# GENERAL METHOD OF MULTIVARIATE NON-LINEAR REGRESSION BASED ON GENETIC PROGRAMMING

ORTIZ TRIVIÑO, JORGE EDUARDO PhD (c), Associate Professor, jeortizt@unal.edu.co Universidad Nacional de Colombia.
FLOREZ CALDERÓN, MAURO PhD, Associate Professor, mflorezc@unal.edu.co Universidad Nacional de Colombia

***Abstract*-In this paper we show the *method* $\psi$ to estimate the structure and parameters of a non-lienar function of real value $f(\cdot;\theta):\mathbb{R}^n \to \mathbb{R}$ from a data set $\left\{\left(x_1, x_2, \cdots, x_j, \cdots, x_n; y\right)_i\right\}_{i=1}^{i=m}$ taken from that function. The technique $\psi$ is based in the *Holland's genetic algorithm*, this employ the genetic operations of *selection*, *crossover* and *mutation*. But unlike this, the individuals are dynamic structures called trees, allowing its size can grow without restrictions and these ones can become a better representation of the desired response. The experimentation shows that the method is efficient for both linear and nonlinear functions as well as for multivaried cases.**

***Index Terms*-Model, non linear function, genetic programming, structure.**

## I. INTRODUCTION

In human affairs, especially in professional and scientific, are managed, generated and obtained numerical data, usually tabulated and organized into records (rows) which in turn are composed of attributes (columns), one of them can be expressed in terms of others, possibly with a sense or maybe without him. In this study I suppose unknown both the function $f$ as the parameters. Table 1 shows graphically the organization of the data. Here is assumed that the attributes $1$ to $n$ are independent of each other and the attribute $(n+1)-ith$ can be explained in terms of all others. This is $y_i = f\left(x_{i1}, x_{i2}, ..., x_{ij}, ,, , x_{in}; \theta\right)$. In these situations it is desirable to estimate the structure of the function $f$ approximating the value of its parameters based on the tabulated data so that $\hat{f}$ with $\hat{\theta}$ describes the behavior of the data. In other words, it seeks to find the relationships between the *independent variables* (first attributes) with the *dependent variable* (last attribute), namely, find a function $\hat{f}$ that describes its dynamics.

There are traditional methods to do this when $f$ is linear (Searle, 1997) or polynomial (Chapra, 2007), the problem is they are limited not only in numbers but also in solutions. Undoubtedly the most important practical method to do this work is the *linear regression* based in Mean Square Error, MSE (Searle, 1997). However, not always have results that can be related by a polynomial function, or results that might be covered by simple methods of linearization. There are also other practical methods of approximation of curves (Sexton R. S., 1998), which have the disadvantage of computational complexity.

| Record | *Attribute* 1 | *Attribute* 2 | ⋯ | *Attribute* *j* | ⋯ | *Attribute* *n* | *Attribute* *n*+1 |
|---|---|---|---|---|---|---|---|
| 1 | $x_{11}$ | $x_{12}$ | ⋯ | $x_{1j}$ | ⋯ | $x_{1n}$ | $y_1$ |
| 2 | $x_{21}$ | $x_{22}$ | ⋯ | $x_{2j}$ | ⋯ | $x_{2n}$ | $y_2$ |
| ⋮ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ |
| *i* | $x_{i1}$ | $x_{i2}$ | ⋯ | $x_{ij}$ | ⋯ | $x_{in}$ | $y_i$ |
| ⋮ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ |
| *m* | $x_{m1}$ | $x_{m2}$ | ⋯ | $x_{mj}$ | ⋯ | $x_{mn}$ | $y_m$ |

TABLE 1. TABULAR REPRESENTATION OF THE EXPERIMENTAL DATA.

Alternatively, also is possible apply heuristic methods [Rubenstein,1981] that approximate an answer to the desired result, examples of such methods include *random searches*, *Monte Carlo simulation* (Law, 2000) or the utilization of *bio-inspired procedures* (Sexton R. S., 1999), the latter is the technique used to development of the method proposed in this research.

This paper is structured as follows, beginning with the problem statement and model formulation, continues with genetic programming section, which presents a brief review of the state of the art of evolutionary strategies and considers the method to solve the problem, including the simulator implemented, then I show the experimentation and the results, which assesses the accuracy of the technique used, finally I present the conclusions.

## II. STATEMENTOF THE PROBLEM AND FORMULATION OF THE MODEL

### A. Statement of the Problem

Is required a procedure that receives as input a set of data, these can be supposed how information included in the unknown function $f:\mathbb{R}^n \to \mathbb{R}$, the procedure must

produce how results a good estimation of its structure and its parameters. Figure 1 summarizes the problem to solve. Formally speaking, is desired the estimation of the conditional expected value to describe the behavior of the values of a random variable in function of the knowledge of the other variables. In other words, is desired to find $\mu_{Y|X_1,X_2,\cdots,X_n} = f(x_1,x_2,\cdots,x_n) = E(Y|X_1=x_1,X_2=x_2,\cdots,X_n=x_n;\theta)$.

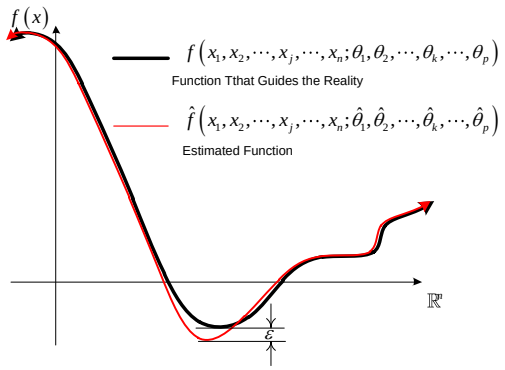


FIGURE 1. PROBLEM TO BE SOLVED.

