12

Variants of Hybrid Genetic Algorithms for Optimizing Likelihood ARMA Model Function and Many of Problems

Basad Ali Hussain Al-Sarray and Rawa'a Dawoud Al-Dabbagh
Baghdad University (Computer Science Dept; Collage of Science)
Iraq

1. Introduction

Optimization is essentially the art, science and mathematics of choosing the best among a given set of finite or infinite alternatives. Though currently optimization is an interdisciplinary subject cutting through the boundaries of mathematics, economics, engineering, natural sciences, and many other fields of human Endeavour it had its root in antiquity. In modern day language the problem mathematically is as follows - Among all closed curves of a given length find the one that closes maximum area. This is called the Isoperimetric problem. This problem is now mentioned in a regular fashion in any course in the Calculus of Variations. However, most problems of antiquity came from geometry and since there were no general methods to solve such problems, each one of them was solved by very different approaches.

Generally, optimization algorithms can be divided in two basic classes: deterministic probability algorithm. Deterministic algorithm are most often used if a clear relation between the characteristic of possible solutions and their utility for a given problem exists. If the relation between a solution candidate and its fitness are not so obvious or too complicated, or the dimensionality of the search space is very high, it becomes harder to solve a problem deterministically. Trying it would possible result in exhaustive enumeration of the search space, which is not feasible even for relatively small problem.

Then, the probabilistic algorithm come in to play. The increased availability of computing power in past two decades has been used to develop new techniques of optimization Today's computational capacity and the widespread Availability of computers have enabled development of new generation of intelligent computing techniques, such as genetic algorithm.

Evolutionary Algorithm are population met heuristic optimization algorithms that use biologic- inspired mechanisms like mutation, crossover, natural selection, and survival of the fittest in order to refine a set of solution candidates iteratively [Weise, 2009].

All evolutionary algorithms proceed in principle according to the scheme illustrated in fig.(1).

A simple Genetic Algorithm sGA is search algorithms based on the mechanics of natural selection and neutral genetics. They combine survival of fittest among string structures with a structure yet randomized information exchange to form a search algorithm with some of

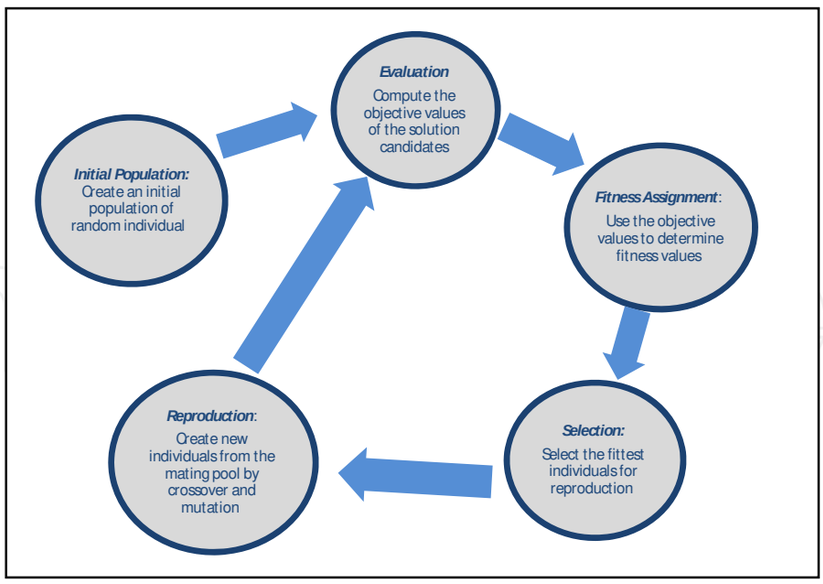


Fig. 1. Cyclic life of an evolutionary algorithms

the innovative flair of human search. In every generation; a new set of artificial creatures (string) is created using bits and pieces of the fittest of the old; an occasional new part is tried for good measure. They efficiently exploit historical information to speculate on a new search points with expected improved performance. A hybrid genetic algorithm (HGA) is the coupling of two processes: the simple GA and a local search algorithm. HGAs have been applied to a variety of problems indifferent fields, such as optical network design [Sinclair,2000], signal analysis [Sabatini, 2000], and graph problems [Magyar et al, 2000], among others. In these previous applications, the local search part of the algorithm was problem specific and was designed using trial-and-error experimentation without generalization or analysis of the characteristics of the algorithm with respect to convergence and reliability. The purpose of this study is to develop variants of hybrid simple genetic algorithm with local search algorithm represent by gradient or global algorithm present by evolution strategy to optimize solution of some functions where classifies as multimodal function and unimodel functions. One of import function of this study is likelihood function of time series autoregressive moving average $ARMA(1,1)$ model, this function defined as a unimodel function it is one of fundamental importance in estimation theory. The other functions used in this study as a test function used widely as benchmark functions. This study presents the($HGA1$) which is represent hybrid of simple genetic algorithm with an widely local search algorithm used steepest decent algorithm the other approach of hybrid denoted by $HGA2$ is coupling simple genetic algorithm with global search algorithm multimember evolution strategy, compares its performance with the simple(sGA), steepest descent algorithm(SDA), multimember evolution strategy ES; to study the behaviours many of functions classified as its kind multimodal or unimodel function which is used as test functions. The reminder of this chapter, section 2 presents definitions needed, section 3 giving a brief overview of genetic algorithms, representation of search points and their fitness evolution, selection, recombination, and mutation mechanisms. Then to be consistent, section 4 introduce the characteristic components of local search and its operators , also section 5 issue of the multimember evolution strategy. Section 6 address the issue of coupling simple genetic algorithm with multimember evolution strategy. Section 7 is an

extension of the results of section, in which are representative of the classes of unimodel , and multimodal function. In which competition is raised.

2. Definitions

Definition 2.1 (Objective Function) An objective function $f: \mathbb{X} \to \mathbb{Y}$ with $\mathbb{Y} \subseteq \mathbb{R}$ is a mathematical function which is subject to optimization.
The co-domain $\mathbb{Y}$ of an objective function as well as its range must be a subset of the real numbers $\mathbb{Y} \subseteq \mathbb{R}$. The domain $\mathbb{X}$ of f is called problem space and can represent any type of element like numbers, lists, construction plans, and so on. It is chosen according to the problem to be solved with the optimization process. Objective functions are not necessarily mere mathematical expressions, but can be complex algorithms that, for example , involve multiple simulations. Global optimization comprises all techniques that can be used to find the best element $x^* \in \mathbb{X}$ with respect to such criteria $f \in F$.
Definition 2.2 (local Maximum) A local maximum $\hat{x}_l \in \mathbb{X}$ of one (objective) function $f: \mathbb{X} \to \mathbb{R}$ *is a*n input element $f(\hat{x}_l) \geq f(x)$ for all x neighbouring $\hat{x}_l$. If $\mathbb{X} \in \mathbb{R}^n$, we can write:

$$\forall \hat{x}_l \exists \epsilon > 0: f(\hat{x}_l) \geq f(x) \forall x \in \mathbb{X}, |x - \hat{x}_l| < \epsilon.$$

Definition 2.3 (Local Optimum). A (local) minimum $\hat{x}_l \in \mathbb{X}$ of one (objective) function $f: \mathbb{X} \to \mathbb{R}$ is an input element with

$f(\hat{x}_l) \leq f(x)$ for all x neighbouring $\hat{x}_l$, $\forall \hat{x}_l \exists \epsilon > 0: f(\hat{x}_l) \leq f(x) \forall x \in \mathbb{X}, |x - \hat{x}_l| < \epsilon$.

Definition 2.4 (Local Optimum).A local optimum $x^* \in \mathbb{X}$ of one (objective) function $f: \mathbb{X} \to \mathbb{R}$ is either a local maximum or a local minimum.
Definition 2.5 (Global Maximum). A global maximum $\hat{x} \in x$ of one (objective) function $f: \mathbb{X} \to \mathbb{R}$ is an input element with

$$f(\hat{x}) \geq f(x) \forall x \in \mathbb{X}.$$

Definition 2.6 (Global Maximum). A global maximum $\hat{x} \in x$ of one (objective) function $f: \mathbb{X} \to \mathbb{R}$ is an input element with

$$f(\hat{x}) \leq f(x) \forall x \in \mathbb{X}.$$

Definition 2.7 (Local Optimum): A global optimum $x^* \in \mathbb{X}$ of one (objective) function $f: \mathbb{X} \to \mathbb{R}$ is either a global maximum or a global minimum. Even a one-dimension function $f: \mathbb{X} = \mathbb{R} \to \mathbb{R}$ may have more than one global maximum, multiple global minimum, or even both in its domain $\mathbb{X}$. Take the sine or cosine function for example; for cosine function it has global maximum $\hat{x}_i = 2i\pi, (i = 0,1,2, \dots)$ and global minimum $\hat{x}_i = (2i + 1)\pi, (i = 1,2, \dots)$.
Definition 2.8 (Solution Candidate): A solution candidate x is an element of the problem space $\mathbb{X}$
Definition 2.9 (Solution Space): we call the union of all solutions of an optimization problem its solution space $\mathbb{S}$. $\mathcal{X}^* \subseteq \mathbb{S} \subseteq \mathbb{X}$
This solution space contain (and can be equal to) the global optimal set $\mathcal{X}^*$. There may exist valid solution $x \in \mathbb{S}$ which are not elements of $\mathcal{X}^*$, especially in the context of constraint optimization.

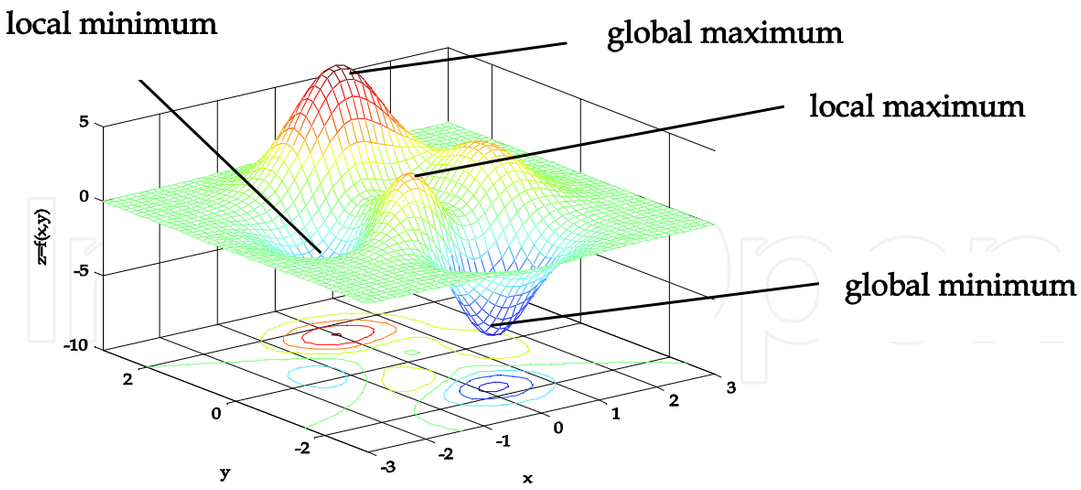


Fig. 2. An example of function with multi global and local maximum and minimum optimal point.

Definition 2.10 (Search space) : The search space $\mathbb{G}$ of an optimization problem is the set of all elements g which can be processed by the search operations. The type of the solution candidates depends on the problem to be solved. Since there are many different applications for optimization, there are many different forms of problem spaces. It would be cumbersome to develop search operations time and again for each new problem space encounter.

Definition 2.11 (Genotype): the elements $g \in \mathbb{G}$ of the search space $\mathbb{G}$ of a given optimization problem are called the genotypes.

