A New Meta-Heuristic Algorithm for Optimization Based on Variance Reduction of Guassian Distribution

A. Namadchian[1], M. Ramezani[2], N. Razmjooy[1]
1- Department of Electrical Engineering, University of Tafresh, Tafresh, Iran.
Email: Ali_namadchian@yahoo.com, navid.razmjooy@hotmail.com,
2- Department of Mathematics, University of Tafresh, Tafresh, Iran.
Email: ramezani@aut.ac.ir (corresponding author)

Received: July 2016 Revised: July 2016 Accepted: August 2016

ABSTRACT:
Meta-heuristic methods are global optimization algorithms which are widely used in the engineering issues, nowadays. The main problem with the classical optimization algorithms is their slow rate of convergence to time-consuming mathematical calculations. In this paper, a new stochastic search for optimization is presented using variable variance Guassian distribution sampling. The main idea of searching in this algorithm is to regenerate new samples around each solution with a Guassian distribution. The proposed algorithm is applied to four popular test functions for optimizations (Griewank, Booth, Rosenbrock, Rastrigin). Numerical simulations have revealed that the new presented algorithm outperformed simulated annealing and genetic algorithms.

KEYWORDS: Optimization, Gaussian distribution, covariance matrix, stochastic search, variance reduction, Probability Density Function (PDF, hereafter)

1. INTRODUCTION

Optimization is the process of finding the best solution to minimize a function according to the problem constraints. In other words, optimization task is to nominate the problem variables in order to optimize the fitness function to achieve a purpose. The general description of the optimization process can be described as below:

$$
\begin{aligned}
&f_{\min} = \min\nolimits_{x\in s} f(x),\\
&f_{\max} = \max\nolimits_{x\in s} f(x),\\
&x = (i = 1,2,3...l)^T \qquad\qquad (1)\\
&c_i(x) = 0,\quad i = 1,2,3...M' \quad i = 1, M' \in N\\
&c_i(x) \geq 0,\quad i = M'+1...M \quad i = 1, M \in N
\end{aligned}
$$

Where $f(x)$ is the fitness function, x is the column vector of the l independent variables and $c_i(x)$ represents the class of constraint functions. Constraint and unconstraint equations form the equality $c_i(x) = 0$ and inequality $c_i(x) \geq 0$, respectively. In the above equations, $f(x)$ and $c_i(x)$ are characterized as the problem constraints [1].

The main problem with the classical optimization algorithms is their slow rate of convergence to time-consuming mathematical calculations; even with the help of advanced technology, solving an extensive multi-dimensional problem needs several years of hard work [2]. In recent years, meta-heuristic algorithms have put up a brilliant or acceptable performance in comparison to classical methods.

The key issue of these algorithms has been inspired, either by nature or by treating human beings in various fields, such as political, social, etc [3]. Meta-heuristic algorithms mainly begin with an initial set of variables, known as the population, and then end in reaching the global minimum or maximum of the fitness function. There are a large number of naturally-inspired optimization algorithms, to which even more algorithms are being added [4-9]. In this article, a new meta-heuristic algorithm is introduced by searching the solution space, based on Gaussian distribution, which is a widely used one in the theory of probability, largely due to the central limit theorem. The main goal of such algorithm is to regenerate Gaussian distributed points around other points in the solution space. But the variance of distribution is variable with respect to the function evaluation of each point. If the function is univariate, the normal distribution employed will be univariate, as well; if, however, it is a multivariate function, the corresponding normal distribution will also be multivariate (MVN).

In section 2, general formulas for univariate normal distribution and its extension to MVN have been discussed; besides, the shape of the densities for a typical MVN, based on different form of covariance matrix, has been inspected. Section 3, introduces the new VRGS (Variance Reducing Gaussian Search) algorithm in detail and delves into the reasons why the best MVN for the algorithm is jointly independent MVN, based on different form of the MVN densities. In section 4, the performance of VRGS is explored and compared to genetic algorithm (GA) and simulate annealing (SA), to investigate which of four popular test functions have been employed. Finally, section 5 presents some concluding points about the algorithm and further suggestions to increase its performance.

