Neural Network Modeling of Hydrodynamics Processes

Sergey Valyuhov, Alexander Kretinin and Alexander Burakov
Voronezh State Technical University
Russia

1. Introduction

Many of the computational methods for equation solving can be considered as methods of weighted residuals (MWR), based on the assumption of analytical idea for basic equation solving. Test function type determines MWR specific variety including collocation methods, least squares (RMS) and Galerkin's method. MWR algorithm realization is basically reduced to nonlinear programming which is solved by minimizing the total equations residual by selecting the parameters of test solution. In this case, accuracy of solving using the MWR is defined by approximating test function properties, along with degree of its conformity with its initial partial differential equations, representing a continuum solution of mathematical physics equations.
On fig. 1, computing artificial neural network (ANN) is presented in graphic form, illustrating process of intra-network computations. The input signals or the values of input

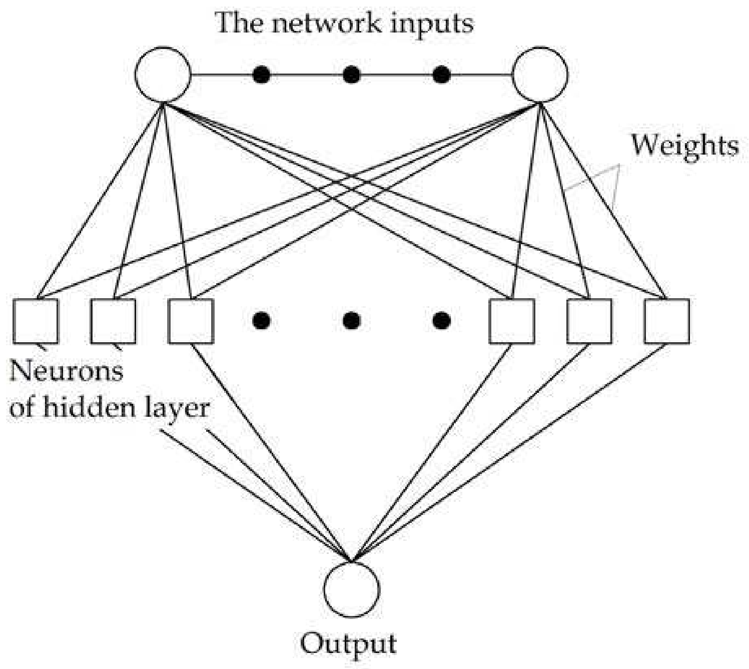


Fig. 1. Neural network computing structure

variables are distributed and "move" along the connections of the corresponding input together with all the neurons of hidden layer. The signals may be amplified or weakened by being multiplied by corresponding coefficient (weight or connection). Signals coming to a certain neuron within the hidden layer are summed up and subjected to nonlinear transformation using so-called activation function. The signals further proceed to network outputs that can be multiple. In this case the signal is also multiplied by a certain weight value, i.e. sum of neuron output weight values within the hidden layer as a result of neural network operation. Artificial neural networks of similar structure are capable for universal approximation, making possible to approximate arbitrary continuous function with any required accuracy.
To analyze ANN approximation capabilities, perceptron with single hidden layer (SLP) was chosen as a basic model performing a nonlinear transformation from input space to output space by using the formula (Bishop, 1995):

$$y(\mathbf{w},\mathbf{x}) = \sum_{i=1}^{q} v_i f_\sigma\left(b_i + \sum_{j=1}^{n} w_{ij} x_j\right) + b_0\,, \tag{1}$$

where $\mathbf{x} \in \mathbf{R}^n$ is network input vector, comprised of x_j values; q – the neuron number of the single hidden layer; $\mathbf{w} \in \mathbf{R}^s$ – all weights and network thresholds vector; w_{ij} – weight entering the model nonlinearly between j-m input and i-m neuron of the hidden layer; v_i – output layer neuron weight corresponding to the i-neuron of the hidden layer; b_i, b_0 – thresholds of neurons of the hidden layer and output neuron; f_σ – activation function (in our case the logistic sigmoid is used). ANN of this structure already has the universal approximation capability, in other words it gives the opportunity to approximate the arbitrary analog function with any given accuracy. The main stage of using ANN for resolving of practical issues is the neural network model training, which is the process of the network weight iterative adjustment on the basis of the learning set (sample) $\{\mathbf{x}_i, y_i\}$, $\mathbf{x}_i \in \mathbf{R}^n$, $i = 1,...,k$ in order to minimize the network error – quality functional

$$J(\mathbf{w}) = \sum_{i=1}^{k} Q\left(f_\varepsilon(\mathbf{w},i)\right), \tag{2}$$

where $\mathbf{w}$ – ANN weight vector; $Q\left(f_\varepsilon(\mathbf{w},i)\right) = f_\varepsilon(\mathbf{w},i)^2$ – ANN quality criterion as per the i-training example; $f_\varepsilon(\mathbf{w},i) = y(\mathbf{w},\mathbf{x}_i) - y_i$ – i-example error. For training purposes the statistically distributed approximation algorithms may be used based on the back error propagation or the numerical methods of the differentiable function optimization.

2. Neuronet's method of weighted residuals for computer simulation of hydrodynamics problems

Let us consider that a certain equation with exact solution $\overline{y}(x)$

$$L(\overline{y}) = 0 \tag{3}$$

for non-numeric value y^s equation (3) presents an arbitrary $\mathbf{x}^s$ within the learning sample. We have $L(y)=R$ with substitution of approximate solution (1) into equation (3), with R as equation residual. R is continuous function $R=f(\mathbf{w},\mathbf{x})$, being a function of SLP inner

parameters. Thus, ANN training under outlet functional is composed of inner parameters definition using trial solution (1) for meeting the equation (3) goal and its solution is realized through the corresponding modification of functional quality equation (2) training.
Usually total squared error at net outlets is presented as an objective function at neural net training and an argument is the difference between the resulted '*s*' net outlet and the real value that is known a priori. This approach to neural net utilization is generally applied to the problems of statistical set transformation along with definition of those function values unknown a priori (net outlet) from argument (net inlet). As for simulation issues, they refer to mathematical representation of the laws of physics, along with its modification to be applied practically. It is usually related to necessity for developing a digital description of the process to be modeled. Under such conditions we will have to exclude the a priori known computation result from the objective function and its functional task. Objective function during the known law simulation, therefore, shall only be defined by inlet data and law simulated:

$$E = \frac{1}{2}\sum_{S}\left(y^{s} - f\left(\mathbf{x}^{s}\right)\right)^{2}. \qquad (4)$$

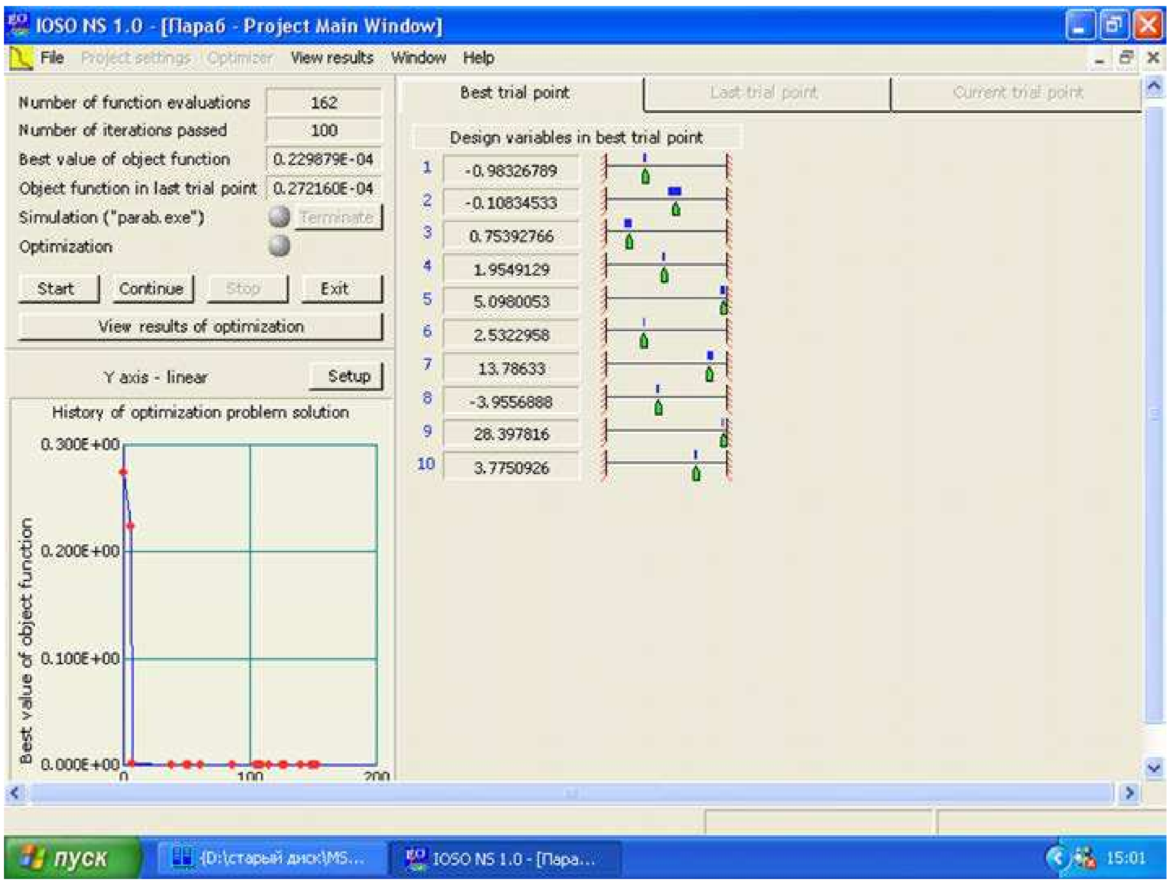
Use of neuronet's method of weighted residuals (NMWR) requires having preliminary systematic study for each specific case, aimed at: 1) defining the number of calculation nodes (i.e. the calculation grid size); 2) defining number of neurons within the network, required for obtaining proper approximation power; 3) choosing initial approximations for training neural network test solution ; 4) selecting additional criteria in the goal function for training procedure regularization in order to avoid possible solution non-uniformity; 5) analyzing the possibilities for applying multi-criteria optimization algorithms to search neural network solution parameters (provided that several optimization criteria are available).
Artificial neural network used for hydrodynamic processes studying is presented by two fundamentally different approaches. The first is the NMWR used for direct differential hydrodynamics equations solution. The NMWR description and its example realization for Navier-Stokes equations solution is presented in papers (Kretinin, 2006; Kretinin et al., 2008). These equations describe the 2D laminar isothermal flow of viscous incompressible liquid. In the paper (Stogney & Kretinin, 2005), the NMWR is used for simulating flows within a channel with permeable wall. Neural network solution results of hydrodynamic equations for the computational zone consisting of two sub-domains are presented below. One is rotating, while another is immobile. In this case, for NMWR algorithm realization specifying the conjugate conditions at the two sub-domains border is not required.
In the second approach, neural network structures are applied to computational experiment results approximation obtained by using traditional methods of computational hydrodynamics and for obtaining of hydrodynamic processes multifactor approximation models. This approach is illustrated by hydrodynamics processes neural network modeling in pipeline in the event of medium leakage through the wall hole.

