Communications in Applied and Industrial Mathematics, ISSN 2038-0909, e-393 DOI: 10.1685/journal.caim.393

A multistart gradient-based algorithm with surrogate model for global optimization

Daniele Peri and Federica Tinti

CNR-INSEAN - National Research Council - Maritime Research Centre
Via di Vallerano, 139 - 00128 - Roma - Italy
daniele.peri@cnr.it
federicatinti@live.it

Communicated by Giorgio Fotia

Abstract

Gradient-based optimization algorithms are probably the most efficient option for the solution of a local optimization problem. These methods are intrinsically limited in the search of a local optimum of the objective function: if a global optimum is searched, the application of local optimization algorithms can be still successful if the algorithm is initialized starting from a large number of different points in the design space (multistart algorithms). As a counterpart, the cost of the exploration is further increased, linearly with the number of adopted starting points. Consequently, the use of a multistart local optimization algorithm is rarely adopted, mainly for two reasons: i) the large computational cost and ii) the absence of a guarantee about the success of the search (in fact, there is not a general indication about the minimum number of starting points able to guarantee the success of global optimization).

In this paper, techniques for reducing the computational cost of the full process together with some techniques able to maximize the efficiency of a parallel multistart search are described and tested. An extensive use of surrogate models is applied in order to drastically reduce the computational effort in practical applications, where the computational cost of a single objective function evaluation is high. De-clustering techniques are also adopted in order to exploit at best the different local searches.

Keywords: `Global optimization, gradient-based methods, trust region methods, surrogate models.`

AMS Subject Classification: `05A16, 65N38, 78M50`

1. Background and motivation.

Gradient-based algorithms, and second order methods in particular, represent one of the most efficient choices for local optimization. They have proof of global convergence under not too strict hypothesis, but they also need an estimate of (at least) first order derivatives: for this reason, their computational price is pretty high and it is also strongly dependent on the number of design variables. Multistart algorithms are an option if global minimum is searched, since they are, in principle, able to explore more

Received 2012 01 10. Accepted 2012 10 15. Published 2012 10 19.

than a single basin of attraction of the objective function, if the starting points are enough and well distributed onto the design space: this gives some chance to detect the basin of attraction containing the global optimum. Anyway, the cost of the exploration is further increased if compared with the single search (growing linearly with the number of starting points).

Practical applications, where the objective function and the constraints come from complex numerical simulations, represent the typical situations in which the objective function is computationally very expensive (days of computations on a parallel machine): as a consequence, the evaluation of the gradient, obtained by finite differences when black-box objective functions are used, may push the optimization costs way out of the assigned design time window. Since we are interested in the application of numerical optimization algorithms to this challenging class of problems, we are used to apply derivative-free optimization algorithms, where only the objective function value is required. Anyway, since gradient-based algorithms are more efficient in principle, the availability of a multistart gradent-based optimization algorithm with reduced computational cost would be preferable. To this aim, the use of a surrogate model[a] of the objective function, like kriging [1,2], may be a key for reducing the cost of the gradient computation. Kriging provides a global model of the objective function, which evaluation may be inexpensive (fractions of a second) in comparison with the evaluation time of the real objective function. As a counterpart, some discrepancies between the real objective function and the surrogate model appear, often hiding the optimal solution if training points are not properly placed. To reduce this risk, techniques for the certification of the quality of the surrogate model, like the trust region approach, would be also applied in order to guarantee the equivalence of the optimization problem solved by surrogates and the original problem.

In this paper, the use of surrogate models with trust region in a global optimization algorithm is proposed. In the next Section, a general description of the base elements for a multistart global optimization algorithm are presented. Then, a first optimization algorithm, in which the use of surrogate models is reducing the total computational time, is introduced. The use of a trust region approach for the certification of the quality of the surrogate model is then described in the third Section and added to the new algorithm. The use of a de-clustering technique, useful for maintain a single element for each detected basin of attraction, is then illustrated in the fourth Section: the final version of the algorithm is here presented. Lastly, some

[a]Surrogate models, or meta-models, are approximation techniques able to fit a generic function by means of algebraic functions. Polynomials, neural networks and kriging are some examples.

DOI: 10.1685/journal.caim.393

numerical results for algebraic test cases are reported and commented. Comparisons with the a derivative-free global search algorithm (Particle Swarm Optimization - PSO - described in Section 5) are presented. A comparison with the same algorithm in which the first and second order information are obtained by central differences using the real objective function is also presented. Some conclusions are drawn in the last Section.

2. Basic multistart optimization algorithm with surrogates.

The general idea of applying a multistart model for the solution of a global optimization problem has been already attempted in literature. In [3] the objective function is simply computed on a random population, and local searches are initiated from the most promising points; at each descent step, a new quasi-random population is derived again around the most promising points, and a local search is started again.

In order to speed up the convergence of the optimization, in [4] the convexity estimation of a kriging model is performed by computing the Hessian of a surrogate model of the objective function, and a clustering technique is adopted in order to detect the basins of attraction of a multimodal function. Also in [5] a multistart search is supported by quadratic local models of the objective function. This because the complexity of reliable simulations implies a huge execution time for the optimization procedure, often well beyond the time limits prescribed by the design team for the numerical activities in real design operations.

Surrogate models are also adopted when constraints are too expensive to compute: in [6] a response surface is applied for mapping structural constraints on a coarse structural model, in order to predict with a good approximation the real constraints, coming from the computation of a refined and more complex structural model. In [7] derivatives information are used for improving the predictive performances of a kriging surrogate model, derived from a set of expensive computations: here two different levels for the estimate of the objective function are also applied.

In this paper, we are suggesting to adopt a local second order model of the objective function for the determination of the search direction, similarly to what is produced by Sequential Quadratic Programming (SQP) algorithms. The difference with standard algorithms is the use of a surrogate model for the determination of the local second order model, instead of the computation of the real objective function: this is drastically reducing the computational time, since a single evaluation of the objective function is required for each iteration (and for each element of the multistart search), while the evaluation of the Hessian by central differences is requiring $(N + 1) * N * 2$ evaluations of the (real) objective function if N is the number of the adopted design variables. SQP is probably one of the first and more efficient techniques for the solution of a local optimization problem. SQP produces a sequence of second order local models of the objective function in order to minimize it. The way in which the second order information about the objective function are obtained is the key for the success of the (efficient) solution of the optimization problem. Following the scheme of the SQP algorithm, at each of the search points a (local) second order model is derived, here basing on the computation of the surrogate model on an appropriate number of points placed in the vicinity of the starting point. Using the local second order model, the Newton step is computed, and the search point is moved toward this position. This operation is performed, in parallel, on all the different search points adopted in the multistart procedure. All the previously computed values of the true objective function are fully re-utilized by the algorithm for the determination of the surrogate model.

