A STOCHASTIC ANALYSIS APPROACH IN THE SEARCH FOR COSTAS ARRAYS

Konstantinos Drakakis, John Healy & Scott Rickard

ABSTRACT: We propose and analyze two families of algorithms that search stochastically for Costas permutations: the first family is based on combinatorial optimization, whereby the Costas property is quantified by cost 0 of an appropriately selected cost function on permutations, and, starting by a randomly selected permutation, transformations (elements swaps) are performed iteratively on it aiming to the gradual reduction of the cost until 0 is hopefully reached. A detailed Bayesian analysis of this method, however, indicates that the rarity of Costas permutations at large orders condemns it to failure. The second family, which can also be construed as optimization, endeavors to construct a permutation incrementally, so that the Costas property is retained at each step of the increment process; we present an empirical analysis of this stochastic method as well, which again shows that it does not perform well for large orders. We conclude that, if a stochastic search is to succeed, its cost function needs to be very carefully selected.

1. INTRODUCTION

Costas permutations appeared for the first time in the literature in the 1960s as a model for optimal configuration of RADARs and SONARs [3, 4], in the sense that they prescribed frequency-hopping patterns with optimal noise robustness. The difficulty of their construction was quickly recognized, and in the 1980s the first (and last so far) mathematical methods for their construction, based on the theory of finite fields, were presented [5, 7, 8]. The Costas arrays so constructed, along with those found by the "brute force" method of exhaustive search, which up to this day has covered all orders up to and including 26 [1, 2, 10], were, until recently, practically the only ones known (a semi-empirical method found 4 new arrays [9]). Orders 32 and 33 are the smallest ones for which no example of a Costas permutation is known.

In this work we present two new approaches towards searching for Costas permutations: one based on combinatorial optimization, and one based on incremental construction (but which can also be construed as an optimization); the latter idea has already been used in some variations of the mathematical methods [8] and in the recent method mentioned above [9]. But before we proceed, let us give some precise definitions and descriptions of the problem.

2. PRELIMINARIES AND DEFINITIONS

2.1 Basics

Let $n \in \square$; we introduce the notation $[n] = \{1, \ldots, n\}$, and we denote, as usual, by a permutation of order n any bijection on $[n]$.

Definition 1: Let $p : [n] \to [n]$ be a permutation of order $n \in \square$. p has the Costas property iff

$$\forall i, j, k \in [n] : i + k, j + k \in [n], p(i + k)\ p(i) = p(j + k)\ p(j)\,) \Rightarrow i = j \vee k = 0$$

We will also need the concept of the *difference triangle* of a permutation:

Definition 2: Let p be a permutation of order n; its difference triangle $T(p)$ is the collection of vectors $\{ti(p) : i \in [n-1]\}$, where $t_i(p) = (t_{ij}(p) = p(i + j)\ p(j) : j = 1, \ldots, n - i)$. The vector $t_i(p)$ is the ith row of $T(p)$.
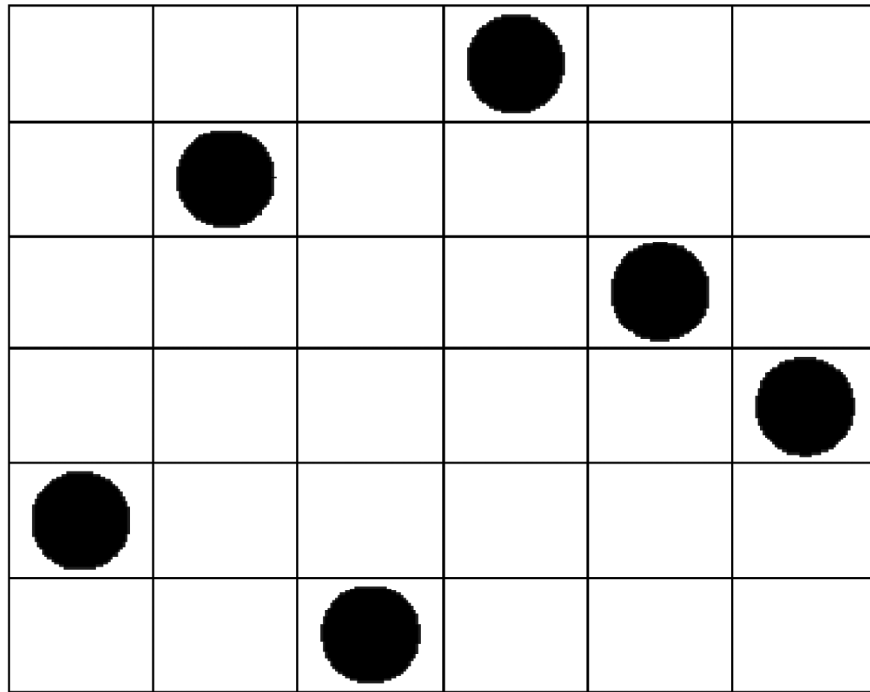
Figure 1: A 6 × 6 Costas Array

The difference triangle is aptly named because it actually looks like a triangle when written its rows are written one under the other and centered; its elements take values in the set $\{(n-1), \ldots, n-1\} - \{0\}$, and it is easy to see that it contains exactly $n-i$ absolute values equal to i in it, $i \in [n-1]$, as the absolute difference between any two integers in $\{1, \ldots, n\}$ is taken exactly once. It is also easy to see that the Costas property has the equivalent formulation that no row of the difference triangle contains two equal entries.

Example 1: The difference triangle for the Costas permutation 251643 (of order 6) is:

$$
\begin{array}{cccccc}
2 & 5 & 1 & 6 & 4 & 3 \\
 & 3 & -4 & 5 & -2 & -1 \\
 & & -1 & 1 & 3 & -3 \\
 & & & 4 & -1 & 2 \\
 & & & & 2 & -2 \\
 & & & & & 1
\end{array}
$$

Permutations correspond bijectively to permutation arrays, namely arrays whose elements are either 0 or 1, so that there is exactly one element equal to 1 per row and column. It is customary to represent 1s by dots and 0s by blanks in the graphical depiction of such an array. There are many alternatives in choosing the permutation array that corresponds to a particular permutation, and a convention must be followed:

Definition 3: Let p be a permutation of order $n \in \square$; the $n \times n$ permutation array $A^p = [a^p_{ij}]$ corresponding to p is constructed by setting $a_{n+1-p(i),i} = 1$, $i \in [n]$, and all other elements equal to 0.

Note that, according to this convention, the $p(i)$th element of column i, starting to count *from the bottom up*, is set to 1, $i \in [n]$.

Example 2: Figure 1 shows the Costas array corresponding to the Costas permutation 251643.

