CREATIVE MATH. & INF.
17 (2008), No. 2, 52 - 60

Online version available at http://creative-mathematics.ubm.ro/
Print Edition: ISSN 1584 - 286X Online Edition: ISSN 1843 - 441X

Symbolic approach for the generalized airfoil equation

ELENA BĂUTU AND ELENA PELICAN

ABSTRACT.

The generalized airfoil equation governs the pressure across an airfoil oscillating in a wind tunnel.
In this paper we analyze the problem for an airfoil with a flap, by means of Gene Expression Programming (GEP). We present the main traits of the GEP metaheuristic and then we define its elements in order to be used for integral equations of the first kind. The results obtained by our symbolic approach confirm the suitability of this method for problems modeled by Fredholm first kind integral equations.

1. INTRODUCTION

In aerodynamics, simulation of real-life phenomena through the use of models in wind tunnels has been used since the early stages of development of the field [1]. Although there exist major differences between two-dimensional and three-dimensional flows, solutions for two-dimensional problems provide important information to be used in understanding three-dimensional phenomena. In this paper we consider the inverse problem of deriving the pressure across a two-dimensional airfoil oscillating in a wind tunnel, knowing the oscillatory motion and the downwash velocity created around the airfoil. Usually, the derived pressure function is not used directly, but by means of integrals of the pressure, such as forces and moments.

The inverse problem we tackle is known in the literature as the generalized airfoil equation and has been studied by many authors in the recent decades (see [15], [13] [17],[5]). This singular integral equation is obtained by an integral transform technique applied to the partial differential equation for the pressure potential (see [5]), and describes the relation between the unknown pressure acting on the wing with a known oscillatory motion and the known downwash velocity.

There is no universal algorithm to obtain an exact solution to this problem. Most papers in the literature deal with numerical methods ([15], [13], [7]) for finding or at least approximating the solution of equation

$$\frac{1}{\pi}\oint_{-1}^{1}\frac{u(t)}{t-s}dt+\frac{1}{\pi}\int_{-1}^{1}k(t,s)u(t)dt=f(s). \tag{1.1}$$

with $-1<s<1$. The approach presented in this paper provides a means to obtain an approximate solution in symbolic form, without any constraints regarding the form of the solution or the numerical coefficients involved.

This paper is organized as follows. In Section 2 we review the mathematical formulation of the problem. In Section 3 we describe the symbolic approach by means of Gene Expression Programming (GEP), its application to a particular form of the generalized airfoil equation and some experimental results. Section 4 concludes the paper.

2. THE GENERALIZED AIRFOIL EQUATION

The form of the generalized airfoil equation is given in (1.1). The function we want to approximate is $u(t)$, $f(s)$ is the known downwash velocity, related to the profile function of the airfoil, and the first integral is to be considered in Cauchy principal value sense. Studies concerning the classical airfoil equation (see [8]) state that it is appropriate to look for a solution that has $(1-t)^{\frac{1}{2}}$ and $(1+t)^{\frac{1}{2}}$ as dominant endpoint singularities, when we consider that the pressure jump tends to 0 at the trailing edge of the airfoil ($u(1)=0$). Therefore, the solution to (1.1) may be considered to be of the form $u(t)=\rho(t)v(t)$, where $\rho(t)=\frac{1-t}{1+t}$. The integral equation becomes:

$$\frac{1}{\pi}\oint_{-1}^{1}\rho(t)\frac{v(t)}{t-s}dt+\frac{1}{\pi}\int_{-1}^{1}\rho(t)k(t,s)v(t)dt=f(s). \tag{2.2}$$

The equation we tackle in this paper models the case of an airfoil with a flap [15].The function $v(t)$ is expected to be smooth, except in the neighborhood of the point where the flap junctions. The following kernel function was considered:

$$k(s,t)=-\frac{1}{\pi}e^{-i(s-t)}\left[C_i(|s-t|)+i\cdot S_i(s-t)+i\cdot\frac{\pi}{2}\right], \tag{2.3}$$

where

$$C_i(z)=0.577215664\cdots+\log z+\sum_{n=1}^{\infty}(-1)^n\frac{z^{2n}}{2n(2n)!},$$

