Arabian Journal for Science and Engineering (2021) 46:11797–11818
https://doi.org/10.1007/s13369-021-05702-8

RESEARCH ARTICLE-MECHANICAL ENGINEERING

Implementation of Stochastic Optimization Method-Assisted Radial Basis Neural Network for Transport Phenomenon in Non-Newtonian Third-Grade Fluids: Assessment of Five Optimization Tools

Vijay K. Mishra[1] · Sumanta Chaudhuri[1]

Received: 30 December 2020 / Accepted: 4 May 2021 / Published online: 20 May 2021

Abstract

Implementation of radial basis neural network is demonstrated by considering a test case of non-Newtonian third-grade fluid flow and heat transfer through two parallel plates. Five commonly used stochastic optimization methods: genetic algorithm, global search algorithm, multiple starting point algorithm, simulated annealing algorithm, and pattern search algorithm, are employed to optimize the RBNN. Flow of a non-Newtonian third-grade fluid through two parallel plates, subjected to uniform heat flux, is considered. At first, governing equations, describing the flow and heat transfer problem, are solved by the least-square method, a semi-analytical tool. The velocity and the temperature profiles are obtained for different values of third-grade fluid parameter 'A', which are then used for training different stochastic optimization method-assisted RBNNs (SOMARBNNs). For proper functioning of RBNN, a suitable value for an important attribute called 'spread' is required. Deciding the value for 'spread' requires experience and knowledge of neural networks. The present approach makes the selection of proper value of 'spread' very easy, and beginners can use the RBNN for problem-solving. With the help of different stochastic optimization methods, the value of spread for the RBNN is determined. Once all SOMARBNNs are trained, the temperature profile and the corresponding third-grade fluid parameter 'A' are obtained as output, corresponding to any new velocity profile fed as input. Further, the data for training are perturbed by different levels of noise, and different SOMARBNNs are successfully employed to get the output. The performance evaluation of different SOMARBNNs is carried out in terms of CPU time and error in result. The results indicate that PSAARBNN is better than other SOMARBNNs, as it is able to generate results with high accuracy for both low noise data and high noise data. Moreover, the CPU time requirement by PSAARBNN is lowest.

Keywords Non-Newtonian fluid · Least square method · Radial basis neural network · Stochastic optimization methods · Parameter retrieval

List of Symbols

A	Third-grade fluid parameter
Ac	Cross-sectional area [m^2]
$A_1, A_2, A_3, \ldots A_n$	Kinematic tensor
a_0, a_2, a_4, a_6, a_8	Constants
an	Layer output vector in RBNN
Br	Brinkman number
$b_0, b_2, b_4,$ b_6, b_8, b_{10}, b_{12}	Constants
bn	Bias in RBNN
C_P	Specific heat at constant pressure [J/kg.K]
$c_1, c_2, \ldots$	Constants
c_i	Ith constant
D	Differential operator
D/Dt	Material derivative
f	Body force per unit volume
g	Function
h	Half depth of channel [m]
IWn^1	Hidden layer weight matrix in RBNN
k_{th}	Thermal conductivity of the fluid [W/m K]
L	Length of the channel [m]
LWn^2	Output layer weight matrix in RBNN
l_1, l_2	Constants
N	Non-dimensional pressure gradient

✉ Vijay K. Mishra
mishra.vdm@gmail.com

[1] School of Mechanical Engineering, Kalinga Institute of Industrial Technology (KIIT) DU, Bhubaneswar 751024, India

Nu	Nusselt number
p^*	Dimensional pressure [N/m^2]
q	Heat flux ratio
q_1, q_2	Heat fluxes at lower and upper walls [W/m^2]
R	Residual
Rn	Number of observation points along y* direction (=201)
S	Sum of square of residual
sn	Size of matrix in RANN
T^*	Dimensional temperature [K]
T^*_m	Bulk mean temperature [K]
$T^*_{w_l}$	Temperature of the lower wall [K]
u	Non-dimensional velocity along axial direction
u_N	Non-dimensional velocity for Newtonian fluid
u^*	Dimensional velocity along axial direction [m/s]
u_0	Average velocity [m/s]
V^*	Velocity vector [m/s]
$v, \tilde{v}$	Functions [m/s]
x, y, z	Non-dimensional coordinate
$x*, y*, z*$	Dimensional coordinates [m]
Xn_o, Xn_1	Central point in PSA indifferent iterations
xn_i	Neuron input
w_i	Ith weight function
Wn_i	Neuron weight

Abbreviations

mse	Mean squared error
ANN	Artificial neural network
RBNN	Radial basis neural network
GA	Genetic algorithm
GAARBNN	Genetic algorithm-assisted RBNN
GSA	Global search algorithm
GSAARBNN	Global search algorithm-assisted RBNN
MSPA	Multiple starting point algorithm
MSPAARBNN	Multiple starting point algorithm-assisted RBNN
OF	Objective function
SAA	Simulated annealing algorithm
SAAARBNN	Simulated annealing algorithm-assisted RBNN
PSA	Pattern search algorithm
PSAARBNN	Pattern search algorithm-assisted RBNN
SOM	Stochastic optimization method
SOMARBNN	Stochastic optimization method-assisted radial basis neural network
TGF	Third-grade fluid

Greek symbols

α_1, α_2	Material constants
β	Constant
$\beta_1, \beta_2, \beta_3, \ldots$	Material constants
ρ	Density of the fluid [kg/m^3]
μ	Dynamic viscosity of the fluid [N s/m^2]
θ	Non-dimensional temperature
θ_N	Non-dimensional temperature for Newtonian fluid
$\tilde{\theta}$	Temperature obtained from SOMARBNNs
Φi	Base function
τ	Stress

Superscript

1, 2	Hidden layer and output layer variables

1 Introduction

Artificial neural network (ANN) can prove to be a very effective statistical tool for handling variety of situations such as (1) absence of mathematical relations for complicated physics, (2) fast and time-effective approach to get the result, (3) use of noisy data (obtained from analytical/numerical/experimental method). One of the very powerful types of neural network, used to solve a wide range of practical problems, is the radial basis network. Radial basis network offers various advantages over perceptron types such as simple to use and better capability to handle noisy data for medium-sized data set [1, 2]. Because of these advantages, radial basis network has been employed by numerous researchers in a wide spectrum of problems of engineering and industrial applications.

Stochastic optimization methods [3, 4] are very powerful optimization methods with capability to handle optimization problems with local minimum traps. Most commonly used stochastic optimization methods (SOMs) are: genetic algorithm (GA) [5], global search algorithm (GSA) [6], multiple starting point algorithm (MSPA) [7], simulated annealing algorithm (SAA) [8], and pattern search algorithm (PSA) [9].

For design and optimization of various systems, inverse engineering has emerged as a new field in this regard [10, 11]. In the inverse engineering problems, independent variables are computed from the knowledge of dependent variables. Or, we can say that from the end product, we have to trace back the governing equations of the system. Inverse engineering proves to be very effective in situation such as [12] measurement of properties is a costly affair or not possible due to inaccessibility of location (re-entry of vehicles), or chances of perturbation of flow pattern due to intrusion of measuring probes. In the field of inverse engineering, ANN-based optimization techniques play a key role in the final analysis.

In the present study, implementation of radial basis neural network (RBNN) has been demonstrated by choosing a problem of engineering applications involving flow and heat transfer of a non-Newtonian third-grade fluid. A variety of engineering equipment such as mixer, heat exchanger and pumps handles different kinds of fluids such as slurry [13], polymers [14], medicines [15], and pastes [16] in industries like mining applications, polymer processing, etc. Flow of fluid through these equipments is not governed by a simple linear stress–strain rate model of Newtonian fluids. Instead, in many cases, a nonlinear stress–strain rate relation is better-suited to describe the intricacies associated with the flow of these substances, classified as non-Newtonian fluids.

A non-Newtonian fluid exhibits strange behavior such as shear thinning/thickening and elastic nature, which are not displayed by Newtonian fluids. But a mathematical modeling of problems involving non-Newtonian fluid is a formidable task. Over the years, several mathematical models such as power law model, visco-elastic model, visco-plastic model, and Rivlin–Erickson model of grade three (also known as third-grade fluids) are put forward by the researchers to describe the non-Newtonian fluids. Third-grade fluid model has been employed by numerous researchers for modeling the flow of polymer, slurry, lubricants. In the present study, flow and heat transfer of a non-Newtonian third-grade fluid has been chosen as a test case for the demonstration of implementation process of RBNN.

Flow of a non-Newtonian third-grade fluid through rectangular parallel plates, subjected to uniform heat fluxes, is considered. First, governing equations of the problem are solved by employing LSM [17]. Even though LSM approach is comparatively simple to apply, yet it takes significant time to find the solution for small alterations such as new type of boundary conditions, changes in thermo-physical properties of fluid and changes in geometry. One pair of velocity profile and temperature profile is computed for one value of third-grade fluid parameter (A). This process is repeated multiple times to generate 26 pairs for 26 different values of ‘A’. This constituted the first part of the problem.

In the second part, RBNN is explored to solve the energy equation, as well as to retrieve the value of A, corresponding to a given velocity profile. So, the present approach intends to two problems simultaneously: (a) to solve the direct problem to find the temperature profile and (b) to apply the inverse method to find TGF parameter ‘A’. To apply RBNN in the problem, one important parameter is required, called spread. Five different SOMs are used, one by one to compute the value of spread (required for RBNN). The data (velocity profile: as input, temperature profile & A: as target) generated by LSM are used to train RBNN. This leads to the development of stochastic optimization method-assisted radial basis neural network (SOMARBNN).

Once the SOMARBNNs are developed, a new velocity profile (not part of training data) is fed as input, and temperature profile along with A is obtained as output. The advantage of this approach is that only few cases need to be solved with any of the means: analytically, numerically or experimentally to generate data. Once sufficient data are generated to train SOMARBNNs, and then, various alterations can be solved by using SOMARBNNs. One case (i.e., producing temperature profile and A corresponding to any velocity profile) is shown in the present work, but the approach can be extended to solve any engineering problem. Present work reports first time use of SOMARBNNs, applied to non-Newtonian fluid flow problem.

Table 1 shows the latest literature available for application of ANN/RBNN on non-Newtonian fluids. It is observed from Table 1 that large number of work is done on non-Newtonian fluids by employing ANN, but optimization of RBNN by different stochastic optimization methods, and comparative assessment of the final result is not available in open literature. The present work intends to find a best solution to optimize RBNN for application in non-Newtonian fluids. The performance of different SOMARBNNs is compared and analyzed under various noises in the input data. PSAARBNN is found to be the most suitable in terms of accuracy, CPU time consumption, and capability to handle noisy data set. (Table 1 to be inserted here).

The advantages of present approach are: (i) the main attribute in RBNN is easily obtained, which otherwise requires lots of technical knowledge of neural networks, (ii) solving any inverse engineering problem becomes easy (iii), if few cases of any engineering problem are solved either experimentally, numerically or analytically, large number of other cases with various levels of alterations can be solved within very short time.

2 Formulation

2.1 First Part: Generation of Data for RBNN

Two parallel plates subject to uniform heat fluxes (q_1 and q_2) are arranged to make channel with the following dimensions: distance H, length L and width W. An incompressible non-Newtonian fluid flow through the parallel plates under the steady-state condition and is shown schematically in Fig. 1. The flow is assumed to be laminar and fully developed (thermally and hydro-dynamically). The dimension of the channel along the x* axis is large as compared to the dimension in other two axes, i.e., $L > > H \quad W$. The equations governing the flow can be presented as: (Fig. 1 to be inserted here).

Equation of continuity

Table 1 Research available in open literature about use of ANN in fluid flows

S. No.	Paper	Work
1	H. Kalani et al. [18]	Used particle swarm optimization-assisted artificial neural networks of different types, for modeling a photo-thermal nanofluid-based collector system
2	M.H. Esfe [19]	Used artificial neural network for predicting heat transfer and pressure drop characteristics of nanofluid in a heat exchanger
3	R. Daneshfar et al. [20]	Employed various alterations of ANN to determine the heat capacity of ionanofluids as a function of nanoparticle concentration, critical temperature, etc.
4	M. H. Esfe [21]	Two alterations of artificial neural network were used to predict thermal conductivity of ethylene glycol–water-based TiO2 nanofluids
5	K. Chhantyal et al. [22]	ANN and support vector machine (SVM) are used to develop empirical models in to be used in ultrasonic-level sensors for flowmetering of non-Newtonian fluids in open Venturi channels
6	H. Eshgarf [23]	ANN models were used to forecast the rheological behavior of MWCNTs–SiO_2/EG–water non-Newtonian hybrid nanofluid
7	H. Wu et al. [24]	They used ANN + GA to improve the pipe flow hydrodynamic and thermal properties such as pressure drop and heat transfer coefficient for a non-Newtonian nanofluid composed of Fe3O4 nanoparticles dispersed in liquid paraffin
8	S. Zhang et al. [25]	Radial basis neural network was used to predict thermal conductivity for a non-Newtonian fluid
9	V.K. Mishra et al. [26]	Used ANN + GA for estimation of one parameter in non-Newtonian fluid flow problem
10	M. Amani et al. [27]	ANN was used to evaluate friction factor and heat transfer coefficient of the aqueous solution of TiO2/CMC-water non-Newtonian nanofluid

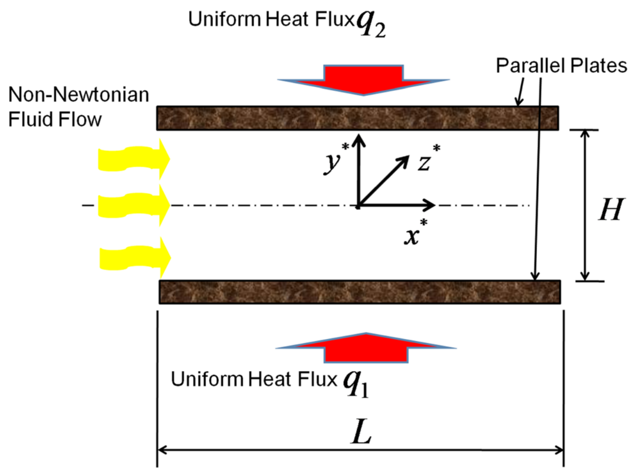


Fig. 1 Schematic of the non-Newtonian fluid flow under uniform heat flux

$$\nabla \cdot V^* = 0 \tag{1}$$

Equation of momentum conservation

$$\rho \frac{DV^*}{Dt} = \nabla \cdot \tau + f \tag{2}$$

It is to be noted that body forces are neglected in Eq. (2).

