# The two-stage evolutionary-neuro computing approach for stochastic optimization problems

Piotr Orantek[1] and Tadeusz Burczyński [1,2]

[1] Department for Strength of Materials and Computational Mechanics,
Silesian University of Technology, Konarskiego 18a, 44-100 Gliwice, e-mail: piotr.orantek@polsl.pl
[2] Institute of Computer Modelling, Artificial Intelligence Department,
Cracow University of Technology, Warszawska 24, 31-155 Kraków, e-mail: tadeusz.burczynski@polsl.pl

**Abstract.** This paper is devoted to the application of the two-stage evolutionary-neuro approach for stochastic optimization problems. The algorithm is based on the stochastic representation of the data. Chromosomes are represented by multidimensional random vectors consisting of random genes in the form of independent random variables with the Gaussian density probability function. The stochastic optimization problem is repalced by deterministic one by evolutionary computing for vector genes consisting of mean values and standard deviations. In the first stage the EA is used. As the second stage in the presented approach the special local gradient method with neuro-computing is proposed.

## 1 Introduction

There are physical problems in which systems and processes have some uncertain parameters, e.g. materials properties, boundary conditions or geometry. The granular type of information about these parameters causes the necessity of using various models of uncertainty in the form of interval, fuzzy and rough sets and the theory of probability.

The concept of the interval and the fuzzy evolutionary algorithm and their applications in the optimization and identification problems have been considered in previous papers.

In the present paper another form of granularity is analyzed - the probability approach to optimum design is considered. The parameters of systems and processes are modelled by random variables characterized by a probability density function. The classical approach to solution of such problems is based on stochastic programming [3].

The new concept two-stage evolutionary-neuro computing approach is proposed. In the first stage the EA is used. As the second stage in the presented approach the special local gradient method with neuro-computing is proposed. The special multilevel artificial neural network for aproximation the stochastic problem is proposed. The gradient of the fitness function is cumputed due to the multilevel artificial neural network.

## 2 The formulation of the stochastic optimization problem

A general non-linear stochastic programming problem can be stated as follows:
*Find a random vector*

$$\mathbf{X}(\gamma) = [X_1(\gamma), X_2(\gamma), ..., X_i(\gamma), ..., X_n(\gamma)] \tag{1}$$

*which minimizes the objective function* $F(\gamma) = F(\mathbf{X}(\gamma))$ *subject to the constraints* $P\left[g_j(\mathbf{X}) \geq 0\right] \geq p_j, j = 1,2,...,m$ .

If the problem is solved by the evolutionary approach, the vector $\mathbf{X}(\gamma)$ is considered as the chromosome, where $X_i(\gamma)$ , $i$=1,2,…,$n$, are random genes.

A gene is represented by a random variable, which is a real function $X_i = X_i(\gamma)$ , $\gamma \in \Gamma$ , defined on a sample space $\Gamma$ and measurable with respect to $P$: i.e., for every real number $x_i$ , the set $\{\gamma : X_i(\gamma) < x_i\}$ is an event in $\mathcal{F}$.

The chromosome $\mathbf{X}(\gamma)$ is a function (measurable with respect to $P$) which takes every element $\gamma \in \Gamma$ into a point $\mathbf{x} \in R^n$ .

The mean value of the chromosome $\mathbf{X}(\gamma)$ is given as follows

$$\mathbf{m} = \mathbf{E}[\mathbf{X}(\gamma)] = \left[m_1, m_2, ..., m_i, ..., m_n\right] \tag{2}$$

where:

$$m_i = \mathbf{E}\left[X_i(\gamma)\right] = \int_{\Gamma} X_i(\gamma) dP(\gamma) = \int_{-\infty}^{+\infty} x_i p_i\left(x_i\right) dx_i \tag{3}$$

is the mean value of the gene $X_i(\gamma)$ and $p_i(x_i)$ is the probability density function (PDF) of this gene [5].
The matrix of covariance is given as follows:

$$\mathbf{K} = [k_{ij}] = \mathbf{E}\left[\left(\mathbf{X}(\gamma) - \mathbf{m}\right)^T \left(\mathbf{X}(\gamma) - \mathbf{m}\right)\right] \tag{4}$$

where the covariance between $X_i(\gamma)$ and $X_j(\gamma)$ is defined by:

