SEARCH FOR COSTAS ARRAYS VIA SPARSE REPRESENTATION

Mojtaba Soltanalian[*+], *Petre Stoica*[+] *and Jian Li*[†]

[+]Dept. of Information Technology, Uppsala University, Uppsala, Sweden
[†]Dept. of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA

ABSTRACT

Costas arrays are mainly known as a certain type of optimized time-frequency coding pattern for sonar and radar. In order to fulfill the need for effective computational approaches to find Costas arrays, in this paper, we propose a sparse formulation of the Costas array search problem. The new sparse representation can pave the way for using an extensive number of methods offered by the sparse signal recovery literature. It is further shown that Costas arrays can be obtained using an equivalent quadratic program with linear constraints. A numerical approach is devised and used to illustrate the performance of the proposed formulations.

Index Terms— Code design, Costas arrays, frequency hopping, radar codes, sparsity

1. INTRODUCTION

Costas arrays have been studied in engineering and mathematics for around half a century— however, many related fundamental questions are not yet answered [1][2]. The definition of a Costas array is fairly straightforward:

Definition 1. *A Costas array is a set of n points lying on the squares of an $n \times n$ checkerboard, such that each row and column contains only one point, and all of the $\binom{n}{2}$ displacement vectors*[1] *between each pair of points are distinct.*

Fig. 1 shows an example of a Costas array of size 8×8. Costas arrays are mainly known as time-frequency coding patterns that optimize the performance of sonars and radars. They have also shown promising applications in data hiding and mobile radio [3][4]. The usefulness of Costas arrays in sonar and radar applications can be seen more clearly by the following alternative definition of Costas arrays:

Definition 2. *A permutation matrix $\boldsymbol{P}$ of size $n \times n$ represents a Costas array if and only if for any pair of integers $(r, s) \neq (0, 0)$, $|r| \leq n$, $|s| \leq n$, the two-dimensional (2D) correlation function $c(r, s)$ associated with $\boldsymbol{P}$ satisfies*

$$c(r,s) = \sum_{k=1}^{n} \sum_{l=1}^{n} \boldsymbol{P}(k,l)\boldsymbol{P}(k+r, l+s) \leq 1 \tag{1}$$

where $\boldsymbol{P}$ is extended with zeros when required.

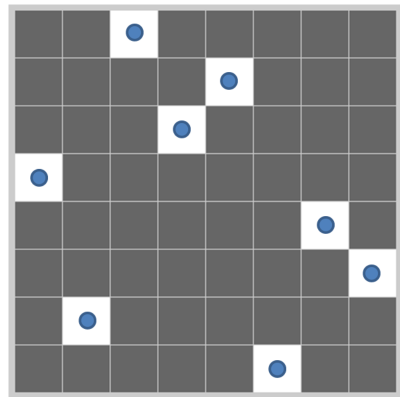


Fig. 1. A Costas array of size 8×8.

Note that the low 2D correlation character described in (1) leads to a low *ambiguity* in detection of moving targets. Due to their importance, several analytical construction methods have been proposed for Costas arrays [5]. Thanks to such constructions, Costas arrays are known for an infinite number of orders n. On the other hand, it is not yet known whether Costas arrays exist for all n.

Using computer clusters, the enumeration of Costas arrays has been accomplished via brute-force computational methods for all $n \leq 29$ [6]. However, the size of the search space is reported to grow exponentially with n, which makes the problem of finding Costas arrays impossible to tackle via exhaustive search when n grows large [7]. With such issues in mind, in this paper (Sections 2 and 3), we propose a sparse formulation of the Costas array search problem:

- We believe that the sparse formulation introduced in this paper lays the ground for using the many methods offered by the extensive literature on sparse signal recovery— which can lead to more effective numerical approaches than the exhaustive search.

This work was supported in part by the European Research Council (ERC), the Swedish Research Council (VR), and the Office of Naval Research (ONR) under Grant No. N00014-12-1-0381.