For modeling a non-Newtonian fluid flow, the nonlinear relationship between stress and the strain rate can be expressed by the third-grade fluid model as [17]:

$$\tau = -pI + \mu A_1 + \alpha_1 A_2 + \alpha_2 A_1^2 + \beta_1 A_3 + \beta_2 \left(A_1 A_2 + A_2 A_1\right) + \beta_3 \left(trA_1^2\right) A_1 \tag{3}$$

$$A_1 = (gradV^*) + (gradV^*)^{Transpose} \tag{4}$$

$$A_n = \frac{dA_{n-1}}{dt} + A_{n-1}(gradV^*) + (gradV^*)^{Transpose} A_{n-1}, \quad n = 1, 2, 3 \tag{5}$$

Equation of energy conservation

$$\rho c_p \frac{dT^*}{dt} = \tau : grad(V^*) - \nabla \cdot (-k_{th} \nabla T^*) \tag{6}$$

With the help of assumption of hydro-dynamically fully developed flow, the only velocity component that remains nonzero is the axial one, with dependency only on y^*. Thus, the velocity vector becomes:

$$v^* = \left[u^*(y^*), 0, 0\right] \tag{7}$$

With the help of Eq. (7), the momentum conservation equation given by Eq. (2) simplifies to:

$$\frac{\partial p^*}{\partial x^*} = \mu \frac{d^2 u^*}{dy^{*2}} + 2(\beta_2 + \beta_3) \frac{d}{dy^*} \left(\frac{du^*}{dy^*} \right)^3 \tag{8}$$

$$\frac{\partial p^*}{\partial y^*} = \left(2\alpha_1 + \alpha_2\right) \frac{d}{dy^*} \left(\frac{du^*}{dy^*} \right)^2 \tag{9}$$

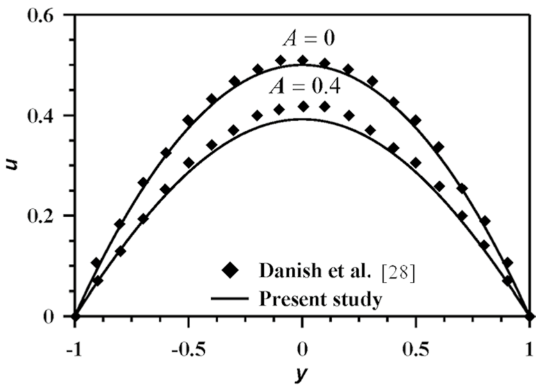


Fig. 2 Validation of the first part by comparing the velocity by LSM with exact solution [28]

$$\frac{\partial p^*}{\partial z^*} = 0 \tag{10}$$

Similarly, by using Eq. (7), energy conservation equation given by Eq. (6) simplifies to:

$$\rho c_p u^* \frac{\partial T^*}{\partial x^*} = k_{th}\left(\frac{\partial^2 T^*}{\partial x^{*2}} + \frac{\partial^2 T^*}{\partial y^{*2}}\right) + \mu\left(\frac{du^*}{dy^*}\right)^2 + 2(\beta_2+\beta_3)\left(\frac{du^*}{dy^*}\right)^4 \tag{11}$$

The effect of third-grade fluid parameter A on energy equation is represented by the third term in the right-hand side of Eq. (11). Because of the assumption of thermally developed conditions, the temperature profile becomes independent of x^* and depends only upon y^*. The non-dimensional temperature can be defined as follows:

$$\theta = \frac{T^* - T^*_{w_l}}{T^*_m - T^*_{w_l}}$$

For constant wall heat flux condition, the following important relations can be deduced:

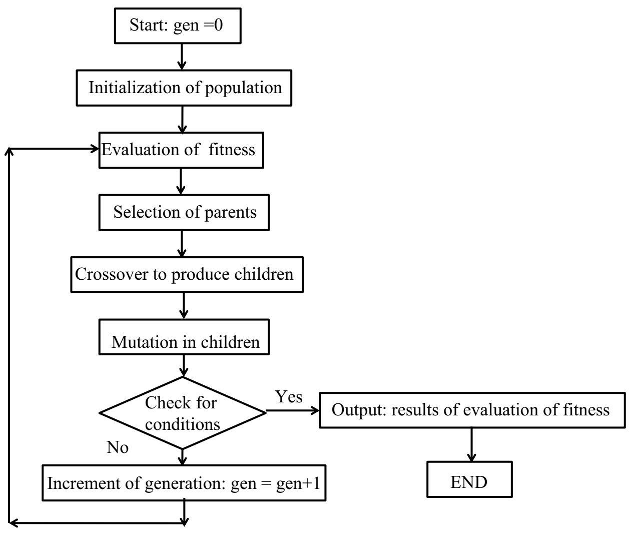

Start: gen =0
Initialization of population
Evaluation of fitness
Selection of parents
Crossover to produce children
Mutation in children
Check for conditions
Yes
No
Output: results of evaluation of fitness
END
Increment of generation: gen = gen+1

Fig. 4 Flowchart of processes in GA

$$\frac{\partial T^*}{\partial x^*} = \frac{dT^*_m}{dx^*} = const., \frac{\partial^2 T^*}{\partial x^{*2}} = 0 \tag{12}$$

Thus, substitution of Eq. (12) into Eq. (11) gives the following form of energy equation

$$\rho c_p u^* \frac{\partial T^*}{\partial x^*} = k_{th}\left(\frac{\partial^2 T^*}{\partial y^{*2}}\right) + \mu\left(\frac{du^*}{dy^*}\right)^2 + 2(\beta_2+\beta_3)\left(\frac{du^*}{dy^*}\right)^4 \tag{13}$$

Boundary conditions: (a) constant and uniform heat fluxes on both the plates, (b) at the plate–fluid interface, no-slip conditions prevail. Thus, the following expressions can be written

$$u*(-h) = 0,\; u*(h) = 0 \tag{14}$$

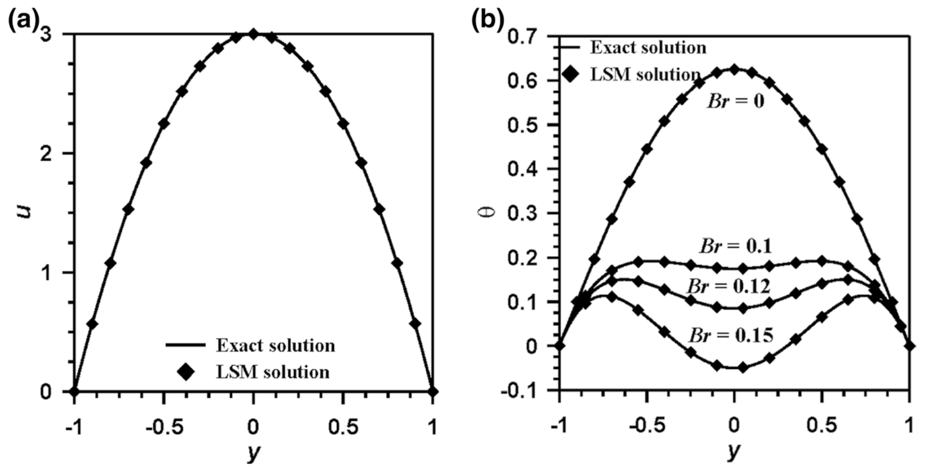


Fig. 3 Validation of the first part with exact solution for Newtonian fluid [28]. **a** Comparison of velocity profiles. **b** Comparison of non-dimensional temperature profiles

$$k_{th}\left(\frac{\partial T^*}{\partial y^*}\right)_{-h} = \mp q_1, k_{th}\left(\frac{\partial T^*}{\partial y^*}\right)_{h} = \pm q_2 \tag{15}$$

To make the analysis independent of requirement of knowledge of plate surface temperature, the temperature is non-dimensionalized in the following way:

$$\theta = \frac{T^* - T^*_{w_l}}{q_1 h/k_{th}} \tag{16}$$

Further, the fluid velocity at any location, and distance (along y^* direction) are non-dimensionalized as:

$$u = \frac{u^*}{u_0}, y = \frac{y^*}{h} \tag{17}$$

With the help of non-dimensional terms defined in Eqs. (16) and (17), conservation equations of momentum and energy can be expressed in the non-dimensional form as:

$$\frac{d^2u}{dy^2} + 6A\left(\frac{du}{dy}\right)^2\frac{d^2u}{dy^2} = N \tag{18}$$

$$\beta u = \frac{d^2\theta}{dy^2} + Br\left(\frac{du}{dy}\right)^2 + 2ABr\left(\frac{du}{dy}\right)^4 \tag{19}$$

$$A = \frac{\beta_2+\beta_3}{\mu}\left(\frac{u_0}{h}\right)^2, Br = \frac{\mu u_0^2}{q_1 h}, u_0 = \frac{1}{N}\frac{dp^*}{dx^*}\frac{h^2}{\mu}, \beta = \frac{\rho c_p u_0}{q_1}\frac{dT^*}{dx^*} \tag{20}$$

In Eq. (20), u_0 is the average velocity of fluid flow.

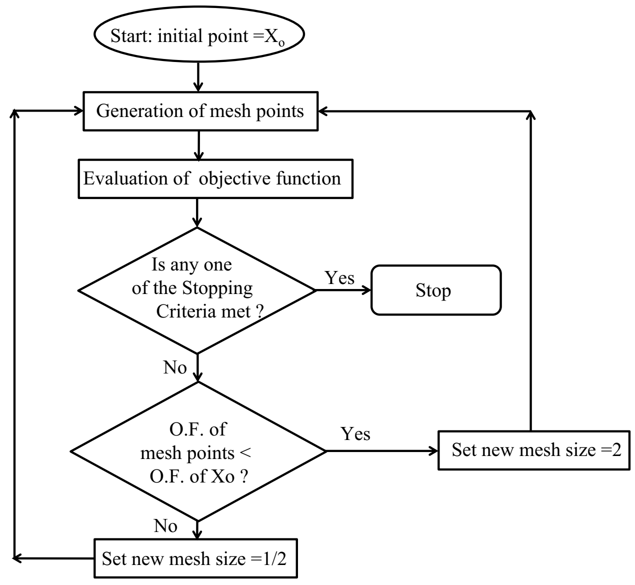

Start: initial point =X$_o$
Generation of mesh points
Evaluation of objective function
Is any one of the Stopping Criteria met ?
Yes
Stop
No
O.F. of mesh points < O.F. of Xo ?
Yes
Set new mesh size =2
No
Set new mesh size =1/2

Fig. 5 Flowchart of pattern search algorithm (PSA)

Equation (14) can now be presented in the non-dimensional form as:

$$u(-1) = 0, u(1) = 0 \tag{21}$$

Further, the boundary condition for energy equation becomes:

$$\theta(-1) = 0 \tag{22}$$

$$\frac{d\theta}{dx}(-1) = \mp 1 \tag{23}$$

$$\frac{d\theta}{dx}(1) = \pm\frac{q_2}{q_1} = q \tag{24}$$

Solution of the governing equations subjected to the boundary conditions is obtained by LSM, and the methodology of LSM application is explained in the next section.