Received: 07.03.2008; In revised form: 12.09.2008; Accepted:
2000 *Mathematics Subject Classification.* 45Q05, 45B05, 68T20, 68W99.
Key words and phrases. *Generalized airfoil equation, Gene Expression Programming.*

and

$$S_i(z) = \sum_{n=0}^{\infty} (-1)^n \frac{z^{2n+1}}{(2n+1)(2n+1)!},$$

and the free term is

$$f(s) = \begin{cases} 0, & \text{if } -1 \leq s < 0 \\ 4(1+i\cdot s), & \text{if } 0 \leq s \leq 1 \end{cases}. \tag{2.4}$$

The solution we look for is in the form $v(t) = \Re v(t) + i \cdot \Im v(t)$. We will find analytical forms for both function $\Re v(t)$ and $\Im v(t)$, at the same time, separately. Thus, equation (2.2) becomes a system of integral equations, with the unknowns $\Re v(t)$, and $\Im v(t)$ respectively:

$$\begin{cases} \frac{1}{\pi} \oint_{-1}^{1} \frac{\rho(t)}{t-s} \Re v(t)dt + \int_{-1}^{1} \rho(t)(\Re k(s,t)\Re v(t) - \Im k(s,t)\Im v(t)dt = \Re f(s) \\ \\ \frac{1}{\pi} \oint_{-1}^{1} \frac{\rho(t)}{t-s} \Im v(t)dt + \int_{-1}^{1} \rho(t)(\Re k(s,t)\Im v(t) + \Re k(s,t)\Im v(t)dt = \Im f(s) \end{cases}. \tag{2.5}$$

There is no known exact solution for such a system of equations. In the next sections we will describe our symbolic approach to this problem and comment on its advantages and disadvantages.

3. The Symbolic Approach

Biological systems and their evolution are the inspiration for many new search algorithms and heuristics [16]. Evolutionary algorithms are stochastic, population-based optimization methods that make use of Darwin's model for the evolution of natural population on the basis of the "survival of the fittest" principle, in order to look for the solution to a given problem. The main trait of these algorithms is the combination of the representation of the potential solutions and the search algorithm that navigates the search space in search of an effective solution for the problem at hand.

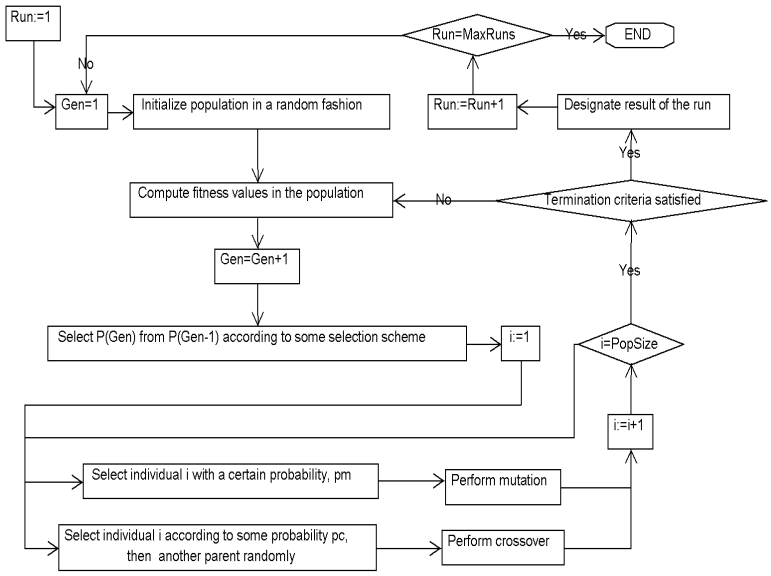
The idea behind the evolutionary algorithm is quite simple. It maintains a population of individuals that encode potential solutions to the problem at hand and evolves this population during several generations. The evolution process consists of two main stages. In terms of natural evolution, the algorithm searches for the individual that adapts best to a given medium. Metaphorically speaking, the problem to be solved is the medium that the individuals in the population maintained by the algorithm are supposed to adapt to. Therefore, in the first stage, the individuals are evaluated with respect to how well they solve the given problem and are awarded a measure of their performance, called a fitness value. According to their fitness, they are selected to survive in the next generation, meaning that they are reproduced unaltered. The second stage consists in introducing genetic variation by means of genetic operators. The traditional genetic operators are inspired by those present in nature: mutation and crossover. Mutation is a unary operator that consists in selecting an individual and performing a slight alteration of its genetic code. Crossover is a binary operator that consists in selecting two individuals and creating two offspring by swapping pieces of genetic code among them. Figure 1 presents the general architecture [11] of an evolutionary algorithm.

