Testing MVMO on Learning-based Real-Parameter Single Objective Benchmark Optimization Problems

José L. Rueda, Senior Member, IEEE
Department of Electrical Sustainable Energy
Delft University of Technology
Delft, The Netherlands
j.l.ruedatorres@tudelft.nl

István Erlich, Senior Member, IEEE
Institute of Electrical Power Systems
University Duisburg-Essen
Duisburg, Germany
istvan.erlich@uni-due.de

Abstract— **Mean-variance mapping optimization (MVMO) is an emerging evolutionary algorithm, which adopts a single-solution based approach and performs evolutionary operations within a normalized range of the search for all optimization variables. MVMO uses a special mapping function for mutation operation, which allows a controlled shift from exploration priority at early stages of the search process to exploitation at later stages. Recently, the MVMO has been extended to a population-based and hybrid variant denoted as MVMO-SH, which includes strategies for local search and multi-parent crossover. This paper provides an study on the performance of MVMO-SH on the IEEE-CEC 2015 competition test suite on learning-based real-parameter single objective optimization. Experimental results evidence the effectiveness of MVMO-SH for successfully solving different optimization problems with different mathematical properties and dimensionality.**

Keywords—Heuristic optimization; mean-variance mapping optimization; single objective optimization; learning-based optimization.

I. INTRODUCTION

Heuristic optimization has become a proven alternative to solve optimization problems with special mathematical properties (e.g. non-convexity, discontinuity, multimodality). Remarkably, most heuristic optimization algorithms offer a conceptual simplicity and are open to further extensions. Different groups of test functions have been proposed within the context of the IEEE-CEC competitions [1], which are of great importance for evaluating and comparing the performance of modified or newly proposed algorithms. To date, the outcomes of several competitions and comparative studies on the performance of heuristic optimization algorithms highlight the potential of different emerging algorithm to become generic optimization engines in the near future [2]-[4].

Among the emerging algorithms in the Mean-variance mapping optimization (MVMO), which was initially conceived as a single-solution based approach (i.e. evolution of a single candidate solution), and whose evolutionary mechanism performs on a normalized search space (i.e. within the range [0,1]) for all optimization variables [5]. Besides, MVMO uses a solution archive as an adaptive memory to record the n-best solutions found so far. Like other state-of-art evolutionary mechanisms, MVMO adopts an elitism criterion to select the parent solution (i.e. first ranked solution in the archive), from which a child solution (offspring) is created. Besides, some dimensions of the child solution inherit the values from the parent, whereas the remaining dimensions are mutated through a special mapping function which accounts for corresponding mean and variance. The mapping function is defined in the interval [0, 1] and its shape is adapted throughout the progress of the search process in order to shift the focus from exploration (at initial stages of the process) to exploitation (at later stages). Denormalization (i.e. conversion to the original dimension) of each dimension of a candidate solution is only performed for each fitness evaluation.

Recently, MVMO has been extended to a population-based approach, which is also hybridized to include the possibility launching a local search strategy. Due to these two extensions, the new variant is hereafter denoted as MVMO-SH. Broadly speaking, each solution is evolved like in the single-solution approach, but multi-parent crossover is incorporated into the offspring creation stage in order to force the particles with worst fitness to explore other sub-regions of the search space [6]. In this paper, the performance of MVMO-SH is further evaluated by using the group of test problems defined for the IEEE-CEC 2015 competition on Learning-based Real-Parameter Single Objective Optimization. The statistical performance of MVMO-SH during the optimization repetition is investigated by considering different problem dimensions ranging from 10D up to 100D along with other predefined experimental settings from IEEE-CEC 2014 problem definitions [7].

The paper is organised as follows: Section II provides the theoretical background behind MVMO-SH. Section III shows the experimental setup and provides a discussion on numerical results of the study. Finally, concluding remarks are summarized in Section IV.

II. THE MVMO-SH PROCEDURE

The flowchart of MVMO-SH is depicted in Fig. 1 [6]. The procedure begins with an initialization stage where the algorithm parameter settings are defined and samples of the optimization variables are randomly drawn within their search boundaries for a population of N_P candidate solutions. Besides, the optimization variables are normalized at this stage, that is, the range of the search space for all variables is transformed from [min, max] to [0, 1] range. This is a precondition for the subsequent mutation operation via mapping function; on top of this, it guarantees that the generated offspring will never violate the search boundaries. The optimization variables are only de-normalized when performing fitness evaluation or launching local search. The core of the algorithm is contained in the inner loop of the flowchart, in which, after execution of fitness evaluation or local search for a given candidate solution, updating the solution archive, fitness based classification of candidate solutions into good or bad solutions, parent selection and offspring generation are performed. The process is terminated upon completion of a specified number of iterations.