2.1 NMWR application: Preliminary studying

There are specific ANN training programs such as STATISTICA NEURAL NETWORKS or NEURAL TOOLBOX in the medium of MATLAB, adjusting the parameters of the network to the known values of the objective function within the given points of its definitional domain. Using these packages in our case, therefore, does not seem possible. At the same time, many of optimization standard methods work well for ANN training, e.g. the conjugate gradients methods, or Newton, etc. To solve the issue of ANN training, we shall use the Russian program IOSO NS 1.0 (designed by prof. I.N. Egorov (Egorov et al., 1998), see www.IOSOTech.com) realizing the algorithm of indirect optimization method based on self-organizing. This program allows minimizing the mathematical model given algorithmically and presented as “black box”, i.e. as external file module which scans its values from running variable file generated by optimization program, then calculates objective function value and records it in the output file, addressed in turn by optimization program. It is therefore sufficient for computer program forming, realizing calculations using the required neural network, where the input data will be network internal parameters (i.e. weights, thresholds); on the output, however, there’ll be value of required equation sum residual based on accounting area free points. Let us suppose that the objective function $y = x^2$ is determined within the interval $[0;1]$. It is necessary to define parameters of ANN perceptron type with one hidden layer, consisting of 3 neurons to draw the near-objective function with given accuracy, computed in 100 accounting points x_i evenly portioned in determination field. Computer program for computing network sum residual depending on its parameters can be as follows (Fortran):

```
      dimension x(100),y(100)
      dimension vs(10)
      common vs
c     vs- values of ANN internal parameters
      open(1,file='inp')
      read(1,*)vs
      close(1)
c     'inp'- file of input data,
c     generated by optimization program
      do i=1,100
      x(i)=(i-1)/99.
      end do
c     calculation by subprogram ANN ynet
c     and finding of sum residual del
      del=0.
      do i=1,100
      y(i)=ynet(x(i))
      del=del+(y(i)-x(i)**2)**2
      end do
c     'out'-file of value of the minimization function ,
c     sent to optimization program
      open(2,file='out')
      write(2,*)del
      close(2)
      end
      function ynet(t)
      dimension vs(10),w(3),b(3),v(3),t1(3),q(3)
      common vs
c     w-weights between neuron and input
c     b-thresholds of neurons
c     v-weights between neuron and output neuron
c     bv-threshold of output neuron
      do i=1,3
      w(i)=vs(i)
      b(i)=vs(i+3)
      v(i)=vs(i+6)
      end do
      bv=vs(10)
      vyh=0.
      do i=1,3
      t1(i)=w(i)*t-b(i)
      q(i)=1./(1.+exp(-t1(i)))
      vyh=vyh+v(i)*q(i)
      end do
      ynet=vyh-bv
      end
```

With IOSO NS 1.0, ANN internal parameter values were obtained with sum residual there were received the values of the internal parameters of the ANN, giving the sum $E = 0.000023$ (fig. 2).



Number of function evaluations	162
Number of iterations passed	100
Best value of object function	0.229879E-04
Object function in last trial point	0.272160E-04

Design variables in best trial point

1	-0.98326789
2	-0.10834533
3	0.75392766
4	1.9549129
5	5.0980053
6	2.5322958
7	13.78633
8	-3.9556888
9	28.397816
10	3.7750926

Fig. 2. Results of using IOSO NS 1.0 for the ANN training

Hence we have neural network approximation for given equation, which can be presented by the formula

$$y = 13.786\frac{1}{1+e^{1.954913+0.983267\cdot x}} - 3.95569\frac{1}{1+e^{5.098+0.108345\cdot x}} + 28.3978\frac{1}{1+e^{2.532-0.75393\cdot x}} - 3.7751 \quad (5)$$

Using nonlinear optimization universal program products for ANN training is limited to neural networks of the simplest structure, for dimension of optimization tasks solved by data packages does not normally exceed 100; however, it frequently forms 10-20 independent variables due to the fact that efficiency of neural network optimization methods generally falls under the greater dimensions of the nonlinear programming free task. On the other hand, the same neural network training optimization methods prove efficient under much greater dimensions of vector independent variables. Within the framework of given functioning, the standard program codes of neural network models are applied, using the well-known optimization procedures, e.g. Levenberg-Markardt or conjugate gradients - and the computing block of trained neural network with those obtained by the analytical expressions for objective function of the training anti-gradient components, which in composition of the equation under investigation acts as a "teacher" is designed.

2.2 Computing algorithm of minimization of neural network decision

Let us consider perceptron operation with one hidden layer from N neuron and one output (1). As training objective function, total RMS error (4) will be considered. The objective function shall be presented as a complex function from neural network parameters; components of its gradient shall be calculated using complex function formula. Network output, therefore, is calculated by the following formula:

$$y\left(\mathbf{x}^s\right) = \sum_j w_j \sigma_j\left(\mathbf{x}^s\right), \quad (6)$$

where **x** - vector of inputs, *s* - number of point in training sample, σ*(x)* - activation function, w_j - weights of output neuron, *j* - number of neuron in hidden layer. For activation functions, logistic sigmoid will be considered

$$\sigma_j(\mathbf{x}) = \frac{1}{1+e^{-t_j\left(\mathbf{x},b_j\right)}}. \quad (7)$$

Here b_j - threshold of *j*-number neuron of hidden layer; the function $t_j(\mathbf{x},b_j)$, however, has form of $t_j\left(\mathbf{x},b_j\right) = \sum_i v_{ij}\cdot x_i - b_j$, where v_i - neuron weight of hidden layer.

While training on each iterations (the epoch) we shall correct the parameters of ANN toward the anti-gradient of objective function - ∇E(**v,w,b**), which components are presented in the following form:

$$\frac{\partial E^s}{\partial w_{ij}} = \sigma_j\left(\mathbf{x}^s,b_j\right)\cdot\left(y^s - f\left(\mathbf{x}^s\right)\right); \quad (8)$$

$$\frac{\partial E^s}{\partial v_{ij}} = \left(y^s - f\left(\mathbf{x}^s\right)\right) \cdot w_j \cdot \sigma_j\left(\mathbf{x}^s, b_j\right) \cdot \left(1 - \sigma_j\left(\mathbf{x}^s, b_j\right)\right) \cdot x_i^s \; ; \tag{9}$$

$$\frac{\partial E^s}{\partial b_j} = -\left(y^s - f\left(\mathbf{x}^s\right)\right) \cdot w_j \cdot \sigma_j\left(\mathbf{x}^s, b_j\right) \cdot \left(1 - \sigma_j\left(\mathbf{x}^s, b_j\right)\right). \tag{10}$$

Thereby, we have got all the components of the gradient of the objective function of minimization, comparatively which iterations will be consecutively realized in accordance with the general formula

$$\Delta \mathbf{w} = -\varepsilon \nabla E(\mathbf{w}) \, . \tag{11}$$

Here **w** is vector of current values of network weights and thresholds.

3. Using NMWR for hydrodynamics equations solving

Parameter optimization of neural network trial solutions is achieved by applying several optimization strategies and by subsequently choosing the maximum effective one (see Cloete & Zurada, 2000). First strategy is to apply totality of effective gradient methods "starting" from various initial points. The other strategy is to apply structural-parametrical optimization to ANN training; this method is based on indirect statistic optimization method on self-organizing basis or parameter space research (see: Egorov et al., 1998; Statnikov & Matusov, 1995).
Any versions for multi-criterion search of several equations system solution are based on different methods of generating multiple solutions, satisfying Pareto conditions. Choosing candidate solution out of Pareto-optimal population must be based on analysis of hydrodynamic process and is similar to identification procedure of mathematical model. In any case, procedure of multi-criterion optimization comes to solving single-criterion problems, forming multiple possible solutions. At the same time particularities of some computational approaches of fluid dynamics allows using iteration algorithms, where on each step solution at only one physical magnitude is generated.

