การประชุมวิชาการเครือข่ายวิศวกรรมเครื่องกลแห่งประเทศไทยครั้งที่ 18
18-20 ตุลาคม 2547 จังหวัด-ขอนแก่น

Structural Compliance Minimisation Using Evolutionary Algorithms with Surface Spline Interpolation

Sujin Bureerat & Jumlong Limtragool
Department of Mechanical Engineering, Khon Kaen University, Thailand, 40002, sujbur@kku.ac.th, jumlim@kku.ac.th

Abstract

In this paper, a number of well-established population-based optimisation methods i.e. genetic algorithms, stud – genetic algorithms, simulated annealing and population based incremental learning are briefly reviewed and compared in terms of their philosophical basis. The use of the optimisation methods for compliance minimisation of plates is demonstrated. The paper also presents the approximated density distribution technique to prevent checkerboard formation in topology design as well as to reduce the size of topological design variables. From the optimum solutions obtained using the various optimisation methods, the performances of the methods are compared and discussed.

1. Introduction

Topology optimisation is a special kind of structural shape optimisation. This design process is employed when designers need to find a new structural configuration for particular use as shown in Figure 1. In topology optimisation problem, with a given design domain, the task is to find structural layout that gives the optimum of desired objective functions e.g. weight, system compliance, deflection and natural frequency whilst fulfilling design constraints. From numerical viewpoint, by the use of Finite Element Method (FEM) for structural analysis, topological design can be performed by discretising a structure into a number of connected finite elements. Design variables determine the distribution of element density, which means that elements with nearly zero density represent voids on the structure whereas other elements indicate the existence of structural material [1].

One of the most preferable optimisation methods for topology design is Optimality Criteria Method (OCM) [2] as it is arguably the most powerful method for this task. Also, the classical gradient-based methods such as Sequential Linear Programming (SLP) and the Method of Moving Asymptotes (MMA) were implemented successfully [3]. There have been a few publications concerning the applications of population-based methods for topology design and most of them referred to Genetic Algorithms (GAs) e.g. [4-6]. Despite the capability of reaching a global optimum of GAs, the methods seem to be ineffective when used in topology design. This is due to the large number of topological design variables and, consequently, a great many of function evaluations are needed when performing GAs for solving such a design problem. However, it cannot be totally concluded that all the population-based or evolutionary methods are inferior since some other evolutionary methods are rarely applied, and there have been a number of research articles indicating that the more successful evolutionary search is usually obtained from performing a few evolutionary methods for one design problem [7].

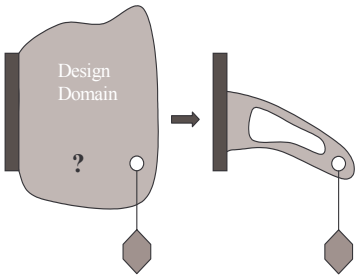

Figure 1 Topology Optimisation

This paper demonstrates the use of evolutionary algorithms for solving structural topology optimisation. The checkerboard suppression scheme here is the application of an Approximated Density Distribution (ADD) technique. Structural topology is represented by interpolation coefficients whereas the objective function is structural compliance. Two sets of design variables are examined. The evolutionary algorithms consist of Genetic Algorithm (GA), Simulated Annealing (SA), Population Based

Incremental Learning (PBIL) and Stud-Genetic Algorithm (Stud-GA). The methods are briefly reviewed and implemented on the topological design problem. The main investigation is aimed at comparing the performance of the evolutionary methods. The optimum results from using the various design strategies are obtained, illustrated and compared in terms of convergence rate and consistency.

2. Topology Optimisation

Figure 2 displays the finite element model of a plate under in-plane loading. The structure is divided into a number of rectangular elements. The topological design variables represent the elements' density (or thickness in cases of 2D structures) and, at the post-process, will define a structural configuration. A typical form of topology design problem can be posed as:

Find $\boldsymbol{\rho}$ densities of the elements such that

Min: $f(\boldsymbol{\rho})$ (1)

Subject to

$$\mathbf{g}(\boldsymbol{\rho}) \le \mathbf{0}$$

$$\mathbf{0} < \boldsymbol{\rho}_{\min} \le \boldsymbol{\rho} \le \mathbf{1}$$

where $\boldsymbol{\rho}$ is the vector of topological design variables that are usually elements' density

f is the objective function

and $\mathbf{g}$ is the vector of inequality constraints.