### B. *Formulation of the Model* $\psi$

Being $f:\mathbb{R}^n \to \mathbb{R}$ a real function with space of parameters $\Theta \in \mathbb{R}^p$ such that $y = f(x_1,x_2,\cdots,x_j,\cdots,x_n;\theta_1,\theta_2,\cdots,\theta_p)$ and $\{(x_1,x_2,\cdots,x_j,\cdots,x_n;y)_i\}_{i=1}^{i=m}$ a given sample drawn from the population $f$. Both $f$ as the space of parameters $\Theta$ are assumed unknown. The procedure $\psi$, based on genetic programming (Goldberg, 1989), is proposed to estimate $f$ y $\Theta$, namely $\hat{f}(x_1,x_2,...,x_j,...,x_n;\hat{\theta}_1,\hat{\theta}_2,...,\hat{\theta}_p)$. Figure 2 shows a general block diagram of the proposed method.

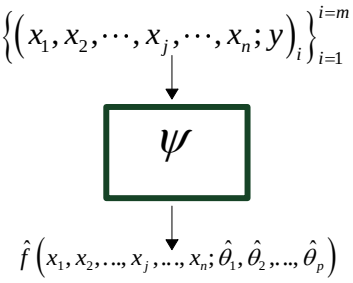


FIGURE 2. GENERAL SCHEME OF THE TECHNIQUE $\psi$.

The technique $\psi$ uses only the data, analyzes it and gives the estimation of $f$, $p$ and $\Theta$. This is done by the representation of the individuals as trees describing a function and its parameters. They correspond in genetic programming to the programs to evolve, being one of the main features of genetic programming that unlike genetic algorithms don't evolve a static data structure. Figure 3 shows a tree for the technique $\psi$.

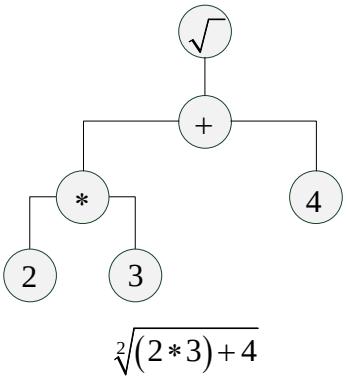


FIGURE 3. REPRESENTATION OF A TREE IN THE TECHNIQUE $\psi$.

To measure the quality of the estimation is used the *MSE*.

## III. GENETIC PROGRAMMING

Genetic programming (GP) is a technique for solving problems, in which apply an exhaustive search method is very expensive. Often these problems are combinatorial and the search space is too large. The Genetic programming seeks to eliminate these problems with the cost of dispense with the direct relationship between the parts of the genome and the parts of the solution (Goldberg, 1989). Instead it uses an indirect representation where the chains have a variable length and are interpreted as programs. The effect of each instruction is to alter the shape of the solution or change the state of the system responsible for interpreting to individuals.

Is possible to see some similarity between this encoding and functioning of the DNA transcription in a normal cell. While the transcript from the sequence of bases in the DNA to the sequence of aminoacids in proteins is relatively direct, the complexity and indirectness of the action of these to form an individual is similar to the way a program is interpreted in genetic programming (Ortiz, 2011).

A *genetic algorithm* (GA) is a stochastic procedure of search, inspired in the theory of *Darwinian evolution* and *Mendelian genetics*. Practical applications in various fields of human activity such as optimization (Chouza, 2008), have shown that its use generates usually good results, for

this reason their popularity and acceptance has been increasing over recent years.

In this paper is used an evolutionary strategy called genetic programming, which is of great importance and has proven to be very good for different applications. In general terms the genetic programming can be described as a computational technique that seeks to find optimal solutions to intractable problems in time, this technique uses an evolutionary approach to generate programs (trees) that resolve a certain task, significant difference compared with other techniques such as genetic algorithms; in this case the programs are the individuals to evolve different to the data rigid structures in GAs, usually these programs are represented by syntactic trees, with this is possible to express complex functions, defined by all possible combinations of symbols, from which the tree can be structured. For the construction of the trees is necessary the specification of the set of symbols that will be used as leaves, and the set of symbols that will be used as internal nodes, which often are the operations used on the set of symbols that are in the leaves. With this, the approach behaves in the same way that the evolutionary strategies in a general level, namely, are used the mechanisms of reproduction, mutation and crossover for the evolution of the trees, which represent solutions to the problem to solve.

To find a relationship between a dependent variable $y$ and $n$ independent variables $X_n$, through of genetic programming, is necessary to define the set of symbols that go on the leaves of the trees; for this work the set of symbols are the independent variables and the set that will be used for internal nodes is given by the arithmetic operators, algebraic functions and transcendental functions.

### A. *Individuals and Populations*

The fundamental idea that implements this technique is based on the selection mechanisms that nature uses (Chouza, 2008), where the fittest individuals in a population are those with a higher probability of survival, by means of a better adaptation to the changes that occur in the environment they inhabit.

According to some genetic results, we know that these changes are made in the genes of an individual (basic unit of encoding each attribute in an individual), and their desirable attributes (i.e., those that let a better adaptation to their environment) are transmitted to their descendants (children) when they reproduce sexually. Namely, are adaptive methods that can be used to solve search problems and optimization, among others. John Holland developed this idea computationally (Goldberg, 1989). His work has been the basis for this research in which the concept of simple genetic algorithm has been generalized to estimate the structure of nonlinear multivariate functions from a set of data available that follow that model.

Often when the evolutionary strategies are used to find the solution of a problem in which we have a very large space, the set of solutions is called population. The population is composed of many individuals, who are independent of each other; then the population is a community composed of individuals who provide independent solutions to the problem, through genetic information that is inherited to their descendants, that is to say, each individual is a possible solution to the problem and is composed of a set of genes which are generally called genotype. These genes represent characteristics and properties that uniquely define each individual in the population; then the genotype provides an interpretation of the individual with respect to the problem attacked, commonly called phenotype.

Firstly, the method uses the initialization procedure to generate the first population. Simulating the individuals in the population, they compete against each other to be used in the reproduction process. That is, each one of the members of the population is evaluated, and according to their "Fitness" is assigned a probability to be selected for reproduction. Speaking mathematically, this probability is used to select some individuals with the genetic operators, thereby are obtained new individuals which are often the most successful in each competition and will be able to produce more offspring.

