JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 2, January 2005

Efficient Search for Trade-Offs by Adaptive Range Multi-Objective Genetic Algorithms

Daisuke Sasaki*
University of Southampton, Southampton, SO17 1BJ, United Kingdom

and
Shigeru Obayashi†
Institute of Fluid Science, Tohoku University, Sendai, 980-8577, Japan

Trade-offs is one of important elements for engineering design problems characterized by multiple conflicting objectives that needs to be simultaneously improved. Further, in many problems such as aerodynamic design, due to computational reasons, only a limited number of evaluations can be allowed for industrial use. This paper proposes new efficient Multi-Objective Evolutionary Algorithms (MOEAs), Adaptive Range Multi-Objective Genetic Algorithms (ARMOGAs), to identify trade-offs among objectives using a small number of function evaluations. The search performance of ARMOGAs is examined by using four different multi-objective analytical test problems. ARMOGAs are also compared with another MOEA. Although the number of evaluations is limited, ARMOGAs showed good performance. In addition, Sequential Quadratic Programming and Dynamic Hill Climber methods are applied to obtain trade-offs for the same problems. These gradient-based methods had some difficulties in identifying trade-offs.

Nomenclature

c_{tol}	=	tolerance of constraint
d_{ij}	=	distance in objective-function space between solutions i and j
F_i	=	fitness value of solution i
F_i'	=	Shared fitness value of solution i
f_k^i	=	k-th Objective-function value of solution i
fl_i	=	half length of plateau region in i-th phenotype design variable
G	=	constraint
M	=	number of objective functions
M_{sa}, M_{ra}		starting generation of range adaptation
M_{ra}	=	interval generation of range adaptation
N	=	number of solutions
$N(0,1)$	=	normal distribution
nc_i	=	niche count of solution i

Received 18 August 2004; revision received 4 November 2004; accepted for publication 20 December 2004.
*Research Fellow, School of Engineering Sciences, Highfield. AIAA Member.
†Professor, Katahira 2-1-1. Associate Fellow AIAA.

p_i = i-th phenotype design variables

$ran1$ = uniform random number defined in [0,1]

R_i = rank of i-th solution

r_i = i-th genotype design variable defined in [0,1]

u_k, l_k = maximum and minimum of k-th objective-function value

x_i = i-th design variable

α, β = weights for utility function

α_r = ratio of population assigned to foot region in range adaptation

α_{share} = sharing function parameter

η_c = SBX parameter

η_m = polynomial mutation parameter

μ(k) = number of solutions having rank k

μ_i = average of i-th phenotype design variable

σ_i = variance of i-th phenotype design variable

σ_{share} = threshold of sharing function

I. Introduction

Industrial design problems often have many design objectives with conflicting requirements. For example, high thrust, low weight and low noise are required when a whole gas-turbine engine is considered. Also, high pressure rise and low total pressure loss are required for compressor design. These problems are typical Multi-Objective (MO) optimization problems. The aim of MO optimization is to determine a single best design that satisfies designers' requirement for each objective. One way is to find a solution based on a trade-off that is determined empirically or interactively by designers. Another way is to first find trade-offs between multiple objectives and then designers select the best solution based on a suitable criterion. In the latter case, the trade-offs are represented by non-dominated solutions, which are solutions that are not dominated by any other solutions as shown in Fig. 1. To select the best solution from a set of non-dominated solutions, it would be better to sample many non-dominated solutions. Ideally, Pareto solutions, which mean global non-dominated solutions that form global trade-offs, should be obtained.

Multi-Objective Evolutionary Algorithms (MOEAs) have gained popularity because of their ability to find global trade-offs.[1-4] MOEAs search from multiple points in the design space simultaneously and stochastically, instead of moving from a single point deterministically like gradient-based methods. This feature prevents design candidates from settling in local optima, and has the capability of finding global optimal solutions. MOEAs can search for optimal solutions in non-smooth design spaces because MOEAs do not require any sensitivity derivatives. It also means that MOEAs are not prone to premature failure even if the design space is very noisy. Various evaluation tools, which compute objective-function values such as Computational Fluid Dynamics (CFD), are easily coupled with MOEAs, because MOEAs only require the objective-function values. In addition, MOEAs can be easily parallelized because of multiple-point search. Due to these advantages, MOEAs have been applied for various multi-objective aerodynamic optimization problems.[5-15]