Classical design objective functions are structural compliance, natural frequencies and buckling factors relying on predetermined design concepts. Structural weight or mass is normally set as a design constraint. Other constraints are unlikely to be included in the problem as it would lead to some more difficulty in solving the problem since this problem is expected to have a great number of design variables.

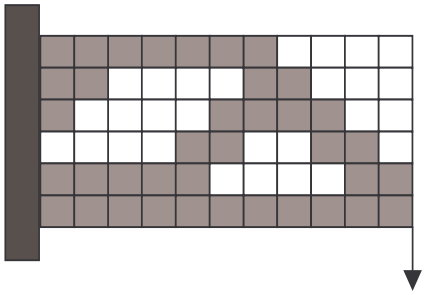

Figure 2 Discretised structure

3. Approximated Density Distribution (ADD) Technique

ADD is a simple numerical technique exploiting the interpolation techniques for approximation of elements' density from the known density at some particular points. From a rectangular design domain being meshed into n elements as shown in Figure 3, let $\mathbf{r}_i^0$ be the position vectors of m sampling points (plus sign) and $\mathbf{r}_k^v$ be the position vector of the centre points of the n elements ('o' sign). With the idea of interpolation with radial-basis functions, the densities at the centre points of the elements, $\boldsymbol{\rho}^v$, can be approximated from the given densities at the sampling points, $\boldsymbol{\rho}^0$, by the relation:

$$\boldsymbol{\rho}^v = \mathbf{C}\mathbf{A}^{-1}\boldsymbol{\rho}^0 = \mathbf{T}\boldsymbol{\rho}^0 \tag{2}$$

where $\mathbf{C} = [c_{ij}]_{n\times m} = [f(d(\mathbf{r}_k^v, \mathbf{r}_j^0))] = [f(d_{kj})]$

$\mathbf{A} = [a_{ij}]_{m\times m} = [f(d(\mathbf{r}_i^0, \mathbf{r}_j^0))] = [f(d_{ij})]$

$f(d_{ij}) = 1 + d_{ij} + d_{ij}^2 + d_{ij}^3$

and $d(\mathbf{r}_i, \mathbf{r}_j) = \sqrt{(\mathbf{r}_i - \mathbf{r}_j)^T(\mathbf{r}_i - \mathbf{r}_j)}$.

For more details, see [8]. By using the relation (2), the topological design problem (1) becomes:

Min: $f(\boldsymbol{\rho}^0)$ (3)

Subject to

$$\mathbf{g}(\boldsymbol{\rho}^0) \le \mathbf{0}$$

$$\mathbf{0} < \boldsymbol{\rho}_{\min} \le \mathbf{T}\boldsymbol{\rho}^0 \le \mathbf{1}.$$

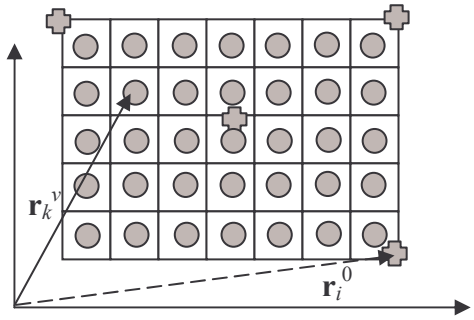

Figure 3 Sampling points and elements' centre points

4. Evolutionary Algorithms

Evolutionary Algorithms (EAs) sometimes referred to as population-based methods are the optimisation methods that search for optima based upon evolutionary and random mechanisms. The methods start the search with a group of initial solutions called population and the population are then evolved, in some manner, generation by generation until reaching the optimum. The evolutionary methods presented here are as follows:

4.1 Genetic Algorithms

Genetic algorithms are probably the best known of the evolutionary algorithms. This approach can be best thought of as mimicking Darwinian natural selection in that a population of solutions (genes) is generated and then the next generation is produced by mating pairs of these genes. The genes have the opportunity of being selected based on their merit. The selected genes are reproduced by means of crossover and mutation yielding the new population or next generation. The next generation is iteratively evolved until an optimum is achieved [9].