The elements of the search space rarely are unconstraint aggregations. Instead, they often consist of distinguishable parts, hierarchical units, or well-type data strictures. The same goes for DNA in biology. It consists of genes, segments of nucleic acid, that contain the information necessary to produce RNA strings in a controlled manner. A fish, for instance, may have a gene for the colour of its scales. This gene, in turn, could have two possible "values" called alleles, determining whether the scales will be brown or grey. The genetic algorithm community has adopted this notation long ago and we can use it for arbitrary search space.

Definition 2.12 (Gene). The distinguishable units of information in a genotype that encode the phonotypical properties are called gene.

Definition 2.13 (Allele): An allele is a value of specific gene.

Definition 2. 14 (Locus): The locus is the position where a specific gene can be found in a genotype.

Definition 2.15 (Search Operation): the search operation search OP are used by optimization algorithm in order to explore the search space $\mathbb{G}$.

Definition 2.16 (individual): An individual p is a tuple $(p.g, p.x)$of an element $p.g$ in the search space $\mathbb{G}$ and the corresponding element $p.x = gpmp.g$ in the problem space $\mathbb{X}$.

Definition 2.17 (Population): A population (pop) is a list of individuals used during an optimization process.

$$Pop \subseteq \mathbb{G} \times \mathbb{X}: \forall p = (p.g, p.x) \in Pop \Rightarrow p.x = gpm(p.g)$$

As already mention, the fitness $v(x)$ of an element $x$in the problem space $\mathbb{X}$ often not solely depends on the element itself. Normally, it is rather a relative measure putting the features of x in to the context of a set of solution candidates x.

2.1 Genotype-phenotype mapping

The genotype –phenotype mapping (GPM, or ontogeny mapping) $gpm: \mathbb{G} \to \mathbb{X}$ is a left-total binary relation which maps the elements of the search space $\mathbb{G}$ to elements in the problem space

$$\mathbb{X}; \forall g \in \mathbb{G}\ \exists x \in \mathbb{X} : \ gpm(g) = x$$

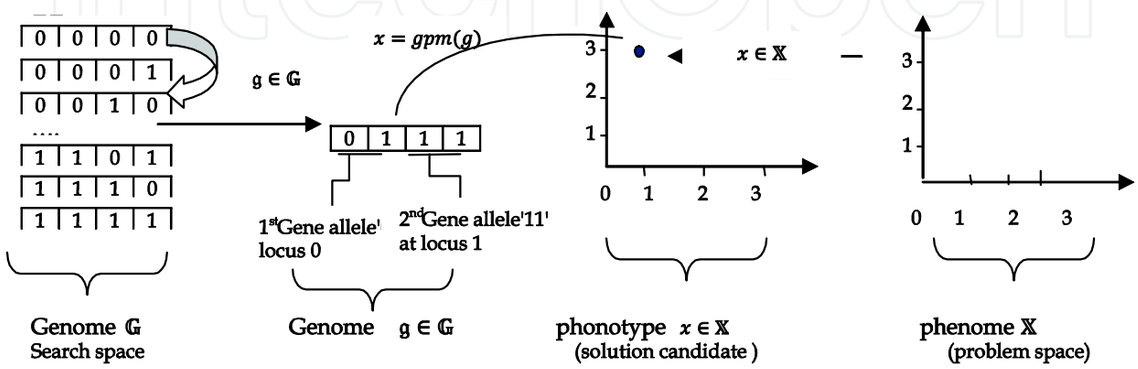


Fig. 3. The relation of genome, genes, and the problem space.

3. Genetic algorithm

3.1.1 Initialization

The first step is the creation of an initial population of solutions, or chromosomes. The populations of chromosomes generally chosen at random, for example, by flicking a coin or by letting a computer generate random numbers. There are no hard rules for determining the size of the population. Larger populations guarantee greater diversity and may produce more robust solutions, but use more computer resources. The initial population must span a wide range of variable settings, with a high degree of diversity among solutions in order for later steps to work effectively.

3.1.2 Fitness evaluation

In the next step, the fitness of the population's individuals evaluated. In biology, natural collection means that chromosomes that are more fit tend to produce more offspring than do those that are not as fit. Similarly, the goal of the genetic algorithm is to find the individual representing a particular solution to the problem, which maximizes the objective function, so its fitness is the value of the objective function for a chromosome. Genetic algorithms can of course also solve minimization problems. The fitness function (also called objective function or evaluation function) used to map the individual's chromosomes or bit strings into a positive number, the individual's fitness. The genotype, the individual's bit string, has to be decoded for this purpose into the phenotype, which is the solution alternative. Once the genotype has been decoded, the calculation of the fitness is simple: we use the fitness function to calculate the phenotype's parameter values into a positive number, the fitness. The fitness function plays the role of the natural environment, rating solutions in terms of their fitness. To apply the GA to real – valued parameters optimization problems of the form $f: \prod[u_i, v_i] \to R(u_i <$

v_i), the bit string is logically divided in to n segments of (in most cases)equal length $l_x (l = nl_x)$ and each segment is interpreted as the binary code of the corresponding object variable $x_i \in [u_i, v_i]$. A segment decoding function $\Gamma^i: \{0,1\}^{l_x} \rightarrow [u_i, v_i]$ typically looks like

$$\Gamma^i(a_{i1}a_{i2} \ldots . a_{il_x}) = u_i + \frac{v_i - u_i}{2^{l_x - 1}} [\sum a_{ij} 2^{j-1}] \quad (1)$$

where $(a_{i1}a_{i2} \ldots . a_{il_x})$ denotes the ith segment of an individual $\vec{a} = \left(a_{i1}a_{i2} \ldots a_{nl_{xx}}\right) \epsilon I^{n.l_x} = I^l$. Associated with each individual is fitness value. This value is a numerical quantification of how good of solution to optimization problem the individual is .Individual with chromosomal strings. Representing better solution has higher fitness values, while lower fitness values attributed to those whose bit string represents inferior solution. Combining the segment-wise decoding function to individual – decoding function $\Gamma = \Gamma^1 \times \ldots . \times \Gamma^n$, fitness values are obtained by setting

$$\Phi(\vec{a}) = \delta(f\left(\Gamma(\vec{a})\right)) \quad (2)$$

where δ denotes a scaling function ensuring positive fitness values such that the best individual receives largest fitness.

3.1.3 Selection

In the third step, the genetic algorithm starts to reproduce. The individuals that are going to become parents of the next generation selected from the initial population of chromosomes. This parent generation is the "mating pool" for the subsequent generation, the offspring. Selection determines which individuals of the population will have all or some of their genetic material passed on to the next generation of individuals. The object of the selection method is to assign chromosomes with the largest fitness a higher probability of reproduction.

3.1.4 Tournament selection

The tournament selection method select μ times the best individual from a random subset β_k of size $|\beta_k| = \xi,\ 2 \leq \xi \prec \mu \ \ \forall k \in \{1, \ldots, \mu\}$ and transfers it to the mating pool (note hat there may appear duplicates). The best individual within each subset β_k selected according to the relation $\succ^k$ (read: better then). A formal definition of the tournament selection operator $S: I^\mu \rightarrow I^\mu$ follows (Schowefel &Bäck, 1997):
Let $\beta_k \subset p(t)\ \forall\ k\ \in (1, \ldots, \mu)$ be random subsets of $P(t)$ each of size$|\beta_k| = \xi.\ \forall\ k\ \in (1, \ldots, \mu)$ choose $a^k \in \beta_k$ such that $\forall \vec{b} \in \beta_k: \vec{a}_k \overset{k}{\succ} \vec{b}$ where

$$\vec{a}_k \overset{k}{\succ} \vec{b}: \Leftrightarrow \Phi(\vec{a}) > 0\ \wedge f\left(\Gamma(\vec{a})\right) > 0\ \leq f\left(\Gamma(\vec{b}_k)\right) \quad (3)$$

3.1.5 Genetic operators

3.1.5.1 Crossover

The primary exploration operator in genetic algorithms is crossover, a version of artificial mating. If two strings with high fitness values mated, exploring some combination of their genes may produce an offspring with even higher fitness. Crossover is a way of searching the range of possible existing solutions. There are many ways in which crossover can implemented, such as one point crossover, two-point crossover, n-point crossover, or uniform crossover. In the following, we will stay with the simplest form, Holland's one-point crossover technique. Single-point crossover is the simplest form, yet it is highly effective.

One point crossover, is often used in sGA, it work first randomly picking a point between 0 and l. The participating parent individuals $\vec{x} = (x_1, \ldots . x_l)$ and $\vec{y} = (y_1, \ldots . y_l)$ are then split at the point , followed by a swapping of the split halves to form two offspring individual $\vec{x}\ and\ \vec{y}$ as follows (Kargupta, 1995):

$$\vec{x} = (x_1, \ldots, x_{\chi-1}, x_\chi, y_{\chi+1}, \ldots, y_l)\vec{y} = (y_1, \ldots, y_{\chi-1}, y_\chi, x_{\chi+1}, \ldots, x_l \quad (4)$$

where $\chi\epsilon\{1, \ldots, l-1\}$ denotes a uniform random variable .

3.1.5.2 Mutation

If crossover is the main operator of genetic algorithms that efficiently searches the solution space, then mutation could called the "background operator" that reintroduces lost alleles into the population. Mutation occasionally injects a random alteration for one of the genes. Similar to mutation in nature, this function preserves diversity in the population. It provides innovation, possibly leading to exploration of a region of better solutions. Mutation performed with low probability. Applied in conjunction with selection and crossover, mutation not only leads to an efficient search of the solution space but also provides an insurance against loss of needed diversity, on a single individual , mutation operator $m\{p_m\}: I \to I$ formally works as follows (Back & Schwefel, 1993): $m\{p_m\}(x_1, \ldots, x_l) = (\acute{x_1}, \ldots, \acute{x_l}), (\forall i \in \{1, \ldots, l\})$:

$$\acute{x_\iota} = \begin{bmatrix} \mathrm{x_i} & , \chi_\mathrm{i} > \mathrm{P_m} \\ 1 - \mathrm{x_i} & , \chi_\mathrm{i} \leq \mathrm{P_m} \end{bmatrix} \quad (5)$$

where $\chi_i\epsilon[0,1]$ is a uniform random variable, sampled anew for each bit.

3.1.6 Conceptual algorithm

The conceptual algorithm of sGA can then formulated as
t:=0; t is the generation number
Initialize $P(0) = \{\vec{a}_1(0), \ldots, \vec{a}_\mu(0)\}\epsilon I^\mu$ Where $I = \{0,1\}^l$
Evaluate $P(0) = \{\Phi(\vec{a}_1(0)), \ldots, \Phi(\vec{a}_\mu(0))\}\epsilon I^\mu$
Where $\Phi(\vec{a}_k(0) = \delta(f\left(\Gamma(\vec{a}_1(0))\right), P(0)$
While $(\tau((P(T)) \neq true$ do / / while termination criterion not fulfilled
Recombine: $\vec{a}_k(k) = r\{p_c\}(P(t))\quad \forall k \in \{1, .., \mu)$
Mutate: $\overrightarrow{a"}_k(t) = m\{p_m\}(\vec{a}(k) \quad \forall k \in \{1, .., \mu\}$
Evaluate $P"(t) = \left\{\overrightarrow{a"}_1(t), \ldots, \overrightarrow{a"}_\mu(t)\right\}:\{\Phi(\overrightarrow{a"}_1(t)), \ldots, \Phi(\overrightarrow{a"}_\mu(t))\}$
Where $\Phi\left(\overrightarrow{a"}_k(0)\right) = \delta\left(f\left(\Gamma\left(\overrightarrow{a"}_k(t)\right)\right), P(t-w)\right);$
end

Fig. 4. Pseudo code of sGA algorithm

3.2 Theorems and definitions needed

Definition 2.1

The directional derivatives of f(x, y) at the point (a, b) and in the direction of the unit vector $\mathrm{u} = \langle \mathrm{u_1}, \mathrm{u_2} \rangle$ is given by

$$\mathrm{D_u f(a,b)} = \lim_{\mathrm{h}\to 0} \frac{\mathrm{f(a+hu_1, b+hu_2) - f(a,b)}}{\mathrm{h}} \quad (6)$$

provided the limit exists.

