Journal of Applied Sciences

ISSN 1812-5654

Journal of Applied Sciences 9 (14): 2501-2518, 2009
ISSN 1812-5654

Computing of Compressible Flow Using Neural Network Based-on Dual-Level Clustering

[1]Nameer N. EL-Emam and [2]Nadia Y. Yousif
[1]Department of Computer Science, Philadelphia University, Jordan
[2]Department of Computer Information Systems, Philadelphia University, Jordan

Abstract: This study presents a new technique based on back-propagation algorithm and Adaptive Neural Networks (ANN) to compute a compressible flow represented by velocity profiles through symmetrical double steps channels. The technique adopts a new clustering algorithm named a Dual-level Clustering (DC), which is based on the speed of flow, the flow deflection and the average of false diffusion error $(\Gamma^{p}_{Average})$. A learning system of three stages is employed, whose first two stages are run simultaneously while the third works as an optimizer stage. The first stage constructs a finite element analysis employing adaptive incremental loading to select appropriate patterns effectively. The second stage concerns an ANN with Modified Adaptive Smoothing Error (MASE), whereas the third stage is to classify a set of patterns into DC to reduce the effect of errors. The proposed training algorithm is fast enough and the simulation results of the learning system are in harmony with the available previous works. The success of such algorithm can be attributed to three reasons. The first is the employment of ANN that is an excellent approximator, especially if training starts from laminar flow patterns. The second reason is the speed of training due to the use of ANN with learning rate $\alpha = 0.7$ on the few number of selected patterns equal to 464 and the third reason is satisfying the stability and the accuracy for high range of Reynolds numbers (Re) Re = 4500 due to the use of MASE and DC based on false diffusion error.

Key words: Velocity formulation, compressible flow, Navier-Stokes, finite element ,training algorithm, neural networks

INTRODUCTION

A computer simulation of compressible fluid flow is of particular importance in designing pipe networks, channels, diffuser and aerodynamic using Navier-Stokes Equations (NSE), which are non-linear Partial Differential Equations (PDE). These equations have widest applications as they govern the motion of every fluid, being a gas or liquid or a plasticized solid material acted upon by forces causing it to change the shape. The popular methods for the numerical solution of PDE's are Finite Difference Analysis (FDA), Finite Element Analysis (FEA), Boundary Element Analysis (BEA) and Finite Volume Analysis (FVA) (Glowinski and Neittaanmaki, 2008).

The earliest solution of (NSE) used non-simultaneous solver through FEA to implement velocity-vorticity (u-v-ω) formulation (Singh and Li, 2003). In recent years, the FEA has been employed quite expensively in predicating laminar, transition and turbulent flow (Glowinski and Neittaanmaki, 2008). It is also costly and not easy to solve specially for high range of Reynolds numbers (Re) due to the long processing time required to attain convergence. Moreover, it is very expensive with respect to the storage for the refinement mesh points (EL-Emam, 2006; EL-Emam and Shaheed, 2008).

Recently, many researchers have demonstrated a neural network approach that is fast and reliable for predicting complex problem solving (Vodinh *et al*., 2005). This approach is based on many types of architectures, such as an artificial neural network with single/multi hidden layer(s) (Lefik and Wojciechowski, 2005; Meybod and Beigy, 2002). Developing neural models in fluid applications was implemented by Gölcü (2006) to study the Head-flow curves of deep well pump impellers with splitter blades.

Neural networks with a clustering approach were developed by many researches. Some methods are classified as Agglomerative hierarchical methods such as: Single Linkage, Complete Linkage, Average Linkage, Median Linkage, Centroid, Ward and others (He *et al*., 2005). Frossyniotis *et al*. (2005) applied a multi-clustering

Corresponding Author: Nameer N. EL-Emam, Department of Computer Science, Philadelphia University, Jordan
Tel: +962 7 777489308 Fax: +962-2-6374440

method based on combining several runs of a clustering algorithm to obtain a distinct partition of the data which is not affected by the initialization.

In this study, a Dual-level Clustering (DC) technique based on back-propagation algorithm and Adaptive Neural Network (ANN) with Modified Adaptive Smoothing Error (MASE) is proposed to compute a compressible flow represented by velocity profiles through symmetrical double steps channels. The clustering is based on the speed of flow, the flow deflection and the average of false diffusion error $(\Gamma^{P}_{Average})$. The clusters and their sub-clusters are fixed to three according to flow regime (laminar, transition and turbulent) while the number of patterns for each sub-cluster is changing according to DC. The proposed training algorithm overcomes a large amount of training time, instability and inaccuracy of results and a large number of patterns for each cluster.

PROBLEM DEFINITION

Fluid agitation and mixing can be generated by the fluid flowing from one step to another in symmetrical double steps channels when sudden expansion of flow diameter occurs (Fig. 1). This, however, increases the main velocity of the separating flow that is produced by the other step. Such problem presents an even greater challenge on the stability and convergence capability of solution procedure. Our motivation is to enhance the stability and convergence criteria through implementing DC algorithm with ANN approach.

PROBLEM FORMULATIONS

Equation 1-3 shown represent the governing equations (NSEs) for two-dimensional steady state of compressible flow in terms of the primitive variables u-v-p (Glowinski and Neittaanmaki, 2008; Vodinh, *et al*., 2005; Eker and Serhat, 2006), where u-v are velocity functions, p is the pressure function, μ is fluid dynamic viscosity, ρ is the fluid density and Re is defined as ρVd/μ, with d being a characteristic length chosen to be the width of channel and V is the inlet velocity.

$$(\rho u)_x + (\rho v)_y = 0 \tag{1}$$

$$\begin{aligned} &\mathrm{Re}\left(\rho u u_x + \rho v u_y\right) = \\ &-\mathrm{Re}\, p_x + \left(2\mu u_x - \frac{2}{3}\mu\left((\rho u)_x + (\rho v)_y\right)\right)_x + \left(\mu u_y + \mu v_x\right)_y \end{aligned} \tag{2}$$

$$\begin{aligned} &\mathrm{Re}\left(\rho u v_x + \rho v v_y\right) = \\ &-\mathrm{Re}\, p_y + \left(2\mu v_y - \frac{2}{3}\mu\left((\rho u)_x + (\rho v)_y\right)\right)_y + \left(\mu u_y + \mu v_x\right)_x \end{aligned} \tag{3}$$

The second order velocity equations are obtained by manipulating the vorticity ω and continuity equation (Eq. 4, 5):

$$\nabla^2 u + \omega_y + \left(\frac{u\rho_x + v\rho_y}{\rho}\right)_x = 0 \tag{4}$$

$$\nabla^2 v - \omega_x + \left(\frac{u\rho_x + v\rho_y}{\rho}\right)_y = 0 \tag{5}$$

The vorticity transport equation (Eq. 6) is obtained by taking the curl of the momentum equations and eliminating the pressure term.

$$\nabla^2(\mu\omega) - \mathrm{Re}\left(\rho u \omega_x + \rho v \omega_y + T_\rho\right) + \left(T_{\mu u}\right)_x + \left(T_{\mu v}\right)_y = 0 \tag{6}$$

where

$$\begin{aligned} T_\rho &= u\left(\rho_x u_y - \rho_y u_x\right) + v\left(\rho_x v_y - \rho_y v_x\right) \\ T_{\mu u} &= 2\left(\mu_x u_y - \mu_y u_x\right) \\ T_{\mu v} &= 2\left(\mu_x v_y - \mu_y v_x\right) \end{aligned} \tag{7}$$

NUMERICAL SOLUTION OF FLUID FLOW USING FEA WITH ADAPTIVE INCREMENTAL LOADING

FEA requires that the domain under consideration is to be partitioned into a number of elements which are defined by a certain fixed topology in terms of nodes. Breaking the original domain into a set of elements is not an easy job, especially for problems with moving boundaries, free surface, or complex boundary (Glowinski and Neittaanmaki, 2008) as in Fig. 1a-c.

Obviously, numerical formulas of Eq.4-6 are always subject to evaluation with unsatisfactory accuracy and possible errors. An approximate solution of numerical formula using FEA gives the high accuracy and stability when adaptive mesh points are considered as in Fig. 1 to overcome the problem of discretization error. The amount of this error depends on four factors: order of shape function, size of elements, shape of elements and arrangement of elements in the domain (El-Emam, 2006; El-Emam and Abdul Shaheed, 2008).

The numerical solution is highly confident in comparison with experimental results introduced by (Glowinski and Neittaanmaki, 2008). Unfortunately, this solution needs more time regarding the amount of computation for Eq. 4-6 to find fluid flow behavior (patterns) for all ranges of Re (0-4500). Consequently, the adaptive incremental loading is effectively introduced in this study to reduce the number of patterns.

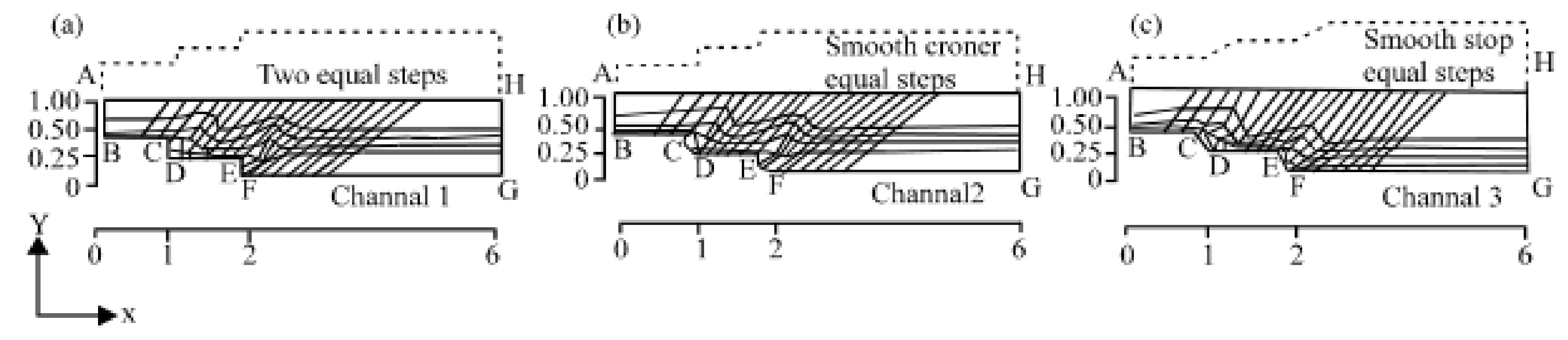


Fig. 1: (a-c) Three types of symmetrical double steps channels with mesh points

The numerical solution of u-v-ω functions on channels such as those shown in Fig. 1 requires the following boundary conditions:

Condition 1: $\forall (x, y) \in AB$ (inlet), Dirichlet (C^0– BC) is used to find u (x, y) and v (x, y) as follows:

$$\begin{aligned} u(0,y) &= 8y - 4y^2 - 3 \\ v(0,y) &= 0 \\ \omega(0,y) &= 8y - 8 \end{aligned} \tag{8}$$

Condition 2: $\forall$ (x, y) $\in$ BC|CD|DE|EF|FG (Lower solid boundaries) and by using Dirichlet (C^0 – BC), we have:

$$\begin{aligned} u(x,y) &= 0 \\ v(x,y) &= 0 \end{aligned} \tag{9}$$

where, ω (x, y) is not specified at this side of boundary. FEM is used to find its approximation value.

Condition 3: $\forall$ (x, y) $\in$ GH and by using C^1 B.C and C^0 B.C for u (6, y) and v (6, y), respectively, we obtain:

$$\begin{aligned} u_n(6,y) &= 0 \\ v(6,y) &= 0 \\ \omega(6,y) &= 0 \end{aligned} \tag{10}$$

where, (n) refers to the normal boundary direction.