4.2 Stud-Genetic Algorithm

A slight modification of classical GA is called Stud-GA which claims to improve significantly the performance of the traditional GA whilst maintaining its simplicity and binary string representation [10]. Rather than maintaining a large population of different solutions, the best gene (stud) from an initial population is chosen and then the new population is generated by mutating the stud until the offspring have the same size as the initial population. After mutation, bit positions of each offspring are allowed to be shuffled by the given probability. The process is repeated until convergence is reached

4.3 Population-Based Incremental Learning

Population Based Incremental Learning (PBIL) [11] has a different feature from GA in that the population is represented by the probability vector of being '1' of each bit position of binary strings. Figure 4 shows probability vectors used in PBIL where row vectors of the population matrix represent genes. It can be concluded that one probability vector can form a variety of populations. Initially, the search procedure starts with the initial probability vector whose elements are full of '0.5'. The probability vector is then updated based upon the given learning rate and the binary string of the current best solution and allowed to be mutated with a predefined probability. The vector is updated iteratively until convergence is achieved.

population 1	population 2	population 3
0 0 1 1,	0 1 1 0,	0 1 0 1
1 1 0 0,	1 1 0 1,	1 0 0 1
0 0 1 1,	1 0 1 0,	0 0 0 1
1 1 0 0,	0 0 0 1,	0 1 0 0
Probability Vectors		
[0.5, 0.5, 0.5, 0.5]	[0.5, 0.5, 0.5, 0.5]	[0.25, 0.5, 0, 0.75]

Figure 4 Probability vector

4.4 Simulated Annealing

Simulated annealing [12] & [13] sometimes is classified to be evolutionary method as GA and the others. The method is based upon mimicking the random behaviour of molecules during the annealing process, which involves slow cooling from a high temperature. As the temperature cools, the atoms line themselves up and form a crystal, which is the state of minimum energy in the system. The search procedure of SA is to start with a single initial solution and it is then adjusted in some manner to produce a few candidates. The best candidate is selected to be a new parent if its objective is better than the parent or it is accepted by the Boltzmann probability. The process is repeated until the optimum is reached.

5. Numerical Implementations

The design case study is the topology optimisation of a cantilever plate under in-plane loading as depicted in Figure 5. The plate is made up of material with 200×10^9 N/m^2 Young modulus and 0.3 Poisson's ratio while **F** = 100 N, L = 3 m and H = 1 m. The plate is discretised into 30×10 elements whereas there are 15×7 sampling points that are equispaced along x and y directions. This means that, by using ADD, the size of design variables is reduced from 300 to 105. Two different sets of design variables are taken into consideration. The first set called SET1 is that the thicknesses at the sampling points are either one or zero. This implies that the design variables are represented by a 105-bit binary string. The second set of design variables, SET2, is a series of binary string as traditionally used in a usual GA search. Note that, with the application of evolutionary algorithms using binary string representing design variables, the bound constraints can be excluded from the optimisation problem as they can be dealt with at the decoding process. Therefore the problem can be written as:

Find $\boldsymbol{\rho}^0$ such that

$$\text{Min:} \quad f(\boldsymbol{\rho}^0) = \mathbf{U}^T\mathbf{K}\mathbf{U} \tag{4}$$

Subject to $g(\boldsymbol{\rho}^0) = m(\boldsymbol{\rho}^0) - \eta m(\mathbf{1}) \leq 0$

where $\boldsymbol{\rho}^0$ is the vector of design variables

U is structural displacement

K is structural stiffness matrix

η is mass reduction ratio

m is structural mass

and $m(\mathbf{1})$ is the initial structural mass.

Since the evolutionary methods can not cope with constrained optimisation directly, the problem (4) has to be

modified by using a penalty function technique leading to an unconstrained optimisation problem having the new objective as

$$F(\rho^0) = \frac{f}{f+1} + 2\mu(\boldsymbol{\rho}^0) \tag{5}$$

where $\mu = \begin{cases} 0 \text{ for } g(\boldsymbol{\rho}^0) \le 0 \\ g(\boldsymbol{\rho}^0)/a \text{ for } 0 < g(\boldsymbol{\rho}^0) < a \\ 1 \text{ for } g(\boldsymbol{\rho}^0) \ge a \end{cases}$

and a is a small number to be specified. This strategy is the modification of the work presented in [14].