Theorem 2.1

Suppose that f is differentiable at (a, b) and $u = \langle u_1, u_2 \rangle$ is any unit vector. Then we can write

$$D_u f(xa, b) = f_x(a, b) + f_y(a, b) \tag{7}$$

Clearly 2.1

For convenience, we define the gradient of a function to be vector –valued function whose component are the first –order partial derivatives of f . we denote the gradient of a function f by grad of f or ∇f read "del f" and define by the given theorem .

Theorem 2.2

If f is a differentiable function of x and y and u is any unit vector, then

$$D_u f(x, y) = \nabla f(x, y).\, u \tag{8}$$

Clearly 2.2

This theorem clear how to compute directional derivatives. Further, writing directional derivatives as a dot products. This theorem generalized to vector valued $F: R^{nx1} \to R^{nx1}$.

Theorem 2.3

Suppose that f is differentiable function of x and y at the point (a, b). Then

i. the maximum rate of change of f at (a, b) is $\|\nabla f(a, b)\|$ and occurs in the direction .

ii. of the gradient, $u = \frac{\nabla f(a,b)}{\|\nabla f(a,b)\|}$ the minimum rate of change of f at (a, b) is $-\|\nabla f(a, b)\|$ and occurs in the direction opposite the gradient $u = -\frac{\nabla f(a,b)}{\|\nabla f(a,b)\|}$.

iii. the gradient $\nabla f(a, b)$ is orthogonal to the level curve f(x,y)=c at the point (a ,b), where c=f(a, b).

Definition 2.2

We call f(a, b) a local maximum of f if there is an open disk R centred at (a, b), for which $f(a, b) \geq f(x, y)$ for $(x, y) \in R$. Similarly , f(a, b) is called a local minimum of f if there is an open disk cantered at (a, b), for which $f(a, b) \leq f(x, y)$ for $(x, y) \in R$. In either case f(a, b) is called a local extreme of f.

Theorem 2.4

suppose that f(x,y) has continuous second order partial derivatives in some open disk containing the point (a ,b) and that $f_x(a, b) = f_y(a, b) = 0$. Define the discriminant D for the point (a, b) by

$$D(a, b) = f_{xx}(a, b) f_{yy} - [f_{xy}(a, b)]^2 \tag{9}$$

if $D(a, b) > 0$ and $f_{xx}(a, b) > 0$, then f has a local minimum at (a, b).

(ii) if $D(a, b) > 0$ and $f_{xx}(a, b) < 0$, then f has a local maximum at (a, b).

(iii) if $D(a, b) < 0$, then f has a saddle point at (a, b).

(iv) if $D(a, b) = 0$, then no conclusion can be drawn.

4. Local search

The local search operator looks for the best solution starting at a previously selected point, in this case a solution in the sGA population. For this application, the steepest descent method was chosen as the local search operator. This method moves along the direction of the steepest gradient until an improved point found, from which a new local search starts.

The algorithm ends when no new relationship shown point can found (this is equivalent to a gradient equal to zero).
For functions with multiple local optimum, the method find one local optima but it is not guaranteed to find the global minimum. For geometric with conical shape, for example, the method finds the local optimum in one local search starting from any point in side the basin of attraction. For other geometries, the local search operator required more than one iteration to achieve the solution.

4.1 Descent method

Cauchy (1847), Kantorovich (1940-1945), Leven berg (1944), and Curry (1944) are the originators of the gradient strategy, which started life as a method of solving equations and systems of equations. It first referred to as aid to solving variation problems by Hadamard (1908) and Courant (1943). This variant of the basic strategy, known under the name optimum gradient method, or method of 'steepest descent'. Theoretical investigations of convergence and rate of convergence of the method can be found e.g. in Akaike (1960),Goldstein (1962), Ostrowski (1967), Zangwill(1969) and Wolfe(1969,1970,1971)[6] The general rule of steepest descent where used to find optimal solutions of nonlinear problems is

$$x^{(k+1)} = x^{(k)} + \alpha_k d^k \tag{10}$$

Where d^k is an a suitably chosen direction and α_k is a positive parameters (called step-size) that measures the step along the direction d^k. This direction is a descent direction if

$$d^{k^T} \nabla f(x^k) < 0 \qquad if \ \nabla f(x^k) \neq 0$$

$$d^k = 0 \qquad \text{if} \quad \nabla f(x^k) = 0 \tag{11}$$

4.1.1 Steepest descent algorithm

To approximate a solution p to the minimization problem $G(p) = \min_{x \in R^n} G(x)$
Given an initial approximation x:
Step 1. set $k = 1$
Step 2. While $(k \leq N)$ do steps (3-8)
Step 3. Set $g_1 = G(x_1, \ldots, x_n)$ / / note: $g_1 = G(x^k)$;
$z = \nabla G(x_1, \ldots, x_n)$ / / note: $z = \nabla G(x^{(k)})$;
$z_0 = ||z||_2$
Step 4. if $z_0 = 0$ then output ("zero Gradient") Output $(x_1, \ldots, x_n, g_1)$/ / [Procedure completed may have minimum check further]
Step 5. choose δ s.t
$g = \min (G(x_0 + \delta z)$
Step 6. Set $x = x + \delta z$
Step 7. if $|g - g_1| < Tol$ then
output $(x_1, \ldots, x_n, g_1)$/ / [Procedure completed successfully]
Stop
Step 8. set k=k+1;
Step 9. Output ('Minimum Iterations Exceeded') / / (Procedure completed unsuccessfully)
Stop

Fig. 5. Pseudo code of gradient algorithm

4.2 Hybrid genetic algorithm

In this section we define a hybrid of sGA with gradient method and we denoted as (HGA1) A hybrid genetic algorithm (HGA) is the coupling of two processes: the simple GA and a local search algorithm. The local search part of the algorithm was problem specific and designed using trial-and-error experimentation without generalization or analysis of the characteristics of the algorithm with respect to convergence and reliability. The HGA algorithm is a standard, which combines an sGA with local search. The local search step defined by three basic parameters: frequency of local search, probability of local search, and number of local search iterations. The first element for the definition of the algorithm is the frequency of local search, which is the switch between global and local search. In the HGA algorithm, this switch performed every "G!" global search generations, where "G!" is a constant number called the local search frequency. The second element of the algorithm is the probability of local search P, which is the probability that local search will be performed on each member of the sGA population in each generation where local search is invoked. This probability is constant and defined before the application of the algorithm. Finally, each time local search is performed, it is performed a constant number of local search iterations before local search is halted.

4.2.1 Basic elements

4.2.1.1 Genetic algorithm

Three basic operators define the simple Genetic Algorithm (sGA): binary tournament selection, single point crossover, elitism, and simple mutation. Through the successive application of these three operators, an initial population of solutions evolved into a highly fit population.

4.2.1.2 Local search

The local search operator looks for the best solution starting at a previously selected point, in this case a solution in the SGA population. For this application, the steepest descent method was chosen as the local search operator. This method moves along the direction of the steepest gradient until an improved point found, from which a new local search starts.
The algorithm ends when no new relationship shown point can found (this is equivalent to a gradient equal to zero) and this satisfied in our formula adaptation [10].

4.3 Hybrid genetic algorithm with local search algorithm

A hybrid genetic algorithm (HGA) is the coupling of two processes: the sGA and a local search algorithm. The local search part of the algorithm was problem specific and was designed using trial-and-error experimentation without generalization or analysis of the characteristics of the algorithm with respect to convergence and reliability. It is defined by three basic parameters: frequency of local search, probability of local search, and number of local search iterations. The first element for the definition of the algorithm is the frequency of local search, which is the switch between global and local search. In the HGA1 algorithm, this switch performed every "G!" global search generations, where "G!" is a constant number called the local search frequency. For example, if G!=3, local search would perform every 3 generations during the sGA. The second element of the algorithm is the probability of local search P, which is the probability that local search will be performed on each member of the sGA population in each generation where local search is invoked. This probability is constant and defined before the application of the algorithm. Finally, each time local search performed; it performed for a constant number of local search iterations before local search halted.

4.3.1 Conceptual algorithm of

The coupling approach in this paper consist in the introduction of generation interval for hybrid activation operator (HAO).Through selection, crossover, and mutation operators, the canonical sGA works on population of bit string encoding scheme generation by generation. When HAO is active (again implemented here every G? generation), the intermediate generation created by GA is fed into an adopted selection strategy which select subpopulation, usually of small size. Then each binary string individual in this subpopulation is convert into a real number vector, to be the initial value of steepest descent (sDA) algorithm that operate on this subpopulation for fixed small of generation. The vectors converted back into bit string values to manipulate again by master GA, sDA used here in this hybridization as a tool operates in small number of generation fashion in an order to enhance the selected points driven from the master GA. It is appropriate that the version of sDA tools to be of preservative survivor property to have worthy adjustment. The conceptual algorithm of the (HGA) given by the following steps:

Input: sample size, number of generation, ..
Output : approximate value
Step1: Initialize GA(generate Initial population of parameters).
Step2.1: for t = 1to number of generation Do
Step2.2: for I = 1: population size Do
The canonical genetic algorithm (sGA) operators:
Step2.2.1: Local Recombination,
Step2.2.2: Mutation,
Step2.2.3 Selection,
Step3: Binary coded sGA individual remapped in to real vector individual.
Step4.1: Select $\left(\frac{1}{3}\text{of population size}\right)$ use as initial solution of sDA perform for each generation
Step4.2. Start local search evaluation// Starting of Steepest Descent Algorithm (sDA).
Step4.3. : Real _coded local search algorithm individual remapped in to binary vector individual.
Step5. Repeat steps until all of generation complete or termination criterion is satisfied.
Step6: end

Fig. 6. Pseudo code of HGA algorithm

5. (μ^+, λ)-Evolution strategies

H.-P. Schwefel proposed the multi-member evolution strategies, the so-called (μ^+, λ)-ES. In their most general form, these strategies are described in the coming subsections.

5.1 Representation and fitness evaluation

An individual $\vec{a} = (\vec{x}, \vec{\sigma}) \epsilon I$ in (μ^+, λ)-ES can consist of the components (Robert, Roland, 2002):
$\vec{x} \epsilon R^n$: The vector of object variables.
$\vec{\sigma} \epsilon R_+^{n_\sigma}$: A vector of step length or standard deviations $(1 \leq n_\sigma \leq n)$ of the normal distribution. The strategy parameter $\vec{\sigma}$ (also called the internal model) determines the variances of the n-dimensional normal distribution, which is used for exploring the search space. The user of an evolution strategy, depending on his feeling about the degree of freedom required, can vary the amount of strategy parameters attached to an individual. As a rule of thumb, the global search reliability increases at the cost of computing time when

the number of strategy parameters is increased. The setting most commonly used which form the extreme cases are:

- $n_\sigma = 1$: (Uncorrelated mutation with one single standard deviation controlling mutation of all components of $\vec{x}$).
- $n_\sigma = n$: (Standard mutations with individual step sizes $\sigma_1, .., \sigma_n$ controlling mutation of the corresponding object variables x_i individually). The only part of $\vec{a}$ entering the objective function evaluation is $\vec{x}$, and the fitness of an individual $\phi(\vec{a})$ is identical to its objective function value $f(\vec{x})$, i.e. $(\phi(\vec{x}) = f(\vec{x}))$.

