Designing unimodular sequence with good auto-correlation properties via Block Majorization-Minimization method

Surya Prakash Sankuru, Prabhu Babu

Abstract

Constant modulus sequence having lower side-lobe levels in its auto-correlation function plays an important role in the applications like SONAR, RADAR and digital communication systems. In this paper, we consider the problem of minimizing the Integrated Sidelobe Level (ISL) metric, to design a complex unimodular sequence of any length. The underlying optimization problem is solved iteratively using the Block Majorization-Minimization (MM) technique, which ensures that the resultant algorithm to be monotonic. We also show a computationally efficient way to implement the algorithm using Fast Fourier Transform (FFT) and Inverse Fast Fourier Transform (IFFT) operations. Numerical experiments were conducted to compare the proposed algorithm with the state-of-the art algorithms and was found that the proposed algorithm performs better in terms of computational complexity and speed of convergence.

Index Terms–Block Majorization-Minimization, Integrated Sidelobe Level, unimodular sequence, aperiodic auto-correlation function, SONAR, RADAR.

I.INTRODUCTION AND PROBLEM FORMULATION

Transmit sequence with an impulse like aperiodic auto-correlation function have many applications, e.g. high resolution SONAR imaging [1], [2], [3], RADAR imaging [4], [5], [6], [7], [2] and CDMA communication systems (to name a few) [6], [8], [2], [9]. Hence, a sequence with lower side-lobe levels in its auto-correlation function is usually desired. In addition to minimizing side-lobe levels, we concentrate on the design of a unimodular sequence due to the practical constraints such as usage of full transmission power available in the system, avoidance of the non-linear side effects and the limitations posed by sequence generation hardware [2], [9], [10].

Let $\{y_i\}_{i=1}^{N}$ be a complex unimodular sequence of length 'N' to be designed. The aperiodic auto-correlation of a sequence $\{y_i\}_{i=1}^{N}$ at any lag 'k' is defined as:

$$r(k) = \sum_{i=1}^{N-k} y_{i+k} y_i^* = r^*(-k), \quad k = 0, ..., N-1. \tag{1}$$

There are two metrics, namely Integrated Side-lobe Level (ISL) and Peak Side-lobe Level (PSL), which are commonly used to measure the degree of correlation of a sequence. The ISL and PSL metrics of a sequence are defined as:

$$\text{ISL} = \sum_{k=1}^{N-1} |r(k)|^2 \tag{2}$$

$$\text{PSL} = \max \left\{ |r(k)| \right\}_{k=1}^{N-1} \tag{3}$$

However, the ISL metric is usually peferred to design a sequence due to its direct applicability to various applications. Hence, our problem of interest would also be

$$\begin{aligned} \underset{\boldsymbol{y}}{\text{minimize}} \quad & \text{ISL} = \sum_{k=1}^{N-1} |r(k)|^2 \\ \text{subject to} \quad & |y_i| = 1, \quad i = 1, ..., N, \end{aligned} \tag{4}$$

where $\boldsymbol{y} = [y_1, y_2, ..., y_N]^T$. The algorithms used to design unimodular sequences can be broadly classified into two categories$-$analytical and computational. Some of the sequences derived using analytical approach are Binary sequences [11], [12], [13], Frank sequence [14], Polyphase sequence [15], Golomb sequence [16]. But these sequences exists only for limited length and has lesser degrees of freedom. On the other hand, computational approaches are able to design a sequence of arbitrary length but at the cost of high computational complexity. Some of the computational approaches available in the literature are CAN algorithm [10], MISL algorithm [17], ADMM approach [18], ISL-NEW algorithm [19].

The following conventions for math symbols are adopted hereafter: boldface uppercase letters denote matrices, boldface lowercase letters denote column vectors and italics denote scalars. Tr() denotes the trace of a matrix. The superscripts $()^T, ()^*, ()^H$ denote transpose, complex conjugate and conjugate transpose, respectively. Re$(.)$ and Im$(.)$ denote real and imaginary parts, respectively. arg$(.)$ denotes the phase of a complex number and y_i denote the i^{th} element of vector $\boldsymbol{y}$. $\boldsymbol{I}_n$ denotes the $n \times n$ identity matrix and vec$(\boldsymbol{G})$ is a column vector consists of all the columns of a matrix-$\boldsymbol{G}$ stacked. Diag$(\boldsymbol{y})$ is a diagonal matrix formed with $\boldsymbol{y}$ as its diagonal. $|.|^2$denotes the absolute squared value. $\mathbb{R}$ and $\mathbb{C}$ represent the real and complex fields. $\lfloor . \rceil$ represents the nearest integer value.

CAN algorithm [10] designs a sequence by minimizing an approximation of the ISL function. The authors in [10] rewrote the objective function in (4) by expressing it in the frequency domain as:

$$\sum_{k=1}^{N-1} |r(k)|^2 = \frac{1}{4N} \sum_{f=1}^{2N} \left[\left| \sum_{i=1}^{N} y_i e^{-j\omega_f (i-1)} \right|^2 - N \right]^2 \tag{5}$$

where $\omega_f = \frac{2\pi}{2N}(f-1),\ f = 1, ..., 2N$ are the Fourier grid frequencies.

Then the problem (4) can be rewritten as:

$$\begin{aligned} \underset{\boldsymbol{y}}{\text{minimize}} \quad & \frac{1}{4N}\sum_{f=1}^{2N}\left[\left|\sum_{i=1}^{N} y_i e^{-j\omega_f(i-1)}\right|^2 - N\right]^2 \\ \text{subject to} \quad & |y_i| = 1, \quad i = 1, ..., N. \end{aligned} \tag{6}$$

The cost function in (6) is a quartic function in the variables $\{y_i\}$ and it is hard to arrive at a minimizer for (6). Thus, instead of solving (6) directly, the authors in [10] solved an almost equivalent problem, which has a quadratic cost function in $\{y_i\}$ as shown below:

$$\begin{aligned} \underset{\boldsymbol{y},\phi_f}{\text{minimize}} \quad & \sum_{f=1}^{2N}\left[\left|\sum_{i=1}^{N} y_i e^{-j\omega_f(i-1)} - \sqrt{N}e^{j\phi_f}\right|^2\right] \\ \text{subject to} \quad & |y_i| = 1, \quad i = 1, ..., N, \end{aligned} \tag{7}$$

where $\phi_f,\ f = 1, 2, ..., 2N$ are auxiliary variables.

The problem in (7) can be rewritten as

$$\begin{aligned} \underset{\boldsymbol{y},\boldsymbol{x}}{\text{minimize}} \quad & \left\|\hat{\boldsymbol{P}}^H\boldsymbol{y} - \sqrt{N}\boldsymbol{x}\right\|_2^2 \\ \text{subject to} \quad & |y_i| = 1, \quad i = 1, ..., N, \end{aligned} \tag{8}$$

where $\hat{\boldsymbol{P}} = [\boldsymbol{p}_1, ..., \boldsymbol{p}_{2N}]$ be a $N \times 2N$ matrix with $\boldsymbol{p}_f \triangleq [1, e^{j\omega_f}, ..., e^{j\omega_f(N-1)}]^T$ and $\boldsymbol{x} \triangleq [e^{j\phi_1}, ..., e^{j\phi_{2N}}]^T$. CAN algorithm solves the problem (8) by alternatively minimizing between $\boldsymbol{y}$ and $\boldsymbol{x}$. For a fixed $\boldsymbol{y}$, minimization of (8) with respect to ϕ_f is given by:

$$\phi_f = \arg(u_f), \quad f = 1, .., 2N, \tag{9}$$

where $\boldsymbol{u} \triangleq \hat{\boldsymbol{P}}^H\boldsymbol{y}$ and for a fixed $\boldsymbol{x}$, minimizer over $\boldsymbol{y}$ would be:

$$y_i = e^{j\arg(g_i)}, \quad i = 1, .., N, \tag{10}$$

where $\boldsymbol{g} \triangleq \hat{\boldsymbol{P}}\boldsymbol{x}$. The pseudocode of the CAN algorithm is summarized in the table Algorithm 1.