Since we need to derive a surrogate model, a training phase is needed. A so-called Design Of Experiments (DOE) is required at the begin of the algorithm: some training points are placed into the design space in order to provide some initial values of the objective function in different locations. Since the multistart algorithm itself requires a number of starting locations well dispersed into the design space for the initialization of the search, we can use the same points from the DOE as starting points for a parallel multistart gradient-based minimization scheme, so that each point of the DOE is an initial point for a local search.

Another issue to be included into the algorithm is the verification of the point at which the Newton method is pointing to. In fact, Newton method is looking for a stationary point of the second order model, but we have not a guarantee that the Hessian matrix of the local model is definite positive at any point. As a consequence, a check about the convexity of the second order model is performed: if the check is positive, the Newton direction is applied, otherwise the anti-gradient is used and golden search algorithm is adopted for the determination of the steplength[b]. The resulting algorithm is sketched in Algorithm 1.

We select kriging as global model of the objective function because it is an interpolation model, that is, the model is exact on the computational point. Furthermore, the dependency of the global model from non-uniformly spaced points can be controlled, and we can recycle all the previously com-

[b]If the Hessian is not positive, the model would be pointing to a local maximum of the objective function.

```
1  Compute a number of initial points (DOE).
2  Initialize a kriging global model of the objective function.
3  Compute a second order model around each DOE point using the
   surrogate.
4  Compute the Newton direction and steplength for each point.
5  Check if Newton direction is a descent direction.
6  Select descent direction.
7  Move each point according to the selected step.
8  Compute the objective function at the new selected point.
9  Update the number of points available for the metamodel
   computation.
10 GO TO 2.
```

Algorithm 1: Multistart Gradient-based algorithm.

puted points, discharging only few of the points if they are too close each other. By the way, techniques for the regularization the model are available [8].

An example of a single iteration of the algorithm is represented in Figure 1. Here the dots are the DOE points, while the global model of the objective function (provided by kriging) is reported as a surface, and the contour levels are colored (in gray scales) according to the local value of the objective function. Around each point, the second order local model is represented. For some points, the curvature of the local model is providing a minimum point for the model, while in other cases the model is providing a maximum (top right point is an example). In this second case, the Newton direction is not used, as previously described.

The Algorithm 1 is similar to what already proposed in [4] and [5]. In this formulation, we can observe two critical points, raising some concerns about the precision and the efficiency of the algorithm, that can be improved by adding two further elements.

Firstly, we need to prevent the algorithm from being driven to wrong direction by an inaccurate approximation of the objective function. As a consequence, some techniques for controlling the quality of the prediction must to be introduced.

Secondly, we need to preserve diversity in the searching points during the optimization, since the objective function could be multimodal, and a large number of different local minima needs to be the identified in order to certify the global minimum. As a consequence, when two points are enter-

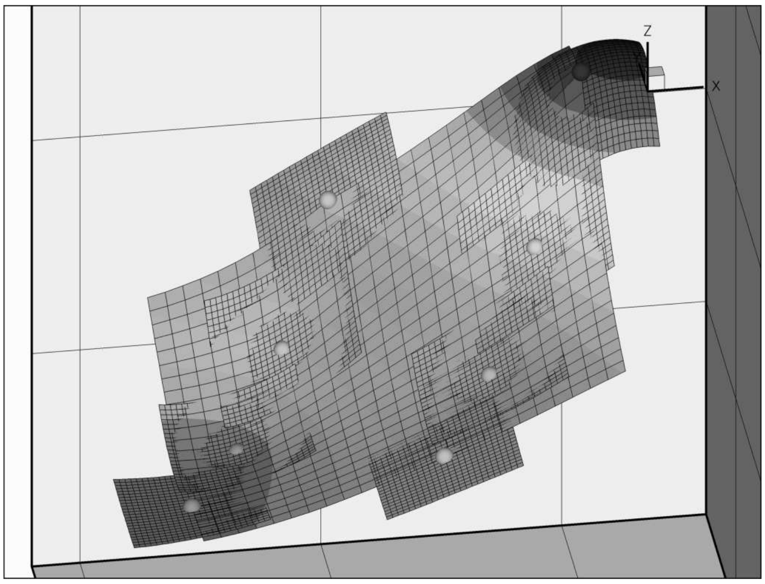

Figure 1. Example of application of a kriging surrogate model in the global and local reconstruction of the objective function.

ing the same basin of attraction, or more generally are too close each other, one of them should be moved away in a different location, previously unexplored, since they are naturally going to collapse to the same local minima, being the search algorithm gradient-based. Techniques for the improvement of the base Algorithm 1 are described in the following.

3. Trust-region method for the control of the global model.

As previously recalled, the surrogate model may be not accurate enough to reproduce the real objective function correctly, due to a coarse sampling of the design space. Consequently, inaccurate estimate of the objective function may led to to incorrect estimation of its derivatives, and derivative-based algorithms are largely sensible to this. To reduce this drawback, a check about the predictive ability of the current surrogate model is of primary importance, in order to be sure to produce a descent direction at each step. Newton algorithm is providing both the search direction and the steplength of the line search: in order to prevent the algorithm to proceed toward regions in which the prediction of the surrogate model is inaccurate, the trust region approach described in [9] can be applied. Trust region approach is providing a measure of the so called *trust region radius*, that represents the upper limit for the line search at each step of a local optimization algorithm. Trust region methods are not able to provide a correction of the local search direction computed via the surrogate model, but they are able to give indications about the limit the region of trustability for the surrogate model, hence limiting the search while preserving a prescribed level of accuracy.

Once initialized, the trust region radius is dynamically adjusted by checking the quality of the prediction at a certain distance from the search point, usually (for local optimization algorithms) at the end of the line search. Check is obtained by comparing the expected improvement at the end of the line search with the real improvement. Examples of this approach include its use in structural optimization, as in [6], as well as microwave design in [10]. In [9], the authors refer to two different level of the approximations, high fidelity model (in our case, the real objective function) and low fidelity model (here, the surrogate model), and they use trust-region methodology to limit the extension of the line search, performed using a local approximation of the objective function. The size of the trust region is dynamically changed as the optimization proceeds. An alternative to the consideration of the discrepancies between low and high fidelity response would be the use of their ratio [11].