3.1 Modeling flows – the first step

The computational procedure described below is analogous to MAC method (Fletcher, 1991), investigating possibility of NMWR application based on neural net trial functions.

Laplace equation solution

Computational capabilities of the developed algorithm can be illustrated by the example of the solution of Navier-Stokes momentum equations, describing two-dimensional isothermal flows of viscous incompressible fluid. On the first stage we will be using this algorithm for Laplace equation solution

$$\frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} = 0 \, . \tag{12}$$

Let us consider the flow of incompressible fluid in the channel (fig. 3).

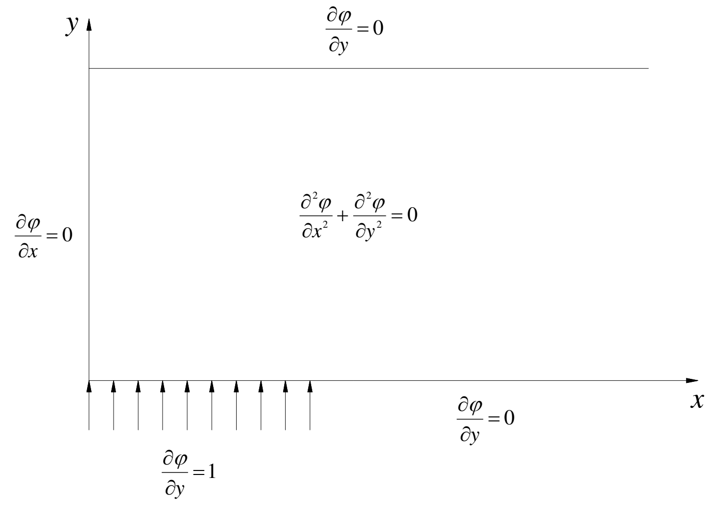

y
$\frac{\partial \varphi}{\partial y}=0$
$\frac{\partial^2 \varphi}{\partial x^2}+\frac{\partial^2 \varphi}{\partial y^2}=0$
$\frac{\partial \varphi}{\partial x}=0$
x
$\frac{\partial \varphi}{\partial y}=0$
$\frac{\partial \varphi}{\partial y}=1$

Fig. 3. Computational area

Here's how the boundary conditions are defined: on solid walls u=v=0, on inflow boundary u=0, v=1, on outflow boundary $\frac{\partial u}{\partial x}=\frac{\partial v}{\partial x}=0$. There are no boundary conditions for pressure except for one reference point, where p=0 is specified (in the absolute values p=p_0), considering which indication of incoming into the momentum equation $\frac{\partial p}{\partial x}$ and $\frac{\partial p}{\partial y}$ is realized.

For solving flow equations by predictor method it is necessary to specify initial velocity distribution within the computational area, satisfying the equation of continuity. For this purpose, velocity potential $\varphi(x,y)$ is introduced and $u=\frac{\partial \varphi}{\partial x}$ and $v=\frac{\partial \varphi}{\partial y}$. As a result of Laplace equation solution, velocity distribution is generated, which can be indicated as free-vortex component of the sought quantity.

If the result of learning sample neuronet calculations is defined by the following formula $\varphi(\mathbf{x}^s)=\sum_j v_j f_j(\mathbf{x}^s)$, where $\mathbf{x}=\{x,y\}^T$ -input variables vector, s - point number in the learning sample, $f(\mathbf{x})$ - activation function, v_j - output neuron weights, j - neuron number in the hidden layer as activation function the logistical sigmoid is used $f_j(\mathbf{x})=\frac{1}{1+e^{-t_j(\mathbf{x},b_j)}}$, where b_j - threshold of the j-number neuron hidden layer, and the function $t_j(\mathbf{x},b_j)$ looks like $t_j(\mathbf{x},b_j)=\sum_i w_{ij}\cdot x_i - b_j$ where w_{ij} - hidden layer neurons weights, then analytical expressions for the second speed potential derivatives can be calculated using the following formula

$$\frac{\partial^2 \varphi}{\partial \left(x_i^s\right)^2}=\sum_j \left(v_j w_{ij}^2 \left(f_j\left(\mathbf{x}^s,b_j\right)-3f_j^2\left(\mathbf{x}^s,b_j\right)+2f_j^3\left(\mathbf{x}^s,b_j\right)\right)\right). \tag{13}$$

Equation summary residual with substituted trial solutions (1) on arbitrary calculation area points with coordinates **x**ˢ with expressions application (13) can also be calculated analytically

$$E = \frac{1}{2}\sum_s \sum_i \left(\frac{\partial^2 \varphi}{\partial \left(x_i^s \right)^2} \right)^2 = \frac{1}{2}\sum_s \left(\frac{\partial^2 \varphi}{\partial \left(x^s \right)^2} + \frac{\partial^2 \varphi}{\partial \left(y^s \right)^2} \right)^2 . \tag{14}$$

Therefore, trial solution (1) training problem of neural network equation consists in SLP hidden layer parameter selection (weights and thresholds) at which the summary residual (14) has the minimal value limited to zero. The computer program described above, with training procedure target function being set functionally by applying analytical expressions for second derivatives $\frac{\partial^2 \varphi}{\partial x^2}$ and $\frac{\partial^2 \varphi}{\partial y^2}$, is used for parameter adjustment of learning model.

Efficiency of searching of neuronet learning solution parameters depends on problem dimension, i.e. weights and perceptron thresholds variable adjusted quantity. The more significant is neurons quantity in trial solution, the higher is ANN approximate capacity; however, achieving high approximation accuracy is more complicated. At the same time, neuron quantity depends not only on simulated function complexity, but also on calculation nodes quantity in which the residual equation is calculated. It is known that generally points' quantity increase in statistical set used for neural network construction is followed by increase in necessary neurons network (Galushkin, 2002; Galushkin, 2007) quantity. Consequently, the dense calculation grids application results in nonlinear programming problems; while applying rare calculation grids, it is necessary to check the solution realization between calculation nodes, i.e. there is a problem of learning solution procedure standardization. In the neuronet solution reception context on known equation, it is convenient using traditional additive parameter of training neural model quality - a control error which is calculated on the set of additional calculation nodes between calculation grid nodes. Number of these additional calculation grid nodes can be much more significant, and they should cover the whole calculation area, because the nodes number increase with control error on known network parameters does not result in essential computing expenses growth. Hence, referring to learning solution neuronet parameters reception, there exists an issue of solving twice-criterion problem of nonlinear optimization along with minimizing simultaneously both summary residual in control points, or the control error can appear as a restriction parameter, in the limited set of calculation nodes and in this case the neural network solution parameters reception is reduced to the conditional nonlinear optimization problem.

At the first stage, residual distribution of the current equation (5) on various calculation nodes and the corresponding speed vector distribution $\mathbf{v} = \{u, v\}^T$, where speed nodes $u = \frac{\partial \varphi}{\partial x}$ and $v = \frac{\partial \varphi}{\partial y}$. As a whole, the received neural network solution satisfies the equation (5) except for calculation nodes group, for example, in the input border right point vicinity, due to a sudden change of the boundary conditions in this point. In areas with the solution insufficient exactness we will place the calculation nodes additional quantity using the following algorithm. Let us formulate the Cohonen neural network with three inlet variables presented by the coordinates of available computation nodes x and y, and also the equations (5) residual value in these nodes, along with the required cluster centers quantity equal to the additional nodes quantity. The cluster center coordinates which will generally be placed in areas with the learning solution low precision (Prokhorov et al., 2001) we will consider additional computation nodes coordinates. The number of these additional nodes in each case is different and defined by iterations, until the decision error does not accept comprehensible value. As a result of the additional formation of received neural network learning solution using additional computation nodes, it turned out to be possible to increase the solution local accuracy in the point B vicinity while maintaining the accuracy high in all other points.

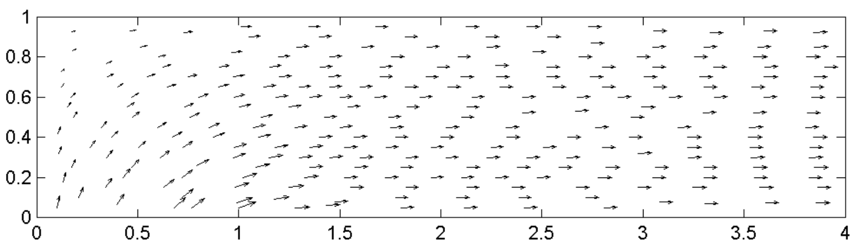


Fig. 4. Formation of additional computation nodes for Laplace equation solution

Therefore, not only has the computing experiment proven reception opportunity of the general neural network solution in the calculation area, but also defined coordinate calculation logic of computational nodes for increasing the accuracy of neural network initial equation solution. Let us study a reception opportunity of the Poisson equation solution using an irregular computational grid, i.e. equation total residual with solutions (1) will be calculated in nodes located in the casual image or certain algorithm, which use has not been connected with the necessity of computational grid coordination and computational area borders.

