International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:1, No:2, 2007

Genetic Combined with a Simplex Algorithm as an Efficient Method for the Detection of a Depressed Ellipsoidal Flaw using the Boundary Element Method

Clio G. Vossou, Ioannis N. Koukoulis, and Christopher G. Provatidis

***Abstract*—**The present work encounters the solution of the defect identification problem with the use of an evolutionary algorithm combined with a simplex method. In more details, a Matlab implementation of Genetic Algorithms is combined with a Simplex method in order to lead to the successful identification of the defect. The influence of the location and the orientation of the depressed ellipsoidal flaw was investigated as well as the use of different amount of static data in the cost function. The results were evaluated according to the ability of the simplex method to locate the global optimum in each test case. In this way, a clear impression regarding the performance of the novel combination of the optimization algorithms, and the influence of the geometrical parameters of the flaw in defect identification problems was obtained.

***Keywords*—**Defect identification, genetic algorithms, optimization.

I. Introduction

EVOLUTIONARY methods in general, and more specifically genetic algorithms (GA), are optimization methods (OM) that were used in a vast majority of optimization problems. The main feature of such applications is the fact that the design process of artificial systems like structural or mechanical components is simulated by biological processes based on heredity principles (genetics) and the natural selection (the theory of evolution) [1].

Inverse problems, particularly crack or void identification problems, can be stated as an optimization task. Inverse problems are defined as the problems where the output is known and the input or source of output remains to be determined. They are contrary to the direct problems, in which output or response are determined using information from input [2]. In the case of the Inverse Elastostatics Problem (IESP) of internal flaw detection, the location, the orientation and the size of the flaw are unknown but the displacements along the boundaries are known. In order to analyse this kind of problems, the boundary displacements, usually called "experimental data", are obtained under known boundary conditions and compared with the calculated ones.

The boundary element (BEM) and the finite element (FEM) methods are the two computational methods mainly used to obtain the displacement field. The FEM is a well-established procedure for structural analysis and has formed the basis of most early inverse methods. On the other hand, the BEM has become a popular alternative, possessing many advantages, like meshing only on the boundaries and low computational effort, over the more established FEM [3]. Especially, in inverse problems these advantages are important, since each iteration step requires the reconstruction of the mesh and a new numerical solution.

The standard BEM equation for static analysis is: $\mathbf{G}(z)\cdot\mathbf{u}(z)=\mathbf{H}(z)\cdot\mathbf{t}(z)$ where vectors **u** and **t** include all boundary displacements and tractions while **G** and **H** are suitable influence matrixes. Theoretical details about the BEM can be found, among others, in [4].

In 1997, Koghuchi and Watabe [5] used GA in order to improve the search of defects in structures using the BEM. In their paper an elastic structure involving several defects of circular and elliptical shapes is considered. The position, the size, the shape and the number of defects were unknown. The results were encouraging in the sense that they managed to identify the correct location of the defect and orientation as well as their number. However, a large number of generations, consequently of function evaluations, were needed [5]. Evolutionary algorithms have been also used in crack identification in rotors [6], and in shape optimization [1]. Burczynski et al. used distributed evolutionary algorithms while Shim and Suh [7] used a parallel computing environment for the performance of crack identification in beams using evolutionary algorithms. Both these studies used distributed and parallel computing to speed up the used algorithms since they are characterized by a large number of evaluations until convergence is achieved.

The idea of the sequential combination of two algorithms with different features is not new. Simulated annealing has

Manuscript received February 21, 2007. This work is co-funded by the European Social Fund (75%) and National Resources (25%) - Operational Program for Educational and Vocational Training II (EPEAEK II) and particularly the Program PYTHAGORAS.

C. G. Vossou is with the School of Mechanical Engineering, National Technical University of Athens, GR-15773 Athens, Greece (e-mail: cvossou@central.ntua.gr).

I. N. Koukoulis, is with the School of Mechanical Engineering, National Technical University of Athens, GR-15773 Athens, Greece (e-mail: jkouk@mail.ntua.gr).