In order to tune the trust-region radius, according to [9], we compare the suggested steplength, provided by the Newton method, with the actual value of the trust-region radius. If the predicted steplength is larger than the trust-region radius, steplength is decreased up to this value, otherwise the suggested steplength is fully applied. Once the new position of the search point is detected, we compute here the real objective function, and we can compare the two levels of approximation:

$$\rho = \frac{f(x) - f^*(x)}{f(x)}$$

in which $f(x)$ represents the high fidelity function and $f^*(x)$ the low fidelity function (following the nomenclature by [9]). Trust-region radius is increased or decreased according to the value of ρ. Low values of ρ are suggesting an increase of the trust region radius, while high values of ρ are indicating that the trust-region radius is too large. Since kriging is an interpolation model, we have the coincidence between high and low fidelity on each training point of the surrogate model: as a consequence, the aforementioned procedure can be reasonably applied if we check the distance from the closest available training point. The original Algorithm 1 is now modified to incorporate these elements, as the resulting algorithm is described in Algorithm 2.

```
1  Compute a number of initial points (DOE).
2  Derive a kriging global model of the objective function.
3  Compute a second order model around each DOE point using
   the surrogate.
4  Compute the Newton direction and steplength for each point.
5  Check if Newton direction is a descent direction.
6  Select descent direction.
7  Select the steplenght (to be inside the trust region radius).
8  Move each point according to the selected step.
9  Compute the objective function at the new selected point.
10 Check the difference $\rho$ and update the trust region radius.
11 Update the number of points available for the metamodel
   computation.
12 Check the trust region radius.
13 GO TO 2.
```

Algorithm 2: Multistart Gradient-based algorithm with trust region.

4. De-clustering.

The global optimization algorithm we are here describing is based on a sequence of parallel local searches. The general idea is that each search is addressed to a different basin of attraction of the objective function, in order to explore the more local minima as possible. As a consequence, we prefer not to have different particles exploring the same basin of attraction: a single particle is enough for the exploration of a single basin of attraction, and the presence of two or more particles in the same area is not helpful for the completeness of the exploration. As a consequence, when two particles become closer than a prescribed limit, one of them is moved away to a different location into the design space, in order to explore a different region, previously unexplored. The selection of the new position is performed according to the methodology described in [12]: here the known values of the objective function are used as training points for two different surrogate models. Interpolators are used in order to guarantee the real value of the surrogate function on the training points: in [12], a linear interpolation method and kriging are adopted. The prediction of these two surrogate models are compared systematically on a large number of samples in the whole design space. Two different criteria are used in selecting the new starting point: one is the maximum discrepancy between the predictions of the two different surrogate models (variance), the second is the expected value provided by the surrogate models. Large variance is indicative of poor information in that region, so that no previous evaluation has been obtained at that location. Among them, points for which the prediction is lower than the mean value of the objective function are preferable: we are asking this quality in order to add points that are supposed to have a low value of the objective function. Among the two closest point selected for the shift, the worst one is moved toward the new position.

This new option is now added to the algorithm described in table Algorithm 2, producing the final version of the algorithm, presented in table Algorithm 3. This new algorithm is called Multi Start Gradient Search with Surrogate Model (MSGS-SM) method.

```
1  Compute a number of initial points (DOE).
2  Derive a kriging global model of the objective function.
3  Compute a second order model around each DOE point
   using the surrogate.
4  Compute the Newton direction and steplength for each
   point.
5  Check if Newton direction is a descent direction.
6  Select descent direction.
7  Select the steplenght (to be inside the trust region radius).
8  Move each point according to the selected step.
9  Compute the objective function at the new selected point.
10 Check the difference $\rho$ and update the trust region radius.
11 Update the number of points available for the metamodel
   computation.
12 Check for clustered points: in case, move closer point to an
   unexplored region.
13 Check the trust region radius.
14 GO TO 2.
```

Algorithm 3: Multistart Gradient-based algorithm with trust region and de-clustering technique (MSGS-SM).

5. Comparative study with a global search algorithm: PSO Algorithm.

In order to check the qualities of the MSGS-SM, a different global search algorithm is adopted for comparison. The Particle Swarm Optimization (PSO), developed by Kennedy and R. Eberhart in 1995, has been selected to this purpose. In PSO, each particle, a collection of which makes up a swarm, has information about its current position and speed in the search space. Instead, original PSO formulation defines each particle as a potential solution of a problem in a N-*dimensional* space, where the i^{th} particle of the swarm can be represented by a N-*dimensional* vector, $\mathbf{X}_i = (x_{i,1}, x_{i,2}, ..., x_{i,N})^T$. The velocity of each particle is given by the vector $\mathbf{V}_i = (v_{i,1}, v_{i,2}, ..., v_{i,N})^T$. Each particle makes experience of the various investigated locations. The best previously visited position by the i^{th} particle is denoted as $\mathbf{P}_i = (p_{i,1}, p_{i,2}, ..., p_{i,N})^T$. Among these position, it is possible to define the best ever visited location for the whole swarm, $\mathbf{P}_{best} = (p_{best,1}, p_{best,2}, ..., p_{best,N})^T$. The original PSO algorithm [13], is formulated as

$$\begin{aligned} v_{i,j}^{k+1} &= w v_{i,j}^k + w_1 r_1 (p_{i,j}^k - x_{i,j}^k) + w_2 r_2 (p_{best,j}^k - x_{i,j}^k) \\ x_{i,j}^{k+1} &= x_{i,j}^k + v_{i,j}^{k+1} \end{aligned} \tag{1}$$

where $j = 1, 2, ..., N$ spans the coordinates of the design variables space; $i = 1, 2, ..., n_{sw}$ spans the particles, being n_{sw} the size of the swarm and k the current iteration; r_1 and r_2 are two random numbers. w is an inertial coefficient, w_1 is a coefficient for individual confidence and w_2 is a coefficient for swarm confidence and the values of parameter are $w = 1$, $w_1 = w_2 = 2.0$. Subsequently a work done by Clerc [14] indicates that use of a constriction factor may be necessary to insure convergence of the PSO algorithm, and in [15], PSO is defined by

$$\begin{aligned} v_{i,j}^{k+1} &= \chi [v_{i,j}^k + c_1 r_1 (p_{i,j}^k - x_{i,j}^k) + c_2 r_2 (p_{best,j}^k - x_{i,j}^k)] \\ x_{i,j}^{k+1} &= x_{i,j}^k + v_{i,j}^{k+1} \end{aligned} \tag{2}$$

where χ is a constriction coefficient, c_1 is a coefficient for individual confidence and c_2 is a coefficient for swarm confidence and the values of parameter are $\chi = 0.729$, $c_1 = c_2 = 2.05$.

