Proceedings of the 6th WSEAS Int. Conf. on EVOLUTIONARY COMPUTING, Lisbon, Portugal, June 16-18, 2005 (pp24-28)

# Solving Non-linear Equations via Genetic Algorithms

Nikos E. Mastorakis
Head of the Department of Computer Science
Military Institutes of University Education (ASEI)
Hellenic Naval Academy, Terma Hatzikyriakou 18539, Piraeus
GREECE
http://www.wseas.org/mastorakis

*Abstract:* - In this paper, a method for solving non-linear equations via GA (Genetic Algorithms) is presented. The method is extended for systems of non – linear equations. The method is compared with a previous one in the literature and it is found that the present one is better. Directions for future research are also provided.

*Key-Words:* - Non – linear equations, genetic algorithms, numerical solutions

## 1 Introduction

Mathematical models for a wide variety of problems in science and engineering can be formulated into equations of the form

$$f(x) = 0 \qquad (1)$$

where $x$ and $f(x)$ may be real, complex or vector quantities. There exist a great variety of methods to find the roots of (1): a) Analytical methods b) Graphical methods c) Trial and error methods d) Iterative methods. In this paper, we present GA (Genetic Algorithms) methods for solving (1).

In section 2, Equation (1) is transformed to a minimization problem and a GA is applied for finding the minimum.

## 2 Problem Formulation and Solution

A brief overview of the GAs methodology could be the following: Suppose that we have to maximize (minimize) the function $Q(x)$ which is not necessary continuous or differentiable. GAs are search algorithms which initially were insiped by the process of natural genetics (reproduction of an original population, performance of crossover and mutation, selection of the best). The main idea for an optimization problem is to start our search no with one initial point, but with a population of initial points. The 2*n* numbers (points) of this initial set (called population, quite analogously to biological systems) are converted to the binary system. In the sequel, they are considered as chromosomes (actually sequences of 0 and 1).

The next step is to form pairs of these points who will be considered as parents for a "reproduction" (see the following figure)

$$\left.\begin{array}{l} 01100|100...11 \\ 00011|101...10 \end{array}\right\} \rightarrow \begin{array}{l} 01100101...10 \\ 00011100...11 \end{array}$$

parents children

"Parents" come to "reproduction" where they interchange parts of their "genetic material". (This is achieved by the so-called crossover, see the previous figure) whereas always a very small probability for a Mutation exists. (Mutation is the phenomenon where quite randomly - with a very small probability though - a 0 becomes 1 or a 1 becomes 0). Assume that every pair of "parents" gives k children.

By the reproduction the population of the "parents" are enhanced by the "children" and we have an increasement of the original population because new members were added (parents always belong to the considered population). The new population has now 2*n*+*kn* members. Then the process of natural selection is applied. According the concept of natural selection, from the *2n*+*kn* members, only 2*n* survive. These *2n* members are selected as the members with the higher values of *Q*Q, if we attempt to achieve maximization of *Q* (or with the lower values of *Q*, if we attempt to achieve minimization of *Q*Q). By repeated iterations of reproduction (under crossover and mutation) and natural selection we can find the minimum (or maximum) of $Q$ as the point to which the best values of our population converge. The termination criterion is fulfilled if the mean value of $Q$ in the 2n-members population is no longer improved (maximized or minimized). More detailed overviews of GAs can be found in [1], [2], [3] and [4]

In this paper, our problem is the solution of the equation

$$f(x) = 0 \quad (1)$$

To this end, the square function $Q(x)$

$$Q(x) = f^2(x) \geq 0 \quad (2)$$

is considered. We can find a solution of (1) by finding the global minimum of $Q(x)$ in (2).

This minimization is achieved by GA (Genetic Algorithm). A slightly different method can be developed using

$$Q(x) = |f(x)| \geq 0 \quad (3)$$

demanding minimization of $Q(x)$. Note that the absolute value in (3) does not cause problem in our GA since GA's do not require the differentiability of the objective function.

Another method for solving (1) is the method developed by Angel Kuri – Morales in [5] and is described as follows:

$\min f(x)$ under the constraint $f(x) \geq 0$ (or $f(x) \leq 0$).

