JOURNAL OF NANO- AND ELECTRONIC PHYSICS
Vol. **17** No 6, 06017(5pp) (2025)

ЖУРНАЛ НАНО- ТА ЕЛЕКТРОННОЇ ФІЗИКИ
Том **17** № 6, 06017(5cc) (2025)

REGULAR ARTICLE

Machine Learning Algorithms for Adaptive Beamforming in Smart Antenna Systems

A.P. Kumar[1,*], Kali Varaprasad B[2], P.R. Kumar[3]

[1] *Department of AIML & ECE, CMR Engineering College, Hyderabad, India*
[2] *Department of ECE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India*
[3] *Department of Artificial Intelligence and Machine Learning, Malla Reddy University, Hyderabad, India*

(Received 08 August 2025; revised manuscript received 17 December 2025; published online 19 December 2025)

Smart antennas are now a day's one of the most important elements of wireless communications technology. This is because of their special characteristics and features. The future application of smart antenna technologies in wireless structures, however, will likely greatly influence spectrum use efficiency, minimize the cost of new networks, improve service quality and simplify the various wireless network technologies. Machine Learning (ML) Techniques are very efficient, powerful, and popular techniques in recent decades. In resolving various nonlinear issues that cannot be simply solved with traditional techniques, Artificial Neural Networks (ANN) have demonstrated their vast capacity to provide higher performance. In this article, the application of artificial neural networks to smart antenna systems is introduced. The most prominent adaptive beam-forming techniques are compared with the smart antenna ANN algorithm. The Python and its powerful library "matplotlib.pyplo" and "pyroomacoustics" functions will be used to decide the weights of the antenna features to decrease the error in the signal received using the ANN back propagation algorithm.

Keywords: Beam forming, ANN, Python, AOA.

DOI: 10.21272/jnep.17(6).06017 PACS numbers: 07.05.Tp, 84.40.Ba

1. INTRODUCTION

Smart antenna systems enhance wireless performance by increasing gain in desired directions and suppressing interference [1]. They improve signal quality, network capacity, and coverage while saving power. Beamforming, a key feature, adjusts signal size and phase using algorithms like LMS to reduce mean square error and prevent interference. Two types of smart antennas, switched beam and adaptive arrays, employ beamforming to track and optimize signal direction. Neural networks, including RBFNN and ADALINE, have been applied for multi-source tracking and efficient beamforming, offering speed and accuracy without prior source knowledge. Reconfigurable antenna systems and modern ML techniques (ANNs, KNN, Lasso) have further optimized antenna performance for IoT and Massive MIMO applications. Advanced algorithms enable adaptive beamforming, side lobe reduction, and DOA estimation, improving efficiency in LTE, 4G, and beyond. New ML-driven adaptive techniques refine beam formation by minimizing signal errors, ensuring smart antennas meet growing wireless demands [2].

2. ADAPTIVE BEAMFORMING ALGORITHM

Adaptive Beamforming is a method using a range of antennae to maximize the receipt of a signal from a chosen direction in a definite path while rejecting signals from the identical frequency from other indication [3].

2.1 Mathematical Model

Consider a uniform linear array (ULA) in which the M omnidirectional sensors are spaced on the incoherent plane waves from distance d-D, pushing {θ_1, θ_2, ... θ_{D-1}}.

$$(k) = s(k) + i(k) + n(k) = s_0(k)a + i(k) + n(k) \qquad (1)$$

Where (k) is the composite vector of array remarks and it stated as:

$$(k) = [x_1(k),\ x_2(k), \dots ,\ x_m(k)]^T \qquad (2)$$

$s(k)$ – the signal waveform, a is the signal steering vector, $i(k)$ – the interference component, $n(k)$ – the noise factor. A narrowband beam former's output is:

$$(k) = \omega^H X(k) \qquad (3)$$

Where ω the composite is vector of beam former weight and is expressed as:

$$\omega = [\omega_1, \omega_2, \dots , \omega_M] \qquad (4)$$

[*] Correspondence e-mail: pramodvce@gmail.com

The signal to interference plus noise ratio (SINR) has the subsequent method

$$SINR = \frac{H_{R_{sW}}}{W^H R_i + n^W} \tag{5}$$

Where Rs is $M\ X\ M$ signal matrix that is the numerical probability of signal vector and it is:

$$\{s(k)s^H(k)\} \tag{6}$$

R_{i+n} is signal plus noise covariance matrix as:

$$R_{i+n} = (i(k) + n(k))(i(k) + n(k))^H \tag{7}$$

To optimize the performance of a specific test, the adaptive beam former weight vector is calculated. Although we can utilize various criteria, we restrict ourselves to the output SINR condition, which is changed to:

$$SINR = \frac{\sigma^2 |w^H a|^2}{w^H R_i + n^w} \tag{8}$$

Where σ_s^2 is the signal power. The challenge of discovery the maximum of (8) is the succeeding difficult of optimization.