Since, CAN algorithm solves an approximation of the problem in (6), the sequence obtained by solving the problem in (7) will not be a minimizer of the original problem in (6). To fix this shortcoming, Song et.al. in [17] proposed the MISL algorithm by solving directly the problem in (6). MISL solves the ISL minimization problem by MM method. Without going into explanation of MISL algorithm, as it would require detailed explaination of MM method, the pseudocode of the MISL algorithm summarized in the table Algorithm 2.

Algorithm 1 :The CAN algorithm proposed in [10]

Require: sequence length ‘N’

1: set $t = 0$, initialize $\boldsymbol{y}^0$

2: **repeat**

3: $\quad \boldsymbol{u} = \hat{\boldsymbol{P}}^H \boldsymbol{y}^t$

4: $\quad x_f = e^{j\arg(u_f)}, f = 1, .., 2N$

5: $\quad \boldsymbol{g} = \hat{\boldsymbol{P}} \boldsymbol{x}$

6: $\quad y_i^{t+1} = e^{j\arg(g_i)}, i = 1, .., N$

7: $\quad t \leftarrow t + 1$

8: **until** convergence

Algorithm 2 :The MISL algorithm proposed in [17]

Require: sequence length ‘N’

1: set $t = 0$, initialize $\boldsymbol{y}^0$

2: **repeat**

3: $\quad \boldsymbol{u} = \hat{\boldsymbol{P}}^H \boldsymbol{y}^t$

4: $\quad u_{\max} = \max_f \left\{ |u_f|^2 : f = 1, .., 2N \right\}$

5: $\quad \boldsymbol{z} = -\hat{\boldsymbol{P}} \left(\mathrm{Diag}\left(|\boldsymbol{u}|^{\boldsymbol{2}} \right) - u_{\max}\boldsymbol{I} - N^2\boldsymbol{I} \right) \boldsymbol{u}$

6: $\quad y_i^{t+1} = e^{j\arg(z_i)}, i = 1, .., N$

7: $\quad t \leftarrow t + 1$

8: **until** convergence

Even though MISL algorithm solves the original problem, it suffers from slower speed of convergence. On the other hand, when compared to CAN algorithm, it converges to the stationary point of problem in (6). Both the algorithms are implemented via FFT and IFFT operations and are computationally viable to be implemented in standard pcs.

In [20], J. Song et.al solve the ISL metric problem to design a sequence set and proposed an algorithm named as MM-Corr, using the MM method. By taking number of sequences as one instead of sequence set, observed that its performance is almost equal to MISL algorithm. In [18], J.Liang et.al proposed a new approach to solve a problem in (8) by using the ADMM method and concludes that, such a technique had a poor performance when compared to MISL algorithm interms of the PSL of an aperiodic auto-correlation function. Y. Li et.al proposed the ISL-NEW algorithm [19] by solving the problem in (6) by using MM method and presented simulation results showing ISL-NEW algorithm as a faster algorithm compared to MISL. J.song et.al had proposed an algorithm based on the MM method named as MM-PSL [21], by solving l_p-norm of the auto-correlation function $(2 < p < \infty)$ as an objective function, which is different from ISL metric.

The main motivation of this paper is to solve the original ISL minimization problem in (4) with a better speed of convergence (with lesser computational complexity) than the existing methods. To achieve this, we have used Block MM technique. We also show a computationally efficient way to implement our algorithm via FFT and IFFT operations.

The major contributions of the paper are as follows:

1) An algorithm based on the Block MM framework is proposed, to design a sequence of any length N by minimizing the ISL metric.
2) We also propose a computationally efficient way to implement our algorithm, which we call as Fast Block MM (FBMM). This is particularly useful for generating sequence of larger lengths.
3) We prove that the proposed algorithm converges to a stationary point of the problem in (4).
4) Numerical experiments were conducted to prove that, our proposed algorithm will perform better when compared to existing methods in terms of speed of convergence.

The rest of the paper is organised as follows. We first give an overview of MM and Block-MM in section II. Next we propose our algorithm and its faster version (FBMM) in section III and discuss its convergence and computational complexity. Numerical experiments are discussed in section IV and finally section V concludes the paper.

II.Majorization-Minimization METHOD

A. MM Procedure:

MM is an iterative procedure, which is used to solve an optimization problem (non-convex or sometimes even a convex) more efficiently. The MM procedure mainly consists of two steps with first step being forming a surroagte function $g(\boldsymbol{y}|\boldsymbol{y}^t)$ which majorizes (upper bounds) the original objective function $f(\boldsymbol{y})$ at any feasible point $\boldsymbol{y} = \boldsymbol{y}^t$, which is followed by minimizing the surrogate function to find the next iterative estimate $\boldsymbol{y}^{t+1}$. The surrogate function $g(\boldsymbol{y}|\boldsymbol{y}^t)$ has to satisfy the following properties:

$$g(\boldsymbol{y}^t|\boldsymbol{y}^t) = f(\boldsymbol{y}^t),\ \forall \boldsymbol{y} \in \chi \tag{11}$$

$$g(\boldsymbol{y}|\boldsymbol{y}^t) \geq f(\boldsymbol{y}),\ \forall \boldsymbol{y} \in \chi \tag{12}$$

where $\boldsymbol{y}^t$ is the value taken by $\boldsymbol{y}$ at t^{th} iteration and χ is a set which consists all possible values of $\boldsymbol{y}$. Hence, the MM procedure will generate the sequence of points $\{\boldsymbol{y}\} = \boldsymbol{y}^0, \boldsymbol{y}^1, \boldsymbol{y}^2, ..., \boldsymbol{y}^m$ according to the following update rule:

$$\boldsymbol{y}^{t+1} \triangleq \arg\min_{\boldsymbol{y} \in \boldsymbol{\chi}} g(\boldsymbol{y}|\boldsymbol{y}^t). \tag{13}$$

The objective value at every iteration will satisfy the following descent property, i.e.

$$f(\boldsymbol{y}^{t+1}) \leq g(\boldsymbol{y}^{t+1}|\boldsymbol{y}^t) \leq g(\boldsymbol{y}^t|\boldsymbol{y}^t) = f(\boldsymbol{y}^t). \tag{14}$$

Computational complexity and the convergence rate of MM based algorithms mainly depends on the choice of the surrogate function $g(\boldsymbol{y}|\boldsymbol{y}^t)$. There are some guideline techniques to construct the surrogate functions as discussed in [22], [23].

B. Block MM:

If one can split an optimization variable into M blocks, then a combination of Block Coordinate Descent [24] and the MM procedure can be applied i.e., the optimization variable is split into blocks and then each block is treated as an independent variable and updated using MM by keeping the other blocks fixed. Hence, the i^{th} block variable is updated by minimizing the surrogate function $g_i(y_i|\boldsymbol{y}^t)$ which majorizes $f(y_i)$ at a feasible point $\boldsymbol{y}^t$ on the i^{th} block. Such surrogate function has to satisfy the following properties:

$$g_i(y_i^t|\boldsymbol{y}^t) = f(\boldsymbol{y}^t), \tag{15}$$

$$g_i(y_i|\boldsymbol{y}^t) \geq f(y_1^t, y_2^t, .., y_i, .., y_N^t), \tag{16}$$

where $\boldsymbol{y}^t$ is the value taken by $\boldsymbol{y}$ at the t^{th} iteration.

The i^{th} block at $(t+1)^{th}$ iteration is updated by solving the following problem:

$$y_i^{t+1} \in \arg\min_{y_i} g_i(y_i|\boldsymbol{y}^t). \tag{17}$$

In Block MM method, every block is updated in a sequential manner and the surrogate function is choosen in a way, such that it is easy to minimize and follow the shape of a objective function.

III.ISL MINIMIZATION USING BLOCK MM TECHNIQUE

In this section, we present our algorithm and discuss its convergence and computational complexity.

