Research in Astronomy and Astrophysics, 22:035020 (11pp), 2022 March https://doi.org/10.1088/1674-4527/ac449d

Solving Surface Deformation of Radio Telescope Antenna by Artificial Neural Network

Bo-Yang Wang[1], Qian Ye[1,2], Li Fu[2], Guo-Xiang Meng[1], Jin-Qing Wang[2], Qing-Hui Liu[2], and Zhi-Qiang Shen[2]
[1] School of Mechanical Engineering, Shanghai Jiaotong University, Shanghai 200240, China; wby920422@sjtu.edu.cn, yeqian@sjtu.edu.cn, guoxiangm@yahoo.co.jp
[2] Shanghai Astronomical Observatory, Chinese Academy of Sciences, Shanghai 200030, China; fuli@shao.ac.cn, jqwang@shao.ac.cn, liuqh@shao.ac.cn, zshen@shao.ac.cn
Received 2021 November 28; accepted 2021 December 13; published 2022 February 25

Abstract

Recent investigations have derived the relation between the near-field plane amplitude and the surface deformation of reflector antenna, namely deformation-amplitude equation (DAE), which could be used as a mathematical foundation of antenna surface measurement if an effective numerical algorithm is employed. Traditional algorithms are hard to work directly due to the complex mathematical model. This paper presents a local approximation algorithm based on artificial neural network to solve DAE. The length factor method is used to construct a trial solution for the deformation, which ensures the final solution always satisfies the boundary conditions. To improve the algorithm efficiency, Adam optimizer is employed to train the network parameters. Combining the application of the data normalization method proposed in this paper and a step-based learning rate, a further optimized loss function could be converged quickly. The algorithm proposed in this paper could effectively solve partial differential equations without boundary conditions such as DAE, which at the same time contains the first-order and the second-order partial derivatives, and constant terms. Simulation results show that compared with the original algorithm by Fast Fourier transform, this algorithm is more stable and accurate, which is significant for the antenna measurement method based on DAE.

Key words: methods: numerical – telescopes – waves – scattering

1. Introduction

The surface deformation of the antenna affects its performance and high frequency observation efficiency seriously, especially for large diameter antenna (Ruze 1966). Many mature schemes such as radio holography (Rahmat-Samii 1984; Morris et al. 1988; Baars et al. 2007), phase retrieval (Morris 1996; Yaccarino & Rahmat-Samii 1997) and photogrammetric (Subrahmanyan 2005) are widely employed to measure the deformation precisely so that the active surface system of the antenna could be controlled accurately to compensate the deformation (Wang et al. 2014). However, these methods are usually limited by elevation angles, measuring frequency, and signal-to-noise ratio (S/N) in practical measurements. As a result, the accuracy and speed of the measurement will be affected to some extent. The near-field measurement method to get the deformation only by scanning the amplitude of a planar grid has been proposed based on geometry optics (GO), which could be realized when the antenna is in any elevation angle, independent of frequency, and the scanning plane could be selected in reactive near-field region which has an ideal S/N (Huang et al. 2017). The core part of this method is the deformation-amplitude equation (DAE), which reveals the relation between the amplitude of the near-field plane and the surface deformation. However, DAE is a partial differential equation (PDE) which simultaneously includes the deformation term and its first-order and second-order terms. The traditional methods for solving PDE, such as the finite difference method (FDM) and the finite element method, are difficult to solve DAE.

With the development of computer science, more and more kinds of approximators are applied to engineering problems. Artificial neural network (ANN) has the property of universal approximation (Cybenko 1989). Let σ be any continuous sigmoidal function. Then finite sums of the form

$$G(x) = \sum_{j=1}^{N} \alpha_j \sigma(y_j^T x + \theta_j), \tag{1}$$

are dense in $C(I_n)$, where $y_j \in R^n$ and a_j, $\theta \in R$ are fixed. I_n denotes the n-dimensional unit cube, $[0,1]^n$. $C(I_n)$ denotes the space of continuous functions on I_n. In other words, given any $f \in C(I_n)$ and $\varepsilon > 0$, there is a sum, $G(x)$, of the above form, for which

$$|G(x) - f(x)| < \varepsilon, \quad \text{for all } x \in I_n. \tag{2}$$

The theorem above proves that for any continuous function in a complete space, there always exists a neural network with as few as a single hidden layer, which has arbitrarily small

difference from the objective function at any point. Therefore, the neural network method has strong approximation ability, which could theoretically achieve arbitrary precision when approximating a continuous multivariate function. The traditional methods for solving PDE are generally limited to regular domains such as rectangle and circle. The neural network method could solve PDEs in irregular domains, and even it does not need to modify the domain with complex geometry structure (Rudd & Ferrari 2015). Besides, with the increase of the number of discrete sampling points, the computational complexity of traditional PDE algorithms may increase exponentially. However, the calculation process of feedforward neural network is independent parallel computing, which means that the computational complexity of the algorithm increases linearly (Shirvany et al. 2009).

ANN has been widely used in engineering fields, such as predicting and solving turbulence models (Ling et al. 2016; Duraisamy et al. 2019), molecular dynamics models (Zhang et al. 2018). The core reason to use ANN is also that these engineering problems involve complex mathematical models including irregular boundary conditions (BCs) and high-order partial differential terms, which are difficult to be solved by traditional algorithms. van Milligen et al. (1995) proposed the use of ANN to solve a two-dimensional ideal magnetohydrodynamic plasma equilibrium without finite differences and coordinate transformations, which proved the technique is especially promising for solving complex PDE and straightforward to implement. The length factor method for solving boundary value problems is proposed using ANNs for irregular domain boundaries with mixed Dirichlet or Neumann BCs, which ingeniously reduces the error caused by irregular BCs (McFall & Mahan 2009). The deep Ritz method combines variational theory and ANN to get the numerical solution of the Poisson equation and its boundary conditions precisely (Yu et al. 2017).

In this paper, an ANN approximator is proposed to fit the deformation function based on the relation between the near-field amplitude and the surface deformation. The satisfactory and repeatable simulation results prove that ANN can be used as a conventional algorithm to solve DAE. One of the problems in solving DAE is that it needs a method which can approximate the deformation function locally at each discrete point because of the unknown boundary condition, so that the error caused by the boundary condition will not spread to the whole domain. Therefore, the length factor method is employed to make the trial solution satisfy the boundary conditions. When using ANN approximator whose training process is nearly a black box, most engineering applications have the problem of too long training time. A special method of preprocessing the training data for solving complex PDE is proposed in this paper, which effectively speeds up the convergence process of the algorithm. A global step-based learning rate is employed to reduce the training time and make the loss decline smoother during the training process.

