Artificial Neural Networks for Modeling Partial Differential Equations Solution: Application to MicroSystems' Simulation.

Efrain Jaime Ang[1], Bruno Jammes[2]

1.- Universidad Autonoma de Tamaulipas, Mexico
ejefrain@laas.fr
2.- Laboratoire d'Analyse et d'Architecture des Systèmes, France
jammes@laas.fr

Abstract. This document presents a method to analytically approach the solution of partial differential equations (PDE) with a Feed-Forward Neural Network (FFNN). Such a solution should allow to introduce PDE into *electrical* or *system simulators*. Virtual design of Micro-Electro-Mechanical Systems (MEMS) would be the principal application. In this paper, the expression of the PDE solution is the sum of two terms. One satisfies the initial and/or boundary conditions. The other one includes a special FFNN, which implements a separated variable structure. The parameters of FFNN are adjusted so that the approximation will satisfy the equation all over its domain of validity. The criterion used to realize this task is the integral of the squared approximation error all over the domain of validity. This method is applied to a steady-state heat transfer problem. For this application, performances of our solution are compared with those of a finite elements method solution.

1.- Introduction.

The current trends of technologies and markets lead engineers to developed more and more complex systems, and in the same time to decrease conception time. The only way to meet these requirements, is the *virtual design*. This method is currently developed for microsystems design. However, the complexity of phenomena to take into account limits the development of such a method in this domain. In fact, partial differential equations (PDE) are often used to describe Microsystems, more particularly Micro-Electro-Mechanical Systems (MEMS) behaviour. Resolution of such equations requires specific software tools and resources of calculations are not compatible with overall simulations of systems. Moreover, current commercial tools for multi-domain system simulation, only support algebraic-differential equations. This problem is generally circumvented by using dedicated tools to solve PDE (most of the time such tools use finite elements methods (FEM)[10], then transferring a data base or a behavioural model [1,2,7] (polynomial interpolation, neural network, etc.) to the *system simulator.*

In this article, we propose to directly approach the solution of PDE with a neural network, without preliminary numerical integration. The parameters of the neural network are adjusted so that the approximation will satisfy the equation all over its domain of validity. We could define the error of approximation at any point of *I* as the error introduced by the substitution of variables by their approximations inside the equation. In this case parameters are adjusted so that they will minimize this error at any point of *I*. By this way, the resolution of PDE becomes a non linear optimisation problem.

The approximation proposed in this paper combines the structure of solution proposed by Lagaris [9] and the FFNN structure proposed by Liu[3,4,5]. The Lagaris' structure intrinsically satisfies boundary conditions, what simplifies the expression of the optimisation problem. The Liu's FFNN has a separated variable structure. This architecture was inspired by the method of separation of variables sometimes used to solve PDE. The objective function used to adjust the FFNN parameters is the integral of the squared approximation errors all over the domain of validity of the equation. Combination of the separated variable structure and the integration of error should simplify the optimisation problem resolution.

This paper is organised as follows : Section 2 describes the method, the structure of the neural network and the criterion used to identify its parameters. Section 3 presents an application of our method to a steady-state heat transfer problem. In section 4, we compare our results with a numerical solution obtained with a FEM. Finally, we conclude and propose directions for future research.

2.- Description of the Method.

Let us consider a variable V(x,y) defined on a domain *I(x,y)* and solution of a partial differential equation :

$$eq(x, y, V, \frac{\partial V}{\partial x}, \frac{\partial V}{\partial y}, ...) = 0 . \quad (1)$$

Furthermore, V(x,y) should satisfy constraints at each boundary of domain *I*:

$$bc(x, y, ...) = 0 . \quad (2)$$

In [9], Lagaris propose to approach the solution of such system of equations by the following expression :

$$V_S(x, y) = A(x, y) + F(x, y) * R_{NN} \quad (3)$$

In this equation, *A(x, y)* is a function that satisfies all the boundary conditions. It does not contain adjustable parameters. In an other hand, the function *F(x,y)* takes the zero value on each boundary. This function restricts the influence of the neural network , $R_{NN,}$ inside the domain of validity of the equation.

$$R_{NN} = R_{NN}(x, y, w_p, b_p, \ldots w_0, b_0) \quad (4)$$

The solution proposed by Lagaris intrinsically satisfies the boundary conditions, so, R_{NN} parameters identification procedure should minimize approximation error at any point of *I*:

$$Ee = eq(x, y, V_S, \frac{\partial V_S}{\partial x}, \frac{\partial V_S}{\partial y}, \ldots) \quad (5)$$

The solution proposed by Liu in [3,4,5] doesn't intrinsically satisfy the boundary conditions. R_{NN} parameters should be adjusted in order to minimize the approximation error at any point of *I* and the boundary conditions. This leads to a more complex optimisation problem. Nevertheless, we will see later that the structure of Neural Network and the goal function proposed by Liu could simplify the optimisation procedure.

