.
4. For each $i \in S$, we define a set $S(i) \subset S - \{i\}$ to be the set of *neighbors* of i.
5. A nonincreasing function $\alpha(t)$ called the *cooling schedule*, which controls how the temperature is lowered.

With the previous terminology in place, a general SA algorithmic scheme looks like:

```
Randomly select initial solution $s_0$;
Select an initial temperature $t_0 > 0$;
Select a temperature reduction function $\alpha$;
Repeat
  Repeat
    Randomly select a neighbor, $s$ of $s_0$;
    $\delta = \mathrm{OF}(s) - \mathrm{OF}(s_0)$;
    if($\delta < 0$)
      then $s_0 = s$;
    else
      generate random $x$ uniformly in the range $(0, 1)$;
        if($x < e^{-\delta/t}$)
          then $s_0 = s$;
  Until $iteration\_count = nrep$
  Set $t = \alpha(t)$;
Until stopping condition = true
$s_0$ is the approximation to the optimal solution
```

Recall that the goal of simulated annealing is to improve on the local search strategies by allowing some uphill moves in a controlled manner. The above SA scheme accepts a solution at each iteration if it is determined to be better than the current "best" solution. However, we also see how a less than optimal solution can be accepted and that it is accepted with probability $e^{-\delta/t}$. Also note that for the purposes of the OF required by SA, one can use OF_1 and OF_2 as defined previously for GAs. Commonly used cooling schedules include $\alpha(t) = \alpha t$, where $\alpha < 1$ and $\alpha(t) = \frac{t}{1 + \beta t}$ where β is small.

The papers [12, 55, 70, 72, 77] use SA techniques to tackle design theory problems.

7 Particle Swarm Optimization (PSO)

Particle swarm Optimization (PSO) is a population-based metaheuristic algorithm. It was introduced by R.C. Eberhart and J. Kennedy in 1995 [35] primarily for solving numerical optimization problems, as an alternative to evolutionary algorithms (EAs) [3]. Its verified efficiency in challenging optimization problems as well as its easy implementation rapidly placed PSO in a salient position among the state-of-the-art algorithms. Today, PSO counts a vast number of applications in diverse scientific fields [4, 5, 78], as well as an extensive bibliography [14, 37, 56, 83], and published scientific software [100].

Putting it formally, consider the n-dimensional global optimization problem:

$$\min_{x \in X \subset \mathbb{R}^n} f(x).$$

PSO probes the search space by using a *swarm*, namely, a set of search points:

$$S = \{x_1, x_2, \ldots, x_N\}, \qquad x_i \in X, \qquad i \in I = \{1, 2, \ldots, N\}.$$

Each search point of S constitutes a *particle*, i.e., a search vector:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^\top \in X, \qquad i \in I,$$

which is randomly initialized in X and allowed to iteratively move within it. Its motion is based on stochastically adapted position shifts, called *velocity*, while it retains in memory the *best position* it has ever visited. These quantities are denoted as

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^\top, \qquad p_i = (p_{i1}, p_{i2}, \ldots, p_{in})^\top, \qquad i \in I,$$

respectively. Thus, if t denotes the current iteration of the algorithm, it holds that

$$p_i(t) = \arg \min_{q \in \{0,1,\ldots,t\}} \left\{ f\left(x_i(q)\right) \right\}, \qquad i \in I.$$

The best positions constitute a sort of experience for the particles, guiding them towards the most promising regions of the search space, i.e., regions that posses lower function values.

In order to avoid the premature convergence of the particles on local minimizers, the concept of *neighborhood* was introduced [96]. A neighborhood of the i-th particle is defined as a set,

$$NB_{i,s} = \{j_1, j_2, \ldots, j_s\} \subseteq I, \qquad i \in NB_{i,s},$$

and consists of the indices of all the particles with which it can exchange information. Then, the neighborhood's best position

$$p_{g_i} = \arg \min_{j \in NB_{i,s}} \{f(p_j)\} \tag{5}$$

is used along with p_i to update the i-th particle's velocity at each iteration. The parameter s defines the *neighborhood size*. In the special case where $s = N$, the whole swarm constitutes the neighborhood. This case defines the so-called *global* PSO model (denoted as *gbest*), while strictly smaller neighborhoods correspond to the *local* PSO model (denoted as *lbest*). The schemes that are used for determining the particles that constitute each neighborhood are called *neighborhood topologies*, and they can have a crucial impact on PSO's performance. A popular scheme is the *ring* topology, where each particle assumes as neighbors of the particles with its adjacent indices, assuming that indices recycle after N. Based on the definitions above, the iterative scheme of PSO is defined as follows [15]:

$$v_{ij}(t+1) = \chi\Big[v_{ij}(t) + c_1\mathcal{R}_1\Big(p_{ij}(t) - x_{ij}(t)\Big) + c_2\mathcal{R}_2\Big(p_{g_i,j}(t) - x_{ij}(t)\Big)\Big], \quad (6)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1), \quad (7)$$