However, as is well known, MOEAs require a large number of evaluations. This could be a major inhibitor in using MOEAs for aerodynamic optimization problems using time-consuming high-fidelity Computational Fluid Dynamics (CFD), especially in aerospace industries. Therefore, more efficient MOEAs are necessary to conduct sophisticated aerodynamic optimizations using limited computer resources. Real-coded Adaptive Range Multi-Objective Genetic Algorithms (ARMOGAs) have been developed for the purpose of reducing the number of evaluations to make it practical to apply MOEAs to aerodynamic optimization problems. The present method will be also useful for various kinds of multi-objective optimization problems, which require time-consuming evaluation tools.

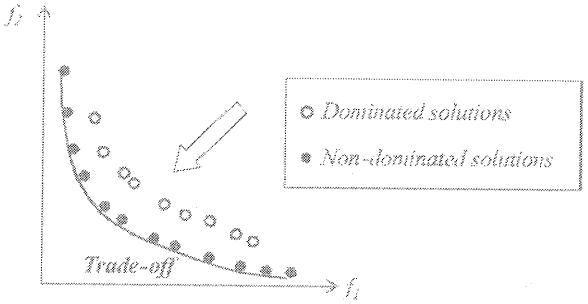


Fig. 1 Trade-off of two-objective minimization problem, which is represented by non-dominated solutions.

The ARMOGAs introduce the range adaptation, which changes the search region according to the statistics of better solutions. Here, a normal distribution is used to represent the design space efficiently, which was originally proposed by Arakawa and Hagiwara in binary-coded Adaptive Range Genetic Algorithms (ARGAs) for single-objective problem.[16,17] Oyama developed real-coded ARGAs for design optimization of transonic wings.[18] In this study, real-coded ARGAs are extended to MO optimization problems to treat multiple solutions and to maintain the diversity of solutions because the aim of MO problems is to collect multiple non-dominated solutions, unlike single-objective problems.

In this paper, the performance of ARMOGAs is studied using four different MO analytical test problems. ARMOGAs are compared to Non-dominated Sorting Genetic Algorithms 2 (NSGA2),[2] which has recently emerged as a powerful MOEA, and two gradient-based methods, Sequential Quadratic Programming method (SQP)[19] and Dynamic Hill Climber method (DHC),[20] both of which are available in Smart Optimization For Turbomachinery (SOFT).[21]

II. Adaptive Range Multi-Objective Genetic Algorithms

Evolutionary Algorithm (EA) is a generic name for population-based optimization methods, such as Genetic Algorithms (GAs), Evolutionary Strategies (ESs), Genetic Programming (GP), etc.[22] The proposed optimization algorithm is based on GAs.[1,3,4,23] GAs simulate the mechanism of natural evolution, where a biological population evolves over generations to adapt to an environment by selection, crossover, and mutation. Fitness, the individual, and genes in the evolutionary theory correspond to the objective function, design candidate, and design variables in design optimization problems, respectively.

GAs have been extended successfully to solve MO problems.[3] GAs use a population to seek optimal solutions in parallel. This feature can be extended to seek Pareto solutions in parallel without specifying weights between the objective functions. The resultant Pareto solutions represent global trade-offs.

Except for the introduction of range adaptation operator, the present ARMOGAs' operators are the same as the MOEAs. Therefore, each genetic operator of the MOEAs adopted here is firstly explained. Figure 2 shows the flowchart of the present MOEAs based on GAs. Then the unique procedure of ARMOGAs is described in this paper.

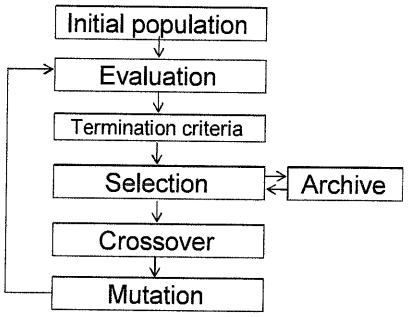


Fig. 2 Flowchart of present MOEAs.

A. Algorithm of Multi-Objective Evolutionary Algorithms

1. Binary and Floating-Point Representation

In GAs, binary numbers are often used to represent design parameter values. However, in many cases such as aerodynamic design optimization, it is more straightforward to use real numbers. Thus, the floating-point representation is adopted here.