The structure proposed by Liu is inspired by the variable separation method sometimes used to solve PDE. Each variable, or input, have its own units, with hyperbolic tangent activating function, in the first hidden layer.

$$f_x(x,k) = \tanh(w_{ix}(k) * x + b_{ix}(k)) \quad (6)$$

$$f_y(y,k) = \tanh(w_{iy}(k) * y + b_{iy}(k)) \quad (7)$$

The structure of the second hidden layer is original. Each unit of this layer receives the information from one hidden unit of each variable, and calculates the product of these values. So, in case of a two-variables equation, the expression of a second hidden layer unit is :

$$O_{HL2}(x,y) = (w_{ox} * f_x + b_{ox}) * (w_{oy} * f_y + b_{oy}) \quad (8)$$

The output layer is more conventional. It sums all the second hidden layer outputs. At the end, the neural network output expression is:

$$R_{NN}(x,y) = \sum\left[(w_{ox} * f_x + b_{ox}) * (w_{oy} * f_y + b_{oy})\right] \qquad (9)$$

$w_{ix}, b_{ix}, w_{ox}, b_{ox}, w_{iy}, b_{iy}, w_{oy}, b_{oy}$ are the parameters (weights) of the neural network.

Liu also proposes to replace the sum of approximation errors, at some judiciously selected points of *I*, by a space integration. This allows to execute the optimisation procedure without validation tests (particularly if we use a variable step-size integration algorithm), and to benefit from the separate variables structure of R_{NN} to reduce number of calculations in the optimisation procedure, if the dimension of the problem is higher than 1. So, the parameters of R_{NN} should minimise :

$$\iint_I Ee^2(x,y)\ dxdy \qquad (10)$$

3.- Application of the Method.

To illustrate our method, we choose to solve a steady-state heat transfer problem. We have selected such a problem because it is typical of phenomena that MEMS' designers would introduce into *system simulators*. The characteristics of our problem are:

Domain *I(x ,y)*	x = [0,1] cm, y = [0,0.6] cm.
Thermal conductivity coef. ***K***	$1.0*10^{-3}$ Wcm $^{-1}$ O C $^{-1}$
Uniform heat power ***W***	1.0 W cm $^{-3}$

If *k* is constant, the heat transfer is described by following equation:

$$k\left(\frac{\partial^2 T(x,y)}{\partial x^2} + \frac{\partial^2 T(x,y)}{\partial y^2}\right) + W = 0 \qquad (11)$$

We selected Dirichlet boundary conditions:

$$T(0,y) = 0 \quad T(1,y) = 0 \quad T(x,0) = 0 \text{ and } T(x,0.6) = 0 \qquad (12)$$

The solution of this problem is approached by:

$$T_S(x,y) = A\ (x,y) + x(1-x)y(0.6-y)\ \ R_{NN} \qquad \textbf{(13)}$$

$$A(x,y) = 0 \ \text{ and } \quad F(x,y) = x(1-x)y(0.6-y) \qquad \textbf{(14)}$$

Thus, the goal function the parameters of R_{NN} should minimize is:

$$\int \left(Ee\right)^2 = \int \left\{ \frac{\partial^2 T_S(x,y)}{\partial x^2} + \frac{\partial^2 T_S(x,y)}{\partial y^2} + W \right\}^2 \qquad \textbf{(15)}$$

4.- Results.

The neural network approximation presented in this section uses 10 units in the first hidden layer. The analytical solution of this problem is not known. So we supposed that the *exact solution* of our problem could be provided by a FEM with a very small grid (100 x100 nodes). We verified that precision of this numerical integration is higher than 10^{-5}.

The following figure (Fig. 1) compares the neural network approximation and the *exact solution*. Figure 2 gives the difference between theses solutions. We notice that the maximum error value of the neural network approximation is order of 0.0026.

Figure 3 presents the R_{NN} contribution in the solution.