2.1.1 Solution

2.1.1.1 Least Square Method (LSM) Starting with any arbitrary function $v(y)$, and operating this with a differential operator D to get a function say $g(y)$, it can be presented in the mathematical form as:

$$Dv(y) = g(y) \tag{25}$$

Also, a group of linear base functions who satisfy the boundary conditions can be arranged linearly to the function v to be $\underset{\sim}{v}$. This can be expressed as:

$$v \cong \tilde{v} = \sum_{i=1}^{n} c_i\phi_i \tag{26}$$

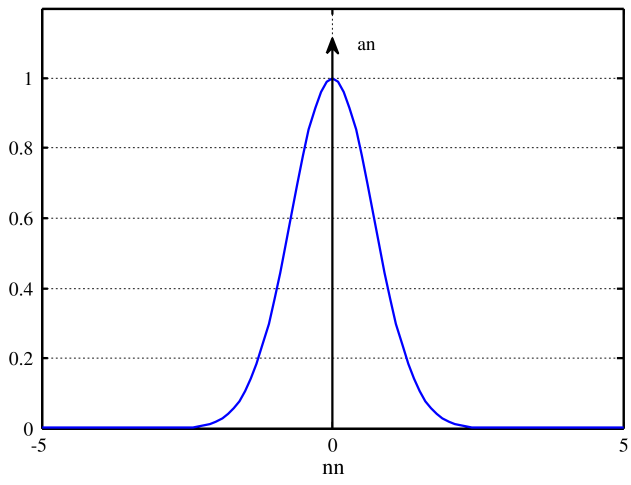


Fig. 6 Radbas activation function

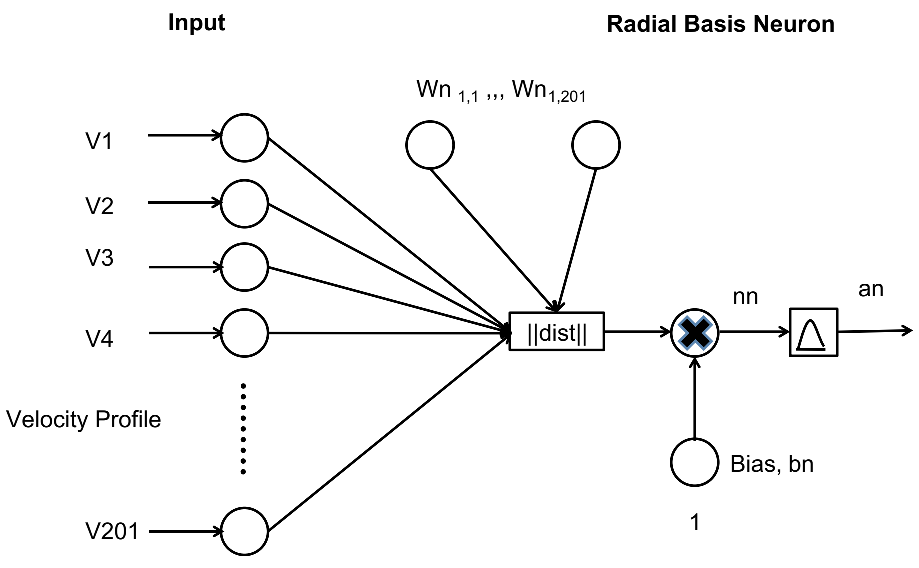


Fig. 7 Radial basis neuron

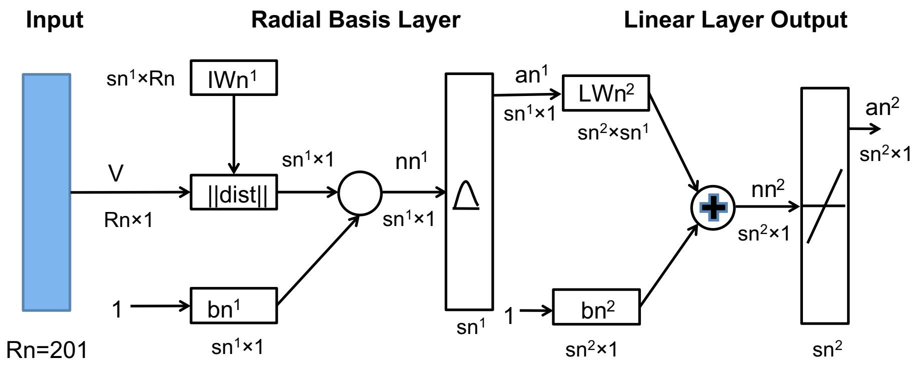


Fig. 8 Architecture of RBNN

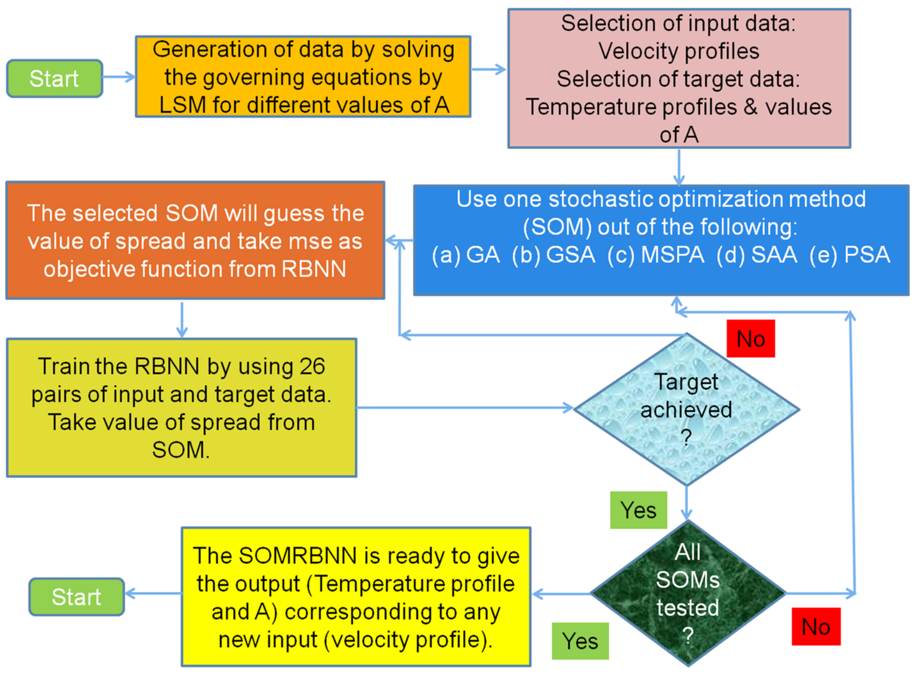

Start
Generation of data by solving the governing equations by LSM for different values of A
Selection of input data: Velocity profiles Selection of target data: Temperature profiles & values of A
Use one stochastic optimization method (SOM) out of the following: (a) GA (b) GSA (c) MSPA (d) SAA (e) PSA
The selected SOM will guess the value of spread and take mse as objective function from RBNN
Train the RBNN by using 26 pairs of input and target data. Take value of spread from SOM.
Target achieved ?
No
Yes
All SOMs tested ?
No
Yes
The SOMRBNN is ready to give the output (Temperature profile and A) corresponding to any new input (velocity profile).
Start

Fig. 9 Flowchart to use SOM-assisted RBNN for generating temperature profile along with A

Table 2 Results of different SOMARBNNs in retrieval of third-grade fluid parameter (A_{ret}) for $A = 0.75$

S. no.	Algorithm	Total time (s)	Fitness value	spread	A_{ret}	Absolute error in retrieved value of A	RBNN time (s)	Noise (%)
1	GA	84	9.7999×10^{-10}	2.5704	0.7500	0.00	2	0
2	GS	370	9.7073×10^{-10}	2.6099	0.7500	0.00	3	0
3	MSP	10	1.4138×10^{-9}	28.4341	0.7500	0.00	2	0
4	SA	65	9.7646×10^{-10}	2.5826	0.7500	0.00	2	0
5	PS	10	1.5105×10^{-9}	0.8168	0.7500	0.00	2	0
6	GA	96	1.7187×10^{-9}	1.8396	0.7501	0.02	2	0.1
7	GS	785	7.5392×10^{-8}	36.1157	0.7539	0.53	3	0.1
8	MSP	199	1.7349×10^{-9}	39.4829	0.7504	0.06	3	0.1
9	SA	71	1.9727×10^{-9}	20.9232	0.7500	0.00	3	0.1
10	PS	9	2.7723×10^{-9}	2.7500	0.7503	0.04	3	0.1
11	GA	81	6.0253×10^{-9}	16.2999	0.7504	0.06	2	0.2
12	GS	730	3.7493×10^{-8}	27.9120	0.7527	0.37	3	0.2
13	MSP	181	4.1118×10^{-8}	27.6489	0.7528	0.38	2	0.2
14	SA	72	3.1172×10^{-8}	25.9298	0.7498	0.03	3	0.2
15	PS	8	9.7416×10^{-9}	0.4375	0.7501	0.02	2	0.2
16	GA	82	7.9762×10^{-8}	16.7495	0.7537	0.50	2	0.5
17	GS	765	$1.5355\text{e} \times 10^{-8}$	14.4924	0.7509	0.13	2	0.5
18	MSP	218	8.0474×10^{-8}	18.7814	0.7539	0.53	17	0.5
19	SA	72	2.6623×10^{-8}	15.6627	0.7492	0.11	3	0.5
20	PS	9	1.8760×10^{-8}	1.5078	0.7503	0.04	3	0.5
21	GA	92	1.7301×10^{-7}	4.6898	0.7494	0.09	2	1
22	GS	840	3.9252×10^{-8}	12.3061	0.7514	0.19	4	1
23	MSP	252	9.4098×10^{-8}	11.8309	0.7531	0.42	3	1
24	SA	72	3.8517×10^{-8}	11.5062	0.7517	0.23	4	1
25	PS	10	7.1505×10^{-8}	0.7500	0.7502	0.03	2	1
26	GA	91	5.4749×10^{-7}	17.8381	0.7595	1.27	3	2
27	GS	593	1.5704×10^{-6}	36.0814	0.7666	2.22	2	2
28	MSP	203	7.3257×10^{-7}	34.5372	0.7613	1.51	3	2
29	SA	73	4.8089×10^{-7}	26.3000	0.7589	1.19	3	2
30	PS	8	3.1130×10^{-7}	0.6211	0.7471	0.39	3	2

Thus, with the help of Eqs. (25) and (26), we can define the residue as:

$$R(y) = D\{\tilde{v}(y)\} - g(y) \neq 0 \tag{27}$$

The residue need not be zero at every point, but their average quantity will reduce to zero.

$$\int_x R(y) w_i(y) dy = 0,\ i = 0, 2, 3..n \tag{28}$$

In Eq. (28), as per LSM method, the weights are taken to be equal to residual:

$$S = \int_y R(y) R(y) dy = \int_y R^2 dy \tag{29}$$

Now, S can be minimized as:

$$\frac{\partial S}{\partial c_i} = 2 \int_x R(y) \frac{\partial R}{\partial c_i} dy = 0 \tag{30}$$

Equation (30) gives equal number of algebraic relations and unknown constants, which can easily be computed.

2.1.1.2 Solution Methodology by Using LSM A group of base functions, which satisfy the boundary conditions, are used to approximate the velocity field:

$$u(y) = c_1(1 - y^2) + c_2(1 - y^4) \tag{31}$$

By using the definition of velocity field in Eq. (31), the residues are obtained from Eq. (18):

$$R(y) = a_0 + a_2 y^2 + a_4 y^4 + a_6 y^6 + a_8 y^8 \tag{32}$$

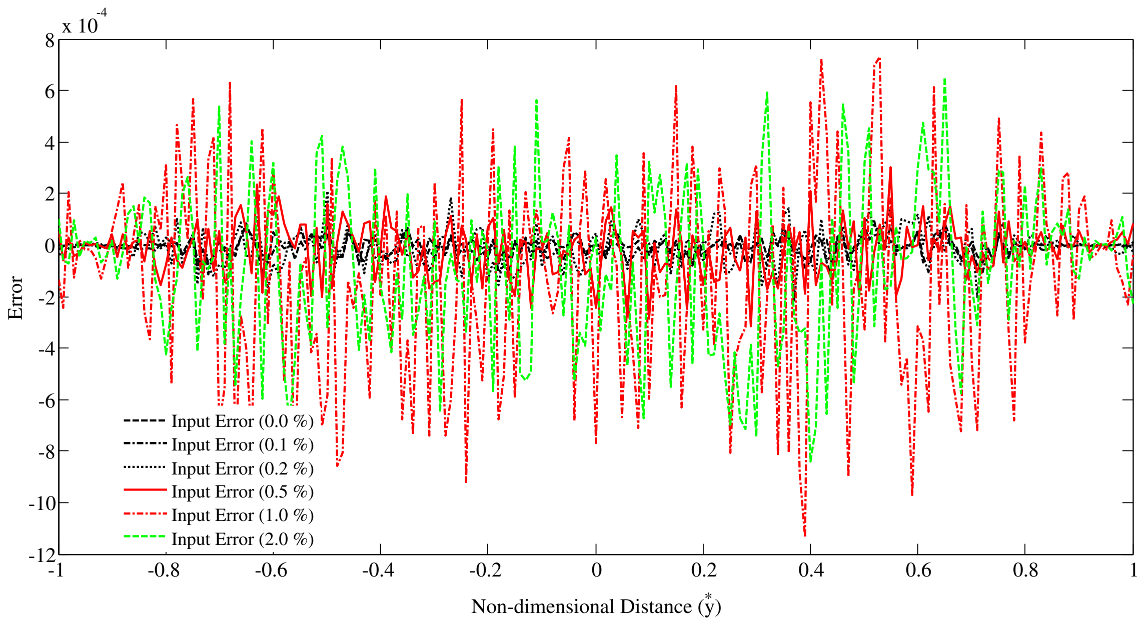


Fig. 10 Enlarged view of error in the temperature generated by GAARBNN for different levels of noise in the input data

$$a_0 = -2c_1 - N,\ a_2 = -12c_2 - 48c_1^3,\ a_4 = -480Ac_1^2c_2,$$
$$a_6 = 6A(32 + 12.16)c_1c_2^2,\ a_8 = -72.16Ac_2^3 \tag{33}$$

With the help of Eq. (27) describing the LSM technique, the following relations are obtained from Eq. (30):

$$\int_{-1}^{1} R\frac{\partial R}{\partial c_1}dy = 0 \tag{34}$$

$$\int_{-1}^{1} R\frac{\partial R}{\partial c_2}dy = 0 \tag{35}$$

Now, Eq. (34) and Eq. (35) are solved for the values of the constants c_1 and c_2, by using practically possible values of Ha, N, H and A. The energy equation of Eq. (19) can be expressed in terms of base function by using the definition of velocity field given by Eq. (31):

$$\frac{d^2\theta}{dy^2} = b_0 + b_2y^2 + b_4y^4 + b_6y^6 + b_8y^8 + b_{10}y^{10} + b_{12}y^{12} \tag{36}$$

$$b_0 = \beta(c_1 + c_2), b_2 = -\beta c_1 - 4Brc_1^2,$$
$$b_4 = -\beta c_2 - 16Brc_1c_2 - 32ABrc_1^4,$$
$$b_6 = -16Brc_2^2 - 256ABrc_1^3c_2,$$
$$b_8 = -2ABr(128 + 1616)c_1^2c_2^2,$$
$$b_{10} = -1024ABrc_1c_2^3, b_{12} = -32.16ABrc_2^4 \tag{37}$$

The temperature field is obtained by integration of Eq. (36) as:

$$\theta = b_0\frac{y^2}{2} + b_2\frac{y^4}{12} + b_4\frac{y^6}{30} + b_6\frac{y^8}{56} + b_8\frac{y^{10}}{90} + b_{10}\frac{y^{12}}{11.12} + b_{12}\frac{y^{14}}{13.14} + r_3y + r_4 \tag{38}$$

Substitution of boundary conditions from Eq. (22) to Eq. (24) gives the value of the constants r_3 and r_4 in Eq. (38) as:

$$\theta_m = \frac{T_m^* - T_1^*}{q_1\frac{h}{k}} = \frac{\int_{-1}^{1}\theta u dy}{\int_{-1}^{1} u dy} \tag{39}$$

Following definitions are used for Nusselt number:

(i) Nusselt number for the bottom plate:

$$\frac{hlw(4h)}{k_{th}} = \frac{4\left.\frac{d\theta}{dy}\right|_{y=-1}}{\frac{T_m^*-T_1^*}{(q_1h/k_{th})}} \tag{40}$$

$$Nu = -\frac{4}{\theta_m} \tag{41}$$

(ii) Nusselt number for the top plate: $Nu_{\text{upper}} = Nu\ q_2/q_1$.