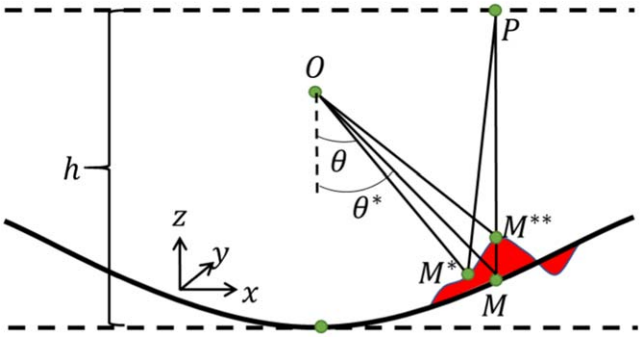


Figure 1. The geometry of a reflector antenna.

2. Mathematics

2.1. DAE Method

The core part of amplitude method is DAE which reveals the relationship between the surface deformation and nearfield amplitude. The geometry of a reflector antenna is shown in Figure 1.

The smooth parabola in the figure represents the ideal state of the antenna when there is no surface deformation. The wavy curve represents the surface deformation of the antenna. Point O is the feed, which is located at the focal point of the antenna where the coordinate is (0, 0, F) and F is the focal length. The dashed line in the top of the figure denotes a planar nearfield and P denotes a random point of this plane. The near-field plane is h meters from the origin of the coordinate. The corresponding reflection point of P on the ideal antenna surface and realistic antenna surface is M and M^*, respectively. The z-direction projection of M on the realistic antenna is M^{**}. The angles between the line segments OM and OM^* and the vertical direction are θ and θ^*.

The surface of ideal antenna, a rotating paraboloid, is given as

$$z_0 = (x^2 + y^2)/4F. \tag{3}$$

Let δ denote the normal component of deformation. The surface of deformed antenna can be expressed as

$$z^* = z_0 + \delta. \tag{4}$$

The electric field of the electromagnetic wave radiated by the feed which is reflected by ideal antenna and realistic antenna with deformation is denoted as E and E^* respectively. To get DAE, E and E^* need to be calculated, which can be expressed as Equations (5) and (6) based on geometrical optics theory, respectively

$$E(P) = D(M)E(M)\exp[-jk(d_{OM} + d_{MP})], \tag{5}$$

$$E^*(P) = D(M^*)E^*(M^*) \times \exp[-jk(d_{OM^*} + d_{M^*P})]. \tag{6}$$

Here $D(M)$ is the divergency coefficient of the ray-tube which connects point P and M and can be calculated by the law of conservation of energy. The j in the above formula denotes imaginary unit. The k denotes wavenumber. The d_{OM} denotes the Euclidean distance between point O and M. The d_{MP} has a similar meaning. According to the geometric properties of the paraboloid, all the rays emitted from focal point become parallel after being reflected by ideal antenna. Therefore, $D(M)$ equals 1 in Equation (5).

It can be shown that $D(M^*)$ can be approximately expressed as (Huang et al. 2017):

$$D(M^*) \approx \frac{1}{\sqrt{1 + G\nabla^2\delta + [(U + G_x)\delta_x + (V + G_y)\delta_y] + (U_x + V_y)\delta}}. \quad (7)$$

Here ∇^2 denotes Laplacian operator. The subscripts x and y means taking the derivative of x and y. The G, U and V in the above equation are expressed as:

$$G = 2F\frac{x^2 + y^2 - 4Fh}{x^2 + y^2 + 4F^2},$$
$$U = \frac{-4Fx(x^2 + y^2 - 4Fh)}{(x^2 + y^2 + 4F^2)^2},$$
$$V = \frac{-4Fy(x^2 + y^2 - 4Fh)}{(x^2 + y^2 + 4F^2)^2}. \quad (8)$$

According to Equations (5)–(8), the DAE is given as below (Huang et al. 2017)

$$\left(\frac{|E(P)|}{|E^*(P)|}\right)^2 \approx 1 + G\nabla^2\delta + [(U + G_x)\delta_x + (V + G_y)\delta_y] + (U_x + V_y)\delta. \quad (9)$$

The Fast Fourier transform (FFT) method is used in the original algorithm (Huang et al. 2017), which omitted $[(U + G_x)\delta_x + (V + G_y)\delta_y] + (U_x + V_y)\delta$ because it is much smaller than $1 + G\nabla^2\delta$. Therefore, the DAE becomes a Poisson equation, and $G\nabla^2\delta$ could be regarded as the convolution of the Laplace operator to the deformation. Then the deformation could be solved in the frequency domain because convolution in time domain is equivalent to multiplication in frequency domain.

2.2. Optimization Function and Trial Solution

Suppose that a second order differential equation can be expressed as:

$$G(\boldsymbol{x}, \psi(\boldsymbol{x}), \nabla\psi(\boldsymbol{x}), \nabla^2\psi(\boldsymbol{x})) = 0, \boldsymbol{x} \in D. \quad (10)$$

It is assumed that the boundary condition S of Equation (10) has been given. $D \subset R^n$ is the domain of definition. $\psi(\boldsymbol{x})$ is the solution to be solved. To solve the above problem, the domain and boundary of the equation must be discretized. Then Equation (10) can be written as:

$$G(\boldsymbol{x}_i, \psi(\boldsymbol{x}_i), \nabla\psi(\boldsymbol{x}_i), \nabla^2\psi(\boldsymbol{x}_i)) = 0, \forall \boldsymbol{x}_i \in \hat{D}. \quad (11H)$$

Here $\hat{D}$ denotes the discretization of D.

Now supposed $\psi_t(\boldsymbol{x}_i, \boldsymbol{p})$ is a trial solution, where $\boldsymbol{p}$ is the adjustment parameter. Then the above formula can be transformed into an optimization problem with constraints, which are the boundary conditions S, as follows:

$$\min \sum_{\boldsymbol{x}_i \in \hat{D}} (G(\boldsymbol{x}_i, \psi_t(\boldsymbol{x}_i, \boldsymbol{p}), \nabla\psi_t(\boldsymbol{x}_i, \boldsymbol{p}), \nabla^2\psi_t(\boldsymbol{x}_i, \boldsymbol{p}))). \quad (12)$$

In this way, we can try to find the numerical solution of the original differential Equation (10) by optimizing the adjustment parameters to find the minimum value of the function G on the premise of satisfying the constraint conditions.

In fact, it is naturally for us to associate the above optimization problem with artificial neural network. The adjustment parameters can be realized by the weight coefficient and deviation in each layer of the neural network. If we select an appropriate form of trial solution, the neural network could keep adjusting each parameter by a certain amount of training. Then the numerical solution of the differential equation can be obtained.