The array representation makes some results entirely obvious: for example, we can see immediately that the Costas property has the equivalent formulation that no four dots form a parallelogram and that for no three dots does one lie at the midpoint of the other two. As these relative dot positions remain unchanged under rotation, horizontal and vertical flip, and transposition of the array, we see that one Costas array leads to the construction of eight in total, or four in case it is symmetric.

We normally do not distinguish between permutations and permutation arrays and use the terms interchangeably: we will prefer below to talk about permutations rather than arrays, unless we want to specifically emphasize array aspects and properties.

2.2 Conflicts

The introduction of a "hard" definition for the Costas property divides the permutations of a given order into those that have the Costas property and those that do not. We would like now to introduce a "soft" definition instead, according to which a permutation has the Costas property to a certain degree:

Definition 4: Let p be a permutation of order $n \in \square$ and set $w_{IJ}(p) = |\{j|t_{Ii}(p) = J\}|$, $I = 1, \ldots, n-1$, $J = (n-1), \ldots, n-1$; the *number of duplicate entries* is $D(p) = \sum\limits_{\{(i,j)|w_{ij}(p)>0\}} (w_{ij}(p) - 1)$, and the *number of conflicts* $C(p) =$

$$\sum\limits_{\{(i,j)|w_{ij}(p)>0\}} \binom{w_{ij}(p)}{2} = \sum\limits_{\{(i,j)|w_{ij}(p)\geq 2\}} \binom{w_{ij}(p)}{2}.$$

Clearly, p has the Costas property iff $D(p) = C(p) = 0$, but it is now natural to think that the smaller $D(p)$ or $C(p)$ is, the "closer" p is to being Costas: if $C(p_1) < C(p_2)$, or equivalently $D(p_1) < D(p_2)$, where p_1 and p_2 are permutations of equal order, then p_1 is "more Costas" than p_2. Both D and C then are appropriate cost functions that "penalize" deviations from the Costas property, and they are clearly equivalent: D leads in general to smaller costs, but because C is more appealing intuitively we will focus on this one from now on. Indeed, revisiting Definition 1, we see that $C(p)$ is the total number of pairs (i, j), $i, j \in [n]$, $i < j$ for which there exists $k > 0$ so that $p(i + k) \ p(i) = p(j + k) \ p(j)$; such pairs are perceived as "conflicts" from the point of view of the Costas property.

We can be even more specific and determine the number of conflicts a particular element of p is involved in:

Definition 5: Let p be a permutation of order $n \in \square$; for $i \in [n]$, set $c_p(i) = |\{j \in [n] : \exists k > 0 : p(i + k) - p(i) = p(j + k) - p(j)\}|$ to be the ith coordinate of the *conflict vector* c_p.

In other words, $c_p(i)$ is the number of pairs of equal entries in the rows of the difference triangle caused by p(i); it easily seen that each entry that has a duplicate in the same row of the difference triangle contributes a "hit" to exactly two elements of p.

Example 3: Consider again the Costas permutation $p = 251643$. If we swap the first two entries, then we obtain the permutation 521643, whose difference triangle becomes

$$
\begin{array}{ccccccccccc}
5 & & 2 & & 1 & & 6 & & 4 & & 3 \\
& -3 & & -1 & & 5 & & -2 & & -1 & \\
& & -4 & & 4 & & 3 & & -3 & & \\
& & & 1 & & 2 & & 2 & & & \\
& & & & -1 & & 1 & & & & \\
& & & & & -2 & & & & &
\end{array}
$$

where the conflicts (repeated values within one row) are shown in bold. Each conflict is associated with two array elements: for example, the first 1 on the first difference row is associated with 2 and 1 while the second 1 on this row is associated with 4 and 3. Similarly, the first 2 on the third difference row is associated with the 4 and the 2, and the second 2 on this row is associated with the 3 and the 1. We construct the conflict vector by counting the number of conflicts each element of the permutation is associated with, and we find it to be $c_p = (0\ 2\ 2\ 0\ 2\ 2)$.

2.3 Hamming Distance

We shall need a measure of the proximity of two permutations p and q of the same order n. The measure we will henceforth adopt is the Hamming distance between them, namely the number of unequal corresponding elements:

Definition 6: Let $n \in \square$, and let p and q be permutations on $[n]$; their Hamming distance is $H(p, q) = |\{i \in [n] : p(i) \neq q(i)\}|$.

Observe that 1 cannot possibly belong in the range of H, but every other value in $[n]$ can, as can 0. H can easily be seen to satisfy all of the defining properties of a distance function on the set of permutations of a given order, namely symmetry, non-negativity, the triangle inequality, and that $H(p, q) = 0 \Rightarrow p = q$.

3. COMBINATORIAL OPTIMIZATION

Thanks to the number of conflicts (or, alternatively, the Hamming distance) we can tell when a permutation is "near" a Costas permutation: in this case, swapping around a small number of elements transforms the permutation into a Costas one. Such knowledge can be useful as it may allow us, from a random permutation, to progressively move towards and eventually find a true Costas permutation. A general algorithm could, for example, be the following:

ALGORITHM 1

1. Start from a random permutation p_0; if it is Costas, stop.

2. Test a set of permutations created by swapping a small number of elements of the permutation p_0 (creating a set of permutations $p_1, p_2, \ldots, p_I$), for some $I \in \square$.

3. Select the permutation p_i that is "most" Costas in the set (the details get filled in by the specific algorithm used).

4. If p_i is Costas, stop. If not,

5. If $C(p_i) > C(p_0)$ p_i, namely p_i is not more Costas than p_0, goto Step 1. If not,

6. Set p_0 p_i and return to Step 2.

If successful and efficient, this algorithm can be used to "fill the gaps" for orders where the existence or the enumeration of Costas arrays is not yet completely settled, namely for orders $n > 26$. But the proposed method relies on the ability to estimate the minimum number of elements of a permutation that need to be swapped in order to turn it into a Costas one (i.e. the Hamming distance from the nearest Costas permutation), as well as the elements themselves; after proposing various intuitively likely methods for estimating this distance, we will provide experimental evidence that search techniques of this type are not likely to succeed for the orders we are interested in.

3.1 Distance to the Nearest Costas Permutation

We ask the following questions?

1. Is it possible to tell how many and which elements need to be swapped to turn a non-Costas permutation array into a Costas one?

2. Is it possible to estimate the Hamming distance to the nearest Costas permutation from the conflict vector of a non-Costas permutation?