The search of the solutions in the problem space is defined under a criterion of adaptation, this process must be very similar to the process of natural selection, where only survive the individuals that are better adapted to the environment, then under this principle, in the strategies evolutionary, the individuals who survive are those that represent the best solutions to the problem. These individuals go through a process of *reproduction* and *mutation*, where the reproduction is often to select two individuals and mix their genes, so that a new individual may be born with a better solution; in regard to *mutation*, this consist in alter mildly the genetic structure of an individual and see if this change is a better solution. Basically the *reproduction* and *mutation* are primarily aimed to increase randomly the search space, in order to find really good solutions.

### B. *Simple Genetic Algorithm*

The most important parts of the genetic algorithm are in the repeated execution of new *generations* and in the *evolution of the population* that makes up the same. For a new generation, the operations of *selection*, *crossover* and *mutation* are realized on individuals of the current population, so the iterations are possible and finally is selected the individual with better level of adaptation which keeps in its coding the best solution in those moment. The

Algorithm 1 shows the main structure of the procedure of *Holland*, while the Algorithm 2 presents the details of each iteration.

```
PROCEDURE SimpleGeneticAlgorithm(SizePop ∈ ℝ, Generations ∈ ℝ, ε ∈ ℝ)
  INITIATION
      gen ← 0;
      Initializevalues(·);
      REPEAT
            INITIATION
                gen ← gen + 1;
                OldGeneration ← NewGeneration;
      UNTIL(gen ≥ Generations) ∨ (Error ≥ ε)
  END _ PROCEDURE SimpleGeneticAlgorithm
```

ALGORITHM 1.GENERAL STRUCTURE OF HOLLAND GENETIC ALGORITHM.

```
PROCEDURE NewGeneration(·)
  INITIATION
      pop ← 0;
      WHILE(pop ≤ TamPop)
      INITIATION
          parent1 ← Seleccion(·);
          parent2 ← Seleccion(·);
          descendant1 ← CruceMutacion(parent1, parent2);
          descendant2 ← CruceMutacion(parent1, parent2);
          Population[pop] ← descendant1;
          Population[pop + 1] ← descendant2;
          Population ← pop + 2;
      END _ WHILE
END _ PROCEDURE NewGeneration
```

ALGORITHM 2. EVOLUTIVE PROCESS OF THE HOLLAND ALGORITHM.

It's clear that their execution, in search of a solution, is stochastic in time and somehow, in storage. However, its probabilistic structure $g_{\Omega}(\omega)$ depends of the particular problem. One of the most important features of this method is the *robustness* that shows, mainly because it can adapt to different environments and optimization problems. Moreover, for the appropriate performance of genetic algorithm are necessary minimum standards (Goldberg, 1989):

1. Its search space (possible solutions) should be bounded properly within a range that can be infinite.
2. It must be possible the definition (estimation) of an adaptive function, which must be capable to indicate how good or bad is a possible answer.
3. The solutions must be encoded in a way that the implementation will be relatively easy in a computer.

The representation of data is key in genetic algorithms, as these manage directly a coded representation of the feasible solutions to the problem. Although the representation of a problem using fixed-length strings can solve a wide variety of problems, has an important limitation. One of these is the direct relationship between the size of the strings used and the size of the feasible solution represented. Another is the relative rigidity of coding systems used in genetic algorithms (Chouza, 2008).

It can be seen that the description of the operation of genetic programming is very similar with the description of the operation of genetic algorithms. The essential difference lies in the evaluation process: while a genetic algorithm evaluates directly the genotype of the individual, the genetic programming execute the genotype as a program, being the result the phenotype to be evaluated. The exact boundary is not clear because the execution of a program could be considered how a sophisticated form of evaluation, but this is the distinctive feature of genetic programming. There are also minor differences given by the fact that programs are composed generally for complex structures of variable size. This forces the development of genetic operators relatively complex; otherwise, would have an excessive number of individuals unable to be implemented (Glover, 1998).

### *C. Implementation*

In GP the programs to evolve are represented as syntactic trees that are non-linear genetic material, unlike genetic algorithms that work with linear genetic material represented in binary strings of symbols.

To find a relationship between a dependent variable $y$ and $n$ independent variables $X_n$ are used tree for the representation of the functions; in this case our tree will be represented by variables and functions, specifically the leaves of our tree represent variables or coefficients of some term of the dependent vector and the internal nodes represent functions with arguments, i.e. , arithmetic functions, transcendent functions and algebraic functions.

Figure 4 shows the typical structure of one of the trees generated as a candidate for the solution of the problem.

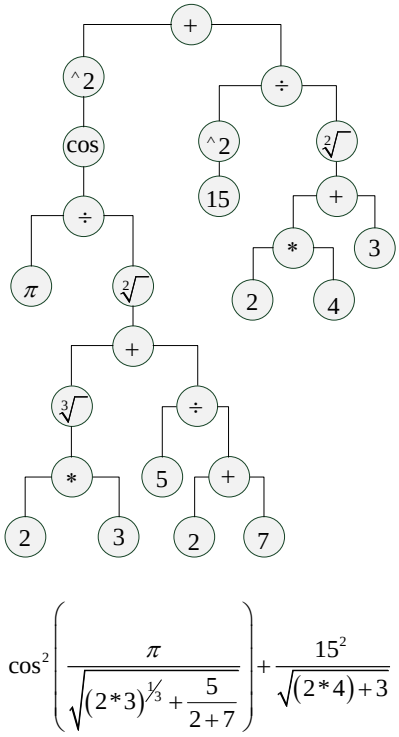


FIGURE 4. TREE: REPRESENTATION OF THE EXPRESSION.