A. FBMM algorithm:

Let us revisit the problem in (4)

$$\begin{aligned} \underset{\boldsymbol{y}}{\text{minimize}} \quad & \sum_{k=1}^{N-1} |r(k)|^2 \\ \text{subject to} \quad & |y_i| = 1, \quad i = 1, ..., N. \end{aligned}$$

After substituting for $r(k)$, the above problem can be rewritten as

$$\begin{aligned} & \underset{\boldsymbol{y}}{\text{minimize}} && \left|\sum_{i=1}^{N-1} y_{i+1} y_i^*\right|^2 + ... + \left|\sum_{i=1}^{2} y_{i+N-2} y_i^*\right|^2 + \left|y_N y_1^*\right|^2 \\ & \text{subject to} && |y_i| = 1, \quad i = 1, ..., N. \end{aligned} \tag{18}$$

Now, to solve the problem in (18), we use the Block MM technique by considering $y_1, y_2, .., y_N$ as an independent block variables. For the sake of clarity, in the following we consider a generic optimization problem in variable y_i, and optimization over any variable of "$\boldsymbol{y}$". would be very similar to the generic problem. Let the generic problem be:

$$\begin{aligned} & \underset{y_i}{\text{minimize}} && f_i(y_i) \\ & \text{subject to} && |y_i| = 1. \end{aligned} \tag{19}$$

where y_i indicates the i^{th} block variable and its corresponding objective function $f_i(y_i)$ is defined as

$$f_i(y_i) \triangleq a_i \left[\sum_{k=1}^{l_1} \mid y_i m_{ki}^* + n_{ki} y_i^* + c_{ki} \mid^2\right] + b_i \left[\sum_{k=l_2}^{l_3} \mid n_{ki} y_i^* + c_{ki} \mid^2\right] \tag{20}$$

where a_i, b_i are some fixed multiplicative constants, l_1, l_2, l_3 are the summation limits and m_{ki}, n_{ki}, c_{ki} are the constants associated with k^{th} auto-correlation lag, which are given by

$$\begin{aligned} & m_{ki} \triangleq y_{i-k} \\ & n_{ki} \triangleq y_{i+k} \\ & c_{ki} \triangleq \sum_{q=k+1}^{N} (y_q y_{q-k}^*), \quad q \neq i, q \neq k+i. \end{aligned} \tag{21}$$

The values that the variables a_i, b_i, l_1, l_2, l_3 take will depend on the variable index (y_i). They can be given as follows:

$$\begin{aligned} & a_i \triangleq \begin{cases} 0 & i = 1, N \\ 1 & else \end{cases} \quad , \forall N. \\ & b_i \triangleq \begin{cases} 1 & \forall i \end{cases} , \forall N \in even. \\ & b_i \triangleq \begin{cases} 0 & i = \lfloor N/2 \rfloor + 1 \\ 1 & else \end{cases} \quad , \forall N \in odd. \end{aligned} \tag{22}$$

$$
\begin{aligned}
l_1 &\triangleq \begin{cases} i-1 & i=2,..,\lfloor N/2 \rfloor, a_i \neq 0 \quad ,\forall N. \\ i-1 & b_i=0, a_i \neq 0 \\ N-i & i=\lfloor N/2 \rfloor+1, a_i \neq 0 \quad ,\forall N \in even. \\ N-i & i=\lfloor N/2 \rfloor+2,..,N-1, a_i \neq 0 \quad ,\forall N. \end{cases} \\
l_2 &\triangleq \begin{cases} i & a_i=0 \\ l_1+1 & b_i \neq 0 \quad ,\forall N. \end{cases} \\
l_3 &\triangleq \begin{cases} N-1 & a_i=0 \\ N-i & i=2,..,\lfloor N/2 \rfloor \quad ,\forall N. \\ i-1 & b_i \neq 0 \end{cases}
\end{aligned} \tag{23}
$$

So, from (20), we have

$$
f_i(y_i) = a_i\left[\sum_{k=1}^{l_1} \mid y_i m_{ki}^* + n_{ki} y_i^* + c_{ki} \mid^2\right] + b_i\left[\sum_{k=l_2}^{l_3} \mid n_{ki} y_i^* + c_{ki} \mid^2\right].
$$

which can be rewritten as

$$
f_i(y_i) = a_i\left[\sum_{k=1}^{l_1}\Big| y_i m_{ki}^* + n_{ki} y_i^* + c_{ki} \Big|^2\right] + b_i\left[\sum_{k=l_2}^{l_3}\Big| n_{ki} + c_{ki} y_i \Big|^2\right] \tag{24}
$$

Further simplification yields:

$$
f_i(y_i) = a_i\left[\sum_{k=1}^{l_1}\Big| y_i m_{ki}^* + n_{ki} y_i^* + c_{ki} \Big|^2\right] + b_i\left[\sum_{k=l_2}^{l_3} w_{ki}\Big| y_i + d_{ki} \Big|^2\right] \tag{25}
$$

where

$$
d_{ki} \triangleq \frac{n_{ki}}{c_{ki}},\ w_{ki} \triangleq |c_{ki}|^2.
$$

Expanding the square term in (25) and by ignoring the constant terms, (25) can be rewritten as

$$
\begin{aligned}
f_i(y_i) = &\sum_{k=1}^{l_1} a_i\Big[(n_{ki}^* m_{ki}^*)(y_i^2) + (c_{ki}^* m_{ki}^* + n_{ki}^* c_{ki})(y_i) + (n_{ki} m_{ki})(y_i^2)^* + (m_{ki} c_{ki} + c_{ki}^* n_{ki})(y_i)^*\Big] \\
&+ \sum_{k=l_2}^{l_3} b_i w_{ki}\Big[y_i d_{ki}^* + d_{ki} y_i^*\Big]
\end{aligned} \tag{26}
$$

$$
f_i(y_i) = \sum_{k=1}^{l_1} a_i\left[2\mathrm{Re}\Big((n_{ki}^* m_{ki}^*)(y_i^2)\Big) + 2\mathrm{Re}\Big((c_{ki}^* m_{ki}^* + n_{ki}^* c_{ki})(y_i)\Big)\right] + \sum_{k=l_2}^{l_3}\left[b_i w_{ki} * 2\mathrm{Re}\Big(y_i d_{ki}^*\Big)\right] \tag{27}
$$

Now if we define:

$$\begin{aligned} &n_{ki}^* m_{ki}^* \triangleq \hat{a}_{1ki} + j\hat{a}_{2ki} \\ &(c_{ki}^* m_{ki}^* + n_{ki}^* c_{ki}) \triangleq \hat{b}_{1ki} + j\hat{b}_{2ki} \\ &d_{ki}^* \triangleq \hat{c}_{1ki} + j\hat{c}_{2ki} \\ &y_i \triangleq u_1 + ju_2 \end{aligned} \tag{28}$$

where $\hat{a}_{1ki}$, $\hat{a}_{2ki}$, $\hat{b}_{1ki}$, $\hat{b}_{2ki}$, $\hat{c}_{1ki}$, $\hat{c}_{2ki}$, u_1, u_2 are real valued quantities.