To address these questions, we perform the following test:

Test 1

1. Given the order n, start with a random Costas permutation of this order from the database.

2. Randomly choose and swap k elements in it.

3. Compute the conflict vector.

4. Check to see if the most conflicted element is among the swapped elements.

5. Check to see if the second most conflicted element is among the swapped elements.

6. Calculate the maximum and the sum of the conflict vector.

7. Calculate the number of non-zero entries in the conflict vector, namely the number of elements with conflicts.

The results of Test 1 (for 100,000 random swappings for each choice of k) are shown in Table 1 for $n = 25$. They show that the conflict score is associated with being out of position and that the sum of the conflict vector (total conflict) is indicative of the Hamming distance of the array.

Table 1
Results of the One Swap Conflict Vector Tests on the 88 Costas Permutations of Order 25, as Described in Test 1. k is the Hamming Distance of the Tested Permutation to the Closest Costas One. The '1st' and '2nd' Refer to the Percentage of Time the Most and Second Most Conflicted Elements are among the Swapped Elements. The Final three Columns are the Average Maximal Value in the Conflict Vector, Average sum of the Entries of the Conflict Vector, and the Average Number of Elements with Conflicts

k	1st	2nd	max conflict	total conflict	non-zero
2	0.999670	0.950300	12.0	77.6	20.23
3	0.997200	0.960690	13.4	111.1	22.42
4	0.984920	0.931600	14.1	137.7	23.47
5	0.954920	0.893030	14.8	161.4	24.00
6	0.906150	0.856240	15.5	181.0	24.30
7	0.852050	0.824030	16.2	197.6	24.48
8	0.801020	0.796310	16.7	211.0	24.59
9	0.757990	0.779380	17.3	222.5	24.66
10	0.724220	0.766190	17.8	231.4	24.71

3.2 Distance between Costas Permutations

We also perform tests to see how far apart Costas permutations are for a given order n. The above interpretation of the results in Table 1 relied on the assumption that an array which is at Hamming distance 10 from a given Costas permutation of order 26 is not closer to any other Costas permutation of order 26. On the other hand, if Costas permutations lived in clusters, then finding one could lead to finding several others as well. With this in mind, we measured the Hamming distances for all pairs of Costas permutations for a given n. We summarize the test results in Table 2. We see that, for $n > 17$, the Costas arrays are far apart. For example, for $n = 22$, the closest any two Costas permutations are involves permuting 12 entries. For $n = 23$, four permutations are just one swap from each other, but beyond this special case, all permutations are at least at Hamming distance 12 from each other.

There is one very unique pair of permutations which, in consideration of its rotations, generates the 4 cases of Hamming distance 2 in the $n = 23$ case. It turns out this permutation of order 23 is representable as a symmetric array with two opposing corner dots (see Definition 3). In this case, the corner dots can be simultaneously swapped to the empty corners and the Costas property is preserved. Because the array is symmetric, each array produces only four arrays via rotation and reflection (instead of the usual eight), and the 4 cases of Hamming distance 2 are explained via this pairing and its rotations. These arrays are shown in Figure 2. Note that these two Costas permutations of order 23 contain embedded Costas permutations of order 21 corresponding to symmetric arrays.

Curiously enough, the various known algebraic techniques for the construction of Costas permutations [5, 7] do not generate these permutations. This leads to the question: what other arrays have this double corner dot swap property? The answer is: none for $2 < n \leq 26$. An exhaustive search of the database of all Costas permutations reveals that, with the exception of the degenerate $n = 2$ case, the arrays pictured in Figure 2 are the only Costas

Table 2
Table of Average Hamming Distances between Pairs of Costas Arrays for $n \leq 26$. n = order of Costas Permutation. Hbar = Average Hamming Distance between all Pairs of Costas Permutations. H2 = Number of Occurrences of Hamming Distance of 2. Hk = First Distance for ($k > 2$) with Non-zero Number of Occurrences

N	Hbar	H2	Hk
2	2	1	n/a
3	2.33	4	3
4	3.27	16	3
5	4.05	61	3
6	5.03	76	3
7	5.96	72	3
8	7	152	3
9	7.97	80	3
10	8.97	176	3
11	9.96	140	3
12	10.98	136	3
13	11.97	88	3
14	12.97	120	3
15	13.97	76	3
16	14.97	24	3
17	15.97	16	3
18	16.96	0	6
19	17.94	0	9
20	18.96	0	11
21	19.95	0	10
22	21.17	0	12
23	21.96	4	12
24	22.97	0	17
25	23.85	0	21
26	25.04	0	22
42 (Welch)	41.04	0	28
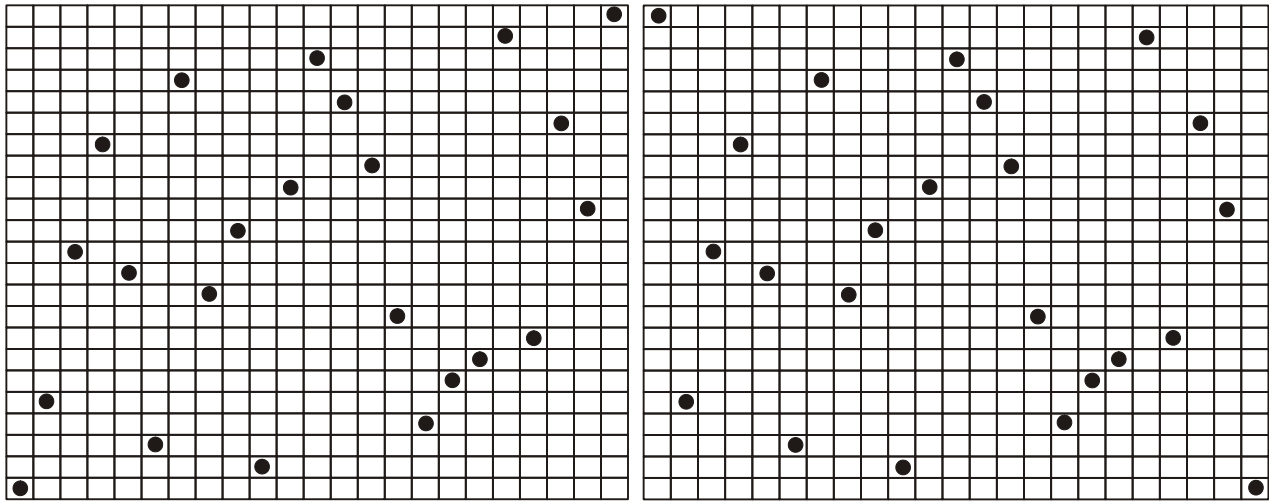
Figure 2: Costas Array Pairing of Order 23 Related by Swapping First and Last Columns, both of which Contain Corner dots. The Embedded 21-by-21 Costas Array Obtained by Ignoring the First and Last Column is Symmetric, but Not a Result of the Algebraic Constructions Available

arrays which have two corner dots which can be swapped to the empty corners without the resulting array losing the Costas property – a very interesting Costas array indeed.

