Robust Design Using Particle Swarm and Genetic Algorithm Optimization

Gerhard Venter[1] (gventer@vrand.com), Raphael T. Haftka[2] (haftka@ufl.edu) and Jaroslaw Sobieszczanski-Sobieski[3] (j.sobieski@larc.nasa.gov)

[1] Vanderplaats R&D Inc, 1767 S 8th Street, Suite 200, Colorado Springs, CO 80906, USA

[2] University of Florida, Dept. of Mechanical and Aerospace Engineering, Gainesville, FL 32611-6250, USA

[3] NASA Langley Research Center, MS 240, Hampton, VA 23681-2199, USA

1. Abstract

The paper makes use of design points evaluated during a conventional genetic algorithm or particle swarm optimization to perform robust optimization that accounts for uncertainty in the design variables. The design points obtained from the two algorithms are used to construct response surface approximations that provide an estimate of the sensitivity to variation in the design variables. As an example, the robust optimization of a composite laminate is considered. The proposed approach allows the inclusion of the effects of uncertainty at little or no additional computational cost. The paper shows that the response surface approximations can be easily updated without keeping track of the entire history of the population. In addition a simple fading procedure is proposed that can effectively emphasize more recent information when constructing the approximation. The paper includes results from both a genetic algorithm and a particle swarm optimization algorithm and a comparison between the two algorithms.

2. Keywords: *Particle Swarm Optimization; Genetic Algorithm; Design for Uncertainty*

3. Introduction

With increased computer power becoming more readily available, robust design problems are gaining more and more interest, both in research and industrial environments. The goal is to accurately model the uncertainty that is inherent to many design problems, for example material properties, manufacturing tolerances, boundary conditions, etc. The challenge is to execute the reliability based design process efficiently without sacrificing accuracy. One approach of reducing the computational cost associated with design for uncertainty is the use of response surface approximations [1].

Similarly, non-gradient based, probabilistic search algorithms have attracted much attention. These algorithms typically require many function evaluations to find an optimum solution, but are easy to program, can efficiently make use of a large number of processors, do not require continuity in the problem definition and generally are better suited for finding global, or near global, solutions. The high computational cost associated with both non-gradient based search algorithms and design for uncertainty makes it impractical to use non-gradient based search algorithms to solve design for uncertainty problems, without specifically addressing the resulting high computational cost. The present paper will explore the assumption that the large number of design points evaluated during a non-gradient based search, may provide free information for describing the uncertainty in design variables as required for solving the corresponding robust optimization problem.

The idea is to perform a conventional non-gradient based search, as one would do for a deterministic problem, using either a genetic algorithm (GA) [2] or particle swarm optimization (PSO) [3, 4] algorithm. The data points evaluated during these searches are then re-used to construct response surface approximations for estimating the sensitivity of the response functions to uncertainty in the design variables. Although a genetic algorithm and particle swarm optimization is considered here, any algorithm that generates a large number of design points would be a candidate for the proposed approach. This method may be appropriate when the goal of the robust optimization is to steer clear of designs that exhibit high sensitivity to uncertainty. In such cases moderate errors in the sensitivity may be tolerated. In addition, the approach can only account for uncertainty associated with the design variables and not for uncertainty associated with other problem parameters. The proposed approach will be demonstrated using a composite laminate design problem with uncertainty in the ply angles.

4. Approach

The proposed approach is based on the fact that the standard deviation of a general response function $Y = Y(\boldsymbol{x})$ with parameters $\boldsymbol{x}$, can be approximated from the standard deviation of the parameters and the first order gradients of the response function with respect to the parameters. This approximation is obtained by using a first order Taylor series to approximate the response function at a point, say $\boldsymbol{x} = \boldsymbol{x}_0$. Calculating the standard deviation of this first order Taylor series approximation, while assuming that all the parameters are uncorrelated, the standard deviation of the response function at point $\boldsymbol{x}_0$ is approximated by Eq. (1) [5]:

$$\sigma^2\left(Y\right) \approx \sum_{i=1}^{n}\left(\sigma^2\left(x_i\right)\left(\left.\frac{\partial Y}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{x}_0}\right)^2\right) \tag{1}$$