A feasible trial solution can be given as (McFall & Mahan 2009):

$$\psi_t(\boldsymbol{x}) = A(\boldsymbol{x}) + F(\boldsymbol{x}, N(\boldsymbol{x}, \boldsymbol{p})). \quad (13)$$

Here $N(\boldsymbol{x}, \boldsymbol{p})$ is a feedforward neural network stochastic objective function with multiple inputs and a single output, containing the adjustment parameter $\boldsymbol{p}$. It is supposed that function F is always zero on the boundary. Function $A(\boldsymbol{x})$ satisfies the boundary conditions, which do not contain adjustment parameters. Noted that through the form of the trial solution in Equation (13), it always satisfies the boundary condition regardless of the output value of the neural network. Therefore, the original optimization problem with constraints has been transformed into an unconstrained optimization problem.

2.3. Network Parameters and Gradient Calculation

For the optimization function G in Equation (12), the optimizing process is the training process of the neural network. In this process, the value of function G, which is the loss of the neural network, decreases continuously and tends to zero. To calculate the loss, not only the output value of the network, but also the first-order and second-order partial derivatives of the network output to each input are needed. At the same time, in the process of loss reduction, we need to calculate not only the gradients of neural network output to network parameters such as weights and deviations in each layer, but also the gradients of these first-order and second-order partial derivatives to network parameters.

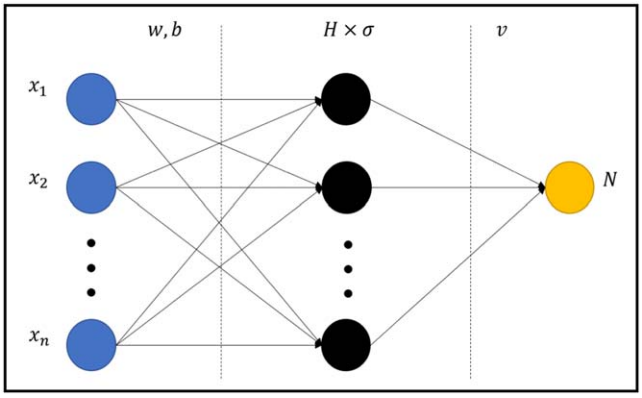


Figure 2. The geometry of a reflector antenna.

Suppose $N(\boldsymbol{x}, \boldsymbol{p})$ has a single hidden layer, as shown in Figure 2, which contains h active neurons, and the output layer is a linear combination of hidden layers. The activation function is sigmoid function, which is given as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{14}$$

The first-order and second-order partial derivatives of the sigmoid function to the inputs can be expressed as Equations (15) and (16), respectively

$$\frac{d}{dx}\sigma(x) = \sigma(1 - \sigma), \tag{15}$$

$$\left(\frac{d}{dx}\right)^2 \sigma(x) = \sigma(1 - \sigma)(1 - 2\sigma). \tag{16}$$

First, for an input vector $\boldsymbol{x} = (x_1, x_2, x_3, \cdots x_n)$, the output of the neural network $N(\boldsymbol{x}, \boldsymbol{p})$ can be expressed as:

$$N = \sum_{i=1}^{h} v_i \sigma(z_i). \tag{17}$$

Here v_i is the weight of the ith neuron in the hidden layer to the output, and z_i is given as:

$$z_i = \sum_{j=1}^{n} w_{ij} x_j + b_i. \tag{18}$$

Here w_{ij} is the weight of the jth neuron in the input layer to the ith neuron in the hidden layer, and b_i is the deviation of the ith neuron in the hidden layer. Therefore, from Equations (17) and (18), the partial derivatives of the output N to each input of the network can be expressed as:

$$\frac{\partial^k N}{\partial x_j^k} = \sum_{i=1}^{H} v_i w_{ij}^k \sigma_i^{(k)}. \tag{19}$$

Here $\sigma_i^{(k)}$ denotes the kth order partial derivatives of the sigmoid function σ_i.

2.4. ANN Algorithm for DAE

It is obvious that DAE, namely Equation (9), is a PDE including constant term, deformation term, the first-order derivative term and the second-order derivative term. Therefore, it is easy to get the amplitudes if the deformation is given, but on the contrary, it is hard to solve the deformation directly by traditional algorithms. However, this problem can be transformed to solve the loss optimization by means of ANN method, where the loss function is constructed as:

$$\begin{aligned} \text{loss} = \{1 &+ G\nabla^2\delta + [(U + G_x)\delta_x \\ &+ (V + G_y)\delta_y] + (U_x + V_y)\delta - EA\}^2. \end{aligned} \tag{20}$$

Here $EA = \left(\frac{|E|}{|E^*|}\right)^2$ is obtained by near-field measurement or simulation. G, U, V are all discrete matrices related to the geometric size of the antenna reflector, the near-field measurement distance and the number of sampling points. Then the solution of DAE is equivalent to a deformation δ which satisfies the minimization of loss in Equation (20).

When using the ANN method to solve the deformation, it is necessary to construct a trial solution first, and a boundary condition is needed according to Equation (13). However, the deformation on the boundary of the reflector surface is never a prior information in actual measurement so that we need to assume a boundary condition. First, the core reason to assume a boundary condition is that the algorithm is difficult to converge when solving partial differential equations by ANN without boundary conditions. Therefore, the trial solution must be constrained by boundary conditions. Second, the essence of the ANN method to solve deformation is loss minimization and function fitting. Compared with traditional methods such as the FDM, the ANN method will lead to a relatively small overall migration error when using inaccurate boundary conditions. In this paper, the subsequent simulation will prove that the error of boundary condition will only affect the boundary and its adjacent region when using neural network method to solve the deformation.