2. Coding and Decoding

GAs require both phenotype and genotype design variables. The phenotype design variable represents the actual design variables, such as length, angle, shape, etc. On the other hand, the genotype design variable is a binary number (Binary GAs) or a real number in [0,1] (Real-coded GAs). The operators of many GAs require genotype representation of design parameters. Therefore, actual design variables (phenotype representation) must be converted to the genotype representation. For real-parameter design problems, such as aerodynamic optimizations, it is not favorable to use binary representation because phenotype design space is not continuous by binary representation. Therefore, the present MOEA adopts a floating-point representation. For the floating-point representation, the *i*-th design parameter pi is coded to genotype value r_i, which is normalized in [0,1]:

$$r_i = \frac{p_i - p_{i,min}}{p_{i,max} - p_{i,min}} \tag{1}$$

3. Initial Population

The results of GAs can be affected by the initial population if the number of individuals per generation is small. It would be better to generate initial individuals which are uniformly spread out in the design space. Here, the initial population is generated randomly.

4. Evaluation

As GAs use only objective-function values for optimization, no modification of evaluation tools is required. In addition, it is easy to apply Master-Slave type parallelization systems to conserve computational resources because GAs do not have to compute design candidates sequentially, unlike gradient-based methods.

5. Selection

GAs choose superior individuals as parents to generate new design candidates. Therefore, selection has a large influence on search performance of GAs. For single-objective optimizations, as the aim is to obtain the best solution, selection is based on the fitness value given by the objective-function value. However, Pareto-optimal solutions must be obtained for MO optimization. To obtain Pareto solutions effectively, each individual is assigned a rank based on the Pareto ranking method and fitness sharing.[1] In the present MOEAs, Fleming and Fonseca's Pareto-ranking method[3] is adopted. Each individual is assigned a rank according to the number of individuals dominating it, as shown in Fig. 3. The fitness value (Fi) of individual *i* is assigned based on the following equation:

$$F_i = N - \sum_{k=1}^{R_i-1} \mu(k) - 0.5\left(\mu(R_i) - 1\right) \tag{2}$$

where N is the number of solutions, and $\mu(R_i)$ is the number of solutions in rank R_i. Thereafter, the standard sharing approach is adopted to prevent choosing similar solutions as parents and to maintain diversity of the population.[4] The assigned fitness values are divided by the niche count:

$$F_i' = F_i / nc_i \tag{3}$$

where niche count nc_i is calculated by summing the sharing function values:

$$nc_i = \sum_{j=1}^{N} sh(d_{ij}) \tag{4}$$

$$sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{share}} \right)^{\alpha_{share}} & d_{ij} < \sigma_{share} \\ 0 & others \end{cases} \tag{5}$$

$$d_{ij} = \sqrt{\sum_{k=1}^{M} \left(\frac{f_k^i - f_k^j}{u_k - l_k} \right)} \tag{6}$$

where u_k is the maximum objective-function value of k at the present generation, l_k is the minimum objective-function value of k at the present generation, and α_{share} is the sharing function parameter. If the distance between individuals i and j is lower than σ_{share}, then niche count increases to reduce the fitness of the solution. The normalized niching parameter σ_{share} is proposed as follows:

$$(1 + \sigma_{share})^M - 1 = N \cdot (\sigma_{share})^M \tag{7}$$

where M is the number of objective functions.

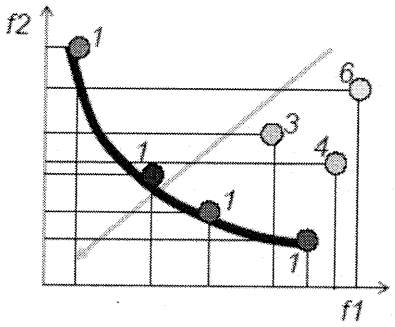


Fig. 3 Rank of solutions based on Pareto ranking method[3] (Rank 1 means non-dominated solutions).

After shared fitness values are determined for all individuals, the stochastic universal selection (SUS)[24] is applied to select better solutions for producing a new generation. Unlike roulette wheel selection method, only one random number is chosen for the whole selection process for SUS. As many different solutions should be chosen to maintain the diversity, a set of N equi-spaced numbers is created as shown in Fig. 4.