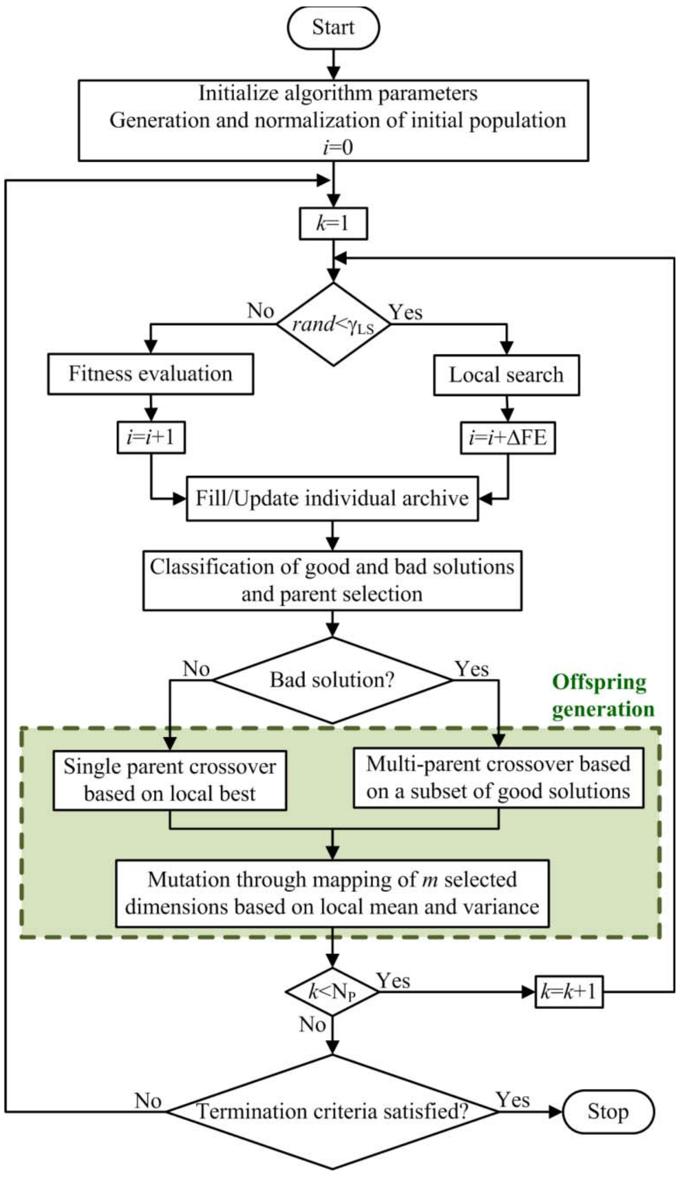
Start
Initialize algorithm parameters
Generation and normalization of initial population
i=0
k=1
No
$rand<\gamma_{LS}$
Yes
Fitness evaluation
Local search
i=i+1
i=i+ΔFE
Fill/Update individual archive
Classification of good and bad solutions and parent selection
No
Bad solution?
Yes
Offspring generation
Single parent crossover based on local best
Multi-parent crossover based on a subset of good solutions
Mutation through mapping of m selected dimensions based on local mean and variance
$k<N_P$
Yes
k=k+1
No
No
Termination criteria satisfied?
Yes
Stop

Fig. 1. Flowchasrt of MVMO-SH. The fitness evaluation and candidate solution counters are denoted by i and k, whereas N_P, ΔFE, and *rand* stand for number of candidate solutions, number of fitness evaluations, and uniform random number between [0, 1], respectively.

A. Fitness evaluation and local search

The elements of the candidate solution array (i.e. optimization variables) are de-normalized from [0, 1] range to their original [min, max] boundaries before fitness evaluation or local search is performed. Besides, since the learning-based test functions of the IEEE-CEC 2015 competition are exclusively bound constrained, no constraint handling strategy is needed, so the fitness value associated to the candidate solution corresponds with the computed error value [6].

Interior-point method (IPM) is included in this step as a local improvement option. After a given number of fitness evaluations, local search is performed with a probability γ_{LS} for any child solution, that is

$$rand < \gamma_{LS} \tag{1}$$

and runs in the range

$$\alpha_{LS\_min} < \alpha < \alpha_{LS\_max}, \quad \alpha = i / i_{max} \tag{2}$$

where i denotes fitness evaluation number, and *rand* is a random number with uniform distribution in [0,1].

B. Fill/update of the solution archive

Each candidate solution has a compact and continually updated solution archive associated to it, which stores its n-best offsprings in a descending order of fitness and serves as the knowledge base for guiding the search direction (cf. Fig. 2) [6].