$$Min\ w^H R_{i+n} w\ subject\ to\ w^H a = 1 \tag{9}$$

The succeeding resolution can be found from (9) for the finest weight vector.

$$W = \frac{R_{i+n}^{-1}}{a^H R_{i+n}^{-1} a} \tag{10}$$

We find that the best SINR is supplied by inserting (10) into (8).

$$SINR_{opt} = \sigma_s^2 a^H R_{i+n}^{-1} a \tag{11}$$

Where equation (11) gives an upper bound on the output SINR (8).

3. ARTIFICIAL NEURAL NETWORK

Artificial neural networks are incredibly influential architectures of brain-stimulated computing units. They were designed for addressing nonlinear difficult problems, which appear to be wasteful in regular mathematical procedures. There is the necessity for learning as the principal commonality between a biological and artificial thinking system [4]. ANN's main transmission functions are the sigmoid vector and logarithm, as we already said. The sigmoid logarithm formula is:

$$(S) = \frac{1}{1+e^{-cs}} \tag{12}$$

Where S is the transmission function input, and c is a constant used for controlling the transfer function pitch. For all real values this function is continuous and its derivatives are given by:

$$o(S) = \frac{ce^{-cS}}{(1+e^{-c})^2} = \frac{1+ce^{-cS}-1}{(1+e^{-cS})^2} \tag{13}$$

The function can be written as:

$$o(S) = \frac{ce^{-cS}}{(1+e^{-c})^2} - \frac{1}{(1+e^{-c})^2} \tag{14}$$

If we choose the value of $c = 1$, then the function decreases to:

$$(S) = \frac{1+e^{-S}}{(1+e^{-})^2} - \frac{1}{(1+e^{-})^2} = \frac{1}{1+e^{-S}}\left(1 - \frac{1}{1+e^{-S}}\right) = (S)(1-(S)) \tag{15}$$

For each layer, the future network calculations can be started by:

$$S = \sum I_n \omega_n + b_n \tag{16}$$

In this context, W_n denotes a matrix containing the weights of each layer, I_n represents the vector of inputs, b_n comprises the bias values utilized to stabilize the network function. In order to ascertain the power output of the computed sum, activation is utilized. The final activation function involves the monitoring of the network output [5]. These outputs ought to resemble the desired output. Error propagation occurs when the discrepancy between the output and the objective value constitutes the error utilized:

$$E = (o_d - o) \tag{17}$$

Where o_d is the anticipated output. Using error E as a value, we extract an error function that looks like:

$$\Delta_k = (o_{dk} - o_k)_k (1 - o_k) \tag{18}$$

The novel values of the past weights are once more determined based on this calculated value. The output layer weights are then specified by:

$$\omega_{k\,new}^h = \omega_{k\,old}^h + \eta \Delta_k^h o^{h-1} + (\delta \omega_{k\,old}^h) \tag{19}$$

Another feature that takes account of the fault in the output layer is used to update the hidden layers. The term for the update is:

$$\Delta^h = o^h (1 - o^h) \sum \omega_k^h \Delta_k \tag{20}$$

The new weights values are:

$$\omega_{k\,new}^h = \omega_{k\,old}^h + \mu \Delta^h o_k^h + (\delta \omega_{k\,old}^h) \tag{21}$$

There are two key parameters in ANN α and μ. The training of a backbone distribution network is controlled. The learning rate μ adjusts how quickly it is. The learning rate of a neural network can be increased or decreased [6, 7]. It is an important factor because it is quite complicated to choose. The higher learning rate can make the network unite quicker but fast convergence can lead to memorization rather than learning phenomena. A low level of learning can significantly extend the time of training. Another key ANN learning parameter based on Error Back propagation is the momentum factor α.