Condition 4: $\forall$ (x, y) $\in$ AH and by using C^1 B.C and C^0 B.C for u (x, 1) and v (x, 1), respectively, we have:

$$\begin{aligned} u_n(x,1) &= 0 \\ v(x,1) &= 0 \\ \omega(x,1) &= 0 \end{aligned} \tag{11}$$

Equations 12-14 are obtained from Eq. 4-6 with the use of Galerkin weighted residual method based on adaptive incremental loading (Glowinski and Neittaanmaki, 2008; Kaczmarczuk and Waszczyszyn, 2005), where, N_{1i}, N_{2i} and N_{3i} are linear Lagrange polynomials (weighted functions) for the quadrilateral elements (Vodinh *et al*., 2005) in the channel's domain Ω.

$$\iint_\Omega N_{1i}\left(\nabla^2 u + \omega_y + \left(\frac{u\rho_x + v\rho_y}{\rho}\right)_x\right) d\Omega = 0 \tag{12}$$

$$\iint_\Omega N_{2i}\left(\nabla^2 v + \omega_x + \left(\frac{u\rho_x + v\rho_y}{\rho}\right)_y\right) d\Omega = 0 \tag{13}$$

$$\iint_\Omega N_{3i}\left(\nabla^2(\mu\omega) - (\mathrm{Re}+\delta\mathrm{Re})\left(\rho u\omega_x + \rho v\omega_y + T_\rho\right) + \left(T_{\mu u}\right)_x + \left(T_{\mu v}\right)_y\right) d\Omega = 0 \tag{14}$$

FEA with modified Newton's method are used to find the variation-vectors δu, δv and δω (Eq. 15-17). The δRe is the adaptive incremental loading and its value is changed from pattern to next patterns according to the value of $\Gamma^p_{Average}$, where, R_{1i}, R_{2i} and R_{3i} are the residuals at node i.

$$\iint_\Omega \left(N_{1ix}(\delta u)_x + N_{1iy}(\delta u)_y + N_{1i}(\delta\omega)_y + N_{1ix}\left(\frac{\rho_x\delta u + \rho_y\delta v}{\rho}\right)\right) d\Omega = -R_{1i} \tag{15}$$

$$\iint_\Omega \left(N_{2ix}(\delta v)_x + N_{2iy}(\delta v)_y + N_{2i}(\delta\omega)_x + N_{2iy}\left(\frac{\rho_x\delta u + \rho_y\delta v}{\rho}\right)\right) d\Omega = -R_{2i} \tag{16}$$

$$\begin{aligned} \iint_\Omega \Big(& N_{3ix}(\mu\,\delta\omega)_x + N_{3iy}(\mu\,\delta\omega)_y + N_{3i}(\mathrm{Re}+\delta\mathrm{Re})\big(\rho u(\delta\omega)_x + \rho v(\delta\omega)_y + \\ & \rho\omega_x\delta u + \rho\omega_y\delta v + \rho_x(u\delta u + v\delta v)_y - \rho_y(u\delta u + v\delta v)_x\big) d\Omega = -R_{3i} \end{aligned} \tag{17}$$

Substituting the shape functions to obtain the following:

$$\sum_{j=1}^{4} K_{1ij}\delta u_j + \sum_{j=1}^{4} K_{2ij}\delta v_j + \sum_{j=1}^{4} K_{3ij}\delta\omega_j = -R_{1i} \tag{18}$$

where

$$K_{1ij} = \iint_\Omega \left(N_{1ix}\left(N^u_{jx} + \frac{\rho_x}{\rho}N^u_j\right) + N_{1iy}N^u_{jy}\right) d\Omega \tag{19}$$

$$K_{2ij} = \iint_\Omega \left(N_{1ix}\frac{\rho_y}{\rho}N^u_j\right) d\Omega \tag{20}$$

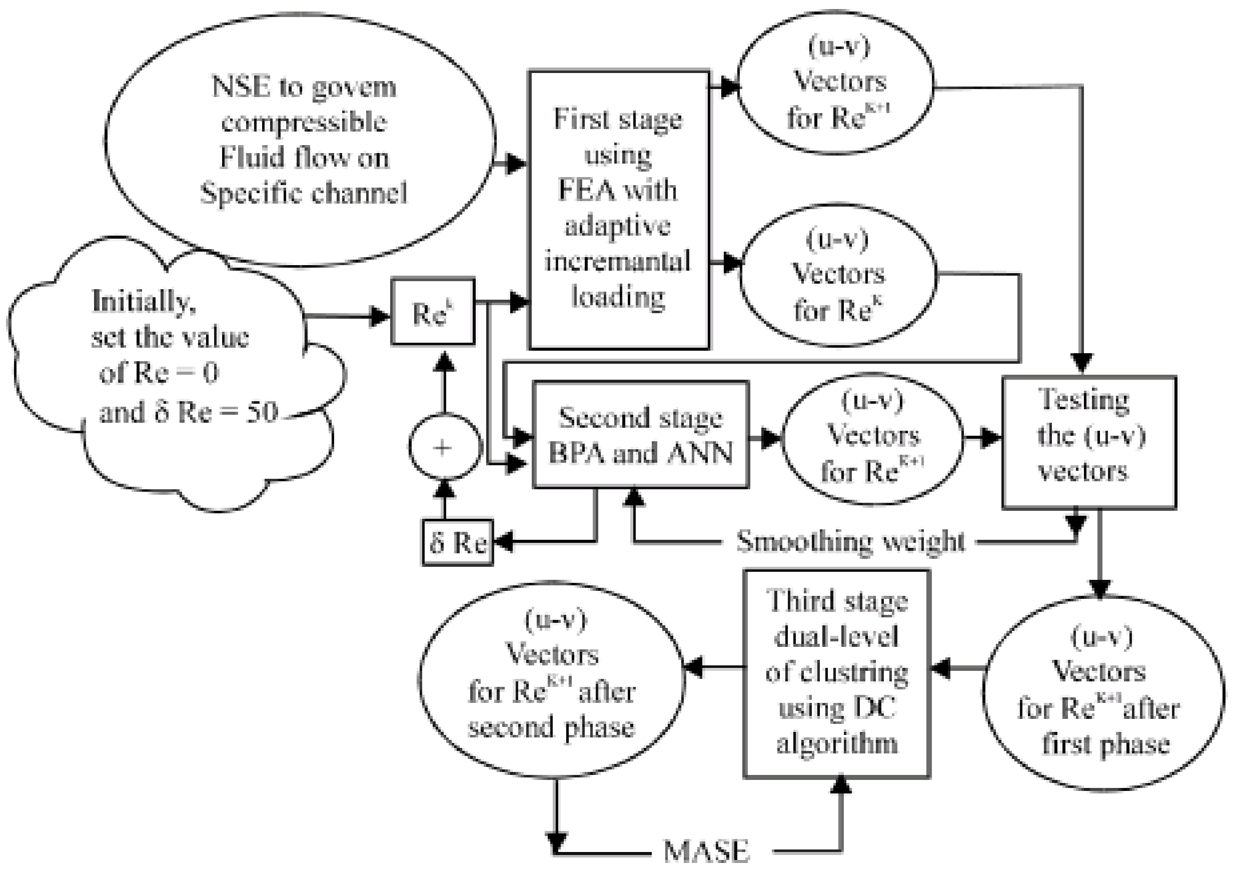

NSE to govem compressible Fluid flow on Specific channel
Initially, set the value of Re = 0 and δ Re = 50
Re^k
First stage using FEA with adaptive incremantal loading
(u-v) Vectors for Re^K+1
(u-v) Vectors for Re^K
Second stage BPA and ANN
(u-v) Vectors for Re^K+1
Testing the (u-v) vectors
δ Re
Smoothing weight
Third stage dual-level of clustring using DC algorithm
(u-v) Vectors for Re^K+1 after second phase
(u-v) Vectors for Re^K+1 after first phase
MASE

Fig. 2: Cooperation between FEA and ANN

$$K_{3ij} = \iint_{\Omega} \left(-N_{1i} N_{jy}^{\omega}\right) d\Omega \tag{21}$$

and

$$\sum_{j=1}^{4} K_{4ij}\delta v_j + \sum_{j=1}^{4} K_{5ij}\delta u_j + \sum_{j=1}^{4} K_{6ij}\delta\omega_j = -R_{2i} \tag{22}$$

where

$$K_{4ij} = \iint_{\Omega} \left(N_{2iy}\left(N_{jy}^{u} + \frac{\rho_y}{\rho} N_j^{u} \right) + N_{2ix} N_{jx}^{u} \right) d\Omega \tag{23}$$

$$K_{5ij} = \iint_{\Omega} \left(N_{2ix} \frac{\rho_x}{\rho} N_j^{u} \right) d\Omega \tag{24}$$

$$K_{6ij} = \iint_{\Omega} \left(N_{2i} N_{jx}^{\omega} \right) d\Omega \tag{25}$$

and

$$\sum_{j=1}^{4} K_{7ij}\delta\omega_j + \sum_{j=1}^{4} K_{8ij}\delta u_j + \sum_{j=1}^{4} K_{9ij}\delta v_j = -R_{3i} \tag{26}$$

where

$$\begin{aligned} K_{7ij} = \iint_{\Omega} \Big(& N_{3ix}\left(\mu_x N_j^{\omega} + \mu N_{jx}^{\omega}\right) + N_{3iy}\left(\mu_y N_j^{\omega} + \mu N_{jy}^{\omega}\right) \\ & N_{3i}(Re + \delta Re)\rho\left(u N_{jx}^{\omega} + v N_{jy}^{\omega}\right)\Big) d\Omega \end{aligned} \tag{27}$$

$$K_{8ij} = \iint_{\Omega} N_{3i}(Re + \delta Re)\left(\rho\,\omega_y N_j^{u} + \rho_x\left(u_y N_j^{u} + u\, N_{jy}^{u}\right) - \rho_y\left(u_x N_j^{u} + u N_{jx}^{u}\right)\right) d\Omega \tag{28}$$

$$K_{9ij} = \iint_{\Omega} N_{3i}(Re + \delta Re)\left(\rho\,\omega_y N_j^{u} + \rho_x\left(v_y N_j^{u} + v\, N_{jy}^{u}\right) - \rho_y\left(v_x N_j^{u} + v N_{jx}^{u}\right)\right) d\Omega \tag{29}$$

In this study, simultaneous solvers on the numerical model represented by Eq. 18, 22 and 26 are implemented to find the fluid flow behavior as an input pattern to the proposed three stages (Fig. 2).

NEURAL NETWORKS MODELING

For nonlinear rheological phenomena, neural networks approach is promising as an alternative technique, where a neural network might consists of a large number of highly interconnected neurons. A fully connected multi-layered neural network is presented in this study. Basically, there are four key-parameters that characterize a neural net architecture: the first is the number of layers, the second is the number of neurons at each layer, the third is the kind of connectivity among layers and the fourth is the kind of activation function used within each neuron (Gölcü, 2006; Kaczmarczyk and Waszczyszyn, 2005; Payne, 2006). Neural network can in principle have any number of layers; each consists of various number of neurons. The most common neural network architecture which is comprised of 3 layers (Hovakimyan *et al*., 2002) (Input, Hidden and Output) is used in this study.

Figure 3 shows simple 3-layers (n-p-m) neural network architecture with (n) neurons in the input layer, (p) neurons in the hidden layer and (m) neurons in the output layer.

Usually, the number of neurons in the input layer is equal to the number of the available features. The present training algorithm uses Re^{k+1} and the velocity functions u-v for Re^k as an input features, where, k is the current iteration index. The number of neurons in the output layer

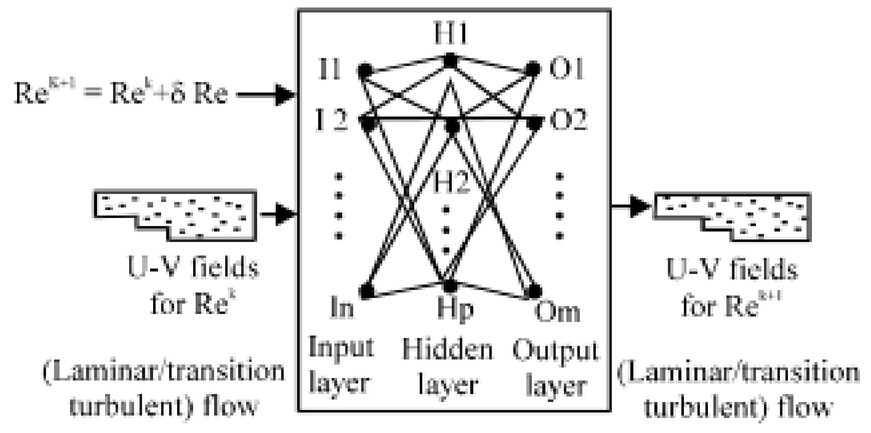


Fig. 3: A 3-Layers neural network architecture with full connections

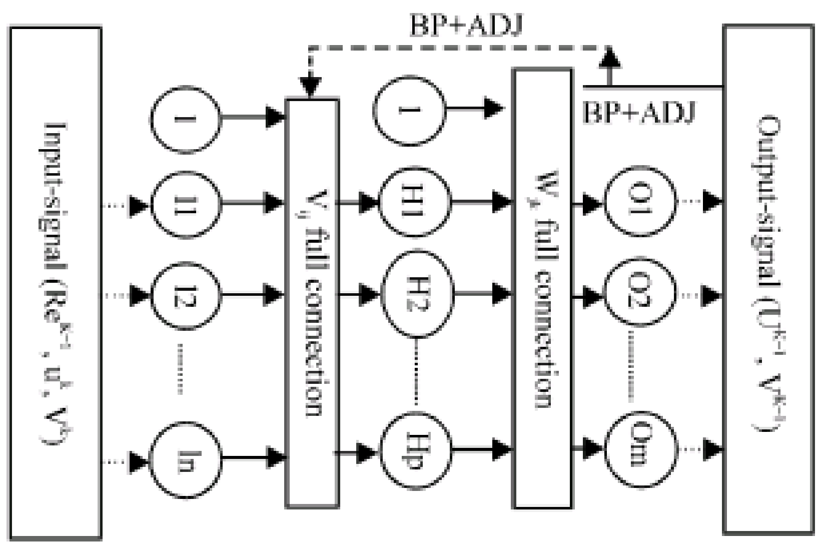


Fig. 4: Neural network layers with BPA

equals to the output features represented by the velocity profile for Re^{k+1}. On the other hand, the number of neurons in the hidden layer needs to be adjusted during training. Usually, there is a trade-off between accuracy of the output and number of neurons in the hidden layer. Complex problems require a large number of neurons at the hidden layer and the kind of connectivity among layers can be full or partial (Frean *et al*., 2006). In the proposed training algorithm, a fully connected architecture is applied where each neuron in a layer is connected to all neurons in the next layer as shown in Fig. 4.