2. MULTIVARIABLE NORMAL DISTRIBUTION

The Multivariate Normal Distribution is a generalization of univariate normal distribution, where the latter is characterized by its mean and variance. Consider a univariate normal probability density with μ (the mean) and σ (the variance) for the random variable x:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

$$x \subset (-\infty, \infty) \qquad (2)$$

$$\mu \subset (-\infty, \infty) \quad \sigma^2 > 0$$

The main parts of this density are $f(x) = \frac{1}{\sqrt{2\pi\sigma^2}}$ (normalizing factor) and $e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$ (kernel of the density). This is a measure of distance of x from μ . Using linear algebra, we can rewrite the argument of the exponential function as:

$$-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2} = \frac{1}{2}(x-\mu)(\sigma^2)^{-1}(\mu - x) \qquad (3)$$

Now considering a $p \times 1$ random vector $X = [X_1, X_2, ..., X_p]$, the kernel could be generalized to:

$$\ker nel = e^{-\frac{1}{2}(x-\mu)' \Sigma^{-1}(x-\mu)} \qquad (4)$$

Where $x = [x_1, x_2, ..., x_p]^T$, $\mu = [\mu_1, \mu_2, ..., \mu_p]^T$ and $\Sigma = \begin{pmatrix} \sigma_{11} & \cdots & \sigma_{1p} \\ \vdots & \ddots & \vdots \\ \sigma_{p1} & \cdots & \sigma_{pp} \end{pmatrix}$ is a $p \times p$ covariance matrix.

After substituting the generalized kernel and changing the normalization constant, the density function of the MVN is:

$$f(x) = \frac{1}{(2\pi)^{\frac{p}{2}} \left|\Sigma\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)' \Sigma^{-1}(x-\mu)} \qquad (5)$$

Where $\left|\Sigma\right|$ is the determinant of the covariance matrix. The normalizing factor makes the volume under the MVN density equal to one. We will discuss about the shape of a MVN based on its mean and variance here. The main factor that forms its shape is its covariance matrix. If that matrix is a diagonal one with $\sigma_{ii}\ i = 1,2,...p$, its determinant is non-zero $\left|\Sigma\right| = \prod_{i=1}^{p} \sigma_{ii}^2$, the inverse of which is also a diagonal one with diagonal entries equal to $\frac{1}{\sigma_{ii}^2}$. In this case, the kernel and the PDF can be re-written as:

$$\ker nel = \sum\nolimits_{i=1}^{p} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma_{ii}^2}} \qquad (6)$$

$$f(x) = \prod_{i=1}^{p} \frac{1}{\sqrt{2\pi\sigma_{ii}^2}} e^{-\frac{1}{2}\frac{(x_i - \mu_i)^2}{\sigma_{ii}^2}} \qquad (7)$$

which is the product of p independent normal distribution.

Visualizing over bi-dimensional MVN is not possible; therefore, the inspection of the shape of MVN is restricted to the bi-variate normal distribution. If there is a need to get a vision of 1D distribution of each random variable based on its bi-variate normal distribution, the density contours of the PDF can be helpful. In the figure below for four different cases, the 3D PDF and density contours of a bi-variate normal distribution with $\mu = [\mu_1, \mu_2]$ and $\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$ have been illustrated, which give a clear idea how the densities of PDF change with respect to the form of covariance matrix.