where $i = 1, 2, \ldots, N$; $j = 1, 2, \ldots, n$; the parameter χ is the *constriction coefficient*; acceleration constants c_1 and c_2 are called the *cognitive* and *social* parameter, respectively; and $\mathcal{R}_1$ and $\mathcal{R}_2$, are random variables uniformly distributed in the range [0, 1]. It shall be noted that a different value of $\mathcal{R}_1$ and $\mathcal{R}_2$ is sampled for each i and j in (6) at each iteration. Also, the best position of each particle is updated at each iteration, as follows:

$$p_i(t+1) = \begin{cases} x_i(t+1), & \text{if } f\big(x_i(t+1)\big) < f\big(p_i(t)\big), \\ p_i(t), & \text{otherwise,} \end{cases} \qquad i \in I. \quad (8)$$

The PSO variant described above was introduced by M. Clerc and J. Kennedy in [15], and it has gained increasing popularity, especially in interdisciplinary applications. This can be attributed to its simplicity, its efficiency, and its theoretical background.

Based on the stability analysis [15], the parameter set $\chi = 0.729$, $c_1 = c_2 = 2.05$, was determined as a satisfactory setting that produces a balanced convergence speed of the algorithm. Nevertheless, alternative settings have been introduced in relevant works [98]. Pseudocode of PSO is reported in Algorithm 1.

Unified particle swarm optimization (UPSO) was proposed as a scheme that harnesses the lbest and gbest PSO models, combining their exploration and exploitation properties [79, 82]. Let $\mathcal{G}_i(t+1)$ be the gbest velocity update of x_i, while $\mathcal{L}_i(t+1)$ be the corresponding lbest velocity update. Then, from Eq. (6), it holds that

$$\mathcal{G}_{ij}(t+1) = \chi\Big[v_{ij}(t) + c_1\mathcal{R}_1\Big(p_{ij}(t) - x_{ij}(t)\Big) + c_2\mathcal{R}_2\Big(p_{g,j}(t) - x_{ij}(t)\Big)\Big], \quad (9)$$

$$\mathcal{L}_{ij}(t+1) = \chi\Big[v_{ij}(t) + c_1\mathcal{R}_1\Big(p_{ij}(t) - x_{ij}(t)\Big) + c_2\mathcal{R}_2\Big(p_{g_i,j}(t) - x_{ij}(t)\Big)\Big], \quad (10)$$

where g denotes the overall best particle. The aggregation of these search directions defines the main UPSO scheme [79]:

$$\mathcal{U}_{ij}(t+1) = u\,\mathcal{G}_{ij}(t+1) + (1-u)\,\mathcal{L}_{ij}(t+1), \quad (11)$$

$$x_{ij}(t+1) = x_{ij}(t) + \mathcal{U}_{ij}(t+1). \quad (12)$$

The parameter $u \in [0, 1]$ is called the *unification factor* and determines the trade off between the two directions. Obviously, the standard gbest PSO is obtained by setting $u = 1$ in (11), while $u = 0$ corresponds to the standard lbest PSO. All intermediate values of $u \in (0, 1)$ define composite UPSO variants that combine the exploration and exploitation properties of the global and local PSO variant.

Besides the basic UPSO scheme, a stochastic parameter can also be incorporated in Eq. (11) to enhance UPSO's exploration capabilities [79]. Thus, depending on which variant UPSO is mostly based, Eq. (11) becomes

$$\mathcal{U}_{ij}(t+1) = \mathcal{R}_3\, u\, \mathcal{G}_{ij}(t+1) + (1-u)\, \mathcal{L}_{ij}(t+1), \qquad (13)$$

which is mostly based on lbest, or,

$$\mathcal{U}_{ij}(t+1) = u\, \mathcal{G}_{ij}(t+1) + \mathcal{R}_3\, (1-u)\, \mathcal{L}_{ij}(t+1), \qquad (14)$$

which is mostly based on gbest. The random variable $\mathcal{R}_3 \sim \mathcal{N}(\mu, \sigma^2)$ is normally distributed, resembling the mutation operator in EAs. Yet, it is biased towards directions that are consistent with the PSO dynamic, instead of the pure random mutation used in EAs. Following the assumptions of Matyas [71], a proof of convergence in probability was derived for the UPSO variants of Eqs. (13) and (14) [79].

Although PSO and UPSO have been primarily designed for real-valued optimization problems, there are various applications also in problems with integer, mixed-integer, and binary variables (e.g., see Eqs. [62, 69, 80, 81, 85, 86]). The most common and easy way to achieve this is by rounding the corresponding variables to their nearest integers when evaluating the particles. Of course, problem-dependent techniques may be additionally needed to enhance performance.

8 Ant Colony Optimization (ACO)

Ant colony optimization (ACO) is a probabilistic algorithm, primarily for solving combinatorial optimization problems. The general algorithmic concept of ACO was introduced by M. Dorigo in 1992 [30], aiming at detecting optimal paths in graphs by imitating the foraging behavior of real ants. The first approaches proved to be very promising, leading to further developments and enhancements [7, 31, 33].