5.2 Mutation operator

The generalized structure of $(\boldsymbol{\mu}^+, \boldsymbol{\lambda})$-ES mutation operator consists of the addition of a normally distributed random number to each component of the object variable vector, corresponding to a step in the search space. The variance of the step-size distribution is itself subject to mutation as a strategy variable. Formally speaking, mutation operator $\boldsymbol{m}\{\boldsymbol{\tau_0}, \boldsymbol{\tau}\}: \boldsymbol{I} \to \boldsymbol{I}$, is defined as follows [5]

$$m\{\tau_0, \tau\}(\vec{a}) = m_x(\vec{x}) \circ m_\sigma(\vec{\sigma}) = (\vec{x}', \vec{\sigma}') \tag{13}$$

Which proceeds by first mutating the strategy parameters $\vec{\boldsymbol{\sigma}}$: $\boldsymbol{m_\sigma}: \boldsymbol{R_+^{n_\sigma}} \to \boldsymbol{R_+^{n_\sigma}}$;

$$m_\sigma(\vec{\sigma}) = \vec{\sigma}' = (\sigma_1 \exp(z_1 + z_0), \dots, \sigma_{n_\sigma} \exp(z_{n_\sigma} + z_0)) \tag{14}$$

Where $z_0 \sim N(0, \tau_0^2)$, $z_i \sim N(0, \tau^2)$ $\forall\, i \in \{1, \dots., n_\sigma\}$ To prevent standard deviations from becoming practically zero, a minimal value of ε_σ is algorithmically enforced for all σ_i.
Secondly, modifying $\vec{x}$ according to the new set of strategy parameters obtained from mutating $\vec{\sigma}: m_\chi: R^n \to R^n$

$$m_x(\vec{x}) = \overrightarrow{x'} = (x_1 + z_1, \dots, x_n + z_n) \tag{15}$$

5.3 Recombination operators

In (μ^+, λ)-ES, different recombination mechanisms are used in either local form, producing one new individual from two randomly selected parent individuals, or in global form, allowing components to be taken for new individual from potentially all individuals available in the parent population. Furthermore, recombination is performed on strategy parameters as well as the object variables, and the recombination type may be different for object variables, and standard deviations.
Depending on the recombination types [4][6]:

$$Rec = \begin{cases} 0 & \text{No recombination} \\ 1 & \text{Discrete recombination of pair of parents} \\ 2 & \text{Intermediate recombination of pair of parents} \\ 3 & \text{Discrete recombination of all parents} \\ 4 & \text{Intermediate recombination of parents in pairs} \end{cases} \tag{16}$$

Sometimes, the choice of a useful recombination operator for a particular optimization problem is relatively difficult and requires performing some experiments [2].The rules of recombination operator $r: I^\mu \to I$ for creating an individual, $r\{rec_x, rec_\sigma\}(P) = \overrightarrow{a'} =$

$(\overrightarrow{x'}, \overrightarrow{\sigma'}) \in I$, are given respectively by referring to arbitrary vectors $\vec{b}$ and $\overrightarrow{b'}$ where $\vec{b}$ and $\overrightarrow{b'}$denote here the part (i.e., either $\vec{x}$ or $\vec{\sigma}$) of a pre-selected parent individuals and the part of an offspring vector receptively. Each of $\vec{b}$ and $\overrightarrow{b'}$ are of length $m \in \{n, n_{\vartheta\sigma}\}, \forall i \in \{1, \dots, m\}$

$$b_i' = \begin{cases} b_i & if \quad rec = 0 \\ b_{\chi_{1,i}} \; or \; b_{\chi_{2,i}} & if \quad rec = 1 \\ (b_{\chi_{1,i}} + b_{\chi_{2,i}}).0.5 & if \quad rec = 2 \\ b_{\chi_{3,i}} & if \quad rec = 3 \\ (b_{\chi_{3,i}} + b_{\chi_{4,i}}).0.5 & if \quad rec = 4 \end{cases} \tag{17}$$

Where $\chi_1, \chi_2 \sim U(\{1, \dots, \mu\}$for each offspring, and $\chi_3, \chi_4 \sim U(\{1, \dots, \mu\}$ for each i.

5.4 Selection operator

There are two main classifications for selection according to the survival property of the parents [6]:

Extinctive_ (μ, λ) strategy; where parents live for a single generation only.$S: I^{\lambda} \to I^{\mu}$

$S(P) = P'$ where $|P| = \lambda$ & $|P'| = \mu$, & $\forall \overrightarrow{a'} \in P': \nexists\, \vec{a} \in P - P': f(\vec{x}) \leq f(\overrightarrow{x'})$

Preservative_$(\mu + \lambda)$ strategy; where selection operates on the joined set of parents and offspring, i.e., very fit individuals may survive indefinitely:

$S: I^{\mu+\lambda} \to I^{\mu}$

$S(P) = P'$ where $|P| = \mu + \lambda$, & $|P'| = \mu$, , and $\forall \overrightarrow{a'} \in P': \nexists\, \vec{a} \in P - P': f(\vec{x}) \leq f(\overrightarrow{x'})$

The ratio μ/λ is known as selection pressure. In the choice of μ and λ, there is no need to ensure that λ is exactly divisible by μ. The association of offspring to parents is made by a random selection of evenly distributed random integers from the range $[1, \mu]$. It is only necessary that λ exceeds μ by a sufficient margin that on average at least one offspring can be better than its parent. Hoffeister and Bäck in [7] have stated that $\mu/\lambda \approx \frac{1}{6}$ are tuned for a maximum rate of convergence, and as a result tend to reduce their genetic variability, i.e., the number of different alleles (specific parameter setting) in a population, as soon as they are attracted by some local optimum.

6. Cross- fertilization space of conical GAS and Standard Variant (μ^+, λ)-ES

The coupling approach followed in this section consists in the introduction of generation interval for hybrid activation operator(HAO). Through selection , crossover, and mutation operators, the simple GA works on population of bit string encoding scheme generation by generation. When HAO is active (implemented for every G? generation), the intermediate generation created by sGA is fed into adopted selection strategy which select ub population, usually of small size. Then each binary string individual in this subpopulation is converted in to a real number vector , to be the parents of the first generation of ES tool that operate on this subpopulation for a fixed small number of generations. The vectors are converted back in to bit string values to be manipulated a gain by the master GA. As ES is used here in this hybridization as a tool operator in a small number of generation fashion in an order to enhance the selected point driven from the master GA, then it is appropriate that the version of the ES too is to be of preservative survivor property to have worthy adjustment i.e., a (μ^+, λ)-ES is used.

t=0; {is the generation number}
ls=t_{max} {is the HAO age}
Initialize $P(0) = \{\vec{a}_1(0), \dots, \vec{a}_\mu(0)\} \epsilon I^\mu$ Where $I = \{0,1\}^l$
where $I \in \{0,1\}^l$;
Evaluate $P(0) = \{\Phi(\vec{a}_1(0)), \dots, \Phi(\vec{a}_\mu(0))\} \epsilon I^\mu$
where $\Phi(\overrightarrow{a_k}(0)) = \delta\left(f\left(\Gamma(\vec{a}_k(0))\right), P(0)\right)$;
while ($\tau(P(t)) \neq$ true) do {while termination criterion not fulfilled}
recombine: $\vec{a}''_k(t) = r\{p_c\}(P(t))$ $\forall k \in \{1, \dots, \mu\}$;
mutate: $\vec{a}''_k(t) = m\{p_m\}(\vec{a}_k(t))$ $\forall k \in \{1, \dots, \mu\}$;
evaluate $P''(t) = \{\vec{a}''_k(t), \dots \vec{a}''_\mu(t)\}: \{\Phi(\vec{a}''_k(t)), \dots \Phi(\vec{a}''_\mu(t))\}$:
where $\Phi(\overrightarrow{a}''_k(t)) = \delta(f(\Gamma(\overrightarrow{a''}_k(t))), (P(t-w))$;
if ($ls > 0$ *and* $HAW = true$)do {if HAO still live and active
Select :$P_{sub} - pop(t) = \{\vec{b}_1(t), \dots \vec{b}_{\mu 1}(t)\} \in I^\mu$
Where $I = R^{n+n_{sp}}$ and
$\vec{b}_k(t) = (x_i, \sigma_j \quad \forall i \in \{1 \dots n\} \forall j \in \{1, \dots n_\sigma\}$;
{binary _coded GA individual is remapped in to $(\mu + \lambda) - ES$}
{real vector individual}
For $t_{\mu+\lambda)-ES} = 1$ to $t_{max\ \_(\mu+\lambda)-ES}$ {do tmax sexual propagation
Recombine : $\overrightarrow{b''}_k(t) = r\{rec_x, rec_\sigma\}(P(t))$ $\forall k \in \{1, \dots, \lambda\}$;
Mutate: $\overrightarrow{b''}_k(t_{(\mu+\sigma)ES}) = m_{\{\tau_0,\tau\}}\left(\vec{b}\left(t_{\mu_1+\lambda_{1)_{ES}}}\right)\right)$ $\forall k \in \{1, \dots, \lambda\}$
Evaluate : $P''\left(t_{(\mu+\lambda)_{ES}}\right) = \left\{\overrightarrow{b''}_1(t_{(\mu 1+\lambda 1)} - ES), \dots, \left\{\overrightarrow{b''}_{\lambda 1}(t_{(\mu 1+\lambda 1)} - ES)\right\}\right.$;
$\left\{\Phi\left(\overrightarrow{b''}_1 t_{(\mu 1+\lambda 1)} - ES\right)\right), \dots, \Phi(\overrightarrow{b''}_1 t_{(\mu 1+\lambda 1)} - ES))\}$
Where $\Phi\left(\overrightarrow{b''}_1 t_{(\mu 1+\lambda 1)} - ES\right) = f\left(\overrightarrow{x''}_1 t_{(\mu 1+\lambda 1)} - ES\right)$;
Select: $P(t_{(\mu 1+\lambda 1)} - ES^{+1}) = S(P(t_{(\mu 1+\lambda 1)-ES^{+1}} \cup P''(t_{(\mu 1+\lambda 1)-ES^{+1}})$
End $(\mu 1 + \lambda 1) - ES$ generation loop
Evaluate
$P''(t) = \left(P''(t) - P_{sub-pop}(t)\right) \cup (P_{t_{max\ \_(\mu 1+\lambda 1)-ES}})$
$= \{\overrightarrow{a''}_1(t), \dots, \overrightarrow{a''}_\mu(t)\} \in I^\mu$
Where $I = \{0,1\}^l$
and $\Phi\left(\overrightarrow{a''}_1(t)\right) = \delta\left(f\left(\Gamma\left(\overrightarrow{a''}_1(t)\right)\right), P(t-w)\right)$; re-evaluate fitness
end
Select $P(t+1) = S(P'')) \epsilon I^\mu$
$ls = ls - 1$; when ls=0 then the loop is turned into pure GA phase
$t = t + 1$
end

Fig. 7. Conceptual algorithm of HGA2 (hybrid Genetic algorithm with multimember evolution strategy)

7.2 Test functions

In order to evaluate the behaviours of hybrid genetic algorithms, a set of test problem have been carefully selected to illustrate the performance of the algorithms and to indicate that it has been successful in practice. The nine test functions, which is classifies as multimodal or unimodel function; these function given with more details in section below.

7.3 Simulations

Multi- functions used as a test functions classified as unimodel and multi model it is deployed to verify the proposed hybrid genetic algorithms. The firs test function is likely hood function of ARMA(1,1) model, this function classifies as a unimodel the simulating experiment described in the following.