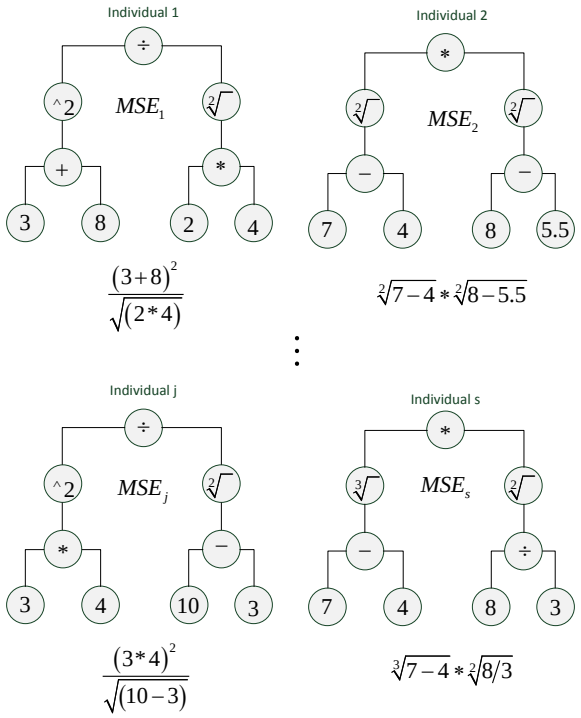


FIGURE 5. OPERATION OF THE SELECTION IN THE POPULATION (FOREST).

### *D. Basic Genetic Operations*

As mentioned above, the evolutive process in the *Holland* algorithm is based in genetic operators, i.e., subprocedures that receive as parameter the current population and transform it, in some way in search of better individuals.

#### *1) Operation Of Selection*

In this strategy the best individuals are evaluated by a square error measure, calculated between the input data and the data generated by the function, returned by each tree; the individual (tree) with the smaller error, is the closest to the desired solution. With this in mind, the best trees are selected to perform the operation of crossover. The adaptation of the individual adaptation is calculated using the concept of mean square error (MSE).Figure 5 shows the operation of selection in the population.

The selection operator is implemented to form a pair of parent trees $(z_1, z_2)$, these are chosen of the actual forest $Z$ in a random and independent form from the density function $h_Z(z) = P(Z = z) = p_z$ that represents the likelihood to select a tree $z$ of the forest $Z$ and whose calculation is performed so that each likelihood is proportional to the individual fitness, this means the selection likelihood of a tree is higher if his fitness is very good, specifically this is achieved by defining $p_z$ as

$$p_z = \frac{\sum_{i=1}^{s} MSE_i - MSE_z}{\sum_{j=1}^{s}\left[\sum_{i=1}^{s} MSE_i - MSE_j\right]}$$

. Finally, $z_i = H_Z^{-1}(u_i)$ where $u_i$ is a random number and $i = 1, 2$.

#### *2) Operation Of Crossover*

The operation of crossover is performed taking the half of a tree with the other half and mix both, providing a greater chance to find the desired function. Usually the crossover operator does not apply to all pairs of trees that have been selected for mating; the crossover operator is applied randomly with likelihood $P_c$, typically between 0.5 and 1.0. Figure 6 shows a pair of trees chosen to perform the

operation of crossover, while Figure 7 shows the result of the operation.

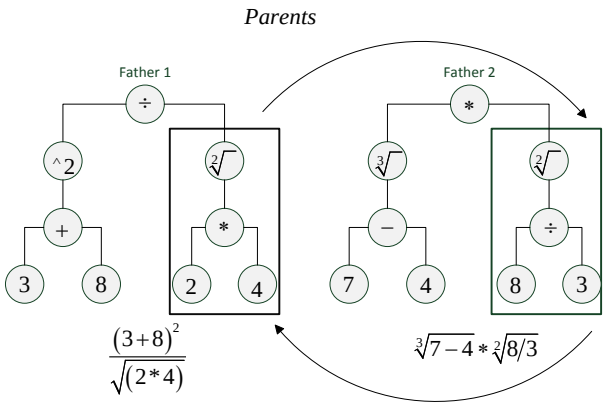


FIGURE 6. TREES (PARENTS) SELECTED FOR THE CROSSING.

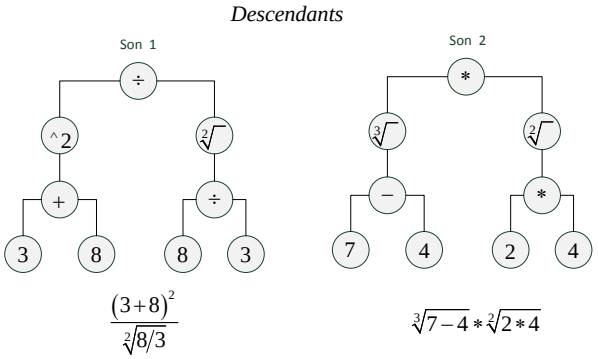


FIGURE 7. TREES (DESCENDANTS) RESULTING FROM THE CROSSING.

### 3) *Operation Of Mutation*

The operation of mutation is a type of probabilistic mutation, which defines two types of mutation and one of these is selected randomly; for the first type of mutation only changes a leaf node or an internal node, for the second type is changed a leaf node and an internal node. The mutation operator is applied to each descendant individually; typically the mutation has a probability $P_m$ between 0.01 and 0.2. Figure 8 shows the types of mutation used in the system $\psi$. In this case the mutation presented is based on the descendant tree obtained in the Figure 7.

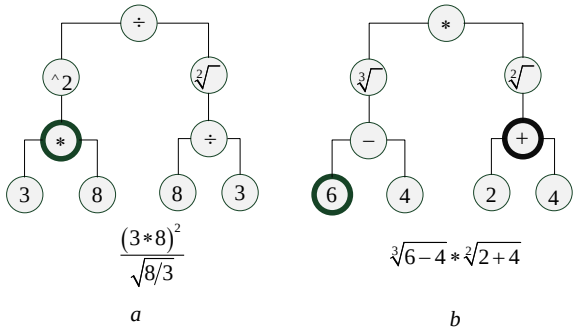


FIGURE 8. TYPES OF MUTATION: A) TYPE 1 B) TYPE 2.

It is necessary remember that the operation of mutation requires special care, because if the mutation probability is too high or it make abrupt changes, noise is generated and isn't possible an improvement to the solution.

If the GP has been properly implemented, the population will evolve over successive generations so that the half adaptation is extended to all individuals of the population, in the same way the adaptation of the best individual will continue to increase towards the global optimum.

### 4) *Convergence and Termination Conditions*

In the execution of the algorithm, first is generated a population (forest), in which the trees (individuals) are conceived randomly with an uniform distribution. This can generate a population that is close to the answer, so that in the subsequent generations the error is smaller or equal. After to calculate the initial population, the three operations (selection, crossover and mutation) are executed.