Here we are using the following formulation

$$v_{i,j}^{k+1} = \chi [w v_{i,j}^k + c_1 (p_{i,j}^k - x_{i,j}^k) + c_2 (p_{best,j}^k - x_{i,j}^k)] \tag{3}$$

$$x_{i,j}^{k+1} = x_{i,j}^k + v_{i,j}^{k+1} \tag{4}$$

where we decide to fix in (3) the parameters r_1 and r_2 equal to 1, thus we eliminate the random factor introduced by these two coefficients. In this way, we transform a pure stochastic method into a deterministic one [16,17]. Initial configuration of the swarm may be a crucial issue for the efficiency of PSO iteration [18].

The relationship between coefficients in (1) and in (2), is

$$\chi = w \qquad \chi c_1 = w_1 \qquad \chi c_2 = w_2 \tag{5}$$

The equation (5) indicates that the constriction coefficient χ has control over the velocity of particles, which is considered in [15] to be crucial in the convergence property of particles. Equation (3) provides the corresponding values of parameter with $\chi = 0.729$; $w = 1$ $c_1 = c_2 = 2.05$ is equivalent to (3) with $\chi = 1$; $w = 0.729$ $c_1 = c_2 = 1.494$ (i.e. $0.792 * 2.05 = 1.49445$).

The above values of parameters in the PSO are frequently adopted in solving various optimization problems. However, these values of the parameters are not necessarily optimal. In the next Section, in order to make a choice on the parameters, a comparison of results has been proposed in two cases. In the first case, we ran the PSO algorithm for all functions using the best performance parameters defined in [15] $c_1 = 2.05$, $c_2 = 2.05$, $\chi = 0.729$, $w = 1$, In the second case, the parameters are fixed as proposed in [19], where $c_1 = 2.04236627$, $c_2 = 1.14957333$, $\chi = 1.04718542$, $w = 0.72004962$[c]. These parameters are coming from the solution of an optimization problem, in which the objective function is the accuracy and speed of the solution of 37 algebraic objective function with a number of design variable ranging from 2 to 20, and the design variables are the PSO coefficients. A complete list of the adopted objective functions is reported also in [17]. A pattern search algorithm has been applied for the optimization of parameters, starting from the values provided in [15] ($c_1 = 2.05$, $c_2 = 2.05$, $\chi = 0.729$, $w = 1$).

6. Numerical experiments.

The aim of this Section is double. Firstly, we want to test PSO algorithm with two different choices of the parameters, in order to detect the more efficient option for the problems in hands. Secondly, we want to test whether the new method MSGS-SM (Multi-Start Gradient Search with Surrogate Model) represent a significant improvement with respect to the deterministic PSO.

To do that, we select a set of test problems (reported in Table 1) for which the global minimum is known (position and value). Dimensions of the problems are ranging from $N = 2$ to $N = 20$. Table 1 reports the problem names, their dimensions and the value of the known global minimum f_{glob}. In order to obtain comparable results, the same stopping criterion for all the

[c] The large number of digits is dictated by the observation reported in [19] about the large sensibility/instability of results for a small change in the parameters (accuracy higher than the third decimal place is still causing strong differences).

test cases is applied. In particular, maximum number of objective function evaluations is fixed to $100\times N$, where N is the number of design variables.

Table 1. Test problems.

Problem	N	f_{glob}
Six humps camel back	2	-1.03162
Treccani	2	0.0000
Quartic	4	-0.3523
Cosine mixture	2	-0.2000
Cosine mixture	4	-0.4000
Exponential	2,4	-1.0000
Hartman	3	-3.8627
Hartman	6	-3.3223
Goldstein and Price	2	3.0000
Branin	2	0.3979
Freudenstein and Roth	2	0.1800
Shekel 10 loc. minima	4	-10.5364
Griewank	2	0.0000
Schubert	2	-186.7309
Schubert pen. 1	2	-186.7309
Schubert pen. 2	2	-186.7309
Levy 5^n loc. minima	2,5,10,20	0.0000
Levy 10^n loc. minima	2,5,10,20	0.0000
Levy 15^n loc. minima	2,5,10	0.0000

Table 2 reports a comparison of the results obtained by two different choices of the parameters χ, w, c_1, c_2 in the PSO algorithm. In particular, for each choice of the parameters, the number of the initial points can be equal to $2\times N$, or $4\times N$. Observing the total error reported in the Table 2, the more accurate choice when the number of initial points is $2\times N$ in the second case as adopted in [19], on the other hand the more accurate choice when the number of points is $4\times N$ in the first case as adopted in [15].

In order to produce some visualizations about the way in which the algorithm proceeds, a first test has been performed using the "Six Humps Camel Back" algebraic function. This function has 6 local minima in the interval $[-2.5, 2.5] \times [-1, 5, 1.5]$ and the number of design variables is 2. This function is also pretty regular, so that the visualization comes easier.

In Figure 2 the initial and final distribution of the computational points are represented. The algorithm is starting with a nearly uniform points distribution, and the final configuration is represented by some clusters of points, all around the local minima of the objective function.

In Figure 3, the sequence of the local second order models (computed by kriging) is reported. The computational points tend to move in the

DOI: 10.1685/journal.caim.393

Table 2. PSO numerical results.

Problem	N	parameters [15] $2\times N$	parameters [15] $4\times N$	parameters [19] $2\times N$	parameters [19] $4\times N$	analitycal solution
Six humps c. b.	2	-0.947908	-1.031532	-1.031600	-0.999993	-1.031520
Treccani	2	0.043494	0.000185	0.000881	0.005761	0.000000
Quartic	2	0.067529	-0.347945	-0.350971	-0.052526	-0.352300
Cosine mixture	2	-0.114725	-0.169944	-0.199801	-0.196393	-0.200000
Cosine mixture	4	-0.030505	-0.345534	-0.233977	-0.041982	-0.400000
Exponential	2	-0.994768	-0.999981	-0.999849	-0.994229	-1.000000
Exponential	4	-0.979737	-0.998668	-0.998717	-0.964866	-1.000000
Hartman	3	-3.517098	-3.678916	-3.848228	-3.806066	-3.862700
Hartman	6	-1.772308	-2.684009	-3.099797	-1.986630	-3.322300
Goldstein-Price	2	33.947090	3.330576	3.084735	3.249493	3.000000
Branin	2	0.399125	0.403750	0.420588	2.355777	0.397900
Freud. -Roth	2	0.399125	49.129360	0.666524	8.800137	0.000000
Shekel 10 loc.min.	4	-4.082606	-3.854712	-4.381017	-2.115418	-10.536400
Griewank	2	0.007403	0.046938	0.007725	0.020060	0.000000
Schubert	2	-186.465600	-160.424300	-174.345700	-156.514200	-186.730900
Schubert pen.1	2	-51.063990	-124.341100	-154.006500	-149.482100	-186.730900
Schubert pen.2	2	-49.587810	-149.224200	-133.498900	-86.081300	-186.730900
5^n loc. min.	2	0.483878	0.008453	0.054982	1.775547	0.000000
5^n loc. min.	5	0.511075	0.060517	0.960994	1.825731	0.000000
5^n loc. min.	10	0.973828	1.443995	1.069051	5.748601	0.000000
5^n loc. min.	20	2.458566	1.402073	1.932469	5.430963	0.000000
10^n loc. min.	2	7.692564	1.778926	0.222846	3.229336	0.000000
10^n loc. min.	5	2.588115	8.820461	6.478716	41.071060	0.000000
10^n loc. min.	10	23.492030	24.490400	23.250480	43.050600	0.000000
10^n loc. min.	20	41.364830	32.757350	19.564760	27.984240	0.000000
15^n loc. min.	2	0.107370	0.020803	0.000830	0.044878	0.000000
15^n loc. min.	5	0.031810	0.008149	0.000311	0.046469	0.000000
15^n loc. min.	10	0.080388	0.018331	0.003361	0.005674	0.000000
Error		5.849909	9.075695	5.686584	11.425308	

