adaptive antenna arrays can dynamically maximize intended signal reception (main lobe) and/or minimize interference or null simultaneously.

Precisely, Banerjee & Dwivedi (2013) stated that adaptive antenna arrays allow the beam to be continually steered to any direction to allow for the maximum signal to be received and/or minimum of nulls. In this case, we can use adaptive beamforming techniques to obtain the desired antenna radiation pattern by adjusting the antenna parameters such as position, excitation current amplitude and excitation current phase weights of the antenna array elements. Radiation pattern nulling optimization techniques are very important to suppress undesired interfering signals while preserving the desired main beam intensity.

Overall, both switched−beam and adaptive antennas are able to provide the directivity. Yet, for antenna designers and engineers, decisions need to be made against cost, complexity and the optimal performance requirements to decide which type should be used to cater the vastly demanding bandwidth, QoS and coverage of wireless communication systems.

Smart antennas design has attracted a widespread interest for several decades due to their implementations in numerous applications (Godara, 2004) and their capabilities to increase system performance. This becomes beneficial in urban and densely populated area where smart antennas can dynamically tuning out interference while focusing on the intended user signal via remarkable DSP advanced techniques (Balanis & Ioannides, 2007). The first issue of *IEEE Transactions of Antennas and Propagation* published in 1964, was followed by special issues of various journals, books, a selected bibliography, and a vast number of specialized research papers (Godara, 2004). Some of the selected papers examples related to smart antenna design include adaptive antenna systems (Widrow et al., 1967), adaptive arrays (Applebaum,

1976), and recently, smart antenna for cellular mobile communications (Stevanović, Skrivervik & Mosig, 2003 and Jain, Katiyar & Agrawal, 2011).

Selected control algorithms with predefined criteria provide adaptive antenna arrays with a unique capability to alter radiation pattern characteristics, e.g. interferers, side lobe level (SLL), main beam direction, and beamwidth. These control algorithms originate from several disciplines and aim for various applications, e.g. underwater, aerospace, and more recently cellular mobile communications. The commercialization of smart antennas has increased the wireless communications system performance in terms of capacity, signal quality or spectral efficiency, and coverage (Balanis & Ioannides, 2007).

1.2 Research Motivation

Antenna array pattern synthesis primarily deals in finding an appropriate complex weighting vector or matrix to produce the anticipated radiation pattern characteristics. The characteristics include the desired SLL, the beamwidth of antenna pattern, and the specific position of the prescribed nulls. Generally, there are three categories of antenna array pattern synthesis (Banerjee & Dwivedi, 2013):

i. The antenna pattern synthesis to generate narrow beams and low SLL, which guarantees the radiating or receiving energy to be more focused in specific directions using various computational techniques, e.g. the binomial method, Dolph−Chebyshev method, and Taylor line source method.

ii. The antenna pattern synthesis to mitigate nulls in desired directions in where smart antenna reduces interferences using effective mathematical techniques, e.g. Schelkunoff polynomial method.

iii. The antenna pattern synthesis to exhibit a desired distribution in the entire visible region referred as beam shaping where the antenna array can obtain a wider angular coverage using few scientific techniques, e.g. the Fourier transform technique and Woodward–Lawson sampling method.

Since mid–1960s, many researchers and engineers have studied on antenna array optimization techniques. The main purpose is to generate the desired radiation pattern specifications by properly initializing the element length, element spacing, feed current amplitude and feed current phase. The specifications include directivity, maximum antenna gain in the desired direction, and minimum antenna gain (e.g. nulls) in the undesired direction. The proposed and verified optimization techniques are ranging from "analytical approach", e.g. Dolph−Chebyshev, Taylor line source, and Orchard−Elliott methods (Balanis, 2005) to "numerical technique", e.g. Fletcher−Powell method, Legendre functions, and conjugate gradient method (CGM).

Recently, due to the higher versatility, flexibility and capability than existing analytical and numerical techniques in complex multidimensional optimization, modern evolutionary algorithm (EA) or evolutionary computation (EC) methods including genetic algorithm (GA), simulated annealing (SA), and particle swarm optimization (PSO) are applied for antenna array beam design to determine the physical layout of the array that produces the radiation pattern closest to the desired pattern.

In this case, EA/EC techniques provide better results relatively than the original gradient and conventional numerical methods. Besides, EA/EC techniques capable to deal with large number of optimization parameters, avoiding getting stuck in local minima, and relatively easy to simulate on computers (Khodier et. al, 2009). For an example, GA is more accurate than conventional analytical method in null locating and maintaining the required null depths (Sattari & Hejazi, 2008). Hence, GA becomes as a

robust random search method, which can solve complicated, multidimensional, and nonlinear problems.

In this research, cuckoo search (CS) metaheuristic algorithm is extensively applied, enhanced, and hybridized to become as an alternative modern EA/EC method in linear antenna array synthesis. There are three reasons of using the newly evolved CS algorithm:

i. CS algorithm is a simple stochastic searching method, which has fewer internal parameters to be fine-tuned (Yang & Deb, 2009).
ii. CS algorithm is more generic and robust than the PSO and GA in optimizing multimodal objective functions model (Yang & Deb, 2009).
iii. CS algorithm is still at the infancy stage and has a lot of potentials to be enhanced and hybridized specifically for electromagnetic single objective (SO) and multiobjective (MO) optimization problem, e.g. antenna array geometry synthesis.

1.3 Problem Statement

Precisely, despite producing successful side lobes suppression and/or nulls mitigation for antenna array synthesis, there are some limitations occur in some well−known conventional techniques. Firstly, let us analyze the amplitude tapering technique used in different window functions such as Kaiser or Dolph−Chebyshev (Balanis, 2005). In tapering process, the main task is to calculate appropriate weights vector, which execute the narrow beam with minimum SLL. Even so, a drawback of amplitude tapering is beamwidth expanding. It means that wider beamwidth occurs spontaneously whilst achieving smaller SLL (Banerjee & Dwivedi, 2013). Moreover, the Dolph−Chebyshev array has inefficiency by suffering directivity saturation when the

number of radiating elements becomes large. Furthermore, the synthesis for the Dolph−Chebyshev array is also complex because we need new complicated polynomial series for every radiation pattern (Alexopoulos, 2006).

Secondly, analytical optimization methods have a drawback where we can use the methods for a specific radiation pattern synthesis, which subjects to one restriction only. Ideally, there should be many possibilities to find the fittest solution. However, this becomes impossible since exhaustive checking of all possible phase−amplitude excitation is impractical as these methods search for a single point or parameter via deterministic rules (Banerjee & Dwivedi, 2013).

Thirdly, the trade−off between the SLL and the half−power beam width (HPBW) stimulus is performed through obtaining the narrowest possible beam width for a given SLL or the smallest SLL for a given beamwidth (Dolph, 1946). This can be done via the orthogonal functions of Chebyshev (Greenberg, 1998) to design an optimum radiation pattern. Positively, the use of modern EA/EC has solved the deficiency and burdensome of matching the array factor (AF) expression with an appropriate Chebyshev function that appears for large number of antenna array elements (Balanis, 2005).

In addition to that, there are also some limitations of some well−known EA/EC techniques, such as PSO and GA in N−dimensional problem including array geometry synthesis. It has been stated that the original PSO algorithm has difficulties in striking a balance between exploration and exploitation accelerators. In this case, the global search ability of PSO algorithm is restricted due to the drawbacks of particle (population) velocity and position updating mechanism (Ho et al., 2005).

Precisely, the original PSO algorithm updates its velocity of particle based on its own and its companion's flying experiences. The particle moves stochastically based

on a weighted average of the previous best points of its own and its neighborhood (Ho et al., 2005). Since there are two weighting parameters independently and randomly generated, two possible cases may occur either two random parameters are too large or too small. If the random parameters are too large, both the personal and social experiences accumulated so far are overused and the particle is driven too far away from the local optimum. However if the random parameters are too small, both the personal and social experiences are not maximum utilized, and the convergence performance of the algorithm is undermined (Liet et al., 2008). These two cases lead to the lack of control in balancing both global and local searches, hence, reduce the optimal solution diversity of the particles in the search space (Ho et al., 2006). This predicament restricts the antenna array synthesis to find the optimal solutions.

Moreover, the standard GA has its own limitations, such as the premature convergence, a tendency to converge towards local optima or even arbitrary points rather than the global optimum of the multi−peak function optimization problem, low speed of convergence, and in some cases, unable to reach the global optimal convergence state. These shortages are primarily due to the restriction of static crossover and mutation operators in the standard GA itself (Shan, 2010). Furthermore, the number of elements which are exposed to mutation is large with an exponential increase in search space size. The further destructive mutation makes the original GA is extremely difficult to be deployed particularly on complex problems, such as optimization of antenna array excitation components to suppress a low SLL and/or mitigate prescribed nulls whilst preserving the main beam.

Perhaps, the newly CS algorithm through certain modification and hybridization processes may overcome the conventional amplitude tapering and analytical optimization methods drawbacks. In other words, the enhanced and hybrid CS

algorithms can be as the alternative of modern EA/EC techniques for antenna array beam design due to its high versatility, flexibility, diversity, and capability to optimize complex multidimensional problems. Precisely, the enhanced and hybrid CS algorithms can directly manipulate the antenna array elements configuration arranged in space, and produce adaptive uniform directional beamform. In some cases, the modified and hybrid CS algorithms are better than existing metaheuristic algorithms to suppress side lobes and/or mitigate nulls in the Dolph−Chebyshev window by optimizing the gradient of cost function(s).

1.4 Research Objectives

In general, there are four main objectives in this research:

i. To analyze the imperative effects of five internal parameters of original CS algorithm in SO optimization to find optimal elements location, which preserves main lobe, suppresses SLL and/or mitigates nulls.

ii. To introduce the Roulette wheel selection operator, dynamic inertia weight (w), and dynamic discovery rate or fraction probability (P_a) in SO optimization of the modified CS (MCS) algorithm to find optimal elements location, which preserves main lobe, suppresses SLL and/or mitigates nulls.

iii. To propose the hybridization of MCS algorithm with two well−known metaheuristics, which are PSO and GA, in SO optimization to find optimal elements location, and weighted−sum MO optimization of three objective functions to find optimal elements location, excitation amplitude, and excitation phase with three aims: (a) to generate a radiation pattern with small HPBW and high directivity of preserved

main beam; (b) to suppress minimum SLL and/or mitigate prescribed nulls; (c) to attain a high absolute dynamic range ratio (DRR) between maximum and minimum excitation amplitudes.

iv. To propose the hybridization of MCS algorithm with strength Pareto evolutionary algorithm (SPEA) defined as MCSSPEA, MCS algorithm with SPEA and hill climbing (HC) methods known as MCSHCSPEA, and MCS algorithm with SPEA and PSO methods referred as MCSPSOSPEA to find the Pareto fronts via trade−offs of three objective functions, which include non−dominated elements location, excitation amplitude, and excitation phase with three aims as mentioned in objective (iii) above.

1.5 Research Scope

In the initial stage of this research, the effects of five internal parameters of CS optimizer are studied systematically. These internal parameters are α value, distribution type, step size factor, number of nest (population), and discovery rate or fraction probability, P_a in linear array geometry synthesis specifically for the broadside case where main beam steered to the direction of 90°. It is important to identify the imperative characteristic of all the tested parameters and fine–tune them with a sufficient maximum number of iterations to achieve the utmost convergence. Besides that, an investigation on the CS application for antenna with main beam steered to the desired direction other than 90° (non–broadside case) and/or prescribed interferers is also conducted.

On top of that, the newly MCS algorithm will also be proposed to optimize antenna array elements location, which generates a radiation pattern with minimum side

SLL suppression and/or prescribed nulls mitigation. In this experiment, Roulette wheel selection operator, dynamic *w*, and dynamic P_a are introduced in the CS optimizer.

Continuing from that, the hybridization of MCS with PSO (MCSPSO) and GA (MCSGA) algorithms in both SO and weighted–sum MO optimization are postulated towards synthesizing antenna array. The basic idea of the MCSPSO hybrid algorithm presented here has two major operations: firstly, running MCS algorithm and obtaining a global optimum solution, and secondly, improving the result with PSO. On the other hand, the MCSGA hybrid algorithm involves an initial search with MCS and refining solution with GA. Both PSO and GA are stochastic algorithms used to produce a better diversity of optimal solution(s).

In the later stage, the proposed MCS algorithm is combined with the strength Pareto evolutionary algorithm (SPEA) denoted as MCSSPEA to find for the Pareto optimum solutions. There is also the combination of the proposed MCS with the SPEA and the hill climbing (HC) technique or simply known as MCSHCSPEA. Moreover, there is the amalgamation of the proposed MCS with the SPEA and the PSO algorithm referred as MCSPSOSPEA. Each of the MCSSPEA, MCSHCSPEA, and MCSPSOSPEA algorithms has the Pareto front represents the trade−off among three fitness functions. The solutions include antenna array optimal position, excitation amplitude, and excitation phase, which generate a radiation pattern with minimum SLL, high antenna directivity, small HPBW, and/or prescribed nulls significant mitigation while preserving the main lobe radiation.

1.6 Research Significance and Contribution

The primary contribution of this research is the proposed MCS hybrid algorithm in SO approach, hybrid MCSGA and MCSPSO hybrid algorithms in both SO and

weighted–sum MO methods, and MCSSPEA, MCSHCSPEA, and MCSPSOSPEA hybrid algorithms in Pareto front MO approaches for linear antenna array synthesis.

Precisely, the newly proposed MCS, MCSGA, and MCSPSO hybrid algorithms performs SO optimization approach to find optimal elements location, which preserves main lobe, suppresses SLL and/or mitigates nulls. On the other hand, the newly proposed MCS, MCSGA, and MCSPSO hybrid algorithms perform normalized weighted–sum MO optimization approach on three objective functions to find optimal elements location, excitation amplitude, and excitation phase with aims of obtaining small HPBW, high directivity of preserved main lobe, minimum SLL suppression, significant prescribed nulls mitigation, and high absolute DRR between maximum and minimum excitation amplitudes. Similarly, the newly proposed MCSSPEA, MCSHCSPEA, and MCSPSOSPEA hybrid algorithms perform Pareto front MO approach on three objective functions to find non–dominated elements location, excitation amplitude, and excitation phase with the same aims of the weighted–sum approach mentioned above.

1.7 Thesis Organization

This thesis mainly consists of seven chapters:

Chapter 1 briefly introduces the research background of two general smart antenna categories: switched−beam antenna, and adaptive antenna. Besides, the research motivation that highlights three categories of antenna array pattern synthesis, and three general optimization techniques deployed in antenna array synthesis are also explained. Then, problem statement is defined where the limitations of conventional analysis approaches and numerical techniques, and the strength of modern evolutionary algorithms (EA) or evolutionary computation (EC) techniques are highlighted.

Moreover, four research objectives and research scope are stated clearly. Finally, research significance and contribution along with level of improvements achieved are also elaborated.

Chapter 2 briefly reviews the theory of radiation pattern and linear antenna array. Besides, there is discussions regarding well−known approaches in linear antenna array synthesis, e.g. analytical techniques, numerical methods and EA/EC−based techniques comprise genetic algorithm (GA), and particle swarm optimization (PSO). The successful implementations via various approaches are then summarized briefly based on previous published literatures.

Chapter 3 explains the methodology applied in this research along with a general block diagram. Precisely, there is a theoretical description on the linear antenna array configuration used throughout this study. This covers the array factor (AF) definition, which becomes the main component of objective function primarily for SLL suppression and/or nulls mitigation. There is also an explanation about linear antenna design specification in terms of required SLL and nulls normalized measurements in decibel (dB). Besides, there is a description of the input data (solutions) to be optimized in SO approach, which is elements location whereas in both weighted−sum and Pareto front MO approaches, which are elements location, excitation amplitude, and excitation phase. There is also the definition of an objective function minimized in SO approach, and three objective functions minimized in weighted−sum and trade−offs in Pareto front MO optimization approaches. The aims of doing both SO and MO optimization approaches are also included. The related flowcharts, pseudo−codes, and differences of standard CS algorithm and all the proposed versions of modified and hybrid CS algorithms are emphasized. There is also an explanation on the digital signal processing (DSP) circuit, which is the specific component of antenna design where the

all the proposed versions of modified and hybrid CS algorithm will do iterative optimization in finding optimal array elements location in SO approach and elements location, excitation amplitude, and excitation phase in MO approach to be multiplied with complex weight, w_i during beamforming process. There is also a discussion on design benchmarking where all the proposed versions of modified and hybrid CS algorithms are compared relatively with few other existing EA/EC techniques used in linear antenna array synthesis. In the last part, there is a brief description in terms of implementations on both software, which is MATLAB and hardware, which is notebook used to develop and validate all the proposed versions of modified and hybrid CS algorithms throughout this research.

Chapter 4 describes the imperative effects of five internal parameters of standard CS algorithm in linear antenna array synthesis. There is also an analysis on the postulation of MCS algorithm through the introduction of Roulette wheel selection operator, dynamic w, and dynamic P_a towards designing a smart antenna radiation pattern with low SLL suppression and/or significant predefined nulls mitigation. Furthermore, there is a detail description on the postulated hybridization of MCS algorithm with two well−known EA/EC methods, which are PSO forming MCSPSO and GA forming MCSGA in SO optimization to find the optimal location of linear antenna array elements. The results obtained in this study will also be compared with other corresponding existing EA/EC SO optimization methods used for linear antenna array synthesis.

Chapter 5 discusses various simulation tests on the proposed MCSPSO and MCSGA in the normalized weighted−sum MO optimization on three objective functions to find the optimal location, excitation amplitude, and excitation phase of linear antenna array elements, respectively. Then, there is a description on the MCS

algorithm performing MO optimization via Pareto front technique. In this case, MCSSPEA, MCSHCSPEA, and MCSPSOSPEA algorithms are proposed to obtain trade−offs between three objective functions in determining the non−dominated set of optimal solutions, which are linear antenna array elements location, excitation amplitude, and excitation phase. Both weighted−sum and Pareto front MO approaches are deployed to generate radiation pattern with small HPBW, high directivity, low SLL suppression and/or significant predefined nulls mitigation, respectively. The results obtained in this study will also be compared with other respective existing EA/EC MO optimization methods used for linear antenna array synthesis.

Chapter 6 discusses and concludes all the findings found through various simulation tests performed in the previous Chapter 4 and 5, respectively.

Chapter 7 summarize the general conclusions, limitations and recommended future works.

Then there are lists of all the related references used and academic publications originated from this study.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter introduces the concept of antenna and its enhanced version called smart antenna. According to Balanis, C. A. (2005), Webster's Dictionary defines an antenna as "a usually metallic device (as a rod or wire) for radiating or receiving radio waves". Furthermore, the Institute of Electrical and Electronics Engineers Standard Definitions of Terms for Antennas (IEEE Standard 145−1983) defines the antenna or aerial or electromagnetic radiator as "a means for radiating or receiving radio waves".

Balanis, C. A. (2005) then precisely states the antenna as the transitional structure between free−space (unbounded medium) and a guiding device. The guiding device or transmission line can be a coaxial line or a hollow pipe (waveguide), which is applied to transport electromagnetic energy from the transmitting source to the antenna or from the antenna to the receiver. In short, the antenna main purpose is to convert the energy of a guided wave into the energy of a free−space wave (or vice versa) as efficiently as possible, while at the same time the radiated power has a certain desired pattern of distribution in space.

The smart antenna has become as an emerging technology since firstly appeared around 1950s, which can be deployed to cater the capacity, quality of service (QoS), and coverage problems faced by wireless communication under heavy traffic or bigger bandwidth (Yilmazer, Burintramart, & Sarkar, 2008). It evolves in various fields such as military, commercial, and medical. Smart antenna systems use a weighted average of the received signals to automatically adjust the beam towards the signal of interest under a dynamically changing environment to radiate or receive desired signals while

nulling the interferers. In a wireless communications system, antenna is essentially used to distribute data to surrounding space or collect data from surrounding space. Thus, by using the components in a "smart" or an "intelligent" fashion, antenna designers can exploit the limited spectrum and increase the quality of the communication network.

Smart antenna technology requires the expertise of multidisciplinary areas, such as antenna design, electronic communications, adaptive digital signal processing (DSP), and recently, evolutionary algorithm (EA) techniques. The antenna is referred as "smart antenna" because it can steer its beam electronically under dynamically changing environments towards the signal of interest. Furthermore, it can also mitigate the interferers by using the DSP techniques in an intelligent way through a weighted average of the received signals (Yilmazer et al., 2008).

Smart antennas can be implemented using two different methods, which are "switched−beam array" and "adaptive array". In the switched−beam array system, a number of predetermined fixed−beam patterns are generated to cover the range of interest. The switching algorithm selects the fixed−beam along the direction of maximum signal strength as demand changes throughout the sector. Based on Figure 2.1(a), instead of shaping the directional antenna pattern with the metallic properties and physical design of a single element (like a sectored antenna), switched−beam systems combine the outputs of multiple antennas in such a way as to form finely sectored (directional) beams with more spatial selectivity than it can be achieved with a conventional single element approach (Stevanović, Skrivervik & Mosig, 2003).

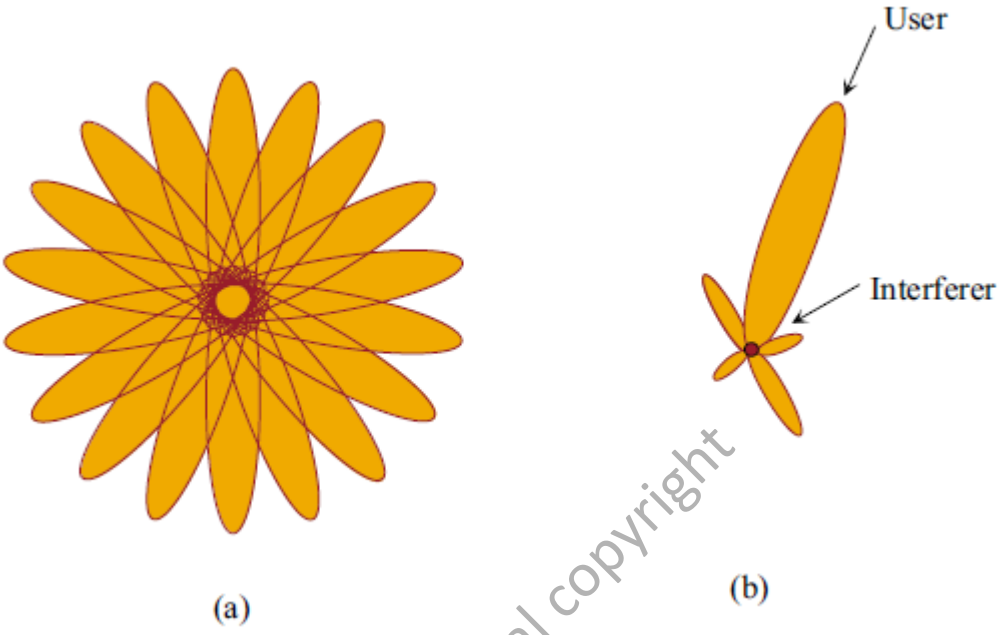


Figure 2.1: (a) Switched–beam system coverage patterns, and (b) Adaptive array coverage (Stevanović, Skrivervik and Mosig, 2003).

The switched−beam array system is easier to implement as compared to the adaptive array system. In the case of strong interferers or nulls, the method is not efficient since the beam pattern is limited by the main beamwidth, so interferers may not be mitigated (Litva, 1996, Liberti, 1999 and Ho, 1998). The most common technique to implement the switched−beam algorithm is done by using the Butler matrix array (Butler, 1961).

In the adaptive array system, the main beam is steered electronically towards the signal of interest, and interferers are nulled at the same time by changing the phase of the voltages at the transmitter and/or receiver antenna array. In this case, a theoretically infinite number of predefined patterns or beams are adjusted in real−time (scenario−based) according to the spatial changes of signal of interest and signal not of interest. This is accomplished by taking a weighted−average of the received signals. Using a variety of new signal processing algorithms, the adaptive array system takes

advantage of its ability to effectively locate and track various types of signals to dynamically minimize interference and maximize intended signal reception as in Figure 2.1(b).

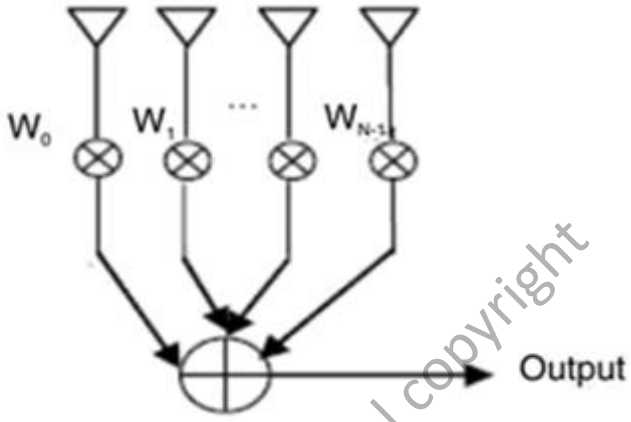


Figure 2.2: Beamforming in an adaptive array system (Yilmazer et al., 2008).

A basic diagram of the adaptive array system is shown Figure 2.2. In beamforming process, each antenna element output is multiplied by a complex weight, w_i. Multiplying with these weights, the amplitude and phase are changed, so that the beam is steered. The antenna array pattern is changed dynamically to mitigate multipath effects and interferences while increasing range and reducing power consumption of the system.

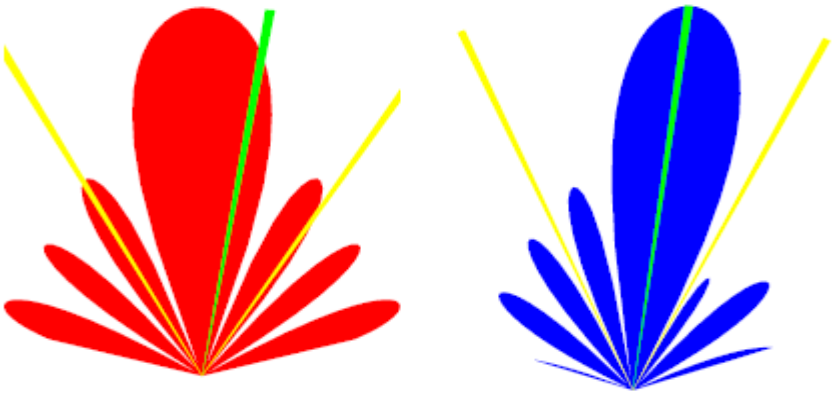


Figure 2.3: Beamforming lobes and nulls that switched–beam (red), and Adaptive array (blue) systems with identical user signals (green line) and co−channel interferers (yellow lines) (Stevanović, Skrivervik and Mosig, 2003).

Figure 2.3 illustrates the beam patterns that each system might choose in the face of a signal of interest and two co−channel interferers in the positions shown. The switched−beam system is shown in red on the left while the adaptive system is shown in blue on the right. The green lines depict the signal of interest while the yellow lines represent the direction of the co−channel interfering signals. Both systems direct the main lobe with the most gain in the general direction of the signal of interest, although the adaptive system chooses a more accurate placement, providing a greater signal enhancement. Similarly, the interfering signals or nulls arrive at places of lower gain outside the main lobe, but again the adaptive system has placed these signals at the lowest possible gain points and ideally, the main signal received maximum enhancement while the interfering signals receive maximum suppression (Stevanović, Skrivervik and Mosig, 2003).

In summary, fixed beam system uses multiple fixed beams with narrow beamwidth. The phase shifts can be implemented by using the Butler matrix so that multiple beams are generated in a simple way. Multipath signals and interferers might be enhanced since the system cannot differentiate between signal of interest and interferers, thus, the co−channel interference reduction will be less than in the adaptive system since there is no null steering (Tsoulos, 2001). In other words, the adaptive array system has a better interference rejection performance as compared to the switched−beam array system, hence has a better coverage and capacity.

2.2 Radiation Pattern Theory

Various parts of a radiation pattern are referred as lobes, which are sub−classified into major or main, minor, side, and back lobes (Balanis, 1997). A

radiation lobe is a portion of the radiation pattern bounded by regions of relatively weak radiation intensity or strength.

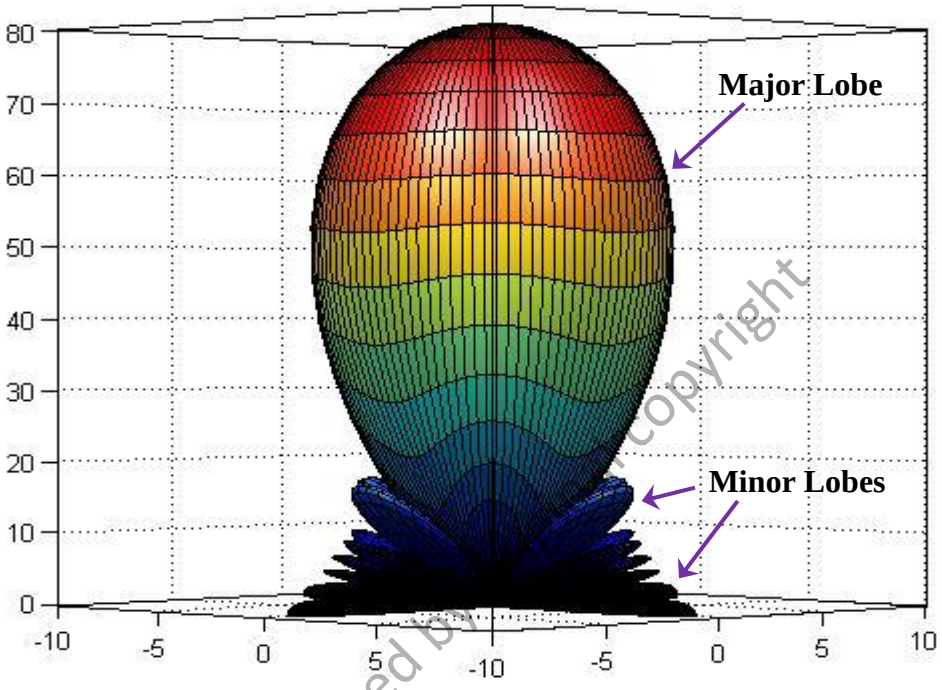


Figure 2.4: Radiation lobes of a three−dimensional (3D) antenna pattern.

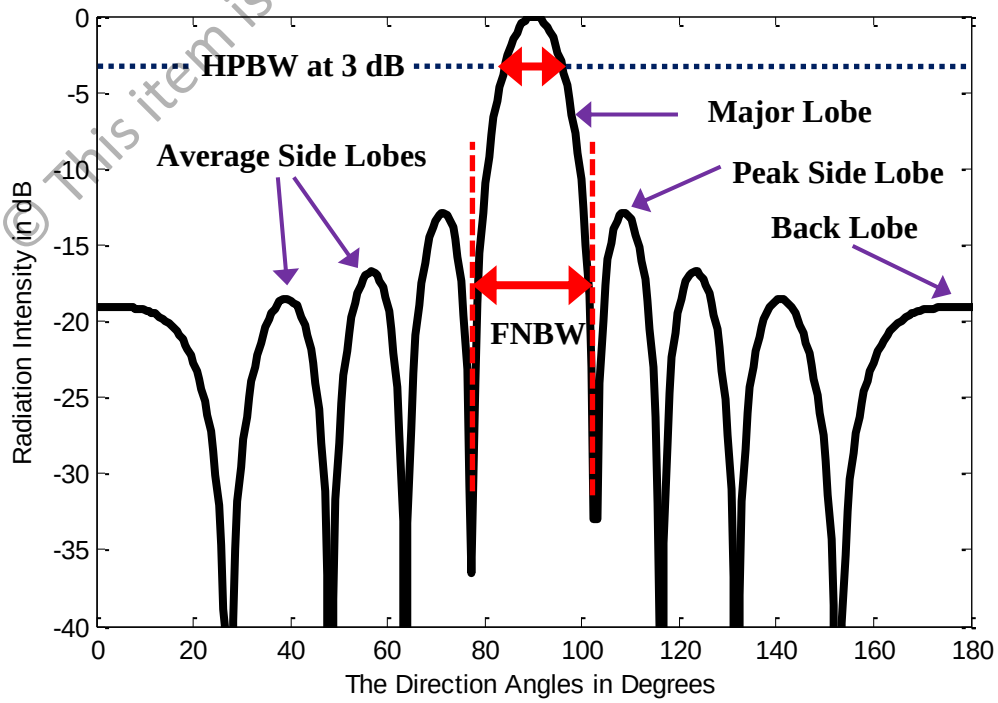


Figure 2.5: Linear plot of power pattern and its associated lobes and beamwidths.

Figure 2.4 depicts a symmetrical three−dimensional (3D) polar pattern with a major and various minor lobes in various sizes. The bigger the lobe the greater radiation intensity it portrays. Moreover, Figure 2.5 illustrates a linear two−dimensional (2D) of normalized radiation pattern where the half−power beamwidth (HPBW) and first null beamwidth (FNBW) are indicated.

A major lobe or main beam is defined as the radiation lobe containing the direction of maximum radiation. In Figure 2.5, the major lobe is pointing in the $\theta = 90°$ direction. In this case, all the lobes with the exception of the major can be categorized as minor lobes. A peak side lobe, average side lobes, and a back lobe are radiation lobes in any direction other than the intended major lobe. Usually, a side lobe is adjacent to the main lobe and occupies the hemisphere in the direction of the main beam. A back lobe is a radiation lobe whose axis makes an angle of approximately 180° with respect to the beam of an antenna. Normally, it refers as a minor lobe, which occupies the hemisphere in a direction opposite to that of the major or main lobe (Balanis, 2005). The HPBW is defined as the angle between the points on the main lobe that are 3 dB lower in gain compared to the maximum. In addition to that, FNBW is referred as the angular span between the first pattern nulls adjacent to the main lobe.

Minor lobes typically represent radiation in undesired directions and should be minimized. Side lobes are normally the largest of the minor lobes. The level of minor lobes is typically expressed as a ratio of the power density in the minor lobe to the major lobe. This ratio is often termed as the side lobe ratio or commonly known as side lobe level (SLL). A SLL of −20dB or smaller is usually desirable in most applications. Achieving a SLL of −30dB or smaller technically requires a very careful design and development approaches. In radar systems, low SLL is critical to minimize the false target indications through side lobes (Balanis, 2005).

2.3 Linear Antenna Array Theory

An antenna can have one or more element (radiator) to emit electromagnetic wave. Usually the radiation pattern of single−element antenna is relatively wide, thus, it has relatively low directivity (gain). In long distance communications, antenna with high directivity is often required. Such antenna is possible to construct by enlarging the dimensions of the radiating aperture (maximum size much larger than wavelength, λ). This approach however may lead to the appearance of multiple side lobes. Besides, the antenna is usually large and difficult to fabricate (Nikolova, 2012).

Another way to increase the electrical size of an antenna is to construct it as an assembly or aggregate of radiating elements in a proper electrical and geometrical configuration known as antenna array. Usually, the elements of the array are identical. This is not necessary but it is practical and simpler for design and fabrication. The arrangement of the array adds up the radiation from individual elements to provide a maximum intensity in a particular direction (Balanis, 2005). The individual element can be wire dipoles, loops, apertures, etc.

The array of elements can reduce power consumption and enhance spectral efficiency for the antenna to perform a high power transmission (Shihab, Najjar, Dib & Khodier, 2008). The fields from the array elements must add constructively in the desired direction, and cancel each other in the remaining spaces to provide a directive pattern. This can reduce interference from the side lobes of the antenna. There are five basic methods to control the overall antenna pattern (Nikolova, 2012 and Banerjee & Dwivedi, 2013):

i. The geometrical configuration of the overall array e.g. linear, circular, spherical, rectangular, etc.
ii. The relative placement or position of the individual elements.

iii. The excitation amplitude of the individual elements.

iv. The excitation phase of each element.

v. The individual pattern of each element.

In designing a smart antenna, various synthesis techniques are performed to determine the physical layout or geometry of the array that produces the radiation pattern closest to the desired pattern. Some of these techniques also focus on the null control to reduce the effects of undesired interference and jamming. This can be achieved by controlling some of the parameters of the array elements, such as the position, amplitude, phase, and complex weights for both the amplitude and phase (Banerjee & Dwivedi, 2013).

Briefly, the shape of the desired pattern can vary widely depending on the application (Pal, Basak, Das & Abraham, 2009). Moreover, the array pattern ideally should possess a high power gain, low side lobes, a controllable beamwidth (Zaharis et al., 2006), and a good azimuthal symmetry.

2.4 Justification of Synthesizing Antenna Array

In this study, the symmetric linear array was the chosen antenna geometry to be synthesized. This is because the geometry assembles a group radiating multi−elements, which enlarge the electrical size of an antenna to gain a high directivity or gain with a narrow main beam. This is necessary to fulfil the existing high bandwidth and quality of service (QoS) demands for long distance wireless communications. The individual elements of the linear array can be of any form of wires, and apertures (Balanis, 2005). Besides to that, cuckoo search (CS) algorithm is modified (enhanced) in this research because it has a significant advantage of being a generic optimizer model regardless types of antenna arrays geometries, such as linear, rectangular, and circular.

2.5 Analytical Techniques in Smart Antenna Design

Two well−known analytical array synthesis methods used are Fourier technique (FT), and Dolph−Chebyshev methods. Keizer (2009), applied the iterative Fourier technique (IFT) for the synthesis of linear arrays with uniform element spacing. In this case, an IFT relationship existed between the array factor (AF) and the element excitations. This relationship was examined iteratively to derive the array element excitations from the prescribed AF. The IFT was found suitable for the large arrays with periodic spacing of the array elements, and capable to handle various design constraints related to both the radiation and aperture domains.

The far−field $F(u)$ of a linear array with M elements arranged along a periodic grid at distance d apart, can be written as the product of the embedded element pattern EF and AF:

$$F(u) = EF(u)AF(u) \tag{2.1}$$

$$AF(u) = \sum_{m=0}^{M-1} A_m e^{jkmdu} \tag{2.2}$$

where A_m is the complex excitation amplitude of the mth element, $k = 2\pi / \lambda$ is the wavenumber, λ is the wavelength, $u = \sin\theta$, and θ is the angular coordinate measured between the far−field direction and the array normal.

Equation (2.2) forms a finite Fourier series that relates the element excitation coefficients A_m of the array to its AF through a discrete IFT. AF is periodic in u−dimension over the interval d/λ. The discrete IFT applied on AF over the period λ/d will yield the element excitations A_m. These Fourier transform relationships are used

iteratively to synthesize a periodic element arrangement of linear array with a low side lobe level (SLL).

Implementation of the IFT algorithm for linear antenna array synthesis using amplitude−only element weighting operates as follows (Keizer, 2009):

1. Start the synthesis using a uniform excitation for M elements in case of the sum pattern and an odd linear taper when the synthesis involves the different pattern.
2. Compute AF from $\{A_m\}$ using a K−point inverse FFT with $K > M$.
3. Adapt AF to the prescribed side lobe constraints.
4. Compute $\{A_m\}$ for the adapted AF using a K−point direct FFT.
5. Truncate $\{A_m\}$ from K samples to M samples by making zero all samples outside the array.
6. Make the phase of the M samples of $\{A_m\}$ equal to the phase of initial excitation at Step 1.
7. Set the magnitude of the excitations violating the amplitude dynamic range constraint to the lowest permissible value.
8. Enforce the optional defective element constraint and take element failures into account by setting their excitation values to zero.
9. Repeat Steps 2−9 until the prescribed side lobe requirements for AF are satisfied or the allowed number of iterations is reached.

In order to apply phase−only synthesis the present Step 6 has to be replaced by: "Make the amplitude of the M samples of $\{A_m\}$ equal to one". When Step 6 is deleted, the synthesis is of the complex weight type. Any low SLL AF consisting of K samples can be realized with K element excitations. The objective of the IFT low SLL synthesis method is to arrange that the M elements of the array took completely over the contribution to AF of the excitations of the $(K - M)$ elements located outside the

aperture. This means that the synthesis is successful when the excitations of $(K - M)$ elements outside the aperture become zero.

The first experiment was performed on a linear antenna array consisting of 80 elements spaced 0.5 wavelength, λ apart and characterized by an isotropic embedded element pattern. For this experiment, a sum and different patterns were synthesized where both featuring a −45 dB maximum peak SLL.

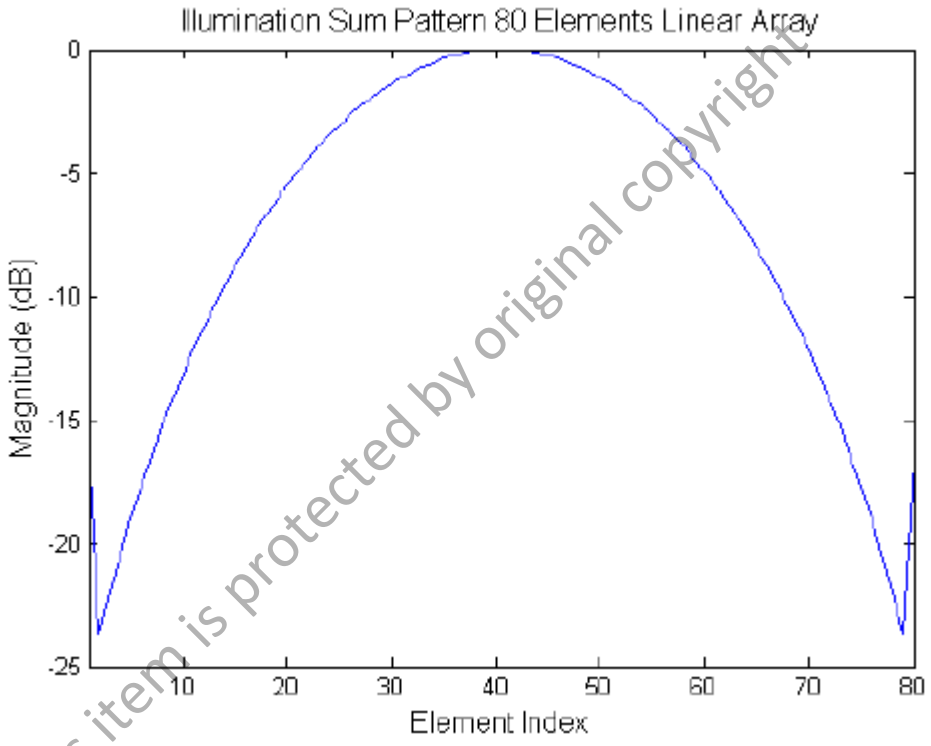


Figure 2.6: Amplitude taper (Keizer, 2009).

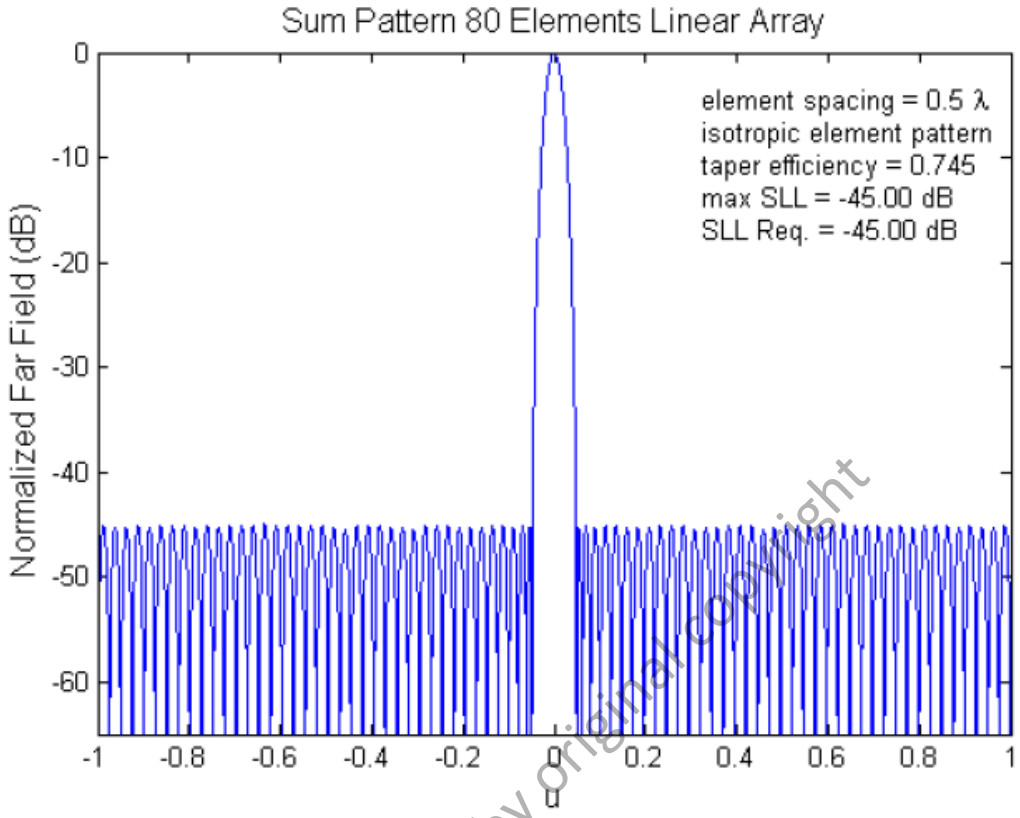


Figure 2.7: Normalized radiation pattern for amplitude–only synthesis (Keizer, 2009).

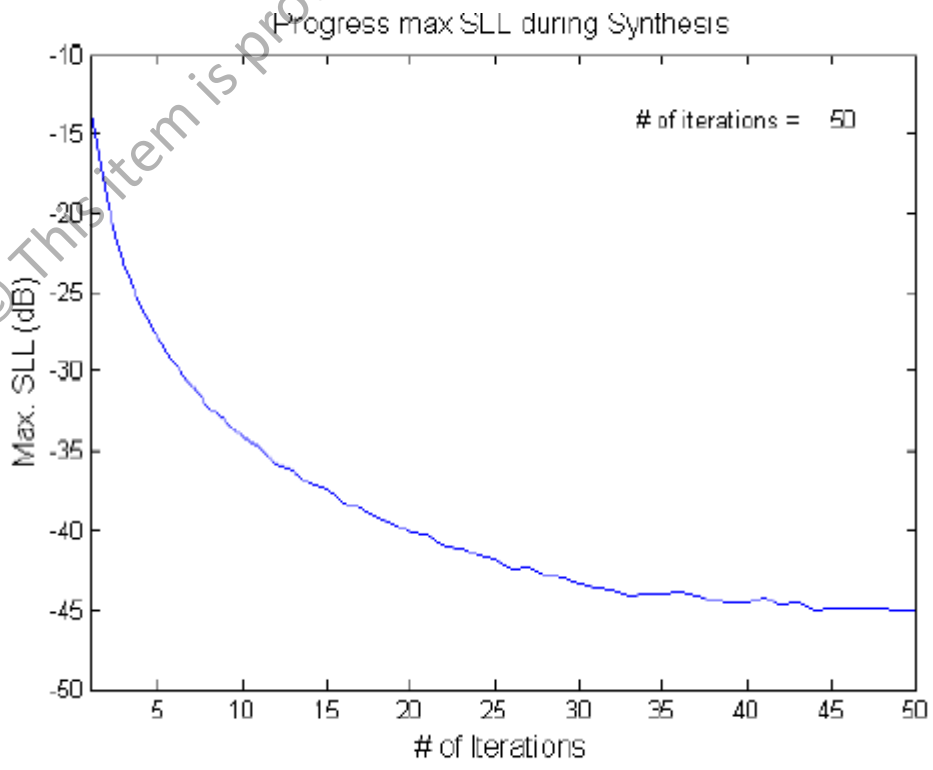


Figure 2.8: Maximum peak SLL vs. Number of iterations (Keizer, 2009).

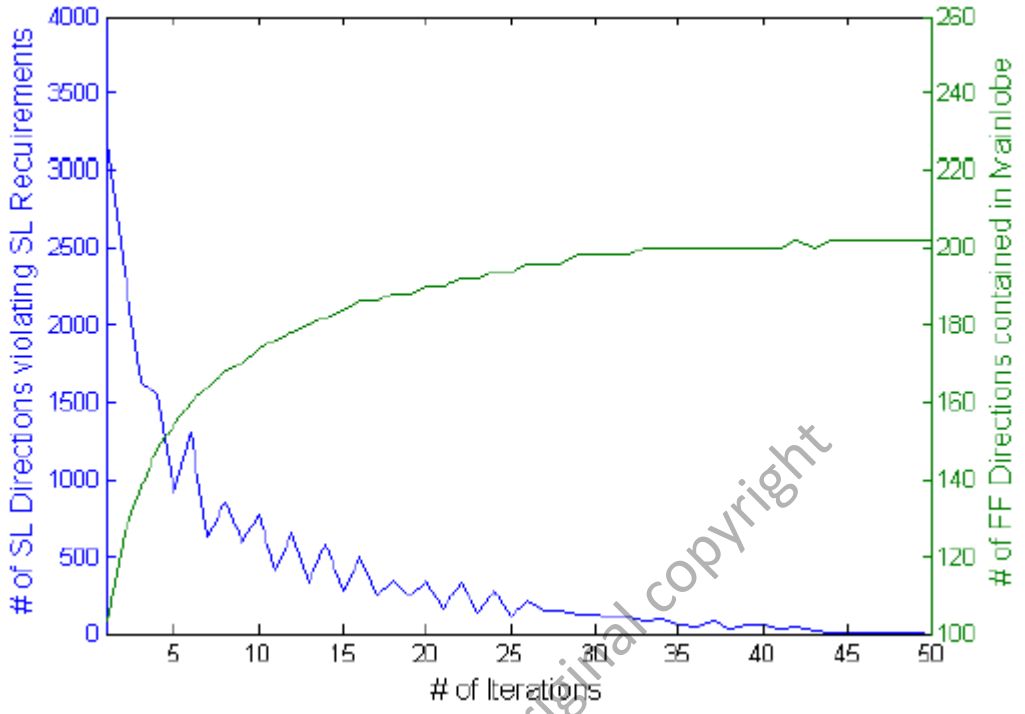


Figure 2.9: Number of far−field directions violating the −45 dB SLL requirement, and number of far−field directions vs. Number of iterations (Keizer, 2009).

Based on Figure 2.6, as the dynamic amplitude range of the element excitations was set 24 dB for the taper, the linear antenna array revealed the sum pattern with a uniform −45 dB peak SLL as depicted in Figure 2.7. The maximum peak SLL suppression curve throughout 50 iterations is shown in Figure 2.8. Besides, the number of far−field directions contained in the main lobe region, which is its direct width measurement is depicted in Figure 2.9. Keizer (2009), also found that the 3 dB larger dynamic range for the different taper produced only a marginal effect on the different normalized pattern, such as generated equal level peak side lobes to the AF.

Another recognized analytical method used in antenna array synthesis is the Dolph−Chebyshev method, which has been originally proposed since 1946 (Dolph, 1946). Since then, many studies have been done to extend the classic Dolph−Chebyshev method to provide new possibilities and choices of communications

and radar systems applications with a low SLL, adjustable beamwidth of main lobe, and high directivity. Alexopoulus (2006) analyzed a phased array design using conventional Dolph−Chebyshev side lobe tapering technique whereas Abreau and Kohno (2002) studied uniform linear array (ULA) and uniform circular array (UCA) designs with low Dolph−Chebyshev like side lobe beam patterns, an adjustable main lobe beamwidth and steering−invariance. Lynch (1997), analysed a simple digital filter based on the Dolph−Chebyshev window, which has properties similar to an optimal low−pass filter (LPF) for a numerical prediction.

Precisely, in 2006, Alexopoulus studied the antenna synthesis characteristics of linear phased rays for both broadside and end fire cases. The modified Chebyshev (MC) side lobe tapering technique is proposed to analyse array parameters, such as the number of elements *N*, the phase factor, side lobe level (SLL), directivity, radiation pattern, element separation, and impact upon signal−to−noise ratio (SNR). The MC method is developed to overcome two inefficiencies of conventional Dolph−Chebyshev arrays: firstly, suffering from directivity saturation when the number of radiating elements becomes large, and secondly, for every radiation pattern a new complicated polynomial series needs to be found.

Theoretically, the MC method is based on the synthesis of arrays using the zeroes of conventional Chebyshev arrays repeatedly. The idea behind the MC formulation is it makes no direct use of Chebyshev polynomials. The array factor (AF) is calculated in terms of cosine and hyperbolic cosine functions respectively, while a system of equations for the excitation amplitudes is obtained from the zeroes of the AF. The solution of the system of equations for the excitation amplitudes is done by allowing one of those excitation amplitudes to be as an independent variable.

For an even and odd number of linear antenna array elements, the AF is defined as below:

$$f(\psi) = 2\sum_{m=1}^{n} a_m \cos\left[\left(m - \frac{1}{2}\right)\psi\right]; N = 2n \tag{2.3}$$

$$f(\psi) = a_0 + 2\sum_{m=1}^{n} a_m \cos[m\psi]; N = 2n + 1 \tag{2.4}$$

where ($\psi = \beta d \cos(\theta) + \alpha$, $\beta = 2\pi/\lambda$); λ is the wavelength, d is the element spacing, α is the phase factor, θ is the angle measured from the line of the array, a_m is the magnitude of the amplitude for the mth element on either side of the array midpoint, and a_0 denotes the amplitude of the centre element when N is odd. Given the fact that Chebyshev polynomials satisfy the relationships $T_n(\cos x) = \cos(nx)$ for $|T_n| < 1$ and $T_n \cos(hx) = \cosh(nx)$ for $T_n \geq 1$, the AF expressed in (2.3) and (2.4) is defined as:

$$|f(\psi)| = \left|f(\bar{\psi})\right| = \begin{cases} C\left|\cos\left[\left(\frac{(N-1)}{2}\bar{\psi}\right)\right]\right|; |f(\psi)| \leq C \\ C\cosh\left[\left(\frac{(N-1)}{2}\bar{\psi}\right)\right]; |f(\psi)| \geq C \end{cases}, \tag{2.5}$$

where C is a positive constant coefficient, and ψ is related to $\bar{\psi}$ by:

$$\gamma\cos\left(\frac{\psi}{2}\right) = \begin{cases} \cos\left(\frac{\bar{\psi}}{2}\right); |f(\psi)| \leq C \\ \cosh\left(\frac{\bar{\psi}}{2}\right); |f(\psi)| \geq C \end{cases} \tag{2.6}$$

Through profound investigation of (2.5), $f(\psi)$ gains its maximum value when $\bar{\psi} = 0$, which corresponds to $\bar{\psi} \equiv \bar{\psi}_0 = 2\cosh^{-1}(\gamma)$. The maximum SLL ratio, R can be expressed as:

$$R = \cosh\left[\left(\frac{(N-1)}{2}\bar{\psi}_0\right)\right] = \cosh[(N-1)\cosh^{-1}(\gamma)]$$

(2.7)

The SLL here is assumed as $-20\log_{10}(R)$ in dB. By rearranging (2.7), γ then can be stated as:

$$\gamma = \cosh\left[\frac{1}{(N-1)}\log_e\left(R + \sqrt{R^2-1}\right)\right]$$

(2.8)

In this case, Alexopoulus (2006) has claimed that we can calculate γ if we know N and the half−power beamwidth (HPBW) or if we know R and the HPBW. The AF can be normalized to unity through determining the coefficient $C = 1/R$. A system of equations that governs the magnitude of the radiating elements can be constructed from the zeroes of the AF in terms of matrices. The zeroes are now obtained using the equation:

$$\bar{\psi}_m = \frac{(2m-1)\pi}{N-1}$$

(2.9)

For $m = 1, 2, 3 \ldots M$, while $M = (N-2)/2$ for N even and $M = (N-1)/2$ for N odd number of radiating elements, respectively. The zeroes in terms of ψ is calculated explicitly as:

$$\psi_m = 2\cos^{-1}\left[\frac{1}{\gamma}\cos\left(\frac{(2m-1)\pi}{2(N-1)}\right)\right]$$

(2.10)

where m = 1, 2, 3…*M*.

The ψ_m's of $f(\psi)$ in (2.3) and (2.4) are zeroes stated as below:

$$f(\psi_m) = 0, m = 1,2,3, \ldots, M$$

(2.11)

From (2.11), Alexopoulus (2006) has stated that we can obtain a system of M equations with $M + 1$ unknown amplitude of the radiating elements a_{M}, one of which is chosen as the independent variable. In the case of an even number of elements, the independent variable is chosen to be $a_{\mathrm{M+1}}$. We can form matrices that allow us to solve for any number of array elements N. The solutions for a_m satisfy the AF as in (2.3) and (2.4). The ratio of the SLL, R is used to define variable z_0:

$$z_0 = \frac{1}{2}\left[\left(R + \sqrt{R^2 - 1}\right)^{\frac{1}{(M-1)}} + \left(R - \sqrt{R^2 - 1}\right)^{\frac{1}{(M-1)}}\right] \tag{2.12}$$

For an even number of elements, $M = N/2$, the amplitude defined as:

$$a_n = \sum_{q=n}^{M} \frac{(2M-1)(q+M-2)!}{(q-n)!\,(q+n-1)!\,(M-q)!}(-1)^{M-q} z_0^{2q-1} \tag{2.13}$$

For an odd number of elements, $M = (N - 1)/2$, the amplitude by the series expansion explicitly yields:

$$a_n = \sum_{q=n}^{M-1} \frac{2M(q+M-2)!}{\varepsilon_n (q-n)!\,(q+n-2)!\,(M-q+1)!}(-1)^{M-q+1} z_0^{2(q-1)} \tag{2.14}$$

such that the additional condition, ε_n is stated as:

$$\varepsilon_n = \begin{cases} 2, n = 1 \\ 1, n \neq 1 \end{cases} \tag{2.15}$$

where n = 1, 2, 3,….

The even and odd solutions of AF then can be expanded as:

$$AF^{(even)} = \sum_{n=1}^{M} a_n \cos[(2n-1)u] \tag{2.16}$$

and

$$AF^{(odd)} = \sum_{n=1}^{M+1} a_n \cos[(2n-1)u]$$

(2.17)

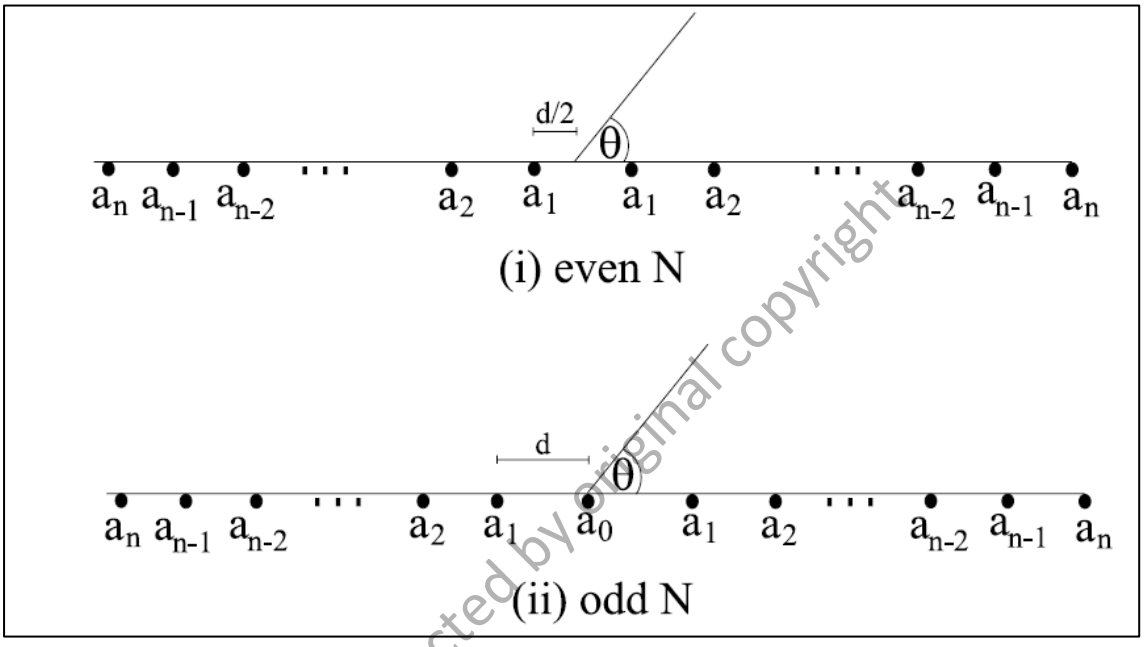


Figure 2.10: (i) Normalizing all other amplitudes by the edge element a_n(ii) The symmetry with amplitude a_0 at a distance of *d* from a_1 (Alexopoulus, 2006).

In the proposed equally spaced MC arrays, the amplitudes of the edge elements are always unity and all other amplitudes are normalized by the former. Based on Figure 2.10, as examples, for *N* = 3 (odd) and, |SLL| = −10 dB the a_0 is 1.0390 whereas $a_2,\ldots,a_{n-1}$, a_n equal to 1; for *N* = 4 (even) and |SLL| = −10 dB, the $a_1$is 0.8794 whereas $a_2,\ldots,a_{n-1}$, a_n equal to 1; for *N* = 5 (odd) and |SLL| = −30 dB, the a_0 is 3.1397 or 2.4123 whereas $a_2,\ldots,a_{n-1}$, a_n equal to 1; for *N* = 6 (even) and |SLL| = −30 dB, the $a_1$is 3.3828 or 2.3129 whereas $a_2,\ldots,a_{n-1}$, a_n equal to 1.

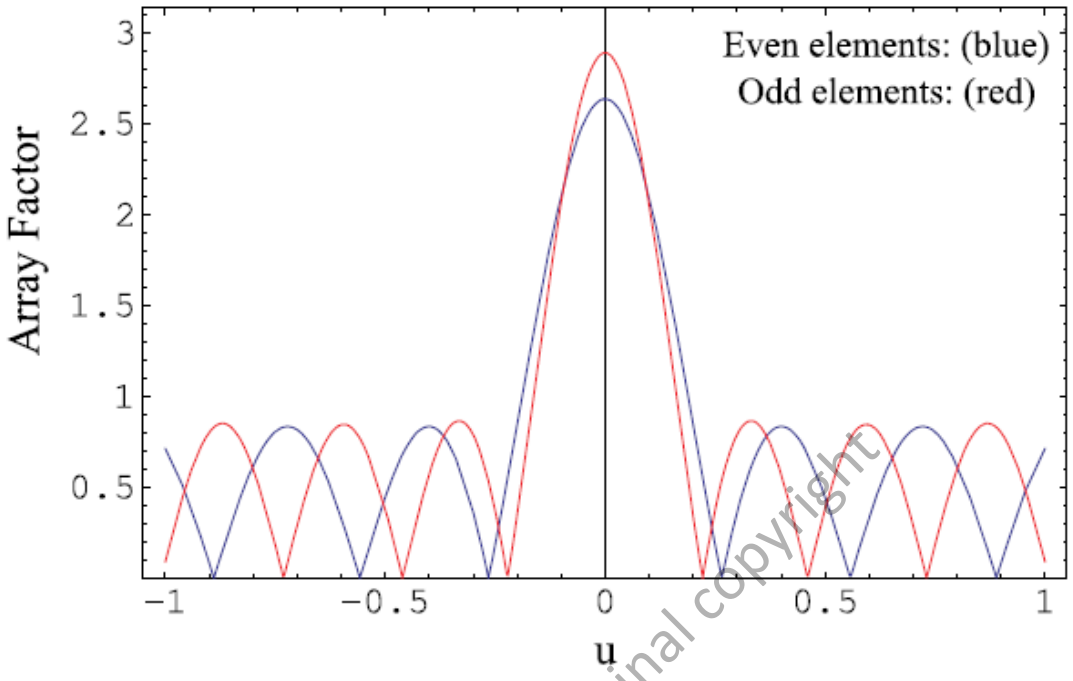


Figure 2.11: AF as a function of *u* for |SLL| = 10 dB with *N* = 11 (red) odd elements, and *N* = 10 (blue) even elements (Alexopoulus, 2006).

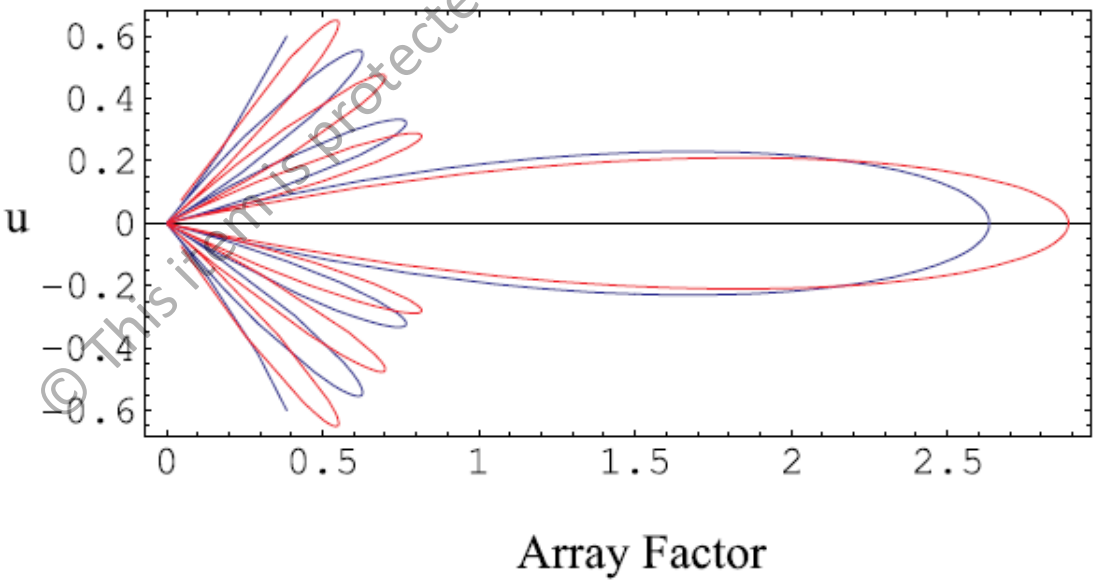


Figure 2.12: Polar plot for AF as a function of *u* for |SLL| = 10 dB with *N* = 11 (red) odd elements, and *N* = 10 (blue) even elements (Alexopoulus, 2006).

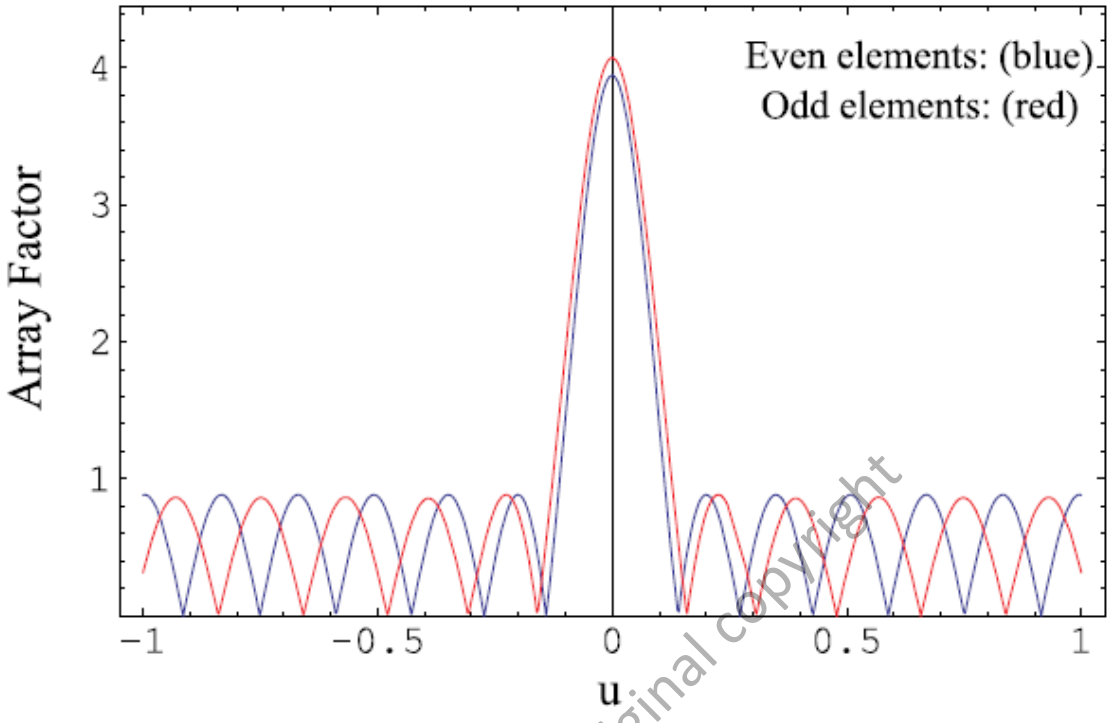


Figure 2.13: AF as a function of u for |SLL| = −13 dB with N = 17 (red) odd elements, and N = 20 (blue) even elements (Alexopoulus, 2006).

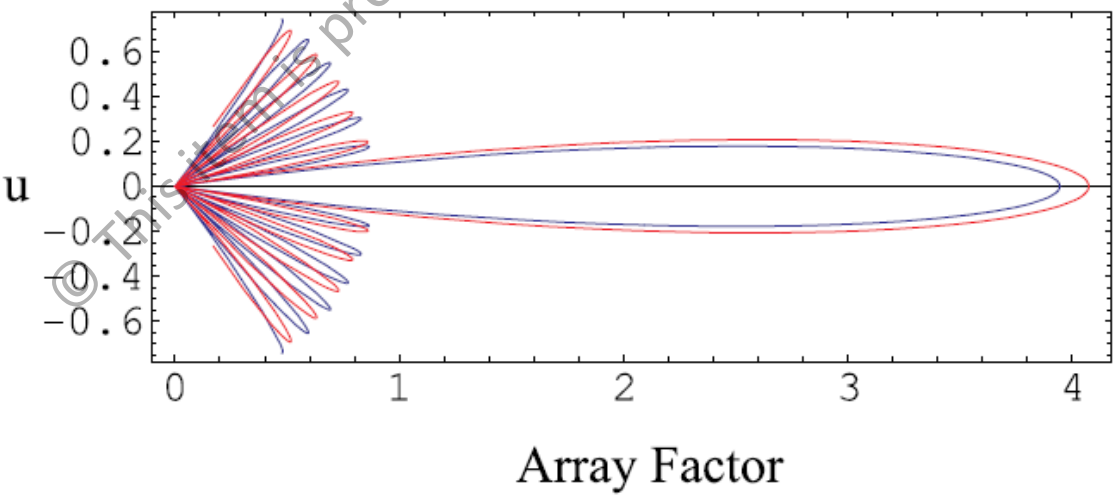


Figure 2.14: Polar plot for AF as a function of u for |SLL| = −13 dB with N = 17 (red) odd elements, and N = 20 (blue) even elements (Alexopoulus, 2006).

Figure 2.11 depicts the modified Chebyshev (MC) arrays radiation pattern with N = 10 and N = 11 elements using (2.16) and (2.17), respectively with the SLL

equiripple at −10 dB. Figure 2.12 on the other hand, shows the respective polar plot. Furthermore, Figure 2.13 depicts the MC arrays radiation pattern with N = 17 and N = 20 elements, respectively with the SLL equiripple at −13 dB whereas Figure 2.14 shows the respective polar plot. Finally, Alexopoulus (2006) has concluded that both the gain (directivity) of main lobe, and number of SLL increase as the number of even or odd elements is increased leading to smaller inter−element spacing.

2.6 Numerical Methods in Smart Antenna Design

Gomez and Covarrubias (2009) have postulated a unified mathematical approach known as "Legendre functions" to nonlinear optimization of multidimensional array geometries. Precisely, the method determines the optimal excitation and spacing as a polynomial problem. In this case, the unified mathematical approach performs the synthesis of the radiation pattern with characteristics of high directivity, minimum SLL, and good adaptability to the radio channel.

In order to establish the AF in terms of Legendre polynomials, Gomez and Covarrubias (2009) have deployed the second−order ordinary differential equation (2nd ODE), which yields solutions called Legendre functions:

$$(1-x^2)\frac{d^2y}{dx^2} - 2x\frac{dy}{dx} + n(n+1)y = 0; n = 0,1,2,3,\dots,\infty \tag{2.18}$$

If n is zero or positive integer, these functions denominate Legendre polynomials. A compact expression of $P_n(x)$ is given by the following expression:

$$P_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}[(x^2-1)^n]; n = 0,1,2,3,\dots,\infty \tag{2.19}$$

The Legendre polynomials have a property of using a recurrence relation to define a recurrent sequence as stated:

$$(n+1)P_{n+1}(x) + nP_{n-1}(x) = (2n+1)xP_n(x); n = 1,2,3, \dots, \infty$$

(2.20)

The Legendre function is a generic model, which can be implemented to diverse types of antenna arrays geometries. Nomenclatures used are enlisted as:

$k = 2\pi/\lambda$: Free space wavenumber;

d_n: Element position;

I_n: Element excitation;

ε_n: Element position perturbation;

Δu: Sampling interval;

M: Sampled points;

α_p, β_n: Transformation vectors;

$P_{(m-1/2)}(\cos \alpha)$: Legendre functions of fractional order;

φ: Progressive phase;

θ_d: Direction of main lobe.

Gomez and Covarrubias (2009) have applied a Legendre function of fractional order, $P_{(m-1/2)}(\cos \alpha)$ in which, a recurrence relation from (2.20) is manipulated for the values of $m \geq 2$, and $n \geq 3$, such as:

$$(n_1 + 0.5)P_n(x) = 2n_1 xP_{n-1}(x) - (n_1 = 0.5)P_{n-2}(x)$$

(2.21)

Using (2.21), Gomez and Covarrubias (2009) have obtained the following values of the Legendre polynomials:

$$P_n(\cos \alpha) = P_{m-1/2}(\cos \alpha) = \frac{2n_1(\cos \alpha)P_{n-1}(\cos \alpha) - (n_1 - 0.5)P_{n-2}(\cos \alpha)}{(n_1 + 0.5)}$$

(2.22)

where $n_1 = n - 2$.

The antenna geometry considered in this study is a symmetric linear array of $2N + 1$ element with AF given as (Gomez & Covarrubias, 2009):

$$AF = E(u) = \sum_{n=0}^{N} \varepsilon_n I_n \cos(k d_n u)$$

(2.23)

where, $u = \cos(\theta) + \varphi$ in the limit $0 \leq \theta \leq \pi$ radians. In order to establish the AF in terms of Legendre polynomials, a desired array pattern is defined as:

$E_d(u)$, in the interval $-1 \leq u \leq 1$.

(2.24)

Since the AF is symmetric, e.g. $E(-u) = E(u)$, the synthesis is performed in the interval $0 \leq u \leq 1$ to obtain:

$$E(u_m) = \sum_{n=0}^{N} \varepsilon_n I_n \cos(m\beta_n - \varphi_n); m = 0,1,2, \dots, M-1$$

(2.25)

where $\Delta u = 1/(M-1)$; $u_m = m\Delta u$; $\varphi_n = k d_n \cos(\theta_d)$; and $\beta_n = k d_n \Delta u$. The following step the Legendre transformation $F(\alpha_p)$ application to the AF to get a triangular set of equations with the final expression:

$$F(\alpha_P) = \sum_{m=0}^{M-1} \varepsilon_m E_d(u_m) P_{m-1/2}(\cos \alpha_p); p = 0,1,2,3, \dots, N$$

(2.26)

where $\varepsilon_m = 1$, $m = 0$; $\varepsilon_m = 2$, $m > 0$.