Today, there is a variety of ant-based algorithms. Perhaps the most popular variants are Ant System (AS) [34], Max–Min Ant System (MMAS) [95], and Ant Colony System (ACS) [32]. Setting aside some differences, most ant-based approaches follow the same procedure flow, which can be summarized in the following actions:

```
Procedure ACO
WHILE (termination condition is false)
Construct_Solutions()
Optional_Further_Actions()
Update_Pheromones()
END WHILE
```

Algorithm 1: Pseudocode of the standard PSO algorithm

```
Input  : Objective function, $f : X \subset \mathbb{R}^n \to \mathbb{R}$; swarm size: $N$; parameters: $\chi$, $c_1$, $c_2$.
Output: Best detected solution: $x^*$, $f(x^*)$.
// Initialization
1  $t \leftarrow 0$.
2  for $i \leftarrow 1$ to $N$ do
3      Initialize $x_i(t)$ and $v_i(t)$, randomly.
4      $p_i(t) \leftarrow x_i(t)$. // Initialize best position
5      Evaluate $f(x_i(t))$. // Evaluate particle
6  end
// Main Iteration
7  while (termination criterion not met) do
       // Position and Velocity Update
8      for $i \leftarrow 1$ to $N$ do
9          Determine $p_{g_i}(t)$ from the $i$–th particle's neighborhood.
10         for $j \leftarrow 1$ to $n$ do
11             $v_{ij}(t+1) \leftarrow \chi \left[ v_{ij}(t) + c_1 \mathcal{R}_1 (p_{ij}(t) - x_{ij}(t)) + c_2 \mathcal{R}_2 (p_{g_i,j}(t) - x_{ij}(t)) \right]$.
12             $x_{ij}(t+1) \leftarrow x_{ij}(t) + v_{ij}(t+1)$.
13         end
14     end
       // Best Positions Update
15     for $i \leftarrow 1$ to $N$ do
16         Evaluate $f(x_i(t+1))$. // Evaluate new position
17         $p_i(t+1) \leftarrow \begin{cases} x_i(t+1), & \text{if } f(x_i(t+1)) < f(p_i(t)), \\ p_i(t), & \text{otherwise.} \end{cases}$
18     end
19     $t \leftarrow t + 1$.
20 end
```

In general, the algorithm assumes a number, K, of search agents, called the *ants*. Each ant constructs a solution component by component. The construction procedure is based on the probabilistic selection of each component's value from a discrete and finite set. For this purpose, a table of *pheromones* is retained and continuously updated. The pheromones play the role of quality weights, used for determining the corresponding selection probability of each possible value of a solution component.

Putting it formally, let

$$x = (x_1, x_2, \dots, x_n)^\top \in D,$$

be a candidate solution of the problem, with each component x_i, $i = 1, 2, \dots, n$, taking values from a discrete and finite set D_i. The construction of a solution begins from an initially empty partial solution, $x^p = \emptyset$. At each step, the next component of x^p is assigned one of the possible values from its corresponding discrete set. Naturally, the values assigned in previous components may affect the feasible set of candidate values for the current one, prohibiting the use of some of them to retain feasibility. Assume that x^p is already built up to the $(i-1)$-th component, and let $D_i = \{d_{i1}, d_{i2}, \ldots, d_{iM}\}$ be the set of candidate values for the i-th component. Each possible value d_{ij} is associated with a pheromone level τ_{ij}. The selection of a specific value for the i-th component is based on the probabilistic selection among the different candidate values, using the selection probabilities:

$$P(d_{ij}|x^p) = \frac{\tau_{ij}^a \, \eta(d_{ij})^b}{\sum\limits_{d_{il} \in D_i} \tau_{il}^a \, \eta(d_{il})^b}, \qquad j = 1, 2, \ldots, M, \tag{15}$$

where $\eta(.)$ is a *heuristic function* that offers a measure of the "desirability" of each component value in the solution (e.g., in TSP problems, it can be associated with the traveled distance). The parameters a and b are fixed, and they offer flexibility in tuning the trade off between pheromone and heuristic information.

Upon constructing a complete solution, it is evaluated with the objective value, and the pheromones are updated so that component values that appear in better solutions increase their pheromone levels and, consequently, their selection probabilities for subsequent iterations of the algorithm. Also, all pheromones undergo a standard reduction in their values, a procedure also known as *evaporation*. Thus, the pheromones are updated as follows:

$$\tau_{ij} = \begin{cases} (1-\rho)\,\tau_{ij} + \rho\,\Delta\tau, & \text{if } d_{ij} \text{ appears in a good candidate solution,} \\ (1-\rho)\,\tau_{ij}, & \text{otherwise,} \end{cases} \tag{16}$$

where $\rho \in (0, 1]$ is the *evaporation rate* and $\Delta\tau$ is an increment that can be either fixed or proportional to the quality of the corresponding candidate solution.