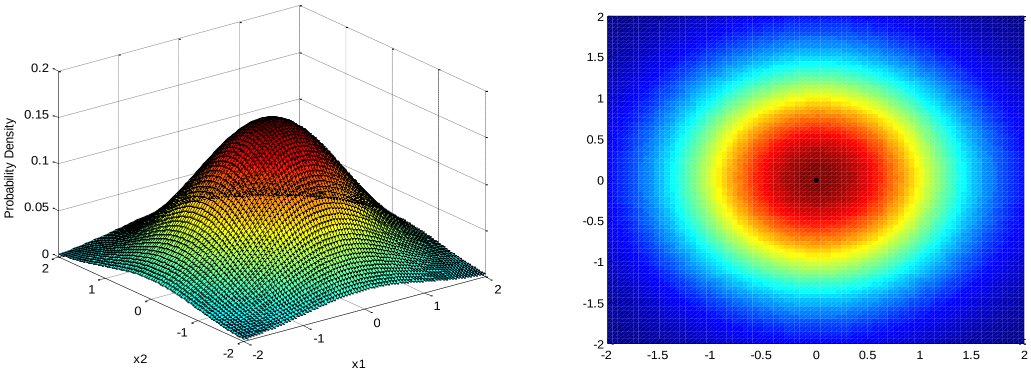


Fig. 1. a) Probability density b) Density contours $\mu_1 = \mu_2 = 0, \sigma_{11} = \sigma_{22} = 1, \sigma_{12} = \sigma_{21} = 0$

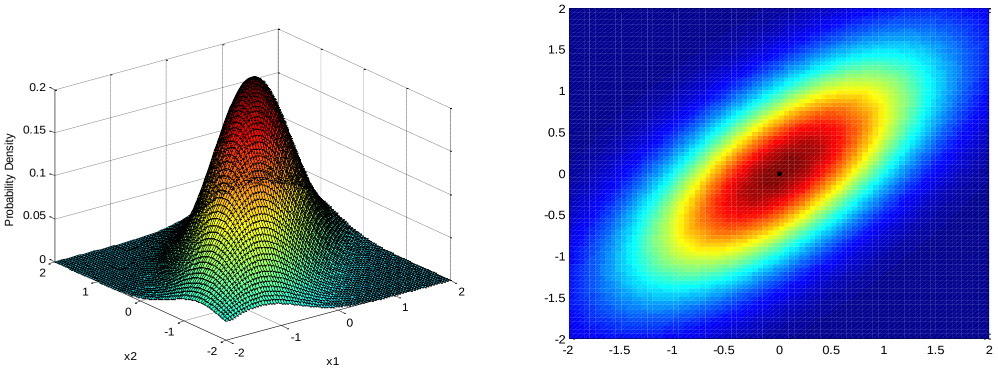


Fig. 2. a) Probability density b) Density contours $\mu_1 = \mu_2 = 0, \sigma_{11} = \sigma_{22} = 1, \sigma_{12} = \sigma_{21} = 0.7$

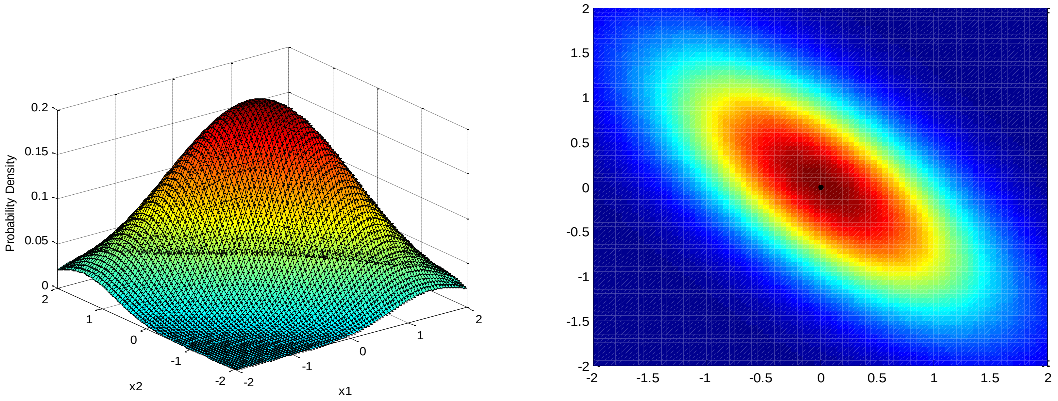


Fig. 3. a) Probability density b) Density contours $\mu_1 = \mu_2 = 0, \sigma_{11} = \sigma_{22} = 1, \sigma_{12} = \sigma_{21} = -0.7$