7.3.1 F_1: Test function / Likelihood function

The likelihood function is one of fundamental importance in estimation theory. This principle says that the data has to tell us about the parameters contained in the likelihood function, all other aspects of the data being irrelevant. In moderate and large samples, the likelihood function will be unimodel and can be adequately approximated over a sufficiently extensive region near the maximum by a quadratic function. Hence, in these cases the log-likelihood function can be described by its maximum and its second derivatives at the maximum. The values of parameters which maximize the likelihood function, or equivalently the log-likelihood function, are called maximum likelihood (ML) estimates. The second derivatives of the log –likelihood provide measurers of "spread" of the likelihood function and can be used to calculate approximate standard errors for the estimates [ ].

Now, to study the likelihood function of $ARMA(1,1)$ let as suppose the $N = n + d$ original observations Z from a time series which can be denoted by $\mathrm{Z}_{-\mathrm{d}+1},..,\mathrm{Z}_0\mathrm{Z}_1,\mathrm{Z}_2,..,\mathrm{Z}_\mathrm{n}$

we assume that this series is generated by an $ARMA(1,1)$ model. From these observations, we can generate a series w of $\mathrm{n} = \mathrm{N} - \mathrm{d}$ differences$w_1, w_2, .., w_n$, where$w_t = \nabla^d z_t$. The stationary mixed $ARMA(1,1)$ model in eq.7 may be written as [2]:

$$a_t = w_t - \phi_1 w_{t-1} + \theta_1 a_{t-1} \tag{18}$$

Where$E(w_t) = 0$. Suppose that $\{a_t\}$ has the normal distribution with zero mean and constant variance equal to $\sigma^2_{\mathrm{a_t}}$, then the likelihood function can get as follows [2]:

$$L = (2\pi\sigma^2)^{\frac{-n}{2}} |M^{(1,1)}|^{\frac{1}{2}} \exp\left(\frac{-s(\phi_1,\theta_1)}{2\sigma_a^2}\right. \tag{19}$$

Where $M^{(1,1)} = var - cov(\phi_1, \theta_1) = I^{-1}(\phi_1, \theta_1)$

$$= \frac{1}{I(\phi_1,\theta_1)} adj(I(\phi_1, \theta_1)) \tag{20}$$

$$I(\phi_1, \theta_1) = \frac{n}{\sigma_a^2}\begin{bmatrix} \frac{\sigma_a^2}{1-\phi_1^2} & \frac{\sigma_a^2}{1-\phi_1\theta_1} \\ \frac{\sigma_a^2}{1-\phi_1\theta_1} & \frac{\sigma_a^2}{1-\theta_1^2} \end{bmatrix} \tag{21}$$

then the log- likelihood function is:

$$\ln(\mathrm{L}) = -\frac{\mathrm{n}}{2}(2\pi\sigma_\mathrm{a}) + \frac{1}{2}\ln\left(\left|\mathrm{M}^{(1,1)}\right|\right) - \frac{\mathrm{S}(\phi_1,\theta_1)}{2\sigma_\mathrm{a}^2} \tag{22}$$

where:

$$S(\phi_1, \theta_1) = \sum_{t=-\infty}^{n} (a_t | \phi_1, \theta_1, w)^2 \tag{23}$$

is the sum squares errors,n is the sample size, and $[a_t|\phi_1, \theta_1, w] = E([a_t|\phi_1, \theta_1, w])$ denotes the expectation of a_t conditional onϕ_1, θ_1 and w. Sum squares errors can be found by unconditional calculation of the $[a]'s$ are computed recursively by taking conditional expectations in eq.13. A back-calculation provides the values $[w_{-j}], j = 0,1,2,..$
This back-forecasting needed to start off the forward recursion.
For moderate and large values of n in eq.17 is dominated by $S(\phi_1, \theta_1)/2\sigma_a^2$ and thus the contours of the unconditional sum squares function in the space of the parameters (ϕ_1, θ_1) are very nearly contours of likelihood and of log likelihood . It follows, in particular, that the parameter estimates obtained by minimizing the sum of squares in eq.17, called least square estimates will usually provide very close approximation to the (maximum likelihood estimator).

7.3.1.1 Drive formula of gradient of likelihood function

These section, we try to drive general form of steepest descent to estimate $ARMA(1,1)$ model parameters, sDA is an iterative strategy depends on the following rule for numerical computation:

$$\beta_{i-1}^{*} = \beta_{i-1} - k\overline{\nabla}e^2 \tag{24}$$

Where
β_{i-1} Parameter model
k Constant value depend
$\overline{\nabla}e^2$ is the gradient which approximate by $\overline{\nabla}e^2 = [\frac{\partial e^2}{\partial \beta_1}, \frac{\partial e^2}{\partial \beta_2}, \dots, \frac{\partial e^2}{\partial \beta_m}]$
we can see that the estimation of parameters depend on iterative algorithm which start with initial value β_i (get by one of traditional estimation methods) this algorithm continue in modified these estimators even we get the value which don't have change in values

$$FMSE = \frac{\sum_{t=1}^{n} (z_t - \hat{z}_t)^2}{n-1} \tag{25}$$

where z_t, actual value of observed time series; $\hat{z}_t$ predicted value of actual value.
We know, ARMA(1,1) model form is

$$z_t = \phi_{1t} z_{t-1} + a_t - \theta_{1t} a_{t-1} \tag{26}$$

Where a_ts are a random variable with standard normal density function known as random shock term.
then

$$a_t^2 = (z_t - \phi_{1t} w_{t-1} + \theta_{1t} a_{t-1})^2 \tag{27}$$

$(\frac{\partial a_t^2}{\partial \phi_{1t}} = -2a_t z_{t-1}, \frac{\partial a_t^2}{\partial \theta_{1t}} = -2a_t a_{t-1})$

S0

$$[\phi_{1t}^{*}, \theta_{1t}^{*}] = [\phi_{1t} + 2k a_t z_{t-1}, \theta_{1t} - 2k a_t a_{t-1}] \tag{28}$$

The value of k gets as follows

$$a_t = z_t - \phi_{1t} w_{t-1} + \theta_{1t} a_{t-1} \tag{29}$$

$$a_t^* = z_t - \phi_{1t}^* w_{t-1} + \theta_{1t} a_{t-1} \tag{30}$$

$$|\Delta a_t| = |a_t^* - a_t| = 2k a_t (z_{t-1}^2 + a_{t-1}^2)$$

$$\because 0 < |\tfrac{\Delta a_t}{a_t}| < 1 \quad \rightarrow 0 < 2k(z_{t-1}^2 + a_{t-1}^2) < 1 \quad \text{so}$$

$$0 < k < \frac{1}{2(z_{t-1}^2 + a_{t-1}^2)^2}$$

7.3.2 Results of likelihood function

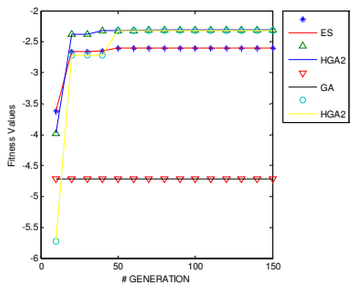
In order to evaluate the behaviour of A , ES, SDA with HGA_1 and HGA_2, we performed several experiments to test the capabilities of the methods. The results of experiments given by the following the conceptual algorithm for simple genetic algorithm and hybrid genetic algorithms adopted for the likelihood estimator of $ARMA(1,1)$. The experimental results performed here are based on different sample size (i.e.$n = 25,75,125$),(ϕ_1, θ_1) set-to $\{(0.8,0.6), (0.4,0.5), (-0.1,0.2), (-0.3, -0.4)$. The random sample are generated using Box-Muller formula which presented by using Delphi Pascal coding programming, MATLAP2008. All results obtained by running each experiment 5 different runs and each iterates with 150 generations for population size 50 and averaging the resulting data for $P_c = 0.75, P_m = 0.1$. Further, the results of (sGA, HGA) compared with those obtained by stepwise descent based on initial value computed by moment method for the same value of (parameters)(ϕ_1, θ_1) and sample size (n) (with 1000 runs). The comparison made based on Mean square error

$$MSE = var(\phi) + bias$$

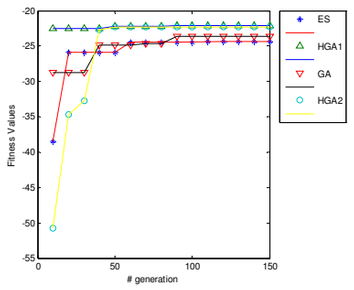
n	ϕ_1	θ_1	SDA best		sGA best		HGA_1 best		ES		HGA_2 best	
			ϕ_1	θ_1	ϕ_1	θ_1	ϕ_1	θ_1	ϕ_1	θ_1	ϕ_1	θ_1
25	0.6	0.8	1.28	1.43	0.473	0.693	0.2300	0.598	0.3668	0.3631	0.01765	0.0342
	0.4	0.5	0.74	1.01	0.517	0.51	0.276	0.419	0.1703	0.0752	0.012116	0.0312
	-0.1	0.2	0.49	0.29	0.191	0.156	0.1 02	0.139	0.1454	0.2404	0.09456	0.0104
	-0.3	-0.4	0.55	0.95	0.393	0.436	0.182	0.421	0.1351	0.2295	0.01023	0.0353
75	0.6	0.8	1.27	1.34	0.415	0.568	0.182	0.484	0.3556	0.3520	0.01198	0.0211
	0.4	0.5	0.71	0.88	0.446	0.391	0.257	0.336	0.1701	0.0751	0.01201	0.0024
	-0.1	0.2	0.21	0.15	0.157	0.154	0.054	0.093	0.1437	0.2374	0.0012	0.0065
	-0..3	-0.4	0.52	0.715	0.381	0.325	0.159	0.317	0.1336	0.2272	0.00543	0.0012
125	0.6	0.8	1.23	1.32	0.324	0.546	0.139	0.462	0.1803	0.0584	0.00156	0.0011
	0.4	0.5	0.66	0.87	0.186	0.342	0.108	0.3106	0.169	0.0746	0.009	0.0013
	-0.1	0.2	0.27	0.1237	0.137	0.136	0.031	0.0 35	0.1451	0.2401	0.0023	0.0015
	-0..3	-0.4	0.432	0.52	0.121	0.306	0.013	`0.213	0.1337	0.2274	0.00161	0.0010

Table 1. Comparisons among (GA,ES,SDA, HGA1,HGA2) algorithm based on MSE of best estimator after averaging 5 runs.

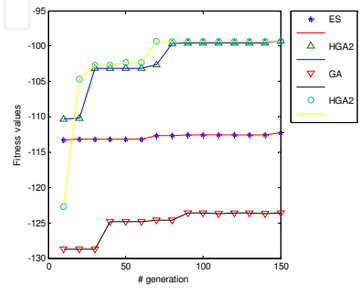
Results given in Fig. 7.and table(1).The experiments on a set of data give some impressions of the behaviours of (sGA, HGA) and sDA. As one can see that, MSE of HGA are smaller than those of steepest descent (sDA). This indicates that HGA is more reliable than HGA and sDA to give estimator of the parameters of the model under study. Moreover, one can see that value of MSE decreases as the sample size increase. For $(sGA, HGA$ we can also see that the value of sum square decreases when increasing the number of generation and sample size. In addition, the behaviour of $(HGA$ when the objective function parameters(ϕ_1, θ_1) take positive values are better than when they are negative. The HGA_2 algorithm was also more robust than the sGA, ES and HGA_1 performing optimally across a broad range of parameter values. In addition we can see the second best algorithm converge to best solution is HGA1.