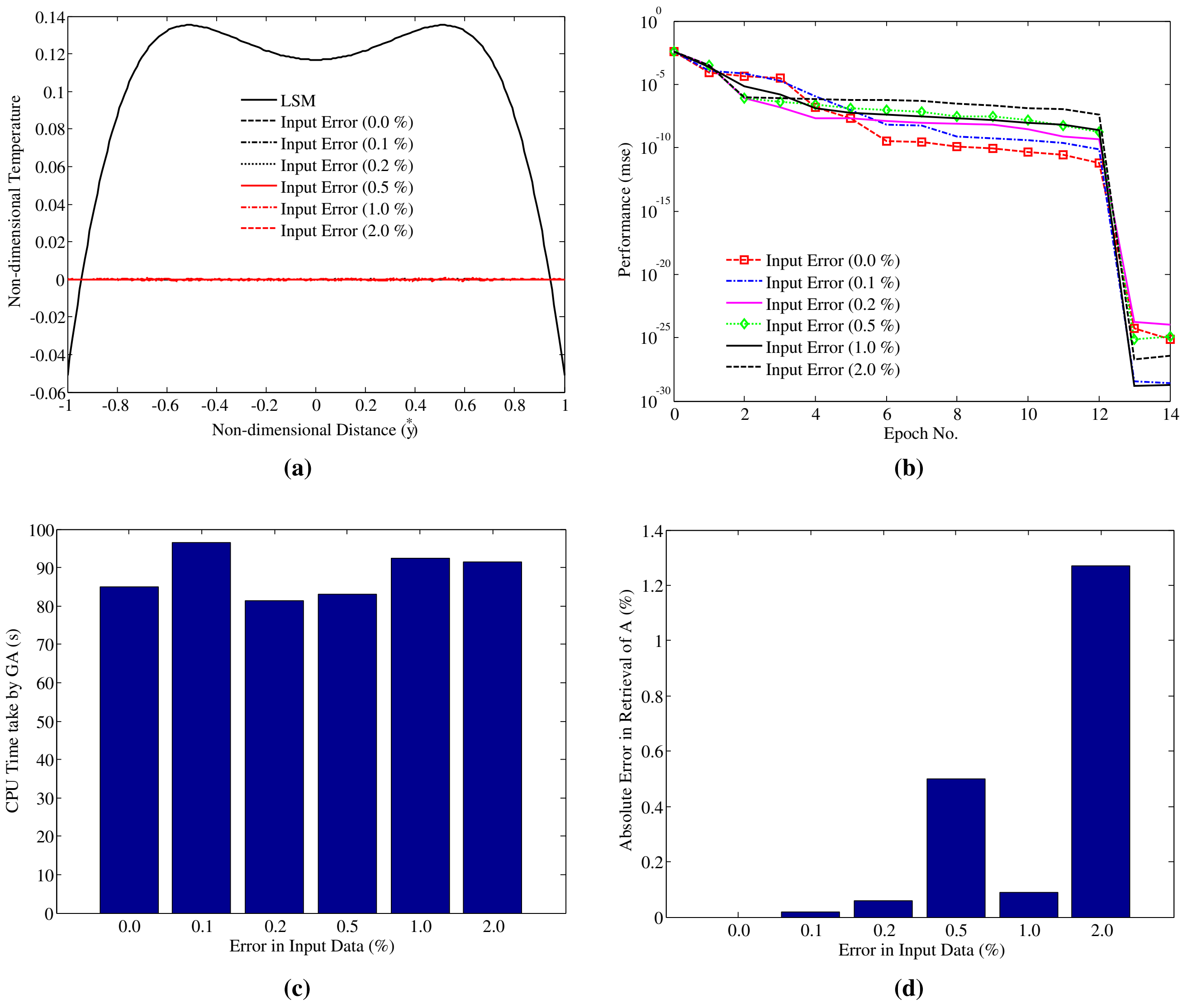


Fig. 11 Performance evaluation of GAARBNN for different levels of noise in the input data **a** Comparison of error in the generated output, **b** Convergence of GAARBNN during training stage, **c** CPU time consumption, **d** error in the retrieval of A

2.1.2 Validation of First Part

Once the governing equations are solved by using LSM, the first part of the problem is over. In the second part, data generated by the LSM are used for the development of SOMARBNNs. Before proceeding to the second part of the problem, the validation of the first part is presented in Figs. 2 and 3. Figure 2 shows that the velocity profile obtained from the LSM is compared with the data available in the literature [28]. Further, the temperature and the velocity profiles from LSM are compared with the result of analytical solution, for the Newtonian fluid, and are shown in Fig. 3. (Figs. 2 and 3 to be inserted here).

2.2 Second Part: Designing of RBNN with the Help of SOMs

After completion of the first part of the problem, the second part of the problem is solved. In the second part, five different SOMs are used to design a RBNN. All the SOMs are implemented in the present work, without taking the results from any other work. A brief description of different SOMs is presented below:

2.2.1 Genetic Algorithm

GA is a mathematical tool to search global minimum and is inspired from nature's process of evolution [29]. It

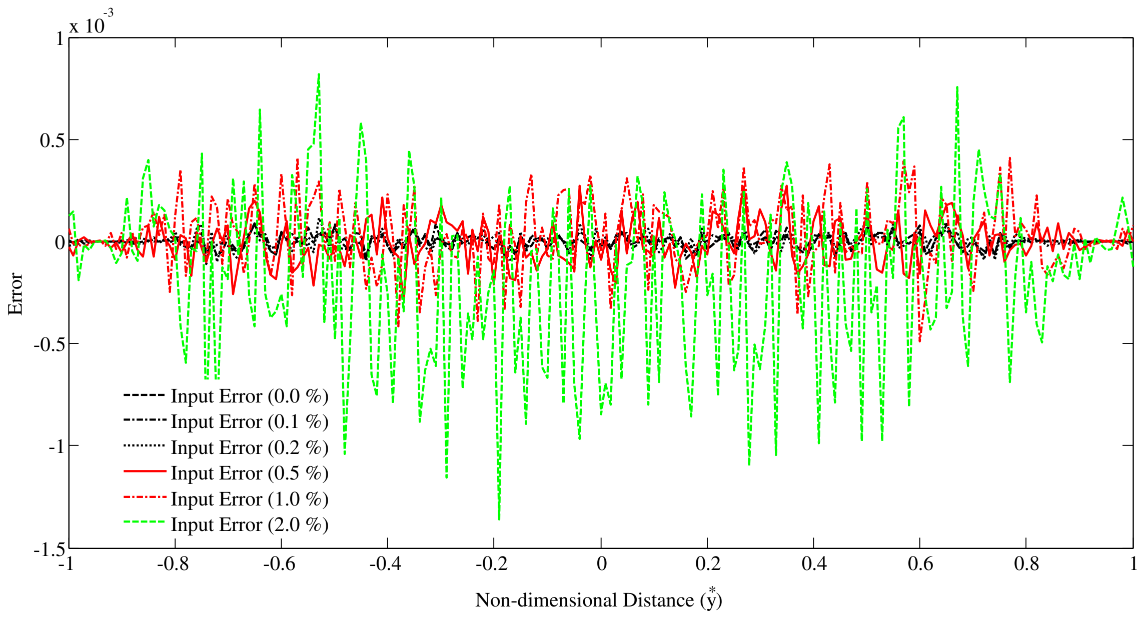


Fig. 12 Enlarged view of error in the temperature generated by GSAARBNN for different levels of noise in the input data

mathematically mimics different processes of evolution such as selection, crossover, and mutation. The flowchart of GA is shown in Fig. 4. First of all, a group of probable solutions are generated, called population of first generation. Then, depending upon the value of fitness value from each individual, selection and mutation are carried out to generate children, also called next generation. In this way, good individuals are generated for few generations till only few good individuals remain. These good individuals correspond to the best solution. (Fig. 4 to be inserted here).

2.2.2 Global Search Algorithm

GSA finds starting points for gradient-based local nonlinear programming (NLP) solvers [30]. Scatter search method OptQuest is employed by GSA to find the starting point for the gradient search algorithm. Global optimization along with powerful local search makes GSA very effective optimization tool. The scatter search is advanced than GA in terms of generating new generation in a deterministic means rather than the random means.

2.2.3 Multiple Starting Point Algorithm

MSPA uses local optimization routine but starts with various different points to solve any optimization problem [31]. The global solution is said to be obtained if best solution is picked. Chances of getting the global minimum increases with the increase in the number of starting points or repeating the program from different sets of starting points. The algorithm gives a set of good solutions, and any of them can be picked depending upon the required stability of the solution parameters.

2.2.4 Simulated Annealing Algorithm

SAA is very commonly used stochastic optimization method to solve unconstrained and bound-constrained problems [32]. It is inspired from the metallurgical process called annealing, where it mimics the physical process of heating a material and then slowly lowering the temperature to decrease defects in crystal structure, thus minimizing the system energy.

In each iteration, a new point is generated in a random way. A probability scale, proportional to the temperature, is used to locate the position of the new point, i.e., the extent of the search. The SAA accepts all the points, which give lower values for objective function. The algorithm also accepts few points with certain probabilities, which give higher values of objective function. This means of accepting the points with higher objective function helps the SAA to overcome the local minima and explore global minima.

2.2.5 Pattern Search Algorithm

PSA is computationally efficient algorithm yet very simple and easy to implement [33]. The flowchart of the steps involved in implementation of PSA is shown in Fig. 5. It starts with a single point Xn_o; then, a mesh is generated around this point. The objective function for all the mesh points is computed and compared with the objective function for point Xn_o.

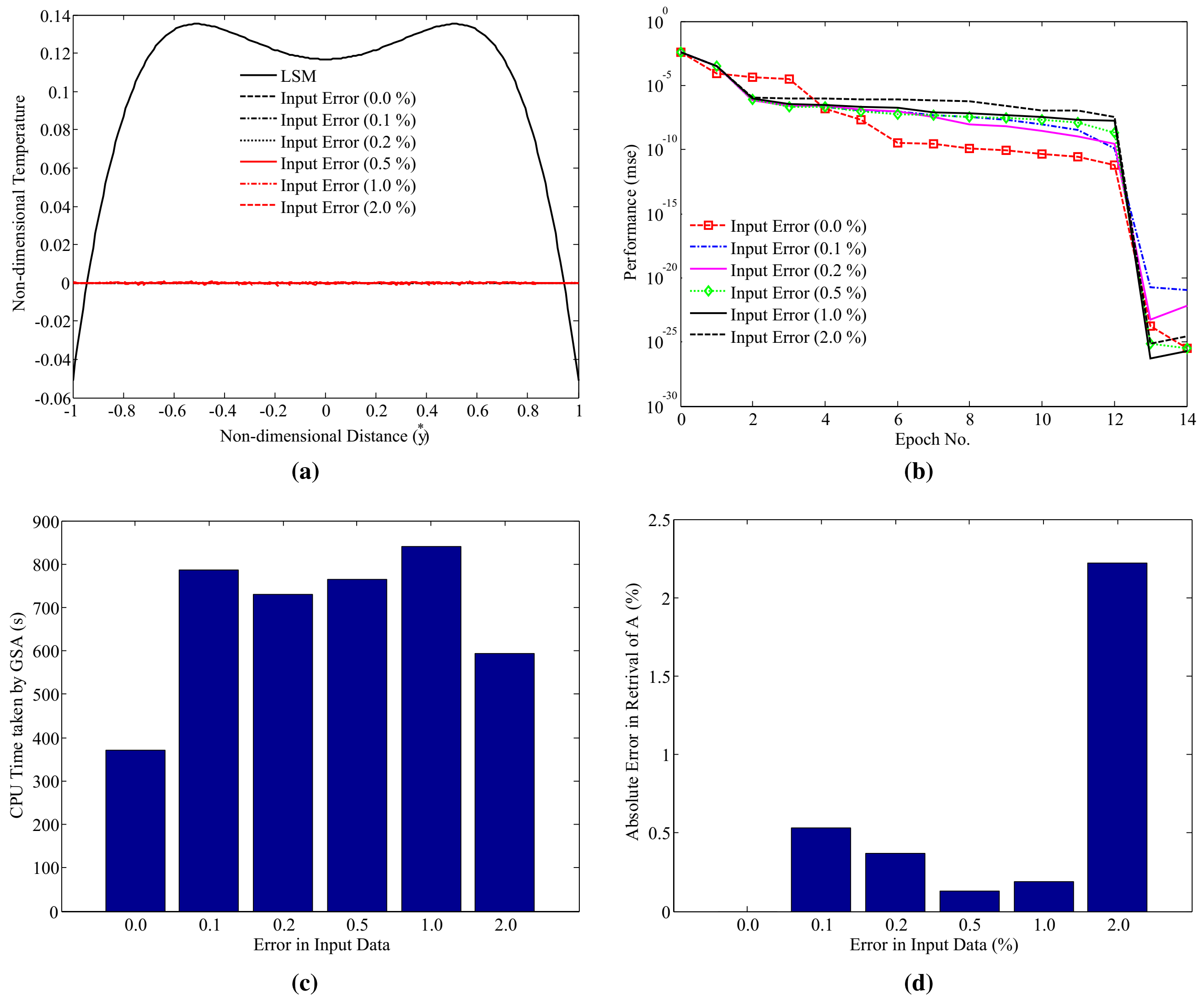


Fig. 13 Performance evaluation of GSAARBNN for different levels of noise in the input data: **a** Comparison of error in the generated output, **b** convergence of GSAARBNN during training stage, **c** CPU time consumption, **d** error in retrieval of A