Thus, individuals are calculated for each new generation, these operations are done until the MSE is less than a value set by the user. In addition, there is a recalculation of the initial population when an individual has a less adaptation to a set value; this is just to ensure a convergence more quickly. The procedure can stop under two situations: when an individual is found and he has met the expectations of the user error or when the maximum number of generations established is reached.

## *E. Simulator Implemented*

The simulator implemented, based on genetic programming, receives the data table and in response generates a function that describes the behavior of the dependent variable $y$ in terms of the $n$ independent variables $(x_1, x_2, ..., x_n)$.

One of the attractions of the system is that it allows to adjust a function for a set of multivaried data. Therefore the

areas of application are several: engineering, mathematics, statistics, physics, economics, and in general all those disciplines that need to adjust the data set obtained in univariate or multivariate functions that represent phenomena and/or natural laws of different kinds.

The approximator of functions $\psi$ is an application developed in Java multiplatform programming language, for this reason is easily used in the most computing environments. The graphical interface is shown in Figure 9 and Figure 10.

Figure 9 shows the graphical interface (GUI), where the user enter the data set, select the number of independent variables and the functions that are desired for evolution. The data shown in Figure 9 correspond to initial experimentation: the convertion function of degrees Fahrenheit to degrees Celsius. As shown in the image, the estimated function has the form $y = f\left(x_1; \theta_1, \theta_2\right) = \theta_1 + \theta_2 x$ with $\hat{\theta}_1 = -18$ and $\hat{\theta}_2 = 0.558333333333333$. The function obtained by the tool has a good fit, which indicates a good measure of adaptation.

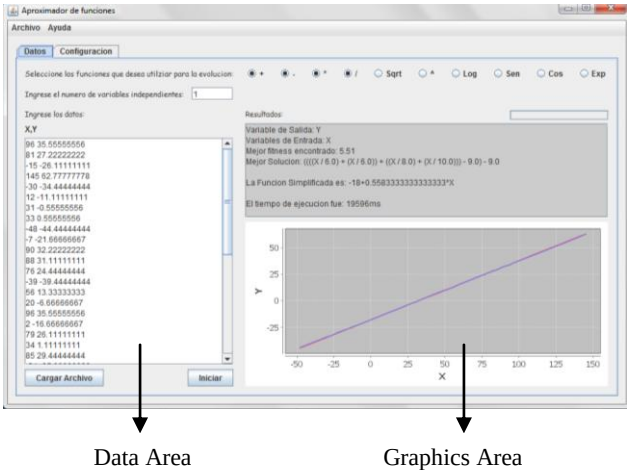


FIGURE 9. GRAPHICAL USER INTERFACE (GUI) OF THE FUNCTIONS APPROXIMATOR $\psi$ –DATA TAB.

Figure 10 shows the graphical user interface in the configuration tab, where is possible to set the parameters to find a better adaptation of the function. The approximator of functions $\psi$ has three default configurations. The configuration shown in Figure 10 is general, one of the two configurations with which the experiments were performed.

The effectiveness of the application will depend largely of the amount of input data, the complexity of the same in the configuration and the parameters set by the user at first. Once the data are supplied, is necessary configure the parameters of the genetic program and begins the process, every time the program find a new function with better measure of adaptation, fitness, the user is informed of this as well as the function that produced this fitness. In addition, to univariate functions, the software shows the graph of the function that best represents the input data for the current generation.

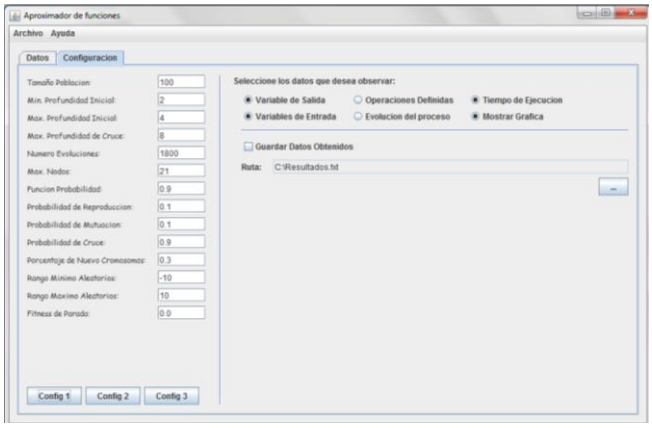

FIGURE 10. GRAPHICAL USER INTERFACE (GUI) OF THE FUNCTIONS APPROXIMATOR $\psi$ - CONFIGURATION TAB.

## IV. EXPERIMENTATION AND RESULTS

The experiments were based on chosen relatively complex functions of represent and set the parameters of the evolutionary strategy to maximize the similarity function between the input and the function that returns the algorithm.

### *GROWTH OF MOBILE CELLULAR SUSCRIBERS IN THE WORLD*

#### *A. Data*

The data were taken from the statistics of the ITU (International Telecommunication Union) of the number of mobile cellular subscribers per 100 inhabitants between 2000 and 2010; these are shown in Table 2.

| $y$ | $x_1$ |
|---|---|
| 12,0 | 0 |
| 15,5 | 1 |
| 18,4 | 2 |
| 22,2 | 3 |
| 27,3 | 4 |
| 33,9 | 5 |
| 41,8 | 6 |
| 50,6 | 7 |
| 59,9 | 8 |
| 68,3 | 9 |
| 70,0 | 10 |

TABLE 2. GLOBAL PENETRATION TASE PHONES (2000-2010).

*B. Method Of Configuration $\psi$*

The second configuration for this example is shown in Table 3 :

| Configuration Parameters 2 | |
|---|---|
| Population size | 2500 |
| Min Initial Depth | 3 |
| Max. Initial Depth | 6 |
| Max. Junction depth | 12 |
| Number of Generations | 3000 |
| Max. Nodes | 60 |
| Probability of reproduction | 0.1 |
| Probability of Mutation | 0.5 |
| Probability of Crossing | 0.9 |

TABLE 3. CONFIGURATION PARAMETERS FOR THE GROWTH OF MOBILE USAGE.