C. G. Provatidis is with the School of Mechanical Engineering, National Technical University of Athens, 9 Iroon Polytecneiou Avenue, GR-15773 Athens, Greece (phone: +30-210-7721520; fax: +30-210-7722347; e-mail: cprovat@central.ntua.gr).

been used combined with GA and artificial neural networks by Plumb et al. [8] in the problem of optimum of a tablet coating formulation requiring minimization of crack velocity and maximization of film opacity. Furthermore, GA has been also combined with neural networks in the crack identification problem [9,10].

The objective of the present work is to present (a) the different performance of GA as stand-alone and of the combination of GA with a simplex method in a defect identification problem, as well as (b) the investigation of the influence of the use of different amount of data, in the cost function, in the above hybrid optimization method. Furthermore, the influence of the location of the center and the orientation of the ellipsoidal flaw to be identified is investigated.

II. Optimization Algorithms

For the purposes of this work, the GA as stand-alone and in combination with *Fminsearch*, as implemented in the optimization toolbox of Matlab [11], was tested. By the term "combination", it is meant that the GA optimization is performed for a standard number of generations and sequentially the Fminsearch is used using the outcome of GA as a starting vector.

Both GA and Fminsearch were used as unconstrained OMs. However, the constraints were introduced into the objective function in a weighted manner through a penalty. Specifically, in the current work a simple scheme was used; if either a small or a large constraint violation occurred then the objective function obtained a very high value. A brief description for GA and Fminsearch follows.

A. Genetic Algorithms

The idea of the GA was firstly introduced by John Holland in the 1970's [12]. A genetic algorithm is a general optimization method that can be used for various kinds of optimization problems.

The general idea of this method is to resemble an "algorithm" that is used in natural evolution processes. The algorithm operates on a set of designs, called population in a current generation, and the approach is to allow its individuals, i.e. designs, to reproduce and cross among themselves in order to obtain designs with better fitness. The fittest designs, i.e. those with low objective function values in the case of minimization, have good genetic characteristics and these are given higher probability of becoming chosen as parents to progenies, i.e. new designs where the characteristics of the parents are combined. A population of N feasible random designs is initially generated where each design is represented by a binary string of 0's and 1's (binary encoding) or by its numerical value (real encoding). The objective function value for each design is calculated and used to compute the corresponding fitness value (a low objective function value imply a higher fitness value). Four basic operators are now used to generate the next generation of designs: reproduction, crossover and mutation and migration.

Reproduction is an operator that basically selects designs from the population at the current generation and transfers them into a mating pool, i.e. a new population of same size, N. More fit designs have higher probability of getting selected and the same design can be selected more than once.

The idea of *crossover* is to generate new designs by exchanging characteristics of designs from the mating pool. Starting and ending positions in two randomly selected design strings from the mating pool are therefore selected using random numbers. The strings between these positions on the two design strings are then exchanged and the two new designs, i.e. progenies, replace their parents in the mating pool.

Mutation is used to generate new designs by a mutation of existing designs. This is accomplished by changing the digit, i.e. 0 to 1 or vice versa in binary encoding, at a random location in a number of randomly selected design strings from the mating pool. The process of operating and updating the population is continued until a stopping criterion is satisfied.

Finally, *migration* specifies how individuals move between subpopulations. When migration occurs, the best individuals from one subpopulation replace the worst individuals in another subpopulation. Individuals that migrate from one subpopulation to another are copied. They are not removed from the source subpopulation.

As mentioned above, the GA implemented in this work belongs to the optimization toolbox of Matlab. The values of the genetic parameters have been determined through a series of runs for algorithm performance comparison and have the values that are mentioned below [13]. The specific implementation has the following structure:

1. Creation of a starting population of 24 individuals. Every individual is created in real encoding with the use of a random number generator
2. Evaluation of every individual.

2.1 Sorting of the individuals according to the value of the function.

2.2 Calculation of the average fitness.

3. Creation of a mating pool.

3.1 Scaling of the raw scores based on the rank of each individual instead of its score. The rank of an individual is its position in the sorted scores. The rank of the fittest individual is 1; the next most fit is 2, and so on.