We first assume that the deformation boundary of the antenna surface is zero. The trial solution of the surface deformation δ can be written as:

$$\begin{aligned} \delta_t(\boldsymbol{x}) &= A(\boldsymbol{x}) + F(\boldsymbol{x}, N(\boldsymbol{x}, \boldsymbol{p})) \\ &= L(\boldsymbol{x}) \cdot N(\boldsymbol{x}, \boldsymbol{p}). \end{aligned} \tag{21}$$

Here $A(\boldsymbol{x}) = 0$. $L(\boldsymbol{x})$ is boundary function which makes the trial solution always zero on the boundary. Because the reflector is a rotating paraboloid, whose boundary is circular, we have

$$L(x, y) = x^2 + y^2 - \left(\frac{D}{2}\right)^2. \tag{22}$$

Obviously, because the inputs are the Cartesian coordinates of the sampling points, the first-order and second-order partial

Table 1
Simulation Conditions

Diameter of the Antenna D/m	Focal Length F/m	Height of Scanning Plane h/m	Frequency f/GHz
110	33	33	0.3
Source	**Taper angle**	**Number of field points($x * y$)**	**Magnitude of δ (M_δ/mm)**
Gaussian feed	79.°61	$256 * 256$	1.2

derivatives of the trial solution $\delta_t(\boldsymbol{x})$ to the inputs could be shown as:

$$\begin{aligned}\delta_x &= 2x \cdot N + L \cdot N_x,\\ \delta_y &= 2y \cdot N + L \cdot N_y,\\ \delta_{xx} &= 2N + 4x \cdot N_x + L \cdot N_{xx},\\ \delta_{yy} &= 2N + 4y \cdot N_y + L \cdot N_{yy}.\end{aligned} \tag{23}$$

Here N_x, N_y, N_{xx}, N_{yy} denote the first-order and second-order partial derivatives of the net output to the inputs, respectively, which are given as Equation (24)

$$\begin{aligned}N_x &= (\sigma \cdot (1-\sigma) \cdot w_{[0]}) \times v,\\ N_{xx} &= (\sigma \cdot (1-\sigma) \cdot (1-2\sigma) \cdot w_{[0]} \cdot w_{[0]}) \times v,\\ N_y &= (\sigma \cdot (1-\sigma) \cdot w_{[1]}) \times v,\\ N_{yy} &= (\sigma \cdot (1-\sigma) \cdot (1-2\sigma) \cdot w_{[1]} \cdot w_{[1]}) \times v.\end{aligned} \tag{24}$$

Once each term in Equation (20) is expressible, the network is ready to start training.

3. Simulation

3.1. Simulation Conditions

Let $R_1 = U + G_x$, $R_2 = V + G_y$, $H = U_x + V_y$, $FA = EA - 1$, and Equation (20) can be changed to Equation (21) as below

$$\text{loss} = [G\nabla^2\delta + (R_1\delta_x + R_2\delta_y) + H\delta - FA]^2. \tag{25}$$

According to Section 2.1, G, H, R_1, R_2 can be calculated directly once the height of scanning plane is determined. To get FA, it should first simulate the amplitudes $|E|$ and $|E^*|$ in Equation (9) by geometry optics (GO) method, respectively. The conditions of simulation are listed as Table 1.

Design the global smooth deformation on the main reflector surface of the antenna as shown as Equation (26) and Figure 3(a). G, H, R_1, R_2 are given as Figures 3(c)–(f), respectively

$$\delta = \frac{\sin\left(\frac{F}{D}\sqrt{x^2+y^2+\frac{F}{2}}\right)}{50F} \frac{1}{1+e^{[-0.6(0.4D-\sqrt{x^2+y^2})]}} \tag{26}$$

where F denotes the focal length and D denotes the diameter of the antenna.

Finally, after simulation we get the near-field plane amplitudes of the antenna without deformation and with deformation, $|E|$ and $|E^*|$, and FA is calculated as Figure 3(b). All the coefficients in the loss function have been obtained so far. They will be fed to the ANN to start training.

3.2. Training Flow

This section mainly describes the process to realize solving DAE by means of ANN. Adam optimizer is employed to train the network parameters, which uses both the first-order and the second-order momentum vector. Using large scale data sets, Kingma & Ba (2014) proved that Adam can effectively solve the actual deep learning problem, and the convergence efficiency is significantly better than other optimization algorithms.

Based on the Adam optimization algorithm, the main steps of the method to solve DAE are listed below and shown in Figure 4.

(a) Define the number of discrete sampling points in the solving area and divide the meshing grid. The coordinates of each point are fed to the neural network as the inputs of the network.
(b) Define the network parameters w, b, v and conduct the forward calculation. In this paper, a single hidden layer with a width of 100–200 neurons is used. Based on an AI framework, the computational graph is then to be constructed.
(c) According to Equation (24), define the first-order and second-order partial derivatives of output N to the inputs by using the characteristics of sigmoid activation function.
(d) Define the boundary function L and its first and second partial derivatives outside the calculation diagram. Import the coefficients G, H, R_1, R_2. After being transformed into corresponding shapes, they are fed to the neural network as the inputs to calculate the loss value.
(e) Start training with appropriate hyperparameters, a step-based global learning rate α is used in this paper.
(f) Calculate the gradient and propagate back to update the network parameters w, b, v, or use an optimizer such as Adam for automatic training, as shown as Figure 5. This step is an iterative process and it is also the core of the algorithm. The Adam optimizer pseudo-code and some default settings of hyperparameters used in this paper is given by Figure 5.

By setting the number of iteration steps or error threshold of the loss to end the training, the solution could be obtained once the network parameters θ_t converge.

3.3. Data Preprocessing

The data preprocessing of ANN mainly has five parts including data importing, data shape transformation, data set