If the objective function for any of these mesh points is less than that for the initial point, then that mesh point is designated as new Xn_1, and it is called successful poll. Then, a new mesh is generated around the point Xn_1 with twice the size, and the whole process is repeated, whereas if the algorithm encounters unsuccessful poll, i.e., all the mesh point gives higher value of objective function than the central point, then a new mesh is generated with half the size. In this way, the algorithm proceeds until it achieves the required accuracy for the objective function or any other stopping criterion is met. (Fig. 5 to be inserted here).

2.2.6 Radial Basis Neural Network

A radial basis neural network (RBNN) function is one of the most commonly used neural networks, apart from perceptron network. In RBNN, the final input to the radbas transfer function is a distance between a weight vector and an input vector and multiplied by a bias. The transfer function calculates a neuron's output as $\text{an} = \exp(-\text{nn}^2)$. When the distance is 0, the function gives it maximum value $\text{an} = 1$, and the function decreases with the increase in distance.

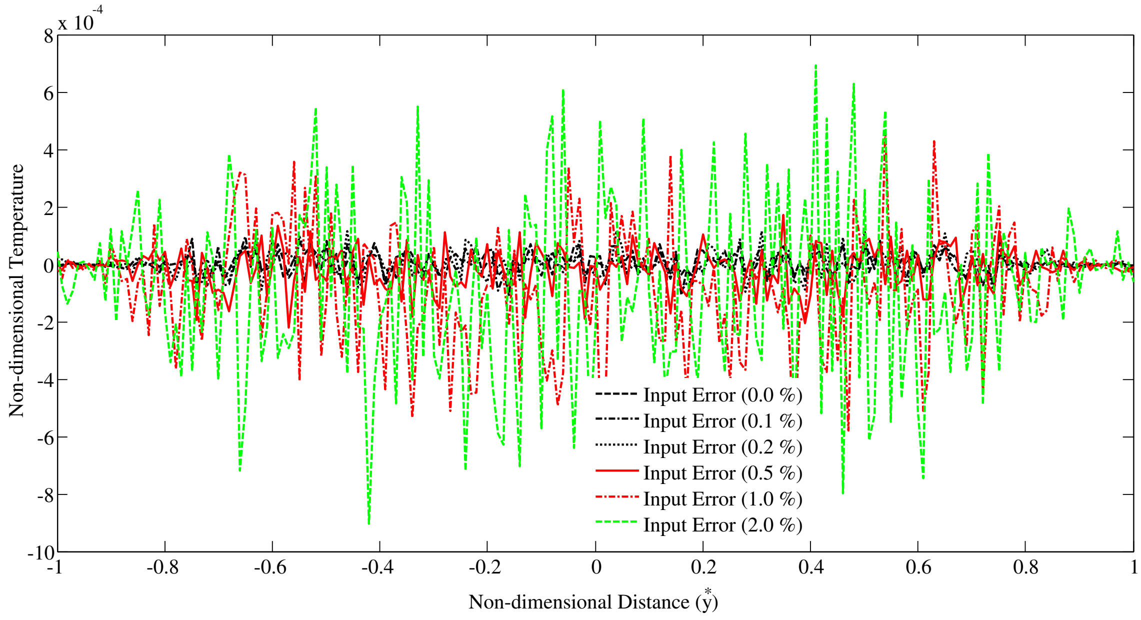


Fig. 14 Enlarged view of error in the temperature generated by MSPAARBNN for different levels of noise in the input data

The sensitivity of neuron is decided by the bias bn. The output of radbas for different inputs is shown in Fig. 6. The radial basis neuron is shown in Fig. 7. (Figs. 6 and 7 to be inserted here).

The network based on the radial basis function is illustrated in Fig. 8. One radial basis layer of sn^1 nodes constitutes the hidden layer, and the output linear layer consists of sn^2 nodes. With the help of input vector V and a weight matrix IWn^1, a distance vector of size sn^1 is generated. After scaling of the distances by bias vector bn1, the radial basis transfer function converts it into hidden layer output vector an1. The output layer in RBNN works similar to perceptron output layer with a linear transfer function to produce output vector an^2 of size sn^2. The coefficients IWn^1, LWn^2, bn^1 and bn^2, are computed automatically in MATLAB software. (Fig. 8 to be inserted here).

One neuron is added at a time in the hidden layer, until the sum-squared error reaches the prescribed value or the prescribed maximum number of neurons is achieved. Thus, number of neurons in the hidden layer is automatically generated.

2.2.7 Stochastic Optimization Methods Assisted RBNN

RBNN is very simple to use, with only one parameter 'spread' is required from user to construct the model. Spread helps to decide the width of an area in the input space to which each neuron in the hidden layer responds. The accuracy and time consumption of the RBNN model depend on the value of spread. Proper selection of the value for spread is very vital for the performance of RBNN model and at the same time it is very difficult, as lot of experience is required to come to a final value. But in the present work, a novel approach of using stochastic optimization method for selecting suitable value for spread is proposed. Different stochastic optimization methods such as GA, GSA, MSPA, SAA, and PSA are used, and the results are compared and analyzed on the basis of accuracy and time consumption.

Flowchart of the stochastic optimization method-assisted RBNN (SOMARBNN) is shown in Fig. 9. First of all, the first part of the problem is used to generate multiple pairs of the velocity- and the temperature- profile for different values of *A*. This gives the data in the form of input (velocity profile) and target (temperature profile and A), to be used in the SOMARBNN for training purpose. Anyone SOM is selected to guess the value for spread, and by using this spread, RBNN is used. The RBNN returns mean-squared error (mse), which acts as objective function (OF) for the selected SOM. Once SOM gets lowest mse, it finalizes the guessed value of 'spread' for RBNN. Thus, this 'spread' is used in RBNN, and output is produced corresponding to any given input. (Fig. 9 to be inserted here).

These steps are repeated for all the SOMs, and comparison of the result is done to select most suitable SOMARBNN. The analysis is also done with different levels (0.0, 0.1, 0.2, 0.5, 1.0, and 2.0) of perturbation in the input data. In all the SOMARBNNs, number of neurons in the hidden layer are 14, and it is automatically decided.

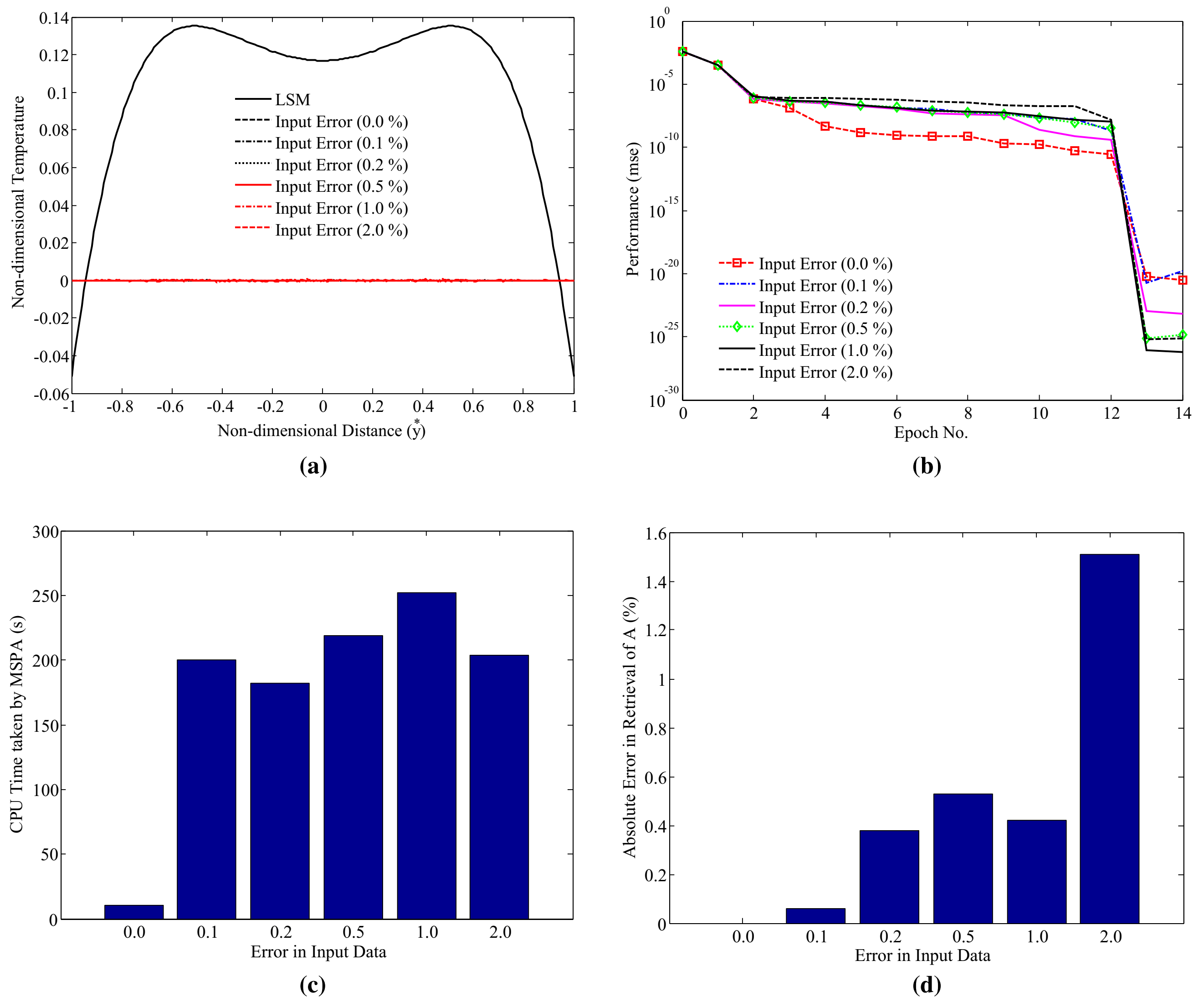


Fig. 15 Performance evaluation of MSPAARBNN for different levels of noise in the input data: **a** Comparison of error in the generated output, **b** convergence of MSPAARBNN during training stage **c** CPU time consumption **d** error in retrieval of A

3 Results and Discussions

In the quest for generating an efficient and robust RBNN, different SOMs are explored to decide the values of spread. The SOMARBNNs are used to get the temperature profile and the value of A corresponding to any velocity profile (not part of initial data). With the help of mse, measured as the difference between the true value (computed by LSM) and the corresponding network output, the convergence of RBNN is monitored [34] and is given by Eq. (42) as:

$$OF = mse = \frac{1}{Rn+1}\left(\left(A - A_{ret}\right)^2 + \sum_{i=1}^{Rn}\left(\theta_i - \tilde{\theta}_i\right)^2\right) \quad (42)$$

where Rn is number of observation points (=201), A and θ is true value of TGF parameter and temperature (obtained from LSM), respectively. RBNN-generated TGF parameter and temperature are represented by A_{ret} and $\tilde{\theta}$, respectively. The mse calculated in the last iteration of RBNN is allocated to the objective function OF, to be used in SOMs. The accuracy of retrieval of A is measured by using the following error function [34]:

$$error = \left|\frac{A - A_{ret}}{A}\right| \times 100 \quad (43)$$

where A_{ret} is retrieved value of TGF parameter, and A is true value of TGF parameter.