Different ant-based approaches can differ from the basic scheme described above. For example, in AS with K ants, the pheromone update takes place after all ants have constructed their solutions and they all contribute to the update:

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \sum_{k=1}^{K} \Delta\tau_{ij}^{(k)},$$

where $\Delta\tau_{ij}^{(k)}$ is the contributed pheromone from the k-th ant and it is related to its quality. On the other hand, ACS uses a different rule for selecting component values, where the value of the i-th component is determined as

$$d_{ig} = \arg \max_{d_{ij} \in D_i} \left\{ \tau_{ij}^a \, \eta(d_{ij})^b \right\}, \quad \text{if } q \leqslant q_0,$$

where $q \in [0, 1]$ is a uniformly distributed random number and $q_0 \in [0, 1]$ is a user-defined parameter; otherwise, the scheme of Eq. (15) is used. Also, only the best ant contributes to the pheromones, instead of the whole swarm. Finally, MMAS employs pheromone bounds $[\tau_{\min}, \tau_{\max}]$. Each pheromone is initialized to its maximum value and updated only from the best ant, either of the current iteration or overall.

There is a rich literature of applications of ant-based approaches in combinatorial optimization problems [7, 31, 33]. Recently, such approaches were used also for the solution of autocorrelation problems [69].

9 Combinatorial Optimization

Combinatorial optimization methods seem like a natural candidate of methods that should be employed to tackle extremely hard optimization problems such as the search for binary and ternary sequences that satisfy autocorrelation constraints. This is because combinatorial optimization methods routinely deal with problems featuring thousands of discrete variables. The missing link that allows one to formalize autocorrelation problems in a manner suitable for combinatorial optimization methods was provided in [66] and [61]. This formalism made it in fact possible to solve in [66] the entire Bömer-Antweiler diagram, which was proposed 20 years before, in [6]. This formalism is based on the fact that autocorrelation can be expressed in terms of certain symmetric matrices, which correspond to the matrices associated to autocorrelation viewed as *quadratic form*. We shall illustrate the formalism for D-optimal matrices, following the exposition in [61].

Let n be an odd positive integer and set $m = \frac{n-1}{2}$. D-optimal matrices correspond to two binary sequences of length n each, with constant periodic autocorrelation 2; see [61]. D-optimal matrices are $2n \times 2n$ $\{-1,+1\}$-matrices that attain Ehlich's determinant bound. See [29] for a state-of-the-art survey of existence questions for D-optimal matrices with $n < 200$. We reproduce the following definition and lemma from [66].

Definition 8 Let $a = [a_1, a_2, \ldots, a_n]^T$ be a column $n \times 1$ vector, where $a_1, a_2, \ldots, a_n \in \{-1,+1\}$ and consider the elements of the periodic autocorrelation function vector $P_A(1), \ldots, P_A(m)$. Define the following m symmetric matrices (which are independent of the sequence a), for $i = 1, \ldots, m$

$$M_i = (m_{jk}), \text{ s.t. } \begin{cases} m_{jk} = m_{kj} = \frac{1}{2}, \text{ when } a_j a_k \in P_A(i), & j,k \in \{1,\ldots,n\} \\ 0, & \text{otherwise} \end{cases}$$

Lemma The matrices M_i can be used to write equations involving autocorrelation in a matrix form:

- For n odd:
$$a^T M_i a = P_A(i), \ \ i = 1, \ldots, m.$$
- For n even:
$$a^T M_i a = P_A(i), \ \ i = 1, \ldots, m-1 \text{ and } a^T M_m a = \frac{1}{2} P_A(m).$$

Now one can formulate the D-optimal matrices problem as a binary feasibility problem.

Binary Feasibility version of the D-optimal matrices problem Find two sequences $a = [a_1, \ldots, a_n]$, $b = [b_1, \ldots, b_n]$, (viewed as $n \times 1$ column vectors) such that

$$a^T M_i a + b^T M_i b = 2, \quad i = 1, \ldots, m,$$

where $a_i, b_i \in \{-1, +1\}, i = 1, \ldots, n$.

It is now clear that one can use combinatorial optimization algorithms to search for D-optimal matrices, as well as several other combinatorial matrices and sequences referenced in Table 1. Definition 8 can easily be adapted for aperiodic autocorrelation as well.

Note that the D-optimal matrices problem also features certain Diophantine constraints, as proved in [61] and subsequently generalized in [29]. It is not clear what is the right way to incorporate this kind of Diophantine constraints into the combinatorial optimization formalism described here, and this is certainly an issue that deserves to be investigated further.

The papers [73–76] demonstrate how to use various optimization techniques to find new designs.

10 Applications

There are several applications of combinatorial matrices and sequences described in Table 1, in such areas as telecommunications, coding theory, and cryptography. The reader can consult [45] for applications in signal design for communications, radar, and cryptography applications. The book [53] describes applications in signal processing, coding theory, and cryptography. The book [104] describes applications in communications, signal processing, and image processing. The paper [99] discusses how partial Hadamard matrices (see [20]) are used in the area of compressed sensing. The paper [93] describes in detail several applications of Hadamard matrices on code division multiple access (CDMA) communication systems as well as in telecommunications. The 55-page paper [49] describes applications of Hadamard matrices in binary codes, information processing, maximum determinant problems, spectrometry, pattern recognition, and other areas. The papers [91] and [92] and the thesis [9] discuss cryptography applications of Hadamard matrices. Chapter 10 of the book [51] is devoted to several combinatorial problems that can be tackled with stochastic search methods.