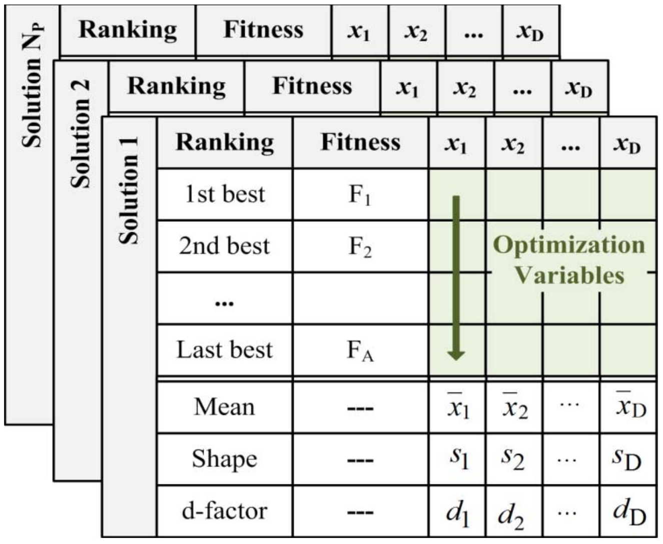

Fig. 2. Structure of the solution archives.

The archive size is fixed for the entire process. For each candidate solution, and after every execution of fitness evaluation/local search, an update of its archive takes place only if the child solution is better than those in the archive. Besides, the mean $\bar{x}_i$, shape s_i, and d-factor d_i associated to each optimization variable are recalculated whenever an update of the archive takes place. However, for the calculation of mean a weighting of the old and new values has been used according to (3)

$$\bar{x}_{update} = 0.2 * \bar{x}_{new} + 0.8 * \bar{x}_{old} \tag{3}$$

That means, the initial value $\bar{x}_{ini}$ represents an additional tuning parameter to be provided by the user. Otherwise it will be generated randomly. Similarly, the initial values of s_i, and d-factor d_i can be defined by the user, but, unlike the mean values $\bar{x}_i$, they will be updated based on the archive only without weighting old and current values. All these parameters influence the change of the shape of the mapping function, which is allows changing the search emphasis from exploration to exploitation. `

C. Selection of parents

In the early stage of the search process, each candidate solution is independently drawn for at least two function evaluations, and the solution that produced the individual best fitness achieved so far (i.e. the one corresponding to the first ranked position in its particular solution archive) is chosen as the parent for the next offspring. Next, the procedure shown in Fig. 3, is used for parent assignment in order to encourage the candidate solutions having poor performance (in terms of individual achieved best fitness) to explore other sub-regions of the search space. The candidate solutions are classified into the set of *GP* "good solutions", and Np-*GP* "bad solutions" based on the individual best fitness. Individual best-based parent assignment is adopted for each solutions classified as good, whereas for each bad solution $\mathbf{x}_k$, the parent $\mathbf{x}_k^{\text{parent}}$ is determined by using the following multi-parent criterion:

$$\mathbf{x}_k^{\text{parent}} = \mathbf{x}_{\text{RG}}^{\text{best}} + \beta\left(\mathbf{x}_{\text{GB}}^{\text{best}} - \mathbf{x}_{\text{LG}}^{\text{best}}\right) \tag{4}$$

where $\mathbf{x}_{\text{GB}}^{\text{best}}$, $\mathbf{x}_{\text{LG}}^{\text{best}}$ and $\mathbf{x}_{\text{RG}}^{\text{best}}$ represent the first (global best), the last, and a randomly selected intermediate solution in the group of good solutions, respectively. The vector of mean values associated to $\mathbf{x}_k$, which are required later for mutation and mapping by the mapping function is also set to $\mathbf{x}_k^{\text{parent}}$. The factor β is a random number, which is determined from:

$$\beta = 2.5\left(rand + 0.25\cdot\alpha^2 - 0.5\right) \tag{5}$$

β is re-drawn and (5) is recalculated for any element of $\mathbf{x}_k^{\text{parent}}$ going outside the range [0, 1]. For the class of "good solutions" the mean used for mapping is selected randomly from the same group.

The relative number *GP* of solutions belonging to the group of good solutions is dynamically determined throughout the search process as follows:

$$GP = \text{round}\left(\text{N}_\text{P}\cdot g_p^{\,*}\right) \tag{6}$$

$$g_p^{\,*} = \text{g}_{\text{p\_ini}}^* - \alpha^2\left(\text{g}_{\text{p\_final}}^* - \text{g}_{\text{p\_ini}}^*\right) \tag{7}$$

Equation (7) is not calculated in the initial stage of the search process, where each solution is evaluated independently. Note that *GP* is linearly narrowed down following the decrease from $\text{g}_{\text{p\_ini}}^*$ to $\text{g}_{\text{p\_final}}^*$.