In Eq. (1), $\sigma(Y)$ is the standard deviation of the response function and $\sigma(x_i)$ is the standard deviation of parameter x_i. The implementation of Eq. (1) presents a challenge when considering a non-gradient search algorithm, since the required gradient information is not readily available. However, an estimate of the missing gradient information can be obtained from an approximation of the response function fitted to data points evaluated during the non-gradient search. In the present work a genetic algorithm and particle swarm optimization algorithm is used to perform the non-gradient search, and a least squares fit is used to construct a full quadratic response surface approximation of the response function. The two algorithms and the response surface approximation procedure are discussed in more detail in the following sub-sections.

4.1. Genetic algorithm

A fairly simple genetic algorithm is used in the present work (e.g., Ref. [6]). The genetic algorithm makes use of the standard genetic operators of selection, crossover and mutation as well as an elitist strategy to ensure monotonic convergence. Single point crossover and tournament selection is used.

The mutation operator selects one gene at random and changes its allele to the next lower or next higher value at random. If the probability of mutation p_m is larger than one, the operator is applied $[p_m]$ times with certainty, where the brackets indicate the integer part of a number, and then once with a probability of $p_m - [p_m]$.

The fitness values are obtained using a penalty function approach to account for the design constraints.

4.2. Particle swarm optimization algorithm

The particle swarm optimization algorithm used here is very similar to that used in Ref. [7] to solve an integer cantilevered beam problem and in Ref. [8] to solve a continuous/discrete multidisciplinary wing design problem. Particle swarm optimization is based on a simplified social model that is closely tied to swarming theory. The swarm is updated based on the social behavior that a population of individuals, the swarm in the case of particle swarm optimization, adapts to its environment by returning to promising regions that were previously discovered. The particle swarm optimization algorithm makes use of a velocity vector to update the current position of each particle in the swarm. The process is stochastic in nature and makes use of the memory of each particle as well as the knowledge gained by the swarm as a whole. The position of each particle is updated using

$$\boldsymbol{x}_{k+1}^{i} = \boldsymbol{x}_{k}^{i} + \boldsymbol{v}_{k+1}^{i}\Delta t \tag{2}$$

where: $\boldsymbol{x}_{k+1}^{i}$ is the position of particle i at iteration $k+1$; $\boldsymbol{v}_{k+1}^{i}$ is the corresponding velocity vector; and Δt is the time step value. A unit time step ($\Delta t = 1$) is used throughout the present work. The scheme for updating the velocity vector of each particle depends on the particular particle swarm optimization algorithm. The scheme used here is slightly different from that used in Refs. [7, 8] and similar to that used in Refs. [9, 10], as follows

$$\boldsymbol{v}_{k+1}^{i} = w\boldsymbol{v}_{k}^{i} + c_1 r_1 \frac{\left(\boldsymbol{p}^{i} - \boldsymbol{x}_{k}^{i}\right)}{\Delta t} + c_2 r_2 \frac{\left(\boldsymbol{p}^{g} - \boldsymbol{x}_{k}^{i}\right)}{\Delta t} \tag{3}$$

where r_1 and r_2 are random numbers between 0 and 1 and $\boldsymbol{p}^{i}$ is the best position found by particle i so far. The difference between Eq. (3) and that used in Refs. [7, 8], is that $\boldsymbol{p}^{g}$, the best point ever found by the swarm, is used instead of $\boldsymbol{p}_{k}^{g}$, the best point found by the swarm during iteration k. The reason for this change is that using $\boldsymbol{p}^{g}$ seemed to work slightly better for the example problem considered here.