3.2 Making of copies of the fittest members for reproduction. Selection is made with the use of stochastic uniform function; this way of selection lays out a line in which each parent corresponds to a section of the line of length proportional to its scaled value. The algorithm moves along the line in steps of equal size. At each step, the algorithm allocates a parent from the section it lands on. The first step is a uniform random number less than the step size.

4. Performance of the crossover operation with the probability *0.8* and give birth to the new individuals.
5. Mutation of some of the individuals.
6. Every *20* generations migration is performed. The direction of migration is set to forward meaning that migration takes place toward the last subpopulation and the migration fraction is set to *0.2*.
7. If the maximum number of generations (*450* for just the GA and *150* for the combination of methods) is reached the algorithm is terminated else go to step 3.

B. Fminsearch

The algorithm of Fminsearch uses the Simplex search method of [14]. This is $X=X_0-l_0, X=X_0+l_0, X=-X_0$ a direct search method that does not use numerical or analytical gradients. If n is the length of the design variables vector, a simplex in n-dimensional space is characterized by the n + 1 distinct vectors that are its vertices. In two-dimensional space, a simplex is a triangle while in three-dimensional space, it is a pyramid. Each simplex defines n+1 solutions in the search space. The simplexes throughout the optimization, using this method, can be expanded, contracted or reflected. These three moves are defined respectively as: where X_0 is the vector of the starting vertex, or of the vertex to be changed. As it is obvious, during a reflection the volume of the simplex is invariant. There are of course several combinations of the above moves.

The structure of the Fminsearch method proposed for the combination with GA is as follows.

1. At each step of the search, a new point in or near the current simplex is generated.
2. The function value at the new point is compared with the values of the function at the vertices of the simplex and, usually, one of the vertices is replaced by the new point, giving a new simplex.

This step is repeated until the diameter of the simplex is less than the specified tolerance.

III. Description of the Test Cases

All the test cases consist of a plane strain plate with one depressed ellipsoidal flaw. The plate is square with edge L=10 and the material constants are the shear modulus G=1×10^5 and Poisson's ratio v = 0.30. The ellipsoidal flaw has axis ratio 1/10, meaning that the semimajor axis is 1.00 and the semiminor axis is 0.10. All quantities are in compatible units.

The flaw is introduced to be in a 4×4 pattern that leads to 16 different locations and to have 4 different angles (-45^o, 0^o, 45^o and 90^o) as shown in Fig. 1.

Moreover, the left-hand side external edge (ad) is fixed in both directions and a tension loading is applied on the right-hand side edge (bc) with a value equal to P = 1000, in the horizontal Ox coordinate direction.

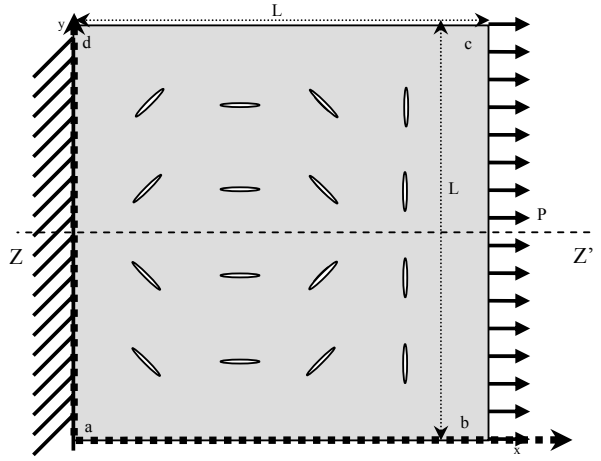


Fig. 1 Geometry and boundary conditions of the test cases with angle -45^o, 0^o, 45^o, and 90^o, from left to right (true scale)

Obviously, the test cases with ellipse orientation -45^o on the lower half of the plate, which is the area under investigation, are symmetric to those with 45^o orientation on the upper half, and vice-versa. Due to symmetry along the line ZZ' (y=5) only the 8 cases of different locations of the center are encountered and a total of 32 different IESP has been solved.

For the solution of the direct problem using the BEM, the boundaries of the plate are discretized by means of quadratic boundary elements. In all test cases presented here, the external boundary is discretized by means of 40 boundary elements (i.e. a total of 80 nodes) and the elliptical flaw is discretized by means of 8 boundary elements (i.e. additional 16 nodes).