11 Conclusion

Combinatorial matrices and the associated binary and ternary sequences provide a wide array of extremely challenging optimization problems that can be tackled with a variety of metaheuristic and combinatorial methods. We tried to summarize some of these methods in the present chapter. Although there are no polynomial complexity algorithms to solve the problem of searching for such matrices and sequences, the combination of new insights, new algorithms, and new implementations will continue to produce new results. The second edition of the Handbook of Combinatorial Designs [16], edited by Charles J. Colbourn and Jeffrey H. Dinitz, is a very valuable resource regarding the state of the art on open cases for several types of such matrices and sequences. The on-line updated webpage maintained by the authors is also quite useful, as many open cases have been resolved since the publication of the handbook in 2007. We firmly believe that cross-fertilization of disciplines is a very important process, from which all involved disciplines benefit eventually. It seems reasonable to predict that the area of combinatorial matrices and their associated sequences will continue to experience strong interactions with other research areas, for many years to come.

Cross-References

▸Algorithms for the Satisfiability Problem
▸Binary Unconstrained Quadratic Optimization Problem

Recommended Reading

1. V. Álvarez, J.A. Armario, M.D. Frau, P. Real, A genetic algorithm for cocyclic Hadamard matrices, in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*. Lecture Notes in Computer Science, vol. 3857 (Springer, Berlin, 2006), pp. 144–153
2. D. Ashlock, Finding designs with genetic algorithms, in *Computational and Constructive Design Theory*. Mathematics and Its Applications, vol. 368 (Kluwer, Dordrecht, 1996), pp. 49–65
3. T. Bäck, D. Fogel, Z. Michalewicz, *Handbook of Evolutionary Computation* (IOP Publishing/Oxford University Press, New York, 1997)
4. A. Banks, J. Vincent, C. Anyakoha, A review of particle swarm optimization. Part i: background and development. Nat. Comput. **6**(4), 467–484 (2007)
5. A. Banks, J. Vincent, C. Anyakoha, A review of particle swarm optimization. Part ii: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications. Nat. Comput. **7**(1), 109–124 (2008)
6. L. Bömer, M. Antweiler, Periodic complementary binary sequences. IEEE Trans. Inf. Theory **36**(6), 1487–1494 (1990)
7. E. Bonabeau, M. Dorigo, G. Théraulaz, *Swarm Intelligence: From Natural to Artificial Systems* (Oxford University Press, New York, 1999)
8. P.B. Borwein, R.A. Ferguson, A complete description of Golay pairs for lengths up to 100. Math. Comput. **73**(246), 967–985 (2004)
9. A. Braeken, *Cryptographic Properties of Functions and S-Boxes*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 2006
10. R.A. Brualdi, *Combinatorial Matrix Classes*. Encyclopedia of Mathematics and Its Applications, vol. 108 (Cambridge University Press, Cambridge, 2006)
11. R.A. Brualdi, H.J. Ryser, *Combinatorial Matrix Theory*. Encyclopedia of Mathematics and Its Applications, vol. 39 (Cambridge University Press, Cambridge, 1991)

