Decreasing Computational effort using a new multi-fidelity metamodeling optimization algorithm

Y. Gholipour¹, M. M. Shahbazi²

¹,² Engineering Optimization Research Group, College of Engineering, University of Tehran, Iran, eorg@ut.ac.ir

Abstract
High-fidelity simulation models are used extensively in engineering optimization. However, most of them require a large number of analyses and high computational effort. Consequently, a new multilevel optimization approach is developed for multidisciplinary structural design optimization based on multi-fidelity modeling to decrease computational effort. Such method is a composition of a statistical estimating method reducing the required number of experiments and a continuous variable Metaheuristic algorithm. It starts with an integrated grid of sample points, and then updates them based on the accurate analysis response and replaces low-fidelity to high-fidelity where needed as well as the high-fidelity analyses, which are far enough from the optimum results, would be omitted according to the corresponding low-fidelity responses. Some examples have been presented to show the influence of the algorithm on the computational effort and the results represent higher performance and lesser number of analyses as compared to different optimization techniques like Ant Colony and Genetic Algorithm. It is shown that the algorithm decreases the high-fidelity computational cost, demonstrates the noticeable effectiveness of the presented method.

Keywords: Structural optimization, multi-fidelity modeling, Metamodeling

1. Introduction
Metamodeling techniques have been widely used in structural optimization and design evaluation. These kinds of methods are generally applied for complicated functions to make a more intelligible overview of response function that helps to have a better optimization parameter adjustment. In addition, they help to predict and approximate the response value of the objective function. Since structural optimization problems may have many optimization variables, those which support multivariable problem definition are usable in this field of study. Some of well-known metamodeling methods are Inverse Distance Weighting (IDW), Polynomial Regression (PR), Moving Least Squares (MLS), Kriging (KG), Multivariate Adaptive Regression Splines (MARS) and Radial Basis Function (RBF). The first model has been introduced in this research and has been combined with a well-known meta-heuristic algorithm Harmony Search (HS). Besides, the IDW algorithm ha been chosen because of its simplicity to be developed and support of multi-variable problems. Moreover, the HS algorithm has been chosen because it supports continuous design variables.

2. Inverse distance weighting model (IDW)
This model is a multivariable model based on interpolation, which is well suited for irregularly spaced data. Existing methods of two-dimensional exact interpolations are of two types: a single global function, often of unmanageable complexity; or a suitably-defined collection of simple, local functions which match appropriately at their boundaries. The IDW method developed in this paper, is of the latter type. Such function is continuously differentiable even at the junction of local function. The mentioned method follows simple concepts and its development has the same simplicity. This method has its own disadvantages compare to similar methods. When the number of sample points increases, calculation of approximate response function at the point P will be very time consuming. The value at any point P in the plane was a weighted average of the values at the data points Dᵢ which, the weighting is a function of distance to those points. Donald Shepard [11] has introduced the interpolated value at P in Eq (1).

\[
f_i(P) = \begin{cases} 
\frac{\sum_{i=1}^{N} d_i^{-u} z_i}{\sum_{i=1}^{N} d_i^{-u}} & d_i \neq 0, u > 0 \\
0 & d_i = 0 
\end{cases}
\]

(1)

Where \(z_i\) is the value at data point Dᵢ, \(d(P,D_i)\) is the Cartesian distance between P and Dᵢ will be shortened to \(d_i\). As P approaches a data point Dᵢ, \(d_i\) approaches to zero. Therefore \(\lim_{P \rightarrow D_i} f_i(P) = z_i\) and the function \(f_i(P)\) is...
continuous. Using coordinates \( P(x,y) \) and \( D_i(x_i,y_i) \), partial differential yields Eq (2).

\[
\sum_{i=1}^{N} \sum_{j=1}^{N} \left( d_{ij} \right)^{-u-2} \left( \begin{array}{c} d_j \\ -u \\ (x-x_i) \end{array} \right) \left( \begin{array}{c} z_i \\ -u \\ (z_i-z_j) \end{array} \right)
\]

\[
f_{1x}(x,y) = \frac{\sum_{i \neq j}^{N} \left( d_{ij} \right)^{-u-2} \left( \begin{array}{c} d_j \\ -u \\ (x-x_i) \end{array} \right) \left( \begin{array}{c} z_i \\ -u \\ (z_i-z_j) \end{array} \right)}{\sum_{i=1}^{N} \left( d_{ij} \right)^{-u-2}}
\]  