In order to compare the *volume*, this means the number of required parameters, of the finite element method solution and the neural network approximation, we looked for a numerical solution that provides the same precision that the neural network approximation. We met this condition with a 12-by-12 nodes grid. Accuracy of this solution is presented in fig. 4. We can see the maximum amplitude of the error is close to 0.0026. In spite of the distribution of the error inside the domain hugely differs from the one obtained with the neural network, we consider the precision of both these solutions are equivalent. In this case, we can conclude that the number of parameter of the numerical solution is more or less 3 times as many as the neural network approximation : 40 parameters for the FFNN and 144 nodes for the FEM solution.

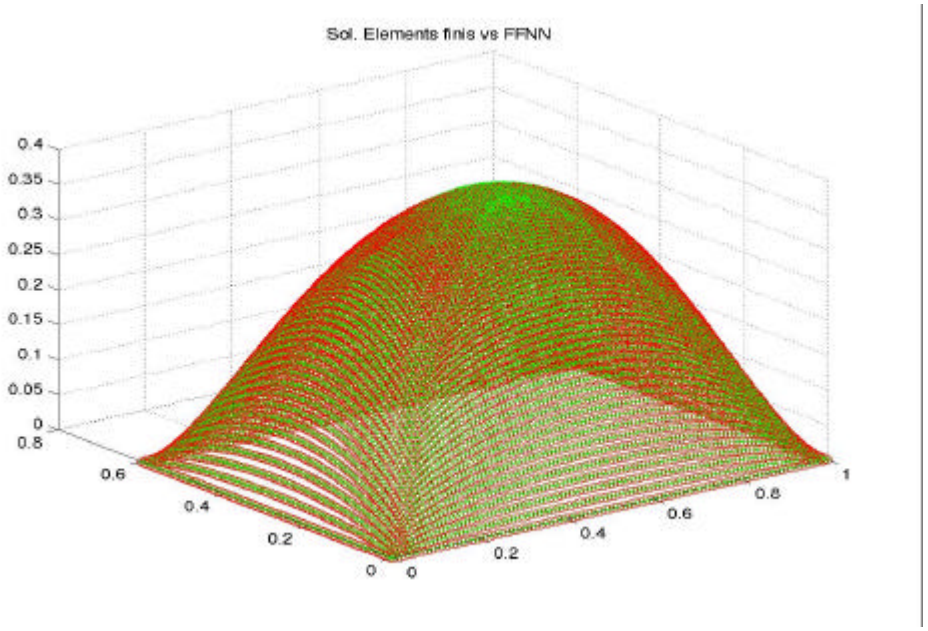

Fig. 1. The Exact Solution and the Neural Network approximation.

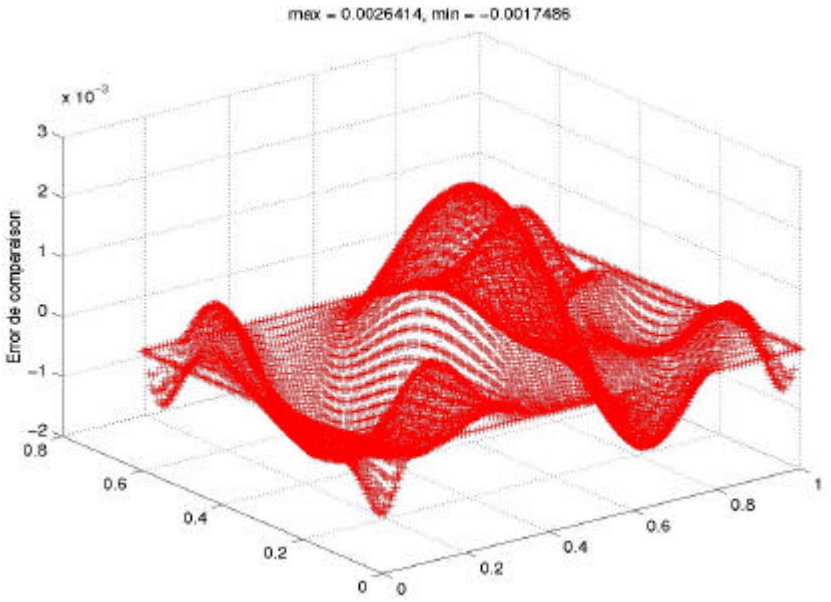

Fig. 2. Accuracy of Neural Network approximation.