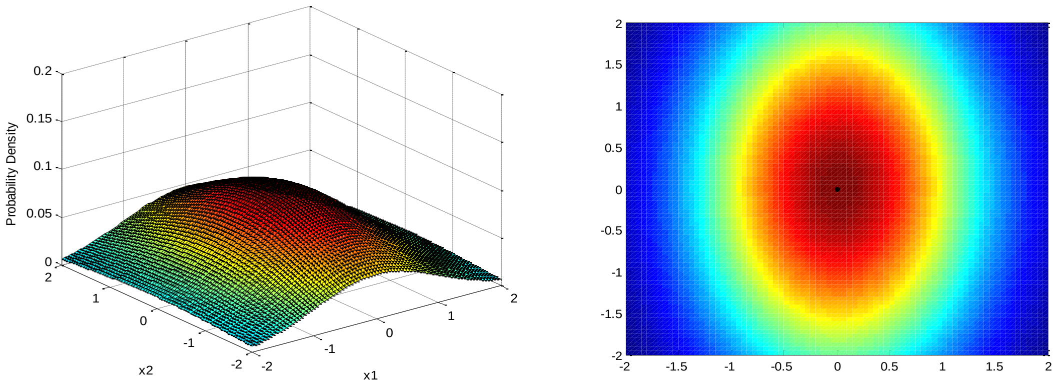


Fig. 4. a) Probability density b) Density contours $\mu_1 = \mu_2 = 0, \sigma_{11} = 1, \sigma_{22} = 3, \sigma_{12} = \sigma_{21} = 0$

The figures above cover nearly all forms of covariance matrix, except for the time when it is a singular one; in that case, the MVN is degenerate and the distribution does not have any density.

3. VARIANCE REDUCING GAUSSIAN SEARCH (VRGS)

In this recently proposed method, search for the solution space X, is not based on a bio-inspired algorithm. First, we choose $N = [n_1 n_2 ....n_k]$ random uniformly distributed individuals in the search space, where k represents the number of individuals, and where each could be a vector. Then, *t*-quantity *of* the best individuals, depending on the type of problem (minimization or maximization), will be kept and the rest will be omitted ($t = round\ \ alpha \times k$).

The main idea is that around the individuals with better function evaluation, the search must be employed more accurately. In the next step, around each t selected individuals, v new individuals will be regenerated by a Gaussian distribution. The mean value of this Gaussian distribution is equal to the old selected individual and the variance of it, proportional to old individuals' function evaluation.

The better the function evaluation is, the lower the variance of the distribution will be. When the problem is minimization, the selected function, which relates the function evaluation of each individual to their corresponding variance, is as follows:

$$\sigma_i = \frac{1}{\beta}(X_{\max} - X_{\min})\left(\frac{\min_{i=1:t}(f(n_i))}{\max_{i=1:t}(f(n_i))}\right)^{\alpha}$$
$$i = 1,2,...,t \qquad (8)$$
$$\mu_i = n_i \quad i = 1,2,....,t$$

where $X_{\max}$ is the upper bound of the solution space and $X_{\min}$ is the lower bound, $f(.)$ is the fitness function and α, β are tuning parameter (usually α is between 1 and 2). For optimizing a multi-variable function instead of a variance, a covariance matrix is needed. One of the main reasons of choosing a Gaussian distribution for a univariate function minimization algorithm is that as we move further from the mean, the probability density will decrease; this means for a set of chosen old individuals in a solution space, the chance of generating new individuals is proportional to the inverse of the distance between the chosen individuals and its neighboring ones; that is, those individuals closer to old ones have higher chances of being picked as new ones. While extending this idea to more-than-one-dimensional space, the chosen PDF for algorithm should have the same property. From figure 1-4, it can be seen that the only appropriate distribution is an MVN with a diagonal covariance, the diagonal entries of it are equal to one another; this means that the random variables of the PDF are not correlated and their distribution is jointly independent (figure 1). As depicted in figure 1, this kind of distribution is better than others, since its density contours are circles, whereas for others, the density contours are ellipses. In cases when the density contours are circular, each two points (neighboring individuals) with equal Euclidean distance from the mean (old individual) have equal probability to be chosen as new individuals. Regarding other distributions, however, there are some further points from the mean which have higher probabilities of being chosen than some closer ones. The VRGS flow chart is illustrated in figure 5, below.