12. Y.W. Cheng, D.J. Street, W.H. Wilson, Two-stage generalized simulated annealing for the construction of change-over designs, in *Designs, 2002*. Mathematics and Its Applications, vol. 563 (Kluwer, Boston, 2003), pp. 69–79
13. M. Chiarandini, I.S. Kotsireas, C. Koukouvinos, L. Paquete, Heuristic algorithms for Hadamard matrices with two circulant cores. Theor. Comput. Sci. **407**(1–3), 274–277 (2008)
14. M. Clerc, *Particle Swarm Optimization* (ISTE Ltd, London/Newport Beach, 2006)
15. M. Clerc, J. Kennedy, The particle swarm–explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**(1), 58–73 (2002)
16. C.J. Colbourn, J.H. Dinitz (eds.), *Handbook of Combinatorial Designs*. Discrete Mathematics and Its Applications (Boca Raton), 2nd edn. (Chapman & Hall/CRC, Boca Raton, 2007)
17. J. Cousineau, I.S. Kotsireas, C. Koukouvinos, Genetic algorithms for orthogonal designs. Australas. J. Comb. **35**, 263–272 (2006)
18. R. Craigen, Boolean and ternary complementary pairs. J. Comb. Theory Ser. A **104**(1), 1–16 (2003)
19. W. de Launey, D. Flannery, *Algebraic Design Theory*. Mathematical Surveys and Monographs, vol. 175 (American Mathematical Society, Providence, 2011)
20. W. de Launey, D.A. Levin, A Fourier-analytic approach to counting partial Hadamard matrices. Cryptogr. Commun. **2**(2), 307–334 (2010)
21. D.Ž. Đoković, Two Hadamard matrices of order 956 of Goethals-Seidel type. Combinatorica **14**(3), 375–377 (1994)
22. D.Ž. Đoković, Equivalence classes and representatives of Golay sequences. Discret. Math. **189**(1–3), 79–93 (1998)
23. D.Ž. Đoković, Hadamard matrices of order 764 exist. Combinatorica **28**(4), 487–489 (2008)
24. D.Ž. Đoković, Skew-Hadamard matrices of orders 188 and 388 exist. Int. Math. Forum **3**(21–24), 1063–1068 (2008)
25. D.Ž. Đoković, Skew-Hadamard matrices of orders 436, 580, and 988 exist. J. Comb. Des. **16**(6), 493–498 (2008)
26. D.Ž. Đoković, On the base sequence conjecture. Discret. Math. **310**(13–14), 1956–1964 (2010)
27. D.Ž. Đoković, Classification of normal sequences. Int. J. Comb. Art. ID 937941, 15 (2011)
28. D.Ž. Đoković, Cyclic $(v; r, s; \lambda)$ difference families with two base blocks and $v \leq 50$. Ann. Comb. **15**(2), 233–254 (2011)
29. D.Ž. Đoković, I.S. Kotsireas, New results on D-optimal matrices. J. Comb. Des. **20**(6), 278–289 (2012)
30. M. Dorigo, *Optimization, Learning and Natural Algorithms*. PhD thesis, Politecnico di Milano, Italy, 2002
31. M. Dorigo, G. Di Caro, The ant colony optimization meta–heuristic, in *New Ideas in Optimization*, ed. by D. Corne, M. Dorigo, F. Glover (McGraw-Hill, London, 1999), pp. 11–32
32. M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Comput. **1**(1), 53–66 (1997)
33. M. Dorigo, T. Stützle, *Ant Colony Optimization* (MIT Press, Cambridge, 2004)
34. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. Part B **26**(1), 29–41 (1996)
35. R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in *Proceedings Sixth Symposium on Micro Machine and Human Science* (IEEE Service Center, Piscataway, 1995), pp. 39–43
36. S. Eliahou, M. Kervaire, B. Saffari, A new restriction on the lengths of Golay complementary sequences. J. Comb. Theory Ser. A **55**(1), 49–59 (1990)
37. A.P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence* (Wiley, Hoboken, 2006)
38. T. Feng, Q. Xiang, Cyclotomic constructions of skew Hadamard difference sets. J. Comb. Theory Ser. A **119**(1), 245–256 (2012)

39. T. Feng, Q. Xiang, Strongly regular graphs from unions of cyclotomic classes. J. Comb. Theory Ser. B **102**, 982–995 (2012)
40. F. Fiedler, J. Jedwab, M.G. Parker, A framework for the construction of Golay sequences. IEEE Trans. Inf. Theory **54**(7), 3114–3129 (2008)
41. F. Fiedler, J. Jedwab, M.G. Parker, A multi-dimensional approach to the construction and enumeration of Golay complementary sequences. J. Comb. Theory Ser. A **115**(5), 753–776 (2008)
42. R.J. Fletcher, M. Gysin, J. Seberry, Application of the discrete Fourier transform to the search for generalised Legendre pairs and Hadamard matrices. Australas. J. Comb. **23**, 75–86 (2001)
43. A. Gavish, A. Lempel, On ternary complementary sequences. IEEE Trans. Inf. Theory **40**(2), 522–526 (1994)
44. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, Reading, 1989)
45. S.W. Golomb, G. Gong, *Signal Design for Good Correlation* (Cambridge University Press, Cambridge, 2005). For wireless communication, cryptography, and radar
46. M.M. Gysin, *Algorithms for Searching for Normal and Near-Yang Sequences*. University of Wollongong, 1993. Thesis (MSc)–University of Wollongong
47. M.M. Gysin, *Combinatorial Designs, Sequences and Cryptography*. PhD thesis, University of Wollongong, Wollongong, NSW, Australia, 1997
48. M. Hall Jr., *Combinatorial Theory* (Blaisdell Publishing Co./Ginn and Co., Waltham/Toronto/London, 1967)
49. A. Hedayat, W.D. Wallis, Hadamard matrices and their applications. Ann. Stat. **6**(6), 1184–1238 (1978)
50. A. Heredia-Langner, W.M. Carlyle, D.C. Montgomery, C.M. Borror, G.C. Runger, Genetic algorithms for the construction of d-optimal designs. J. Qual. Technol. **35**(1), 28–46 (2003)
51. H.H. Hoos, T. Stützle, *Stochastic Local Search: Foundations & Applications* (Elsevier/Morgan Kaufmann, Oxford/San Francisco, 2004)
52. K.J. Horadam, *Hadamard Matrices and Their Applications* (Princeton University Press, Princeton, 2007)
53. K.J. Horadam, Hadamard matrices and their applications: progress 2007–2010. Cryptogr. Commun. **2**(2), 129–154 (2010)
54. Y.J. Ionin, M.S. Shrikhande, *Combinatorics of Symmetric Designs*. New Mathematical Monographs, vol. 5 (Cambridge University Press, Cambridge, 2006)
55. J.A. John, D. Whitaker, Construction of resolvable row-column designs using simulated annealing. Aust. J. Stat. **35**(2), 237–245 (1993)
56. J. Kennedy, R.C. Eberhart, *Swarm Intelligence* (Morgan Kaufmann Publishers, San Francisco, 2001)
57. H. Kharaghani, B. Tayfeh-Rezaie, A Hadamard matrix of order 428. J. Comb. Des. **13**(6), 435–440 (2005)
58. S. Kirkpatrick, C.D. Gelatt Jr., M.P. Vecchi, Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
59. I.S. Kotsireas, C. Koukouvinos, Genetic algorithms for the construction of Hadamard matrices with two circulant cores. J. Discret. Math. Sci. Cryptogr. **8**(2), 241–250 (2005)
60. I.S. Kotsireas, C. Koukouvinos, Hadamard ideals and Hadamard matrices from two circulant submatrices. J. Comb. Math. Comb. Comput. **61**, 97–110 (2007)
61. I.S. Kotsireas, P.M. Pardalos, D-optimal matrices via quadratic integer optimization. J. Heuristics (2012) (to appear)
62. I.S. Kotsireas, C. Koukouvinos, K.E. Parsopoulos, M.N. Vrahatis, Unified particle swarm optimization for Hadamard matrices of Williamson type, in *Proceedings of the 1st International Conference on Mathematical Aspects of Computer and Information Sciences (MACIS 2006)*, Beijing, China, 2006, pp. 113–121
63. I.S. Kotsireas, C. Koukouvinos, J. Seberry, Hadamard ideals and Hadamard matrices with two circulant cores. Eur. J. Combin. **27**(5), 658–668 (2006)