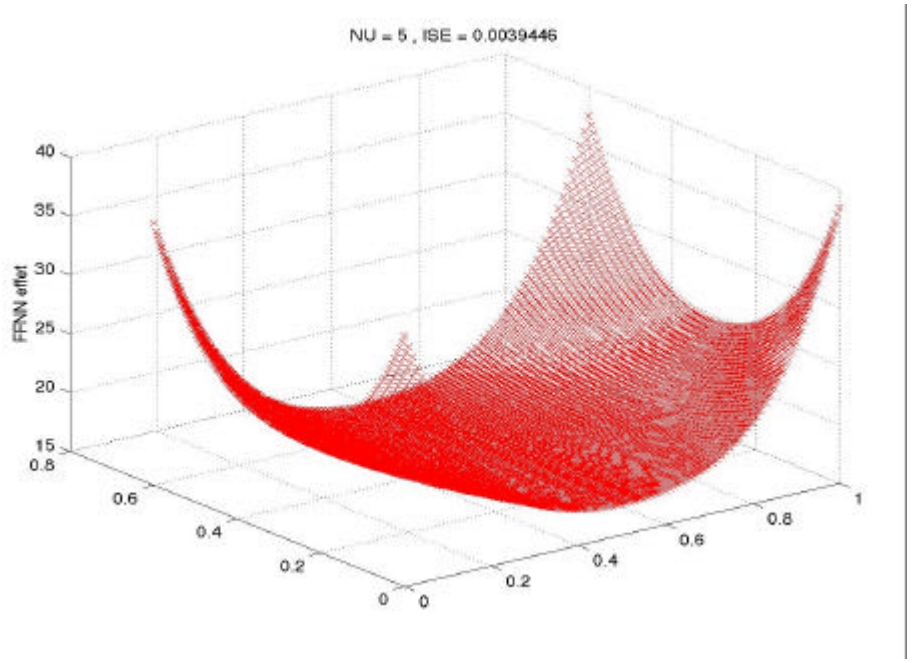

Fig.3. R_{NN} contribution in the solution.

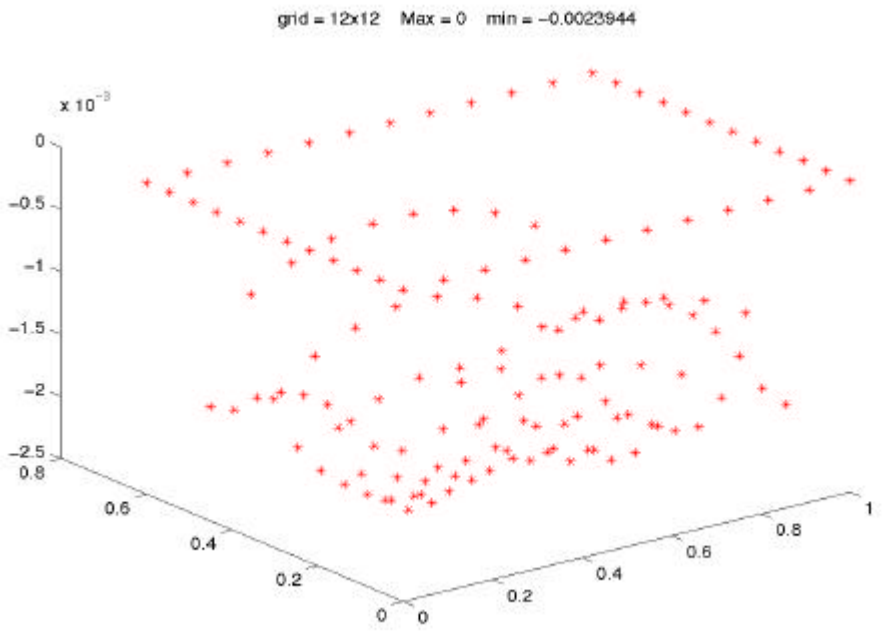

Fig. 4. Accuracy of the 12-by-12 nodes FEM solution.

5.-Conclusions and Future Research.

The approximation of PDE solution proposed in this paper combines the structure of solution proposed by Lagaris, and the FFNN structure and the goal function proposed by Liu. In brief, we have associated a structure of solution that explicitly satisfies the boundary conditions with a new goal function that takes into account the approximation error all over the domain of validity of the equation, and a neural network architecture that simplifies the resolution of the optimisation problem.