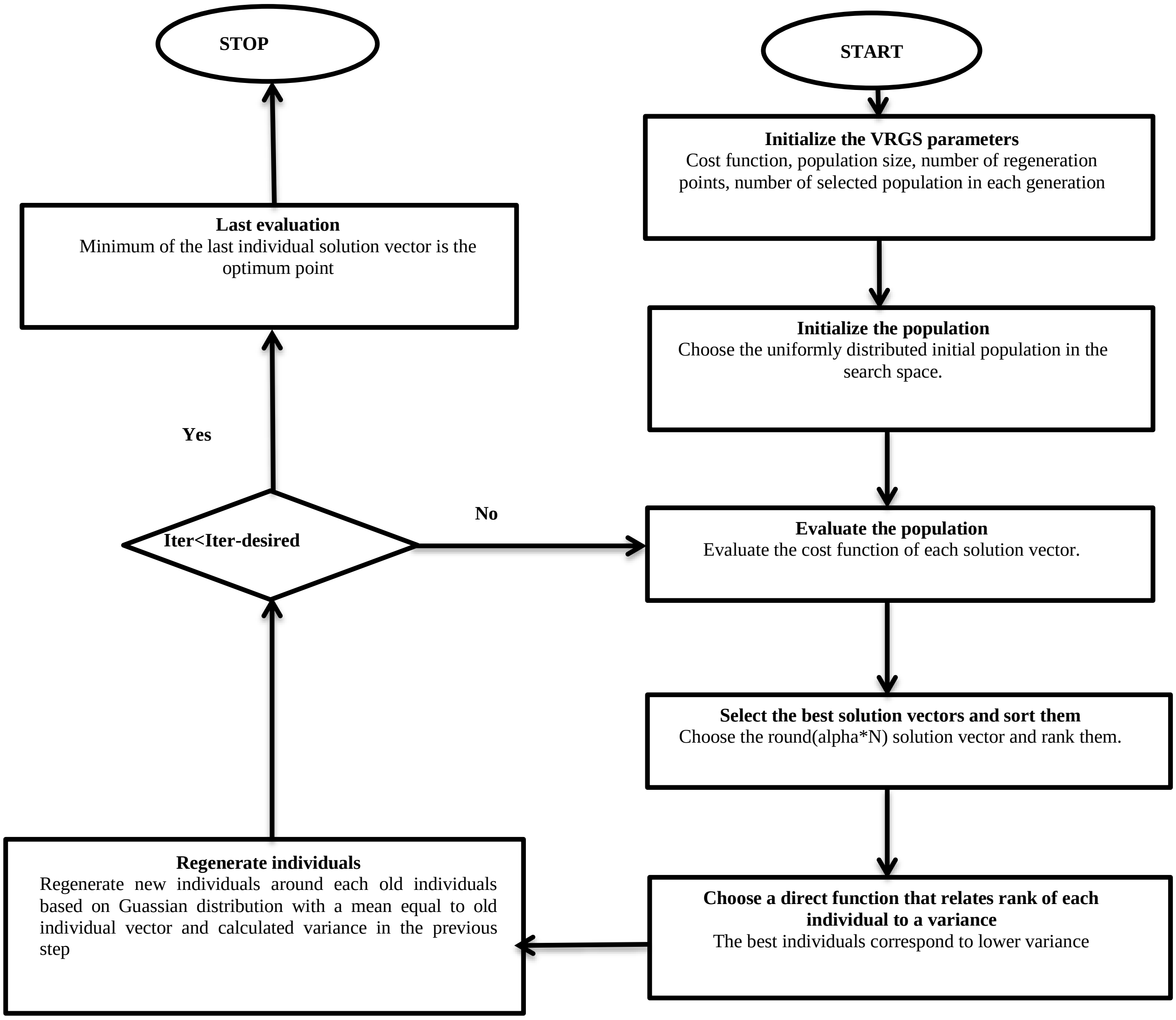

STOP
START
Initialize the VRGS parameters
Cost function, population size, number of regeneration points, number of selected population in each generation
Last evaluation
Minimum of the last individual solution vector is the optimum point
Initialize the population
Choose the uniformly distributed initial population in the search space.
Yes
No
Iter<Iter-desired
Evaluate the population
Evaluate the cost function of each solution vector.
Select the best solution vectors and sort them
Choose the round(alpha*N) solution vector and rank them.
Regenerate individuals
Regenerate new individuals around each old individuals based on Guassian distribution with a mean equal to old individual vector and calculated variance in the previous step
Choose a direct function that relates rank of each individual to a variance
The best individuals correspond to lower variance

Fig. 5. VGRS flowchart

4. SIMULATION RESULT