Then $f_i(y_i)$ in (27) can be further simplified as:

$$\begin{aligned} f_i(u_1,u_2) = &\left[2a_i \sum_{k=1}^{l_1} \hat{a}_{1ki}\right](u_1)^2 - \left[4a_i \sum_{k=1}^{l_1} \hat{a}_{2ki}\right] u_1 u_2 + \left[2a_i \sum_{k=1}^{l_1} \hat{b}_{1ki} + 2b_i \sum_{k=l_2}^{l_3} w_{ki}\hat{c}_{1ki}\right] u_1 \\ &- \left[\sum_{k=1}^{l_1} 2a_i \hat{b}_{2ki} + \sum_{k=l_2}^{l_3} 2b_i w_{ki} \hat{c}_{2ki}\right] u_2 - \left[2a_i \sum_{k=1}^{l_1} \hat{a}_{1ki}\right](u_2)^2 \end{aligned} \tag{29}$$

Again introducing,

$$\begin{aligned} &a \triangleq 2a_i \sum_{k=1}^{l_1} \hat{a}_{1ki} \\ &b \triangleq 4a_i \sum_{k=1}^{l_1} (\hat{a}_{2ki}) \\ &c \triangleq 2a_i \sum_{k=1}^{l_1} \hat{b}_{1ki} + 2b_i \sum_{k=l_2}^{l_3} w_{ki}\hat{c}_{1ki} \\ &d \triangleq 2a_i \sum_{k=1}^{l_1} \hat{b}_{2ki} + 2b_i \sum_{k=l_2}^{l_3} w_{ki}\hat{c}_{2ki} \end{aligned} \tag{30}$$

Then $f_i(u_1, u_2)$ in (29) is simplified as:

$$f_i(u_1,u_2) = au_1^2 - bu_1u_2 + cu_1 - du_2 - au_2^2 \tag{31}$$

Thus the problem in (19) has become the following problem with real valued variables.

$$\begin{aligned} &\underset{u_1,u_2}{\text{minimize}} \quad f_i(u_1,u_2) \\ &\text{subject to} \quad u_1^2 + u_2^2 = 1. \end{aligned} \tag{32}$$

Now, the problem in (32) can be written in (matrix-vector) form as:

$$\begin{aligned} &\underset{\boldsymbol{v}}{\text{minimize}} \quad \boldsymbol{v}^T\boldsymbol{A}\boldsymbol{v} + \boldsymbol{e}^T\boldsymbol{v} \\ &\text{subject to} \quad \boldsymbol{v}^T\boldsymbol{v} = 1, \end{aligned} \tag{33}$$

with

$$\begin{aligned} \boldsymbol{A} &\triangleq \begin{bmatrix} a & \frac{-b}{2} \\ \frac{-b}{2} & -a \end{bmatrix} \\ \boldsymbol{e} &\triangleq \begin{bmatrix} c \\ -d \end{bmatrix} \\ \boldsymbol{v} &\triangleq \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \end{aligned} \tag{34}$$

The problem in (33) has an objective function which is a non-convex quadratic function in the variable $\boldsymbol{v}$ because of $(-a)$ in the diagonal of $\boldsymbol{A}$ and also the constraint is a quadratic equality constraint, so the problem in (33) is a non convex problem and hard to solve. So, we decided to employ MM technique to solve the problem in (33). Let us introduce the following lemma, which would be useful to develop our algorithm.

Lemma-1: Let $\boldsymbol{Q}$ be an $n \times n$ Hermitian matrix and $\boldsymbol{R}$ be another $n \times n$ Hermitian matrix such that $\boldsymbol{R} \geq \boldsymbol{Q}$. Then for any point $\boldsymbol{y}^0 \in \boldsymbol{C}^n$, the quadratic function $\boldsymbol{y}^H\boldsymbol{Q}\boldsymbol{y}$ is majorized by $\boldsymbol{y}^H\boldsymbol{R}\boldsymbol{y} + 2Re(\boldsymbol{y}^H(\boldsymbol{Q}-\boldsymbol{R})\boldsymbol{y}^0) + (\boldsymbol{y}^0)^H(\boldsymbol{R}-\boldsymbol{Q})\boldsymbol{y}^0$ at $\boldsymbol{y}^0$.

Proof: Although the proof can be find in [17], we replicate it here for the sake of clarity.

As $\boldsymbol{R} \geq \boldsymbol{Q}$, we have

$$\begin{aligned} \boldsymbol{y}^H\boldsymbol{Q}\boldsymbol{y} &= (\boldsymbol{y}^0)^H\boldsymbol{Q}\boldsymbol{y}^0 + 2Re((\boldsymbol{y}-\boldsymbol{y}^0)^H\boldsymbol{Q}\boldsymbol{y}^0) + (\boldsymbol{y}-\boldsymbol{y}^0)^H\boldsymbol{Q}(\boldsymbol{y}-\boldsymbol{y}^0) \\ &\leq (\boldsymbol{y}^0)^H\boldsymbol{Q}\boldsymbol{y}^0 + 2Re((\boldsymbol{y}-\boldsymbol{y}^0)^H\boldsymbol{Q}\boldsymbol{y}^0) + (\boldsymbol{y}-\boldsymbol{y}^0)^H\boldsymbol{R}(\boldsymbol{y}-\boldsymbol{y}^0) \\ &= \boldsymbol{y}^H\boldsymbol{R}\boldsymbol{y} + 2Re(\boldsymbol{y}^H(\boldsymbol{Q}-\boldsymbol{R})\boldsymbol{y}^0) + (\boldsymbol{y}^0)^H(\boldsymbol{R}-\boldsymbol{Q})\boldsymbol{y}^0 \end{aligned}$$

for any $\boldsymbol{y} \in \boldsymbol{C}^n$. ■

Now, by using Lemma-1, we will majorize only the quadratic term in the objective function of problem in (33) at any feasible point $\boldsymbol{v} = \boldsymbol{v}^t$ and get

$$g_i(\boldsymbol{v}|\boldsymbol{v}^t) = \boldsymbol{v}^T\boldsymbol{A_1}\boldsymbol{v} + 2\text{Re}[\boldsymbol{v}^T(\boldsymbol{A}-\boldsymbol{A_1})\boldsymbol{v}^t] + (\boldsymbol{v}^t)^T(\boldsymbol{A_1}-\boldsymbol{A})\boldsymbol{v}^t, \tag{35}$$

where $\boldsymbol{A_1} = \lambda_{\text{max}}(\boldsymbol{A}).\boldsymbol{I}_n$. Since $\lambda_{\text{max}}(\boldsymbol{A})$ is a constant value and $\boldsymbol{v}^T\boldsymbol{v} = 1$, so the first and the last terms in the above surrogate function are constants. Hence, after ignoring the constant terms from (35) we get,

$$g_i(\boldsymbol{v}|\boldsymbol{v}^t) = 2\text{Re}[\boldsymbol{v}^T(\boldsymbol{A}-\boldsymbol{A_1})\boldsymbol{v}^t] \tag{36}$$

Now, the problem (33) is equal to

$$\begin{aligned} \underset{\boldsymbol{v}}{\text{minimize}} \quad & 2\text{Re}[\boldsymbol{v}^T(\boldsymbol{A}-\boldsymbol{A_1})\boldsymbol{v}^t]+\boldsymbol{e}^T\boldsymbol{v} \\ \text{subject to} \quad & \boldsymbol{v}^T\boldsymbol{v}=1 \end{aligned} \tag{37}$$

which can be further rewritten as

$$\begin{aligned} \underset{\boldsymbol{v}}{\text{minimize}} \quad & g_i(\boldsymbol{v}|\boldsymbol{v}^t)=\| \boldsymbol{v}-\boldsymbol{z} \|_2^2 \\ \text{subject to} \quad & \boldsymbol{v}^T\boldsymbol{v}=1 \end{aligned} \tag{38}$$

where $\boldsymbol{z}=-[(\boldsymbol{A}-\boldsymbol{A_1})\boldsymbol{v}^t+(\boldsymbol{e}/2)]$.