The coordinates of the eight different center locations are 22(2.00,2.00), 24(2.00,4.00), 42(4.00,2.00), 44(4.00,4.00), 62(6.00,2.00), 64(6.00,4.00), 82(8.00,2.00), 84(8.00,4.00). The data concerning the possible location and orientation of the flaw was selected in order to be representative of what happens on the whole area of the plate.

For the formulation of the identification problem as an optimization process, the spatial coordinates of the centre of the ellipsoidal flaw (x,y) and the semimajor axis r are considered as design variables, the semiminor axis is considered known and has the value of 0.05.

Two different cost functions were used for the combination of the OM. The first one uses the normal displacements of the nodes on each of the three free edges while the second one uses only the normal displacements of the nodes on the edge that the tensile loading is applied (bc). The cost functions are defined as cost function 1 (CF1):

$$f(u)=1000+ln\left(\begin{array}{l}10^{-6}+\sum_{side\_ab}\left(\Delta u_y\right)^2\Big/_{side\_ab}\,max\left(u_y^{real}\right)^2+\\ \quad\sum_{side\_bc}\left(\Delta u_x\right)^2\Big/_{side\_bc}\,max\left(u_x^{real}\right)^2+\\ \quad\sum_{side\_cd}\left(\Delta u_y\right)^2\Big/_{side\_cd}\,max\left(u_y^{real}\right)^2\end{array}\right) \quad (1)$$

and cost function 2 (CF2):

$$f(u)=1000+ln\left(10^{-6}+\sum_{side\_bc}\left(\Delta u_x\right)^2\Big/_{side\_bc}\,max\left(u_x^{real}\right)^2\right) \quad (2)$$

where Δu_i is the difference between the calculated displacement on ith direction, $u_i^{calculated}$, and the corresponding experimental data, u_i^{real} [15].

The four design variables are used in an automatic procedure to create a suitable mesh for the flaw and the displacements u are calculated at the corresponding boundary for every cost function.

In Fig. 2 the displaced boundaries of the plate and the ellipsoidal flaw (grey) under the abovementioned load are shown. In order to make the displacements visible, they have been scaled by 10.

In the present work, the design variables corresponding to the location of the center lies in the interval of [0.5, 9.5], the semimajor axis in [0.1, 2.0] and the orientation in [–90^o, 90^o] for the individuals of the initial population. The constraints of the optimization problem, which were applied for all the generations, are set so as for all the designs to have a physical

meaning, referring to the fact that the flaw specified by the design variables must lie completely inside the plate. This condition is ensured in the minimization process through the imposition of geometric constraints.

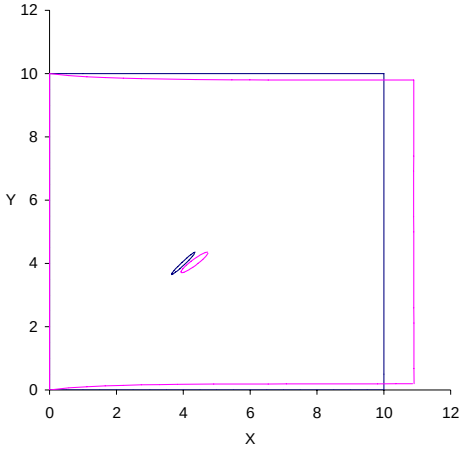


Fig. 2 displaced (grey) and undisplaced (black) plate with a flaw with center location (4.0, 4.0) and orientation 45°

The resulting 32 test have been firstly solved with GA with the use of CF1 and then with the abovementioned combination of the GA and Fminsearch with the use of both cost functions. In all cases the results concern 10 runs for each test case in order to minimize the effect of a particular random seed used in the generation of the initial population of the GA.

IV. Results

Since the main subject of this paper is the investigation of the performance of the combination of the two OM, the results concerning GA are presented briefly. In Fig. 3 the successful runs per location of the center and orientation are presented in a 3D diagram for the use of CF1. For all test cases GA performed 10824 iterations. The height of each cone indicates the number of runs for which the OM identifies successfully the defect. By the term "successful runs" are meant those with variations of the design variables less than 10%.