Replacing \((x-x_i)\) by \((y-y_i)\) in the Eq (2) gives the corresponding expression for \( f_1y(x,y) \). The partial derivative of \( f_i \) exists at all points. For \( u>1 \), \( f_{ix} \) approaches 0 with \( x-x_i \) (or with \( d_i \)) as \( P \to D_i \). For \( u=1 \), both left and right partial derivatives exist and for \( u<1 \), no derivative exists. Empirical tests showed that higher exponents \((u>2)\) tend to make the surface relatively flat near all data points, with very steep gradients over small intervals between data points. Lower exponents produce a surface relatively flat, with short blips to attain the proper values at data points. An exponent of \( u=2 \) not only gives seemingly satisfactory empirical results for purposes of general surface mapping and description, but also presents the easiest calculation. In Cartesian coordinates

\[
d_{ij}^{-2} = \frac{1}{\left( (x-x_i)^2 + (y-y_i)^2 \right)^{u/2}}.
\]

The above method has several shortcomings. When the number of data points is large, the calculation of \( z=f_i(P) \) becomes proportionally longer. As a result the method will become impractical; the directions to \( P \) from data points \( D_i \) are not considered in weighting function; computational errors becomes more significant in the neighborhood of points \( D_i \), as the predominant term results from the difference of two almost equal numbers. To improve the mentioned shortcomings, Shepard [11] made some changes on the Eq (1). He selected the nearby points to get the weighting function. To select the nearby points, either (1) and arbitrary distance criterion (all data within some radius \( r \) of the point \( P \)), or (2) an arbitrary number criterion (the nearest \( n \) data points) could be employed. Since an unmanageable number of data points might be found within radius \( r \), as well as the latter is harder than the former, both methods were employed as Eq (3) in two-dimensions problems as Shepard advised. However, the latter choice requires more detailed searching and ranking procedure for data points.

\[
\pi r^2 = C \left( \frac{A}{N} \right)
\]

Where \( A \) is the area of the largest polygon enclosed by the data points and \( N \) is the total number of data points and \( C \) is the number of data points are included in the circle with radius \( r \) (Shepard advised \( C=7 \)).

\[
r^n = C \frac{\prod_{i=1}^{n} \left( x_i^{mn} - x_i^{nn} \right)}{N}
\]

The equation may be generalized to \( n \)-dimensional problems as written in Eq (4).

For a reference point \( P \), new weighting factor \( s(d) \) may be defined from the following functions:

\[
s(d) = \begin{cases} 
1 & 0 < d \leq \frac{r}{3} \\
\frac{27}{4r} \left( \frac{d}{r} - 1 \right)^2 & \frac{r}{3} < d \leq r \\
0 & r < d 
\end{cases}
\]

The improved interpolation function \( f_2(P) \) which is similar to the original function \( f_1(P) \), but much easier to be employed for large number of data points is:

\[
f_2(P) = \sum_{d_i \neq 0} \frac{S_i^2 z_i}{\sum_{d_j} S_j^2} d_i \\
\sum_{d_i = 0} z_i
\]

Figure 1. illustrates the influence of the algorithm to the Six-Hump Camel Back function.
It is shown that (d) has a smoother surface than (c), that shows the efficiency of the improvement algorithm on the original function. Although Shepard had more improvement steps on the original function like applying directions and etc., they increase computational effort; therefore, they have not been employed in this research; besides, the new algorithm does not need a very well-fitted approximation.