The swarm includes a craziness operator, which is similar to the mutation operator of a genetic algorithm and helps to avoid premature convergence of the swarm. The craziness operator is applied at each design iteration where the coefficient of variation ($COV = StdDev/Mean$) of the objective function values falls below a specified threshold value, in the current work 0.1. The craziness operator selects points located far from the best point to go crazy. All particles that are located more than 1 standard deviation from the best point is selected to have the craziness operator applied. The craziness operator changes the position to a random location and creates a new velocity vector that points back to the best position found so far for a particle, as follows:

$$\boldsymbol{v}_{k+1}^{i} = c_1 r_1 \frac{\left(\boldsymbol{p}^{i} - \boldsymbol{x}_{k}^{i}\right)}{\Delta t} \tag{4}$$

There are three problem dependent parameters, the inertia of the particle w and two trust parameters c_1 and c_2. The inertia parameter controls the exploration properties of the algorithm with larger values facilitating a more global behavior and smaller values facilitation a more local behavior. The trust parameters indicate how much trust a particle has in itself (c_1) and how much it trusts the swarm (c_2). Throughout the present work, slightly more trust is placed in the swarm with $c_1 = 1.5$ and $c_2 = 2.5$. The inertia parameter w is dynamically reduced from an initial value of 1.4 throughout the optimization run. The idea is to start with a more global search and end with a more local search. The dynamic reduction scheme for w is based on the distribution of the swarm at each design iteration and is described in more detail in Refs. [7, 8].

The initial swarm is created such that the particles are randomly, distributed throughout the design space, using a uniform distribution, each with a random initial velocity vector. The random initial position and velocity vectors are obtained from

$$\boldsymbol{x}_{0}^{i} = \boldsymbol{x}_{min} + r_3\left(\boldsymbol{x}_{max} - \boldsymbol{x}_{min}\right) \qquad \boldsymbol{v}_{0}^{i} = \frac{\boldsymbol{x}_{min} + r_4\left(\boldsymbol{x}_{max} - \boldsymbol{x}_{min}\right)}{\Delta t} \tag{5}$$

where r_3 and r_4 are random numbers between 0 and 1, $\boldsymbol{x}_{min}$ is the vector of lower bound values and $\boldsymbol{x}_{max}$ is the vector of upper bound values for the design variables.

The basic particle swarm optimization algorithm is an unconstrained algorithm, similar to a genetic algorithm. Typically, constraints are included using a penalty function formulation. However, the particle swarm optimization algorithm used here includes an additional enhancement for dealing with violated design points. The velocity vector of all points that violate constraints is modified as follows:

$$\boldsymbol{v}_{k+1}^{i} = c_1 r_1 \frac{\left(\boldsymbol{p}^{i} - \boldsymbol{x}_{k}^{i}\right)}{\Delta t} + c_2 r_2 \frac{\left(\boldsymbol{p}_{k}^{g} - \boldsymbol{x}_{k}^{i}\right)}{\Delta t} \tag{6}$$

This new velocity vector will point back to the feasible design space in most cases and help the violated design points to overcome their constraint violation more rapidly. The idea is similar to that of the Modified Method of Feasible Directions [11].

4.3. Response surface approximation

The example problem considered in the present paper designs the in-plane stiffness coefficients A_{11}, A_{22} and A_{66} of a symmetric and balanced composite laminate. The uncertainty is related to the ply angles of each layer. The uncertainty in the in-plane stiffness coefficients due to the uncertainty in the play angels is approximated, using Eq. (1). Quadratic response surfaces are constructed for each of the three stiffness parameters as a function of the ply orientations to estimate the required first order gradient information of Eq. (1). In matrix form, the approximation may be written as follows

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{b} \tag{7}$$

where $\boldsymbol{y}$ is a vector of response data for the data points considered, $\boldsymbol{X}$ is a matrix of monomials evaluated at the data points and $\boldsymbol{b}$ is a vector of unknown polynomial coefficients. For a quadratic approximation with two design variables, Eq. (7) may be expanded at point i as follows:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1^2 + b_4 x_2^2 + b_5 x_1 x_2 \tag{8}$$