$$k_{ij} = \mathbf{E}\left[\left(X_i(\gamma) - m_i\right)\left(X_j(\gamma) - m_j\right)\right] = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\left(x_i - m_i\right)\left(x_j - m_j\right) p\left(x_i, x_j\right) dx_i x_j \tag{5}$$

where $p\left(x_i, x_j\right)$ is the joint PDF of $X_i(\gamma)$ and $X_j(\gamma)$ [5]. If $i = j$, the covariance is represented by a variance [5]. In the present paper the random chromosome $\mathbf{X}(\gamma) = [X_1(\gamma), X_2(\gamma), ..., X_i(\gamma), ..., X_n(\gamma)]$ has an $n$-dimensional Gaussian distribution of the probability density function [5]. It is assumed that random genes are independent random variables. The joint probability density function is expressed by the probability density functions of single random genes as follows:

$$p(x_1, x_2, ..., x_i, ..., x_n) = p_1(x_1) p_2(x_2) .... p_i(x_i) .... p_n(x_n) \tag{6}$$

where:

$$p_i\left(x_i\right) = N\left(m_i, \sigma_i\right) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left[-\frac{\left(x_i - m_i\right)^2}{2\sigma_i^2}\right] \tag{7}$$

is the probability density function of the random gene $X_i(\gamma)$ .
It can be seen that if the random genes $X_i(\gamma)$, $i$=1,2,…,$n$, are random independent Gaussian variables, two parameters: the mean value $m_i$ and the standard deviation $\sigma_i$ describe the probability density function.

## 3 Evolutionary-neural strategy

The main idea of the creating the stochastic strategy consists of two stages based on coupling the advantages of evolutionary and gradient optimization methods aided by neuro-computing.

The evolutionary algorithms can find the global optimum, but it is very time consuming. The gradient methods can find the optimum precisely, but they need information about sensitivity to the objective function.

The proposed strategy in the first stage uses some properties of the evolutionary algorithms (EA). Those algorithms are procedures to search the optimum in the feasible space of solutions.

The EA generates a cluster of points in the feasible domain. The cluster is positioned closely to the optimum. There is a great possibility that the optimum is the global optimum.

There is a risk that the points are located close to the more than one optimum. In this case the second stage (local method) can work unstably. It can be solved in a few ways [5].

In the second stage of the proposed strategy, several best points in this region are selected. Then, these points play the role of the cloud and as previously shown, the local method is beginning. This method is based on the gradient method, but the sensitivity analysis is evaluated by the neuro-computing.

The special multilevel artificial neural network (RBF) is used as the approximation method of the stochastic problem. Each level corresponds with a selected moment of the stochastic number and is modeled as a simple perceptron. In presented approach the random variable is modeled using two moments, therefore the 2-levels neural network is assumed.

The second stage of the two-stage strategy is a combination of the classical gradient method (the steepest descent method) and the RBF network. In order to perform the optimization process the special procedures to defined the stochastic gradient are proposed. Due to coupling the stochastic gradient with the special multilevel neural network, the local stochastic optimization gives promissing results.

## 4 The evolutionary algorithm for stochastic programming

Stochastic optimization problems emerge when some parameters of the objective function or constraints have probabilistic nature. The application of evolutionary algorithms to solve such problems requires some modifications of the traditional evolutionary approaches because chromosomes consist of random genes $X_i(\gamma)$, $i$=1, 2,…, $n$, described by moments, e.g. by the mean value $m_i$ and the standard deviation $\sigma_i$ in the case of Gaussian independent random genes. The mean idea of this algorithm is similar to the traditional evolutionary algorithm but all steps of the algorithm must be modified to the stochastic data and their moments. Each individual expresses a stochastic solution. Each solution is evaluated, and a stochastic value of the fitness function is obtained as the result. The next generation is constructed on the basis of better stochastic chromosomes of the previous generation. In this case the special types of relations are defined. Also the stochastic types of operators (mutation and crossover) are constructed. It can be observe that the next population in the stochastic evolutionary algorithm is better than the previous one.

The stochastic problem is solved by using mathematical operations on moments (e.g. the mean value $m_i$ and the standard deviation $\sigma_i$, etc.). Therefore, the original stochastic problem is considered as an equivalent deterministic one. This technique is known very well in other problems, e.g. in solving the stochastic programming, stochastic differential equations, etc.