3. Harmony search algorithm
The HS algorithm was based on natural musical performance processes that occur when a musician searches for a better state of harmony [21] and find musically pleasing harmony (a perfect state) just as the optimization process seeks to find a global solution (a perfect state) as determined by an objective function.
The pitch of each musical instrument determines the aesthetic quality, just as the objective function value is determined by the set of values assigned to each decision variable. Figure 2 shows the optimization procedure of the HS algorithm. Optimization problem is defined as follows:

$$\text{Minimize } f(x)$$

$$\text{Subject to } x_i \in X_i, \ i = 1, 2, \ldots, N$$

Where \( f(x) \) is objective function, \( x_i \) is optimization variable, \( X_i \) is the set of possible range of each decision variable and \( N \) is the number of decision variables. First of all the HS algorithm parameters that are required to solve the optimization problem Eq (7) are specified: harmony memory size (number of solution vectors, HMS), harmony memory considering rate (HMCR), pitch adjusting rate (PAR), and termination criterion (maximum number of searches). Here, HMCR and PAR are parameters that are used to improve the solution vector. In the next step, the Harmony Memory (HM) matrix shown in Eq (8) is filled with as many randomly generated solution vectors as the size of the HM (HMS) and sorted by the values of the objective function.

$$HM = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^{\text{HMS}} \end{bmatrix}$$

After initializing the HM, the algorithm will generate a new harmony from the HM new harmony vector based on HMCR, PAR and randomization. For instance, the value of the first decision variable \( x'_1 \) for the new vector can be chosen from any value in the specified HM range \( \{ x'^1, x'^2, \ldots, x'^{\text{HMS}} \} \). Values of the other decision variables \( x'_i \) can be chosen in the same manner. Here, there is a possibility that the new value can be chosen using the HMCR parameter, which varies between 0 and 1 as follows shown in Eq (9).

$$x'_i \left\{ \begin{array}{ll} \in \{ x'^1, x'^2, \ldots, x'^{\text{HMS}} \} & \text{w.p. } \text{HMCR} \\ \in X_i & \text{w.p. } (1-\text{HMCR}) \end{array} \right.$$  

(9)

The HMCR sets the rate of choosing one value from the historic values stored in the HM, and \((1-\text{HMCR})\) sets the rate of randomly choosing one value from the possible range of values. Every component of the new harmony vector \( x' = (x'_1, x'_2, \ldots, x'_N) \) is examined to determine whether it should be pitch-adjusted. This procedure uses the PAR parameter that sets the rate of adjustment for the pitch chosen from the HM as follows:

$$x'_i \left\{ \begin{array}{ll} \text{YES} & \text{w.p. } \text{PAR} \\ \text{No} & \text{w.p. } (1-\text{PAR}) \end{array} \right.$$  

(10)

The solution vector is considered if is better than the worst vector in the HM list and will be substituted with it and finally, the HM will be reordered according to the response value. The computation are terminated when the termination criterion is satisfied. If not the last two steps will be repeated.

4. The new multi-fidelity algorithm

The optimization method which is presented in this research is a combination of Inverse Distance Weighting (IDW) as an approximation algorithm and Harmony Search (HS). The main purpose of presenting this method is reduction of the computational effort in optimization problems. Figure 3 show the optimization procedure of the proposed algorithm. The algorithm has four steps:

Step 1: The algorithm initializes the sample points of solution vectors and calculates the response values using a high-fidelity analysis. These points are employed in both IDW algorithm as initial data points and HS algorithms as harmony memory (HM).

Step 2: In this step, the algorithm generates a new solution vector based on possible range \( X_i \) as mentioned in HS algorithm and then calculates the corresponding approximate response value using the IDW algorithm. The algorithm will go to the next step, if the approximate value is better than the worst value stored in HM list. If not, another solution vector is generated.
Figure 3. Optimization procedure of the proposed algorithm

Step 3: A high-fidelity analysis is applied on the new solution vector. If the response value is still better than the worst vector in HM, it is substituted with the new vector and finally the new HM list is reordered.

Step 4: If the termination criterion is satisfied, the algorithm finishes. Otherwise the steps 2 and 4 will be repeated.

Numerical examples show that most of the unnecessary calculations are omitted in step 2. In other words, only the solution vectors, which tend to convergence is employed in accurate analysis. This process has a noticeable influence in decreasing the computational effort.