It appears that for $n > 17$, Costas permutations get further and further apart, with the exception of the notable $N = 23$ pairings. Based on the results from the two tables in this section, we conclude that:

1. Costas permutations of size $n > 17$ are on average far apart from each other.

2. The sum of the conflict vector of a permutation correlates well with the Hamming distance between the permutation and the closest Costas permutation.

3.3 Algorithms

Motivated by the results above, we consider five implementations of the iterative scheme stated in Algorithm 1 to search for a Costas permutation starting with a random one: each one uses the summation of the conflict vector as a measure of distance from a Costas permutation, but they differ in the number I, and the selection of permutations in the consideration set $p_1, p_2, \ldots, p_I$. These five methods are:

1. **max-max:** $I = 1$ and the only permutation considered is the one created by swapping the two most conflicted elements.

2. **simple-max:** $I = N - 1$ and the permutations considered are the ones created by swapping the most conflicted element with each one of the other elements in the permutation.

3. **greedy-2:** $I = \binom{N}{2}$ and the permutations considered are created by all possible pairwise swaps of elements.

4. **greedy-3:** $I = \binom{N}{2} + 2\binom{N}{3}$ and the permutations considered are all possible pairwise or three-way swaps of elements.

5. **greedy-4:** $I = \binom{N}{2} + 2\binom{N}{3} + 9\binom{N}{4}$ and the permutations considered are created by all possible pairwise, three-way or four-way swaps of elements.

In order to compare the performance of the five proposed versions of the stochastic search algorithm, the following tests are performed: for a Costas permutation of order $n = 26$, a random pair of elements is swapped and then the five stochastic methods are run to determine which ones are able to recover a Costas permutation from this starting point. If at any point during the stochastic search pi fails to be more Costas than p0, then the specific method used is declared to have failed (instead of returning to Step 1). This is repeated for every Costas array of size 26, for every Hamming distance 2 through 9, for 56 random tests. The success rate (the number of times a Costas permutation was recovered out of the 1,456 tests) for each method for each Hamming distance is determined and the results are shown in Figure 3 and Table 3.

Table 3

Table of Stochastic Search Algorithms Performance. Percentage Success in Finding a Costas Permutation from a Starting Permutation which is at Hamming Distance k from a Costas Permutation of order 26 for the Five Methods Proposed: Max-max, Simple-max, Greedy-2, Greedy-3, and Greedy-4

k	*max-max*	*simple-max*	*Greedy-2*	*Greedy-3*	*Greedy-4*
2	88.26	58.86	100	100	100
3	7.69	33.38	81.87	100	100
4	0	17.79	85.44	89.42	100
5	0	7.21	71.7	85.44	89.63
6	0	2.75	48.49	69.17	84.07
7	0	1.24	25.96	46.81	60.65
8	0	0.14	6.52	14.15	25.62
9	0	0.07	2.23	3.43	6.46
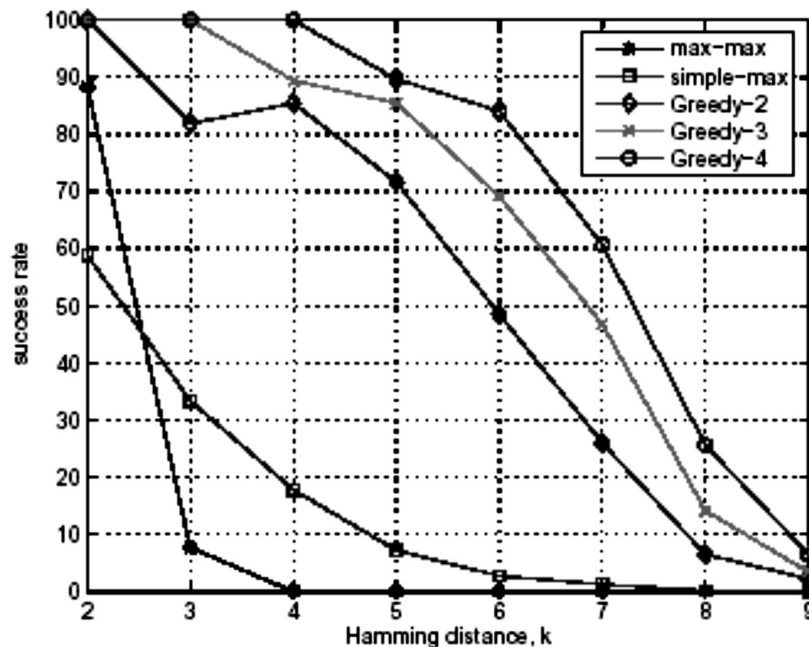
Figure 3: Table of Stochastic Search Algorithms Performance. Percentage Success in finding a Costas Permutation from a Starting Permutation which is at Hamming Distance k from a Costas Permutation of Order 26 for the Five Methods Proposed: Max-max, Simple-max, Greedy-2, Greedy-3, and Greedy-4

The results demonstrate that it is necessary to be fairly close (in terms of Hamming distance) to a Costas permutation in order to be reasonably certain to find it and that the larger the number of permutations consider at each step is, the more likely it is to find a Costas permutation. Note that this comparison unfairly penalizes (in some sense) the simpler methods in that it treats each run from a starting permutation as a unit of cost despite the fact that the simplest method (max-max) considers one potential permutation based on the starting point at each iteration, while the most complicated (Greedy-4) considers over 140,000 potential permutations at each iteration.

3.4 Analysis of the Algorithms

Test runs of the above algorithms fail to yield any Costas permutations (except for small n). To see why this is the case, we examine the histograms of sum conflict scores as a function of Hamming distance for a given n. These histograms (appropriately normalized) can be thought of, in some sense, as the probability density functions of the total conflict score for each Hamming distance. If the probability density functions separate nicely (for example, if they had disjoint support), then the proposed methods would be justified. If however, the probability density functions overlap and the most likely Hamming distance is large, regardless of the value of the sum conflict score, the proposed methods are doomed.

Figure 4 shows the histograms of sum conflict scores for various Hamming distances k for Costas permutations of order 25. Each line in the graph is the histogram of 50,000 sum conflict scores calculated by selecting random Costas permutations and swapping k elements.