Poisson equation

Let us study a neural network solution precision on irregular calculation scales for Poisson equation

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = \Delta \neq 0 \,. \tag{15}$$

This equation is particularly used for calculating the pressure distribution as well as for time iterations organization at the Navier-Stokes equations solution by pseudo-non-stationary algorithms (Fletcher, 1991). For the solution we shall use an irregular calculation grid, because, in contrast to fluid dynamics classical numerical methods, it does not result in the neutral network learning functions algorithm complication. Meanwhile, advantages using calculation nodes located in calculation area for the complex geometry study current are obvious. The decision is defined by the equations (15) with the right side as follows

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = \Delta = 2\left(\frac{\partial u}{\partial x}\frac{\partial v}{\partial y} - \frac{\partial v}{\partial x}\frac{\partial u}{\partial y}\right), \tag{16}$$

where speed nodes $u = \frac{\partial \varphi}{\partial x}$ also $v = \frac{\partial \varphi}{\partial y}$ are received as a result of the Laplace equation solution (12). Calculation grid points are formed as centers of Cohonen network clusters constructed on units coordinates of the uniform rectangular scale and on the right part of the equation (16) corresponding to these units values Δ. Fig. 6 (a) presents formation re sults of the calculation grid and the speed distribution on the pseudo-non-stationary algorithm first iterative step of the Navier-Stokes equation solution. Here it was possible to receive an exact neural network solution for the whole calculation area without using additional set of calculation nodes.

Let us now study am incompressible fluid internal flow within a channel with a stream turning (fig. 3). Navier-Stokes equation system describing two-dimensional isothermal flows of the viscous incompressible fluid (Fletcher, 1991):

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0\,; \tag{17}$$

$$\frac{\partial p}{\partial x} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - \frac{1}{\mathrm{Re}}\left\{\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right\} = 0\,; \tag{18}$$

$$\frac{\partial p}{\partial y} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \frac{1}{\mathrm{Re}}\left\{\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right\} = 0\,. \tag{19}$$

Here *u, v* – nodes speed, Re - Reynolds number. Hydrodynamics equations system is written in the non-dimensional view; i.e. it includes non-dimensional values $u^* = \frac{u}{u_\infty}$, $v^* = \frac{v}{u_\infty}$, $r^* = \frac{r}{D}$, $p^* = \frac{p}{\rho \cdot u_\infty^2}$, $\mathrm{Re} = \frac{\rho \cdot u_\infty \cdot D}{\mu}$. Quality of u_∞ and any speed and linear size values can be chosen in the current field, for example an input fluid speed value in the channel and the channel width h.

Boundary conditions are stated as follows: on solid walls *u=v=0*, on the input border *u=0, v=1*, on the output border $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial x} = 0$. Let us consider that there is rectangular region [a,b]×[c,d] within the plane *XY*, and there is a rectangular analytical grid, specified by Cartesian product of two one-dimensional grids $\{x_k\}$, *k=l,…,n* and $\{y_l\}$, *l*=l,…,*m*.

We will understand neural net functions $u, v, p = f_{NET}(\mathbf{w}, x, y)$ as the (17)-(19) system solution giving minimum of the total squared residual in the knot set of computational grid. The trial solution (fig. 5) of the system (17)-(19) *u, v, p* can be presented in the form equation (1):

$$u(\mathbf{w}, x, y) = \sum_{i=1}^{q} v_i f_\sigma(b_i + w_{i1}x + w_{i2}y) + b_u\,; \tag{20}$$

$$v(\mathbf{w}, x, y) = \sum_{i=q+1}^{2q} v_i f_\sigma(b_i + w_{i1}x + w_{i2}y) + b_v\,; \tag{21}$$

$$p(\mathbf{w},x,y)=\sum_{i=2q+1}^{3q} v_i f_\sigma(b_i+w_{i1}x+w_{i2}y)+b_p\,. \tag{22}$$

Here again, **w** is the vector of all the weights and thresholds of the net. In this case the amount of q neurons in the trial solutions remains the same for each decision variable set. This is the parameter on which depend approximated capabilities of neural net trial solution. Result of computational algorithm functioning should be achievement of necessary accuracy level of solution at q minimum value.

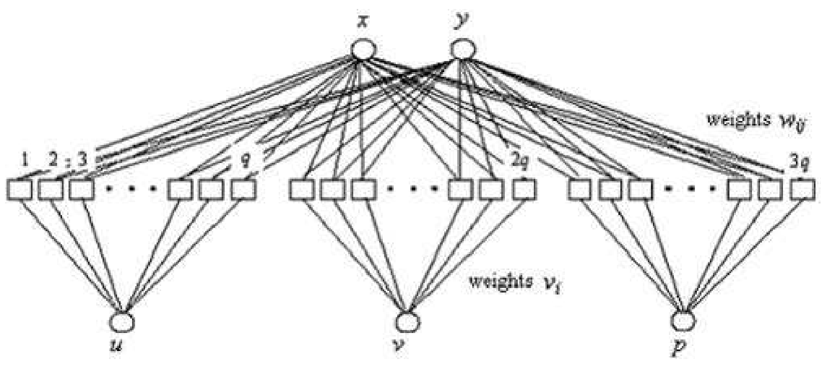


Fig. 5. Neural net trial solution

Let us name the residuals of equations (17)-(19) R_1, R_2 and R_3 correspondingly, then for the vase of NMWR realization for parameters setup of the trial solution it is necessary to minimize three objective functions $R_1^2;R_2^2;R_3^2\to\min$. In the simplest case, the only solution of the multi-criterion problem of minimization can be generated substituting of three criterions by one, presented in compression form; for example, $R^2=R_1^2+R_2^2+R_3^2\to\min$. Presenting the trial solution in the form of continuous functions (20)-(22) allows to define analytically the first and the second differential coefficient in the equations (17)-(19), knowing which one can generate analytic expressions of the function of residuals $R^s(\mathbf{w},x,y)$ and further for antigradient component of the total residual in the s-reference point at ANN inner parameters $\frac{\partial R}{\partial v_j}$, $\frac{\partial R}{\partial w_{ij}}$ and $\frac{\partial R}{\partial b_j}$, being later used in the minimization algorithm in accordance with anti-gradient direction.

For the momentum equations solution by MAC method, it is necessary to specify an initial speeds distribution in the calculation area satisfying to the continuity equation. For this purpose, the speed potential $\varphi(x,y)$ is introduced, $u=\frac{\partial\varphi}{\partial x}$ and $v=\frac{\partial\varphi}{\partial y}$. As a Laplace equation solution result, we obtain speed distribution which can be called non-vortex required value.Final speeds and pressure distribution are results of the momentum equations solution according to the following algorithm.

The speed distribution on the following time layer is calculated according to the formula

$$\mathbf{u}^{n+1}=\mathbf{F}^n-\Delta t\cdot\nabla p^{n+1}\,, \tag{23}$$

where pressure distribution to each iterative step is defined upon the Poisson equation solution

$$\nabla^2 p^{n+1} = \frac{1}{\Delta t}\nabla \mathbf{F}^n = f_{NET}(x,y) \quad . \tag{24}$$

Vector $\mathbf{F} \equiv (F,G)^T$ introduced to this algorithm can be calculated by the momentum equations (18-19); or Poisson equation (16) can be used for pressure calculation. Thus, the solution for pressure received from the equation (16) results in the continuity equation realization at the moment of time $n+1$. Once p^{n+1} is calculated, substitution of these values in the formula $\mathbf{u}^{n+1} = \mathbf{F}^n - \Delta t \cdot \nabla p^{n+1}$ allows to determine v^{n+1}. The iterative process goes on until speed distribution stops varying.

Let us briefly generalize the above mentioned results of calculation experiment in methodic form to set dynamic calculation scale at the Navier-Stokes equations (17)-(19) solution by the establishment method. First, Laplace equation (12) solution is calculated with the received scale or with rectangular scale, or by means of random numbers generator, or by using the Sobol - Statnikov generator of LP_τ (Statnikov & Matusov, 2002); then, distribution solving equation residual in the grid nodes; third, the additional multitude of calculation nodes is generated with using the Cohonen network; then, if the precision is not achieved, points 2 and 3 are realized and the additional calculation components quantity grows until exact neural network solution for the whole calculation area is found; further, pseudo-non-stationary algorithm iterations are organized by using the equations (23)-(24) where equation (16) solution is found on each time step on the multitude of calculation components, the coordinates of which vary depending on the Poisson equation right part distribution for each iterative step; finally, at steps 3 and 5 realization of structural optimization algorithms and learning neural network solutions standardization formation