The Legendre transformation of the desired array pattern $E_d(u)$ is motivated by the following limiting relation for the Legendre polynomial of fractional order:

$$f(\alpha, \beta) = \sum_{m=0}^{\infty} \varepsilon_m P_{m-1/2}(\cos \alpha) \cos(m\beta)$$

$$f(\alpha,\beta) = [2/(\cos\beta - \cos\alpha)]^{1/2}, 0 \le \beta < \alpha$$

$$f(\alpha,\beta) = 0, \alpha < \beta < \pi$$

(2.27)

The triangular system of equations can be expressed using (2.26) and (2.27) as:

$$F(\alpha_p) = \sum_{n=0}^{p} I_n f(\alpha_p, \beta_n)$$

(2.28)

Based on (2.28), the triangular system becomes invertible to find the value of the first element current and the *p*−th element current as stated:

$$I_0 = F(\alpha_0)/f(\alpha_0,\beta_0)$$

$$I_p = \frac{F(\alpha_p) - \sum_{n=0}^{p-1} I_n f(\alpha_p,\beta_n)}{f(\alpha_p,\beta_p)}; p = 1,2,3,\ldots,N$$

(2.29)

with the above parameter (*I*), it is possible to synthesize the radiation pattern of linear antenna arrays easily and quickly.

In the first simulation, the number of the elements of the linear array was set to 17 and the main lobe was steered to 90^{o}, respectively. This was done to extend the previous unequally spaced arrays study presented by Kumar and Branner (1999).

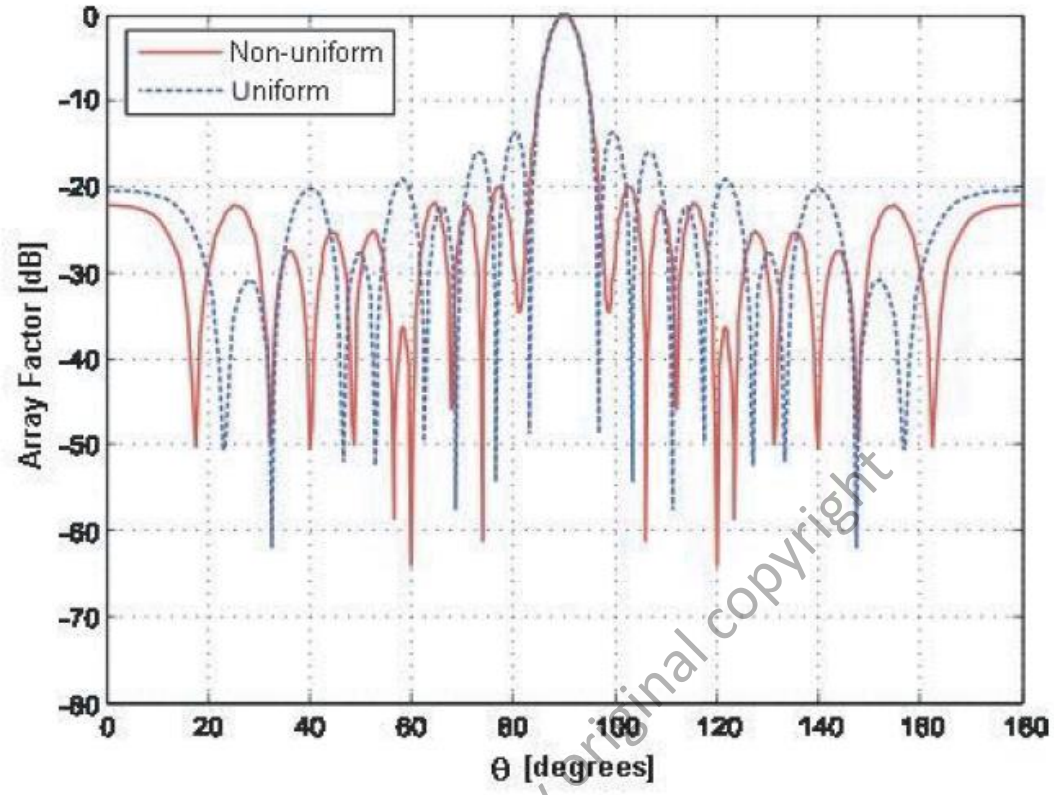


Figure 2.15: Array pattern for the Legendre functions synthesis (Gomez & Covarrubias, 2009).

Figure 2.15 illustrates the results comparatives obtained for both uniform and non−uniform patterns in a linear array. It was found that, a side lobe level (SLL) suppression improvement by about 6 dB for the non−uniform array (−20.13 dB) with respect to the uniform array (−13.89 dB). Conversely, since the focus of the technique was the suppression of SLL, half−power beamwidth (HPBW) generated a slight increment of 3% (about 0.18^{o}) in a non−uniform linear array.

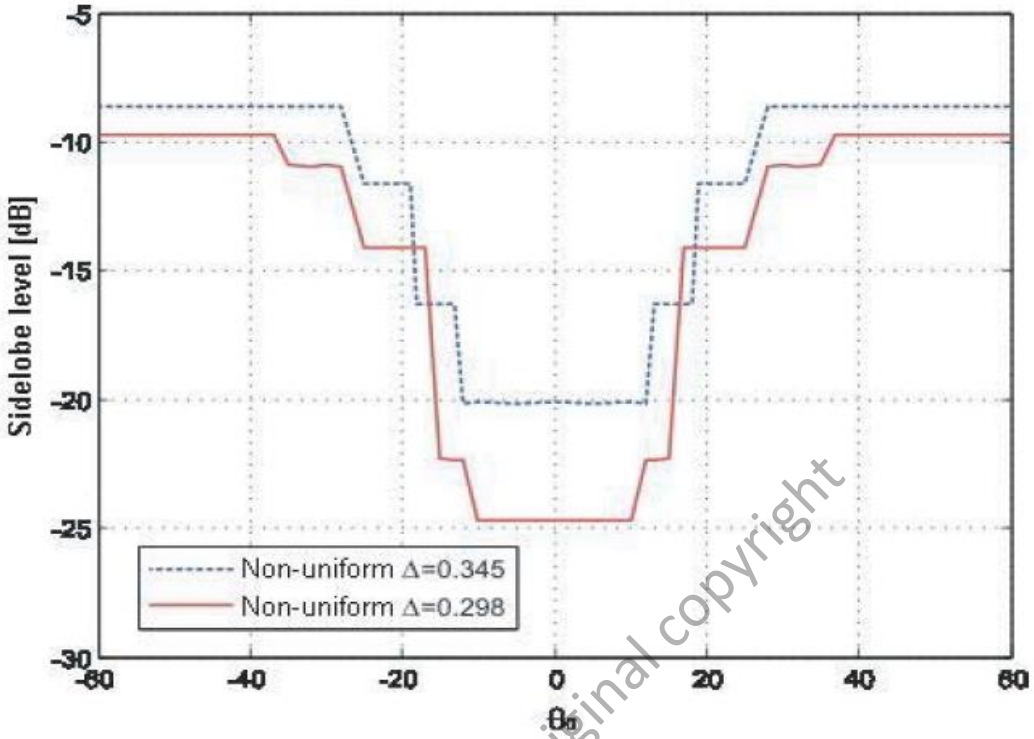


Figure 2.16: SLL when the main lobe steered in the range $-60° \leq \theta_0 \leq 60°$ for different space broadening factors, Δ (Gomez & Covarrubias, 2009).

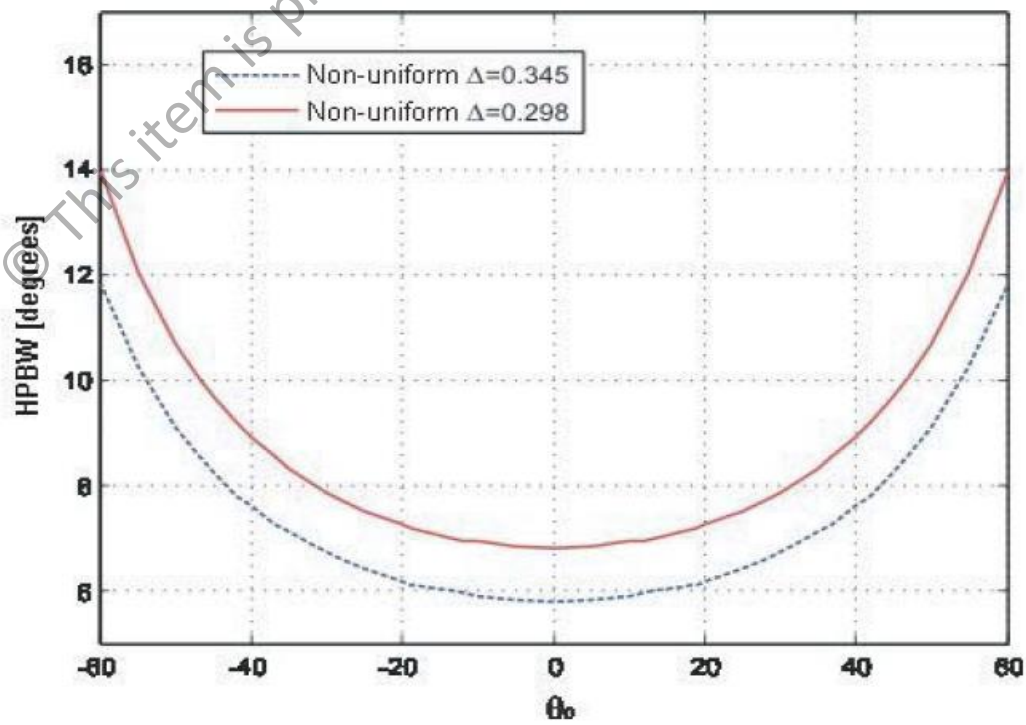


Figure 2.17: HPBW when the main lobe steered in the range $-60° \leq \theta_0 \leq 60°$for different space broadening factors, Δ (Gomez & Covarrubias, 2009).

Figure 2.16 depicts the SLL when the main lobe was steered in the range $-60° \leq \theta_0 \leq 60°$ using the non−uniform spacing. With the same procedure, the HPBW was also evaluated as shown in Figure 2.17. This numerical technique could change the SLL through modifying the space broadening factor, Δ. Both cases exhibited the performance of the Legendre functions with two different Δ values. In the range of $|\theta_0| \leq 12.29°$, there was a reduction in the SLL of 1.13 dB, and an increment of 0.16°, in HPBW for $\Delta = 0.345$. However, if the angular region was within $|\theta_0| \leq 15.80°$, there was a bigger reduction of 5.7 dB in SLL, but a higher increment of maximum 0.29° in HPBW for $\Delta = 0.298$.

Gomez and Covarrubias (2009) have concluded that these results were having a downside in comparison with the differential evolution (DE) algorithm (Rocha et al., 2007) due to the range of steering diminished 47.33% or 28.40°. On the contrary, the Legendre function technique is better than the DE in terms of having a narrower beamwidth without affecting the SLL, and a better computation time, respectively.

2.7 Evolutionary Computation or Evolutionary Algorithm Methods in Smart Antenna Design

Evolutionary computation (EC) is the field of research that draws ideas from evolutionary biology in order to develop search and optimization techniques for solving complex problems (De Castro, 2006). EC is also referred as evolutionary algorithm (EA). More than 40 years ago, computer scientists and engineers began developing various EA/EC methods to generate solutions to problems, which were too difficult and complicated to tackle with other classic or conservative analytical methods. EA/EC methods then rapidly have become major fields of machine learning and system optimization. More recently, EA/EC methods spread into the area of hardware design

including reconfigurable electronic circuits, computer−assisted manufacturing (CAM), and robotics (Floreano & Mattiussi, 2008).

EA/EC methods are examples of approximate algorithm basically try to combine basic heuristic methods in higher level frameworks aimed at efficiently and effectively exploring optimal solutions in a search space. These heuristic methods are nowadays commonly called "metaheuristics". The term metaheuristic, firstly introduced by Glover, F. (1986), derived from the composition of two Greek words. "Heuristic" comes from the verb "heuriskein", which means "to find", while the suffix "meta" means "beyond, in an upper level". Before this term was widely adopted, metaheuristics were often called modern heuristics (Reeves, 1993).

Blum, C. & Roli, A. (2003) have classified two crucial attributes in the modern heuristics or metaheuristics, which are intensification, and diversification. Precisely, intensification aims to search around the current best candidates or potential optimal solutions. Through this accumulated search experience, the algorithm selects the best optimal solutions. Meanwhile, diversification ensures the algorithm can explore the search space efficiently. The balance between intensification and diversification as mentioned above is important. Firstly, the aim is to quickly identify regions in the search space with high quality solutions and secondly, not to waste too much time in regions of the search space, which are either already explored or do not provide high quality solutions.

Nowadays, EA/EC methods, such as genetic algorithm or GA (Zhang, Li, Yuan & Yin, 2014; Laseetha & Sukanesh, 2011; Gómez, Melde, McNeill, & Rodriguez, 2006; Panduro, Brizuela, Balderas & Acosta, 2009), simulated annealing or SA (Lee, 2005), particle swarm optimization or PSO (Recioui, 2012; Ho, Liao & Chiu, 2010; Nik Abd Malik, Esa, Syed Yusof & Marimuthu, 2009; Khodier & Christodoulou, 2005;

Khodier & Al−Aqeel, 2009), tabu search or TS (Cengiz & Tokat, 2008), ant colony optimization or ACO (Karaboga, Guney & Akdagli, 2004), memetic algorithm or MA (Mandal, Zafar, Das & Vasilakos, 2012), artificial bee colony or ABC (Zaman, Md. Mushfiqul Alam, Mamun & Abdul Matin, 2011; Basu & Mahanti, 2011), firefly algorithm or FA (Zaman & Abdul Matin, 2012; Basu & Mahanti, 2011), invasive weed optimization or IWO (Pappula & Ghosh, 2014), and big bang crunch algorithm or BBCA (Sharma & Cecil, 2014) have been extensively studied and applied for antenna array beam design or geometry synthesis.

The reason for the use of these EA/EC methods or normally referred as metaheuristic algorithms, which imitate the best features in nature is mainly due to the high versatility, flexibility and capability to optimize complex multi−dimensional problems with a non−linear and non−convex dependence of design parameters (Panduro, Brizuela, Balderas & Acosta, 2009). Moreover, through the stochastic search process, the EA/EC methods have the ability in dealing with large number of optimization parameters, avoiding getting stuck in local minima, and relatively easy to simulate on computers (Khodier & Al−Aqeel, 2009). Besides, the wide deployment of the metaheuristic algorithms also lies in the computational drawbacks of existing numerical conventional methods. For examples, classical derivative−based optimization techniques are prone to getting trapped in local optima as well as strongly sensitive to initialization, and the gradient based methods fail to obtain significant solution for complex optimization problems. Due to these inherent weaknesses of the classical methods, modern metaheuristic algorithms are developed to achieve low SLL and/or null control from the designed arrays (Mandal, Zafar, Das & Vasilakos, 2012).

2.8 Genetic Algorithm

Now let us discuss briefly, all the EA/EC techniques, which will be compared and manipulated (hybridized) in this study. Firstly, we start with the GA technique. The GA is an EA/EC technique inspired by evolutionary biology phenomenon, which has been conceived since the end of the 1950s (Dreo, Petrowski, Siarry & Taillard, 2006). Precisely, GA simulates the natural selection and survival of the fittest concept among individuals over consecutive generation for solving a problem. Each generation consists of a population of bit strings, which are analogous to the *deoxyribonucleic acid* (DNA) genome or chromosome. Each individual represents a point in a search space and a possible solution.

The individuals in the population then will go through a process of genetic evolution including inheritance, mutation, selection, and crossover (also called recombination). The new population is then used in the next generation for future reproduction. Commonly, the algorithm terminates when a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the whole population.

At first the initial population is generated. A fitness function is then used to evaluate the solution performance. The aim of the genetic operators is to get a maximum or minimum fitness value. The reproduction operator performs a natural selection function mechanism, such as fitness proportional selection or Roulette wheel selection. Individuals are copied from one set to the next according to their fitness value. The individuals that give better results will be reselected for the next generation (Cengiz & Tokat, 2008).

Precisely, in Roulette wheel selection method, the probability of selecting a chromosome (individual of the population) is directly proportional to its fitness value. Each chromosome $\mathbf{x}_i, i = 1, \ldots, N$, of the population is assigned to a part of the wheel whose size is proportional to its fitness value. The wheel is then tossed as many times as parents (*N*) are needed to create the next generation, and each winning individual is selected for reproduction. This selection method allows an individual to be selected more than once and the deletion of some other individuals as well (De Castro, 2006).

The crossover operator chooses pairs of individuals at random and produces new pairs. Crossover is the primary operator that increases the exploratory power of GA. In order to successfully achieve the cross−fertilizing type of innovation, crossover operator must ideally inter−mix good sub solutions without any disruption of the partitions (Dreo, Petrowski, Siarry & Taillard, 2006).

The number of crossover operations is governed by a probability known as the crossover rate, p_c. Crossover is applied to the parents with a high p_c, which has the typical value between 0.80 and 0.95. The simplest single−point crossover operation is to cut the original parents at a randomly selected point and exchange their tails. Precisely, there are three steps of crossover process (De Castro, 2006):

i. Two strings $\mathbf{x} = x_1 x_2 x_3 \ldots x_l$ and $\mathbf{y} = y_1 y_2 y_3 \ldots y_l$ are selected from the current population **P**.

ii. A number *r* indicating the crossover point is randomly selected from {1,2,…,*l*−1}.

iii. Two new strings are formed from **x** and **y** by exchanging the set of attributes to the right of position *r*, yielding $\mathbf{x}' = x_1 \ldots x_i y_{i+1} \ldots y_l$ and $\mathbf{y}' = y_1 \ldots y_i x_{i+1} \ldots x_l$ where the two new chromosomes (strings), $\mathbf{x}'$and $\mathbf{y}'$, are offspring of **x** and **y.**

The mutation operator randomly mutates or reverses the values of bits in an individual. The number of mutation operations is determined by a probability known as the mutation rate, p_m. The process of flipping the bits of offspring is done with a small p_m, which is between 0.100 and 0.001.The mutation of each bit string in the population P (with chromosomes represented by binary numbers) is operated as follows (De Castro, 2006):

i. The numbers $r,\ldots,u$ indicating the positions to undergo mutation are determined by a random process where each position has a small probability p_m of undergoing mutation, independently of the other positions.

ii. A new string $\mathbf{x}' = x_1 \ldots x_r \ldots x_u \ldots x_l$ is generated where $x_r \ldots x_u$ are drawn at random from the set of alleles for each chromosome where in the case of bit strings, if a position has an allele '0', then it becomes '1', else if it is originally '1', then it becomes '0'.

2.9 Genetic Algorithm in Antenna Array Synthesis

Goswami and Mandal (2012), analyzed and applied a real−coded genetic algorithm (RGA) to determine the optimum element current excitation weights, and inter−element spacing. This should impose deeper nulls in the interference direction of uniform linear antenna arrays under the constraints of a reduced side lobe level (SLL) and a fixed first null beamwidth (FNBW).

Considering a broadside linear antenna array of 2M isotropic radiators, each element is excited with a non−uniform current. The array elements are assumed to be uncoupled and equally spaced along the z−axis, and the center of the array is located at

the origin. Assume that the array is symmetric in both geometry and excitation with respect to the center.

In this study, the array factor (AF) in the *xy*–plane with symmetric amplitude distributions is stated as:

$$AF(I,\varphi,d) = 2\sum_{n=1}^{M} I_n \cos\left[\left(\frac{2n-1}{2}\right) kd\cos(\theta) + \varphi_n\right] \tag{2.30}$$

where θ denotes the zenith angle measured from the broadside direction of the array, I_n is the current excitation amplitude, φ_n is the excitation phase, d is the spacing between two consecutive elements, and $k = 2\pi/\lambda$ is the wave number. In this experiment, φ_n is fixed at zero while the array elements are numbered from 1 to M from the origin in a symmetric array with the total number of elements equivalent to 2M.

Goswami and Mandal (2012) included the objective function to be minimized with the RGA to obtain the low nulls and SLL as:

$$f = C_1 \times \frac{\left|\prod_{i=1}^{m} AF(null_i)\right|}{|AF_{max}|} + C_2 \times \sum_{k=1}^{K} H(k) \times (Q_k - \delta) + C_3 \times \left(FNBW_{computed} - FNBW(I_n = 1)\right) \tag{2.31}$$

where m is the maximum number of positions wherever the null can be imposed with the value of either one or two. AF(null$_i$) is the value of the AF at the particular null position, and AF$_{max}$ is the maximum value of the AF, respectively.

The second term in (2.31) is summed to reduce the SLL to a desired level. Besides, K denotes the number of side lobes in the original pattern, Q_k is the SLL in dB generated by the individual population at some peak point θ_k, and δ is the desired value of the SLL in db. Furthermore, the $H(k)$ is defined as:

$$H(k) = \begin{cases} 1, (Q_k - \delta) > 0 \\ 0, (Q_k - \delta) \leq 0 \end{cases}$$

(2.32)

In this case, the side lobes whose peaks exceed the threshold, δ must be suppressed, so that $H(k)$ can be adopted in the objective function. The first null beamwidth (FNBW) denotes the angular width between the first nulls on either side of the main beam. The third term in (2.31) is introduced to keep FNBW of the optimized pattern the same as in the initial pattern with $I_n = 1$, and $d = \lambda/2$, respectively. The two beamwidths $\text{FNBW}_{\text{computed}}$, and FNBW($I_n = 1$) refer to the computed first null beamwidth in radian for the non–uniform excitation for the optimal spacing case and for the uniform excitation ($I_n = 1$) with a uniform inter–element spacing ($d = \lambda/2$).

The actual value of FNBW for a uniform linear array can be calculated by:

$$\theta_n = 2\lambda / Nd$$

(2.33)

where $N = 2M$ is the total number of elements in the array. C_1, C_2, and C_3 are weighting coefficients to control the relative importance of each term of (2.31). The value of C_1 is set to the highest one due to the primary aim is to achieve a deeper null. A smaller value of the objective function meant that the AF values at predefined positions are lower. Consequently, the RGA controls the amplitude excitations and the inter–element spacing to minimize the objective function. Precisely, the RGA applies floating–point number representations for the real variables, which is free of binary encoding and decoding. Hence, the RGA is faster than the binary–coded GA (BGA).

In this study, the chromosomes correspond to the current excitation weights and the inter–element spacing of the antenna elements. Because of symmetry, each chromosome consists of $M + 1$ number of genes, where M is the number of antenna elements on either side of the array center. Here, the 1st to Mth genes represent the

current excitation weights of the antenna elements, and the $(M + 1)^{th}$ gene represents the inter–element spacing. For example, chromosome one, $\overline{W_1}$ can be represented by:

$$\overline{W_1} = [W_{11}, W_{12}, \dots, W_{1M}, W_{1(M+1)}]$$

(2.34)

where $W_{11}, W_{12}, \dots, W_{1M}$ are the antenna element weights or genes, and, $W_{1(M+1)}$ is the inter–element spacing. Each of these current excitation weights and the inter–element spacing has both the upper and lower limits.

The random set of chromosomes can be generated using the following relation represented by:

$$\overline{W_n} = (u_1 - u_2) \times \bar{r} + u_2, u_2 < \overline{W_n} \ll u_1$$

(2.35)

where u_1 and u_2 are the maximum and minimum limit values of the weights or genes, respectively, and $\bar{r}$ is a real random vector between zero and one. All of the current excitation weights or genes are restricted to be between 0 and 1, and the inter–element spacing or gene is restricted to lie between $\lambda/2$ and λ.

Linear antenna arrays composed of 12 isotropic radiating elements, with an inter–element spacing of $\lambda / 2$, is considered. The RGA is executed to obtain deeper nulls and to reduce the SLL through 500 iterations with the population size is fixed at 120, the mutation probability is set to 0.05, and the uniform crossover is used. The RGA algorithm is initialized using random values of the excitation ($0 < I_n < 1$) and the spacing between the elements ($\lambda / 2 \leq d < \lambda$). The nulling performances are improved for predefined nulls of the radiation pattern. Besides, nulls are also imposed at predefined peak positions.

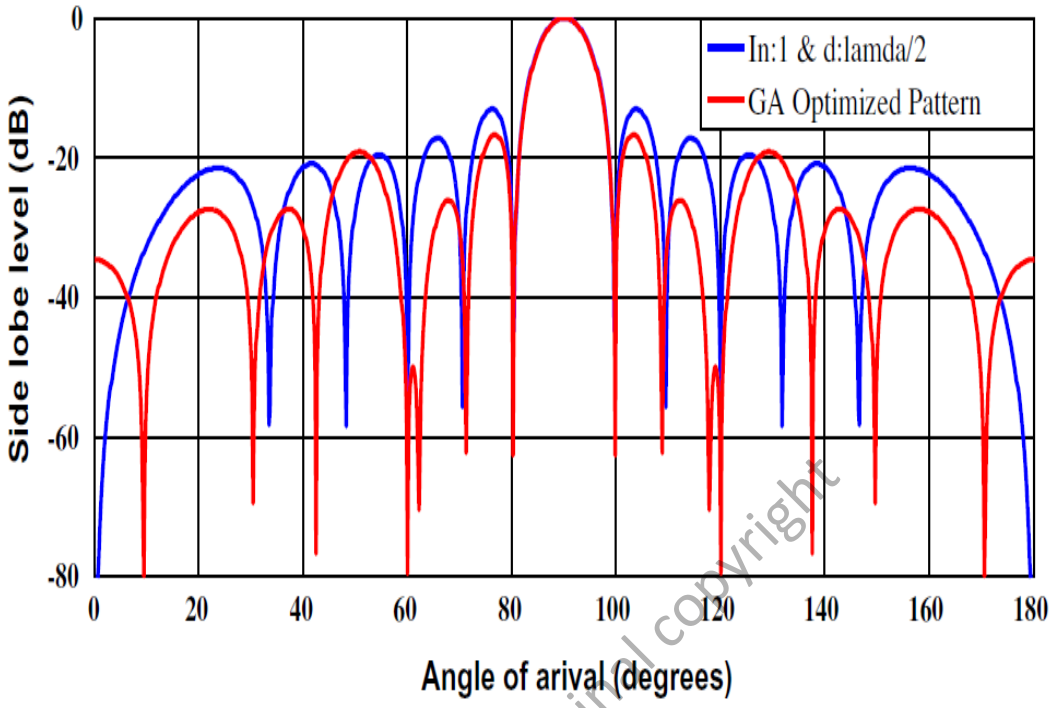


Figure 2.18: Best array pattern found by RGA for the 12–element array case with an improved null; i.e., h = 60° and h = 120° (Goswami and Mandal, 2012).

Based on Figure 2.18, for a uniform excitation (I_n = 1) of 12–elements linear array sets with an inter–element spacing of $\lambda / 2$, the SLL is –13.06 dB, and FNBW is 19.10°, respectively. The optimization I_n and d of the 12–element arrays significantly improve or deepen the nulls from –51.90 dB to –79.54 db. Lastly, the normalized d values with respect to the $\lambda / 2$ generated by the RGA are 0.84511, 0.6556, 0.8444, 0.71671, 0.47139, 0.40992, and 1.1601.

2.10 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population–based stochastic approach for solving continuous and discrete optimization problems. In PSO, simple software agents, called particles, move in the search space of an optimization problem. The position of a particle represents a candidate solution to the optimization problem at

hand. Each particle searches for better positions in the search space by changing its velocity according to rules originally inspired by behavioural models of bird flocking. PSO belongs to the class of swarm intelligence (SI) technique that are used to solve optimization problems ("Particle Swarm Optimization", 2011).

PSO was introduced by Kennedy and Eberhart in 1995 (Kennedy & Eberhart, 1995). It has roots in the simulation of social behaviours using tools and ideas taken from computer graphics and social psychology research. Since its introduction, the PSO has gained an increasing popularity as an efficient alternative to GA and simulated annealing (SA) in solving optimization design problems including antenna arrays (Khodier & Christodoulou, 2005).

In (Eberhart & Shi, 2001), initial simulations are modified to incorporate nearest−neighbour velocity matching, eliminate auxiliary variables, and incorporate multi−dimensional search and acceleration by distance. At some point in the evolution of the algorithm, it is realized that the conceptual model is, in fact, an optimizer. Through a process of trial and error, a number of parameters extraneous to optimization are eliminated from the algorithm, resulting in the very simple original implementation (Eberhart, Simpson & Dobbins, 1996).

Similarly to GA, the PSO optimizer is initialized with a population of random solutions. Nonetheless, each PSO potential solution is also assigned a randomized velocity, and the potential solutions (particles) are then "flown" through the problem space or domain.

In this case, each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far in which the fitness value is also stored. This value is called personal best or *pbest*. Another "best" value that is tracked by the global version of the particle swarm optimizer is the overall

best value, and its location obtained so far by any particle in the population. The best value is called global best or *gbest*.

The PSO concept changes the velocity of each particle toward its *pbest* and *gbest* locations at each time step. Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *gbest* locations. There is also a local version of PSO in which, in addition to *pbest*, each particle keeps track of the best solution, called local best or *lbest* attained within a local topological neighbourhood of particles. The standard process of the global version of PSO is as follows (Eberhart & Shi, 2001):

i. Initialize a population (array) of particles with random positions and velocities in the *d*−dimensional problem space.

ii. For each particle, evaluate the desired optimization fitness function in the *d* search space.

iii. Compare particle's fitness evaluation with the particle's *pbest* where if current value is better than the *pbest*, then set the *pbest* value equal to the current value, and the *pbest* location equal to the current location in the *d* search space.

iv. Compare fitness evaluation with the population's overall previous best where if current value is better than the *gbest*, then reset the *gbest* to the current particle's array index and value.

v. Change the velocity and position of the particle according to equations:

$$v_{id} = v_{id} + c_1 \times \text{rand}() \times (p_{id} - x_{id}) + c_2 \times \text{rand}() \times \left(p_{gd} - x_{id}\right) \quad (2.36)$$

$$x_{id} = x_{id} + v_{id} \quad (2.37)$$

vi. Loop to the step (ii) until the criterion is met, usually the sufficiently good fitness or the maximum number of iterations.

Particles' velocities on each dimension are braced to the maximum velocity or V_{max}. If the sum of accelerations would cause the velocity on that dimension to exceed the V_{max}, which is a parameter specified by the user, then the velocity on that dimension is limited to the V_{max}. Thus, the V_{max} is important because it determines the resolution or refinement with which regions between the present position and the target (best so far) position are searched. If the V_{max} is too high, particles might fly past good solutions. However, if the V_{max} is too small, particles may not explore sufficiently beyond locally good regions. In fact, they could become trapped in local optima, unable to move far enough to reach a better position in the problem space (Eberhart & Shi, 2001).

The acceleration constants c_1 and c_2 in equation (2.36) represent the weighting of the stochastic acceleration terms that pull each particle toward both the *pbest* and *gbest* positions. Thus, adjustment of these constants changes the amount of "tension" in the system. Low values allow particles to roam far from target regions before being tugged back, while high values result in abrupt movement toward or fly past target regions (Eberhart & Shi, 2001).

The population size selected is problem−dependent. Population sizes of 20−50 are probably most common. The small population are optimal for PSO in terms of minimizing the total number of evaluations (population size times the number of generations) needed to obtain a sufficient solution (Eberhart & Shi, 2001).

The concept of an inertia weight is developed to better control exploration and exploitation. The motivation is to eliminate the need for the V_{max}. Equations (2.36) and (2.37) describe the velocity and position update equations with an inertia weight, w included. The use of the inertia weight, w has successfully improved performance in a

number of applications. Precisely, *w* frequently decreases linearly from about 0.9 to 0.4 during execution. Suitable selection of the *w* provides a balance between global and local exploration and exploitation, and requires less number of iteration to find a sufficient optimal solution (Eberhart & Shi, 2001).

$$v_{id} = w \times v_{id} + c_1 \times \text{rand}( ) \times (p_{id} - x_{id}) + c_2 \times \text{rand}( ) \times \left(p_{gd} - x_{id}\right) \tag{2.38}$$

$$x_{id} = x_{id} + v_{id} \tag{2.39}$$

It has been demonstrated that the PSO algorithm can be also successfully applied to tracking and optimizing dynamic systems. A slight adjustment is made to the *w* for this purpose. The *w* in equation (2.38) is set equal to [0.5 + rand/2.0]. This randomly produces a number varying between 0.50 and 1.00, with the mean of 0.75 (Eberhart & Shi, 2001).

2.11 Particle Swarm Optimization in Antenna Array Synthesis

Goudos et al. (2010) have postulated a design technique based on comprehensive learning particle swarm optimization (CLPSO) algorithm to establish unequally linear array synthesis with SLL suppression under constraints to beamwidth and null control. The CLPSO algorithm applies a new learning strategy to accelerate the convergence of classical PSO. In this array synthesis, the CLPSO algorithm is used to find the optimum element spacing between the antenna array elements.

In the proposed CLPSO algorithm, each particle's velocity vector is updated by using not only its own *pbest* but also any other particle's *pbest*, which improves the diversity in the population. The velocity update equation in the CLPSO algorithm is:

$$V_{in}^{t} = \omega . V_{in}^{t-1} + c . rand_{in}^{t} \left(pbest_{fi(n)n}^{t} - x_{in}^{t-1}\right)$$

(2.40)

where $f_i = [f_i(1), f_i(2),\ldots, f_i(n),\ldots, f_i(D)]$ states which particle's *pbest* that the particle, *i* should follow, while $pbest^{t}_{fi(n)n}$ is the corresponding dimension of any particle's *pbest* including its own *pbest*. The decision on which *pbest* to be followed depends on the learning probability, P_c where it can take different values for different particles. In this case, a random number is generated for each dimension to achieve the diversity. The comparison of this number with the P_c decides whether the corresponding dimension will learn from its own *pbest* or from other particle's *pbest*. The particle will learn from its own *pbest* if the random number is larger than the P_c. Otherwise, it would learn from another particle's *pbest*. In case a particle learns from another particle's *pbest*, the following tournament selection procedure is used (Goudos et al., 2010):

a. Two particles are chosen out of the population using a uniform random distribution, excluding the particle whose velocity is updated.
b. The fitness values of these two particles' *pbest* are compared, and the best one is selected.
c. The particle's *pbest* selected in the previous step will be used as the exemplar to be learned from for that dimension where if all exemplars of a particle are its own *pbest*, then randomly one dimension is chosen, so as to learn from another particle's *pbest* for that dimension.

The updating strategy used above will let the CLPSO algorithm to do more exploration in the search space than the original PSO, hence, will increase the diversity. In other words, the CLPSO algorithm can exploit a wider search domain in finding the global optimum.

However, the CLPSO algorithm is found more complex than the two most common PSO algorithms, which are the classical inertia weight PSO (IWPSO) and

constriction factor PSO (CFPSO), respectively (Clerc, 1999). Consequently, the computation load of CLPSO is found slightly higher than the two counterparts.

The synthesis is done on the $2N-$element linear array symmetrically placed along the $x-$axis. The AF used in this study is stated as:

$$AF(\theta) = 2\sum_{n=1}^{N} I_n e^{j\left(\frac{2\pi}{\lambda}x_n \sin\theta + \phi_n\right)} \tag{2.41}$$

where λ is the wavelength, whereas I_n, x_n, ϕ_n are the excitation amplitude, position, and phase of the nth element, respectively. The primary optimization goal is the SLL suppression, while setting the main lobe to a desired beamwidth within ±1° by finding the optimum element positions through minimizing the objective function below:

$$F(\bar{x}) = \max_{\theta \epsilon S}\left\{AF_{dB}^{\bar{x}}(\theta)\right\} + \Xi.\max\{0, |BW_c - BW_d| - 1\} \tag{2.42}$$

where $\bar{x}$ is the vector of the element position, S is the space spanned by the angle, θ excluding the main lobe, BW_c is the calculated beamwidth, BW_d is the desired beamwidth, and Ξ is a very large number.

Through this optimization approach, the feasible region is expanded, but a large cost or "penalty" is added to the original objective function for solutions that lie outside of the original feasible region. Therefore, Ξ is chosen large enough to ensure that solutions can fulfill constraints in case of large fitness values.

Furthermore, in order to perform the SLL suppression whenever there are predefined nulls, the below objective function is expressed:

$$F_N(\bar{x}) = F(\bar{x}) + \Xi.\left[\sum_{k=1}^{K} \max\left\{0, AF_{dB}^{\bar{x}}(\theta_k) - C_{dB}\right\}\right] \tag{2.43}$$

where K is the number of the predefined null, C_{dB} is the desired null level in dB, and θ_k is the direction of the kth null.

Goudos et al. (2010) simulated all the tested algorithms with the population size set to 40 and number of generations fixed to 1000. For CLPSO, c in (2.40) was set to 1.0 whereas Ξ in (2.43) was set to 10^6. The simulation was done on a 10−element linear array with the desired beamwidth was set to 23° with the tolerance of ±5%.

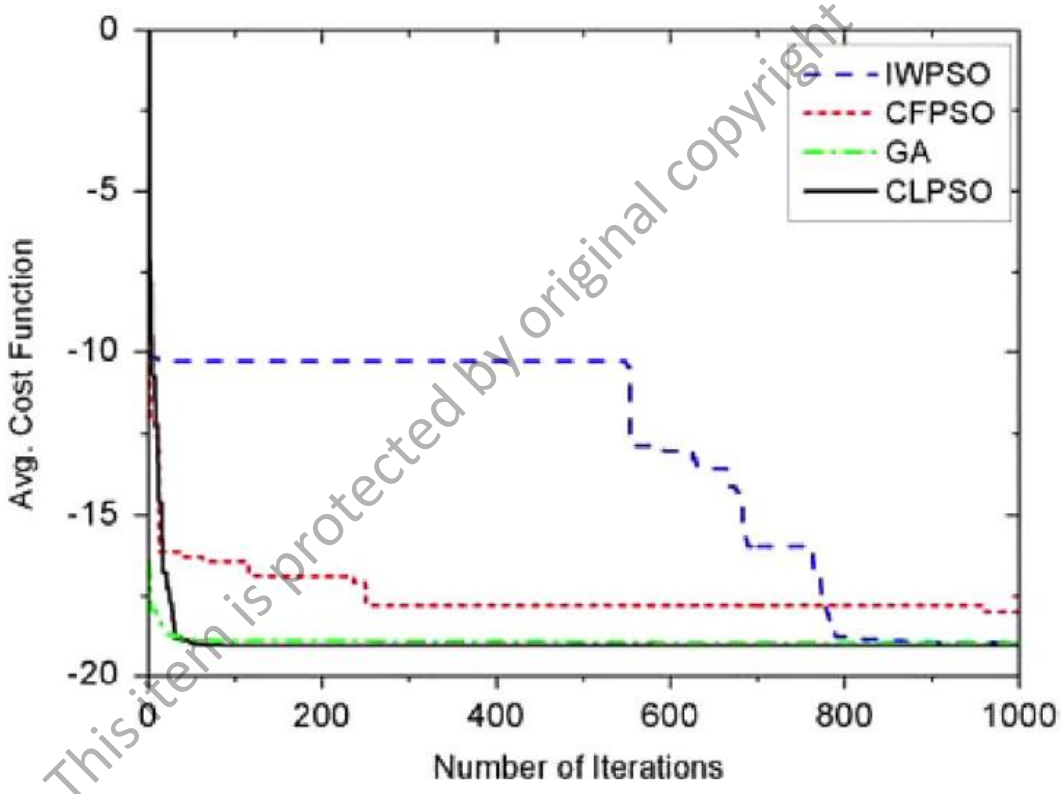


Figure 2.19: Convergence rate plot for the 10−element array case (Goudos et al., 2010).

Based on Figure 2.19, the proposed CLPSO algorithm clearly converged faster than the other two PSO algorithms where in this case, the CLPSO algorithm required less than 100 iterations to converge to its final value. Even so, the real−coded GA was found converged a little bit faster than the CLPSO algorithm.

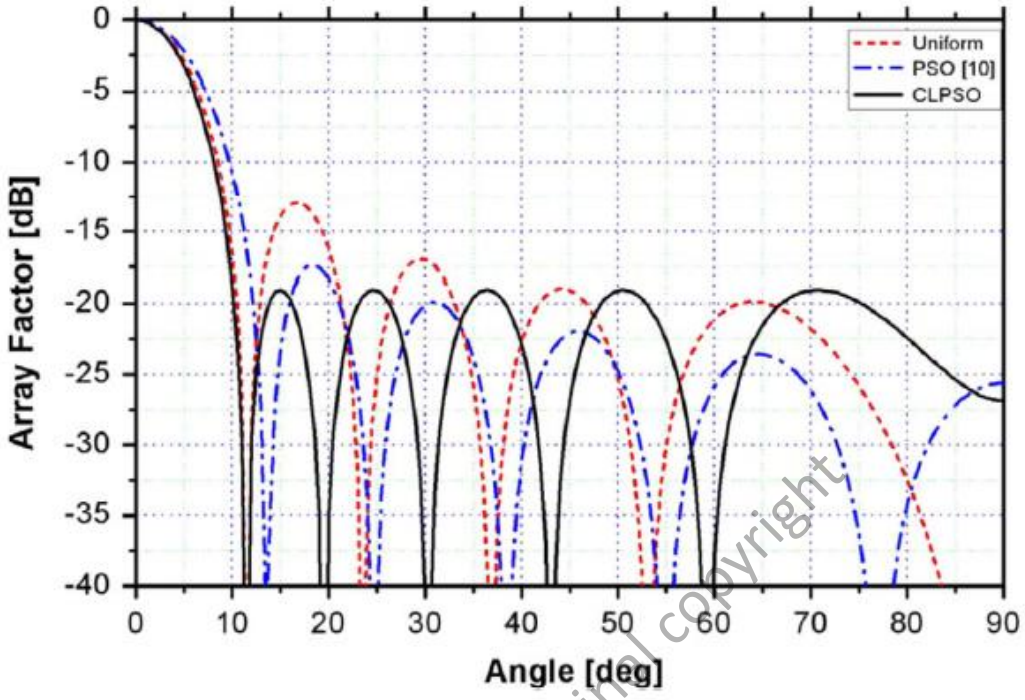


Figure 2.20: Array pattern for the 10–element array case with SLL suppression and desired beamwidth at 23° (Goudos et al., 2010).

In addition, according to Figure 2.20, the postulated CLPSO optimizer achieved a lower SLL of −19.07 dB, while in (Khodier & Christodoulou, 2005), the SLL found was −17.44 dB. In this case, the beamwidth found was close to the uniform array. In fact, the normalized symmetric $2N$−element positions with respect to $\lambda/2$ generated by the CLPSO algorithm were 0.443, 1.422, 2.416, 3.670, and 5.117, respectively.

Finally, Goudos et al. (2010) prove that CLPSO outperforms both the common PSO algorithms and real−coded GA in terms of deeper average SLL suppression. In sum, the CLPSO algorithm can be as an alternative EA/EC method for solving complex multimodal electromagnetic optimization problem using the proposed updating strategy, which leads to a bigger potential search space.

2.12 Summary of Optimization Methods in Array Geometry Synthesis

As discussed earlier, there are various numerical, analytical, and EA/EC metaheuristic methods have been deployed by researchers and engineers in designing adaptive smart antenna. Some of the recent publications regarding the optimization methods are shown in the table below:

Table 2.1: Optimization Methods for Antenna Array Synthesis

Author	***Optimization Method***
Keizer (2009)	Linear antenna array synthesis using iterative Fourier analytical technique.
Alexopoulus (2006)	Phased array design using conventional Dolph−Chebyshev side lobe tapering analytical technique.
Abreau & Kohno (2002)	Uniform linear array and uniform circular array designs with low Dolph−Chebyshev tapering analytical technique.
Gomez & Covarrubias (2009)	Multidimensional array geometries nonlinear optimization using a unified Legendre functions numerical method.
Zhang et al. (2014) Walia et al. (2013) Laseetha & Sukanesh, (2011) Goswami & Mandal (2012) Panduro et al. (2009) Sattari & Hejazi (2008) Gómez et al. (2006)	Array geometry synthesis using genetic algorithm method.
Walia et al. (2013) Lee (2005)	Array geometry synthesis using simulated annealing method.
Recioui (2012) Ho et al. (2010) Goudos et al. (2010) Nik Abd Malik et al. (2009) Khodier & Al−Aqeel (2009) Khodier & Christodoulou (2005)	Array geometry synthesis using particle swarm optimization method.
Cengiz & Tokat (2008)	Array geometry synthesis using tabu search method.
Karaboga et al. (2004)	Array geometry synthesis using ant colony optimization method.
Mandal et al. (2012)	Array geometry synthesis using memetic algorithm method.
Rocha et al. (2007)	Array geometry synthesis using differential evolution algorithms method.
Pappula & Ghosh	Array geometry synthesis using invasive weed

(2014) Pal et al. (2009)	optimization method.
Singh et al. (2010)	Array geometry synthesis using biogeography based optimization method.
Zaman et al. (2012) Basu et al. (2011)	Array geometry synthesis using firefly algorithm method.
Walia et al. (2013) Zaman et al. (2011) Basu et al. (2011)	Array geometry synthesis using artificial bee colony algorithm method.
Sharma & Cecil (2014)	Array geometry synthesis using big bang crunch algorithm method.

2.13 Hybrid Optimization Algorithm

Wolpert and Macready (1997) have claimed that all algorithms that search for a maximum or minimum of a cost function perform exactly the same, when averaged over all possible cost functions. According to the authors, if algorithm *A* outperforms algorithm *B* on some cost functions, then loosely speaking there must exist exactly as many other functions where *B* outperforms *A*.

In other words, Wolpert and Macready (1997) have pointed out that from a problem solving perspective it is difficult to formulate a universal optimization algorithm that could solve all the problems. In this case, hybridization may be the key to solve practical problems.

2.14 Multiobjective Optimization: Weighted−Sum and Pareto Front Optimum

Multiobjective (MO) optimization problems are common (Zitzler, 1999). In addition to that, the majority of engineering design problems can also be categorized as MO problems, which requires optimizing multiple conflicting. For an example, designing a part made of composite materials needs simultaneous optimization of the structural performance and the manufacturing cost or time. These objectives are often

conflicting and strongly coupled and thus, the corresponding MO problem does not have a single optimum solution, but a set of solutions, called “Pareto optimum”.

In this case, the maximum structural performance is desired while the manufacturing cost or time of such composite materials design is to be minimized. These Pareto optimum solutions represent the trade−off among multiple objective functions, e.g. structural performance and manufacturing cost or time functions, respectively (Ghiasi, Pasini & Lessard, 2011).

Zitzler (1999) defines a general MO problem consists of a set of *n* parameters (decision variables), a set of *k* objective functions, and a set of *m* constraints. Objective functions and constraints are functions of the decision variables. Generally, the optimization goal is to:

maximize or minimize $y = f(x) = (f_1(x), f_2(x),\ldots, f_k(x))$

subject to $e(x) = (e_1(x), e_2(x),\ldots, e_m(x)) \leq \text{or } \geq 0$

where $x = (x_1, x_2,\ldots, x_n) \in X$

$y = (y_1, y_2,\ldots, y_k) \in Y$

(2.44)

and *x* is the decision vector, *y* is the objective vector, *X* is denoted as the decision space, and *Y* is called the objective space. The constraints $e(x) \leq \text{or} \geq 0$ determine the set of feasible solutions.

The feasible set X_f is defined as the set of decision variables *x* that satisfy the constraint:

$e(x)$: $X_f = \{x \in X | e(x) \leq \text{or} \geq 0\}$

(2.45)

The image of X_f, e.g. the feasible region in the objective space is denoted as:

$$Y_f = f(X_f) = \cup_{x \in X_f} \{f(x)\}$$

(2.46)

Assume that there are two objectives (structural) performance (f_1) and (inverse of manufacturing) cost (f_2), which will be maximized under size constraints (e_1). Then, an optimal design might be an architecture, which achieves maximum performance at minimal cost and does not violate the size limitations.

However, what makes MO problems difficult is the common situation when the individual optima corresponding to the distinct objective functions are sufficiently different. Then, the objectives are conflicting and cannot be optimized simultaneously. Instead, a satisfactory "trade−off" has to be found. In the abovementioned MO problem example, (structural) performance and (inverse of manufacturing) cost are generally competing where high−performance architectures substantially increase the cost, while inexpensive architectures usually provide a low performance. Depending on the market requirements, an intermediate solution (medium performance, medium cost) might be an appropriate trade−off. To sum up, there is a requirement for a new notion of optimality in MO problems (Zitzler, 1999).

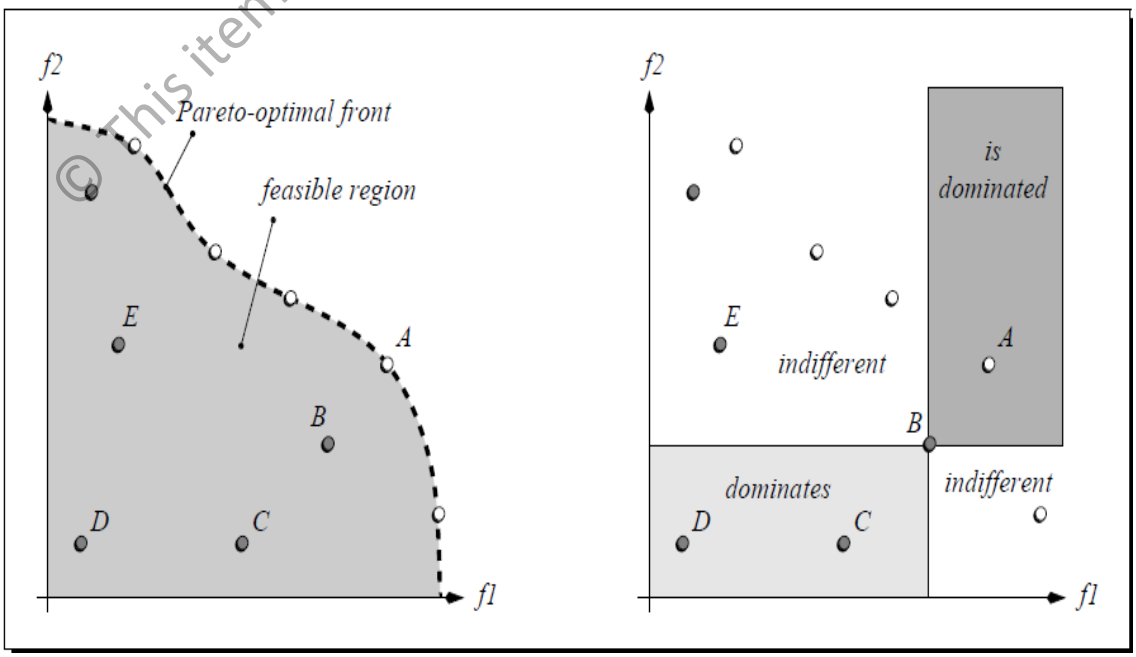


Figure 2.21: Left: Pareto optimality in objective space, and Right: Possible relations of solutions in objective space (Zitzler, 1999).

In the stated MO maximization problem, the feasible set is arranged in a particular order entirely based on two objective functionsf_1 and f_2 for two solutions $a, b \in X_f$, either $(f_1(a) > f_1(b)) \,||\, (f_2(a) > f_2(b))$ or $(f_1(b) > f_1(a)) \,||\, (f_2(b) > f_2(a))$. The goal is to find the solution (or solutions), which provides the maximum value of f_1 and f_2. Nevertheless, since the optimization involves with two objectives, X_f is arranged partially in order as depicted in Figure 2.21 on the left side.

It is preferably to have both objectives improved (maximized) as in the case for *B*and *C*. In this case, the solution represented by point *B* is better than the solution represented by point *C* where it provides a higher performance at a lower cost. There is also a possibility where one of two objectives is improved (maximized) as in the case for *C* and *D*. Despite equal cost, *C* achieves better performance than *D*. In order to express this situation mathematically, the relations =, ≥, and > are extended to objective vectors by analogy to the MO case.

In defining a Pareto maximization dominance, let us assume for any two decision vectors $\boldsymbol{a}$ and $\boldsymbol{b}$,

$\boldsymbol{a} \succ \boldsymbol{b}$ ($\boldsymbol{a}$ dominates $\boldsymbol{b}$) iff $(f_1(\boldsymbol{a}) > f_1(\boldsymbol{b}))$ $||$ $(f_2(\boldsymbol{a}) > f_2(\boldsymbol{b}))$

$\boldsymbol{a} \succcurlyeq \boldsymbol{b}$ ($\boldsymbol{a}$ weakly dominates $\boldsymbol{b}$) iff $(f_1(\boldsymbol{a}) \geq f_1(\boldsymbol{b}))$ $||$ $(f_2(\boldsymbol{a}) \geq f_2(\boldsymbol{b}))$

$\boldsymbol{a} \sim \boldsymbol{b}$ ($\boldsymbol{a}$ is indifferent to $\boldsymbol{b}$) iff $(f_1(\boldsymbol{a}) = f_1(\boldsymbol{b}))$ && $(f_2(\boldsymbol{a}) = f_2(\boldsymbol{b}))$

(2.47)

On the other hand, the definitions for Pareto minimization dominance $(\prec, \preccurlyeq, \sim)$ are analogical. In Figure 2.21 on the right, the light gray rectangle encapsulates the region in objective space that is dominated by the decision vector represented by *B*. The dark gray rectangle contains the objective vectors whose corresponding decision vectors

dominate the solution associated with *B*. All solutions for which the resulting objective vector is in neither rectangle are indifferent to the solution represented by *B*.

Based on the concept of "Pareto dominance", the optimality criterion can be introduced for MO problems. In this case, *A* is unique among *B*, *C*, *D*, and *E* where its corresponding decision vector $\boldsymbol{a}$ is not dominated by any other decision vector. That means, $\boldsymbol{a}$ is optimal in the sense that it cannot be improved in any objective without causing a degradation in at least one other objective. Such solutions are denoted as "Pareto optimal" or "Pareto front" solutions (Zitzler, 1999).

The most widely used approach for MO optimization is the weighted−sum method. The method transforms multiple objectives into an aggregated objective function by multiplying each objective function with a weighting factor and summing up all weighted objective functions (Kim & De Weck, 2004):

$$f_{(\text{weighted}-\text{sum})} = w_1.f_1 + w_2.f_2 + \cdots + w_m.f_m \tag{2.48}$$

where w_i ($i = 1,\ldots,m$) is a weighting factor for the ith objective function. The weighting factor is generated through dividing each objective by a scaling factor. The weighted−sum is classified to be a convex combination of objectives if $\sum_{i=1}^{m} w_i = 1$ and $0 \leq w_i \leq 1$. Under certain conditions, the solution to $f_{(\text{weighted}-\text{sum})}$ optimization is a Pareto optimal point, and by appropriately changing the weight vector w_i ($i = 1,\ldots,m$) one can approximate the "Pareto front" (Ryu, Kim & Wan, 2009).

In this case, each SO optimization determines one particular optimal solution point on the Pareto front. The weighted−sum method then changes weights systemically, and each different SO optimization determines a different optimal solution. The solutions obtained approximate the Pareto front (Kim & De Weck, 2004).

Studies have found some of the major drawbacks of the standard weighted−sum method, such as the optimal solutions distribution is not uniform, optimal solutions in non−convex regions are not detected, and possibility of solutions duplication with different weight combinations (Ryu, Kim & Wan, 2009). The conventional weighted−sum method is unable to generate the non−convex part of the Pareto front is shown in Figure 2.22.

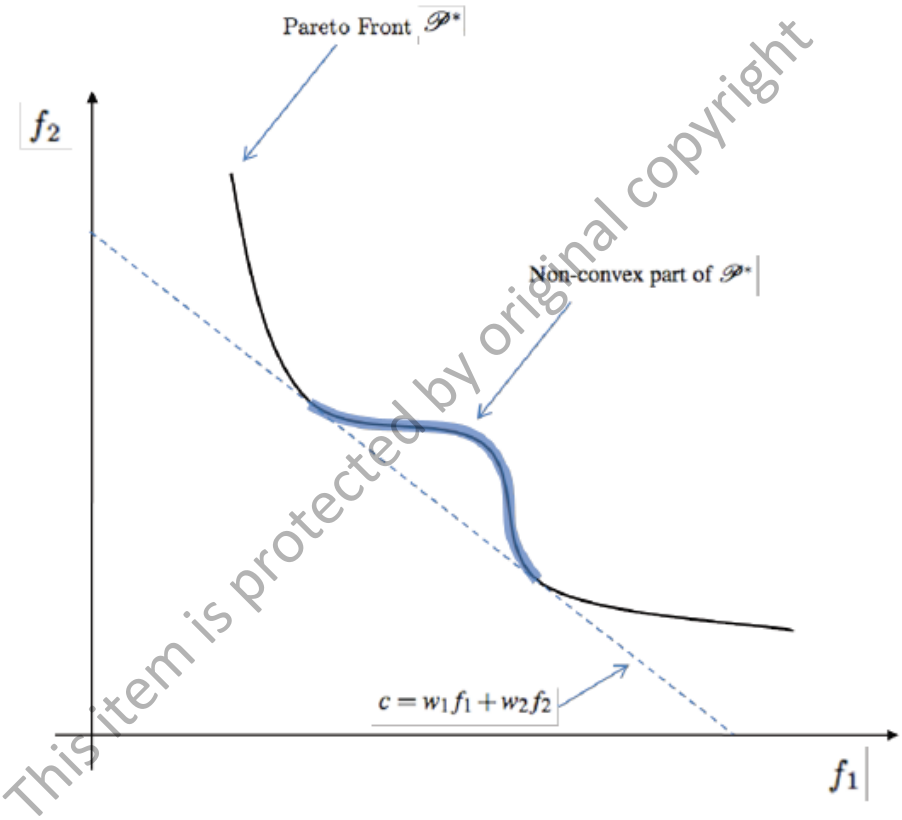


Figure 2.22: Non−convex part of the Pareto front (Ryu, Kim & Wan, 2009).

As a result, the adaptive weighted−sum method is developed to address these two drawbacks. By imposing additional inequality constraints in the usual weighted sum method, the optimization is performed only in a newly−defined feasible region where more exploration is needed. The adaptive weighted−sum method successfully solves MO optimization problems through producing well−distributed solutions, finding

Pareto optimal solutions in non−convex regions, and disregarding non−Pareto optimal solutions (Kim & De Weck, 2004).

The notion of "optimum" changes when several objective functions are involved. In MO optimization problems, the aim is to find good compromises or "trade−offs" rather than a single solution as in global optimization. The notion of "optimum" most commonly adopted is that originally proposed by Francis Ysidro Edgeworth and later generalized by Vilfredo Pareto. Although some authors call this notion the Edgeworth−Pareto optimum, the most commonly accepted term is "Pareto optimum". In some situations, the global Pareto front approach, which approximates the Pareto optimum in MO optimization through trade–offs is more practical than the weighted–sum method. This is because the weighted–sum method may possibly have a biased optimal solution influenced by big differences among relative weighting factors in the MO functions.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 System Description

In this study, various modified and hybrid CS algorithms are proposed and validated in both SO and MO optimizations for symmetric linear antenna array synthesis. The general overview of the methodology applied in this research is shown in Figure 3.1 below.

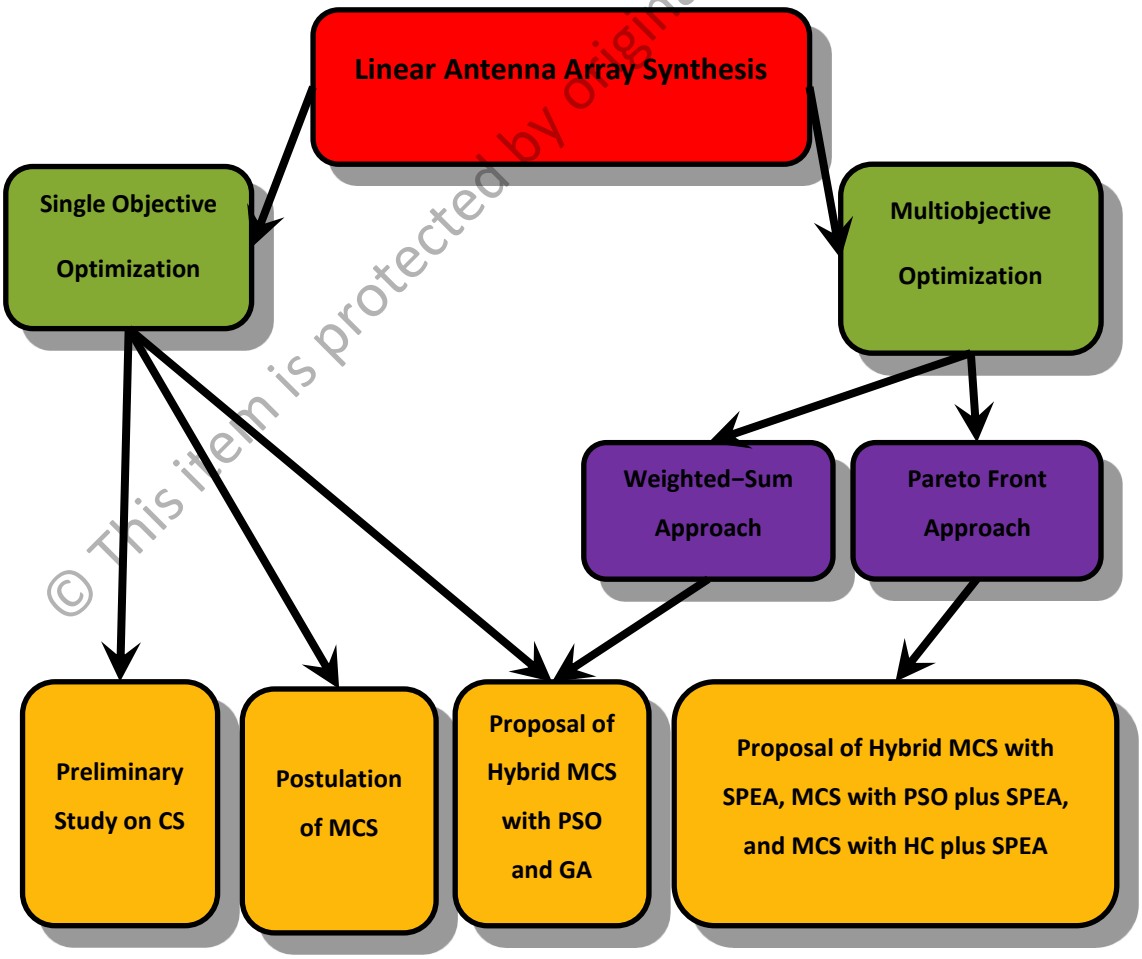


Figure 3.1: Block Diagram of Research Methodology

Based on Figure 3.1, the research is performed in five stages comprising both SO and MO optimization approaches. Precisely, the optimal solution for SO

optimization is the antenna array elements location whereas the optimal solutions for both weighted−sum and Pareto front MO optimization approaches are the antenna array elements location, current amplitude, and current phase. Figure 3.2 below shows the *n*th elements of the symmetric linear antenna array configuration in the *xy*−plane.

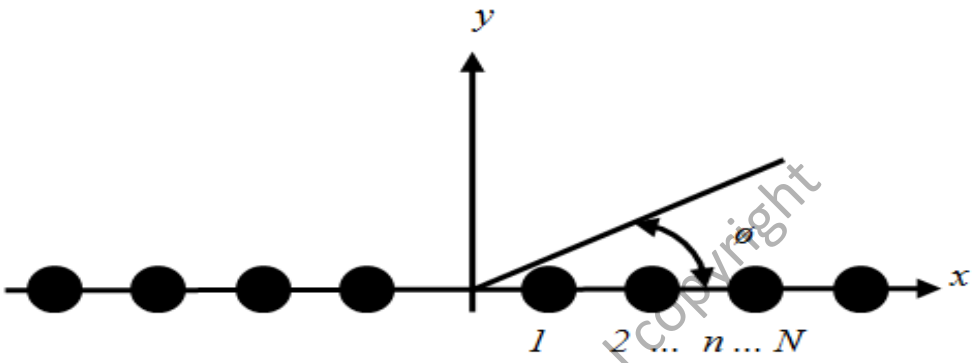


Figure 3.2: Geometry of the 2*N*−element symmetric linear array

It is assumed throughout the experiment that the 2*N*−isotropic radiators are placed symmetrically along the *x*−axis as depicted in Figure 4.2. The array factor (AF) in the azimuth plane can be defined as (Khodier & Christodoulou, 2005):

$$AF(\theta) = 2\sum_{n=1}^{N} I_n cos[kx_n cos(\theta + \emptyset_n)] \tag{3.1}$$

where $k = 2\pi / \lambda$ is the wave number, and I_n, $\emptyset_n$, and x_n are the excitation amplitude, phase, and location of the *n*th element, respectively. Based on the equation (3.1), the newly developed modified and hybrid CS algorithms are tested to find the optimal x_n, I_n, and $\emptyset_n$ of linear antenna array elements.

In SO optimization, for a uniform excitation amplitude and excitation phase, the I_n is assumed to be 1.0 whereas the $\emptyset_n$ is set to 0^o for all elements. Hence, the AF of the

linear antenna array can be simplified as (Khodier & Christodoulou, 2005):

$$AF(\theta) = 2\sum_{n=1}^{N} I_n cos[kx_n cos(\theta)] \quad (3.2)$$

Through the above simplification as in (3.2), the newly developed modified and hybrid CS algorithms are specifically used in SO optimization to find the linear array elements optimal location, x_n only.

Throughout this research, all the proposed modified and hybrid CS algorithm source codes are written and debugged using the MATLAB 7.0 (R14) and MATLAB 7.14 (R2012a) scientific software editions. In this case, all the MATLAB iterative simulations are executed via a notebook deploying the Intel® Core ™ i5−3210M (X64−based processor) operated on 64−bit operating system (OS) at 2.50 GHz processing cycle with 4.00 GB of random access memory (RAM) capacity.

3.2 Cuckoo Search Algorithm

This study primarily enhances and hybridizes the newly evolved CS metaheuristic algorithm developed recently by Xin−She Yang and Suash Deb in 2009. The original CS algorithm is based on the interesting breeding behaviour known as brood parasitism of certain species of cuckoos (Yang & Deb, 2009). Now, let us look on the interesting concept behind the nature−inspired CS algorithm.

Cuckoos are fascinating birds, not only because of the beautiful sounds they can make, but also because of their aggressive reproduction strategy. Some species such as the *Ani* and *Guira* cuckoos lay their eggs in communal nests, though they may remove others' eggs to increase the hatching probability of their own eggs (Payne, 2005). Quite a number of species engage the obligate brood parasitism by laying their eggs in the

nests of other host birds (often other species). There are three basic types of brood parasitism behaviour, which are intraspecific brood parasitism, cooperative breeding, and nest takeover.

Some host birds can engage direct conflict with the intruding cuckoos. For an example, if a host bird discovers the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Some cuckoo species such as the new world brood−parasitic *Tapera* have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colours and pattern of the eggs of a few chosen host species, thus having a greater chance for the cuckoo's eggs hatch successfully (Payne, 2005).

Many studies have shown that flight behaviour of many animals and insects has demonstrated the typical characteristics of Lévy flights (Brown, Liebovitch & Glendon, 2007, Pavlyukevich, 2007a, Pavlyukevich, 2007b and Reynolds & Frye, 2007). Lévy flight is defined as a random walk with the step−lengths based on a heavy−tailed probability distribution. When defined as a walk in a space of dimension greater than one, the steps made are in isotropic random directions. The "Lévy" in "Lévy flight" is a reference to the French mathematician Paul Pierre Lévy ("Lévy Flight", 2013).

Benoít Mandelbrot coined the term "Lévy flight" as one specific definition of the distribution of step sizes in 1982. He used the term "Cauchy flight" for the case where the distribution of step sizes is a Cauchy distribution, and "Rayleigh flight" for when the distribution is a normal distribution, which is not an example of a heavy–tailed probability distribution. Later researchers have extended the use of the term "Lévy flight" to include cases where the random walk takes place on a discrete grid rather than on a continuous space ("Lévy Flight", 2013). Consequently, such behavior has been

emulated to optimization and global optimal search strategy with a promising capability (Pavlyukevich[a], 2007 and Shlesinger, 2006).

A recent study by Reynolds and Frye (2007) shows that fruit−flies or *Drosophila Melanogaster,* explore their landscape using a series of straight flight paths punctuated by a sudden 90° turn, leading to a Lévy−flight−style intermittent scale free search pattern. Likewise, studies on human behaviour such as the Ju/'Hoansi hunter−gatherer foraging patterns also show the typical feature of Lévy flights (Brown, Liebovitch & Glendon, 2007). Moreover, the development of an optical material in which Lévy statistics govern the diffusive transport of light might allow new optical functionalities go beyond normal light diffusion (Barthelemy, Bertolotti & Wiersma, 2008).

The simplest approach of using new CS metaheuristic algorithm can be done through three idealized assumptions, which are (Yang & Deb, 2010):

i. Each cuckoo lays one egg at a time, and dumps its egg in randomly chosen nest of other species of host bird.
ii. The best nests with high quality of eggs will carry over to the next generations.
iii. The number of available host nests is fixed where the egg laid by a cuckoo is discovered by the host bird with a measured discovery rate or fraction probability, $P_a \in [0, 1]$.

In this case, the host bird may throw the egg away or may abandon the nest, hence build a completely new nest. The third assumptions can be approximated as the fraction P_a of the n nests is replaced by new nests (new random solutions). Based on these three idealized rules, the basic steps of the original CS can be summarized as the flowchart shown in Figure 3.3

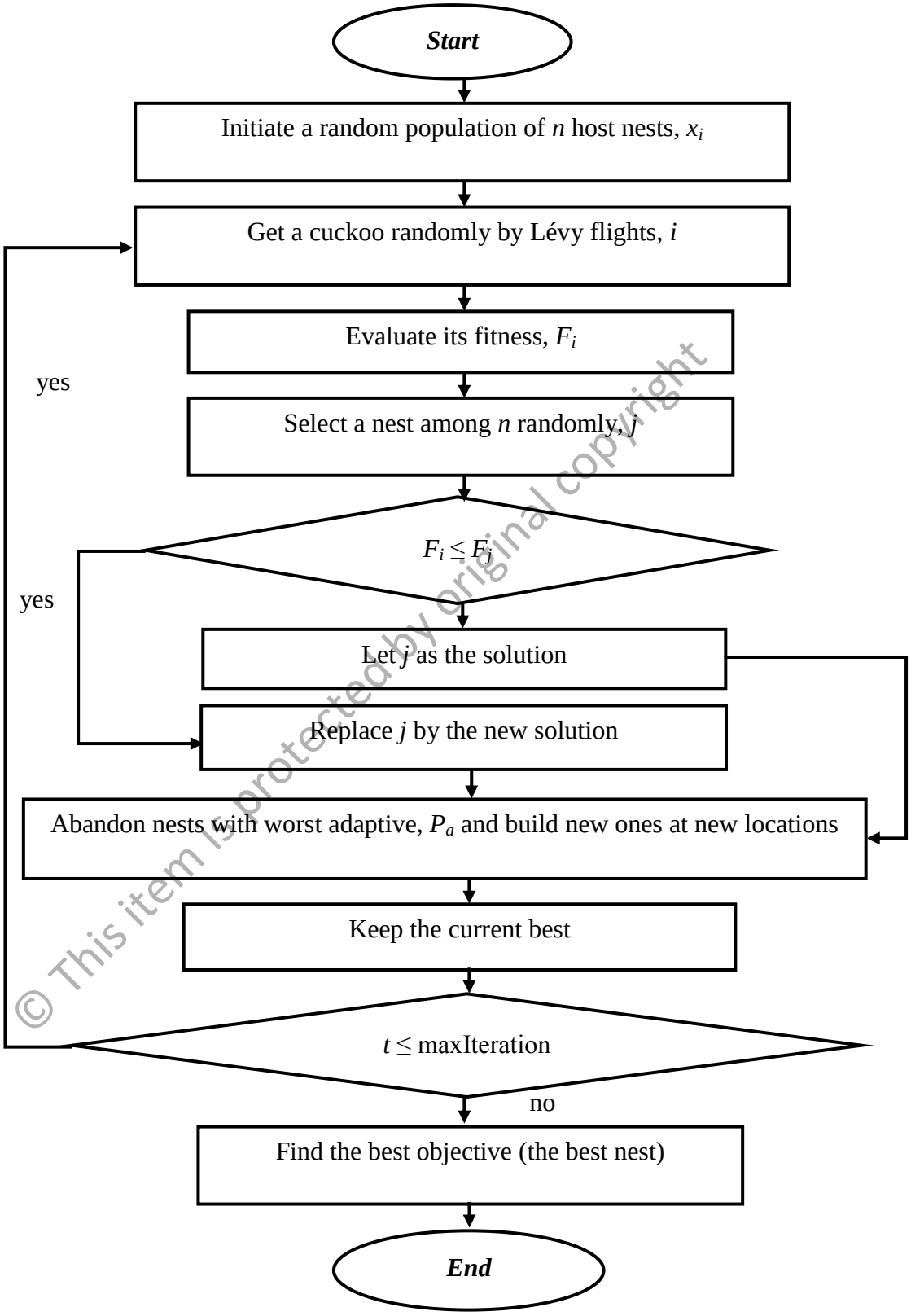


Figure 3.3: Flowchart of the Original CS Algorithm

In standard CS algorithm, we generate new solutions, x_i^{t+1} for a cuckoo, i by explicitly performing a Lévy flight (Yang & Deb, 2010):

$$x_i^{t+1} = x_i^t + \alpha \oplus \text{Lévy}\ (\lambda) \tag{3.3}$$

Note that $\alpha > 0$ is the step size related to the scales of the problem of interest while the product $\oplus$ means entry−wise multiplications. The above equation (3.3) is essentially the stochastic equation for random walk, which is a Markov chain whose next status or location only depends on the current location (the first term in the above equation) and the transition probability (the second term). As in Markov process, after a large number of steps, the distance from the origin of random walk in Lévy flight tends to form a stable distribution. Statistically, the stochastic process with both stationary and independent increments leads the stable distribution in Lévy flight. The random walk via Lévy flight is more efficient than the entry-wise product applied in PSO in exploring the search space as its step length is much longer in the long run (Yang & Deb, 2009).

Conceptually, CS random step length drawn from Lévy distribution applies power law hence it has an infinite variance with infinite mean depicted as:

$$\text{Lévy} \sim u = t^{-\lambda},\ (1 < \lambda \leq 3) \tag{3.4}$$

Here, the consecutive jumps or steps of a cuckoo essentially form a random walk process obeys a power law step length distribution with a heavy−tail. Some of the new solutions obtained within a local search via a low speed Lévy walk around the current best solution. On the contrary, other new solutions generated by a far field randomization whose locations are distant enough from the current best solution ensuring the optimizer is not trapped within a local optimum space. In sum, the

randomization in CS becomes more efficient than GA and PSO as the step−length is heavy−tailed, and any large step is possible (Yang & Deb, 2009).

We can apply three types of α−stable distribution in Lévy flight for generating new solutions in CS algorithm, which are Mantegna's algorithm, McCulloch's algorithm, and standard random walk. In Mantegna's algorithm, $\alpha \in [0.3, 1.99]$ usually becomes as the input parameter with step, v is calculated as (Mantegna, 1994):

$$v = \frac{x}{|y|^{1/\alpha}} \tag{3.5}$$

where x and y are normally distributed variables with standard deviations as follow, respectively (Mantegna, 1994):

$$\sigma_x = \left[\frac{\Gamma(1+\sigma)\sin(\pi\sigma/2)}{\Gamma(1+\alpha)/2\alpha 2^{(2\alpha-1)/2}}\right]^{1/\alpha} \tag{3.6}$$

$$\sigma_y = 1 \tag{3.7}$$

(3.5) till (3.7) indicate that the resulting distribution will have the same behaviour of a Lévy distribution for large values of random variable ($|v| \geq 0$).

To calculate the step size of Lévy flights, the v will be then multiplied with n factor where $n \in$ real number, R. Normally, n is set to 0.01 from the fact that $L/100$ is the step size of walks or flights where L is the typical length scale. It is important to set the proper factor to ensure the Lévy flights do not become too aggressive, which makes new solutions jump outside of the design or search space.