TRAINING OF COMPRESSIBLE FLUID FLOW

The training process presented in this paper employs back-propagation with ANN through three steps: the feedforward of the input training pattern, the back-propagation of the associated error (BP) and the adjustment of the weights (ADJ). Figure 4 shows the (n-p-m) neural network, where the solid arrow refers to many-to-one or one-to-many transitions, the doted arrow refers to one-to-one transition and the dashed arrow refers to send action for adjustment process.

During feedforward step, each input neuron I_i, i =1…n receives an input signal and broadcasts this signal to each of the hidden neurons H_j, j =1… p, as in Eq. 30.

$$H_j = \Phi(V_{0j} + \sum_{i=1}^{n} I_i V_{ij}) \tag{30}$$

Each hidden neuron computes its activation and sends its signal to each of the output neurons O_k, for k = 1…m. Each output neuron O_k computes its activation as in Eq. 31 to form the output signal of the network.

$$O_k = \Phi(W_{0k} + \sum_{j=1}^{n} H_j W_{jk}) \tag{31}$$

The type of activation function implemented in this work is bipolar sigmoid working in the range [1, -1]. This function is given in Eq. 32 and its first derivative is shown in Eq. 33.

$$\Phi(x) = \frac{2}{1+\exp(-x)} - 1 \tag{32}$$

$$\Phi_x(x) = \frac{1}{2}\big(1+\Phi(x)\big)\big(1-\Phi(x)\big) \tag{33}$$

The first derivative of the error factor represented by Δ_k, k = 1…m (Eq. 34) is computed to show the associated error for the specific pattern at the output layer (Meybod and Beigy, 2002; Hovakimyan *et al*., 2002). This error is used to adjust the weight W_{jk} between the hidden neuron H_j and the output neuron O_k as illustrated in Eq. 35, where $\beta \in [0, 1]$ is the damping parameter.

$$\Delta_k = (t_k - O_k)\,\Phi_o(W_{0k} + \sum_{j=1}^{n} H_j W_{jk}) \tag{34}$$

$$\Delta W_{jk}^{new} = \beta \alpha_{jk}^{new} \Delta_k H_j. + (1-\beta).\Delta W_{jk}^{old} \tag{35}$$

Similarly, the first derivative of the error factor Δ_j, j = 1,…, p (Eq. 36) is computed to show the associated error for the specific pattern at the hidden layer. The adjustment to the weight V_{ij} between input neuron I_i and hidden neuron H_j is based on the factor Δ_j and the activation of the output neuron as illustrated in Eq. 37.

$$\Delta_j = \Phi_h(V_{0j} + \sum_{i=1}^{n} I_i V_{ij}) \sum_{k=1}^{m} \Delta_k W_{jk} \tag{36}$$

$$\Delta V_{ij}^{new} = \beta\alpha\Delta_j I_i + (1-\beta)\Delta V_{ij}^{old} \tag{37}$$

The adjustment on the weight function is defined in Eq. 38-39.

$$W_{jk}^{new} = W_{jk}^{old} + \Delta W_{jk} \tag{38}$$

$$V_{ij}^{new} = V_{ij}^{old} + \Delta V_{ij} \tag{39}$$

In this study, $\beta = 0.1$ is assumed and the Adaptive Learning rate AL is used to improve the speed of training by changing the rate of learning α during training process as shown in Eq. 40a (EL-Emam, 2006; Lefik and Wojciechowski, 2005; Meybod and Beigy, 2002; Gölcü, 2006; Maira *et al*., 2002).

$$\alpha_{jk}^{new} = \begin{cases} \alpha_{jk}^{old} + K & \text{if} \quad \Delta W_{jk}^{new} \Delta W_{jk}^{old} > 0 \\ (1-\gamma)\alpha_{jk}^{old} & \text{if} \quad \Delta W_{jk}^{new} \Delta W_{jk}^{old} < 0 \\ \alpha_{jk}^{old} & \text{otherwise} \end{cases} \tag{40a}$$

The training is repeated several times to update the old values of the two dimensional arrays V and W until convergence criteria given in Eq. 40b is satisfied.

$$\underset{\forall k}{\text{Max}} \ (t_k - O_k)^2 < 10^{-6} \quad k = 1,\ldots,m \tag{40b}$$

In addition, the present study introduces a new technique to improve the training process. This technique is implemented to reduce the effect of errors and speed up training by using DC. The proposed algorithm performs two clustering levels, the first level includes three clusters depending on the type of flow (laminar, transition and turbulent) while the second level (sub-clusters) depends on the following:

- The average of all angles θ_3 for all elements in the specific pattern (Eq. 45), where θ_3 angle represents an inclination of velocity from the element's direction as in Fig. 5
- The patterns ordered in ascending order within each sub-cluster according to the value of $\Gamma_{Average}^{P}$ (Eq. 42a-b). In the next section, this point is discussed to show the effect of sorted patterns on the performance of the training algorithm

Figure 5 shows inclination of an element represented by $\xi-\eta$ axis with respect to the X-Y axis and velocity direction V, where θ_1 is the angle between ξ-axis and X- axis, θ_2 is the angle between velocity vector V and X-axis and θ_3 is the angle between ξ-axis and velocity vector V.

Figure 6a shows a finite state automata transition graph to describe formally the first phase of the present training algorithm, where, T is the transition, S is the smoothing error and N is moving to the next pattern.

In this study, three clusters denoted by C_i i = 1, 2, 3 are proposed. Each cluster includes a number of selective appropriate patterns corresponding to diverse Re, which

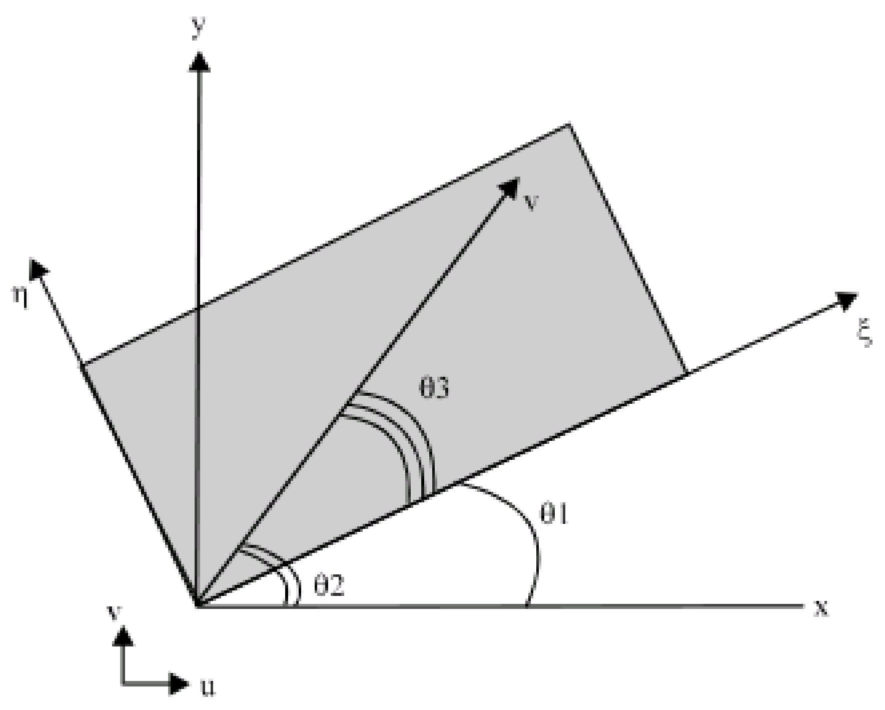


Fig. 5: Velocity inclination for one element

Table 1: Clusters' intervals

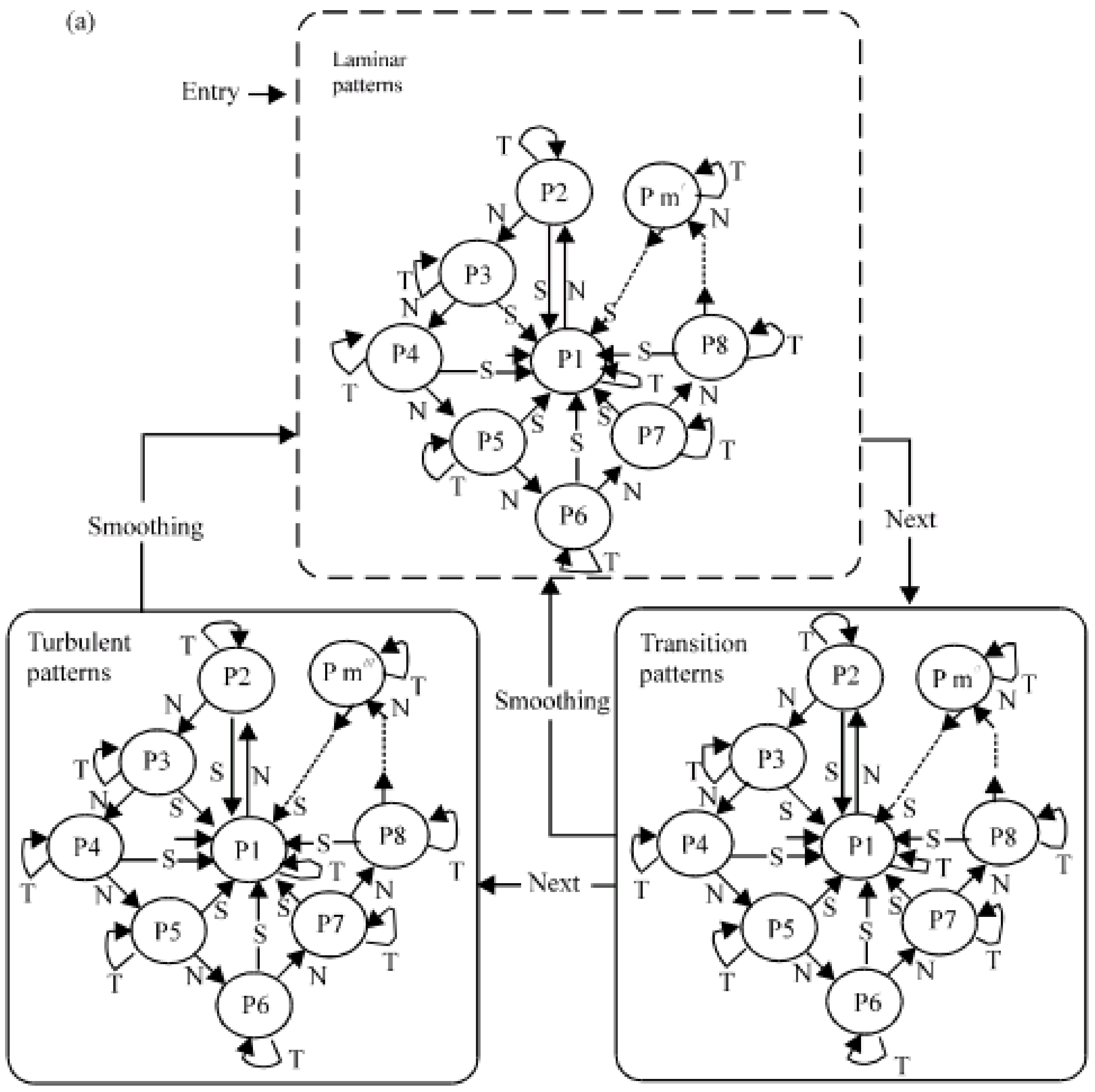
Cluster name	Cluster interval
Laminar	Re ∈ [0, 500]
Transition	Re ∈ [550, 1500]
Turbulent	Re ∈ [1550, 4500]

is equal to m', m'' and m''' patterns for laminar, transition and turbulent clusters, respectively. These patterns are selected by using FEA with adaptive incremental loading on specific type of double steps channel. In the next section, we discuss that a few number of patterns is enough to produce an effective learning system for compressible flow. The proposed intervals for each cluster used in this work are shown in Table 1.