In order to benchmark the performance of the algorithms, the methods start with the same initial solution and have the same number of generations i.e. 200 generations with the population size being 100. This implies that there are 20000 function evaluations on each run. To serve the feature of SA which requires a few candidates on each loop, the number of loops is set to be 500 with 40 candidates being created. Thus, there are equal numbers of function evaluations for every method. Each method is operated five attempts to measure its consistency. Also, note that the number of iterations and population size are assigned merely for measuring the algorithms performance. As a result, it cannot be assured that the obtained results will reach or close to the optimum.

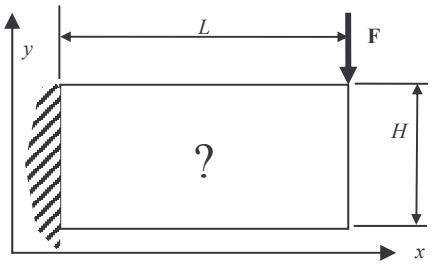

Figure 5 Cantilever plate

6. Results

The optimum here means the best solution obtained from last generation of an evolutionary method. The optimum results from SA and PBIL using the SET1 design variables are illustrated in Figure 6 while the results from GA and Stud-GA are shown in Figure 7. Table 1 displays the optimum solutions after performing the algorithms five times. The results show that SA has the best convergence rate while the most consistent method is PBIL. Most of the topologies are said to be checkerboard-free.

Figure 6 Optimum results of SA & PBIL with SET1 variables

Figure 7 Optimum results of GA & Stud-GA with SET1 variables

No.	SA	PBIL	GA	StudGA
1	0.7367	0.8190	0.7732	0.9074
2	0.7672	0.8351	0.7785	0.8822
3	0.7744	0.8060	0.8061	0.8926
4	0.7556	0.7681	0.8545	0.8474
5	0.8315	0.8016	0.8039	0.8516
AV	0.7731	0.8060	0.8032	0.8762
STD	0.0356	0.0249	0.0322	0.0261

Table 1 Optimum results using SET1 variables

The optimum results of SA and PBIL for SET2 design variables are displayed in Figure 8 and the results from using GA and Stud-GA are in Figure 9. Table 2 shows the objective function values obtained from operating the methods five times. It can be said that PBIL gives the best convergence rate whilst SA and GA are the most consistent for the design variables of SET2. Most of the solutions are inferior to that obtained by using the SET1 design variables and they are said to be checkerboard-free.

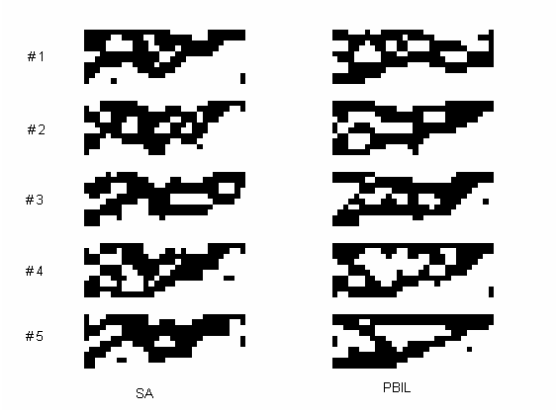

Figure 8 Optimum results of SA & PBIL with SET2 variables

Figure 9 Optimum results of GA & Stud-GA with SET2 variables

No.	SA	PBIL	GA	StudGA
1	0.8880	0.8391	0.9392	0.9033
2	0.9197	0.8278	0.9572	0.9550
3	0.9292	0.8179	0.9500	0.9451
4	0.8955	0.7723	0.9083	0.8284
5	0.9269	0.7462	0.9341	0.9372
AV	0.9118	0.8007	0.9377	0.9138
STD	0.0189	0.0396	0.0188	0.0515

Table 2 Optimum results using SET2 variables

Figure 10 illustrates the optimum solution obtained from solving the problem (3) using the gradient-based method, sequential quadratic programming with 105 design variables. In figure 11, the optimum topology of the plate obtained from solving the design problem (1) directly using the optimality criteria method (with 300 design variables being used) is shown. When comparing these two topologies to those obtained from implementing the evolutionary algorithms, it can be said that the use of evolutionary algorithms for this task is still inferior to the gradient-based method. In order to obtain the better solution by using EA, a great many of function evaluations must be executed. The evolutionary algorithms are, nevertheless, still useful when the objective function is not continuous or when it is difficult or even impossible to compute the function derivatives. Apart from that, the gradient-based methods with poor derivative approximation often lead to unsuccessful search but this problem never occurs when using EAs. Therefore, the population-based optimisation methods are always proposed as an alternative choice since the solution will usually be achieved (at slower rate of convergence).