Histograms are generated for all values of k from 2 to 23; we see that each histogram is very well approximated by a Gaussian, as the Kolmogorov-Smirnov distance between each histogram and the corresponding Gaussian with the same mean and variance is in all cases less than 0.04. The estimated means and variances, as functions of k, are shown in Figure 5. Figure 6 contains the Gaussian approximation of the probability density functions for each k. From these plots, we can indeed see that increasing conflict scores can be associated with increasing Hamming distances. One might be tempted to conclude from the graphs that if a given permutation of order 25 has a conflict score of 20, the most likely explanation is that it is at Hamming distance 2 away from being a Costas permutation. Unfortunately, this is not the case, as we must, of course, consider the number of permutations at each Hamming distance from a given Costas permutation: due to the fact that this number grows exponentially as the Hamming distance increases, we are about to see, using Bayes' law, that, regardless of the total conflict score, the permutation is highly likely to be at least at Hamming distance 10 from its closest Costas permutation.
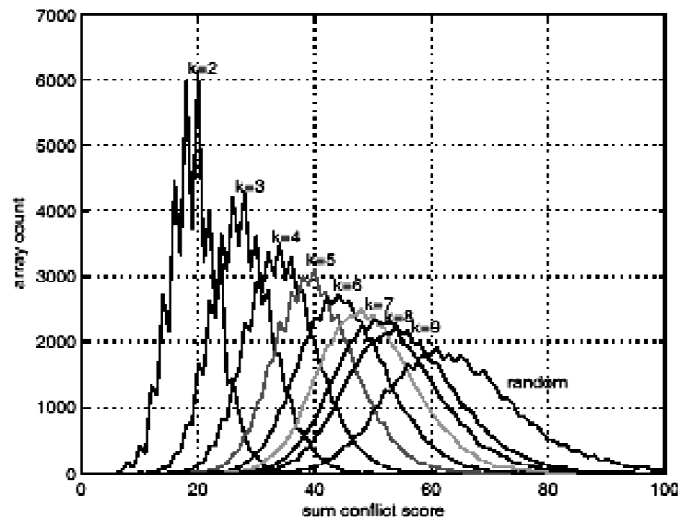

Figure 4: Histograms of Sum Conflict Scores for Permutations of Hamming Distance k away from a Costas Permutation of Order 25. The Rightmost Line is the Histogram of Sum Conflict Scores for 50,000 Random Permutations of order 25

Assuming $n = 25$, Figure 6 and Figure 5 essentially show $p(s|k)$, the probability of a given sum conflict score given the permutation is at Hamming distance k from the closest Costas permutation, which is the likelihood term in Bayes' law below. Also, we assume that we have an estimate of $p(s)$ which is the distribution of sum conflict scores for a random permutation; this is the evidence term in Bayes' law below, which, as is often the case, we treat as a (rather indifferent) normalization constant. Finally, we can calculate explicitly the probability that a permutation be at Hamming distance k from the closest Costas one (for $k \leq 10$), using the results in Table 2, according to which no two Costas permutations of order 25 share more than 4 entries, and the following theorem:

Theorem 1: Assuming than no two Costas arrays of order n share more than n_s entries, the probability that a permutation be at Hamming distance $1 \leq k \leq k_{\max} := \dfrac{n - n_s}{2}$ from its closest Costas permutation is given by

$$p(k) = d(k)\binom{n}{k}\frac{C(n)}{n!} \approx \frac{C(n)}{e(n-k)!}, k = 0, 1 \ldots, k_{\max}$$

where $C(n)$ is the number of Costas arrays of order n and $d(k)$ the number of derangements of order k, namely the number of permutations with no fixed point.

Proof: It is clear that, within the range for k given, the spheres of radius $k_{\max}$ around different Costas permutations do not intersect: therefore, the total number of permutations lying at distance k from the closest Costas permutation is just the sum of the numbers of all permutations lying at distance k from each Costas permutation, and this number in turn is the number of ways k elements of the Costas permutation can be chosen out of n and be deranged. This number needs to be divided by the total number of permutations $n!$ and this proves the equality in the statement.

The approximation is obtainable by the well known asymptotics of derangements, according to which $\dfrac{d(k)}{k!} \approx \dfrac{1}{e}$. This completes the proof.
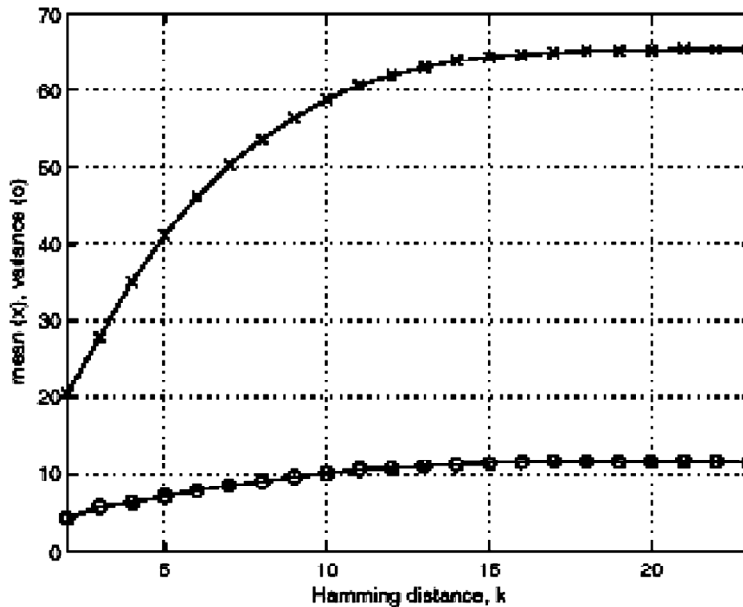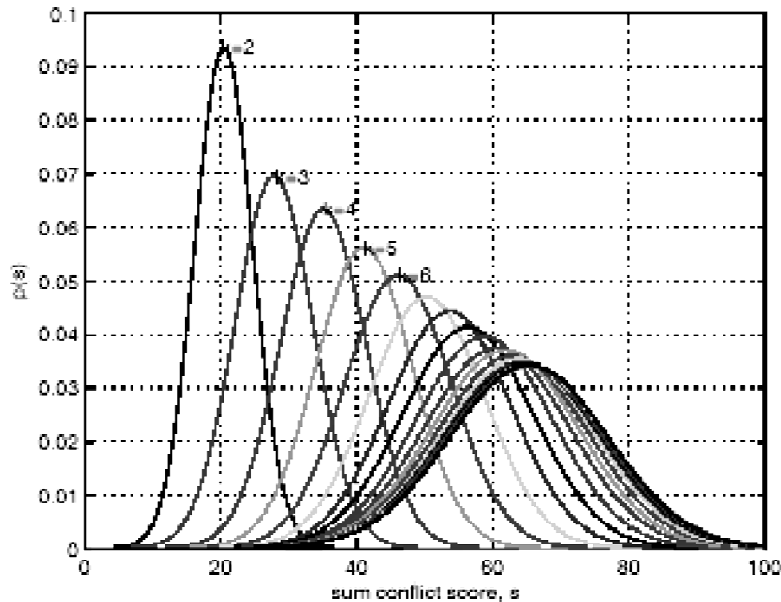