*C. Estimate*

When you run the ψ method with the general configuration is obtained a fitness of 6.48 that in this as take the data of the growth of mobile in the world which is a variable amount that is not associated directly to an equation is a value acceptable and in a time of 2053ms which is very fast to get a good estimate and threw function is of the form $y = f(x;\theta_1,\theta_2,\theta_3,\theta_4) \quad = \theta_1 + x\left(\theta_2 + \sqrt{\theta_3 + \theta_4 x}\right)$ $= 12 + x\left(-2 + \sqrt{14 + 6x}\right)$. By changing the settings for the specified configuration is achieved by reducing the fitness to $2.12$ with a significant increase in execution time $(30226)$ and the equation found in this configuration is as follows $y = (x;\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6,\theta_7)$

$$= \sqrt{\theta_1 - \theta_2 x + \theta_3 x\sqrt{\theta_4 - \theta_5 x + \theta_6 x^2 + \theta_7 x^3}}$$

$$= \sqrt{144 - 3x + 7x\sqrt{207 - 2x + 18x^2 + 9x^3}}$$

Figure 11 shows the structure of the genetic tree found for the approximator of functions $\psi$ in the current experiment.

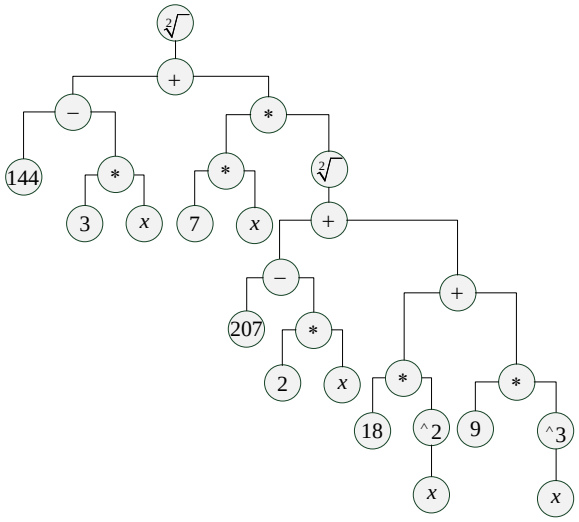


FIGURE 11. GENETIC TREE FOUND IN THE EXPERIMENT.

Figure 11 is only an example of the found tree (solution) in this case; in the next experiments it isn't show.

*D. Analysis*

Although the data try a phenomenon that is not controllable and hence it does not fit into a particular function is achieved through of this method a very close approximation to the real behavior of the growth of mobile cellular subscribers in the world in the last decade.

*LINEAR SYNTHETIC EXAMPLE*

*A. Data*

For this example the data was obtained from a multivariate function choosing the parameters randomly which had the form: $y = \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4$ with $\theta_1 = 3.8,\ \theta_2 = 7.2,\ \theta_3 = 15.9$ and $\theta_4 = -20.25$, the generation of data is also made randomly through the Excel tool.

*B. Method Of Configuration $\psi$*

As for all the examples we used two configurations, the first of these is the general configuration shown in Figure 10 and the second configuration is found by trial and error and experience in order to obtain a fitness close to zero indicating that the function found to adequately fit the data, this second configuration is shown in the Table 4.

| Configuration Parameters 2 | |
|---|---|
| Population size | 2500 |
| Min Initial Depth | 3 |
| Max. Initial Depth | 6 |
| Max. Junction depth | 12 |
| Number of Generations | 3000 |
| Max. Nodes | 60 |
| Probability of reproduction | 0.1 |
| Probability of Mutation | 0.5 |
| Probability of Crossing | 0.9 |

TABLE 4. CONFIGURATION PARAMETERS LINEAR SYNTHETIC EXAMPLE.

### C. *Estimate*

When you run the ψ method with the general configuration is obtained a fitness of 38.89 so you can understand why the general configuration is for estimating functions not very complex and therefore the result is obtained in a short run time (2383ms) and the function obtained is the follows $y = f\left(x_1, x_2, x_3; \theta_1, \theta_2, \theta_3, \theta_4, \theta_5,\right) = \theta_1 + \frac{\theta_2}{x_1} + \frac{\theta_3 x_1}{\theta_4 - \theta_5 x_1 x_2}$

$= 2 + \frac{16}{x_1} + \frac{7x_1}{8 - 0.1 x_1 x_2}$ ,Using the specific configuration for this example, perform better fitness $1.421085*10^{-14}$ that In practical terms it could be said to be zero in a runtime $311537\text{ms}$ which is greater than the general settings but you get a good approximation to the output function that is $y = f(x_1, x_2, x_3; \theta_1, \theta_2, \theta_3, \theta_4) = \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4$ with $\theta_1 = 3.8,\ \theta_2 = 7.2,\ \theta_3 = 15.9$ and $\theta_4 = -20.25$ with this is evidence that with different configurations is find better results depending on the problem to be treated.

### D. *Analysis*

When running the tool with the two configurations is observed that depending on the difficulty of the problem to obtain a good approximation by the tool must be configured properly the parameters and with this in some case the processing time is increased and according the complexity this time will be higher, in this specific case to the adjust the settings is get a good approximation that resembles to the real although it increases the execution time $(311537\text{ms})$ this time is not so great if you think about the similarity between the input and the function which gives the tool.

## *INTERNET USERS*

### A. *Data*

The data were taken from the statistics of the ITU of the number of Internet users per 100 habitants between 2000 and 2010 years; these are shown in Table 5.

| $y$ | $x_1$ |
|---|---|
| 12,0 | 0 |
| 15,5 | 1 |
| 18,4 | 2 |
| 22,2 | 3 |
| 27,3 | 4 |
| 33,9 | 5 |
| 41,8 | 6 |
| 50,6 | 7 |
| 59,9 | 8 |
| 68,3 | 9 |
| 70,0 | 10 |

TABLE 5. GLOBAL INTERNET USERS (2000-2010).