direction of the two different basins of attraction of the two global minima, concentrating around them.

The number of starting points for a multistart algorithm represents a critical choice: a large number of starting points gives more confidence about the identification of the global minimum, because a larger number of basins of attraction is potentially investigated, but the computational cost of the algorithm is increased if the number of iterations is fixed, elsewhere we have a small number of iteration of the achievement of the local minimum. As a consequence, two different numerical tests have been produced in order to quantify the effect of the number of starting points.

Comparisons of the results obtained by two different versions of PSO

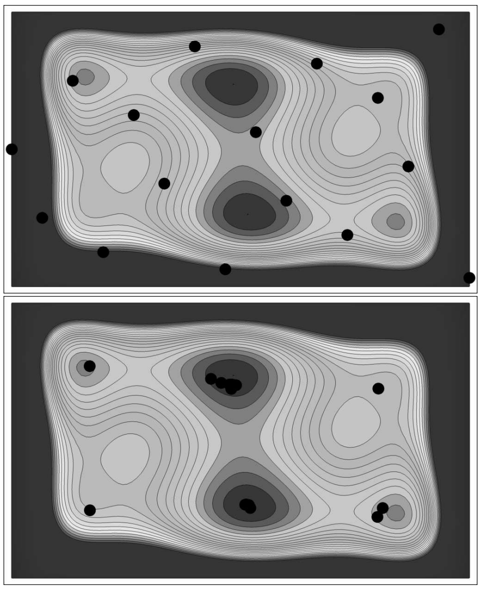

Figure 2. Example of application of MSGS-SM algorithm for the "Six Humps Camel Back" function. On top, position of the starting points. On bottom: final position. In background, the contour levels of the objective function (analytically computed).

(with parameters as in [15]) and MSGS-SM on the Shubert test problem in two different cases are represented in Figures 4 and 5. In particular, for the first case the number of the initial points is $2 \times N$, while for the second case the number of the initial points is $4 \times N$. The effect of the increasing number of starting points seems to be beneficial for the MSGS-SM algorithm: in fact, PSO is more accurate for the lower number of starting points (Figure 4), while the order is swapped when the number of starting points is increased (Figure 5). Accidentally, the precision of the PSO algorithm is decreasing when the number of starting points is increased. Since the stopping criteria is fixed on the base of the global number of objective function evaluations, an increase of the number of starting points is reflected on a decrease of the number of iterations. In this case, we can conclude that PSO is more efficient if the number of iterations of the swarm is higher. On the other hand, the effect of the increased number of starting point is largely beneficial for MSGS-SM: a possible explanation comes with the higher accuracy of the surrogate model. In fact, the initial uniform distribution of points is increasing the quality of the resulting surrogate model since from the start, because a larger and more uniform exploration of the design space is provided from the beginning.

Table 3 reports the problem names, dimensions, the more accurate al-

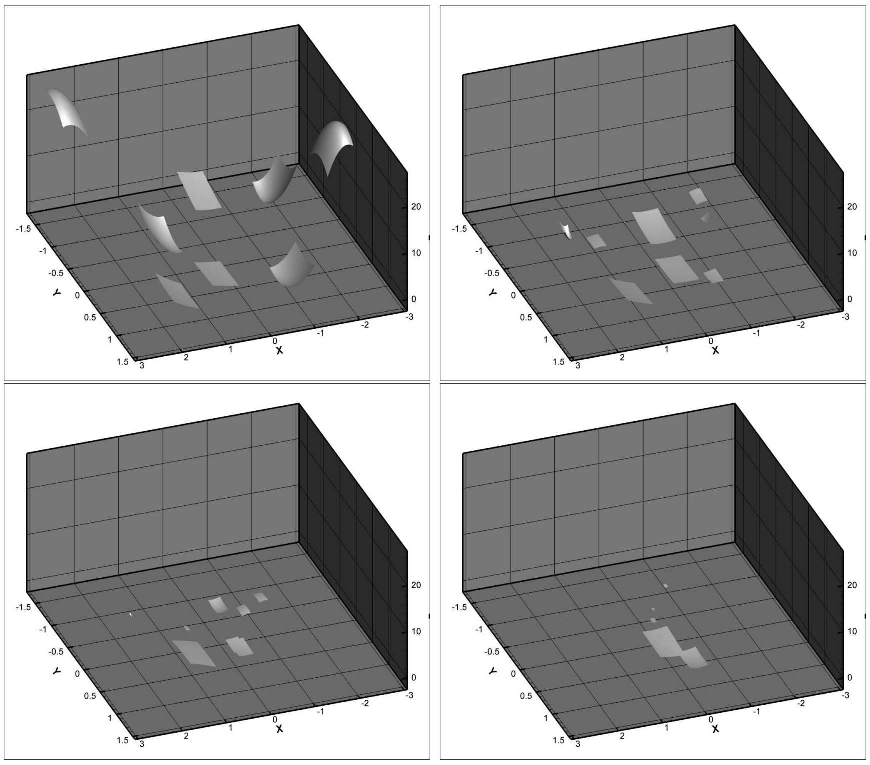

Figure 3. Sequence of the local models adopted for the detection of the minima of the objective function. The objective function is reported for the corresponding value of the design variables.

gorithm when the number of initial points is $2\times N$ and the more accurate algorithm when the number of initial points is $4\times N$.

Observing Table 3, the MSGS-SM outperforms PSO when the number of initial points is $4\times N$ in 26 test problems over 28. Similar situation occurs when the number of initial points is $2\times N$: here MSGS-SM outperforms PSO in 24 test problems over 28. These results are confirming the preliminary consideration drawn observing Figures 4 and 5.