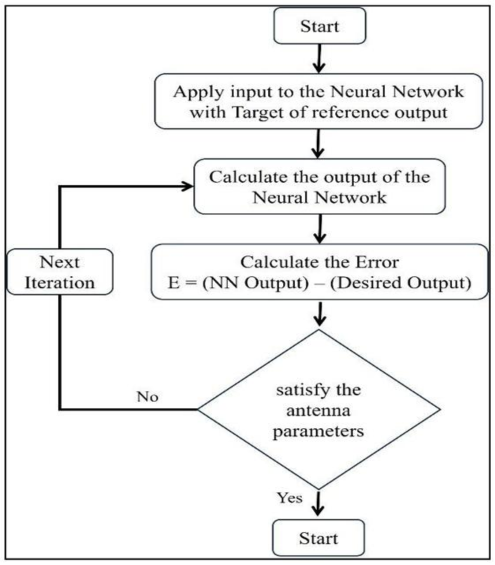
Start
Apply input to the Neural Network with Target of reference output
Calculate the output of the Neural Network
Next Iteration
Calculate the Error E = (NN Output) – (Desired Output)
satisfy the antenna parameters
No
Yes
Start

Fig. 1 – Flow Chart of ANN Training Program

4. SIMULATION AND RESULTS

In this simulation procedure, the performance of common adaptive methods including LMS, RLS, and CMA are analyzed and compared. The intended signal's amplitude reaction, convergence time, acquisition, and monitoring were just few of the metrics used in the comparison. DOA 30 degrees of desire signal and DOA - 60 degrees of interference signal were employed as simulation parameters [8, 9]. The simulations vary a parameter of the antenna array for the same process, such as the sum of sensor elements or the space among the two parts of the array. The simulation outcomes are shown in the following figures for a range of element numbers and inter-element distances. Arrays containing 4, 8, and 16 items were employed at $d = 0.125\lambda$, 0.25λ, and 0.5λ distances. To avoid spatial aliasing, the maximum distance between two elements was 0.5λ.

4.1 LMS Algorithm

The Least Mean Square (LMS) algorithm was tested for various array sizes at spacing $d = 0.5\lambda$. It amplified the desired signal while suppressing interference.

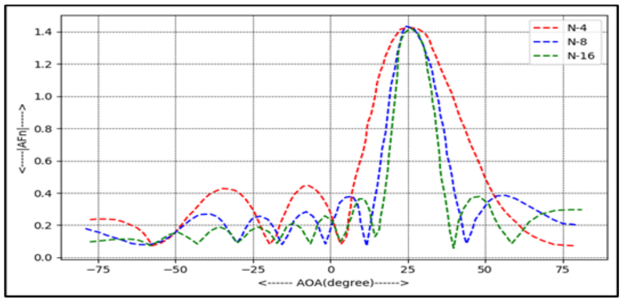

Fig.2 – Beam formation of LMS algorithm for desire angle at 30° and null at – 60° for ($d = 0.5\lambda$)

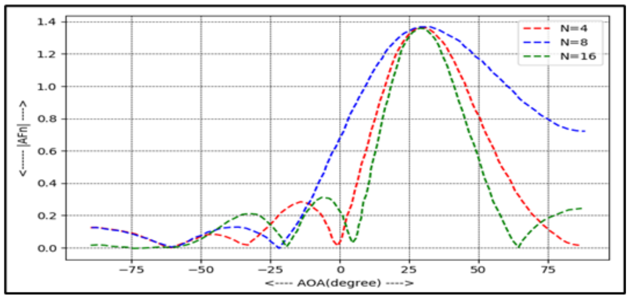

Fig. 3 – Beam formation of LMS algorithm for desire angle at 30° and null at – 60° for ($d = 0.25\lambda$)

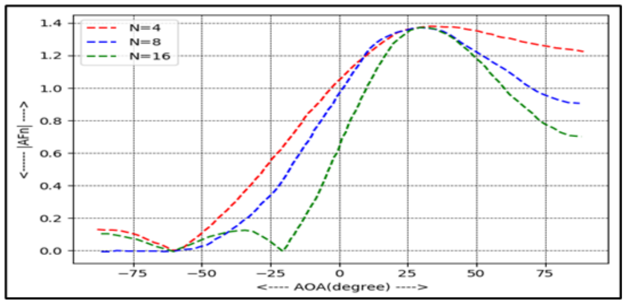

Fig. 4 – Beam formation of LMS algorithm for desire angle at 30° and null at – 60° for ($d = 0.125\lambda$)

Using 16 array elements improved responsiveness by narrowing the reaction band. Similar results were observed for $d = 0.25\lambda$ and $d = 0.125\lambda$. However, reducing array element spacing decreased response accuracy, highlighting that larger arrays enhance algorithm performance

4.2 RLS Algorithm

The RLS algorithm was used for the same trials. In this portion Figs. 5, 6, 7 show the outcomes. The results suggest that when bigger numbers of pieces are used the reaction is better. At certain unwanted angles of reach, the answer shows an amplitude of around 0.4. These values are regarded as disadvantages of the algorithm. The results for $N = 16$ have been enhanced for $d = 0.5\lambda$, however for the use of $d = 0.25\lambda$ and $d = 0.125\lambda$ this was not the case. The $N = 8$ instance has shown improved results. For a reduced inter-elemental distance, the performance in terms of passing belt has proven better.