The unknown coefficients of the approximation, $\boldsymbol{b}$, are obtained from a least squares fit that results in

$$\boldsymbol{X}^T \boldsymbol{X}\boldsymbol{b} = \boldsymbol{X}^T \boldsymbol{y} \tag{9}$$

In the present work special attention is paid to updating the approximations with each new set of population data. The goal is to minimize the storage requirements while placing more emphasis on the data of more recent populations. To minimize the memory requirements for storing previous data, the data is partitioned into two sets ($\boldsymbol{X}_1$, $\boldsymbol{y}_1$) and ($\boldsymbol{X}_2$, $\boldsymbol{y}_2$). With this partitioning, Eq. (9) may be re-written as

$$\left(\boldsymbol{X}_1^T \boldsymbol{X}_1 + \boldsymbol{X}_2^T \boldsymbol{X}_2\right) \boldsymbol{b} = \boldsymbol{X}_1^T \boldsymbol{y}_1 + \boldsymbol{X}_2^T \boldsymbol{y}_2 \tag{10}$$

which implies that there is no need to keep a record of previous generations for the purpose of fitting the response surface approximations. It is enough to simply accumulate the $\boldsymbol{X}^T\boldsymbol{X}$ matrix and the $\boldsymbol{X}^T\boldsymbol{y}$ vector from all previous generations.

In the application of response surface approximations, it is quite common to place more emphasis on specific points by changing the relative weight between the design points, with more important points having higher weights. In the present work, it is desirable to have data points from more recent iterations contribute more to the approximation. A simple fading scheme is proposed that works well with the fact that we do not store all previous design points. The approach used here is to simply fade out previous data at a constant geometric rate, using a fading constant that is less than 1. For example for a fading constant of 0.8, points from the current population are weighted by 1, points from the previous generation by 0.8, points from two generations ago by 0.64, and so on. The combination of the history saving and the weighting schemes result in the following set of equations for obtaining the approximations of the responses when adding new population data

$$\boldsymbol{S}_{Xi}\boldsymbol{b} = \boldsymbol{s}_{yi} \tag{11}$$
$$\boldsymbol{S}_{Xi} = f\boldsymbol{S}_{X(i-1)} + \left(\boldsymbol{X}^T\boldsymbol{X}\right)_i \qquad \boldsymbol{s}_{yi} = f\boldsymbol{s}_{y(i-1)} + \left(\boldsymbol{X}^T\boldsymbol{y}\right)_i$$

5. Example Problem

The design problem considered is to maximize the in-plane stiffness coefficient A_{11} subject to constraints on the A_{22} and A_{66} stiffness coefficients for a symmetric and balanced composite laminate. The laminate is assumed to be composed of n stacks of $\pm\theta$ plies (e.g., $(\pm\theta_1$ / $\pm\theta_2$ / ... / $\pm\theta_n)_s$). Each stack can have a ply angle that is any integer multiple of 15 ˚, between 0 ˚ and 90 ˚. For the current example, $n = 8$ so that the total number of plies is 32. The material properties used are $E_1 = 128$ GPa, $E_2 = 13$ GPa, $G_{12} = 6.4$ GPa and $v_{12} = 0.3$. The ply thickness is 1 mm and the standard deviation of the ply angles is assumed to be 1 ˚. The deterministic design problem is then defined as

$$\begin{aligned} \textbf{Maximize}: &\quad A_{11} \\ \textbf{Such that}: &\quad g_1 = 1 - A_{22r}/A_{22} \geq 0 \\ &\quad g_2 = 1 - A_{66r}/A_{66} \geq 0 \end{aligned} \tag{12}$$

where A_{22r} and A_{66r} represents lower bound values for A_{22} and A_{66} respectively.