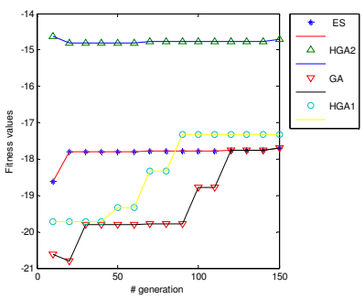
A1
Simple size
n=25,(ϕ_1=.6, θ_1=.8)



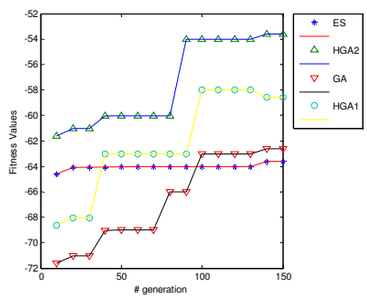
A2
Simple size
n=75,(ϕ_1=.6, θ_1=.8)



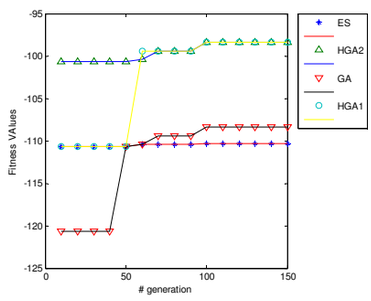
A3
Simple size
n=125,(ϕ_1=.6, θ_1=.8)



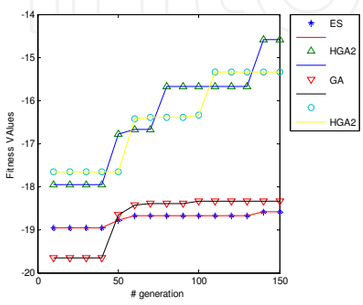
B1
Simple size
n=25,(ϕ_1=-.1, θ_1=.2)



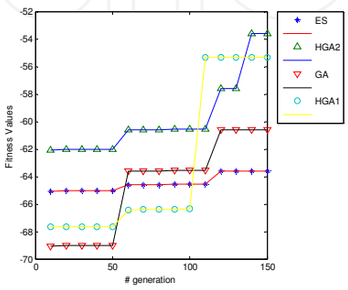
B2
Simple size
n=75,(ϕ_1=-.1, θ_1=.2)



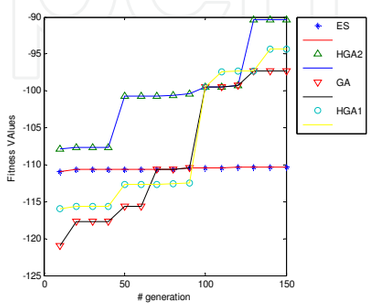
B3
Simple size
n=125,(ϕ_1=-.1, θ_1=.2)



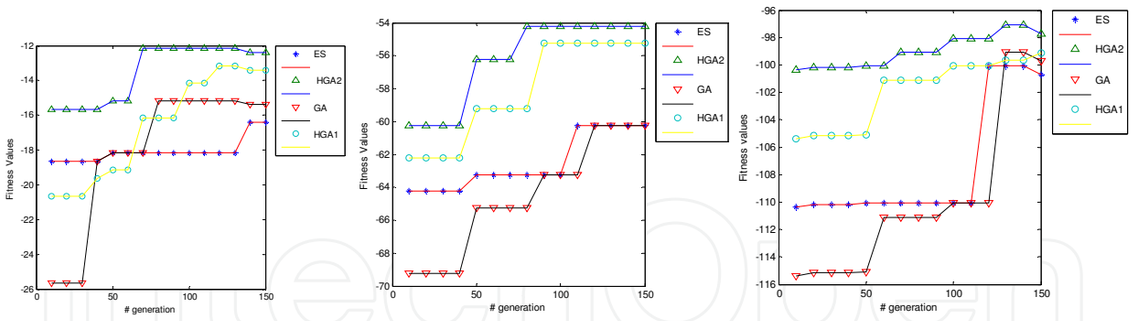
C1
Simple size
n=25,(ϕ_1=.4, θ_1=.5)



C2
Simple size
n=75,(ϕ_1=.4, θ_1=.5)



C3
Simple size
n=125,(ϕ_1=.4, θ_1=.5)



D1
Simple size
n=25,(ϕ_1=-.3, θ_1=-.4)

D2
Simple size
n=75,(ϕ_1=-.3, θ_1=-.4)

D3
Simple size
n=125,(ϕ_1=-.3, θ_1=-.4)

Fig. 8. Compression Among best fitness values respect to sample size and ARMA model parameters getting by algorithms under study

7.3.2 Benchmark test functions

The reminder of test functions using the following parameterization of the algorithms compared are used for experimental test runs

- Genetic Algorithm with population size $\mu = 200$ mutation rate $p_m = 0.1$, crossover rate $p_c = 0.75$ one-point crossover , binary code, and bit string length l =24, this algorithm denoted sGA
- Evolution strategy $((30^+, 200) - ES$ with self adaption of $n_\sigma = n$ standard deviation, no correlated mutation, local discrete recombination on object variables $rec_x = 1$, global intermediate recombination on standard deviation $rec_\sigma = 4$ and standard deviation initialize at 3.0. $(30^+, 200) - ES$ was used for unimodel functions, while $(30{,}200) - ES$ used for multimodal function.
- Steepest decent Algorithm(SDA): use maximum of iteration =15; with tolerance =10^{-3}. And initial size take randomly from x range .
- Hybrid Es algorithm with the algorithm GA a master and (30+200) ES as a tool for the master , for unimodel function, the best fit 30 GA individual are selected to be delivered to the (30+200)ES tool. while for multimodal functions, an evenly random selected 30 GA individuals are delivered to that ES tool.HAO is active after 3 generations of the cross-fertilization phase. HGA1
- Hybrid steepest descent with genetic algorithm Genetic Algorithm with population size $\mu = 200$ mutation rate $p_m = 0.1$, crossover rate $p_c = 0.75$ one-point crossover , binary code, and bit string length l =24, this algorithm denoted sGA, use maximum of iteration =15; with tolerance =10^{-3}. And initial solution take as the best individual from genetic algorithm for values equal (pop_size/ 3) from population. All results were obtained by running 5 experiments per algorithm and averaging the resulting data.

F_2: Test function

Ackley Function(multi), this function is named after Ackley who invented it

$$f(\vec{x}) = -20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i{}^2)}} - e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)} + 20 + e \qquad (31)$$

The original version was a two- dimensional function and it was later generalized to n dimension by Back. In this study form defined on n=2. The values of $\vec{x}$ defined on [-32.0,32.0]. The global minimum is located at the origin and its value is zero. The Ackley function is a nonlinear multimodal function with regularly distributed local optima

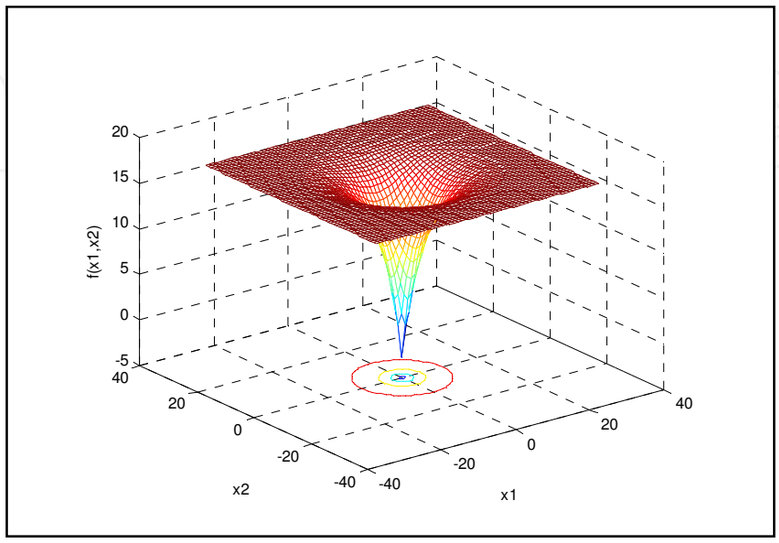


Fig. 9. F_2 Test function shape.

The results showing the ability of HGA2 to give more robust results. Also HGA1 promise to give robust results cleared in fig.16 .

F_3: ***Test Function***

$$f(x_1, x_2) = [25 - (x_1 - 5)^2 - (x_2 - 5)^2]^{\frac{1}{2}} \tag{32}$$

With constraints
$$-x_1^2 + 4x_2 \leq 0$$
$$4x_1 - x_2^2 + 12x_2 \leq 58$$
$$x_1, x_2$$

$$(x_1, x_2) \in [0,7]^2.$$

Optimal solution get when x_1, x_2=5.

The first set of results of this function given in table(2) for describing the SDA algorithm which is depended on experiment designed for studying the behaviour of algorithm under study, where results in table (2) explained how the gradient algorithm depend on three operators the first one is initial values x_0 generated randomly, tolerance of accuracy take equal to 10^{-3}; number of iterations (determined at the begging of experiment designed equal to 50. from results increasing number iterations when increasing variation of parameter (S), the best results get at (5.5419,5. 7225)where maximum value is(4.9178) where S=1.691, number of iterations is 6. With x_0 = (3.5,3).

When applying sGA we get the best solution get at generation 15 with X=(4.5035, 4.8091) and $F(x_1,x_2)$ = 4.9716. Hybrid of sDA with GA gave the best results at generation 95 with

x = (4.9904, 4.96268) *and* $F(x_1,x_2)$ = 4.99976 the compression among algorithms used refer to HGA2 And HGA1 to give more robust results with

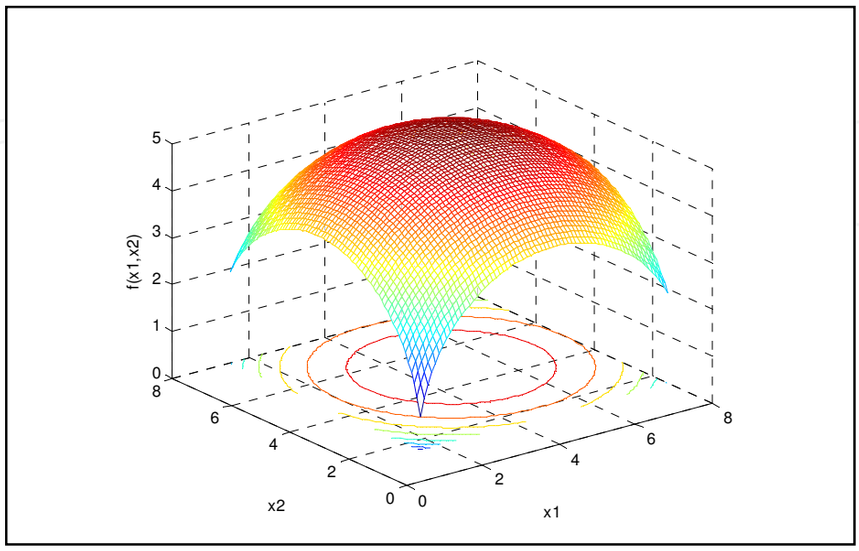


Fig. 10. F3 Test function shape.