Figure 5: Estimated Mean and Variance of the Probability Density Functions of Sum Conflict Scores for Permutations at Hamming Distance k away from a Costas Permutation of Order 25

$p(k)$ is the prior probability in Bayes' law below. For $n = 25$, we indeed obtain $k_{max} = 10$ by the theorem and

$$p(k = 0) \;=\; 88/25!$$

$$p(k = 1) \;=\; 0$$

$$p(k = 2) \;=\; \binom{25}{2} *88/25!$$

$$p(k = 3) \;=\; 2\binom{25}{3} * 88/25!$$

$$p(k = 4) \;=\; 9\binom{25}{4} * 88/25!$$

$$p(k = 10) \;=\; d(10)\binom{25}{10} *88/25!$$

The approximation yields $p(k) \approx \dfrac{88}{e(25 - k)!}$. We are interested in $p(k|s)$, the probability of a certain Hamming distance k given the sum conflict score, the posterior probability in Bayes' law, which takes the form:

$$p(k|s) = \frac{p(s|k)p(k)}{p(s)}$$

We are particularly interested in the maximum likely Hamming distance argmax $_k p(k|s)$ as a function of s: that is, if we have a permutation with a sum conflict score of s, what is the most likely Hamming distance? Analysis of the probability ratios show that, for example, we are more than 1,000,000 times more likely to be at Hamming distance 10 from a Costas permutation than at Hamming distance 2, for all values of the conflict score. For

Figure 6: Gaussian Approximations of the Probability Density Functions of Sum Conflict Scores for Permutations of Hamming Distance k away from a Costas Permutation of Order 25

illustrative purposes, we plot the ratio $\dfrac{p(k=i+1|s)}{p(k=i|s)}$ for $i = 2, 3, \ldots, 9$ in Figure 7. For $s > 25$, the ratios increase exponentially and the larger Hamming distances are exponentially more likely than the smaller ones. Coupled with the search algorithm results, we conclude that the stochastic search is unlikely to yield a Costas permutation.

4. INCREMENTAL CONSTRUCTIONS

Let $p = p(1) \ldots p(n)$ be a Costas permutation for some $n \in \square$ and let $k < n$: then $p(1) \ldots p(k)$ and $p(k) \ldots p(n)$ are both strings that have the Costas property, but are not complete permutations. This simple observation can give rise to a new family of construction algorithms for Costas permutations:

4.1 Costas Strings

We propose the following algorithm:

Algorithm 2

1. Let $n \in \square$, let $S_n = \{1, \ldots, n\}$ with no numbers flagged, consider s_0 to be the empty string, and set $i = 0$.

2. If S_n is empty, signal successful stop; if all numbers are flagged, signal unsuccessful stop; otherwise, choose randomly a number in S_n that is not flagged, say x, and remove it from S_n.

3. Check whether $s_i x$ or $x s_i$ has the Costas property; as soon as one is found to have it, set it to be s_i+1, stop searching, set $i \leftarrow i+1$, and unflag all elements in S_n; if none of the two has it, flag x, put it back in S_n, and go to step 2.

We could perhaps modify the algorithm so that we test new integers by placing them within the existing string as well as at its side. The last step then should be substituted by:

3'. For all possible decompositions u and v such that $s_i = uv$ (u and v are allowed to be empty), check whether uxv has the Costas property; as soon as one is found to have it, set it to be s_{i+1}, stop searching, set $i \leftarrow i+1$, and unflag all elements in S_n; if none has it, flag x, put it back in S_n, and go to step 2.

Experiments show, however, that the gain from testing the extra cases is negligible compared to the increase in complexity. So we will not follow this alternative, but we will return to this idea in Section 4.3, when we talk about anti-Costas strings.
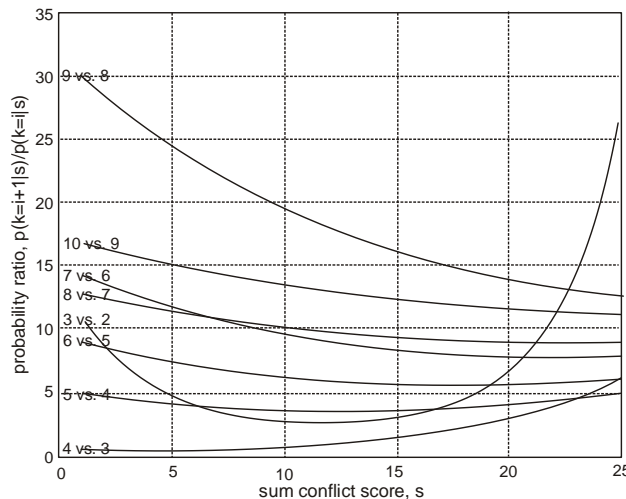

Figure 7: Ratio of $p(k = i + 1|s)$ to $p(k = i|s)$ for $i = 2, 3, \ldots, 9$.

It is easy to see that this algorithm is essentially is a completely random search with no additional criteria: a successful stop means that a Costas permutation has been found, whereas the unsuccessful stop signals that the string with the Costas property found so far does not contain all numbers yet, and also that it cannot be augmented further without losing the Costas property; therefore, we face a dead end, and we need to start a new search.

Example 4: Suppose that we are working with order $n = 6$, and we have formed $s_4 = 6513$ (this is a Costas string). We want to introduce $x = 2$, and the possible alternatives are 26513 and 65132. The former has the Costas property, while the latter does not, and so the former gets chosen as s_5.

4.2 Mean Maximum Length

When we run Algorithm 2 with a particular n, we get as output a string of length $L(n)$, so that $3 \leq L(n) \leq n$; if $L(n) = n$, the permutation constructed has the Costas property. Obviously, $L(n)$ is a random variable. Imagine now we run this algorithm m times in independent trials, and that we get the lengths $L_i(n)$, $i \in [m]$; then, we compute $L^m(n) = \max_{i \in [m]} L_i(n)$, which is also a random variable. What is $f_n^m = \mathbb{E}[L^m(n)]$?