Individual	Fitness	Ratio	Selection Probability
I - 1	6.1	0.339	1.000
I - 2	3.2	0.178	0.711
I - 3	4.1	0.228	0.911
I - 4	2.9	0.161	0.644
I - 5	1.7	0.094	0.378

< Choice of 4 Parents >

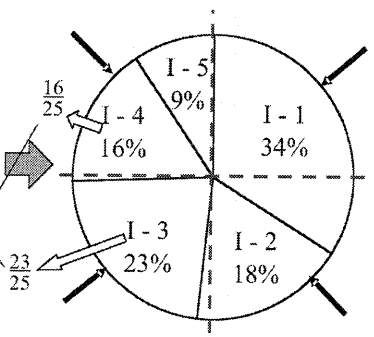


Fig. 4 SUS method.[24]

6. Crossover

Crossover is an operator that interchanges the genotype parameters of selected parents and produces two different design candidates. The probability of crossover and the crossover method can have a significant impact on the search performance of GAs.

For binary representations, crossover interchanges the bit strings of selected parents. However, many crossover methods have been proposed for real-parameter GAs. Simulated binary crossover (SBX)[1] operator creates offspring based on the distance between the parents as shown in Fig. 5. If the two parents are closely related to each other, SBX is likely to generate new offspring near the parents. On the other hand, if the two parents are more distantly related, it is possible for solutions to be created away from the parents. This operator is described as follows:

$$Child1 = 0.5\left[\left(1+\beta_q\right)\cdot Parent1 + \left(1-\beta_q\right)\cdot Parent2\right] \tag{8a}$$

$$Child2 = 0.5\left[\left(1-\beta_q\right)\cdot Parent1 + \left(1+\beta_q\right)\cdot Parent2\right] \tag{8b}$$

$$\beta_q = \begin{cases} (2\cdot ran1)^{1/(\eta_c+1)} - 1 \\ \left(\dfrac{1}{2\cdot(1-ran1)}\right)^{1/(\eta_c+1)} \end{cases} \tag{8c}$$

where *ran1* is an uniform random number in [0,1].

7. Mutation

Mutation maintains diversity and expands the search space by changing the design parameters. If the mutation rate is high, a GA search is close to a random search and results in slow convergence. Therefore, an adequate value is required for the mutation rate. For binary representation, mutation is performed to reverse the bit strings. It is not as simple for real-coded GAs as for binary GAs. This is realized by adding disturbances to the design parameters.

Polynomial mutation[1], which is similar to the SBX operator described in Sec. 2.1.6, has been proposed:

$$Child_{mutation} = Child_{crossover} + \left(r_{i,\max} - r_{i,\min}\right).\delta \tag{9}$$

where $r_{i,max}$, $r_{i,min}$ are upper and lower boundaries of i-th genotype design variable, and δ is calculated from the polynomial probability distribution:

$$\delta = \begin{cases} (2\cdot ran1)^{1/(\eta_m+1)} - 1 \\ 1 - \left[2\cdot(1-ran1)^{1/(\eta_m+1)}\right] \end{cases} \tag{10}$$

where the value of η_m determines the perturbation size of mutation.

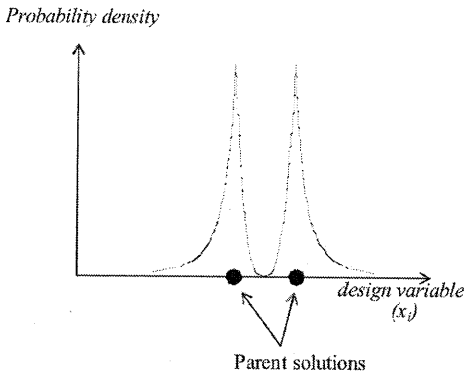


Fig. 5 Sketch of simulated binary crossover.

8. Archiving

To obtain Pareto solutions efficiently, it would be better to include past excellent solutions as current solutions. In the present MOEAs, two archiving techniques are combined. The first is the *Best-N* technique, which keeps the latest better solutions and parent generation of *(x-1)N* size and uses these solutions for the selection process. The second is the standard archiving technique, which is comprised of all previous solutions to prevent the loss of previous excellent solutions.[1] These two methods are combined in the present MOEAs as shown in Fig. 6. The procedure is as follows:

1) Fitness values based on the fitness assignment operators in Sec. 2.1.5 are assigned to the present population and the *Best-xN* group. Here, x is set to 2.
2) According to the fitness value, the top N individuals are chosen for the next step. In addition, the top *(x-1)N* individuals are preserved as the *Best-xN* group.
3) Fitness values are assigned to chosen N individuals.
4) SUS is used to select the parents. Then, crossover and mutation are applied to generate new individuals.
5) Several individuals in the *Best-xN* group are replaced by the same number of individuals from the archives.