Now, the problem in (38) has a closed form solution of:

$$\boldsymbol{v}=\frac{\boldsymbol{z}}{||\boldsymbol{z}||_2}. \tag{39}$$

Then the update y_i^{t+1} can be calculated by:

$$y_i^{t+1}=v_1+jv_2 \tag{40}$$

The constants $(c_{1i},c_{2i},...,c_{(N-1)i})$ in (21) which are calculated at every iteration, which form the bulk of the computations, can be computed via FFT and IFFT operations as follows: For example, the constant (c_{1i}) can be intreperated as the auto-correlation of a sequence (with $y_i=0$) which inturn can be calculated by an FFT and IFFT operation. So, to calculate all the constants of N variables, we would require N number of FFT and N number of IFFT operations. To avoid implementing FFT and IFFT operations N number of times, we propose an computationally efficient way to calculate the constants. To achieve this, we would exploit the cyclic pattern in the expression of the constants. First we define $\boldsymbol{s}$ which includes original variable $\boldsymbol{y}$ along with some pre-defined zero padding structure as shown below:

$$\boldsymbol{s}=[\boldsymbol{0}_{1\times N-2},\boldsymbol{y}^T,\boldsymbol{0}_{1\times N}]^T \tag{41}$$

Let us define the variables $\boldsymbol{b}_i$ and $\boldsymbol{D}_i$ as:

$$\boldsymbol{b}_i=[-y_i^*,y_{i-1}^*,-y_i,y_{i-1}]^T \tag{42}$$

$$\boldsymbol{D}_i = \begin{bmatrix} \boldsymbol{s}_{(N+i-1)} & . & . & \boldsymbol{s}_{(2N+i-4)} & 0 \\ 0 & \boldsymbol{s}_{(N+i-1)} & . & . & \boldsymbol{s}_{(2N+i-4)} \\ 0 & \boldsymbol{s}^*_{(N+i-4)} & . & . & \boldsymbol{s}^*_{(i-1)} \\ \boldsymbol{s}^*_{(N+i-4)} & . & . & \boldsymbol{s}^*_{(i-1)} & 0 \end{bmatrix} \tag{43}$$

So, to calculate the i^{th} variable constants $(c_{1i}, c_{2i}, .., c_{(N-1)i})$, we will use the constants associated with the $(i-1)^{th}$ variable $(c_{1(i-1)}, c_{2(i-1)}, .., c_{(N-1)(i-1)})$ as follows:

$$\begin{bmatrix} c_{1i}, & . & . & , c_{(N-1)i} \end{bmatrix} = \begin{bmatrix} c_{1(i-1)}, . & . & . & , c_{(N-1)(i-1)} \end{bmatrix} + \boldsymbol{b}_i^T \boldsymbol{D}_i \qquad \forall\, i = 2, .., N \tag{44}$$

Therefore, all the $(N-1)$ number of constants associated with each of the N variables are implemented using only one FFT and IFFT operation. The steps of our algorithm which is named as FBMM is shown in the table Algorithm 3.

Algorithm 3 :FBMM algorithm

Require: sequence length 'N'
set $t = 0$, initialize $\boldsymbol{y}^0$
repeat
    set $i = 1$
        repeat
            calculate $\{c_{ki}\}_{k=1}^{N-1}$ using (44)
            calculate $d_{ki} = \frac{n_{ki}}{c_{ki}}$, $w_{ki} = |c_{ki}|^2$, $k = 1, ..., N-1$.
            $\boldsymbol{A}_1$=$\lambda_{\max}(\boldsymbol{A}).\boldsymbol{I}_2$
            $\boldsymbol{z} = -[(\boldsymbol{A} - \boldsymbol{A_1})\boldsymbol{v}^t + (\boldsymbol{e}/2)]$
            $\boldsymbol{v} = \frac{\boldsymbol{z}}{||\boldsymbol{z}||_2}$
            $y_i^{t+1} = v_1 + jv_2$
            $i \longleftarrow i + 1$
        until length of a sequence
    $t \longleftarrow t + 1$
until convergence

B. Proof of convergence:

The proposed algorithm is based on a Block MM technique. As Block MM is a combination of coordinate descent and the MM procedure, it is ensured that the cost function evaluated at every limit point is monotonic. Also, since the cost function in (4) is bounded below by zero, the sequence of objective values is guaranteed to converge to a finite value. In [25], Theorem 2.a Razaviyayn et.al stated that a limit point generated at each iteration by a Block MM algorithm is a coordinate wise minimum point with respect to original cost function, iff the upper bound $g_i(.)$ is a quasi-convex function. We now have to prove that $g_i(\boldsymbol{v}|\boldsymbol{v}^t)$ in (35) is indeed a quasi convex function.

So, from (35), we have,

$$g_i(\boldsymbol{v}|\boldsymbol{v}^t) = \boldsymbol{v}^T \boldsymbol{A_1} \boldsymbol{v} + 2\text{Re}[\boldsymbol{v}^T(\boldsymbol{A} - \boldsymbol{A_1})\boldsymbol{v}^t] + (\boldsymbol{v}^t)^T(\boldsymbol{A_1} - \boldsymbol{A})\boldsymbol{v}^t$$

which is a quadratic function in $\boldsymbol{v}$. The Hessian of $g_i(\boldsymbol{v}|\boldsymbol{v}^t)$ is $2\boldsymbol{A}_1$, where $\boldsymbol{A}_1 = \lambda_{\max}(\boldsymbol{A}).\boldsymbol{I}_n$. Since $\lambda_{\max}(\boldsymbol{A})$ is a positive value, $\boldsymbol{A}_1$ is a diagonal matrix with positive entries. Hence $g_i(\boldsymbol{v}|\boldsymbol{v}^t)$ is a convex function. Since every convex function is also a quasi-convex function, $g_i(\boldsymbol{v}|\boldsymbol{v}^t)$ is also a quasi-convex function. Therefore, according to Theorem 2.a of [25] the sequence of points generated by FBMM will converge to the stationary point of problem in (4).

C. Computational complexity:

The per iteration computational complexity of the proposed algorithm is dominated in the calculation of constants c_{ki} ,$k = 1, .., N - 1$,$i = 1, .., N$. These constants can be calculated using one FFT and IFFT operation and the approach as mentioned in the end of subsection (A), where we exploit some cyclic pattern and calculate the constants, then the computational complexity per iteration would be $\mathcal{O}(N^2) + \mathcal{O}(N\, logN)$.

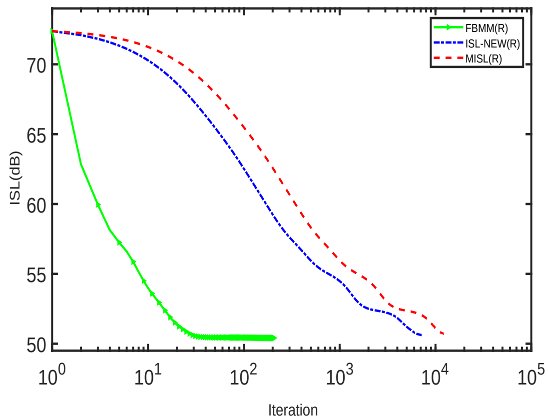
IV.NUMERICAL EXPERIMENTS

In this section, we present the numerical results of our proposed algorithm and compare its performance with the state-of-the art algorithms. As CAN algorithm and ADMM method developed in [18] solves an approximate problem, we will not include them for numerical comparision. So, we compared our results with MISL and ISL-NEW algorithm. All the simulations were performed in MATLAB on a PC with two core 2.40GHz processor. Experiments has been conducted to design a sequence of lengths $N = 50, 100, 200, 300, 400, 500$ using different initialization sequences like Random, Golomb [16] and Frank [14] sequences. In case of random initialization, 30 monte carlo runs has been conducted for every length and for every run, initialization sequence $\{y_i^0\}_{i=1}^N$ is choosen as $\{e^{j2\pi\theta_i}\}_{i=1}^N$, where $\{\theta_i\}$ are drawn randomly from the uniform distribution $[0, 1]$. The convergence criterion which we used to stop all the algorithms in the comparision is

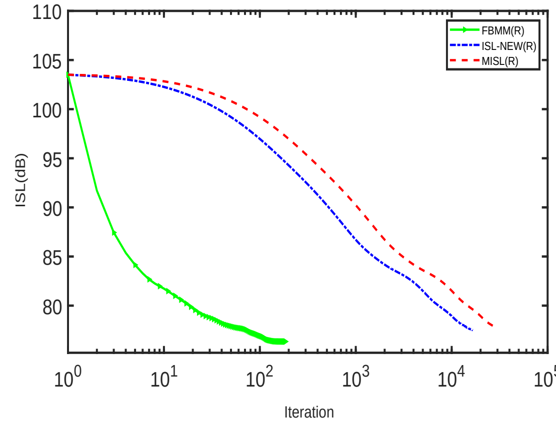
$$\left| \frac{(\text{ISL}(t+1) - \text{ISL}(t))}{\max(1, \text{ISL}(t))} \right| \leq 10^{-5}, \tag{45}$$

where $\text{ISL}(t)$ is the ISL metric value at t^{th} iteration.