The previous ideas can also be extended in the case of a system of $n$ equations in $n$ unknown variables.

$$\begin{aligned} f_1(x_1, x_2, ..., x_n) &= 0 \\ f_2(x_1, x_2, ..., x_n) &= 0 \\ &\vdots \\ f_n(x_1, x_2, ..., x_n) &= 0 \end{aligned} \quad (4)$$

The square function $Q(x) = f_1^2 + f_2^2 + \cdots + f_n^2$ or the absolute value function $Q(x) = |f_1| + |f_2| + \cdots + |f_n|$ are defined (or in general any suitable norm of $\vec{f} = (f_1, f_2, ..., f_n)$) and our problem is min $Q(x)$

If the global minimum of $Q(x)$ is 0 at the point $(x_1^*, x_2^*, \cdots, x_n^*)$ then $x_1^*, x_2^*, \cdots, x_n^*$ is a solution of (4)

Another method for solving (4) is $\min(f_1 + f_2 + ... + f_n)$ under the constraints $f_1(x) \geq 0, f_2(x) \geq 0, \cdots, f_n(x) \geq 0$ (method developed by Angel Kuri – Morales, [5])

In this paper, we can see that the proposed methodologies are better than the methodology of Angel Kuri – Morales, [5]).

Let us consider the following example

Consider the system of the equations

$$\begin{aligned} x_1^2 + x_1 \cdot x_2 - 6 &= 0 \\ x_1^2 + x_2^3 + 2 \cdot x_1 \cdot x_2^2 - 3 &= 0 \end{aligned} \quad (5)$$

1st method

We demand

$$\min(x_1^2 + x_1 \cdot x_2 - 6)^2 + (x_1^2 + x_2^3 + 2 \cdot x_1 \cdot x_2^2 - 3)^2$$

2nd method

We demand

$$\min|x_1^2 + x_1 \cdot x_2 - 6| + |x_1^2 + x_2^3 + 2 \cdot x_1 \cdot x_2^2 - 3|$$

3rd method (Method Angel – Kuri Morales)

We demand

$$\min(x_1^2 + x_1 \cdot x_2 - 6 + x_1^2 + x_2^3 + 2 \cdot x_1 \cdot x_2^2 - 3)$$

under the constraints

$$x_1^2 + x_1 \cdot x_2 - 6 \geq 0$$

and

$$x_1^2 + x_2^3 + 2 \cdot x_1 \cdot x_2^2 - 3 \geq 0$$

The following figures show the results in each case. where the following notation is used

$$F_{ga} = (x_1^{\ 2} + x_1 \cdot x_2 - 6)^2 + (x_1^{\ 2} + x_2^{\ 3} + 2 \cdot x_1 \cdot x_2^{\ 2} - 3)^2$$

for the first method

$$F_{ga} = \left| x_1^{\ 2} + x_1 \cdot x_2 - 6 \right| + \left| x_1^{\ 2} + x_2^{\ 3} + 2 \cdot x_1 \cdot x_2^{\ 2} - 3 \right|$$

for the second method

$$F_{ga} = x_1^{\ 2} + x_1 \cdot x_2 - 6 + x_1^{\ 2} + x_2^{\ 3} + 2 \cdot x_1 \cdot x_2^{\ 2} - 3$$

for the third method

In each case (i.e. in each method) we use:

$n$ = 20 (i.e. the number of parents is 2 $n$ = 40)

$k$=4

$p_{mutation} = 0.1$ (probability for mutation in every generation)

Also our chromosome is of 20 bits

And we search for $x_1, x_2$ in the range -100, +100

That means: $x_1 \in [-100,100]$, $x_2 \in [-100,100]$

We have the following results (in the horizontal axis we present the number of generations, i.e. iterations of the GA)