For the reliability based design problem, it is assumed that the n ply angles θ_k are normally distributed and uncorrelated, with the mean equal to the design variable value and the same standard deviation of 1 ˚. From Eq. (1), the standard deviation of the in-plane stiffness properties A_{ij} is then approximated as:

$$\sigma^2\left(A_{ij}\right) \approx \sigma^2\left(\theta\right) \sum_{k=1}^{n} \left(\frac{\partial A_{ij}}{\partial \theta_k}\right)^2 \tag{13}$$

The robust design problem replaces the stiffness properties with the stiffness properties reduced by a number m_{ij} of standard deviations as follows:

$$A_{ij}^u = A_{ij} - m_{ij}\sigma\left(A_{ij}\right) \tag{14}$$

In practice different m_{ij} values can be used for each of the A_{ij} response values. However, in the current implementation the same m_{ij} value (referred to as m) equal to two was used for each of the response quantities. To accommodate the genetic and particle swarm optimization algorithms, the problem is defined in terms of a penalty function formulation. Using Eqs. (12) and (14), the penalized objective function may be defined as

$$F = A_{11}^u[1 + 0.05 \min(g_1, g_2)] \quad \text{if } g_1 > 0 \text{ and } g_2 > 0$$
$$F = 0.8A_{11}^u + p \min(g_1, g_2)] \qquad \text{otherwise} \tag{15}$$

where p is a penalty parameter, which is selected large enough to ensure that the design with the highest value of F does not violate any constraints. The term 0.05 in the expression for feasible designs is used to reward designs that satisfy the constraints with larger margins. The same penalized objective function with the same penalty parameters were used for both the genetic and the particle swarm optimization algorithms. The constraints g_1 and g_2 are calculated using properties reduced due to the uncertainty, using Eq. (14).

6. Results

Throughout the results section both algorithms were repeated 100 times for each of three cases to obtain the reliability of the algorithms. The three cases considered are: (1) deterministic optimization; (2) robust optimization using exact gradient information; and (3) robust optimization using approximate gradient information. The reliability is defined as the number of times (out of 100) that the optimizer found a design within 2% of the best design found in any of the 100 optimization runs. In most cases this is equivalent to the reliability of finding the best design.

After some numerical experimentation, the genetic algorithm was run with a mutation rate of 2 and a population size of 10 for all cases. The particle swarm optimization algorithm was run with a larger swarm size of 20. Also, the particle swarm optimization algorithm used an initial inertia weight value of $w = 1.4$ and trust parameters $c_1 = 1.5$ and $c_2 = 2.5$. In all cases the algorithms are compared based on the total number of function evaluations to account for the differences in the population and swarm sizes.

As a first step, different values of the constraints in the optimization problem were tried in order to find an example where the introduction of uncertainty has a clear effect on the choice of the design. The clearest choice was obtained for $A_{22r} = 1220$ MN/m, and $A_{66r} = 830$ MN/m. The five top designs for this choice are given in Table 1. The first design has an A_{11} value that is about one percent higher than the other designs, but it has almost no margin on the A_{22} requirement, so that with practically any amount of uncertainty it becomes infeasible. The third, fourth and fifth designs have almost no margin on the A_{66} requirement, so that with practically any amount of uncertainty it too becomes infeasible. For $m = 2$ only the second design will satisfy all the constraints.

Table 1: Best designs obtained for $A_{22r} = 1220$ MN/m and $A_{66r} = 830$ MN/m

Laminate	A_{11} (MN/m)	$\sigma(A_{11})$ (MN/m-rad)	A_{22} (MN/m)	$\sigma(A_{22})$ (MN/m-rad)	A_{66} (MN/m)	$\sigma(A_{66})$ (MN/m-rad)
$(\pm15_2 / \pm30 / \pm45_4 / \pm60)_s$	2025.89	11.07	1221.79	9.78	857.77	3.29
$(\pm30_5 / \pm45_2 / \pm75)_s$	2003.12	12.87	1244.56	7.98	857.77	4.03
$(0_2 / \pm30_3 / \pm45_2 / \pm60_2)_s$	1999.23	10.93	1302.86	9.82	830.57	3.68
$(\pm15 / \pm30_3 / \pm45_3 / \pm75)_s$	1999.23	11.94	1302.86	8.56	830.57	3.68
$(0_4 / \pm45_5 / \pm60)_s$	1999.23	9.24	1302.86	10.42	830.57	1.65

For the deterministic case, both algorithms converged very fast to the best designs. After 500 function evaluations, both algorithms had a reliability of 100%, with the genetic algorithm finding the best design 87 times and the particle swarm optimization algorithm 69 times.