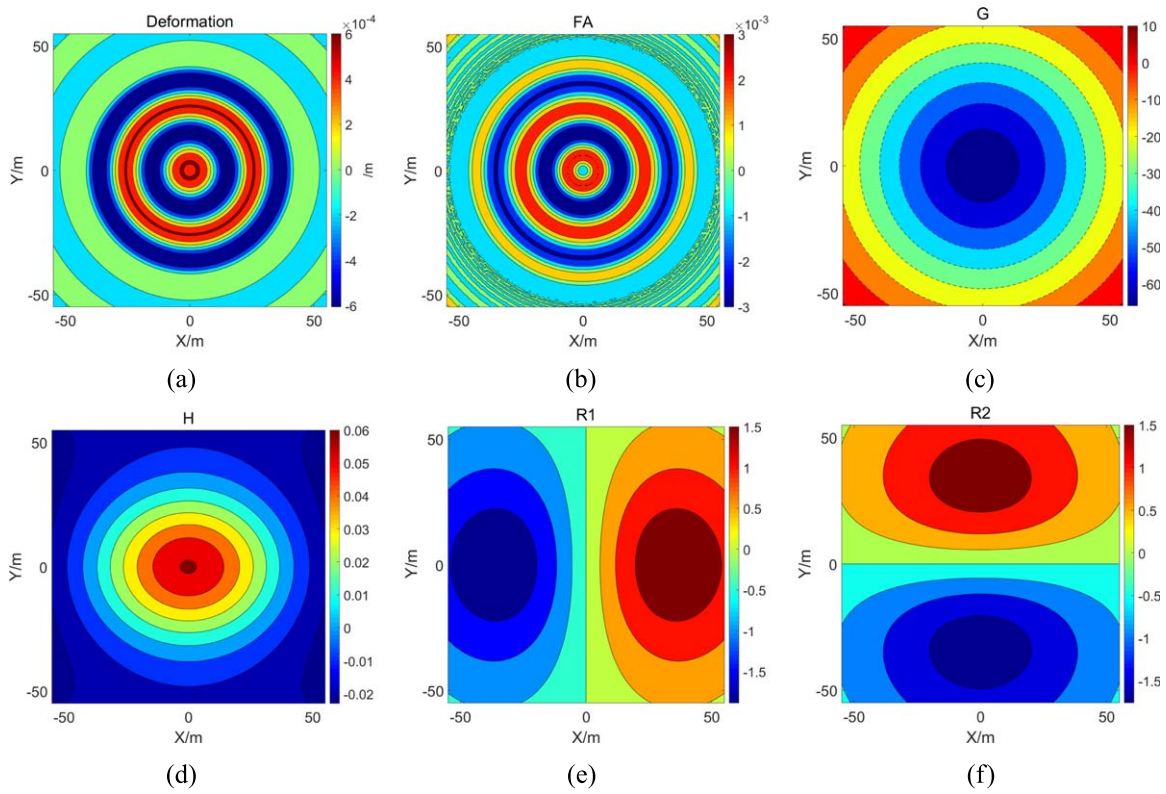


Figure 3. The assumed deformation 1, coefficients in loss function and the near-field simulation results. (a) The first assumed deformation in this paper. (b) The FA simulation result in near-field. (c)–(f) The values of G, H, R_1, R_2.

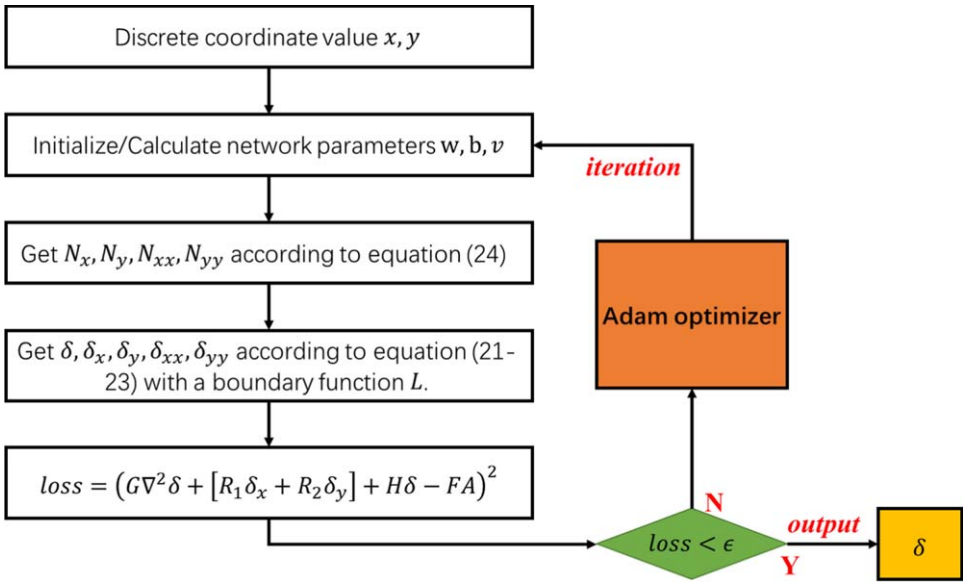


Figure 4. The flow chart of the ANN algorithm to solve the deformation.

Algorithm: Adam optimizer is employed to solve the DAE. See section2 for detail.

Default settings: $\alpha = 0.2 \times 0.96^{\frac{t}{50}}, n = 1000, \mu = 0.9, \beta = 0.99$

Require: α: Global learning rate
Require: $\mu, \beta \in [0,1)$: Exponential decay rates for the first-order and second-order moment vectors
Require: $f(\theta)$: Stochastic optimization function, including w, b, v
namely the network $N(\vec{p})$ with the adjustment parameters $\vec{p}$ described in section 2.2
Require: θ_0: Initial network parameter vector
$t \leftarrow 0$: Initialize timestep
$m_0 \leftarrow 0$: Initialize the first-order moment vector
$n_0 \leftarrow 0$: Initialize the second-order moment vector
Require: n: Training steps or k: threshold
While $loss > k$ or $t < n$, **do**
$t < t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$: Get gradients of the network function to the network parameters at each t
$m_t \leftarrow \mu \cdot m_{t-1} + (1-\mu) \cdot g_t$: Update biased first-order moment vector
$n_t \leftarrow \beta \cdot n_{t-1} + (1-\beta) \cdot g_t^2$: Update biased second-order moment vector
$\theta_t \leftarrow \theta_{t-1} - \frac{\eta}{\sqrt{n_t + \varepsilon}} \cdot m_t$: Update network parameters
End while
Return θ_t

Figure 5. The main loop and the default settings of the Adam algorithm.

dividing, data normalization and data encapsulation. This section mainly discusses the data normalization method for solving DAE by ANN.

Different kinds of the original training data have different measurement units and the numerical distribution ranges differ a lot, which will make the training of neural network difficult to converge. Therefore, we must preprocess the sampling data first, and normalize the range of each kind of data to the same level and eliminate the correlation between different groups so that to obtain the ideal results.

The input data of the neural network are all the coordinates of the sampling points on the *X*–*Y* plane parallel to the aperture surface, which ranges from −55 (m) to 55 (m). However, FA ranges from -3×10^{-3} to 3×10^{-3}, and the supposed deformation ranges from -6×10^{-4} (m) to 6×10^{-4} (m). Obviously, each array of training data has a huge numerical difference from each other. We have tried to solve DAE using the original data without normalization, which results in a huge initial loss. During 200,000 steps of training in nearly 7h, the loss value is always fluctuating and the decline is slow, as shown in Figure 6(a). It is obviously that we failed to get the accurate solution.

The normalization method used first is that suppose $x' = \frac{x}{55} \in [-1, 1]$, $y' = \frac{y}{55} \in [-1, 1]$, $FA' = \frac{FA}{0.003} \in [-1, 1]$, and then the loss function becomes

$$\text{loss} = \left(\frac{G\nabla^2\delta'}{55 \times 55} + \frac{R_1\delta'_x + R_2\delta'_y}{55} + H\delta' - FA' \right)^2. \quad (27)$$

Here $\delta' = \delta/0.003 \in [-0.2, 0.2]$ is the solution to be solved of Equation (27), and the deformation δ could be easily obtained by denormalization from δ' after training. By means of this kind of normalization, the network parameters will converge fast, as shown in Figure 6(b). After data normalization, the decline speed of loss is significantly improved, and the fluctuation of the loss curve is reduced. Although we could already solve the deformation in this way, this algorithm still cannot be applied to real-time compensation for the deformation of large reflector antenna, because its operation time is not fast enough to meet the requirement of a real-time measurement.