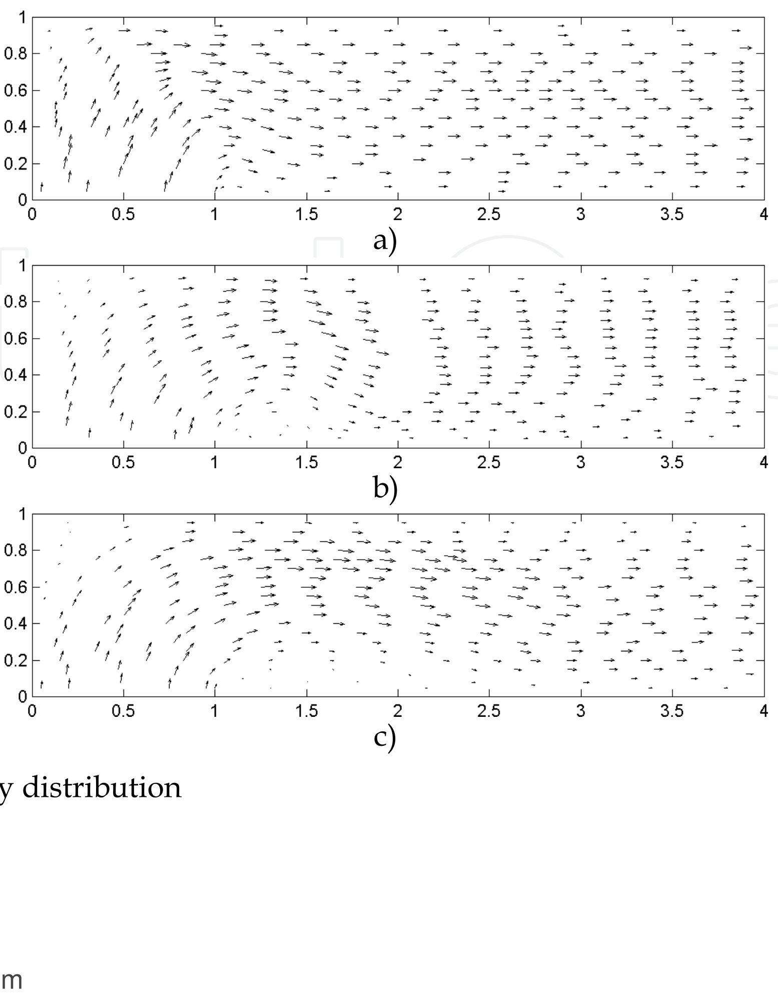


Fig. 6. Net velocity distribution

stated in (Kretinin et al., 2010). On fig. 6 (a-c), changing dynamics of calculation components during realizing various moments of time of equations (23)-(24) algorithm is shown, the speed vector distribution reorganization during the transition from the equation (12) solution to the equations (17)-(19) solution is also illustrated.
Finally, analytical solution (as neural network function) for Navier-Stokes equations (17)-(19) systems within the channel with stream turning at $\mathrm{Re} = 100$ which is expressed by the formula (23) on last iterative layer and neural network dependence of the pressure distribution on this layer

$$p(\mathbf{w},x,y) = \sum_{i=2q+1}^{3q} v_i f_\sigma(b_i + w_{i1}x + w_{i2}y) + b_p$$

with weights calculated array and the boundary network $\mathbf{w} = \{w,v,b\}$.

3.2 Modeling flows in rotating ring zone - the equations that are applicable to rotating reference frame

For this flow, NMWR solves conservation equations for mass and momentum describing incompressible flows of viscous Newtonian fluid.
Continuity equation

$$\frac{\partial \overline{u}_j}{\partial x_j} = 0 \ . \tag{25}$$

Momentum equations

$$\frac{\partial}{\partial x_j}\left(\overline{u_i u_j}\right) + \frac{\partial}{\partial x_j}\left(\overline{u_i' u_j'}\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)\right] + f_i \ . \tag{26}$$

For flows in rotating domain (fig. 7), the equations for conservation of mass and momentum are written for the relative velocity formulation, where f_i in right hand side is given by

$$\vec{f}_i = -\rho\left(2\vec{\omega}\times\vec{u} + \vec{\omega}\times\left(\vec{\omega}\times\vec{r}\right)\right). \tag{27}$$

The absolute velocity formulation is used in the non-rotating domain, and f_i=0. Thus, the standard $k-\varepsilon$ turbulence model is used.
Let us consider the neural network functions $\overline{u}_i, p, k, \varepsilon = f_{NN}(\mathbf{w},x,y)$ as equations solution, where $\mathbf{w}$– all weights and network thresholds vector that assure summary quadratic residuals minimums for each equation in optional totality of computation nodes coordinating each neural network solution learning iteration, are generated using the random number generator.
Neuronet learning solution parameter search process efficiency depends on problem dimension, i.e. weights and perceptron thresholds varying adjusted quantity. The more significant is neurons quantity in trial solution, the higher is ANN approximate capacity; however, achieving high approximation accuracy is more complicated.

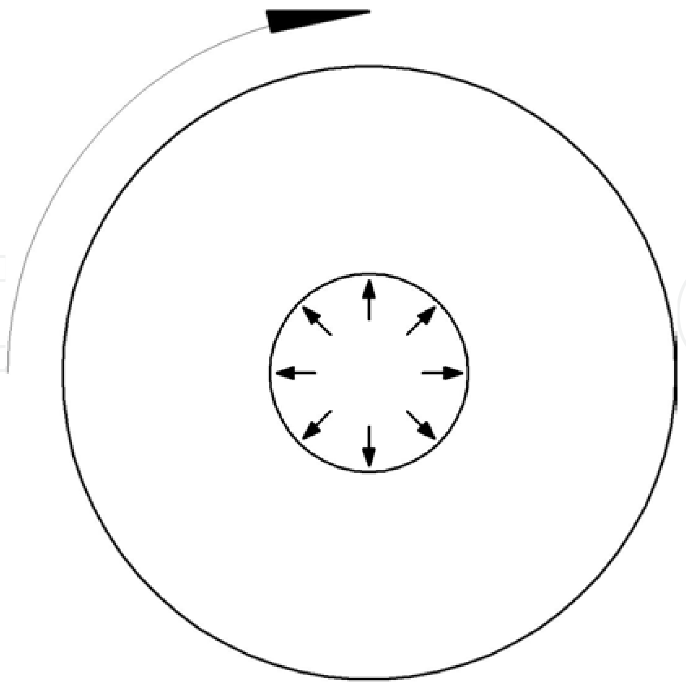


Fig. 7. Rotating Domain

At the same time, neuron quantity depends not only on simulated function complexity, but also on calculation nodes quantity in which the residual equation is calculated. It is known that generally points' quantity increase in statistical set used for neural network construction is followed by increase in necessary neurons network (Galushkin, 2002; Galushkin, 2007) quantity. Consequently, the dense calculation grids application results in nonlinear programming problems; while applying rare calculation grids, it is necessary to check the solution realization between calculation nodes, i.e. there is a problem of learning solution procedure standardization. In the neuronet solution reception context on known equation, it is convenient using traditional additive parameter of training neural model quality - a control error which is calculated on the set of additional calculation nodes between calculation grid nodes. The number of these additional calculation grid nodes can be much more important and they should cover all calculation area because the nodes number increase in which the control error on known network parameters does not result in essential computing expenses growth. Hence, at the learning solution neuronet parameters reception there is a problem of solving the multi-criterion problem of the nonlinear optimization and to minimize simultaneously both the summary residual in control points, or the control error can appear as a restriction parameter, in the limited set of calculation nodes and in this case the neural network solution parameters reception is reduced to the conditional nonlinear optimization problem.
On figure 8, velocity distribution vector within the current random computation nodes of revolving ring on one of neural network training iteration of equation (25)-(26) decisions is presented. Contours of stream function within the computational zone obtained by using NMWR are presented on fig. 9.