* Please address all the correspondence to Mojtaba Soltanalian, Phone: (+46) 18-471-3168; Fax: (+46) 18-511925; Email: mojtaba.soltanalian@it.uu.se

[1]Note that the points in the checkerboard can be associated with position vectors (initiated from an arbitrary origin). Then the displacement vectors are the differences of such position vectors.

- To the best of our knowledge, this work is the first to cast the Costas array search problem as an optimization problem, in a form which is well-known in the signal processing community.

Based on the proposed formulation, and in order to show its performance, a numerical approach is devised and used to find a Costas array in Section 4.

Notation: We use bold lowercase letters for vectors and bold uppercase letters for matrices. $(\cdot)^T$ denotes the vector/matrix transpose. $\mathbf{1}$ and $\mathbf{0}$ are the all-one and all-zero vectors/matrices. $\boldsymbol{e}_k$ and $\widetilde{\boldsymbol{e}}_k$ are the k^{th} standard basis vectors in $\mathbb{R}^n$, and $\mathbb{R}^{n^2}$, respectively. $vec(\boldsymbol{X})$ is a vector obtained by stacking the columns of $\boldsymbol{X}$ successively. $\|\boldsymbol{x}\|_n$ or the l_n-norm of the vector $\boldsymbol{x}$ is defined as $(\sum_k |\boldsymbol{x}(k)|^n)^{\frac{1}{n}}$ where $\{\boldsymbol{x}(k)\}$ are the entries of $\boldsymbol{x}$; for $n = 0$, $\|\boldsymbol{x}\|_0$ is given by the number of nonzero entries of $\boldsymbol{x}$. Finally, the symbol $\otimes$ stands for the Kronecker product of matrices.

2. COSTAS ARRAYS: A LINEAR FORMULATION

In this section, we introduce a linear interpretation of the Costas array constraints. Particularly, we show that such constraints can be expressed as an under-determined linear system of equations along with some linear inequalities. To this end, we propose the following geometrically equivalent definition of Costas arrays:

Definition 3. *A Costas array is a set of n points lying on the squares of an $n \times n$ checkerboard, such that (i) each row and column contains only one point (**permutation property**), and that (ii) no four points form a parallelogram; moreover no three equidistant points occur on the same line (**distinctness property**).*

Let the binary matrix $\boldsymbol{X}_{n\times n}$ (with entries in $\{0, 1\}$) denote a Costas array, and let $\boldsymbol{x} = vec(\boldsymbol{X})$. Assuming binary variables, the permutation property in Definition 3 can be expressed by the linear equality constraint:

$$\boldsymbol{A}\boldsymbol{x} = \mathbf{1}_{2n\times 1} \triangleq \boldsymbol{b} \tag{2}$$

where

$$\boldsymbol{A}_{2n\times n^2} = \begin{pmatrix} \boldsymbol{O}_1 & \boldsymbol{O}_2 & \cdots & \boldsymbol{O}_n \\ \boldsymbol{I} & \boldsymbol{I} & \cdots & \boldsymbol{I} \end{pmatrix} \tag{3}$$

and $\boldsymbol{O}_k = \boldsymbol{e}_k \otimes \mathbf{1}_n^T$. We further note that the distinctness property in Definition 3 can be formulated using a number of linear inequalities. Suppose that the location indices $\{i_1, i_2, i_3, i_4\}$ in $\boldsymbol{x}$ represent the vertices of a parallelogram in $\boldsymbol{X}$. To avoid forming such a parallelogram, it is sufficient to add the inequality

$$\boldsymbol{x}(i_1) + \boldsymbol{x}(i_2) + \boldsymbol{x}(i_3) + \boldsymbol{x}(i_4) \leq 3, \quad \text{i.e. } (\widetilde{\boldsymbol{e}}_{i_1} + \widetilde{\boldsymbol{e}}_{i_2} + \widetilde{\boldsymbol{e}}_{i_3} + \widetilde{\boldsymbol{e}}_{i_4})^T\boldsymbol{x} \leq 3 \tag{4}$$