It is observed that the numerical change of the input data can be compensated by adding adjustment coefficients in the loss function. Now consider to further adjust the normalization method. The main idea is to enlarge the numerical range of the deformation terms and reduce the constant terms so that the impact of the network output on the loss function is increased.

Assume $x' = \frac{x}{55} \in [-1, 1]$, $y' = \frac{y}{55} \in [-1, 1]$, $FA' = \frac{FA}{0.015} \in [-0.2, 0.2]$, and the loss function is changed into

$$\text{loss} = \left(\frac{G\nabla^2\delta'}{55 \times 55 \times 100} + \frac{R_1\delta'_x + R_2\delta'_y}{55 \times 100} + \frac{H\delta'}{100} - FA' \right)^2. \quad (28)$$

Here $\delta' = \delta/(1.5 \times 10^{-4}) \in [-4, 4]$. The loss curve is shown in Figure 7. Compared to the former curves, the initial loss is

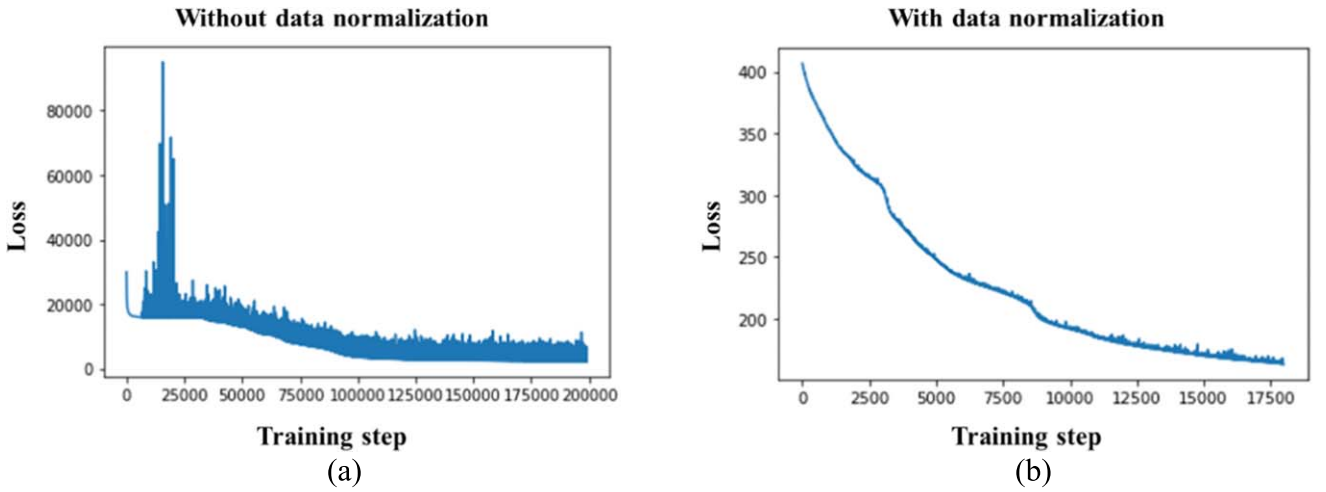


Figure 6. The loss curve of the training without and with data normalization.

small and we can get the solution only by 500–1000 steps of training with 25–41 s.

3.4. Simulation Results

Based on the algorithm described in Section 3.2 and the data normalization method as Equation (3.4), we solved the deformation from DAE and compared the results δ_r with the assumed deformation δ. To make a quantitative comparison, root-mean-square error (rms) and relative root-mean-square error (RRMS) are introduced as

$$\mathrm{rms} = \sqrt{\frac{\sum_m \sum_n [\delta_r(m, n) - \delta(m, n)]^2}{N^2}}, \tag{29}$$

where $N = 256$ denotes the number of discrete points in the x or y direction

$$\mathrm{RRMS} = \frac{\mathrm{rms}}{M_\delta}. \tag{30}$$

Here M_δ denotes the magnitude of δ.

As is shown in Figure 8(b), the error of the solution by ANN is RRMS = 9.76%. Figure 8(a) shows the comparison of the solving result, dotted line in red, and the original supposed deformation, the full line in black, on the cutline of $X = 0$. We perform 1000 steps of training on GPU, RTX2060, with the training time of 40 s. In fact, we have already obtained an accurate solution by 500 steps of training in 24.59 s. The training messages are listed in Table 2. In engineering practice, one of the disadvantages of using the ANN method to solve PDE is that the fitting time is too long. However, by using the normalization method in Section 3.3 of this paper, the time-consuming of about 30 s is acceptable.

If we use the original algorithm by the FFT method described in Section 2.1 to solve the deformation 1, the accuracy is excellent, which has an error of RRMS = 9.13%.

