Learning of The Neural System Based on Nonlinear Aggregation Operations of Quaternionic-Valued Signals in Neuron Cell

Sushil Kumar[1] and B K Tripathi
Department of Computer Science & Engineering
Harcourt Butler Technical University, Kanpur, India

***Abstract*—**This paper illustrates the new structure of artificial neuron which is based on the non-linear aggregation operation of quaternionic-valued signals in neuron cell. The main aim of this neuron is to present the comparative potential capabilities of non-linear aggregation over conventional aggregation (summation) of the quaternionic-valued signals. The root-power mean is used as a nonlinear aggregation operation which has an ability to construct a wide spectrum of aggregation means in between minima and maxima depending on degree of compensation (exponent). It has a beautiful property of changing its degree of compensation in the natural way which emulates the various existing neuron models as its special cases. The three-layer multilayer perceptron networks are designed with proposed and conventional neurons in quaternionic domain separately. These networks are trained through quaternionic-valued back-propagation (ℍ-BP) learning algorithm for various benchmark problems. The results demonstrate the improvement on training cycles and exhibits better approximation accuracy. A wide spectrum of benchmark problems are considered to evaluate the performance of proposed quaternionic root-power mean neuron with ℍ-BP learning algorithm.

Keywords—Quaternion, quasi-arithematic means, quaternionic-valued root-power means, quaternionic multilayer perceptron, and quaternionic backpropagation.

I. Introduction

The information processing in cell body is an important functionality of a neuron, which emulates the computational power of a neuron [1]-[4]. In the last few years, various neurocomputing researches have confirmed the computational capability of a neuron with nonlinear aggregation operations on synaptic inputs [1], [2], [5]-[8] and presented various higher order neurons based on the nonlinear correlation among different impinging signals. These attempts resulted in various classes of neural structure as pi-sigma [9], [10] second order neuron [11], Compensatory neuron [12] and other higher order neurons [13]-[16]. However, the higher order neurons have proved to be efficient, but they face the problem of explosion of terms as the number of inputs increases hence demanding sparseness in representation. The problem worsens when neurons are implemented in high dimension. It is highly demanding to investigate a neuron model like conventional neuron in higher dimension, but is free from problem of higher order neurons. This paper presents a neuron model with complete specification for quaternionic-valued that employs the non-linear correlation among input components; but it is free from above problem even when there is an increase in the degree of approximation. The corresponding neural network with learning algorithm in quaternionic domain provides a better learning and generalization opportunity for problems in three or four dimensions. The weighted root-power mean covers the various classes of aggregation in the interval between minima to maxima operations [17], [18]. It provides the flexibility to approximate appropriate operation in the widest range of aggregation through variation of power coefficient. The weighted root-power mean as an aggregation function of the proposed neuron model with quaternionic-valued signals exhibits the natural and general model that presents the various existing neuron models as its special cases, depending on the domain of input signals and value of power coefficient.

The back-propagation (BP) learning algorithm has gained popularity due its simplicity in implementation. Therefore, some of other proposals have been given, like modified error function [19], [20], addition of variable learning rate [21], [22], addition of momentum [23], [24], delta-bar-delta algorithm [25], [26], and quick prop [27], to accelerate the convergence to a significant amount. The fast convergence with efficacious performance along with less complexity of neural network is the important matter for a variety of applications in higher dimensions. This paper presents a quaternionic-valued based backpropagation (ℍ-BP) learning algorithm of the network employed with proposed quaternionic-valued root-power mean neurons (ℍ-RPMN) in hidden and output layers.

The proposed quaternionic-valued root power mean neuron along with the derivation of learning rules through a bounded but non-holomorphic activation function is compared with conventional neuron in quaternionic domain which demonstrates the better generalization in simulation results. This paper is organized as follows. The section II presents a new structure of neuron in quaternionic domain which emulates many neuron models as its special cases. The BP weight update rules for a three layer network based on ℍ-RPMN are derived here. In section III, the comparative performance is given for a wide spectrum of benchmark problems. Section IV concludes the paper with future scope.

II. Neural Network and Learning

In variety of researches it is observed that the approximation capabilities of an artificial neuron are governed through spatial aggregation input signals [1], [2], [5]-[7]. This paper focuses on the design and assessment of a generalized artificial neuron whose aggregation operation provides wide spectrum of approximation in between minima to maxima.

A. Root Power Mean Neuron in Quaternionic Domain

The aggregation operation for proposed neuron is conceptually based on weighted root-power means of input signals which belongs to the family of quasi-arithmetic means [28], [29] and expressed as:

$$\mathcal{M}_\varphi(x_1,\dots,x_n;\omega_1,\dots,\omega_n)=\varphi^{-1}\left(\textstyle\sum_{i=1}^{n}\omega_i\,\varphi(x_i)\right) \quad (1)$$

where $\varphi:\mathbb{R}^n\to\mathbb{R}$ is a continuous and strictly monotonic function which is a generator of quasi-arithmetic mean ($\mathcal{M}_\varphi$). The φ^{-1} is the inverse of function $\varphi: x\to x^\alpha$, where $\alpha\in\mathbb{R}-\{0\}$, thus weight root-power mean [30] is defined as:

$$\mathcal{M}(x_1,x_2,\dots,x_n;\omega_1,\omega_2,\dots,\omega_n;\ \alpha)=\left(\textstyle\sum_{i=1}^{n}\omega_i\,x_i^\alpha\right)^{\frac{1}{\alpha}} \quad (2)$$

This function models a compensation operation to adjust the degree of approximation by varying the power coefficient α, from $-\infty$ (minimum) to $+\infty$ (maximum). The specific power coefficients determine various classical means. When $\alpha\to0$, then $\mathcal{M}$ converges to geometric mean and when $\alpha=-1,\ 1,\ 2$, then $\mathcal{M}$ acts as harmonic, arithmetic, and quadratic means, respectively. Most of the common averaging operations come under the family of root-power mean (2), which motivated us to define new aggregation function for a neuron in a quaternionic domain ($\mathbb{H}$). This neuron is named as $\mathbb{H}$-RPMN whose net potential (V) is defined as follows :

$$V(q_0,q_1,\dots,q_n;w_0,w_1,\dots,w_n;\ \alpha)=\left(\textstyle\sum_{i=0}^{n}w_i\otimes q_i^\alpha\right)^{\frac{1}{\alpha}} \quad (3)$$

where $V,\ w_i,\ q_i\in\mathbb{H}$ and $\alpha\in\mathbb{R}$. $\mathbb{H}$ and $\mathbb{R}$ are the set of quaternion and complex numbers respectively. The symbol $\otimes$ denotes the quaternion multiplication, which satisfies the Hamilton's properties as $\boldsymbol{i}^2=\boldsymbol{j}^2=\boldsymbol{k}^2=\boldsymbol{ijk}=-1$, $\boldsymbol{ij}=-\boldsymbol{ji}=\boldsymbol{k}$, $\boldsymbol{jk}=-\boldsymbol{kj}=\boldsymbol{i}$, and $\boldsymbol{ki}=-\boldsymbol{ik}=\boldsymbol{j}$ [31]. The net potential through quaternionic variables is further transformed through quaternionic-valued split activation function $f_{\mathbb{H}}$ ($f_{\mathbb{H}}:\mathbb{H}\to\mathbb{H}$). This presents the four dimensional extension with the suitable real activation function f ('split-quaternion') [32]. This idea has been motivated from split-type action function in complex domain whose approximation capabilities have been thoroughly justified in [12], [30], [33]. The split-type activation function in quaternionic domain is bounded but non-holomorphic since Cauchy–Riemann-Fueter (CRF) condition does not hold for it [34]. Let $V=\Re(V)+\Im_1(V)\boldsymbol{i}+\Im_2(V)\boldsymbol{j}+\Im_3(V)\boldsymbol{k}$ is a quaternionic variable, then $f_{\mathbb{H}}$ is defined as :

$$f_{\mathbb{H}}(V)=f(\Re(V))+f(\Im_1(V))\boldsymbol{i}+f(\Im_2(V))\boldsymbol{j}+f(\Im_3(V))\boldsymbol{k} \quad (4)$$

where, $\Re$, $\Im_1$, $\Im_2$ and $\Im_3$ denote a real and other three imaginary components of a quaternionic variable respectively. The various types of non-linear real valued activation functions f have been defined in literatures. The choice of a suitable activation function depends on the intended applications. The output of proposed $\mathbb{H}$-RPMN can be defined as :

$$Y(q_0,q_1,\dots,q_n;w_0,w_1,\dots,w_n;\alpha)=f_{\mathbb{H}}\left(\left(\textstyle\sum_{i=0}^{n}w_i\otimes q_i^\alpha\right)^{\frac{1}{\alpha}}\right) \quad (5)$$

where, the power coefficient α provides wide ranging functionality of $\mathbb{H}$-RPMN. The many existing neuron models can be realized as special cases of (5) by substituting the specific value of power coefficient α. Some special type of existing neuron models are extracted from (5) as follows :

On substituting $\alpha=1$ in (5), then the output of neuron

$$Y(q_0,q_1,\dots,q_n;w_0,w_1,\dots,w_n)=f_{\mathbb{H}}\left(\textstyle\sum_{i=0}^{n}w_i\otimes q_i\right) \quad (6)$$

which is conventional type of neuron model proposed in the real [35], complex [33], [36] and quaternionic domain [37] as cases apply in consideration of imaginary components. Let consider all parameters in complex domain, then the output of a neuron

$$Y(z_0,z_1,\dots,z_n;w_0,w_1,\dots,w_n;\ \alpha)=f_{\mathbb{C}}\left(\left(\textstyle\sum_{i=0}^{n}w_i\,z_i^\alpha\right)^{\frac{1}{\alpha}}\right) \quad (7)$$

which presents the $\mathbb{C}$-RPMN model in complex domain, whose functional capabilities are investigated in [30].