A comparison is performed among MSGS-SM and PSO with $4\times N$ point, in other test case: Six Hump Camel Back test case in Figure 6, Freudenstein and Roth test case in Figure 7, Shubert pen.2 test case in Figure 8.

Table 3. Numerical results. PSO=P [15] and MSGS-SM=M.

Problem	N	$2\times N$	$2\times N$		$4\times N$	$4\times N$	
		(PSO)	(MSGS-SM)		(PSO)	(MSGS-SM)	
Six humps c. b.	2	-9.479080E-1	-10.316280E-1	M	-10.315320E-1	-10.316274E-1	M
Treccani	2	0.434940E-1	0.000000E-1	M	0.001856E-1	0.000004E-1	M
Quartic	4	0.675290E-1	-3.523853E-1	M	-3.479452E-1	-3.523671E-1	M
Cosine mixture	2	-1.147250E-1	-1.966118E-1	M	-1.699445E-1	-1.999999E-1	M
Cosine mixture	4	-0.305050E-1	-3.999991E-1	M	-3.455344E-1	-3.999926E-1	M
Exponential	2	-9.947680E-1	-9.999999E-1	M	-9.999814E-1	-10.000000E-1	M
Exponential	4	-9.797370E-1	-9.999998E-1	M	-9.986687E-1	-9.999998E-1	M
Hartman	3	-0.351709E+1	-0.375166E+1	M	-0.367891E+1	-0.378821E+1	M
Hartman	6	-0.177230E-1	-0.332059E-1	M	-0.268400E-1	-3.322309E+1	M
Goldstein-Price	2	3.394709E+1	0.476678E+1	M	0.333057E-1	0.303772E-1	M
Branin	2	3.991250E-1	3.982143E-1	M	4.037504E-1	3.981867E-1	M
Freud. -Roth	2	3.991250E-1	0.101480E-1	M	491.293600E-1	0.002839E-1	M
Shekel 10 loc.min.	4	-40.826060E-1	-105.363060E-1	M	-38.547120E-1	-59.527054E-1	M
Griewank	2	0.740300E-2	0.913870E-2	P	4.693840E-2	0.300140E-2	M
Schubert	2	-1.864656E+2	-0.685473E+2	P	-1.604243E+2	-1.861671E+2	M
Schubert pen. 1	2	-5.106399E+1	-16.699240E+1	M	-12.434110E+1	-12.706039E+1	M
Schubert pen. 2	2	-4.958781E+1	-4.510490E+1	M	-14.922420E+1	-18.633875E+1	M
5^n loc. min.	2	4.838780E-1	0.120000E-12	M	0.084534E-1	0.424979E-1	P
5^n loc. min.	5	5.110750E-1	3.994055E-1	M	0.605175E-1	1.342643E-1	M
5^n loc. min.	10	9.738280E-1	10.649290E-1	P	14.439950E-1	2.467353E-1	M
5^n loc. min.	20	24.585660E-1	3.956077E-1	M	14.020720E-1	0.971480E-1	M
10^n loc. min.	2	76.925640E-1	2.513270E-1	M	17.789260E-1	0.037131E-1	M
10^n loc. min.	5	25.881150E-1	62.483301E-1	M	88.204610E-1	36.758010E-1	M
10^n loc. min.	10	2.349203E+1	1.924040E+1	M	2.449040E+1	0.884098E+1	M
10^n loc. min.	20	4.136483E+1	6.498911E+1	P	3.275735E+1	0.091156E+1	M
15^n loc. min.	2	1.073700E-1	0.000049E-1	M	0.208033E-1	0.059972E-1	P
15^n loc. min.	5	0.318100E-1	0.251390E-1	M	0.081494E-1	0.159709E-1	P
15^n loc. min.	10	0.803880E-1	0.069515E-1	M	0.183312E-1	0.049360E-1	M

7. Effect of kriging surrogate model on the solving optimization problem.

Application of a surrogate model to a multistart algorithm is motivated by the reduced computational effort when the objective function is expensive to compute. The side effect of this operation is virtually the application of a filter to the objective function, represented by the difference between the kriging interpolation and the true value of the objective function. The extreme effect of the interposition of the surrogate model could be the complete hiding of the true global minimum, if we do not put a sufficient number (a least one) of training points in the neighborhood of the global minimum.

If we exclude the use of surrogate models from MSGS-SM, we obtain the MSGS algorithm, where all the derivatives are computed by using the

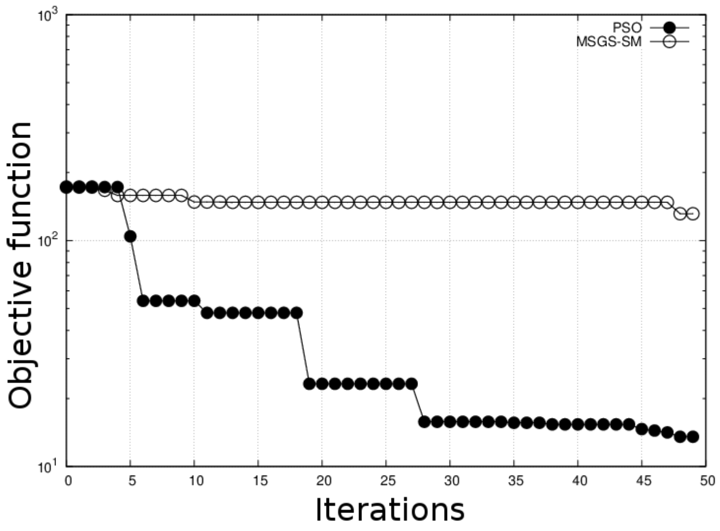

Figure 4. Comparison of the results obtained by two different optimization algorithms on the Shubert test case. Comparison is performed among MSGS-SM (Multi Start Gradient Search applied to the surrogate models) and PSO (Particle Swarm Optimization) with $2{\times}N$ points.