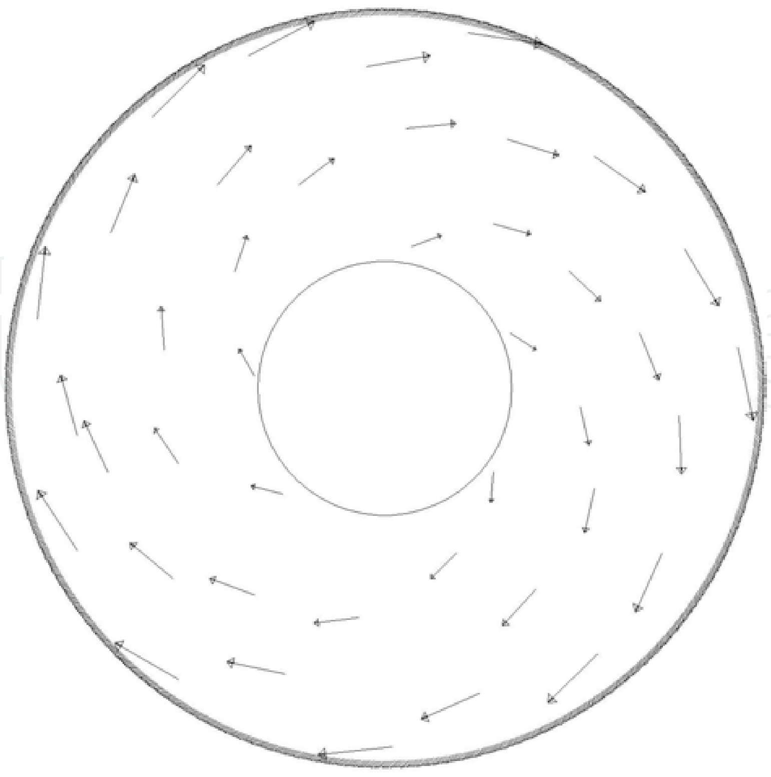


Fig. 8. Velocity distribution on one of neural network training iteration

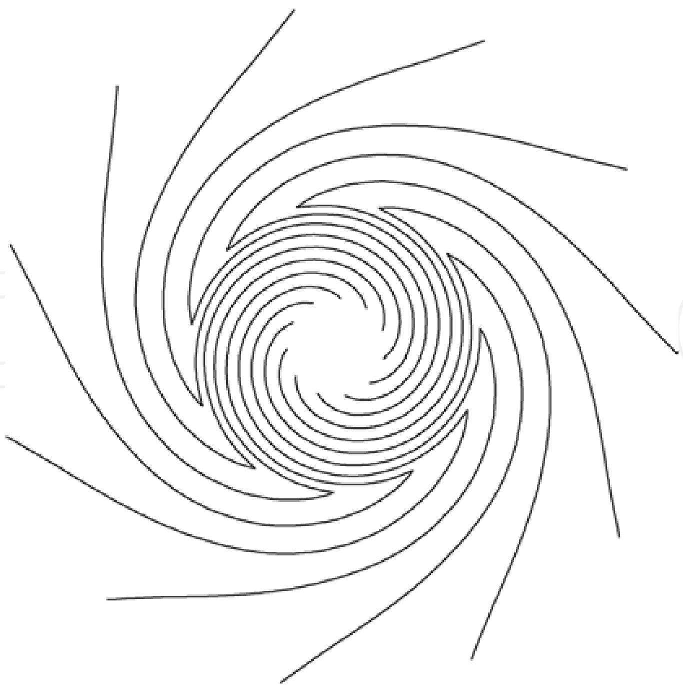


Fig. 9. The contours of stream function

4. Modeling leakage in a fuel transfer pipeline

Method of leakage zonal location (Zverev & Lurie, 2009), which generalizes the known method of leakage detection by hydraulic gradient line salient point, can be used to define of pipeline leakage position. When using the “base” zone location method variant, it is supposed that in case of stationary process, hydraulic gradients i_1 and i_2 before and after the leakage are constant and can be designed by the known values of liquid charge at the ends of controllable site. I.e., the distribution indignation of hydrodynamic parameters is not taken into account in comparison with the established flow which arises because of the environment outflow influence through the leakage aperture. It results in speed profile deformation within the cross sections of the pipeline near the leakage and the nonlinear dependence of hydraulic gradient in the function of distance from position of leakage downwards the stream. In order for pipe cross section speed distribution of the pipe to become appropriate for the established flow, the distance up to 40-60 calibres from leakage coordinate x_y can be required. Defining nonlinear dependence of hydraulic gradient function near the leakage is possible either based on special experimental researches, or based on the numerical decision of hydrodynamics equations in three-dimensional statement. The given function can be added to the algorithm of zone location methode to reduce the leakage position coordinate definition error.

One pipeline section of 300 m length and diameter $D = 1$ m is used for the nonlinear dependence of the total pressure drop determination along the pipeline length in the leakage neighborhood through the wall hole. The leakage position coordinate is fixed $x_y = 155.5$ m . On the segment $x \in [140, 200]$ m computational faces are formed with an interval $h = 1\,m$. In segments $x \in [100, 140]$ m and $x \in [200, 250]$ m computational faces are disposed with an interval $h = 10$ m . Such a leakage location (approximately in the middle of the pipeline sector into a question) and computational faces disposition have been chosen for the influence elimination of the boundary conditions setting in the computational zone inlet and outlet on the flow distribution near leakage position. The leakage hole D_l/D relative diameter during the computational planed experiment is constant. The mathematical model includes conservation equations for mass and momentum enclosed by the standard $k-\varepsilon$ turbulence model. The local hydraulic friction coefficient in any section of the pipeline is defined by the ratio

$$\lambda = -\left[8\mu \left.\frac{\partial u}{\partial r}\right|_{r=R} \right] \Big/ \left(\rho \bar{u}^2\right), \tag{27}$$

where $|\bar{u}| = \frac{1}{R^2}\int_0^R 2urdr$ - design section average speed, μ - dynamic factor of liquid viscosity.

Turbulent flow speed distribution within the pipe cross section is described by the universal logarithmic dependence

$$\frac{u}{u_*} = A \log \frac{u_* y}{\nu} + B, \tag{28}$$

where $u_* = \bar{u}\sqrt{\frac{\lambda}{8}}$ - the dynamic speed, y - the distance from the wall, ν - the kinematic viscosity.

At the statement of boundary conditions on the input, mass charge $\dot{m}_1$ or the speed $\overline{u}_1$ appropriate to this charge is set. In the grid units belonging to the leakage aperture, liquid outflow speed u_y is appointed. The given parameter at numerical researches varies for the modeling leakages of various intensity. For any point of the calculating area, any operational value of pressure $\partial/\partial x$ of all the hydrodynamical parameters are equaled to zero, for the concerning of which differences of pressure will be calculated, is set. On the output from the calculating area the conditions of the established current are set, i.e. the derivatives modeling of the boundary layer the standard wall functions for parameters of turbulence k and ε (Fletcher, 1991) are used.
The equations discretization is effectuated on the base of finite volumes method in the combination with hexagonal grid. On fig. 10 the formation of the grid in the calculating sections of the pipeline is represented.

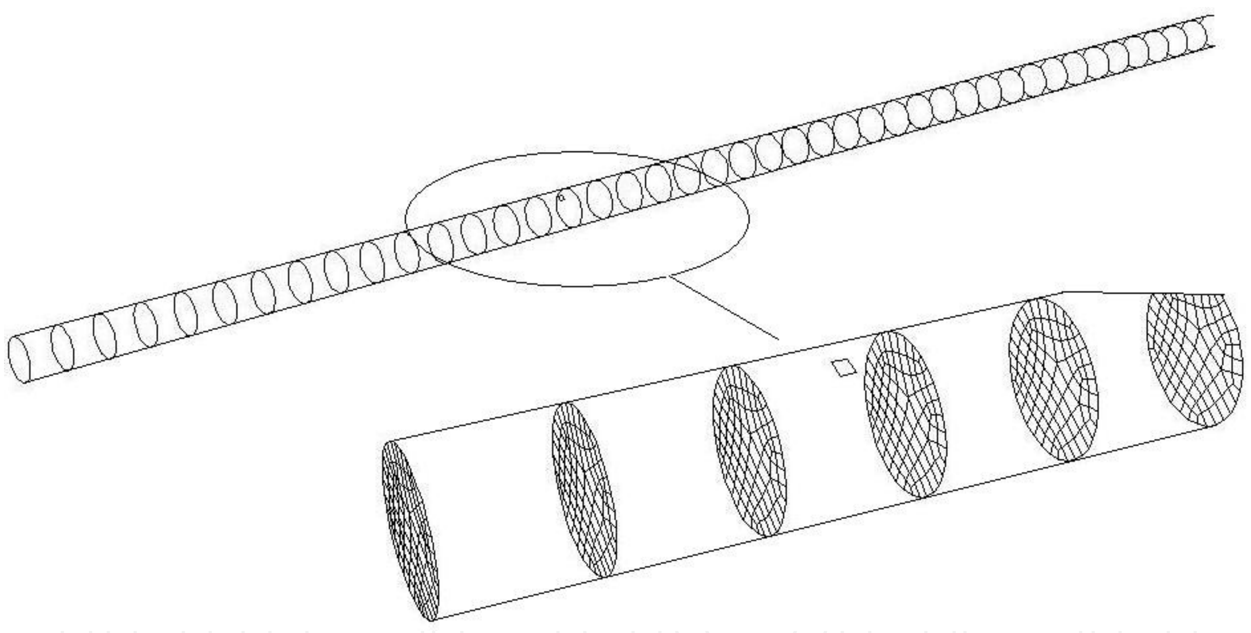


Fig. 10. Calculation grid in the pipeline sections

4.1 The plan of computing experiment

If x_1, x_2 are the coordinates of the beginning and the end of the controllable pipeline sector, then i_1, i_2 are hydraulic gradients before and after the leakage respectively, Δp_0 is total pressure drop of the sector, so in case of $i_1 = const$ from the beginning of the sector to the leakage and $i_2 = const$ from the leakage up to the end of the sector, in the stationary hydrodynamic mode, leakage coordinate shall be defined by the formula

$$x_y = \frac{\Delta p_0 + \rho g\left(i_1 x_1 - i_2 x_2\right)}{\rho g\left(i_1 - i_2\right)}. \tag{29}$$

Let us present the total pressure drop nonlinear dependence for its calculation in leakage neighbourhood in nonlinear function form $i = f_{NN}^{i}\left(i_1, i_2, x - x_{NN}\right)$ on the pipeline sector $x \in \left[x_{NN}, x_{NN} + l_{NN}\right]$, where for our computational model the left area limit coordinate of the nonlinear leakage function determination $x_{NN} = 150\,\text{m}$ and the determination length fragment area $l_{NN} = 60\,\text{m}$ (leakage is situated in the point with coordinate $x_y = 155{,}5\,\text{m}$). In this case, total pressure drop in the controlled sector $\left[x_1, x_2\right]$ is calculated by the following formula

$$\frac{\Delta p_0}{\rho g} = i_1\left(x_{NN} - x_1\right) + i_2\left(x_2 - \left(x_{NN} + l_{NN}\right)\right) + \int_0^{l_{NN}} f_{NN}^{i}\left(i_1, i_2, x - x_{NN}\right)dx\,. \tag{30}$$

In the known function $f_{NN}^{i}\left(i_1, i_2, x - x_{NN}\right)$, the present equation is nonlinear with one unknown value x_{NN}. After solving this equation, we can define the pipeline leakage coordinate x_l.