Table 1. An algorithmic construction of $(\boldsymbol{A}', \boldsymbol{b}')$

Step 1: For all triples (i_1, i_2, i_3) (not on the same row/column) sorted by their row number in the checkerboard do:
Step 1-1: If i_2 is in the middle of i_1 and i_3,
Step 1-1-1: Add the row vector $(\widetilde{\boldsymbol{e}}_{i_1} + \widetilde{\boldsymbol{e}}_{i_2} + \widetilde{\boldsymbol{e}}_{i_3})^T$ at the bottom of $\boldsymbol{A}'$, and 2 at the bottom of $\boldsymbol{b}'$.
Step 1-2: Else,
Step 1-2-1: Find the 4^{th} vertices $\{i_4\}$ corresponding to the two parallelograms that can be formed using the available vertices (i_1, i_2, i_3) with i_1 as the vertex with the minimum row number.
Step 1-2-2: For any of the two possible locations $\{i_4\}$ which occur inside the checkerboard, include the row vector $(\widetilde{\boldsymbol{e}}_{i_1} + \widetilde{\boldsymbol{e}}_{i_2} + \widetilde{\boldsymbol{e}}_{i_3} + \widetilde{\boldsymbol{e}}_{i_4})^T$ at the bottom of $\boldsymbol{A}'$, and 3 at the bottom of $\boldsymbol{b}'$.

to the constraint set. Furthermore, any pattern of equidistant points on the same line, represented by the location indices $\{i_1, i_2, i_3\}$ in $\boldsymbol{x}$, can be avoided by the linear inequality constraint

$$\boldsymbol{x}(i_1) + \boldsymbol{x}(i_2) + \boldsymbol{x}(i_3) \leq 2, \quad \text{i.e. } (\widetilde{\boldsymbol{e}}_{i_1} + \widetilde{\boldsymbol{e}}_{i_2} + \widetilde{\boldsymbol{e}}_{i_3})^T\boldsymbol{x} \leq 2. \tag{5}$$

By including the linear constraints associated with all possible parallelograms and three equidistant points on a same line in the checkerboard, one can formulate the distinctness property in the following unified form:

$$\boldsymbol{A}'\boldsymbol{x} \leq \boldsymbol{b}'. \tag{6}$$

Note that although $\boldsymbol{A}'$ and $\boldsymbol{b}'$ can be hardly formulated in an explicit form, they are easy to generate algorithmically. An example of such an algorithmic construction of $(\boldsymbol{A}', \boldsymbol{b}')$ is given in Table 1. Finally, a binary vector $\boldsymbol{x}_{n^2\times 1}$ represents a Costas array if and only if it is a solution to the linear system

$$\begin{cases} \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \\ \boldsymbol{A}'\boldsymbol{x} \leq \boldsymbol{b}'. \end{cases} \tag{7}$$

As indicated earlier, the linear system in (7) contains $2n$ equality constraints. We refer the interested reader to the Appendix for enumeration results regarding the inequality constraints in (7).

3. SPARSE REPRESENTATION OF COSTAS ARRAYS

It is interesting to observe that the binary constraint on $\boldsymbol{x}$ can be omitted if one seeks for the *sparsest* solution of the linear system in (7). Consider the following optimization problem:

$$P_0 : \min_{\boldsymbol{x}\in\mathbb{R}^{n^2}} \quad \|\boldsymbol{x}\|_0 \tag{8}$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \\ \boldsymbol{A}'\boldsymbol{x} \leq \boldsymbol{b}', \\ \mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1} \end{cases} \tag{9}$$

Theorem 1. *Every solution of P_0 represents a Costas array, and vice versa.*

Proof: According to the equality constraint in (9), every solution of P_0 has at least one nonzero element in the location indices corresponding to each row/column of the checkerboard. This implies that every solution $\boldsymbol{x}$ to P_0 has an l_0-norm of at least n. On the other hand, if $\|\boldsymbol{x}\|_0 = n$, the satisfaction of the equality constraint in (9) implies that $\boldsymbol{x}$ denotes a permutation matrix, and as a result, the satisfaction of the inequality constraint in (9) shows that $\boldsymbol{x}$ represents a Costas array. Moreover, any $\boldsymbol{x}$ representing a Costas array is a feasible solution to the linear system in (9) with the minimum l_0-norm, i.e. n. ■