The multiplicative neuron model proposed in [38], whose capability has been proven there, can be realized by (5) when $\alpha\to0$

$$Y(x_0,x_1,\dots,x_n;w_0,w_1,\dots,w_n)=f\left(\lim_{\alpha\to0}\left(\textstyle\sum_{i=0}^{n}w_i\,x_i^\alpha\right)^{\frac{1}{\alpha}}\right)=f\left(\textstyle\prod_{i=0}^{n}x_i^{w_i}\right) \quad (8)$$

The harmonic neuron model proposed in [39] can be obtained when $\alpha=-1$ in (5), thus output of neuron :

$$Y(x_0,x_1,\dots,x_n;w_0,w_1,\dots,w_n)=f\left(\frac{1}{\sum_{i=0}^{n}\frac{w_i}{x_i}}\right) \quad (9)$$

Similarly, when $\alpha=2$ and all variables are in real domain, then the output from (5) is conceptually similar to the quadratic neuron model proposed in [40] and expressed as follows :

$$Y(x_0,x_1,..,x_n;w_0,w_1..,w_n;\ \alpha=2)=f\left(\left(\textstyle\sum_{i=0}^{n}w_i\,x_i^2\right)^{\frac{1}{2}}\right) \quad (10)$$

B. *Learning in $\mathbb{H}$-RPMN Network*

The multilayer network with proposed neuron model is similar to the network of conventional neuron. Consider a three-layer ($L-M-N$) network of $\mathbb{H}$-RPMN where first layer possesses $L(l=1,\dots,L)$ inputs, and second and third layer contain $M(l=1,\dots,M)$ and $N(l=1,\dots,N)$ neurons respectively. All weights, biases, and input-output signals are quaternionic numbers. Let $Q=[q_1,q_2,\dots,q_L]$ be the vector of quaternionic input $q=\Re(q)+\Im_1(q)\boldsymbol{i}+\Im_2(q)\boldsymbol{j}+\Im_3(q)\boldsymbol{k}$ and $\bar{q}=\Re(q)-\Im_1(q)\boldsymbol{i}-\Im_2(q)\boldsymbol{j}-\Im_3(q)\boldsymbol{k}$ be the conjugate. Let f be the real-valued activation function and f' be its derivative and $\eta\in[0,1]$ be the learning rate. Let weight w_{lm} be from lth input to mth hidden neuron and w_{mn} be from mth hidden neuron to nth output neuron of the network. Let w_0 be the bias weight and q_0 be the bias input. Let V be net internal potential of a neuron and output can be computed through the split-quaternion activation function $f_{\mathbb{H}}$. For mth hidden neuron $V_m=\Re(V_m)+\Im_1(V_m)\boldsymbol{i}+\Im_2(V_m)\boldsymbol{j}+\Im_3(V_m)\boldsymbol{k}$, then output :

$$Y_m=f_{\mathbb{H}}(V_m)=f_{\mathbb{H}}\left(\left(\textstyle\sum_{l=0}^{L}w_{lm}\otimes q_l^\alpha\right)^{\frac{1}{\alpha}}\right) \quad (11)$$

Let $u_m=\sum_{l=0}^{L}w_{lm}\otimes q_l^\alpha$ and V_m is expressed as power of quaternionic variable $u_m=\Re(u_m)+\Im_1(u_m)\boldsymbol{i}+\Im_2(u_m)\boldsymbol{j}+\Im_3(u_m)\boldsymbol{k}$. Let $\vec{V}_{(u_m)}=\Im_1(u_m)\boldsymbol{i}+\Im_2(u_m)\boldsymbol{j}+\Im_3(u_m)\boldsymbol{k}$ is the vector component of u_m then u_m can be expressed as a real and its vector part together as $\Re(u_m)+\vec{V}_{(u_m)}$ and it can be represented in the polar form [41] as :

$$\|u_m\|\left(Cos(\theta_{(u_m)})+\frac{\vec{V}_{(u_m)}}{\|\vec{V}_{(u_m)}\|}Sin(\theta_{(u_m)})\right),$$

where,

$$\theta_{(u_m)}=\tan^{-1}\left(\|\vec{V}_{(u_m)}\|/\Re(u_m)\right),$$

the norm of quaternionic variable u_m is $\|u_m\|=\left((\Re(u_m))^2+\right.$ *ℑ1um2+ℑ2um2+ℑ3um212,* and

$\|\vec{V}_{(u_m)}\|=\left((\Im_1(u_m))^2+(\Im_2(u_m))^2+(\Im_3(u_m))^2\right)^{\frac{1}{2}}$, which denotes the magnitude of the vector component ($\vec{V}_{(u_m)}$) of u_m. The net internal potential V_m of mth hidden neuron can be obtained by applying De Moivre's theorem on u_m [42] as :

$$V_m = \|u_m\|^{\frac{1}{\alpha}} \left(Cos\left(\frac{\theta_{(u_m)}}{\alpha}\right) + \frac{\vec{V}_{(u_m)}}{\|\vec{V}_{(u_m)}\|} Sin\left(\frac{\theta_{(u_m)}}{\alpha}\right) \right) \quad (12)$$

The output of the nth neuron in output layer can be expressed as similar to (11) as :

$$Y_n = f_{\mathbb{H}}(V_n) = f_{\mathbb{H}}\left(\left(\sum_{m=0}^{M} w_{mn} \otimes Y_m^{\alpha}\right)^{\frac{1}{\alpha}} \right) \quad (13)$$

Let $u_n = \sum_{m=0}^{M} w_{mn} \otimes Y_m^{\alpha}$ and V_n be the net internal potential of nth neuron which can be derived as similar to (12)

$$V_n = \|u_n\|^{\frac{1}{\alpha}} \left(Cos\left(\frac{\theta_{(u_n)}}{\alpha}\right) + \frac{\vec{V}_{(u_n)}}{\|\vec{V}_{(u_n)}\|} Sin\left(\frac{\theta_{(u_n)}}{\alpha}\right) \right) \quad (14)$$

The gradient-descent-based error back-propagation learning scheme for feed-forward neural network has been extended in quaternionic domain. Let error $e_n = D_n - Y_n = \Re(e_n) + \Im_1(e_n)\boldsymbol{i} + \Im_2(e_n)\boldsymbol{j} + \Im_3(e_n)\boldsymbol{k}$ be the difference between desired (D_n) and actual (Y_n) output of nth neuron at output layer. The weight update formula can be derived by minimizing the real-valued error function (E) as follows :

$$E = \frac{1}{2N}\sum_{n=1}^{N}\|e_n\|^2 = \frac{1}{2N}\sum_{n=1}^{N}\left\{ \left(\Re(e_n)\right)^2 + \left(\Im_1(e_n)\right)^2 + \Im 2en2 + \Im 3en2 \right. \quad (15)$$

The real-valued error function (E) does not follow the Cauchy-Riemann condition, therefore it is not holomorphic. The error function is minimized by recursively altering the weight coefficients as :

$$w^{new} = w^{old} - \eta\, \nabla_w(E) \quad (16)$$

where, $\nabla_w(E)$ presents the gradient of the error function (E) which is derived with respect to real and other three imaginary components of quaternionic weights. The weight update (Δw) is proportional to the negative gradient of the error function with respect to quaternionic weight as :

$$\Delta w = -\eta\, \nabla_w(E) = -\eta\left(\frac{\partial E}{\partial \Re(w)} + \frac{\partial E}{\partial \Im_1(w)}\boldsymbol{i} + \frac{\partial E}{\partial \Im_2(w)}\boldsymbol{j} + \partial E \partial \Im 3w\boldsymbol{k} \right. \quad (17)$$