Genetic Programming (GP) [11] is an evolutionary technique that belongs to the family of evolutionary computation algorithms. GP uses individuals represented as nonlinear entities of different sizes and shapes – parse trees of mathematical expressions. The GP individuals encode complex compositions of functions and variables. This is a distinctive feature of GP that allows it to search spaces of complex nonlinear and nonparametric functions. GP's capability to automatically generate models via symbolic regression ([11], [2]) permits it to obtain equally fit solutions on different runs. It has been successfully implemented in several application areas like inferential sensors, emulators, accelerated new product development. Studies concerning the use of GP to first kind integral equations of Fredholm type led to promising results (see [4] and [3]). This is the one of the reasons we considered applying this technique in the study of the generalized airfoil equation.

3.1. Individual Representation for the Generalized Airfoil Equation Problem.

Since GP works with expression trees, its application to find approximate solutions in functional form for equation (1.1) in the form of complex compositions of functions, variables and constants comes natural.

Figure 1. Flowchart for the evolutionary algorithm.



The set of functions, variables and constants are provided to the algorithm at the beginning of the run. Usually, the set of functions includes arithmetic operators and mathematical functions (trigonometric, exponential, logarithmic, etc.) that may seem to help in finding a good solution. Insight into the problem domain is needed to decide this. There is a downside though - the dimension of the search space increases with every new function included in the function set, therefore we must include only those functions that we expect to help. The variable set is composed of the variables involved in the problem we tackle - in our case, since we know the functions we are looking for depend only on t, t it will be the only variable provided to the function finding algorithm. The constants are created randomly within the algorithm, from the uniform distribution on $[0, 1]$. Any other constant that may be necessary to the solution will be derived by the algorithm, combining the functions in the function set as it sees fit.

Unlike other model building techniques, GP does not impose any particular form for the solutions it evolves, nor a specific dimension. The optimization process works to find the optimal form and coefficients that best fit equation (2.2). It is important to note that GP will almost never yield an exact solution to the problem, but it will find alternative diverse solutions that fit as well as possible.

In this paper, we used a GP-like algorithm - Gene Expression Programming (GEP) (introduced in [6]), which enriches the complex structure of the individuals in GP with the benefits of the linear representation. This way, it assures separation between the phenotype and the genotype [6]. From a computational point of view, this allows the algorithm to benefit from all the advantages of the linear structure, such as the ease of manipulation of the individuals and the straightforward implementation of the genetic operators.

GEP individuals are multigenic structures that encode multiple expression trees, linked by a linking function. The genes composing the chromosomes are strings of symbols of fixed size. The gene length, the number of genes, and the linking function are among the parameters of the algorithm. The translation of a gene into the expression it encodes is a simple process [6], based on reconstructing a mathematical expression from the breadth first traversal of its expression tree.

For the problem we are treating in this paper, we need to derive two separate functions for each solution, namely $\Re v(t)$ and $\Im v(t)$. The multigenic structure of GEP usually encodes only one function. To adjust it to our needs, we extended the definition of a GEP chromosome. The first $n/2$ genes of the chromosome encode the approximate solution for finding $\Re v(t)$, and the latter half encodes the approximate solution for finding $\Im v(t)$. The number of genes (n) shall be even.