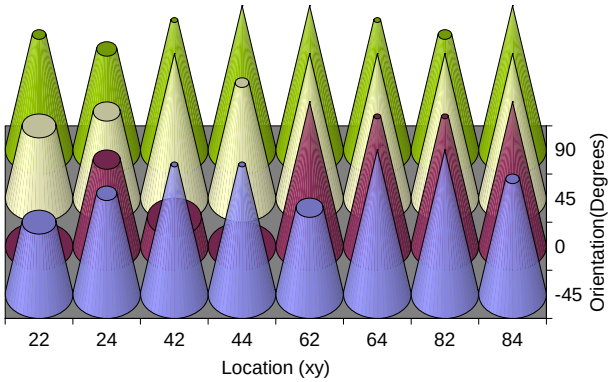


Fig. 3 successful runs for GA per angle and location with the utilization of CF1

In the following figures the results of the combination of the OM with both CF1 and CF2 are presented. For all test cases GA performed 3624 iterations. To retain a better insight, the results are presented in the form of diagrams according to common parameters. In Figs. 4 – 7, the results concern the normalized success which is the ratio of "successful runs" to the amount of total runs performed, with respect to different parameters of the problem. In this case, by the term "successful runs" it is meant the runs that lead to the identification of the correct defect with variation less than 0.005%.

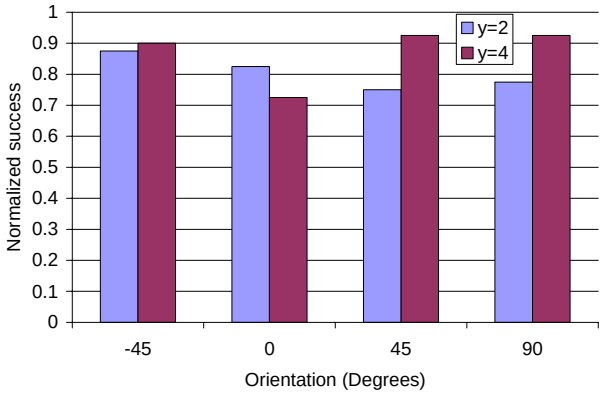


Fig. 4 Normalized success per angle with respect to the *y* location of the center using CF1

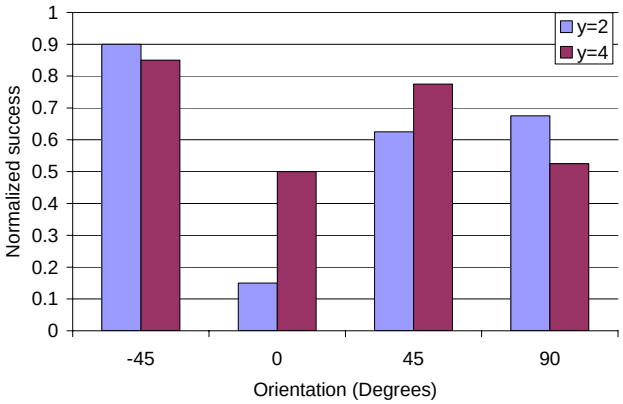


Fig. 5 Normalized success versus orientation in respect to the *y* location of the center using CF2