9. Constraint-Handling Technique

In many real-world problems, it is common to have several constraints. Many constraint-handling techniques have been proposed,[1] however, it is not easy for GAs to solve constrained-problems compared to gradient-based methods, such as the Modified Method of Feasible Direction (MMFD)[19]. A popular and easy constraint-handling strategy is the penalty function approach in which a penalty value is added to the objective-function value if the design violates the constraint. Although several penalty functions have been proposed, it is difficult to choose appropriate penalty values a priori.

In the present MOEAs, an extended Pareto ranking method based on constraint-dominance is used. Constraint-dominance is defined as follows[1]:

A solution $\mathbf{x}_i$ is said to 'constrain-dominate' a solution $\mathbf{x}_j$, if any of the following conditions are true:

1) $\mathbf{x}_i$ is feasible and $\mathbf{x}_j$ is not.
2) $\mathbf{x}_i$ and $\mathbf{x}_j$ are both infeasible, but $\mathbf{x}_i$ has a smaller constraint violation.
3) $\mathbf{x}_i$ and $\mathbf{x}_j$ are feasible and $\mathbf{x}_i$ dominates $\mathbf{x}_j$ in the usual sense.

Figure 7 shows the example of a Pareto ranking method based on constrain-dominance for the two-objective minimization problem with one constraint. Based on this approach, it would be easy to generate new offspring that satisfy the constraints because feasible solutions are likely to be chosen as the parents. However, it is possible for good solutions to lie close to the edge of the feasible and infeasible region in many design problems. Therefore, an adequate tolerance of the constraint (c_{tol}) should be introduced to the constraint violation:

$$G - c_{tol} \leq 0 \tag{11}$$

where G is an original constraint less than zero. As the tolerance c_{tol} is introduced, solutions having smaller violation than ctol are assumed to be feasible for constraint-dominance. This enables EAs to search for solutions near the boundary between feasible and infeasible solutions.

To tackle aerodynamic optimization using time-consuming CFD codes, it is unfavorable to generate many violated candidates. If it is possible to estimate the feasibility of any candidate designs efficiently before time-consuming CFD computation, it is reasonable to prevent generating such solutions, as it would be a waste of computational time in CFD.

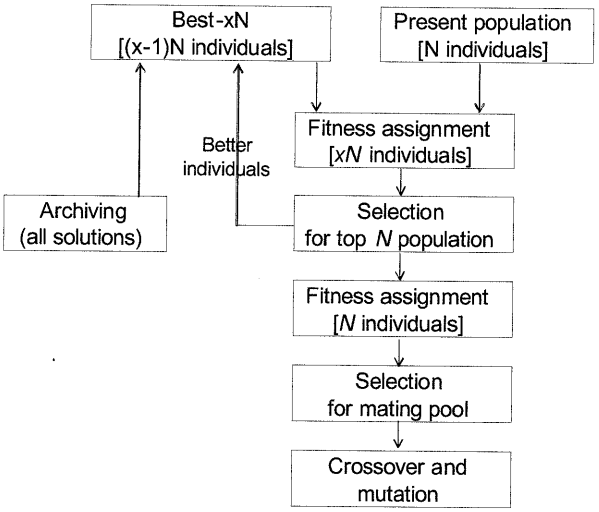


Fig. 6 Archiving procedure used in the present MOEAs.

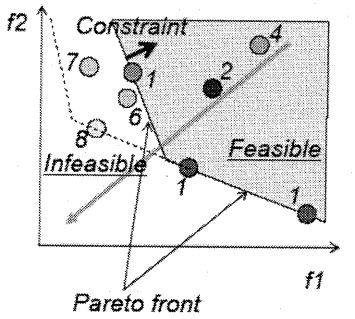


Fig. 7 Example of rank based on constraint-dominance definition.