Secondly, we can use McCulloch's algorithm to generate α−stable generation of Lévy noise (Chambers, Mallows & Stuck, 1976). The algorithm returns matrix of

random numbers with characteristic exponent α, scale c, and location parameter τ. In this case, α must be greater than 0.1 due to the non−negligible probability of overflow and no skewers ($\beta = 0$) is assumed. There are three cases to calculate the simplified step (v) of α−stable distribution:

1) Cauchy case (α=1):
$v = c \tan(\varphi) + \tau$ (3.8)

2) Gaussian case (α=2):
$v = c2\sqrt{w} \sin(\varphi) + \tau$ (3.9)

3) Other cases ($\alpha \neq 1$ or $\alpha \neq 2$):
$$v = c\left(\frac{\cos\big((1-\alpha)\varphi\big)}{\xi}\right)^{\frac{1}{\alpha}-1}\left(\frac{\sin(\alpha\varphi)}{\cos(\varphi)}\right)^{\frac{1}{\alpha}} + \tau \quad (3.10)$$

where $c > 0$, ξ are negative logarithm of random numbers, φ are random angles in radians, and for simplicity, $\tau = 0$. Thirdly, the simplest way to generate a stable Lévy distribution is by generating standard random walk where step, $v = 1$ constantly.

In addition, CS is also proven more generic and robust than the PSO and GA in optimizing multimodal objective functions. Through simulations running on various standard test functions, CS is more efficient in finding the global optima with higher success rates (Yang & Deb, 2009 and 2010). This is partly due to the fact that there are fewer parameters to be fine−tuned in CS, hence, potentially more generic to adapt to a wider class of optimization problems (Yang & Deb, 2009).

3.3 Cuckoo Search Algorithm in Linear Antenna Array Synthesis

In the first stage of this research, five internal parameters of the standard CS algorithm are tested specifically upon their implicit effects of the normalized antenna radiation pattern. The five parameters include Lévy flights distribution type or α value,

α–stable distribution method, length step factor, number of host nest (population), and discovery rate or fraction probability or P_a. In this case, the CS algorithm is simulated using $2N$ = 10 and $2N$ = 20 linear array configurations. Then, the performance results are compared with the conventional array. Based on (3.2), the experiment involves the usage of the CS algorithm to optimize the inter−element spacing with respect to the $\lambda/2$ while preserving a uniform excitation amplitude and phase over the array aperture. The $\lambda/2$ spacing is suitable to avoid mutual coupling impairment occurs among antenna array elements.

In this study, based on (3.2), the fitness optimization is primarily performed to design the geometry of a symmetric linear antenna steering at the desired direction with a minimum average SLL and/or nulls control using the following objective function:

$$\text{Fitness}f = \sum_{i} \frac{1}{\Delta\theta_i} \int_{\theta_{li}}^{\theta_{ui}} |AF(\theta)|^2 d\theta + \sum_{k} |AF(\theta_k)|^2 \tag{3.11}$$

where $[\theta_{li}, \theta_{ui}]$ is the spatial region in which the SLL is suppressed, $\Delta\theta_i = \theta_{ui} - \theta_{li}$, and θ_k are the directions of the prescribed nulls or interferers. Precisely, the first−term on the right−hand side of the fitness function focuses on SLL suppression whereas the second−term on the right−hand side is used for nulls control. In this study, the nest's location vector resulted the minimum value of the fitness function is chosen as the best nest's location (the best normalized locations of antenna array isotropic radiators or elements).

The second stage involves the postulation of modified CS (MCS) algorithm in linear antenna array synthesis. In this section, the MCS algorithm is proposed by integrating the standard CS algorithm with the Roulette wheel selection operator, and adaptive inertia weight, w and adaptive fraction probability, P_a primarily to control the

Lévy flight search motion ability. The Roulette wheel operator performs the possible selections of potential solutions (assigned in a portion of the wheel) based on their fitness value. This is done by dividing the fitness of a selection by the total fitness of all the selections, thus normalizing them to one before a random selection is made similar to how the Roulette wheel is rotated. The main purpose is to ensure that candidate solutions with a superior fitness had a larger possibility to be selected. On the other hand, the aim of introducing the adaptive *w* is primarily to control the exploration ability towards optimal solutions in *N*–dimensional search space. Figure 3.4 shows the flowchart of the proposed MCS algorithm.

Ideally, this breakthrough should make the MCS algorithm more robust and efficient than the original CS algorithm particularly in SLL suppression and/or prescribed nulls control. In the MCS algorithm, the process of generating new solutions, $x^{(t+1)}$ for a cuckoo, *i* via the Lévy flight with adaptive *w* can be restated as:

$$x_i^{t+1} = w \times x_i^t + \alpha \oplus \text{Lévy}(\lambda) \tag{3.12}$$

As a matter of fact, a larger *w* would perform further global search ability whereas a smaller *w* would execute further local search ability. Based on (3.12), the *w* is linearly decreased from a relatively large value to a small value as number of iterations increased. To achieve this, the adaptive *w* is defined as:

$$w = w_{max} - [(w_{max} - w_{min}) \times \text{iter}] / \text{maxIter} \tag{3.13}$$

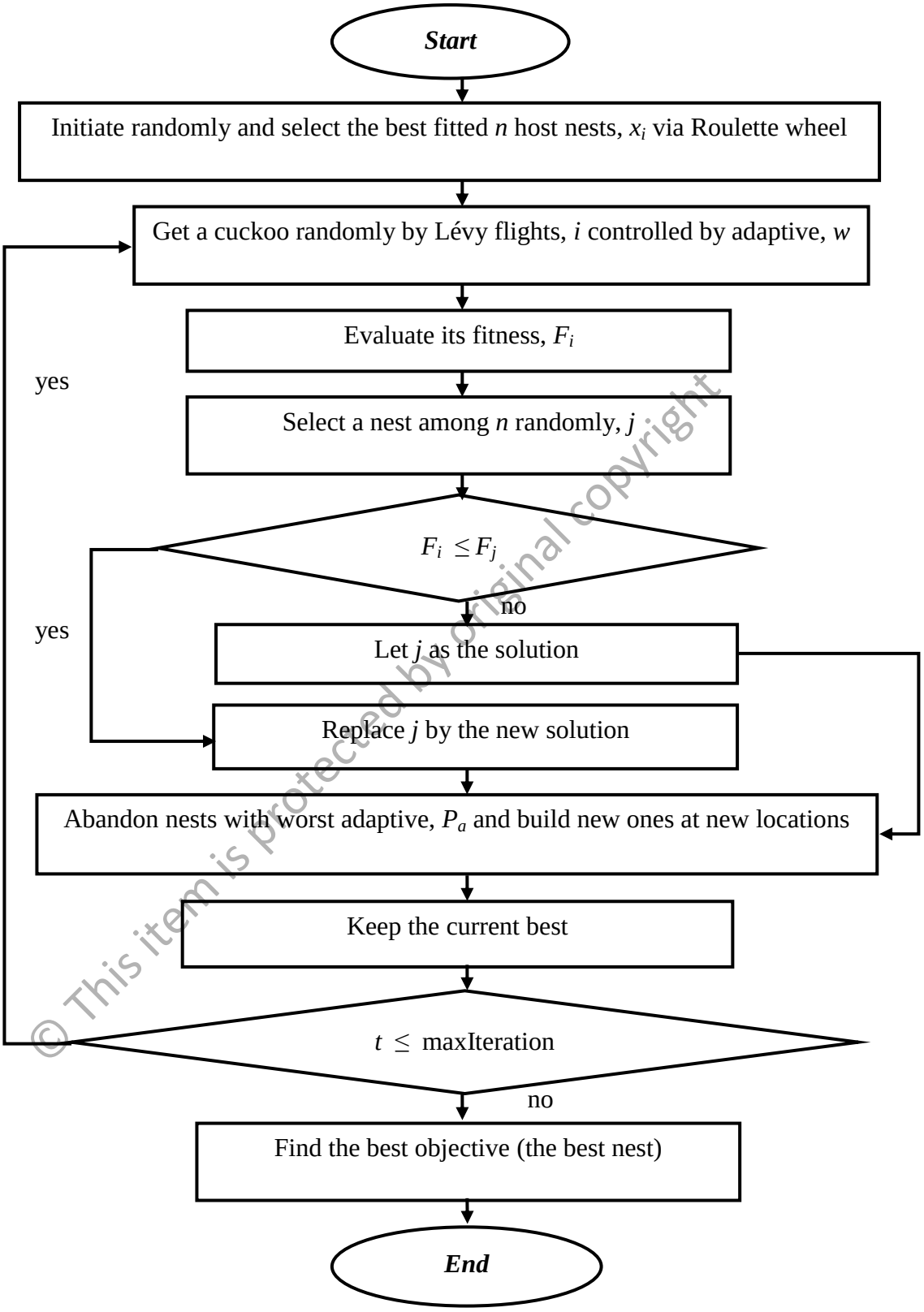


Figure 3.4: Flowchart of the Proposed MCS Algorithm

In this study, the fitness minimization is done with the aim to determine the optimal locations of *n*th element based on (3.2) and (3.11), respectively. Based on a flowchart in Figure 3.4, the following pseudo–code of the newly developed MCS algorithm is proposed and tested:

```
begin
    Let iter denote the iteration number of MCS.
    Iter←1;
    Initialize population of host nests with size n at Iter=1;
    for each iteration
        Operate the Roulette wheel selection to obtain the
        "fittest" host nests with size n;
        Generate a new solution (host nest) but keep the
        current best (say, i) randomly by Lévy flights
        incorporating with inertia weight, w, which controls
        the search ability according to (3.12);
        Evaluate new solution fitness, F_i according to
        (3.11);
        Get a selected host nest among n (say, j) and
        calculate its fitness, F_j according to (3.11);
        if (F_i < F_j)
            Replace j by the new solution, i;
        end
        A fraction probability, P_a of worse nests is
        abandoned and a new nest (solution) is built;
        Keep the best nests with quality solutions;
        Rank the solutions and find the current best nest;
    end
    Post-process results and visualization;
end
```


Initially, the study focuses on the four internal parameters of the MCS algorithm specifically on their imperative effects in the antenna array geometry synthesis. The parameters are Lévy flight distribution type (α value), α–stable distribution method, number of host nest (population), and discovery rate or fraction probability, P_a. The optimal inter–element spacing solutions obtained by the MCS algorithm are then validated through comparisons with the standard CS–optimizer and the conventional antenna array within the uniform and Dolph–Chebyshev envelope patterns.

In the third stage, the MCS algorithm is hybridized with two EA/EC techniques, which are GA and PSO in linear antenna array synthesis. Both the proposed MCSPSO and MCSGA hybrid algorithms use the fitness, *f* function to guide the Lévy flight motions towards the optimal solutions in *N*–dimensional search space. Precisely, the optimal solutions refer to the linear antenna array elements location along in the *x*–axis. The resulting optimal positions taken from the global minimum value of equation (3.11) are presumed to be as the best particle in PSO or best chromosome in GA. Based on a flowchart in Figure 3.5, the following is the postulated pseudo–code of MCSPSO hybrid algorithm, which is developed and validated in this study:

begin

Let *iter* denote the iteration number of MCSPSO.

iter ←1;

Initialize population of host nests with size *n* at *iter*=1;

for each iteration

Operate the Roulette wheel selection to obtain the "fittest" host nests with size *n*;

Generate a new set of solutions (host nests) but keep the Current best (say, *i*) randomly by Lévy flights incorporating with inertia weight, *w*, which controls the search ability according to (3.12);

Evaluate new solution fitness, F_i according to (3.11);

Get a selected set of host nests among n (say, *j*) and calculate its fitness, F_j according to (3.11);

if ($F_i < F_j$) % *fitness minimization* %

Replace j by the new set of solutions, *i*;

end

A dynamic fraction probability, P_a of worse nests is abandoned and a new nest (set of solution) is built;

Keep the best nests with quality solutions;

```
        Let the best nests become as initial particles;
        for each particle
            Calculate fitness value according to (3.11);
            if the fitness value is better than the best
            fitness value (pbest) in history
                Set current value as the new pbest;
            end
        end
        Choose the particle with the best fitness value of
        all the particles as the gbest;
        for each particle
            Calculate particle velocity according equation
            (2.38);
            Update particle position according equation
            (2.39);
        end
        Evaluate the updated current fitness value according
        to (3.11);
        if the new current fitness value is better than the
        fitness of pbest;
            Set current value as the new pbest;
        end
        if the new current fitness value is better than the
        fitness of gbest
            Set current value as the new gbest;
        end
        Keep the best particles with quality solutions;
        Rank the solutions and find the current best
        particle;
    end
    Post-process results and visualization;
end
```

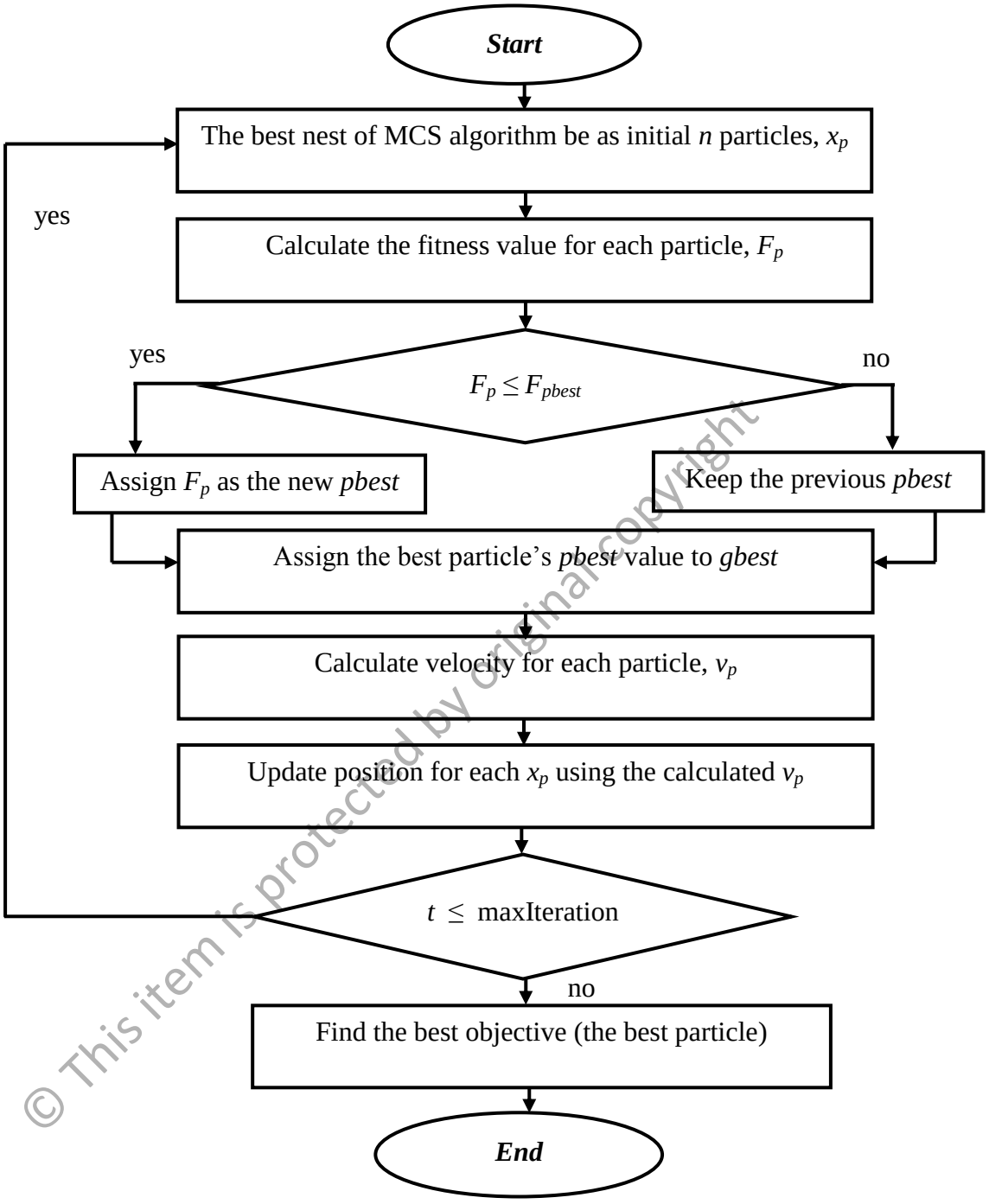


Figure 3.5: Flowchart of the Proposed MCSPSO Algorithm

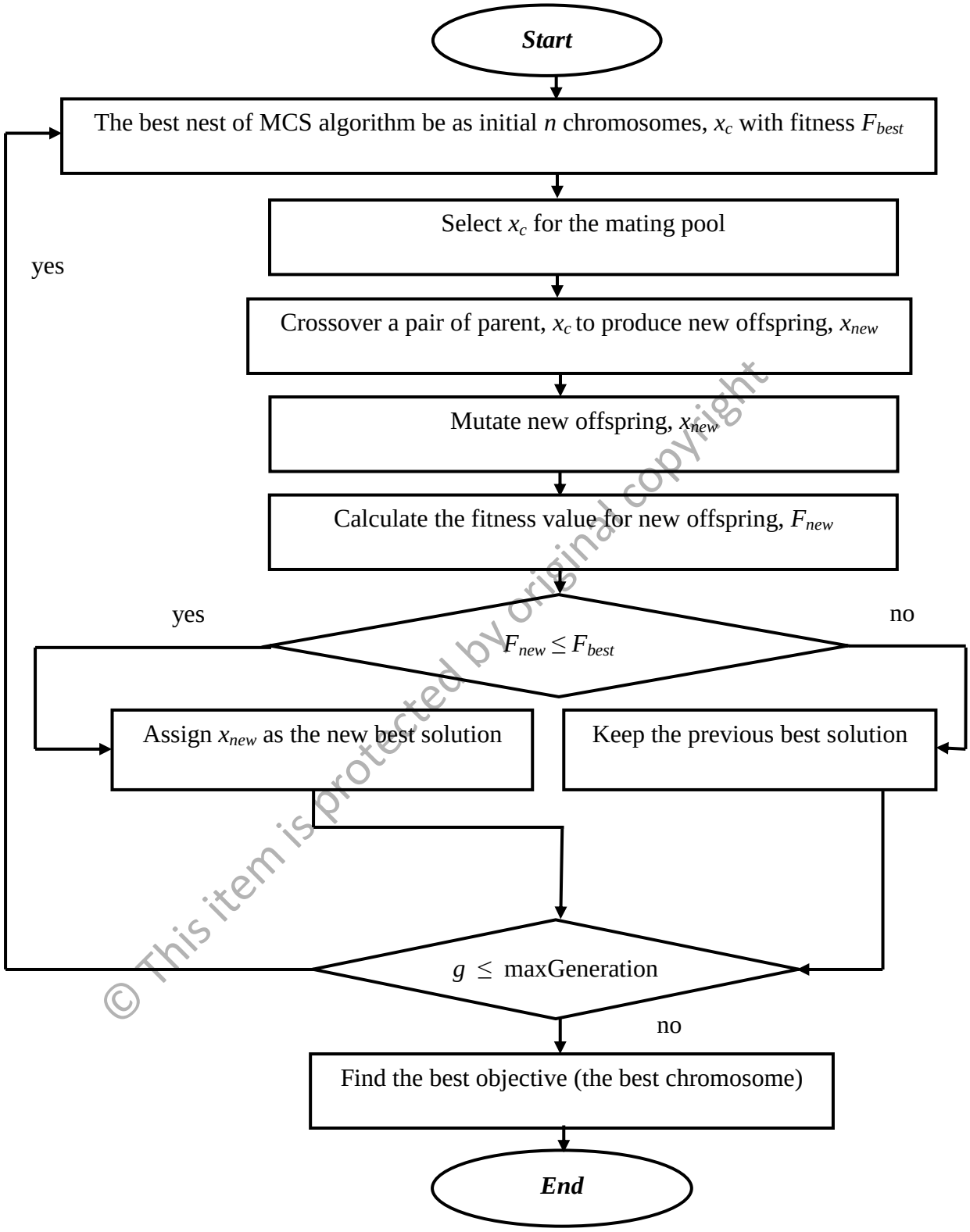


Figure 3.6: Flowchart of the Proposed MCSGA Algorithm

Based on a flowchart in Figure 3.6, the following is the proposed pseudo–code for the MCSGA hybrid algorithm, which is developed and validated in this study:

begin

Let *gen* denote the generation number of MCSGA.

gen←1;

Initialize population of host nests with size *n* at *gen*=1;

for each generation

Operate the Roulette wheel selection to obtain the "fittest" host nests with size *n*;

Generate a new set of solutions (host nests) but keep the current best (say, *i*) randomly by Lévy flights incorporating with inertia weight, *w*, which controls the search ability according to (3.12);

Evaluate new solution fitness, F_i according to (3.11);

Get a selected set of host nests among n (say, *j*) and calculate its fitness, F_j according to (3.11);

if (F_i< F_j) % *fitness minimization* %

Replace *j* by the new set of solutions, *i*;

end

A dynamic fraction probability, P_a of worse nests is abandoned and a new nest (set of solution) is built;

Keep the best nests with quality solutions;

Let the best nests become as initial chromosomes;

Evaluate each individual's fitness according to (3.11);

Select pairs to mate from best–ranked individuals;

Mate pairs at random;

Apply crossover operator;

Apply mutation operator;

for each chromosome

Calculate new fitness value according to (3.11);

if the new fitness value is better than the best fitness value in history

```
                Set current value as the new best
                chromosomes;
            end
        end
        Keep the best chromosomes with quality solutions;
        Rank the solutions and find the current best
        chromosome;
    end
    Post-process results and visualization;
end
```

In the fourth stage, the proposed MCS, hybrid MCSGA and hybrid MCSPSO algorithms are deployed to perform weighted−sum multiobjective (MO) optimization for linear antenna array synthesis. Precisely, the MCSGA leads the MCS algorithm to control effectively exploration of the best–fitted chromosomes (host nests) in search space after undergoing the unique crossover and mutation processes. On the other hand, the MCSPSO counterpart helps the MCS algorithm in controlling the velocity (direction and speed) of particles (cuckoos) Lévy flight motions towards optimal solutions. The position updating process determines the personal best fitness (*pbest*) for all particles and through it to locate a particle with the global best fitness (*gbest*) in search space.

In this case, the MCS algorithm performed optimization involving three objective functions simultaneously until the maximum number of iteration achieved using the weighted–aggregation objective or total normalized weighted–sum fitness function as defined below:

$$Fitness(\text{iter}) = \frac{f_1}{\text{mean}\left(f_1(\text{iter}=1)\right)} + \frac{f_2}{\text{mean}\left(f_2(\text{iter}=1)\right)} + \frac{f_3}{\text{mean}\left(f_3(\text{iter}=1)\right)} \tag{3.14}$$

The approach primarily exploits the weighted–sum of three objective functions, f_1, f_2, and f_3 where the fitness, f_1 is defined as below:

$$f_1 = \min\{1/directivity\}$$

(3.15)

The directivity is the ratio of the radiation intensity in a given direction from the antenna to the radiation intensity averaged over all directions. Mathematically, the directivity in terms of beam solid angle can be defined as:

$$D(\theta,\varphi) = \frac{U(\theta,\varphi)}{U_{avg}} = 4\pi\frac{U(\theta,\varphi)}{P_{rad}}$$

(3.16)

where $U(\theta,\varphi) = B_o F(\theta,\varphi)$is the antenna radiation intensity, and U_{avg} is radiation intensity averaged over all directions. In this study, the directivity is measured in decibel (dB) unit through a formula below:

$$D_{dB}(\theta,\varphi) = 10\log_{10} D(\theta,\varphi)$$

(3.17)

The fitness, f_2 is defined as below:

$$f_2 = \min\left\{\sum_i \frac{1}{\Delta\emptyset_i}\int_{\emptyset_{li}}^{\emptyset_{ui}} |AF(\emptyset)|^2 d\emptyset + \sum_k |AF(\emptyset_k)|^2\right\}$$

(3.18)

where $[\emptyset_{li}, \emptyset_{ui}]$'s are the spatial regions in which the SLL is suppressed, $\Delta\emptyset_l = \emptyset_{ui} - \emptyset_{li}$, and $\emptyset_k$'s are the directions of the prescribed nulls or interferers. Moreover, the fitness, f_3 is stated as below:

$$f_3 = \min\{1 - \text{DRR}\}$$

(3.19)

where the dynamic range ratio (DRR) is mathematically defined as:

$$\text{DRR} = |\max \text{excitation amplitude}/\min \text{excitation amplitude}|$$

(3.20)

Basically, the fitness, f_1 is related to the evaluation of minimizing the $\frac{1}{directivity}$ ratio or in other words, maximizing the antenna directivity. The fitness, f_2 is concerned on the minimizing the average side lobes radiated by the antenna. Furthermore, the fitness, f_3 is mainly considered for minimizing the deviations between |maximum excitation amplitude| and |minimum excitation amplitude| for all linear array elements.

The postulated MCSPSO, MCSGA, and MCS algorithms use the dynamic discovery rate, P_a, and the inertia weight, w. In this case, the dynamic P_a reduces the possibility of host birds of other species discover the cuckoo's egg as the iteration increases. In other words, the dynamic discovery rate or fraction probability is getting smaller gradually as the number of iteration rises initiating the brood–parasitism behavior successes is calculated by:

$$P_a = P_{a_{max}} - \left[\left(P_{a_{max}} - P_{a_{min}}\right) \times \text{iter}\right]/\text{maxIter} \tag{3.21}$$

where $P_{a_{max}}$ is the maximum discovery rate, and $P_{a_{min}}$ is the minimum discovery rate, respectively. On the other hand, the dynamic inertia weight, w is calculated as in (3.13).

For simplicity, it is assumed that the weight given for all objectives f_1, f_2, and f_3 are equal to 1.0. The weighted–sum fitness in (3.14) is normalized through dividing fitness, all objectives f_1, f_2, and f_3 in all iterations with their respective mean values of the first iteration. The purpose is to reduce the possible bias caused by differences in terms of magnitude among three objective functions, respectively. The resulting optimal location, amplitude, and phase vectors taken from the global minimum value of (3.14) are declared to be the optimal solutions.

Lastly, the fifth stage postulates various hybrid MCS algorithms via the global Pareto front MO approach to find the three optimal decision variables, which are

positions, excitation amplitudes, and excitation phases of the symmetric linear antenna array elements, respectively. The approach primarily exploits the trade–offs of three objective functions, f_1, f_2, and f_3 as defined in (3.15), (3.18), and (3.19), respectively. In the function, f_3 the dynamic range ratio (DRR) is calculated using (3.20).

The strength Pareto evolutionary algorithm (SPEA) method is implemented to perform the MO optimization for the linear antenna array synthesis. In this case, the postulated MCS algorithm will be hybridized with the SPEA method known as MCSSPEA to find the Pareto front non–dominated solutions. Furthermore, there is the hybridization of hill climbing (HC) algorithm to improve the local search capability of MCS and SPEA algorithms referred as MCSHCSPEA algorithm, and also the hybridization of both MCS and PSO algorithms with SPEA known as MCSPSOSPEA algorithm, respectively.

The MCSSPEA, MCSHCSPEA, and MCSPSOSPEA hybrid algorithms deploy the Roulette wheel selection operator, dynamic P_a, and dynamic w within the MCS optimizer. In this case, the dynamic P_a reduces the possibility of host birds of other species discover the cuckoo's egg as the iteration increases. In other words, the dynamic P_a is getting smaller gradually as the number of iteration rises initiating the brood–parasitism behavior successesas defined in (3.21). Moreover, the dynamic w is calculated using (3.13).

Moreover, there is a calculation mechanism for MCSSPEA, MCSHCSPEA, and MCSPSOSPEA optimizers to increase the spread of Pareto front. The purpose is to reduce the local trap predicament as sometimes appeared in the original SPEA. In this study, the distance for three MO functions with p non–dominated solutions is calculated as below:

$$Distance_{i,j} = \left| \sum_{i=1}^{p} \sum_{j=i}^{p} f_{1_i} - f_{1_j} \right| + \left| \sum_{i=1}^{p} \sum_{j=i}^{p} f_{2_i} - f_{2_j} \right| + \left| \sum_{i=1}^{p} \sum_{j=i}^{p} f_{3_i} - f_{3_j} \right| \tag{3.22}$$

The following equations (3.23) – (3.25) show the Pareto front spread fitness formulation of p non–dominated solutions (Zitzler, 2004):

$$MinDistance_i = \min Distance_{i,j} \tag{3.23}$$

$$Density_i = \frac{1}{(MinDistance_i + \delta)} \tag{3.24}$$

$$SpreadFitness_i = RawFitness_i + Density_i \tag{3.25}$$

The equation (3.24) shows that the smaller distances between the inter–Pareto raw points, the higher density would be. The denominator of (3.24) has the minimum distance added with a constant value, δ to ensure the density is enough to spread the raw points as calculated in (3.25). Hence, the simulation can achieve the big spread of optimal non–dominated solutions in the Pareto front trade–offs domain.

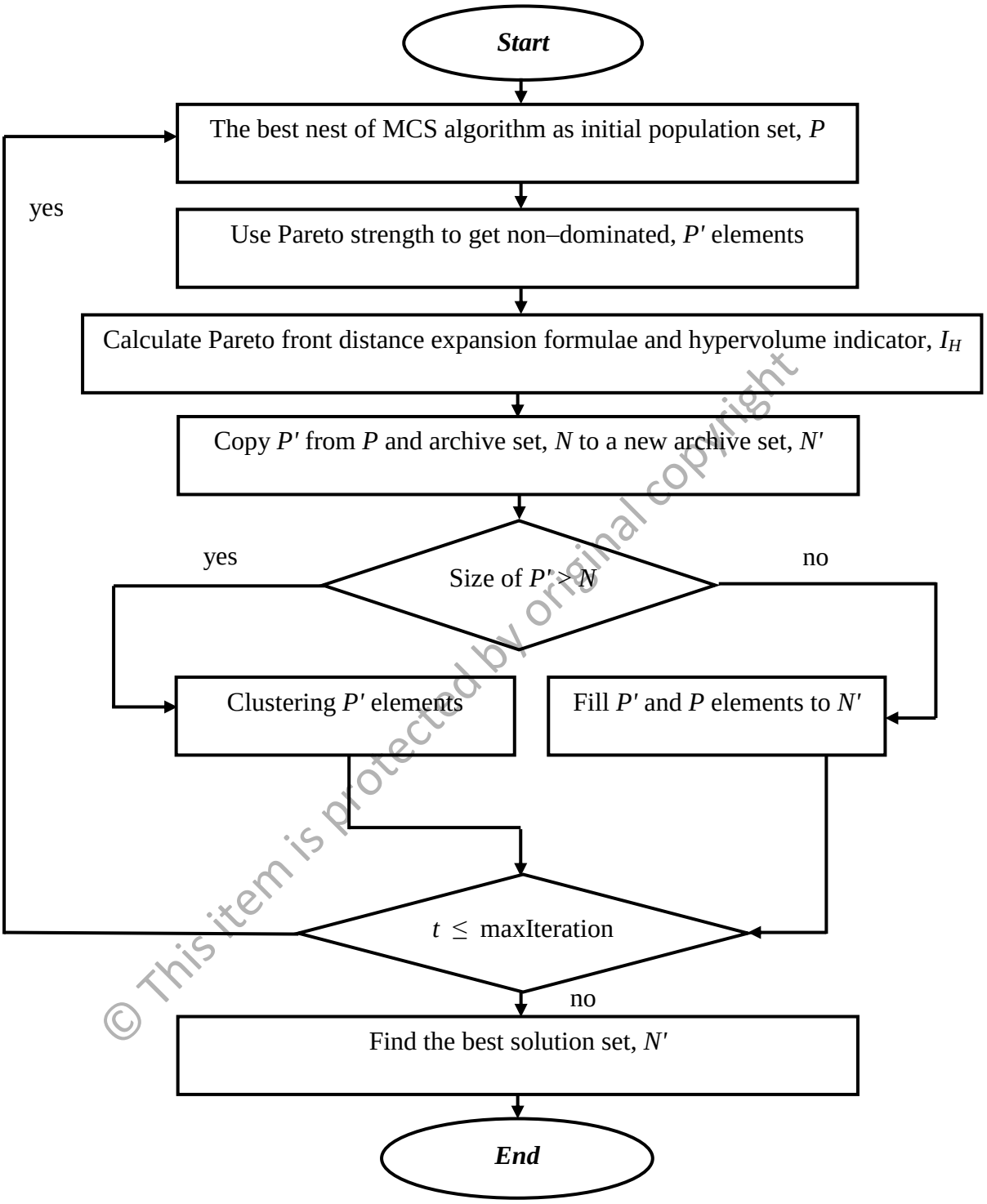


Figure 3.7: Flowchart of the Proposed MCSSPEA Algorithm

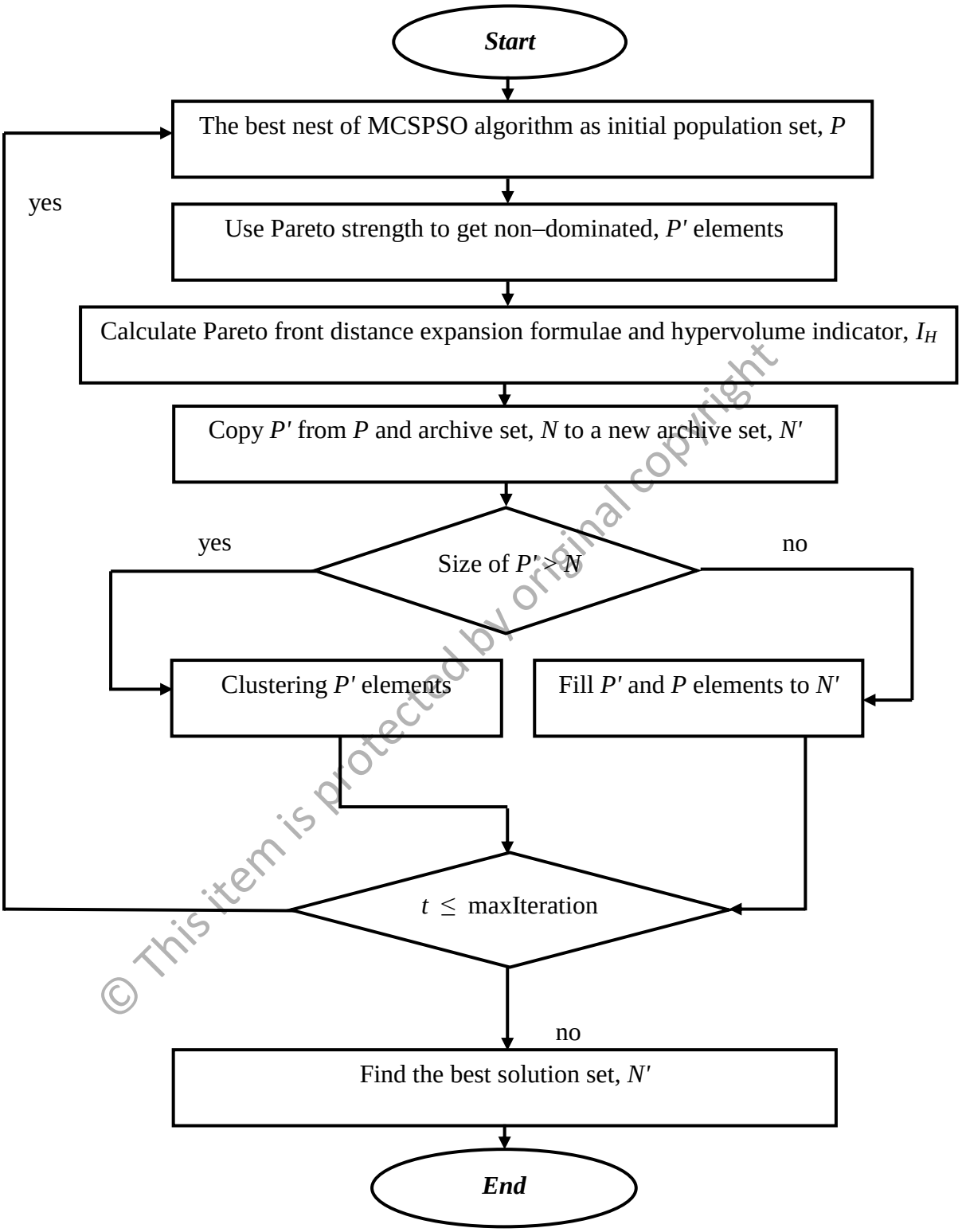


Figure 3.8: Flowchart of the Proposed MCSPSOSPEA Algorithm

Since this study involves MO functions, hypervolume measure or indicator of the dominated portion of the objective space is applied as a quality measure for Pareto set approximations. The hypervolume indicator, $I_H(A)$ of a solution set $A \subseteq X$ can be defined as the hypervolume of the space with $N = 3$ objective functions that is dominated by the set A and bounded by a reference point $r = (r_1, r_2, ,r_3) \in \mathbb{R}^3$(Brockhoff et al., 2008):

$$I_H(A) = vol(\cup_{a \in A} [f_1(a), r_1] \times [f_2(a), r_2] \times [f_3(a), r_3]) \tag{3.26}$$

where *vol*(.) is the Lebesgue measure and ([f_1(*a*), r_1]×[f_2(*a*), r_2]×[f_3(*a*), r_3]) denotes as the 3–dimensional hypercuboid consisting of all points, which are weakly dominated by the point *a* but not weakly dominated by the reference point.

Based on a flowchart shown in Figure 3.7, the following is the postulated pseudo–code of MCSSPEA algorithm, which is developed and tested in this study:

begin

Let *iter* denote the iteration number of MCSSPEA.

iter←1;

Initialize population of host nests with size *n* at *iter*=1;

for each iteration

Operate the Roulette wheel selection to obtain the "fittest" host nests with size *n*;

Generate a new set of solutions (host nests) but keep the Current best (say, *i*) randomly by Lévy flights incorporating with inertia weight, *w*, which controls the search ability according to (3.13);

Evaluate new solution MO fitness, F_i according to (3.15),(3.18), and (3.19);

Get a selected set of host nests among n (say, *j*) and calculate its MO fitness, F_j according to (3.15), (3.18), and (3.19);

```
        if (F_i ≤ F_j) % fitness minimization %
            Replace j by the new set of solutions, i;
        end
        A dynamic fraction probability, P_a of worse nests is
        abandoned and a new nest (set of solution) is built;
        Keep the best nests with quality solutions;
        Rank the solutions;
        Population←Current best individuals;
        Archive←∅;
        for S_i ∈Population
            S_i_objectives ←CalculateObjectives(S_i);
        end
        Union←Population+Archive;
        for S_i ∈Union
        S_i_raw ←CalculateRawFitness(S_i,Union);
        S_i_density ←CalculateSolutionDensity(S_i,Union);
        S_i_fitness ←S_i_raw +S_i_density;
        end
    Archive←GetNonDominated(Union);
    if Size(Archive)<Archive_size
        PopulateWithRemainingBest(Union,Archive,Archive_size);
    elseif Size(Archive)>Archive_size
        RemoveMostSimilar(Archive,Archive_size);
        end
        Return(GetNonDominated(Archive));
    end
    Post-process result (Archive) and visualization;
end
```

Based on a flowchart shown in Figure 3.8, the postulated pseudo–code of MCSPSOSPEA algorithm, which is developed and tested in this study:

begin

Let *iter* denote the iteration number of MCSPSOSPEA.

iter←1;

Initialize population of host nests with size *n* at *iter*=1;

for each iteration

Operate the Roulette wheel selection to obtain the "fittest" host nests with size *n*;

Generate a new set of solutions (host nests) but keep the Current best (say, *i*) randomly by Lévy flights incorporating with inertia weight, *w*, which controls the search ability according to (3.13);

Evaluate new solution MO fitness, F_i according to (3.15),(3.18), and (3.19);

Get a selected set of host nests among n (say, *j*) and calculate its MO fitness, F_j according to (3.15), (3.18), and (3.19);

if ($F_i \leq F_j$) *% fitness minimization %*

Replace *j* by the new set of solutions, *i*;

end

A dynamic fraction probability, P_a of worse nests is abandoned and a new nest (set of solution) is built;

Keep the best nests with quality solutions;

Let the best nests become as initial particles;

for each particle

Calculate MO fitness value according to (3.15), (3.18), and (3.19);

if the fitness value is better than the best MO fitness value (*pbest*) in history

Set current value as the new *pbest*;

end

end

for each particle

Calculate particle velocity according to (2.38);

```
            Update particle position according to (2.39);
        end
        Evaluate the updated current MO fitness value
        according to (3.15), (3.18), and (3.19);
        if the new current MO fitness value is better than
        the fitness of pbest;
            Set current value as the new pbest;
        end
        Keep the best particles with quality solutions;
        Rank the solutions and find the current best
        particles;
        Population←Current best particles;
        Archive←∅;
        for S_i ∈Population
            S_i_objectives ←CalculateObjectives(S_i);
        end
        Union←Population+Archive;
        for S_i ∈Union
        S_i_raw ←CalculateRawFitness(S_i,Union);
        S_i_density ←CalculateSolutionDensity(S_i,Union);
        S_i_fitness ←S_i_raw +S_i_density ;
        end
    Archive←GetNonDominated(Union);
    if Size(Archive)<Archive_size
        PopulateWithRemainingBest(Union,Archive,Archive_size);
    elseif Size(Archive)>Archive_size
        RemoveMostSimilar(Archive,Archive_size);
        end
        Return(GetNonDominated(Archive));
    end
    Post-process result (Archive) and visualization;
end
```

In this stage, there is also a postulation of MCS algorithms with two stochastic algorithms, which are hill climbing (HC) and SPEA metaheuristics. HC is a local search method that uses an iterative improvement strategy. The strategy is applied to a single point, such as the current point (or state) in the search space. At each iteration, a new point $\mathbf{x}'$ is selected by performing a small displacement or perturbation in the current point **x**, i.e., the new point is selected in the neighbourhood of the current point: $\mathbf{x}:\mathbf{x}'+\Delta\mathbf{x}$ (De Castro, 2006).

If that new point provides a better value for the evaluation function, then the new point becomes the current point. Else, some other displacement is promoted in the current point (a new neighbour is chosen) and tested against its previous value. Termination occurs when one of the following stopping criteria is met:

i. No further improvement can be achieved.

ii. A fixed number of iterations have been performed.

iii. A goal point is attained.

Let **x** be the current point, **g** the goal point, and `max_iteration` a maximum number of iterations allowed. The pseudo−code of a standard (simple) HC is as follow:

```
function [x] = hill_climbing(max_iteration,g)
     initialize x
     eval (x)
     t ← 1
     while t < max_iteration & x ~= g && no_improvement do
          x′ ← perturbation (x)
          eval (x′)
          if eval (x′) is better than eval (x)
               then x ← x′
          end
          t ← t + 1
     end
endfunction
```

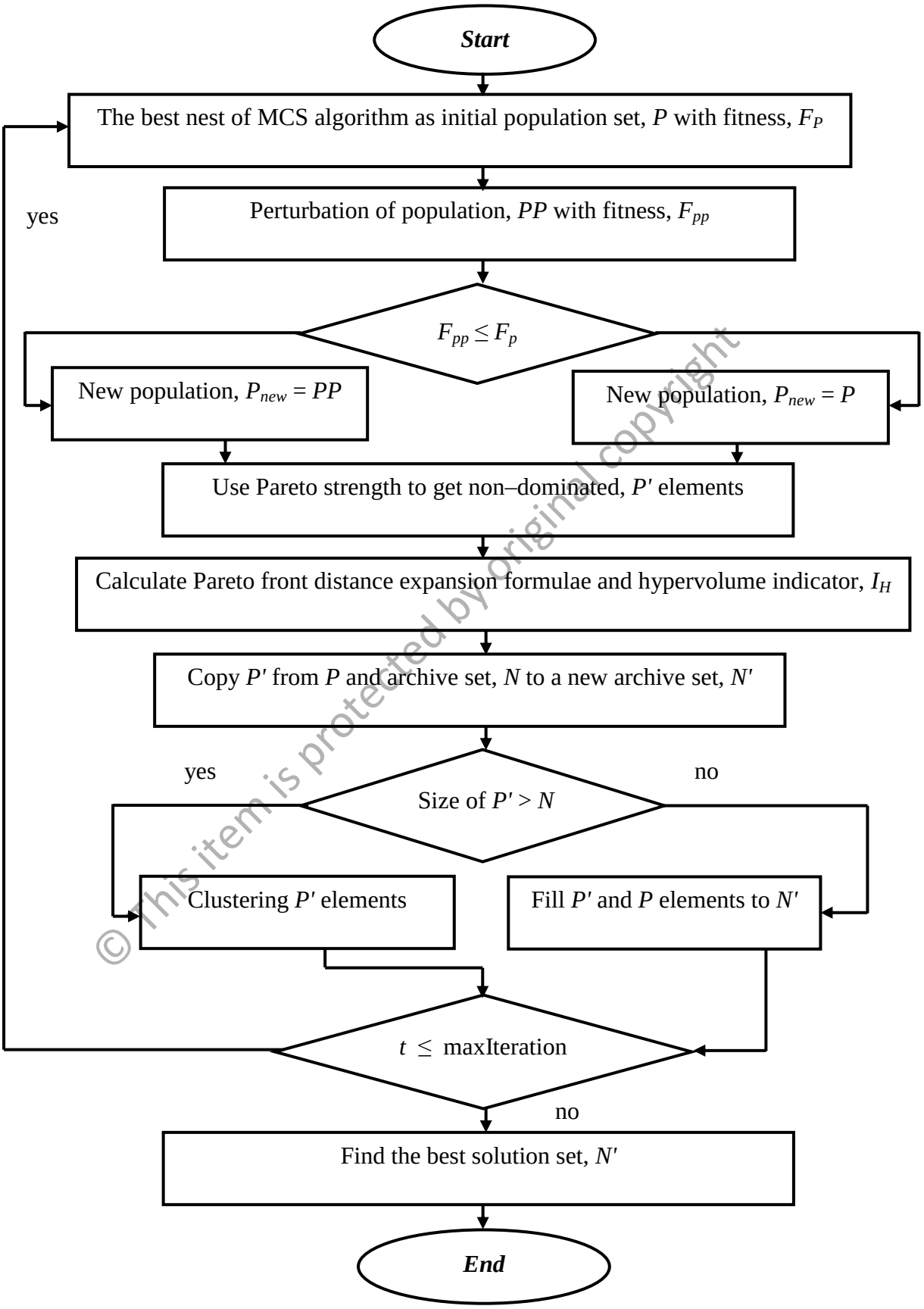


Figure 3.9: Flowchart of the Proposed MCSHCSPEA Algorithm

Indeed, HC algorithms have some limitations, which are usually terminate at local optima solutions, no information about the distance between the solution found and the global optimum. The optimum found depends on the initial configuration, and generally not possible to provide an upper bound for the computational time of the algorithm (De Castro, 2006).

Based on a flowchart in Figure 3.9, the following is the postulated pseudo–code of MCSHCSPEA algorithm, which is also developed and validated in this simulation:

begin

Let *iter* denote the iteration number of MCSHCSPEA.

iter←1;

Initialize population of host nests with size *n* at *iter*=1;

for each iteration

Operate the Roulette wheel selection to obtain the "fittest" host nests with size *n*;

Generate a new set of solutions (host nests) but keep the Current best (say, *i*) randomly by Lévy flights incorporating with inertia weight, *w*, which controls the search ability according to (3.13);

Evaluate new solution MO fitness, F_i according to (3.15),(3.18), and (3.19);

Get a selected set of host nests among n (say, *j*) and calculate its MO fitness, F_j according to (3.15), (3.18), and (3.19);

if ($F_i \leq F_j$) *% fitness minimization %*

Replace *j* by the new set of solutions, *i*;

end

A dynamic fraction probability, P_a of worse nests is abandoned and a new nest (set of solution) is built;

Keep the best nests with quality solutions;

Let the best nests become as initial individuals;

x←best nests;

```
        for each individual, x
            Calculate MO fitness, Fx value according to
            (3.15), (3.18), and (3.19);
    end
    x'←perturbation (x)
    for each individual, x'
            Calculate multiobjective fitness, Fx' value
            according to (3.15), (3.18), and (3.19);
    end
        if (Fx' ≤ Fx) % fitness minimization%
            Replace x by the new set of solutions, x';
        end
        Keep the best individuals with quality solutions;
        Rank the solutions;
        Population←Current best individuals;
        Archive←∅;
        for S_i ∈Population
            S_i_objectives ←CalculateObjectives(S_i);
        end
        Union←Population+Archive;
        for S_i ∈Union
        S_i_raw ←CalculateRawFitness(S_i,Union);
        S_i_density ←CalculateSolutionDensity(S_i,Union);
        S_i_fitness ←S_i_raw +S_i_density ;
        end
        Archive←GetNonDominated(Union);
    if Size(Archive)<Archive_size
        PopulateWithRemainingBest(Union,Archive,Archive_size);
    elseif Size(Archive)>Archive_size
        RemoveMostSimilar(Archive,Archive_size);
        end
        Return(GetNonDominated(Archive));
    end
    Post-process result (Archive) and visualization;
end
```

In this study, all the modified and hybrid CS algorithms are proposed and deployed to perform SO and MO optimization simulations for linear antenna array synthesis with $2N$ = 10 or 20 or 30 symmetric elements based on design specifications as enlisted in Table 3.1 below. All the modified and hybrid CS algorithms in SO optimization should accomplish the desired average SLL suppression and null mitigation values in both uniform and Dolph–Chebyshev current amplitude distributions. Besides, all the newly proposed CS algorithms in weighted–sum and Pareto front MO approaches should fulfill the displayed average SLL suppression, null mitigation, antenna directivity improvement, and HPBW reduction in both uniform and Dolph–Chebyshev current filtering windows.

Table 3.1: Design Parameter Specification

Uniform Amplitude Distribution				
No. of Elements	***Average SLL***	***Null***	***Directivity***	***HPBW***
$2N$ = 10	–35 dB ≤ SLL ≤ –15 dB	≤ –35 dB	≥ 6.0 dB	≤ 12^o
$2N$ = 20	–35 dB ≤ SLL ≤ –15 dB	≤ –35 dB	≥ 6.0 dB	≤ 08^o
$2N$ = 30	–35 dB ≤ SLL ≤ –15 dB	≤ –35 dB	≥ 6.0 dB	≤ 04^o
Dolph–Chebyshev (Relative SLL, R = –30 dB) Amplitude Distribution				
No. of Elements	***Average SLL***	***Null***	***Directivity***	***HPBW***
$2N$ = 10	–40 dB ≤ SLL ≤ –25 dB	≤ –40 dB	≥ 7.0 dB	≤ 14^o
$2N$ = 20	–40 dB ≤ SLL ≤ –30 dB	≤ –40 dB	≥ 7.0 dB	≤ 07^o
$2N$ = 30	–40 dB ≤ SLL ≤ –30 dB	≤ –40 dB	≥ 7.0 dB	≤ 05^o

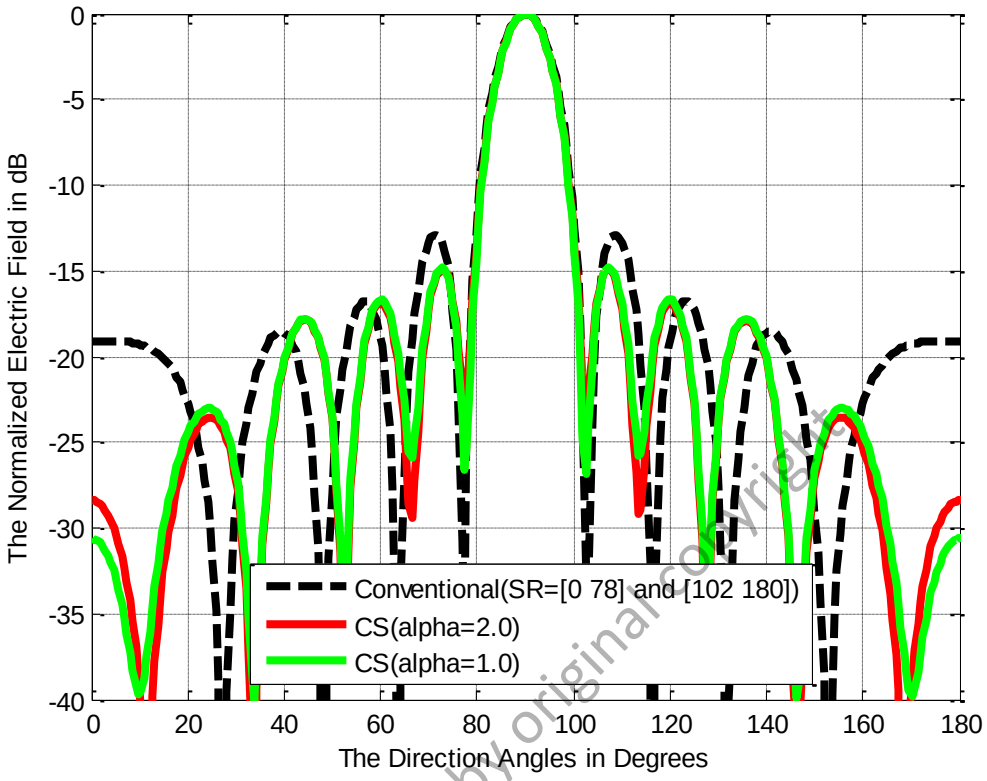
CHAPTER FOUR

SINGLE OBJECTIVE OPTIMIZATION

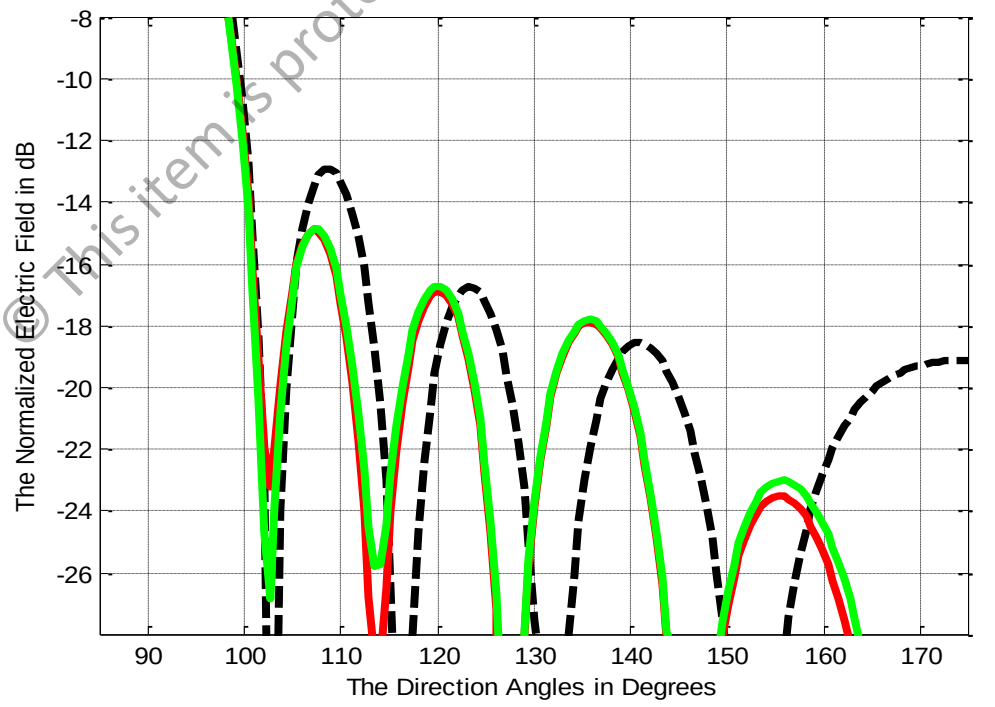
4.1 The Preliminary Study on Cuckoo Search Algorithm Internal Parameters

Firstly, there is an analysis on the Lévy flights distribution type or α parameter. The CS–optimizer with the host nest (population) = 20, fraction probability, P_a = 25% or 0.25, length step factor = L/100 or 0.01, along with both α = 1.0 (Lévy flight Cauchy distribution), and α = 2.0 (Lévy flight Gaussian distribution) are examined, respectively.Figure 4.1(a) – (b) depict two normalized patterns for $2N$ = 10 linear array in which the $\lambda / 2$ inter−element distance are optimized by the original CS algorithm. Figure 4.2 and Figure 4.3 shows the polar pattern for both linear arrays with α = 2.0 and α = 1.0, respectively. The normalized and polar patterns are identical for both cases primarily due to the same minimum fitness or f_{min} convergence, which is about 0.01185 after simulating 500 iterations.

Based on Figure 4.4, as α smaller, the convergence rate becomes faster. In this case, the CS–optimizer with Mantegna's algorithm shows that for α = 1.0, about 212 iterations are needed to achieve the convergence whereas for α = 2.0 about 304 iterations are needed, respectively. Moreover, for α = 2.0, bigger location fluctuations occur in an attempt to search for the optimal solutions before 465 iterations.



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.1: Normalized Pattern for α Comparison ($2N$ = 10, Uniform, maxIter = 500)

Table 4.1: Optimal Location for α Comparison ($2N$ = 10, Uniform, maxIter = 500)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [α = 2.0]	±0.4881	±1.4793	±2.4626	±3.4937	±4.5702
CS [α = 1.0]	±0.5060	±1.5235	±2.5196	±3.5552	±4.6319

In addition, Table 4.1 above enlists the optimal locations for both CS−optimizers with the Lévy flight Gaussian and Cauchy distributions, which have differences between them less than |±0.1000| for all the $2N$ = 10 array elements. In other aspect, the CS−optimizer with the Gaussian distribution generates the bigger optimal location deviations compared to the conventional array between |±0.0060| and |±0.1319|. In contrast, the CS−optimizer with the Cauchy distribution has the smaller variations between |±0.0119| and |±0.0702|. In short, the CS−optimizer with the Gaussian distribution generates a bigger diversity in terms of optimal array element locations through a further exploration in search space producing a better SLL suppression.

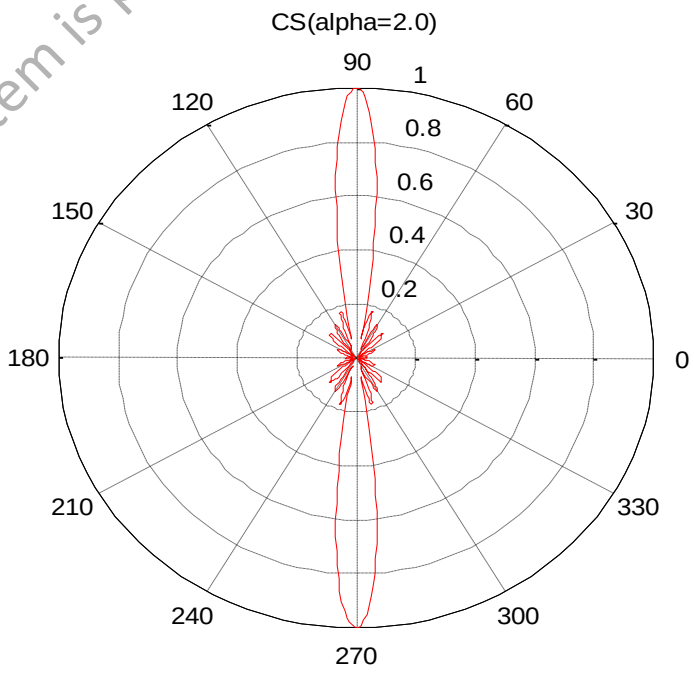


Figure 4.2: Polar Pattern for CS−based Array
($2N$=10, Gaussian, Uniform, maxIter = 500)

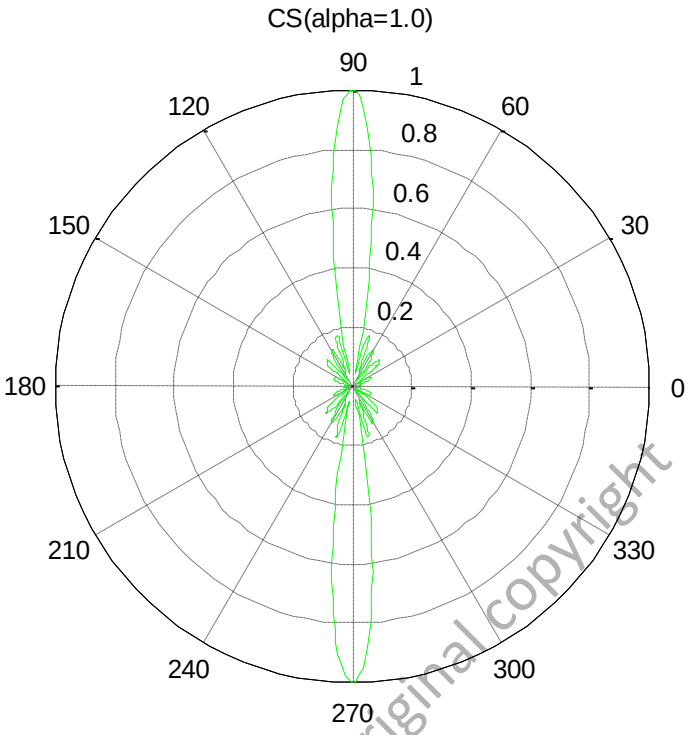


Figure 4.3: Polar Pattern for CS−based Array
(2*N*=10, Cauchy, Uniform, maxIter = 500)

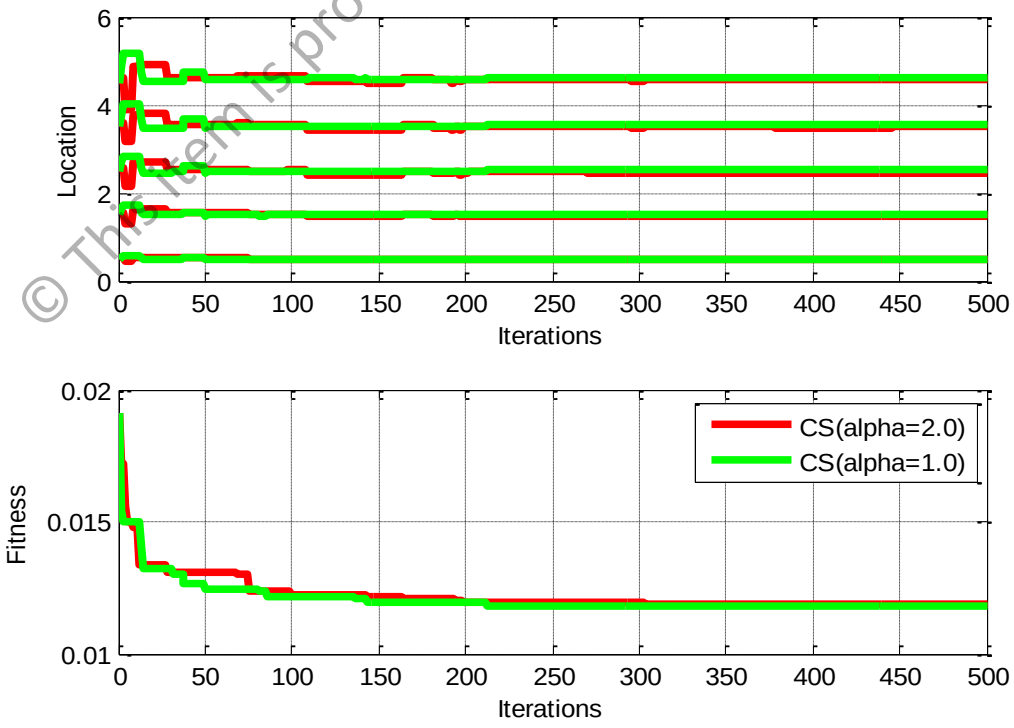
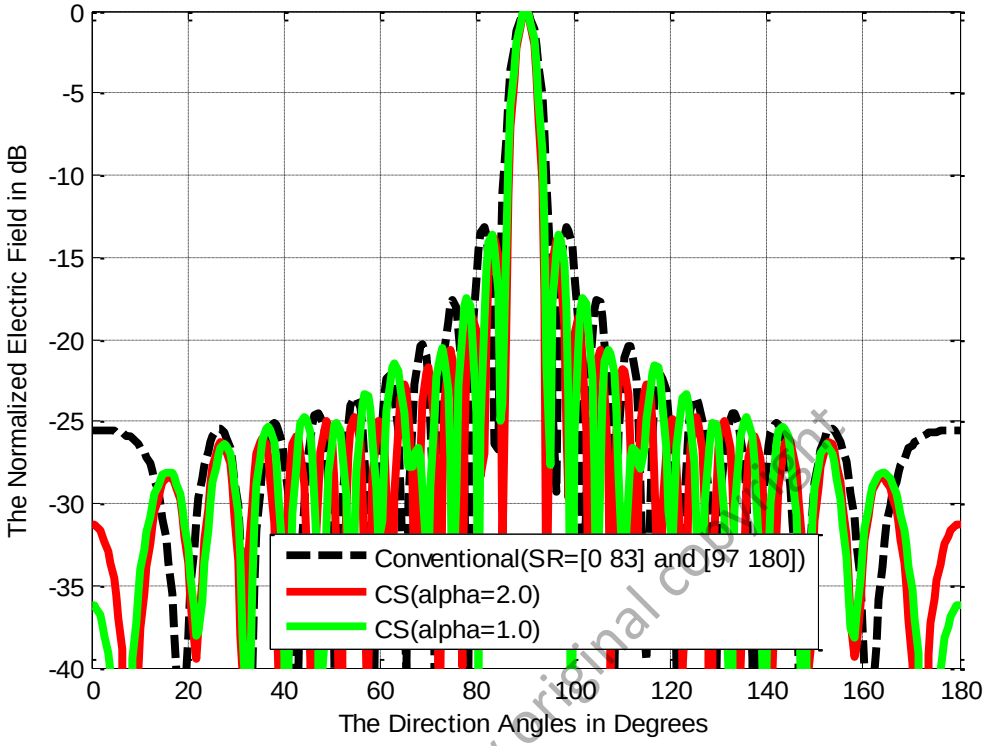
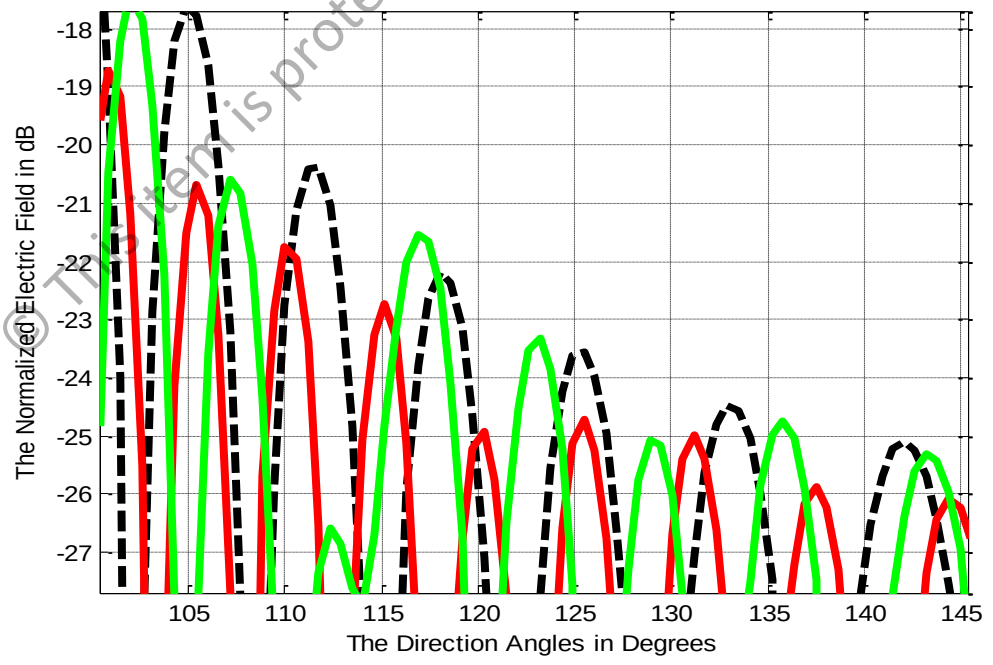


Figure 4.4: Location and Fitness Curves for α Comparison
(2*N*=10, Uniform, maxIter = 500)

Then, there is also an analysis on the effect of α using the $2N = 20$ linear arrays through a MATLAB simulation with 5000 iterations. For a standardization, the CS−optimizer with Mantegna's algorithm parameters, e.g. nest (population) = 20, discovery rate, P_a = 25% or 0.25, length step factor = $L/100$ or 0.01, along with both α = 1.0 (Cauchy distribution), and α = 2.0 (Gaussian distribution) are maintained, respectively. Figure 4.5(a) – (b) show that the CS−optimizer with α = 2.0 performs better by having a lower SLL suppression, e.g. between 0.5 dB and 3.5 dB lower than the CS−optimizer with α = 1.0 within [30° 83°] and [97° 150°] domains, respectively. In addition, Figure 4.6 shows that the CS−optimizer with α = 2.0 had bigger location fluctuations, and a lower f_{min} convergence of 0.0145 after 4400 iterations. Moreover, Table 4.2 enlists the CS−optimizer with the Gaussian distribution generated the bigger optimal location deviations compared to the conventional array between |±0.1653| and |±3.1577|. In sum, the CS algorithm with the Gaussian α–stable distribution has a better diversity via larger fluctuations in optimal excitation locations for the linear array with larger number of elements.



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.5: Normalized Pattern for α Comparison ($2N$ = 20, Uniform, maxIter = 5000)

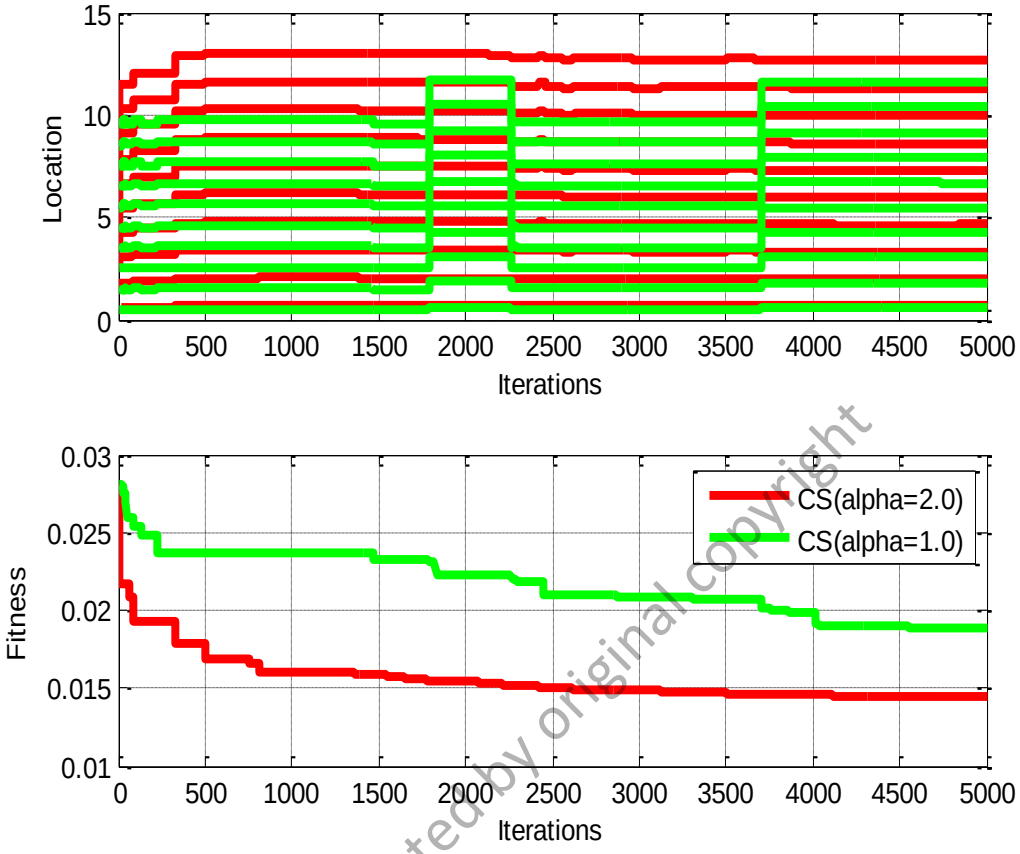
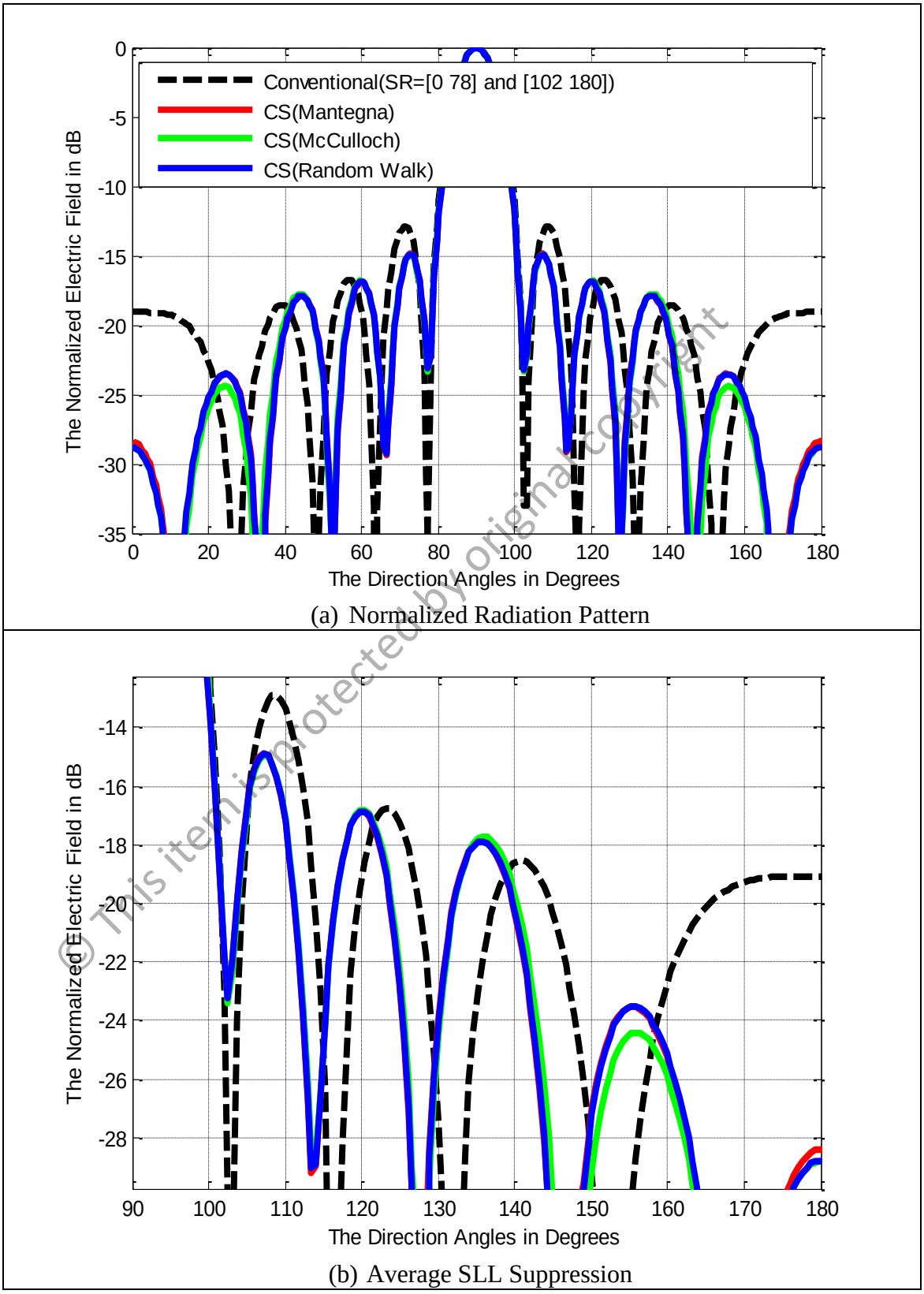


Figure 4.6: Location and Fitness Curves for α Comparison
($2N = 20$, Uniform, maxIter = 5000)

Table 4.2: Optimal Location for α Comparison ($2N = 20$, Uniform, maxIter = 5000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [α = 2.0]	±0.6653	±1.9971	±3.3248	±4.6470	±5.9711
CS [α = 1.0]	±0.6047	±1.8184	±3.0369	±4.2449	±5.4526
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [α = 2.0]	±7.2922	±8.6136	±9.9481	±11.2996	±12.6577
CS [α = 1.0]	±6.6861	±7.9265	±9.1531	±10.3858	±11.6309

Secondly, there is an experiment on the imperative effect of three different α–stable distribution methods. The distribution methods are Mantegna's algorithm, McCulloch's algorithm, and standard random walk, which are tested on both $2N = 10$ and $2N = 20$ linear arrays.