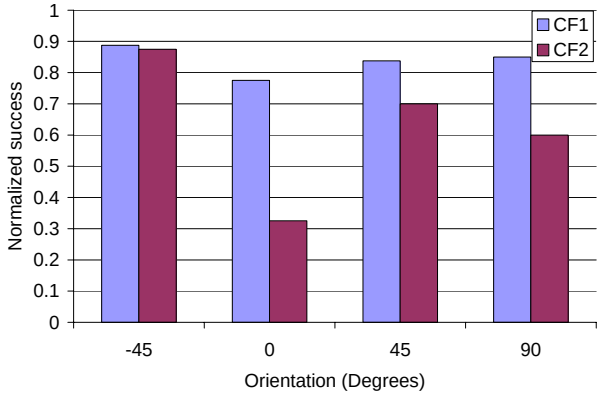


Fig. 6 Normalized success versus orientation using CF1 and CF2

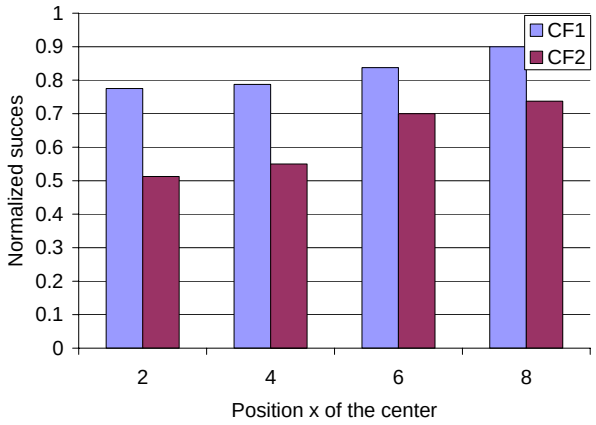


Fig. 7 Number of function evaluations needed for the convergence of *FminSearch* for the use of different amount of data per angle

The importance of the orientation of the depressed ellipse is presented in Fig. 4 and Fig. 5 for all the test cases with the use of CF1 and CF2, respectively. The results are grouped according to y coordinate of the center location. The first column represents the defects with the center lying on the line $y = 2$ while by the second column the defects with a center lying on the line $y = 4$ are represented.

The effect of the different cost function in accordance with the orientation and the location of the center on the x coordinate are presented in Fig. 6 and Fig. 7. In both figures the perpendicular axis is this of the normalized success.

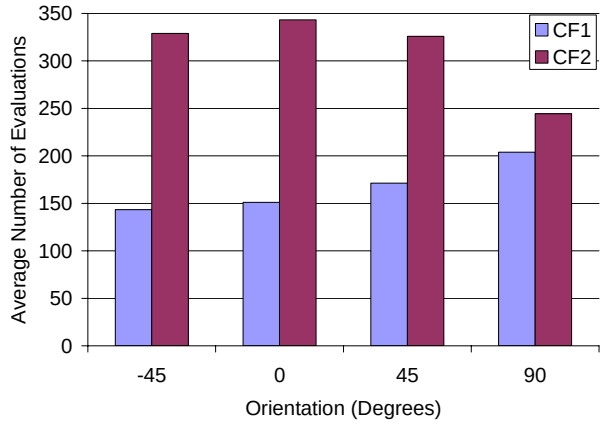
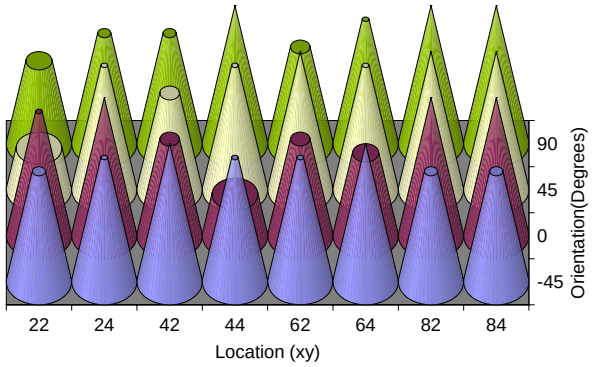


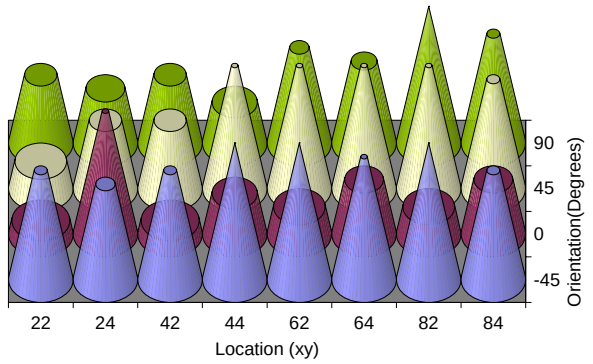
Fig. 8 Normalized success versus x location using CF1 and CF2

In Fig. 8 the average number of the performed evaluations versus the orientation of the flaw is plotted. In each couple of columns, the first one represents the use of CF1 while the second the use of CF2. The average number of evaluations concerns the FminSearch algorithm since as it is mentioned above the GA is set so as to have a population size of 24 and be executed for 150 generations which leads to a total of 3624 iterations for all runs.