Note that the sparse formulation above paves the way for employing many existing, as well as emerging sparse signal recovery techniques in the literature. The recovery of sparse signals subject to linear constraints has been studied widely in the past decade, see [8]-[14] and the references therein. We note that as the l_1-norm of any feasible $\boldsymbol{x} \geq \mathbf{0}$ of P_0 is constant, it is not possible to employ the convex l_1 *minimization* alternative of P_0, although it is a common practice in sparse signal recovery [11]. From a geometrical point of view, the solutions to P_0, i.e. the sparsest solutions to (7), have the maximum Euclidean distance from the origin. In other words, P_0 is equivalent to

$$P_2: \max_{\boldsymbol{x}\in\mathbb{R}^{n^2}} \quad \|\boldsymbol{x}\|_2 \tag{10}$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \\ \boldsymbol{A}'\boldsymbol{x} \leq \boldsymbol{b}', \\ \mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1} \end{cases} \tag{11}$$

To observe this fact, we recall that the equality constraint $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ implies a constant sum of entries corresponding to each row/column of $\boldsymbol{X}$. Using such a constant-sum property, the connection between the solutions of P_0 and P_2 can be studied in a constructive way. Let $\boldsymbol{x}(k) \geq \boldsymbol{x}(l)$, and $\alpha > 0$. Then it is easy to verify that

$$(\boldsymbol{x}(k)+\alpha)^2 + (\boldsymbol{x}(l)-\alpha)^2 > (\boldsymbol{x}(k))^2 + (\boldsymbol{x}(l))^2. \tag{12}$$

The latter inequality implies that the l_2-norm of $\boldsymbol{x}$ can be increased monotonically by decreasing the small elements of $\boldsymbol{x}$ and increasing the large elements of $\boldsymbol{x}$ at the same rate— i.e. enhancing sparsity while increasing the l_2-norm. The latter result can be applied to a feasible $\boldsymbol{x}$ of P_0, meaning that a solution $\boldsymbol{x}$ of P_0 has the minimal l_0-norm (i.e. n) and at the same time the maximal l_2-norm (i.e. $\sqrt{n}$) among all feasible candidates in the search space of P_0.

We end this section with two remarks. First of all, the linear system in (9) and (11) describes a convex polytope, which we call the *Costas ball* in the sequel. Based on the above discussion, Costas arrays can be viewed as the farthest subset (in Euclidean sense) of vertices of the Costas ball from the origin, that intersect with the n^2-sphere

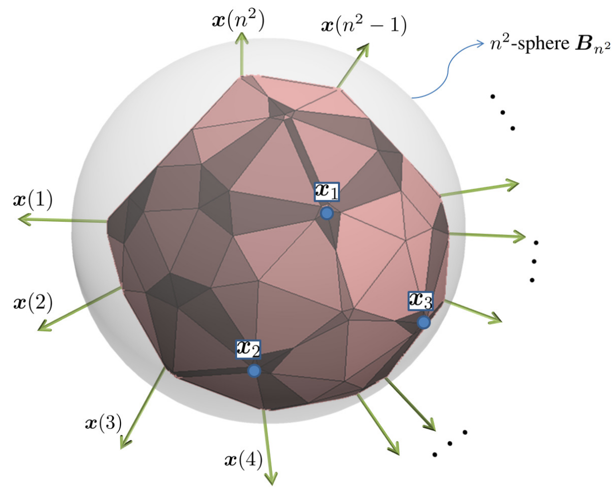


Fig. 2. An intuitive illustration of the Costas ball and the Costas arrays ($\{\boldsymbol{x}_k\}$, shown by the blue dots) possessing maximal l_2-norm. Costas arrays intersect with the n^2-sphere $\boldsymbol{B}_{n^2}$.

$$\boldsymbol{B}_{n^2} = \{\boldsymbol{x} \in \mathbb{R}^{n^2} : \|\boldsymbol{x}\|_2^2 = n\}. \tag{13}$$

This observation is illustrated in Fig. 2. Second, both optimization problems P_0 and P_2 are NP-hard in general; see [14], and [15],[16]. Nevertheless, such new formulations might be employed to devise numerical methods that can handle the Costas array search problems more effectively than the brute-force methods.