Results of different SOMARBNNs are presented in Table 2. For no noise in the input data, all the SOMARBNNs retrieve the value of A with zero error. Under the total time column, CPU time consumption by different SOMARBNNs

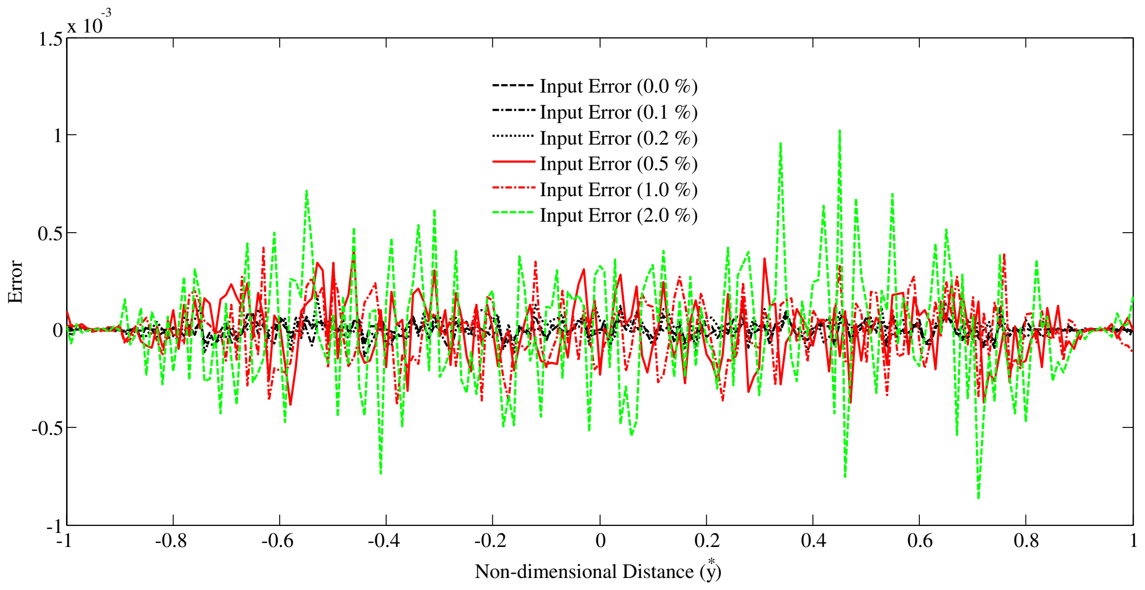


Fig. 16 Enlarged view of error in the temperature generated by SAAARBNN for different levels of noise in the input data

is listed. RBNNs assisted by MSPA and PSA take minimum time of 10 s. During the training stage of the SOMARBNNs, the performance of the RBNN is displayed under the column, fitness value (mse). Low value of fitness value indicates higher accuracy in the output (temperature profile and *A*) of SOMARBNNs. Value of spread for RBNN, as given by SOMs, is listed under the column 'spread'. Once spread is given by the SOMs, RBNN runs very fast and takes 2–3 s for almost all the cases and is displayed under the column RBNN time.

It is observed from Table 2 that the fitness value increases with the increase in noise level. Generation of spread by different SMOs is on the basis *OF*, with no observable trend. Error in the retrieved value of *A,* increases with the increase in the noise. Detailed interpretation and plots for different SOMARBNNs are discussed below: (Table 2 to be inserted here).

3.1 Performance Evaluation of GAARBNN

Figure 10 shows the error in the temperature profile generated by GAARBNN for different levels of noise in the input data. It is observed that for all the cases the error in the output is in the order of 10^{-4}, which is highly accurate. Also, the error in the temperature profile generated by GAARBNN is proportional to the noise in the input data (Fig. 10 to be inserted here).

In Fig. 11, a detailed performance evaluation of GAARBNN is presented. Temperature profile generated by LSM, along with the error in the temperature profile generated by GAARBNN, for different levels of noise in the input data, is shown in Fig. 11a. Robustness of the GAARBNN is evident from the observation that the error in the output is very less even for high-level noise (2.0%) in the input data. Figure 11b shows the performance of the GAARBNN during the training stage, for different levels of noise in the input data.

It is observed that the mse decreases continuously with the increase in iteration (epoch). CPU time consumption by GAARBNN for different cases is shown in Fig. 11c. For all the cases, time consumption is approximately the same. Error in the retrieved value of A for different cases is shown in Fig. 11d. For high noise (2.0%) case, the error in retrieval is more than 1%, whereas for all the other cases the error is less than 1%. This indicates that the GAARBNN is capable of giving accurate results, even under the situations where the input data are noisy. (Fig. 11 to be inserted here).

3.2 Performance Evaluation of GSAARBNN

Error in the temperature profile obtained from GSAARBNN for different levels of noise in the input data is compared in Fig. 12. For all the cases, the order of error is 10^{-3}, which is poor as compared to GAARBNN. For high level of noise in the input data, the error in temperature profile is high. Figure 13 shows the results of GSAARBNN for different cases of noise in the input data. Figure 13a shows that along the whole height (distance along *y**) the error in the result is very less. (Fig. 12 to be inserted here).

Convergence of GSAARBNN during the training stage is shown in Fig. 13b. For no noise case, the rate of convergence is faster than other cases. All the cases converge very fast during 12th iteration. CPU time consumption by

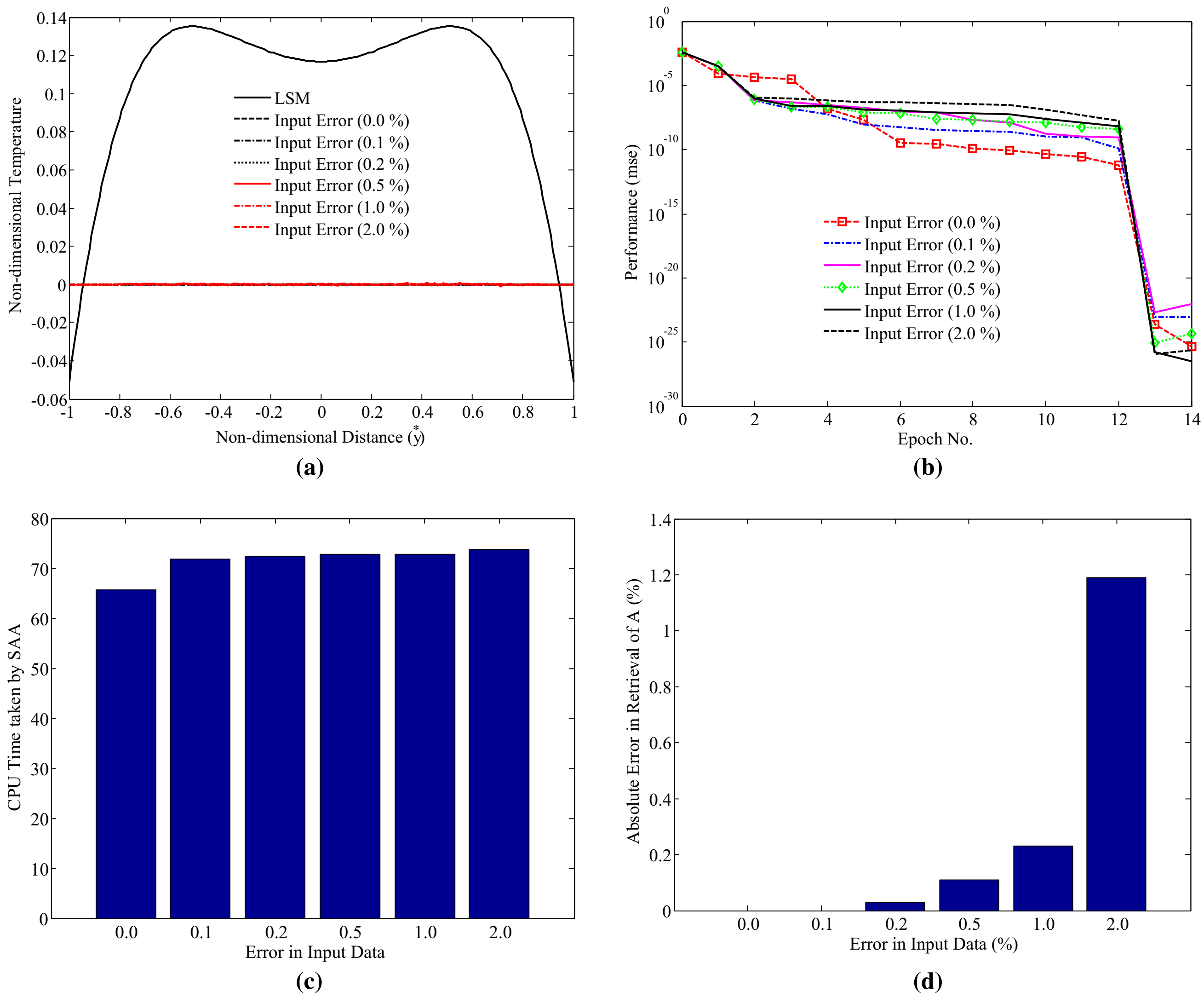


Fig. 17 Performance of SAAARBNN for different levels of noise in the input data: **a** error in the output generated, **b** convergence during the training stage **c**, consumption of CPU time, **d** error in retrieval of A

GSAARBNN for all the cases is shown in Fig. 13c. Time consumption for no noise case is very less, as compared to noisy cases. The error in the retrieved value of A from GSAARBNN for all the cases is shown in Fig. 13d. Except for high noise (2.0%) case, for all the cases the error in the retrieved value is less than 0.5. (Fig. 13 to be inserted here).

3.3 Performance Evaluation of MSPAARBNN

An enlarged view of the comparison of the error in the temperature profile generated by MSPAARBNN for different levels of noise in the input data is shown in Fig. 14. It is observed that for all the cases, the error is in the order of 10^{-4}, which indicates high accuracy in results. Also, the errors for low noise level are less and high for high noise levels. Detailed performance evaluation of MSPAARBNN is presented in Fig. 15. Figure 15a shows the error in temperature profile generated by MSPAARBNN along the y* axis and compared against the LSM solution. It is observed that for all the cases the output is very accurate. (Fig. 14 to be inserted here).

Plots of iteration verses mse are plotted for different noise cases in Fig. 15b. For all the cases, accuracy improves significantly after iteration 12. CPU time consumption by MSPAARBNN for different cases is compared in Fig. 15c. Drastic difference in time consumption is observed between no noise and noisy cases. Error in the retrieved value of A is shown in Fig. 15d. Up to 1.0% noise, the error in result is found to be less than 0.6%, whereas for high level of noise (2.0%), the error in the result is more than 1.4%. (Fig. 15 to be inserted here).

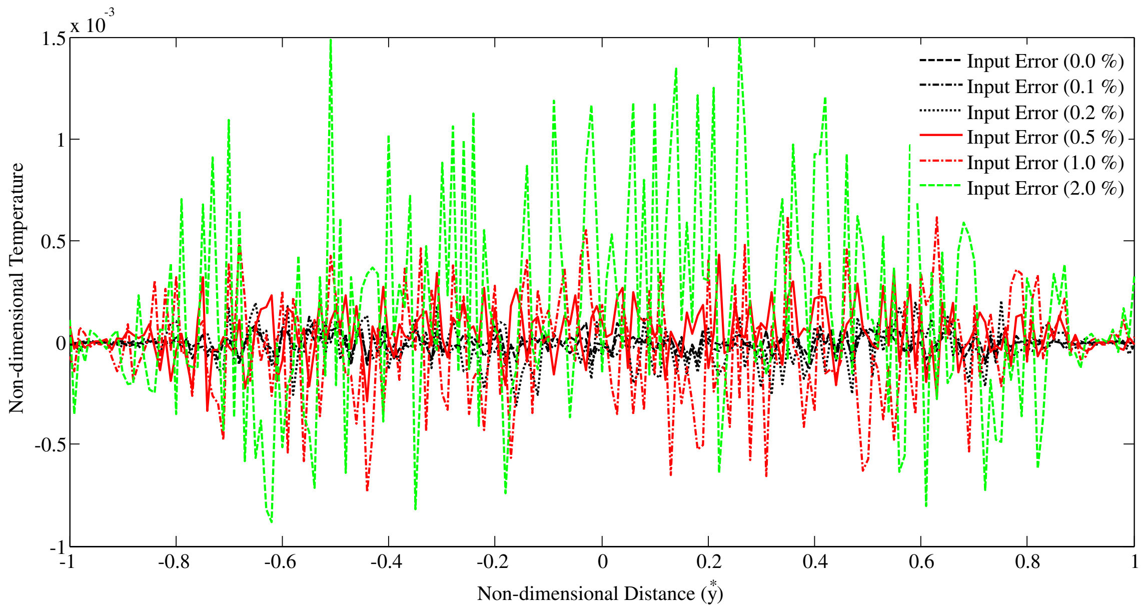


Fig. 18 Enlarged view of error in the temperature generated by PSAARBNN for different levels of noise in the input data

3.4 Performance Evaluation of SAAARBNN

Enlarged view of the comparison of the error in the temperature profile generated by SAAARBNN for different levels of noise in the input data is shown in Fig. 16. Error in all the cases is of the order of 10^{-3}, which is not so good as compared to some of the other SOMARBNNs. The error increases with the increase in noise in the input data. Detailed performance evaluation of SAAARBNN is presented in Fig. 17. Figure 17a shows the error in the temperature profile produced by SAAARBNN for different noise levels in the input data. It is observed that temperature along the y^* axis, as produced by SAAARBNN, is having very less error for all the cases. (Fig. 16 to be inserted here).

Convergence of SAAARBNN is shown in Fig. 17b, and the mse falls continuously with the increase in iteration. Also, after iteration 12, the *mse* falls very rapidly. CPU time consumption by SAAARBNN is shown in Fig. 17c, and it is observed that CPU time is approximately independent of noise in the input data. Error in the retrieved value of A is shown in Fig. 17d. Error in the output, in all the cases except high noise (2.0%), is less than 0.4%. (Fig. 17 to be inserted here).