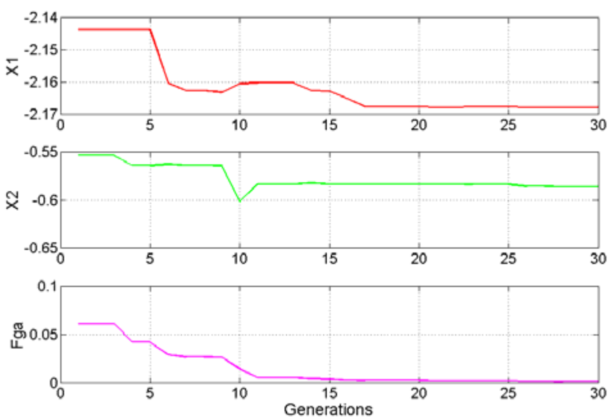


Fig.1: 1st Method

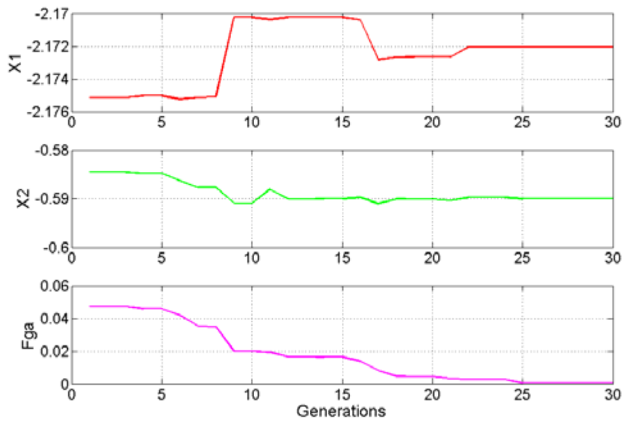


Fig.2: 2nd Method

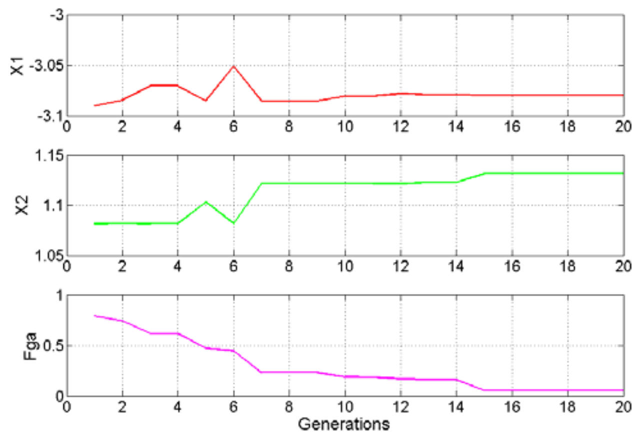


Fig.3: 3rd Method

| Method | $F_{ga}$ | $x_1$ | $x_2$ |
|---|---|---|---|
| 1st | 0.00003591 | -2.17316835 | -0.59035358 |
| 2nd | 0.00061440 | -2.17206208 | -0.59006747 |
| 3rd | 0.05382000 | -3.08009441 | 1.13125909 |

That means that the first method is better than that of [5].

On the other hand the 3rd method has a serious disadvantage. It cannot converges to $x_1 = -2.17$ and $x_2 = -0.59$, because $x_1 = -2.17$ and $x_2 = -0.59$ the following constraint (inequality) fails $x_1^{\ 2} + x_1 \cdot x_2 - 6 \geq 0$

# 3 Determining all possible roots

## 3.1 Case of one equation

The previous developed methodology estimate only one root. If we are interested for all the roots, we can use the so called "incremental search" [6] as follows. Let's start with one equation like (1).
We choose a lower limit **a** and an upper limit **b** of the interval covering all the roots. We also have to choose the size of incremental interval $\Delta x$. A major problem is to decide the increment size $\Delta x$. A small $\Delta x$ means more iterations, more execution time, but we avoid missing some root or roots.
Our algorithm for determining all possible roots of (1) is formulated as follows:

**Step1:** Choose lower limit **a** and upper limit **b** of the interval covering all the roots.

**Step 2:** Decide the size of the incremental interval $h = \Delta x$

**Step 3:** Set $x_1 := \alpha$ and $x_2 := x_1 + h$

**Step 4:** Compute $f_1 := f(x_1)$ and $f_2 := f(x_2)$

**Step 5:** If $f_1, f_2$ are of the same sign (i.e $f_1 \cdot f_2 > 0$) (that means the interval $[x_1, x_2]$ does not bracket any root), then go to **Step 7**

**Step 6:** Apply GA and find the root $x^*$ inside the interval $[x_1, x_2]$.

**Step 7:** If $x_2 + h < b$ then set $x_1 := x_2$ and $x_2 := x_2 + h$ and go to Step 4
**Step 8:** Stop

## 3.2 Case of system of equations

Suppose that the system of equations are as in (4). We choose lower limit and upper limit for each variable say: $\underline{x}_i, \overline{x}_i$ for $i = 1,2,\cdots,n$
That means that we seek roots in the closed subset

$$S = [\underline{x}_1, \overline{x}_1] \times [\underline{x}_2, \overline{x}_2] \times \cdots \times [\underline{x}_n, \overline{x}_n] \subset I\Re^n \qquad (6)$$

A generalized incremental search can be achieved considering the incremental intervals

$$h_1 = \Delta x_1, h_2 = \Delta x_2, \cdots, h_n = \Delta x_n$$

Consider that $[\underline{x}_i, \overline{x}_i]$ has been divided into $N_i$ parts, i.e.

$h_i = \dfrac{\overline{x}_i - \underline{x}_i}{N_i}$ $i = 1,2,\cdots,n$ and considering the elementary cell

$$C_{k_1,k_2,\cdots k_n} = [\underline{x}_1 + k_1 h_1, \underline{x}_1 + k_1 h_1 + h_1] \times$$
$$[\underline{x}_2 + k_2 h_2, \underline{x}_2 + k_2 h_2 + h_2] \times \cdots$$
$$\times [\underline{x}_n + k_n h_n, \underline{x}_n + k_n h_n + h_n]$$

Note that

$$\overline{x}_1 = \underline{x}_1 + N_1 h_1$$
$$\overline{x}_2 = \underline{x}_2 + N_2 h_2$$

$$\overline{x}_n = \underline{x}_n + N_n h_n$$

where

$$0 \le k_1 \le N_1 - 1,\ 0 \le k_2 \le N_2 - 1, \cdots,\ 0 \le k_n \le N_n - 1 \quad (7)$$

So **S** in (6) has been devided into $N_1 N_2 \cdots N_n$ elementary cells.