Figure 6b shows the finite state automata transition graph for the second phase of the training algorithm, where the dashed state refers to the final state of transition graph. In addition, the transition label MCE refers to maximum error of clusters (Eq. 44a, b), the label ME refers to the maximum error of neurons (Eq. 44c) and NF refers to the next value of $\Gamma_{Average}^{P}$ (Eq. 42b). Additionally, the pattern P_{ji}^{s} in Fig. 6b represents the jth pattern in the ith cluster and sth sub-cluster for all i = 1, 2, 3 and s = 1, 2, 3. The numbers of patterns in sub-cluster 1 for laminar, transition and turbulent clusters are equal to N', N'' and N''', respectively; M', M'' and M'', respectively for sub-cluster 2 and K', K'' and K''', respectively for sub-cluster 3, where:

$$\begin{aligned} m' &= N' + M' + K' \\ m'' &= N'' + M'' + K'' \\ m''' &= N''' + M''' + K''' \end{aligned} \tag{41}$$

Equation 42a is used to compute the error Γ_e for each element in the channel domain, where this error depends on Re, mesh size, channel shape and inclination of



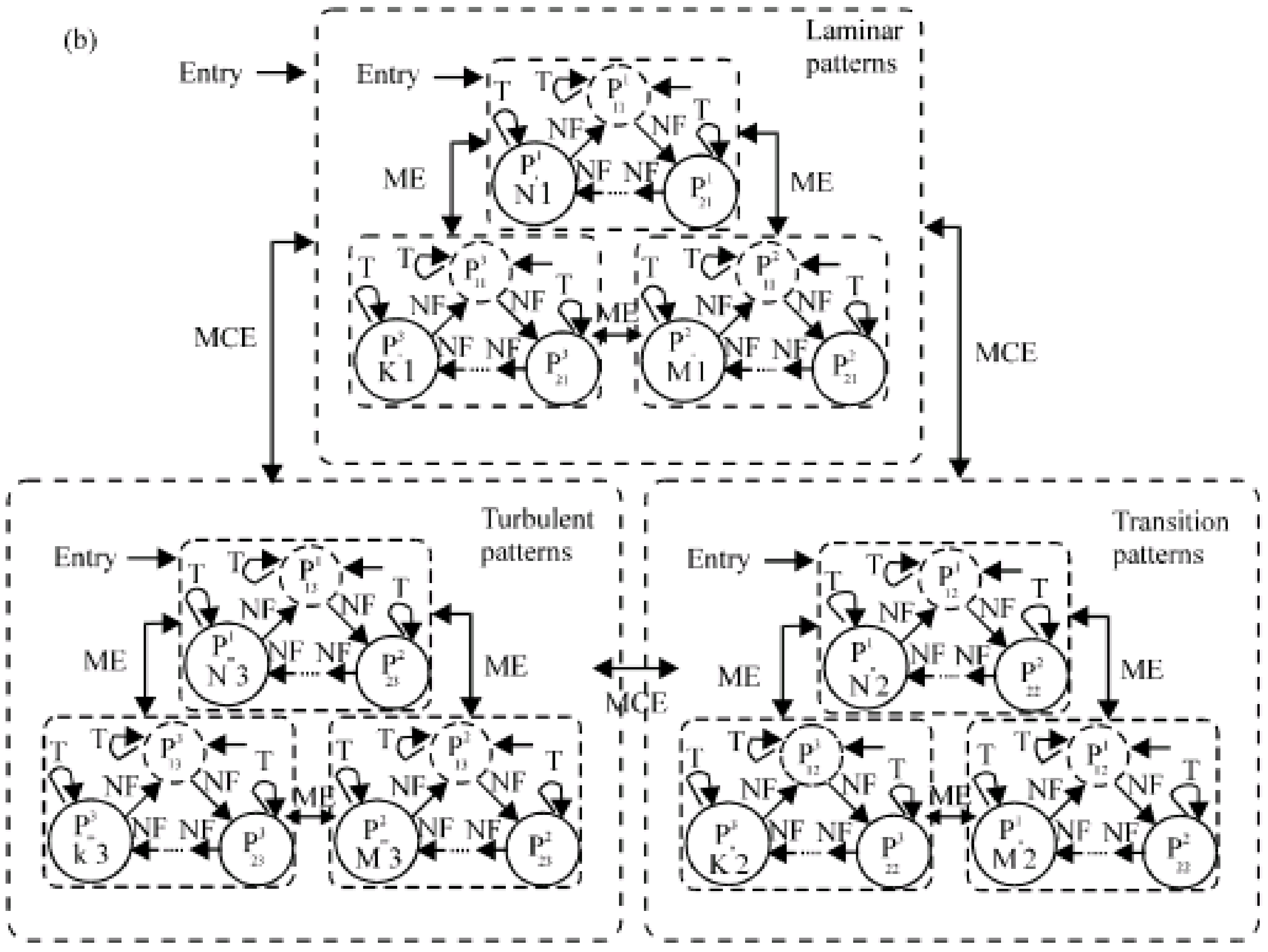


Fig. 6: Transitions automata graph for the (a) 1st phase of training on three clusters and (b) 2nd phase of training algorithm

velocity direction to the grid. The maximum value of Γ_e is reached when the angle between velocity vector V and X-coordinate axis is equal to $\pi/4$.

$$\Gamma_e = \frac{\sqrt{2}}{4}\,\mathrm{Sin}\,(\frac{\pi}{4}+\theta_3)|V|L\,\sin(2\theta_3) \tag{42a}$$

$$\Gamma^P_{\mathrm{Average}} = \frac{1}{100}\sum_{e=1}^{100}\Gamma^P_e \tag{42b}$$

THE TRAINING ALGORITHM

The training algorithm is implemented through two phases. The first phase is to find appropriate patterns with their training process and the second phase is to reduce the effect of errors. The second phase requires DC sub-algorithm to apply hierarchical clustering (He *et al*., 2005; Silvestre *et al*., 2008) for each cluster. This kind of clustering is worked iteratively to agglomerate patterns into three sub-clusters and it is based on two steps: the first is used to find the amount of flow deflection for each pattern while the second is to sort patterns in ascending order at each sub-cluster according to $\Gamma^P_{\mathrm{Average}}$. The two phases of the training algorithm and the sub-algorithm are presented below:

First phase of the training algorithm: Find appropriate patterns with their training process as the following steps:

```
Step 1: Let $q_1$ be the index of external iterations for training clusters
        Let $q_2$ be the index of internal iterations for training patterns
        Let u be the size of the current cluster
        Let $\rho$ be iteration's index for training patterns
        Let $\sigma$ be the maximum number of iterations for training patterns, which is large enough
        Let $\Pi_{\max1}$ be maximum number of iterations for the external loop (from cluster to next cluster)
        Let $\Pi_{\max2}$ be maximum number of iterations for the internal loop (from pattern to next pattern)
        Let m' = m'' = m''' = 1// the initial setting of the number of patterns for each cluster
        Let $\delta Re^1 = 50$;        //the initial value of the incremental loading

Step 2: For ($q_1$=0; $q_1 < \Pi_{\max1}$; next $q_1$)

Step 2-1: i = ($q_1$+1) mod 3; // we have 3 clusters

Step 2-2: If ( i = 1) then u = m';
              elseif (i = 2) then u = m'';
              elseif (i = 0) then u = m''';

Step 2-3: For ($q_2$=0; $q_2 < \Pi_{\max2}$; next $q_2$)

Step 2-3-1: j = ($q_2$+1) mod u;        // since there are u patterns in the cluster
Step 2-3-2: Implement FEA to find $P^s_{ji}$ for $Re^{j+1}$
            (Eq. 26-29)        // if it is not calculated before

Step 2-3-3:        For ($\rho$ =1; $\rho$ <= $\sigma$; next $\rho$)

Step 2-3-3-1:      Apply training on $P^s_{ji}$;

Step 2-3-3-2:      Update the value of weight functions (Eq. 38-39), with the smoothing (Eq. 35, 37) for the cluster $C_i$.

Step 2-3-3-3:      Using Eq. 40b to check the convergence of the training process for the pattern $P^s_{ji}$

Step 2-3-3-3-1:    If convergence is satisfied then break training.

Step 2-3-4:        Implement the proposed adaptive incremental loading:

                   if ( $\rho = \sigma$ )
                      if ($\delta Re \le 50$ )
                          $\delta Re = 50$;
                     else
                          $\delta Re = \delta Re /2$;
                 else if ($\rho<\sigma$)
                   if ($\delta Re \ge 250$ )
                      $\delta Re = 250$
                   else
                       $\delta Re$ = äRe * 2;

Step 2-3-5: Increment by one the number of patterns for the current $C_s$ {m', m'', or m'''}

Step 3:     Set the final number of patterns to each cluster, m', m'', m''', (Eq. 41).
```

Second phase of the training algorithm: Using DC sub-algorithm to reduce the effect of errors as the following steps:

```
Step 1: Call DC sub-algorithm (It is presented below).
```

Step 2: From clusters domain, select the ith cluster C_i that has a maximum sum of sub-cluster's error (MSE) (Eq. 43-44)

$$\mathrm{MCE} = \underset{\forall C_i}{\mathrm{Max}}\,(\mathrm{MSE}),\ i = 1...3 \tag{43}$$

Step 2-1: From sub-cluster SC at cluster i, select sub-cluster indexed s that has a maximum sum of neuron's error (Eq. 44a,b).

$$\mathrm{MSE} = \underset{\forall SC_s}{\mathrm{Max}}\,(\mathrm{ME}),\ s = 1...3 \tag{44a}$$

Step 2-1-1: For each pattern P^s_{ji} at the sub-cluster SC_s, find the maximum sum of neurons' errors Eq. 44b.

$$\mathrm{ME} = \underset{\forall P_j}{\mathrm{Max}} \left| \sum_{\forall\,\mathrm{neurons}} \left(\mathrm{Error}^{\mathrm{neurons}}\right) \right|,\ j = 1...3 \tag{44b}$$

Where:

$$\mathrm{Error}^{\mathrm{neurons}} = \left| O^{\mathrm{neurons}} - t^{\mathrm{neurons}} \right| \tag{44c}$$

Step 2-1-2: Apply training on the jth pattern P^s_{ji}

Step 3: If clusters' errors are not flat then go to step 2.