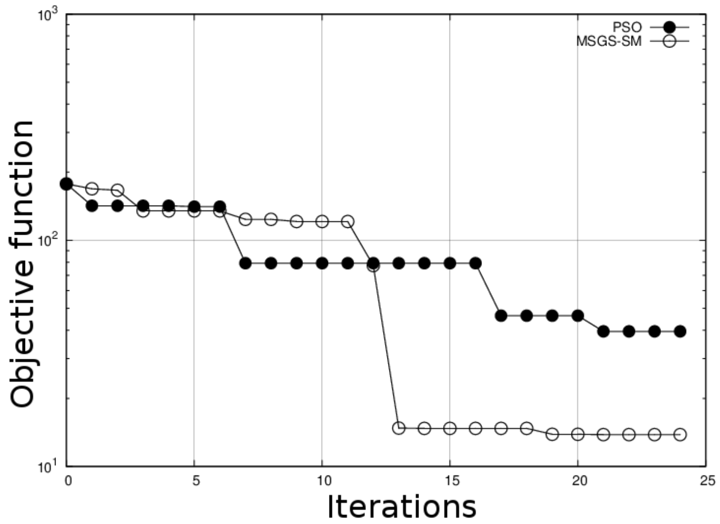

Figure 5. Comparison of the results obtained by two different optimization algorithms on the Shubert test case. Comparison is performed among MSGS-SM (Multi Start Gradient Search applied to the surrogate models) and PSO (Particle Swarm Optimization) with $4{\times}N$ points.

real objective function; also trust region is not applied in this case. MSGS-

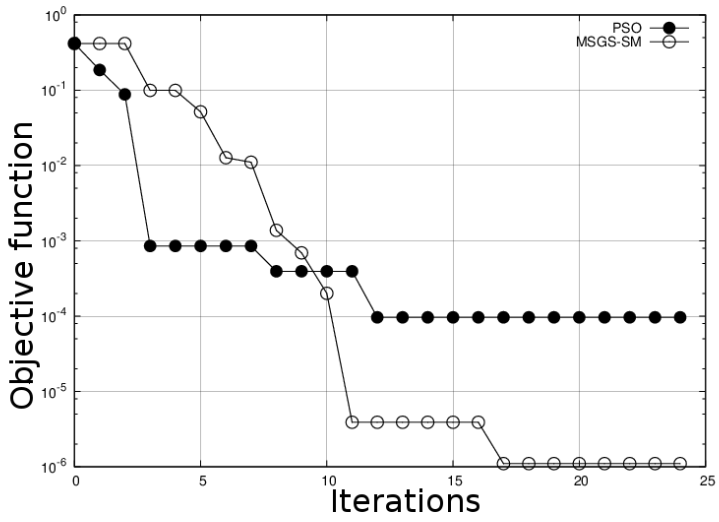

Figure 6. Comparison of the results obtained by two different optimization algorithms on the Six Hump Camel Back test case with. Comparison is performed among MSGS-SM (Multi Start Gradient Search applied to the surrogate models) and PSO (Particle Swarm Optimization) with $4 \times N$ points.

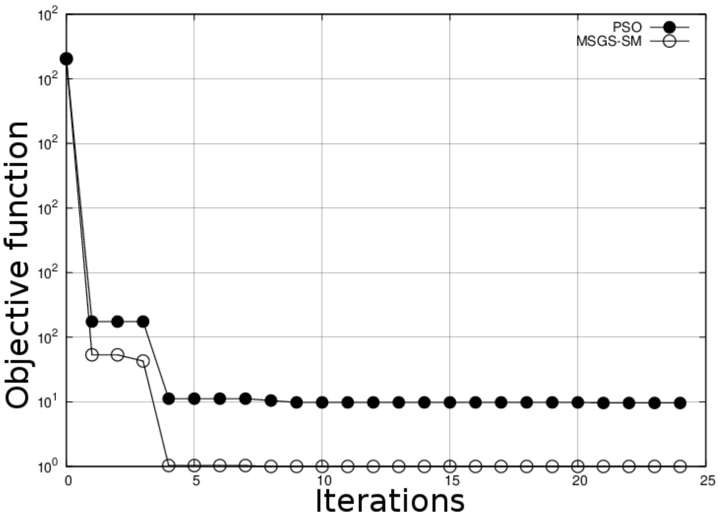

Figure 7. Comparison of the results obtained by two different optimization algorithms on the Freudenstein and Roth test case. Comparison is performed among MSGS-SM (Multi Start Gradient Search applied to the surrogate models) and PSO (Particle Swarm Optimization) with $4 \times N$ points.

SM is here compared with MSGS, in order to check accuracy, speed and

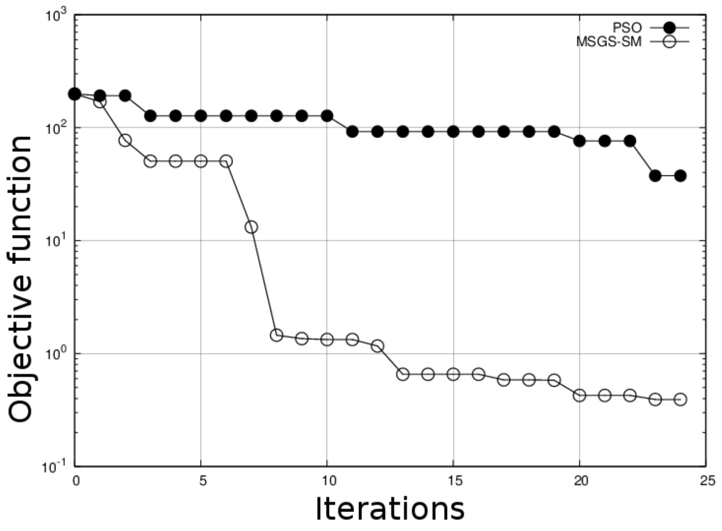

Figure 8. Comparison of the results obtained by two different optimization algorithms on the Shubert pen.2 test case. Comparison is performed among MSGS-SM (Multi Start Gradient Search applied to the surrogate models) and PSO (Particle Swarm Optimization) with $4{\times}N$ points.

convergence of MSGS-SM. As a test case, the Six Hump Camel Back test case is adopted, and results are reported in Figure 9. For graphical purposes, the plotted objective function is shifted in order to have zero minimum value instead of the value reported in Table 1: this shift is not adopted during the solution of the optimization problem. In this figure, logarithmic scale is adopted for the vertical axis, reporting the objective function and horizontal axis is reporting the progressive number of iterations.