Instead of the random chromosome $\mathbf{X}(\gamma)$ one can consider a deterministic chromosome **chr** which consists of *n*-vector pars of genes $\mathbf{g}_i=(m_i,\sigma_i)$ , $i$=1,2,…,$n$ which corresponding to random variable $X_i(\gamma)$

$$\mathbf{chr}=\left[\mathbf{g}_1;\mathbf{g}_2;...\mathbf{g}_i;...;\mathbf{g}_n\right]=\left[(m_1,\sigma_1);(m_2,\sigma_2);...;(m_i,\sigma_i);...;(m_n,\sigma_n)\right] \tag{8}$$

In the more general cases the gene $\mathbf{g}_i$ consists of more moments of random variable $X_i(\gamma)$. Due to the Gaussian distribution two moments are sufficient to describe random genes.

In presented algorithm the evolutionary operators concern to the random genes $X_i(\gamma)$ by modification of mean values $m_i$ and standard deviations $\sigma_i$.

It is worth to stress some similarities between the proposed approach to the stochastic representation of the EA and the evolutionary strategies.

For each vector gene $\mathbf{g}_i=(m_i,\sigma_i)$ $i$=1,2,…,$n$, two kinds of constraints are imposed:

$$m_i^{\min}\le m_i\le m_i^{\max} \tag{9}$$

$$\sigma_i^{\min}\le \sigma_i\le \sigma_i^{\max} \tag{10}$$

where indices: *min* and *max* mean here the maximum and minimum values.

The special stochastic kind of mutation, crossover and selection operators are applied. This algorithm is more detailed described in [5].

## 5 The artificial neural network

Consider the artificial neural network with the radial activation functions (RBF). The RBF for approximation the fitness function is used. The number of neurons in the input layer is equal to the number of design variables of the fitness function. In the output layer there is only one neuron, its output value plays the role of the fitness function.

The number of neurons in the hidden layer depends on the degree of difficulty of the function.

The output value of $i$-th neuron in the hidden layer is expressed by:

$$e_i=f\left(u_i\right)=e^{-0.5u_i} \tag{11}$$

where:

$$u_i=\sum_{k=1}^{N}\left(\frac{e_k-t_k^i}{\sigma_k^i}\right)^2 \tag{12}$$

where: $t_k^i$ – centrum of $k$-th radial function, $\sigma_k^i$ – standard deviation of $k$-th radial function, $e_k$ - input value of the neural network, $N$ – number of inputs.

The output value of the neuron in the output layer is computed as follows:

$$E=\sum_{j=1}^{M}w_je_j+w_0 \tag{13}$$

where: $M$ – number of neurons, $w_j$ - weigths.

Computing of sensitivity to the radial activation function is very easy. The sensitivity to the output signal $E$ of the RBF network in the some $z$ input value $e_k$ is expressed by [8]:

$$\frac{dE}{de_k} = \sum_{j=1}^{M} \left( -w_j \exp\left( -\frac{1}{2} u_j \right) \frac{e_k - t_k^j}{\left( \sigma_k^j \right)^2} \right) \tag{14}$$

## 5.1 The multilevel artificial neural network

The special multilevel artificial neural network (Fig. 1) is used as the approximation tool of the stochastic problem. Each level corresponds with selected parameter of the random number and is modelled as a simple (RBF) perceptron.

The number of the levels of multilevel neural network depends on the number of moments, which describe the random variable. In the present paper the random variable is describe using two moments: $m$ and $\sigma$, therefore the number of level is equal to 2.

All levels can be connected with each others (Fig.1). The sensitivities to the output signals of the multilevel network are expressed by formula (14).

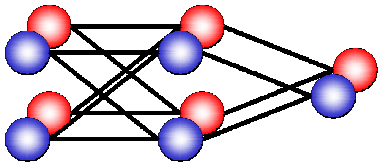

**Figure 1.** The scheme of multi-level neural networks.
(a-all levels are connected with each others; b- all levels are separated)

## 6 The local optimization method with neuro-computing

The proposed local optimization method is a combination of the classical gradient method (the steepest descent method) and the multilevel RBF network. In the first step of the algorithm a set (cloud) of stochastic points in the function domain stochastic is generated.