DC Sub-algorithm: Apply hierarchical clustering for each cluster as the following steps:

Step 1: For each cluster C_i, i = 1, 2, 3 and their sub-cluster SC_s, s = 1, 2, 3
Put $SC_s = \{\ \}\ \forall s$

Step 1-1: For every pattern P^s_{ji}

Step 1-1-1: For every element e in P^s_{ji} e = 1…100

Step 1-1-1-1: Find the angle θ_3 (Eq. 45).

$$\theta_3 = \begin{cases} |\theta_2 - \theta_1| & \text{for } \left(\frac{3\pi}{2} \le \theta_1, \theta_2 \le 2\pi\right) \vee \left(0 \le \theta_1, \theta_2 \le \frac{\pi}{2}\right) \\ & \vee \left(\frac{\pi}{2} \le \theta_2 \le \pi \ \&\ 0 \le \theta_1 \le \frac{\pi}{2}\right) \\ & \vee \left(\frac{\pi}{2} \le \theta_2 \le \pi \ \&\ \frac{2\pi}{3} \le \theta_1 \le 2\pi\right) \\ |\theta_2 + \theta_1| & \text{otherwise} \end{cases} \tag{45}$$

Step 1-1-2: Find Av (θ_3) (Eq. 46).

$$\mathrm{Av}(\theta_3) = \frac{1}{100} \sum_{e=1}^{100} \theta_3(e) \tag{46}$$

```
Step 1-1-3: if $(0 \le Av(\theta_3) \le \pi/2)$ then
        if $(40 \le Av(\theta_3) \le 50)$ then
            $SC_1 = SC_1 \cup \{ P^s_{ji} \}$
    else if $(20 \le Av(\theta_3) < 40) \vee (50 < Av(\theta_3) \le 70)$ then
            $SC_2 = SC_2 \cup \{ P^s_{ji} \}$
    else if $(0 \le Av(\theta_3) < 20) \vee (70 < Av(\theta_3) \le \pi/2)$ then
            $SC_3 = SC_3 \cup \{ P^s_{ji} \}$
```

```
Step 1-1-4: if $(\pi/2 \le Av\ (\theta_3) \le \pi)$ then
        if $(\pi - 50 \le Av\ (\theta_3) \le \pi - 40)$ then
            $SC_1 = SC_1 \cup \{ P^s_{ji} \}$
    else if $(\pi - 40 < Av\ (\theta_3) \le \pi - 20 \vee$
        $\pi - 70 \le Av\ (\theta_3) < \pi - 50)$ then
            $SC_2 = SC_2 \cup \{ P^s_{ji} \}$
    else if $(\pi - 20 < Av(\theta_3) \le \pi \ \vee$
        $\pi/2 \le Av\ (\theta_3) < \pi - 70)$ then
            $SC_3 = SC_3 \cup \{ P^s_{ji} \}$
```

```
Step 1-1-5: if $(\pi \le Av\ (\theta_3) \le 3\pi/2)$ then
    if $3\pi/2 - 40 \le Av\ (\theta_3) \le 3\pi/2 - 50$ then
            $SC_1 = SC_1 \cup \{ P^s_{ji} \}$
    else if $(3\pi/2 - 40 < Av\ (\theta_3) \le 3\pi/2 - 20 \vee$
        $(3\pi/2 - 70 \le Av\ (\theta_3) < 3\pi/2 - 50$ then
            $SC_2 = SC_2 \cup \{ P^s_{ji} \}$
    else if $(3\pi/2 - 20 < Av\ (\theta_3) \le 3\pi/2 \ \vee$
        $\pi \le Av\ (\theta_3) < \pi/2 - 70)$ then
            $SC_3 = SC_3 \cup \{ P^s_{ji} \}$
```

```
Step 1-1-6: if $(3\pi/2 \le Av\ (\theta_3) \le 2\pi)$ then
        if $(2\pi - 50 \le Av\ (\theta_3) \le 2\pi - 40$ then
            $SC_1 = SC_1 \cup \{ P^s_{ji} \}$
else if $(2\pi - 40 < Av\ (\theta_3) \le 2\pi - 20 \vee (2\pi - 70 \le Av\ (\theta_3) < 2\pi - 50)$
then
            $SC_2 = SC_2 \cup \{ P^s_{ji} \}$
else if $(2\pi - 20 < Av\ (\theta_3) \le 2\pi \vee 3\pi/2 \le Av\ (\theta_3) < 2\pi - 70)$ then
            $SC_3 = SC_3 \cup \{ P^s_{ji} \}$
```

Step 2: Using $\Gamma^{P}_{\mathrm{Average}}$ Eq. 42a-b to sort patterns for each sub-cluster s.

RESULTS AND DISCUSSION

To construct a learning system for compressible fluid flow on two-dimensional channels as shown in Fig. 1, it is necessary to find number of patterns for each cluster and sub-cluster for those channels. These patterns are selected through the training system stages (Fig. 2). The numerical model using FEA with adaptive incremental loading is constructed from Eq. 18, 22 and 26 and executed them simultaneously to achieve these patterns.

The symmetrical double steps channel's domain is divided into finite elements using adaptive mesh point technique (El-Emam, 2006; El-Emam and Shaheed, 2008) as shown in Fig. 1. This technique is used to provide an adequate mesh point refinement at the critical region of the specific channel. We suggest that the sufficient number of elements used in this study is equal to 100 elements.

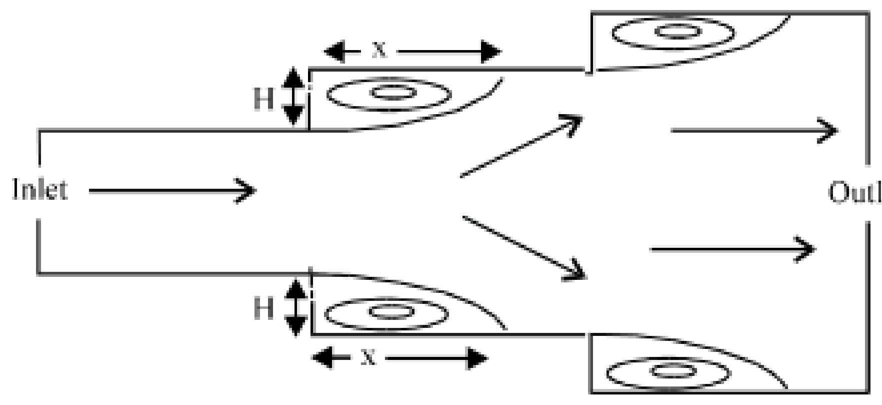


Fig. 7a: Symmetrical double backward step channel with expansion ratio = 2/3

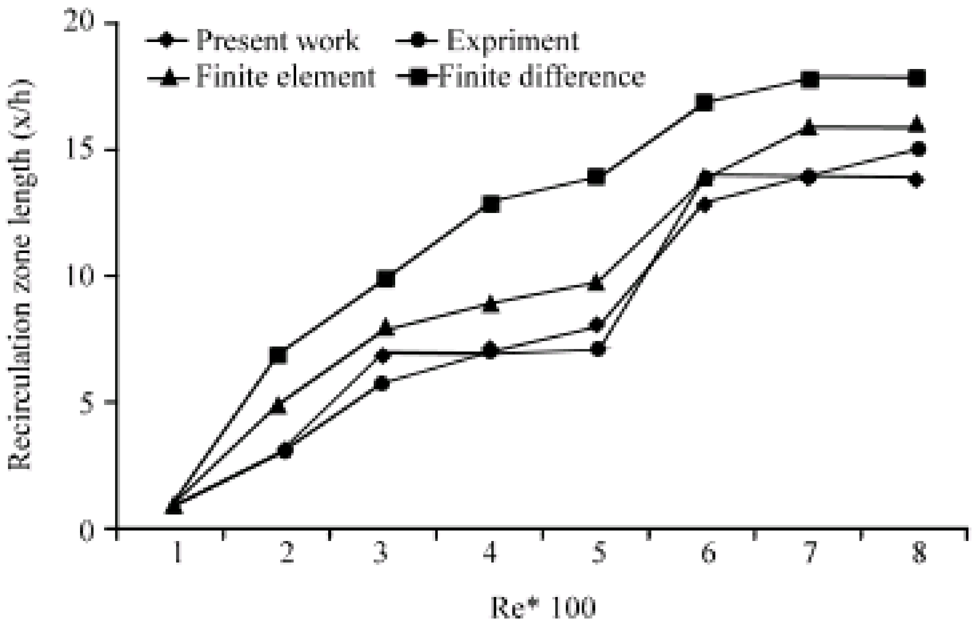


Fig. 7b: Recirculation length in a single step channel vs Reynold No.

It is important to check if the proposed numerical simulation using FEM with adaptive incremental loading leads to physically coherent and accurate results. As a consequence, the proposed approach is compared with the existing experimental and numerical literatures (Glowinski and Neittaanmaki, 2008; Vodinh *et al*., 2005). This comparison is implemented on steps channel with expansion ratio equal to 2/3 (Fig. 7a) to show the variation of the length of the separated region as function of Re numbers. The agreement appears to be fairly good over all Re numbers as shown in Fig. 7b.

In addition, the dependence of the recirculation length (normalized with respect to the step height) on Re is shown in Fig. 7b and it is apparent that the ratio X/h increases almost linearly with Re, reaches a peak value (~7) at laminar regime conditions and reaches a constant value (~14) at transition/turbulent conditions.

This study concentrates on the training system and percept the essential problems that are playing basic role on the training efficiency. Such problems are evaluated and removed eventually through AL with DC technique. In this section, we define a set of factors to speed up training system and to reduce the effects of errors.

Channel geometry factor: The training system is applied on three types of channels that were shown in Fig. 1. The channel geometry affects the speed of training as shown in Fig. 8, in which we observe that channel 3 consumes a minimum number of iterations due to the fully slopped enlargements of the channel shape. This type of channel makes two vortices near the steps are slipped to the outlet with minimum mixing, whereas channel 2 consumes a maximum number of iterations due to the partially slopped the channel's steps which generates higher velocities near the wall and push vortex region into the center of channel to generate the maximum mixing. Reduction of errors is achieved when the training is forwarded through clusters starting from the laminar regime.