(a) Normalized Radiation Pattern

(b) Average SLL Suppression

Figure 4.7: Normalized Pattern for Distribution Type Comparison
($2N$ = 10, Uniform, maxIter = 500)

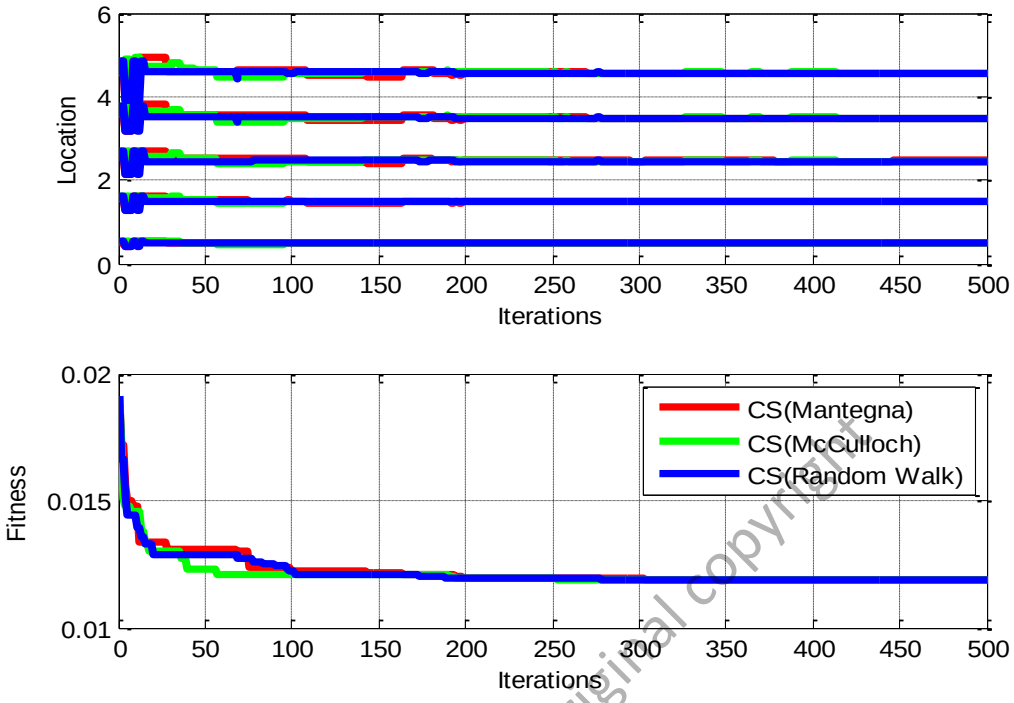
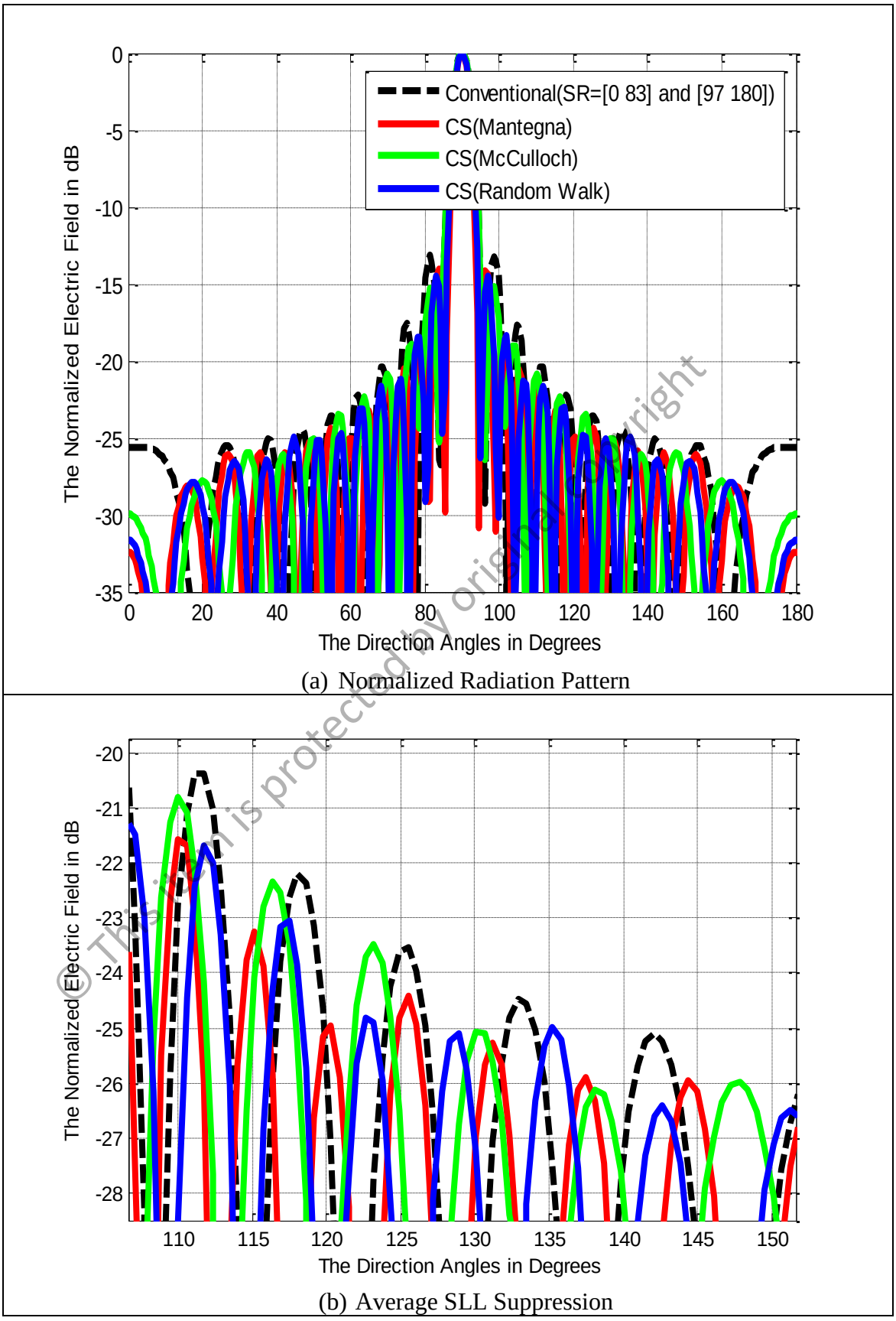


Figure 4.8: Location and Fitness Curves for Distribution Type Comparison
($2N$ = 10, Uniform, maxIter = 500)

Table 4.3: Optimal Location Distribution Type Comparison
($2N$ = 10, Uniform, maxIter = 500)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.4881	±1.4793	±2.4626	±3.4937	±4.5702
CS [McCulloch]	±0.4862	±1.4750	±2.4521	±3.4777	±4.5488
CS [Random Walk]	±0.4875	±1.4773	±2.4591	±3.4888	±4.5641

Based on Figure 4.7(a) – (b), all the three α–stable distribution types, which are applied in $2N$ = 10 symmetric linear array have almost similar SLL suppression. This is due to the same fitness convergence attainment of 0.011913 after about 330 iterations as depicted in Figure 4.8. Furthermore, Table 4.3 enlists the optimal locations for all CS−optimizers with three different distribution types, which are nearly identical with differences of less than |±0.1000| for all elements.



(a) Normalized Radiation Pattern

(b) Average SLL Suppression

Figure 4.9: Normalized Pattern for Distribution Type Comparison
($2N$ = 20, Uniform, maxIter = 10000)

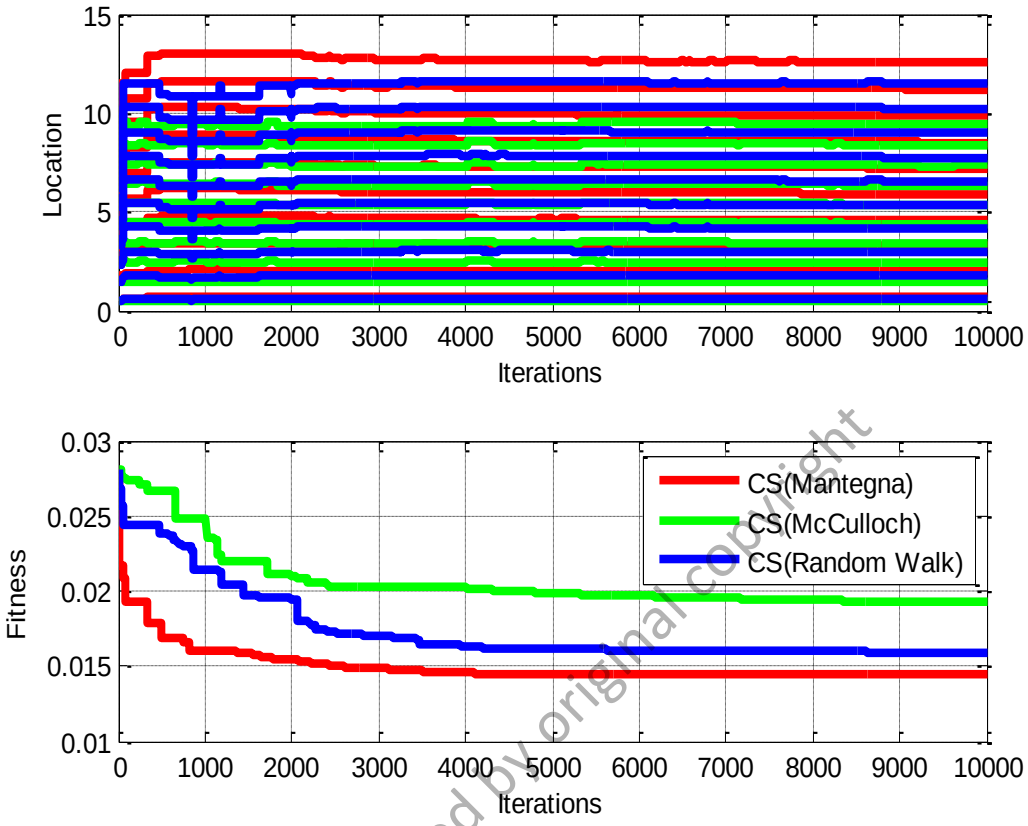


Figure 4.10: Location and Fitness Curves for Distribution Type Comparison ($2N$ = 20, Uniform, maxIter = 10000)

Table 4.4: Optimal Location Distribution Type Comparison ($2N$ = 20, Uniform, maxIter = 10000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.6513	±1.9552	±3.2598	±4.5679	±5.8822
CS [McCulloch]	±0.4921	±1.4735	±2.4534	±3.4288	±4.4025
CS [Random Walk]	±0.5959	±1.7850	±2.9713	±4.1561	±5.3480
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Mantegna]	±7.1960	±8.5146	±9.8490	±11.2008	±12.5595
CS [McCulloch]	±5.3787	±6.3612	±7.3618	±8.3931	±9.4448
CS [Random Walk]	±6.5493	±7.7592	±8.9860	±10.2343	±11.4929

However, as the number of element increases to 20, the Mantegna's algorithm clearly has the lowest SLL suppression compared to other two counterparts. Based on Figure 4.9, the Mantegna's algorithm suppresses 0.5 dB to 2.5 dB relatively lower than

the McCulloch's algorithm and standard random walk distributions within the [0° 83°] and [97° 180°] domains, respectively. The Mantegna's algorithm produces the best result due to the capability of finding further optimal solutions (excitation locations) in a search space, hence has a better diversity of optimal solutions. This is proved by the bigger fluctuations and lower f_{min} convergence attainment as depicted in Figure 4.10. In this study, the Mantegna's algorithm has the f_{min} convergence of 0.0144, which is followed by the standard random walk with the f_{min} convergence of 0.0159, and the McCulloch's algorithm with the f_{min} convergence of 0.0193, respectively.

Table 4.4 shows that the Mantegna's algorithm generates the optimal locations with more than |±0.2000| differences compared to other rivals for all elements. With bigger fluctuations, the Mantegna's algorithm has the optimal locations deviate between |±0.1513| and |±3.0595| from the conventional array after 10000 iterations.

Thirdly, there is a study on the imperative effect of the length step factor or L towards the linear array beam pattern where the Mantegna's algorithm with $\alpha = 2.0$ is used constantly. Three Lévy flights step factors, which are L/10 or 0.1, L/100 or 0.01, and L/1000 or 0.001 are tested. Precisely, L is the length scale of cuckoo's motion in searching for a new (other host bird's) nest for a brood parasitism purpose. Figure 4.11 and Figure 4.13(a) – (b) clearly depict that the normalized patterns for both $2N = 10$ and $2N = 20$ arrays are uniform regardless the Lévy flights step factors. This confirms that all the length step factors attain the same array element positions and f_{min} convergence. Moreover, all the length step factors ensure the same new nests (potential solutions) are discovered within the search space. This can be seen in Figure 4.12 where all the three length step factors have the f_{min} convergence of 0.011911 for $2N = 10$ array, and the f_{min} convergence of 0.014524 for $2N = 20$ array as shown in Figure 4.14, respectively.

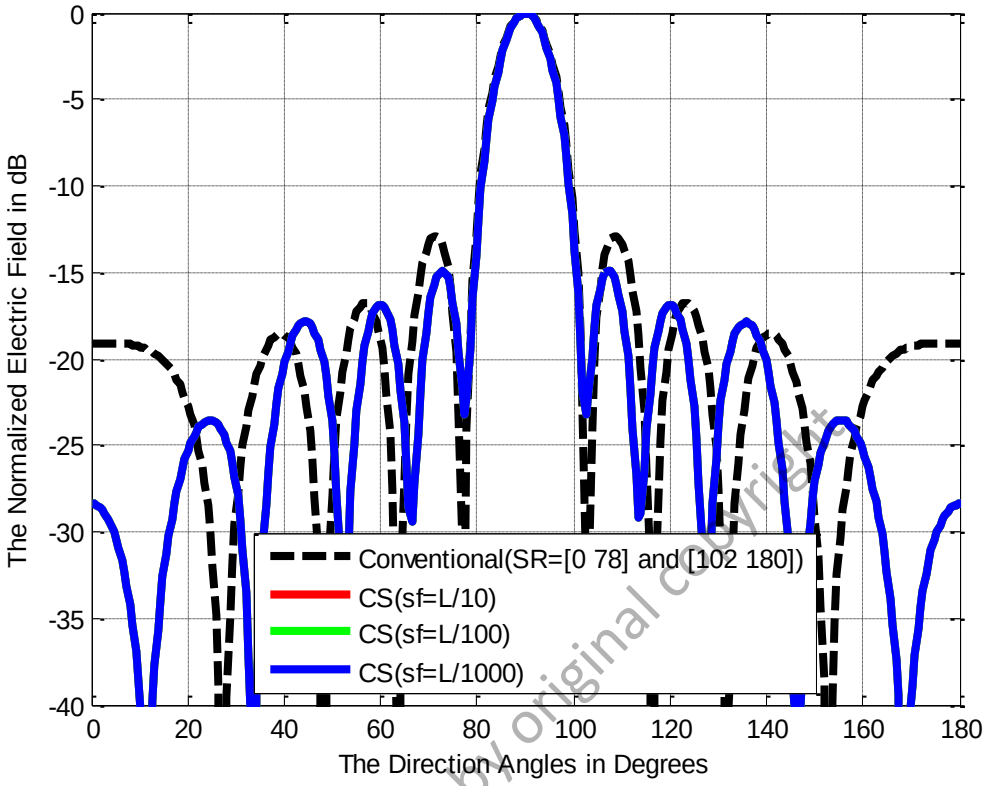


Figure 4.11: Normalized Pattern Step Factor Comparison
($2N$ = 10, Uniform, maxIter = 500)

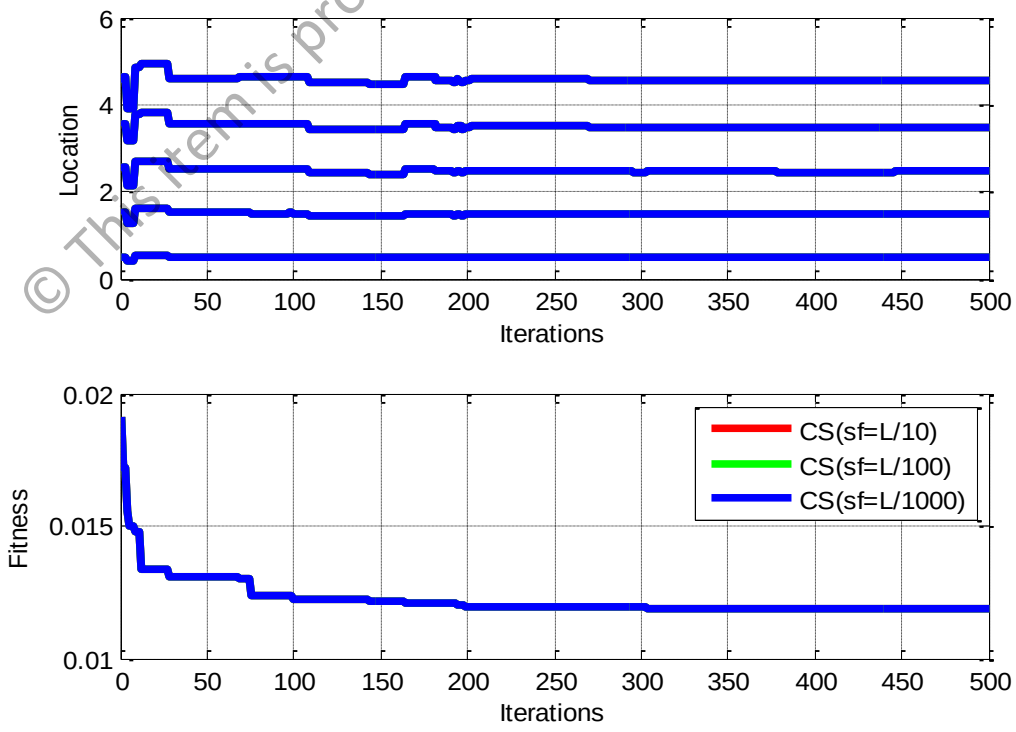
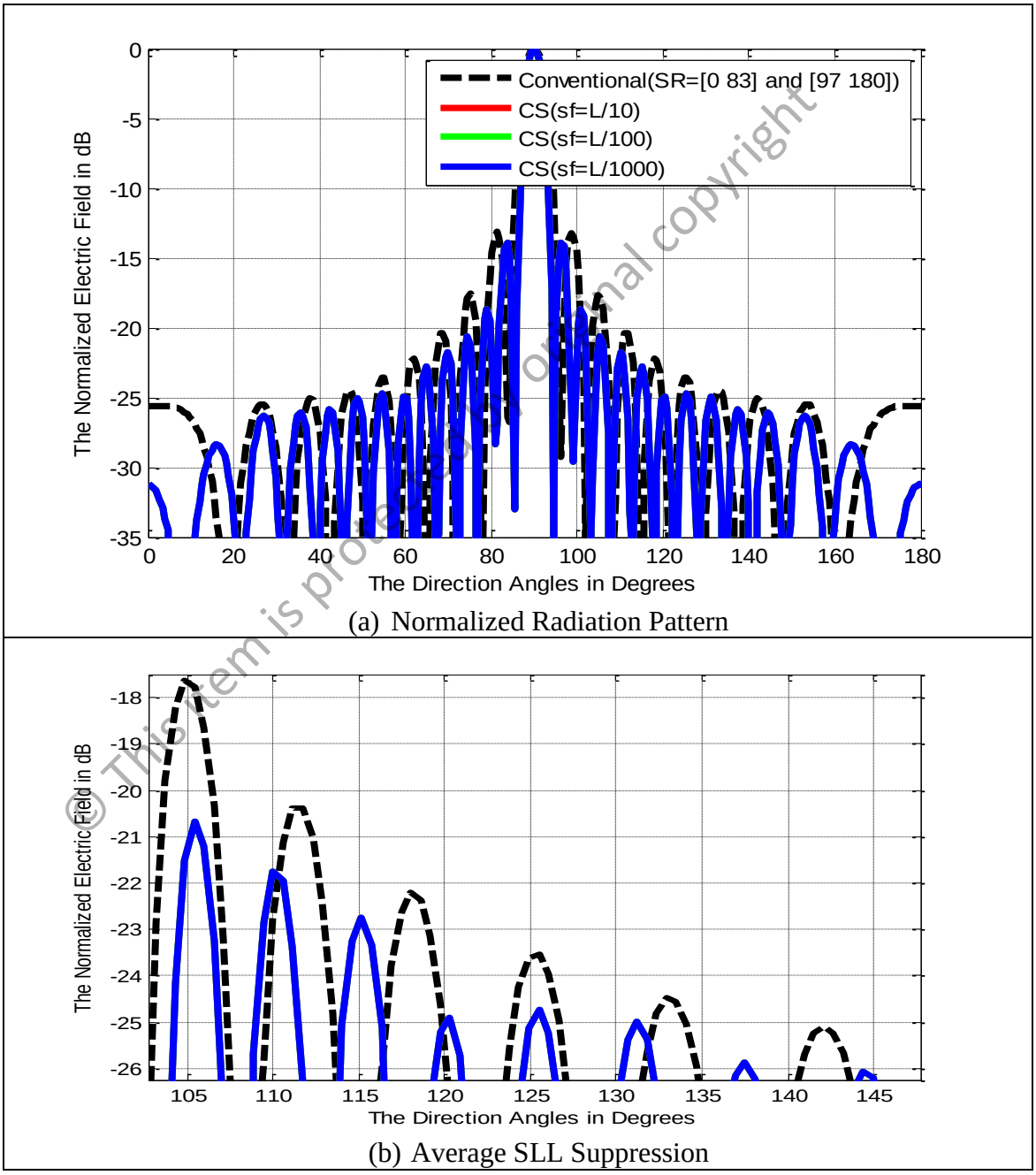


Figure 4.12: Location and Fitness Curves for Step Factor Comparison
($2N$ = 10, Uniform, maxIter = 500)

Table 4.5: Optimal Location for Step Factor Comparison
(2*N* = 10, Uniform, maxIter = 500)

Element	**1**	**2**	**3**	**4**	**5**
X_n [$\lambda/2$]	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [*sf* = *L*/10]	±0.4881	±1.4793	±2.4626	±3.4937	±4.5702
CS [*sf* = *L*/100]	±0.4881	±1.4793	±2.4626	±3.4937	±4.5702
CS [*sf* = *L*/1000]	±0.4881	±1.4793	±2.4626	±3.4937	±4.5702



(a) Normalized Radiation Pattern

(b) Average SLL Suppression

Figure 4.13: Normalized Pattern for Step Factor Comparison
(2*N* = 20, Uniform, maxIter = 5000)

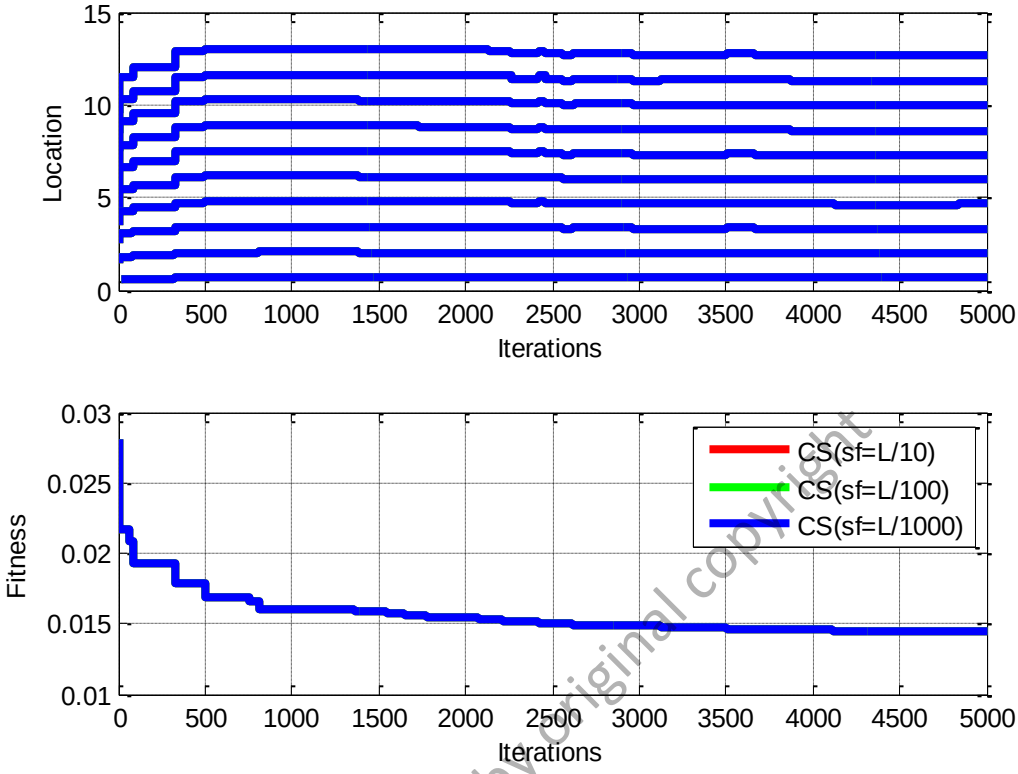


Figure 4.14: Location and Fitness Curves for Step Factor Comparison ($2N$ = 20, Uniform, maxIter = 5000)

Table 4.6: Optimal Location vs. Step Size Factor ($2N$ = 20,Uniform, maxIter = 5000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [*sf* = *L*/10]	±0.6653	±1.9971	±3.3248	±4.6470	±5.9710
CS [*sf* = *L*/100]	±0.6653	±1.9971	±3.3248	±4.6470	±5.9710
CS [*sf* = *L*/1000]	±0.6653	±1.9971	±3.3248	±4.6470	±5.9710
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [*sf* = *L*/10]	±7.2922	±8.6136	±9.9481	±11.2996	±12.6577
CS [*sf* = *L*/100]	±7.2922	±8.6136	±9.9481	±11.2996	±12.6577
CS [*sf* = *L*/1000]	±7.2922	±8.6136	±9.9481	±11.2996	±12.6577

Since all the length step factors have the identical f_{min} convergence for both $2N$ = 10, and $2N$ = 20 linear arrays, the optimal solutions (element excitation locations) are also same as stated in both Table 4.5, and Table 4.6, respectively. Precisely, for $2N$ = 10 linear array, all the length step factors deviate optimal locations less than

|±0.1000| compared to the conventional array. In addition, for $2N$ = 20 linear array, all the length step factors deviate optimal locations between |±0.1653| and |±3.1577| compared to the conventional array. In sum, the use of any length step factor or L, which is less than 1.0 generates the same outcome and most importantly, prohibits the excessive Lévy flight motions performed by cuckoo in finding new nests (potential solutions) within the search space. As a result, the f_{min} convergence and optimal solutions are identical regardless any length step factors applied.

Fourthly, there is a study on the imperative effect of the number of host nest (population) on the symmetric linear array. In this experiment, the CS–optimizer with the discovery rate, P_a = 0.25, Mantegna's stable algorithm, α = 2.0, and length step factor = L/100 or 0.01 are simulated along with three different sizes of host nest (population) = 10, 20, and 30, respectively.

It is found that as the number of host nest bigger, the convergence rate becomes faster. This is due to the higher capability and larger probability in finding global minimum solutions since a bigger number of populations occupy in the search space. As shown in Figure 4.15, the performances for $2N$ = 10 linear arrays are identical as the same f_{min} convergence is achieved regardless the number of host nest.

However, based on Figure 4.16, we can see the differences of the convergence rate where the CS–optimizer with the nest = 30 achieves the f_{min} convergence after just 130 iterations, nest = 20 around 230 iterations, and nest = 10 about 380 iterations, respectively. In addition, the fluctuations of optimal solutions stabilize first as optimal solutions are achieved for nest = 30 (after about 270 iterations). This is followed by nest=20 (after around 310 iterations) and nest = 10 (roughly after 430 iterations), respectively. In sum, the optimal solutions stability is found directly related to the f_{min} convergence rates of the CS–optimizer.

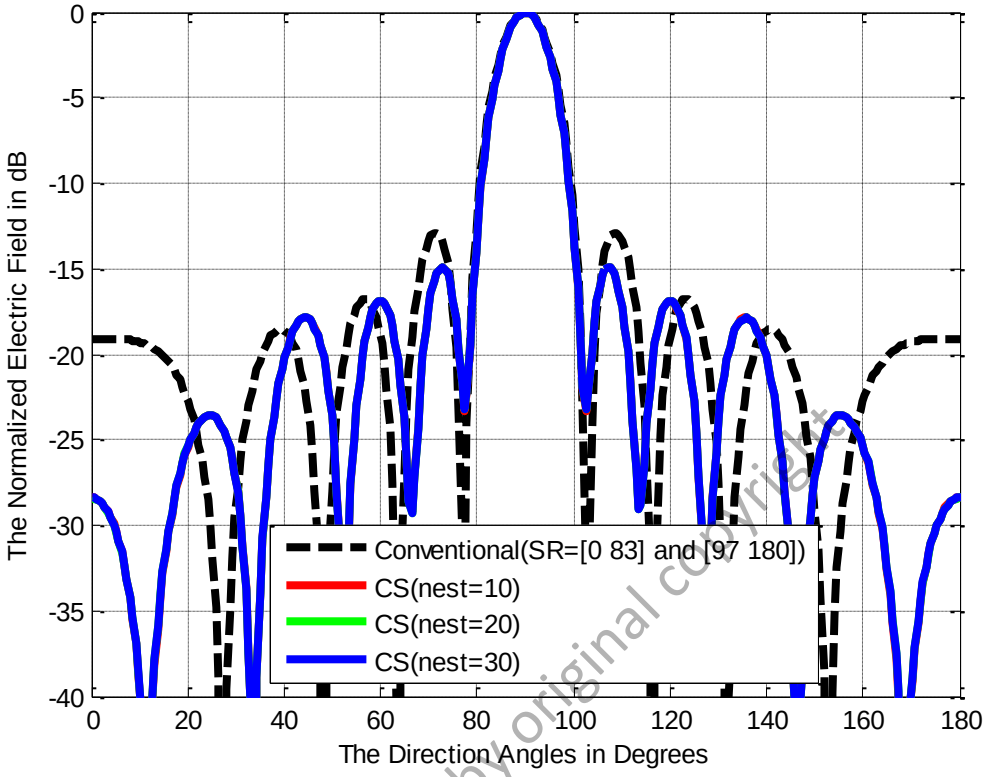


Figure 4.15: Normalized Pattern vs. Population ($2N$ = 10, Uniform, maxIter = 500)

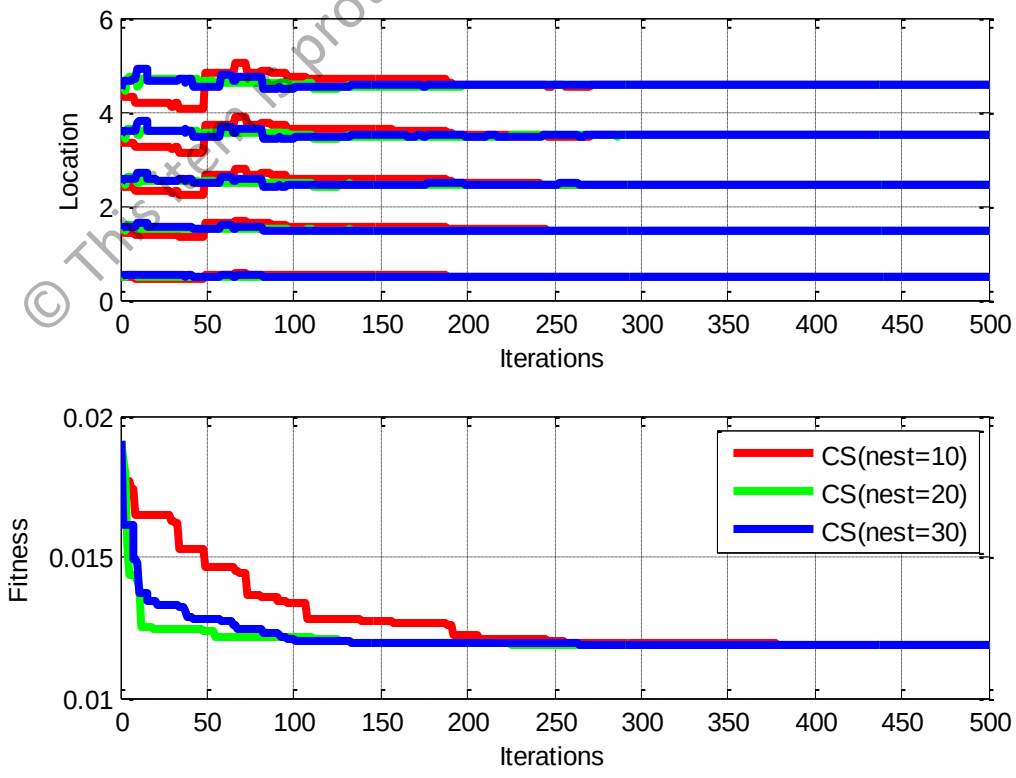


Figure 4.16: Location and Fitness Curves ($2N$ = 10, Uniform, maxIter = 500)

Table 4.7: Optimal Location vs. Population ($2N$ = 10, Uniform, maxIter = 500)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [nest = 10]	±0.4888	±1.4813	±2.4651	±3.4962	±4.5725
CS [nest = 20]	±0.4881	±1.4794	±2.4624	±3.4932	±4.5694
CS [nest = 30]	±0.4881	±1.4795	±2.4625	±3.4933	±4.5695

As each of population sizes has the identical f_{min} convergence of 0.011911, the optimal solutions (array element locations) are also almost similar. This can be seen in Table 4.7, where all the population sizes have the small differences of location among them, which are less than |±0.0100|. Besides, all the three tested CS–optimizers generate the optimal solution (element location with respect to λ/2) deviations less than |±0.1000| compared to the conventional linear array.

Fifthly, an investigation is done to analyze the fraction probability (P_a) or discovery rate imperative effect on the linear array. Three different P_a values, which are 0.25 (discovery rate of 25%), 0.50 (discovery rate of 50%), and 0.95 (discovery rate of 95%) are compared in the $2N$ = 10 linear array. As shown in Figure 4.17(a) – (b), both the CS–optimizers with P_a = 0.25 and P_a = 0.50 have the best identical performances whereas the CS–optimizer with P_a = 0.95 is the worst one in suppressing side lobes. In this case, both the CS–optimizers with P_a = 0.25 and P_a= 0.50 suppress the side lobes relatively between 0.40 dB and 3.35 dB lower than the CS–optimizer with P_a= 0.95.

Besides to that, both P_a = 0.05 and P_a = 0.25 have similar f_{min} convergence too, which is about 0.0119 after 300 iterations. On the other hand, P_a = 0.95 has a bigger f_{min} convergence of around 0.0136 after 130 iterations as depicted in Figure 4.18.

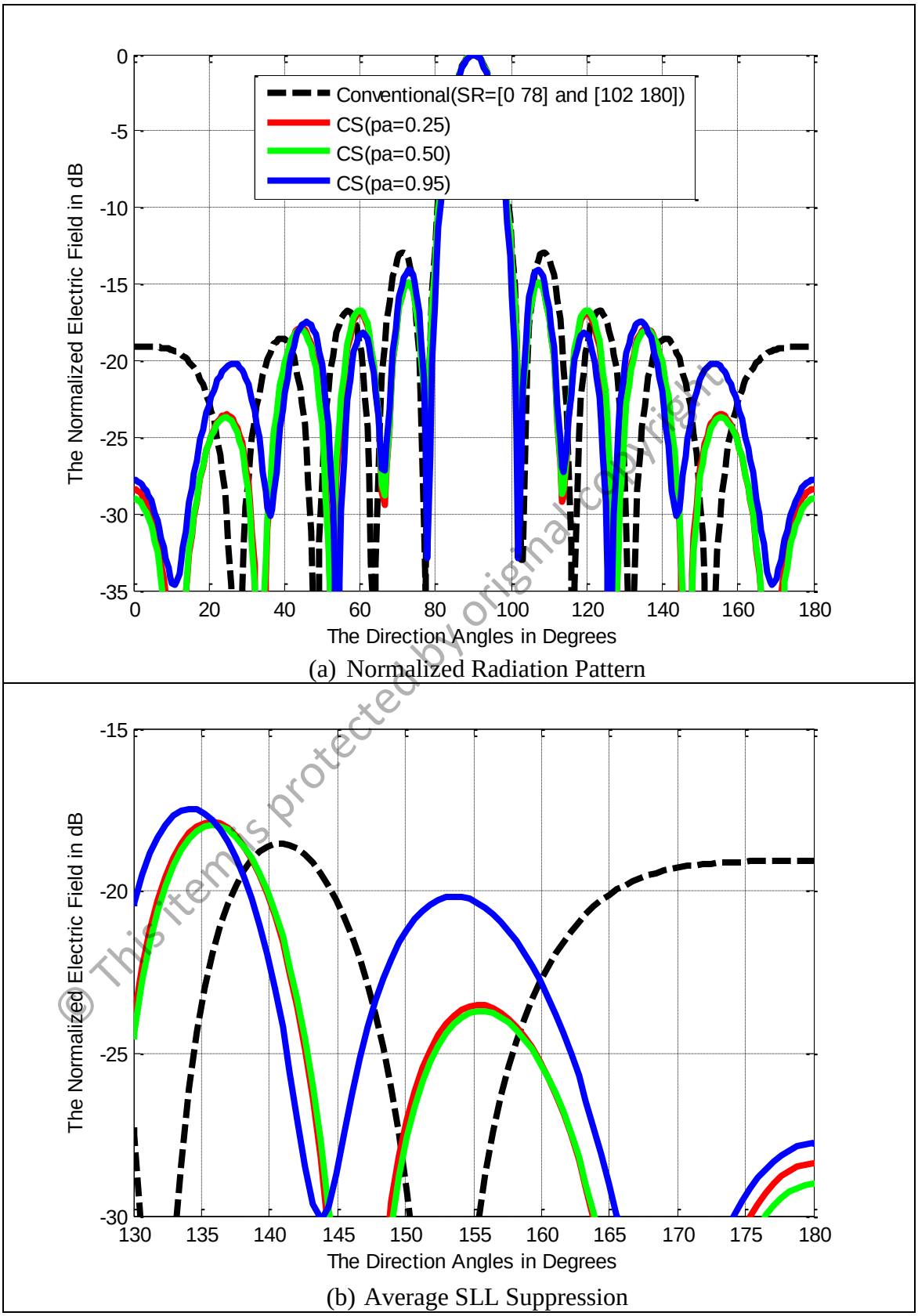


Figure 4.17: Normalized Pattern for P_a Comparison ($2N$ = 10, Uniform, maxIter = 500)

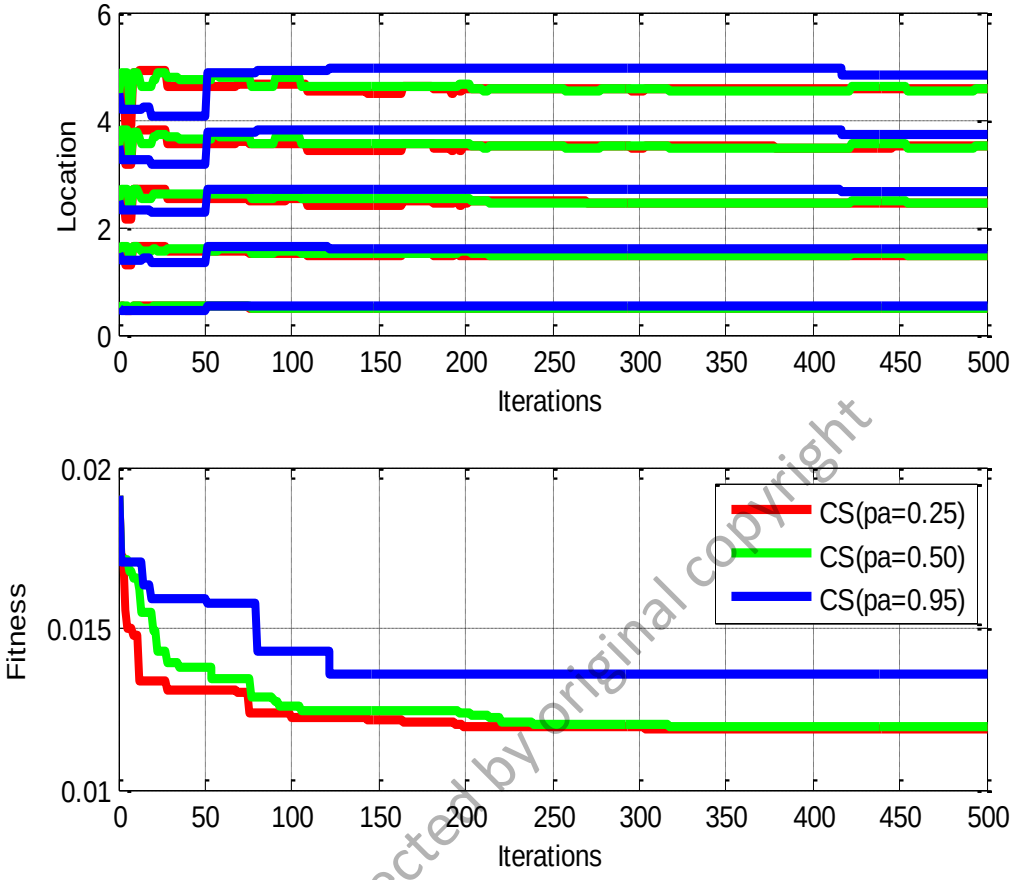


Figure 4.18: Location and Fitness Curves for P_a Comparison ($2N$ = 10, Uniform, maxIter = 500)

Table 4.8: Optimal Location for P_a Comparison ($2N$ = 10,Uniform, maxIter = 500)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [P_a = 0.25]	±0.4881	±1.4793	±2.4626	±3.4937	±4.5702
CS [P_a = 0.50]	±0.4890	±1.4795	±2.4600	±3.4895	±4.5638
CS [P_a = 0.95]	±0.5273	±1.5962	±2.6451	±3.7088	±4.8086

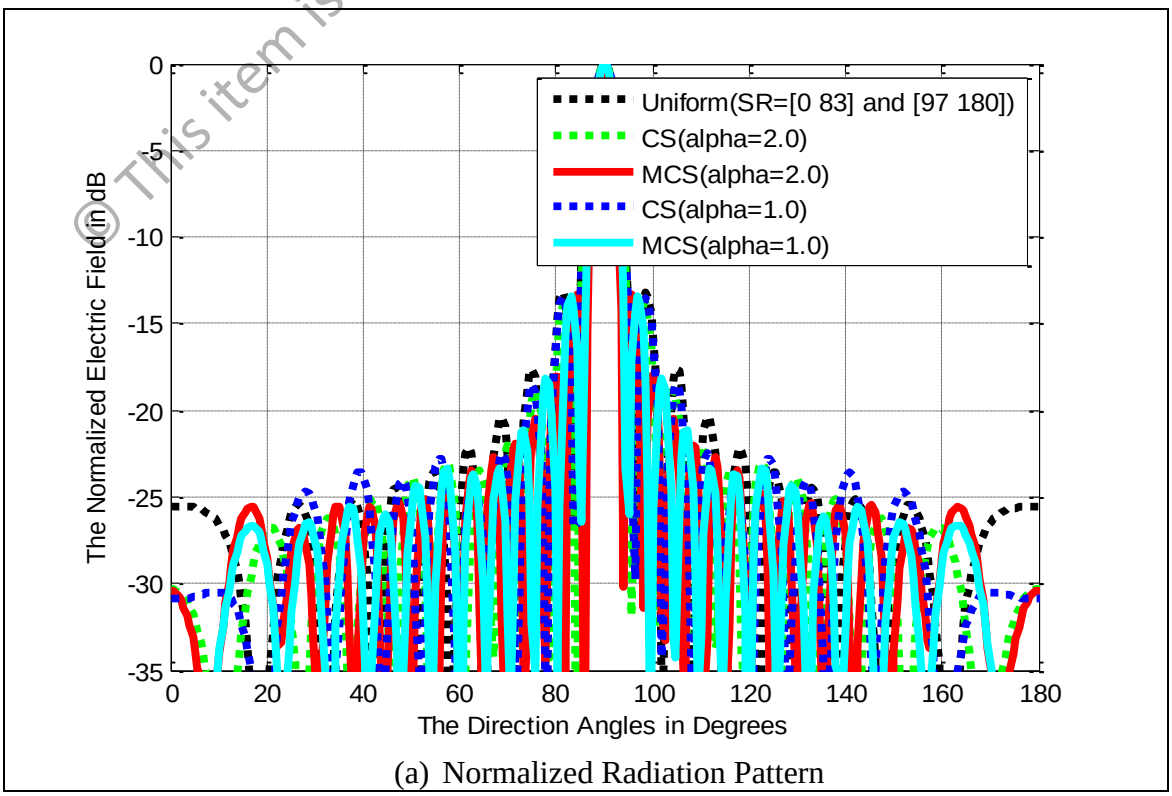
Based on Table 4.8, both the CS–optimizers with P_a = 0.25 and P_a = 0.50 have almost similar optimal locations with the deviations of less than |±0.1000| compared to the conventional linear array for all $2N$ = 10 array elements. In contrast, the CS–optimizer with P_a = 0.95 generates the optimal location deviations with the differences between |±0.0273| and |±0.3086| compared to the conventional linear array.

Overall, this agrees with the statistical assumption that as the P_a bigger, the possibility of an egg laid by a cuckoo to be discovered by the host bird of other species becomes higher. As a result, the cuckoo's egg (candidate solution) will be abandoned or thrown away leading to a new nest searching or a new nest replacement. Ideally, the fittest cuckoo's egg (best optimal solution) survival should be preserved where the egg hatching and brood–parasitism processes well–taken by the host bird unknowingly.

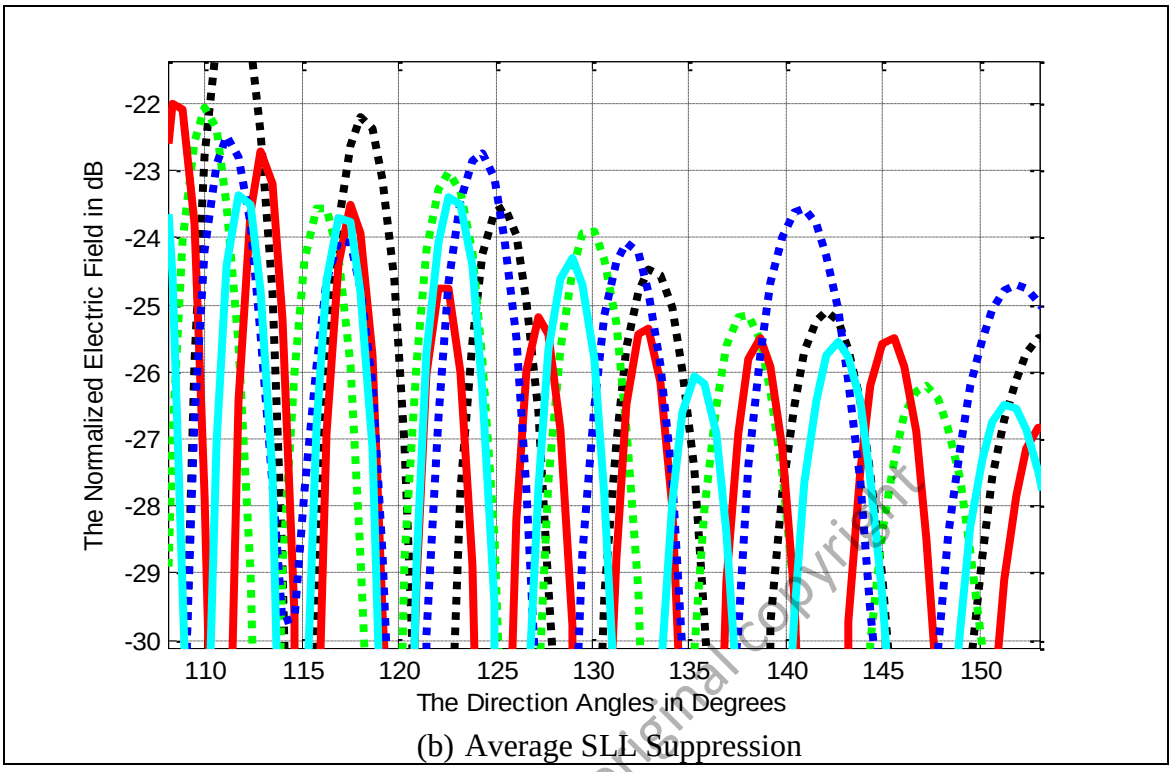
4.2 The Postulation of Modified Cuckoo Search Algorithm in Linear Antenna Array Synthesis

Firstly, the characteristic of α is analyzed thoroughly. In this case, both CS and MCS–optimizers employ Mantegna's algorithm as the selected α–stable distribution method, host nest = 30, P_a = 25% or 0.25, length step factor = L/100 or 0.01, along with both α = 1.0 (Lévy flight Cauchy distribution), and α = 2.0 (Lévy flight Gaussian distribution) are examined on the $2N$ = 20 antenna array, respectively. Both of the tested MCS–optimizers have the adaptive w magnitude domain of [0.95 1.00].

As illustrated in Figure 4.19(a) – (b), the MCS–optimizer with the Gaussian distribution (α = 2.0) slightly outperforms other optimizers in SLL suppression. It is closely followed by the MCS counterpart with α = 1.0, and then by both CS–optimizers, respectively. Specifically, the MCS algorithm with α = 2.0 generates SLL between 0.5 dB and 1.5 dB relatively lower than the conventional array in the [20° 130°] and [97° 160°] domains.



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.19: Normalized Pattern for CS vs. MCS in α Value
($2N$ = 20, Uniform, maxIter = 2000)

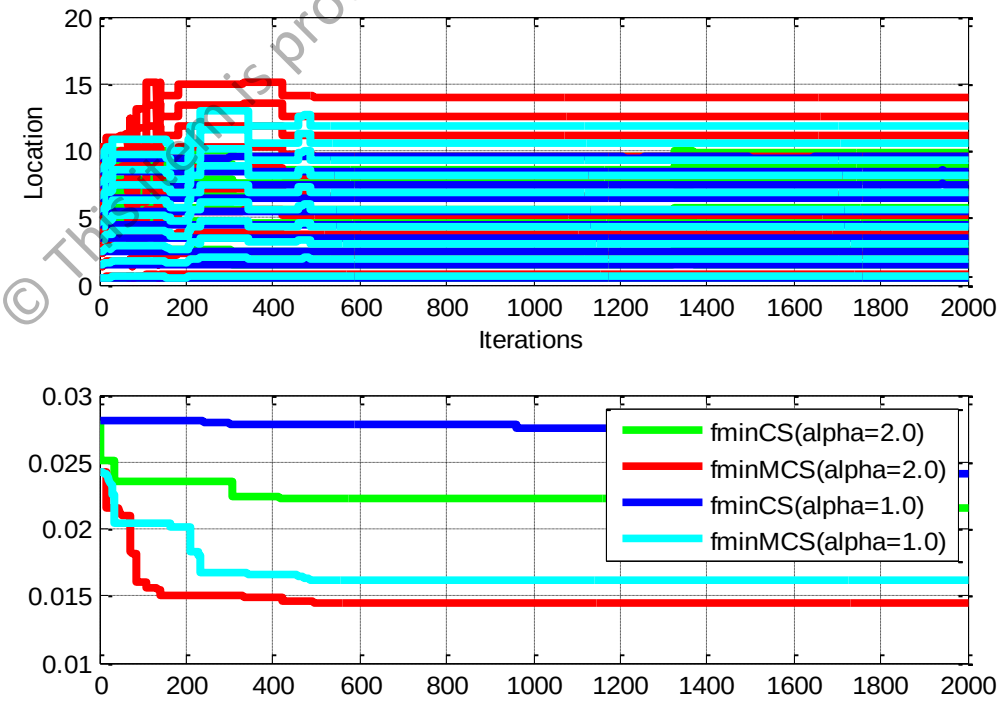


Figure 4.20: Location and Fitness Curves for CS vs. MCS in α Value
($2N$ = 20, Uniform, maxIter = 2000)

Based on Figure 4.20, the proposed MCS algorithm (α = 2.0) produces the biggest oscillation of locations before it stabilizes after about 500 iterations. This is followed by the MCS counterpart (α = 1.0), which fluctuates and stabilizes after approximately 500 iterations. The postulated MCS algorithm (α = 2.0) achieves the minimum fitness, f_{min} convergence after around 500 iterations with the lowest value corresponds to 0.0145. This is trailed by the MCS algorithm (α = 1.0), which converges after about 500 iterations with the f_{min} of 0.0161 then, by the CS algorithm (α = 2.0) converges nearly after 450 iterations with the f_{min} of 0.0215, and finally, the CS algorithm (α = 1.0), which converges nearly 950 iterations with the f_{min} of 0.0242.

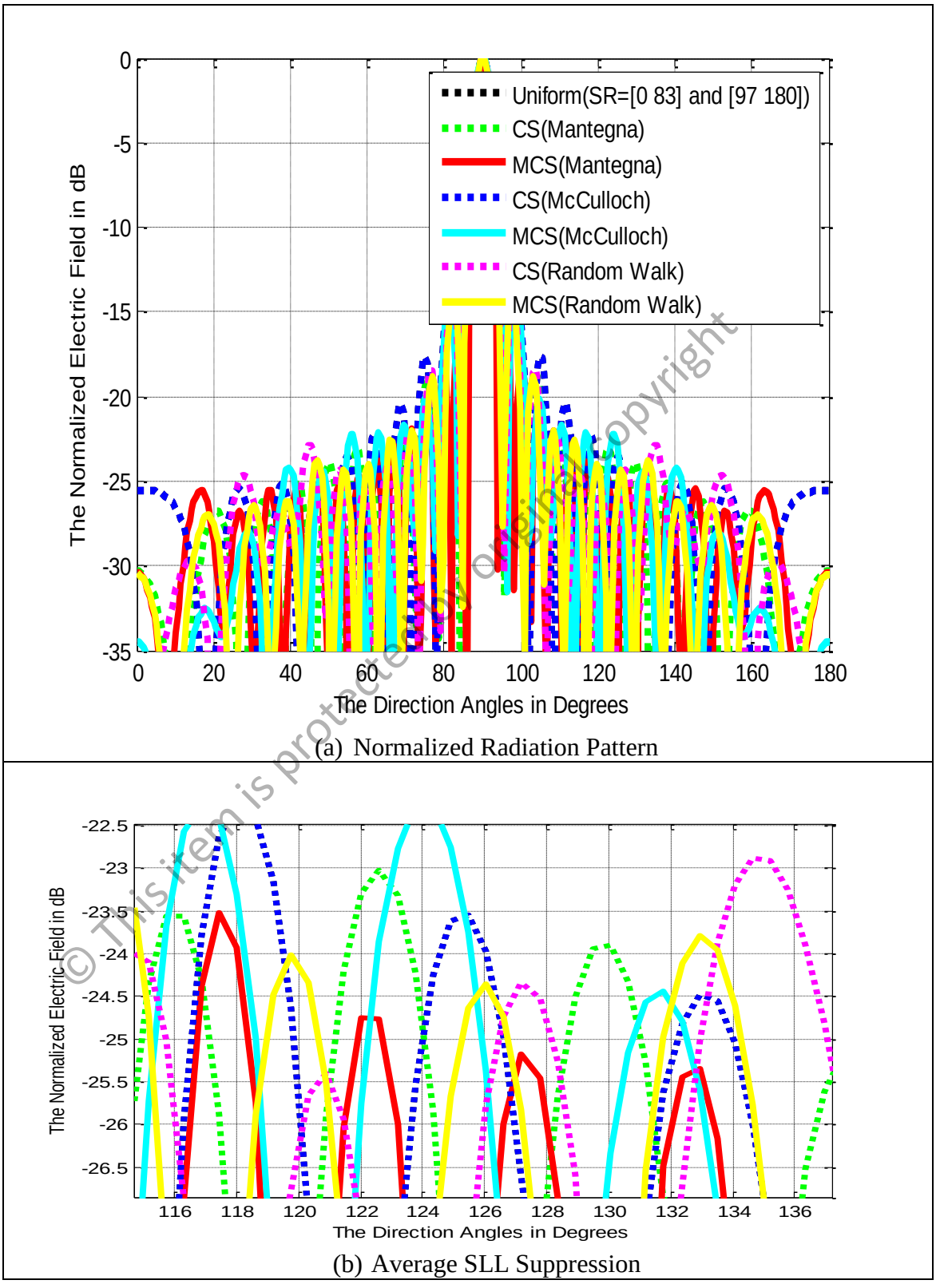
Table 4.9: Optimal Location for CS vs. MCS in α Value
($2N$ = 20, Uniform, maxIter = 2000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [α = 2.0]	±0.5177	±1.5535	±2.5898	±3.6245	±4.6596
MCS [α= 2.0]	±0.7386	±2.2139	±3.6916	±5.1681	±6.6451
CS [α = 1.0]	±0.5008	±1.5027	±2.5032	±3.4969	±4.4975
MCS [α = 1.0]	±0.6225	±1.8689	±3.1149	±4.3607	±5.6010
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [α = 2.0]	±5.6975	±6.7348	±7.7696	±8.8171	±9.8788
MCS [α= 2.0]	±8.1216	±9.5943	±11.0699	±12.5519	±14.0308
CS [α = 1.0]	±5.4970	±6.4975	±7.4876	±8.4890	±9.5047
MCS [α = 1.0]	±6.8461	±8.0908	±9.3314	±10.5810	±11.8378

Table 4.9 also indicates that the proposed MCS–optimizer with Lévy flight Gaussian distribution (α = 2.0) has the highest optimal location deviations between |±0.2386| and |±4.5308| compared to the conventional array. This is followed by the MCS–optimizer (α = 1.0) with the deviations between |±0.1225| and |±2.3378|. In sum, despite having the bigger size of antenna aperture in $2N$ = 20 linear array, the proposed MCS–optimizer could perform better than the CS–optimizer regardless α values. In other words, the MCS algorithm is capable to search further the optimal element

locations in N–dimensional space via Lévy flight motions. Consequently, this provides a bigger diversity of optimal solutions, which suppress side lobes radiation lower.

Secondly, there is an analysis on the imperative effect of three different α–stable distribution methods, which are Mantegna's algorithm, McCulloch's algorithm, and standard random walk for the broadside case (main beam direction at 90°) on $2N = 20$ linear array. All the tested MCS–optimizers deploy the Roulette wheel selection operator and adaptive weight, w with the magnitude domain of [0.95 1.00].



(a) Normalized Radiation Pattern

(b) Average SLL Suppression

Figure 4.21: Normalized Pattern for CS vs. MCS in Distribution Type
($2N$ = 20, Uniform, maxIter = 2000)

The MCS–optimizer with Mantegna's algorithm again produces the best SLL suppression between 0.1 dB – 3.0 dB lower than conventional antenna array within the [20° 83°] and [97° 160°] domains for $2N = 20$ linear array. The second best optimizer is the MCS algorithm (standard random walk) with the SLL suppression between 0.1 dB and 2.3 dB lower than the conventional antenna array as depicted in Figure 4.21(a) – (b).

Furthermore, the proposed MCS algorithm (Mantegna) executes the biggest oscillation of locations before it stabilizes after about 500 iterations as shown in Figure 4.22. This is followed by the MCS counterpart (standard random walk), which fluctuates and stabilizes after nearly 700 iterations. The postulated MCS algorithm (Mantegna) achieves the lowest f_{min} convergence of 0.0145 after about 500 iterations. This is followed by the MCS algorithm (standard random walk), which has f_{min} convergence of 0.0196 nearly after300 iterations, and the MCS algorithm (McCulloch) with f_{min} convergence of 0.0207 after 250 iterations, respectively. In contrast, the CS algorithm (McCulloch) has the highest f_{min} convergence of 0.0237 after 650 iterations.

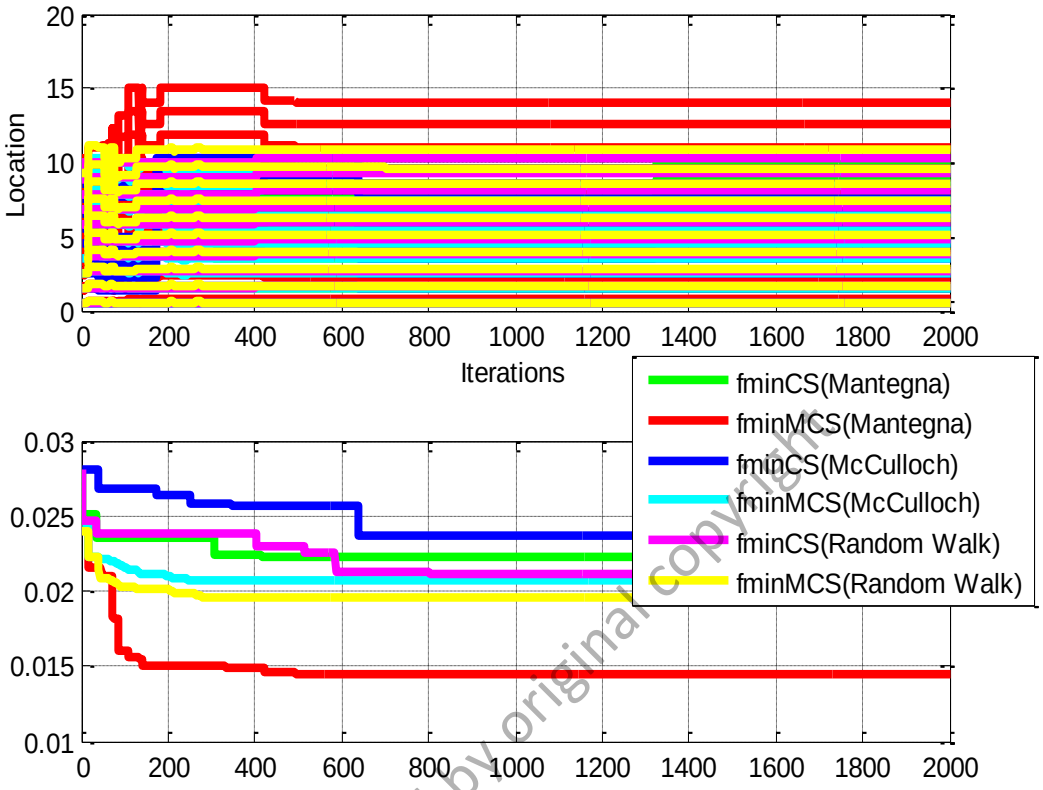


Figure 4.22: Location and Fitness Curves for CS vs. MCS in Distribution Type ($2N$ = 20, Uniform, maxIter = 2000)

As tabulated in Table 4.10, the proposed MCS–optimizer with Mantegna's algorithm has the highest optimal location fluctuations between |±0.2386| and |±4.5308| compared to the conventional array. This is followed by the MCS–optimizer (standard random walk) with the deviations between |±0.0661| and |±1.2922|. Overall, the hypothesized MCS–optimizer with Mantegna's algorithm as the α–stable distribution method outperforms other MCS and CS counterparts, specifically in SLL suppression for $2N$ = 20 symmetric array. Theoretically, this is mainly because the MCS–optimizer with Mantegna's algorithm produces the best diversity (highest variations) of optimal solution resulting from the far–reaching metaheuristic search.

Table 4.10: Optimal Location for CS vs. MCS in Distribution Type
(2*N* = 20,Uniform, maxIter = 2000)

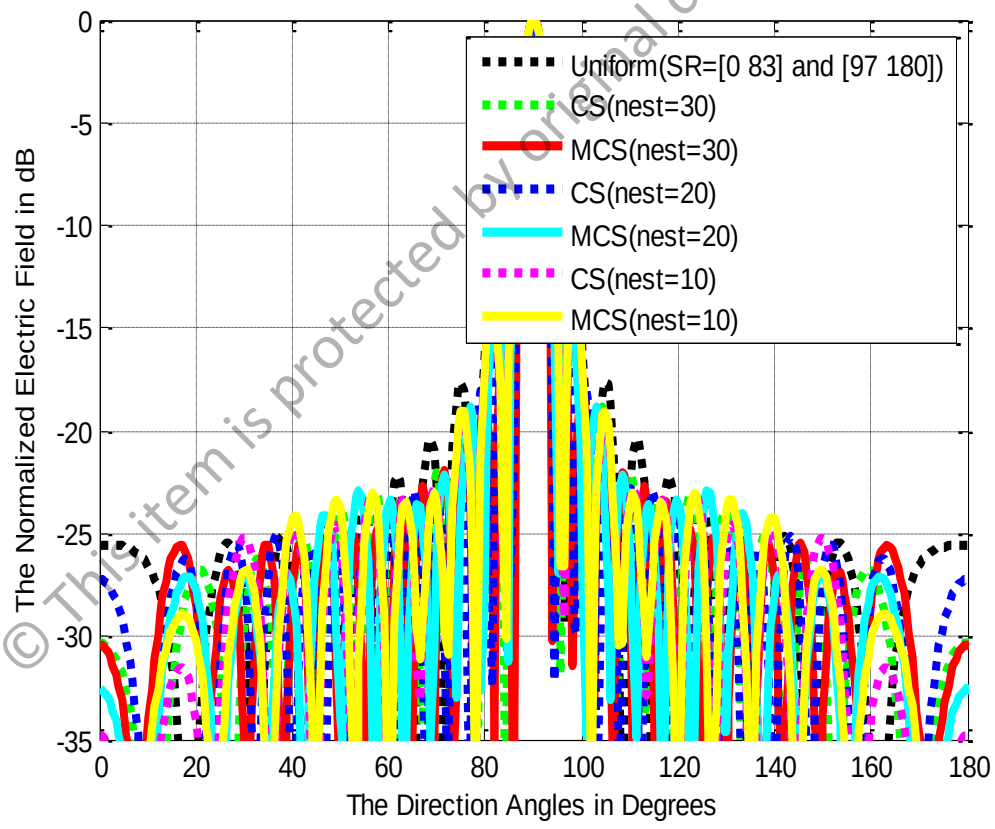
Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.5177	±1.5535	±2.5898	±3.6245	±4.6596
MCS [Mantegna]	±0.7386	±2.2139	±3.6916	±5.1681	±6.6451
CS [McCulloch]	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
MCS [McCulloch]	±0.4951	±1.4881	±2.4838	±3.4740	±4.4673
CS [Random Walk]	±0.5391	±1.6247	±2.6976	±3.7763	±4.8576
MCS [Random Walk]	±0.5661	±1.7008	±2.8327	±3.9705	±5.1088
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Mantegna]	±5.6975	±6.7348	±7.7696	±8.8171	±9.8788
MCS [Mantegna]	±8.1216	±9.5943	±11.0699	±12.5519	±14.0308
CS [McCulloch]	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
MCS [McCulloch]	±5.4527	±6.4382	±7.4213	±8.4286	±9.4538
CS [Random Walk]	±5.9408	±7.0347	±8.1215	±9.3029	±10.3420
MCS [Random Walk]	±6.2401	±7.3707	±8.4982	±9.6382	±10.7922

Thirdly, there is a study on the imperative effect of the number of host nest (population) for 2*N* = 20 linear array. In this study, all the modified CS and standard CS metaheuristic optimizers are simulated using the discovery rate, P_a = 25% or 0.25, Mantegna's algorithm, α = 2.0 (Lévy flight Gaussian distribution), and length step factor = *L*/100 or 0.01 along with three different population sizes (host nest = 10, 20, and 30), respectively.

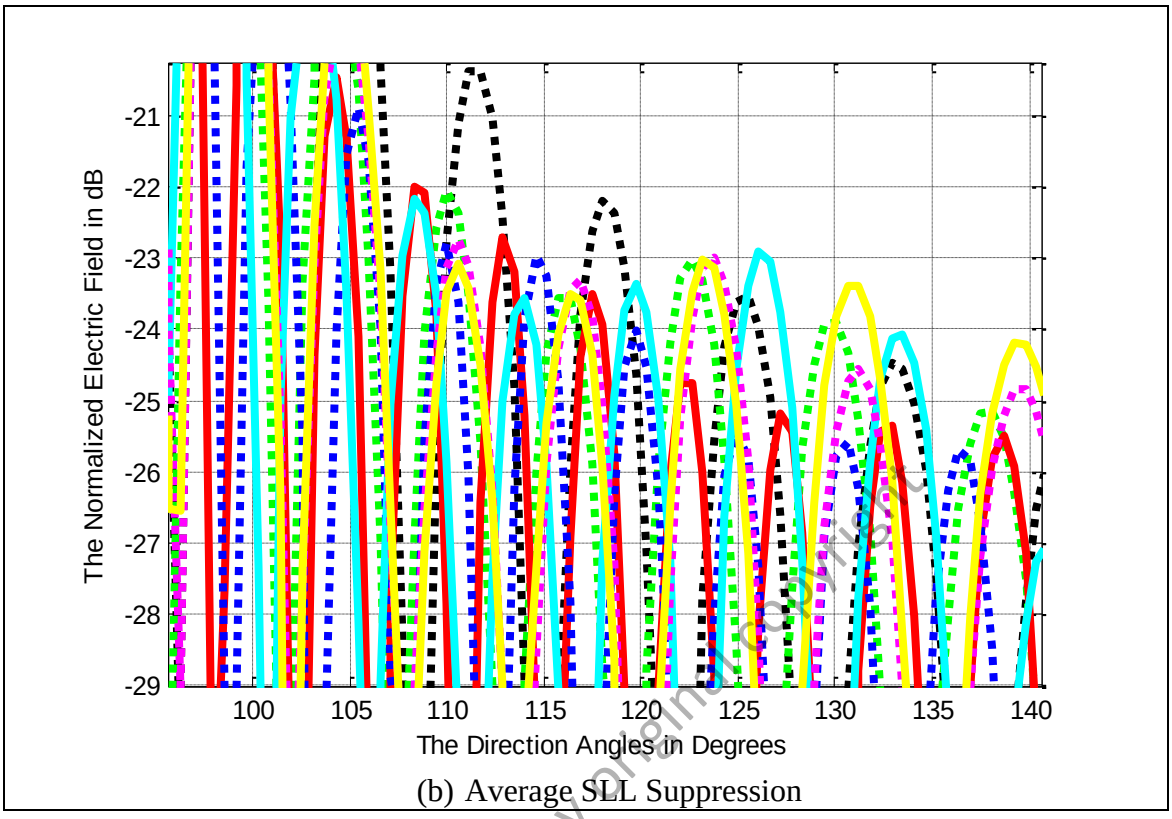
As shown in Figure 4.23(a) – (b), the proposed MCS–optimizers (nest = 20 and 30) generates approximately SLL of 5.0 dB lower than the MCS–optimizer (nest = 10), and the three standard CS–optimizers (nest= 10, 20, and 30) within the [0° 83°] and [97° 180°] suppression regions.It is found that the MCS–optimizer (host nest = 30) executes the best SLL suppression between 1.0 dB and 6.0 dB lower than conventional array within the [20° 83°] and [97° 160°] domains. This is followed by two other MCS–optimizers (host nest = 10 and 20).

Based on Figure 4.24, the proposed MCS algorithm (nest = 30) executes the biggest oscillation of locations before it stabilizes after 500 iterations. Moreover, the

postulated MCS algorithm (nest = 30) achieves the lowest f_{min} convergence of 0.0145. This proves that for a more complex synthesis (e.g. larger number of array elements), the MCS–optimizer with the highest number of population (nest = 30) delivers the best SLL suppression while maintaining the main beam. Precisely, with the larger number of population, the MCS–optimizer gains a higher possibility and more capability in exploring further optimal solutions within search domain through iterative searching process. Consequently, this produces a better diversity of optimal solutions.