In Fig. 9 that follows the "successful runs" per location of the center and orientation are presented analytically in a 3D diagram for both (a) CF1 and (b) CF2. The height of each cone indicates the number of runs for which the combination of the OMs identifies successfully the defect.



(a)



(b)

Fig. 9 successful runs for the optimization algorithm per angle and location with the utilization of (a) CF1 and (b) CF2

V. Discussion

By the use of GA as stand alone, the need of a large number of iterations it was observed, since the value of the cost function even after 10000 iterations was still high. However, if as successful runs are considered those with variations of the design variables less than 10%, the total success is 78%. On the other hand, the total success of the combination, after less than 4000 iterations, with successful runs those with variation of design variables less than 0.005%, rises to 84%. This different definition of successful runs was imposed to us by the different nature of the algorithms. The use of Fminsearch cancels the stochastic nature of the GA and leads either to the finding of the global optimum or to the entrapment of the method in a local optimum, explaining why differences in the order of 10% did not exist.

From this point on, the discussion of the results concerns not only the effect of the geometry of the flaw to be identified but also the role of the amount of data used in the cost function.

The geometrical parameters, i.e. the location and the orientation of the ellipsoidal flaw, seem to play an important role in the performance of the OM. When CF1 is used the influence of the y coordinate of the center seems to be important (Fig. 4). For three out of the four orientations the OM has higher normalized success for $y = 4$. The opposite happens for 0^o where the OM performs better when the flaw center lies on $y = 2$. The flaws with this orientation are parallel

to x axis as well as to the traction that has been applied, fact which makes these test cases particular.

Observing the same results for the use of CF2 in Fig. 5 it is obvious that there is no clear effects of the y coordinate. It is remarkable that only for the orientation of 0^o there is an advantage for the location of $y = 4$, which opposes to the results obtained for the use of CF1.

In Fig. 6 it is shown that the use of CF2 generally leads to lower normalized success ratio than the use of CF1. However, at the orientation of -45^o CF2 achieves to retain almost the same normalized success with CF1 and at the orientation of 45^o their difference is less than 15%.

On the other hand, the x coordinate of the center seems to influence the performance of the OM in a straightforward manner (Fig. 7). Regardless of the cost function that is used the higher value of x gives better results, in the sense that it leads to higher normalized success ratio. This can be explained by the influence of the fully constrained edge (da). Furthermore, the difference of the performance of the OM between the use of CF1 and CF2, for the x locations 2 and 4, is considerably bigger than for the two other locations. This means that as long as the defect lies on the left half the use of less data is less efficient than for the other half.

It has been already mentioned that while for the execution of GA the iterations are the same regardless the cost function and the test case, in Fminsearch the iterations for convergence differ. For the convergence of FminSearch (Fig. 8) in the first three out of four different orientations almost half of iterations are needed for the use of CF1 compared to the use of CF2. Briefly, it can be said that the more data is given the fastest it is for this OM to converge. The only case that differentiates from this rule is the orientation of 0^o. In this case Fminsearch with the use of CF2 can perform successfully only if the outcome of GA is close to the correct one so few iterations are needed.

In Fig. 9, where the results are presented in details, the behavior of the OM for each test case is shown. In this figure some exceptions to the abovementioned general results are shown. Firstly, concerning the results with the use of CF1, the two symmetric orientations (-45^o and 45^o) should have similar results but this does not happen for the center 22. This flaw has a different behavior since it is close to the lowest corner (a) and its orientation (45^o) is the one of the bisector.

Another exception is the few successes of the center 44 and orientation 0^o, where better results were expected because it is close to the center of the plate, though due to the boundary conditions seems to have so small effect to the boundary displacements that the algorithm results in a local minimum.