6.1. Exact gradients

Next, the design for uncertainty problem was considered using exact gradient information in Eq. (13). With the uncertainty set at two standard deviations, only the second design in Table 1 remains feasible. After 1500 function evaluations, the genetic algorithm had a reliability of 26%, while the particle swarm optimization algorithm had a reliability of 36%. The detailed results obtained from each algorithm is presented in Table 2. For this case, numerical experimentation found that for the same number of total function evaluations, the particle swarm optimization algorithm performed better when increasing the swarm size from 20 to 30.

Table 2: Design for uncertainty results after 1500 function evaluations, using exact gradient information and $A_{22r} = 1220$ MN/m and $A_{66r} = 830$ MN/m

GA Success	PSO Success	Laminate	A_{11} (MN/m)	A_{11}^u (MN/m)	A_{22} (MN/m)	A_{22}^u (MN/m)	A_{66} (MN/m)	A_{66}^u (MN/M)
26/100	36/100	$(\pm30_5 / \pm45_2 / \pm75)_s$	2003.12	1977.37	1244.56	1228.60	857.77	849.71
0/100	3/100	$(0_2 / \pm30_2 / \pm45_4 / \pm60)_s$	1944.81	1922.70	1248.45	1228.53	884.98	879.28
0/100	3/100	$(\pm30_5 / \pm45 / \pm60_2)_s$	1944.81	1919.82	1248.45	1229.81	884.98	876.28
74/100	58/100	$(0_2 / \pm15 / \pm45_5 / \pm60)_s$	1940.92	1921.02	1306.75	1285.90	857.77	849.71

For this example the reliability criterion of 2% isolates only the best design. Although both algorithms had a low reliability of finding the best design, even after 1500 function evaluations, both algorithms were very successful in finding near optimum designs that are within 3% of the best design. The genetic algorithm converged on only two designs, while the particle swarm optimization algorithm found four designs.For a 3% reliability criterion, both algorithms had a reliability of close to 100% after 1500 function evaluations. These results indicate that for this example problem, the reliability of the algorithms is very sensitive to the reliability criterion used. However, the reliability parameter is still useful in comparing the relative performance of the two algorithms.

6.2. Approximate gradients

The final case obtains the gradient information of Eq. (13) from the response surface approximations created for the A_{ij} values. Again the uncertainty was set to two standard deviations. The initial population size had to be increased since a minimum of 45 data points are required to construct a full quadratic approximation for 8 design variables. Numerical experimentation found that the genetic algorithm performed better when distributing the initial population according to a DOE, rather than using a random distribution, and that a fading constant of 0.9 performed better than one of 0.5. A simple DOE consisting of 49 points was used to distribute the initial genetic algorithm population because the functions are separable. A nominal design with all the ply angles at $\pm45^\circ$ was perturbed so that only one pair of angles (that is $\pm\theta$) at a time was different from $\pm45^\circ$. With each of the 8 angles being able to take six other values. The total number of function evaluations for the genetic algorithm is obtained from the 49 initial points plus 149 design iterations with 10 points each, giving a total of 1539 points.