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.23: Normalized Pattern for CS vs. MCS in Nest
(2*N* = 20, Uniform, maxIter = 2000)

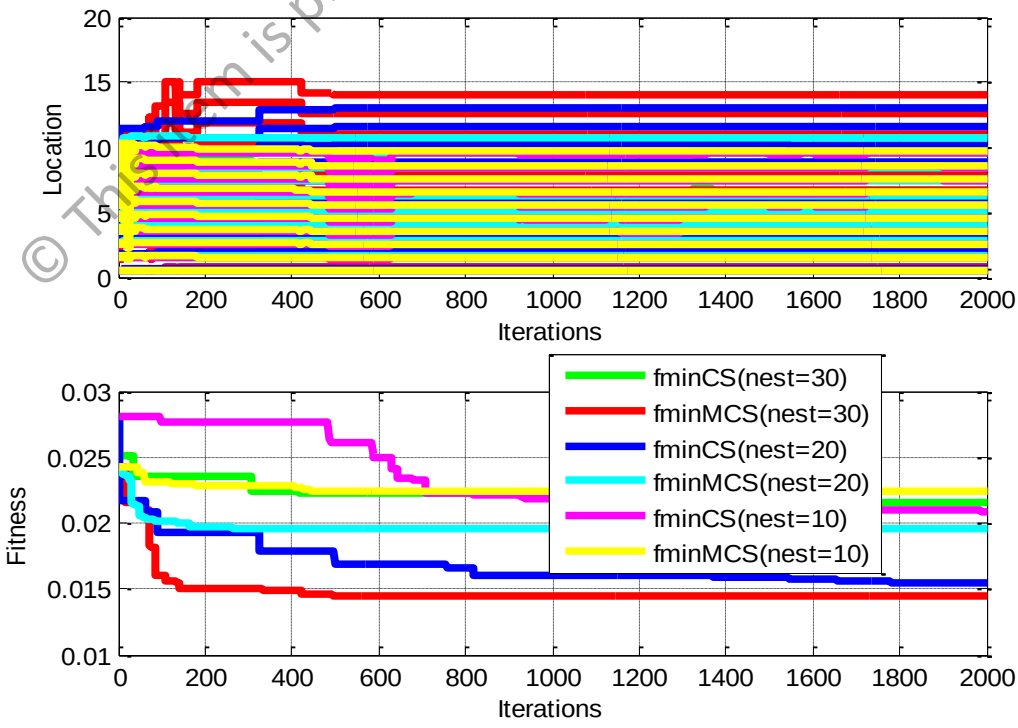


Figure 4.24: Location and Fitness Curves for CS vs. MCS in Nest
(2*N* = 20, Uniform, maxIter = 2000)

Table 4.11: Optimal Location for CS vs. MCS in Nest
($2N$ = 20, Uniform, maxIter = 2000)

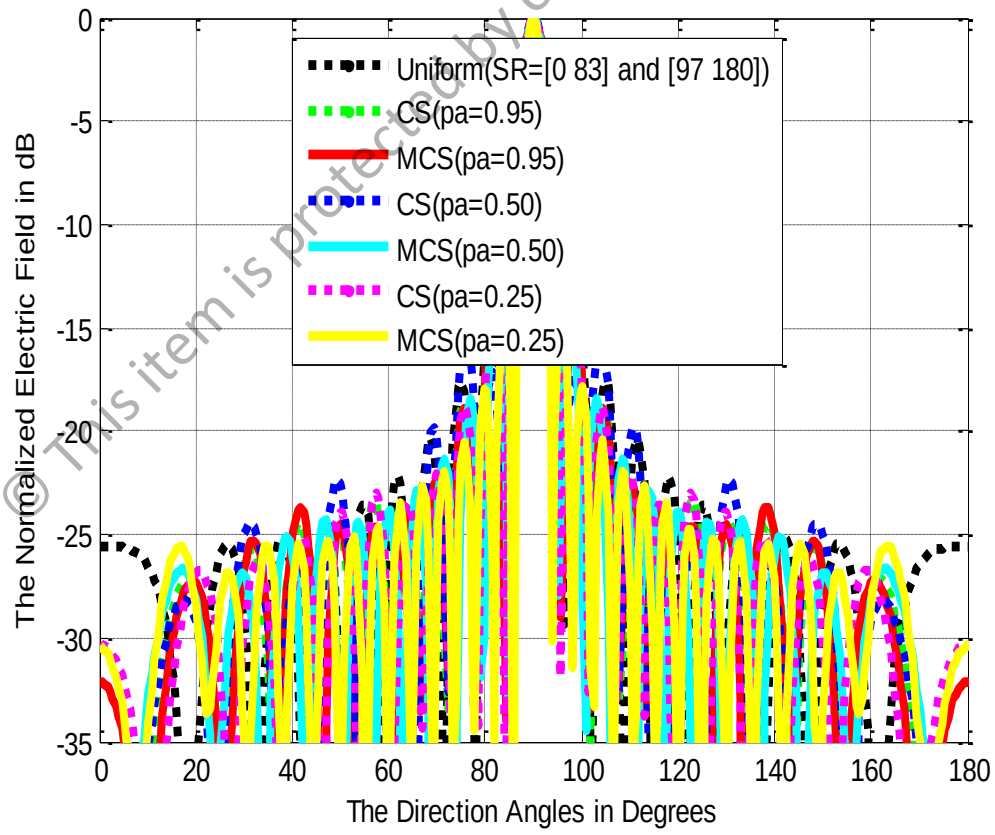
Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Nest = 30]	±0.5177	±1.5535	±2.5898	±3.6245	±4.6596
MCS [Nest = 30]	±0.7386	±2.2139	±3.6916	±5.1681	±6.6451
CS [Nest = 20]	±0.6813	±2.0445	±3.4066	±4.7634	±6.1229
MCS [Nest = 20]	±0.5647	±1.6952	±2.8248	±3.9567	±5.0849
CS [Nest = 10]	±0.4984	±1.4932	±2.4918	±3.4856	±4.4844
MCS [Nest = 10]	±0.5080	±1.5240	±2.5393	±3.5544	±4.5714
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Nest = 30]	±5.6975	±6.7348	±7.7696	±8.8171	±9.8788
MCS [Nest = 30]	±8.1216	±9.5943	±11.0699	±12.5519	±14.0308
CS [Nest = 20]	±7.4873	±8.8485	±10.2117	±11.5825	±12.9585
MCS [Nest = 20]	±6.2142	±7.3438	±8.4657	±9.6022	±10.7531
CS [Nest = 10]	±5.4810	±6.4787	±7.4727	±8.4822	±9.5128
MCS [Nest = 10]	±5.5867	±6.6011	±7.6053	±8.6234	±9.6594

As can be seen in Table 4.11, the MCS algorithm (nest = 30) has the biggest fluctuations of optimal radiator location with the differences between |±0.2386| and |±4.5308| compared to the conventional antenna array. This agrees with the simulation results in Figure 4.23(a) – (b), where the biggest deviations of optimal solution stimulates the postulated MCS–optimizer (nest = 30) to generate a better performance than other MCS and CS competitors.

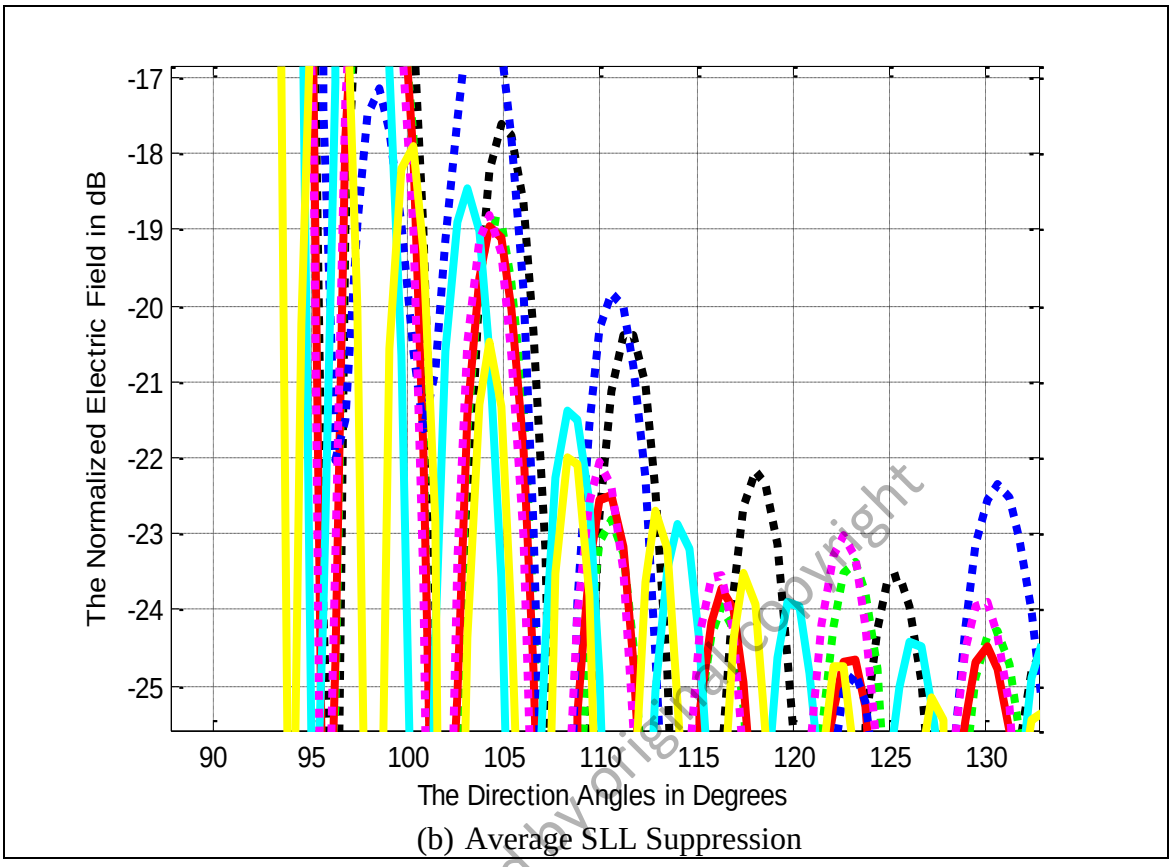
Fourthly, an investigation is also done on $2N$ = 20 symmetric array to find the imperative effect of discovery rate or fraction probability (discovery rate = 25% or P_a =0.25, discovery rate = 50% or P_a= 0.50, and discovery rate = 95% or P_a = 0.95). Figure 4.25(a) – (b) show that the postulated MCS–optimizer (P_a = 0.25) has the best SLL suppression between 0.12 dB and 2.85 dB lower than the conventional array within the suppression domains [20° 83°] and [97° 160°]. This is followed by the MCS algorithm (P_a = 0.50). Moreover, Figure 4.26 displays that both the proposed MCS algorithm (P_a = 0.25) generate the biggest location variations before reaching the constant optimum location after 500 iterations. This agrees with Table 4.12 where the

proposed MCS–optimizer (P_a = 0.25) has the biggest location deviations compared to the conventional array between |±0.2386| and |±4.5308|. Moreover, the MCS–optimizer (P_a = 0.25) attain the lowest f_{min} convergence. In sum, the proposed MCS–optimizer with the smaller P_a tends to have the lower f_{min} convergence and the bigger fluctuations of optimal location, thus, suppress the lower side lobes for $2N$ = 20 linear array.

Theoretically, this agrees with the CS (and so is MCS) algorithm assumption: Whenever discovery rate or fraction probability, P_a bigger, the possibility of egg laid by a cuckoo to be discovered by the host bird of other species becomes higher. As a result, the cuckoo's egg (candidate solution) could be abandoned or thrown away leading to a new host nest searching or replacement done by cuckoo.



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.25: Normalized Pattern for CS vs. MCS in P_a
($2N$ = 20, Uniform, maxIter = 2000)

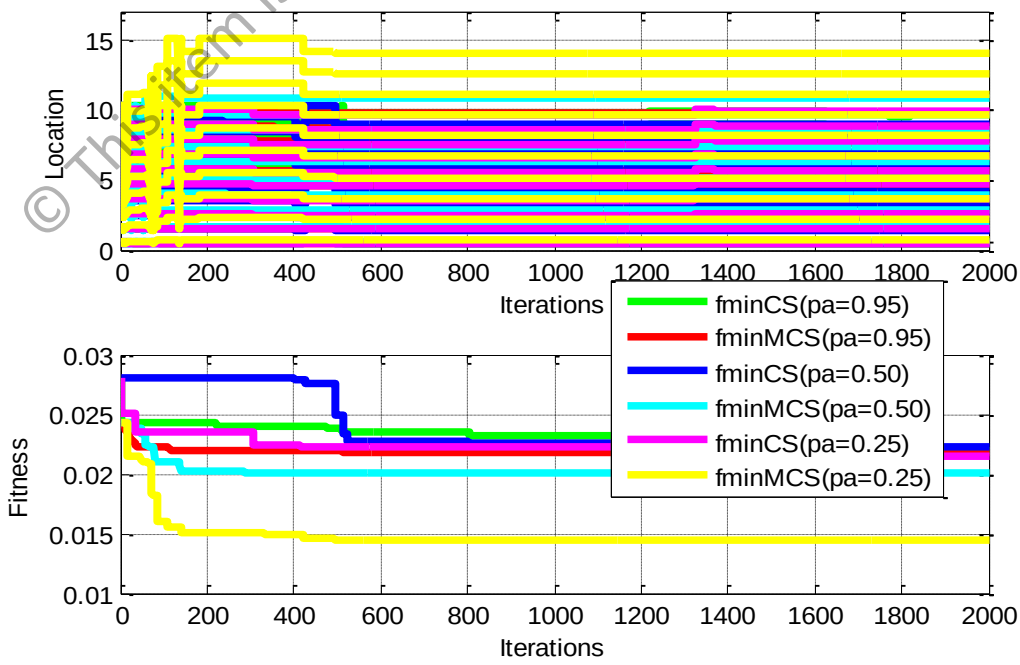


Figure 4.26: Location and Fitness Curves for CS vs. MCS in P_a
($2N$ = 20, Uniform, maxIter = 2000)

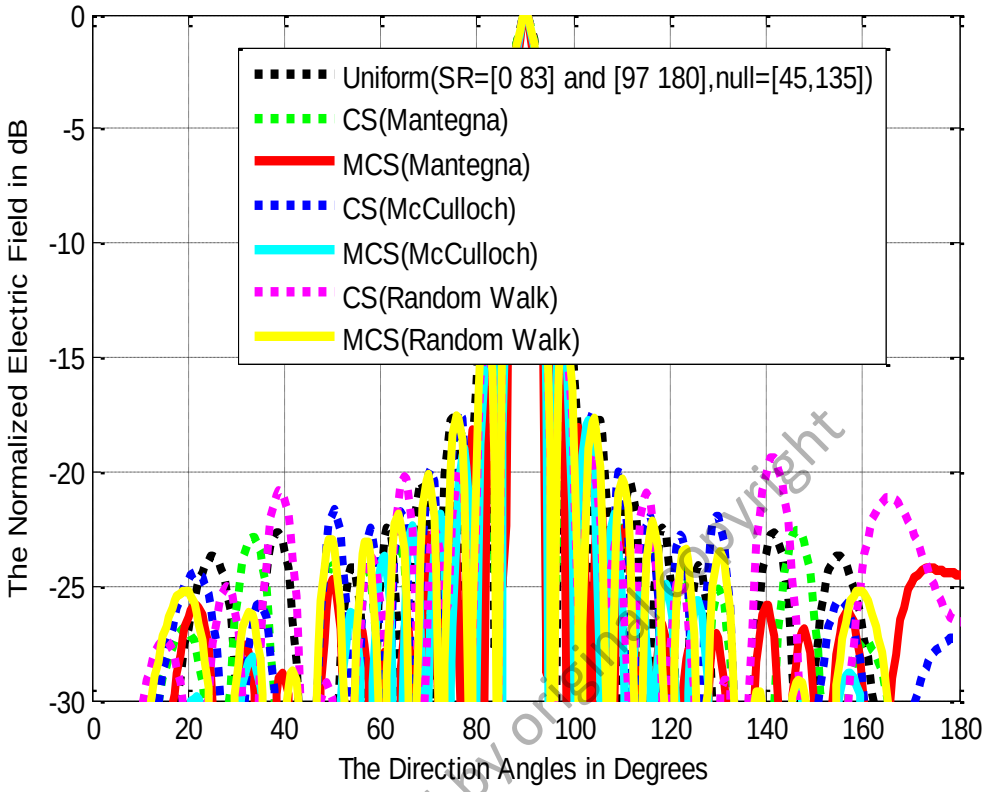
Table 4.12: Optimal Location for CS vs. MCS in P_a ($2N$ = 20, Uniform, maxIter = 2000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [P_a = 0.95]	±0.5107	±1.5324	±2.5534	±3.5723	±4.5948
MCS [P_a = 0.95]	±0.5130	±1.5393	±2.5652	±3.5913	±4.6198
CS [P_a = 0.50]	±0.4693	±1.3993	±2.3272	±3.2442	±4.1589
MCS [P_a = 0.50]	±0.5682	±1.7058	±2.8438	±3.9809	±5.1175
CS [P_a = 0.25]	±0.5177	±1.5535	±2.5898	±3.6245	±4.6596
MCS [P_a = 0.25]	±0.7386	±2.2139	±3.6916	±5.1681	±6.6451
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [P_a = 0.95]	±5.6178	±6.6402	±7.6559	±8.6816	±9.7263
MCS [P_a = 0.95]	±5.6435	±6.6696	±7.6916	±8.7222	±9.7719
CS [P_a = 0.50]	±5.0598	±5.9880	±6.9604	±7.9722	±9.0101
MCS [P_a = 0.50]	±6.2501	±7.3835	±8.5150	±9.6552	±10.8044
CS [P_a = 0.25]	±5.6975	±6.7348	±7.7696	±8.8171	±9.8788
MCS [P_a = 0.25]	±8.1216	±9.5943	±11.0699	±12.5519	±14.0308

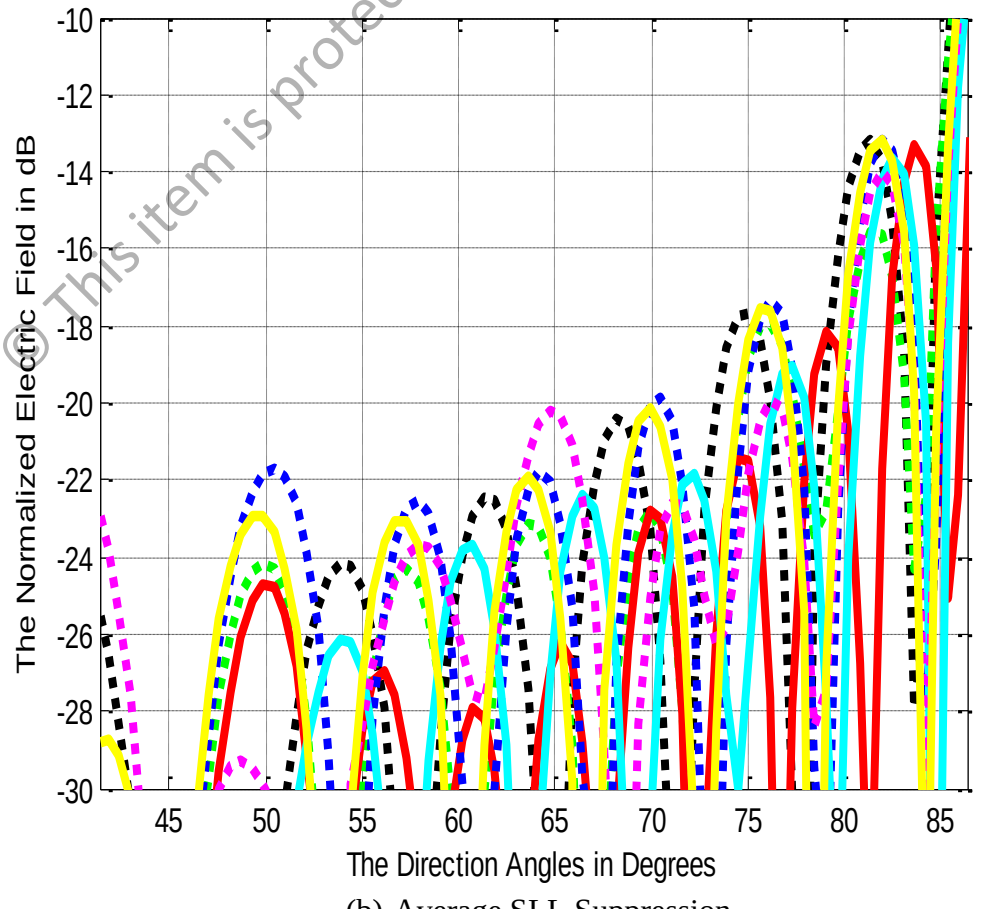
Fifthly, there is a more stiff experiment on the broadside case (main beam steered to 90°) for $2N$ = 20 linear array with two prescribed nulls at 45° and 135°. All the MCS and CS–optimizers with three α–stable distribution methods, host nest (population) = 30, discovery rate, P_a = 25% or 0.25, α = 2.0 (Gaussian), and the length step factor = L/100 or 0.01 are simulated for 1000 iterations.

Figure 4.27(a) – (b) show generally that the hypothesized MCS–based array with the Mantegna's algorithm outperforms other rivals with the SLL between 0.001 dB and 4.256 dB relatively lower than the conventional array.

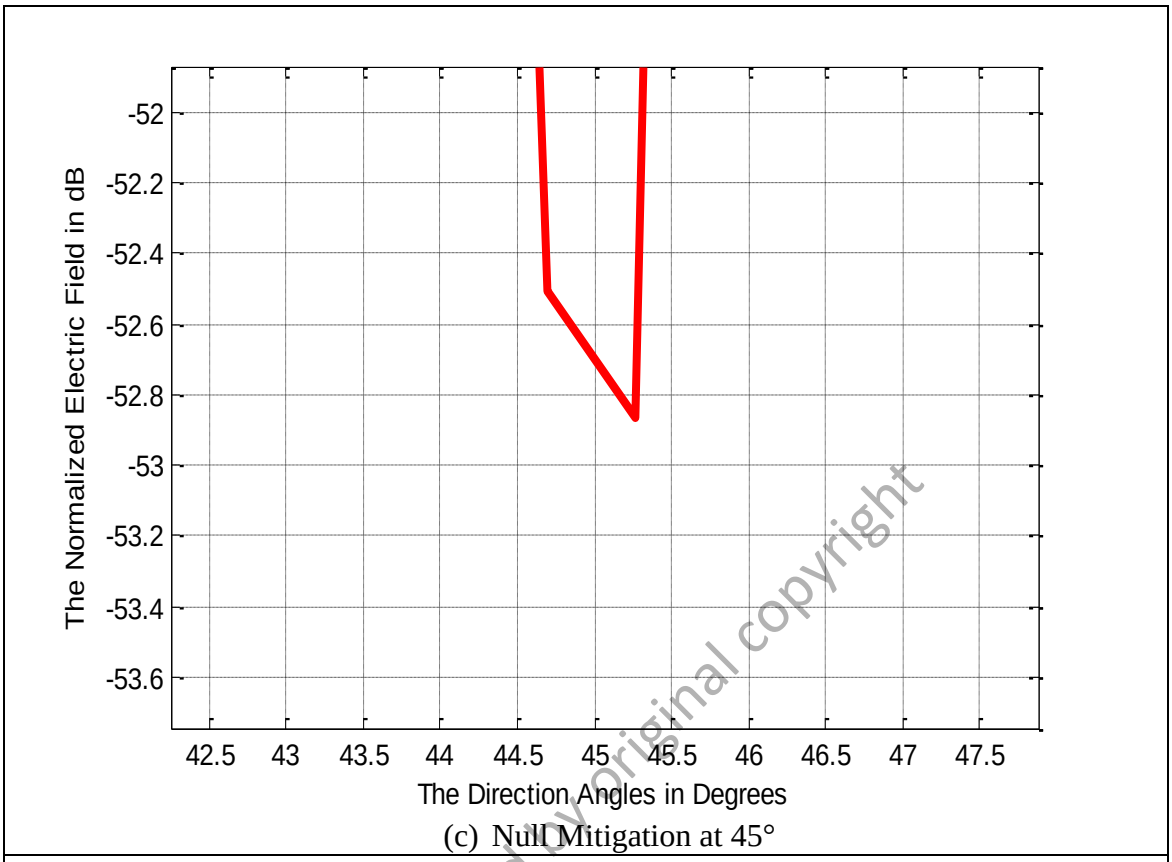
Figure 4.27(c) – (d) display that the MCS–optimizer (Mantegna's algorithm) clearly demonstrates the best prescribed nulls mitigation at direction of 45° with SLL of –52.865 dB, and at direction of 135° with SLL of –53.615 dB, respectively.



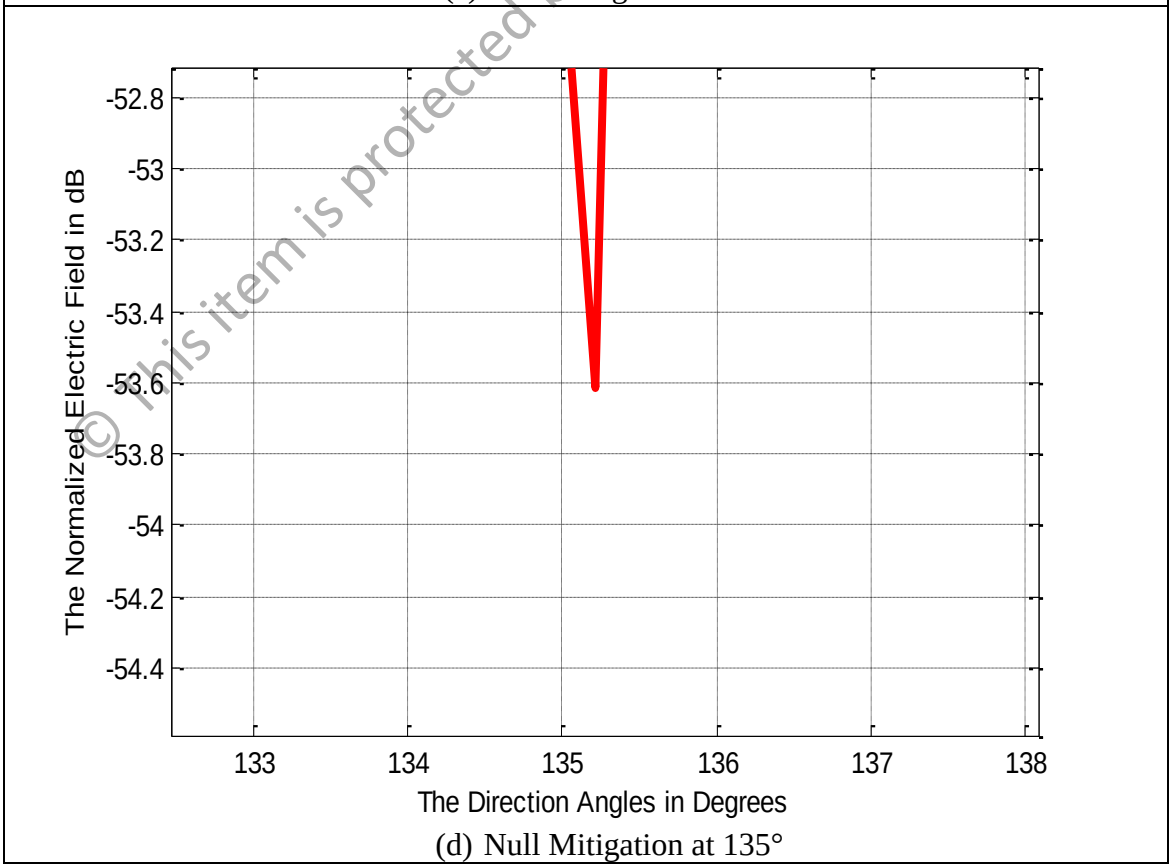
(a) Normalized Radiation Pattern



(b) Average SLL Suppression



(c) Null Mitigation at 45°



(d) Null Mitigation at 135°

Figure 4.27: Normalized Pattern for CS vs. MCS in Distribution Type ($2N$ = 20, Main Beam = 90°, Null = [45°, 135°], maxIter = 1000)

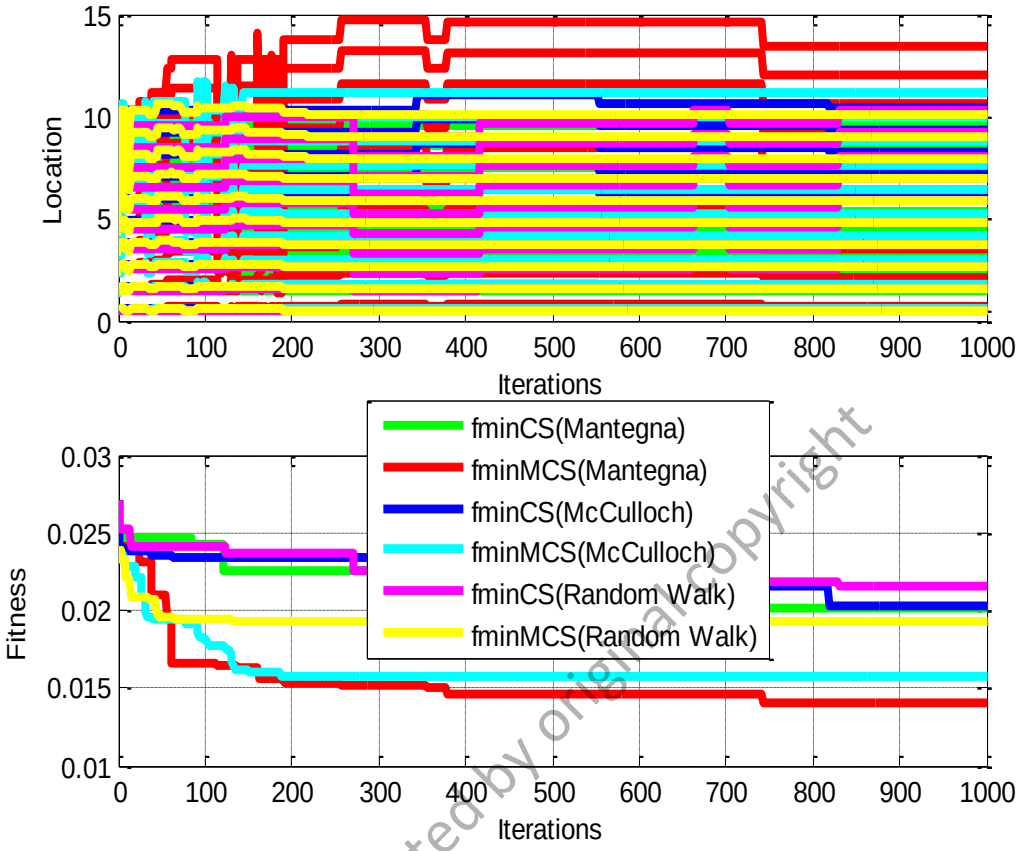


Figure 4.28: Location and Fitness Curves for CS vs. MCS in Distribution Type ($2N$ = 20, Main Beam = 90°, Null = [45°, 135°])

Table 4.13: Optimal Location for CS vs. MCS in Distribution Type ($2N$ = 20, Main Beam = 90°, Null = [45°, 135°], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.4961	±1.4846	±2.4652	±3.4373	±4.4084
MCS [Mantegna]	±0.7056	±2.1159	±3.5259	±4.9363	±6.3482
CS [McCulloch]	±0.5492	±1.6466	±2.7435	±3.8403	±4.9364
MCS [McCulloch]	±0.5860	±1.7585	±2.9345	±4.1110	±5.2872
CS [Random Walk]	±0.5368	±1.6158	±2.6897	±3.7601	±4.8504
MCS [Random Walk]	±0.5321	±1.5964	±2.6607	±3.7250	±4.7893
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Mantegna]	±5.3896	±6.3739	±7.4809	±8.6066	±9.5032
MCS [Mantegna]	±7.7584	±9.1684	±10.5780	±11.9900	±13.4071
CS [McCulloch]	±6.0358	±7.1355	±8.2336	±9.3341	±10.4292
MCS [McCulloch]	±6.4631	±7.6292	±8.7984	±9.9805	±11.1713
CS [Random Walk]	±5.9251	±6.9828	±8.0584	±9.2452	±10.2804
MCS [Random Walk]	±5.8535	±6.9178	±7.9821	±9.0464	±10.1107

According to Figure 4.28, the MCS (Mantegna's algorithm) has the most location fluctuations before stabilizing. This is followed by the MCS (McCulloch's algorithm). In short, the proposed MCS (Mantegna's algorithm) produce the biggest diversity of optimal solutions. Furthermore, the MCS–optimizer (Mantegna's algorithm) also generates the lowest f_{min} convergence of 0.0141. This is followed by the MCS (McCulloch' algorithm), with the f_{min} convergence of 0.0158 and the MCS (standard random walk) with the f_{min} convergence of 0.0193, respectively. Overall, the proposed MCS optimizer regardless α–stable distribution methods used converge with the lower f_{min} values compared to the CS opponents.

As enlisted in Table 4.13, the proposed MCS algorithms generate a significant optimal location deviations compared to the conventional array. In details, the MCS (Mantegna' algorithm) has the biggest deviations between |±0.2056| and |±3.9071| for all $2N$ = 20 array elements. This is trailed by the MCS (McCulloch's algorithm) with the deviations between |±0.0860| and |±1.6713|. The biggest deviations become the main factor for the postulated MCS–optimizer (Mantegna's algorithm) to achieve superior optimal solutions diversity that demonstrates best side lobes suppression and/or nulls mitigation whilst preserving the main beam.

Sixthly, the MCS and CS–optimizers with three α–stable distribution methods, host nest (population) = 20, discovery rate, P_a = 25% or 0.25, α = 2.0 (Lévy flight Gaussian distribution) and the length step factor = L/100 or 0.01 are simulated for $2N$ = 20 linear array within the Dolph–Chebyshev amplitude envelope. All the tested MCS–optimizers deploy the Roulette wheel selection operator and adaptive weight, w with the magnitude domain of [0.95 1.00]. Figure 4.29 depicts the theoretical Dolph–Chebyshev filter window, which is applied in this study.

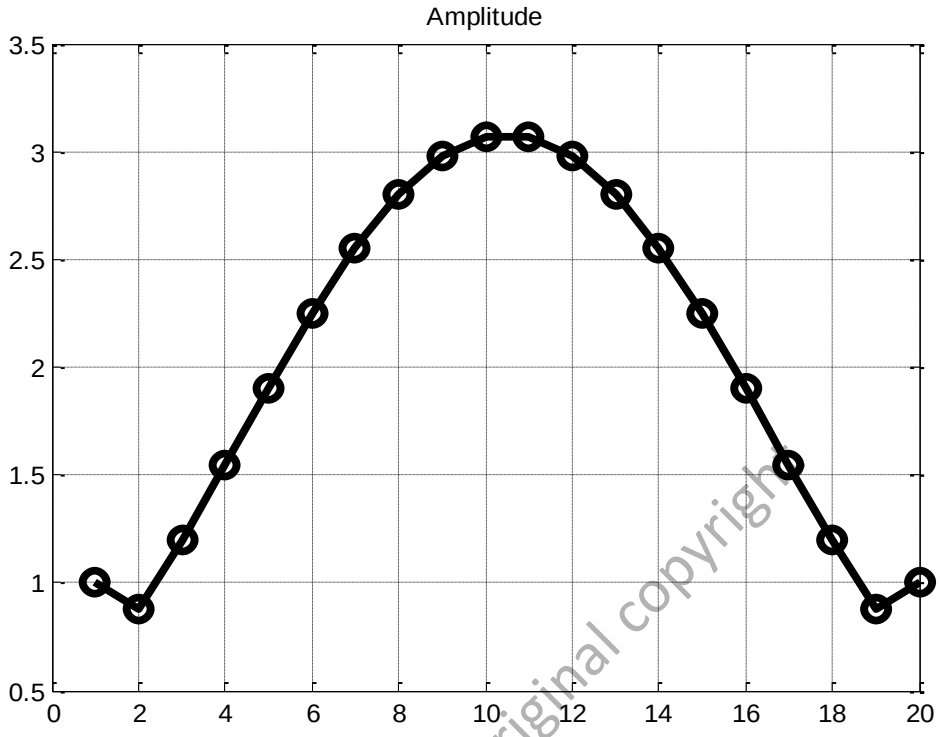
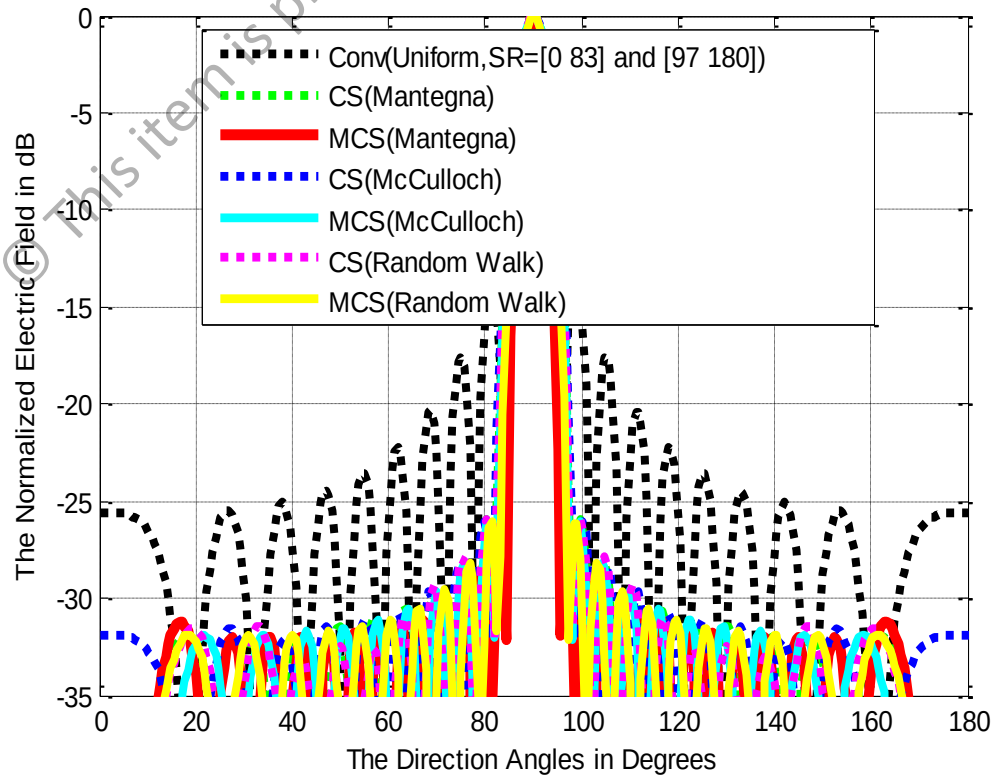
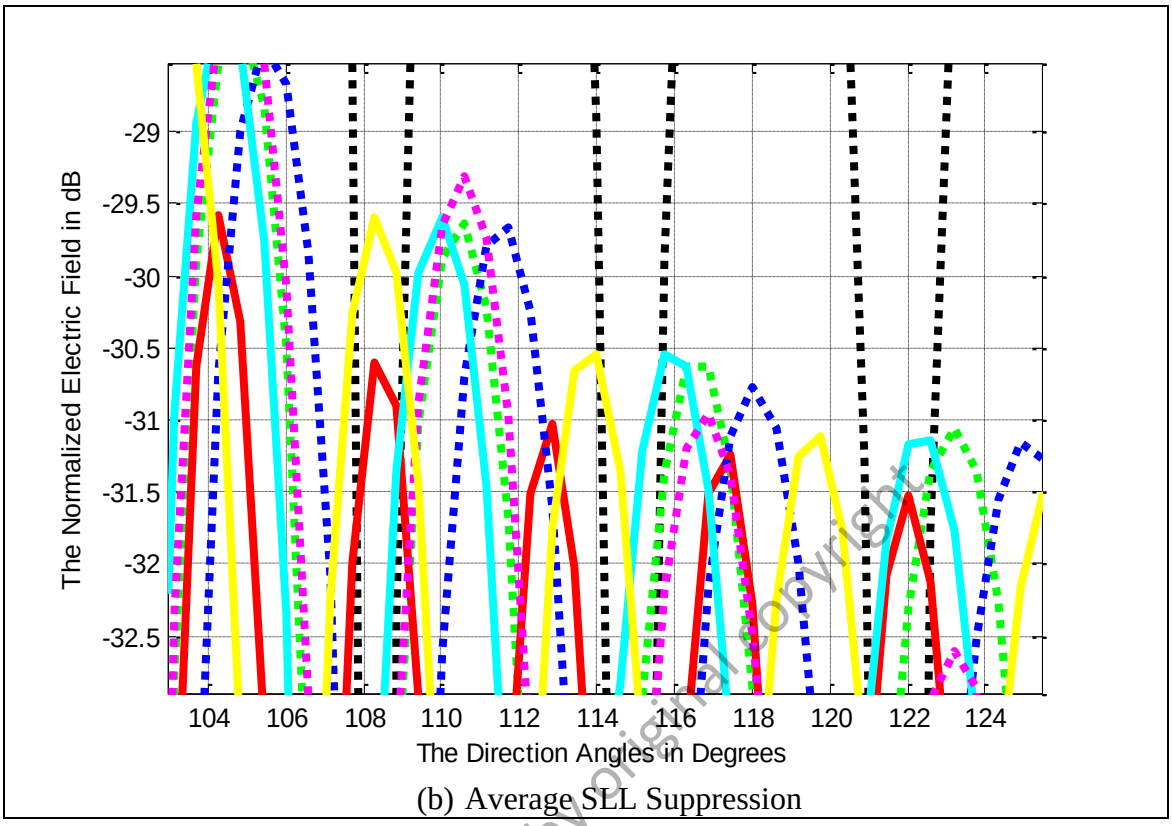


Figure 4.29: The Dolph–Chebyshev Excitation Amplitude for $2N = 20$ Linear Array



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.30: Normalized Pattern for CS vs. MCS in Distribution Type
($2N$ = 20, Dolph–Chebyshev, maxIter = 1000)

Based on Figure 4.30(a) – (b), the proposed MCS–optimizer (Mantegna) generates the best equiripple side lobes suppression with the gain relatively 32.1 dB below the main beam, narrowly followed by the MCS–optimizer (McCulloch) and the MCS–optimizer (standard random walk) with equally 31.9 dB below the main beam. The CS opponent with the McCulloch's algorithm is the worst one with 31.6 dB below the main beam. It can be seen that in Figure 4.31, the MCS–optimizer (Mantegna's algorithm) has the largest fluctuations in optimal position of antenna radiators. In addition, the MCS–optimizer (Mantegna) converges quickly after 107 iterations and achieves the f_{min} of 0.0111. This is closely trailed by the MCS–optimizer (McCulloch) with the f_{min} convergence of 0.0114 after 280 iterations, and the MCS–optimizer (standard random walk) with f_{min} convergence of 0.0121 after 47 iterations.

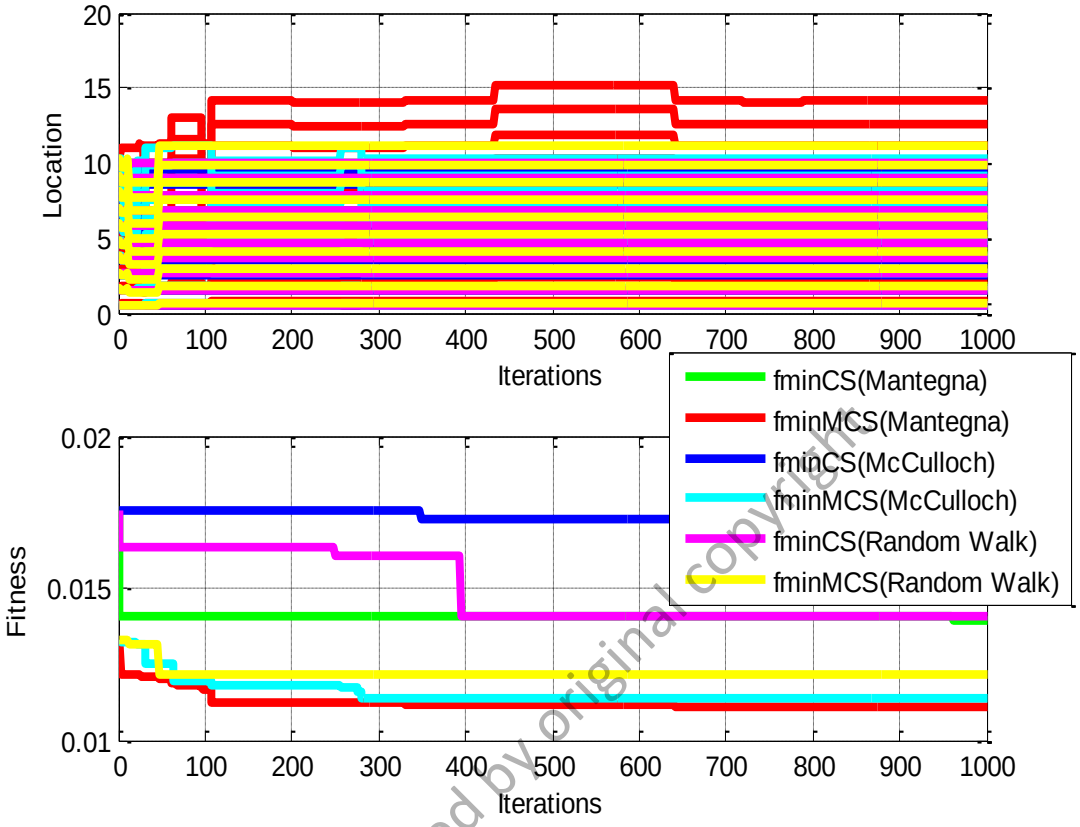


Figure 4.31: Location and Fitness Curves for CS vs. MCS in Distribution Type (2*N* = 20, Dolph–Chebyshev, maxIter = 1000)

Table 4.14: Optimal Location for CS vs. MCS in Distribution Type (2*N* = 20, Dolph–Chebyshev, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.5259	±1.5777	±2.6294	±3.6812	±4.7329
MCS [Mantegna]	±0.7433	±2.2297	±3.7164	±5.2027	±6.6892
CS [McCulloch]	±0.5000	±1.5000	±2.5000	±3.4991	±4.4988
MCS [McCulloch]	±0.5382	±1.6146	±2.6909	±3.7673	±4.8437
CS [Random Walk]	±0.5264	±1.5778	±2.6305	±3.6824	±4.7342
MCS [Random Walk]	±0.5835	±1.7506	±2.9176	±4.0847	±5.2517
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Mantegna]	±5.7848	±6.8364	±7.8882	±8.9400	±9.9919
MCS [Mantegna]	±8.1760	±9.6625	±11.1487	±12.6356	±14.1217
CS [McCulloch]	±5.4985	±6.4983	±7.4982	±8.4985	±9.5000
MCS [McCulloch]	±5.9201	±6.9964	±8.0728	±9.1492	±10.2255
CS [Random Walk]	±5.7872	±6.8404	±7.8932	±8.9421	±9.9953
MCS [Random Walk]	±6.4188	±7.5858	±8.7529	±9.9199	±11.0870

Based on Table 4.14, the MCS–optimizer (Mantegna) obviously has the largest fluctuations of array element location (with respect to λ/2) between |±0.2433| and |±4.6217| compared to the conventional array. This is followed by both MCS counterparts (standard random walk and McCulloch). In sum, this agrees with Figure 4.31 that the significant fluctuations of optimal location and small f_{min} convergence stimulate the three proposed MCS–optimizers to generate a better SLL suppression in the Dolph–Chebyshev envelope compared to the three original CS competitors. In other words, the proposed MCS algorithms are capable to explore and exploit a better diversity of optimal solutions within search domain.

Seventhly, there is an experiment on $2N$ = 30 linear Dolph–Chebyshev array in which the proposed MCS algorithm based array is relatively compared with the original CS, genetic algorithms (GA), and particle swarm optimization (PSO)–based arrays. All the evaluated MCS and CS–metaheuristic optimizers are simulated for 1000 iterations using host nest (population) = 30, discovery rate, P_a = 25% or 0.25, length step factor = L/100 or 0.01, α–stable distribution method = Mantegna's algorithm, and α = 2.0 (Lévy flight Gaussian distribution). Once again, the MCS–optimizer deploys the Roulette wheel selection operator, and adaptive w domain = [0.95 1.00]. The fixed excitation Dolph–Chebyshev filter envelope as illustrated in Figure 4.32 is used whereas for a simplification, the excitation phase is set to 0° for all $2N$ = 30 symmetric linear array elements. Precisely, the feed current Dolph–Chebyshev amplitudes, which are applied in this study are [0.4235 0.2477 0.3127 0.3827 0.4564 0.5322 0.6083 0.6826 0.7532 0.8182 0.8756 0.9238 0.9613 0.9870 1.0000].

The PSO–optimizer is constructed using particle (population) = 30, max/min velocity limit of particle = [–0.1 +0.1], individuality accelerator = 1.0, and sociality accelerator = 1.0, respectively. The GA–optimizer with the Roulette wheel selection

operators also uses chromosome or gene (population) = 30, gene crossover probability, P_c = 90% or 0.9, and gene mutation probability, P_m = 10% or 0.1.

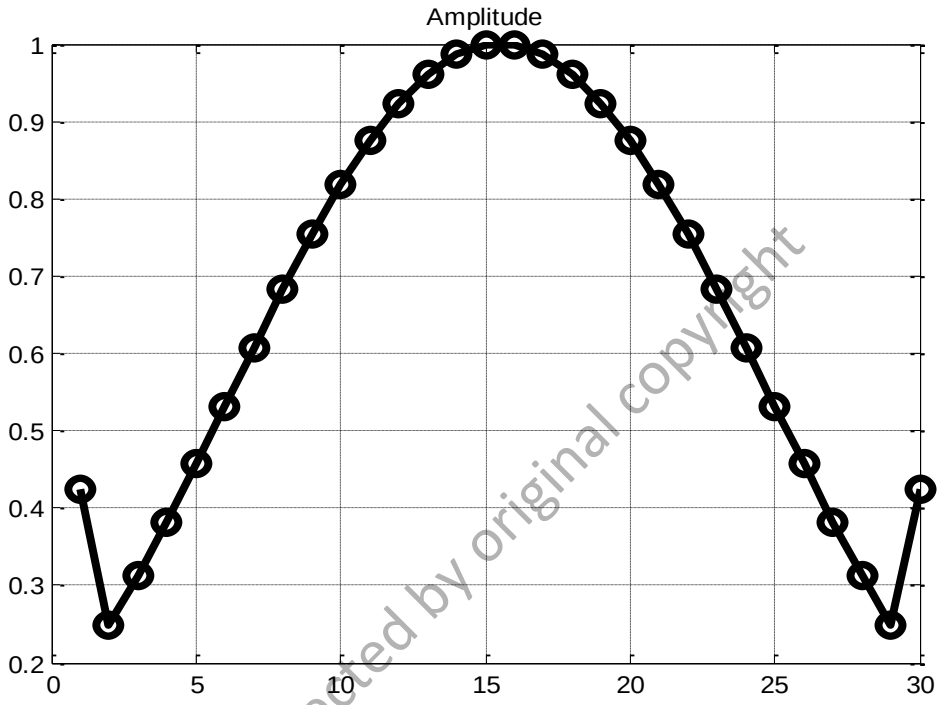
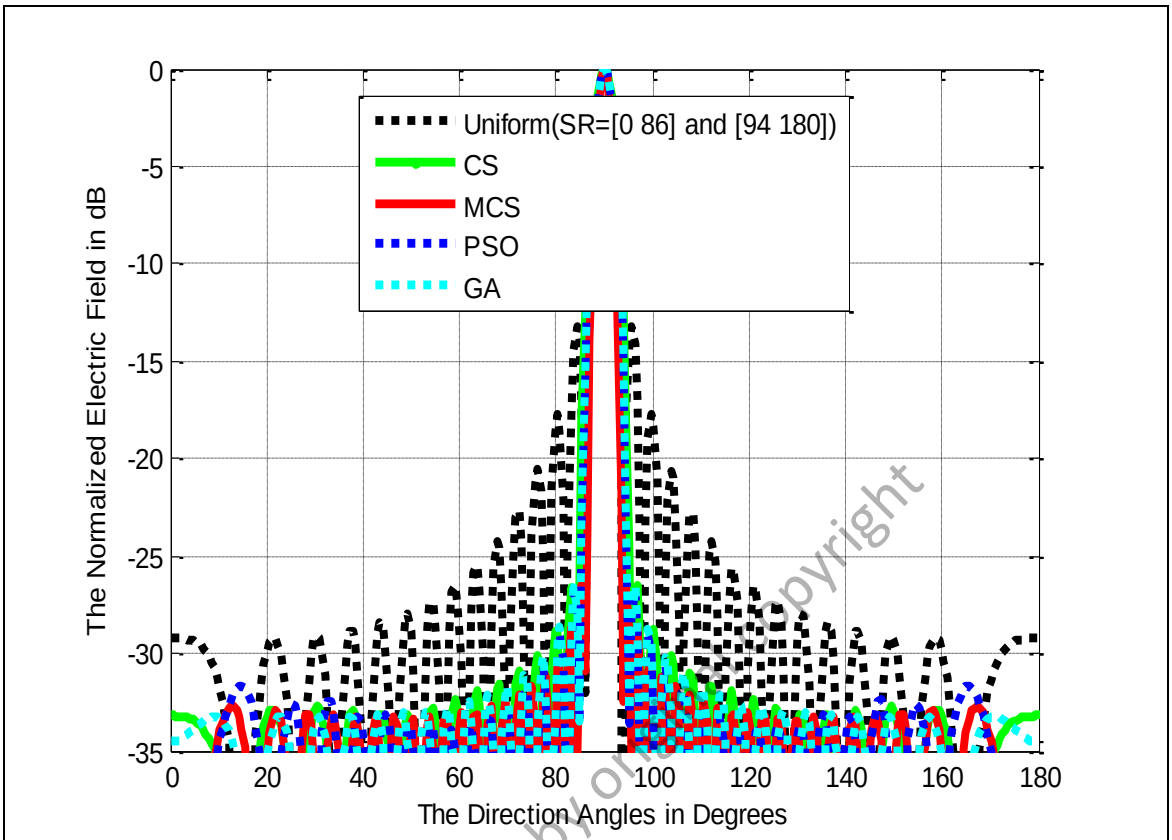
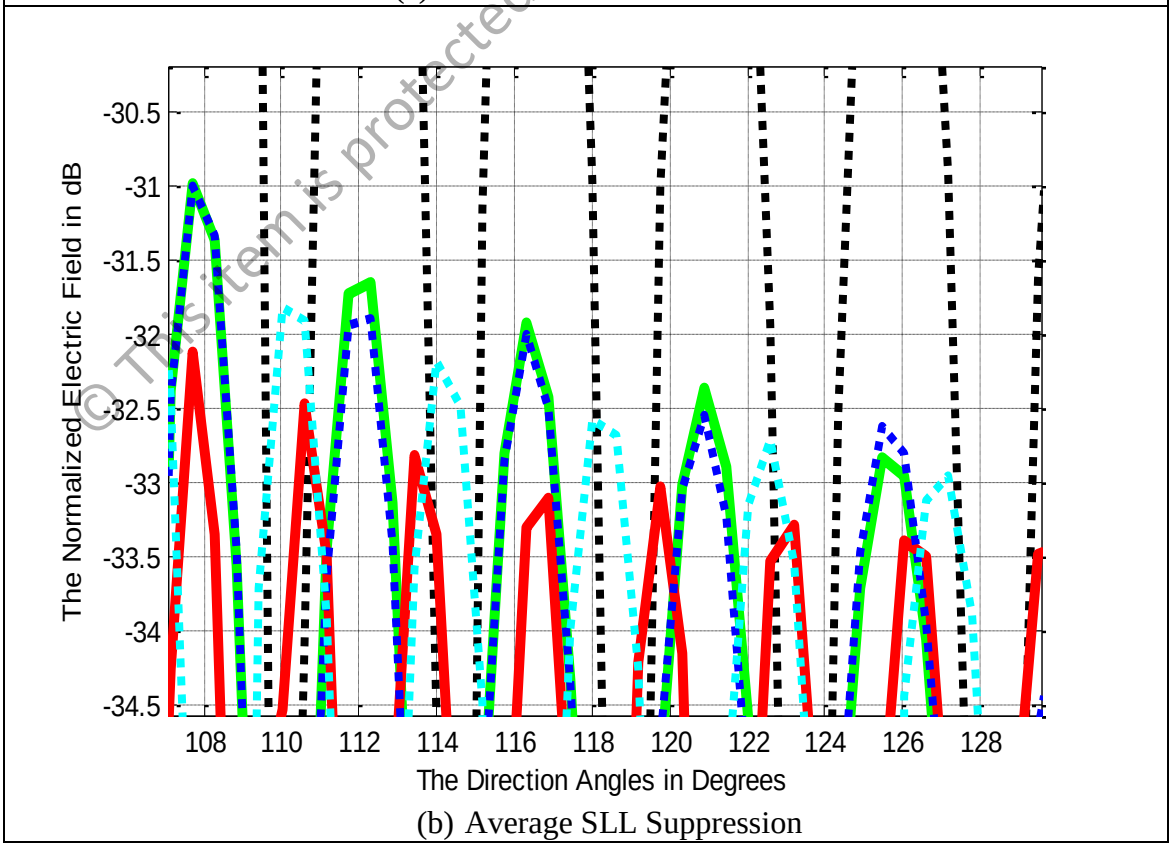


Figure 4.32: The Dolph–Chebyshev Excitation Amplitude for 2*N* = 30 Linear Array



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.33: Normalized Pattern for MCS vs. Other EC–Optimizers
(2*N* = 30, Dolph–Chebyshev, maxIter = 1000)

Based on Figure 4.33(a) – (b), the MCS–optimizer with the Mantegna's algorithm has the identical highest peak of equiripple side lobes gain relatively about 33.35 dB below the main beam, and relatively 20.10 dB below the highest SLL peak of the uniform (conventional) pattern, respectively due to the lowest f_{min} convergence of 0.0126 after about 400 iterations as shown in Figure 4.34. The highest SLL peak of the uniform pattern, which is produced by the conventional array, is comparatively 13.25 dB below the main beam. On the other hand, the standard CS–optimizer with the Mantegna's algorithm generates the equiripple side lobes of relatively about 32.86 dB below the main beam or relatively about 19.61 dB below the highest SLL peak of the uniform pattern. In addition, both of the GA and PSO counterparts has the equiripple SLL of relatively about 33.10 dB below the main beam or relatively 19.85 dB below the highest SLL peak of the uniform pattern.

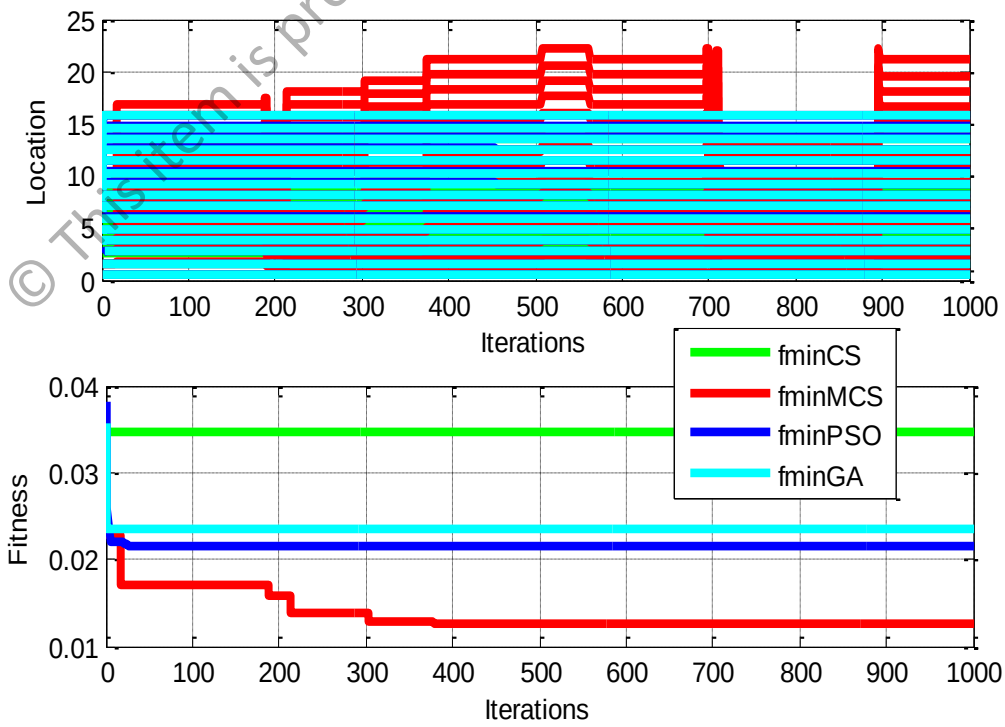


Figure 4.34: Location and Fitness Curves for MCS vs. Other EC–Optimizers ($2N$ = 30, Dolph–Chebyshev, maxIter = 1000)

Table 4.15: Optimal Location for MCS vs. Other EC–Optimizers
(2*N* = 30, Dolph–Chebyshev, maxIter = 1000)

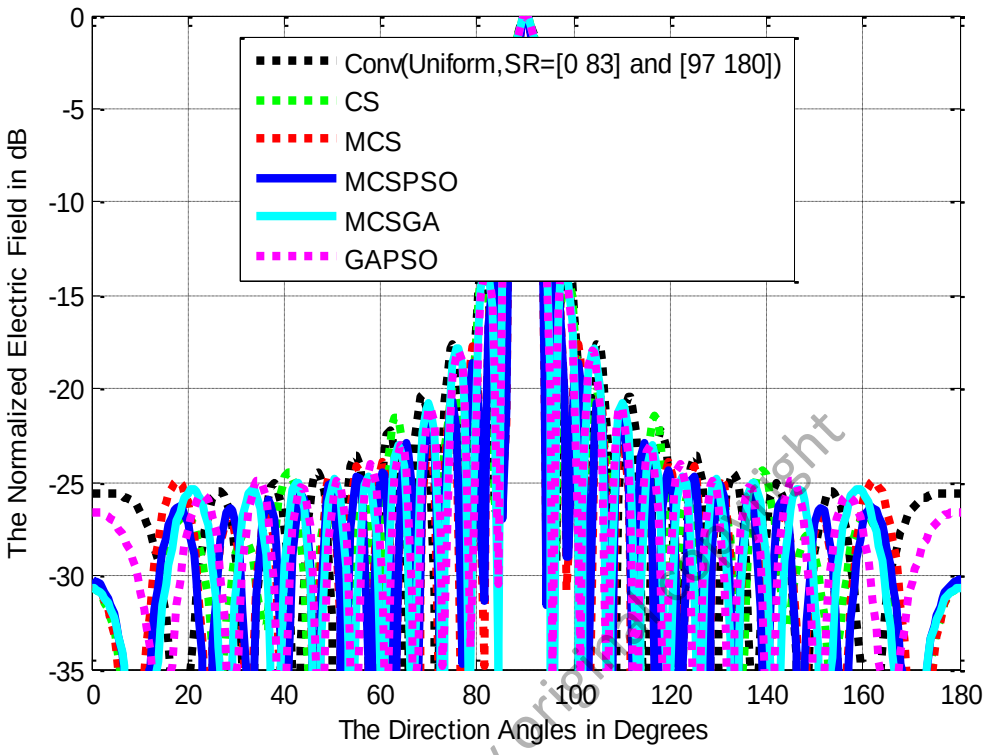
Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
MCS [Mantegna]	±0.7269	±2.1807	±3.6344	±5.0882	±6.5420
CS [Mantegna]	±0.5000	±1.5002	±2.5004	±3.5005	±4.5007
PSO	±0.5457	±1.6380	±2.7302	±3.8224	±4.9146
GA	±0.5417	±1.6251	±2.7085	±3.7918	±4.8752
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
MCS [Mantegna]	±7.9958	±9.4495	±10.9033	±12.3571	±13.8109
CS [Mantegna]	±5.5009	±6.4996	±7.4996	±8.4996	±9.4996
PSO	±6.0068	±7.0990	±8.1912	±9.2834	±10.3756
GA	±5.9586	±7.0420	±8.1254	±9.2087	±10.2921
Element	**11**	**12**	**13**	**14**	**15**
X_n **[λ/2]**	±10.5000	±11.5000	±12.5000	±13.5000	±14.5000
MCS [Mantegna]	±15.2646	±16.7184	±18.1722	±19.6259	±21.0797
CS [Mantegna]	±10.5013	±11.5018	±12.5020	±13.5021	±14.5018
PSO	±11.4678	±12.5600	±13.6522	±14.7445	±15.8447
GA	±11.3755	±12.4589	±13.5423	±14.6256	±15.7090

It can be found from Table 4.15 that the postulated MCS algorithm (Mantegna's α–stable distribution method) demonstrates the biggest array element location fluctuations (with respect to λ/2) compared to standard CS, PSO, and GA counterparts. This implies that the MCS algorithm is capable to explore further optimal solutions within the global search space with the deviations compared to the conventional antenna array between |±0.2269| and |±6.5797|. Consequently, this becomes as the key factor for the proposed MCS–optimizer to produce the best diversity of optimal solutions (array element locations), which generates the lowest f_{min} convergence, and the best equiripple SLL suppression.

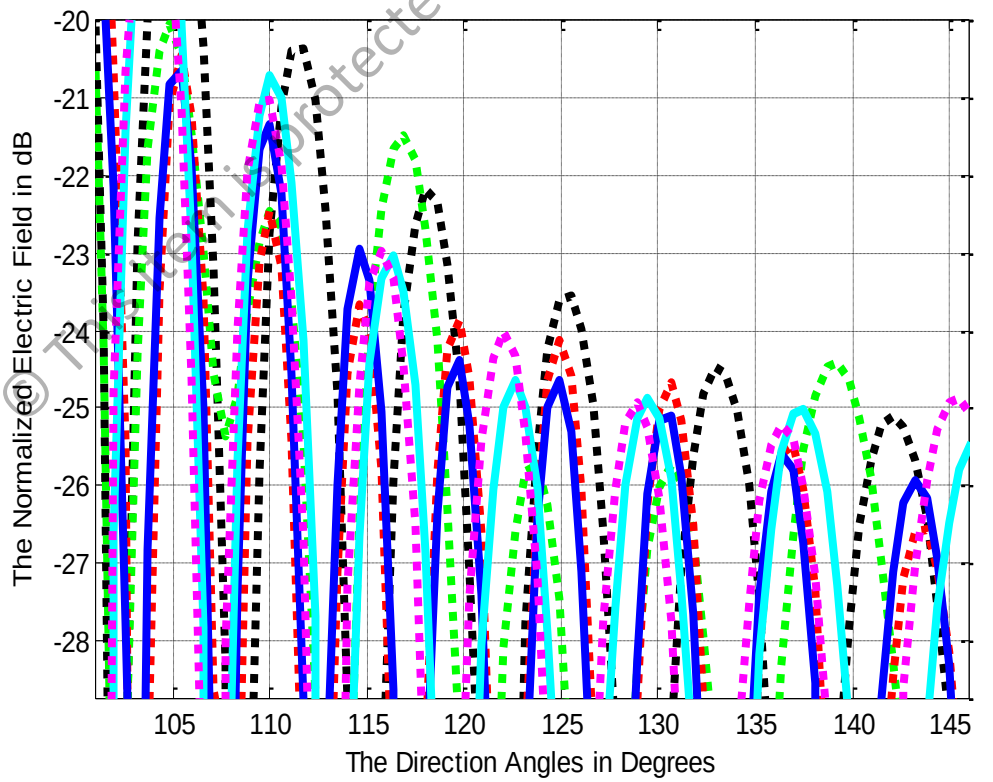
4.3 The Proposition of Modified Cuckoo Search Algorithm through Hybridization in Linear Antenna Array Synthesis

In this stage, both the postulated MCSPSO and MCSGA–hybrid algorithms with Mantegna's α–stable distribution method, host nest (population) = 30, discovery rate, P_a = 25% or 0.25, length step factor = $L/100$ or 0.01, and α = 2.0 (Lévy flight Gaussian distribution) are tested on $2N$ = 20 linear antenna array. Both MCSPSO and MCSGA hybrid algorithms along with MCS algorithm deploy the Roulette wheel selection operator and adaptive inertia, w domain of [0.95 1.00]. The proposed MCSPSO and MCSGA–hybrid optimizers are compared with the hybrid GAPSO, MCS, and standard CS algorithms, respectively.

It is found that the proposed MCSPSO hybrid algorithm outperforms other competitors with the SLL suppression of 0.30 dB – 4.70 dB lower than the conventional array within the [20°83°] and [97° 180°] regions whilst maintaining the main beam intensity at direction angle of 90°. This is closely followed by the proposed MCS metaheuristic algorithm with the SLL suppression of 0.17 dB – 4.46 dB as can be seen in Figure 4.35(a) – (b).



(a) Normalized Radiation Pattern



(b) Average SLL Suppression

Figure 4.35: Normalized Pattern for MCS Hybrids vs. others
($2N$ = 20, Uniform, maxIter = 1000)

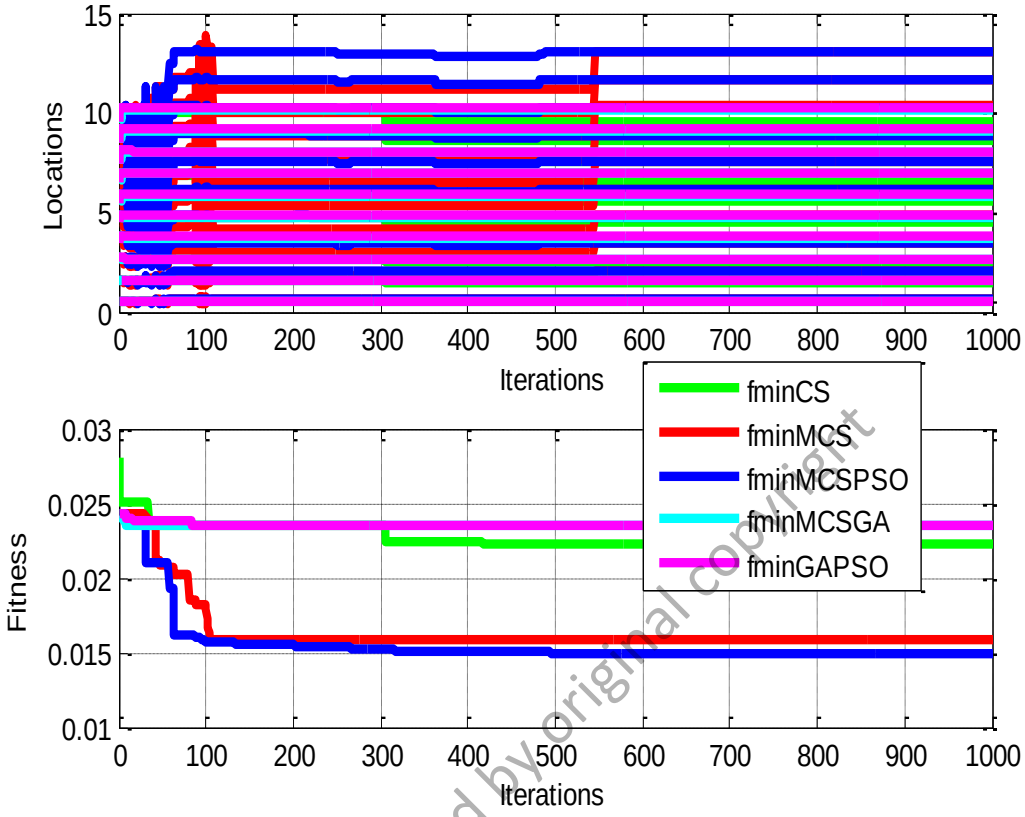


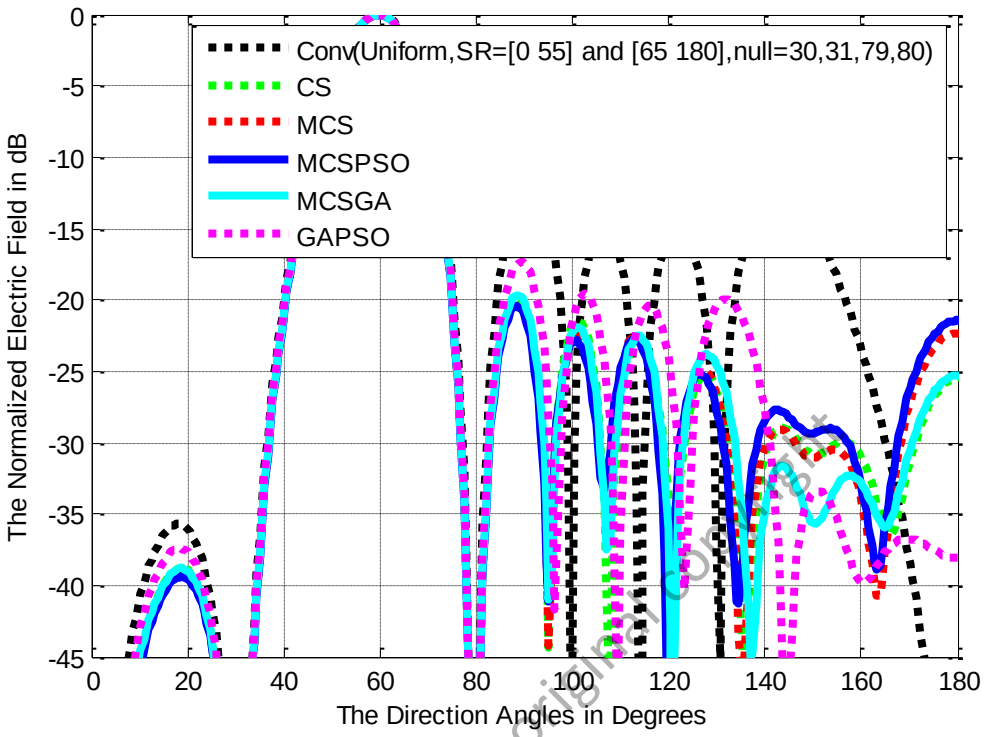
Figure 4.36: Location and Fitness Curves for MCS Hybrids vs. others
($2N$ = 20, Uniform, maxIter = 1000)

Table 4.16: Optimal Location for MCS Hybrids vs. others
($2N$ = 20, Uniform, maxIter = 1000)

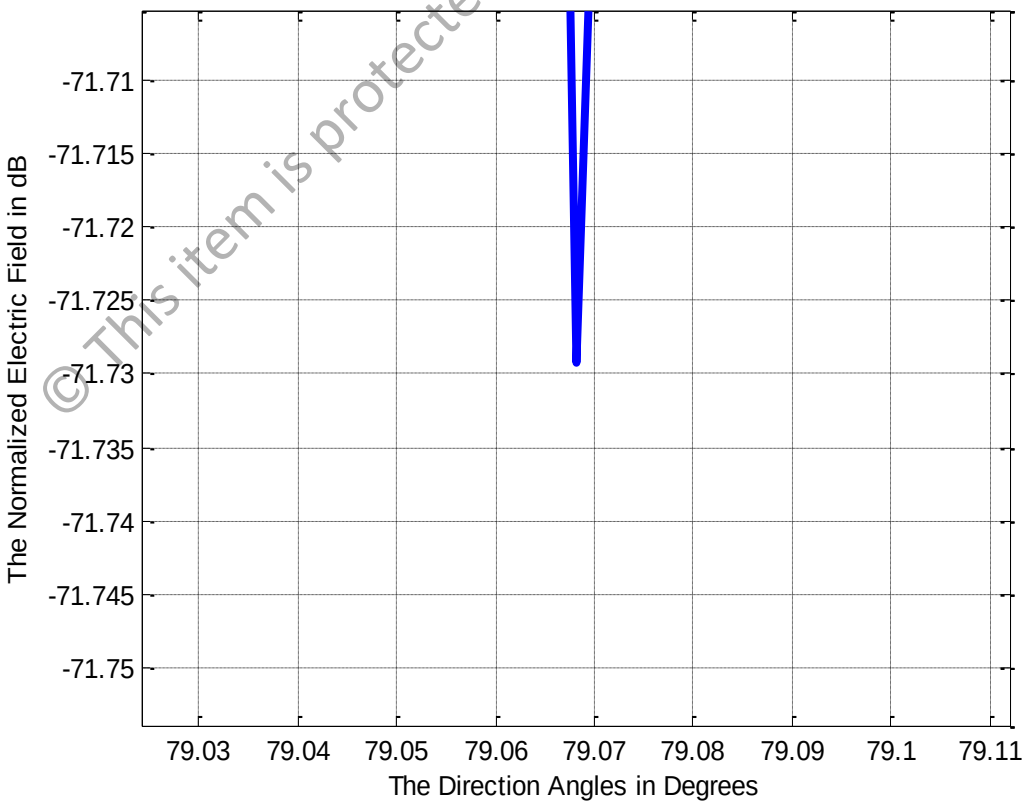
Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.5032	±1.5097	±2.5133	±3.5257	±4.5425
MCS [Mantegna]	±0.6885	±2.0652	±3.4428	±4.8200	±6.1961
MCSPSO [Mantegna]	±0.6863	±2.0565	±3.4256	±4.7964	±6.1639
MCSGA [Mantegna]	±0.5321	±1.5963	±2.6607	±3.7255	±4.7890
GAPSO	±0.5359	±1.6110	±2.6914	±3.7697	±4.8494
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Mantegna]	±5.5418	±6.5398	±7.5332	±8.6300	±9.5861
MCS [Mantegna]	±7.5732	±8.9516	±10.3255	±11.7039	±13.0822
MCSPSO [Mantegna]	±7.5229	±8.8808	±10.2481	±11.6280	±13.0103
MCSGA [Mantegna]	±5.8532	±6.9188	±7.9813	±9.0437	±10.1121
GAPSO	±5.9274	±7.0053	±8.0814	±9.1594	±10.2439

Based on Figure 4.36, the MCSPSO–hybrid optimizer has the lowest f_{min} convergence of 0.0150 after executing nearly 500 iterations. This is closely followed by the MCS counterpart with the f_{min} convergence of 0.0158. In this case, both the proposed MCSPSO and MCS algorithms produce the two largest optimal location deviations compared to the conventional array as appeared in both Figure 4.36 and Table 4.16, respectively. In this case, the MCSPSO hybrid algorithm has the location (with respect to $\lambda/2$) deviations between |±0.1863| and |±3.5103|.