The individuals in the initial population are generated as random strings of symbols (functions, variables and constants). The only variable used was t since both functions we want to approximate depend only on it. The function set we used is composed of functions that are related to those appearing in the mathematical formulation of the problem, namely arithmetic operators $(+,\ -,\ x,\ /)$, natural logarithm, sinus and cosine. We used Koza style [11] protected versions of the division and the logarithm. Division by 0 will be evaluated to 1, and logarithm will be evaluated over the absolute value of its parameter, and will return a 1 in case of called with parameter 0, as in [11]. In the results Table 2, protected division is represented by % and protected logarithm by $rlog$.

3.2. **Fitness Measure.** Each individual in each generation of the algorithm must be evaluated according to a measure of fitness that quantifies how well it solves the given problem. We will denote an individual as c; the real part function encoded in the individual will be denoted by $\Re c(t)$ and the imaginary part will be $\Im c(t)$. The fitness of the individual

c will be computed with respect to the system of equations (2.5). The fitness of the individual is based on the errors between the left hand side of the integral equations and the right hand side:

$$error_c = \sum_{s \in S} \sqrt{(\Re\Phi_c(s) - \Re f(s))^2 + (\Im\Phi_c(s) - \Im f(s))^2}, \tag{3.6}$$

where $\Re\Phi_c(s)$ is the numerically approximated value of the sum of integrals on the left hand side of the first equation in (2.5), computed using the decoding of the first half of the current individual c as the function $\Re v(t)$, and $\Im\Phi_c(s)$ represents the approximated value of the sum of integrals on the right hand side of the second equation in (2.5). S is a set of collocation points uniformly sampled in $[-1, 1]$, in such a manner that the margins of the interval are not among the collocation points.

In order to compute the fitness, we will need to approximate numerically the integrals that appear in (2.5). The first integrals in both equations, denoted by $\oint$ are defined in the Cauchy principal value sense. For their numerical approximation, we used formula (3.7) (see [14] and [9]).

$$\oint \frac{g(t)}{t-s} = \sum_{i=1}^{n} \frac{g(t_{i,n}) - g(s)}{t_{i,n} - s} + q_0(s)g(s), \tag{3.7}$$

where $t_{i,n}$ are the roots of P_n - the Legendre polynomial of the order n, and

$$\mu_{i,n} = \frac{2(1-t_{i,n})^2}{n^2[P_{n-1}(t_{i,n})]^2},\ q_0(s) = log\frac{1-s}{1+s}.$$

The choice of value for n is not always clear, and experiments are useful to see the influence of choosing a different number of points for specific problems. In particular, we used $n = 5$ in our tests (the roots for P_5 are depicted in Table 1, and P_4 is defined by $P_4(x) = x^4 - \frac{6}{7}x^2 + \frac{3}{35}$). The remaining integrals in (2.5) were approximated by means of Gaussian quadrature, formula (3.8), using the same nodes t_i as in Table 1:

$$\int_{-1}^{1} g(t)dt = \sum_{i=1}^{n} \omega_i g(t_i). \tag{3.8}$$

Values of $t_{i,5}$	Weights ω_i
0.0	0.5688889
$\pm$ 0.53846931	0.47862867
$\pm$ 0.90617985	0.23692689

TABLE 1. The roots of the Legendre polynomial of order 5.

Choosing the collocation points and the quadrature nodes in this way ensures that all computations are valid. The number of collocation points is set at the beginning of a run. After several tests with different values, we decided to use a number of 16 points for the experiments. Evidently, this parameter may be subject to further tuning too, and we expect to obtain better results with a larger number of collocation and quadrature points, respectively.

The set of fitness cases for the symbolic regression formulation of the problem consists in two sets of pairs, consisting in a collocation point and the value of the left hand side $\Re f$, respectively $\Im f$ in that point. This way, we solve a complex symbolic regression problem ([11], [2]) on the data set obtained, with the objective of minimizing the error expressed in formula (3.6).

