Application of Genetic Algorithm in Numerical Solution of Two-dimensional Laplace's Equation

Mehran Qate¹, Majid Pourabdian¹, ²

¹Department of Aerospace Engineering, Sharif University of Technology, Tehran, Iran

²Corresponding Author; Department of Aerospace Engineering, Sharif University of Technology, Tehran, Iran, Email: m.pourabdian@gmail.com

Abstract
This paper describes the impact of special application of Genetic Algorithm (GA) in solving CFD problems. This investigation deals with numerical solving of 2-D Laplace’s equation in uniformly-spaced square domain. Present study reflects the ability of GA in the convergence rate of first steps of the solution in comparison to other CFD methods. However, it does not guarantee the convergence of the whole solution, therefore, multigrid method is used to compensate the inaccuracy of GA in the final steps. Results have shown that the problem’s solution applying only GA is not accurate enough and it decreases the convergence rate, whereas using GA and multigrid together increases the convergence rate and keeps the accuracy of the solution.

Keywords: Genetic Algorithm, Laplace’s Equation, GA, Multigrid

1. Introduction
Genetic Algorithms (GAs) are adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic. The basic concept of GAs is designed to simulate processes in natural system necessary for evolution. The first basis and theory for Genetic algorithms (GAs) as stochastic search and optimization algorithms is provided by John Holland [1]. They stem from the idea of biological evolution where organisms mate and natural selection guarantees the route of individuals with better qualities to the next generation. In a genetic algorithm, a random population of possible solutions, called chromosomes is initialized and let evolve, loosely following the same Darwinian principle of “survival of the fittest.” In the biological sense, all living organisms consist of cells. In each cell there is the same set of chromosomes. Chromosomes are strings of DNA and serves as a model for the whole organism. A chromosome consist of genes, blocks of DNA. Each gene encodes a particular protein. Basically can be said, that each gene encodes
a trait, for example color of eyes. Possible settings for a trait (e.g. blue, brown) are called alleles. Each gene has its own position in the chromosome. The ‘mating’ of individuals involves crossing over the second parent, producing two different offspring. Occasionally, but rarely, gene is mutated at random: one or more allele is changed (flipped if binary genes are used, and perturbed if real-valued genes are used). The two genetic operators, crossover and mutation, are the two main components of a GA simulation. The performance is influenced mainly by these two operators. The general aim is that the better qualities (solutions) are preserved with high probability and the worse solutions rapidly discarded. Genetic algorithms have been used in many fields of optimization. They were also applied to other areas of optimization like radio antennas, Jones and Joines [2], fin profile designs, Fabri [3], inverse initial-value boundary-value problems, Karr et al. [4], and even areas as curious as fashion design, Kim and Cho [5], and music composition, Haupt and Haupt [6]. Various problems in aerodynamics, ranging from wing shape optimization to active noise control, Milano and Kounoutsakos [7], have also been tackled using GA. More recently, however, genetic algorithms have been employed to an increasing number of engineering problems in the areas of heat transfer and fluid mechanics. They also have been applied to basic heat transfer problems, Davalos and Rubinsky [8], multiphase flow functions estimation, Akin and Demiral [9], and pipeline flow optimization, Vuković and Sopeta [10]. The methodology has also been applied to fluid flow problems; Fan et al. [11] used GA to solve a potential flow problem for a simple two-dimensional circular diffuser cascade with 40 node points. It has been reported that the real-coded GAs outperformed binary-coded GAs in many types of design problems. However, even the real-coded GAs lead to premature convergence (to a local extremum) when applied to problems with a large number of design variables (constraints). In [12], Bourisli and Kaminski introduced a new strategy for adapting an evolutionary algorithm to act as a go to solver to be activated when common methods fail to achieve convergence. The outline of this paper is as follows: In Section 2, the basic concepts of genetic algorithms are described. Section 3 expresses the application of GA in solving of two-dimensional Laplace’s equation. Numerical solution and results are presented in Section 4. Finally, some conclusions are given in Section 5.

2. Concepts of Genetic Algorithm
Initially, the basic concepts of Genetic Algorithm method are introduced for solving numerical problems. For this aim, it is necessary to know some definitions:

Gene: The basic unit capable of transmitting characteristics from one generation to the next such as temperature as a variable in a specific node.

Chromosome: Structure carrying genes that determine the characteristics on organism inherits from its parents like variables.

Population: A group of chromosomes.

Parents: The candidates to generate next generation.

Algorithm is started with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are selected to form new solutions (offspring) are selected according to their fitness, the more suitable they are the more chances they have to reproduce. In other words, weaker chromosomes are doomed to be annihilated. This procedure is used in parallel with the old CFD methods and is capable to boost the rate of solution.

2.1. Crossover. Crossover is the most principle operator to generate new generation. Crossover selects genes from parent chromosomes and creates a new offspring. By cross over, each child has some information from parent’s chromosomes. This is the reason that children inherit some father’s characteristics
and some mother’s characteristics and this prevents them to be exactly like one of the parents. In this work as can be seen in Fig. 1, two-point crossover operator is used in order to prevent unreasonable children, two chromosomes break from two points, and thus new chromosomes are generated from the crossover of the first part of parents.