In this section, the VRGS algorithm is applied to four popular test functions for optimizations; figure 6 displays the 3D plot of these functions, all of which have a global minimum at $X=[0,0]$, and each of which has variable bounds same as the ones shown in figure 6. Booth and Rosenbrock functions have just one minimum but Griewank and Rastrigin have many local ones, but just one global minimum. Besides, the proposed algorithm is compared to two well-known EAs, simulated annealing and genetic algorithms. Table one depicts the results of optimization through these three algorithms, being investigated during 30 independent runs, along with a mean and variance of all results.

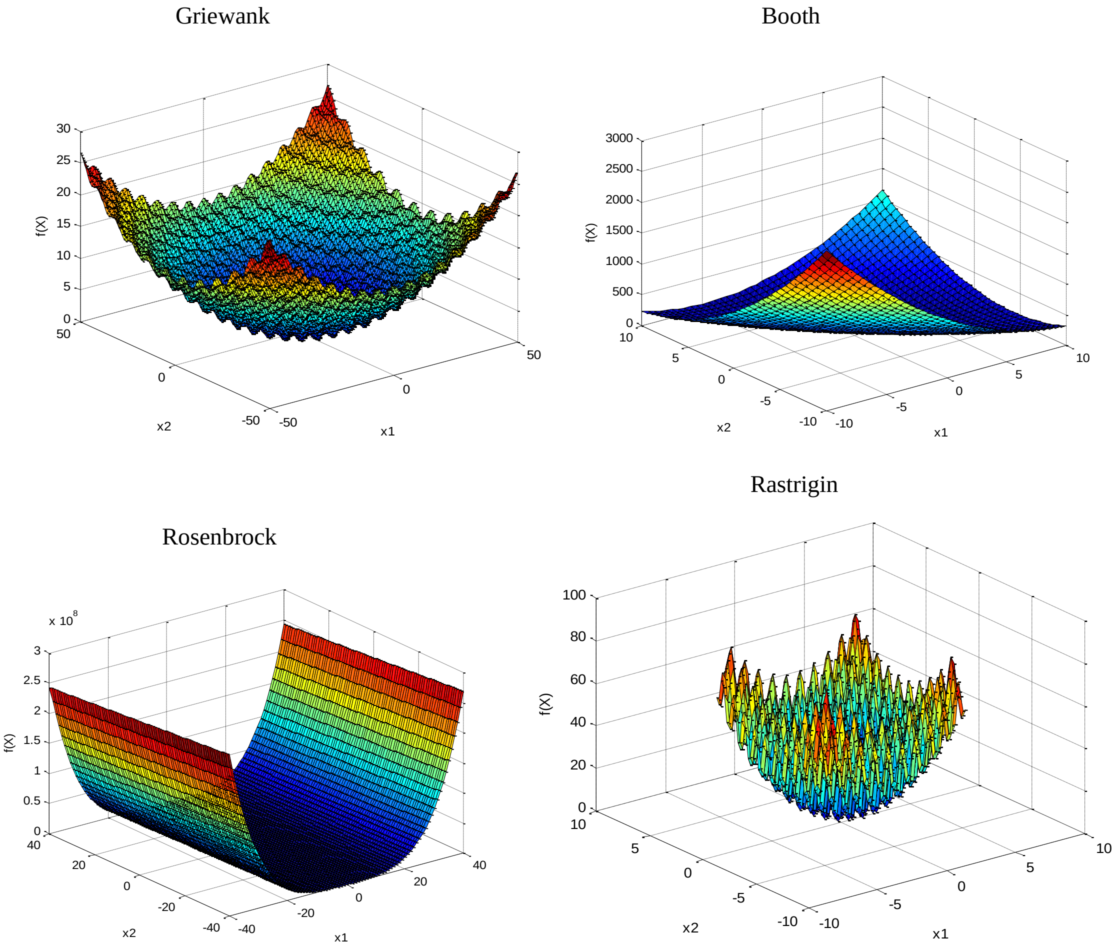
The new proposed VRGS algorithm outperformed the SA and GA for Griewank, Booth and Rosenbrock functions; nonetheless, as for Rastrigin function, although its performance outdid GA, it does not show better results in comparison to SA.