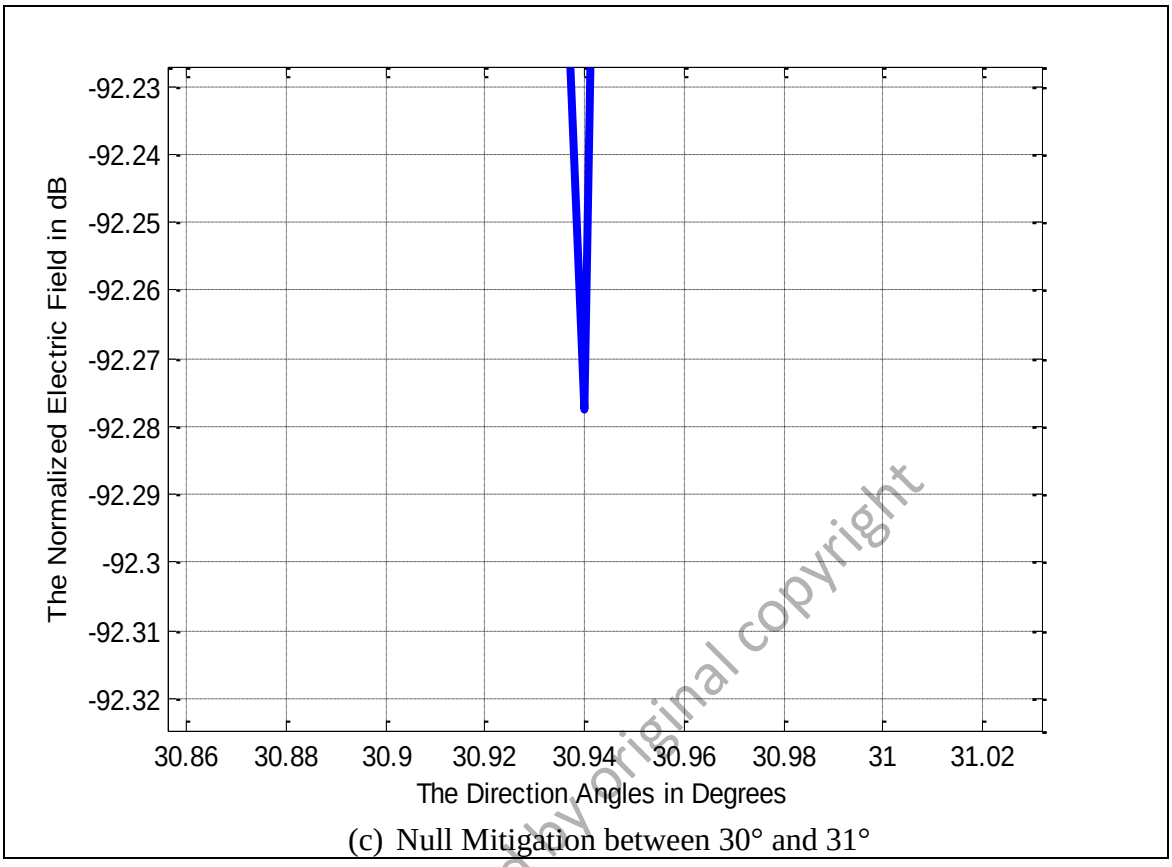
In the second experiment, a more vigorous condition is tested on the postulated MCSPSO, MCSGA, and other competitors where the main beam is steered to 60°**,** with four prescribed nulls at 30°, 31°, 79°, and 80°, respectively. The simulation shows that the MCSPSO–based array slightly surpasses other counterparts in SLL suppression specifically within the [0° 30°] and [80° 130°] suppression regions. This is narrowly followed by both of the proposed MCSGA and MCS–optimizers. In this case, the MCSPSO algorithm yields the average SLL magnitude of 3.7 dB – 10.3 dB below the conventional linear array as shown in Figure 5.37(a).



(a) Normalized Radiation Pattern



(b) Null Mitigation between 79° and 80°



(c) Null Mitigation between 30° and 31°

Figure 4.37: Normalized Pattern for MCS Hybrids vs. others
(2*N* = 10, Main Beam = 60°, Null = [30°, 31°, 79°, 80°], maxIter = 100)

According to Figure 5.37 (b) – (c), the proposed MCSPSO hybrid algorithm also has the best null mitigation through attaining the lowest magnitude of 92.28 dB and 71.73 dB below the main beamnearly at direction angles of 30.94° and 79.07°, respectively. In addition, Figure 4.38, Figure 4.39, and Figure 4.40 portray the adaptive polar pattern, which is generated by conventional, MCSPSO, and MCSGA–based arrays. It is clearly found that all the three arrays have the main lobe with the largest radiation intensity steered to 60° and perform nulls mitigation at the four prescribed directions. Precisely, both the postulated MCSPSO and MCSGA hybrid algorithms generate side lobe suppression relatively lower than the conventional array as shown in both Figure 4.39 and Figure 4.40, respectively.

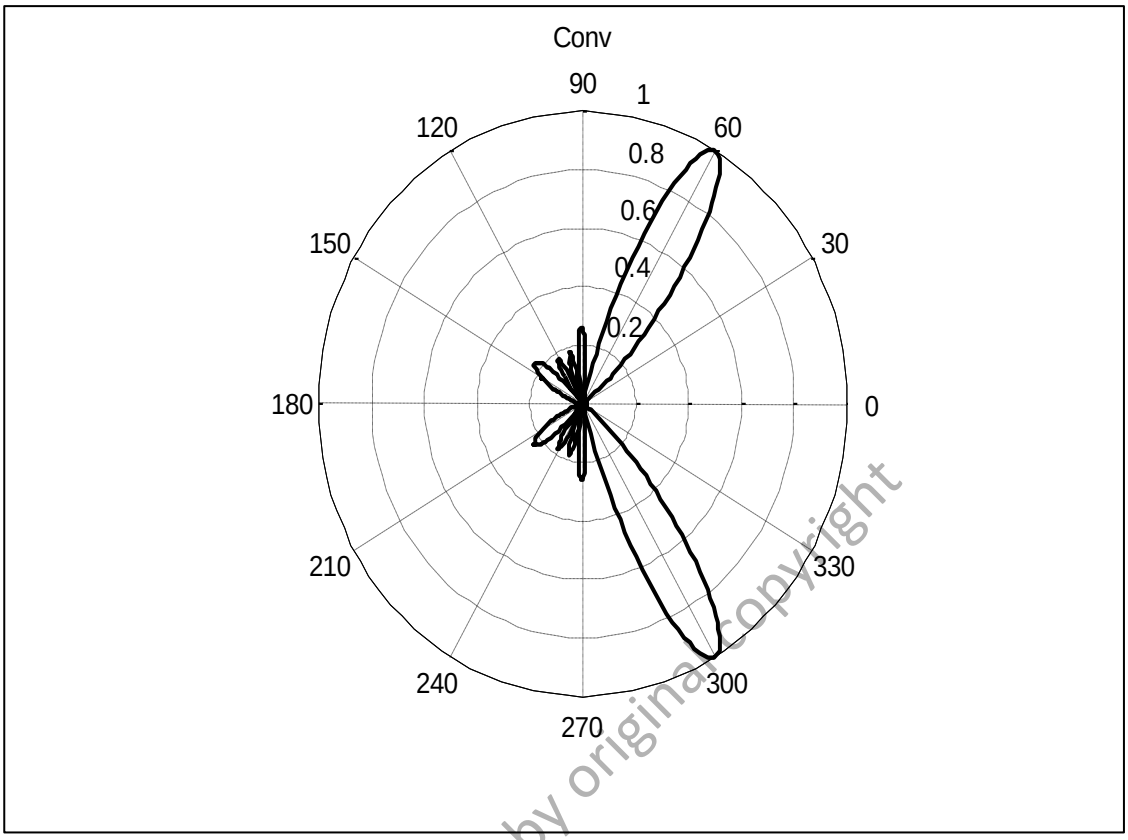


Figure 4.38: Polar Pattern for Conventional Array
(2*N* = 10, Main Beam = 60°, Null = [30°, 31°, 79°, 80°])

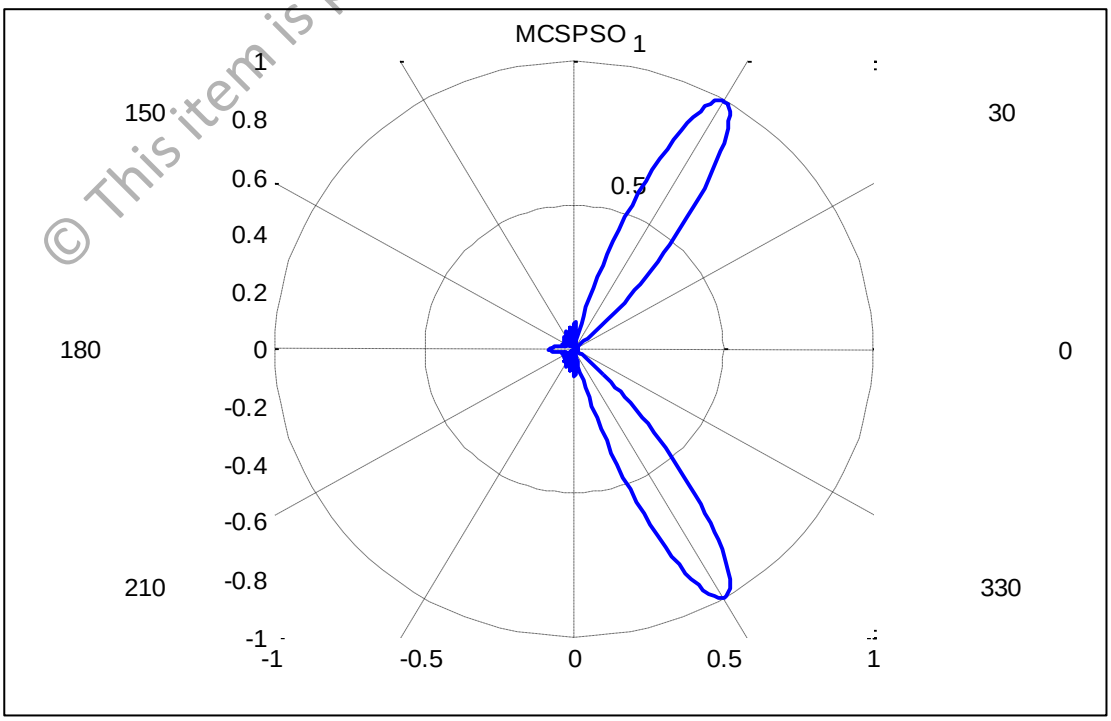


Figure 4.39: Polar Pattern for MCSPSO Array
(2*N* = 10, Main Beam = 60°, Null = [30°, 31°, 79°, 80°])

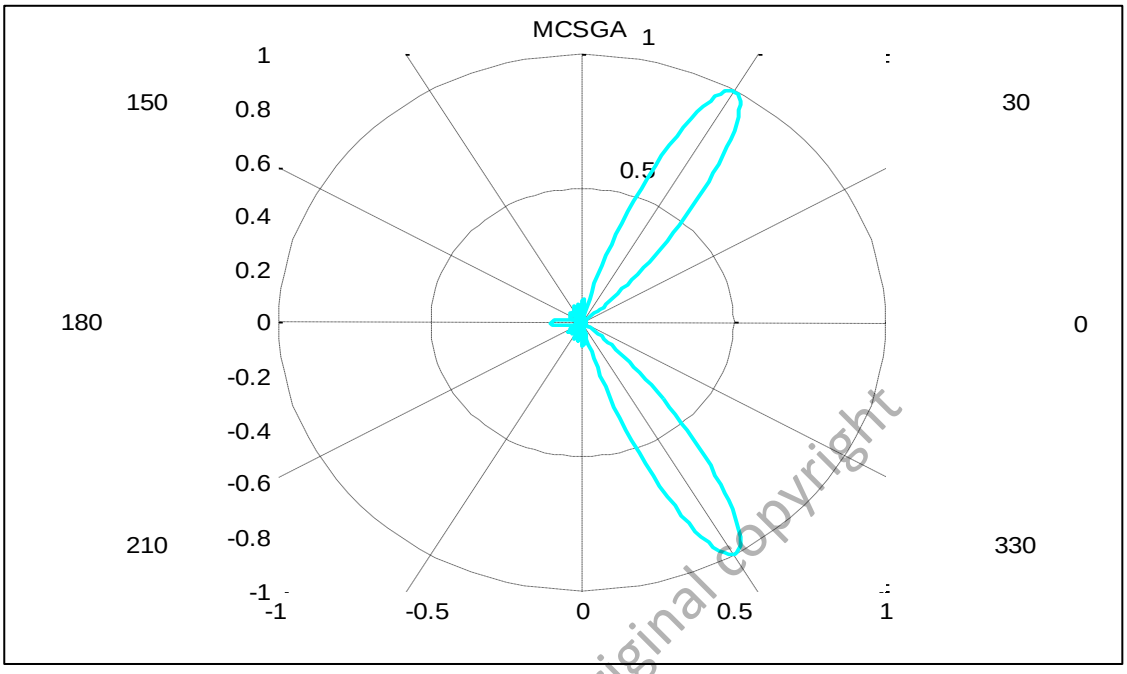


Figure 4.40: Polar Pattern for MCSGA Array
(2*N* = 10, Main Beam = 60°, Null = [30°, 31°, 79°, 80°])

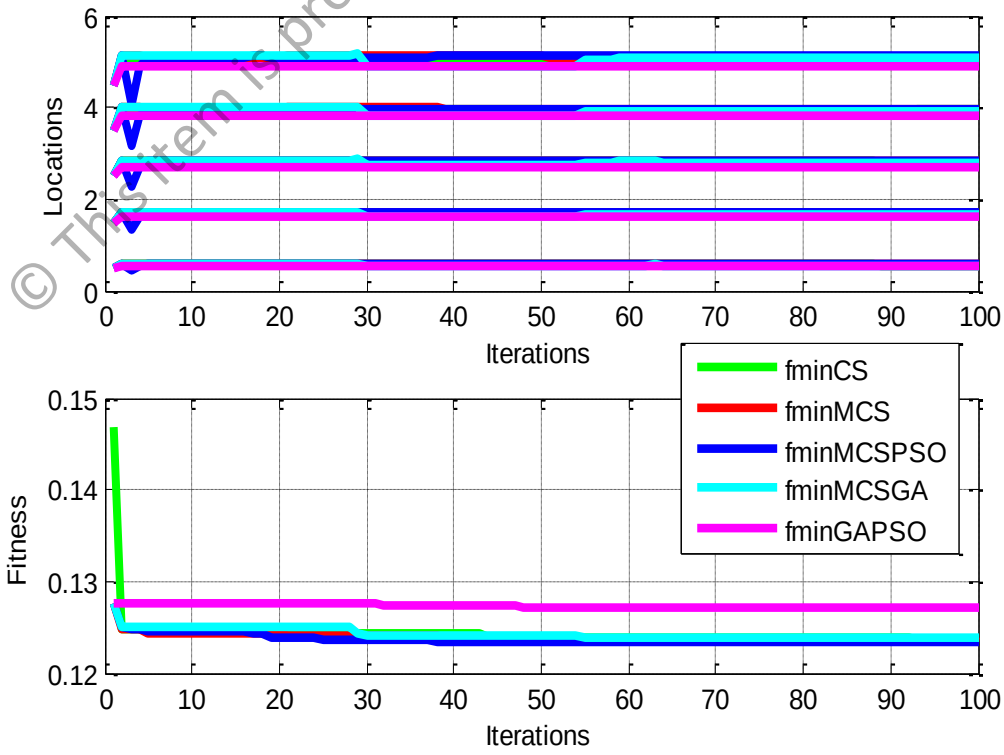


Figure 4.41: Location and Fitness Curves for MCS Hybrids vs. others
(2*N* = 10, Main Beam = 60°, Null = [30°, 31°, 79°, 80°])

Table 4.17: Optimal Location for MCS Hybrids vs. others
(2*N* = 10, Main Beam = 60°, Null = [30°, 31°, 79, 80], maxIter = 100)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.5624	±1.6879	±2.8124	±3.9379	±5.0748
MCS [Mantegna]	±0.5673	±1.6985	±2.8332	±3.9632	±5.1084
MCSPSO [Mantegna]	±0.5684	±1.7021	±2.8389	±3.9707	±5.1204
MCSGA [Mantegna]	±0.5620	±1.6858	±2.8120	±3.9350	±5.0704
GAPSO	±0.5421	±1.6289	±2.7146	±3.7993	±4.8903

Figure 4.41 clearly shows that the postulated MCSPSO–based array generates the largest optimal location oscillations and attains the lowest f_{min} convergence of 0.1234 after about 57 iterations. Moreover, based on Table 4.17, the MCSPSO is proven to have the largest optimal location deviations (with respect to λ/2) compared to the conventional array between |±0.0684| and |±0.6204| for 2*N* = 10 linear array. Overall, the MCSPSO outperforms MCSGA and MCS counterparts in side lobes suppression and/or nulls mitigation. In this case, the hybridization process manipulates the PSO algorithm particles (population) significant velocity and position updating processes. Consequently, the MCSPSO–hybrid optimizer is able to control effectively the Lévy flight direction (motion) and speed of cuckoos (population), and through it can locate the global best (*gbest*) solutions in search space.

CHAPTER FIVE

MULTIOBJECTIVE OPTIMIZATION

5.1 Multiobjective Optimization Techniques using Modified Cuckoo Search Algorithm in Linear Antenna Array Synthesis

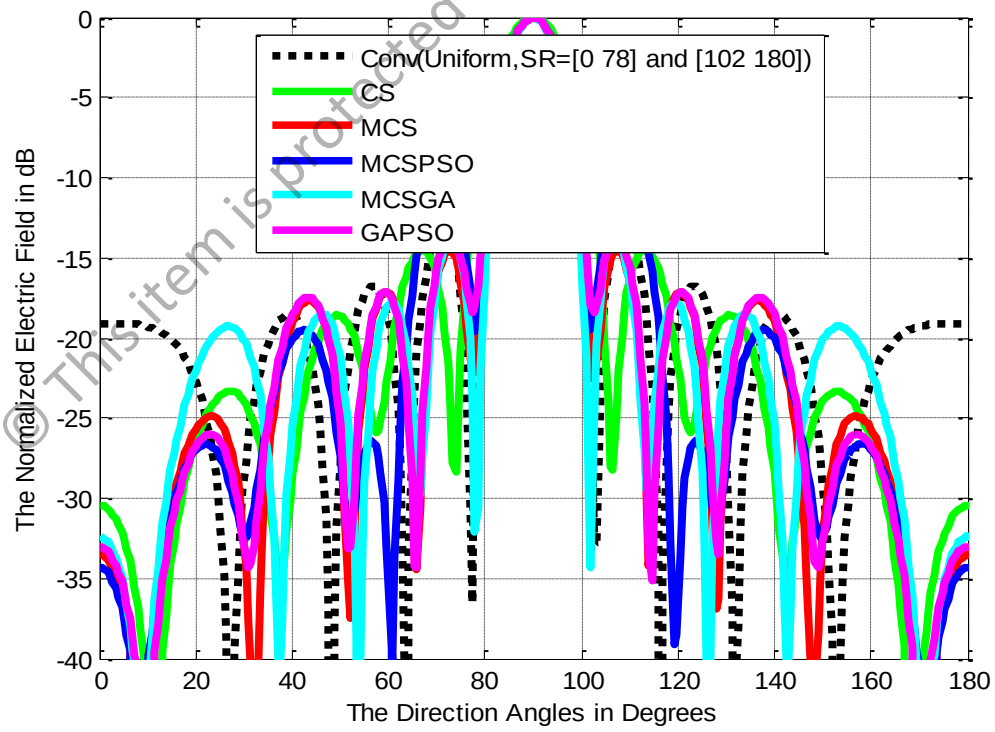
This chapter discusses the simulations of proposing modified and hybrid CS algorithms in MO optimization methods for linear antenna array synthesis. In this case, two approaches are deployed, which are weighted–sum and global Pareto front.

5.1.1 Weighted–Sum Approach

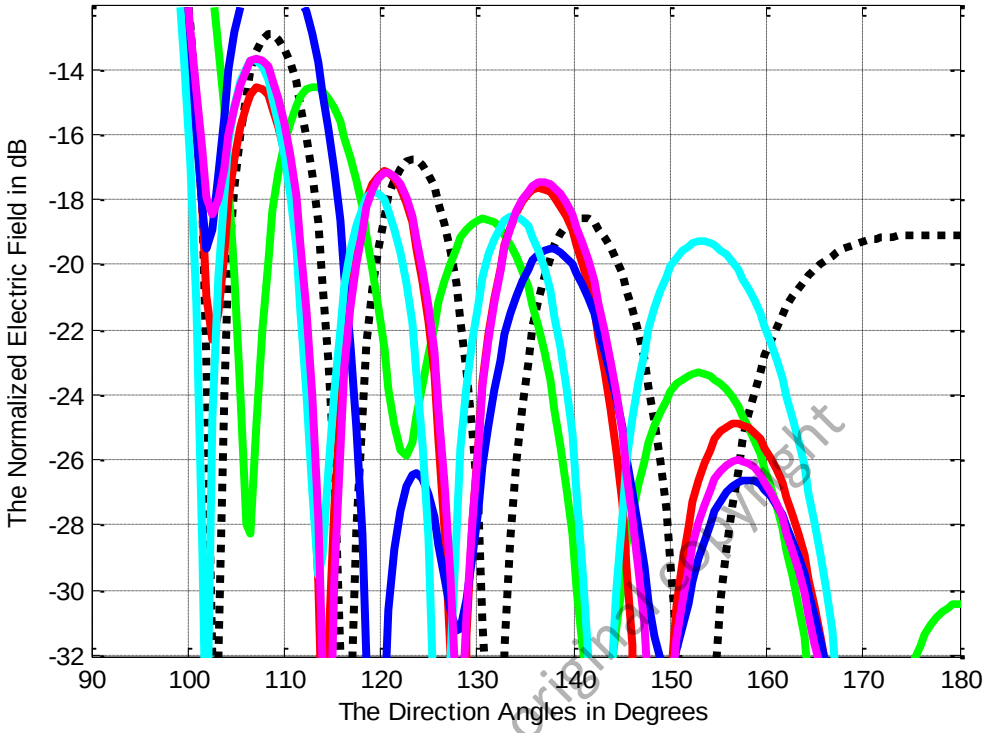
In the first weighted–sum MO simulation, the postulated MCSPSO, MCSGA, and MCS algorithms with Mantegna's algorithm as the selected α–stable distribution method, host nest (population) = 30, length step factor = $L/100$ or 0.01, and α = 2.0 (Lévy flight Gaussian distribution) are examined on the $2N$ = 10 linear array. For uniformity, all the proposed MCSPSO, MCSGA, and MCS algorithms have the dynamic P_a magnitude domain of [0.01 0.25] and dynamic, w magnitude domain of [0.95 1.05], respectively. The proposed algorithms are deliberately compared with hybrid GAPSO, and original CS algorithm. Precisely, both the MCSPSO and GAPSO optimizers use the PSO algorithm with the dynamic random particle velocity domain of [–0.1 +0.1]. Moreover, the MCSGA and GAPSO algorithms apply the GA optimizer with the gene crossover probability, P_c = 90% or 0.9, and gene mutation probability, P_m = 10% or 0.1.

Based on Figure 5.1(a), the normalized radiation pattern for the postulated MCSPSO optimizer outperforms other competitors by having the lowest average SLL

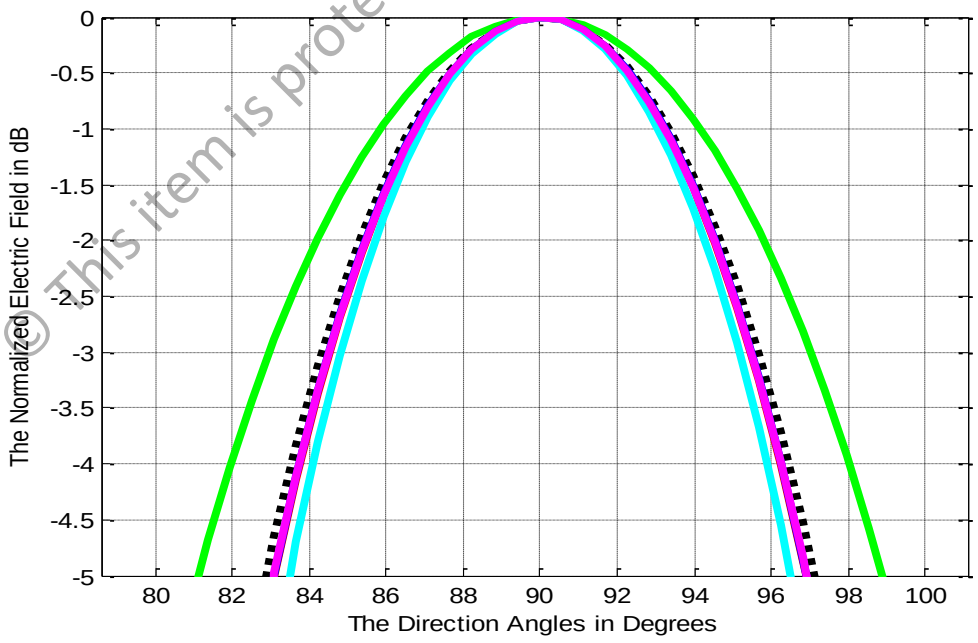
suppression and whereas the MCSGA counterpart has the highest intensity or the smallest half–power beamwidth (HPBW) of the main beam. Precisely, the MCSPSO algorithm suppresses SLL between –0.97 dB and –15.20 dB compared to the conventional array within the [120^o 180^o] and [0^o 60^o] regions as can be seen in Figure 5.1(b). The HPBW is the angular separation in which the magnitude of the radiation pattern decreases by 50% (or –3 dB) from the peak of the main beam. Figure 5.1(c) shows the pattern for MCSGA optimizer has the smallest HPBW, which decreases to –3 dB at 84.825^o and 95.175^o. Hence, the HPBW is 95.175^o – 84.825^o = 10.35^o. Furthermore, the postulated MCSGA array generates a higher directivity of 8.4474 dB whereas the MCSPSO counterpart has the smaller directivity of 8.2567 dB, respectively.



(a) Normalized Radiation Pattern



(b) Average SLL Suppression



(c) HPBW Pattern

Figure 5.1: Normalized Pattern for Weighted–Sum MCS Hybrids vs. others
(2*N* = 10, Uniform, maxIter = 1000)

As displayed in Figure 5.2, the MCSPSO hybrid optimizer produces the lowest weighted–sum fitness, f_{min} of 0.6654 that leads to the best SLL suppression performance. It executes the largest optimal location fluctuations followed by the MCSGA counterpart. Besides, the MCSPSO–based array also generates the lowest optimal amplitude magnitudes for all the $2N$ = 10 symmetric elements as shown in Figure 5.3. Furthermore, Figure 5.4 portrays that the MCSPSO optimizer produces the biggest optimal phase fluctuations compared to other tested algorithms, respectively. All the optimal locations, amplitudes, and phases of the MCSPSO optimizer produce a better diversity of excitation components, which increase the linear array beam scanning capability with low side lobes.

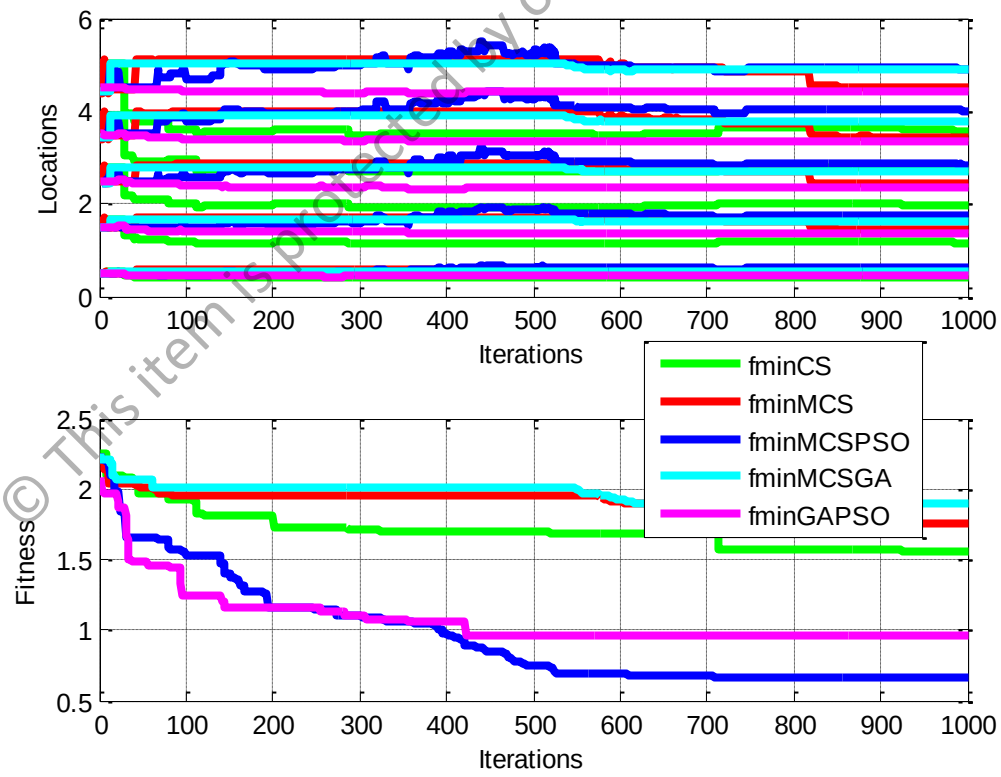


Figure 5.2: Optimal Location and Total Fitness Curves for Weighted–Sum MCS Hybrids vs. others ($2N$ = 10, Uniform, maxIter = 1000)

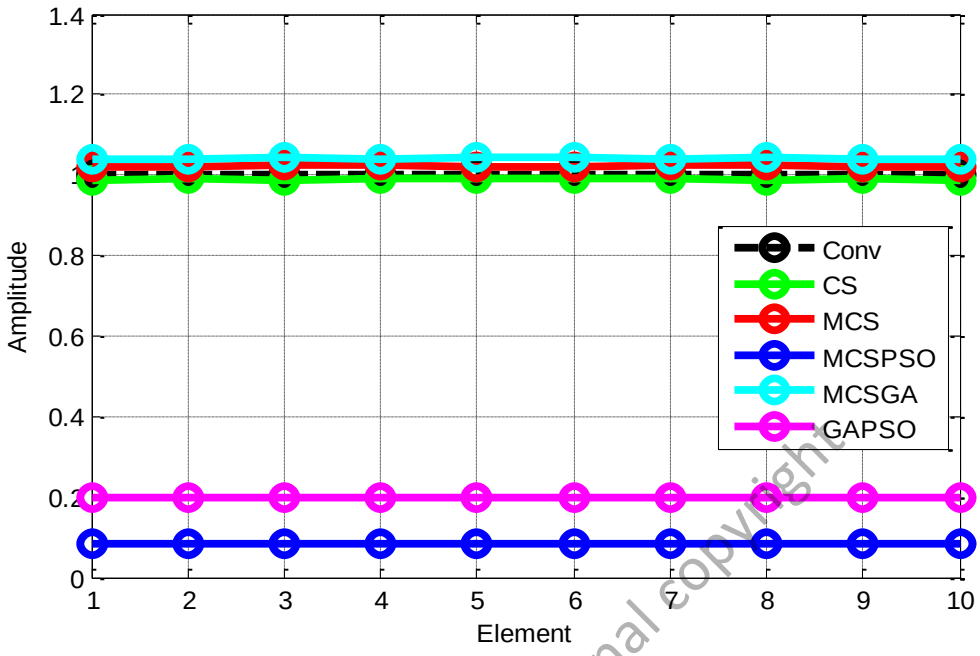


Figure 5.3: Optimal Amplitude for Weighted–Sum MCS Hybrids vs. others
($2N$ = 10, Uniform, maxIter = 1000)

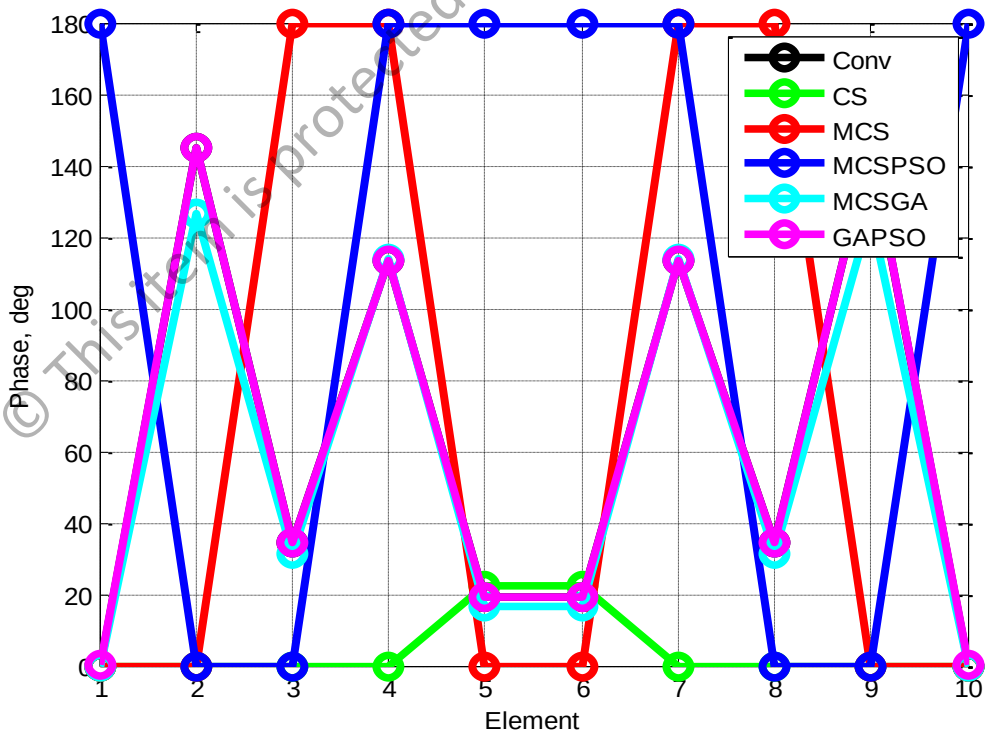


Figure 5.4: Optimal Phase for Weighted–Sum MCS Hybrids vs. others
($2N$ = 10, Uniform, maxIter = 1000)

Table 5.1 precisely conscripts the optimal location variations for all the tested weighted–sum optimizers for $2N$ = 10 linear array after 1000 iterations. The MCSPSO hybrid algorithm evidently generates the largest position variations compared to the conventional array which, is between |±0.1119| and |±0.3954| followed by the MCSGA counterpart. Moreover, the MCSPSO array has the lowest optimal excitation amplitude which is 0.0844, hence the biggest amplitude differences compared to the conventional array as tabulated in Table 5.2.Looking on the aspect of the array excitation phase, the MCSPSO hybrid algorithm has the magnitude domain of $[0^o 180^o]$ for all $2N$ = 10 linear array elements.

Table 5.3 shows that the MCSPSO–based optimizer generates the biggest optimal phase deviations compared to the conventional array, which are between 34.5848^o and 180^o.

Table 5.1: Optimal Location for Weighted–Sum MCS Hybrids vs. others
($2N$ = 10, Uniform, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.4055	±1.1638	±1.9590	±2.7393	±3.5674
MCS [Mantegna]	±0.4769	±1.4518	±2.4265	±3.4521	±4.5170
MCSPSO [Mantegna]	±0.6119	±1.7458	±2.8382	±4.0097	±4.8954
MCSGA [Mantegna]	±0.5429	±1.6285	±2.7025	±3.7869	±4.8959
GAPSO	±0.4489	±1.3770	±2.3313	±3.3593	±4.4227

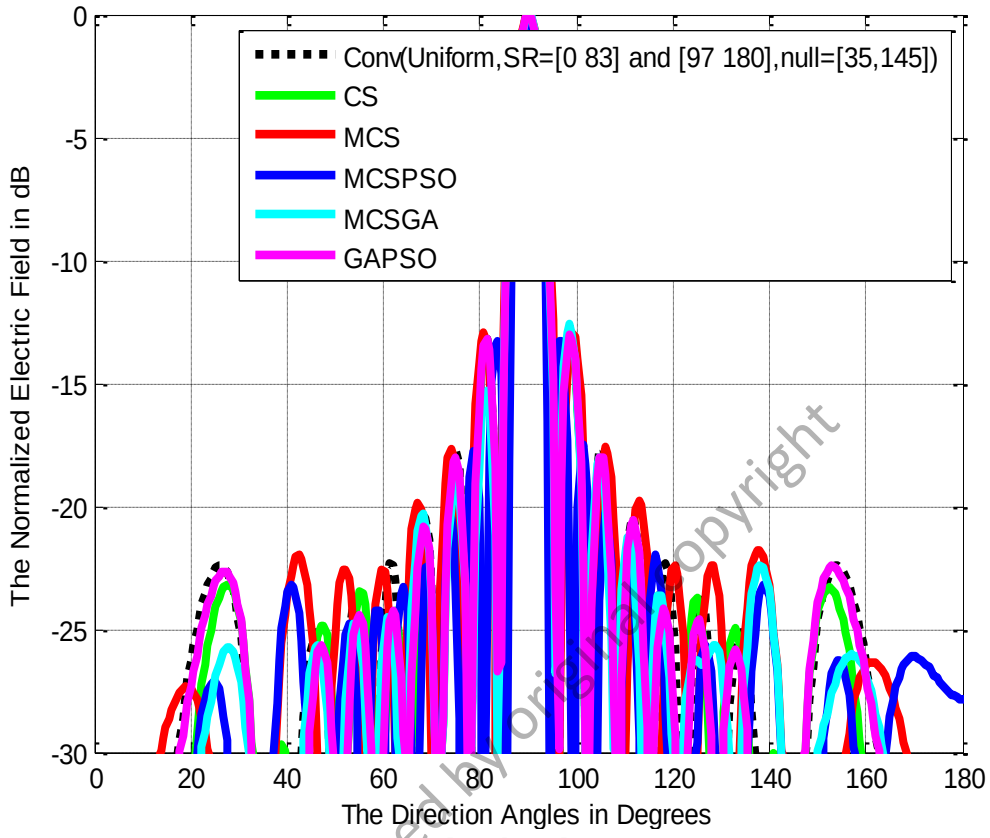
Table 5.2: Optimal Amplitude for Weighted–Sum MCS Hybrids vs. others
($2N$ = 10, Uniform, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
CS [Mantegna]	0.9873	0.9906	0.9881	0.9908	0.9911
MCS [Mantegna]	1.0195	1.0200	1.0271	1.0241	1.0193
MCSPSO [Mantegna]	0.0844	0.0844	0.0844	0.0844	0.0844
MCSGA [Mantegna]	1.0378	1.0375	1.0445	1.0388	1.0418
GAPSO	0.2000	0.2000	0.2000	0.2000	0.2000

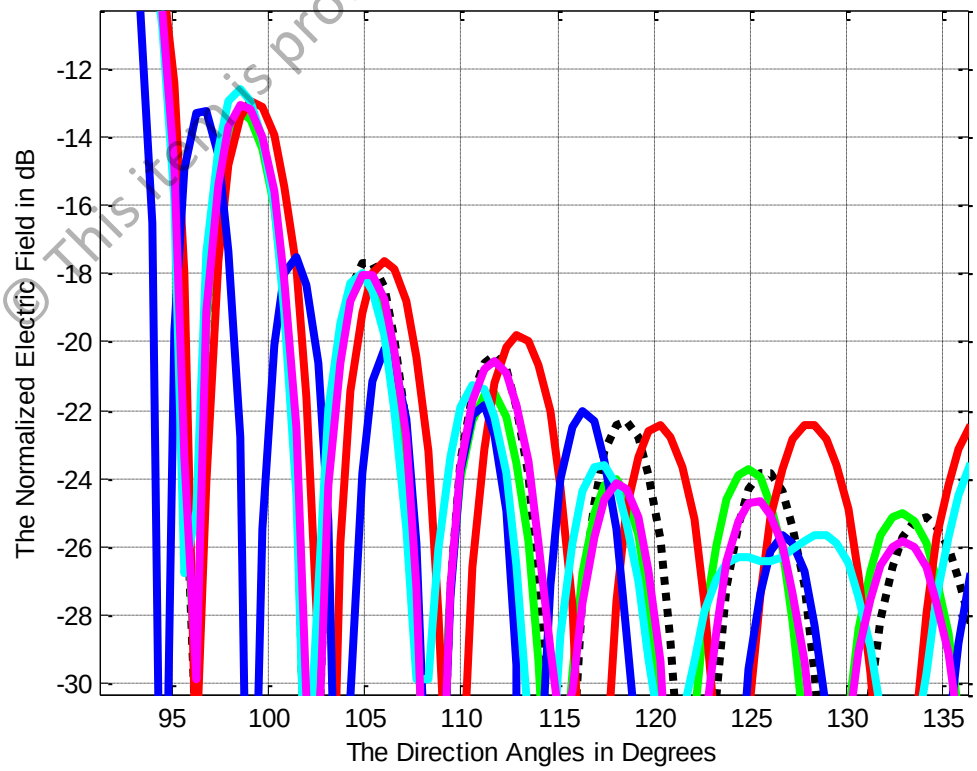
Table 5.3: Optimal Phase for Weighted–Sum MCS Hybrids vs. others
($2N$ = 10, Uniform, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
ϕ_n	0^o	144.8614^o	34.5848^o	113.2354^o	19.0016^o
CS [Mantegna]	0^o	0^o	0^o	0^o	22.4561^o
MCS [Mantegna]	0^o	0^o	180^o	180^o	0^o
MCSPSO [Mantegna]	180^o	0^o	0^o	180^o	180^o
MCSGA [Mantegna]	0^o	126.4717^o	31.1931^o	114.1964^o	16.7280^o
GAPSO	0.2285^o	144.8629^o	34.6929^o	113.3098^o	19.1002^o

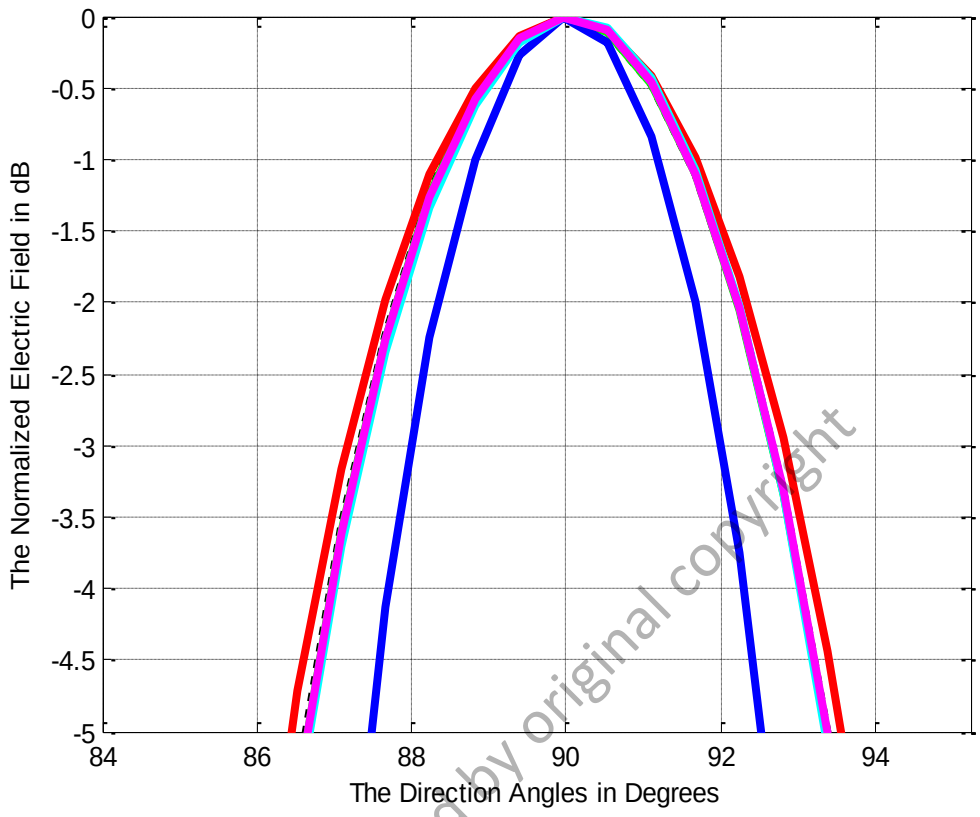
Secondly, a more substantial simulation is done on the $2N$ = 20 linear array with main beam radiates at the desired direction angle of 90^o and two prescribed interferers at direction angles of 35^o and 145^o, respectively. In this simulation, MCSPSO, MCSGA, MCS and original CS algorithms deploy the Mantegna's α–stable distribution method, host nest (population) = 30, length step factor = L/100 or 0.01, and α = 2.0 (Lévy flight Gaussian distribution). All the MCS–based algorithms have a dynamic P_a magnitude domain of [0.01 0.25] and a dynamic w magnitude domain of [0.95 1.05], respectively. Both the MCSPSO and GAPSO optimizers deploy the PSO algorithm with the dynamic random particle velocity domain of [–0.1 +0.1]. Furthermore, the MCSGA and GAPSO algorithms use the GA optimizer with the gene crossover probability, P_c = 90% or 0.9, and gene mutation probability, P_m = 10% or 0.1.



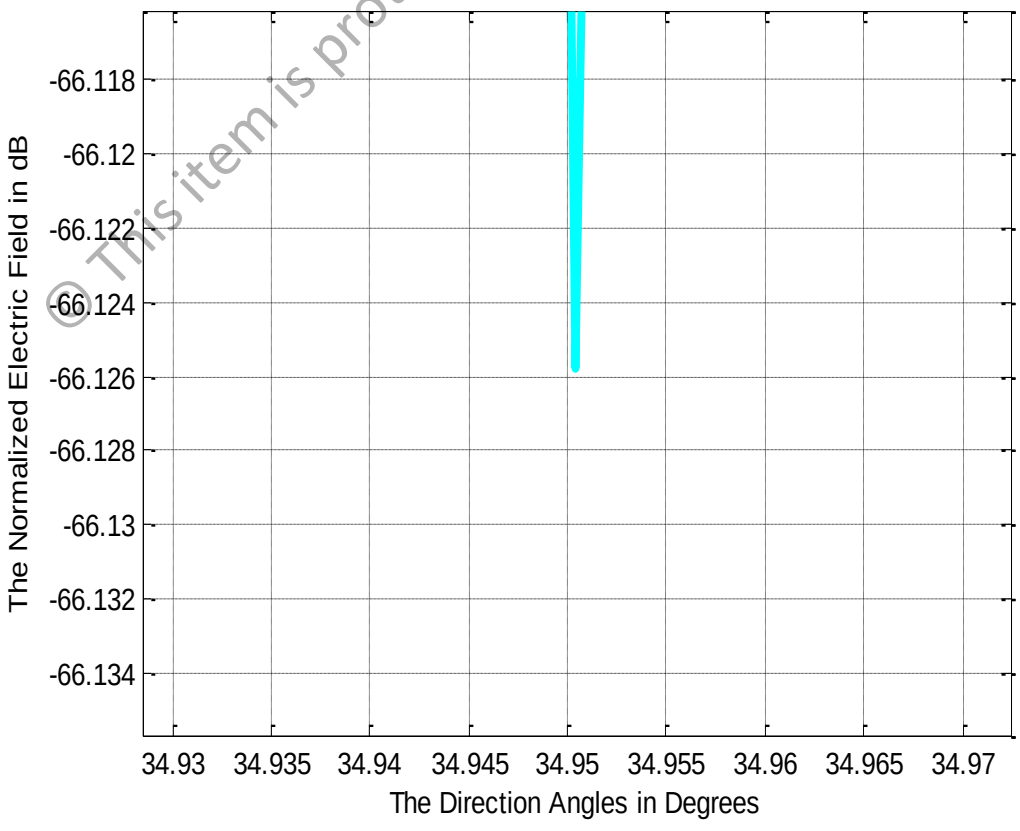
(a) Normalized Radiation Pattern



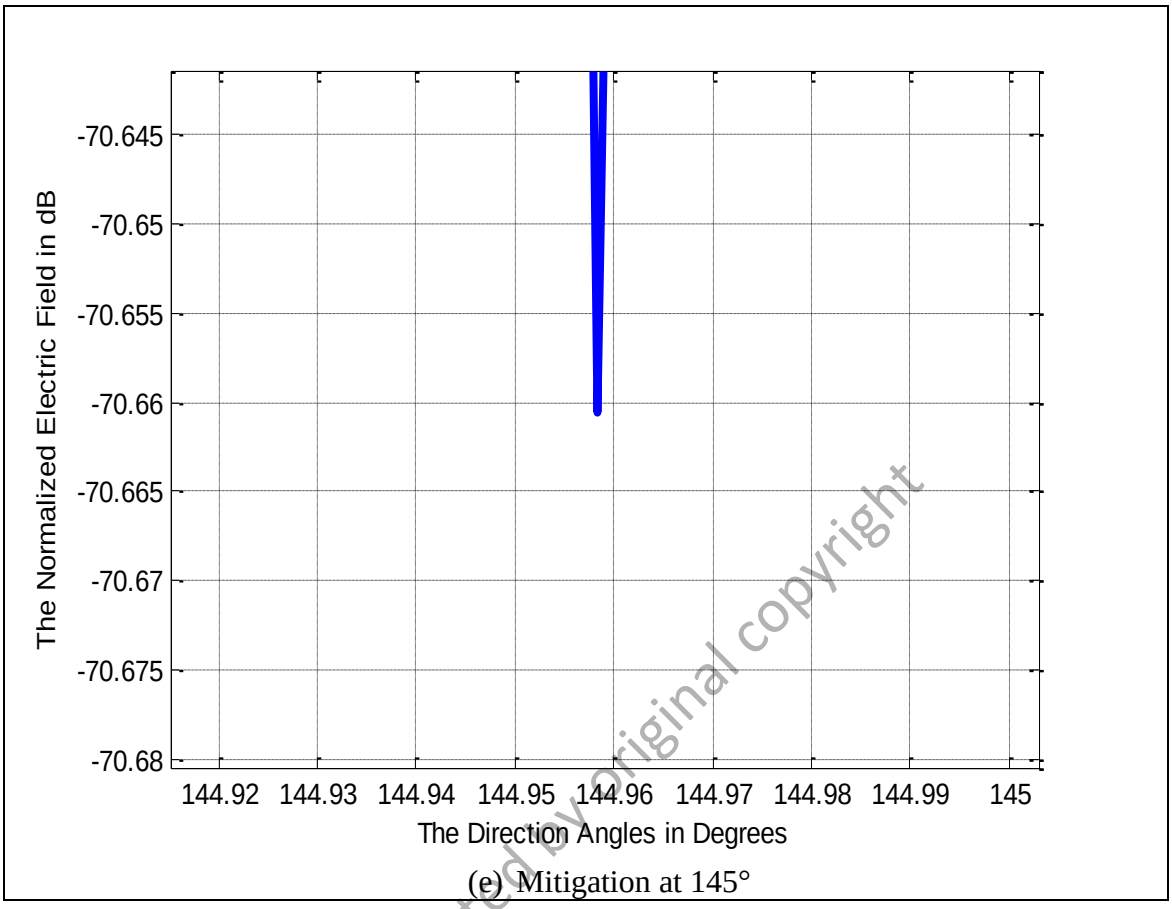
(b) Average SLL Suppression



(c) HPBW Pattern



(d) Mitigation at 35°



(e) Mitigation at 145°

Figure 5.5: Normalized Pattern for Weighted–Sum MCS Hybrids vs. others
(2*N* = 20, Uniform, Null = [35°, 145°], maxIter = 1000)

Figure 5.5(a) shows that the MCSPSO–based array outperforms other arrays in SLL suppression particularly between the [0^o 83^o] and [97^o 180^o] regions, respectively. In this case, the MCSPSO hybrid algorithm generates the SLL between 0.047 dB and 3.826 dB below the conventional array as depicted in Figure 5.5(b). Moreover, the MCSPSO–based array as in Figure 5.5(c) demonstrates the highest radiation intensity at the main beam with the smallest HPBW of $92^o - 88^o = 4^o$ with the directivity of 11.5831 dB. This is trailed by the MCSGA counterpart with the calculated HPBW of $92.67^o - 87.39^o = 5.28^o$, and the directivity of 11.3074 dB. In addition, Figure 5.5(d) shows that the proposed MCSPSO algorithm has the significant null mitigation, with the measurements of –70.661 dB nearly at 144.96°, whereas Figure 5.5(e) indicates that the

proposed MCSGA counterpart has the remarkable null mitigation of –66.126 dB at about 34.95°.

Based on Figure 5.6, the MCSPSO hybrid–optimizer produces the largest optimal location fluctuations with the lowest weighted–sum f_{min} convergence of 1.0994. Besides, the proposed MCSPSO hybrid–optimizer generates the lowest optimal amplitude measurements for all $2N = 20$ array elements as shown in Figure 5.7. In another aspect, Figure 5.8 portrays that the MCSPSO based–optimizer produces the biggest optimal phase variations for the $2N = 20$ linear array with two prescribed nulls.

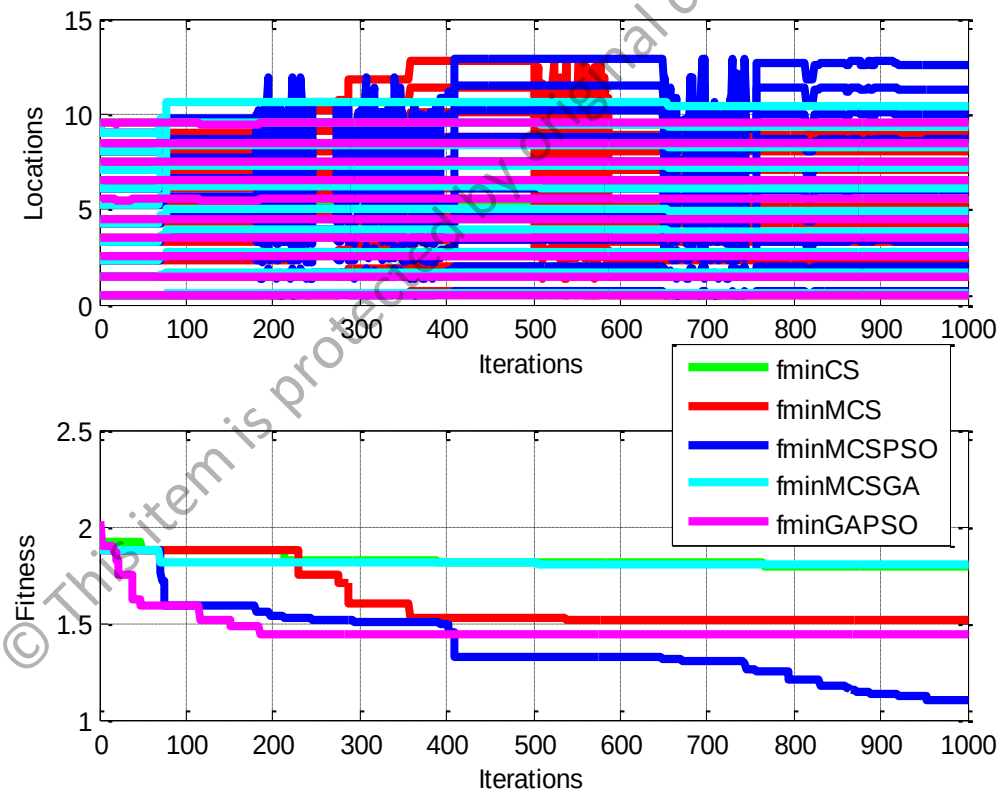


Figure 5.6: Optimal Location and Total Fitness Curves for Weighted–Sum MCS Hybrids vs. others
($2N = 20$, Uniform, Null = [35°, 145°], maxIter = 1000)

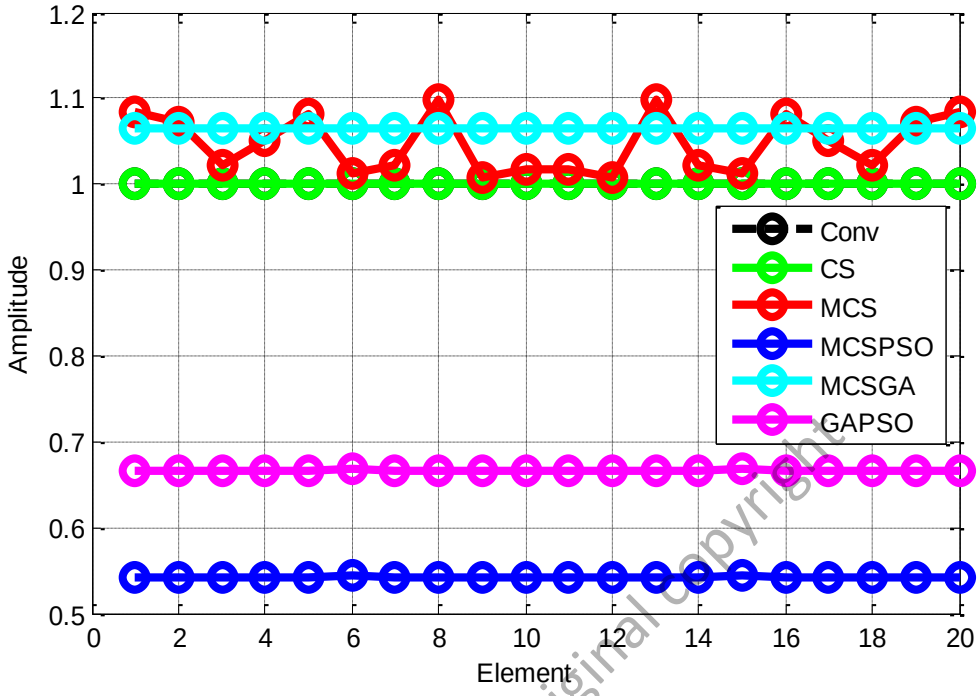


Figure 5.7: Optimal Amplitude for Weighted–Sum MCS Hybrids vs. others
(2*N* = 20, Uniform, Null = [35°, 145°], maxIter = 1000)

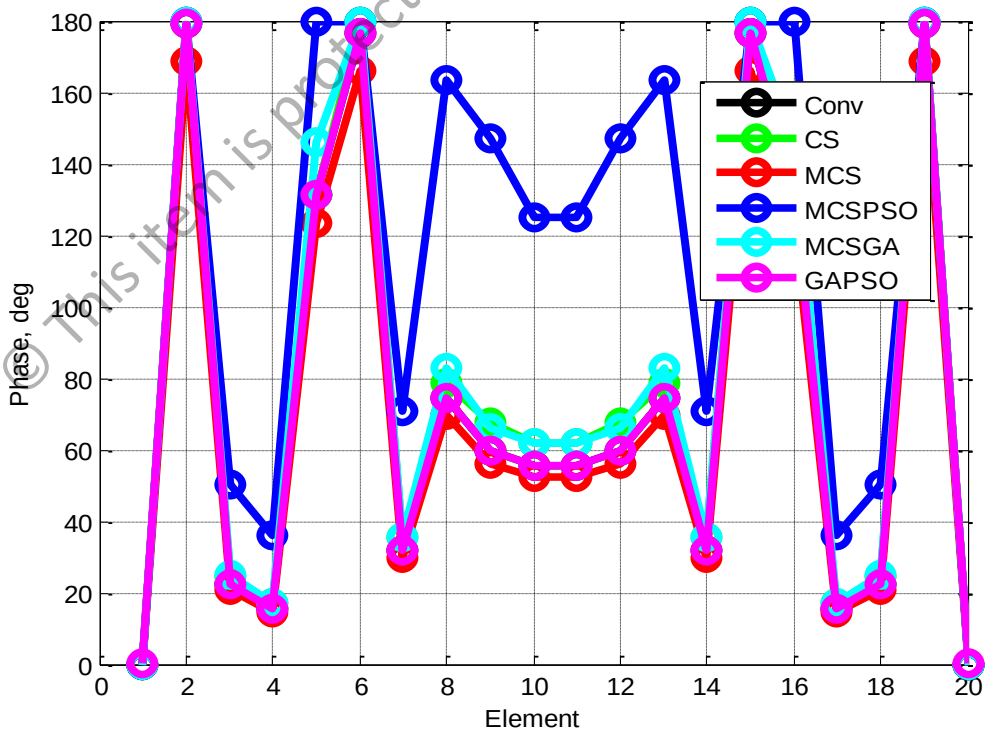


Figure 5.8: Optimal Phase for Weighted–Sum MCS Hybrids vs. others
(2*N* = 20, Uniform, Null = [35°, 145°], maxIter = 1000)

As mentioned earlier, the MCSPSO hybrid algorithm executes the largest optimal location oscillations with the measurements between |±0.1620| and |±3.1255| compared to the conventional array for all $2N = 20$ linear array as enlisted in Table 5.4. In addition, the MCSPSO hybrid algorithm has the lowest optimal amplitude deviations compared to the conventional array between 0.5427 and 0.5432 for all $2N = 20$ array elements as presented in Table 5.5. As can be seen in Table 5.6, the MCSPSO hybrid algorithm has the largest optimal phase deviations compared to the conventional array between 0^o and 88.9306^o. In this case, an element has the excitation phase of 0^o and three elements have the excitation phase of 180^o. This indicates that the proposed MCSPSO–based optimizer is capable to search the optimal solutions (optimal phases) to the minimum and maximum extents, which improves the scanning capability with low side lobes, a narrow main beam, and well–mitigated nulls.

Table 5.4: Optimal Location for Weighted–Sum MCS Hybrids vs. others
($2N = 20$, Uniform, Null = [35°, 145°], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
CS [Mantegna]	±0.4996	±1.4969	±2.4978	±3.4928	±4.4918
MCS [Mantegna]	±0.4703	±1.4108	±2.3514	±3.2919	±4.2325
MCSPSO [Mantegna]	±0.6620	±1.9907	±3.3192	±4.6481	±5.9780
MCSGA [Mantegna]	±0.5499	±1.6497	±2.7488	±3.8493	±4.9491
GAPSO	±0.5093	±1.5102	±2.5140	±3.5113	±4.5116
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
CS [Mantegna]	±5.4933	±6.4910	±7.4846	±8.4826	±9.4916
MCS [Mantegna]	±5.1730	±6.1136	±7.0541	±7.9947	±8.9353
MCSPSO [Mantegna]	±7.3084	±8.6368	±9.9648	±11.2959	±12.6255
MCSGA [Mantegna]	±6.0489	±7.1487	±8.2485	±9.3460	±10.4481
GAPSO	±5.5158	±6.5135	±7.5116	±8.5057	±9.5117

Table 5.5: Optimal Amplitude for Weighted–Sum MCS Hybrids vs. others
(2*N* = 20, Uniform, Null = [35°, 145°], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
CS [Mantegna]	1.0004	1.0012	1.0009	1.0003	1.0003
MCS [Mantegna]	1.0853	1.0716	1.0213	1.0516	1.0819
MCSPSO [Mantegna]	0.5428	0.5429	0.5427	0.5428	0.5427
MCSGA [Mantegna]	1.0656	1.0656	1.0656	1.0656	1.0656
GAPSO	0.6666	0.6666	0.6666	0.6666	0.6666
Element	**6**	**7**	**8**	**9**	**10**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
CS [Mantegna]	1.0008	1.0014	1.0002	1.0010	1.0010
MCS [Mantegna]	1.0117	1.0228	1.0986	1.0071	1.0161
MCSPSO [Mantegna]	0.5432	0.5428	0.5428	0.5430	0.5430
MCSGA [Mantegna]	1.0656	1.0656	1.0656	1.0656	1.0656
GAPSO	0.6666	0.6666	0.6666	0.6666	0.6666

Table 5.6: Optimal Phase for Weighted–Sum MCS Hybrids vs. others
(2*N* = 20, Uniform, Null = [35°, 145°], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
ϕ_n	0^o	179.3495^o	22.3307^o	15.5222^o	131.2926^o
CS [Mantegna]	0^o	180^o	24.3387^o	17.2827^o	131.4126^o
MCS [Mantegna]	0^o	168.6877^o	21.0032^o	14.5994^o	123.4877^o
MCSPSO [Mantegna]	0^o	180^o	50.3968^o	36.2676^o	180^o
MCSGA [Mantegna]	0^o	180^o	24.8503^o	17.2736^o	146.1066^o
GAPSO	0.0948^o	179.4246^o	22.3733^o	15.5829^o	131.3076^o
Element	**6**	**7**	**8**	**9**	**10**
ϕ_n	176.3662^o	31.7982^o	74.2622^o	59.5501^o	55.6673^o
CS [Mantegna]	180^o	35.3167^o	78.8409^o	67.8246^o	61.9853^o
MCS [Mantegna]	165.8818^o	29.9079^o	69.8475^o	56.0100^o	52.3581^o
MCSPSO [Mantegna]	180^o	70.7653^o	163.1928^o	146.8616^o	124.8258^o
MCSGA [Mantegna]	180^o	35.3860^o	82.6413^o	66.2692^o	61.9484^o
GAPSO	176.3985^o	31.8015^o	74.3409^o	59.6257^o	55.7297^o

5.1.2 Global Pareto Front Approach

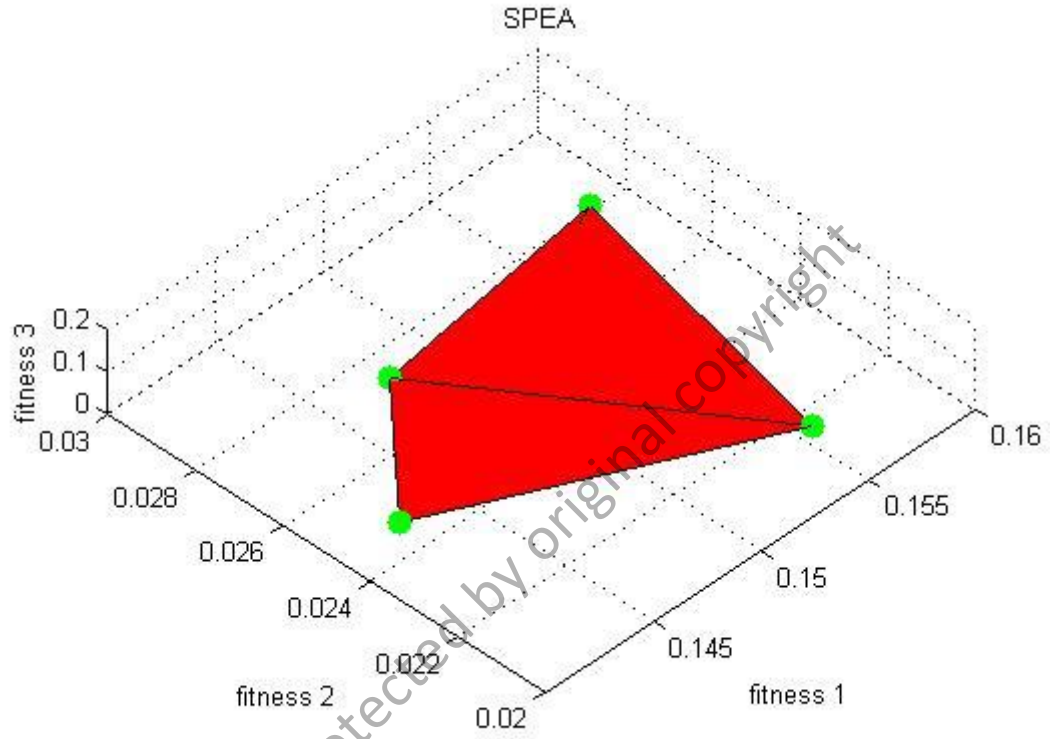
In the first global Pareto simulation, the proposed MCSSPEA, MCSHCSPEA and MCSPSOSPEA optimizers with Mantegna's algorithm as the selected α–stable

distribution method, host nest (population) = 20, discovery rate, P_a = 25% or 0.25, length step factor = L/100 or 0.01, and α = 2.0 (Lévy flight Gaussian distribution) are analyzed on $2N$ = 20 linear array. In this case, the proposed MCSSPEA, MCSHCSPEA and MCSPSOSPEA optimizers with the dynamic inertia weight, w magnitude domain of [0.80 1.20] are directly compared with the standard SPEA and conventional arrays in the normal uniform window. The bigger, w magnitude domain leads the MCS algorithms to gain a more control on the Lévy flight motions with a heavy–tailed and α–stable distribution towards the best host nest (candidate solution) in search space.

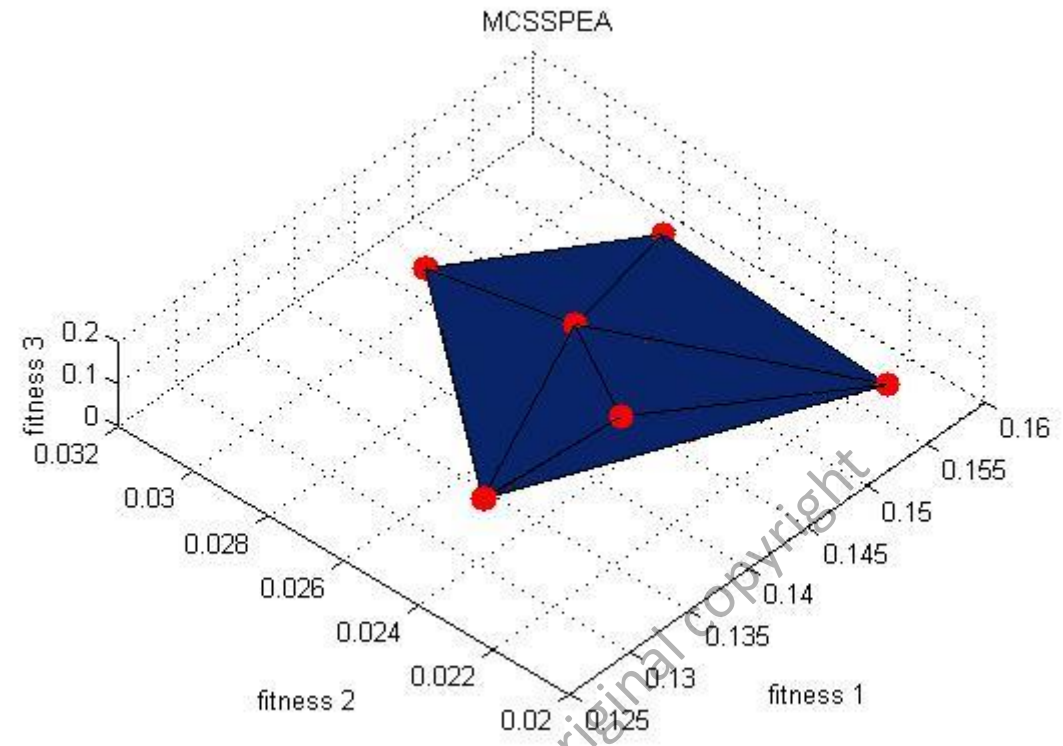
The MCSPSOSPEA optimizer uses the particle swarm optimization (PSO) algorithm with the dynamic random particle velocity domain of [–0.1 +0.1]. The MATLAB simulation executes 1000 iterations of Pareto optimization to find the set of three decision variables or optimal solutions simultaneously, which are linear array excitation locations, amplitudes, and phases.

The spread Pareto fitness domain are $f_1 \in [0.12, 0.18]$, $f_2 \in [0.018, 0.032]$, and $f_3 \in [0.0, 0.2]$, respectively, as shown in Figure 6.9(b) – (e). Since, this is a three–dimensional (3D) Pareto fronts plot, hypervolume calculation are done for all the tested optimizers. Ideally, the hypervolume near or equals to zero is preferred for a global Pareto minimization process. In this case, the SPEA has the smallest hypervolume, which is 0.6361×10^{-5} unit^3, which is followed by the MCSHCSPEA with the hypervolume of 1.5907×10^{-5} unit^3, the MCSPSOSPEA with the hypervolume of 2.9925×10^{-5} unit^3, and the MCSSPEA with the hypervolume of 3.5477×10^{-5} unit^3, respectively. This indicates that at overall Pareto MO trade–offs aspect, both the MCSPSOSPEA and MCSHCSPEA algorithms have some non–dominated points that are dominated by SPEA, which has the smallest hypervolume. In sum, all the optimizers have very small hypervolumes, which close to zero after going through a MO

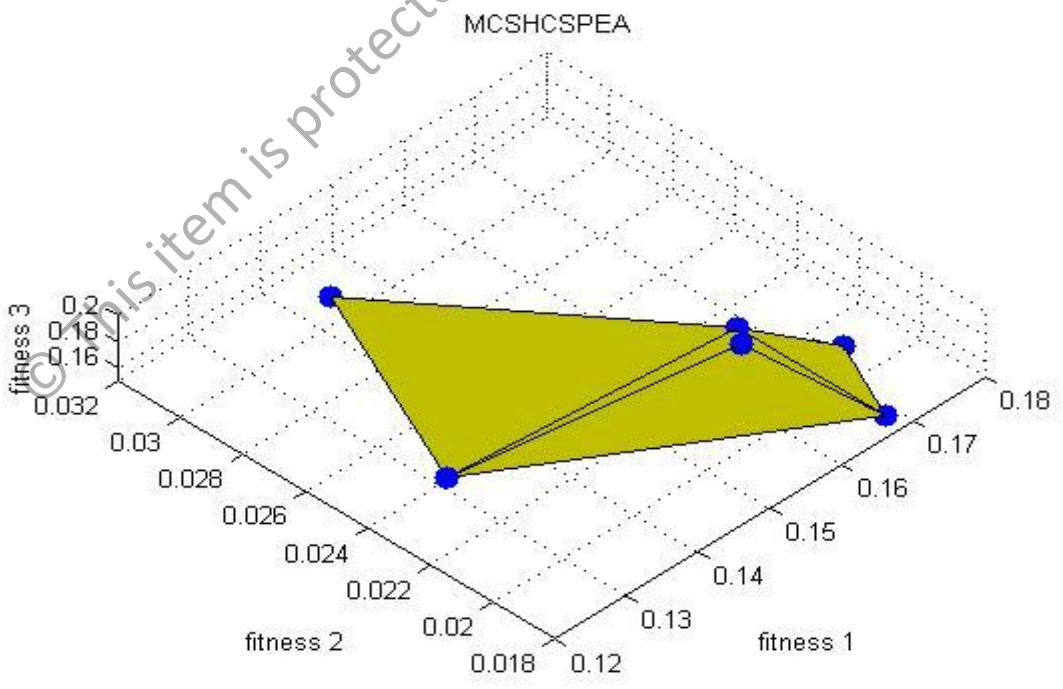
minimization process. Besides, there are small hypervolume differences among the optimizers, which are less than 3.0×10^{-5} unit^3 (estimated difference between the largest and smallest hypervolumes).