Starting from above-mentioned, computational experiment is effectuated with the purpose of function determination $f_{NN}^{i}\left(i_1, i_2, x - x_{NN}\right)$, i.e. the total pressure drop dependence in the leakage neighbourhood *i* (criterion) of three variables $i_1^{NN} = \rho g i_1$, $i_2^{NN} = i_2 / i_1$ and $\Delta x_{NN} = x - x_{NN}$ (factors). Earlier, the interval forfactor variation $\Delta x_{NN} \in \left[0{,}60\right]$ was determined. Let us determine the intervals $i_1^{NN} \in \left[10{,}20\right]$, $i_2^{NN} \in \left[0{,}5{,}0{,}95\right]$. The plan of computing experiment is made with the points received with the help of Sobol-Statnikov generator of quasiuniform number sequences (Statnikov & Matusov, 2002). The working hypercube of space $\mathbf{R}^3$ is filled with the points $\left(i_1^{NN}, i_2^{NN}, \Delta x_{NN}\right)$ according to LP_τ algorithm (Statnikov & Matusov, 2002). Choice of this algorithm for the experiment plan formation is made by high efficiency of the research method of parameter space based on the sounding of computational area by points of the uniform distributed sequence.For each variable, parameter combinations i_1^{NN}, i_2^{NN} from the experiment plan limitation boundary conditions in the computational model are selected by determining values corresponding tothe inlet mass flow $\dot{m}_{in}$ and leakage intensity $\dot{m}_l$, where the hydraulic friction coefficient is calculated by the implicit formula of Altschul for "smooth" pipes

$$\frac{1}{\sqrt{\lambda}} = -2{,}04\log\left(\frac{2{,}82}{\mathrm{Re}\sqrt{\lambda}}\right).$$

As a result of computational hydrodynamics equations decision, we obtain the value distribution of total pressure drop on the length of the controlled pipeline sector, from which we are exterminating the criteria value *i* for the corresponding value Δx_{NN} from the experiment plan. Let us now register the obtained vector $\left(i_1^{NN}, i_2^{NN}, \Delta x_{NN}, i\right)$ in database for the posterior generation of neural network dependence $i = f_{NN}^{i}\left(i_1^{NN}, i_2^{NN}, \Delta x_{NN}\right)$.

For the illustration of numerical calculations, the distribution of speed in the leakage neighbourhood is presented on Fig. 11, and on fig. 12 the distribution of the hydraulic gradient on the pipeline sector $x \in \left[100{,}250\right]\text{m}$ for factor values $i_1^{NN} \approx 14\,\frac{\text{Pa}}{\text{m}}$, $i_2^{NN} = i_2 / i_1 = 0{,}5;\, 0{,}57;\, 0{,}64;\, 0{,}71;\, 0{,}8;\, 0{,}9$ is presented.

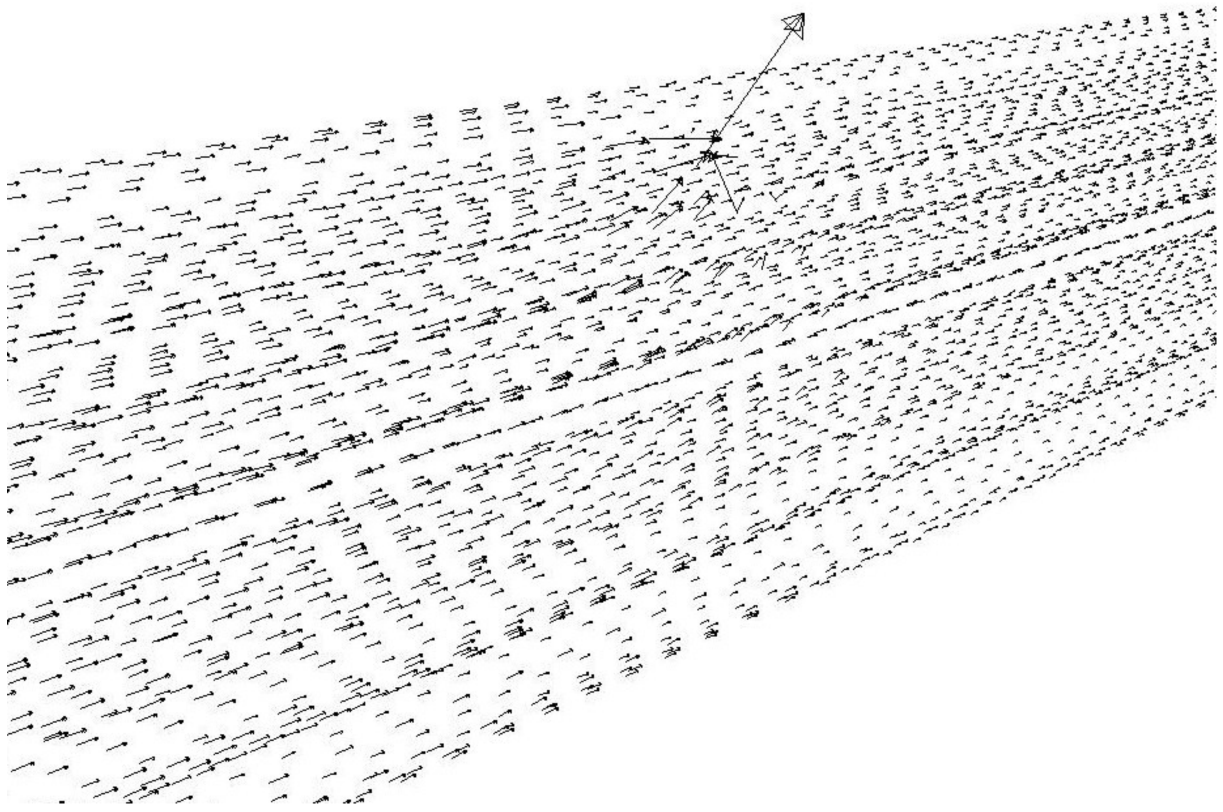


Fig. 11. Speed distribution in the leakage neighbourhood

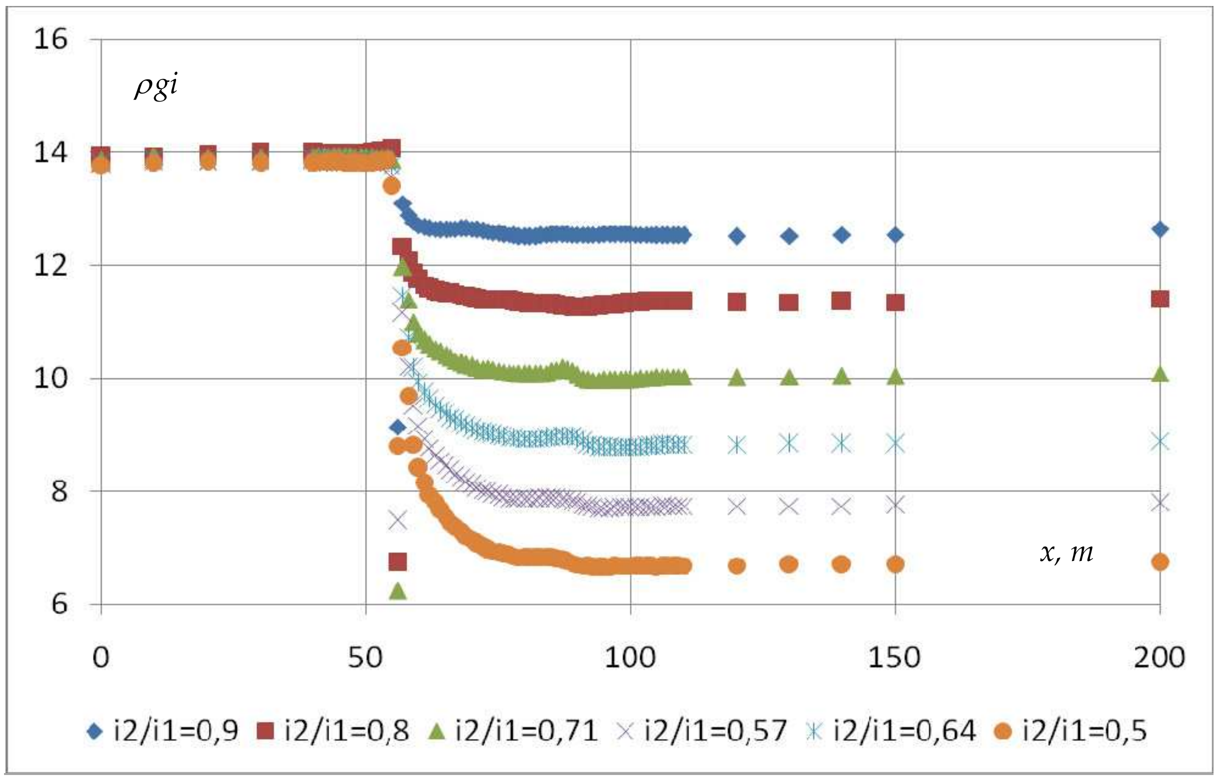


Fig. 12. Total pressure drop distribution for 1 meter of pipeline length

4.2 The neural network regressive model

For the regressive dependence construction, $i = f_{NN}^{i}\left(i_1^{NN}, i_2^{NN}, \Delta x_{NN}\right)$ artificial neural networks device is used. For the formation of display $i = f_{NN}^{i}\left(i_1^{NN}, i_2^{NN}, \Delta x_{NN}\right)$, the standard structure of the multilayer perceptron (MLP) with 3 inputs, one output and two latent layers with 7 and 5 neurons accordingly was used. At training, MLP algorithm of Levenberg-Markardt was used. The total RMS error on 512 points of statistical sample was *E=0.001*.