Simulations were performed by a laptop with CORE i7 2.2GHZ processor and 6G installed RAM. Table 2 compared the average calculation time needed for each run using different algorithms. Furthermore, it can be seen that in this case, the new VRGS algorithm is faster; in other words, it is approximately 4 times speedier than GA and more than 10 times quicker than SA. For all of the four functions optimization, the parameters of VRGS are as follows:

$\alpha = 1, \beta = 10$, $v = 10$ $, alpha = 0.6$, $k = 60$

$iteration = 20$

One of the major and most time consuming parts of the VRGS algorithm is generating Gaussian distributed samples. There are many algorithms available in the literature for sampling a Gaussian PDF; the most popular ones include Marsaglia polar method [10], Box-Muller transform [11] and Ziggurat algorithm [12], the last of which is one of the fastest methods used for generating such samples. For generating new individuals in VRGS, Ziggurat algorithm has been employed, a method belonging to rejection sampling ones, based on sectioning the probability density functions to a set of horizontal rectangles in order to obtain points within each. Using the same computer that ran simulations, Ziggurat algorithm is able to generate more than 6e07 samples per second which proves how fast it can be. Figure 7 illustrates the convergence of VRGS for the test functions. It should be noted that although there is no such thing as mutation in VRGS, it produces good performance for those with many local minimums. This could be due to the fact that the regeneration around individual with the worst cost function ($\max_{i=1:t}(f(n_i))$) will be done with the variance of $\frac{1}{\beta}(X_{\max} - X_{\min})$ among t selected individuals; consequently, it can easily escape local minimums and find the global one.



Griewank

Booth

Rosenbrock

Rastrigin

Fig. 6. The 3D plot of test functions for optimization

Table 1. Comparison among SA, GA and VRGS for different typical fitness functions (the mean and variance of the minimum of 30 independent simulations)

Function	SA	GA	VRGS
Griewank	6.25e-2±3.23e-3	4.07e-10±2.48e-16	4.3e-7±9.9e-13
Booth	6.62e-5±1.25e-8	4.65e-5±2.58e-8	7.62e-9±2.15e-16
Rastrigin	1.13e-8±6.34e-16	6.6e-2±1.31e-1	2.2e-6±3.1e-11
Rosenbrock	8.72±2.2e2	2.83e-3±8.98e-5	1.1e-4±4.5e-8

Table 2. Comparison between SA, GA and VRGS for different typical fitness functions (the mean of calculation time)

Function	SA	GA	VRGS
Griewank	0.81 s	0.28s	0.071s
Booth	0.9s	0.28	0.056s
Rastrigin	0.71s	0.28	0.058s
Rosenbrock	0.85s	0.28s	0.056s