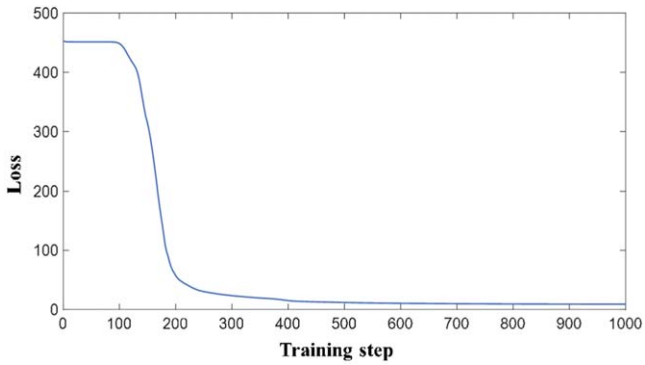


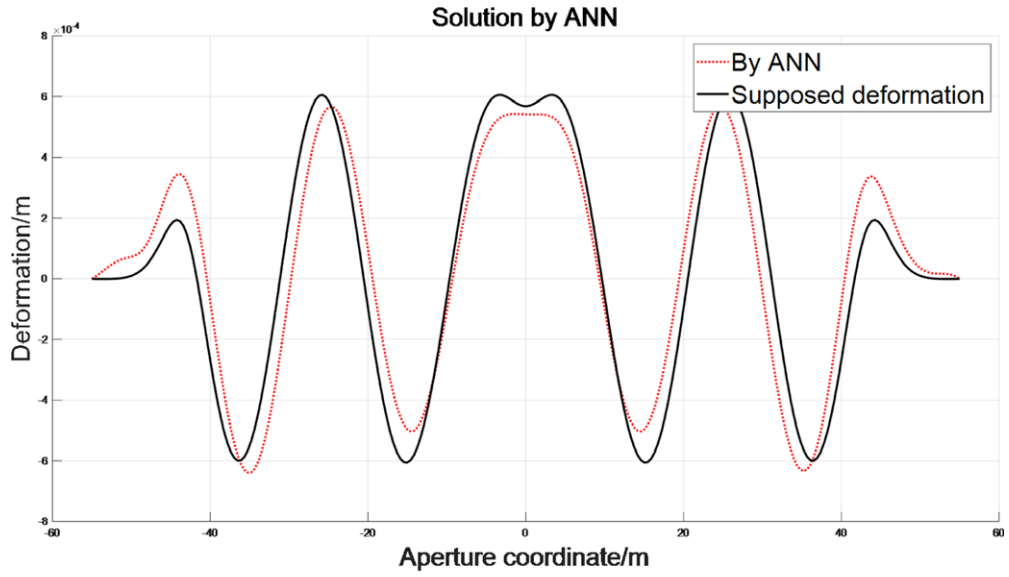
Figure 7. The loss curve of the training with data normalization as Equation (28).

However, each point on the boundary of the assumed deformation 1 as Equation (26) is zero, which is unrealistic for the real antenna. Now suppose deformation 2 is expressed as Equation (31), which is the result of deformation 1 shifted in the X–Y plane

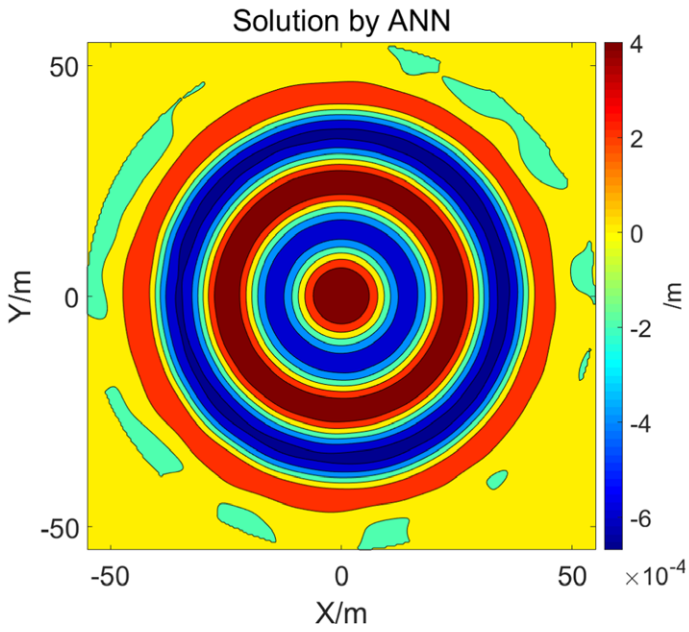
$$\delta = \frac{\sin\left(\frac{F}{D}\sqrt{(x+25)^2 + (y+25)^2 + \frac{F}{2}}\right)}{50F} \times \frac{1}{1 + e^{[-0.6(0.4D - \sqrt{(x+25)^2+(y+25)^2})]}}. \tag{31}$$

We keep the electromagnetic simulation conditions unchanged, and use the totally same algorithm and the same network parameters. The supposed deformation 2 and the final solution by ANN are shown as Figures 9(a) and (b), respectively.

The error of the solution for deformation 2 is RRMS = 9.11%, as shown in Table 3, which keeps the same level from the accuracy of the solution for deformation 1.



(a)



(b)

Figure 8. The solution by ANN for the assumed deformation 1.

Table 2
The Results of Solving Deformation 1 by ANN

Deformation 1	$M_\delta = 1.2$ mm	Adam Optimizer	$256 * 256$	$f = 0.3$ GHz		
Total steps	Network structure	Initial learning rate	Decay steps	Decay rate	Time/s(GPU)	RRMS/%
500	2, 200 * 1, 1	0.2	50	0.96	24.59	9.88
1000	2, 200 * 1, 1	0.2	50	0.96	40.90	9.76

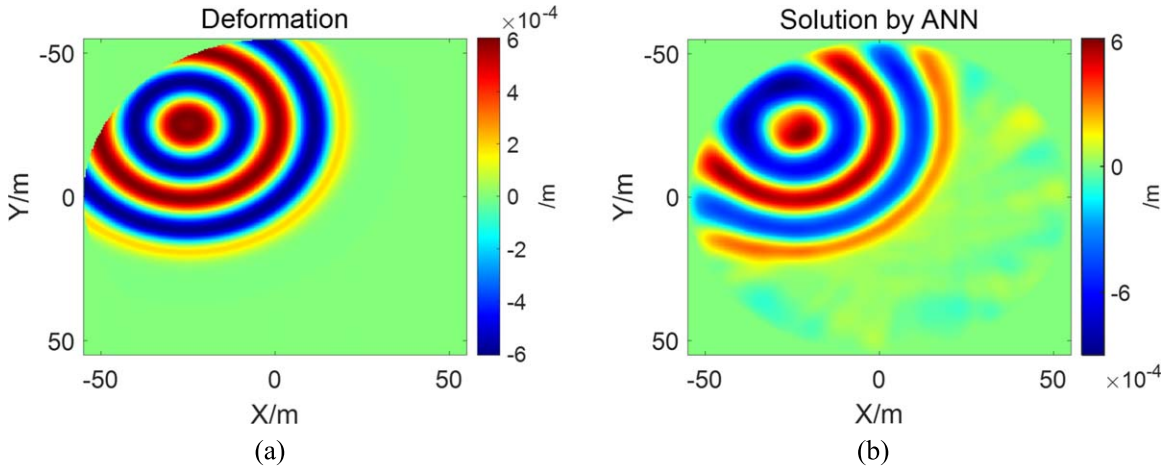


Figure 9. The assumed deformation 2 and the solution by ANN for it.

Table 3
The Results of Solving Deformation 2 by ANN

Deformation 2 $M_\delta = 1.2$ mm Adam Optimizer $256 * 256$ $f = 0.3$ GHz						
Total steps	Network structure	Initial learning rate	Decay steps	Decay rate	Time/s(GPU)	RRMS/%
500	$2, 200 * 1, 1$	0.2	50	0.96	23.91	9.25
1000	$2, 200 * 1, 1$	0.2	50	0.96	40.33	9.11

Note that in the process of solving deformation 2, we still assume that the deformation boundary of the antenna surface is zero, which is not consistent with the actual deformation 2. However, the accuracy of the final solution is not significantly reduced, which proves the feasibility of the proposed algorithm for solving DAE with unknown boundary conditions. Figure 10(a) shows the error between the calculated deformation 2 and the setting value. It is obvious that the error mainly comes from the boundary due to the fact that the length factor we set is all zero on the boundary. The error inside the boundary is not increased because the ANN algorithm is a local approximation method.