3.3. **Algorithm Settings.** The settings for the parameters of the algorithm are as recommended in the literature [6]. Fine tuning of these parameters in order to improve the behavior of the algorithm is possible, but such a study is beyond the purpose of this paper.

3.4. **Experimental Results.** Evolutionary algorithms are of probabilistic nature. Therefore, for purposes of statistical validity, the results in this paper are reported after a number of 50 independent runs, of 300 generations each. The individuals consisted of 6 genes. The solution of a run is designated the best individual in all generations. The raw error of the solution presented in Figure 2, computed according to formula (3.6), is 4.225148. The symbolic form of the solution is very complex, as expected. Table 2 presents it in edited form, where simplifications through manual editing were performed when possible, and numerical constants were truncated to 3 decimal places. All computations have been performed using double precision arithmetic (sixteen-digit accuracy).

In 72% of the runs, the solution depicted by GEP had error values in the range $[3.5, 5.5]$. In almost 10% of the runs, the algorithm prematurely converges to a solution of very poor fitness that becomes dominant in the population.

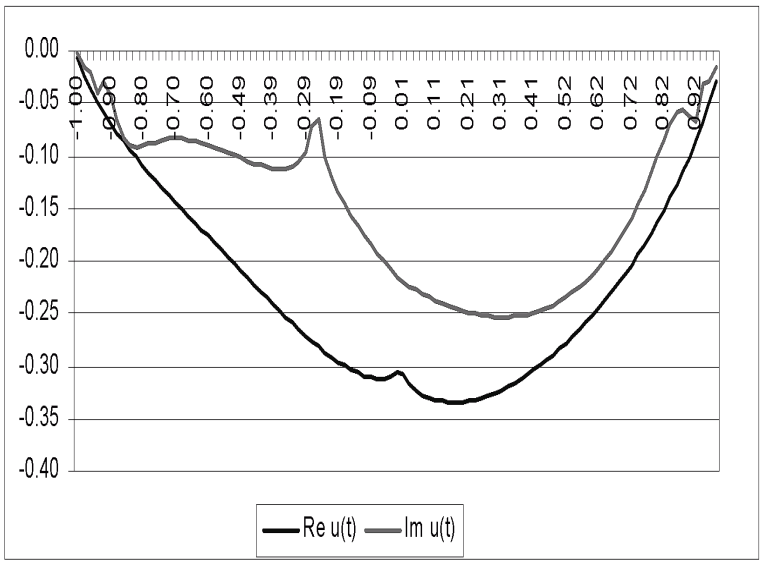


Figure 2. The solution obtained by GEP in 50 runs.

$\Re v_c(t)$	$(0.8910 - cos(0.4692 * t))*$ $(rlog(0.3062\%rlog(t))*$ $(sin(0.8814 + 2 * cos(sin(t))))+$ $sin(t + 0.2601) + 2.2601 + cos(sin(sin(t)))$
$\Im v_c(t)$	$sin(sin(0.8910 - cos(0.4692 * t)))*$ $(rlog(t + 0.2601) * 0.5695 + t\%(sin(t)) + 3)*$ $cos(sin(0.3062\%rlog(t) + t - 0.3684))*$ $cos(sin(cos(t + 0.2601)))$

TABLE 2. The symbolic form of the solution encoded by the best individual c.

4. CONCLUSIONS

The expressions evolved by GEP get quite complex, and it is often extremely difficult to understand them, even with symbolic post processing with tools like Mathematica (as suggested in [10]). The solution presented above is for exemplification purposes only.

The evolutionary approach provides multiple diverse analytical models without the need to enforce any supplementary assumptions on the problem. They may be used as alternatives to known solutions from other approaches. By minimizing the error in (3.6), GEP created high quality models that fit well the given input data. It can be noted that the solution provided here is different from the solutions provided in [8] and [15]. A possible explanation for this kind of behavior is the fact that the problem we study is an inverse problem and since different causes may produce the same effects, the solution of the problem may be not unique.