$X_0=(x_{10},x_{20})$	Max iteration for 5 runs	s	x_1	x_2	$f(x_1,x_2)$	Min iteration for 5 runs	s	x_1	x_2	$f(x_1,x_2)$
(7 ,1)	20	2.6676	4.349	6.302	4.7834	6	2.381	4.5177	5.9647	4.8823
(5 ,4.5)	20	2.6412	5	7.1406	4.5186	2	1.839	5	6.3387	4.8175
(3.5,3)	38	2.4953	5.8697	6.1596	4.7853	6	1.691	5.5419	5.7225	4.9178
(2.5,1.2)	24	2.836	5.8308	6.2628	4.766	8	2.729	5.5737	5.872	4.8898

Table 2. Relation among number of iteration and(stepsize –s-) with initial value

F_4: Test Function

This function with multiple basins of attraction

$$f(x,y) = \frac{d_i}{r_i^2}(\bar{x}_1^2 - \bar{y}_1^2)(2 - \frac{\bar{x}_1^2+\bar{y}_1^2}{r_i} - d_i \qquad \bar{x}_1^2 + \bar{y}_1^2 \le r_i^2 \tag{33}$$

with constraints d_i , r_i generated randomly. The test functions given in eq.29 are multi-modal functions with multiple basins of attraction. The coordinates $(x_{o,i}, y_{o,i})$ are the coordinates of the basin of attraction "i", which has random geometry (radius and depth $r_i$depth d_i), The basins of attraction for functions are randomly distributed. (Goldberg and Voessner, 1999) has conical basins of attraction represents the best case for local search, in which only one local search is required to find the local minimum.

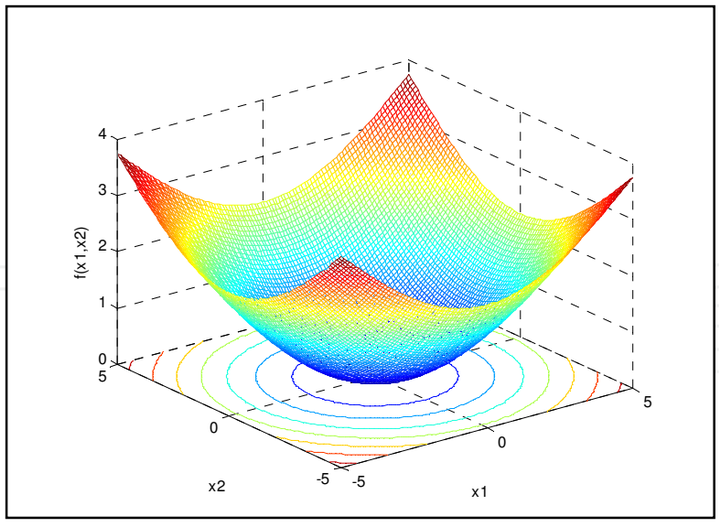


Fig. 11. F_4 Test function shape

The simulation results cleared in fig.16.

F5. Single test function

This function is a nonlinear function with single input variable

$$f(x) = e^{-2(ln2)(\frac{x-0.1}{0.8})^2}|\sin(5\pi x)| \tag{34}$$

Where $x \in [-1,1]$. Actually, this simple function has several local maximum. However, there is only one global maximum, as shown in Fig.11.

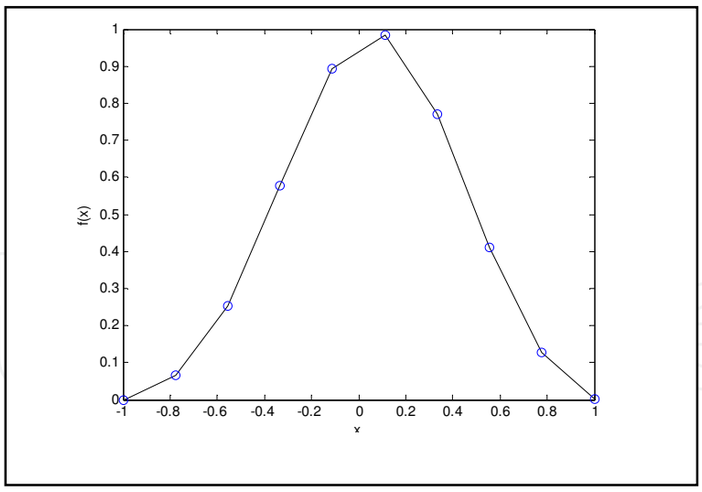


Fig. 12. F_5 Test function shape

F6: Test function

$$f(x,y) = \cos(x)^2 + \sin(y)^2 \tag{35}$$

$\vec{x}$ defined in[-5,5], this function has infinite global maximum in R^2 at points $\left(\frac{m\pi}{2}, n\pi\right), m, n = \mp 1, \mp 2 \ldots$

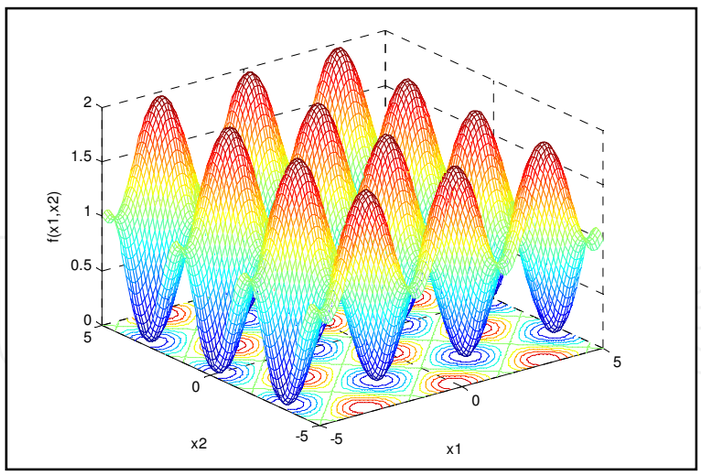


Fig. 13. F_6 Test function shape

F7: Test Function

This function known as Rosenbrock function was invented by Rosenbrock, mathematically defined as

$$f(\vec{x}) = \sum_{i=1}^{n-1}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \quad (36)$$

Where $\vec{x}$ is an n-dimension vector located within the range$[-30.0,30.0]^n$. the global optimum is located at (1,…,1) with a function value of zero. This function exhibit a parabolic-shaped deep valley. In the optimization literature it is considered a difficult problem due to the nonlinear interaction between variables [1].

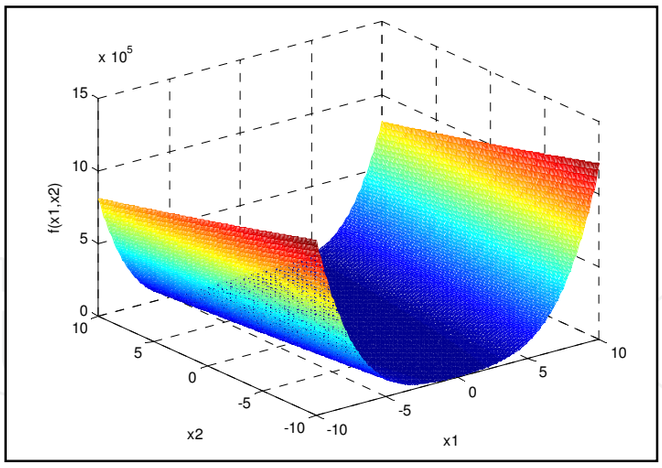


Fig. 14. F_7 Test function shape

F8: Test function

The Salmon function is rotation – invariant and was proposed by Salmon ,it is defined as

$$f(\vec{x}) = 1 - \cos(2\pi\sqrt{\sum_{i=1}^{n} x_i^2}) + 0.1\sqrt{\sum_{i=1}^{n} x_i^2} \quad (37)$$

Where $\vec{x}$ is an n-dimensional vector located within the range $[-100,100]$ the global optimum located at the origin with a function value of zero.

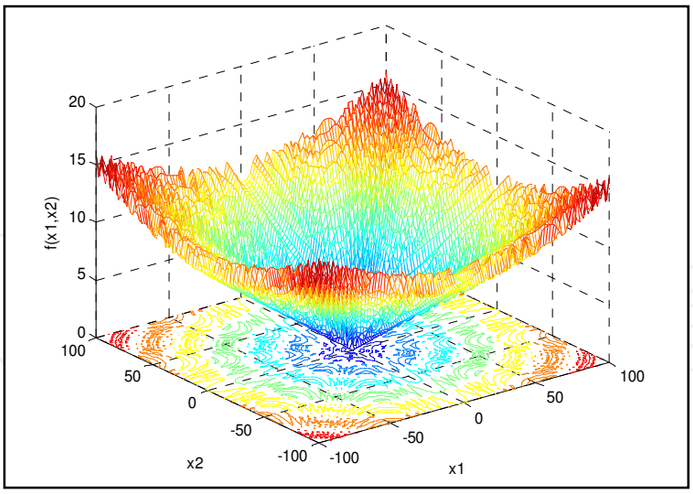


Fig. 15. F_8 Test function shape

F9. Test function

This function also known as Schaffer's function or the sine envelope sine wave. Mathematically define as

$$f(\vec{x}) = 0.5 + \frac{\sin^2(\sqrt{x_1^2+x_2^2})}{(1+0.001(x_1^2+x_2^2))^2} \tag{38}$$

Where $\vec{x}$ is a two-dimension vector located within the range $[[-100.0,100.0]^n$. The global optimal is located at the origin with a function value equal to zero.

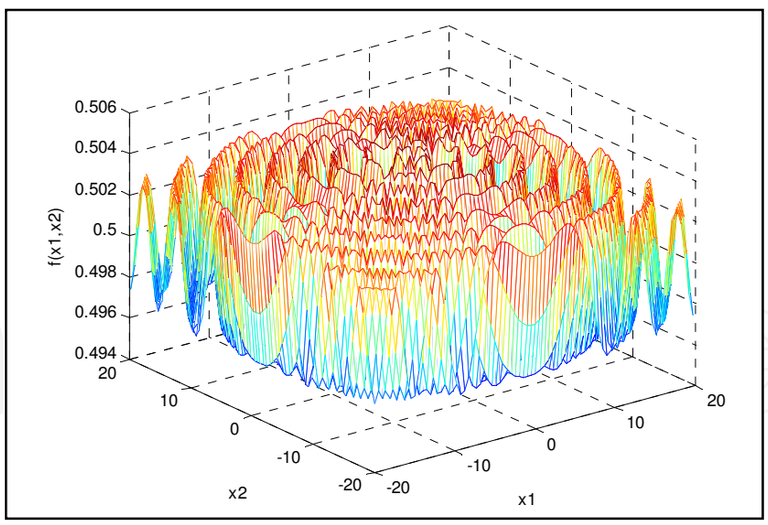


Fig. 16. F_9 Test function shape

F10 : Esom function

This function was proposed by Easom to evaluate global optimization techniques. It is n-dimensional function with single minimum that is also the global optimum. The mathematical expression of this function is

$$f(\vec{x}) = -\prod_{i=1}^{n} \cos(x_i) \cdot e^{-\sum_{i=1}^{n}(x_i-\pi)^2} \tag{39}$$

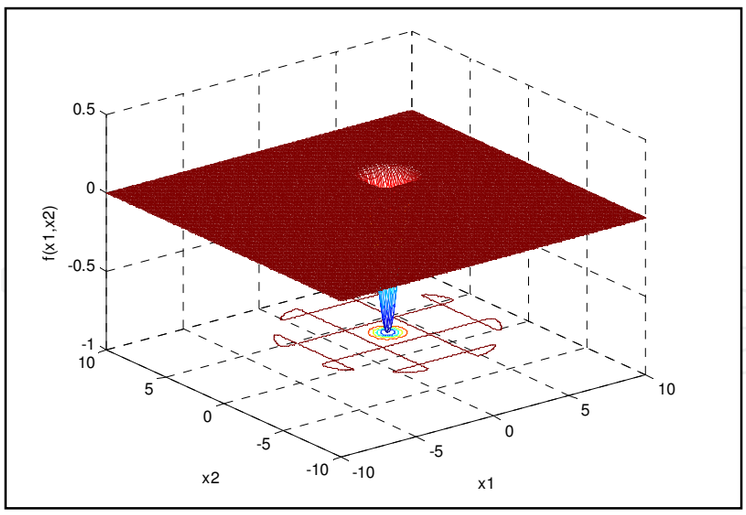
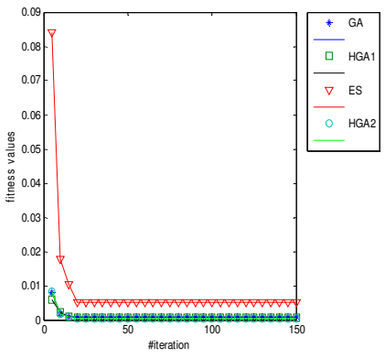
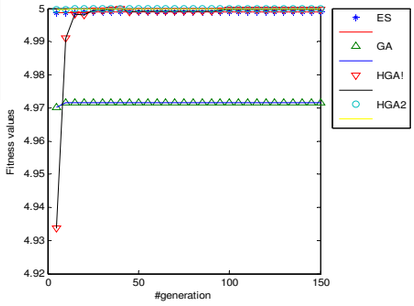