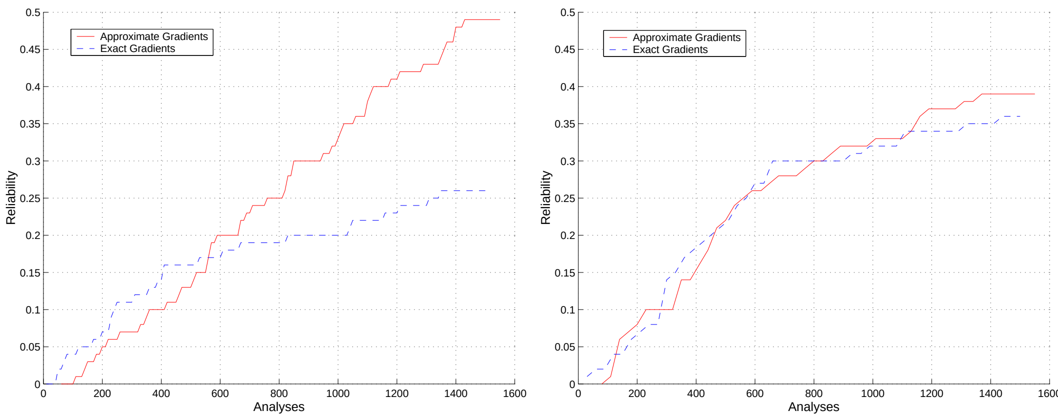
For the particle swarm optimization algorithm a fading constant of 0.9 also performed better than one of 0.5 and no initial DOE was required. For the particle swarm optimization algorithm the initial design was randomly created, using a uniform distribution and 49 design points to be consistent with the genetic algorithm. The total number of function evaluations for the particle swarm optimization algorithm is obtained from the 49 initial points plus 49 design iterations with 30 points each, giving a total of 1519 points.

The detailed results from each algorithm are summarized in Table 3 while Fig. 1 compare the reliability of each algorithm for the cases with exact and approximate gradient information.

Table 3: Design for uncertainty results after 1539 function evaluations for the GA and 1519 function evaluations for the PSO, using approximate gradient information and $A_{22r} = 1220$ MN/m and $A_{66r} = 830$ MN/m

GA Success	PSO Success	Laminate	A_{11} (MN/m)	$A_{22}^{u^*}$ (MN/m)	A_{22} (MN/m)	$A_{22}^{u^*}$ (MN/m)	A_{66} (MN/m)	$A_{66}^{u^*}$ (MN/m)
49/100	39/100	$(\pm30_5 / \pm45_2 / \pm75)_s$	2003.12	1977.37	1244.56	1228.60	857.77	849.71
51/100	42/100	$(0_2 / \pm30_2 / \pm45_4 / \pm60)_s$	1944.81	1922.70	1248.45	1228.53	884.98	879.28
0/100	15/100	$(\pm30_5 / \pm45 / \pm60_2)_s$	1944.81	1919.82	1248.45	1229.81	884.98	876.28
0/100	4/100	$(\pm15 / \pm30_3 / \pm45_2 / \pm60_2)_s$	1940.92	1917.86	1306.75	1287.11	857.77	849.71

* - *Exact A_{ij}^u values*



(a) Genetic Algorithm Reliability

(b) Particle Swarm Optimization Reliability

Figure 1: Algorithm Reliability

Table 3 indicates that the genetic algorithm again converged to only two designs, while the particle swarm optimization converged to four designs. Again, although the overall reliability is low, both algorithms found near optimum designs in all cases. One striking feature of Fig. 1 is that the use of approximate gradient information does not increase the required computational cost for solving the robust design problem. In fact, the use of approximate gradient information increases the reliability of both algorithms. The reliability of the genetic algorithm is increased from 26% to 49% and that of the particle swarm optimization from 36% to 39%. It would appear that the response surface approximations help both algorithms to explore the design space more thoroughly by including additional randomness into the search process.

The procedure described here is applicable only for dealing with uncertainty associated with design variables and not for uncertainty associated with any other problem parameters, such as geometry or material properties. To include the effect of uncertainty in problem parameters that are not considered as design variables, future work may study the inclusion of these parameters in the optimization as additional design variables. These design variables should be restricted to vary within narrow ranges typical for the uncertainty variations. However, even with the current limitation, in many optimization problems the number of design variables is much larger than the number of uncertain geometry or material parameters. Thus, the ability to handle the design variable uncertainty efficiently will be useful. Furthermore, it is increasingly recognized (e.g., [12]) that in design for uncertainty, it may be important to design the magnitude of the uncertainty rather than take it as given. That is, the designer may want to trade the cost of reduced uncertainty achieved via quality control, additional tests, or stricter tolerances, against the cost of living with the uncertainty. When the uncertainty is controlled via uncertainty design variables, the present approach allows additional savings for an expanded set of uncertainty variables.