For the weight $(w = w_{mn})$ that connects mth hidden neuron to nth output neuron, the weight update is obtained using chain rule of derivation as :

$$\Delta w_{mn} = -\eta\, \nabla_{w_{mn}}(E) = \frac{\eta}{N}\left\{ \Re(e_n) f'\left(\Re(V_n)\right) \nabla_{w_{mn}}\left(\Re(V_n)\right) + i{=}13\Im ienf'\Im iVn\ \nabla wmn\Im iVn \right. \quad (18)$$

The weight update (Δw_{mn}) depends on gradients of each component of the net potential (V_n) of nth output neuron with respect to weight (w_{mn}). The gradient of the real component of the net potential can be obtained using real part of (14) e.g. $\|u_n\|^{\frac{1}{\alpha}} Cos\left(\frac{\theta_{(u_n)}}{\alpha}\right)$ as :

$$\nabla_{w_{mn}}\left(\Re(V_n)\right) = \frac{\|u_n\|^{\frac{1}{\alpha}-2}}{\alpha}\left\{ A_1 \Re(u_n) \nabla_{w_{mn}}\left(\Re(u_n)\right) + B1j{=}13\Im jun\ \nabla wmn\Im jun \right. \quad (19)$$

and the gradients of three imaginary components of the net potential $(\Im_i(V_n), i = 1,\ 2,\ 3)$ can be obtained as :

$$\nabla_{w_{mn}}\left(\Im_i(V_n)\right) = \frac{\|u_n\|^{\frac{1}{\alpha}-2}}{\alpha\|\vec{V}_{(u_n)}\|}\Im_i(u_n)\left\{ A_2 \Re(u_n) \nabla_{w_{mn}}\left(\Re(u_n)\right) + B_2\sum_{j=1}^{3}\Im_j(u_n)\, \nabla_{w_{mn}}\left(\Im_j(u_n)\right) \right\} + \frac{\|u_n\|^{\frac{1}{\alpha}}}{\|\vec{V}_{(u_n)}\|} Sin\left(\frac{\theta_{(u_n)}}{\alpha}\right) \nabla_{w_{mn}}\left(\Im_i(u_n)\right) - \frac{\|u_n\|^{\frac{1}{\alpha}}}{\|\vec{V}_{(u_n)}\|^3}\Im_i(u_n) Sin\left(\frac{\theta_{(u_n)}}{\alpha}\right)\left\{ \sum_{j=1}^{3}\Im_j(u_n)\nabla_{w_{mn}}\left(\Im_j(u_n)\right) \right\} \quad (20)$$

where,

$\|u_n\| = \left(\left(\Re(u_n)\right)^2 + \left(\Im_1(u_n)\right)^2 + \left(\Im_2(u_n)\right)^2 + \Im 3un212, \right.$

$\|\vec{V}_{(u_n)}\| = \left(\left(\Im_1(u_n)\right)^2 + \left(\Im_2(u_n)\right)^2 + \left(\Im_3(u_n)\right)^2 \right)^{\frac{1}{2}}$,

$A_1 = Cos((1 - 1/\alpha)\theta_{(u_n)})/Cos(\theta_{(u_n)})$,

$B_1 = Sin((1 - 1/\alpha)\theta_{(u_n)})/Sin(\theta_{(u_n)})$,

$A_2 = -Sin((1 - 1/\alpha)\theta_{(u_n)})/Cos(\theta_{(u_n)})$,

$B_2 = Cos((1 - 1/\alpha)\theta_{(u_n)})/Sin(\theta_{(u_n)})$, and

$\theta_{(u_n)} = \tan^{-1}\left(\|\vec{V}_{(u_n)}\|/\Re(u_n)\right)$.

The gradients of real and three imaginary components of u_n with respect to the weight (w_{mn}) can be obtained as :

$$\nabla_{w_{mn}}\left(\Re(u_n)\right) = \left(\overline{Y}_m\right)^{\alpha}, \quad (21)$$

$$\nabla_{w_{mn}}\left(\Im_1(u_n)\right) = \boldsymbol{i} \otimes \left(\overline{Y}_m\right)^{\alpha}, \quad (22)$$

$$\nabla_{w_{mn}}\left(\Im_2(u_n)\right) = \boldsymbol{j} \otimes \left(\overline{Y}_m\right)^{\alpha}, \quad (23)$$

$$\nabla_{w_{mn}}\left(\Im_3(u_n)\right) = \boldsymbol{k} \otimes \left(\overline{Y}_m\right)^{\alpha}. \quad (24)$$

For the weight $(w = w_{lm})$ that connects lth input to mth hidden neuron

$$\Delta w_{lm} = -\eta\, \nabla_{w_{lm}}(E) = \frac{\eta}{N}\sum_{n=1}^{N}\left\{ \Re(e_n) f'\left(\Re(V_n)\right)\nabla_{w_{lm}}\left(\Re(V_n)\right) + i{=}13\Im ienf'\Im iVn\nabla wlm\Im iVn \right. \quad (25)$$

The weight update (Δw_{lm}) between lth input and mth hidden neuron depends on gradients of each component of the net potential (V_n) of nth output neuron with respect to weight (w_{lm}). The gradient of the real component of the net potential can be obtained as :

$$\nabla_{w_{lm}}\left(\Re(V_n)\right) = \frac{\|u_n\|^{\frac{1}{\alpha}-2}}{\alpha}\left\{ A_1 \Re(u_n)\nabla_{w_{lm}}\left(\Re(u_n)\right) + B1j{=}13\Im jun\ \nabla wlm\Im jun \right. \quad (26)$$

and the gradients of three imaginary components of the net potential $(\Im_i(V_n), i = 1, 2, 3)$ can be obtained as :

$$\nabla_{w_{lm}}\left(\Im_i(V_n)\right) = \frac{\|u_n\|^{\frac{1}{\alpha}-2}}{\alpha\|\vec{V}_{(u_n)}\|}\Im_i(u_n)\left\{ A_2 \Re(u_n)\nabla_{w_{lm}}\left(\Re(u_n)\right) + B_2\sum_{j=1}^{3}\Im_j(u_n)\, \nabla_{w_{lm}}\left(\Im_j(u_n)\right) \right\} + \frac{\|u_n\|^{\frac{1}{\alpha}}}{\|\vec{V}_{(u_n)}\|} Sin\left(\frac{\theta_{(u_n)}}{\alpha}\right)\nabla_{w_{lm}}\left(\Im_i(u_n)\right) - \frac{\|u_n\|^{\frac{1}{\alpha}}}{\|\vec{V}_{(u_n)}\|^3}\Im_i(u_n) Sin\left(\frac{\theta_{(u_n)}}{\alpha}\right)\left\{ \sum_{j=1}^{3}\Im_j(u_n)\nabla_{w_{lm}}\left(\Im_j(u_n)\right) \right\} \quad (27)$$

The gradients of real and three imaginary components of u_n with respect to the weight (w_{lm}) can be obtained as :

$$\nabla_{w_{lm}}\left(\Re(u_n)\right) = \Re(w_{mn})\,\nabla_{w_{lm}}\left(\Re(Y_m^\alpha)\right) - \Im_1(w_{mn})\nabla_{w_{lm}}\left(\Im_1(Y_m^\alpha)\right) - \Im_2(w_{mn})\,\nabla_{w_{lm}}\left(\Im_2(Y_m^\alpha)\right) - \Im_3(w_{mn})\,\nabla_{w_{lm}}\left(\Im_3(Y_m^\alpha)\right) \quad (28)$$

$$\nabla_{w_{lm}}\left(\Im_1(u_n)\right) = \Re(w_{mn})\,\nabla_{w_{lm}}\left(\Im_1(Y_m^\alpha)\right) + \Im_1(w_{mn})\,\nabla_{w_{lm}}\left(\Re(Y_m^\alpha)\right) + \Im_2(w_{mn})\,\nabla_{w_{lm}}\left(\Im_3(Y_m^\alpha)\right) - \Im_3(w_{mn})\,\nabla_{w_{lm}}\left(\Im_2(Y_m^\alpha)\right) \quad (29)$$

$$\nabla_{w_{lm}}\left(\Im_2(u_n)\right) = \Re(w_{mn})\,\nabla_{w_{lm}}\left(\Im_2(Y_m^\alpha)\right) - \Im_1(w_{mn})\nabla_{w_{lm}}\left(\Im_3(Y_m^\alpha)\right) + \Im_2(w_{mn})\,\nabla_{w_{lm}}\left(\Re(Y_m^\alpha)\right) + \Im_3(w_{mn})\nabla_{w_{lm}}\left(\Im_1(Y_m^\alpha)\right) \quad (30)$$

$$\nabla_{w_{lm}}\left(\Im_3(u_n)\right) = \Re(w_{mn})\nabla_{w_{lm}}\left(\Im_3(Y_m^\alpha)\right) + \Im_1(w_{mn})\,\nabla_{w_{lm}}\left(\Im_2(Y_m^\alpha)\right) - \Im_2(w_{mn})\,\nabla_{w_{lm}}\left(\Im_1(Y_m^\alpha)\right) + \Im_3(w_{mn})\,\nabla_{w_{lm}}\left(\Re(Y_m^\alpha)\right) \quad (31)$$