The ratio $\dfrac{f_n^m}{n}$ is a measure of the difficulty to construct a Costas permutation for a given n, if we follow the completely random method of Algorithm 2. We will argue now that if m is large, then $\mathrm{Var}[L^m(n)]$ is negligible, so that $L^m(n)$ is, for all practical purposes, deterministic. We use the following conjecture, which seems to be corroborated by experiments:

Conjecture 1: For sufficiently large values of L, and for a given n and large m:

- There exists a slowly increasing sequence $\{k_m\}_{m \in \mathbb{N}}$ such that $m\mathbb{P}(L(n) = k_m) = O(1)$;

- $\mathbb{E}(L(n) = k)$ is decreasing fast.

If we now set $\mathbb{P}(L(n) \leq k) = P(k)$, $k \in [n]$, a well-known result in Probability Theory is that $\mathbb{P}(L^m(n) \leq k) = P^m(k)$. Given that $P^m(n) = 1$ and $P^m(0) = 0$, the exponentiation acts like a hard thresholder. There will be a value $k_m < n$ so that:

- $P^m(k) \approx 0$ if $k < k_m$, as $P^m(k) < 1$ there;

- $P^m(k) \approx 1$ if $k > k_m$;

- $P^m(k_m)$ will be small but perhaps not completely negligible, because $P^m(k_m) \approx (1-\alpha/m)m \approx e^{-\alpha}$, for some $\alpha > 0$.

This amounts to saying that all of the probability mass is effectively concentrated on k_m+1, with a slight correction perhaps needed for k_m, so that $L^m(n)$ is "deterministic". Figure 8 shows these distributions for $n = 35$, and is typical for all values of n in general.

Now that we can be sure that $f_n^m = \mathbb{E}[L^m(n)] \approx L^m(n)$ (for one set of m runs) is easy to compute accurately (by one set of m runs only), we can focus on its growth with respect to n. We run Algorithm 2 for all values of n between 20 and 100 with an increment of 10, using $m = 1000000$ throughout; the results are shown in Figure 9, where we also show the linear t to the log-log plot, which is extremely good, and returns an exponent close to 2/3.

Clearly, if we let $m \to \infty$, we eventually get all the probability mass concentrated on $k = n$; our point is that in practice we are bound to use a finite value for m, and for such a value k_m may be quite far from n. Also, the slow increase of k_m implies that even if we double or triple the m we use, k_m will only increase by little, an will remain far from n.

Unfortunately, this result implies that this incremental construction will almost always result to a Costas string of length substantially less than n (about $n^{2/3}$ for large n in our example), hence it will almost always fail to produce a Costas permutation.
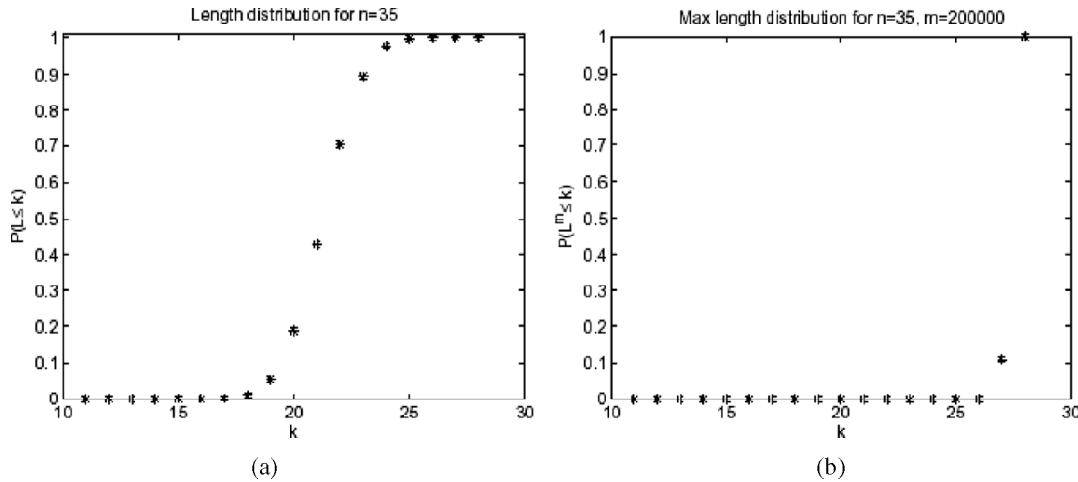
Figure 8: Distributions of Length of the Output of Algorithm 2 for $n = 35$: (a) One run; (b) Maximum of 200000 Runs. The Thresholding Effect is Clear: $k_m = 27$ and the Maximum is almost Deterministic and Equal to Approximately 28 (with a Small Correction for the Small Probability of 27)

Note finally that this algorithm too can be perceived as minimization of a cost function: in our case, the cost function is $n - L$, where L is the length of the string used.

4.3 Anti-Costas Strings

The data used in Figure 8(a) represents the upper bound on the length of the string that Algorithm 2 can return (for a given number of repetitions m); the largest value of this upper bound is, of course, n, and when the algorithm produces a string of such length, it produces a Costas permutation. But it seems equally interesting to look into the lower bound on the length of the string that Algorithm 2 can return: such short strings have the very interesting property that they are the shortest strings determined that cannot be extended to larger Costas strings. The smallest value of this lower bound is certainly well defined for a given n and depends only on n. This suggests the idea for the following definition of *anti-Costas strings*:

Definition 7: A string s of integers in $[n]$ will be called an anti-Costas string of order n iff it has the following properties:

- Every integer in $[n]$ is contained in s at most once;

- There are no strings u and v (possibly empty), and there is no integer $x \in [n]$ not already belonging to u or v, such that $s = uv$ and uxv has the Costas property.

- There is no string ts satisfying the previous two properties that is shorter than s.

In other words, anti-Costas strings are those Costas strings that can in no way be extended into longer strings while retaining the Costas property, even if we attempt to interpolate in the middle of the string the integers not already included in it, instead of just appending them at the sides; this slight difference between this definition and Algorithm 2 (remember the discussion in Section 4.1) is really negligible: simulations show it is very unlikely to be able to extend a Costas string through interpolation but not through appending.