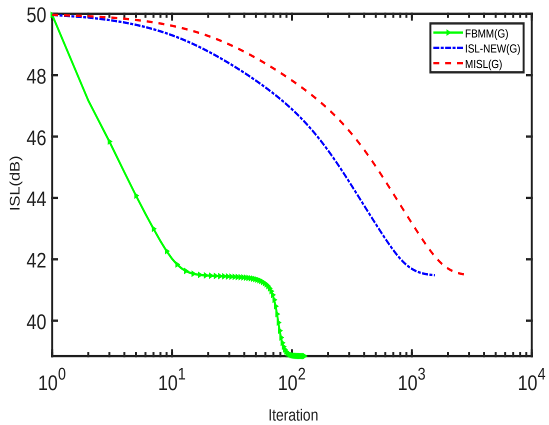
In each experiment, execution time of the proposed algorithm and property of a designed sequence such as auto-correlation side-lobe levels, cost function value were observed and compared with the MISL and ISL-NEW algorithms. Since the algorithms under comparision, MISL and ISL-NEW and also our algorithm are based on MM, all of them can be accelerated using standard acceleration schemes [26], [27], [28], but for the sake of comparison, we didn't implement acceleration scheme for any of the methods.



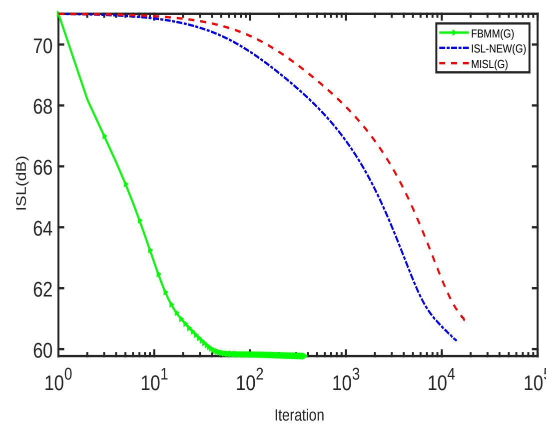
(a) $N = 100$



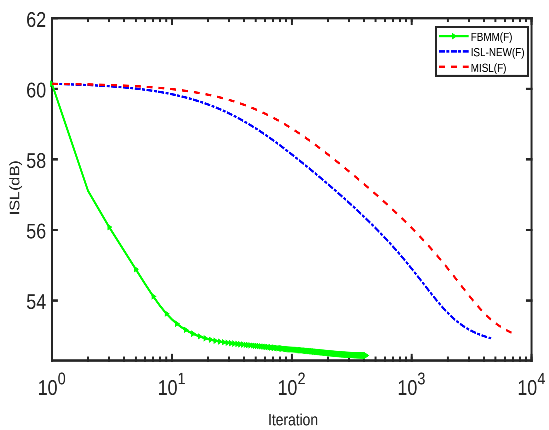
(b) $N = 500$



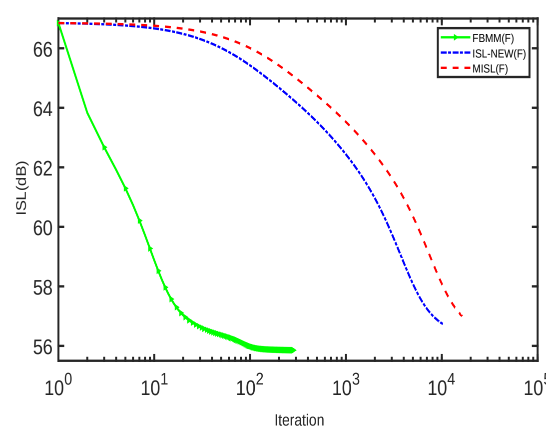
(c) $N = 100$



(d) $N = 500$



(e) $N = 289$



(f) $N = 484$

Figure 1: ISL vs Iteration for a sequence length $N = 100, 289, 484, 500$. (a) and (b) are for initialization via Random sequence. (c) and (d) are for initialization via Golomb sequence. (e) and (f) are for initialization via Frank sequence.

Figure. 1 consists the plots of ISL value vs iteration number for different lengths using three different initialization sequences. Here FBMM(R), FBMM(G), FBMM(F) indicates FBMM algorithm initialized with Random, Golomb and Frank sequences, respectively. We initialize all the algorithms at the same initial point and observed that, almost they also ended up at the same minimum but with different speed of convergence. From the plots, it can be observed that for the different lengths N, MISL and ISL-NEW are taking much larger number of iterations and FBMM is taking lesser number of iterations to converge to the same objective minimum value.

Figure. 2 shows the ISL value at every iteration vs time for different lengths using different initialization sequences. Each plot also has zoomed version to show the subtle difference in the speed of convergence of FBMM and ISL-NEW. From the plots, it can be observed that, irrespective to the initialization sequence and length N, FBMM is always taking less time to converge to the same objective minimum value when compared to MISL and ISL-NEW algorithms.

Figure. 3 shows the auto-correlation plots of the generated sequence via FBMM, ISL-NEW and MISL algorithms using different initialization sequences. From plots, we observed that in the case of Frank and Golomb sequence initializations, most of the side-lobe levels in the initial sequence itself are too low but PSL is high. It can be seen that all the three algorithms improves performance from the initialized sequence interms of PSL.

Figure. 4 has a comparision of three algorithms interms of average running time for different lengths using random initialization sequence. For better comparision, all the three algorithms are initialized with a same sequence and stopped using the same convergence criterion. From the figure, it can be observed that, irrespective of length N, FBMM is taking lesser time to converge when compared to rest of the two algorithms: MISL and ISL-NEW.

VI.CONCLUSION

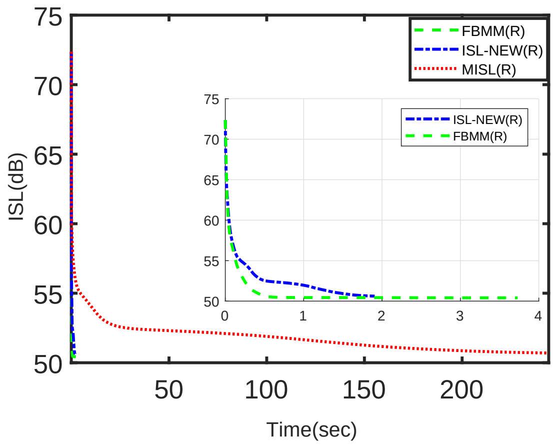
To design a sequence of any length (N), we have proposed an algorithm by minimizing the ISL metric. We also shown a computationally efficient way of implementing our proposed algorithm and named as FBMM. Proposed algorithm is derived based on a Block MM technique and implemented using FFT, IFFT operations, hence computationally efficient for large lengths. Numerical experiments shows that, proposed algorithm is performing well when compared to state-of-the art algorithms in terms of convergence rate, computational complexity and average running time.

REFERENCES

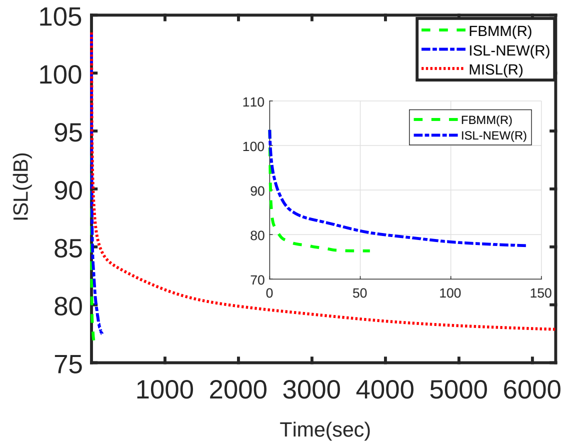
[1] W. C. Knight, R. G. Pridham, and S. M. Kay, "Digital signal processing for sonar," *Proceedings of the IEEE*, vol. 69, no. 11, pp. 1451–1506, Nov 1981.