The gradient of real component of Y_m^α can be obtained as :

$$\nabla_{w_{lm}}\left(\Re(Y_m^\alpha)\right) = \alpha\|Y_m\|^{\alpha-2}\left\{A_3\Re(Y_m)f'\left(\Re(V_m)\right)\nabla_{w_{lm}}\left(\Re(V_m)\right) + \right.$$
$B3j=13\Im jYmf'\Im jVm\nabla wlm\Im jVm$ (32)

and the gradients of three imaginary components of the net potential ($\Im_i(Y_m^\alpha), i = 1,2,3$) can be obtained as :

$$\nabla_{w_{mn}}\left(\Im_i(Y_m^\alpha)\right) = \frac{\alpha\|Y_m\|^{\alpha-2}}{\|\vec{V}_{(Y_m)}\|}\Im_i(Y_m)\left\{A_4\Re(Y_m)f'\left(\Re(V_m)\right)\nabla_{w_{lm}}\left(\Re(V_m)\right) + B_4\sum_{j=1}^{3}\Im_j(Y_m)f'\left(\Im_j(V_m)\right)\nabla_{w_{lm}}\left(\Im_j(V_m)\right)\right\} + \frac{\|Y_m\|^\alpha}{\|\vec{V}_{(Y_m)}\|}Sin\left(\alpha\theta_{(Y_m)}\right)f'\left(\Im_i(V_m)\right)\nabla_{w_{lm}}\left(\Im_i(V_m)\right) - \frac{\|Y_m\|^\alpha}{\|\vec{V}_{(Y_m)}\|^3}\Im_i(Y_m)Sin\left(\alpha\theta_{(Y_m)}\right)\times\left\{\sum_{j=1}^{3}\Im_j(Y_m)f'\left(\Im_j(V_m)\right)\nabla_{w_{lm}}\left(\Im_j(V_m)\right)\right\} \quad (33)$$

where,

$\|Y_m\| = \left(\left(\Re(Y_m)\right)^2 + \left(\Im_1(Y_m)\right)^2 + \left(\Im_2(Y_m)\right)^2 + \right.$
$\Im 3Ym212,$

$\|\vec{V}_{(Y_m)}\| = \left(\left(\Im_1(Y_m)\right)^2 + \left(\Im_2(Y_m)\right)^2 + \left(\Im_3(Y_m)\right)^2\right)^{\frac{1}{2}},$

$A_3 = Cos((1-\alpha)\theta_{(Y_m)})/Cos(\theta_{(Y_m)}),$

$B_3 = Sin((1-\alpha)\theta_{(Y_m)})/Sin(\theta_{(Y_m)}),$

$A_4 = -Sin((1-\alpha)\theta_{(Y_m)})/Cos(\theta_{(Y_m)}),$

$B_4 = Cos((1-\alpha)\theta_{(Y_m)})/Sin(\theta_{(Y_m)})$, and

$\theta_{(Y_m)} = \tan^{-1}\left(\|\vec{V}_{(Y_m)}\|/\Re(Y_m)\right).$

The gradient of real component of the net potential (V_m) can be obtained as :

$$\nabla_{w_{lm}}\left(\Re(V_m)\right) = \frac{\|u_m\|^{\frac{1}{\alpha}-2}}{\alpha}\left\{A_5\Re(u_m)\,\nabla_{w_{lm}}\left(\Re(u_m)\right) + \right.$$
$B5j=13\Im jum\ \nabla lm\Im jum$ (34)

and the gradients of three imaginary components of the net potential ($\Im_i(V_m), i = 1,2,3$) can be obtained as :

$$\nabla_{w_{lm}}\left(\Im_i(V_m)\right) = \frac{\|u_m\|^{\frac{1}{\alpha}-2}}{\alpha\|\vec{V}_{(u_m)}\|}\Im_i(u_m)\left\{A_6\Re(u_m)\,\nabla_{w_{lm}}\left(\Re(u_m)\right) + B_6\sum_{j=1}^{3}\Im_j(u_m)\,\nabla_{w_{lm}}\left(\Im_j(u_m)\right)\right\} + \frac{\|u_m\|^{\frac{1}{\alpha}}}{\|\vec{V}_{(u_m)}\|}Sin\left(\frac{\theta_{(u_m)}}{\alpha}\right)\nabla_{w_{lm}}\left(\Im_i(u_m)\right) - \frac{\|u_m\|^{\frac{1}{\alpha}}}{\|\vec{V}_{(u_m)}\|^3}\Im_i(u_m)Sin\left(\frac{\theta_{(u_m)}}{\alpha}\right)\left\{\sum_{j=1}^{3}\Im_j(u_m)\,\nabla_{w_{lm}}\left(\Im_j(u_m)\right)\right\} \quad (35)$$

where,

$\|u_m\| = \left(\left(\Re(u_m)\right)^2 + \left(\Im_1(u_m)\right)^2 + \left(\Im_2(u_m)\right)^2 + \right.$
$\Im 3um212,$

$\|\vec{V}_{(u_m)}\| = \left(\left(\Im_1(u_m)\right)^2 + \left(\Im_2(u_m)\right)^2 + \left(\Im_3(u_m)\right)^2\right)^{\frac{1}{2}},$

$A_5 = Cos((1-1/\alpha)\theta_{(u_m)})/Cos(\theta_{(u_m)}),$

$B_5 = Sin((1-1/\alpha)\theta_{(u_m)})/Sin(\theta_{(u_m)}),$

$A_6 = -Sin((1-1/\alpha)\theta_{(u_m)})/Cos(\theta_{(u_m)}),$

$B_6 = Cos((1-1/\alpha)\theta_{(u_m)})/Sin(\theta_{(u_m)})$, and

$\theta_{(u_m)} = \tan^{-1}\left(\|\vec{V}_{(u_m)}\|/\Re(u_m)\right).$

The gradients of real and three imaginary components of u_m with respect to the weight (w_{lm}) can be obtained as :

$$\nabla_{w_{lm}}\left(\Re(u_m)\right) = \left(\overline{q}_l\right)^\alpha. \quad (36)$$

$$\nabla_{w_{lm}}\left(\Im_1(u_m)\right) = \boldsymbol{i}\otimes\left(\overline{q}_l\right)^\alpha. \quad (37)$$

$$\nabla_{w_{lm}}\left(\Im_2(u_m)\right) = \boldsymbol{j}\otimes\left(\overline{q}_l\right)^\alpha. \quad (38)$$

$$\nabla_{w_{lm}}\left(\Im_3(u_m)\right) = \boldsymbol{k}\otimes\left(\overline{q}_l\right)^\alpha. \quad (39)$$

III. Performance Evaluation through Benchmark Problems

In this paper, we present the effectiveness of proposed neuron ℍ-RPMN and algorithm ℍ-RPMN through wide spectrum of benchmark problems. Comparative evaluation is done by networks designed by conventional neurons (MLP) and root-power-mean neurons with ℍ-BP learning algorithms in quaternionic domain. Four components of all quaternionic weights and biases for both networks are randomly initialized in the range -1 to 1. The quaternionic varaiable $\boldsymbol{q}_0 = 1 + \boldsymbol{i} + \boldsymbol{j} + \boldsymbol{k}$ is assumed as bias input and the hyperbolic tangent function is used as activation function. The comparison of training and testing performance through function approximation is thoroughly evaluated by statistical parameters like error variance, correlation, and AIC [43]. Another class of benchmark problem is the learning of linear transformations (rotation, scaling, and translation and their combinations) through a set of points lying on line whose generalization abilities are tested over complicated 3D geometric structure. In last subsection, two primary experiments are presented for 3D face recognition which surely it will be stepping stone for prospective researchers.

A. Function Approximations

1) Approximation of Model for Spread of Tuberculosis: The spread of tuberculosis model [44] is the system of four differential equations with respect to time along with four variables that can be denoted together as a quaternion number instead of four real numbers. This model needs the intelligent behavior and automated analysis of bacterial effect to reach equilibrium from different initial conditions. A mathematical model aims to analyze the effect of accumulation of bacteria which survive due to conducive ecological factors such as flower pots, plants, grasses, human clothes, etc. in the habitat, acting as reservoir, on the spread of tuberculosis (TB) in human population. The total population ($N(t)$) is categorized into classes, susceptible ($S(t)$) and infective ($T(t)$). $B(t)$ governs the bacteria density in the environment and $E(t)$ is cumulative density of ecological factors which is conducive to the accumulation of bacteria population. All these factors of this model are treated as quaternion number ($S(t)+T(t)\boldsymbol{i}+B(t)\boldsymbol{j}+E(t)\boldsymbol{k}$) with respect to time. In this model, it is assumed that TB is spread by direct contacts with infective in the population and indirect contacts with bacteria which is emitted by infective in the habitat. The dynamics of the spread of TB is governed by the system of nonlinear differential equations as :

$$
\begin{aligned}
\frac{dS}{dt} &= A-\beta ST-\lambda SB-dS+\nu T,\\
\frac{dT}{dt} &= \beta ST+\lambda SB-(\nu+\alpha+d)T,\\
\frac{dB}{dt} &= sT-s_0B+s_1BE,\\
\frac{dE}{dt} &= \gamma E-\gamma_0E^2+\gamma_1(S+T)E,\\
&S(0)>0,\, T(0)\geq 0,\, B(0)\geq 0,\, E(0)\geq 0
\end{aligned}
\qquad (40)
$$

where, A is the immigration rate of susceptibles. β and λ are the transmission coefficients of TB by contact of susceptibles with infectives and by inhalation of bacteria from environment; d is the natural death rate, ν is the therapeutic treatment rate of infected individuals and α is the death rate due to TB infection. The parameter s is the release rate of bacteria from the TB infected individuals, s_0 is the decrease coefficient due to natural factors and s_1 is the rate of survival and accumulation of bacteria population due to conducive ecological factors in the habitat. γ is the growth rate, γ/γ_0 is the carrying capacity in the habitat and γ_1 is the increase coefficient due to total human population. All these parameters in this model are assumed only for positive values. The TB model presented in (40) with parameters $A=500$, $\beta=0.0003$, $\lambda=0.0001$, $d=0.15$,$\nu=0.01$,$\alpha=0.2$, $s=0.1$, $s_0=0.3$, $s_1=0.0001$, $\gamma=25$, $\gamma_0=0.1$, and $\gamma_1=0.002$ generates four dataset containing 200 data points in 100 hours with time interval $\Delta t=0.5$ and four initial conditions are given as :

(i) $S(0)=1400, T(0)=1600, B(0)=358,E(0)=290$
(ii) $S(0)=200, T(0)=1300, B(0)=358, E(0)=290$
(iii) $S(0)=2600, T(0)=400, B(0)=358, E(0)=290$
(iv) $S(0)=600, T(0)=400, B(0)=358, E(0)=290$

The system presented in (40) reaches an equilibrium point $S(100)=1068.5313355$,$T(100)=970.6294275$, $B(100)=358.2693836$, and $E(100)=290.7832153$ from given four different initial conditions given (i), (ii), (iii) and (iv). With the help of each equilibrium point with its parameters as given above, the system generates four datasets containing four components ($S(t)$, $T(t)$, $B(t)$, and $E(t)$). All four datasets are further normalized between -0.9 to 0.9 and its 80 data points are used for training by ℍ-BP (learning rate η =0.001). The normalized datasets containing 200 points are used for testing of networks trained by both algorithms.

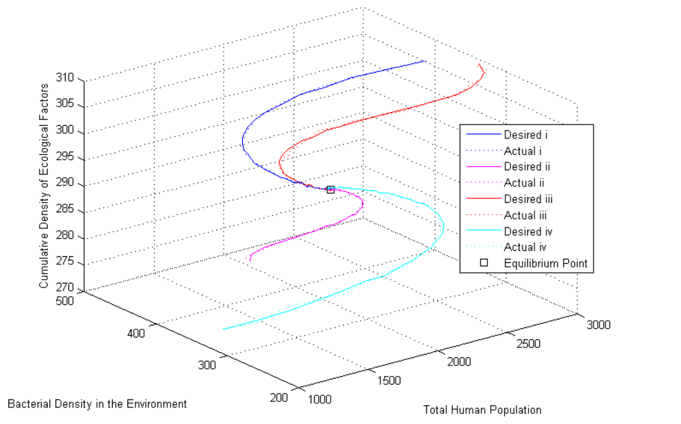


Fig. 1. Comparison of testing result of ℍ-RPMN based network trained by ℍ-BP algorithm with normalized desired.

TABLE I
COMPARISON OF TRAINING AND TESTING PERFORMANCE FOR THE SPREAD OF TB MODEL

	ℍ-MLP	ℍ-RPMN (α =0.94)
Learning Algorithm	ℍ-BP(η=0.001)	ℍ-BP(η=0.001)
Network	4-8-4	4-8-4
Parameters	76	76
MSETraining	0.0005	0.0004
Learning Cycles	6000	4000
MSETesting	3.4615e-04	2.9362e-04
Error Variance	5.8870e-04	5.0581e-04
Correlation	0.9965	0.9970
AIC	-6.6409	-7.0640

The comparative analysis for training and testing data through conventional and proposed neuron based networks performed by ℍ-BP algorithm are presented in Table I. It clearly shows that ℍ-BP with ℍ-RPMN has significantly faster convergence and has better testing results in terms of error, variance, correlation, and AIC. Fig. 1 demonstrates the testing results by ℍ-BP with ℍ-RPMN and compares with desired result in 3D for different initial conditions, where normalized total human population ($S(t)+T(t)$), bacterial population density ($B(t)$) and cumulative population density ($E(t)$) are in x, y and z direction respectively. The overall training and testing performance infer the superiority of ℍ-BP algorithm with ℍ-RPMN over quaternionic-valued conventional neuron.

2) Approximation of Vector Operations: A quaternion number correspond to radius vector from origin to the point in space. The interpretation of vectors and operations over them has the important and valuable impact on the analysis of various geometrical relationships in space. The quaternion addition (QADD), subtraction (QSUB), multiplication (QMULT) and normalized division (N-QDIV) as basic algebraic operations facilitate to define different vector operations in the four dimensional space. The learning and generalization capability of a single network for the above four operations is evaluated in this experiment.

TABLE II
COMPARISON OF TRAINING AND TESTING PERFORMANCE FOR QUATERNION OPERATIONS

	ℍ-MLP	ℍ-RPMN (α =0.9)
Algorithm	ℍ-BP(η=0.001)	ℍ-BP(η=0.001)
Network	2-20-4	2-20-4
Parameters	144	144
MSE training	0.0009	0.0008

Average Learning Cycles		25000	15000
MSE(Testing)	QADD	0.0008	0.0006
	QSUB	0.0008	0.0007
	QMULT	0.0035	0.0028
	N-QDIV	0.0064	0.0040
Error Variance	QADD	0.0008	0.0008
	QSUB	0.0009	0.0007
	QMULT	0.0046	0.0027
	N-QDIV	0.0072	0.0041
Correlation	QADD	0.9985	0.9981
	QSUB	0.9987	0.9972
	QMULT	0.2836	0.6178
	N-QDIV	0.1794	0.5381
AIC		-6.12	-6.28

Let $q_1 = \|q_1\| e^{\hat{n}_1 \theta_1}$ and $q_2 = \|q_2\| e^{\hat{n}_2 \theta_2}$ be the two quaternion numbers presented in polar form, where $-\pi \leq \theta_1, \theta_2 \leq \pi$, $0.1 \leq \|q_1\|, \|q_2\| \leq 0.4$, and three components of both unit vectors ($\hat{n}_1$ and $\hat{n}_2$) lie between 0 to 1. A set of 200 samples was randomly chosen for training and a set containing 5000 samples was used for testing of trained network. Table II analyses the performance with ℍ-BP ($\eta = 0.001$) algorithm for conventional and RPMN neurons in quaternionic domain. Results demonstrate that ℍ-RPMN based network requires reasonably smaller network topology with comparatively better accuracy in terms of statistical parameters like error variance, correlation, and AIC. The performance for QADD and QSUB operations do not change on increasing the number of neurons at hidden layer, but slow improvement is observed in case of QMULT and N-QDIV operations. ℍ-RPMN based network with ℍ-BP performs better, especially in QMULT and N-QDIV operations with fewer numbers of learning cycles and learning parameters.

B. Linear Transformations

This experiment presents the capability to learn 3D motion patterns through a training set containing points on a line and motion or transformation generalization over complicated geometrical structures in space. As a benchmark problem, this section presents the learning and generalization of linear transformations (rotation, scaling, and translation and their combinations) through ℍ-BP algorithm for the network based on ℍ-RPMN and ℍ-MLP. This facilitates the viewing of 3D objects from different orientations as well as the interpretation of their motion in space.

We have considered a three layer network (2-4-2), as defined in II.B, in learning process for input-output mapping over a straight line containing a reference point (like mid of the line) in 3D space for all experiments. First input receives set of point that lies on a straight line and second input passes the reference point. The simulation results show that the ℍ-BP algorithm with ℍ-RPMN drastically reduces the number of training epochs and also able to generalize more accurately as compare to conventional.

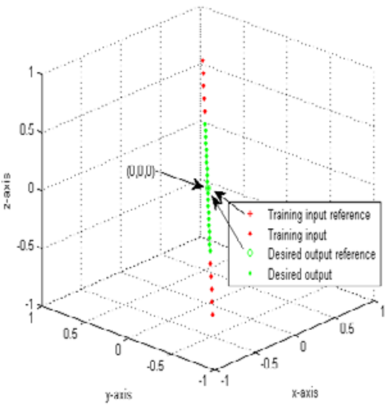
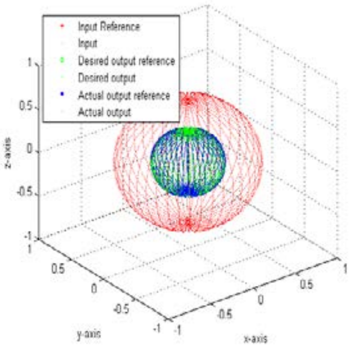


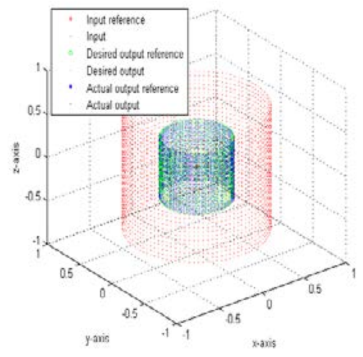
Fig. 2.Training with input-output mapping of straight line with scaling factor ½.

The learning of a three layer network is performed for different class of transformations, which are as follows : The input-output mapping for scaling with factor ½ is shown in Fig. 2; scaling with factor ½ followed by 0.3 unit translation along the positive z-direction is shown in Fig. 4; and scaling with factor ½ followed by 0.3 unit translation along the positive z-direction and π/2 radian rotation around the unit vector (***i***) is shown in Fig. 6. This mapping is defined over straight line containing 21 points and referenced at (0, 0, 0), as shown in Fig. 2, 4 and 6. The training through ℍ-BP with conventional neural network requires comparatively larger average epochs then ℍ-BP with ℍ-RPMN neuron to achieve similar MSE, as shown in Fig. 2, Fig. 4 and Fig. 6, and presented in Table III, IV and V respectively. Thus, the convergence of proposed algorithm is faster over conventional ℍ-BP.

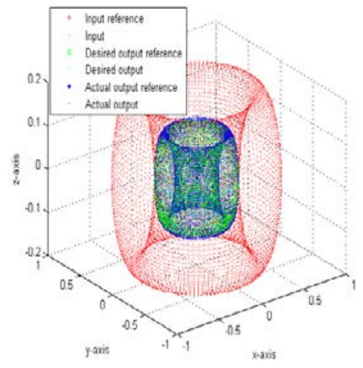
The generalization of trained networks has been performed over complicated 3D objects like sphere (4141 data points), cylinder (2929 data points) and torus (10201 data points). The ℍ-BP with ℍ-RPMN (Fig. (3), Fig. (5) and Fig. (7)) show excellent generalization for all three cases of transformations over rest of the algorithms. Tables III, IV and V clearly demonstrate the superiority of ℍ-BP with RPMN in all experiments.



(a)



(b)



(c)

Fig. 3.The generalization through ℍ-BP algorithm with ℍ-RPMN: Transformations with scaling factor ½; over (a) Sphere (b) Cylinder and (c) Torus.

TABLE III
COMPARISON OF TRAINING AND TESTING PERFORMANCE FOR SCALING

		ℍ-RPMN ($\alpha = 0.95$)	ℍ-MLP
Algorithm		ℍ-BP	ℍ-BP
Network		2-4-2	2-4-2
Parameters		88	88
MSE training through straight line		0.0007	0.0007
Average Learning Cycles		12000	25000
MSE testing through	Sphere	0.0017	0.0052
	Cylinder	0.0014	0.0034
	Torus	0.0033	0.0098

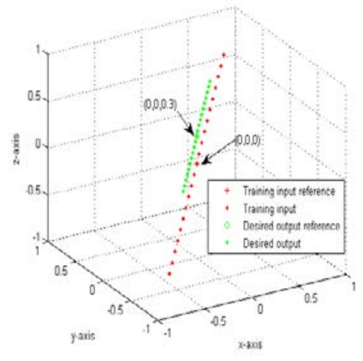
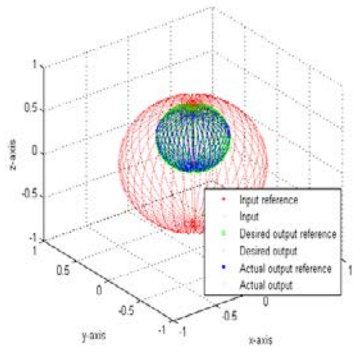
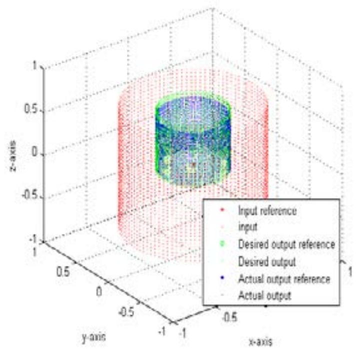


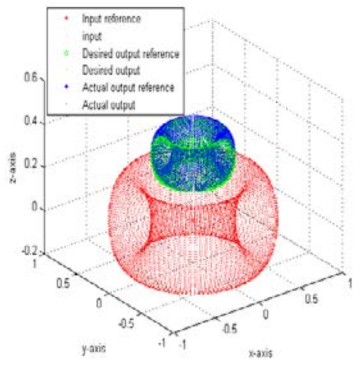
Fig. 4.Training with input-output mapping over straight line with scaling factor ½ and 0.3 unit translation in positive z-direction.



(a)



(b)



(c)

Fig. 5.The generalization through ℍ-BP algorithm : Transformations with scaling factor ½ and 0.3 unit translation in positive z-direction; over (a) Sphere (b) Cylinder and (c) Torus.

TABLE IV
COMPARISON OF TRAINING AND TESTING PERFORMANCE FOR SCALING AND TRANSLATION

		ℍ-RPMN ($\alpha = 0.95$)	ℍ-MLP
Algorithm		ℍ-BP	ℍ-BP
Network		2-4-2	2-4-2
Parameters		88	88
MSE training through straight line		0.0007	0.0007
Average Learning Cycles		15040	28000
MSE testing through	Sphere	0.0021	0.0063
	Cylinder	0.0016	0.0054
	Torus	0.0041	0.0095

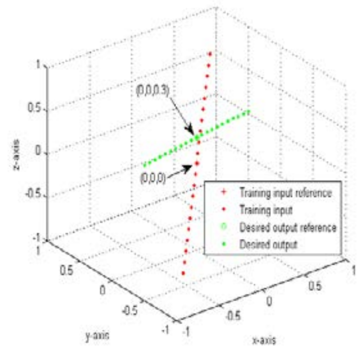
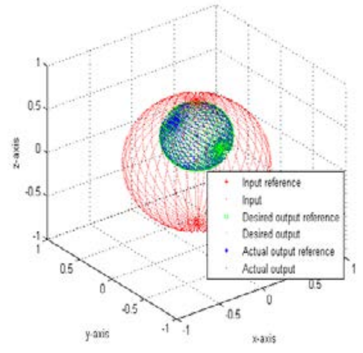
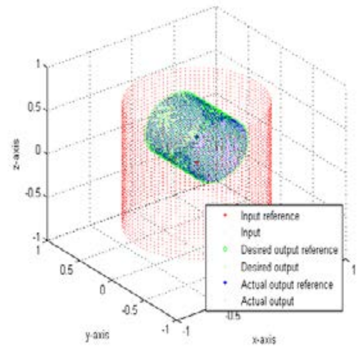


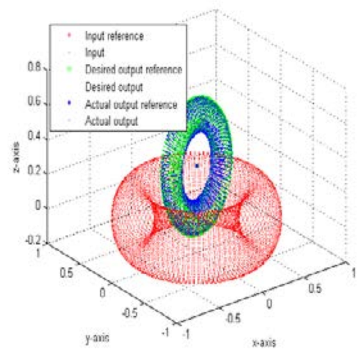
Fig. 6. Training with input-output mapping of straight line with scaling factor ½, 0.3 unit translation in positive z-direction, and $\pi/2$ radian rotation around the unit vector ($\boldsymbol{i}$).



(a)



(b)



(c)

Figure 7. The generalization through ℍ-BP algorithm: Transformations with scaling factor ½, 0.3 unit translation in positive z-direction, and $\pi/2$ radian rotation around the unit vector ($\boldsymbol{i}$); over (a) Sphere (b) Cylinder and (c) Torus.

TABLE V
COMPARISON OF TRAINING AND TESTING PERFORMANCE FOR SCALING, TRANSLATION AND ROTATION

		ℍ-RPMN (α = 0.95)	ℍ-MLP
Algorithm		ℍ-BP	ℍ-BP
Network		2-4-2	2-4-2
Parameters		88	88
MSE training through straight line		0.0007	0.0007
Average Learning Cycles		16500	30000
MSE testing through	Sphere	0.0025	0.0062
	Cylinder	0.0018	0.0035
	Torus	0.0047	0.0088

C. 3D Face Recognition

In this section, we have focused on 3D face identification as biometrics application through proposed methodology and compare it with related methods. The two human face datasets of 3D points cloud containing variable head position, orientation, and facial expressions, have been considered for training and testing. The first set consists of five faces of same person and other set have five faces of different persons. In both experiments, one face has been used for training of the 1-2-1 network and the rest for testing. Thus, it is a basic and primitive experiment that learns the complex geometrical surface of one face and classify the rest of the faces through quaternionic signal based networks; surely it will put a leading direction to future researchers to work with large dataset using such a simple and small neural network.

The first experiment is performed on first dataset containing 05 faces of same person with different orientation and poses; the learning of NN in quaternionic domain is done with one face (Fig. 8(a)) and testing with all faces where each 3D face consists of 4654 points cloud data. Table VI presents the training and testing analysis of faces through learning algorithms of each face. The Table VI also presents the comparative analysis of threshold MSEs with respect to average epochs for all algorithms. The threshold MSE reaches significantly faster during training in case of RPMN model (power coefficient α = 0.90) in ℍ-BP (learning rate η = 0.001). This table shows that the testing error of all five faces are less comparable to each other for all algorithms which demonstrate they are faces of same person irrespective of minor variations in face orientation and poses. These results infer the learning and generalization capability of neural network in quaternionic domain.

Similarly, the second experiment is performed on another dataset containing 05 faces of different persons; the learning of NN in quaternionic domain is done with one face (Fig. 9(a)) and testing with all faces where each 3D face consists of 6397 points cloud data. Table VII presents the training and testing analysis of faces through learning algorithms. The Table VII also presents the comparative analysis of threshold MSEs with respect to average epochs for all algorithms. The threshold MSE reaches significantly faster during training in case of RPMN model (power coefficient α = 0.90) in ℍ-BP (learning rate η= 0.001). For all algorithms, the table shows the testing error of all five faces but MSE of other four faces are much higher in comparison to the face (Fig. 9(a)) which is used in training of the network. This demonstrates that the network classifies the faces of same or different person with different orientation and poses. These results also infer the learning and generalization capability of neural network.

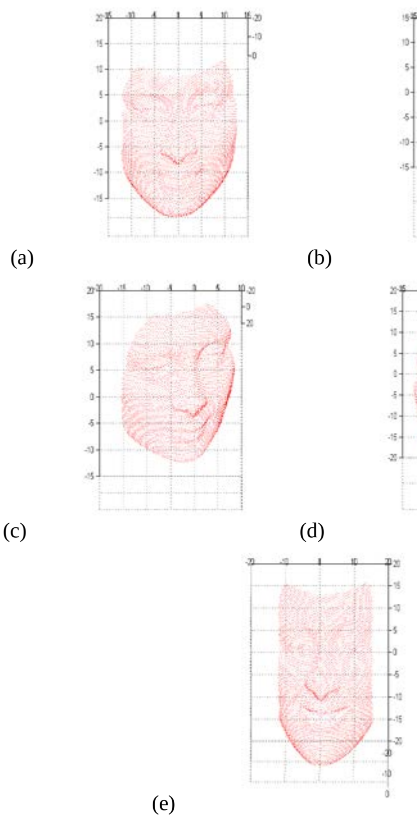
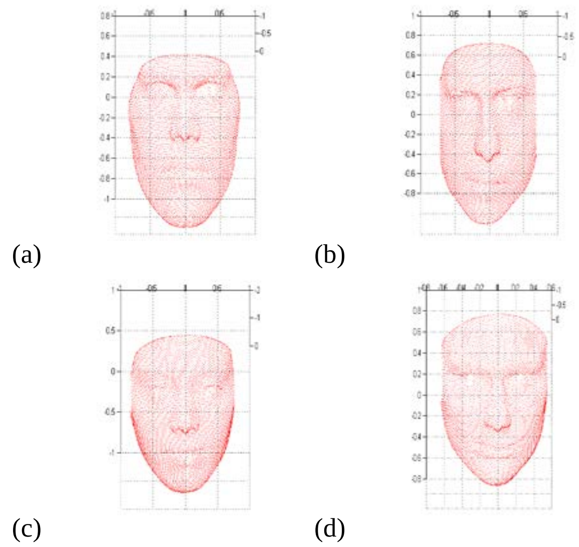


Fig. 8. Five 3D faces of same person with different orientation and poses.

TABLE VI
COMPARISON OF TESTING ERROR OF EACH FACE OF SAME PERSON WITH DIFFERENT ORIENTATION AND POSES

		ℍ-RPMN (α = 0.90)	ℍ-MLP
Algorithm		ℍ-BP	ℍ-BP
Network		1-2-1	1-2-1
Parameters		28	28
MSE training through Fig. 8(a)		0.0001	0.0001
Average Learning Cycles		12000	28000
MSE testing through Fig.	8(a)	2.4842e-04	2.7214e-04
	8(b)	3.8822e-04	3.5431e-03
	8(c)	3.13943-04	5.1153e-03
	8(d)	4.8824e-04	4.5212e-04
	8(e)	3.6904e-04	3.9148e-04



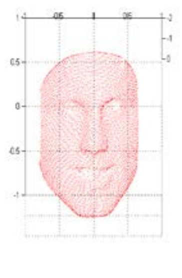


Fig. 9. Five 3D faces of different persons

TABLE VII
COMPARISON OF TESTING ERROR OF EACH FACE OF DIFFERENT PERSON

		ℍ-RPMN (α = 0.90)	ℍ-MLP
Algorithm		ℍ-BP	ℍ-BP
Network		1-2-1	1-2-1
Parameters		28	28
MSE training through Fig. 8(a)		0.0001	0.0001
Average Learning Cycles		13000	29000
MSE testing through Fig.	8(a)	1.7721e-04	1.8521e-04
	8(b)	8.2840e-01	8.7296e-01
	8(c)	3.3772e-00	3.5742e-00
	8(d)	5.5721e-02	6.2996e-02
	8(e)	3.7327e-01	3.9274e-01

IV. CONCLUSION

This paper presents an efficient neuron model with nonlinear aggregation function of quaternionic-valued signals and its evaluation is performed through error propagation (BP) learning algorithm in quaternionic domain. The proposed methodology is systematically evaluated and compared with convention neuron in quaternion domain through a wide spectrum of 3D and 4D problems. The root-power mean of quaternionic signal is conceptually used as an aggregation function of the proposed neuron which emulates better performance than conventional neuron. The quicker convergence with better performance is the significant advantage of algorithm with RPMN which always revealed. Its computational power is also demonstrated through various benchmark problems (function approximation, prediction, linear transformation and 3D face recognition). The power coefficient (α) is an important parameter in which exhibits the approximation capabilities of root-power mean neuron (ℍ-RPMN). The development of an adaptive algorithm to define the power coefficient of ℍ-RPMN will be interesting work for future research.

REFERENCES

[1] B. W. Mel, "Information processing in dendritic trees," *Neural Comput.*, vol. 6, no. 6, pp. 1031–1085, Nov. 1994.

[2] C. Koch and I. Segev, "The role of single neurons in information processing," *Nat Neurosci.*, 3(Suppl), pp. 1171–1177, Nov. 2000.

[3] A. Polsky, B. W. Mel, and J. Schiller, "Computational subunits in thin dendrites of pyramidal cells," *Nat Neurosci.*,7, pp. 621–627, May, 2004.

[4] K. Sidiropoulou, E. K. Pissadaki, and P. Poirazi, "Inside the brain of a neuron," EMBO Rep., vol. 7, no. 9, pp. 886–892, Sep. 2006.

[5] M. Lavzin, S. Rapoport, A. Polsky, L. Garion, and J. Schiller, "Nonlinear dendritic processing determines angular tuning of barrel cortex neurons *in vivo*," *Nature*, vol. 490, no. 7420, pp. 397–401, Sep. 2012.

[6] Y. Todo, H. Tamura, K. Yamashita, and Z. Tang,"Unsupervisedlearnableneuronmodelwithnonlinearinteractiononde ndrites," *Neural Netw.*, vol. 60, pp. 96–103, Dec. 2014.

[7] T. Jiang,D. Wang, J. Ji,Y. Todo, andS. Gao, "Single dendritic neuron with nonlinear computation capacity: A case study on XOR problem," in *Proc. Int. Conf. on Progress in Informatics and Computing (PIC)*, pp. 20–24, Dec. 2015.

[8] W. Chen, J. Sun, S. Gao, J.-J. Cheng, J. Wang, and Y. Todo, “Using a Single Dendritic Neuron to Forecast Tourist Arrivals to Japan,” *IEICE Trans. Inf. & Syst.*, vol. E100–D, no. 1, pp. 190–202, Jan. 2017.

[9] Y. Xiong, W. Wu, X. Kang, and C. Zhang, “Training Pi-Sigma Network by Online Gradient Algorithm with Penalty for Small Weight Update,” *Neural Comput.*, vol. 19, no. 12, 3356–3368, Jan. 2008.

[10] C.-K. Li, “A sigma-pi-sigma neural network (SPSNN),” *Neural Process. Lett.*, vol. 17, no. 1, pp. 1–19, Mar. 2003.

[11] N. Homma and M. M. Gupta, “A general second-order neural unit,”*Bull. Coll. Med. Sci. Tohoku Univ.*, vol. 11, no. 1, pp. 1–6, 2002.

[12] B. K. Tripathi and P. K. Kalra, “The novel aggregation function-based neuron models in complex domain,” *Soft Comput.*, vol. 14, no. 10, pp. 1069–1081, Aug. 2010.

[13] C. L. Giles and T. Maxwell, “Learning, invariance, and generalization in high-order neural networks,” *Appl. Opt.*, vol. 26, no. 23, pp. 4972–4978, 1987.

[14] M. Zhang, S. Xu, and J. Fulcher, “Neuron-adaptive higher order neural network models for automated financial data modeling,” *IEEE Trans. Neural Netw.*, vol. 13, no. 1, pp. 188–204, Jan. 2002.

[15] S. Xu, “Adaptive higher order neural network models and their applications in business,”*IGI Global*, pp. 314–329, 2009.

[16] E. B. Kosmatopoulos,M. M. Polycarpou, M. A. Christodoulou, and P.A. Ioannou, “High-order neural network structures for identification of dynamical systems,” *IEEE Trans. Neural Netw.*, vol. 6, no. 2, pp. 422–431, Mar. 1995.

[17] H. Dyckhoff and W. Pedrycz, “Generalized means as model of compensative connectives,” *Fuzzy Sets Syst.*, vol. 14, no. 2, pp. 143–154, Nov. 1984.

[18] R. R. Yager, “Generalized OWA aggregation operators,” *Fuzzy Optimization and Decision Making*, vol. 3, no. 1, pp. 93–107, Mar. 2004.

[19] A. V. Ooyen and B. Nienhuis, “Improving the convergence of the backpropagationalgorithm,” *Neural Netw.*, vol. 5, no. 3, pp. 465–472, 1992.

[20] X. Chen, Z. Tang, and S. Li, “An modified error function for the complex-value backpropagation neural network,” *Neural Inf. Process.*, vol. 8, no. 1, pp. 1–8, Jul. 2005.

[21] G. D. Magoulas, N. V. Michael, and S. A. George, “Effective backpropagation training with variable stepsize,” *Neural Netw.*, vol. 10, no.1, pp. 69–82, 1997.

[22] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, “Accelerating the convergence of the back-propagation method,” *Biological Cybernetics*, vol. 59, no. 4, pp. 257–263, 1988.

[23] C. C. Yu and D. B. Liu, “A backpropagation algorithm with adaptive learning rate and momentum coefficient,” in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2, 2002, pp. 1218–1223.

[24] E. Istook and T. Martinez, “Improved backpropagation learning in neural networks with windowed momentum,” *Int. J. Neural Sys.*, vol. 12, no. 3 and 4,pp. 303–318, Jan. 2002.

[25] R. A. Jacobs, “Increased rates of convergence through learning rate adaptation,” *Neural Netw.*, vol. 1, no. 4, pp. 295–307, 1988.

[26] A. A. Minai and R. D. Williams, “Back-propagation heuristics: a study of the extended delta-bar-delta algorithm,” in *Proc. Int. Jt. Conf. Neural Netw.*, June 1990, pp. 595–600.

[27] S. E. Fahlman, “An empirical study of learning speed in backpropagation networks,” Tech. Rep. CMU-CS-88-162, Sep. 1988.

[28] A. N. Kolmogoroff, “Sur la notion de la moyenne,” *Acad. Naz. Lincei Mem. Cl. Sci. Fis. Mat. Natur. Sez.*, vol. 12, no. 6, pp. 388–391, 1930.

[29] M. Nagumo, “Über eine klasse der mittelwerte,” *Jpn. J. Math.*, vol. 7, pp. 71–79, 1930.

[30] B. K. Tripathi and P. K. Kalra, “On efficient learning machine with root-power mean neuron in complex domain,” *IEEE Trans. Neural netw.*, vol. 22, no. 5, pp. 727–738, May 2011.

[31] W. R. Hamilton, “On a new species of imaginary quantities connected with a theory of quaternions,” in *Proc. Royal Irish Academy*, vol. 2, no. 1843, pp. 424–434, Nov. 1844.

[32] B. C. Ujang, C. C. Took, and D. P. Mandic, “Split quaternion nonlinear adaptive filtering,” *Neural Netw.*, vol. 23, no. 3, pp. 426–434, Apr. 2010.

[33] T. Nitta, “An extension of the back-propagation algorithm to complex numbers,” *Neural Netw.*, vol. 10, no. 8, pp. 1391–1415, Nov. 1997.

[34] B. K. Tripathi, “High dimensional neurocomputing: Growth, Appraisal and Applications,” Springer, India, 2015.

[35] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bull. Math. Biophys.*, vol. 4, no. 4, pp. 115–133, Dec. 1943.

[36] A. Hirose, “Complex-valued neural networks: theories and applications,” vol. 5., World Scientific, 2003.

[37] T. Nitta, “A quaternary version of the back-propagation algorithm,” in *Proc. Int. Conf. Neural Netw.*, vol. 5, Nov. 1995, pp. 2753–2756.

[38] M. Schmitt, “On the complexity of computing and learning with multiplicative neural networks,” *Neural Comput.*, vol. 14, no. 2, pp. 241–301, Feb. 2002.

[39] P. K. Kalra, B. Chandra, and M. Shiblee, “New neuron model for blind source separation,” in *Proc. Int. Conf. Neural Inf. Process.*, Auckland, New Zealand, Nov. 2008, pp. 27–36.

[40] G. M. Georgiou, “Exact interpolation and learning in quadratic neuralnetworks,” in *Proc. Int. Joint Conf. Neural Netw.*, Vancouver, BC, Canada, Jul. 2006, pp. 230–234.

[41] S. J. Sangwine and N. L. Bihan, “Quaternion polar representation with a complex modulus and complex argument inspired by the Cayley-Dickson form,” *Advances in Applied Clifford Algebras*, vol. 20, no. 1, pp. 111–120, Mar. 2010.

[42] E. Cho, “De moivre's formula for quaternions,” *Appl. Math. Lett.*,vol. 11, no. 6, pp. 33–35, Nov. 1998.

[43] D. B. Foggel, “An information criterion for optimal neural network selection,” *IEEE Trans. Neural Netw.*, vol. 2, no. 5, pp. 490-497, Sep.1991.

[44] R. Naresh, S. Pandey, and JB Shukla, “Modeling the cumulative effect of ecological factors in the habitat on the spread of tuberculosis,” *International Journal of Biomathematics*, vol. 2, no. 3, pp. 339-355, Sep. 2009.