4. A NUMERICAL APPROACH— ALONG WITH AN ILLUSTRATIVE EXAMPLE

In order to show the potential of the proposed sparse formulation, we consider a reweighted iterative approach to tackle P_2. More precisely, starting from a random positive weight vector $\boldsymbol{w}^{(1)} \in \mathbb{R}^{n^2}$, we obtain the variable $\boldsymbol{x}^{(k)}$ (at the k^{th} iteration) using the linear program:

$$P_2 - reweighted: \max_{\boldsymbol{x}^{(k)}\in\mathbb{R}^{n^2}} \quad \boldsymbol{w}^{T\ (k)}\boldsymbol{x}^{(k)} \tag{14}$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{A}\boldsymbol{x}^{(k)} = \boldsymbol{b}, \\ \boldsymbol{A}'\boldsymbol{x}^{(k)} \leq \boldsymbol{b}', \\ \mathbf{0} \leq \boldsymbol{x}^{(k)} \leq \mathbf{1} \end{cases} \tag{15}$$

where the weight vector is updated as

$$\boldsymbol{w}^{(k+1)} = \boldsymbol{x}^{(k)} + \boldsymbol{\epsilon}, \quad k \geq 1 \tag{16}$$

and $\boldsymbol{\epsilon}$ is a random vector in $\mathbb{R}^{n^2}$ consisting of elements with i.i.d. standard Gaussian distribution $\mathcal{N}(0,\sigma)$. Note that $P_2 - reweighted$ aims to (approximately) maximize the l_2-norm, while it also provides a randomized leverage to skip the local optima of P_2 (corresponding to some vertices of the Costas ball with an l_2-norm less than $\sqrt{n}$). We stop the iterations when $\boldsymbol{x}$ becomes binary, and hence will represent a Costas array.

We note that, the reweighted method in (14)-(16) typically requires more iterations until convergence as n grows large. As an example, we consider using the proposed iterative approach to find a Costas array of size 8×8; a size which enables us to present the results through iterations. Due to the binary (i.e. $0/1$) nature of the ultimate results, we use a standard deviation σ of 0.5. The resultant 2D patterns $\boldsymbol{X}^{(k)} = vec^{-1}\left(\boldsymbol{x}^{(k)}\right)$ are shown in Fig. 3. In this example, the algorithm finds a Costas array after 6 iterations. It is worth observing that the 2D patterns obtained from successive iterations appear to be rather correlated (i.e. similar); at the same time, the obtained Costas array looks rather different from the pattern obtained at the end of the first iteration.

Finally, it is worthwhile to mention that, for large n, efficient implementation of the proposed approaches is crucial, and may be considered as an interesting topic for future works.

A. APPENDIX: THE NUMBER OF INEQUALITY CONSTRAINTS IN (7)

We begin by observing that the number of distinct (k,l)-vectors[2] in an $n \times n$ checkerboard is given by $(n-|k|+1)(n-|l|+1)$. Note that (i) the pattern described by three equidistant points on the same line is also a parallelogram whose two vertices share the same location. Moreover, (ii) each parallelogram including a (k,l)-edge is uniquely determined by placing two (k,l) vectors in the checkerboard. The number of all parallelograms including a (k,l)-edge in the $n \times n$ checkerboard is thus given by