5. Numerical examples
Some standard examples are considered in this research to evaluate the efficiency of the algorithm on computational effort. The OpenSees software has been used in these examples.
5.1. 10-bar truss
Figure 4 shows the geometry, loading and support conditions of two-dimensional ten-bar truss. Each member is numbered by 1 to 10.

![Figure 4. Configuration of 10 bar truss](image)

The material employed in this structure has modulus of elasticity of $E=10^7$ psi, density of $\rho=0.1 \frac{lb}{in^3}$, maximum allowable stress in each member is $\sigma_{max} = \pm 25 \text{ ksi}$, maximum nodal displacement in both directions is $\delta_{max} = 2 \text{ in}$ and possible value range for each design variable (cross section area) is $0.1 \text{ in}^2 \leq A \leq 35 \text{ in}^2$. Figure 5 illustrates the convergence diagram of new algorithm in comparison with other ordinary algorithms, such as Harmony Search (HS), Genetic Algorithm (GA) and Ant Colony Optimization (ACO).

![Figure 5. Convergence diagram of ten bar truss optimization using Harmony Search and the new algorithm](image)

As shown in Figure 5 and Table 1, the new algorithm causes a noticeable decrease in required number of analysis to reach to optimum value. For example, ACO algorithm [8] designed the truss with a weight of 4994 lb. This design was found using 12000 analyses. Such weight is a result of about 516 analyses (4.3%) in new algorithm. In addition, the GA [26] results a weight of 5076 lb using 15000 analyses.

The similar analysis number in the new algorithm is about 310 analyses (2%). The HS algorithm results a weight of 4982 lb in 5000 analysis. The equivalent number of analysis for that weight in new algorithm is about 717 analyses (14.3%). Most of unnecessary analyses happen in the early start of the algorithm, where, the response value of new generated vectors is far enough from the optimum value. The algorithm omits the high-fidelity analysis, replaces it with a low-fidelity approximation and eventually causes a noticeable decrease in computational effort and CPU time.

5.2. 25-bar truss
Geometry and support conditions of 3 dimensional 25-bar truss problems have been shown in Figure 6. The material employed in this structure has the following properties: modulus of elasticity of $E=10^7$ psi; density of $\rho=0.1 \frac{lb}{in^3}$; maximum allowable stress of $\sigma_{max} = \pm 40 \text{ ksi}$; maximum nodal displacement in three dimensions is $\delta_{max} = 35 \text{ in}$; design variable range is $0.1 \text{ in}^2 \leq A \leq 3.4 \text{ in}^2$. 

![Figure 6. Configuration of 25 bar truss](image)
Table 1. Optimum solution vectors of ordinary optimization algorithm compare to new algorithm for ten-bar truss

<table>
<thead>
<tr>
<th>Member</th>
<th>ACO</th>
<th>Camp et al. (1998)</th>
<th>Mahfouz (1999)</th>
<th>Harmony Search</th>
<th>This Research (IDM+HS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>29.90</td>
<td>28.92</td>
<td>33.50</td>
<td>30.05</td>
<td>29.85</td>
</tr>
<tr>
<td>2</td>
<td>0.10</td>
<td>0.10</td>
<td>1.62</td>
<td>0.27</td>
<td>0.53</td>
</tr>
<tr>
<td>3</td>
<td>26.10</td>
<td>24.07</td>
<td>22.90</td>
<td>22.86</td>
<td>22.81</td>
</tr>
<tr>
<td>4</td>
<td>15.40</td>
<td>13.96</td>
<td>14.20</td>
<td>14.58</td>
<td>14.54</td>
</tr>
<tr>
<td>5</td>
<td>0.10</td>
<td>0.10</td>
<td>1.62</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>6</td>
<td>0.50</td>
<td>0.56</td>
<td>1.62</td>
<td>0.28</td>
<td>0.73</td>
</tr>
<tr>
<td>7</td>
<td>20.90</td>
<td>21.95</td>
<td>22.90</td>
<td>20.64</td>
<td>20.58</td>
</tr>
<tr>
<td>8</td>
<td>7.40</td>
<td>7.69</td>
<td>7.97</td>
<td>8.11</td>
<td>7.31</td>
</tr>
<tr>
<td>9</td>
<td>0.10</td>
<td>0.10</td>
<td>1.62</td>
<td>0.24</td>
<td>0.10</td>
</tr>
<tr>
<td>10</td>
<td>18.70</td>
<td>22.09</td>
<td>22.00</td>
<td>20.69</td>
<td>21.01</td>
</tr>
<tr>
<td>Weight</td>
<td>4994</td>
<td>5076</td>
<td>5491</td>
<td>4982</td>
<td>4963</td>
</tr>
<tr>
<td>Analysis</td>
<td>12000</td>
<td>15000</td>
<td>8000</td>
<td>5000</td>
<td>1870</td>
</tr>
</tbody>
</table>