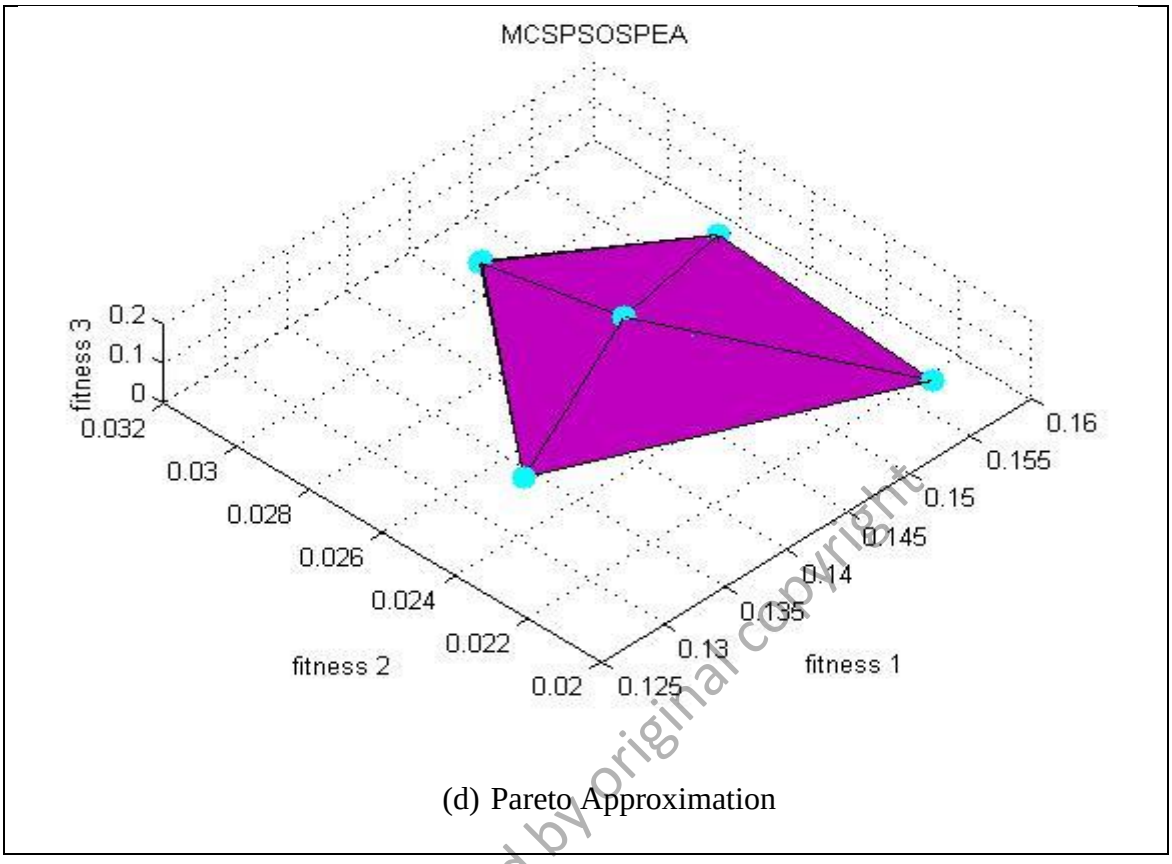
(a) Pareto Approximation



(b) Pareto Approximation



(c) Pareto Approximation



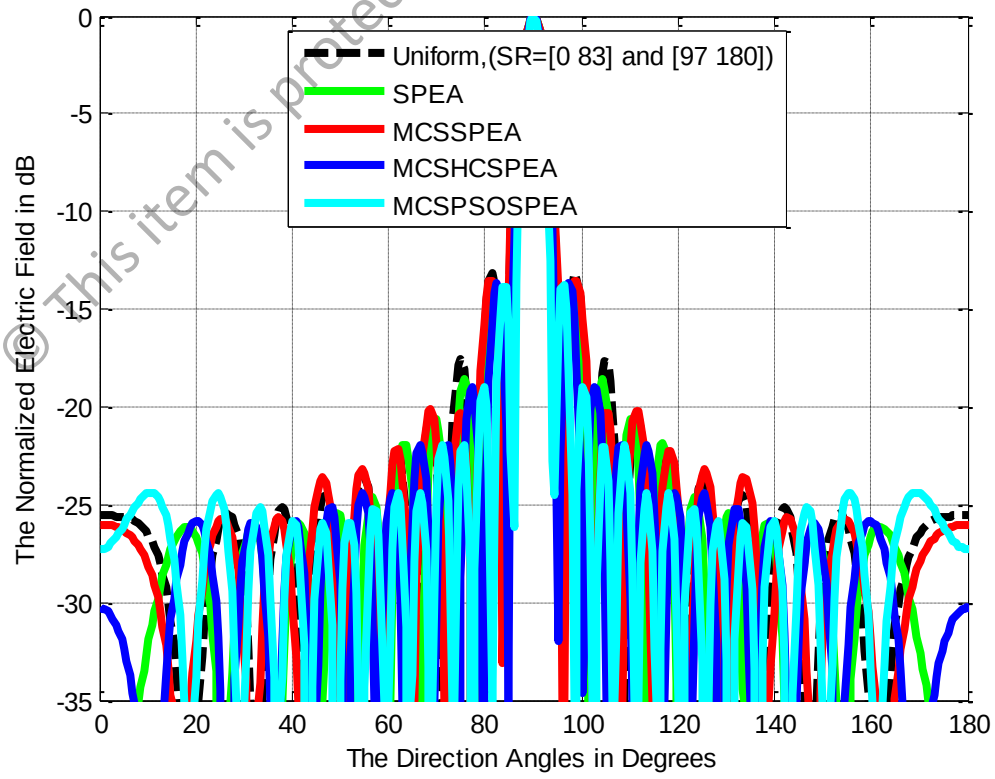
(d) Pareto Approximation

Figure 5.9: Strength Pareto Evolutionary Algorithm (SPEA) Front Approximations ($2N$ = 20, Uniform, maxIter = 1000)

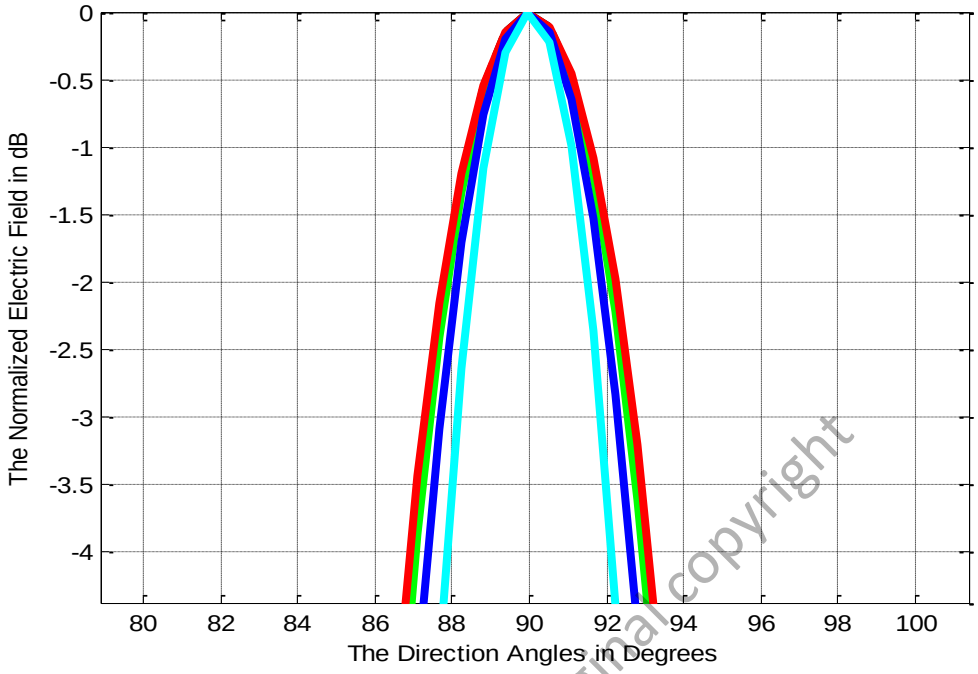
Based on Figure 5.10(a), the normalized radiation pattern for the hypothesized MCSPSOSPEA optimizer evidently outperforms other competitors by having the lowest average SLL suppression and highest directivity of the main beam. The half–power beamwidth (HPBW) is the angular separation in which the magnitude of the radiation pattern decreases by 50% (or –3 dB) from the peak of the main beam. Figure 5.10(b) shows the pattern for MCSPSOSPEA optimizers similarly decrease to –3 dB at 88.1426^{o} and 91.8487^{o}. Hence, the HPBW is $91.8487^{o} - 88.1426^{o} = 3.7061^{o}$. Figure 5.10(b) also depicts the MCSPSOSPEA–based array generates the highest directivity measurement of 7.4124 dB. This is followed by the MCSHCSPEA counterpart with the directivity of 6.9339 dB. Besides, the postulated MCSPSOSPEA optimizer has the best average SLL suppression between –0.545 dB and –2.232 dB compared to the

conventional array within the suppression regions of [40^o 83^o] and [97^o 140^o] as shown in Figure 5.10(c).

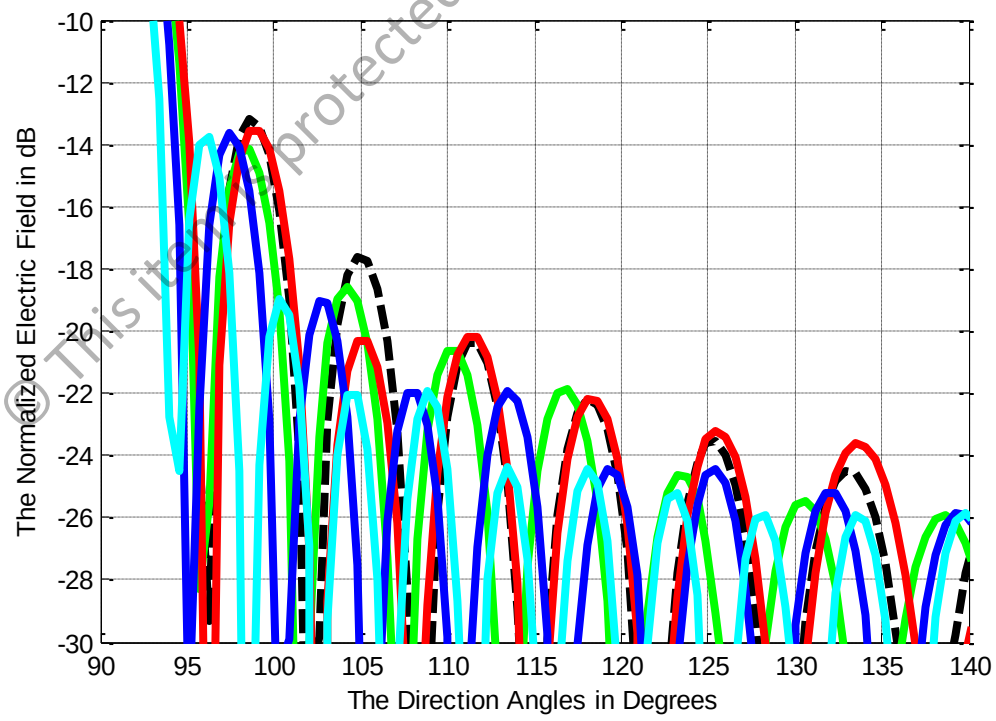
Figure 5.11 shows that compared to other opponents, the MCSPSOSPEA algorithm fluctuates furthest its excitation amplitude with respect to the conventional array. Furthermore, it also has the biggest optimal phase deviations compared to other rivals as depicted in Figure 5.12. All of these findings become the key factor for the MCSPSOSPEA algorithm applying the selected non–dominated solutions to design a linear array, which can suppress the average lowest side lobes while preserving the main beam intensity. Based on the selected non–dominated solutions aspect, the MCSPSOSPEA algorithm has better Pareto front trade–offs especially with respect to f_1 and f_3 despite having a bigger hypervolume, which leads to the best directivity and smallest half–power beamwidth (HPBW).



(a) Normalized Radiation Pattern



(b) HPBW Pattern



(c) Average SLL Suppression

Figure 5.10: Normalized Pattern for SPEA–based Arrays
($2N$ = 20, Uniform, maxIter = 1000)

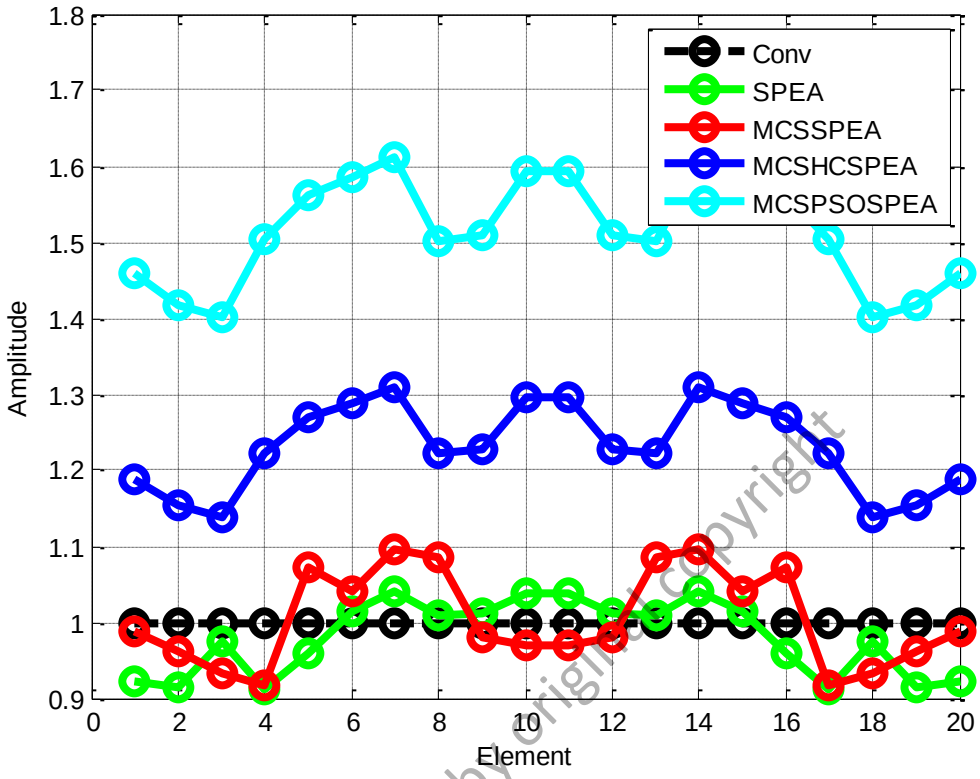


Figure 5.11: Optimal Amplitude for SPEA–based Arrays
(2*N* = 20, Uniform, maxIter = 1000)

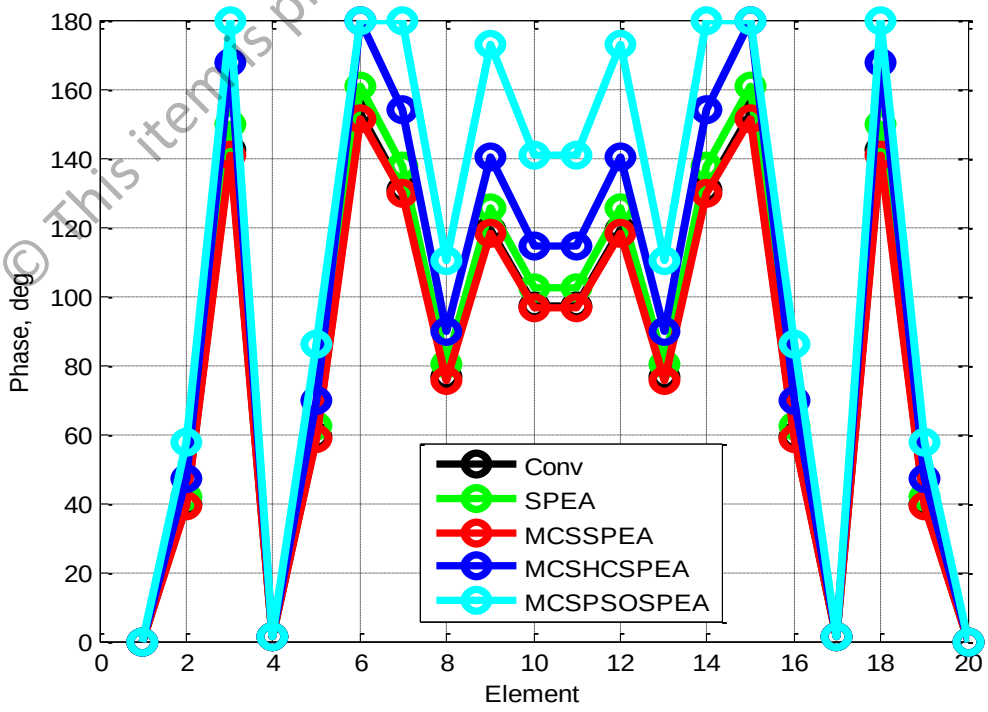


Figure 5.12: Optimal Phase for SPEA–based Arrays
(2*N* = 20, Uniform, maxIter = 1000)

In this experiment, the MCSPSOSPEA algorithm generates 11 non–dominated solutions whereas MCSHCSPEA algorithm generates 12 non–dominated solutions, respectively. Table 5.7 shows the fitness trade–off values of the selected non–dominated solutions for all the four tested Pareto algorithms for the comparison purposes as shown in Figure 5.10, Figure 5.11, and Figure 5.12, respectively. It can be seen that the MCSPSOSPEA has the smallest relative values of Pareto front with respect to f_1 and f_3 after running 1000 iterations of a Pareto MO minimization.

Table 5.7: Selected Optimal Pareto Fitness for SPEA–based Arrays
(2*N* = 20, Uniform, maxIter = 1000)

Fitness	f_1	f_2	f_3
SPEA	0.1535	0.0206	0.1406
MCSSPEA	0.1461	0.0248	0.1277
MCSHCSPEA	0.1442	0.0307	0.1505
MCSPSOSPEA	0.1349	0.0249	0.1277

Table 5.8: Optimal Location for SPEA–based Arrays
(2*N* = 20, Uniform, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[**$\lambda/2$**]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
SPEA	±0.5262	±1.5786	±2.6310	±3.6835	±4.7359
MCSSPEA	±0.4956	±1.4869	±2.4782	±3.4695	±4.4608
MCSHCSPEA	±0.5888	±1.7663	±2.9438	±4.1214	±5.2989
MCSPSOSPEA	±0.7241	±2.1724	±3.6206	±5.0688	±6.5171
Element	**6**	**7**	**8**	**9**	**10**
X_n **[**$\lambda/2$**]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
SPEA	±5.7883	±6.8407	±7.8931	±8.9456	±9.9980
MCSSPEA	±5.4521	±6.4434	±7.4347	±8.4260	±9.4173
MCSHCSPEA	±6.4765	±7.6540	±8.8315	±10.0091	±11.1866
MCSPSOSPEA	±7.9653	±9.4135	±10.8618	±12.3100	±13.7582

Table 5.9: Optimal Amplitude for SPEA–based Arrays
($2N$ = 20, Uniform, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
SPEA	0.9224	0.9142	0.9751	0.9119	0.9584
MCSSPEA	0.9887	0.9629	0.9332	0.9178	1.0716
MCSHCSPEA	1.1873	1.1529	1.1385	1.2220	1.2689
MCSPSOSPEA	1.4603	1.4180	1.4002	1.5029	1.5606
Element	**6**	**7**	**8**	**9**	**10**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
SPEA	1.0142	1.0401	1.0098	1.0105	1.0375
MCSSPEA	1.0402	1.0965	1.0863	0.9800	0.9692
MCSHCSPEA	1.2886	1.3098	1.2218	1.2280	1.2954
MCSPSOSPEA	1.5848	1.6109	1.5026	1.5103	1.5932

Table 5.10: Optimal Phase for SPEA–based Arrays ($2N$ = 20, Uniform, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
ϕ_n	0^o	39.8173^o	142.2216^o	1.0740^o	59.3696^o
SPEA	0^o	41.9045^o	149.6768^o	1.1303^o	62.4817^o
MCSSPEA	0^o	39.4706^o	140.9834^o	1.0647^o	58.8527^o
MCSHCSPEA	0^o	46.8864^o	167.4716^o	1.2647^o	69.9101^o
MCSPSOSPEA	0^o	57.6648^o	180.0000^o	1.5555^o	85.9812^o
Element	**6**	**7**	**8**	**9**	**10**
ϕ_n	152.6461^o	130.6266^o	76.3159^o	119.2334^o	97.2496^o
SPEA	160.6477^o	137.4740^o	80.3164^o	125.4836^o	102.3473^o
MCSSPEA	151.3171^o	129.4893^o	75.6515^o	118.1953^o	96.4029^o
MCSHCSPEA	179.7468^o	153.8180^o	89.8650^o	140.4021^o	114.5152^o
MCSPSOSPEA	180.0000^o	180.0000^o	110.5235^o	172.6782^o	140.8404^o

As enlisted in Table 5.8, the MCSPSOSPEA hybrid algorithm executes the largest optimal location oscillations with respect to $\lambda/2$ compared to the conventional array. Precisely, the MCSPSOSPEA has the oscillations between $|\pm 0.2241|$ and $|\pm 4.2582|$ for all $2N$ = 20 linear antenna array elements. Table 5.9 shows that the MCSPSOSPEA hybrid algorithms also attain the biggest optimal amplitude deviations compared to the conventional array. In this case, the MCSPSOSPEA has the deviations between 0.4002 and 0.6109. Besides, Table 5.10 indicates that the postulated

MCSPSOSPEA algorithm produces the largest optimal phase variations compared to the conventional array with the variations between 0^o and 53.4448^o, respectively.

In sum, the dynamic hybridization of MCS with PSO and SPEA in MCSPSOSPEA algorithm is proven deliver a better control of Lévy flight motion of cuckoo in searching for a host nest (optimal solution), which provides the brood with a higher probability of survival. Hence, the proposed MCSPSOSPEA hybrid algorithm performs a better minimization trade–offs using the selected non–dominated solutions with Pareto fitness functions tabulated in Table 5.7 by producing a bigger diversity of optimal excitation position, amplitude, and phase values.

The next experiment compares the proposed MCSSPEA, MCSHCSPEA and MCSPSOSPEA hybrid algorithms along with the standard SPEA and conventional arrays under the Dolph–Chebyshev signal processing window with the relative SLL, R = –30 dB for the $2N$ = 20 linear array. In this experiment, all the SPEA–based optimizers with Mantegna's algorithm as the selected α–stable distribution method, host nest (population) = 20, discovery rate, P_a = 25% or 0.25, length step factor = L/100 or 0.01, and α–stable = 2.0 (Lévy flight Gaussian distribution) are simulated for 1000 iterations on $2N$ = 20 linear antenna array. All the proposed MCSSPEA, MCSHCSPEA, and MCSPSOSPEA optimizers apply the dynamic inertia weight, w with the magnitude domain of [0.80 1.20]. Furthermore, the proposed MCSPSOSPEA algorithm uses the PSO velocity of particle (cuckoo) updating mechanism with the domain of [–0.1 +0.1]. This smaller domain is to avoid an excessive Lévy flight motion towards the global best (*gbest*) solution in N–dimensional search space. The excessive Lévy flight motion produces too much excitation solution fluctuations compared to the conventional array, which are not optimum for equiripple side lobes suppression in a "highly–sensitive" Dolph–Chebyshev signal processing window.

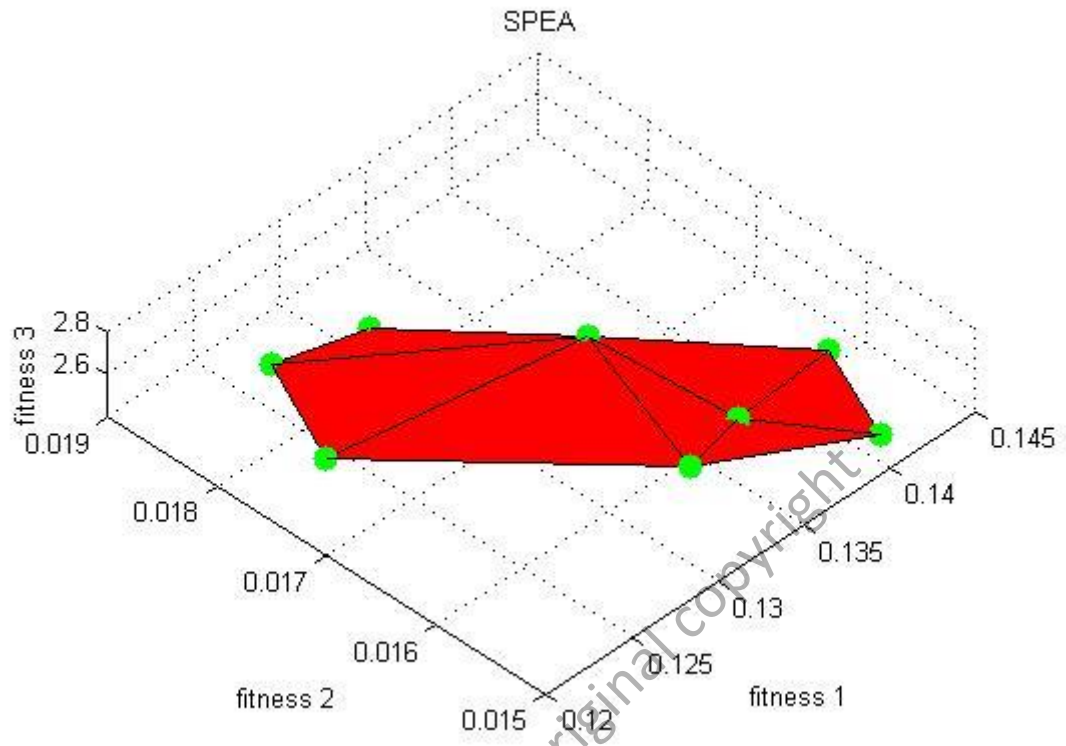
Figure 5.13(a) shows the initial Pareto non–dominated solutions with respect to the three fitness functions. The spread Pareto fitness domain are $f_1 \in$ [0.05, 0.25], $f_2 \in$ [0.01, 0.06], and $f_3 \in$ [0.0, 2.8], respectively, as depicted in Figure 5.9(b) – (e). In this study, the MCSPSOSPEA has the hypervolume of 0.2041×10^{-4} unit^3, the MCSSPEA has the hypervolume of 0.2083×10^{-4} unit^3, and the standard MCSHCSPEA has the hypervolume of 7.3015×10^{-4} unit^3, respectively. It is found that each of the tested algorithms has a very small hypervolume (closes to zero) due to the Pareto minimization. Besides, all the optimizers have similar hypervolumes with the differences less than 7.3×10^{-4} unit^3 (estimated difference between the largest and smallest hypervolumes).

The MCSPSOSPEA algorithm outperforms other competitors in side lobes suppression particularly within the [0^o 60^o] and [120^o 180^o] suppression regions as shown in Figure 5.14(a). In addition, the MCSPSOSPEA algorithm as in Figure 5.14(b) generates the smallest HPBW of $92.67^o - 87.33^o = 5.34^o$ with the highest directivity of 7.8733 dB. This is followed by the MCSHCSPEA algorithm with the HPBW of $93.00^o - 87.00^o = 6.00^o$ with the directivity of 7.7769 dB. The hybrid MCSSPEA–based array has the directivity of 7.2022 dB whereas the SPEA counterpart has the directivity of 7.1259 dB, respectively. Precisely, the MCSPSOSPEA algorithm generates the best SLL suppression between 0.18 dB and 0.98 dB relatively lower than the conventional linear array. The MCSSPEA counterpart is the second best optimizer in the SLL suppression between 0.28 dB and 0.78 dB relatively lower than the conventional linear array. Figure 5.14(c) shows that the MCSPSOSPEA algorithm generates the lowest side lobes radiation intensity for $2N = 20$ linear antenna array within the Dolph–Chebyshev signal filtering window.

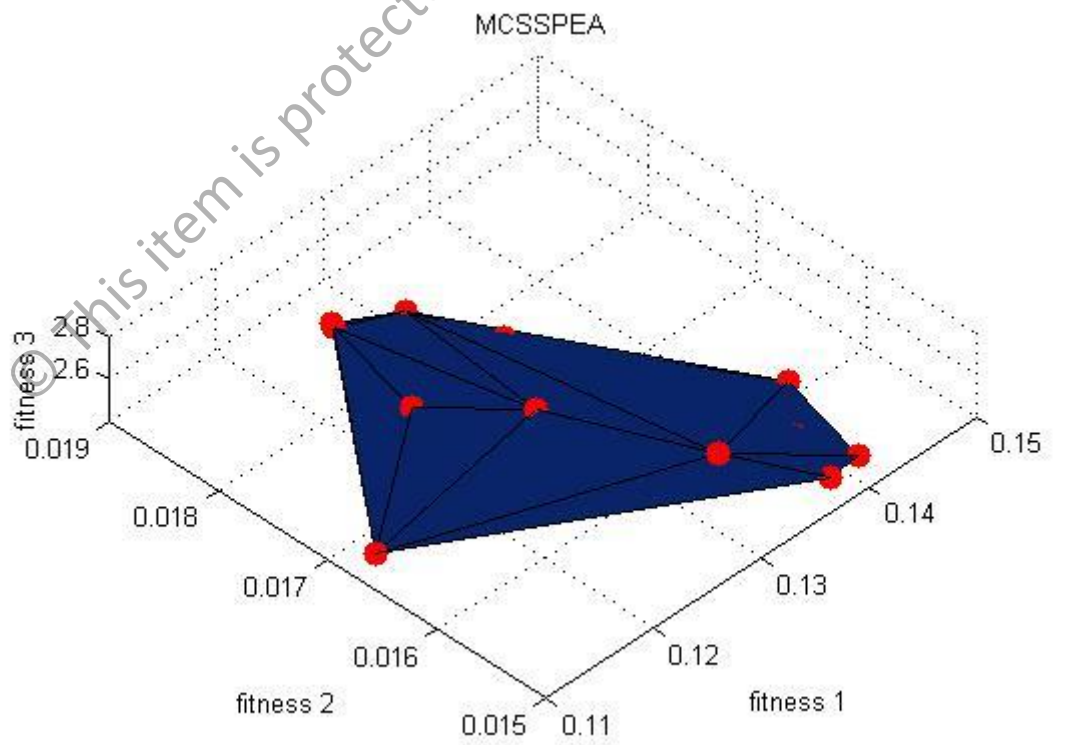
Table 5.11: Selected Optimal Pareto Fitness for SPEA–based Arrays
(2*N* = 20, Dolph–Chebyshev, maxIter = 1000)

Fitness	f_1	f_2	f_3
SPEA	0.1403	0.0151	2.4990
MCSSPEA	0.1388	0.0156	2.4983
MCSHCSPEA	0.1286	0.0165	2.5288
MCSPSOSPEA	0.1270	0.0186	2.4206

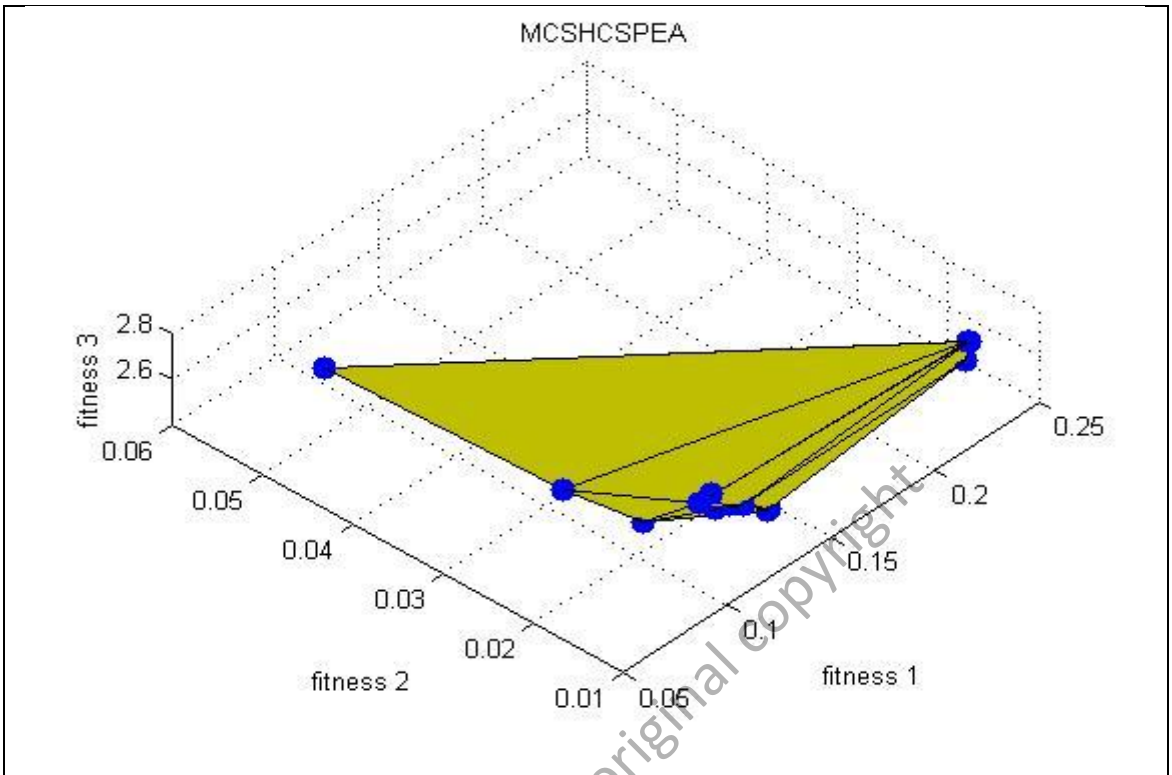
In this experiment, both the MCSSPEA and MCSPSOSPEA hybrid algorithms generate 19 non–dominated solutions whereas MCSHCSPEA algorithm produces 14 non–dominated solutions. Table 5.11 shows the fitness trade–off values of the selected non–dominated solutions for all the four tested SPEA–based Pareto algorithms, respectively. The MCSPSOSPEA generates the smallest values of fitness f_1 and f_3 after going through 1000 iterations of minimization process. Once again, looking at the selected non–dominated solutions aspect, the MCSPSOSPEA algorithm has better Pareto front trade–offs especially with respect to f_1 and f_3 despite having a bigger hypervolume than the standard SPEA counterpart, which leads to the best antenna directivity and smallest half–power beamwidth (HPBW).



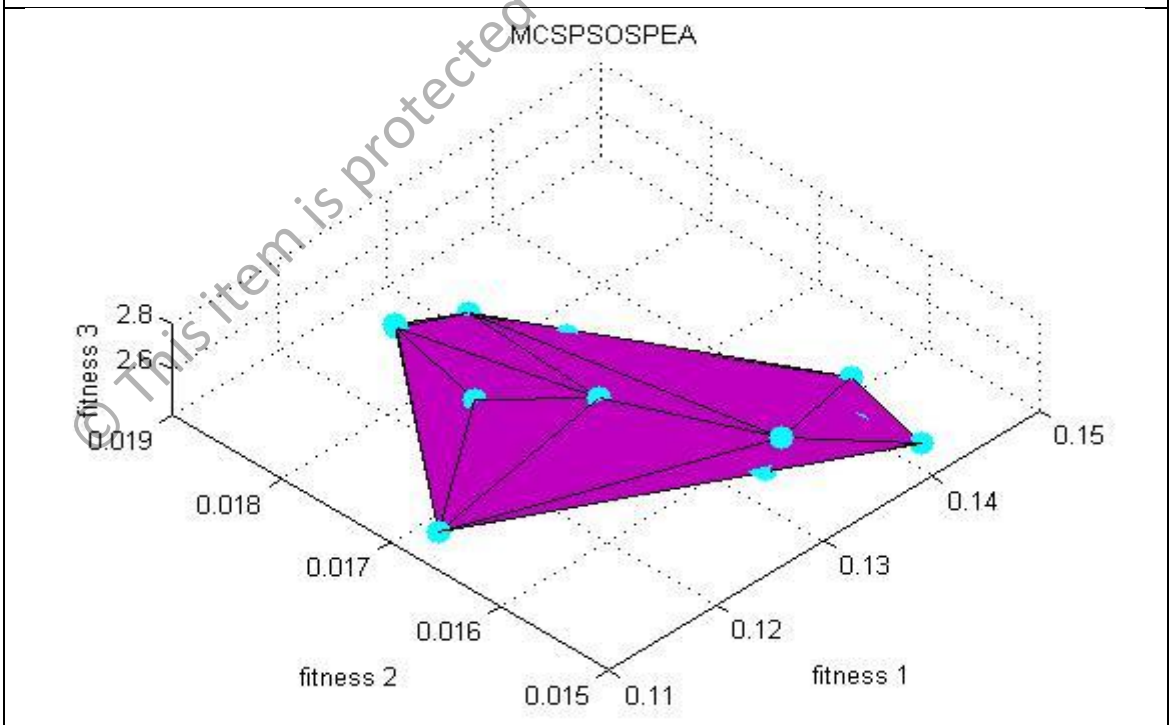
(a) Pareto Approximation



(b) Pareto Approximation

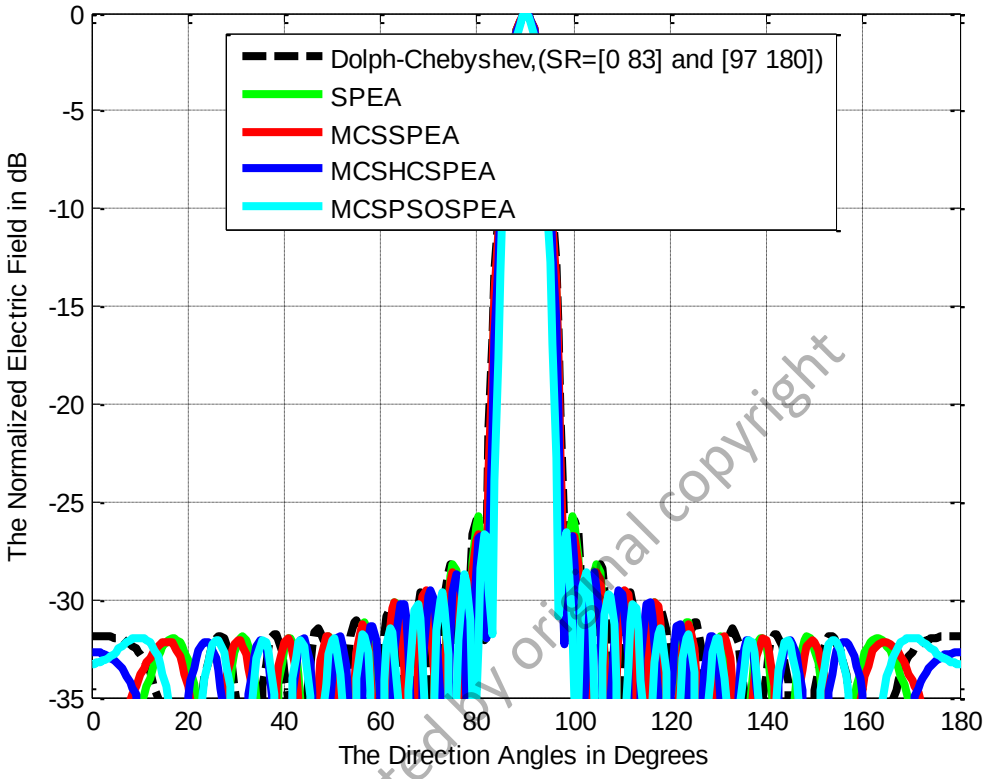


(c) Pareto Approximation

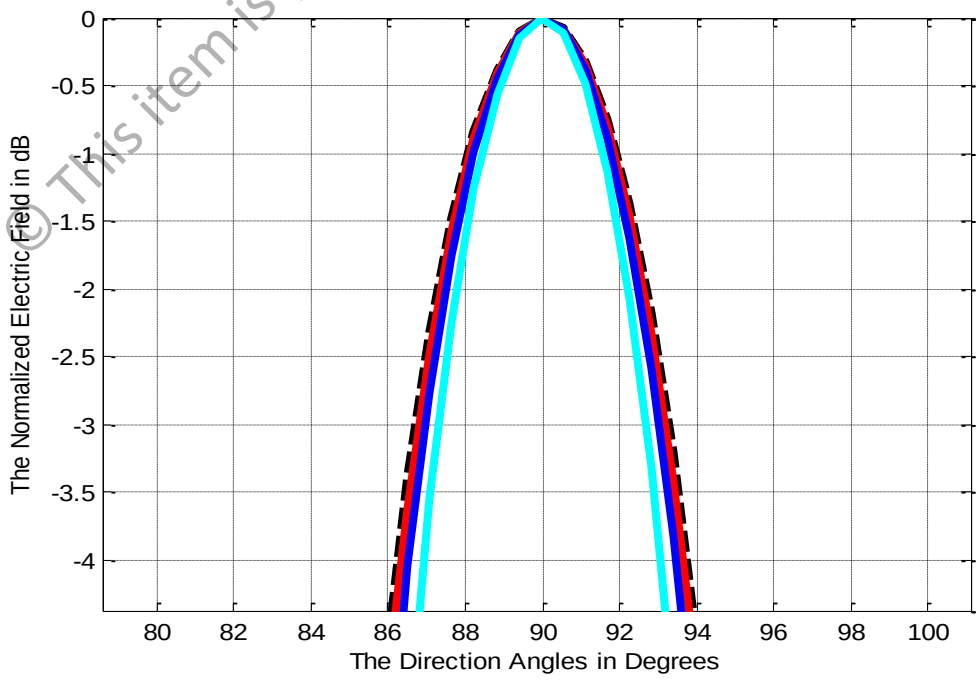


(d) Pareto Approximation

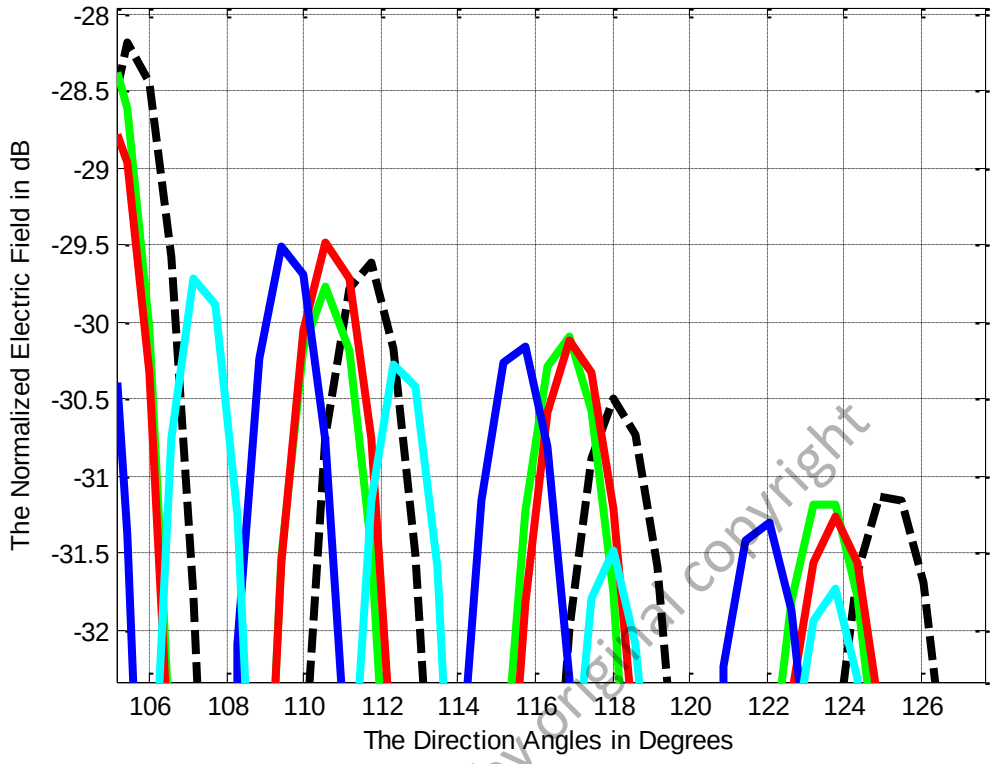
Figure 5.13: Strength Pareto Evolutionary Algorithm (SPEA) Front Approximations (2N = 20, Dolph–Chebyshev, maxIter = 1000)



(a) Normalized Radiation Pattern



(b) HPBW Pattern



(c) Average SLL Suppression

Figure 5.14: Normalized Pattern for SPEA–based Arrays
($2N$ = 20, Dolph–Chebyshev, maxIter = 1000)

Furthermore, Figure 5.15 shows that the MCSPSOSPEA hybrid algorithm has the biggest optimal Dolph–Chebyshev amplitude deviations compared to the conventional linear array. This is followed by the MCSHCSPEA, MCSSPEA and SPEA algorithms. Similarly for optimal phase values as in Figure 5.16, the MCSPSOSPEA hybrid algorithm generates the biggest fluctuations followed by the MCSHCSPEA counterpart. Nevertheless, both the MCSSPEA and SPEA algorithms produce almost similar fluctuations in the excitation location, Dolph–Chebyshev amplitude and phase, respectively.

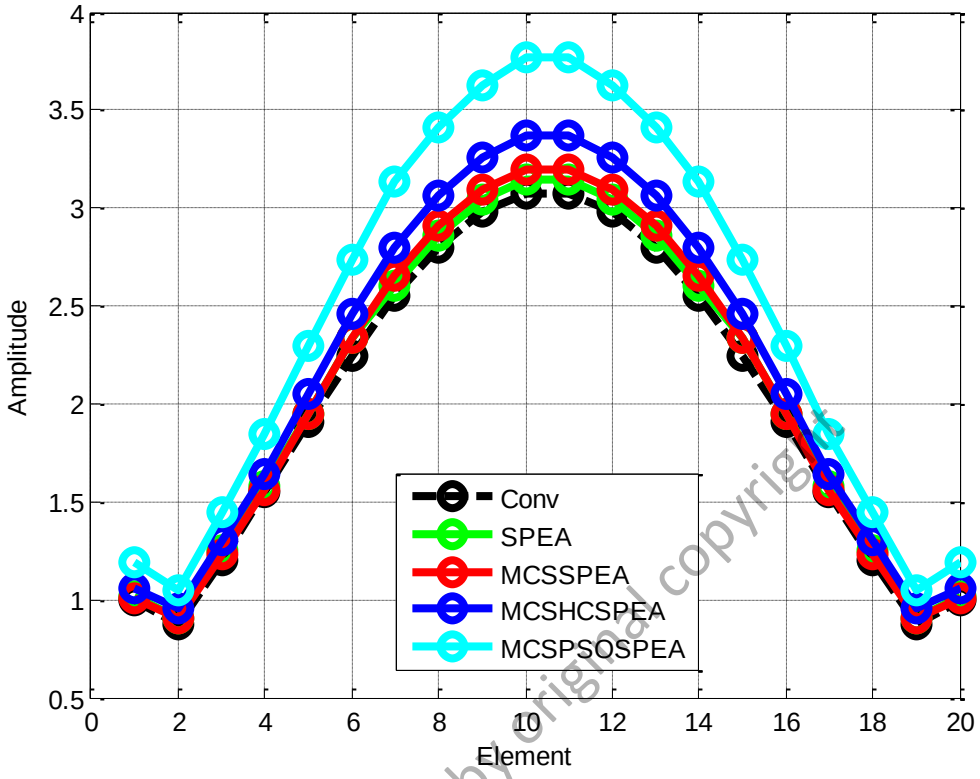


Figure 5.15: Optimal Amplitude for SPEA–based Arrays
($2N$ = 20, Dolph–Chebyshev, maxIter = 1000)

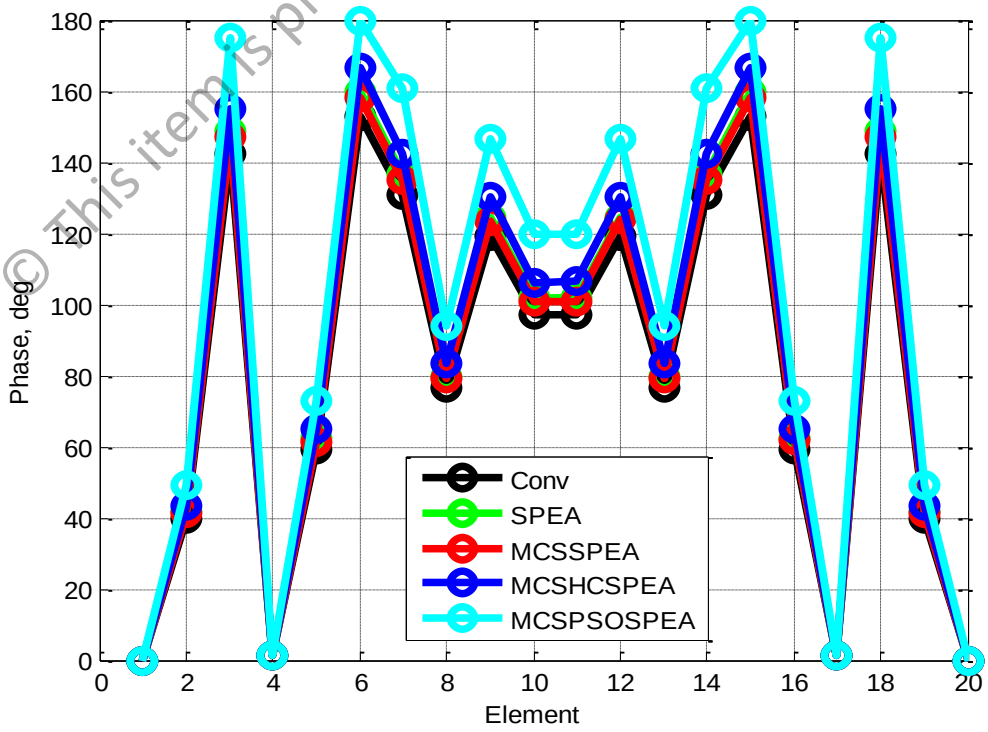


Figure 5.16: Optimal Phase for SPEA–based Arrays
($2N$ = 20, Dolph–Chebyshev, maxIter = 1000)

Table 5.12 clearly indicates that the proposed MCSPSOSPEA–based array produces the largest optimal element location fluctuations compared to the conventional linear array. Precisely, the MCSPSOSPEA fluctuations occurred between |±0.1157| and |±2.1974| for all $2N = 20$ linear array elements. This is followed by the MCSHCSPEA with the fluctuations between |±0.0467| and |±0.8866|. Based on Table 5.13, the MSCPSOSPEA hybrid optimizer generates the largest amplitude variations compared to the standard Dolph–Chebyshev window (as generated by the conventional linear array), which is between 0.1669 and 0.6967. This is followed by the MCSHCSPEA optimizer, with the fluctuations between 0.0591 and 0.2965 for all the $2N = 20$ linear array elements. In addition, the MCSPSOSPEA algorithm also generates the biggest optimal phase fluctuations compared to the conventional array, which are between 0^o and 32.8966^o as enlisted in Table 5.14. Once again, the MCSHCSPEA algorithm has the second largest optimal phase deviations which are between 0^o and 13.8217^o. In sum, the postulated MCSPSOSPEA algorithm is capable to explore furthest the optimal solutions in the search domain based on the biggest fluctuations as mentioned earlier. As a result, the MCSPSOSPEA algorithm produces the best diversity of Pareto front trade–offs, which surpass other competitors in suppressing side lobes while attaining the best directivity and smallest half–power beamwidth (HPBW) of main beam for $2N = 20$ linear antenna array within the Dolph–Chebyshev signal processing window.

Table 5.12: Optimal Location for SPEA–based Arrays
(2*N* = 20, Dolph–Chebyshev, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
SPEA	±0.5230	±1.5691	±2.6152	±3.6613	±4.7074
MCSSPEA	±0.5191	±1.5572	±2.5953	±3.6334	±4.6715
MCSHCSPEA	±0.5467	±1.6400	±2.7333	±3.8266	±4.9200
MCSPSOSPEA	±0.6157	±1.8470	±3.0783	±4.3096	±5.5409
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
SPEA	±5.7535	±6.7996	±7.8457	±8.8918	±9.9379
MCSSPEA	±5.7096	±6.7477	±7.7858	±8.8239	±9.8620
MCSHCSPEA	±6.0133	±7.1066	±8.2000	±9.2933	±10.3866
MCSPSOSPEA	±6.7722	±8.0035	±9.2348	±10.4661	±11.6974

Table 5.13: Optimal Amplitude for SPEA–based Arrays
(2*N* = 20, Dolph–Chebyshev, maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
A_n	1.0000	0.8771	1.2009	1.5497	1.9052
SPEA	1.0131	0.8992	1.2532	1.5752	1.9466
MCSSPEA	1.0056	0.9062	1.2335	1.5597	1.9431
MCSHCSPEA	1.0591	0.9544	1.2991	1.6427	2.0464
MCSPSOSPEA	1.1928	1.0440	1.4512	1.8436	2.2927
Element	**6**	**7**	**8**	**9**	**10**
A_n	2.2465	2.5522	2.8022	2.9793	3.0712
SPEA	2.3367	2.5988	2.8579	3.0419	3.1464
MCSSPEA	2.3370	2.6522	2.9066	3.0941	3.1976
MCSHCSPEA	2.4613	2.7933	3.0612	3.2587	3.3677
MCSPSOSPEA	2.7406	3.1333	3.4149	3.6315	3.7679

Table 5.14: Optimal Phase for SPEA–based Arrays
(2*N* = 20, Dolph–Chebyshev, maxIter = 1000)

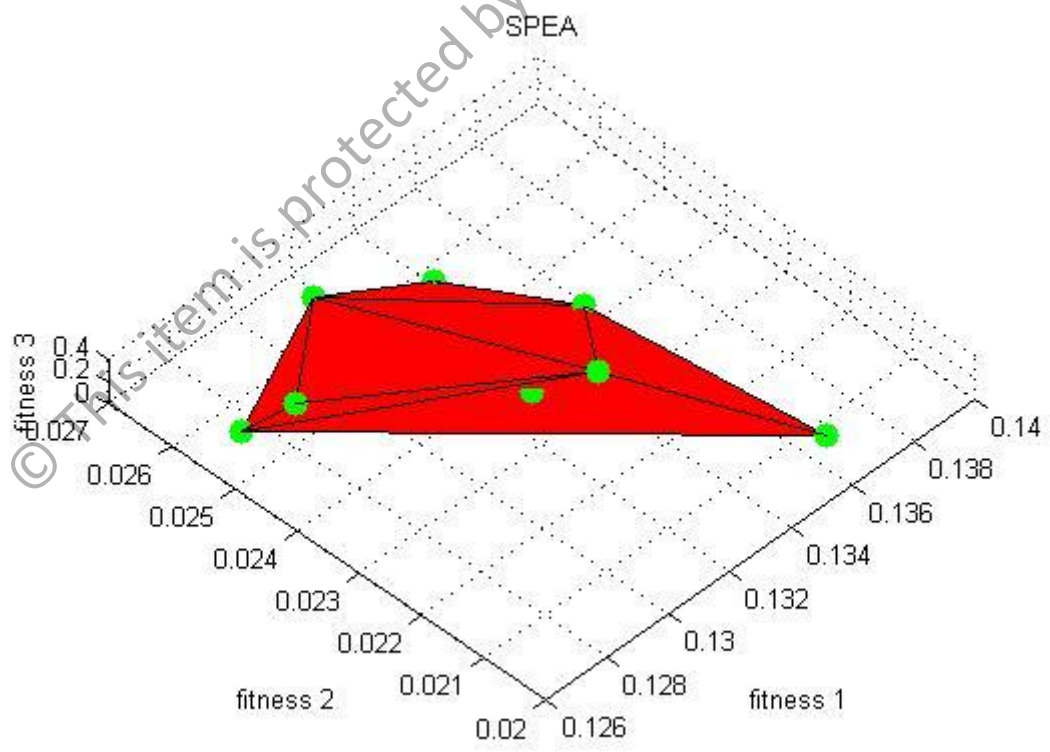
Element	**1**	**2**	**3**	**4**	**5**
ϕ_n	0^o	39.8173^o	142.2216^o	1.0740^o	59.3696^o
SPEA	0^o	41.6525^o	148.7767^o	1.1235^o	62.1060^o
MCSSPEA	0^o	41.2294^o	147.2655^o	1.1121^o	61.4751^o
MCSHCSPEA	0^o	43.4226^o	155.0994^o	1.1713^o	64.7454^o
MCSPSOSPEA	0^o	49.0272^o	175.1182^o	1.3225^o	73.1021^o
Element	**6**	**7**	**8**	**9**	**10**
ϕ_n	152.6461^o	130.6266^o	76.3159^o	119.2334^o	97.2496^o
SPEA	159.6816^o	136.6473^o	79.8334^o	124.7290^o	101.7319^o
MCSSPEA	158.0596^o	135.2593^o	79.0224^o	123.4620^o	100.6985^o
MCSHCSPEA	166.4678^o	142.4545^o	83.2261^o	130.0297^o	106.0553^o
MCSPSOSPEA	180.0000^o	160.8412^o	93.9682^o	146.8127^o	119.7439^o

In the last test, a more complex broadside case simulation is conducted where $2N = 20$ linear array radiates at the desired angle of 90^o with four predefined interferers at 30^o, 31^o, 149^o and 150^o, respectively. For a uniformity, all the tested SPEA–based optimizers deploy the Mantegna's algorithm as the α–stable distribution type, host nest (population) = 20, discovery rate or fraction probability, P_a = 25% or 0.25, length step factor = $L/100$ or 0.01, α = 2.0 (Lévy flight Gaussian distribution), and dynamic inertia weight, w domain of [0.80 1.20]. The proposed MCSSPEA, MCSHCSPEA and MCSPSOSPEA hybrid–based arrays are compared with the standard SPEA–based and conventional arrays in terms of SLL suppression, and interferermitigation whilst improving the main beam radiation through 1000 iterations of MATLAB simulations. The Pareto front non–dominated decision variables include the linear array excitation location, amplitude, and phase, respectively.

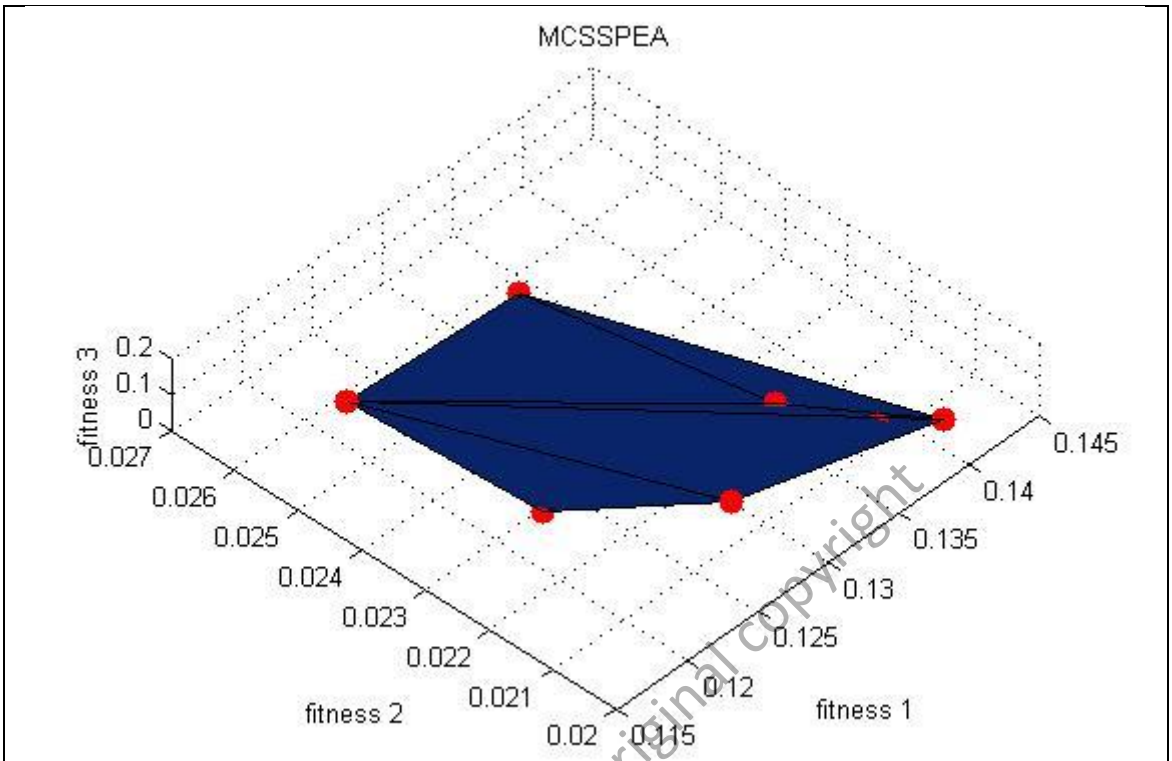
Figure 5.17(a) – (d) portray the spread Pareto fitness domain, which are bounded by $f_1 \in [0.115, 0.155]$, $f_2 \in [0.015, 0.045]$, and $f_3 \in [0.0, 0.4]$, respectively. In details, the SPEA counterpart has the smallest hypervolume of 4.3996×10^{-6} unit^3 whereas the postulated MCSPSOSPEA has the hypervolume of 6.3902×10^{-6} unit^3, followed by the MCSSPEA counterpart with the hypervolume of 9.6473×10^{-6} unit^3, and the MCSHCSPEA hybrid algorithm with the hypervolume of 2.6184×10^{-5} unit^3, respectively. In sum, all the hypervolume values are so small (near to zero) due to the Pareto fitness minimization process. Besides, all the tested SPEA optimizers have almost identical hypervolumes with the differences less than 2.2×10^{-5} unit^3 (estimated difference between the largest and smallest hypervolumes).

The MCSPSOSPEA–based array outperforms other arrays in SLL suppression particularly between the [60^o 83^o] and [97^o 120^o] regions, respectively as in Figure 5.18(a). Moreover, the MCSPSOSPEA–based array as in Figure 5.14(b)

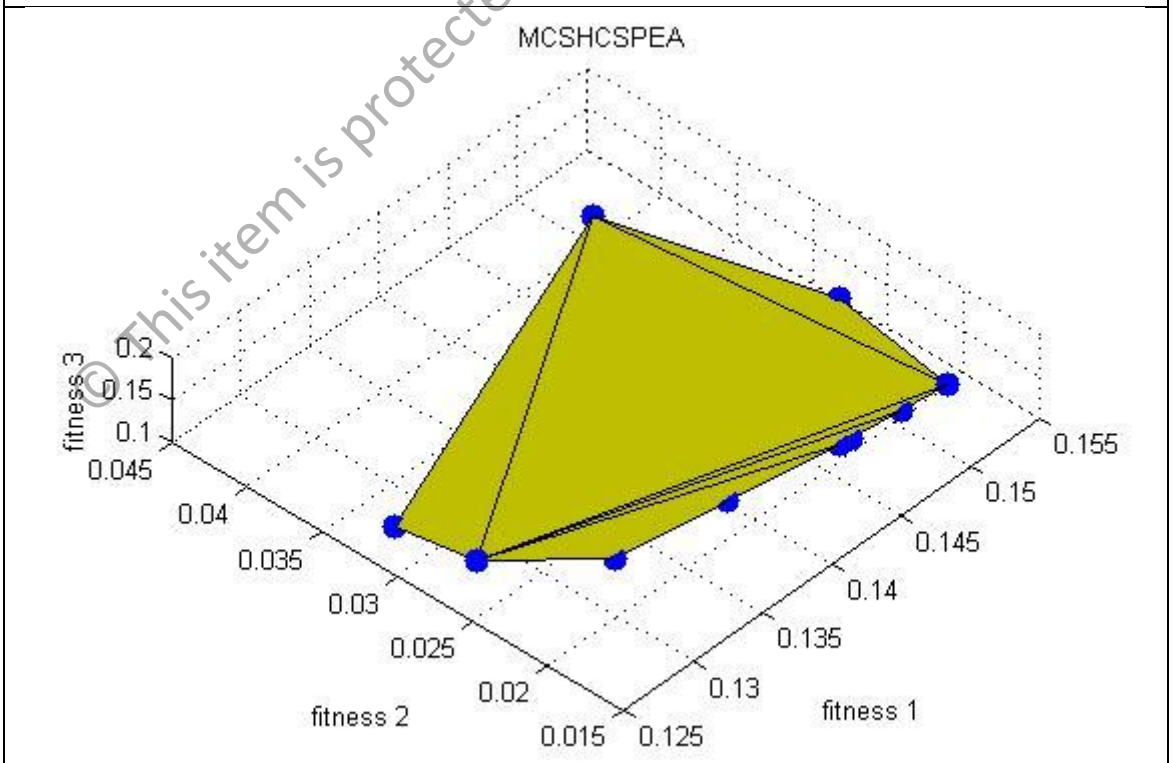
generates the smallest half–power beamwidth (HPBW) of 91.8954^{o} – 88.1072^{o} = 3.7882^{o} with the highest directivity of 8.3578 dB. This is trailed by the MCSHCSPEA counterpart with the calculated HPBW of 92.2838^{o} – 87.7614^{o} = 4.5224^{o}, and the radiation directivity of 7.6527 dB. On the other hand, the MCSSPEA–based array has the smaller directivity of 7.4503 dB followed by the original SPEA–based array with the directivity of 7.3769 dB, respectively. Figure 5.18(c) displays that the MCSPSOSPEA algorithm executes the lowest SLL between 0.0597 dB and 4.1265 dB relatively lower than the conventional array. In addition, Figure 5.18(d) and Figure 5.18(e) show that the proposed MCSPSOSPEA algorithm demonstrates the best nulls mitigation compared to other rivals, with the significant measurements of –52.8246 dB at approximately 31.5127°, and –56.9966 dB at around 148.3961°, respectively.



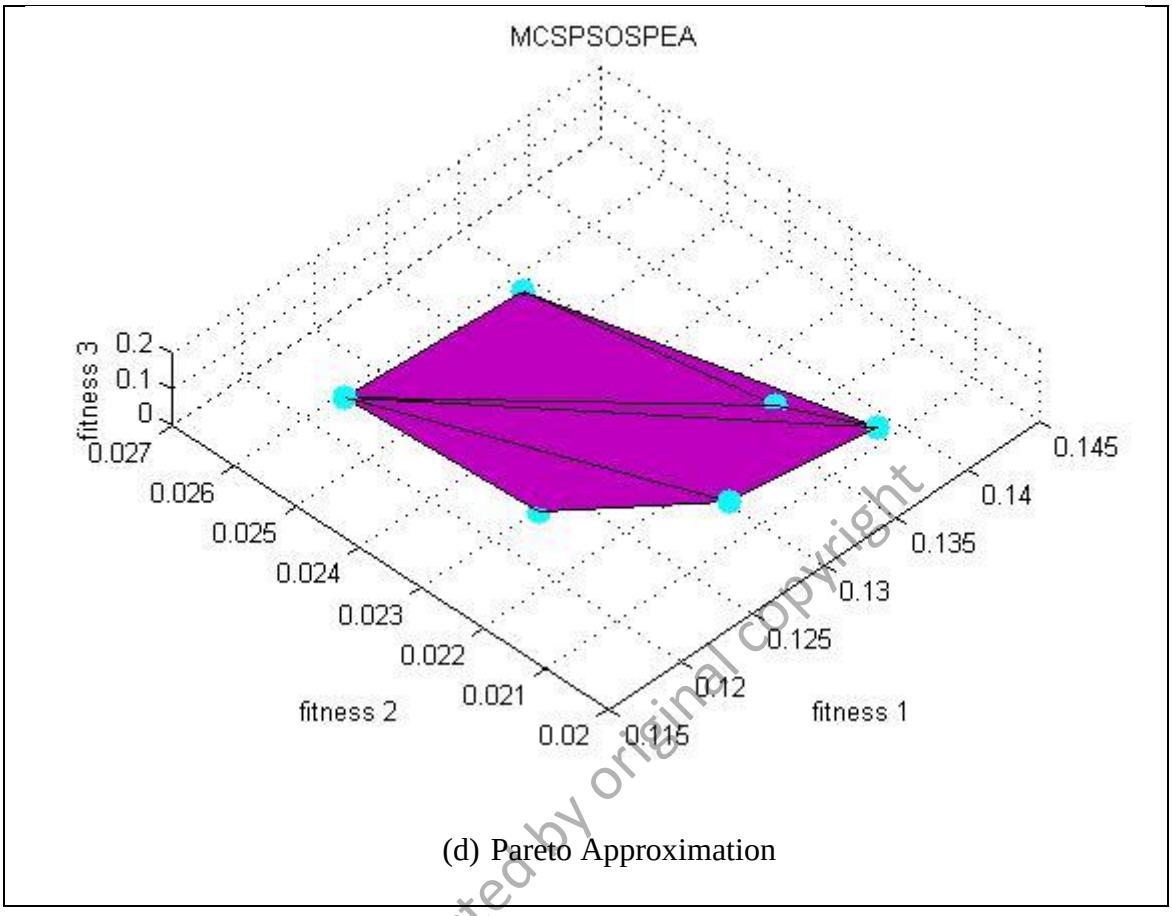
(a) Pareto Approximation



(b) Pareto Approximation

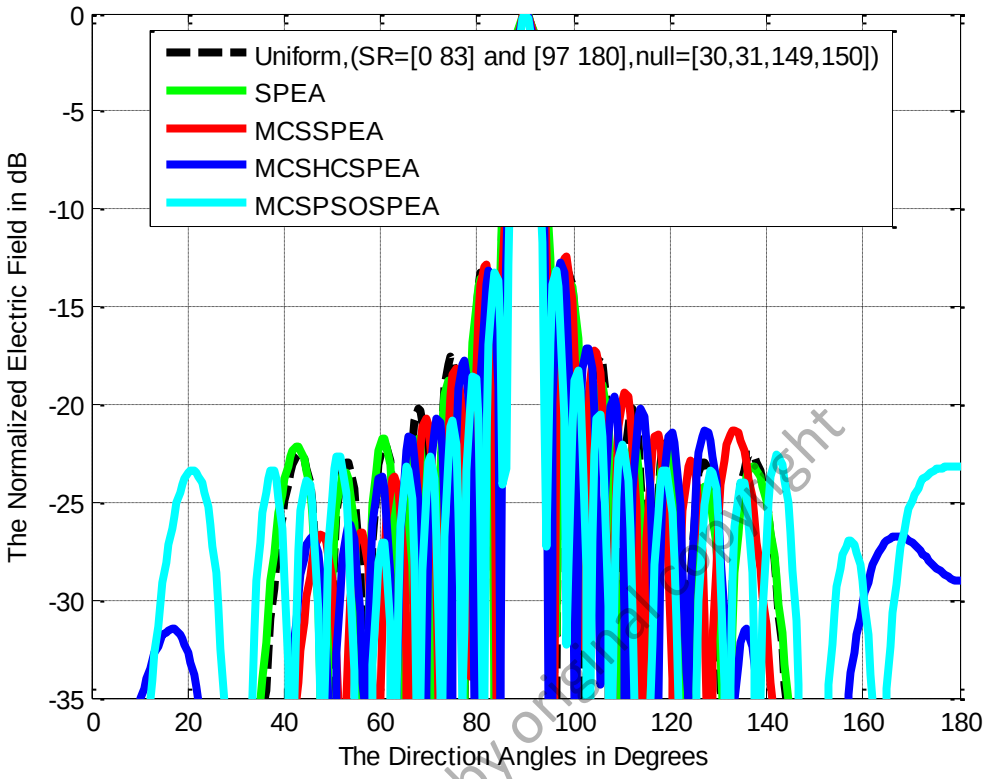


(c) Pareto Approximation

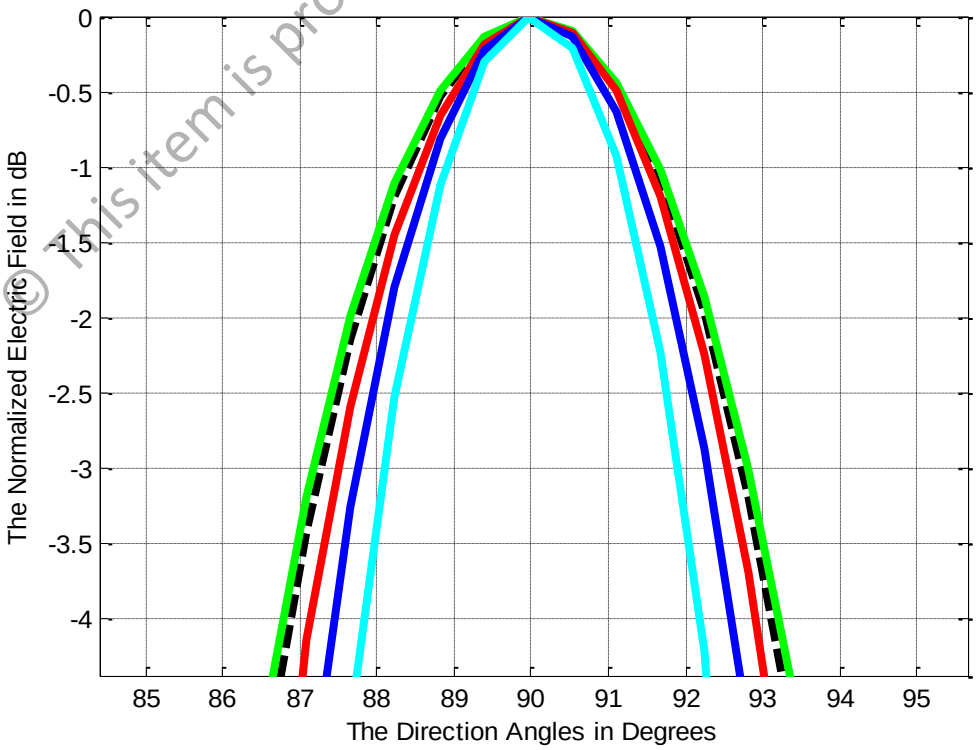


(d) Pareto Approximation

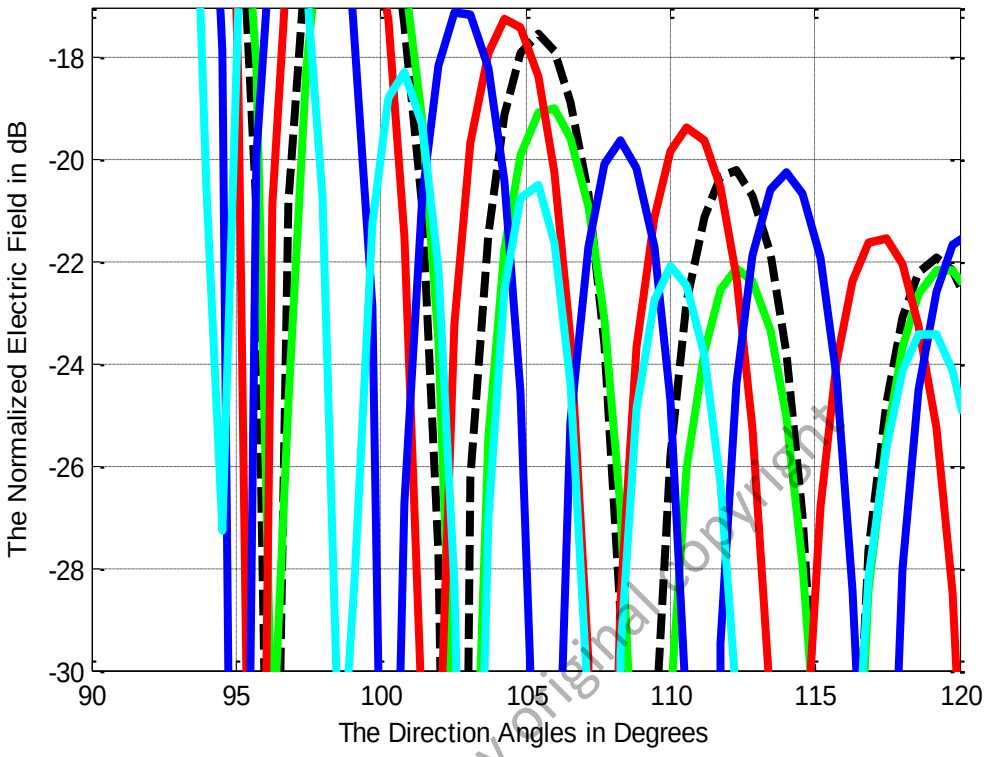
Figure 5.17: Strength Pareto Evolutionary Algorithm (SPEA) Front Approximations ($2N$ = 20, Uniform, Null = [30^o, 31^o, 149^o, 150^o], maxIter = 1000)



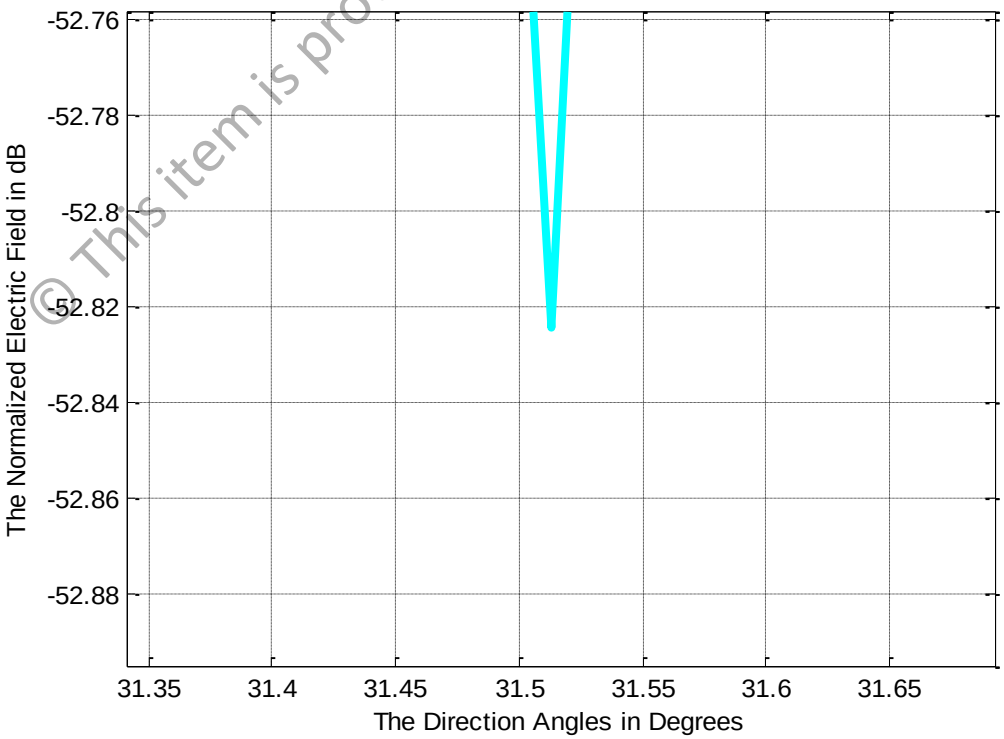
(a) Normalized Radiation Pattern



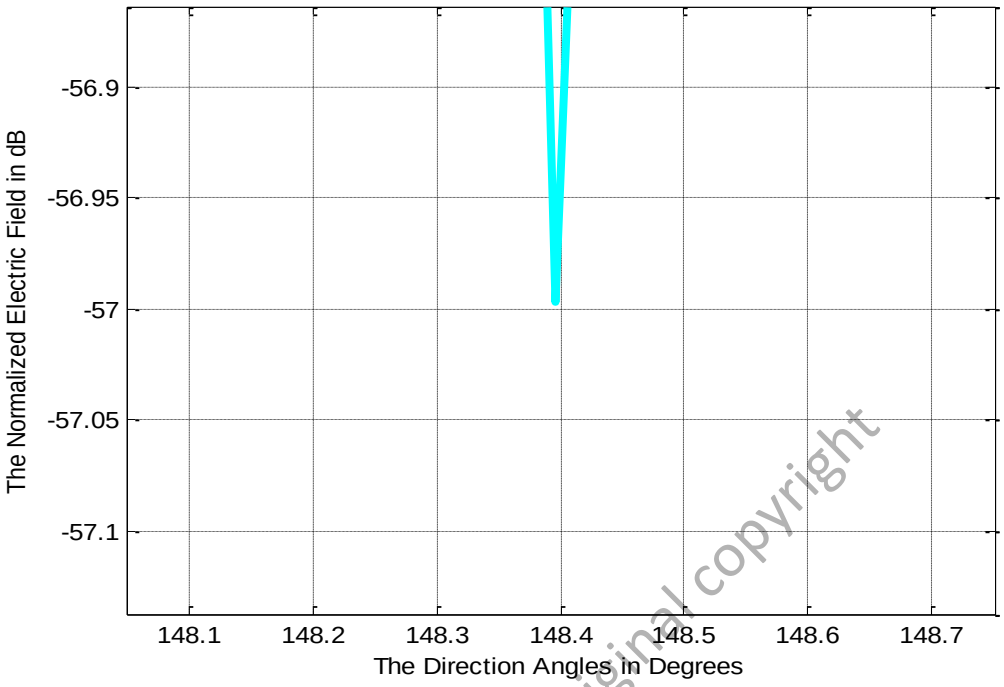
(b) HPBW Pattern



(c) Average SLL Suppression



(d) Null Mitigation near to 31^{o}



(e) Null Mitigation near to 149^{o}

Figure 5.18: Normalized Pattern for SPEA–based Arrays
($2N$ = 20, Uniform, Null = [30^{o}, 31^{o}, 149^{o}, 150^{o}], maxIter = 1000)

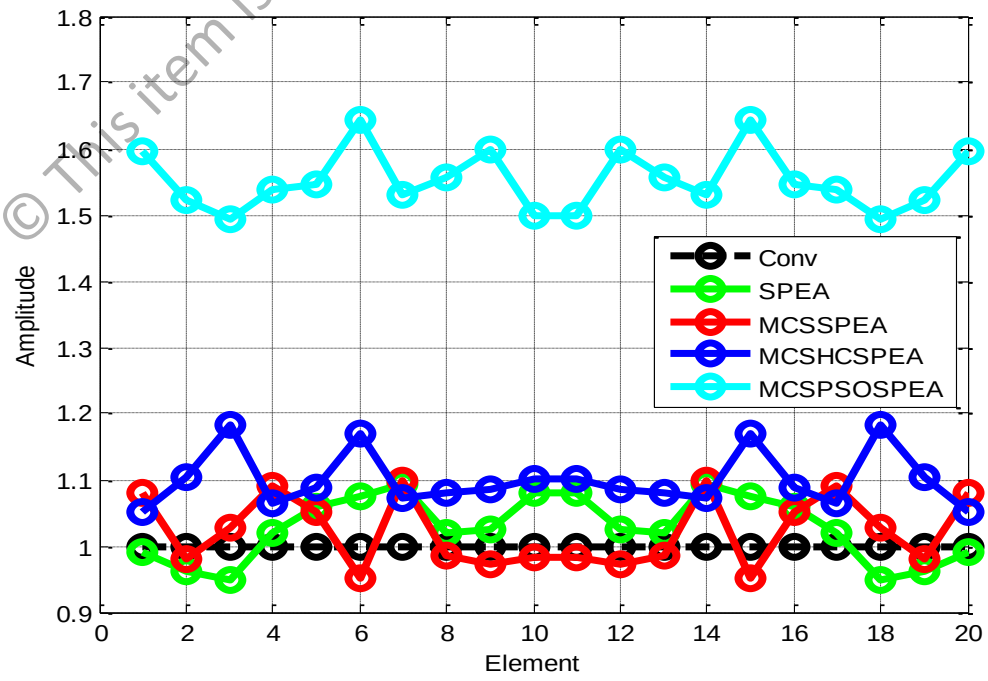


Figure 5.19: Optimal Amplitude for SPEA–based Arrays
($2N$ = 20, Uniform, Null = [30^{o}, 31^{o}, 149^{o}, 150^{o}], maxIter = 1000)

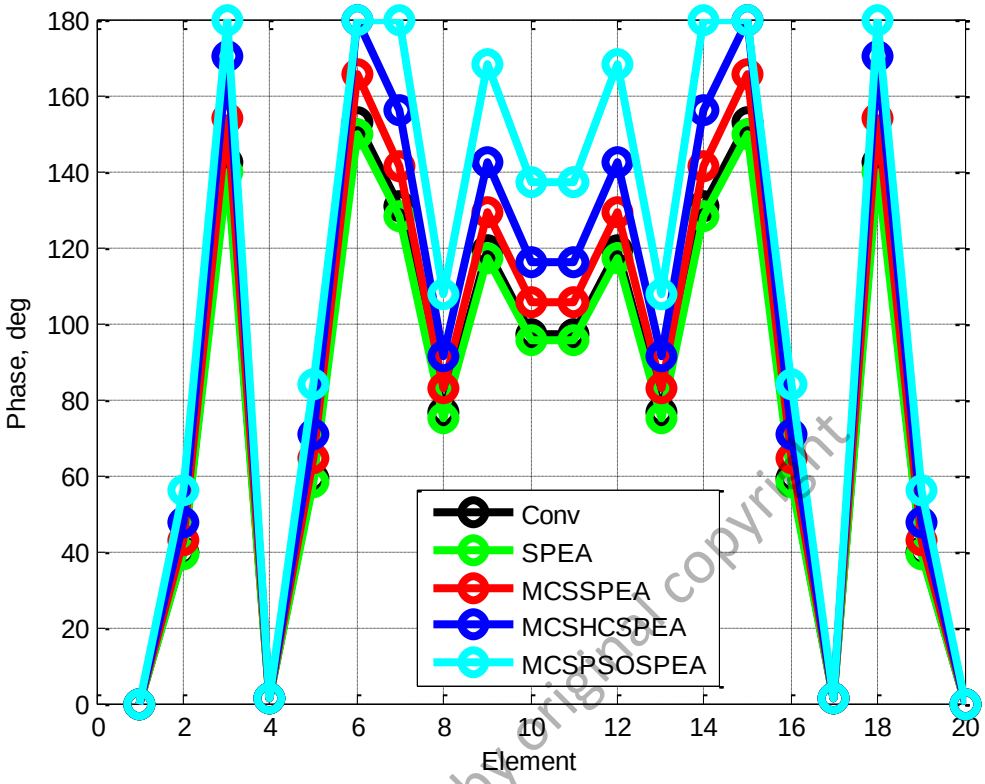


Figure 5.20: Optimal Phase for SPEA–based Arrays
($2N$ = 20, Uniform, Null = [30^o, 31^o, 149^o, 150^o], maxIter = 1000)

The MCSPSOSPEA algorithm produces the biggest amplitude fluctuations. This is followed by the MCSHCSPEA counterpart as shown in Figure 5.19. Similarly, the MCSPSOSPEA algorithm also generates the biggest excitation phase deviations compared to the conventional array, which is also followed by the MCSHCSPEA algorithm as depicted in Figure 5.20. In this simulation, the MCSSPEA algorithm generates 12 non–dominated solutions, the MCSHCSPEA algorithm has 14 non–dominated solutions, and the MCSPSOSPEA algorithm has 10 non–dominated solutions, respectively.