### B. *Method Of Configuration ψ*

The second configuration used for this example is shown in Table 6 :

| Configuration Parameters 2 | |
|---|---|
| Population size | 900 |
| Min Initial Depth | 4 |
| Max. Initial Depth | 8 |
| Max. Junction depth | 16 |
| Number of Generations | 2500 |
| Max. Nodes | 60 |
| Probability of reproduction | 0.3 |
| Probability of Mutation | 0.15 |
| Probability of Crossing | 0.9 |

TABLE 6. CONFIGURATION PARAMETERS FOR INTERNET USERS EXAMPLE.

### C. *Estimate*

When you run the ψ method with the general configuration is obtained a fitness of 2.33for this type of problems in which taken real data which are not associated with a particular function is very good and indicates that the resulting function resembles the shape of the inputs also are achieved these results in a very short time of 2924ms with the following function, $y = f(x; \theta_1, \theta_2, \theta_3, \theta_4)$

$= \theta_1 + \log\left(x^x + \sqrt{x^{\theta_2} + \theta_3 x^{\theta_4}}\right) = 7 + \log\left(x^x + \sqrt{x^8 + 2x^9}\right)\cdot$

Changing the configuration of the tool to get best results were achieved by reducing the fitness to 0.85 but increases the execution time 151089ms reaching the following function:

$$y = 9 + 2x - \cos\left(1.125 + \frac{0.125\sqrt{10\log(x)\sqrt{-8+x}}}{x} + 0.875x\right) - \sqrt{-10 + x - \sqrt{-9 + x - \cos\left(5x + x\sqrt{x}\right)}}$$

*D. Analysis*

Results from the tool for this example are good because the behavior of the function provided by the tool is very close to that of the data entered in both configurations, but as expected with the specific configuration provides better results but with an increase considerable execution time.You can do more experiments to get a better fitness but not significant reduction in relation to the execution time required to obtain this reduction.

*BELL GAUSS*

*A. Data*

The data used in this experiment were created using the Gaussian model that is of the form:

$$y = f(x_1, x_2; \theta_1, \theta_2, \theta_3, \theta_4, \theta_5) = \frac{1}{2\pi\theta_1\theta_2\sqrt{1-\theta_3^2}} e^{\left(-\frac{1}{2\left(1-\theta_3^2\right)}\left(\left(\frac{x-\theta_4}{\theta_1}\right)^2 - 2\theta_3\frac{x-\theta_4}{\theta_1}\frac{y-\theta_5}{\theta_2} + \left(\frac{y-\theta_5}{\theta_2}\right)^2\right)\right)}.$$

The data used in the tool as in previous examples were generated with Excel.

*B. Method Of Configuration $\psi$*

The configuration parameters for this specific case are shown in the Table 7.

| **Configuration Parameters 2** | |
|---|---|
| Population size | 100 |
| Min Initial Depth | 2 |
| Max. Initial Depth | 4 |
| Max. Junction depth | 10 |
| Number of Generations | 2500 |
| Max. Nodes | 21 |
| Probability of reproduction | 0.1 |
| Probability of Mutation | 0.4 |
| Probability of Crossing | 0.9 |

TABLE 7. CONFIGURATION PARAMETERS 2 FOR A GAUSSIAN FUNCTION.

*C. Estimate*

For this expression the exact parameters were used: $\theta_1 = 7$, $\theta_2 = 4, \theta_3 = 0.75, \theta_4 = -2$ and $\theta_5 = 3$ of the proposed model. This is a case in which the researcher knows exactly fluctuation ranges for different parameters.

The results with the general settings are good as you get a fitness of $0.01$ with a runtime of $2206ms$ and the resulting function is:

$$y = f(x_1, x_2; \theta_1, \theta_2, \theta_3, \theta_4) = \frac{\theta_1}{\theta_2 + \theta_3 e^{\theta_4 + \frac{x_1}{x_2}}} = \frac{9}{1099.07 + 0.135335e^{7.38906 + \frac{x_1}{x_2}}}.$$

With the specific configuration for this example yields a fitness of zero that is desired without a significant increase in execution time $(3183ms)$ and the simplified function obtained is: $f(x_1, x_2; \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) = \theta_1 e^{-\theta_2 - e^{x_1 - \frac{\theta_3 x_1}{\theta_4 - 1} - x_2 + \frac{x_2}{\theta_5 - x_1}} + \theta_6 e^{-x_2} x_2} = 64e^{-9 - e^{x_1 - \frac{2x_1}{9-x_1} - x_2 + \frac{x_2}{9-x_1}} + 0.1e^{-x_2} x_2}$

*D. Analysis*

Note that the first configuration fitness is $0.1$ close to zero, but due to changes in the configuration parameters yields a fitness of zero on the specific configuration for this example adjusting these parameters based on the experience and understand the behavior of this function.

*MULTIPLE LINEAL REGRESSION*

*A. Data*

The data presented in Table 8 were taken from (Searle, 1997), where estimates were calculated traditional multiple linear regression. In this table $x_1$ represents the values of age, while $x_2$ represents the level of schooling and is assumed that the income $y$ of a person is functionally dependent on these variables.

| $x_1$ | $x_2$ | $y$ |
|---|---|---|
| 6 | 28 | 10 |
| 12 | 40 | 20 |
| 10 | 32 | 17 |
| 8 | 36 | 12 |
| 9 | 34 | 11 |

TABLE 8. REVENUE PERFORMANCE BASED ON AGE AND EDUCATIONAL LEVEL.

*E. Method Of Configuration $\psi$*

The specific configuration that was used to obtain better results in this example is shown in the Table 9.

| Configuration Parameters 2 | |
|---|---|
| Population size | 2500 |
| Min Initial Depth | 2 |
| Max. Initial Depth | 6 |
| Max. Junction depth | 18 |
| Number of Generations | 5000 |
| Max. Nodes | 35 |
| Probability of reproduction | 0.2 |
| Probability of Mutation | 0.4 |
| Probability of Crossing | 0.9 |

TABLE 9. SPECIFIC CONFIGURATION PARAMETERS FOR MULTIPLE LINEAR REGRESSION.