D. Crossover

For the next generation, a child vector (array) $\mathbf{x}^{\text{new}} = \left[x_1, x_2, x_3, \ldots, x_\text{D}\right]$, where D is the number of problem dimensions, is created for each solution by combining a subset of D-*m* directly inherited dimensions from $\mathbf{x}_\text{p}^{\text{parent}}$ (i.e. crossover) and m selected dimensions that undergo mutation operation through mapping function based on the actual values of the parameters $\bar{x}_i$, s_i, and d_i associated to each solution. The number *m* of dimensions to be selected for mutation operation is progressively decreased as follows:

$$m = \text{round}\left(\text{m}_{\text{final}} + irand\left(m^* - \text{m}_{\text{final}}\right)\right) \tag{8}$$

$$m^* = \text{round}\left(\text{m}_{\text{ini}} - \alpha^2\left(\text{m}_{\text{ini}} - \text{m}_{\text{final}}\right)\right) \tag{9}$$

Where *irand* represents a random integer in the range of zero and the value given in the brackets. The selection of the *m* variables to be mutated can be done by using any of the strategies given in [13].

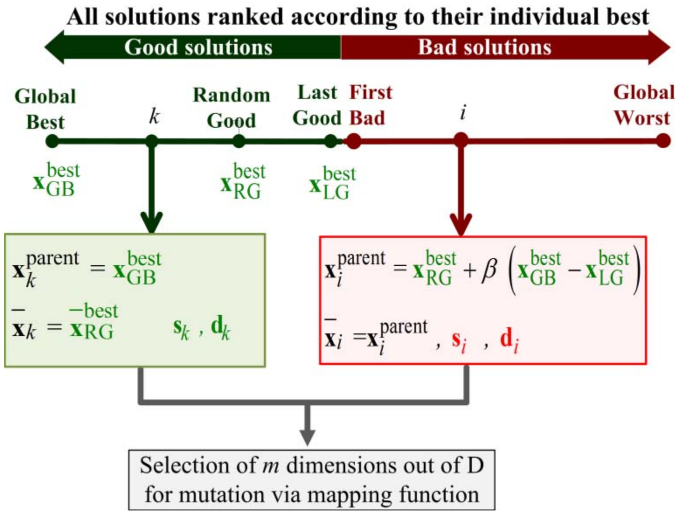
All solutions ranked according to their individual best
Good solutions
Bad solutions
Global Best
k
Random Good
Last Good
First Bad
i
Global Worst
$\mathbf{x}_{\text{GB}}^{\text{best}}$
$\mathbf{x}_{\text{RG}}^{\text{best}}$
$\mathbf{x}_{\text{LG}}^{\text{best}}$
$\mathbf{x}_k^{\text{parent}} = \mathbf{x}_{\text{GB}}^{\text{best}}$
$\bar{\mathbf{x}}_k = \bar{\mathbf{x}}_{\text{RG}}^{\text{best}}$
$\mathbf{s}_k$, $\mathbf{d}_k$
$\mathbf{x}_i^{\text{parent}} = \mathbf{x}_{\text{RG}}^{\text{best}} + \beta\left(\mathbf{x}_{\text{GB}}^{\text{best}} - \mathbf{x}_{\text{LG}}^{\text{best}}\right)$
$\bar{\mathbf{x}}_i = \mathbf{x}_i^{\text{parent}}$, $\mathbf{s}_i$, $\mathbf{d}_i$
Selection of *m* dimensions out of D for mutation via mapping function

Fig. 3. Procedure for parent selection in MVMO-SH.

E. Mutation

The new value of each selected dimension x_r of $\mathbf{x}^{\text{new}}$ is determined by

$$x_\text{r} = h_\text{x} + (1 - h_1 + h_0)\cdot x_\text{r}^* - h_0 \tag{10}$$

where x_r^* is a randomly generated number with uniform distribution between [0, 1], and the term *h* (subscripts specified below) represents the transformation mapping function defined as follows.

$$h(\bar{x}, s_1, s_2, x) = \bar{x}\cdot(1 - e^{-x\cdot s_1}) + (1 - \bar{x})\cdot e^{-(1-x)\cdot s_2} \tag{11}$$

h_x, h_1 and h_0 are the outputs of the mapping function calculated for

$$h_\text{x} = h(x = x_\text{r}^*),\quad h_0 = h(x = 0),\quad h_1 = h(x = 1) \tag{12}$$

Thus, x_r is always within the range [0, 1]. s_r is the shape factor calculated as follows:

$$s_\text{r} = -\ln(v_\text{r})\cdot f_\text{s} \tag{13}$$

where v_r is the variance computed from the stored values of x_r in the solution archive, and f_s is a scaling factor.

In the first evaluation of every solution, the mean $\bar{x}_r$ corresponds with the initial value of x_r and the variance v_r is set to 1.0 which corresponds with s_r=0. But as the optimization progresses, they are recalculated after every update of the particle's solution archive for each selected optimization variable. Both input and output of the mapping function cover the range [0, 1]. From (11), it can be noticed that the shape of the mapping function is influenced by the mean $\bar{x}_r$ and shape factors s_{r1} and s_{r2} . So, the search diversity can be enhanced through proper variation of the shape factors. To this aim, the scaling factor f_s can be additionally used to change the shape of the function. Thus, f_s is increased as the optimization progresses from a small initial value (e.g. $f^*_{s\_ini} = 1$) up to a higher final value (e.g. $f^*_{s\_final} = 2$) by using (14) and (15):

$$f_s = f_s^* \left(1 + \alpha^2 \cdot rand\right) \tag{14}$$

$$f_s^* = f^*_{s\_ini} + \alpha\left(f^*_{s\_final} - f^*_{s\_ini}\right) \tag{15}$$

The shape factors s_{r1} and s_{r2} of the variable x_r are assigned by using the procedure given in (15).

```
$s_{i1} = s_{i2} = s_i$
if $s_i > 0$ then
    $\Delta d = (1 + \Delta d_0) + 2.5 \cdot \Delta d_0 \cdot (rand - 0.5)$
    if $s_i > d_i$
        $d_i = d_i \cdot \Delta d$
    else
        $d_i = d_i / \Delta d$
    end if
    if $d_i > s_i$ then
        $s_a = d_i$;   $s_b = s_i$
    else
        $s_a = s_i$;   $s_b = d_i$
    end if
    if $rand < 0.5$ then
        $s_{i1} = s_a$;   $s_{i2} = s_b$
    else
        $s_{i1} = s_b$;   $s_{i2} = s_a$
    end if
end if                                                    (16)
```

Initially, the value of d_i is set to 1 for all variables. Thereafter, each d_i is scaled up or down over the iterations by the factor Δd , which is randomly varied between 1 and $1 + 2 \cdot \Delta d_0$. If $d_i > s_i$, the current d_i is divided by Δd which is always larger than 1.0 and thus leads to reduced value of d_i. In case $d_i < s_i$, d_i will be multiplied by Δd resulting in increased d_i. In this way d_i will always oscillate around the current shape factor s_i. The assignment of s_i and d_i to s_{i1} or s_{i2} depends on the actual values of s_i and d_i and $\bar{x}_i$.

It is recommended to take a non-zero value equal to or smaller than 0.4 for Δd_0. A high value of Δd_0 will indirectly entail wider global search diversification over the entire space whereas a smaller one would lead to concentrated local search aiming at accuracy improvement. By using the described procedure the asymmetric characteristic of the mapping function is also exploited by using different values for si1 and s_{i2} leading to enhanced search performance and zero variance handling. Zero variance can occur when all values of xi in the archive are identical. In this case the previous non-zero value can be used further. However, this value may result, under circumstances, in stagnating convergence behaviour. The procedure overcomes this problem as the mean and variance are calculated only for non-identical values of xi saved in the archive.

The mean and variance are not calculated before a certain number of solutions are available in the archive. The authors usually start the calculation immediately after two solutions have been obtained. However, it is possible to wait until the archive is filled up completely which will result in more robust initial solutions. In this stage, the search is performed with s_{i1}=s_{i2}=0 which corresponds with a straight line between zero and one as the mapping function. The mean value in this case does not have any effect on the mapping function.

F. Parameter tuning

In summary, the following set of parameters of MVMO-SH should be optimized for each test function of the IEEE-CEC 2015 test suite:

P1: Number of particles
P2: Archive size
P3: Initial-final *fs* factor ($f^*_{s\_ini} - f^*_{s\_final}$)
P4: Initial-final ratio of good particles ($g^*_{p\_ini} - g^*_{p\_final}$)
P5: Initial-final number of mutated variables ($m_{ini} - m_{final}$)
P6: Local search probability (γ_{LS})
P7: Initial-final bound of range of local search probability ($\alpha_{LS\_min} - \alpha_{LS\_max}$)
P8: initial value of the shape factor s_i and dynamic shape factor d_i
P9: initial value of the mean value of optimization variables save in the archive

For each test function, the algorithm is tuned by performing sensitivity analysis of the achieved fitness value under a single parameter change within 10 independent optimization runs. The execution of local search may require performing tens, hundreds or even thousands of fitness evaluations. Thus, the use of this option is recommended for optimization problems that can be solved without considerable computing time concerns (i.e. large number of fitness evaluation budget), whereas, for optimization problems to be solved within reduced time (i.e. limited amount of function evaluations), the value of γ_{LS} should be set to zero in order to give preference to the underlying evolutionary mechanism MVMO-SH.

G. Further development of MVMO-SH

This section outlines some prospective future work on possible extensions of MVMO-SH, which will be further tested and presented in a future publication.

First, it is worth pointing out that other alternative methods can be incorporated into the core MVMO-SH to perform local search. Among these are classical methods like Sequential Quadratic Programming (SQP) and heuristic methods like Hill. The parameter γ_{LS} can also be defined to vary over the iterations to account for accuracy targets and computing budget constraints.

Instead of linear decrease, GP could be reduced quadratically in order to pursue a progressive selection of a smaller set of the most successful particles while having more particles with poor performance being forced to visit other unexplored and attractive sub-regions. Similarly, another decreasing strategy can be defined for f_s and m in order to better contribute to gradually turning the emphasis of the search from exploration to intensification (exploitation) in the region where the parent is located.

III. Numerical Tests

Numerical experiments were performed on a computer with Intel® Core™ i7-3770 CPU, 3.4 GHz and 16 GB RAM, under Windows 8.1 pro, 64 bit OS. The implementation of MVMO was done in Matlab® Version R2013b and the functionalities of the Parallel Computing Toolbox are used to set a cluster with 7 cores to perform the optimization trials in a distributed manner. Stochastic integrity is guaranteed by performing independent initialization of random number streams on individual processes with respect to time plus the process identifier.

A. Experimental setting

The IEEE-CEC 2015 learning-based benchmark problems listed in TABLE I. are used to evaluate the performance of MVMO-SH. The details of these problems, which are treated as black-box problems for the competition, are given in [8]. Statistical tests on convergence performance and quality of final solution provided by MVMO-SH were performed under the following considerations:

- Problem dimension D = 10, 30, 50, 100.
- Search range: $[-100,100]^D$.
- Max. number of function evaluations: 10000*D.
- Optimization trials per problem: 51.
- Uniform random initialization within the search space. The random seed is based on time, which is done using the command rand('state', sum(100*clock)) in Matlab environment.
- The objective function is defined as the error value $OF=TF_i(x)-F_i^*$, where F_i^* is the theoretical global optimum of the i-th benchmark function TF given in TABLE I. . The values of OF smaller than 1E-08 are taken as zero.
- The optimization is terminated upon completion of the maximum number of function evaluations.

TABLE I. IEEE-CEC 2015 Expensive Optimization Problems

Type	No.	Description	F_i*
Unimodal function	TF1	Rotated High Conditioned Elliptic Function	100
	TF2	Rotated Bent Cigar Function	200
Simple Multimodal functions	TF3	Shifted and Rotated Ackley's Function	300
	TF4	Shifted and Rotated Rastrigin's Function	400
	TF5	Shifted and Rotated Schwefel's Function	500
Hybrid function	TF6	Hybrid Function 1 (N=3)	600
	TF7	Hybrid Function 2 (N=4)	700
	TF8	Hybrid Function 3 (N=5)	800
Composition function	TF9	Composition Function 1 (N=3)	900
	TF10	Composition Function 2 (N=3)	1000
	TF11	Composition Function 3 (N=5)	1100
	TF12	Composition Function 4 (N=5)	1200
	TF13	Composition Function 5 (N=5)	1300
	TF14	Composition Function 6 (N=7)	1400
	TF15	Composition Function 7 (N=10)	1500

The MVMO parameters P1 to P9 were empirically tuned (trial and error basis) by sequentially changing only one parameter at a time and running MVMO for 4-5 independent repetitions for each function. A systematic tuning method is under development and will be introduced in a future publication. The parameters used for optimizing 10D, 30D, 50D and 100D dimensions are included and highlighted in the

Matlab source code, which will be published after the competition.

B. *Performance statistics and discussion*

The results provided in the appendix paper for 10D, 30D, 50D and 100D were obtained by using different tunings for each function based on the typical values suggested in [6].

The statistical attributes of the error value OF (i.e. best, worst, mean, median, and standard deviation values), which were calculated after 51 runs, are summarized in TABLE II. to TABLE V. in the Appendix, for each dimension. In addition, and for each problem dimensionality, the computational complexity measures for MVMO-SH, as defined in [8], are given in TABLE VI in the Appendix. The following remarks can be deduced from these results:

- *Solving unimodal functions:* For all dimensions, MVMO-SH is capable of finding close to zero error values (i.e. smaller than 1E-08) for OF in all runs. Besides, it was found out that convergence to these values was achieved short after the local search call. MVMO-SH performs in this case just the global adjustment and the interior-point method used as local search function improves the optimization until zero error is reached. Therefore, MVMO-SH constitutes a powerful tool to effectively tackle unimodal problems irrespective of dimensionality and the underlying mathematical features (e.g. asymmetrical, separable/non-separable).

- *Solving simple multimodal functions*: For TF3 most of the solutions converge to the local minimum around 20. However, in case of 10-D almost 50% of the solutions reaches the global minimum of zero. TF4 and TF5 represent harder to solve problems. Nevertheless, good solutions have been achieved for all dimensions. The variance among the 51 runs is the largest for TF5.

- *Solving hybrid functions*: Even though the problems here are more challenging MVMO-SH is also capable of providing near to zero error values (in the order of 10^{-2}) for all functions in almost all runs. The errors are in the order of 10^0, 10^1 and 10^2 for all dimensions in most optimization repetitions.

- *Solving composition functions*: For the TF13, the proposed algorithm is capable of providing small fitness values in the range of 10^{-2} for all dimensions. For the remaining functions the errors are in the order of 10^2, and 10^4. TF15 converges always to the local minimum of 100.

IV. Conclusions

This paper presents an analysis of the performance of MVMO-SH on 15 novel benchmark functions belonging to the IEEE-CEC 2014 competition test suite on Learning-based Real-Parameter Single Objective Optimization. Basically, MVMO-SH conducts the search process by evolving a population of candidate solutions, each having its own memory, which is represented by the associated solution archive and mapping function. Besides, a fitness-based classification is performed to discriminate between good solutions (e.g. smaller fitness values) and bad solutions (e.g. higher fitness values). For each good solution, the parent assignment is done by considering the first ranked solution in its particular knowledge archive, whereas a multi-parent crossover is used to reorient each bad solution towards different sub-regions of the search space. Additionally, it offers the possibility of launching an interior-point method based strategy for local improvement purpose. Numerical results evidence the effectiveness of MVMO as for tackling different optimization problems

Further research effort is being focused on the study of other types of local search strategies. Performance comparisons with other state-of-art heuristic optimization algorithms, along with application of MVMO-SH to different large-scale power system optimization problems, are also currently under investigation.

References

[1] P.N. Suganthan, "Testing Evolutionary Algorithms on Real‐World Numerical Optimization Problems". [Online]. Available at: http://www.ntu.edu.sg/home/epnsugan/

[2] S. Das, and P.N. Suganthan,"Differential Evolution: A Survey of the State-of-the-Art," IEEE Transactions on Evolutionary Computation, vol. 15, no. 1, pp. 4-31, Feb. 2011.

[3] S.M.Elsayed, R.A.Sarker, and D.L. Essam, "A new genetic algorithm for solving optimization problems," Engineering Applications of Artificial Intelligence, vol. 27, pp. 57–69, Jan. 2014.

[4] D. Simon, Evolutionary optimization algorithms: Biologically inspired and population-based approaches to computer intelligence. Hoboken: John Wiley & Sons, 2013.

[5] I. Erlich, G. K. Venayagamoorthy, and W. Nakawiro, "A mean-variance optimization algorithm," 2010 IEEE Congress on Evolutionary Computation, pp.1-6, July 2010.

[6] I. Erlich, J.L. Rueda, and S. Wildenhues, "Evaluating the Mean-Variance Mapping Optimization on the IEEE-CEC 2014 Test Suite", in Proc. 2014 IEEE World Congress on Computational Intelligence, March 2014.

[7] J.J. Liang, B.Y. Qu, and P.N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization," Technical Report, Nanyang Technological University (Singapore) and Zhengzhou University (China), Dec. 2013. [Online]. Available at: http://www.ntu.edu.sg/home/epnsugan/

[8] J.J. Liang1, B.Y. Qu, and P.N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-based Real-Parameter Single Objective Optimization," Technical Report, Nanyang Technological University (Singapore) and Zhengzhou University (China), Nov. 2014. [Online]. Available at: http://www.ntu.edu.sg/home/epnsugan/

APPENDIX

TABLE II. RESULTS FOR 10D

Type	Function	Best	Worst	Median	Mean	Std.
Unimodal function	TF1	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
	TF2	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
Simple Multimodal functions	TF3	0.000E+00	2.000E+01	2.000E+01	1.091E+01	1.044E+01
	TF4	9.950E-01	5.970E+00	1.990E+00	2.261E+00	1.729E+00
	TF5	3.665E+00	1.218E+02	1.183E+01	2.201E+01	3.388E+01
Hybrid function	TF6	2.081E-01	4.000E+01	1.411E+00	5.893E+00	1.182E+01
	TF7	2.683E-02	1.236E-01	3.654E-02	5.795E-02	3.271E-02
	TF8	3.611E-03	6.605E-01	3.196E-01	2.467E-01	2.153E-01
Composition function	TF9	1.002E+02	1.003E+02	1.002E+02	1.002E+02	4.040E-02
	TF10	2.166E+02	2.167E+02	2.167E+02	2.167E+02	4.014E-02
	TF11	2.158E+00	3.001E+02	3.567E+00	8.412E+01	1.387E+02
	TF12	1.004E+02	1.008E+02	1.007E+02	1.007E+02	1.243E-01
	TF13	3.042E-02	3.053E-02	3.042E-02	3.044E-02	4.260E-05
	TF14	1.000E+02	1.000E+02	1.000E+02	1.000E+02	0.000E+00
	TF15	1.000E+02	1.000E+02	1.000E+02	1.000E+02	0.000E+00

TABLE III. RESULTS FOR 30D

Type	Function	Best	Worst	Median	Mean	Std.
Unimodal function	TF1	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
	TF2	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
Simple Multimodal functions	TF3	2.000E+01	2.000E+01	2.000E+01	2.000E+01	5.424E-04
	TF4	3.980E+00	1.990E+01	8.955E+00	9.540E+00	3.538E+00
	TF5	5.515E+02	1.650E+03	1.156E+03	1.141E+03	2.813E+02
Hybrid function	TF6	4.348E+01	9.289E+02	2.769E+02	3.102E+02	1.794E+02
	TF7	2.393E+00	5.337E+00	3.242E+00	3.408E+00	7.585E-01
	TF8	1.174E+01	5.093E+02	5.153E+01	8.142E+01	8.390E+01
Composition function	TF9	1.027E+02	1.036E+02	1.031E+02	1.031E+02	1.596E-01
	TF10	3.991E+02	1.057E+03	6.458E+02	6.513E+02	1.369E+02
	TF11	3.002E+02	3.012E+02	3.009E+02	3.009E+02	1.467E-01
	TF12	1.031E+02	1.044E+02	1.037E+02	1.037E+02	3.331E-01
	TF13	2.576E-02	2.917E-02	2.702E-02	2.710E-02	9.009E-04
	TF14	3.122E+04	3.240E+04	3.150E+04	3.157E+04	3.021E+02
	TF15	1.000E+02	1.000E+02	1.000E+02	1.000E+02	0.000E+00

TABLE IV. RESULTS FOR 50D

Type	Function	Best	Worst	Median	Mean	Std.
Unimodal function	TF1	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
	TF2	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
Simple Multimodal functions	TF3	2.000E+01	2.000E+01	2.000E+01	2.000E+01	5.497E-06
	TF4	3.482E+01	8.656E+01	5.671E+01	5.882E+01	1.171E+01
	TF5	1.600E+03	3.825E+03	2.719E+03	2.833E+03	5.345E+02
Hybrid function	TF6	7.039E+02	1.941E+03	1.481E+03	1.414E+03	2.970E+02
	TF7	5.134E+00	8.112E+01	4.172E+01	3.545E+01	2.501E+01
	TF8	2.950E+02	1.386E+03	8.687E+02	8.716E+02	2.242E+02
Composition function	TF9	1.040E+02	1.050E+02	1.045E+02	1.045E+02	1.978E-01
	TF10	1.021E+03	2.563E+03	1.408E+03	1.425E+03	2.746E+02
	TF11	3.008E+02	5.014E+02	3.020E+02	3.256E+02	6.041E+01
	TF12	1.057E+02	2.003E+02	1.064E+02	1.102E+02	1.839E+01
	TF13	8.805E-02	1.349E-01	1.041E-01	1.065E-01	1.005E-02
	TF14	4.952E+04	7.227E+04	4.952E+04	5.080E+04	5.157E+03
	TF15	1.000E+02	1.000E+02	1.000E+02	1.000E+02	0.000E+00

TABLE V. RESULTS FOR 100D

Type	Function	Best	Worst	Median	Mean	Std.
Unimodal function	TF1	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
	TF2	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00
Simple Multimodal functions	TF3	2.000E+01	2.000E+01	2.000E+01	2.000E+01	6.026E-07
	TF4	1.184E+02	2.408E+02	1.642E+02	1.660E+02	2.128E+01
	TF5	7.211E+03	1.228E+04	1.007E+04	9.928E+03	1.124E+03
Hybrid function	TF6	2.626E+03	4.403E+03	3.727E+03	3.665E+03	3.929E+02
	TF7	2.066E+01	1.383E+02	2.617E+01	4.578E+01	3.823E+01
	TF8	1.355E+03	2.681E+03	2.002E+03	1.991E+03	3.408E+02
Composition function	TF9	1.071E+02	1.086E+02	1.079E+02	1.079E+02	3.722E-01
	TF10	2.663E+03	7.084E+03	3.763E+03	3.908E+03	7.785E+02
	TF11	3.025E+02	1.429E+03	1.115E+03	1.010E+03	3.286E+02
	TF12	1.125E+02	2.004E+02	1.139E+02	1.257E+02	3.010E+01
	TF13	8.532E-02	1.334E-01	1.044E-01	1.063E-01	1.156E-02
	TF14	1.088E+05	1.090E+05	1.089E+05	1.089E+05	2.834E+01
	TF15	1.000E+02	1.027E+02	1.007E+02	1.008E+02	8.827E-01

TABLE VI. COMPUTATIONAL COMPLEXITY

Dimension	T_0	T_1	$\hat{T}_2$	$(\hat{T}_2 - T_1)/T_0$
D=10	1.0920070e-01	5.7205567e+01	6.0930871e+01	3.4114283e+01
D=30	1.2480080e-01	1.4659414e+02	1.4955504e+02	2.3724998e+01
D=50	1.2480080e-01	2.3262869e+02	2.4102779e+02	6.7300010e+01
D=100	1.0920070e-01	4.7962507e+02	4.7397784e+02	-5.1714245e+01