[2] W. Roberts, H. He, J. Li, and P. Stoica, "Probing waveform synthesis and receiver filter design," *IEEE Signal Processing Magazine*, vol. 27, no. 4, pp. 99–112, July 2010.

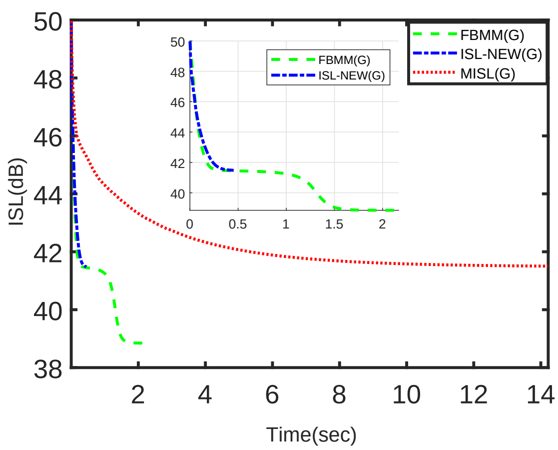
[3] Zhaofu Chen, J. Li, X. Tan, H. He, Bin Guo, P. Stoica, and M. Datum, "On probing waveforms and adaptive receivers for active sonar," in *OCEANS 2010 MTS/IEEE SEATTLE*, Sep. 2010, pp. 1–10.



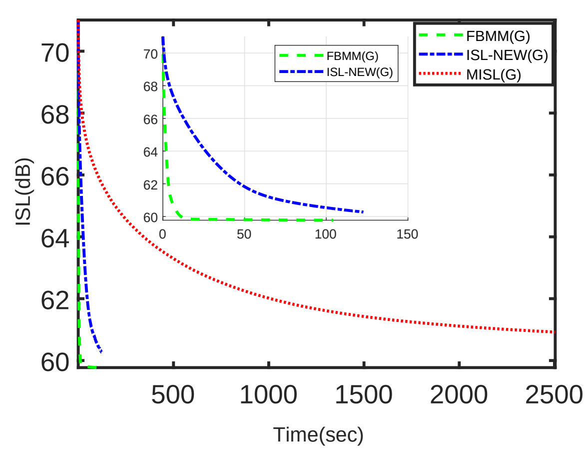
(a) $N = 100$



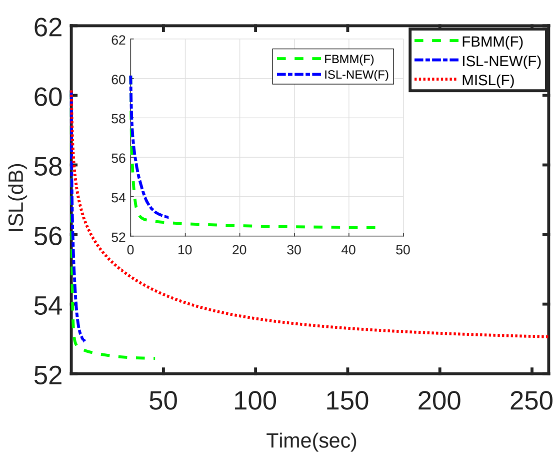
(b) $N = 500$



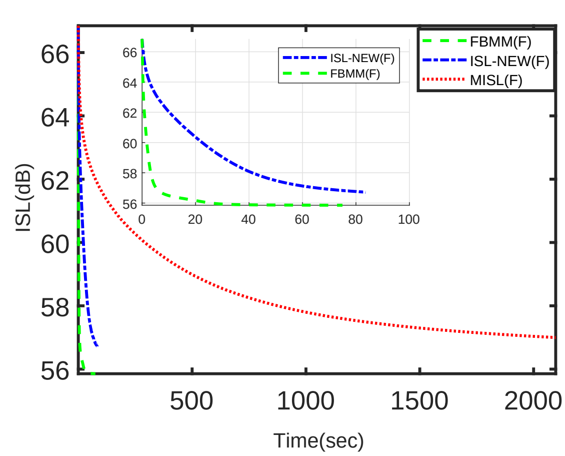
(c) $N = 100$



(d) $N = 500$

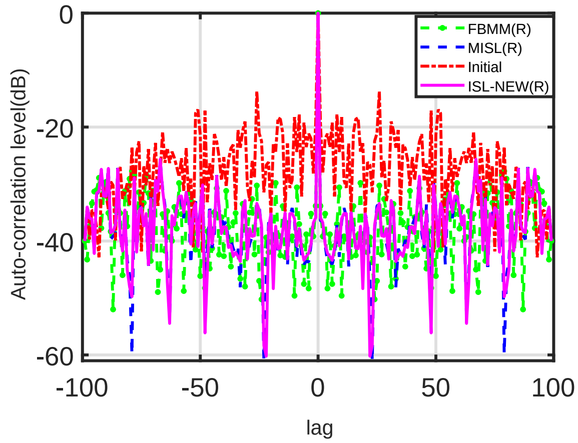


(e) $N = 289$

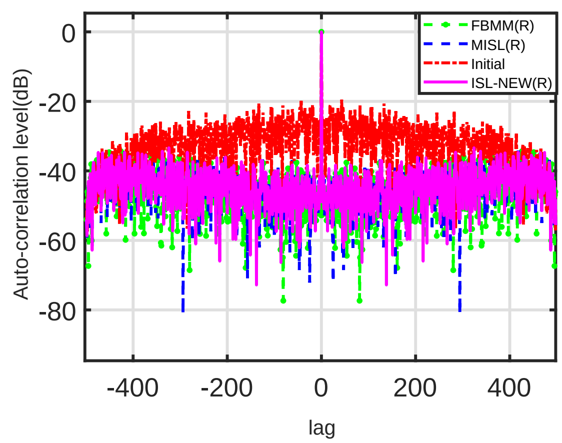


(f) $N = 484$

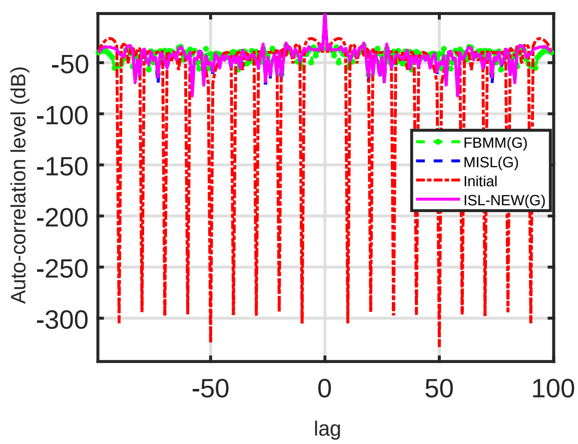
Figure 2: ISL vs Time for a sequence length $N = 100, 289, 484, 500$. (a) and (b) are for initialization via Random sequence. (c) and (d) are for initialization via Golomb sequence. (e) and (f) are for initialization via Frank sequence.