3.5 Performance Evaluation of PSAARBNN

Enlarged view of the comparison of the error in the temperature profile generated by PSAARBNN for different levels of noise in the input data is shown in Fig. 18. Maximum error out of all the cases is of the order of 10^{-3}. The error increases with the increase in noise in the input data. Detailed performance evaluation of PSAARBNN is presented in Fig. 19. Figure 19a shows the error in the temperature profile produced by PSAARBNN for different noise levels in the input data. It is observed that the temperature along the y^* axis, as produced by PSAARBNN is having very less error for all the cases. (Fig. 18 to be inserted here).

Convergence of PSAARBNN is shown in Fig. 19b, and the *mse* falls continuously with the increase in iteration. Further, after iteration 12, the convergence speeds up rapidly. CPU time consumption by PSAARBNN is shown in Fig. 19c, and it is observed that the CPU time is approximately independent of noise levels in the input data. The error in the retrieved value of A is shown in Fig. 19d, for all the cases of noise levels in the input data. The error in the output is less than 0.05% for all the cases except for high noise (2.0%) (Fig. 19 to be inserted here).

3.6 Performance Evaluation of different SOMARBNN

Comparison of error in the temperature profile generated by different SOMARBNNs is shown in Fig. 20. Figure 20a shows error in the output temperature for the case of no noise in the input data. Output from RBNN, assisted by PSA and MSPA, has a maximum error, at the center of channel height. Similarly, Fig. 20b shows the error in the output temperature for the case of high noise (2.0%) in the input data. The error in the output from the RBNN assisted by PSA and GSA is higher than, by other SOMs. (Fig. 20 to be inserted here).

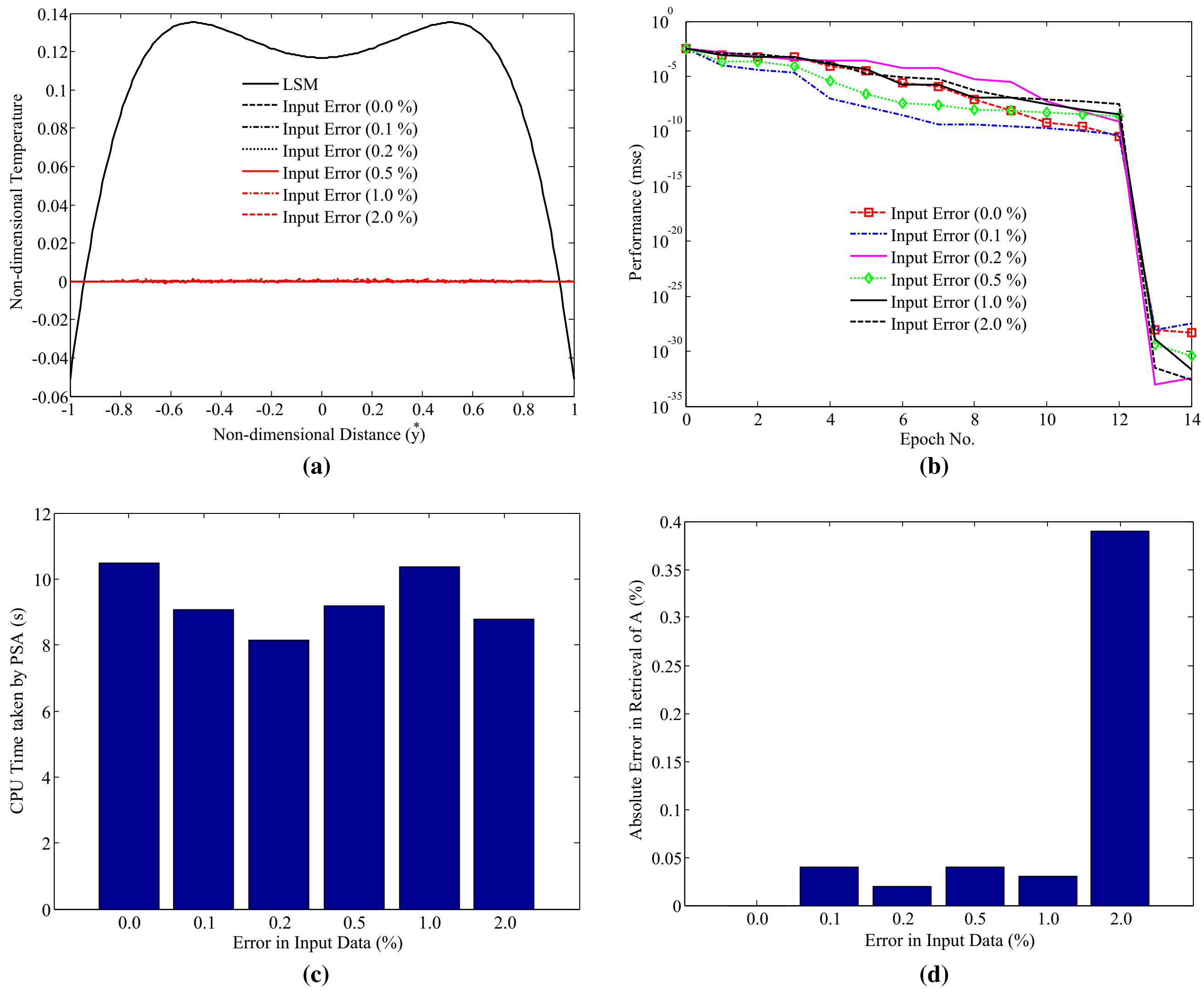
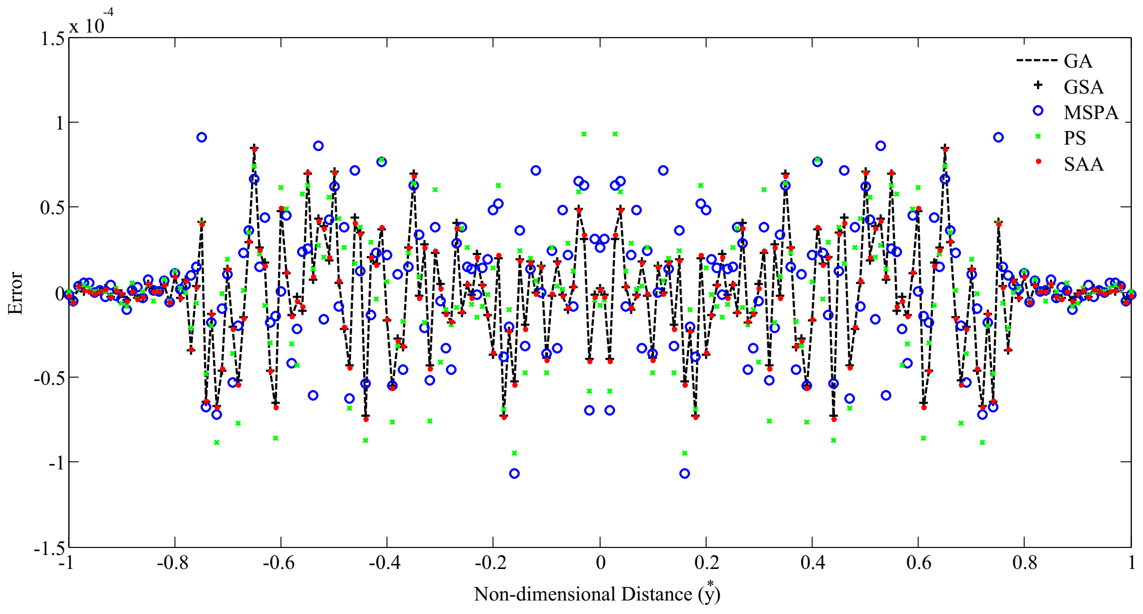


Fig. 19 Performance evaluation of PSAARBNN for different levels of noise in the input data: **a** comparison of the output generated, **b** convergence of PSAARBNN during training stage, **c** CPU time consumption, **d** error in the retrieval of A

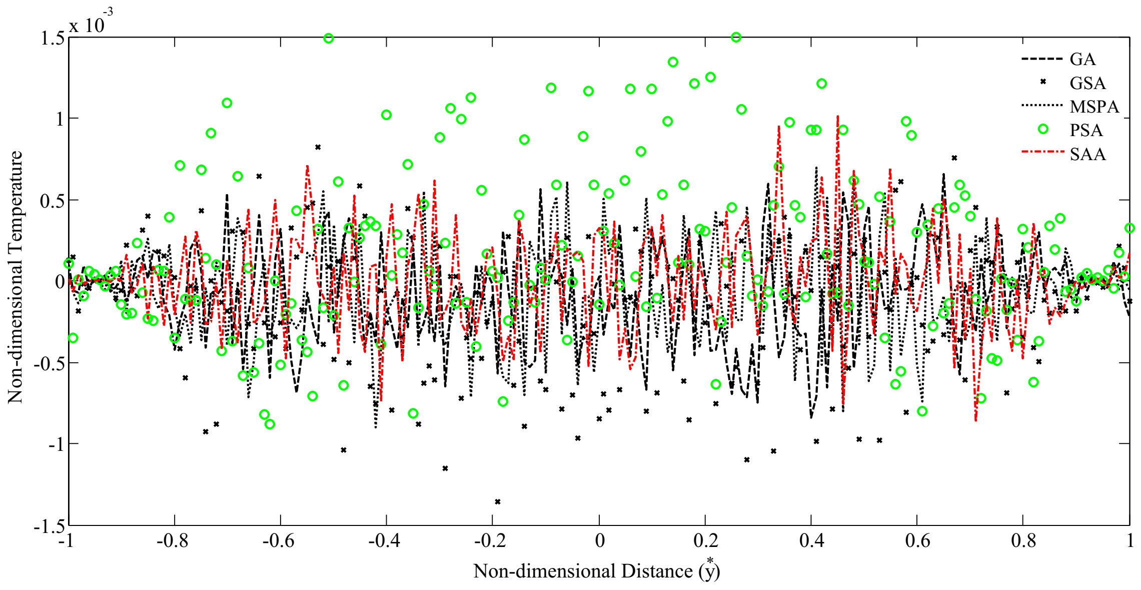
Convergence comparison of different SOMARBNNs is shown in Fig. 21, for different cases of noise in the input data. Figure 21a and b shows the convergence for the case of no noise and high noise (2.0%) in the input data, respectively. All the SOMARBNNs converge slowly up to 12 iteration and then speed up to finally converge within 14 iterations. PSAARBNN gives lowest *mse* for both the cases of noise in the input data, thus indicating higher accuracy in the output (Fig. 21 to be inserted here).

Figure 22 shows the performance evaluation of different SOMARBNNs in the retrieval of *A*, for different cases of noise in the input data. Figure 22a compares the CPU time consumed by different SOMARBNNs for giving the value of spread for a RBNN, in case of no noise in the input data. GSA is found to take maximum time to achieve the target. Error in the retrieved value of A by different SOMARBNNs under no noise is shown in Fig. 22b. Through this graph also, GSAARBNN is found to be least suitable due to low accuracy. However, for all other SOMARBNNs, the error in the output is less than 0.1%.

CPU time requirement by different SOMARBNNs is shown in Fig. 22c, for the case of high noise (2.0%) in the input data. PSA is found to take minimum time even in case of high noise. Figure 22d shows the error in the retrieved value of *A* by different SOMARBNNs under various level of noise in the input data. RBNN assisted by PSA is most accurate, while GSA gives poorest accuracy. All the runs were taken on 2.4 GHz processor with 8 GB RAM (Fig. 22 to be inserted here).



(a)



(b)

Fig. 20 Comparison of error in the output of different SOMARBNNs **a** Noise in the input data is 0.0% **b** Noise in the input data is 2.0%

4 Conclusions

Implementation of radial basis neural network (RBNN) is demonstrated by choosing a test case of non-Newtonian third-grade fluid flow with heat transfer, through two parallel plates. First part of the problem involves solving the governing equations subjected to the prescribed boundary conditions, by employing LSM. Pairs of velocity profile and temperature profile along with A, are prepared for the second part of the problem.

In the second part, RBNN is designed with the help of the following SOMs: GA, GSA, MSPA, SAA, and PSA. The data generated in the first part are then fed to train SOMARBNNs. After successful training of the SOMARBNNs, a velocity profile is used as input, and outputs obtained are: (i) temperature profile (ii) A. Performance of different SOMARBNNs under varying noise levels in the input data is analyzed and evaluated in terms of CPU time consumption and error in the result.