![Two-point crossover operator](image1.png)

**Fig. 1. Two-point crossover operator**

**2.2. Mutation.** *Mutation* is the second operator which alters one part of chromosome randomly. After a crossover is performed, mutation takes place. This is to prevent falling all solutions in population into a local optimum of solved problem. Mutation changes randomly the new offspring. By this operator, we can hope good chromosomes are omitted and revived again in generations. A random choosing of a bit by mutation operator is shown in Fig. 2. Mutation also ensures that regardless of the distribution of the initial population, the probability of searching points will not be zero in the problem’s domain.

![Random choosing of a bit by mutation operator](image2.png)

**Fig. 2. Random choosing of a bit by mutation operator**

**3. Application of GA in Solving of Two-dimensional Laplace’s Equation**

This method is applied to solve two-dimensional Laplace’s equation as a test case to examine the capability of GA. It is known that steady-state temperature (T) distribution satisfies Laplace’s equation if thermal conductivity is constant (Eq. (1)). The domain is discretized with a uniformly-spaced square grid and boundary conditions stated in (2).

\[
\nabla^2 T = 0 \quad (1)
\]

\[
T(0, y) = T(x, 0) = 0, T(1, y) = \sin(\pi y), T(x, 1) = \sin(\pi x) \quad (2)
\]

Using second-order central differences, Eq. (1) is discretized as

\[
\frac{\partial^2 T}{\partial x^2} \approx \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{\delta x^2} \quad (3a)
\]

\[
\frac{\partial^2 T}{\partial y^2} \approx \frac{T_{i,j+1} - 2T_{i,j} + T_{i,j-1}}{\delta y^2} \quad (3b)
\]

By substituting and simplifying above approximations, the nodal equation will be as Eq. (4):
\[ T_{i,j}^{n+1} = \frac{1}{4} \left( T_{i+1,j}^n + T_{i-1,j}^n + T_{i,j+1}^n + T_{i,j-1}^n \right) \] (4)

Where \( n \) is the time step and \((i,j)\) is one sample interior node. In the method used, each node and its neighbors are in a matrix, then in each step with implementation of a random function with the same distribution; two chromosomes are selected to be parents and they crossover together in order to generate children as many as initial chromosomes. Subsequently, mutation operator is exerted on generated children. To find the best chromosome and put it as the temperature of the node for the next step, error function should be utilized for more homogenization (Eq. (5)).

\[ r = \frac{1}{4} \left( T_{i+1,j} + T_{i-1,j} + T_{i,j+1} + T_{i,j-1} \right) - T_{i,j} \] (5)

Fig. 3 shows the generation of children from parents after crossover and mutation for the first population. The chromosome with the value of 4 has minimum error compared to others. In the next step, the code will use this value for nodes leading to be more homogenous domain. Crossover and Mutation operators are random number generators. In the final steps of the solution, the temperature values in each node are close to answer, so applying simple version of these operators might decrease the rate of convergence. Though some changes can be used in these operators in a way that in final steps they only change initial bits. This action causes the data to be more precise and prevent to produce coarse children. For this aim, usual CFD methods can also be applied in the final steps. Now our method’s rate increases enough without change in accuracy because of using usual CFD methods in final steps.

<table>
<thead>
<tr>
<th>Temperature Value</th>
<th>binary Value</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>1 0 0 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>112</td>
<td>0 0 0 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>220</td>
<td>0 0 1 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>45</td>
<td>1 0 1 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>66</td>
<td>0 1 0 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>3042</td>
<td>0 0 0 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>2032</td>
<td>0 0 0 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>4</td>
<td>0 0 1 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>114</td>
<td>0 0 0 ....</td>
<td>0 0 0</td>
</tr>
<tr>
<td>75</td>
<td>0 0 0 ....</td>
<td>0 0 0</td>
</tr>
</tbody>
</table>

**Fig. 3. Generation of children from parents after crossover and mutation for the first population**

**4. Results and Discussion**

Fig. 4 and Fig. 5 present the final solution of temperature distribution in a uniformly-spaced square domain using GA and multigrid methods together and only GA method, respectively. As Fig. 5 indicates accurate solution was not obtained, because when only GA is implemented for the numerical solution, it does not improve the residuals in the final steps, while to gain more accurate results, GA needs more time to converge. This is due to the fact that a coarse change in binary amount by crossover and mutation operators might give worse result in comparison with previous step, therefore the operators have to repeat the process. In another hand, when GA and multigrid methods are utilized simultaneously, the solution is more accurate.
From Fig. 6, it is obvious that convergence rate of GA is much higher than Multigrid method at the first steps (till Error e-3). Afterwards, as we march in the time, GA loses its convergence rate. To compensate this drawback, multigrid method was applied after error e-3. Thus, for any types of CFD methods, GA can be used at the first steps to improve convergence rate and for final steps the other methods (multigrid in our example) replaces so it doesn’t change the order of error. For final steps (below e-3 for this example) GA is going to produce coarse solutions and has considerable changes in binary amounts and is not capable as other CFD methods.

Fig. 6. Convergence of three methods vs. time step

5. Summary
In this paper a genetic algorithm is used and applied in solving of 2-D Laplace’s equation to improve the rate of convergence. Results have shown that the problem’s solution using only GA is not accurate enough and it decreases the convergence rate, whereas employing GA and multigrid together increases the convergence rate and keeps the accuracy of the solution. Thus, this method can be used with other types of
CFD methods as a supplementary tool. In other words, GA is utilized in each first steps of the solution and in the final steps multigrid or another specific CFD method is used to make the error better in addition to improvement in solution rate.

References