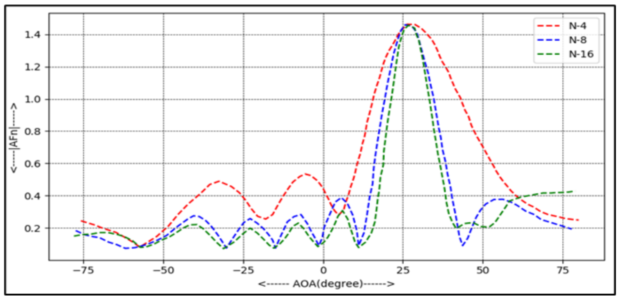

Fig. 5 – Beam formation of RLS algorithm for desire angle at 30° and null at – 60° for ($d = 0.5\lambda$)

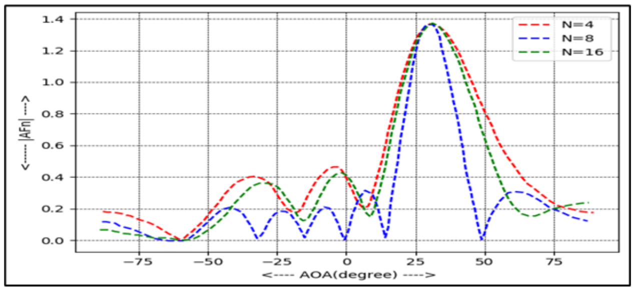

Fig. 6 – Beam formation of RLS algorithm for desire angle at 30° and null at – 60° for ($d = 0.25\lambda$)

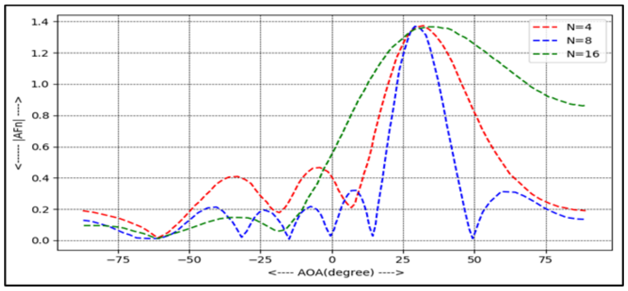

Fig. 7 – Beam formation of RLS algorithm for desire angle at 30° and null at – 60° for ($d = 0.125\lambda$)

4.3 Effect of N and d using ANN

The same experiments were carried out using Python and its powerful library to draw an array factor for ANN for varying element numbers and positioning. At AOA = 30°, the intended signal is reached, while the desired signal is reached at AOA = – 60°. The utilization in the array of 16 components has enhanced the neural network's reaction by decreasing the main response band, which is vital in certain applications, which need exact tracking, such as in the mobile communication system. However, with the reduction in the space among array elements and the passing band, the precision of the response is less than 0.5λ, when 0.125λ is achieved. It is also clear that raising the quantity of pieces leads to increasing the amount of side labels, an inconvenience that increases the chances of getting signals from undesirable directions. The act of ANN in terms of tracing the specified signal is analyzed using Python with $N = 8$ antennas spaced at half-wavelength ($d = 0.5\lambda$). Intend signal is received at AOA = 30 degrees, whereas desired signal is received at AOA = – 60 degrees. Figs. 8, 9 depicts the mean square error that emerges from this process, which approaches zero after 8 repetitions. The convergence rate is good; the display output attains and follows the required signal after about 8 iterations.