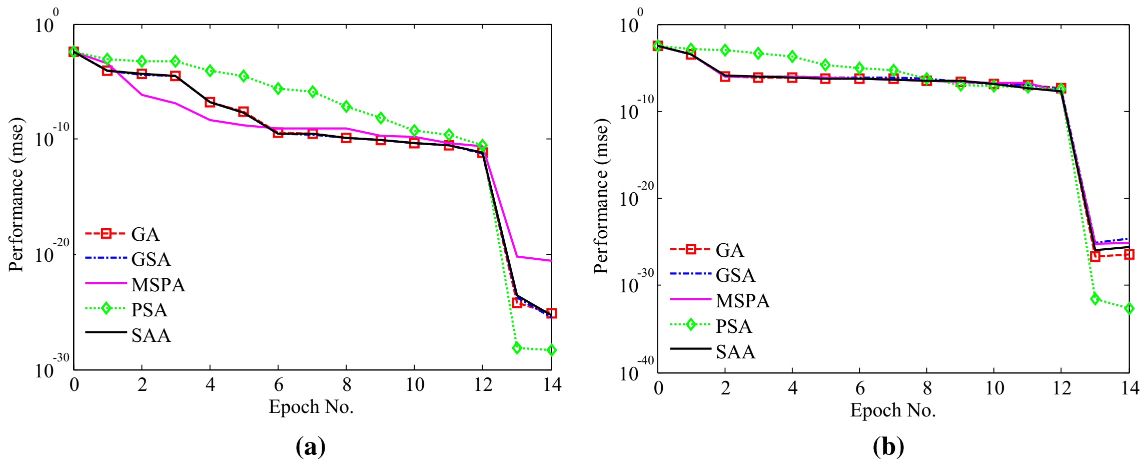


Fig. 21 Convergence comparison of different SOMARBNNs **a** noise in the input data is 0.0% and **b** noise in the input data is 2.0%

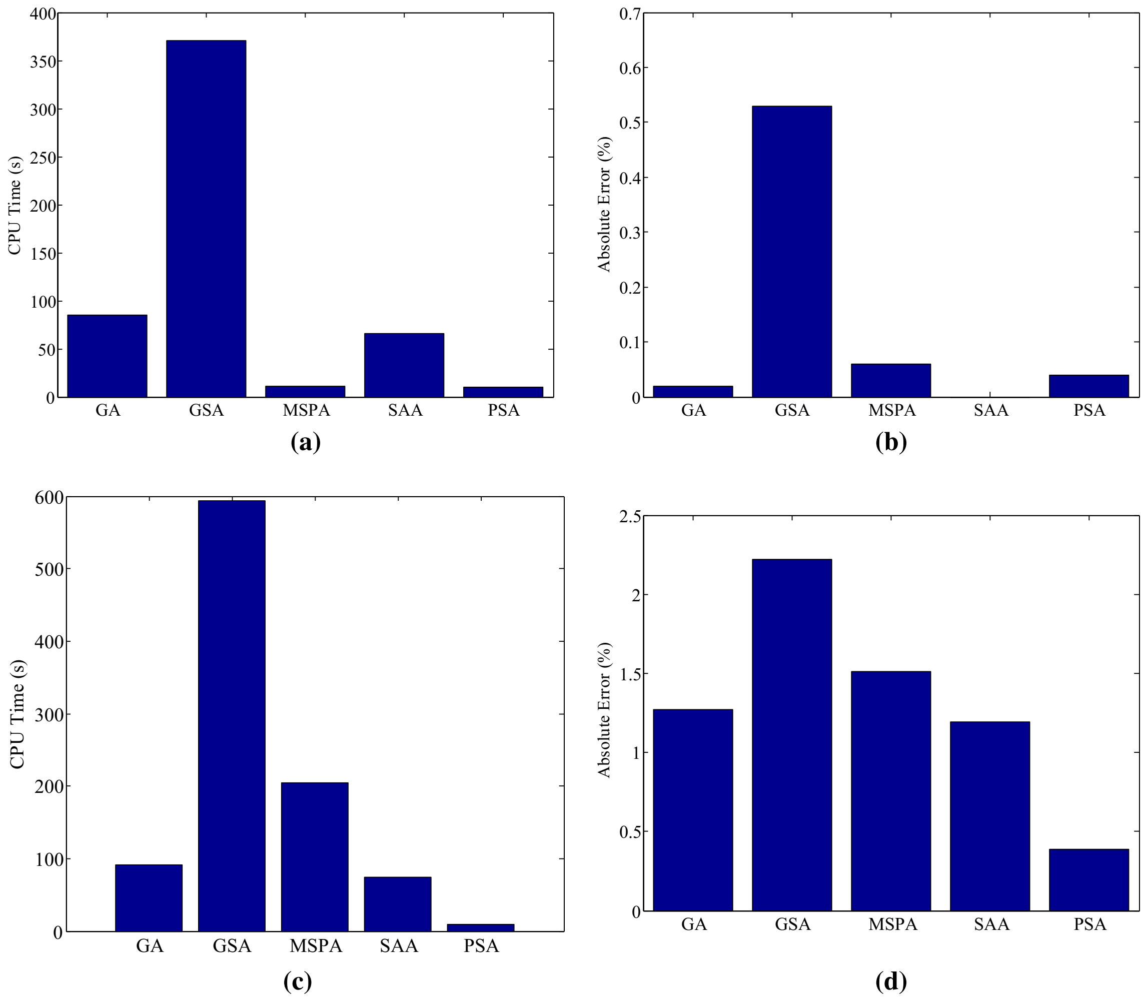


Fig. 22 Comparison of different SOMARBNNs in retrieval of A for different levels of noise in the input data **a** CPU time consumption for 0.0% noise, **b** error in retrieval of A for 0.1% noise, **c** CPU time consumption for 2.0% noise, **d** error in retrieval of A for 2.0% noise

Overall, all the SOMARBNNs are able to generate highly accurate temperature and value of *A,* corresponding to any velocity field, for different noise levels in the input data. GSAARBNN is found to be performing worst than other SOMARBNNs. PSAARBNN is found to be better than other SOMARBNNs in terms of CPU time consumption, accuracy, and ability to handle noisy input data.

References

1. Mai-Duy, N.; Tanner, R.I.: Computing non-Newtonian fluid flow with radial basis function networks. Int. J. Numer. Methods Fluids. **48**, 1309–1336 (2005)
2. Esfe, M.H.: Designing an artificial neural network using radial basis function (RBF-ANN) to model thermal conductivity of ethylene glycol–water-based TiO_2 nanofluids. J Therm Anal Calorim. **127**(3), 2125–2131 (2017)
3. Abualigah, L.; Diabat, A.: A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. Clust. Comput. **24**, 205–223 (2021)
4. Abualigah, L.; Diabat, A.; Mirjalili, S.; Elaziz, M.A.; Gandomi, A.H.: The arithmetic optimization algorithm. Comput. Methods Appl. Mech. Eng. (2021). https://doi.org/10.1016/j.cma.2020.113609
5. Karimi, H.; Yousefi, F.; Rahimi, M.R.: Correlation of viscosity in nanofluids using genetic algorithm-neural network (GA-NN). Heat Mass Transf. **47**, 1417–1425 (2011)
6. Li, S.J.; Liu, Y.X.; He, X.; Liu, Y.J.: Global search algorithm of minimum safety factor for slope stability analysis based on annealing simulation. Chinese J. Rock Mech. Eng. **22**(2), 236–240 (2003)
7. Miri, T.; Tsoukalas, A.; Bakalis, S.; Pistikopoulos, E.N.; Rustem, B.; Fryer, P.J.: Global optimization of process conditions in batch thermal sterilization of food. J. Food Eng. **87**, 485–494 (2008)
8. Saruhan, H.: Designing optimum oil thickness in artificial human knee joint by simulated annealing. Math. and Comp. Applic. **14**(12), 109–117 (2009)
9. Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**, 1–25 (1997)
10. Mishra, V.K.; Mishra, S.C.; Basu, D.N.: Simultaneous estimation of four parameters in a combined mode heat transfer in a 2-D rectangular porous matrix with heat generation. Numerical Heat Transfer- A. **71**(6), 677–692 (2017)
11. Mishra, V.K.; Mishra, S.C.; Basu, D.N.: Simultaneous estimation of properties in a combined mode conduction-radiation heat transfer in a porous medium. Heat Transf. Asian Res **45**(8), 699–713 (2016)
12. Mulani, U.K.; Talukdar, P.; Das, A.; Alagirusamy, R.: Performance analysis and feasibility study of ant colony optimization, particle swarm optimization and cuckoo search algorithms for inverse heat transfer problems. Int. J. Heat Mass Transf. **89**, 359–378 (2015)
13. Inaba, H.; Zhang, Y.; Horibe, A.; Haruki, N.: Numerical simulation of natural convection of latent heat phasechange-material microcapsulate slurry packed in a horizontal rectangular enclosure heated from below and cooled from above. Heat Mass Transfer. **43**, 459–470 (2017)
14. Subba Rao, A.; Ramachandra Prasad, V.; Rajendra, P.; Sasikala, M.; Anwar Beg, O.: Numerical study of non-Newtonian polymeric boundary layer flow and heat transfer from a permeable horizontal isothermal cylinder. Front. Heat Mass Transfer. **9**, 2 (2017)
15. Lim, H.; Back, S.M.; Hwang, M.H.; Lee, D.H.; Choi, H.; Nam, J.: Sheathless high-throughput circulating tumor cell separation using viscoelastic non-Newtonian fluid. Micromachines (Basel). **10**, 462 (2019)
16. Glinski, G.P.; Bailey, C.; Pericleous, K.A.: A non Newtonian of the stencil printing process. Proc. Inst. Mech. Eng. **215**(C4), 437–446 (2001)
17. Chaudhuri, S.; Das, P.K.: Semi-analytical solution of the heat transfer including viscous dissipation in the steady flow of a Sisko fuid in cylindrical tubes. J. Heat Transf (ASME). **140**, 071701 (2018)
18. Kalani, H.; Sardarabadi, M.; Passandideh-Fard, M.: Using artificial neural network models and particle swarm optimization for manner prediction of a photovoltaic thermal nanofluid based collector. Appl. Therm. Eng. **113**, 1170–1177 (2017)
19. Esfe, M.H.: Designing a neural network for predicting the heat transfer and pressure drop characteristics of Ag/water nanofluids in a heat exchanger. Appl. Therm. Eng. **126**, 559–565 (2017)
20. Daneshfar, R.; Bemani, A.; Hadipoor, M.; Sharifpur, M.; Ali, H.M.; Mahariq, I.; Abdeljawad, T.: Estimating the heat capacity of non-Newtonian ionanofluid systems using ANN, ANFIS, and SGB tree algorithms. Appl. Sci. **10**, 6432 (2020)
21. Esfe, M.H.: Designing an artificial neural network using radial basis function (RBF-ANN) to model thermal conductivity of ethylene glycol–water-based TiO_2 nanofluids. J. Therm. Anal. Calorim. **127**, 2125–2131 (2017). https://doi.org/10.1007/s10973-016-5725-y
22. Chhantyal, K.; Viumdal, H.; Mylvaganam, S.; Elseth, G.: Ultrasonic level sensors for flowmetering of non-Newtonian fluids in open Venturi channels: using data fusion based on artificial neural network and support vector machines. IEEE Sensors Applications Symposium (SAS), Catania. 1–6 (2016), https://doi.org/10.1109/SAS.2016.7479829
23. Eshgarf, H.; Sina, N.; Esfe, M.H., et al.: Prediction of rheological behavior of MWCNTs–SiO2/EG–water non-Newtonian hybrid nanofluid by designing new correlations and optimal artificial neural networks. J Therm Anal Calorim. **132**, 1029–1038 (2018). https://doi.org/10.1007/s10973-017-6895-y
24. Wu, H.; Bagherzadeh, S.A.; D'Orazio, A.; Habibollahi, N.; Karimipour, A.; Goodarzi, M.; Bach, Q.-V.: Present a new multi objective optimization statistical Pareto frontier method composed of artificial neural network and multi objective genetic algorithm to improve the pipe flow hydrodynamic and thermal properties such as pressure drop and heat transfer coefficient for non-Newtonian binary fluids. Phys. Stat. Mech. Appl. **535**, 122409 (2019)
25. Zhang, S.; Ge, Z.; Fan, X., et al.: Prediction method of thermal conductivity of nanofluids based on radial basis function. J. Therm. Anal. Calorim. **141**, 859–880 (2020). https://doi.org/10.1007/s10973-019-09067-x
26. Mishra, V.K.; Chaudhuri, S.: Genetic algorithm-assisted artificial neural network for retrieval of a parameter in a third grade fluid flow through two parallel and heated plates. Heat Transfer (Wiley). (2020). https://doi.org/10.1002/htj.21970
27. Amani, M.; Amani, P.; Bahiraei, M., et al.: Prediction of hydrothermal behavior of a non-Newtonian nanofluid in a square channel by modeling of thermophysical properties using neural network. J Therm Anal Calorim. **135**, 901–910 (2019). https://doi.org/10.1007/s10973-018-7303-y
28. Danish, M.; Kumar, S.; Kumar, S.: Exact analytical solutions for the Poiseuille and Couette-Poiseuille flow of third grade fluid between parallel plates. Commun. Non Linear Sci. **17**, 1089–1097 (2012)
29. Balaji, C.: Thermal system design and optimization, 2nd edn. Ane Books Pvt. Ltd., New Delhi (2019)
30. Zsolt, U.; Lasdon, L.; Plummer, J.; Glover, F.; Kelly, J.; Martí, R.: Scatter Search and Local NLP Solvers: a Multistart Framework

for Global Optimization. INFORMS J. Comput. **19**(3), 328–340 (2007)

31. Glover, F.: A template for scatter search and path relinking. In: Hao, J.K.; Lutton, E.; Ronald, E.; Schoenauer, M.; Snyers, D. (Eds.) Artificial Evolution. AE 1997. Lecture Notes in Computer Science. Springer Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0026589
32. Zhou, H.; Jiang, Z.; Li, W.; Wang, G.; Tu, Y.: Optimal design for an extruder head runner based on response surface method and simulated annealing algorithm. Int. J. Polym. Sci. **7239618**, 10 (2018). https://doi.org/10.1155/2018/7239618
33. Wu, C.; Wang, S.-S.; Jiang, X.; Li, J.: Thermodynamic analysis and performance optimization of transcritical power cycles using CO_2-based binary zeotropic mixtures as working fluids for geothermal power plants. Appl. Therm. Eng. **115**, 292–304 (2017)
34. Bar, N.; Bandyopadhyay, T.K.; Biswas, M.N.; Das, S.K.: Prediction of pressure drop using artificial neural network for non-Newtonian liquid flow through piping components. J. Pet. Sci. E. **71**, 187–194 (2010)