In order to perform the optimization process the network is constructed. The multilevel RBF network has architecture: (2/8/1). The number of levels of neural network depends on the number of moments, which describe the random number.

The starting number of training vectors is equal to $3m$, where $m$ – number of random design variables of minimizing function. In each iteration of the optimization algorithm a few steps are performed (Fig.2).

In the first step the set of training vectors of the network is created. In the first iteration the set is created on the basis of the cloud of points. The coordinates of points (moments) play the role of the input values of the network, the random fitness values in points play the role of output value of the network.

In the second step the network is trained.

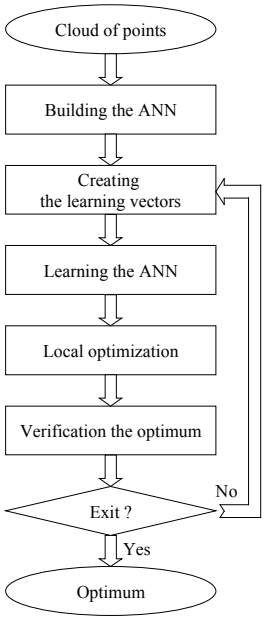


**Figure 2.** The schema of local optimization method

In the next, third step, the optimization process is carried out. The gradient method (the steepest descent method) of optimization is used. The network as the fitness function approximation is used. The gradient formula (14) is employed in computation.

The special kind of the stochastic gradient is introduced. For a point, which is a result of optimization (found in step 3), the actual fitness function is computed.

In the last step the stop condition is checked. In the case, in which the condition is true, the point is treated as the result of the optimization process. If this condition is false, this point is added to the training vector set and the next iteration is carried out (go to step 1).

This method was tested for the real and fuzzy problems and results were satisfactory.

## 7 Example

Consider the body Ω (Fig.3), bounded by boundary Γ. The material parameters of the body, shape and boundary conditions as the uncertain values can be considered. The aim of the identification problem is to find the stochastic parameters of mechanical structure.

From the mathematical point of view, the identification problem is expressed as the minimization of the special stochastic function.

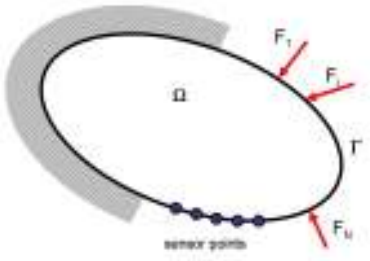


**Figure 3**. The elastic body

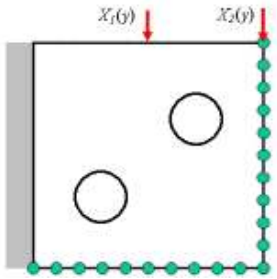


**Figure 4**. The 2-D elastic structure

In the previous works (the real problem) the identification problem was expressed as the minimization of the displacement function:

$$f = \sum_{i=1}^{n} \int_{\Gamma} \left( \left| \mathbf{d}(\gamma, \underline{\mathbf{x}}) - \hat{\mathbf{d}}(\gamma, \underline{\mathbf{x}}) \right| \right)^2 \delta\left( \underline{\mathbf{x}} - \underline{\mathbf{x}}^i \right) d\Gamma \tag{15}$$

where: $\mathbf{d}(\gamma)$ - the random measured physical quantity in sensor point $\underline{\mathbf{x}}^i$, $\hat{\mathbf{d}}(\gamma)$ - the random computed physical quantity in this same point for parameters generated by the evolutionary algorithm, $n$ – number of sensor points, $\delta$ - Dirac function. The formula (15) can be transformed to the simpler form:

$$f = \sum_{i} \left( d_i(\gamma) - \hat{d}_i(\gamma) \right)^2 \tag{16}$$

where: $d_i(\gamma)$ - the random measured physical quantity, $\hat{d}_i(\gamma)$ - the random displacements physical quantity for the structure with the parameters generated by the evolutionary algorithm.

The fitness function depends on measured physical quantity at sensor points. The sensor points are located on the surface of the body. The stochastic problem is solved using the stochastic finite element method (SFEM).