Concerning the optimization of the test cases with the use of CF2 the exception is this of the center 24 with orientation 0^o where the results are much better than the results of every other test case with the same orientation. The case of the center 22 at the orientation of 45^o has the same performance as it had with the use of CF1. Also for the center 44 and orientation 90^o the successful runs are less than expected.

VI. Conclusion

This paper shows that although the use of GA in a complicated inverse problem could have prohibitive computational cost, if it is combined with a pattern search method results in a lower computational cost increasing actually the success in most of the test cases, regardless the amount of "experimental" data used. The combination of the two optimization methods performs satisfactory with the use of CF1 and identifies successfully the defect in all the test cases and in the majority of runs. Although with the use of CF2 the normalized success is lower, it still leads to total success of 62%.

The parametric study concerning the geometrical parameters of the IESP has shown that it is possible to find the location, the size and the orientation of an ellipsoidal flaw that is significantly depressed in all of the combinations. The coordinates of the center influence the results and there is a tendency to facilitate the identification of the flaws away from the boundaries. Concerning the orientation of the flaw, it is obvious that the most difficult test cases are those with the angle of 0^o for both the use of CF1 and CF2.

References

[1] T. Burczynski, W. Kus, A. Dlugosza and P. Orantek, "Optimization and defect identification using distributed evolutionary algorithms", *Engineering Applications of Artificial Intelligence*, vol. 17, 2004, pp. 337–344.

[2] S. Kubo, Inverse analyses and their applications to nondestructive evaluations, in *Proc. 12th A-PCNDT 2006 – Asia-Pacific Conference on NDT*, Auckland, 2006, pp

[3] S. C. Mellings and M. H. Aliabadi, "Flaw identification using the Boundary Element Method", *International Journal For Numerical Methods In Engineering*, vol. 38, 1995, pp. 399-419.

[4] C. A. Brebbia and J. Dominguez, *Boundary Elements: an introductory course*. New York CA: Computational Mechanics Publications, McGraw-Hill Company, 1992.

[5] H. Koguchi and H. Watabe, "Improving defects search in structure by boundary element and genetic algorithm scan method", *Engineering Analysis with Boundary Elements*, vol. 19, 1997, pp. 105-116.

[6] Y. He, D. Guo and F. Chu, "Using genetic algorithms and finite element methods to detect shaft crack for rotor-bearing system", *Mathematics and Computers in Simulation*, vol. 57, 2001, pp. 95–108.

[7] M. Shim and M. Suh, "Crack identification using evolutionary algorithms in parallel computing environment", *Journal of Sound and Vibration,* vol. 262, 2003, pp. 141–160.

[8] A. P. Plumb, R. C. Rowe, P. York and C. Doherty, "Effect of varying optimization parameters on optimization by guided evolutionary simulated annealing (GESA) using a tablet film coat as an example formulation", *European Journal of Pharmaceutical Sciences*, vol. 18, 2003, pp. 259–266.

[9] M. W. Suh and M. B. Shim, "Crack identification using hybrid Neuro-genetic technique", *Journal of Sound and Vibration*, vol. 238(4), 2000, pp. 617-635.

[10] M. Engelhardt, M. Schanz, G. E. Stavroulakis and H. Antes, "Defect identification in 3-D elastostatics using a genetic algorithm", *Optim Eng*, vol. 7, 2006 pp. 63–79.

[11] Matlab Optimization toolbox.

[12] A. D. Belegundu and T. R. Chandrupatla, *Optimization concepts and applications in engineering.* New Jersey, CA: Prentice Hall, 1999.

[13] A.A. Papageorgiou, D.T. Venetsanos and C.G. Provatidis, "Investigating the Influence of Typical Genetic Algorithm Parameters on the Optimization of Benchmark Mathematical Functions", in: D.Tsahalis (ed.), Proc. *2nd International Conference "From Scientific Computing to Computational Engineering"*, Athens, 2006.

[14] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the Nelder-Mead simplex method in low dimensions," *SIAM Journal of Optimization*, vol. 9(1), 1998, pp.112-147.

[15] G. E. Stavroulakis and H. Antes, "Nondestructive elastostatic identification of unilateral cracks through BEM and neural networks", *Computational Mechanics*, vol. 20, 1997, pp. 439–451.