7. Concluding Remarks

A simple example of the robust design of a composite laminate for specified in-plane stiffness was used to demonstrate the use of population data from a genetic algorithm and particle swarm optimization algorithm for estimating uncertainty in the design variables. For each algorithm the population data were used to create approximations of the objective function and constraints for estimating the sensitivity to uncertainty in the design variables. The paper also addressed the issue of updating the response surface without keeping track of the entire population history.

For the present example both algorithms converged very fast in the deterministic case. However, both algorithms had a low reliability when solving the robust problem. When approximating the gradient information with response surface approximations, both algorithms performed better as compared to the case where exact gradient information was used. It appears that for the current example problem, the robust design problem can be solved at no additional computational cost when using the response surface approximations for estimating the gradient information.

Finally, it appears that the two algorithms performed similarly for solving the current example problem, with the genetic algorithm gaining more from the use of the approximate gradient information as compared to the particle swarm optimization algorithm.

References

[1] X. Qu, S. Venkataraman, and R. T. Haftka. Response Surface Options for Probabilistic Optimization of Stiffened Panels. In *Proceedings, Joint Speciality Conference on Probabilistic Mechanics and Structural Reliability, Notre Dame, IN*, July 24–26 2000.

[2] Z. Michalewicz and D. Dasgupta, editors. *Evolutionary Algorithms in Engineering Applications*. Springer Verlag, 1997.

[3] J. Kennedy and R. C. Eberhart. Particle Swarm Optimization. *Proceedings of the 1995 IEEE International Conference on Nerual Networks, Perth, Australia*, pages 1942–1948, 1995.

[4] R. C. Eberhart and J. Kennedy. A New Optimizer Using Particles Swarm Theory. *Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan*, pages 39–43, 1995.

[5] R. E. Melchers. *Structural Reliability Analysis and Prediction*. John Wiley and Sons, Inc., New York, NY, 2nd edition, 1999.

[6] Z. Gürdal, R. T. Haftka, and P. Hajela. *Design and Optimization of Laminated Composite Materials*. John Wiley and Sons, Inc., New York, NY, 1999.

[7] G. Venter and J. Sobieszczanski-Sobieski. Particle Swarm Optimization. In *Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Denver, CO*, volume AIAA-2002-1235, April 22–25 2002.

[8] G. Venter and J. Sobieszczanski-Sobieski. Multidisciplinary Optimization of a Transport Aircraft Wing using Particle Swarm Optimization. In *Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA*, volume AIAA-2002-5644, September 4–6 2002.

[9] P. C. Fourie and A. A. Groenwold. Particle Swarms in Size and Shape Optimization. *Proceedings of the International Workshop on Multidisciplinary Design Optimization, Pretoria, South Africa*, pages 97–106, August 7-10 2000.

[10] P. C. Fourie and A. A. Groenwold. Particle Swarms in Topology Optimization. *Extended Abstracts of the Fourth World Congress of Structural and Multidisciplinary Optimization, Dalian, China*, pages 52–53, June 4-8 2001.

[11] G. N. Vanderplaats. *Numerical Optimization Techniques for Engineering Design*. Vanderplaats Research and Development, Inc., Colorado Springs, CO, 3rd edition, 1999.

[12] X. Qu, S. Venkataraman, R. T. Haftka, and T. F. Johnson. Reliability, Weight, and Cost Tradeoffs in the Design of Composite Laminates for Cryogenic Environments. In *Proceedings of the 42rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Seattle, WA*, volume AIAA-2001-1327, April 2001.