Table 2. Single load case for 25-bar truss

<table>
<thead>
<tr>
<th>Node</th>
<th>Fx (kips)</th>
<th>Fy (kips)</th>
<th>Fz (kips)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0</td>
<td>-10.0</td>
<td>-10.0</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>-10.0</td>
<td>-10.0</td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>0.6</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 3. Multiple load case for 25-bar truss

<table>
<thead>
<tr>
<th>Case</th>
<th>Node</th>
<th>Fx (kips)</th>
<th>Fy (kips)</th>
<th>Fz (kips)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1.0</td>
<td>10.0</td>
<td>-5.0</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>10.0</td>
<td>-5.0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.5</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.0</td>
<td>20.0</td>
<td>-5.0</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>-20.0</td>
<td>-5.0</td>
<td></td>
</tr>
</tbody>
</table>

In this example, designs for both a single load case and a multiple load case are developed and the results are compared with traditional optimization techniques. Table 2 and Table 3 Show the single and multiple load conditions of 25 bar truss problem respectively.

Figure 6. Configuration of 25-bar truss

Figure 7 illustrates the convergence diagram of 25-bar truss for single load condition; in addition, it makes a comparison between the results of new algorithm and ordinary algorithms.
The algorithm has a significant effect on decreasing computational effort. For example, ACO algorithm designed the truss with a weight of 484.85 lb in 7700 analyses. The new algorithm got the same result in about 384 analyses that are only 5% of the ACO result; in addition, Genetic Algorithm got a weight of 485.05 lb in 15000 analyses which, the new algorithm reached the same weight in 382 analyses (2.54% of GA). Compare to HS algorithm which designed the truss with a weight of 482.24 lb in 5000 analyses, the new algorithm got the same result in about 542 analyses (10.8%) which has a considerable decrease in analysis number. Table 4 shows the best solution vectors earned by new algorithm and ordinary algorithms. It can be seen that the optimum design in the new algorithm is lighter than the other algorithms with lower analysis number. As mentioned above, the second load case for 25-bar truss is a multiple load case.

Table 4. Optimum weight of 25-bar truss under single load case

<table>
<thead>
<tr>
<th>Group</th>
<th>Members</th>
<th>ACO</th>
<th>Camp et al. (GA)</th>
<th>Harmony Search</th>
<th>Current Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0.10</td>
<td>0.10</td>
<td>0.12</td>
<td>0.28</td>
</tr>
<tr>
<td>2</td>
<td>2-5</td>
<td>0.30</td>
<td>0.50</td>
<td>0.17</td>
<td>0.30</td>
</tr>
<tr>
<td>3</td>
<td>6-9</td>
<td>3.40</td>
<td>3.40</td>
<td>3.40</td>
<td>3.40</td>
</tr>
<tr>
<td>4</td>
<td>10-11</td>
<td>0.10</td>
<td>0.10</td>
<td>0.12</td>
<td>0.11</td>
</tr>
<tr>
<td>5</td>
<td>12-13</td>
<td>2.10</td>
<td>1.90</td>
<td>1.23</td>
<td>1.73</td>
</tr>
<tr>
<td>6</td>
<td>14-17</td>
<td>1.00</td>
<td>0.90</td>
<td>0.82</td>
<td>0.91</td>
</tr>
<tr>
<td>7</td>
<td>18-21</td>
<td>0.50</td>
<td>0.50</td>
<td>0.94</td>
<td>0.53</td>
</tr>
<tr>
<td>8</td>
<td>22-15</td>
<td>3.40</td>
<td>3.40</td>
<td>3.35</td>
<td>3.39</td>
</tr>
<tr>
<td>Weight</td>
<td></td>
<td>485</td>
<td>485</td>
<td>482</td>
<td>476</td>
</tr>
<tr>
<td>Analysis</td>
<td></td>
<td>7700</td>
<td>15000</td>
<td>5000</td>
<td>2838</td>
</tr>
</tbody>
</table>