The example presented in this paper shows that the neural network approximation requires, for a given precision, less parameters than the node number of an FEM solution. It is important also to notice that the approximation of PDE solution proposed in this paper is analytical. This means it can directly be exploited at any point of its domain of validity, contrary to FEM solutions, which requires interpolations between nodes.

The ability of the solution proposed here to solve the steady-state heat transfer problem, let us to think this is the good approach for introducing PDE in *system simulators*.

The future researches of this work will be in two direction :

- We need an efficient algorithm to solve the optimisation problem. We have implemented several algorithms : Levenberg-Marquardt, Broyden-Fletcher-Goldfard-Shanno and Powell (one-dimension research) [6,11,12], but we were never satisfied : they are either too slow (Powell) or too sensitive to local minimum.
- We need to validate this method on more complex surfaces or equations.

References.

1. Meade A. and A. Fernandez, Solution of no-linear ordinary differential equations by feed-forward neural networks, Math. Comput. Mod, Vol. 20, No. 9, pp 19-44.1994.

2. Meade A. and A. Fernandez, The numerical solution of linear ordinary differential equations by FFNN. Math. Comput. Mod, Vol. 19, No. 12, pp 1-25. 1994.

3. Boan Liu and Bruno Jammes, Solving Ordinary Differential Equations by Neural Networks , Proceeding of 13th European Simulation Multi-conference Modelling and Simulation: A Tool for the Next Millennium. June 1-4, 1999, Warsaw, Poland.

4. Boan Liu, Analytical Approximation of the Solution of Ordinary and Partial Derivative Equations with Artificial Neural Networks, Rapport LAAS/CNRS No. 00623, 6 April 2000.

5. Boan Liu, Xin Zhou and Jammes Bruno, Solving Ordinary Differential Equations by Neural Networks: Application to Steady-state Heat Transfer Problem, 8th International Conference On Neural Information Processing (ICONIP-2001), pp.1069-1074,Shanghai China, Nov. 14-18,2001.

6. Eric Walter, Luc Pronzato, Identification de Modèles Paramétriques à Partir de Données Expérimentales**,** Masson, Edition 1994.

7. Freeman J., Skapura D., Neural Networks,Algorithms, Applications and Programming Techniques**.** Addison-Wesley Publishing Company. 1992.

8. George F Carrier, Carl E. Pearson, Partial Differential Equations theory and Technique**,** Academic Press, Second Edition, 1998.

9. Issac Lagaris, Aristidis Likas and Dimitrios Fotiadis, Artificial Neural Networks for Solving Ordinary and Partial Differential Equations, IEEE Transactions on Neural Networks, Vol. 9, No. 5, September 1998.

10. J.N. Reddy, An Introduction to the Finite Element Method, Mc Graw Hill, Second Edition 1993.

11. J. F. Bonnans,J.C.Gilbert,C.Lemarechal, C. Sagastizabal, Optimisation Numerique Aspects Theoriques et Pratiques, Springer-Verlag, 1997.

12. S.S. Rao, Optimization Theory and Applications, Wiley Eastern Limited, Second Edition 1985.

Title :

Artificial Neural Networks for Modeling Partial Differential Equations Solution: Application to MicroSystems' Simulation.

Authors:

Efrain Jaime Ang.
4,impasse de Champlain 31520.
Ramonville St. Agne France.
Tel. 05.62.33.62.62
ejefrain@laas.fr

Bruno Jammes
7 Avenue du Colonel Roche
31077 Cedex 4
Toulouse, France.
Tel. 05.61.33.69.91
jammes@laas.fr

Abstract. This document presents a method to analytically approach the solution of partial differential equations (PDE) with a Feed-Forward Neural Network (FFNN). Such a solution should allow to introduce PDE into *electrical* or *system simulators*. Virtual design of Micro-Electro-Mechanical Systems (MEMS) would be the principal application. In this paper, the expression of the PDE solution is the sum of two terms. One satisfies the initial and/or boundary conditions. The other one includes a special FFNN, which implements a separated variable structure. The parameters of FFNN are adjusted so that the approximation will satisfy the equation all over its domain of validity. The criterion used to realize this task is the integral of the squared approximation error all over the domain of validity. This method is applied to a steady-state heat transfer problem. For this application, performances of our solution are compared with those of a finite elements method solution.

Index Terms.- Partial differential equations, system simulators, Feed-Forward Neural Network, Micro-Electro-Mechanical Systems, separation of variable method.

Topic:

1.- Artificial Neural Model.
2.-soft-computing.
3.-Optimization.

PAPER TRACK