64. I.S. Kotsireas, C. Koukouvinos, J. Seberry, Weighing matrices and string sorting. Ann. Comb. **13**(3), 305–313 (2009)
65. I.S. Kotsireas, C. Koukouvinos, P.M. Pardalos, An efficient string sorting algorithm for weighing matrices of small weight. Optim. Lett. **4**(1), 29–36 (2010)
66. I.S. Kotsireas, C. Koukouvinos, P.M. Pardalos, O.V. Shylo, Periodic complementary binary sequences and combinatorial optimization algorithms. J. Comb. Optim. **20**(1), 63–75 (2010)
67. I.S. Kotsireas, C. Koukouvinos, P.M. Pardalos, A modified power spectral density test applied to weighing matrices with small weight. J. Comb. Optim. **22**(4), 873–881 (2011)
68. I.S. Kotsireas, C. Koukouvinos, P.M. Pardalos, D.E. Simos, Competent genetic algorithms for weighing matrices. J. Comb. Optim. **24**, 508–525 (2012)
69. I.S. Kotsireas, K.E. Parsopoulos, G.S. Piperagkas, M.N. Vrahatis, Ant–based approaches for solving autocorrelation problems, in *Eighth International Conference on Swarm Intelligence (ANTS 2012)*, Lecture Notes in Computer Science (LNCS), Brussels, Belgium, Vol. 7461, pp. 220–227 Springer (2012)
70. P.C. Li, Combining genetic algorithms and simulated annealing for constructing lotto designs. J. Comb. Math. Comb. Comput. **45**, 109–121 (2003)
71. J. Matyas, Random optimization. Autom. Remote Control **26**, 244–251 (1965)
72. R.K. Meyer, C.J. Nachtsheim, Constructing exact D-optimal experimental designs by simulated annealing. Am. J. Math. Manag. Sci. **8**(3–4), 329–359 (1988)
73. L.B. Morales, Constructing difference families through an optimization approach: six new BIBDs. J. Comb. Des. **8**(4), 261–273 (2000)
74. L.B. Morales, Constructing some PBIBD(2)s by tabu search algorithm. J. Comb. Math. Comb. Comput. **43**, 65–82 (2002)
75. L.B. Morales, Constructing cyclic PBIBD(2)s through an optimization approach: thirty-two new cyclic designs. J. Comb. Des. **13**(5), 377–387 (2005)
76. L.B. Morales, Constructing 1-rotational NRDFs through an optimization approach: new (46,9,8), (51,10,9) and (55,9,8)-NRBDs. J. Stat. Plann. Inference **139**(1), 62–68 (2009)
77. K.J. Nurmela, P.R.J. Östergård, Upper bounds for covering designs by simulated annealing, in *Proceedings of the Twenty-fourth Southeastern International Conference on Combinatorics, Graph Theory, and Computing*, Boca Raton, FL, 1993, vol. 96, pp. 93–111
78. K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through particle swarm optimization. Nat. Comput. **1**(2–3), 235–306 (2002)
79. K.E. Parsopoulos, M.N. Vrahatis, UPSO: a unified particle swarm optimization scheme, in *Proceedings of the International Conference of Computational Methods in Sciences and Engineering (ICCMSE 2004)*. Lecture Series on Computer and Computational Sciences, vol. 1 (VSP International Science Publishers, Zeist, 2004), pp. 868–873
80. K.E. Parsopoulos, M.N. Vrahatis, Unified particle swarm optimization for tackling operations research problems, in *Proceedings of the IEEE 2005 Swarm Intelligence Symposium*, Pasadena, CA, USA, 2005, pp. 53–59
81. K.E. Parsopoulos, M.N. Vrahatis, Studying the performance of unified particle swarm optimization on the single machine total weighted tardiness problem, in ed. by A. Sattar, B.H. Kang Lecture Notes in Artificial Intelligence (LNAI), vol. 4304 (Springer, 2006), pp. 760–769
82. K.E. Parsopoulos, M.N. Vrahatis, Parameter selection and adaptation in unified particle swarm optimization. Math. Comput. Model. **46**(1–2), 198–213 (2007)
83. K.E. Parsopoulos, M.N. Vrahatis, *Particle Swarm Optimization and Intelligence: Advances and Applications* (Information Science Publishing (IGI Global), Hershey, 2010)
84. C.B. Pheatt, *Construction of D-Optimal Experimental Designs Using Genetic Algorithms*. ProQuest LLC, Ann Arbor, MI, 1995. Thesis (Ph.D.)–Illinois Institute of Technology
85. G.S. Piperagkas, C. Voglis, V.A. Tatsis, K.E. Parsopoulos, K. Skouri, Applying PSO and DE on multi–item inventory problem with supplier selection, in *The 9th Metaheuristics International Conference (MIC 2011)*, Udine, Italy, 2011, pp. 359–368
86. G.S. Piperagkas, I. Konstantaras, K. Skouri, K.E. Parsopoulos, Solving the stochastic dynamic lot–sizing problem through nature–inspired heuristics. Comput. Oper. Res. **39**(7), 1555–1565 (2012)