Figure 8 shows the convergence diagram of 25-bar truss under multiple load case. As shown in Table 5, the ACO algorithm results a weight of 545.53 lb in 16500 analyses, while the new algorithm has the same result in 389 analyses (2.36%); moreover, the GA shows a weight of 545.80 lb in 15000 analyses; therefore the new algorithm has the same result in only 2.6% of analysis compare to GA. Although The HS algorithm shows a weight of 559.45 lb in 5000 analyses, yet the new algorithm gets the same weight in 153 analyses (3% of HS). It is noticeable that, the new algorithm has the lighter result in all algorithms that shows the efficiency of the algorithm on computational load.

Table 5. Optimum weight of 25-bar truss under multiple load case

<table>
<thead>
<tr>
<th>Group</th>
<th>Member</th>
<th>ACO</th>
<th>Camp et al. (GA)</th>
<th>Harmony Search</th>
<th>Current Research</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0.01</td>
<td>0.01</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>2</td>
<td>2-5</td>
<td>2.00</td>
<td>2.01</td>
<td>1.65</td>
<td>2.05</td>
</tr>
<tr>
<td>3</td>
<td>6-9</td>
<td>2.97</td>
<td>2.95</td>
<td>3.12</td>
<td>2.97</td>
</tr>
<tr>
<td>4</td>
<td>10-11</td>
<td>0.01</td>
<td>0.01</td>
<td>0.13</td>
<td>0.19</td>
</tr>
<tr>
<td>5</td>
<td>12-13</td>
<td>0.01</td>
<td>0.03</td>
<td>0.11</td>
<td>0.12</td>
</tr>
<tr>
<td>6</td>
<td>14-17</td>
<td>0.69</td>
<td>0.68</td>
<td>0.53</td>
<td>0.67</td>
</tr>
<tr>
<td>7</td>
<td>18-21</td>
<td>1.68</td>
<td>1.68</td>
<td>2.04</td>
<td>1.57</td>
</tr>
<tr>
<td>8</td>
<td>22-15</td>
<td>2.67</td>
<td>2.68</td>
<td>2.80</td>
<td>2.64</td>
</tr>
<tr>
<td>Weight</td>
<td></td>
<td>545.53</td>
<td>545.80</td>
<td>559.45</td>
<td>542.14</td>
</tr>
<tr>
<td>Analysis</td>
<td></td>
<td>16500</td>
<td>5000</td>
<td>1492</td>
<td></td>
</tr>
</tbody>
</table>
6. Conclusion
A new multi-fidelity optimization algorithm proposed here which, is invented by combination of a response approximation method and a meta-heuristic optimization algorithm. Since most of meta-heuristic algorithms need a large number of iterations to reach an optimum solution, the new algorithm has been proposed here to decrease computational effort. In addition, it can be used to get a better layout of objective function using the approximation part of algorithm. Although the approximation algorithm is employed to omit unnecessary high-fidelity analyses, yet it is not very important to have an accurate approximation algorithm. The numerical examples has been considered in this research, show a significant decrease in computational load for standard truss examples compare to ordinary optimization algorithms such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Harmony Search (HS) and etc. Besides, the new algorithm has better results in all case studies with lesser number of analyses in the same conditions that shows high efficiency of the algorithm on the optimization problems. As shown above, the algorithm has decreased the number of analysis from 0.5% to 10.8%.

References
[5] Solving optimization problems in metal forming using Finite Elements simulation and metamodeling techniques
[6] A study on the use of Kriging models to approximate deterministic computer models
[9] A study of alternative formulations for optimization of structural and mechanical systems subjected to static and dynamic loads