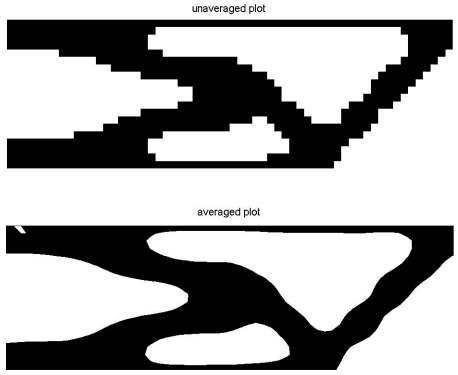

Figure 10 Optimum topology by SQP

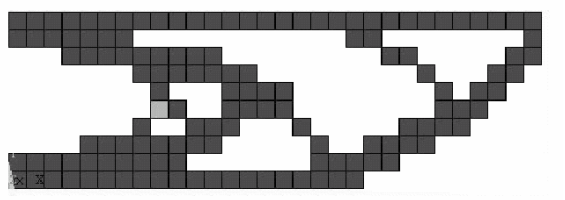

Figure 11 Optimum topology by OCM

7. Conclusions

The proposed evolutionary algorithms are applicable to topology optimisation. The ADD technique is a powerful tool for preventing checkerboard pattern in topological design. Using SET1 design variables gives the better design solutions when compared to the use of SET2 design variables. SA is the best among the four methods in terms of convergence rate when SET1 is applied and PBIL is superior to the others with SET2 being used.

8. References

[1] T. Kunakote and S. Bureerat, “Structural Topology Optimisation Using Evolutionary Algorithms”, *17th ME-NETT*, Thailand, 2003. (in Thai).

[2] O. Simund, “A 99 Line Topology Optimization Code Written in MATLAB”, *Struct. Multidisc. Optim*, 21, 120 – 127, 2001.

[3] M.P. Bensøe and O. Sigmund, *Topology Optimization Theory, Method and Applications*, Springer-Verlag, Berlin Heidelberg, 2003.

[4] C. Kane, F. Jouve & M. Schoenauer, Structural Topology Optimization in Linear and Nonlinear Elasticity Using Genetic Algorithms, *21st ASME Design Automatic Conf.*, Boston, 1995.

[5] C. Kane and M. Schoenauer, Topological Optimum Design Using Genetic Algorithms, *Control & Cybernetics*, 25, **5**, 1059 – 1088, 1996.

[6] M. Jakiela, C. Chapman, J. Duda, A. Adewuya, and K. Saitou, Continuum structural topology design with genetic algorithms, Comp. Methods in App. Mech. and Eng., 86:2, 339–356, 2000.

[7] S. Bureerat and J. E. Cooper, Evolutionary Methods for the Optimisation of Engineering Systems, *Opt. in Control: Methods and Applications, IEE*, London, 1/1-1/10, 1998.

[8] Bureerat S. Structural Compliance Minimisation Using Approximated Distribution of Material Density. *KKU-Engineering Conference*, 2004. Khon Kaen University.

[9] G. Lindfield and J. Penny, *Numerical Methods Using MATLAB*, Ellis Horwood, 1995, p. 284 - 294.

[10] W. Khatib and P. Fleming, The Stud GA: A Mini-Revolution, *5th Int. Conf. on Parallel Problem Solving From Nature*, 1998.

[11] S. Baluja, Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning, *CMU_CS_163*, 1994.

[12] S. Kirkpatrik, C. D. Gelatt Jr. and M. P. Vecchi, Optimization by Simulated Annealing, *Science*, 220, **4598**, 671-680, 1983.

[13] W.A. Benage and A. K. Dhingra, Single and Multiobjective Structural Optimization in Discrete-continuous Variables Using Simiulated Annealing, Int. J. of Num. Methods in Eng., 38, 2753-2773, 1994.

[14] Cheng F.Y. and Li D., “Fuzzy Set Theory with Genetic Algorithms in Constrained Structural Optimization”, *ASCE Proceeding of US-Japan Joint Seminar on Structural Optimization*, New York, Advances in Structural optimization, pp. 55-66, 1997.