$$\binom{(n-|k|+1)(n-|l|+1)}{2}. \tag{17}$$

Now we should exclude the parallelograms which have two vertices on the same row/column, except those for which the two mentioned vertices are exactly at the same location. The number of such parallelograms with two vertices on the same row is given by the number of $(t, 2l)$-vectors with $-(n-2|l|+2) \le t \le n$. Similarly, the number of such parallelograms with two vertices on the same column is given by the number of $(2k, t)$-vectors with $-(n-2|k|+2) \le t \le n$. On the other hand, the number of parallelograms with a (k,l)-edge and two vertices sharing the same location is given by the number of $(2k, 2l)$-vectors in the checkerboard. In sum, the number of parallelograms including a (k,l)-edge for which no two vertices occur on the same row/column unless they share a same location is equal to

$$\begin{aligned} f_{k,l} \;=\; & \binom{(n-|k|+1)(n-|l|+1)}{2} \\ & - \sum_{t=-(n-2|l|+2)}^{n} (n-2|l|+1)(n-|t|+1) \\ & - \sum_{t=-(n-2|k|+2)}^{n} (n-2|k|+1)(n-|t|+1) \\ & +2\,(n-2|k|+1)(n-2|l|+1) \end{aligned} \tag{18}$$

We note that by considering all $1 \le k \le n-1$ and $1 \le |l| \le n-1$, each parallelogram (except those with two vertices in the same location) will be counted two times in $\{f_{k,l}\}$. Therefore, the number of all parallelograms (corresponding to the linear inequality constraints) becomes

$$\begin{aligned} T_n \;=\; & \frac{1}{2}\sum_{k=1}^{n-1} \sum_{1 \le |l| \le n-1} f_{k,l} \\ & + \frac{1}{2}\sum_{k=1}^{n-1} \sum_{1 \le |l| \le n-1} (n-2|k|+1)(n-2|l|+1). \end{aligned} \tag{19}$$

Finally, it is worthwhile to observe that $f_{k,l} = \mathcal{O}(n^4)$ which implies that $T_n = \mathcal{O}(n^6)$ according to (19).

[2] i.e. the displacement vectors that map a point location (i,j) to the location $(i+k, j+l)$ in the checkerboard.

B. REFERENCES

[1] S. Rickard, "Open problems in Costas arrays," in *IMA International Conference on Mathematics in Signal Processing*, Cirencester, UK, 2006.

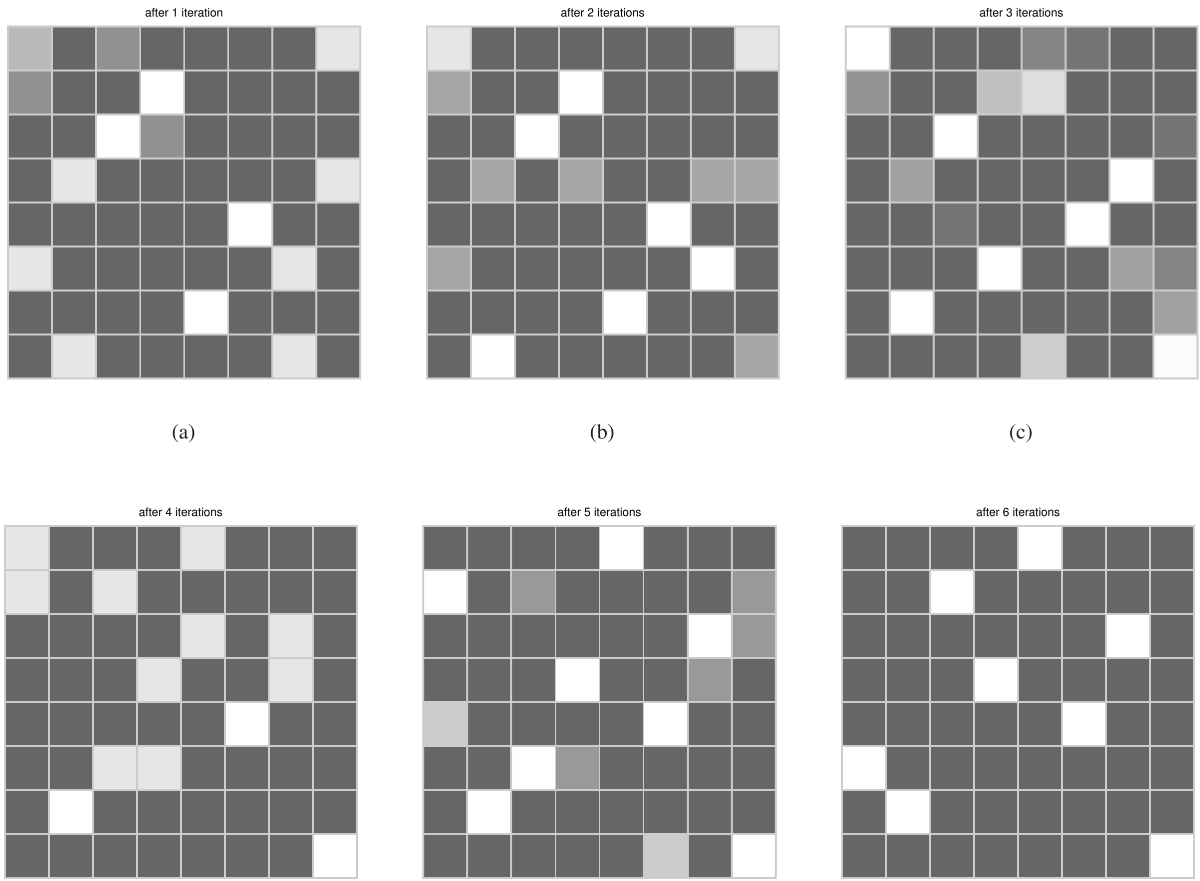
[2] K. Drakakis, "A review of Costas arrays," *Journal of Applied Mathematics*, vol. 2006, pp. 1–32, 2006.