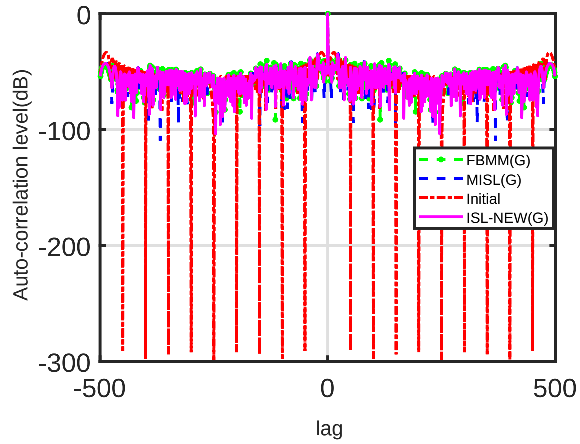
(a) $N = 100$



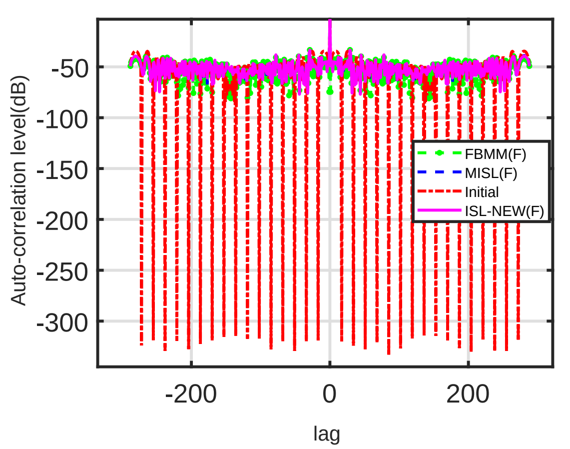
(b) $N = 500$



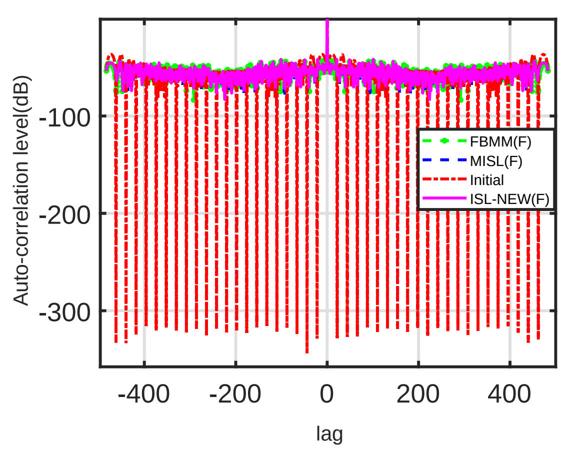
(c) $N = 100$



(d) $N = 500$



(e) $N = 289$



(f) $N = 484$

Figure 3: Autocorrelation value vs lag for a sequence length $N = 100, 289, 484, 500$. (a) and (b) are for initialization via Random sequence. (c) and (d) are for initialization via Golomb sequence. (e) and (f) are for initialization via Frank sequence.

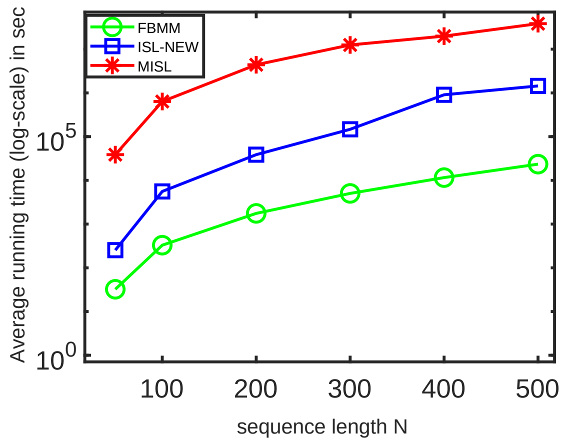


Figure 4: Avg-Time vs sequence length using Random initialization sequence.

[4] M. Skolnik, "Radar handbook," *McGraw-Hill*, 1990.

[5] N. Levanon and E. Mozeson, "Basic radar signals," *John Wiley and Sons*, vol. 64, no. 11, pp. 53–73, 2004.

[6] S. W. Golomb and G. Gong, *Signal Design for Good Correlation: For Wireless Communication, Cryptography, and Radar*. Cambridge University Press, 2005.

[7] J. J. Benedetto, I. Konstantinidis, and M. Rangaswamy, "Phase-coded waveforms and their design," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 22–31, Jan 2009.

[8] H. B. Mann, *Error correcting codes; proceedings of a symposium. Edited by Henry B. Mann*. Wiley New York, 1968.

[9] H. He, J. Li, and P. Stoica, *Wave form Design for Active Sensing Systems: A Computational Approach*. Cambridge University Press, 2012. [Online]. Available: https://books.google.co.in/books?id=syqYnQAACAAJ

[10] P. Stoica, H. He, and J. Li, "New algorithms for designing unimodular sequences with good correlation properties," *IEEE Transactions on Signal Processing*, vol. 57, no. 4, pp. 1415–1425, April 2009.

[11] I. Dotú and P. Van Hentenryck, "A note on low autocorrelation binary sequences," in *Principles and Practice of Constraint Programming - CP 2006*, F. Benhamou, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 685–689.

[12] S. Mertens, "Exhaustive search for low-autocorrelation binary sequences," *Journal of Physics A: Mathematical and General*, vol. 29, no. 18, pp. 473–481, sep 1996.

[13] S. E. Kocabas and A. Atalar, "Binary sequences with low aperiodic autocorrelation for synchronization purposes," *IEEE Communications Letters*, vol. 7, no. 1, pp. 36–38, Jan 2003.

[14] R. Frank, "Polyphase codes with good nonperiodic correlation properties," *IEEE Transactions on Information Theory*, vol. 9, no. 1, pp. 43–45, January 1963.

[15] P. Borwein and R. Ferguson, "Polyphase sequences with low autocorrelation," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1564–1567, April 2005.

[16] N. Zhang and S. W. Golomb, "Polyphase sequence with low autocorrelations," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 1085–1089, May 1993.

[17] J. Song, P. Babu, and D. P. Palomar, "Optimization methods for designing sequences with low autocorrelation sidelobes," *IEEE Transactions on Signal Processing*, vol. 63, no. 15, pp. 3998–4009, Aug 2015.

[18] J. Liang, H. C. So, J. Li, and A. Farina, "Unimodular sequence design based on alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5367–5381, Oct 2016.

[19] Y. Li and S. A. Vorobyov, "Fast algorithms for designing unimodular waveform(s) with good correlation properties," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1197–1212, March 2018.

[20] J. Song, P. Babu, and D. P. Palomar, "Sequence set design with good correlation properties via majorization-minimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 11, pp. 2866–2879, June 2016.

[21] J. Song, P. Babu, and D. P. Palomar, "Sequence design to minimize the weighted integrated and peak sidelobe levels," *IEEE Transactions on Signal Processing*, vol. 64, no. 8, pp. 2051–2064, April 2016.

[22] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 3, pp. 794–816, Feb 2017.

[23] D. R. Hunter and K. Lange, "A tutorial on mm algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004. [Online]. Available: https://doi.org/10.1198/0003130042836

[24] A. Breloy, Y. Sun, P. Babu, and D. P. Palomar, "Block majorization-minimization algorithms for low-rank clutter subspace estimation," in *2016 24th European Signal Processing Conference (EUSIPCO)*, Aug 2016, pp. 2186–2190.

[25] M. Razaviyayn, M. Hong, and Z. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 1126–1153, 8 2013.

[26] R. VARADHAN and C. ROLAND, "Simple and globally convergent methods for accelerating the convergence of any em algorithm," *Scandinavian Journal of Statistics*, vol. 35, no. 2, pp. 335–353, 2008. [Online]. Available: http://www.jstor.org/stable/41548597

[27] M. Raydan and B. F. Svaiter, "Relaxed steepest descent and cauchy-barzilai-borwein method," *Computational Optimization and Applications*, vol. 21, no. 2, pp. 155–167, Feb 2002. [Online]. Available: https://doi.org/10.1023/A:1013708715892

[28] J. BARZILAI and J. M. BORWEIN, "Two-Point Step Size Gradient Methods," *IMA Journal of Numerical Analysis*, vol. 8, no. 1, pp. 141–148, 01 1988. [Online]. Available: https://doi.org/10.1093/imanum/8.1.141