87. C.R. Reeves, J.E. Rowe, *Genetic Algorithms: Principles and Perspectives*. Operations Research/Computer Science Interfaces Series, vol. 20 (Kluwer, Boston, 2003). A guide to GA theory
88. H.J. Ryser, *Combinatorial Mathematics*. The Carus Mathematical Monographs, vol. 14 (The Mathematical Association of America, Buffalo, 1963)
89. J. Sawada, Generating bracelets in constant amortized time. SIAM J. Comput. **31**(1), 259–268 (2001)
90. J. Sawada, A fast algorithm to generate necklaces with fixed content. Theor. Comput. Sci. **301**(1–3), 477–489 (2003)
91. J. Seberry, X. Zhang, Y. Zheng, Cryptographic Boolean functions via group Hadamard matrices. Australas. J. Comb. **10**, 131–145 (1994)
92. J. Seberry, X. Zhang, Y. Zheng, Pitfalls in designing substitution boxes (extended abstract), in *Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '94* (Springer, London, 1994), pp. 383–396
93. J. Seberry, B.J. Wysocki, T.A. Wysocki, On some applications of Hadamard matrices. Metrika **62**(2–3), 221–239 (2005)
94. D.R. Stinson, *Combinatorial Designs, Constructions and Analysis* (Springer, New York, 2004)
95. T. Stützle, H.H. Hoos, MAX MIN ant system. Future Gener. Comput. Syst. **16**, 889–914 (2000)
96. P.N. Suganthan, Particle swarm optimizer with neighborhood operator, in *Proceedings of the IEEE Congress on Evolutionary Computation*, Washington, D.C., USA, 1999, pp. 1958–1961
97. M.N. Syed, I.S. Kotsireas, P.M. Pardalos, D-optimal designs: a mathematical programming approach using cyclotomic cosets. Informatica **22**(4), 577–587 (2011)
98. I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection. Inf. Process. Lett. **85**, 317–325 (2003)
99. Y. Tsaig, D.L. Donoho, Extensions of compressed sensing. Signal Process. **86**(3), 549–571 (2006)
100. C. Voglis, K.E. Parsopoulos, D.G. Papageorgiou, I.E. Lagaris, M.N. Vrahatis, MEMPSODE: a global optimization software based on hybridization of population–based algorithms and local searches. Comput. Phys. Commun. **183**(5), 1139–1154 (2012)
101. J. Wallis, On supplementary difference sets. Aequ. Math. **8**, 242–257 (1972)
102. J. Wallis, A note on supplementary difference sets. Aequ. Math. **10**, 46–49 (1974)
103. M. Yamada, Supplementary difference sets and Jacobi sums. Discret. Math. **103**(1), 75–90 (1992)
104. R.K. Yarlagadda, J.E. Hershey, *Hadamard Matrix Analysis and Synthesis*. The Kluwer International Series in Engineering and Computer Science, vol. 383 (Kluwer, Boston, 1997). With applications to communications and signal/image processing