Table 5.15: Selected Optimal Pareto Fitness for SPEA–based Arrays
($2N$ = 20, Uniform, Null = [30°, 31°, 149°, 150°], maxIter = 1000)

Fitness	f_1	f_2	f_3
SPEA	0.1356	0.0241	0.1505
MCSSPEA	0.1342	0.0213	0.1346
MCSHCSPEA	0.1307	0.0258	0.1277
MCSPSOSPEA	0.1196	0.0253	0.1485

Table 5.15 shows the spread Pareto fitness trade–off of the selected non–dominated solutions for all the four tested SPEA–based algorithms for the comparison purposes as presented in

Figure 5.18 – Figure 5.20, respectively. Overall, the MCSPSOSPEA algorithm has the smallest value of f_1 whereas the MCSHCSPEA counterpart has the smallest f_3 and MCSSPEA algorithm has the smallest value of f_2, respectively after 1000 iterations of Pareto MO minimization process. Viewing at the selected non–dominated solutions aspect, the MCSPSOSPEA algorithm has better Pareto front trade–offs especially with respect to f_1 despite having a bigger hypervolume than the standard SPEA counterpart, which leads to the best directivity and smallest half–power beamwidth (HPBW).

Table 5.16: Optimal Location for SPEA–based Arrays
($2N$ = 20, Uniform, Null = [30°, 31°, 149°, 150°], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
X_n **[λ/2]**	±0.5000	±1.5000	±2.5000	±3.5000	±4.5000
SPEA	±0.4910	±1.4731	±2.4551	±3.4372	±4.4192
MCSSPEA	±0.5417	±1.6251	±2.7085	±3.7918	±4.8752
MCSHCSPEA	±0.5978	±1.7935	±2.9891	±4.1847	±5.3804
MCSPSOSPEA	±0.7061	±2.1184	±3.5306	±4.9429	±6.3551
Element	**6**	**7**	**8**	**9**	**10**
X_n **[λ/2]**	±5.5000	±6.5000	±7.5000	±8.5000	±9.5000
SPEA	±5.4013	±6.3834	±7.3654	±8.3475	±9.3295
MCSSPEA	±5.9586	±7.0420	±8.1254	±9.2087	±10.2921
MCSHCSPEA	±6.5760	±7.7716	±8.9673	±10.1629	±11.3585
MCSPSOSPEA	±7.7674	±9.1796	±10.5919	±12.0041	±13.4163

As enlisted in Table 5.16, the postulated MCSPSOSPEA–based array generates the largest optimal location fluctuations compared to the conventional array (with respect to $\lambda/2$) for all $2N = 20$ array elements between $|\pm 0.2061|$ and $|\pm 3.9163|$. The MCSPSOSPEA algorithm once again executes the biggest amplitude deviations compared to the conventional array between 0.4932 and 0.6445 as tabulated in Table 5.17. This is followed by the MCSHCSPEA counterpart with the fluctuations between 0.0501 and 0.1842 for all $2N = 20$ linear array elements.

Table 5.17: Optimal Amplitude for SPEA–based Arrays
($2N = 20$, Uniform, Null = [30^o, 31^o, 149^o, 150^o], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
SPEA	0.9902	0.9615	0.9495	1.0191	1.0582
MCSSPEA	1.0806	0.9804	1.0280	1.0912	1.0509
MCSHCSPEA	1.0501	1.1040	1.1842	1.0649	1.0882
MCSPSOSPEA	1.5957	1.5225	1.4932	1.5394	1.5452
Element	**6**	**7**	**8**	**9**	**10**
A_n	1.0000	1.0000	1.0000	1.0000	1.0000
SPEA	1.0747	1.0923	1.0189	1.0242	1.0803
MCSSPEA	0.9507	1.0999	0.9851	0.9736	0.9824
MCSHCSPEA	1.1694	1.0729	1.0799	1.0849	1.1005
MCSPSOSPEA	1.6445	1.5302	1.5569	1.5988	1.4982

Moreover, Table 5.18 depicts that both the hybrid MCSPSOSPEA and MCSHCSPEA algorithms generate the optimal phase domain of [0^o 180^o]. On the other hand, the MCSSPEA counterpart has the optimal phase domain of [0^o 165.3738^o] and is followed by the standard SPEA algorithm with the phase domain of [0^o 149.9067^o], respectively. In this case, the MCSPSOSPEA algorithm produces the biggest phase deviations compared to the conventional array, which are between 0^o and 49.3734^o. The MCSHCSPEA counterpart has the second biggest phase deviations between 0^o and 27.8236^o. This is followed by the MCSSPEA algorithm with the deviations between 0^o

and 12.7277^{o} and the original SPEA algorithm between 0^{o} and 2.7394^{o} for all 2N = 20 linear array elements.

Table 5.18: Optimal Phase for SPEA–based Arrays
(2N = 20, Uniform, Null = [30^{o}, 31^{o}, 149^{o}, 150^{o}], maxIter = 1000)

Element	**1**	**2**	**3**	**4**	**5**
ϕ_n	0^{o}	39.8173^{o}	142.2216^{o}	1.0740^{o}	59.3696^{o}
SPEA	0^{o}	39.1027^{o}	139.6693^{o}	1.0548^{o}	58.3041^{o}
MCSSPEA	0^{o}	43.1373^{o}	154.0802^{o}	1.1636^{o}	64.3199^{o}
MCSHCSPEA	0^{o}	47.6069^{o}	170.0452^{o}	1.2842^{o}	70.9844^{o}
MCSPSOSPEA	0^{o}	56.2318^{o}	180.0000^{o}	1.5168^{o}	83.8445^{o}
Element	**6**	**7**	**8**	**9**	**10**
ϕ_n	152.6461^{o}	130.6266^{o}	76.3159^{o}	119.2334^{o}	97.2496^{o}
SPEA	149.9067^{o}	128.2824^{o}	74.9464^{o}	117.0937^{o}	95.5043^{o}
MCSSPEA	165.3738^{o}	141.5184^{o}	82.6792^{o}	129.1752^{o}	105.3583^{o}
MCSHCSPEA	180.0000^{o}	156.1817^{o}	91.2460^{o}	142.5596^{o}	116.2750^{o}
MCSPSOSPEA	180.0000^{o}	180.0000^{o}	107.7770^{o}	168.3871^{o}	137.3404^{o}

In sum, the bigger deviations of optimal solutions (position, amplitude, and phase) prove that the postulated MCSPSOSPEA hybrid algorithm is capable to explore furthest in the search space producing the best diversity of the selected non–dominated solutions. Consequently, this stimulates the MCSPSOSPEA algorithm to synthesize linear antenna arrays with the lowest average SLL suppression, smallest main beam HPBW, highest normalized radiation directivity, and best predefined nulls mitigation compared to other SPEA–based competitors.

CHAPTER SIX

RESULTS AND DISCUSSIONS

6.1 Cuckoo Search Algorithm Internal Parameters Analysis

Generally, five internal parameters of CS algorithm are tested in linear array geometry synthesis, which are Lévy flights distribution type (α), α–stable distribution method, length step factor (L), numbers of host nest (population), and discovery rate or fraction probability (P_a). Firstly, two types of Lévy flights distribution, which includes α = 1.0 (Cauchy) and α = 2.0 (Gaussian) are analyzed on $2N$ = 10 and $2N$ = 20 linear antenna arrays, accordingly. It is found that as α smaller, the convergence rate becomes faster. This is due to the fact that the CS–optimizer agreed with the Cauchy distribution property. It means that the CS algorithm can find the optimal solutions and converge faster due to the presence of larger steps or jumps compared to the random Brownian motion existed in optimizer with the Gaussian distribution. In other words, the stochastic CS–optimizer with the Cauchy distribution demonstrates a mix of long trajectories, and short random movements while finding the optimal solutions within N−dimensional search space.

Overall, the performance for both optimizers are identical in $2N$ = 10 linear array since both optimizers have the same minimum fitness, f_{min} convergence and almost similar optimal locations with the differences less than |±0.1000| for all array elements. However, the CS−optimizer with α = 2.0 performs better by having a lower SLL suppression in $2N$ = 20 symmetric array, mainly due to the bigger location fluctuations compared to the conventional array, and a lower f_{min} convergence. In sum, the CS algorithm with the Lévy flight Gaussian distribution performed better for a larger number of array elements. In this case, the CS algorithm with the Lévy flight Gaussian

distribution is capable to explore further in finding the optimal solutions (antenna excitation locations) for the $2N = 20$ linear array.

Secondly, three α–stable distribution methods, which are Mantegna's algorithm, McCulloch's algorithm, and standard random walk are analyzed on both $2N = 10$ and $2N = 20$ linear arrays. It is found that the performances for all the three α–stable distribution methods are alike for $2N = 10$ linear array. This situation is due to the same f_{min} convergence and nearly same optimal solutions (locations) with the differences less than $|\pm 0.1000|$ for all antenna elements. Even so, as the number of element is increased to 20, the CS algorithm with the α–stable Mantegna's algorithm generates the lowest SLL suppression compared to other two competitors. This is because the Mantegna's algorithm is capable to control the Lévy flight motion based on the α–stable distribution more effectively in finding further the optimal solutions in a search space. This is proven by the largest optimal location deviations (as well as magnitudes) compared to the conventional linear array, and a lower f_{min} convergence.

Thirdly, three length step factors (L), which are $L/10$ (factor of 0.1), $L/100$ (factor of 0.01), and $L/1000$ (factor of 0.001) are examined on both $2N = 10$ and $2N = 20$ linear arrays. Precisely, L is the length scale of cuckoo's motion in searching for a new (other host bird's) nest for a brood parasitism purpose. It is found that the performance for all the three length step factors applied in both $2N = 10$ and $2N = 20$ linear arrays are same. These outcomes are primarily due to the same f_{min} convergence and optimal locations for all array elements.

Overall, this indicates that all the three small length step factors, which are smaller than 1.0 ensure that the Lévy flight motions performed by cuckoo are not being too aggressive, e.g. have a uniform steady searching motion. As a result, new identical solutions are found in all the length step factors within the N−dimensional search space.

In another aspect, this also shows that the small length step factors will not be enough to overpower the increasing lengths since the Lévy flight motions with the heavy−tailed distribution has an infinite variance or possible length. In other words, any small length step factor will not distinguish the random motions process, which obeys a power−law step length distribution with a heavy−tail. Thus, the optimal solutions found in N−dimensional search space will be the same for any given step factor.

Fourthly, three numbers of host nest (population), which are 10, 20, and 30are examined on $2N = 10$ linear array. It is found that the performances for all the three numbers of population are alike. This is mainly due to the same f_{min} convergence and almost similar optimal location magnitudes with the differences less than |±0.1000| for all array elements.

However, all the three numbers of host nest (population) applied in the CS algorithm converge at different times. As the number of host nest bigger, the convergence rate becomes faster due to the higher capability and larger probability to find global minimum solutions since bigger number of individuals occupied in the search space.

In other aspect, there should be the ideal number of population applied in the CS−optimizer due to the process complexity and computing execution time (CPU processing time). In this case, bigger number of population means higher process complexity and more execution elapsed time.

Fifthly, three discovery rates or fraction probabilities, P_a, which are 25% or 0.25, 50% or 0.50, and 95% or 0.95 are examined on $2N = 10$ symmetric array. It is found that both the CS–optimizers with $P_a = 0.25$ and $P_a = 0.50$ have the best identical performances whereas the CS–optimizer with $P_a = 0.95$ is the worst one in SLL suppression. This is primarily due to the lower f_{min} convergence and almost similar

optimal locations for all $2N$ = 10 array elements in both CS–optimizers with P_a= 0.25 and P_a = 0.50.

In sum, this finding agrees with the principle that as the P_a larger, the possibility of egg laid by a cuckoo to be discovered by the host bird of other species becomes higher leading to a new nest searching or replacement. As a result, the likelihood of getting the best possible optimum solutions will be diminished or vanished in some extents. In other words, the lower P_a applied in CS algorithm will lead the cuckoo's egg (optimal solution) hatching and brood–parasitism processes well–taken by the host bird unknowingly. Hence, the lower P_a the better SLL suppression due to the superior optimal solutions found in the N–dimensional optimization problem.

6.2 Modified Cuckoo Search Algorithm Analysis

In this part of analysis, there is a development of modified cuckoo search (MCS) algorithm by integrating the standard CS algorithm with the Roulette wheel selection operator, and the adaptive inertia weight, w. The Roulette wheel operator generates a random selection (through a wheel rotation process) where candidate solutions with a superior fitness have a larger possibility to be selected. Moreover, the aim of introducing the adaptive w isprimarily to control the MCS algorithm exploration ability better towards optimal solutions in search space.

Firstly, two types of Lévy flight distribution, which are α = 1.0 (Cauchy) and α = 2.0 (Gaussian) are analyzed on $2N$ = 20 linear array. It is found that the MCS algorithm (α = 2.0) outperforms the MCS counterpart (α = 1.0), and the standard CS algorithms with α = 1.0 and 2.0, in terms of SLL suppression. This is due to the fact that MCS algorithm (α = 2.0) has the highest optimal element location magnitudes and

fluctuations compared to the conventional array. In addition to that, the postulated MCS algorithm (α = 2.0) also achieves the lowest f_{min} convergence.

In sum, the combination of Roulette wheel selection operator, adaptive *w*, and Lévy flight Gaussian distributions in MCS algorithm evidently is capable to search further best−fitted solutions (linear array excitation locations) in *N*–dimensional space. As a result, the proposed MCS algorithm delivers the bigger diversity of optimal solutions, which generate the normalized antenna radiation pattern with lower side lobes whilst preserving the main beam intensity.

Secondly, three α–stable distribution methods, which are Mantegna's algorithm, McCulloch's algorithm, and standard random walk are analyzed on $2N$ = 20 linear array. It is found that the MCS–optimizer with Mantegna's algorithm demonstrates the best SLL suppression. This is primarily due to the biggest location deviations compared to the conventional arrayand the lowest f_{min} convergence.

In sum, the introduction of Roulette wheel selection operator, adaptive *w*, and Mantegna's algorithm enhance the capability of MCS algorithm in generating random numbers (candidate solutions) based on a symmetric α–stable distribution. As a result, the MCS–optimizer with Mantegna's algorithm is capable to demonstrate a far–reaching metaheuristic search, which outperforms other competitors in SLL suppression whilst maintaining the main lobe.

Thirdly, three numbers of population (host nest), which are 10, 20, and 30 are examined on $2N$ = 20 linear array. It is found that the hypothesized MCS–optimizers with the largest number of population (nest = 30) outperforms other counterparts in suppressing SLL. This is due to the highest location deviations compared to the conventional array and the lowest f_{min} convergence.

In sum, the introduction of Roulette wheel selection operator and adaptive inertia weight, *w* along with a larger population size can provide a higher possibility in finding the optimal solutions in the search space. Hence, a better diversity of optimal solutions is directly generated from a larger number of individuals (candidate solutions).

Nevertheless, there should be the appropriate number of population chosen for the MCS−optimizer due to the process complexity and execution time (CPU processing time) limitations. This is important since the bigger number of population will cause the higher complexity and the more processing time for execution.

Fourthly, three discovery rates or fraction probabilities, which are P_a = 25% or 0.25, 50% or 0.50, and 95% or 0.95 are examined on $2N$ =20 linear array. It is clearly found that the MCS–optimizer with P_a = 0.25 has the best SLL suppression compared to other two MCS counterparts. This is primarily due to the lowest f_{min} convergence, and biggest optimal location fluctuations compared to the conventional array for all $2N$ = 20 array elements.

To conclude, the simulation result confirms the hypothesis that as the P_a smaller, the possibility of egg laid by a cuckoo to be discovered by the host bird of other species becomes lower. Hence, this leads to a bigger odd for the survival of its brood or offspring, possibly then becomes as a candidate solution. In this experiment, the application of Roulette wheel selection operator and adaptive, *w* along with the lowest P_a significantly improves the proposed MCS−optimizer performance in SLL suppression whilst maintaining the main beam due to the higher probability of finding the best–fitted optimal solutions (element excitation locations) in search space.

Fifthly, a rigid broadside case experiment is performed on the proposed MCS–optimizer for $2N$ = 20 linear array. In this case, the linear array is steered to 90° and has two predefined nulls at 45° and 135°, respectively. All the tested optimizers

with three α–stable distribution methods, host nest (population) = 30, P_a = 25% or 0.25, α = 2.0 (Lévy flight Gaussian distribution), and length step factor = L/100 or 0.01 are compared along with the conventional array in terms of SLL suppression, and prescribed nulls mitigation. In this study, the MCS algorithm deploys the Roulette wheel selection operator and dynamic inertia weight, w with the domain of [0.95 1.00].

The MATLAB simulation shows that the hypothesized MCS–based array with the Mantegna's algorithm significantly outperforms other rivals in SLL suppression and null mitigation at both direction angles of 45° and 135°, respectively. This is due to the proposed MCS–optimizer (Mantegna's algorithm) generates relatively the lowest f_{min} convergence and biggest optimal location fluctuations than the other opponents.

To recap, the introduction of Roulette wheel selection operator and adaptive w significantly improves the MCS–optimizer (with Mantegna's algorithm as the α–stable distribution method applied) in generating a group of potential best–fitted host nests or candidate solutions, and controlling cuckoo's Lévy flight motion towards the best host nest or best optimal solution in search space. As a result, the proposed MCS–optimizer (Mantegna's algorithm) is able to demonstrate the best SLL suppression and/or nulls mitigation whilst maintaining the main beam, simultaneously.

Sixthly, there is also an investigation done on the proposed MCS algorithm for $2N$ = 20 linear array within a window or tapering function, known as the Dolph–Chebyshev window. Similarly to the previous experiment, all the evaluated MCS and CS–optimizers with three α–stable distribution methods, host nest (population) = 20, discovery rate, P_a = 25% or 0.25, α = 2.0 (Lévy flight Gaussian distribution), and the length step factor = L/100 or 0.01 are compared along with conventional array in terms of SLL suppression. In this experiment, all the tested

MCS–optimizers deploy the Roulette wheel selection operator and adaptive weight, w with the magnitude domain of [0.95 1.00].

The output reveals that the postulated MCS–based array with the Mantegna's algorithm outperforms other rivals in equiripple side lobe level (SLL) suppression followed by both MCS–based array with the McCulloch's algorithm and the standard random walk, respectively. The MCS algorithm with the Mantegna's α–stable distribution method demonstrates the best performance is directly driven bythe largest optimal location magnitudes and the lowest f_{min} convergence. In this case, all the three MCS–optimizers produce a bigger optimal location fluctuations compared with the conventional array, and a lower f_{min} convergence than the original CS competitors.

To recapitulate, the integration of the Roulette wheel selection operator, adaptive w, and Mantegna's algorithm improves the performance of the MCS–based linear array in SLL suppression within the Doppler–Chebyshev signal processing window. In other words, the proposed MCS algorithm (Mantegna's α–stable distribution method) is capable to perform a far–reaching metaheuristic exploration for a better diversity of optimal solutions in N–dimensional search space regardless signal processing window or tapering function used.

Seventhly, there is an experiment on $2N$ = 30 linear array to compare the proposed MCS algorithm with the original CS algorithm and two other evolutionary computation (EC) methods, which are particle swarm optimization (PSO) and genetic algorithms (GA). Likewise to the earlier study, the Dolph–Chebyshev becomes the selected tapering function. Both MCS and CS–optimizers are simulated for 1000 iterations using host nest (population) = 30, discovery rate, P_a = 25% or 0.25, length step factor = L/100 or 0.01, and α = 2.0 (Lévy flight Gaussian distribution). Once again,

the MCS–optimizer uses the Roulette wheel selection operator, and adaptive *w* domain of [0.95 1.00].

All the corresponding internal parameters of PSO and GA are set compatible with both the MCS and CS metaheuristic algorithms to avoid any predisposition or bias. The PSO algorithm is simulated using particle (population) = 30, particle min/max velocity domain = [–0.1 +0.1], individuality accelerator = 1.0, and sociality accelerator = 1.0, respectively. The GA uses the Roulette wheel selection operators with chromosome or gene (population) = 30, gene crossover probability, P_c = 90% or 0.9, and gene mutation probability, P_m = 10% or 0.1.

The iterative simulation shows that the MCS algorithm marginally outperforms other opponents in suppressing narrow–width side lobes of $2N = 30$ linear array. In this case, the largest optimal location magnitudes, and the lowest f_{min} convergence are the key factors for the postulated MCS algorithm to perform the best equiripple SLL suppression whilst preserving the main lobe intensity at the direction angle of 90°.

In sum, the introduction of Roulette wheel selection operator and adaptive *w* mechanisms in the postulated MCS algorithm successfully produces a better diversity of *N*–dimensional solutions via an effective controllable cuckoo's Lévy flight motions in search space. Here, the *N*–dimensional solutions refer to the $2N = 30$ element or radiator symmetric positions along the *xy*–plane. As a result, the proposed MCS algorithm is able to suppress side lobes better than other well–known EC techniques for a large number of array elements within the Dolph–Chebyshev feed current tapering window.

6.3 Hybrid Modified Cuckoo Search Algorithm Analysis

In this part of analysis, the MCS algorithm undergoes a hybridization process with PSO algorithm known as MCSPSO, and GA merely referred as MCSGA. The

newly developed MCSPSO and MCSGA hybrid–optimizers manipulate the fitness, *f* function in guiding cuckoo's α–stable Lévy flight motions towards a potential host nest (optimal element locations) in *N*–dimensional search space.

In the first simulation, the postulated MCSPSO and MCSGA–hybrid optimizers with Mantegna's algorithm as the selected α–stable distribution method, host nest (population) = 30, discovery rate, P_a = 25% or 0.25, length step factor = $L/100$ or 0.01, and α = 2.0 (Lévy flight Gaussian distribution) are executed on $2N$ =20 linear arrays. Other compatible EC algorithms are simulated for a comparison purpose include hybrid GAPSO, MCS, and standard CS algorithms, respectively. For a uniformity, the proposed MCSPSO, MCSGA, and MCS algorithms deploy the Roulette wheel selection operator and adaptive inertia *w* domain of [0.95 1.00].

The MATLAB simulation shows that the MCSPSO hybrid–optimizer has the best SLL suppression. This is primarily due to the lowest f_{min} convergence. Moreover, the despite worse than the MCSPSO hybrid algorithm, the MCSGA counterpart has a lower SLL suppression than the GAPSO, and original CS rivals.

To sum up, the hybridization of the proposed MCS algorithm and standard PSO stochastic algorithm enhance the capability of searching further the optimal solutions (array element locations) in search space. This is initially done through the usage of MCS value–added parameters, which are the Roulette wheel selection operator, and adaptive *w* to control effectively exploration of the best–fitted chromosomes (host nests) in the search space. Furthermore, the hybridization process manipulates the PSO algorithm in controlling the velocity of cuckoo Lévy flight motions towards optimal solutions. Besides, the position updating process in PSO stochastic algorithm is used to calculate the personal best fitness or *pbest* for all particles (host nests), and from that to locate the particle (host nest) with the global best fitness or *gbest* found in search space.

Therefore, the hybrid MCSPSO algorithm is proven can generate a better diversity of optimal solutions (array element positions), which demonstrate the desired radiation pattern with a lower SLL suppression whilst maintaining a main beam intensity.

Another experiment involves a more robust circumstance of $2N = 10$ symmetric array where all the tested metaheuristic algorithms have the main lobe steered to 60°**,** with four prescribed nulls at 30°, 31°, 79°, and 80°, respectively. All the fundamental parameters used for all the algorithms are same as in the previous experiment. It is found that after executing 100 iterations, the MCSPSO–based array generates the best SLL suppression especially within the [0° 30°] and [80° 130°] regions, respectively. Besides to that, the MCSPSO hybrid algorithm also demonstrates the best nulls mitigation approximately at the predefined direction angles. These findings are driven by two key factors, which are the biggest optimal solution fluctuations compared to the conventional array, and the lowest f_{min} convergence.

In sum, the hybridization of MCS and PSO metaheuristic algorithms can produce a significant diversity of optimal solutions (array element locations). This is done through the integration of MCS algorithm value–added features, which are Roulette wheel selection operator, and adaptive inertia, *w* with the PSO stochastic algorithm routines, which are velocity and position of particle (host nest) iterative updating mechanisms. As a result, the MCSPSO–hybrid optimizer is able to control the cuckoo's Lévy flight direction and speed, and through it can locate the global best (*gbest*) solutions, which generates a radiation pattern with the lowest side lobes suppression and/or the best prescribed nulls mitigation while preserving and steering the main lobe to the desired direction angle.

6.4 Multiobjective Optimization Approach of Modified Cuckoo Search Algorithm Analysis

In this study, the proposed MCS algorithm optimizes three objective functions simultaneously to identify three decision variables, which are optimal excitation locations, amplitudes, and phases. Two approaches are done in the MO optimization, which are weighted–sum, and global Pareto front.

6.4.1 Weighted–Sum Approach

Firstly, the postulated MCSPSO, MCSGA, and MCS metaheuristic algorithmswith Mantegna's α–stable distribution method, host nest (population) = 30, discovery rate, P_a = 25% or 0.25, length step factor = L/100 or 0.01, and α = 2.0 (Lévy flight Gaussian distribution) are examined on the $2N$ = 10 linear array. Both optimizers are directly compared with hybrid GAPSO, MCS, and CS algorithms through a weighted–sum method. All the MCS–based algorithms has a dynamic P_a magnitude domain of [0.01 0.25] and an adaptive w domain of [0.95 1.05]. Both domains are set to gain a more flexibility in exploring and controlling cuckoo's Lévy flight motions towards the best host nest (potential solution) in search space. Moreover, Both the MCSPSO and GAPSO optimizers deploy the PSO algorithm with the dynamic random particle velocity domain of [–0.1 +0.1]. Furthermore, the MCSGA and GAPSO algorithms use the GA optimizer with the gene crossover probability, P_c = 90% or 0.9, and gene mutation probability, P_m = 10% or 0.1.

The MATLAB simulation shows that the proposed MCSPSO hybrid algorithm outperforms all other stochastic algorithms in SLL suppression whereas the MCSGA counterpart has the smallest half–power beamwidth (HPBW) of the main beam. Four

main factors lead the outstanding performance in SLL suppression and high directivity of MCSPSO hybrid algorithm via the weighted–sum technique. The first factor: the MCSPSO hybrid algorithm generates the lowest weighted–aggregation f_{min} convergence. The second factor: the MCSPSO hybrid algorithm generates the largest position variations compared to the conventional array with respect to $\lambda/2$. The third factor: the MCSPSO hybrid algorithm has the lowest optimal amplitude, hence the biggest amplitude variations compared to the conventional array. The fourth factor: the MCSPSO hybrid algorithm produces the biggest optimal phase deviations compared to the conventional array. Notes that, the f_{min} is normalized in all iterations by dividing the fitness functions, f_1, f_2, and f_3 with their corresponding mean calculated in the first iteration. The purpose is to reduce the possible bias due to magnitude differences among the three fitness functions.

Secondly, a more stringent experiment is conducted using the weighted–sum method to simulate $2N$ = 20 linear array with main beam radiates to the desired direction angle of 90^o and prescribed interferers at the direction angles of 35^o and 145^o, respectively. All the CS and MCS optimizers have the same internal parameters. Besides, all the MCS–based algorithms have a dynamic P_a magnitude domain of [0.01 0.25] and a dynamic w magnitude domain of [0.95 1.05], respectively. Both the MCSPSO and GAPSO optimizers deploy the PSO algorithm with the dynamic random particle velocity domain of [–0.1 +0.1]. Moreover, the MCSGA and GAPSO algorithms use the GA optimizer with the gene crossover probability, P_c = 90% or 0.9, and gene mutation probability, P_m = 10% or 0.1.

It is found that the postulated MCSPSO based–optimizer has the best average side lobes suppression, highest radiation intensity with the smallest half–power beamwidth (HPBW), significant directivity, and best prescribed null mitigation at about

145°. This is followed by the MCSGA counterpart in terms of SLL suppression, small HPBW, high directivity, and the best predefined null mitigation nearly at 35°.

Once again, the outstanding performance of MCSPSO hybrid algorithm is primarily due to the four factors similarly to the previous experiment. This includes the lowest weighted–aggregation f_{min} convergence, and the largest variations compared to the conventional array in terms of optimal location, amplitude, and phase for all the $2N = 20$ array elements.

Overall, the proposed MCSPSO stochastic algorithm is able to further search the best host nest (optimal solution) in search space. Hence, this produces a better diversity of optimal solution (array element location, amplitude, and phase). This is driven by the use of value–added attributes, e.g. Roulette wheel selection operator, adaptive w, dynamic P_a, and both the velocity and position of particle iterative effective updating mechanisms in PSO optimizer. As a result, the MCSPSO hybrid algorithm can control more effectively the Lévy flight searching motion (via velocity and position updating processes), and through it can locate the global best host nest (optimal solution) within N–dimensional search space.

6.4.2 Global Pareto Front Approach

The final analysis focuses on the MO optimization using various MCS algorithms through the global Pareto front method. Altogether, four Pareto algorithms (including the three proposed algorithms) will be compared in performing a minimization trade–off of three objective functionsto find a set of non–dominated solutions, which are antenna array excitation locations, amplitudes, and phases. The three proposed hybrid algorithms tested are the modified cuckoo search–strength Pareto evolutionary algorithm (MCSSPEA), modified cuckoo search–hill climbing–strength

Pareto evolutionary algorithm (MCSHCSPEA) and modified cuckoo search–particle swarm optimization–strength Pareto evolutionary algorithm (MCSPSOSPEA). The other competitors simulated are the standard strength Pareto evolutionary algorithm (SPEA) and the conventional array.

In the first Pareto experiment, the MCSSPEA, MCSHCSPEA and MCSSPEA algorithms deploy Mantegna's algorithm as the selected α–stable distribution method, host nest (population) = 20, fraction probability, P_a = 25% or 0.25, step length factor = $L/100$ or 0.01, and α = 2.0 (Lévy flight Gaussian distribution) for $2N$ = 20 linear array. In this study, the proposed MCSSPEA, MCSHCSPEA and MCSPSOSPEA optimizers use the dynamic inertia weight, w with the magnitude domain of [0.80 1.20]. The bigger w magnitude domain leads the MCS algorithms to gain a more control on the Lévy flight motions with a heavy–tailed and α–stable distribution towards the best host nest (candidate solution) in search space. Furthermore, the proposed MCSPSOSPEA algorithm uses the particle swarm optimization (PSO) optimizer with the dynamic random particle velocity domain of [–0.1 +0.1]. The comparison is conducted within the uniform distribution window.

Overall, all the optimizers have very small hypervolumes, which close to zero (due to the Pareto MO minimization) and are almost identical with the differences among them less than 3.0×10^{-5} unit^3. Based on the selected trade–off solutions, the MCSPSOSPEA algorithm has better Pareto front trade–offs especially with respect to f_1 and f_3 despite having a bigger hypervolume than the standard SPEA counterpart, which leads to the best directivity and smallest HPBW. The selected trade–off solutions are chosen from the nearest point of all compared Pareto–based algorithms since hypervolume differences among them are small.

In this case, the proposed MCSPSOSPEA algorithm generates the highest antenna directivity, smallest HPBW of the main beam, and lowest average SLL suppression. In other words, the MCSPSOSPEA algorithm executes the highest radiation intensity at the main beam and lowest electromagnetic field at side lobes. The key factors for these findings are the proposed MCSPSOSPEA hybrid algorithm has the biggest optimal location (with respect to $\lambda/2$), amplitude, and phase deviations compared to the conventional array, respectively. In other words, the MCSPSOSPEA algorithm is capable to do metaheuristic search further and provide a better diversity of global Pareto front trade–offs.

The second Pareto experiment focuses on the performance comparison within the Dolph–Chebyshev window with the relative SLL, R = –30 dB for $2N$ = 20 linear array. All the internal parameters for all the tested SPEA–based optimizers are kept the same as in the previous experiment.

It is found that all the tested algorithms have the Pareto fitness hypervolumes, which are very small (close to zero) and almost similar with the differences among them less than 8.0×10^{-4} unit^3. Once again, using the selected trade–off solutions, the proposed MCSPSOSPEA algorithm has the smallest values of f_1 and f_3. Besides, the simulation also demonstrates that the proposed MCSPSOSPEA–hybrid optimizer has the best average SLL suppression, smallest HPBW of main lobe, and highest directivity followed by the MCSHCSPEA counterpart. Similarly to the previous experiment, the findings are primarily driven by the biggest fluctuations of optimal excitation position, Dolph–Chebyshev amplitude, and phase compared to the conventional array, respectively. This shows that the proposed MCSPSOSPEA hybrid algorithm is capable to do stochastic search further, and deliver a better diversity of Pareto front trade–offs in a complex Dolph–Chebyshev window too.

Then, the third Pareto simulation is conducted on an intricate broadside analysis for $2N = 20$ linear antenna array with a main beam at 90^o and four predefined interferers at 30^o, 31^o, 149^o and 150^o, respectively. For uniformity, all the tested SPEA–based optimizers deploy the same internal parameters as in the previous simulations. In this simulation, all the SPEA–based algorithms and conventional array are compared in terms of average SLL suppression, HPBW reduction, directivity improvement, and prescribed nulls mitigation whilst improving the main lobe intensity.

The simulation shows that all the tested algorithms generate the Pareto fitness hypervolumes, which are very near to zero and alike with the differences among them less than 2.2×10^{-5} unit^3. In this case, the proposed MCSPSOSPEA–hybrid optimizer finds the selected Pareto non–dominated solutions with the smallest values of f_1. In this case, the MCSPSOSPEA algorithm demonstrates the best SLL suppression whilst maintaining and improving the main lobe electromagnetic field strength with the smallest HPBW and highest antenna directivity. In addition, the MCSPSOSPEA also has the best mitigation near to the four prescribed interferers.

The best linear antenna array performance demonstrated by the MCSPSOSPEA algorithm is directly driven by the biggest optimal position, amplitude, and phase variations compared to the conventional array for all $2N = 20$ array elements.

In summary, the combination of the proposed MCS algorithm with PSO and SPEA algorithms is proven successfully produce significant non–dominated solutions through the Pareto MO trade–offs mechanism. In this case, the MCS maximum utilizes the advantage of SPEA metaheuristic method, which has a lot of potentials in deploying 3–dimensional optimization and Pareto dominance calculation based on relative fitness strengths. Furthermore, the deployment of six significant attributes:

i. The Roulette wheel selection operator.

ii. Dynamic discovery rate or fraction probability.

iii. Dynamic inertia weight.

iv. Fitness strength comparisons in SPEA (to locate and preserve Pareto non–dominated solutions).

v. The distances expansion formula (to reduce the local trap of global Pareto fronts).

vi. Both velocity and position of particle updating mechanism in PSO (to navigate cuckoo Lévy flight motions) backing the MCSPSOSPEA hybrid algorithm to be as an alternative and efficient evolutionary computation (EC) technique in the average SLL suppression and/or prescribed nulls mitigation whilst increasing the main lobe intensity (small HPBW) and antenna directivity (in dB) regardless signal processing window applied.

6.5 Result Comparison

In this section, all the proposed modified and hybrid CS algorithm performances are compared relatively with other existing EA/EC stochastic techniques in terms of maximum SLL suppression, nulls mitigation and HPBW values. Other existing comparable EA/EC techniques are taken from previous literatures. The following is the comparison details.

Table 6.1: EA/EC Stochastic Method Performance Comparison

Uniform Amplitude ($2N$ = 10)			
EA/EC Technique	**SLL Suppression**	**Null Mitigation**	**HPBW**
MCS	–23.84 dB ≤ SLL ≤ –14.50 dB	–82.42 dB	10.9^{o}
MCSPSO	–29.67 dB ≤ SLL ≤ –09.59 dB	–85.76 dB	8.3^{o}
MCSPSOSPEA	–22.85 dB ≤ SLL ≤ –12.09 dB	–36.64 dB	7.3^{o}
BBCA (Sharma, 2014)	–18 dB ≤ SLL ≤ –14 dB	n/a	13^{o}
PSO (Khodier, 2009)	–24 dB ≤ SLL ≤ –18 dB	n/a	12^{o}
CLPSO (Goudos, 2010)	–19.07dB ≤ SLL ≤ –19.07 dB	n/a	11.5^{o}
Dolph–Chebyshev R = –30 dB Amplitude ($2N$ = 10)			
EA/EC Technique	**SLL Suppression**	**Null Mitigation**	**HPBW**
MCS	–32.30 dB ≤ SLL ≤ –21.61 dB	–71.22 dB	11.63^{o}
MCSPSO	–34.81 dB ≤ SLL ≤ –19.89 dB	–51.88 dB	11.64^{o}
MCSPSOSPEA	–27.46 dB ≤ SLL ≤ –24.12 dB	–41.33 dB	9^{o}
PS(Günes, 2010)	–31 dB ≤ SLL ≤ –23 dB	n/a	12.2^{o}
Uniform Amplitude ($2N$ = 20)			
EA/EC Technique	**SLL Suppression**	**Null Mitigation**	**HPBW**
MCS	–26 dB ≤ SLL ≤ –13.67 dB	–48 dB	5.27^{o}
MCSPSO	–26 dB ≤ SLL ≤ –13.15 dB	–70.65 dB	4^{o}
MCSPSOSPEA	–25.94 dB ≤ SLL ≤ –13.86 dB	–57 dB	3.79^{o}
GA (Laseetha, 2011)	–24 dB ≤ SLL ≤ –15 dB	n/a	7.20^{o}
RGA (Goswami, 2012)	–30 dB ≤ SLL ≤ –15 dB	–86.36 dB	5.81^{o}
PSO (Zaman, 2012)	–32.50 dB ≤ SLL ≤ –22.50 dB	n/a	6.70^{o}
FA (Zaman, 2012)	–28 dB ≤ SLL ≤ –23 dB	n/a	6.70^{o}
FA (Kaur, 2013)	–22.50 dB ≤ SLL ≤ –13 dB	–50 dB	7^{o}
TM (Kaur, 2013)	–23 dB ≤ SLL ≤ –12.50 dB	–48 dB	7^{o}
SADE (Kaur, 2013)	–16 dB ≤ SLL ≤ –12.50 dB	–45 dB	7.50^{o}
FA (Basu, 2011)	–20.50 dB ≤ SLL ≤ –15.57 dB	n/a	5^{o}
ABC (Basu, 2011)	–27.50 dB ≤ SLL ≤ –15.56 dB	n/a	5^{o}

Based on the comparison table, the proposed MCSPSO algorithm generate the reasonable average SLL suppression and the best in null mitigation for both $2N$ = 10 and $2N$ = 20 linear array elements under the uniform current amplitude distribution. The postulated MCSPSOSPEA technique had the competitive average SLL suppression, significant null mitigation, and best HPBW (best main beam intensity) for both $2N$ = 10 and $2N$ = 20 linear array elements under the uniform and Dolph–Chebyshev windows. The PSO and RGA techniques are the two best existing EA/EC competitors for both MCSPSO and MCSPSOSPEA metaheuristics in terms of average SLL suppression and null mitigation despite having larger HPBW.

CHAPTER SEVEN

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

On the whole, the contributions of this study can be divided into two general aspects. Firstly, looking at the theoretical aspect, the introduction of MCS algorithm, hybridization of MCSPSO algorithm (in both single objective and weighted–sum MO optimizations techniques), and integration of MCSPSOSPEA, MCSHCSPEA and MCSSPEA algorithms in the Pareto front MO optimization are proven scientifically can be as alternative EA/EC techniques for linear antenna array synthesis.

In this case, all the postulated MCS, MCSPSO, MCSSPEA, MCSPSOSPEA and MCSHCSPEA algorithms with the fine–tuned internal parameters are capable to perform side lobes suppression and/or predefined nulls mitigation while preserving/enhancing the main lobe either for a broadside/non–broadside case. The proposed MCS, MCSPSO, MCSPSOSPEA, and MCSHCSPEA are verified perform better than their compatible opponents in their respective analysis as discussed in Chapter 8. Since the CS metaheuristic algorithm is still at the infancy stage in electromagnetic optimization, there are large opportunities for the algorithm to be manipulated through various enhancement and hybridization mechanisms particularly, for array geometry synthesis. This research is a pioneering study, which only exposes a small portion of the modified CS algorithm highly potential breakthrough.

Looking at the practical aspect, the findings of the various proposed MCS–based hybrid algorithms in this study can be put together as a new foundation for

researchers/engineers to design smart antenna specifically, adaptive antenna array. In this case, the design can be developed in both software and hardware parts.

7.2 Limitations

Generally, there are seven limitations appeared in this pioneering study. Despite of becoming more robust compared to other well−known EA/EC techniques, the original CS algorithm itself has three disadvantages, which has been improved in this study. Firstly, the original CS algorithm uses fixed value for both P_a and α parameters. Both internal parameters are critical and sensitive in fine−tuning a local and global explorative random motion towards optimal solutions, and adjusting convergence rate (Valian, Mohanna & Tavakoli, 2011). In this case, the initialization of both P_a and α parameters are fixed, hence cannot be amended in the next iterations. Consequently, the main drawback appears in terms of number of iterations required to find optimal solutions. In case if the value of P_a is small and the value of α is large, the performance of the original CS algorithm will be ineffective, which leads to significant increase in number of iterations. In contrast, if the value of P_a is large and the value of α is small, the convergence rate will be high (small number of iterations needed) but still unable to find optimal solutions (Kamat & Karegowda, 2014). In this research, an adaptive P_a is introduced to control the convergence rate so that optimal solutions can be found at the reasonable number of iterations. Hence, the dynamic P_a can increase the diversity of solutions and approximation capability.

Secondly, based on the assumption that a cuckoo lays one egg at a time at a nest of other host bird, the egg survival is highly risk. In other words, it is barely for one egg hatches successfully after being discovered by the host bird. In this regard, there is a need of introducing a dynamic parameter along with the fixed step size, α for cuckoos

to search around the potential good nests (solutions) for laying egg. Precisely, this can been achieved by introducing and embedding an adaptive inertia weight, *w* parameter along with the fixed α to search for new nests (solutions), $x^{(t+1)}$ via the Lévy flight motion. Hence, cuckoos are capable to explore more rigorously for a better environment if the current habitat is not suitable for breeding.

Thirdly, the assumption is made in the original CS algorithm where the number of available of random host nests is fixed. The host birds spot the cuckoos' eggs with a probability $P_a \in [0, 1]$. For such incidents, the host birds will either evict the parasitic eggs or abandon the nests totally and seeks for a new site to rebuild the nests (Ong 2014). In this case, there is a missing mechanism to select the fittest nests (solutions) with the best probability of cuckoos' eggs survival. This becomes crucial with the scarcity number of host nests. In this research, a Roulette wheel selection operator is introduced to do an initial stochastic selection of highly potential host nests (solutions).

Fourthly, this study synthesizes a linear antenna array only. Geometrically, the linear antenna array is simple to be synthesized since all the isotropic radiators or symmetric elements are aligned straight along the *xy*–plane. Hence, it is easy to implement and verify the postulated algorithms on the linear antenna array to search for optimal solutions with the aim of a better SLL suppression and/or significant predefined nulls mitigation.

Fifthly, this study just applies two signal processing windows or tapering functions only, which are the uniform and Doppler–Chebyshev windows. These two windows are the two most common windows used in geometry linear array synthesis. Most literatures state that many previous geometry array syntheses used these two windows due to its straightforwardness. Hence, it is compatible to compare the

postulated algorithms performance with the competitors using the same signal processing windows as stated in literatures.

Sixthly, the postulated MCS algorithm is hybridized and compared with only two stochastic algorithms, which are particle swarm optimization (PSO) and genetic algorithms (GA). These two well–known metaheuristic algorithms are used because of their simplicity and flexibility in finding optimal solutions in *N*–dimensional search space. Besides, there is the hybridization of MCS algorithm with one Pareto optimum technique only, which is the strength Pareto evolutionary algorithm (SPEA) to find MO trade–off solutions. The SPEA is chosen because of its simple routines and has a significant capability in approximating Pareto dominance.

Seventhly, this study is completely based on the software design of adaptive antenna signal processing component only where various metaheuristic algorithms development and verification are done via MATLAB iterative simulations. Hence, there is no study on the hardware design, such as circuit fabrication and prototyping which can verify the postulated algorithms.

7.3 Future Work

Since this is a pioneering study on the postulation of enhanced and hybrid versions of CS algorithm, there are large opportunities for further study in the near future. Firstly, the synthesis can be done in a more complex array geometry, e.g. planar/rectangular, circular, cylindrical, and spherical array. This requires a more intensive and complex 3D–simulations with certain fine–tuned CS internal parameters.

Secondly, the array geometry synthesis possibly can be executed in few other signal processing windows or tapering functions e.g. Tukey, Kaiser, Poisson, Hamming, Hann, Blackman, and Taylor. Two possible purposes are to increase the result

variations, and to find the scientific relations between postulated algorithms and various current amplitude tapering techniques. Different signal processing window may need alteration of CS internal parameters.

Thirdly, the postulated MCS algorithm can be hybridized and compared with few other popular stochastic algorithms, e.g. ant colony optimization (ACO), harmony search (HC), artificial bee colony (ABC), artificial immune systems (AIS), artificial neural network (ANN), firefly algorithm (FA), invasive weed optimization (IWO), and intelligent water drops (IWD). These metaheuristic algorithms are either good in global search or local search. Besides, there can be also a hybridization of MCS algorithm with other Pareto optimum methods, e.g. non–dominated sorting genetic algorithm (NSGA), vector evaluated genetic algorithm (VEGA), and niched Pareto genetic algorithm (NPGA). Hence, this can generate more varieties of simulation result.

Fourthly, the process of circuit fabrication and prototyping can be done in the future to develop the respective hardware design. In this case, certain feasibility studies, analyses and adjustments are required to fulfil hardware design requirements and specifications. The verifications in both software (via the CST Studio Suite software) and hardware design can enrich the values and contributions of postulated algorithms in adaptive array geometry synthesis.

REFERENCES

Abreu, G. & Kohno, R. (2002). Chebyshev–Like Low Side Lobe Beampatterns with Adjustable Beamwidth and Steering−Invariance. *The European Wireless 2002*.

Alexopoulos, A. (2006). *Phased Array Analysis using a Modified Chebyshev Approach* (DSTO−TR−1806). Edinburgh, Australia: Defence Science and Technology Organisation of Australia. Retrieved from http://www.dsto.defence.gov.au/publications/4425/DSTO−TR−1806.pdf

Applebaum, S. P. (1976). Adaptive Arrays. *IEEE Transactions of Antennas and Propagation*, 24, 585–598.

Balanis, C. A. & Ioannides, P. (2007). *Introduction to Smart Antennas*. (1st Ed). San Francisco, CA: Morgan and Claypool Publishers.

Balanis, C. A. (2005). *Antenna Theory: Analysis and Design*. (3rd Ed). New York: John Wiley & Sons.

Banerjee, S. & Dwivedi, V. V. (2013). Review of Adaptive Linear Antenna Array Pattern Optimization. *International Journal of Electronics and Communication Engineering*, 2(1), 25–42.

Barthelemy, P., Bertolotti, J. & Wiersma, D. S. (2008). A Lévy Flight for Light. *Nature*, 453, 495−498.

Basu, B. & Mahanti, G. K. (2011). Fire Fly and Artificial Bees Colony Algorithm for Synthesis of Scanned and Broadside Linear Array Antenna. *Progress in Electromagnetics Research B*, 32, 169−190.

Blum, C. & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 35(3), 268−308.

Brockhoff, D., Friedrich, T. & Neumann, F. (2008). Analyzing Hypervolume Indicator Based Algorithms. *Proceedings of the 2008 10th International Conference on Parallel Problem Solving from Nature−Indicator based Evolutionary Algorithm* (*PPSN−IBEA*).

Brown, C. T., Liebovitch, L. S. & Glendon, R. (2007). Lévy Flights in Dobe Ju / 'Hoansi Foraging Patterns. *Human Ecology*, 35, 129−138.

Butler, J., & Lowe, R. (1961). Beam Forming Matrix Simplifies Design of Electrically Scanned Antennas. *Electronic Design*.

Chambers, J. M., Mallows, C. L. & Stuck, B. W. (1976). A Method for Simulating Stable Random Variables. *Journal of the American Statistics Association*, 71, 340−344.

Cengiz, Y. & Tokat, H. (2008). Linear Antenna Array Design with use of Genetic, Memetic and Tabu Search Optimization Algorithms. *Progress in Electromagnetics Research C*, 1, 63−72.

Clerc, M. (1999). The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. *1999 Congress on Evolutionary Computation*, 3, 1951−1957.

De Castro, L. N. (2006). *Fundamentals of Natural Computing*. Boca Raton, FL: Chapman & Hall/CRC Press.

Dolph, C. A. (1946). A Current Distribution for Broadside Arrays which Optimizes the Relationship between Beamwidth and Side Lobe Level. *Institute of Radio Engineers*, 335–348.

Dreo, J., Petrowski, A., Siarry, P. & Taillard, E. (2006). *Metaheuristics for Hard Optimization*, Berlin, Germany: Springer.

Eberhart, R. C. & Shi, Y. (2001). Particle Swarm Optimization: Developments, Applications and Resources. *IEEE Congress on Evolutionary Computation.*

Eberhart, R. C., Simpson, P. K., & Dobbins, R. W. (1996). *Computational Intelligence PC Tools.* Boston, MA: Academic Press Professional.

Floreano, D. & Mattiussi, C. (2008). *Bio−Inspired Artificial Intelligence: Theories, Methods, and Technologies*. Cambridge, MA: The MIT Press.

Gómez, N. G., Melde, K. L., McNeill, K. M., & Rodriguez, J. J. (2006). Development of Array Distributions for Smart Antennas with Low Side Lobes, Interference−Nulling, and Effective Radiated Voltage Constraints. *The 2006 Antennas and Propagation Society International Symposium*, 3323−3326.

Ghiasi, H., Pasini, D. & Lessard, L. (2011). A Non−Dominated Sorting Hybrid Algorithm for Multi-objective Optimization of Engineering Problems. *Engineering Optimization*, 43(1), 39−59.

Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers and Operations Research*, 5, 533−549.

Godara, L. C. (2004). *Smart Antennas*. (1st Ed). Boca Raton, FL: CRC Press.

Gomez, J. S. & Covarrubias, D. H. (2009). A Synthesis of Unequally Spaced Antenna Arrays using Legendre Functions. *Progress in Electromagnetics Research M*, 7, 57–69.

Goswami, B. & Mandal, D. (2012). A Genetic Algorithm for the Level Control of Nulls and Side Lobes in Linear Antenna Arrays. *Journal of King Saud University–Computer and Information Sciences*.

Goudos, S. K. et al. (2010). Application of a Comprehensive Learning Particle Swarm Optimizer to Unequally Spaced Linear Array Synthesis with Side Lobe Level

Suppression and Null Control. *IEEE Antennas and Wireless Propagation Letters*, 9, 125−129.

Greenberg, M. (1998). *Advanced Engineering Mathematics*. (3rd Ed.). Englewood Cliffs, NJ: Prentice Hall.

Günes, F. & Tokan, F. (2010). Pattern Search Optimization with Applications on Synthesis of Linear Antenna Arrays. *Science Direct Expert Systems with Applications*, 37, 4698–4705.

Ho, M. H., Liao, S. H. & Chiu, C. C. (2010). A Novel Smart UWB Antenna Array Design by PSO. *Progress in Electromagnetics Research C*, 15, 103−115.

Ho, S. L. et al. (2006). A Particle Swarm Optimization Method with Enhanced Global Search Ability for Design Optimizations of Electromagnetic Devices. *IEEE Transactions on Magnetics*, 42 (4), 1107–1110.

Ho, S. L. et al. (2005). A Particle Swarm Optimization–Based Method for Multiobjective Design Optimizations. *IEEE Transactions on Magnetics*, 41 (5), 1756–1759.

Ho, M. J., Stuber G. L., & Austin, M. D. (1998). Performance of Switched−Beam Smart Antennas for Cellular Radio System. *IEEE Transactions on Vehicle Technology*, 47 (1), 10−19.

Jain, R. K., Katiyar, S. & Agrawal, N. K. (2011). Smart Antenna for Cellular Mobile Communications. *VSRD International Journal of Electrical, Electronics & Communication Engineering*, 1(9), 530–541.

Kamat, S. & Karegowda, G. (2014). A Brief Survey on Cuckoo Search Applications. *International Journal of Innovative Research in Computer and Communication Engineering*. 2(2), 7–14.

Karaboga, D., Guney, K. & Akdagli, A. (2004). Antenna Array Pattern Nulling by Controlling both Amplitude and Phase using Modified Touring Ant Colony Optimization Algorithm. *International Journal of Electronics*, 91(4), 241–251.

Kaur, K. & Banga, V. K. (2013). Synthesis of Linear Antenna Array using Firefly Algorithm. *International Journal of Scientific & Engineering Research*, 4(8), 601–606.

Keizer, W. P. M. N. (2009). Low Side Lobe Pattern Synthesis using Iterative Fourier Techniques Coded in MATLAB. *IEEE Transactions of Antennas and Propagation*, 51(2), 137–150.

Kennedy, J. & Eberhart, R. C. (1995). Particle Swarm Optimization. *Proceedings of the IEEE Conference on Neural Networks*, 4, 1942−1948.

Khodier, M. M. & Al−Aqeel, M. (2009). Linear and Circular Array Optimization: A Study using Particle Swarm Intelligence. *Progress in Electromagnetics Research B*, 15, 347−373.

Khodier, M. M. & Christodoulou, C. G. (2005). Linear Array Geometry Synthesis with Minimum Side Lobe Level and Null Control using Particle Swarm Optimization. *IEEE Transactions on Antennas and Propagation*, 53(8), 2674–2679.

Kim, Y. & De Weck, O. (2004). Adaptive Weighted−Sum Method for Multiobjective Optimization. *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*.

Kumar, B. P. & Branner, G. R. (1999). Design of Unequally Spaced Arrays for Performance Improvement. *IEEE Transactions of Antennas and Propagation*, 47 (3), 511–523.

Laseetha, T. S. J. & Sukanesh, R. (2011). Synthesis of Linear Antenna Array using Genetic Algorithm to Maximize Sidelobe Level Reduction. *International Journal of Computer Applications*, 20(7), 27−33.

Lee, K. C. (2005). Frequency−Domain Analyses of Nonlinearly Loaded Antenna Arrays using Simulated Annealing Algorithms. *Progress in Electromagnetics Research C*, 53, 271−281.

Lévy Flight. (2013, February 28). Retrieved May 2, 2013, from Wikipedia website: http://en.wikipedia.org/wiki/ Lévy _flight

Li, W. T. et al. (2008). An Improved Particle Swarm Optimization Algorithm for Pattern Synthesis of Phased Arrays. *Progress in Electromagnetics Research*, 82, 319–332.

Liberti, J. & Rappaport, T. S. (1999). *Smart Antennas for Wireless Communications*. Englewood Cliffs, NJ: Prentice Hall.

Litva, J., & Lo, T. K. (1996). *Digital Beam Forming in Wireless Communications*. Norwood, MA: Artech House.

Lynch, P. (1997). The Dolph−Chebyshev Window: A Simple Optimal Filter. *American Meteorological Society*, 125, 655−660.

Mandal, A., Zafar, H., Das, S. & Vasilakos, A. V. (2012). Efficient Circular Array Synthesis with a Memetic Differential Evolution Algorithm. *Progress in Electromagnetics Research B*, 38, 367−385.

Mantegna, R. N. (1994). Fast, Accurate Algorithm for Numerical Simulation of Lévy Stable Stochastic Processes. *Physical Review E*, 49, 4677−4683.

Mouhamadou, M. & Vaudon, P. (2006). Smart Antenna Array Patterns Synthesis: Null Steering and Multiuser Beamforming by Phase Control. *Progress in Electromagnetics Research*, 60, 95–106.

Nik Abd Malik, N. N., Esa, M., Syed Yusof, S. K. & Marimuthu, J. (2009). Suppression of Antenna's Radiation Side Lobes using Particle Swarm Optimization. *Proceedings of the Progress in Electromagnetics Research Symposium*, 683−686.

Nikolova, N. (2012). *Linear Array Theory − Part I* [PDF document]. Retrieved from the Lecture Notes Online Web Site: http://www.ece.mcmaster.ca/faculty/nikolova/antenna_dload/current_lectures/L13_Arrays1.pdf

Ong, P. (2014). Adaptive Cuckoo Search Algorithm for Unconstrained Optimization. *The Scientific World Journal*. 2014, 1−8.

Pal, S., Basak, A., Das, S. & Abraham, A. (2009). Linear Antenna Array Synthesis with Invasive Weed Optimization Algorithm. *Proceedings of the International Conference of Soft Computing and Pattern Recognition*, 161−166.

Panduro, M. A., Brizuela, C. A., Balderas, L. I. & Acosta, D. A. (2009). A Comparison of Genetic Algorithm, Particle Swarm Optimization and the Differential Evolution Methods for the Design of Scannable Circular Antenna Arrays. *Progress in Electromagnetics Research B*, 13, 171−186.

Pappula, L. & Ghosh, D. (2014). Constraint−based Synthesis of Linear Antenna Array using Modified Invasive Weed Optimization. *Progress in Electromagnetics Research M*, 36, 9−22.

Particle Swarm Optimization. (2011, October 21). Retrieved April 28, 2013, from the Scholarpedia Web Site URL: http://www.scholarpedia.org/article/Particle_swarm_optimization

Payne, R. B. (2005). *The Cuckoos* (M. D. Sorensen, Ed.) (K. Klitz, Illustrator) New York: Oxford University Press.

Pavlyukevich, I. (2007a). Lévy Flights, Non−Local Search and Simulated Annealing. *Journal of Computational Physics*, 226(2), 1830−1844.

Pavlyukevich, I. (2007b). Cooling Down Lévy Flights. *Journal of Physics. A: Mathematical and Theoretical*, 40, 12299−12313.

Recioui, A. (2012). Side Lobe Level Reduction in Linear Array Pattern Synthesis Using Particle Swarm Optimization. *Journal of Optimization Theory and Applications*, 153(2), 497−512.

Reeves, C. R. (Ed.) (1993). *Modern Heuristic Techniques for Combinatorial Problems*. Oxford, UK: Blackwell Scientific Publishing.

Reynolds, A. M. & Frye, M. A. (2007). Free−Flight Odor Tracking in Drosophila is Consistent with an Optimal Intermittent Scale-Free Search. *PLoS One*, 2, e354.

Rocha, C., Covarrubias, D. H., Brizuela, C. A. & Panduro, M. A. (2007). Differential Evolution Algorithm Applied to Side Lobe Level Reduction on a Planar Array. *International Journal of Electronics and Communications*, 61(5), 286–290.

Ryu, J−H., Kim, S. & Wan, H. (2009). Pareto Front Approximation with Adaptive Weighted−Sum Method in Multiobjective Simulation Optimization. *Proceedings of the 2009 Winter Simulation Conference*, 623−633.

Sattari, P. & Hejazi, N. (2008). Array Pattern Null Steering using Genetic Algorithm by Element Position Perturbations. *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering*, 423–428.

Shan, Y. T. (2010). An Improved Genetic Algorithm and its Blending Application with Neural Network. *The Proceedings of the 2010 2nd International Workshop on Intelligent Systems and Applications*, 1−4.

Sharma, A. & Cecil, K. (2014). Optimization of Linear Antenna Array using Big Bang Crunch Algorithm for Reduction in Side Lobe Levels. *International Journal of Engineering and Innovative Technology*, 3(7), 97−99.

Shlesinger, M. F. (2006). Search Research. *Nature*, 443, 281−282.

Shihab, M., Najjar, Y., Dib, N. & Khodier, M. (2008). Design of Non−Uniform Circular Antenna Arrays using Particle Swarm Optimization. *Journal of Electrical Engineering*, 59, 216−220.

Singh, U. Kumar, H. & Kamal, T. S. (2010). Linear Array Synthesis using Biogeography−based Optimization. *Progress in Electromagnetics Research M*, 11, 25−36.

Stevanović, I., Skrivervik, A. & Mosig, J. R. (2003). Smart Antenna Systems for Mobile Communications. *Swiss Federal Office of Communications Activity Final Report*.

Tsoulos, G. V. (2001). *Adaptive Antennas for Wireless Communication Systems*. New York: IEEE Press.

Valian, E., Mohanna, S. & Tavakoli, S. (2011). Improved Cuckoo Search Algorithm for Feedfoward Neural Network Training. *International Journal of Artificial Intelligence & Applications*, 2 (3), 36–43.

Walia, A. & Patterh, M. S. (2013). Antenna Array Failure Correction using Optimization Techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(9), 764–769.

Widrow, B., Mantey, P. E., Griffiths, L. J. & Goode, B. B. (1967). Adaptive Antenna Systems. *Proceedings of the IEEE*, 55(12), 2143–2159.

Wolpert, D. H. & Macready, W. G. (1997). No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1), 67–82.

Yang, X. S. & Deb, S. (2010). Engineering Optimization by Cuckoo Search. *International Journal of Mathematical Modelling and Numerical Optimisation*, 1, 330−343.

Yang, X. S. & Deb, S. (2009). Cuckoo Search via Lévy Flights. *The 2009 World Congress on Nature & Biologically Inspired Computing*, 210−214.

Yilmazer, N., Burintramart, S., & Sarkar, T. K. (2008). Smart Antennas. K. Fujimoto (Ed.), *Mobile Antenna Systems Handbook*, 647–673.

Zaharis, Z. et al. (2006). Optimal Design of a Linear Antenna Array using Particle Swarm Optimization. *The Proceedings of the 5th WSEAS International Conference on Data Networks, Communications and Computers*.

Zaman, M. A. & Abdul Matin, M. (2012). Nonuniformly Spaced Linear Antenna Array Design using Firefly Algorithm. *International Journal of Microwave and Optical Technology*, 2012, 1–8.

Zaman, M. A., Md. Mushfiqul Alam, M. G., Mamun, S. A. & Abdul Matin, M. (2011). Synthesis of Antenna Arrays using Artificial Bee Colony Optimization Algorithm. *International Journal of Microwave and Optical Technology*, 6(4), 234–241.

Zhang, Z. Li, T., Yuan, F. & Yin, L. (2014). Synthesis of Linear Antenna Array using Genetic Algorithm to Control Side Lobe Level. *Computer Engineering and Networking, Lecture Notes in Electrical Engineering*. (W. W. Wong & T. Zhu, Eds.) Switzerland: Springer, 277, 39−46.

Zitzler, E., Laumanns, M. & Bleuler, S. (2004). A Tutorial on Evolutionary Multiobjective Optimization. *Metaheuristics for Multiobjective Optimization, Lecture Notes in Economics and Mathematical Systems*, (X. Gandibleux,

M. Sevaux, K. Sörensen & V. T'kindt, Eds.) Berlin Heidelberg: Springer−Verlag, 535, 3−37.

Zitzler, E. (1999). *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications* (Doctoral Dissertation). Swiss Federal Institute of Technology Zurich.

LIST OF PUBLICATIONS

1. **Abdul Rani, K. N.**, Abd Malek, M. F., Neoh, S. C. & Abd Wahab, A. (2014). Modified and Hybrid Cuckoo Search Algorithms via Weighted–Sum Multiobjective Optimization for Symmetric Linear Array Geometry Synthesis. *International Journal of Advanced Research in Computer and Communication Engineering*, 3, 5, 6774−6781. **Impact Factor: 1.770**.
2. **Abdul Rani, K. N.**, Abd Malek, M. F. & Neoh, S. C. (2012). Nature−Inspired Cuckoo Search Algorithm for Side Lobe Level Suppression in a Symmetric Linear Antenna Array. *Radioengineering*, 21, 3, 865−874. **Impact Factor: 0.687. Indexed by INSPEC and SCOPUS**.
3. **Abdul Rani, K. N.**, Abd Malek, M. F., Neoh, S. C., Jamlos, F., Mohd Affendi, N. A., Mohamed, L., Saudin, N. & Abdul Rahim, H. (2012). Hybrid Multiobjective Optimization using Modified Cuckoo Search Algorithm in Linear Array Synthesis. *The Proceedings of the 2012 Loughborough Antennas & Propagation Conference*. **Indexed by IEEE XPLORE and SCOPUS**.
4. **Abdul Rani, K. N.**, Abd Malek, M. F., Neoh, S. C., Wee, F. H., Mohd Affendi, N. A., Mohamed, L., Saudin, N. & Ali, A. (2012). Modified Cuckoo Search Algorithm in Weighted Sum Optimization for Linear Antenna Array Synthesis. *The Proceedings of the 2012 IEEE Symposium on Wireless Technology and Applications*, 223−228. **Indexed by IEEE XPLORE and SCOPUS**.
5. **Abdul Rani, K. N.** & Abd Malek, M. F. (2011). Symmetric Linear Antenna Array Geometry Synthesis using Cuckoo Search Metaheuristic Algorithm. *The Proceedings of the 2011 17th Asia-Pacific Conference on Communications*, 374−379. **Indexed by IEEE XPLORE.**

6. **Abdul Rani, K. N.** & Abd Malek, M. F. (2011). Preliminary Study on Cuckoo Search Parameters for Symmetric Linear Array Geometry Synthesis. *The Proceedings of the 2011 Trends and Developments in Converging Technology towards 2020 Conference*, 568−572. **Indexed by IEEE XPLORE and SCOPUS.**