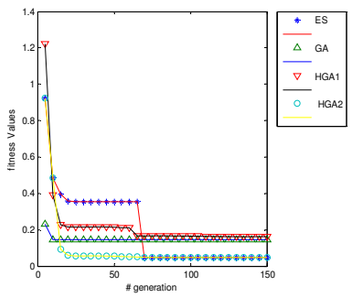
Fig. 17. F_9 Test function shape



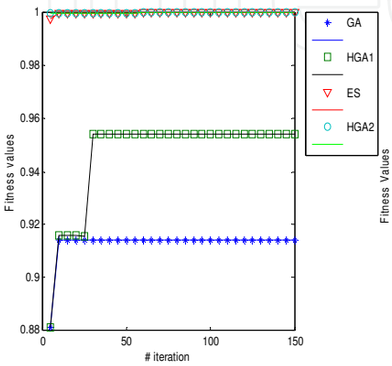
F_2 Test Function Comparison



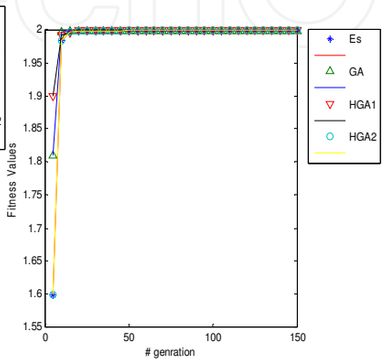
F_3 Test Function Comparison



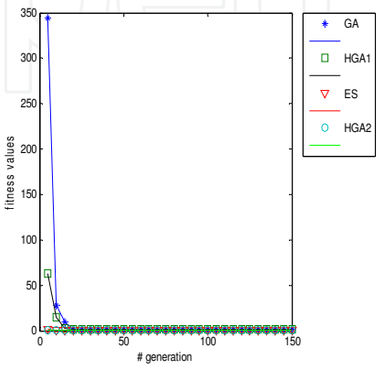
F_4 Test Function Comparison



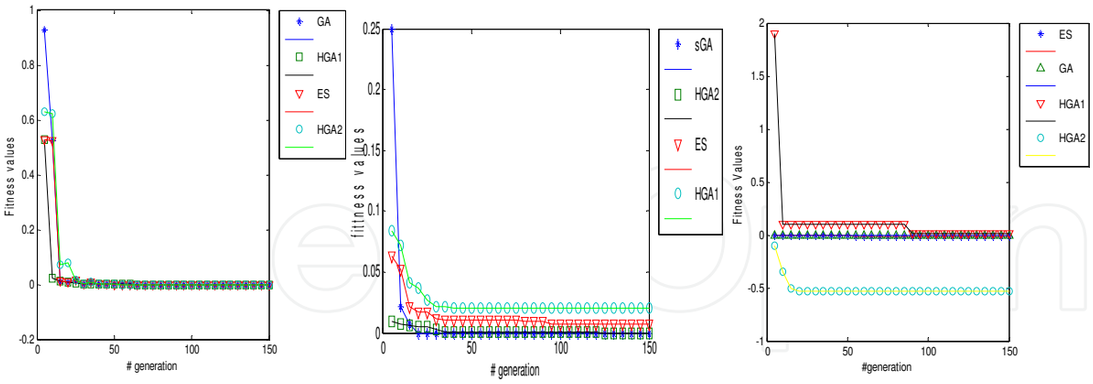
F_5 Test Function Comparison



F_6 Test Function Comparison



F_7 Test Function Comparison



F_8 Test Function Comparison F_9 Test Function Comparison F_{10} Test Function Comparison

Fig. 18. Compression among fitness values respect to number of generations for benchmark test functions

7.3.3 General results getting from designing experiments

According to the results presented above, a general trend will be drawn about the course of actions of the competent algorithms. Results show that sGA alone with its bit level crossover and mutation operators can act as a heuristic for exploration with somewhat little emphasis on search focus.

Sample GA showed to be trapped by local plateaus. One could return this behavior to the main distinguished operator of the master GA, the one point crossover operator. one can easily see that the canonical GA is the slowest of the algorithms under study. The behavior of GA is almost identical on all the unimodel functions.

The collective nature of GA tournament selection, one point crossover, and mutation operators give a clear demonstration of its missing emphasis on the convergence and local optimization. On the other hand , the ES with self adaption of $n_\sigma = n$ standard deviation is , on overall, the faster by far and its results are superior to that obtained from (GA,HGA1). The combination of self-adaption , recombination , and relatively strong selective pressure as used in ES algorithm. The nature of the preservative survival of the best individual implied by the plus selection strategy. Also, the self adaption role of the strategy parameters through intermediate recombination and mutation is shown to be fascinating. Even if all the parents start with equal $\sigma_i = \sigma = 3.0\ \forall i = 1, \ldots, n_\sigma$, and all the step length components are varied by a common random factor in the production of the offspring , the σ_i of all individuals will differ from each other in the subsequent generations through self adaption. So in this way a better combination affords a higher chance of survival to its bearer. It can therefore be expected that in the course of the optimum search, the currently best combination of the $\{\sigma_i; \forall i = 1, .., n_\sigma\}$prevails.

The HGA2 reduces the speed gap between the canonical GA and standard variant ESs convergence , it does not outperformed ES with both its, variants, except for some cases . The deviation in convergence velocity of the HGA2 from ES variants can be attributed to the fact that although in the first cross-fertilization phase of HGA2, the best GA individuals are enhanced by the coupling EA algorithm, the exploration power of the master GA still remained an order of magnitude. A closer look is given here to compare the behavior of ES and HGA2 one hand , and HGA2 with HGA1,GA, SDA. comparing the results of the overall

hybridization established in HGA1 with the hybridization of HGA2, one could see that the presented results of HGA2 are more powerful than that of HGA1.

8. Conclusions

This chapter is devoted to global optimization algorithms, which are methods to find optimal solutions for given problems. It especially focuses on two major groups of optimization algorithms evolutionary computation by discussing evolutionary algorithms, genetic algorithms, evolution strategy. Second group represent by hybrid algorithms which are coupling simple GA with local algorithm steepest descent algorithm(HGA1) and GA with self adaptive global algorithm evolution strategy (HGA2). The results , depending on the standard functions presented in the test suite, it campers the performance of (sGA, (μ^+, λ)-ES, SDA, HGA1,HGA2) algorithms. The simulating experiments designed for sets of benchmark test functions classifies as unimodel and multimodal, Four unimodel functions, the hybridization was found to be advantageous for speeding up the performance of the canonical GA so the speed gap difference between the very general purpose optimizer algorithms as canonical GA and the specialized parametric optimization algorithm as multimember ES is diminished . Also, the hybridization was found to be beneficial in multimodal functions where convergence reliability is of interest. By taking the advantages of both exploration power of the GA and the exploitation power of the multimember ES, the HGA introduces more reliable solutions than GA or ES when worked individually.

9. References

A.Montes Marco; Rolda'n Oca (2006).On the optimization of Particle Swarm Optimization, Annee Academique

W. W. S. Wei, 1990, Time Series Analysis: Univariate and Multivariate Methods, Wesley Publishing com. INC, New York.

G. E. B. Box, and G. M. Jenkins(1976).Time Series Analyses is: forecasting and control", Holden-Day, San Francisco.

S. Makridakis, S. C. Wheelwright (1999). Forecasting Method and Application, John Wiley& sons, New York.

T. Bäck, and H,-P, Schwefel, 1993, An overview of evolutionary algorithms for parameter optimizations", Evolutionary Computation, Cambridge, Vol. 1, No. 1, pp. 1-23,.

T. Bäck, and H,-P, Schwefel (1995). Evolution strategies I: variants and their computational implementation", Genetic Algorithms in Engineering and Computer Science, pp. 11-26.

H,-P, Schwefel(1981). Numerical Optimization of Computer Models. New York: John Wiley & Sons,.

F.Hoffmeister, and T.Back, "Genetic Algorithms and Evolutionary Strategies: Similarities and Differences", Parallel Problem Solving from Nature PPSN I,. Vol. 496, pp.455-

B.A.Attea, B.A.Husssain (2006). An evolution strategy for likelihood Estimators of ARMA (1, 1) Mode The 4th International Multiconference and Information Technology CSIT2006 Vol.1, Amman-Jordan.

B.A.Hussain, R.D.AL.Dabbagh (2007). A Conical Genetic Algorithm for Likelihood Estimation of first order Moving Average Model parameter. Neural Network World Vol.4, pp(271-285).

Adrian E. Drake, Robert E. Marks, Genetic algorithm in economic and Finance: forecasting Stock Market Prices and Foreign Exchange, A Review, Internet

T. Smith Robert, B. Minton Roland (2002). Multivariable Calculus, second edition, McGraw-Hill.

Thomas Weise (2009). Global Optimization Algorithms– Theory and Application – ;Version: 2009-06-26; Newest Version: http:/ / www.it-weise.de/ .

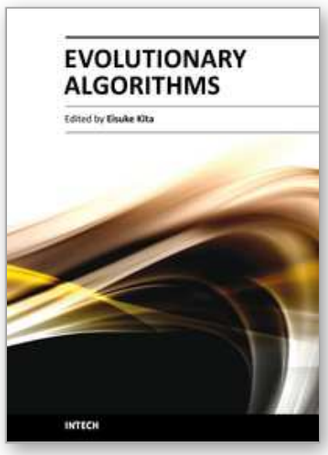
Attea, Bara'a Ali (2001). Interdigitation : Hybrid of Genetic Algorithms and Evolution Strategies for Parametric Optimization; Doctoral theses, Iraq/ Baghdad.

Hussain . Basad Ali (2002) Comparison Among Forecasting Methods of Markov and Mixed Model by Using Simulation; Master thesis; Iraq Baghdad.

Rabunal . Juan R, Julian Dorado(2006).Artificial Neural Network in Real-Life Applications, Idea Group Publishing,.

X. Wang, X.Z. Gao and S.J Ovaska (2008). A Novel Particle Swarm – based Method for Nonlinear Function Optimization, International journal of Computational Intelligence Research,Vol.4, No.3, pp.281-289.

Guoli Zhang, Haiyan Lu(2006). Hybrid Real coded Genetic Algorithm with Quasi-Simplex Technique ,IJCSNS International Journal of computer Science and Network Security, Vol.6 No.10,October .



Evolutionary Algorithms

Edited by Prof. Eisuke Kita

ISBN 978-953-307-171-8
Hard cover, 584 pages
Publisher InTech
Published online 26, April, 2011
Published in print edition April, 2011

Evolutionary algorithms are successively applied to wide optimization problems in the engineering, marketing, operations research, and social science, such as include scheduling, genetics, material selection, structural design and so on. Apart from mathematical optimization problems, evolutionary algorithms have also been used as an experimental framework within biological evolution and natural selection in the field of artificial life.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen