# A Technique for the Visualization of Population–Based Algorithms

K.E. Parsopoulos, V.C. Georgopoulos and M.N. Vrahatis

***Abstract*— A technique for the visualization of stochastic population–based algorithms in multidimensional problems with known global minimizers is proposed. The technique employs projections of the populations in the 2–dimensional vector space spanned by the two extremal eigenvectors of the Hessian matrix of the objective function at a global minimizer. This space condenses information regarding the shape of the objective function around the given minimizer. The proposed approach can provide intuition regarding the behavior of the algorithm in unknown high–dimensional problems. It also provides an alternative visualization framework for problems of any dimension, which alleviates drawbacks of the most popular projection methods. The proposed technique is illustrated for three well–known population–based algorithms, namely, Differential Evolution, Covariance Matrix Adaptation Evolution Strategies and Particle Swarm Optimization, on three test problems of different dimensionality.**

## I. Introduction

VISUALIZATION is the process of converting numbers into pictures [1], [2]. Visualizing the evolution of the population as well as the trajectories of specific individuals, such as the best individual, during the evolution process of a stochastic population–based algorithm may provide intuition regarding the algorithm's dynamics and rate of convergence. This information is considered very important for the development of more efficient variants of the algorithm, as well as for the selection of the most proper algorithm for a given class of problems. Thus, visualization techniques are very useful tools for the design and evaluation of algorithms.

In contrast to low–dimensional cases where visualization is simple, high–dimensional cases exhibit difficulties and require special treatment. The most trivial technique for data visualization in an $n$–dimensional search space, $S \subset \mathbb{R}^n$, is the projection of the original vectors, $x = (x_1, x_2, \ldots, x_n)^\top \in S$, on a 2–dimensional subspace, $S' \subset \mathbb{R}^2$, by setting $(n-2)$ arbitrarily selected components of $x$ equal to fixed values and varying the remaining ones. Although simple, this approach does not guarantee the reliability of the obtained information regarding the algorithm's performance and dynamics, due to its heavy dependence on the choice of the subspace $S'$, i.e., the choice of the components that will vary. Indeed, the projections of two distinct $n$–dimensional points, $y, z \in S$, may be located very near to each other in a projection space $S'$, and very far in a different projection space, $S''$, regardless of their actual distance in $S$. Thus, the lack of information regarding the choice of the most proper projection space may induce misleading conclusions regarding the behavior of the algorithm, and, although in low–dimensional spaces all the possible combinations of the projections can be drawn and evaluated, in high–dimensional cases this is computationally expensive and time consuming.

In this paper, a technique for the visualization of stochastic population–based optimization algorithms in any dimension, is proposed [3]. This technique visualizes the individuals of a population by projecting them on the 2–dimensional vector space spanned by the extremal eigenvectors of the Hessian matrix of the objective function at a global minimizer of the objective function. The choice of the specific vector space is based on its properties regarding the shape of the objective function locally around the minimizer. The proposed approach has already been applied for the visualization of gradient–based methods with very promising results [3]–[5]. It also provides a unified framework for the visualization of population–based algorithms in any dimension, exposing the behavior of the algorithm in problems with known characteristics and minimizers, which can be a very useful hint regarding its performance in similar unknown problems.

The rest of the paper is organized as follows: the proposed technique is described in Section II, while Section III is devoted to a brief description of the algorithms that are employed to illustrate its workings. The results of the application of the proposed approach are reported in Section IV, and the paper concludes in Section V.

## II. The Proposed Visualization Technique

Consider the unconstrained optimization problem (without loss of generality only the minimization case is considered),

$$\min_{x \in \mathcal{D} \subset \mathbb{R}^n} f(x),$$

and assume that

$$x^* = \arg\min_{x \in \mathcal{D}} f(x),$$

is a global minimizer of $f$, i.e.,

$$f(x^*) \leqslant f(x), \quad \forall x \in \mathcal{D}.$$

Let $\nabla f(x)$ and $\nabla^2 f(x)$ be the gradient and the Hessian matrix, respectively, of the objective function $f$ at the point $x = (x_1, x_2, \ldots, x_n)^\top$. Also, let $\lambda_{\min}$ and $\lambda_{\max}$ be the minimum and maximum eigenvalue of $\nabla^2 f(x^*)$, respectively, and, $e_{\min}$, $e_{\max}$, be the corresponding eigenvectors. Since $\nabla^2 f$ is real and symmetric, all eigenvalues and eigenvectors are real, and eigenvectors that correspond to distinct eigenvalues are orthogonal. Consider the 2–dimensional subspace, $\mathcal{E}^2$,

K.E. Parsopoulos and M.N. Vrahatis are with the Department of Mathematics, University of Patras, Greece (email: {kostasp, vrahatis}@math.upatras.gr). V.C. Georgopoulos is with the Department of Speech and Language Therapy, T.E.I. of Patras, Greece (email: voula@teipat.gr).

978-1-4244-1823-7/08/$25.00©2008 IEEE

which is spanned by $\{e_{\min}, e_{\max}\}$. Then, the points $x \in \mathcal{E}^2$ can be determined by a pair $(c_1, c_2)$ such that [3], [4],

$$x = x^* + c_1 e_{\min} + c_2 e_{\max}, \quad c_1, c_2 \in \mathbb{R}, \tag{1}$$

i.e., the pair $(c_1, c_2)$ determines the coordinates of $x$ in $\mathcal{E}^2$.

It is known that in a sufficiently small neighborhood of $x^*$, the directions of the principal axes of the elliptical contours ($n$–dimensional ellipsoids) are given by the eigenvectors of $\nabla^2 f$, while the lengths of the axes are inversely proportional to the square roots of the corresponding eigenvalues. Thus, a variation along $e_{\max}$ causes the largest change in $f$, while $e_{\min}$ corresponds to the least sensitive direction of change. Consequently, the subspace $\mathcal{E}^2$ provides all the important information regarding $f$ around its global minimizer, thereby constituting a very appealing choice as a subspace for the visualization of points of arbitrary dimensionality [3].

The proposed approach has been applied for the visualization and investigation of the performance of traditional gradient–based algorithms, such as Steepest Descent with Armijo Line Search, Fletcher–Reeves, Polak–Ribiere, Davidon–Fletcher–Powell, and Broyden–Fletcher–Goldfarb–Shanno [3], [5]. It has also been applied for studying the behavior of well–known methods on problems with noisy objective functions as well as for the visualization of neural networks training algorithms, with promising results [4], [5].

The aforementioned technique can be directly applied for visualizing the population of a population–based algorithm during the evolution process in the original $n$–dimensional search space $S$. This is achieved by projecting the individuals under consideration on $\mathcal{E}^2$, i.e., by determining for each individual $x \in S$, the corresponding pair $(c_1, c_2) \in \mathcal{E}^2$, such that Eq. (1) holds. The proposed technique is independent of the individuals' dimension and provides a unique projection subspace, which captures all relevant information regarding the shape of the objective function in the neighborhood of the global minimizer.

Besides the application of the aforementioned technique for the direct visualization of the population during the evolution in the original $n$–dimensional space, the proposed technique can also be applied as follows: the algorithm is initialized with a uniformly distributed population of pairs $(c_1, c_2) \in \mathcal{E}^2$ and applied for the minimization of $f$ directly in the subspace $\mathcal{E}^2$. The global minimizer $x^*$ of $f$ in the $n$–dimensional space $S$ corresponds, by definition, to the origin of $\mathcal{E}^2$. For the evaluation of a pair $(c_1, c_2)$ with $f$, the pair is first transformed to the corresponding point $x$ in the $n$–dimensional space using Eq. (1), and, subsequently the obtained point $x$ is passed as an argument to $f$ to obtain the objective function value. Since $\mathcal{E}^2$ captures all possible changes of $f$, the 2–dimensional optimization can reveal the dynamics of the algorithm with the advantage of direct visualization on a plane. Moreover, 2–dimensional optimization requires smaller population sizes, since the problem's dimension is reduced.

The main drawbacks of the proposed technique is the requirement to foreknow the global minimizer of $f$, i.e., the algorithms can be visualized and studied only on problems with known global minimizers and computable Hessian matrix on it. However, it can provide useful knowledge regarding the algorithm's performance on classes of similar test functions by visualizing the algorithm on some representative problems with known minimizers.

## III. The Employed Evolutionary Algorithms

For completeness purposes, we briefly describe the algorithms that were selected for the illustration of the proposed technique.

### A. Differential Evolution

Differential Evolution (DE) was developed by Storn and Price [6]. It utilizes a population of $N$, $n$–dimensional vectors, $x_{i,G}$, $i = 1, 2, \ldots, N$, for each iteration (generation), $G$, of the algorithm. The initial population is taken to be uniformly distributed in the search space. At each generation, the *mutation* and *crossover* (recombination) operators are applied on the individuals, giving rise to a new population, which is subsequently subjected to the selection phase. This phase effectively identifies the $N$ best points from both populations to comprise the next generation.

According to the *mutation* operator, for each vector $x_{i,G}$, $i = 1, 2, \ldots, N$, a *mutant vector* is generated through the equation:

$$v_{i,G+1} = x_{r_1,G} + F\ (x_{r_2,G} - x_{r_3,G}), \tag{2}$$

where $r_1, r_2, r_3 \in \{1, 2, \ldots, N\}$ are mutually different random indices, and, $F \in (0, 2]$. The indices $r_1, r_2, r_3$, also need to differ from the current index, $i$. Consequently, to apply mutation, $N$ must be greater than or equal to 4.

Following the mutation phase, the *crossover* operator is applied on the population. Thus, a *trial vector*,

$$u_{i,G+1} = (u_{1_{i,G+1}}, u_{2_{i,G+1}}, \ldots, u_{n_{i,G+1}}), \tag{3}$$

is generated, where,

$$u_{j_{i,G+1}} = \begin{cases} v_{j_{i,G+1}}, & \text{if } (R_j \leqslant CR) \text{ or } j = \mathrm{rnbr}(i), \\ x_{j_{i,G}}, & \text{if } (R_j > CR) \text{ and } j \neq \mathrm{rnbr}(i), \end{cases}$$

where $j = 1, 2, \ldots, n$; $R_j$ is the $j$–th evaluation of a uniform random number generator in the range $[0, 1]$; $CR$ is a user–defined crossover constant in the range $[0, 1]$, and $\mathrm{rnbr}(i)$ is a randomly chosen index from the set $\{1, 2, \ldots, n\}$.

To decide whether or not the vector $u_{i,G+1}$ will be a member of the population of the next generation, it is compared to $x_{i,G}$. Thus,

$$x_{i,G+1} = \begin{cases} u_{i,G+1}, & \text{if } f(u_{i,G+1}) < f(x_{i,G}), \\ x_{i,G}, & \text{otherwise.} \end{cases}$$

The procedure described above is considered as the standard variant of the DE algorithm. Different mutation and crossover operators have been applied with promising results [6]. In order to classify the different variants, the scheme

$$DE/x/y/z,$$

is used, where $x$ specifies the mutated vector ("rand" for randomly selected individual or "best" for selection of the best individual); $y$ is the number of difference vectors used; and, $z$ denotes the crossover scheme (the scheme described above is due to independent binomial experiments and it is denoted as "bin") [6]. According to this description scheme, the DE variant described above is denoted as DE/rand/1/bin.

A highly beneficial scheme that deserves special attention is the DE/best/2/bin, where,

$$v_{i,G+1} = x_{best,G} + F\ (x_{r_1,G} + x_{r_2,G} - x_{r_3,G} - x_{r_4,G}). \quad (4)$$

The use of two difference vectors in this scheme seems to improve the diversity of the population in cases where $N$ is adequately high [6].

## B. Covariance Matrix Adaptation Evolution Strategies

Evolution Strategies (ES) have been developed by Rechenberg and Schwefel [7]–[10]. They exploit a population of $\mu$ individuals to probe the search space. At each iteration of the algorithm, $\lambda$ offsprings are produced by recombining and mutating a set of randomly chosen individuals from the current population (parents). After the offsprings' generation, a selection phase takes place, where either the $\mu$ best individuals among the offsprings or the $\mu$ best individuals among both the parents and the offsprings are selected to comprise the next generation. These two variants are denoted as $(\mu, \lambda)$–ES and $(\mu + \lambda)$–ES, respectively.

ES use a set of parameters, called *endogenous parameters*, to control certain statistical properties of the genetic operators, especially those of the mutation operator. These parameters can either be fixed or evolve during the evolution process resulting in self–adaptive ES variants [11].

Covariance Matrix Adaptation Evolution Strategies (CMA–ES) were developed by Hansen and Ostermeier [12]. They are self–adaptive $(\mu_W, \lambda)$–ES, where $W$ denotes weighted recombination from $\mu$ out of $\lambda$ individuals. CMA–ES use a set of parameters,

$$p_{c,G} \in \mathbb{R}^n, \quad C_G \in \mathbb{R}^{n\times n}, \quad p_{\sigma,G} \in \mathbb{R}^n, \quad \sigma_G \in \mathbb{R}^+,$$

where $G$ denotes the generation number. The parameters are initialized as follows: $p_{c,0} = p_{\sigma,G} = 0$ and $C_0 = I$ (the unity matrix), while $\sigma_0$ and the initial weighted mean of the $\mu$ best individuals, $\langle x\rangle_{W,0}$, have to be chosen problem dependent [12]. The object parameter vector $x_{k,G+1}$, $k = 1, 2, \ldots, \lambda$, is determined by the equation:

$$x_{k,G+1} = \langle x\rangle_{W,G} + \sigma_G \underbrace{B_G D_G z_{k,G+1}}_{\sim\mathcal{N}(0,C_G)},$$

where $\mathcal{N}(0, C_G)$ denotes the Gaussian distribution with zero mean and covariance matrix $C_G$, and,

$$\langle x\rangle_{W,G} = \frac{1}{\sum_{i=1}^{\mu} w_i} \sum_{i=1}^{\mu} w_i x_{i:\lambda,G}, \quad w_i \in \mathbb{R}^+,$$

with $i : \lambda$ denoting the $i$–th best individual, is the weighted mean of the $\mu$ best individuals at generation $G$; $\sigma_G \in \mathbb{R}^+$ is the step size; $z_{k,G+1} \in \mathbb{R}^n$ are independent realizations of a $(0, I)$–normally distributed random vector; and $B_G$, $D_G$ are determined by the symmetrical positive definite $n \times n$ covariance matrix $C_G$ as follows [12]:

$$C_G = B_G D_G (B_G D_G)^\top = B_G (D_G)^2 B_G^\top.$$

This is actually a singular value decomposition of $C_G$. Thus, the matrix $D_G$ is an $n \times n$ diagonal matrix with its diagonal elements being equal to the square roots of the eigenvalues of $C_G$, while $B_G$ is an $n \times n$ orthogonal matrix that determines the coordinate system, where the scaling with $D_G$ takes place and its columns are the normalized eigenvectors of $C_G$. A complete analysis of the adaptation procedure of $C_G$ can be found in [12].

## C. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a swarm intelligence algorithm, inspired from and based on the social dynamics and emergent behavior in socially organized colonies [13]–[18].

PSO exploits a population of individuals to synchronously probe promising regions of the search space. In this context, the population is called a *swarm* and the individuals (i.e., the search points) are called the *particles*. Each particle moves with an adaptable velocity within the search space, and retains a memory of the best position it has ever encountered. In the *global* variant of PSO, the best position ever attained by all individuals of the swarm is communicated to all the particles at each iteration. In the *local* variant, each particle is assigned to a topological neighborhood consisting of a prespecified number of particles. In this case, the best position ever attained by the particles that comprise a neighborhood is communicated among them [14], [16].

Assume an $n$–dimensional search space, $S \subset \mathbb{R}^n$, and a swarm consisting of $N$ particles. The $i$–th particle is in effect an $n$–dimensional vector $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^\top \in S$. The velocity of this particle is also an $n$–dimensional vector, $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^\top$. The best previous position encountered by the $i$–th particle in $S$ is denoted by $p_i = (p_{i1}, p_{i2}, \ldots, p_{in})^\top \in S$. Assume $g_i$ to be the index of the particle that attained the best previous position among all the particles in the neighborhood of the $i$–th particle, and $G$ to be the iteration counter. Then, the swarm is manipulated by the equations [19]:

$$\begin{aligned} v_{i,G+1} &= \chi \Big[ v_{i,G} + c_1 r_1 \big(p_{i,G} - x_{i,G}\big) \\ &\quad + c_2 r_2 \big(p_{g_i,G} - x_{i,G}\big) \Big], && (5) \\ x_{i,G+1} &= x_{i,G} + v_{i,G+1}, && (6) \end{aligned}$$

where $i = 1, 2, \ldots, N$; $\chi$ is a parameter called *constriction factor*; $c_1$ and $c_2$ are two parameters called *cognitive* and *social* parameters respectively; and $r_1$, $r_2$, are random numbers uniformly distributed within the range $[0, 1]$.

The constriction factor constitutes a mechanism for controlling the magnitude of velocities and it is derived analytically through the formula [19]:

$$\chi = \frac{2\kappa}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|}, \quad (7)$$

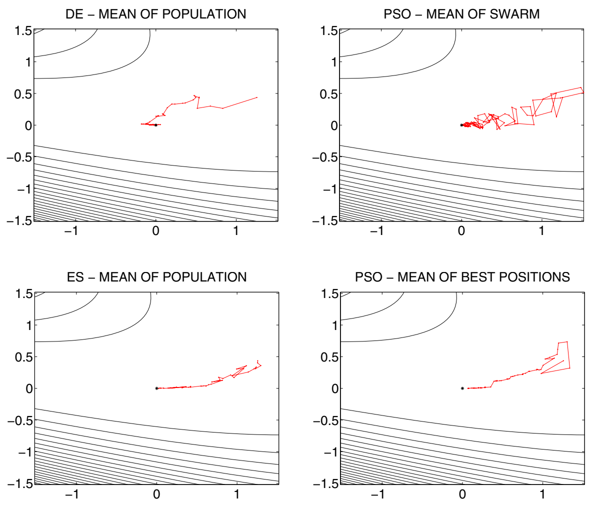

Fig. 1. The trajectory of the population's mean while optimizing $F_1$ in $S$, projected in the vector space $\mathcal{E}^2$.

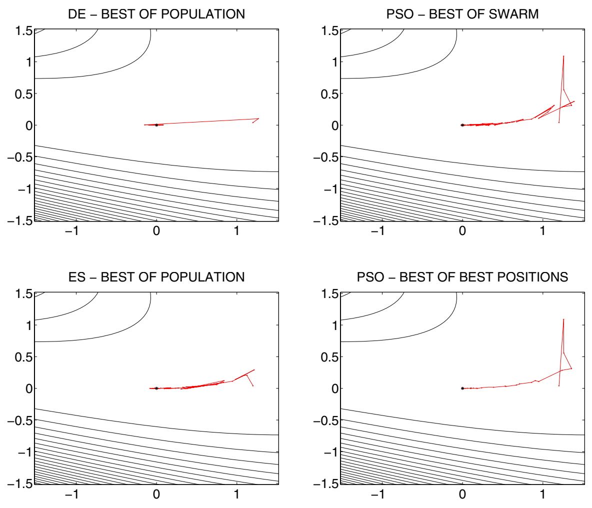

Fig. 2. The trajectory of the population's best while optimizing $F_1$ in $S$, projected in the vector space $\mathcal{E}^2$.

for $\phi > 4$, where $\phi = c_1 + c_2$ and $\kappa = 1$. Different configurations of $\chi$ as well as a thorough theoretical analysis of the derivation of Eq. (7) can be found in [19], [20].

## IV. Application of the Proposed Technique

The proposed visualization technique has been suggestively applied for DE, CMA–ES and PSO on the following test problems:

Test Problem 1 (Rosenbrock function) [21]. This problem is 2–dimensional and it is defined by:

$$F_1(x) = \left[10\left(x_2 - x_1^2\right)\right]^2 + (1 - x_1)^2 .$$

It has the global minimum $F_1^* = 0$ at $x^* = (1, 1)^\top$.

Test Problem 2 (Extended Powell Singular function) [21]. This problem is 12–dimensional and it is defined by:

$$F_2(x) = \sum_{i=1}^{12} f_i^2(x),$$

where,

$$\begin{aligned} f_{4i-3}(x) &= x_{4i-3} + 10x_{4i-2}, \\ f_{4i-2}(x) &= \sqrt{5}(x_{4i-1} - x_{4i}), \\ f_{4i-1}(x) &= (x_{4i-2} - 2x_{4i-1})^2, \\ f_{4i}(x) &= \sqrt{10}(x_{4i-3} - x_{4i})^2, \end{aligned}$$

with $i = 1, 2, 3$. It has the global minimum $F_2^* = 0$ at $x^* = (0, 0, \ldots, 0)^\top$.

Test Problem 3 (Schwefel Double Sum function) [10]. This problem is 20–dimensional and it is defined by:

$$F_3(x) = \sum_{i=1}^{20} \left( \sum_{j=1}^{i} x_j \right)^2 .$$

It has the global minimum $F_3^* = 0$ at $x^* = (0, 0, \ldots, 0)^\top$.

The DE/best/2/bin version of the DE algorithm, the global variant of the constriction factor PSO, as well as the default CMA–ES algorithm proposed in [12] have been used to illustrate the proposed visualization technique on the aforementioned test problems. The choice of the specific versions of the algorithms has been motivated by their developers' suggestions as well as their superior performance in several applications. Here, our intention is rather the illustration of the proposed technique than the comparison of the algorithms.

Initially, the proposed visualization technique was applied for the visualization of specific individuals of the populations during the minimization in the $n$–dimensional space, $S$, where $n$ is the dimension of the problem under consideration. For this purpose, a population of $10 \times n$ individuals was used for each algorithm, and a maximum number of $10^4 \times n$ function evaluations was allowed to be performed. The desired accuracy was $10^{-5}$ for all test problems. The parameters of the DE algorithm were set equal to $F = CR = 0.5$. The PSO parameters were also set to their default values $\chi = 0.729$, $c_1 = c_2 = 2.05$ [19]. For CMA–ES, $\lambda = 2$ and $\mu = 20 \times n$ were used. All algorithms were initialized with the same initial population, which was randomly (uniformly distributed) selected within $S = [-2, 2]^n$ for all test problems.

The behavior of each algorithm in a single run can be illustrated through the proposed visualization technique, as

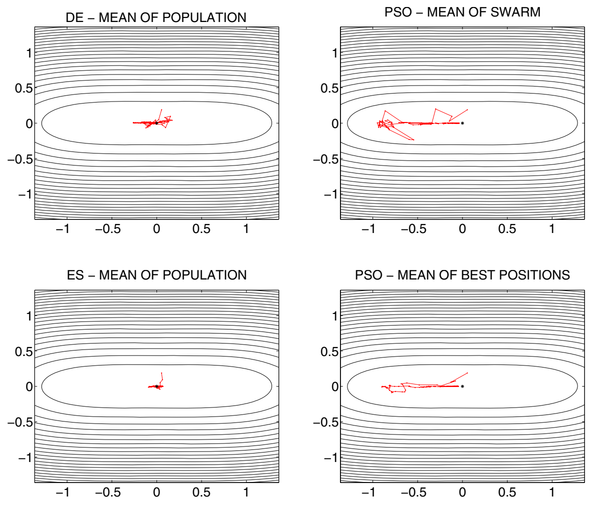

Fig. 3. The trajectory of the population's mean while optimizing $F_2$ in $S$, projected in the vector space $\mathcal{E}^2$.

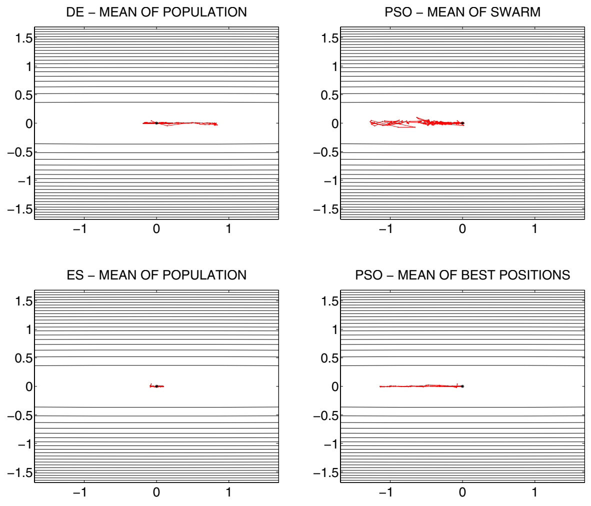

Fig. 5. The trajectory of the population's mean while optimizing $F_3$ in $S$, projected in the vector space $\mathcal{E}^2$.

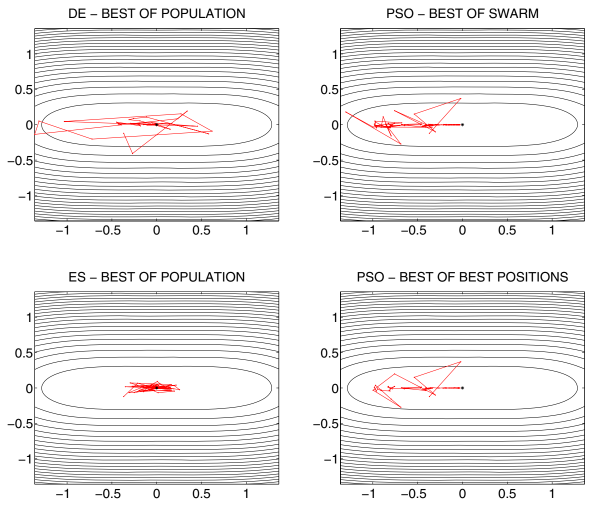

Fig. 4. The trajectory of the population's best while optimizing $F_2$ in $S$, projected in the vector space $\mathcal{E}^2$.

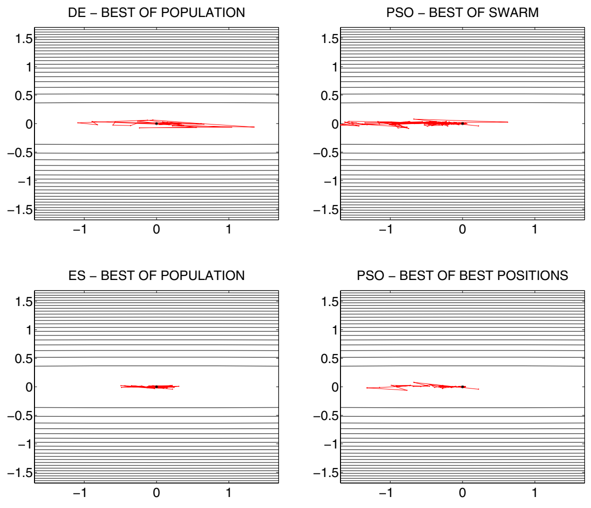

Fig. 6. The trajectory of the population's best while optimizing $F_3$ in $S$, projected in the vector space $\mathcal{E}^2$.

follows: for each algorithm, the mean and the best individual (i.e., the individual with the lowest function value) of the population were recorded at each iteration (generation). Also, the extremal eigenvectors of the Hessian matrix at the global minimizer were determined for each test problem. The recorded data were then projected on the 2–dimensional vector space $\mathcal{E}^2$, which is spanned by the extremal eigenvectors as described in Section II, and the corresponding (projected) trajectories for the mean and best individual are depicted in the contour plots of Figs. 1 and 2 for Test Problem 1, in Figs. 3 and 4 for Test Problem 2, and in Figs. 5 and 6 for Test Problem 3, respectively. The black star at the origin of each figure denotes the global minimizer of the objective function (which is always projected at the origin of $\mathcal{E}^2$). PSO employs actually two populations, the swarm for the current iteration and the best positions to store the best position ever visited by each particle. For this reason, the mean and best individual of both populations are reported in the figures.

As we observe in Figs. 1 and 2, for all algorithms, both the mean and the best individual spent more time rather on horizontal than vertical moves. This effect can be attributed to the highest variation of $f$ along the vertical axis, which corresponds to eigenvector $e_{\max}$ of the largest eigenvalue, while, on the other hand, the horizontal axis corresponds to the least sensitive direction of change. Thus, the algorithms were able to detect rapidly the most promising region of

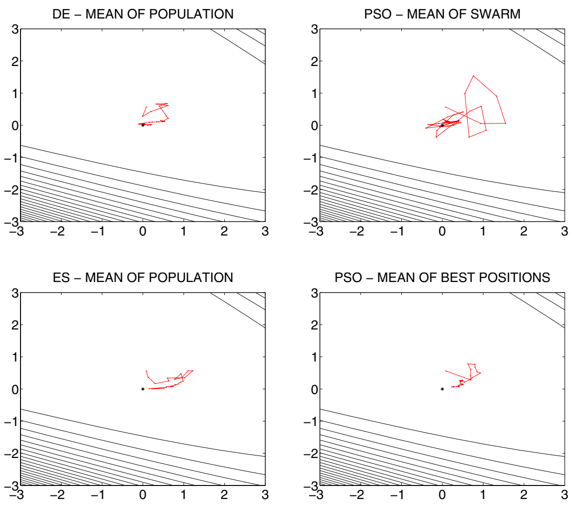

Fig. 7. The trajectory of the population's mean while optimizing $F_1$ directly in $\mathcal{E}^2$.

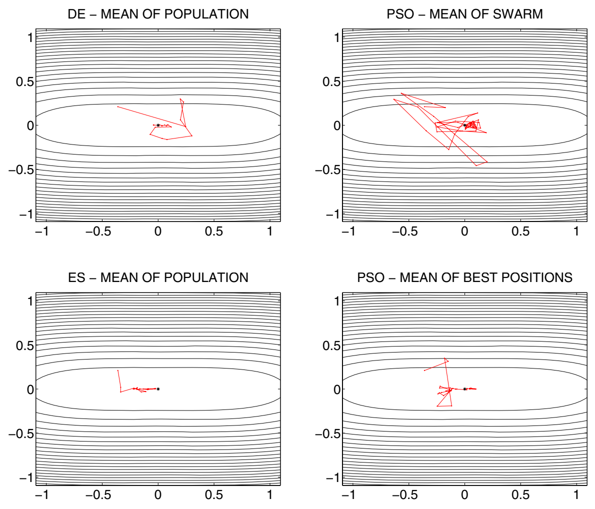

Fig. 9. The trajectory of the population's mean while optimizing $F_2$ directly in $\mathcal{E}^2$.

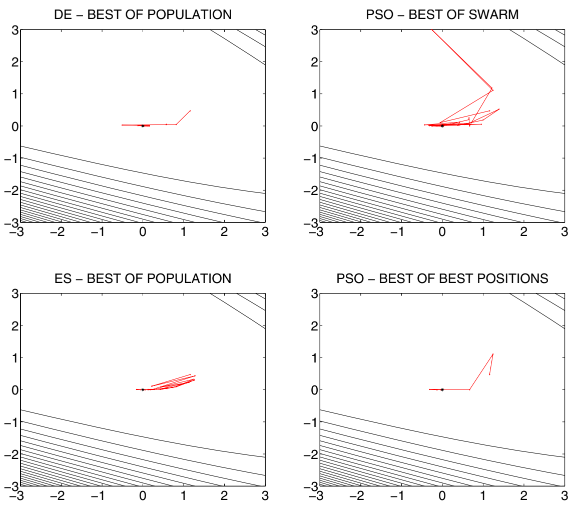

Fig. 8. The trajectory of the population's best while optimizing $F_1$ directly in $\mathcal{E}^2$.

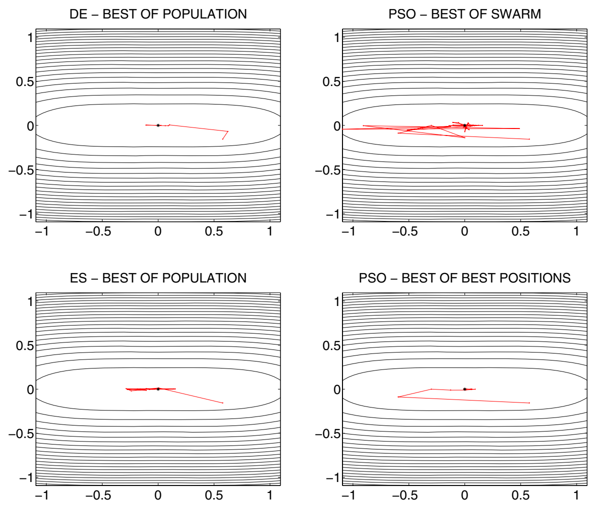

Fig. 10. The trajectory of the population's best while optimizing $F_2$ directly in $\mathcal{E}^2$.

the search space along the steepest direction and spent the rest iterations for approaching gradually the global minimizer from the flatter direction. This is an indication that the employed algorithms were able to identify the shape of the objective function locally.

Also, we can observe that the trajectory of the mean and best particle of the PSO's swarm is characterized by more intensive fluctuations than that of its best positions as well as the rest of the algorithms. This is an expected observation, since the swarm does not incorporate any form of selection, in contrast to the best positions of PSO as well as the populations of DE and CMA–ES.

Similar conclusions can be derived for Test Problems 2 and 3 from Figs. 3–6. Moreover, we can clearly observe that the trajectories of DE and PSO are more similar than CMA–ES. This could be attributed to the nature of the algorithms. While DE and PSO are based on difference vectors of the current populations to produce the points for the next generation, CMA–ES employ a rather probabilistic approach, drawing points from adaptive probability densities. Therefore, the proposed approach can be also used to roughly identify algorithms of similar dynamics.

Besides the aforementioned investigation of the algorithms' trajectories projected in the space $\mathcal{E}^2$, we performed a second round of experiments, however, this time the optimization was performed directly in $\mathcal{E}^2$. More specifically,

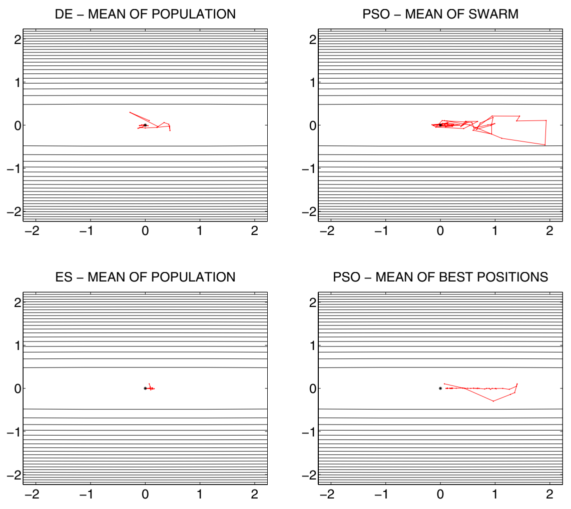

Fig. 11. The trajectory of the population's mean while optimizing $F_3$ directly in $\mathcal{E}^2$.

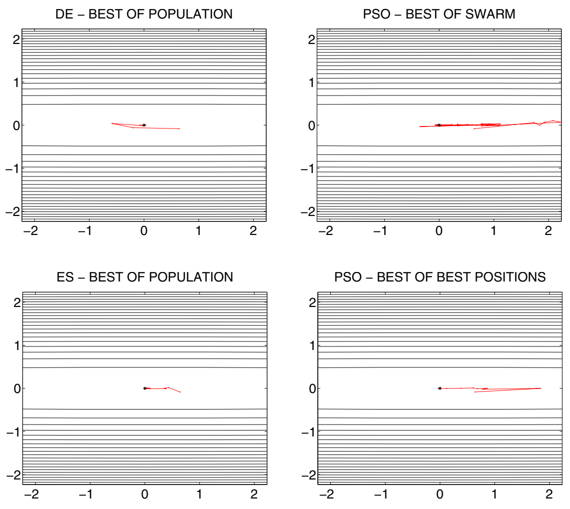

Fig. 12. The trajectory of the population's best while optimizing $F_3$ directly in $\mathcal{E}^2$.

instead of having individuals defined and updated in the original search space $S$ and then projected in $\mathcal{E}^2$ to obtain the corresponding figures, we assumed individuals defined directly in $\mathcal{E}^2$. Thus, each individual was a 2–dimensional vector defined and updated in $\mathcal{E}^2$, consisting of a pair of coordinates, while, for its evaluation with the objective function $f$, it was transformed to the corresponding $n$–dimensional vector through Eq. (1).

Following this approach, each problem was transformed to a 2–dimensional problem, thus, only small populations of 20 individuals were employed for each algorithm. The rest of the algorithms' parameters remained the same as in the previous experiments. The initial populations were always uniformly distributed in $[-3, 3]^2$. The same data were recorded as for the $n$–dimensional case. The obtained trajectories are depicted in Figs. 7–12.

It is clear that the figures obtained through the second round of experiments reveal similar behavior of the algorithms as with the case of Figs. 1–6. This is an indication that there was no crucial loss of information regarding the algorithm's dynamics by applying it directly on $\mathcal{E}^2$. Since the experiments in $\mathcal{E}^2$ are 2–dimensional using small population sizes, thereby requiring significantly less time and resources, this approach could be very useful in cases where the objective function is high–dimensional and each function evaluation requires significant time.

Regarding the algorithms performance, it worths noting that PSO approaches the minimizer in a different manner than DE and CMA–ES. Specifically, in most cases, PSO's swarm "encircles" the minimizer and approaches it from all directions, in contrast to DE and CMA–ES, which capture the shape of the contour lines and distribute the population in narrow channels parallel to the axis that corresponds to the eigenvector $e_{\min}$. This is also a direct implication of the lack of selection in PSO's swarm, which retains high diversity during optimization, since the best positions of the particles are stored in a separate population.

## V. Conclusions

A technique for the visualization of stochastic population–based optimization algorithms in any dimension has been introduced. The proposed technique uses the 2–dimensional vector space spanned by the extremal eigenvectors of the Hessian matrix of the objective function at a global minimizer to project the $n$–dimensional search points. This vector space captures all information regarding the shape of the objective function at this region. A population–based algorithm can either be applied on the original $n$–dimensional test problem and project its population on the subspace or it can be directly applied on the subspace and visualized, revealing almost the same information, yet reducing significantly the expected computational cost.

The proposed approach enables the user to obtain intuition about the performance of an algorithm on high–dimensional spaces. It also provides a unified visualization framework in any dimension, which alleviates the drawbacks of common projection methods that can result in false conclusions due to their heavy dependence on the selection of the projection subspace. Thus, it contributes towards the directions of developing more efficient algorithms and selecting the most proper algorithm for specific classes of similar test problems.

The proposed technique has been illustrated for three well–known population–based algorithms, DE, CMA–ES and PSO, on 2, 12 and 20–dimensional test problems. The reported results support the claim that it can be a very useful tool for visualization in any dimension.

Further research is needed to fully reveal the potential of the proposed technique in visualizing population–based algorithms as well as to investigate specific characteristics of the underlying algorithms.

## Acknowledgement

This project is co–funded by the European Social Fund and National Resources – (EPEAEK–II) ARCHIMIDIS. The work of K.E. Parsopoulos is supported by the State Scholarship Foundation of Greece (I.K.Y.).

## References

[1] M. Fisher, D. Mandic, J. A. Bangham, and R. Harvey, "Visualising error surfaces for adaptive filters and other purposes," in *Proc. IEEE Int. Conf. Accoustics Speech & Signal Processing (ICASSP)*, Instabul, Turkey, 2000.

[2] D. Silver and N. J. Zabusky, "Scientific visualization and computer vision," in *Proc. IEEE Workshop on Visualization and Machine Vision*, 1994, pp. 55–61.

[3] G. S. Androulakis and M. N. Vrahatis, "Optac: A portable software package for analyzing and comparing optimization methods by visualization," *Journal of Computational and Applied Mathematics*, vol. 72, pp. 41–62, 1996.

[4] G. S. Androulakis, G. D. Magoulas, and M. N. Vrahatis, "Geometry of learning: Visualizing the performance of neural network supervised training methods," *Nonlinear Analysis, Theory, Methods & Applications*, vol. 30, no. 7, pp. 4539–4544, 1997.

[5] G. S. Androulakis, M. N. Vrahatis, and T. N. Grapsa, "Studying the performance of optimization methods by visualization," *Systems Analysis–Modelling–Simulation*, vol. 25, pp. 21–42, 1996.

[6] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimization*, vol. 11, pp. 341–359, 1997.

[7] I. Rechenberg, "Evolutionsstrategie: Optimierung technischer systeme nach prinzipien der biologischen evolution," Ph.D. dissertation, Department of Process Engineering, Technical University of Berlin, Germany, 1971.

[8] H.-P. Schwefel, "Evolutionsstrategie und numerische optimierung," Ph.D. dissertation, Department of Process Engineering, Technical University of Berlin, Germany, 1975.

[9] ——, *Numerical Optimization of Computer Models*. Chichester: Wiley, 1981.

[10] ——, *Evolution and Optimum Seeking*. New York: Wiley, 1995.

[11] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies: A comprehensive introduction," *Natural Computing*, vol. 1, no. 1, pp. 3–52, 2002.

[12] N. Hansen and A. Ostermeier, "Completely derandomized self–adaptation in evolution strategies," *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[13] E. Bonabeau, M. Dorigo, and G. Théraulaz, *From Natural to Artificial Swarm Intelligence*. New York: Oxford University Press, 1999.

[14] R. C. Eberhart, P. Simpson, and R. Dobbins, *Computational Intelligence PC Tools*. Academic Press, 1996.

[15] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Networks*, vol. IV. Piscataway, NJ: IEEE Service Center, 1995, pp. 1942–1948.

[16] ——, *Swarm Intelligence*. Morgan Kaufmann Publishers, 2001.

[17] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2–3, pp. 235–306, 2002.

[18] ——, "On the computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, 2004.

[19] M. Clerc and J. Kennedy, "The particle swarm–explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, 2002.

[20] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, pp. 317–325, 2003.

[21] J. J. More, B. S. Garbow, and K. E. Hillstrom, "Testing unconstrained optimization software," *ACM Transactions on Mathematical Software*, vol. 7, no. 1, pp. 17–41, 1981.