The comparative analysis of dependences calculation results obtained for inlet parameters values $\rho g i_1 = 16\,\frac{\text{Pa}}{\text{m}}$, ${i_2}/{i_1} = 0.9$ is presented on Fig. 13 and $\rho g i_1 = 10\,\frac{\text{Pa}}{\text{m}}$, ${i_2}/{i_1} = 0.8$ for $\Delta x_{NN} \in [0,60]\,\text{m}$ from the differential equation computational solution– continuous lines and determined with neural network dependence – markers. It is also necessary to note the high approximation precision.

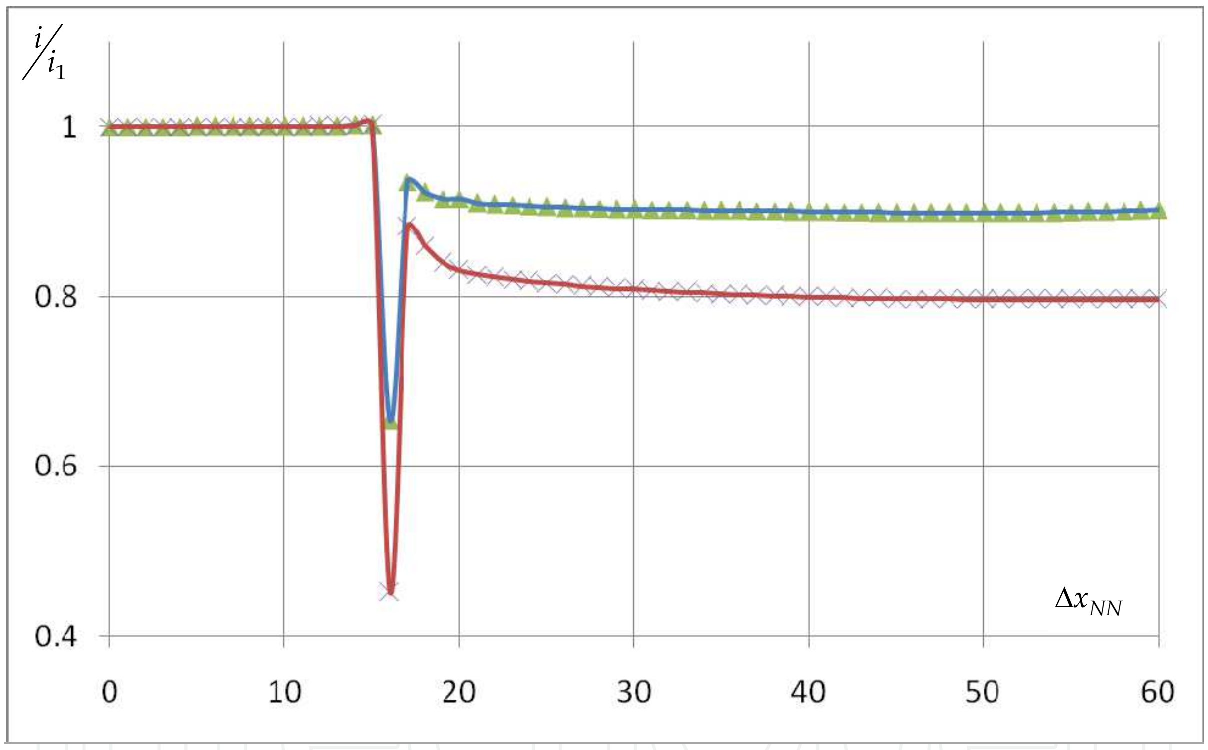


Fig. 13. Comparative analysis of neural network data and numerical solution results

5. Conclusion

Results presented in this chapter concern one of many applications of neural network learning functions to the mathematical physics problems solution, which is solution of hydrodynamics equations. Using learning functions of neural network allows to exclude the errors in solving differential equations caused by derivatives discretization and the borders low precision representation. Numerical algorithm of Navier-Stokes equations system solution differing from known fluid dynamics methods by the possibility of computational grid arbitrary formationis developed. The formation technique of the additional nodes file depending on learning solution local accuracy is offered.

The formation of the finite-elemental models for all extended pipelines and the numerical decision of equations of liquid movement for modeling the hydrodynamics processes now is limited to computer resources as by the quantity of the final elements of model calculated grid, so as by the time of finding of the decision. Even with such model, its use for system functioning operative analysis is quite doubtful. However, the numerical models of hydrodynamics processes can be used for the formation of information databases based on the neural network computing architecture which, after "training", have high speed of calculation. Neural networks are the universal approximation tool of multivariate nonlinear dependences, capable "to be arranged" under appearing of the new information of the researched process, i.e. they can serve as an intellectual tool for monitoring, which is constantly filled up and clarified. Thus, the introduction of neural algorithms in the approved and well-recommending methods of pipeline leakage detection is expedient for increased accuracy and the efficiency of accepted decisions.

6. References

Bishop, C.M. (1995) *Neural networks for pattern recognition*, Oxford University Press

Cloete, L. & Zurada, J.M. (2000) *Knowledge-Based Neurocomputing,* MIT Press

Egorov, I. N.; Kretinin, G.V.; Matusov I.B. & Statnikov, R.B. Problems of design and multicriteria control of controlled engineering systems. *J. Reports of RF Academy of Sciences,* Vol. 359, No. 3, 1998, pp. 330-333. ISSN 0869-5652

Fletcher, K. (1991). *Computational techniques for fluid dynamics,* Mir, Moscow, ISBN 5-03-001881-6

Galushkin, A.I. (2002). *Neuromathematics,* IPRJR, Moscow

Galushkin, A.I. (2007). *Neural networks theory,* Springer

Kretinin, A.V. The weighted residuals method based on neuronet approximations for simulation of hydrodynamics problems, *Siberian J. Num. Math*, 2006, Vol. 9, No.1, pp. 23-35.

Kretinin, A.V.; Bulygin, Yu. A. & Kirpichev, M.I. Method of Weighted Residuals on the Base of Neuronet's Approximations for Computer Simulation of Hydrodynamics Problems, *Proceedings of IEEE 6th International Conference on Computational Cybernetics ICCC 2008*. pp. 237-240, Stara Lesná, Slovakia, 2008.

Kretinin, A.V.; Bulygin, Yu. A.; Kirpichev, M.I. & Darneva, T.I. Neuronet's method of weighted residuals for computer simulation of hydrodynamics problems, Proceedings of 2010 International Joint Conference on Neural Networks. ISBN 978-1-4244-6916-1, Barcelona, Spain, 2010.

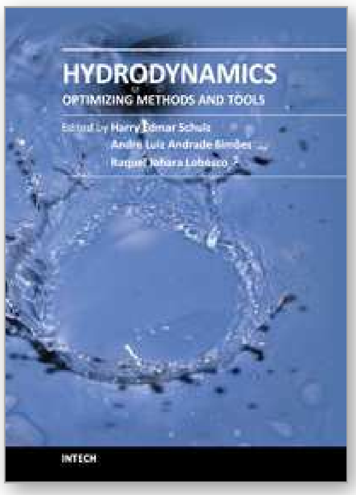
Prokhorov, D.; Feldkamp, L. & Feldkamp, T. A new approach to cluster weighted modeling. *Proceedings of International Joint Conference on Neural Networks (IJCNN),* Washington DC, July 2001

Statnikov, R.B. & Matusov, J.B. (1995). *Multicriteria Optimization and Engineering,* Chapman & Hall, N.Y.

Statnikov, R.B. & Matusov, J.B. (2002). *Multicriteria Analysis in Engineering,* Kluwer Academic Publishers, Dordrecht/ Boston / London

Stogney, V.G. & Kretinin*, A.V.* Simulation of Flows in a Channel with Permeable Wall Using Artificial Neural Networks. *Izv.Vuz. Av. Tekhnika*, Vol. 48, No. 1, 2005, pp. 34–38 [*Russian Aeronautics* (Engl.Transl.)]

Zverev, F.C. & Lurie, M.V. Generalized zonal location method of pipeline leakage detection, *Oil industry*, No 8, 2009, pp. 85-87



Hydrodynamics - Optimizing Methods and Tools
Edited by Prof. Harry Schulz

ISBN 978-953-307-712-3
Hard cover, 420 pages
Publisher InTech
Published online 26, October, 2011
Published in print edition October, 2011

The constant evolution of the calculation capacity of the modern computers implies in a permanent effort to adjust the existing numerical codes, or to create new codes following new points of view, aiming to adequately simulate fluid flows and the related transport of physical properties. Additionally, the continuous improving of laboratory devices and equipment, which allow to record and measure fluid flows with a higher degree of details, induces to elaborate specific experiments, in order to shed light in unsolved aspects of the phenomena related to these flows. This volume presents conclusions about different aspects of calculated and observed flows, discussing the tools used in the analyses. It contains eighteen chapters, organized in four sections: 1) Smoothed Spheres, 2) Models and Codes in Fluid Dynamics, 3) Complex Hydraulic Engineering Applications, 4) Hydrodynamics and Heat/Mass Transfer. The chapters present results directed to the optimization of the methods and tools of Hydrodynamics.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Sergey Valyuhov, Alexander Kretinin and Alexander Burakov (2011). Neural Network Modeling of Hydrodynamics Processes, Hydrodynamics - Optimizing Methods and Tools, Prof. Harry Schulz (Ed.), ISBN: 978-953-307-712-3, InTech, Available from: http://www.intechopen.com/books/hydrodynamics-optimizing-methods-and-tools/neural-network-modeling-of-hydrodynamics-processes