The aim of the test is to find $n$=2 random loads: $X_1(\gamma)$ and $X_2(\gamma)$ (Fig.4). The actual stochastic parameters of the load $X_1(\gamma)$ is described by: $\mathbf{g}_1=(m_1,\sigma_1)$, where: $m_1$=20.0 [kN], and $\sigma_1$=0.16 [kN]. The actual stochastic parameters of the load $X_2(\gamma)$ is described by: $\mathbf{g}_2=(m_2,\sigma_2)$, where: $m_2$ =25.0 [kN], and $\sigma_2$=0.33 [kN]. The loads are independent random variables. The stochastic chromosome $\mathbf{X}(\gamma)=[X_1(\gamma), X_2(\gamma)]$ is replaced by deterministic one containing moments of $X_i(\gamma)$ $\mathbf{chr}=\left[\mathbf{g}_1;\mathbf{g}_2\right]=\left[(m_1,\sigma_1);(m_2,\sigma_2)\right]$.

As the sensor points $N$=21 boundary nodes were selected.

The following parameters of two-stage evolutionary-neuro computing approach in the interval case were assumed: population size: 30, number of generations: 50, probability of mutation: 0.2, probability of crossover 0.1, number of iteration of local method was equal to 200.

The 25 independent experiments were performed. The results (the worst, the average and the best one) are included in Table 1.

**Table 1.** The found moments of the random loads

| Results | | The moments of random loads | | | |
|---|---|---|---|---|---|
| | | $m_1$ [kN] | $\sigma_1$ [kN] | $m_2$ [kN] | $\sigma_2$ [kN] |
| After SEA | the worst | 19.54 | 0.14 | 26.23 | 0.28 |
| | average | 20.06 | 0.16 | 25.86 | 0.33 |
| | the best | 20.01 | 0.16 | 25.09 | 0.32 |
| After two-stage evolutionary-neuro computing approach | the worst | 20.00 | 0.16 | 25.00 | 0.33 |
| | average | 20.00 | 0.16 | 25.00 | 0.33 |
| | the best | 20.00 | 0.16 | 25.00 | 0.33 |

## Conclusions

An effective two stage strategy based on the stochastic evolutionary algorithm and multilevel artificial neural networks has been presented. This approach can be applied in the optimization and identification problems.

The optimum can be found in less number of iteration due to application of the multilevel neural network for local approximation of fitness function. In the some tests the time was decreased even to 56%.

The future task is testing the influence of the parameters on the sensitivity to the algorithm: the parameters of the stochastic evolutionary algorithm and control parameters of the selection.

In the general case uncertain conditions have the granular form. The models based on the rough sets, interval and fuzzy variables can be used. The granular evolutionary algorithm can be created as a general method for all described models.

## Acknowledgements

The research is financed from the budget resource of the Foundation for Polish Science (2005-2008) and Ministry of Education and Science in the years 2006-2007 as the research project.

## Bibliography


[1] J. Arabas, *Wykłady z algorytmów ewolucyjnych*. WNT, 2001.

[2] A. Bargiela, W. Pedrycz, *Granular Computing: An introduction*. Kluwer Academic Publishers Boston/Dordrecht/London 2002.

[3] A. M. Brandt, *Criteria and Methods of Structural Optimization*. PWN Warszawa 1984.

[4] T. Burczyński, P. Orantek: Algorytm ewolucyjny bazujący na interwałowej i rozmytej reprezentacji danych. Testy numeryczne. *Zeszyty Naukowe WSInf*. Łódź 2005.

[5] T. Burczyński, P. Orantek, The evolutionary algorithm in stochastic optimization and identification problems *Evolutionary computation and global optimization 2006*, Praca zbiorowa pod red. J. Arabasa, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2006

[6] T. Burczyński, P. Orantek, The fuzzy evolutionary algorithm in structural optimization and identification problems. *Proc. 16th International Conference on Computer Methods in Mechanics CMM-2005*, Częstochowa 2005.

[7] M. Kleiber T. D. Hien, *The Stochastic Finite Element Method*. John Wiley & Sons. 1992.

[8] P. Orantek, Fuzzy evolutionary algorithms and neural networks in uncertain optimization problems. *Proc. Neural Networks and Soft Computing 2005*, Cracow, Poland

[9] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, New York 1991.

[10] R. Schaefer, *Podstawy genetycznej optymalizacji globalnej*. Wydawnictwo Uniwersytetu Jagiellońskiego, Kraków 2002.