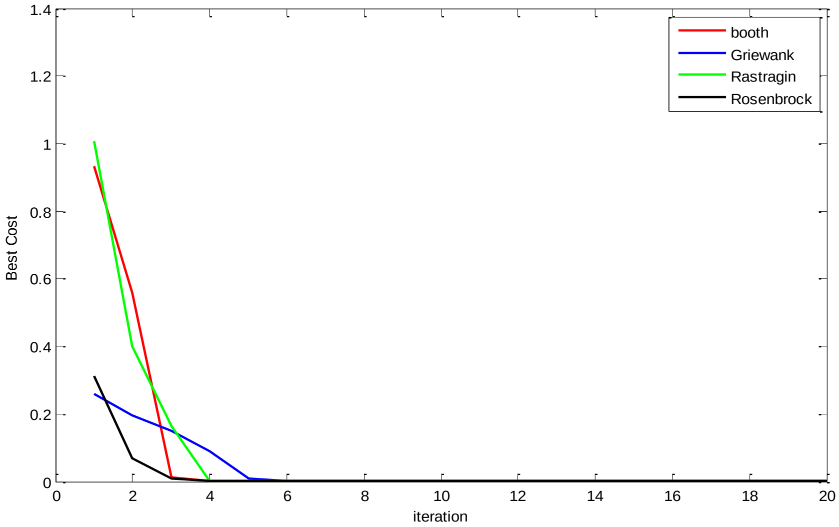


Fig. 7. Convergence of VRGS algorithm for the test functions

5. CUNCLUSION AND FUTURE WORK

This paper presents a new stochastic search for optimization based on Gaussian distribution. The algorithm was compared with GA and SA and revealed a better and faster performance. There remains much discussion about choosing the inverse function, relating the function evaluation to the variance of Gaussian distribution. Needless to say, the paper does not claim the chosen relating function has been the best; there could be other such functions, employing which in the algorithm may increase its performance. In addition, another improvement that could be exerted is the well-known elitism behavior in GA. By choosing some best points in every iteration and transferring them to the next, some surprisingly better results might be achieved.

REFERENCES

[1] Malek, R, **“Collaboration of metaheuristic algorithms through a multi-agent system,”** *in International Conference on Industrial Applications of Holonic and Multi-Agent Systems*. 2009. Springer.

[2] Ramezani, F. Lotfi, S.,**“Social-based algorithm (SBA) ,”** *Applied Soft Computing*, Vol. 13(5), pp. 2837-2856, 2013.

[3] Atashpaz-Gargari, E. Lucas, C., **“Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition, ”** *in 2007 IEEE Congress on Evolutionary Computation*, 2007, IEEE.

[4] Riedmiller, M. Braun, H., **“A direct adaptive method for faster backpropagation learning: The RPROP algorithm,”** *in Neural Networks, 1993., IEEE International Conference On*, 1993, IEEE.

[5] Tsoulos, I.G. and I.E. Lagaris, **“MinFinder: Locating all the local minima of a function,”** *Computer Physics Communications*, Vol. 174(2), pp. 166-179, 2006.

[6] Colorni, A., Dorigo, M., and Maniezzo, V., **“Distributed optimization by ant colonies,”** in *Proceedings of the first European conference on artificial life*, 1991, Paris, France.

[7] Muller, S. D., et al, **“Optimization based on bacterial chemotaxis,”** I*EEE transactions on Evolutionary Computation*, Vol. 6(1), pp. 16-29, 2002.

[8] Simon, D., **“Biogeography-based optimization,”** *IEEE transactions on evolutionary computation*, Vol. 12(6), pp. 702-713, 2008.

[9] Philipse, A. P. Maas, D., **“Magnetic colloids from magnetotactic bacteria: chain formation and colloidal stability,”** *Langmuir*, Vol. 18(25), pp. 9977-9984, 2002.

[10] Marsaglia, G. Bray, T. A., **“A convenient method for generating normal variables,”** *SIAM review*, Vol. 6(3), pp. 260-264, 1964.

[11] Box, G. E. Muller, M. E. **“A note on the generation of random normal deviates,”** T*he annals of mathematical statistics*, Vol. 29(2), pp. 610-611, 1958.

[12] Marsaglia, G. Tsang, W.W., **“The ziggurat method for generating random variables,”** *Journal of statistical software*, Vol. 5(8), pp. 1-7, 2000.