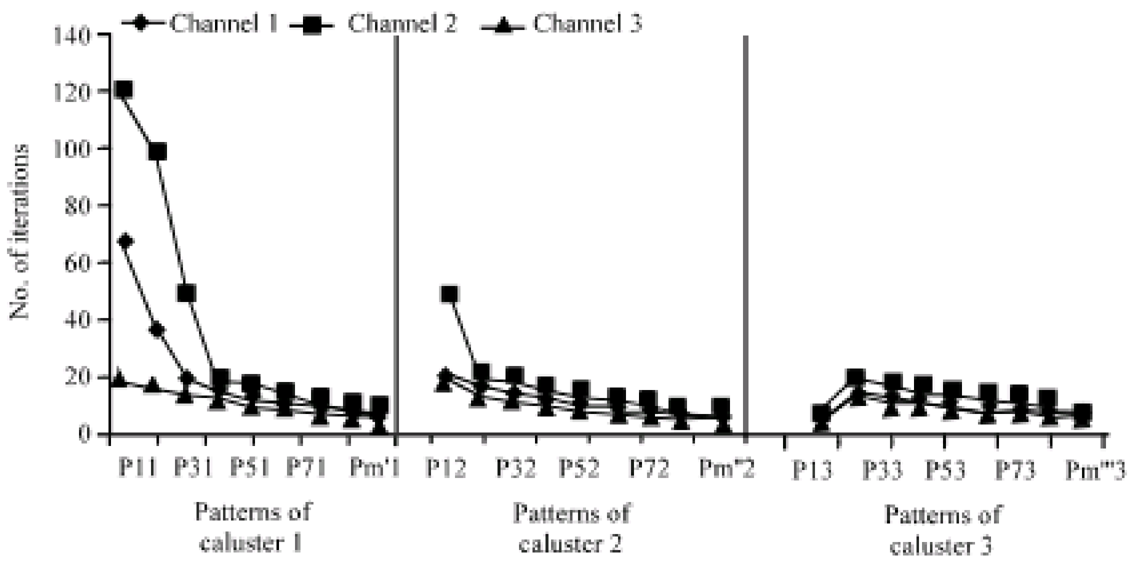


Fig. 8: Speed of training on three types of channels

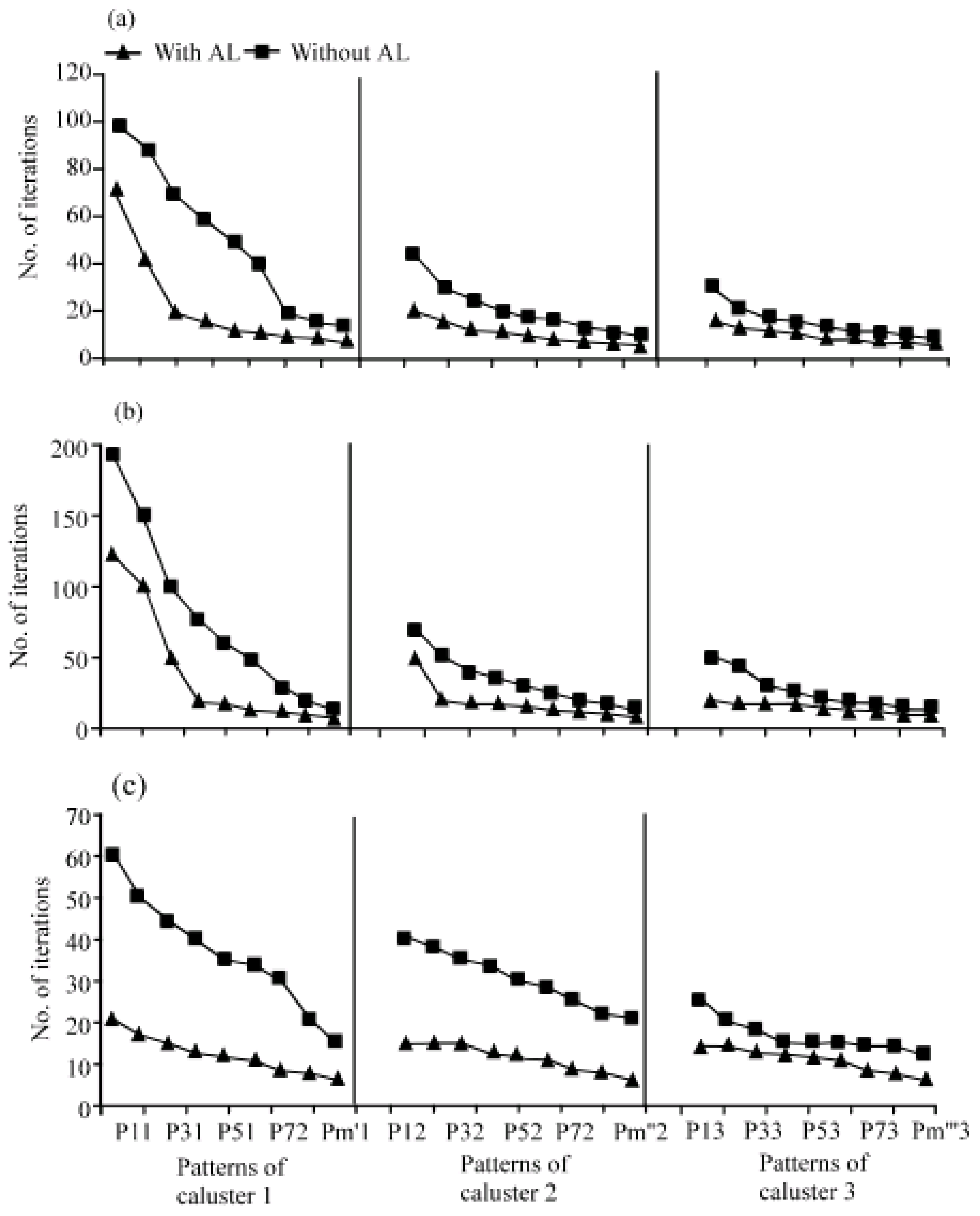


Fig. 9: Speed of training with /without AL on (a) channel 1, (b) channel 2 and (c) channel 3

Rate of learning factor: For training process, two approaches are proposed. The first is based on using adaptive learning rate (AL) on the learning factor α according to Eq. 40a, where, the values of parameters K and γ are estimated to 0.015 and 0.8, respectively. The second concept is to fix the value of learning rate α. Figure 9a-c show that the speed of training process with AL is more stable and faster than the training without AL.

The initial rate of learning factor: This factor is very important to speed up training process and to get high accuracy too. Accordingly, we study the behavior of the present training process with respect to the number of iterations and errors distribution. Figure 10 shows the speed of training process with respect to different values of learning rate (α) on three channels. The maximum peak is appeared at (α = 0.1) while the minimum peak appears at (α = 0.8). The best result is reached when (α = 0.7) is selected due to the minimum error distribution (Fig. 11).

Using DC factor: In the second phase of the proposed training algorithm, DC is used effectively to reduce the effects of errors and the order of patterns for each sub-cluster. These are playing basic roles on the amount of damping error, so that it is necessary to sort patterns in ascending order according to $\Gamma^{P}_{\text{Average}}$ for each sub-cluster. On the other hand, DC is not promising when patterns are selected randomly or sorted according to Euclidean distance presented in Eq. 47a, where, I^P and O^P are the input and output signals of the pattern P.

$$D_P(I,O)=\sqrt{\sum_{i=1}^{n}(I_i^P-O_i^P)^2} \qquad (47a)$$

Figure 12a-c show the comparison between sum of pattern's errors and rate of learning with/without DC and with/without sorting patterns for each sub-cluster in the three channels.

In addition, the learning process with DC is worked well when a pattern with a maximum total error (Eq. 43, 44) is selected rather than selecting the one that includes neuron with maximum error (Eq. 47a-c).

$$MCE' = \underset{\forall C_i}{Max}(MSE'),\ i = 1..3 \quad (47b)$$

$$MSE' = \underset{\forall SC_s}{Max}(ME'),\ s = 1..3 \quad (47c)$$

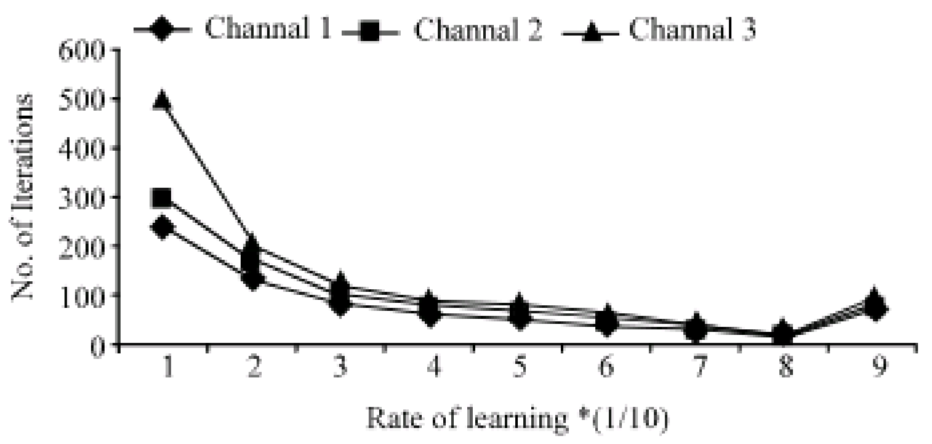


Fig. 10: Speed of training process with respect to the rate of learning

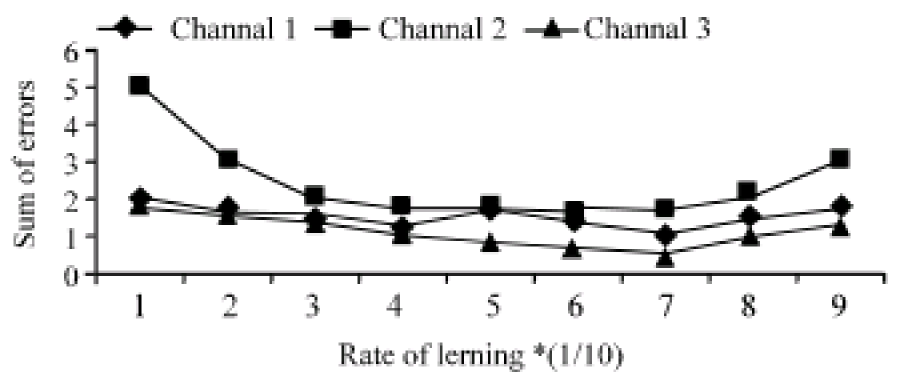


Fig. 11: Sum of pattern's errors with respect to the rate of learning

$$ME' = \underset{\forall P}{Max}\left|\left(Error^{neuron}\right)_{\forall neuron}\right|,\ P = 1..3 \quad (47d)$$

Figure 13a-c show the level of error if the pattern is selected according to Eq. 43, 44 and Eq. 47b-d for the three channels. It appears that the error factor is oscillated through the steps of dapping error, whereas training process is more stable if pattern is selected according to Eq. 43, 44. In addition, the error remains the same for all types of channels when Eq. 43, 44 are implemented on the contrary to that implementing Eq. 47b-d, which show instability and changes from channel to another. It is obvious that the maximum error occurs at channel 2 while the minimum error occurs at channel 3.

Fluid flow behavior and number of patterns factor: The behavior of the fluid flow is also affected when patterns are selected by using adaptive incremental loading γRe. Figure 14a shows the number of patterns for each cluster at each channel and Fig. 14b-d show the number of patterns for each sub-cluster at the three channels. It appears that channel 2 needs a maximum number of patterns equal to 223 that are apportioned on three clusters C_1, C_2 and C_3 with 140, 43 and 40 patterns, respectively. While channel 1 needs 137 patterns; these patterns are apportioned on three clusters C_1, C_2, C_3 with 70, 37 and 30 patterns, respectively. Finally, channel 3 needs a minimum number of patterns equal to 104; these patterns are apportioned on three clusters C_1, C_2, C_3 with 43, 31 and 30 patterns, respectively.

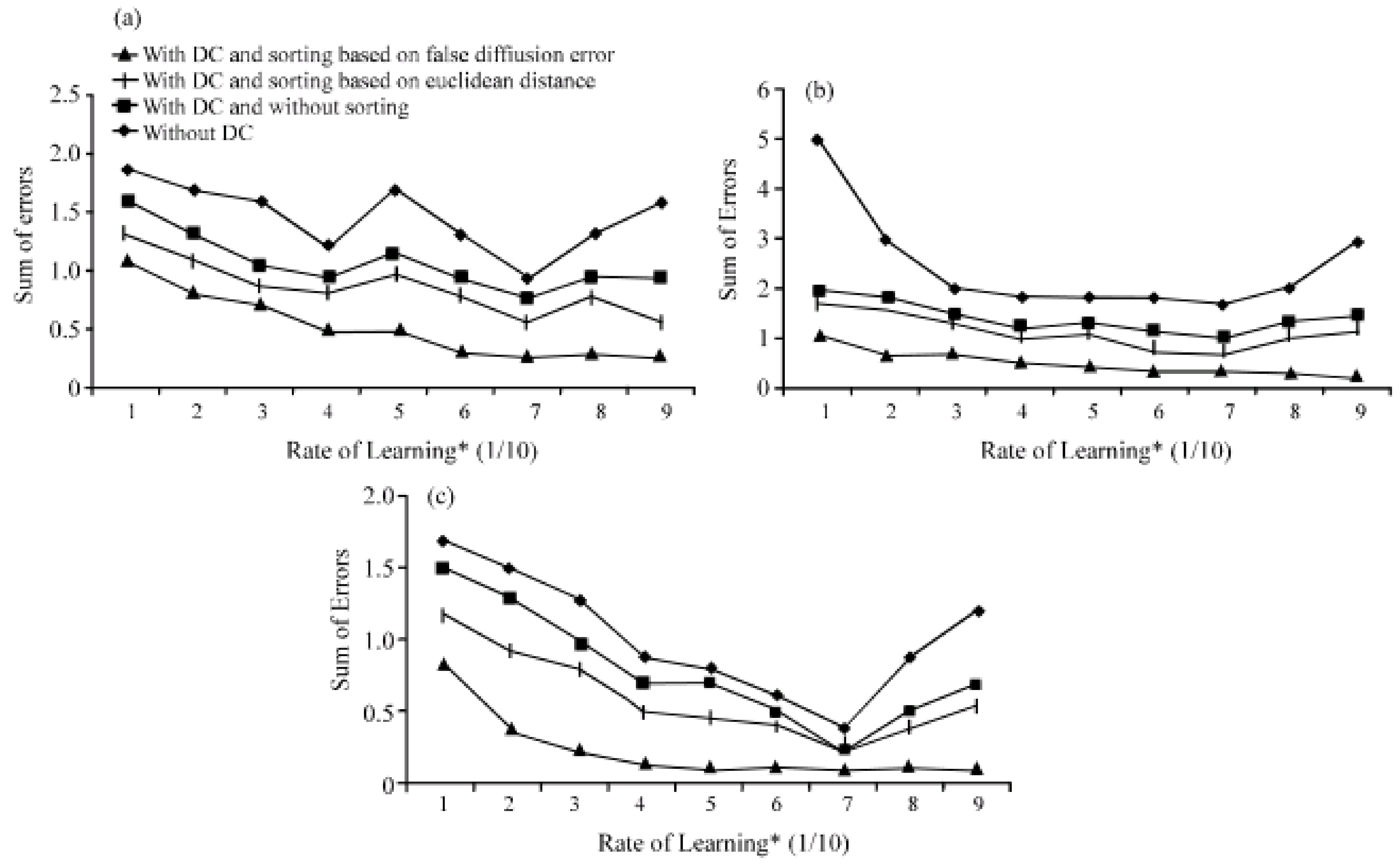


Fig. 12: Comparison between sums of pattern's errors vs. rate of learning for (a) channel 1, (b) channel 2 and (c) channel 3

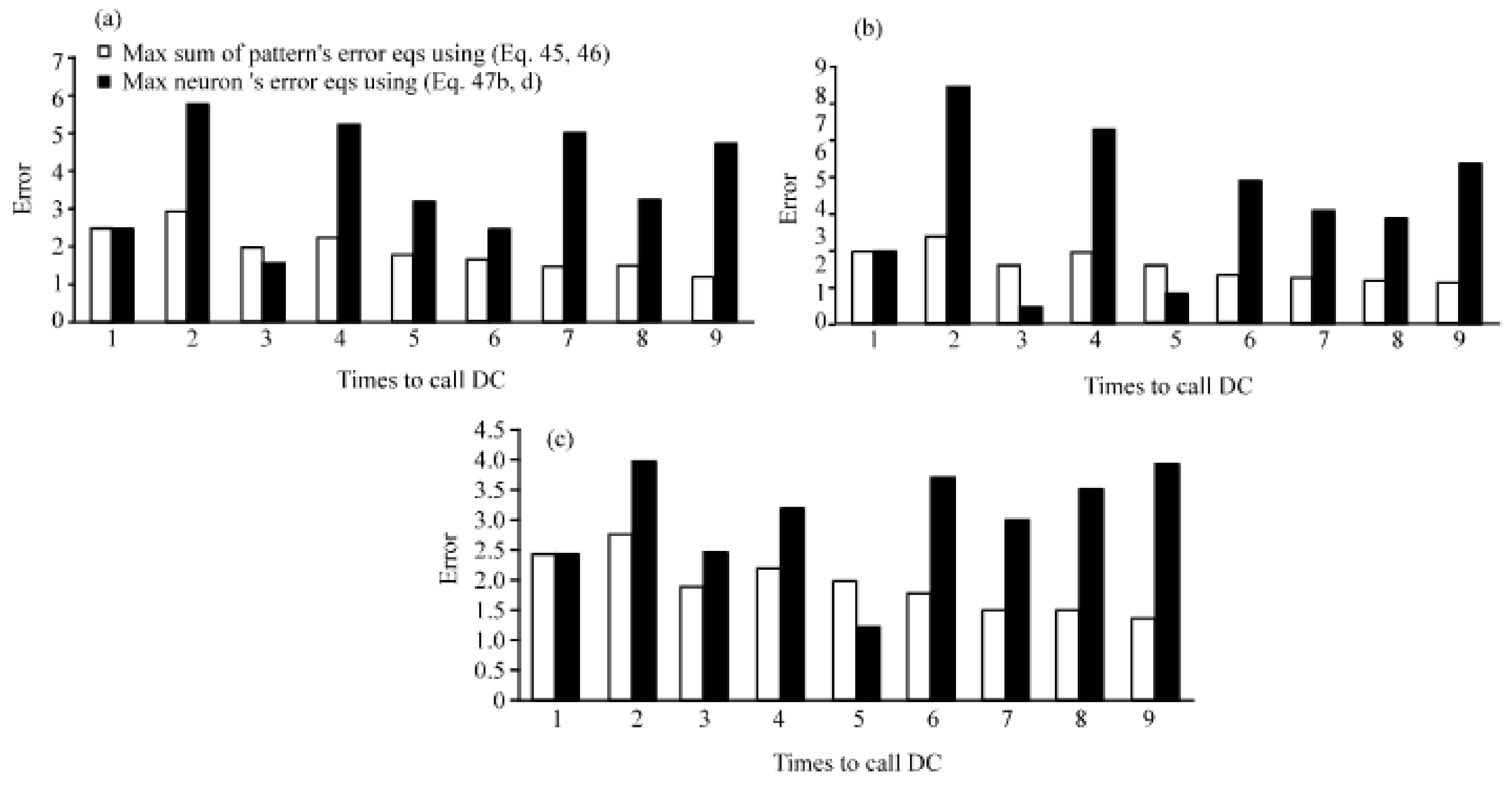


Fig. 13: Comparison between methods of selecting patterns for (a) channel 1, (b) channel 2 and (c) channel 3

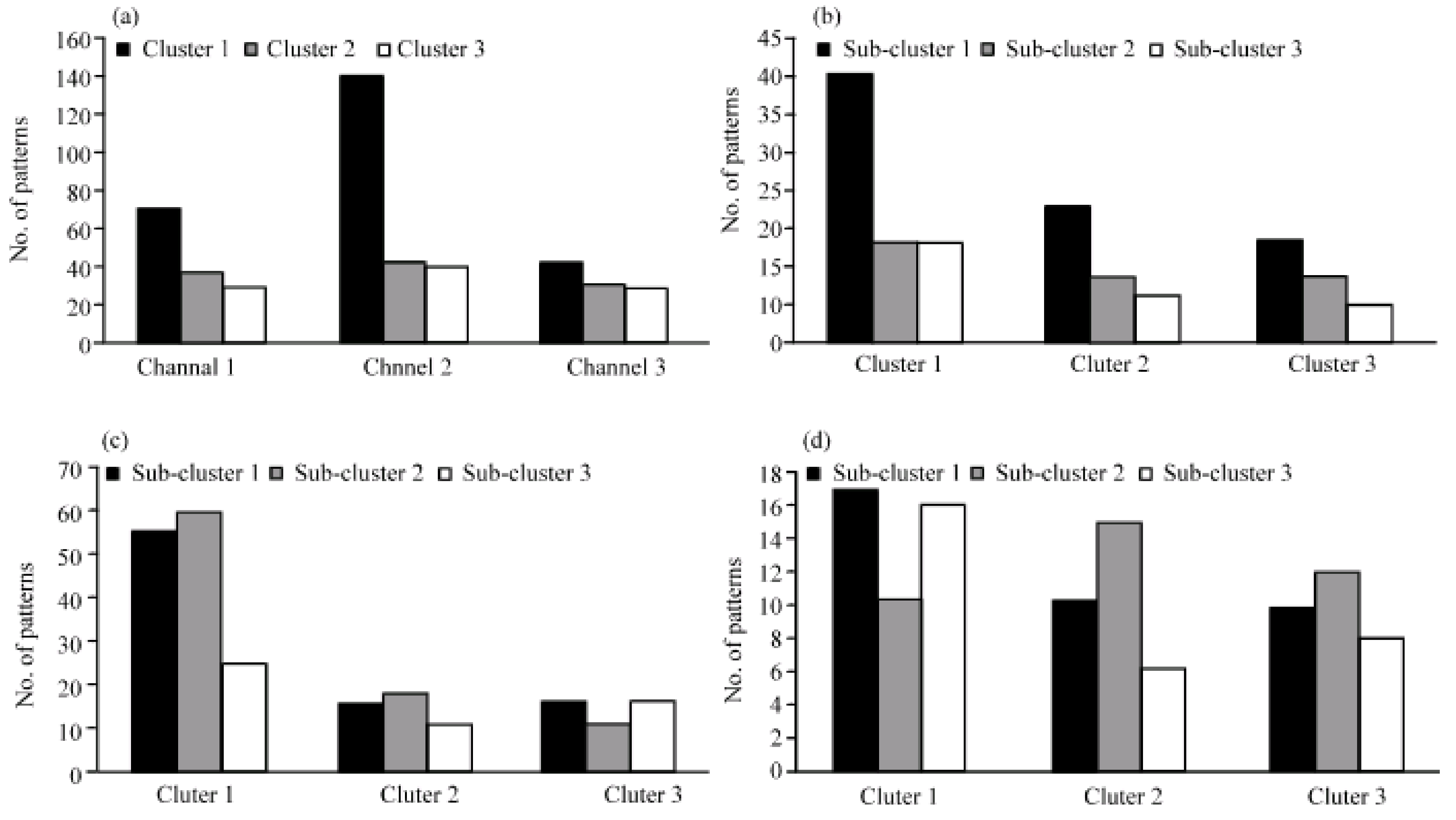


Fig. 14: No. of patterns for each (a) cluster at each channel, (b) sub-cluster at channel 1, (c) sub-cluster at channel 2 and (d) sub-cluster at channel 3

COMPUTER SIMULATION RESULTS

The proposed learning system calculates the velocity of the compressible flow through double steps channels for various values of Re in the interval (100-4000). Figure 15a-d-17a-d show these results. It appears that small vortexes exist at the steps of channels. As Re increases, the vortices region extends in length and the upper vortex spins over the lower vortex. The interaction of two vortices diverts both of them slightly upward into the center of channel especially for channel 2. Strong interaction has appeared when the value of Re is

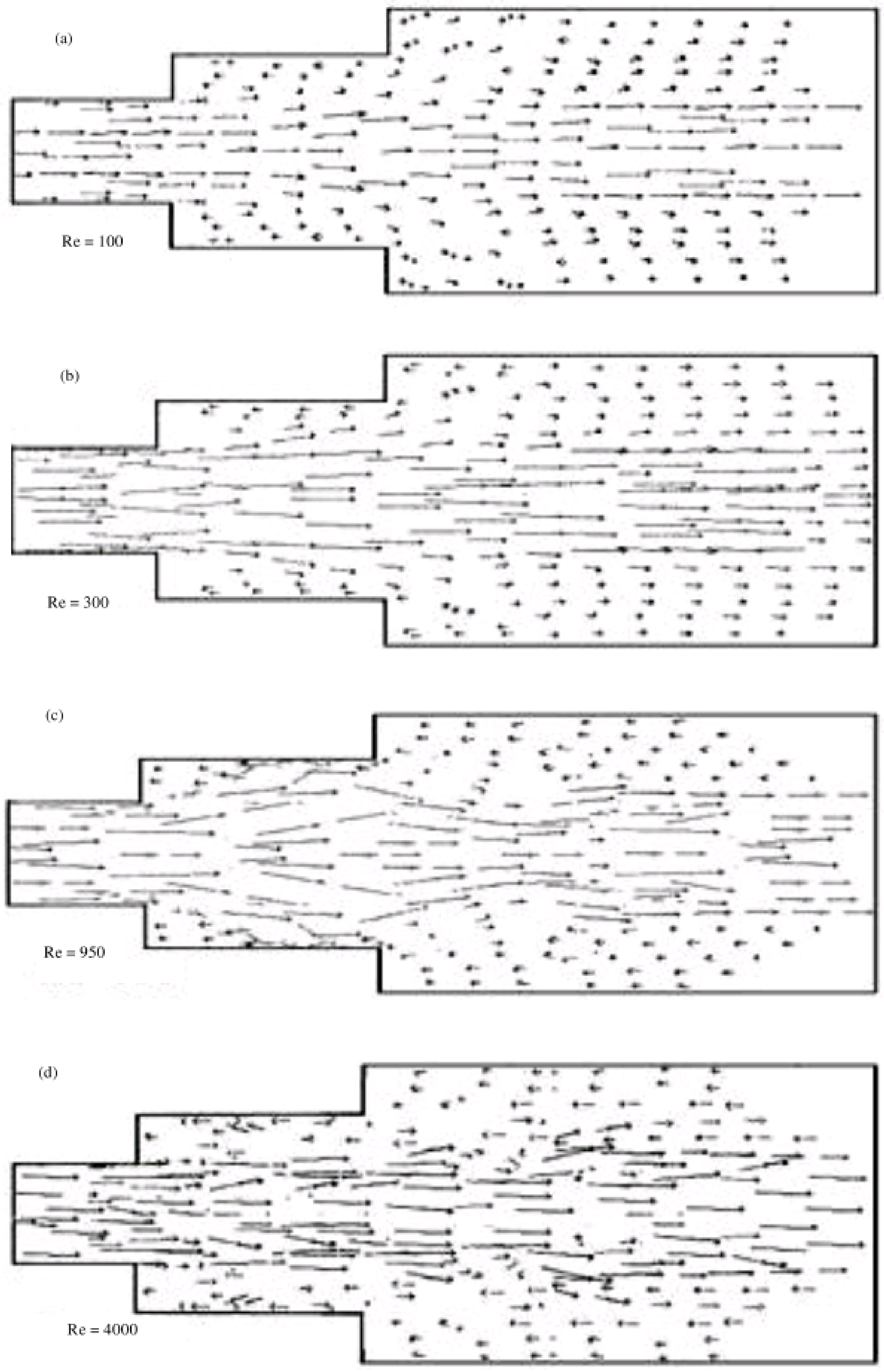


Fig. 15: (a-d) Velocity profile for compressible flow through channel 1 (Re = 100-4000)