We examine the sign of $f_1, f_2, \cdots f_n$ on the $2^n$ edge points of each cell. If at least one $f_i$ maintains the same sign on the $2^n$ edge points $(e_1, e_2, ..., e_n)$ of the elementary cell $C_{k_1,k_2,\cdots k_n}$ of (7) then: do not seek for possible roots of (4) in $C_{k_1,k_2,\cdots k_n}$, do not apply the GA and go to next cell.

Note that: $(e_1, e_2, ..., e_n)$ are the $2^n$ edge points

$$e_1 = \underline{x}_1 + k_1 h \quad or \quad \underline{x}_1 + k_1 h + h_1$$
$$e_2 = \underline{x}_2 + k_2 h \quad or \quad \underline{x}_2 + k_2 h + h_2$$
$$\vdots$$
$$e_n = \underline{x}_n + k_2 h \quad or \quad \underline{x}_n + k_2 h + h_2$$

If each $f_i$ change signs on the $2^n$ edge points $(e_1, e_2, ..., e_n)$ of the elementary cell $C_{k_1,k_2,\cdots k_n}$ of (7) then seek for a root of (4) inside $C_{k_1,k_2,\cdots k_n}$ by applying GA.

Repeat this procedure until exhausting all the $N_1 N_2 \cdots N_n$ elementary cells.

## 4 Conclusion

GA (Genetic Algorithms) is really a powerful tool for many problems in numerical analysis and scientific computation. In this paper, we apply GA to solve a non-linear equation as well as systems of non-linear equations. Work is in progress by the author in the direction of applying Gas in many open problems of numerical analysis such a polynomial factorization, solution of differential equations, etc. See [7]÷[10]. Other relevant studies can be found in [11], [12].

*References:*


[1] Goldberg D.E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Second Edition, 1989

[2] Grefenstette J.J., Optimization of control parameters for Genetic Algorithms, IEEE Trans. Systems, Man and Cybernetics, SMC 16, Jan/Feb 1986, pp. 128

[3] Eberhart R., Simpson P. and Dobbins R. (1996), *Computational Intelligence PC Tools*, AP Professionals.

[4] Kosters W.A., Kok J.N. and Floreen P., Fourier Analysis of Genetic Algorithms, Theoretical Computer Science, Elsevier, 229, 199, pp. 143-175.

[5] Angel Fernando Kuri-Morales, “Solution of Simultaneous Non-Linear Equations using Genetic Algorithms”, WSEAS Transactions on SYSTEMS, Issue 1, Volume 2, January 2003, pp.44-51

[6] E. Balagusuramy, *Numerical Methods*, Tata McGraw Hill, New Delhi, 1999

[7] Ioannis F. Gonos, Lefteris I. Virirakis, Nikos E. Mastorakis, M.N.S. Swamy, "Evolutionary Design of 2-Dimensional Recursive Filters via the Computer Language GENETICA" to appear in IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications. (2005)

[8] Gonos I.F., Mastorakis N.E., Swamy M.N.S.: “A Genetic Algorithm Approach to the Problem of Factorization of General Multidimensional Polynomials”, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, Part I, Vol. 50, No. 1, pp. 16-22, January 2003.

[9] Mastorakis N.E., Gonos I.F., Swamy M.N.S.: “Design of 2-Dimensional Recursive Filters using Genetic Algorithms”, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, Part I, Vol. 50, No. 5, pp. 634-639, May 2003.

[10] Mastorakis N.E., Gonos I.F., Swamy M.N.S.: “Stability of Multidimensional Systems using Genetic Algorithms”, IEEE Transactions on Circuits and Systems, Part I, Vol. 50, No. 7, pp. 962-965, July 2003.

[11] I. F. Gonos and I. A. Stathopulos, “Estimation of multi-layer soil parameters using genetic algorithms”, IEEE Transactions on Power Delivery, vol. 20, no. 1, Jan. 2005.

[12] I. F. Gonos, F. V. Topalis and I. A. Stathopulos, “A genetic algorithm approach to the modeling of polluted insulators”, IEE Proceedings Generation, Transmission and Distribution, vol. 149, No. 3, May 2002, pp. 373-376.