As a comparison, the solution of the original algorithm by FFT method has an error of RRMS = 42.77%, as shown as Figure 10(b), which means that it totally fails to obtain the deformation. From the scale mark in the figure, we can see that the solution by FFT algorithm has an overall deviation, which is caused by the boundary deformation. The simulation results for deformation 2 prove the instability of the original algorithm. In particular, when the boundary of the antenna reflector is deformed seriously, the solution contains a huge error. On the contrary, the algorithm based on ANN proposed in this paper is stable enough due to the property of local approximation.

4. Summary

Based on DAE, a local approximation algorithm to solve the surface deformation of antenna reflector using ANN as the stochastic objective function. First, Length factor method is employed to construct the trial solution of the objective deformation, which converts solving the DAE into an unconstrained optimization problem because the trial solution always satisfies the boundary conditions. Simulation results show that this method can minimize the impact of the unknown boundary conditions on the overall solution accuracy. As is a fact, the boundary of the deformation is hard to get in advance. We use a neural network with a single hidden layer and sigmoid activation function. The advantage of sigmoid function is to make the partial derivatives of the stochastic objective function to each network input to be expressed easily. As a result, it is convenient to define each term in DAE and the design of loss function could be done. The Adam approximator is used to train the network parameters, which adds the first-order and second-order momentum factors besides the global learning rate we have set. This optimization method intelligently adjusts the gradient descent direction and speed. In addition, we compare the effects of different data normalization methods, and finally select the optimal data normalization

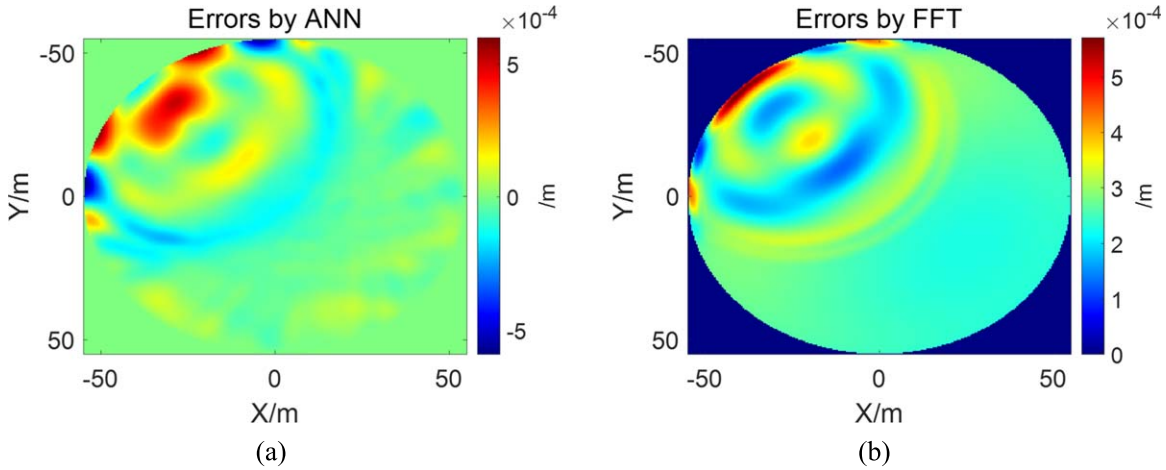


Figure 10. The errors for solving deformation 2 by ANN and FFT, respectively.

method based on changing the design of the loss function. Combined with a step-based global learning rate, the overall convergence time of the algorithm is greatly reduced, and the solution accuracy of the deformation is improved.

The main contributions of this paper are as follows.

(1) A local approximation algorithm based on ANN is proposed to solve PDE with unknown boundary containing constant term, zero-order term, the first-order derivative term and the second-order derivative term simultaneously, which is hard to be solved by traditional algorithms. It is applied to the engineering model of antenna deformation measurement, which effectively solves the problem of low accuracy of the original algorithm when the boundary deformation is large. As a result, the RRMS of the solution could be stabilized within 10% and the error by the original algorithm may reach the level of RRMS $= 42.77\%$ in some cases.

(2) A data preprocessing method for PDE is proposed so that the accuracy of the solution is improved and the convergence speed of the ANN reach a satisfactory level. In this paper, it only needs 500 steps of training to get the solution with 256×256 sampling points in 25–41 s, which is significant for the actual antenna real-time measurement.

Acknowledgments

This work is supported by the National Key Research and Development Program of China: Research on Key Technologies of real-time shape control and ultra wideband pulsar signal processing for large aperture radio telescope, 2021YFC2203501, and the National Natural Science Foundation of China under project U1931137.

References

Baars, J., Lucas, R., Mangum, J., & Lopez-Perez, J. 2007, IAPM, 49, 24
Cybenko, G. 1989, Math. Control Signals Syst., 2, 303
Duraisamy, K., Iaccarino, G., & Xiao, H. 2019, AnRFM, 51, 357
Huang, J., Jin, H., Ye, Q., & Meng, G. 2017, OExpr, 25, 24346
Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
Ling, J., Kurzawski, A., & Templeton, J. 2016, JFM, 807, 155
McFall, K. S., & Mahan, J. R. 2009, ITNN, 20, 1221
Morris, D. 1996, IEEEP, 143, 298
Morris, D., Baars, J., Hein, H., Steppe, H., & Thum, C. 1988, A&A, 203, 399
Rahmat-Samii, Y. 1984, RaSc, 19, 1205
Rudd, K., & Ferrari, S. 2015, Neurocomputing, 155, 277
Ruze, J. 1966, Proc. IEEE, 54, 633
Shirvany, Y., Hayati, M., & Moradian, R. 2009, Applied Soft Computing, 9, 20
Subrahmanyan, R. 2005, ITAP, 53, 2590
van Milligen, B. P., Tribaldos, V., & Jiménez, J. 1995, PhRvL, 75, 3594
Wang, W., Wang, C., Duan, B., Leng, G., & Li, X. 2014, IET Microw. Antennas Propag., 8, 158
Yaccarino, R., & Rahmat-Samii, Y. 1997, in IEEE Antennas and Propagation Society Int. Symp., 2, 1472
Yu, B., & E, W. 2017, Commun. Math. Stat., 6, 1
Zhang, L., Han, J., Wang, H., Car, R., & Weinan, E. 2018, PhRvL, 120, 143001