Looking at Figure 9, at first glance it seems that MSGS method outperforms MSGS-SM. In particular, the MSGS appears faster than MSGS-SM in approaching the global minimum. MSGS is applied to an exact objective function whereas the MSGS-SM is applied to the kriging model that is an interpolation model, and a surrogate model needs a certain number of training point in order to be accurate. For the aforementioned reasons, the speed of convergence of the MSGM-SM cannot be the same of the MSGS. Once a significant number of training points becomes available, MSGS-SM gains accuracy, and the final result detected by MSGS-SM is absolutely comparable with MSGS. But we have to remember how a single iteration of MSGS has a cost of 13 objective function evaluations (12 are required to compute gradient and Hessian, one for the end point of the line search),

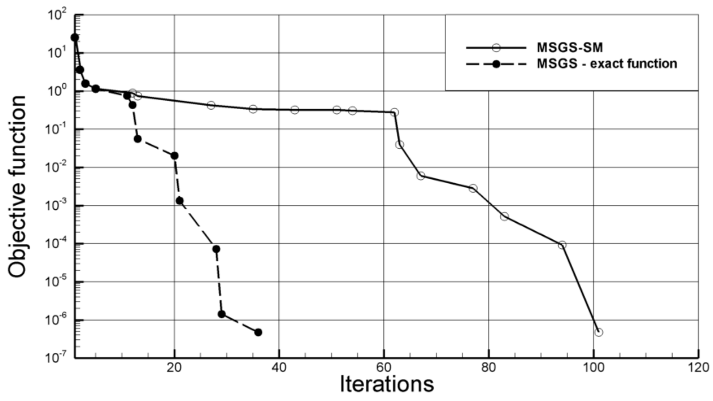

Figure 9. Comparison of the results obtained by two different optimization algorithms on the Six Hump Camel Back test case. Comparison is performed among MSGS-SM (Multi Start Gradient Search applied to the surrogate models) and MSGS (Multi Start Gradient Search applied to the exact objective function).

while the cost is simply one for MSGS-SM. MSGS converges in 36 iterations while MSGS-SM needs 102 iterations. But the total price of MSGS is 36*13=468 evaluations of the objective function per particle, while the total price of MSGS-SM is 102 evaluations per particle: this is lass than a quarter of the total expense of MSGS-SM. As a consequence, MSGS-SM is largely reducing the overall computational cost of the MSGS algorithm, preserving the same accuracy.

8. Conclusions.

A combined use of a global and local approximation models of the objective function is applied to the minimization of a general function. Trust region method is applied in order to certify the quality of the global model. A de-clustering technique is applied in order to preserve diversity in the search. Numerical experiments give the evidence of the capabilities of the proposed methodology.

Numerical experiments are also indicating how the initial iterations of the MSGS-SM algorithm are crucial for the global efficiency of the algorithm: in fact, the initial approximation of the objective function provided by the surrogate model may be very smooth, due to the small number of available training points: as a consequence, all the starting elements could point to the same position, and the algorithm could rapidly collapse. A de-clustering technique, here introduced in the algorithm, is able to recover from this situation, but the initial lack of information about the objective function is slowing down the convergence process.

In order to remove this drawback, the next step will be the investigation of a combined use of MSGS-SM and PSO. In particular, PSO can be used as the main algorithm, and the PSO step could be replaced by the local search when PSO is not producing a descendent step.

Acknowledgements.

This work has been partially supported by the research project *Hybrid Marine* (HyMAR) funded under the EUs Seventh Framework Research Programme (FP7).

REFERENCES

1. G. Matheron, Principles of geostatistics, *Journal of Global Optimization*, vol. 58, no. 8, pp. 1246–1266, 1963.
2. T. Simpson, A. Booker, A. Ghosh, A. Giunta, P. Koch, and R. Yang, Approximation methods in multidisciplinary analysis and optimization: a panel discussion, *Structural and Multidisciplinary Optimization*, vol. 27, no. 5, pp. 302–313, 2004.
3. F. Hickernell and Y. Yuan, A simple multistart algorithm for global optimization, *OR Transactions*, vol. 1, no. 2, pp. 1–12, 1997.
4. S. Sakata and F. Ashida, Approximate global optimziation with convexity estimation of response surface using kriging method, *Structural and Multidisciplinary Optimization*, vol. 40, pp. 417–431, 2010.
5. D. Mayne and C. Meewella, A multistart non-clustering algorithm for global optimization, *Lecture Notes in Control and Information Sciences*, vol. 111, pp. 334–345, 1988.
6. S. Leary, A. Bhaskar, and A. Keane, A constraint mapping approach to the structural optimization of an expensive model using surrogates, *Optimization and Engineering*, vol. 2, pp. 385–398, 2001.
7. S. Leary, A. Bhaskar, and A. Keane, A derivative based surrogate model for approximation and optimizing the output of an expensive computer simulation, *Journal of Global Optimization*, vol. 30, pp. 39–58, 2004.
8. J. Matias and W. Gonzalez-Manteiga, Regularized kriging as a generalization of simple, universal, and bayesian kriging, *Stochastic Environmental Research and Risk Assessment*, vol. 20, pp. 243–258, 2006.
9. N. Alexandrov, J. D. Jr., R. Lewis, and V. Torczon, A trust-region framework for managing the use of approximation models in optimization, *Structural Optimization*, vol. 15, pp. 16–23, 1998.
10. P. Watson and K. Gupta, Em-ann models for microstrip vias and interconnects in dataset circuits, *IEEE Transactions on Microwave Theory and Techniques*, vol. 44, pp. 2495–2503, 1996.

11. K. Chang, R. Haftka, G. Giles, and P. Kao, Sensitivity-based scaling for approximating structural response, *Journal of Aircraft*, vol. 30, pp. 283–287, 1993.
12. D. Peri, Self-learning metamodels for optimization, *Ship Technology Research*, vol. 56, no. 2, pp. 94–108, 2009.
13. J. Kennedy and R. Eberhart, Particle swarm optimization, in *1995 IEEE International Conference on Neural Networks*, pp. 1942–1948, IEEE, 1995.
14. M. Clerc, The swarm and the queen: toward a deterministic and adaptive particle swarm optimization, in *Congress on Evolutionary Computation*, vol. 3-30, pp. 1951–1957, 1999.
15. R. Eberhart and Y. Shi, Comparing inertia weights and constriction factor in particle swarm optimization, in *Congress on Evolutionary Computing*, vol. 1, pp. 84–88, 2000.
16. A. Pinto, D. Peri, and E. Campana, Global optimization algorithms in naval hydrodinamics, *Ship Technology Research*, vol. 51, no. 3, pp. 123–133, 2004.
17. E. Campana, G. Liuzzi, S. Lucidi, D. Peri, A. Pinto, and V. Piccialli, New global optimization methods for ship design problems, *Optimization and Engineering*, vol. 10, pp. 533–555, 2009.
18. E. Campana, G. Fasano, D. Peri, and A. Pinto, Particle swarm optimization: efficient globally convergent modifications, in *III European Conference on Coupled Problems in Engineering* (C. M. Soares, J. Martins, H. Rodrigues, J. Ambrosio, C. Pina, C. M. Soares, E. Pereira, and J. Folgado, eds.), pp. 412–413, Springer, 2006.
19. D. Peri, Collaborative use of a particle swarm optimization algorithm and an adaptive covering method for global optimization, Tech. Rep. 2011-TR-010, CNR-INSEAN, 2001.