It seems very likely that determining anti-Costas strings is as hard as determining Costas permutations, if not harder. In a relevant experiment we carried out, we ran Algorithm 2 1000000 times with $n = 35$, and the shortest non-extensible string it produced was of length 16:

| 26 | 8 | 33 | 10 | 22 | 3 | 19 | 14 | 2 | 12 | 18 | 17 | 24 | 25 | 28 | 6 |

At this moment, though, we don't have a calculation method for the length of anti-Costas strings of order n, for a given n, nor do we have constructing algorithms like the ones for Costas permutations (see [5, 7, 8]). In the
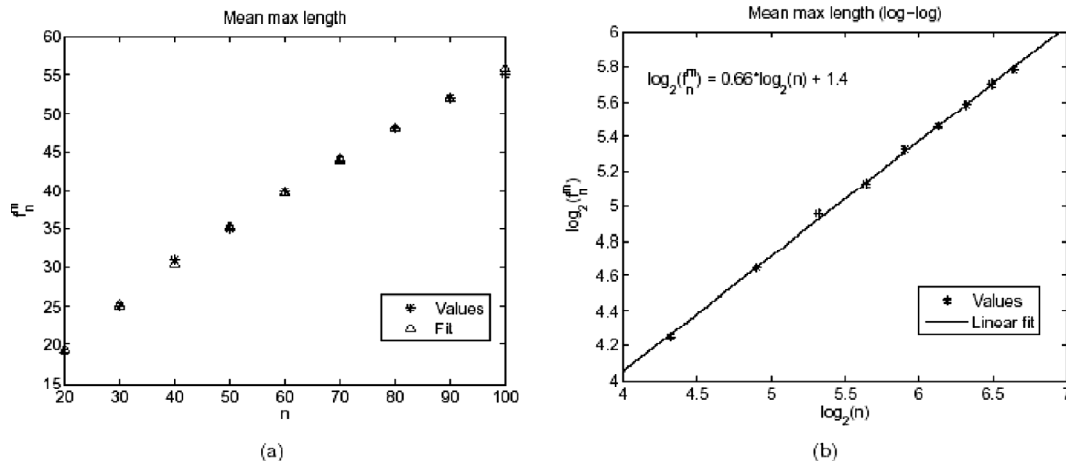
Figure 9: Mean Max Length f_n^m for $m = 105$ for Values of n between 20 and 100 with step 10; the Log Plot Shows that the Logarithms of the Lengths lie with High Accuracy on a Straight Line, So that the Lengths Themselves Follow a Power Law with a Power Close to 2/3. Observe that the Linear t is Remarkably Good

case of Costas permutations, the former was not needed, as it is always n, but, on the other hand, their existence needed to be proved (we don't know whether Costas permutations exist for all n [5, 8]), whereas anti-Costas strings always exist.

5. SUMMARY AND CONCLUSION

In this paper, we presented two families of algorithms that search randomly for Costas permutations by shaping an object progressively and iteratively until it becomes a permutation with the Costas property: the first family modifies permutations till they acquire the Costas property (or cannot be modified further), while the second family extends strings of integers that have the Costas property till they become full permutations (or cannot be further extended). In a sense, then, the two families exhibit a duality. A common feature they share is that they can both be considered minimization procedures over appropriately selected cost functions.

More specifically, we showed that:

1. The number of conflicts a permutation element is associated with is an indicator about whether it should be swapped with another element to make the permutation closer to a Costas one.

2. For $n > 17$ the Costas arrays are far apart, that is, they require a relatively large number of swaps to transform one Costas permutation into another. There is one notable exception: there are two pairs Costas permutations of order 23 that only require one swap to permute one into the other. Moreover, the dots that need to be swapped are both corner dots, and the core arrays are symmetric.

3. The Hamming distance of a permutation to the nearest Costas one is a good indicator of the number of conflicts in the difference triangle of the permutation.

4. The number of conflicts in the difference triangle of a permutation is not a good indicator of the Hamming distance of the permutation to the nearest Costas one.

5. In the specific case of order 25, which we studied in detail, stochastic search (iteratively swapping elements so as to minimize the total number of conflicts in the difference triangle) only succeeds in nding a Costas permutation if the initial permutation is near a Costas one; but random permutations are not likely to be near a Costas one.

6. Incremental constructions that extend a string with the Costas property into a Costas permutation are very likely to get stuck before reaching their goal. In the process of studying these strings, we wondered

whether it is possible to characterize the non-extendable strings of the shortest length, but we made no progress towards this direction.

Overall then, it seems that our random searches, which are essentially optimization processes based on some simple and intuitively promising cost functions, fail to deliver. This should not be perceived, however, as a failure of the concept of random search based on optimization techniques, but rather of the specific implementations of the random search we presented, namely of the specific cost functions we considered. It is likely that better suited cost functions are available, and we relegate their discovery to future work.

REFERENCES

[1] J. K. Beard. "Combinatoric Collaboration on Costas Arrays and Radar Applications", Slide presentation in\ RADARCON (2004).

[2] J. K. Beard, J. C. Russo, K. Erickson, M. Monteleone, M. Wright. "Combinatoric Collaboration on Costas Arrays and Radar Applications", IEEE Radar Conference, Philadelphia, PA, USA (2004).

[3] J. P. Costas. "Medium constrains on sonar design and performance", Technical Report Class 1 Rep. R65EMH33, GE Co.

[4] J. P. Costas. "A study of detection waveforms having nearly ideal range-doppler ambiguity properties", Proceedings of the IEEE, Vol. **72**, No. 8, pp. 996–1009.

[5] K. Drakakis. "A review of Costas arrays". Journal of Applied Mathematics, Volume 2006 (2006).

[6] K. Drakakis. "Data mining and Costas arrays", Turkish Journal of Electrical Engineering & Computer Sciences, Volume 1, No. 4 (2007).

[7] S. W. Golomb. "Algebraic Constructions for Costas Arrays", Journal of Combinatorial Theory, Series A, Volume **37**, pp. 13–21 (1984)

[8] S. W. Golomb, H. Taylor. "Constructions and Properties of Costas Arrays", Proceedings of the IEEE, Volume **72**, No. 9, September 1984

[9] S. Rickard. "Searching for Costas arrays using periodicity properties", IMA International Conference on Mathematics in Signal Processing, The Royal Agricultural College, Cirencester, December 2004.

[10] S. Rickard, E. Connell, F. Duignan, B. Ladendorf, and A. Wade. "The enumeration of Costas arrays of size 26", Conference on Information Sciences and Systems, Princeton, NJ, Mar. 2006.

[11] J. Silverman, V. E. Vickers, J. M. Mooney. "On the number of Costas arrays as a number of array size", Proceedings of the IEEE, Volume **76**, No. 7, July 1988.

Konstantinos Drakakis
Complex & Adaptive Systems Laboratory
University College Dublin
Ireland
Email: Konstantinos.Drakakis@ucd.ie

John Healy
Electronic & Electrical Engineering
University College Dublin
Ireland
Email: johnjhealy@gmail.com

Scott Rickard
Electronic & Electrical Engineering,
University College Dublin,
Ireland.
Email: Scott.Rickard@ucd.ie