[3] M. Sterling, E. L. Titlebaum, X. Dong, and M. F. Bocko, "An adaptive spread-spectrum data hiding technique for digital audio," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Philadelphia, PA, USA: IEEE, 2005, pp. 685–688.

[4] O. Moreno and S. Maric, "A new family of frequency-hop codes," *IEEE Transactions on Communications*, vol. 48, no. 8, pp. 1241–1244, 2000.

[5] S. Golomb and G. Gong, "The status of Costas arrays," *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4260–4265, 2007.

[6] K. Drakakis, F. Iorio, S. Rickard, and J. Walsh, "Results of the enumeration of Costas arrays of order 29," *Advances in Mathematics of Communications*, vol. 5, no. 3, pp. 547–553, 2011.



(a) (b) (c)

(d) (e) (f)

Fig. 3. The 2D patterns $\boldsymbol{X}^{(k)} = vec^{-1}\left(\boldsymbol{x}^{(k)}\right)$ obtained using the iterative reweighted approach proposed in (14)-(16). The values are shown using grayscale intensity. In this example, the method finds a Costas array after 6 iterations.

[7] L. Barker, K. Drakakis, and S. Rickard, "On the complexity of the verification of the Costas property," *Proceedings of the IEEE*, vol. 97, no. 3, pp. 586–593, 2009.

[8] M. Elad, *Sparse and redundant representations: from theory to applications in signal and image processing*. New York, NY: Springer, 2010.

[9] T. A. Davis, *Direct methods for sparse linear systems*. Philadelphia, PA: SIAM, 2006.

[10] P. Knee, "Sparse representations for radar with MATLAB examples," *Synthesis Lectures on Algorithms and Software in Engineering*, vol. 4, no. 1, pp. 1–85, 2012.

[11] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[12] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[13] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Las Vegas, NV, March 2008, pp. 3869–3872.

[14] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[15] P. M. Pardalos and S. A. Vavasis, "Quadratic programming with one negative eigenvalue is NP-hard," *Journal of Global Optimization*, vol. 1, no. 1, pp. 15–22, 1991.

[16] S. Zhang, "Quadratic maximization and semidefinite relaxation," *Mathematical Programming*, vol. 87, no. 3, pp. 453–465, 2000.