Other than statistical testing of the performance of the GEP algorithm, limited convergence and stability results are provided in the GP literature [12], mainly due to its stochastic nature. Different numerical integration methods for the approximation of the integrals in (2.5), and other fitness measures will be investigated. Comparisons with other approaches shall be performed in order to study further the validity and usefulness of the results.

REFERENCES

[1] Baals, D. D., Corliss, W. R., *Wind Tunnels Of Nasa*, electronic edition available online at http://www.hq.nasa.gov/office/pao/History/SP-440/cover.htm

[2] Băutu, E., Bąutu, A., Luchian, H., *Symbolic Regression on Noisy Data with Genetic and Gene Expression Programming*, Proceedings of SYNASC 2005, IEEE Computer Society Press, Los Alamitos, CA, USA, ISBN 0-7695-2453-2

[3] Băutu, E., Bąutu, A., Luchian, H., *A GEP-Based Approach For Solving Fredholm First Kind Integral Equations*, Proceedings of SYNASC 2005, IEEE Computer Society Press, Los Alamitos, CA, USA, ISBN 0-7695-2453-2

[4] Băutu, E., Pelican, E., *Numerical Solution For Fredholm First Kind Integral Equations Occurring In Synthesis of Electromagnetic Fields*, Rom. Journ. Phys., Vol. 52, Nos. 3-4, P. 225-235, Bucharest, 2007

[5] Bland, S. R., *The Two-Dimensional Oscillating Airfoil in a Wind Tunnel in Subsonic Flow*, SIAM J. Numer. Anal., 18 (1970), 830-848

[6] Ferreira, C., Gene Expression Programming: A New Adaptive Algorithm for Solving Problems, Complex Systems, 2001

[7] Fromme, J., Goldberg, M. A., *Numerical Solution of a Class of Integral Equations Arising in Two-Dimensional Aerodynamics*, Journal of Optimization Theory and Applications, Vol. 24, No. 1, jan. 1978

[8] Goldberg, M. A., *Introduction To The Numerical Solution Of Cauchy Singular Integral Equations*, Numerical Solution of Equations, Plenum Press, New York, 1990, 183-308
[9] Hunter, D. B., *Some Gauss-Type Formulae For The Evaluation Of Cauchy Principal Values Of Integrals*, Numer. Math. 19 (1972), 419–424, Math. Sci. Net.
[10] Kirshenbaum, E., Suermondt, H. J., *Using Genetic Programming to Obtain a Closed-Form Approximation to a Recursive Function*, GECCO 2004, 2004, pp. 543-556
[11] Koza, J. R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, 1992
[12] Langdon, W. B., Poli, R, *Foundations of Genetic Programming*, Springer-Verlag, 2002
[13] Mastroianni, G., Themistoclakis, W. 2005, *A Numerical Method For The Generalized Airfoil Equation Based On The De La Valle Poussin Interpolation*,. J. Comput. Appl. Math. 180, 1 (Aug. 2005), 71-105
[14] Monegato, G., *On The Weights Of Certain Quadratures For The Numerical Evaluation Of Cauchy Principal Value Integrals And Their Derivatives*, Numer. Math. 50, 1987, pp. 273-281
[15] Monegato, G., Sloan, I. H., *Numerical Solution of the Generalized Airfoil Equation for an Airfoil with a Flap*, SIAM J. Numer. Anal. 34, 6 (Dec. 1997), 2288-2305
[16] Paton, R. C., Nwana, H. S., Shave, M.J.R., and Bench-Capon, T.J.M., *An Examination Of Some Metaphorical Contexts For Biologically Motivated Computing*, British Journal for the Philosophy of Science, Vol. 45, 1994, 505-525
[17] Scuderi, L., *A Collocation Method for the Generalized Airfoil Equation for an Airfoil with a Flap*, SIAM J. Numer. Anal. 35, 5 (Oct. 1998), 1725-1739

UNIVERSITY OVIDIUS
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
BD. MAMAIA 124, 900527 CONSTANŢA, ROMANIA
E-mail address: ebautu@univ-ovidius.ro
E-mail address: epelican@univ-ovidius.ro