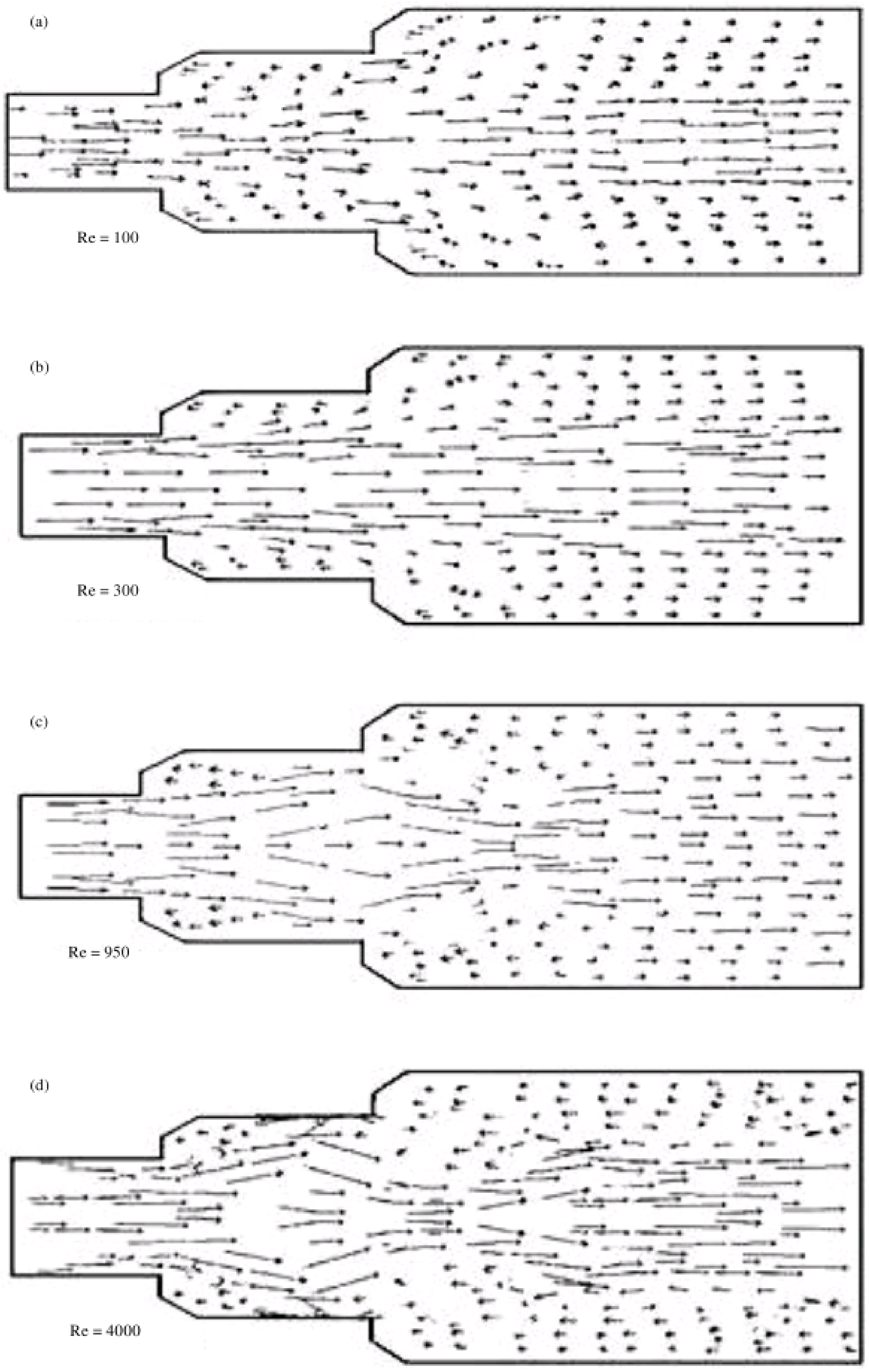


Fig. 16: (a-d) Velocity profile for compressible flow through channel 2 (Re = 100-4000)

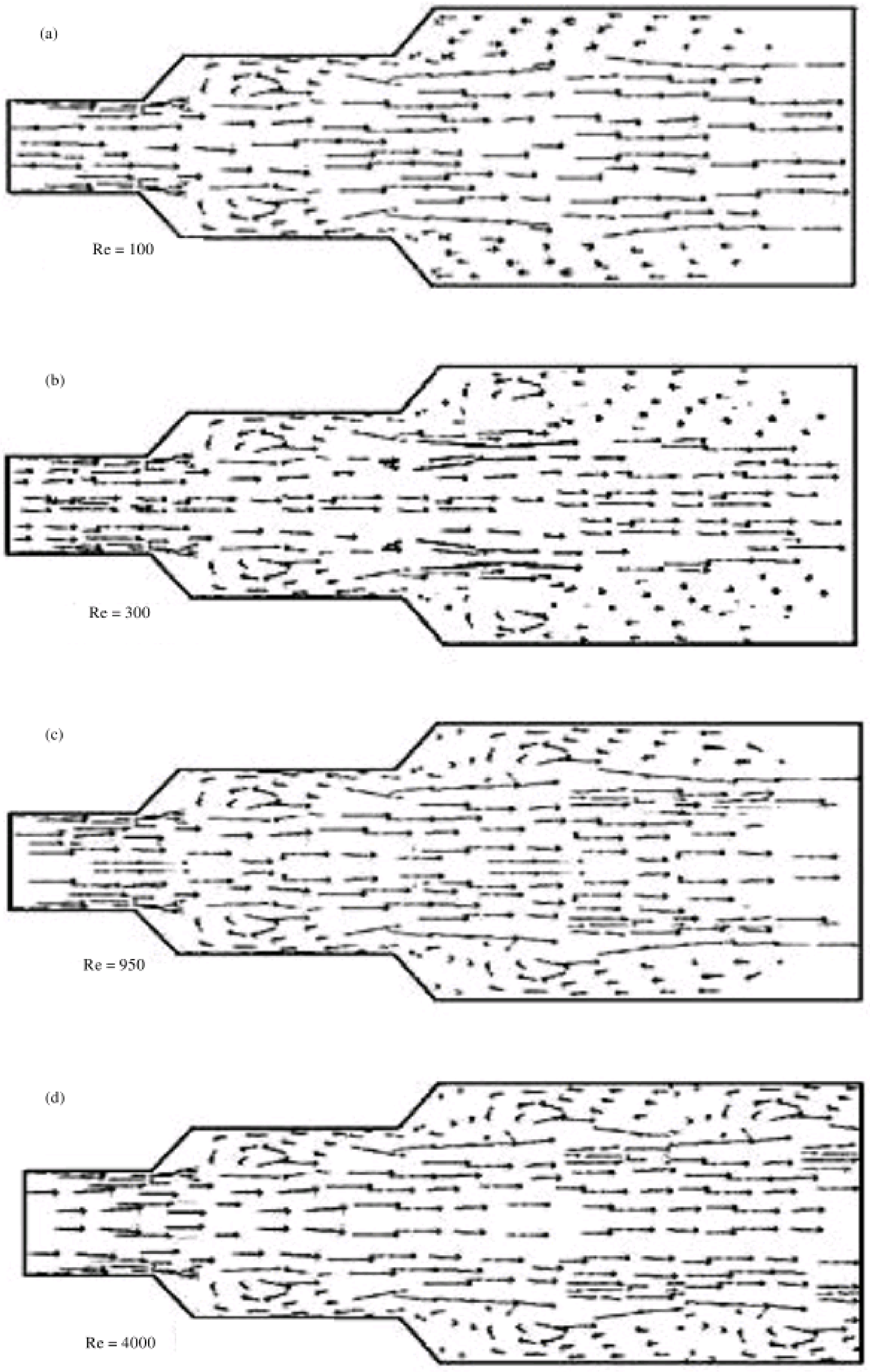


Fig. 17: (a-d) Velocity profile for compressible flow through channel 3 (Re = 100-4000)

increased. In addition, the new small vortices are generated between two main vortices especially for channel 1 and channel 2.

CONCLUSION

The present study demonstrates a new training algorithm based on two phases through a successful ANN with DC. The proposed algorithm is used to predict compressible fluid flow through specific type of channels geometry (Fig. 1). FEA with a new approach named adaptive incremental loading is utilized at the first phase of the training algorithm to prepare appropriate patterns. This studies simultaneously beside a training system using ANN with DC. The proposed structure of the neural network, which works through back propagation algorithm, includes one input, one hidden and one output layers. It has been shown that it is possible to use few patterns to simulate data with reasonable accuracy for three clusters (laminar, transition and turbulent) that are represented by velocity profile function and it gives encouraging results in many fields of applications (fluid mixing, fluid agitation, etc.) (Yu and Morales, 2005; Rajkumar and Bardina, 2002; Creusé and Mortazavi, 2004). The success of the proposed training algorithm can be attributed to three factors. The first is the employment of neural network that is an excellent approximator to any type of flow through any type of channel geometry (regular or irregular shapes) specially, if training process starts from laminar flow patterns (cluster 1). The second factor is the speed of training due to the use of ANN with proper value of AL on the minimum number of selected patterns and the third factor is satisfying the stability and the accuracy for high range of Re due to the use of MASE and DC based on $\Gamma^{P}_{\text{Average}}$. Finally, a channel's shape is playing basic role on the speed of training, accuracy of results and the number of patterns.

ACKNOWLEDGMENT

Authors would like to thank Prof. R.H. Al- Rabeh from Cambridge University for his support and help. This support is gratefully acknowledged.

REFERENCES

Creusé, E. and I. Mortazavi, 2004. Simulation of low Reynolds number flow control over a backward-facing step using pulsed inlet velocities. Applied Math. Res. Express, 4: 133-152.

EL-Emam, N.N., 2006. Reallocation of mesh points in fluid problems using Back-Propagation algorithm. J. Inform., 9: 175-184.

EL-Emam, N.N. and R.A. Shaheed, 2008. Computing an adaptive mesh in fluid problems using neural network and genetic algorithm with adaptive relaxation. Int. J. Artif. Intell. Tool, 17: 1089-1108.

Eker, E. and A. Serhat, 2006. Simulation of fluid Flow in synthetic fractures. Transport Porous Media, 65: 363-384.

Frean, M., M. Lilley and P. Boyle, 2006. Implementing Gaussian process inference with neural networks. Int. J. Neural Syst., 16: 321-327.

Frossyniotis, D.S., C. Pateritsas and A. Stafylopatis, 2005. A multi-clustering fusion scheme for data partitioning. Int. J. Neural Syst., 15: 391-401.

Glowinski, R. and P. Neittaanmaki, 2008. Partial Differential Equations: Modeling and Numerical Simulation. Springer, New York.

Gölcü, M., 2006. Neural network analysis of head-flow curves in deep well pumps. Energy Convers. Manage., 47: 992-1003.

He, Z., X. Xu and S. Deng, 2005. Tcsom: Clustering transactions using self-organizing map. Neural Process. Lett., 22: 249-262.

Hovakimyan, N., F. Nardi and A.J. Calise, 2002. Adaptive output feedback control of uncertain systems using single hidden layer neural networks. IEEE Trans. Neural Network., 13: 1420-1431.

Kaczmarczyk, L. and Z. Waszczyszyn, 2005. Neural procedures for the hybrid FEA/NN analysis of elastoplatic plates. Comput. Assist. Mech. Eng. Sci., 12: 379-391.

Lefik, M. and M. Wojciechowski, 2005. Artificial neural network as a numerical form of effective constitutive law for composites with parameterized and hierarchical microstructure. Comput. Assist. Mech. Eng. Sci., 12: 183-194.

Meybod, M.R. and H. Beigy, 2002. New learning automata based algorithms for adaptation of back propagation algorithm parameters. Int. J. Neural Syst., 12: 45-67.

Payne, S.J., 2006. A model of the interaction between auto regulation and neural activation in the brain. Math. Biosci., 204: 260-281.

Rajkumar, T. and J. Bardina, 2002. Prediction of aerodynamic coefficients using neural networks for sparse data. Proceedings of the FLAIRS Conference, May 2002, FLAIRS, pp: 242-246.

Silvestre, M.R., S.M. Oikawa, F.H.T. Vieira and L.L. Ling, 2008. A clustering based method to stipulate the number of hidden neurons of MLP neural networks: Applications in pattern recognition. Tend. Mat. Applied Comput., 9: 351-361.

Singh, R.P. and Z. Li, 2007. A mass conservative streamline tracking method for three-dimensional CFD velocity fields. J. Flow Visual Image Process., 14: 107-120.

Vodinh, L., B. Podvin and P. Le Quéré, 2005. Flow estimation using neural network. Proceedings of the Turbulence and Shear Flow Phenomena, Jun. 27-27, Williamsburg, VA, USA., pp: 1-1.

Yu, W. and A. Morales, 2005. Neural networks for the optimization of crude oil blending. Int. J. Neural Syst., 15: 377-389.