### F. Estimate

To calculating the coefficient of linearity is concluded that the functional structure is linear nature [SEAR1971]. For this reason the function that approximates to this behavior is: $y = f\left(x_1, x_2; \theta_1, \theta_2, \theta_3\right) = \theta_1 x_1 + \theta_2 x_2 + \theta_3$ Applying the classical estimation methods for multiple linear regression, we arrive at $\theta_1 = 2.083333$, $\theta_2 = -0.208333$ and $\theta_3 = 2.333333$, for its part the execution of the $\psi$ method with the general parameter settings yielded the following function $y = f(x_1, x_2, \theta_1, \theta_2, \theta_3)$

$$= -\cos(x_1 - x_2) + \cos(x_2(\theta_1 - \theta_2 x_1) + \theta_3 \log\left[\left(e^{x_2}\right)^{x_1}\right]$$

$$= -\cos(x_1 - x_2) + \cos(x_2(5 - 0.5x_1) + \frac{1}{2}\log\left[\left(e^{x_2}\right)^{x_1}\right]$$

with a runtime of $1956ms$ and a fitness of $1.47$.

With the specific configuration is intended to reduce the fitness that is a little high to reach a close to zero indicating that the behavior of the function generated by the tool is closer to the shape shown by the data, with this configuration the results are the follows: the function is

$$y = f\left(x_1, x_2, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7\right)$$

$$= \sqrt{\theta_1}\sqrt{x_1} + x_1 + \cos\left[\sqrt{\left(e^{x_1} + \sqrt{\theta_2}\sqrt{e^{\theta_3 x_1}}\right)^{\theta_4}} + x_1\right] - \sin\left[x_1\right]$$

$$-\sin\left[\theta_5 x_1\right] - \sin\left[\sin\left[\sin\left[\theta_6 x_1 - \sin\left[\sin\left[(\theta_7 + x_1)^{x_1}\right]\right]\right]\right]\right]$$

$$= \sqrt{2}\sqrt{x_1} + x_1 + \cos\left[\sqrt{\left(e^{x_1} + \sqrt{2}\sqrt{e^{2x_1}}\right)^{1/4}} + x_1\right]$$

$$-\sin\left[x_1\right] - \sin\left[3x_1\right] - \sin\left[\sin\left[\sin\left[3x_1 - \sin\left[\sin\left[(-14 + x_1)^{x_1}\right]\right]\right]\right]\right]$$

with a runtime of $331242ms$ and with a fitness of $0.04$.

### G. Analysis

Although the example is presented as a linear regression, in this case the adjustment of the function to the data is not completely linear because the all the data are not included in a plane and the tool does not have the assumption of linearity that is have SEARLE in the book, but the shape of the function is very close to the data.

*EXPERIMENTS CONCLUSIONS*

On the basis of the initial configuration of parameters, the fitness found are goods so to the general configuration as to the configuration specified in each example, the function approaches to the theoretical basis of the data, as was expected when used the second configuration, the overall execution time increases but in the most examples is improved the fitness and in many cases reaching a fitness of zero is ideal.

The most notable changes for improvements were: The increase of the initial population of individuals, the increased number of generations, increasing the chances of mutation and reproduction and the depth of nodes in the tree, could be implemented as extensive configuration as the user wishes, taking into account the capabilities of the machine, which is almost exact results can be expected, but this could even take days processing program.

Noting the results of experiment 4 that is a function composed of several functions and with which is reached to fitness very valid and verified in the results that the error is only 1 hundredth. This is where you can see the results of genetic programming and we can imagine how through of a series of evolutions is obtained results as optimal which with more time could be very close to perfection.

## V. CONCLUSIONS

Genetic programming can be a very good approach to solve these types of problems, the representation that technique has helped enough to find very good results. For best results you can use this technique in combination with machine learning techniques to restrict the search space and improve timing and accuracy of results.

The evolutionary strategies are closely linked to their configuration parameters, these parameters should be adjusted by experience depending on the type of problem being approaching, even as in this case has to consider the search space that is facing. Because of this the evolutionary strategies often do not converge to the best solution, although you can always ensure that the solution is very close to the best. We also realize that you can explore as much as time allows us, however is possible that the algorithm may find a very good solution in 1 second and not be able to overcome this solution, or there the possibility that the algorithm will last long time to find a good solution, this is due to the local minimums and maximums that we found in the search space.

Due to the use of square difference method, when the program check decimal data close to zero the fitness that can be found can be below zero, but because the decimal data is very close to zero the fitness too much would not rise for a function that this wrong according to the original but that its points happen very near those of entrance.

A good future work is to implement other measures to the fitness such as "generalized square minimums ", "the maximum verisimilitude", among others.

## REFERENCES


Chapra, S. C. (2007). *Métodos numéricos para ingenieros.* Barcelona. 5Ta Edición: MacGrawHill.

Chouza, M. (2008). *Técnicas de optimización para Programación Genética*. Madrid: Prentice Hall.

Glover, F. (1998). *A Template for Scatter Search and Path Relinking.* New York: Springer-Verlag.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Boston: Addison-Wesley.

Kelly, J. P. (1996). A Scatter Search-Based Learning Algorithm for Neural Networks Training. *Journal of Heuristics, vol.2, n.2* , 129-146.

Laguna, M. a. (2000). Neural Network Prediction in a System for Optimizing Simulations. *IEEE Transactions* , 50-57.

Law, A. M. (2000). *Simulation Modeling and Analysis.* Arizona: McGraw-Hill.

Ortiz, J. (2011). Una técnica de regresión no lineal multivariada basada en algoritmos genéticos. *Revista Colombiana de Estadística.* , 89-101.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo method.* Chicago: Wiley Series in Probability and Statistics-Interscience.

Searle, S. R. (1997). *LinearMmodels.* New York: Wiley Classics Library.

Sexton, R. S. (1998). Global Optimization for Artificial Neural Networks: A Tabu search Application. *European Journal of Operational Research, vol. 106* , 570-584.

Sexton, R. S. (1999). Optimization of Neural Networks: A Comparative Analysis of the Genetic Algorithm and Simulated Annealing. *European Journal of Operational Research, vol. 114* , 589-601.

Werbos, P. (1974). *Beyond Regression: New tools for prediction and analisys in the behavioral sciences.* Cambridge. Harvard.