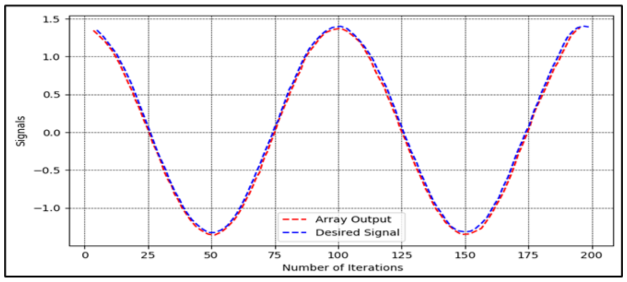

Fig. 8 – Achieve and track the desired ANN signal

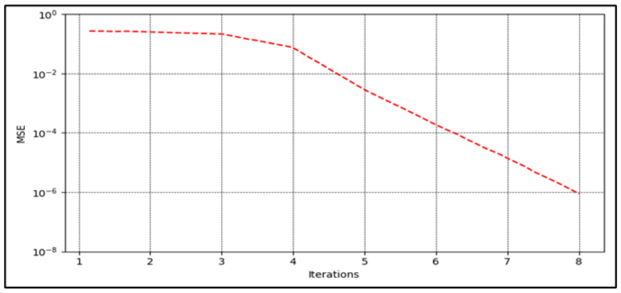

Fig. 9 – Mean Square Error in ANN method

Table 1 – Parameters of the used Neural Networks

Layers	5	Second hidden layer	6
Hidden Layer	3	Transfer functions	Linear
MSE	1e-5	Bias Weights	No
Iteration	1000	Learning rate	0.06

5. CONCLUSION

In recent years, the wireless communication industry has taken significant advantage of the range, speed, and capacity of such systems as a result of their high efficiency, which is based on the incorporation of the antenna array with a digital signal processing system. This study compared three widely used adaptive technologies: LMS, RLS, and CMA with back propagation neural networks to determine which is the most effective strategy for optimizing antenna performance. Here it is shown that the performance of Back propagation neural network performs best among others.

REFERENCES

1. R. Sharma, A. Senapati, J.S. Roy, *International Conference on Emerging Trends in Electronic Devices and Computational Techniques* (EDCT) IEEE Xplore, 1 (2018).
2. P. Mainkar, G.N. Jagtap, G. Mulay, *International Conference on Internet of Things and Applications* (IOTA), 213 (2016).
3. D. Rajeswaran, A.K. Nair, *3rd International Conference on Signal Processing and Integrated Networks (SPIN)* IEEE Xplore, 406 (2016).
4. B. Samantaray, K.K. Das, J.S. Roy, *2nd National Conference on Mechatronics, Computing, and Signal Processing*, MCSP-2017 (2017).
5. K. Lazarus, N.H. Noordin, Z. Ibrahim, K.H. Abas, Asia Multi-Conference on Modelling and Simulation, 19 (2016).
6. A. Rathore, D.K. Panda, 2017 *International Conference on Information, Communication, Instrumentation and Control* (ICICIC), 1 (2017).

7. S. Patra, A. Pandey, N. Nandni, S. Kumar, V. Jha, M. Kumar, *Int. J. of Adv. Res.* **3** No 5, 1459 (2015).
8. Y.M. Tabra, B.M. Sabbar, C. Science, *International Journal of Recent Technology and Engineering* (IJRTE) **16** No 2, 715 (2019).
9. A.H. El Zooghby, C.G. Christodoulou, M. Georgiopoulos, *IEEE Transactions on Antennas and Propagation* **48** No 5, 768 (2000).

Алгоритми машинного навчання для адаптивного формування променя в інтелектуальних антенних системах

A.P. Kumar[1], Kali Varaprasad B[2], P.R. Kumar[3]

[1] *Department of AIML & ECE, CMR Engineering College, Hyderabad, India*
[2] *Department of ECE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India*
[3] *Department of Artificial Intelligence and Machine Learning, Malla Reddy University, Hyderabad, India*

Розумні антени сьогодні є одним з найважливіших елементів технології бездротового зв'язку. Це пов'язано з їхніми особливими характеристиками та особливостями. Однак майбутнє застосування технологій інтелектуальних антен у бездротових структурах, ймовірно, значно вплине на ефективність використання спектру, мінімізує вартість нових мереж, покращить якість обслуговування та спростить різні технології бездротових мереж. Методи машинного навчання (МН) є дуже ефективними, потужними та популярними методами останніх десятиліть. Вирішуючи різні нелінійні проблеми, які неможливо просто вирішити традиційними методами, штучні нейронні мережі (ШНМ) продемонстрували свою величезну здатність забезпечувати вищу продуктивність. У цій статті представлено застосування штучних нейронних мереж до систем інтелектуальних антен. Найвідоміші методи адаптивного формування променя порівнюються з алгоритмом ШНМ для інтелектуальних антен. Python та його потужна бібліотека "matplotlib.pyplo" та функції "pyroacoustics" будуть використані для визначення вагових коефіцієнтів характеристик антени, щоб зменшити похибку в сигналі, отриманому за допомогою алгоритму зворотного поширення ШНМ.

Ключові слова: Формування балки, Штучні нейронні мережі, Python, AOA.