B. Algorithm of Adaptive Range Multi-Objective Genetic Algorithms

To reduce the large computational burden, the total number of function evaluations need to be reduced On the other hand, a large string length is necessary for real parameter problems. ARGAs, originally proposed by Arakawa and Hagiwara, are a quite unique approach to solve such problems efficiently.[16,17] Oyama developed real-coded ARGAs and applied them to transonic wing optimization.[18] According to the encoding system (Fig.8) based on normal distribution built by population statistics consisting of better designs computed before, ARGAs can find near optimal designs efficiently.

The basis of ARMOGAs is the same as ARGAs, but a straightforward extension may cause problems with the diversity of the population. Therefore, ARMOGAs have been developed based on ARGAs to deal with multiple Pareto solutions for multi-objective optimization. In addition, archiving and constraint-handling techniques are considered to select better solutions to decide new search range.

This section describes the genetic operators of ARMOGAs. ARMOGAs differ from MOEAs described above with regard to the application of range adaptation. Therefore, before starting range adaptation, the MOEAs and

ARMOGAs in the present study are identical. A flowchart of ARMOGAs is shown in Fig. 9. The range adaptation starts at M_{sa} generation and is carried out every M_{ra} generations. The new decision space is determined based on the statistics of selected better solutions, and then the new population is generated in the new decision space. Thereafter, all the genetic operators are applied to the new design space.

ARMOGAs are able to find Pareto solutions more efficiently than conventional MOEAs because of the concentrated search of the promising design space out of the large, initial design space. ARMOGAs can adapt their search region as shown in Fig. 10. In contrast, the search region of conventional EAs remains unchanged. The encoding system is based on the normal distribution with the plateau region as shown in Fig. 10. The selected designs are located in the plateau region, and the normal distribution region is determined based on the population statistics to preserve the diversity of candidate solutions. Re-initialization helps to maintain the population diversity. However, there is a conflict between concentration of the search space and the maintenance of population diversity.

1. Sampling for Range Adaptation

Range adaptation needs to select superior solutions to determine the new design space based on some statistics. The solutions, which have higher fitness values based on Pareto ranking method, are selected to determine the reasonable search range. It would be better to select many solutions to prevent the creation of new search regions that do not include the global optimum. On the other hand, many solutions for range adaptation generally interfere with the decrease in size of the search space. The solutions are selected at random according to their fitness given by the following solution sets:

1) PR_{non}% non-dominated solutions from all solutions. (PR_{non}=100)
2) PR_{arc}% solutions from the archive. (PR_{arc}=0)
3) PR_{prs}% solutions from the latest generation. (PR_{prs}=0)
4) PR_{vio}% solutions that violate the constraint. (PR_{vio}=1, at least one design)

Solution set 4 is introduced to search near the boundary between feasible and infeasible solutions, as the global-optimum for constraint problems is often located there. According to the amount of violation, violated designs are sampled. The probabilities in bracket are used in this optimization. In this case, only non-dominated solutions with several infeasible designs are selected to determine new design range.

2. Range Adaptation

In ARMOGAs, the search region is changed according to the population statistics of the average and the standard deviation. The range adaptation adopts the Normal distribution to search global solutions efficiently. Figure 8 shows the normal distribution used for encoding in the real-coded ARGAs. The real value of the i-th design variable p_i is encoded to a real number r_i defined in (0,1) such that r_i is equal to the integrations of the normal distribution from -∞ to $p_{n,i}$:

$$r_i = \int_{-\infty}^{p_{n,i}} N(0,1)(z)dz \tag{12a}$$

$$p_{n,i} = \frac{p_i - \mu_i}{\sigma_i} \tag{12b}$$

where μ_i is the average of i-th phenotype design variable, and σ_i is the standard deviation of i-th phenotype design variable.

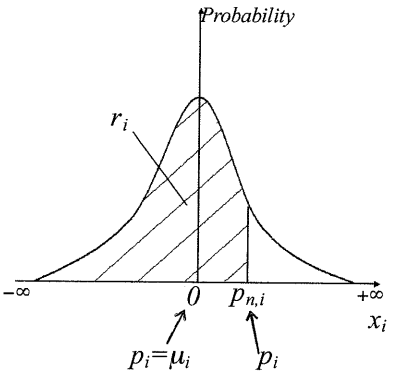


Fig. 8 Normal distribution for encoding in real-coded ARGAs.

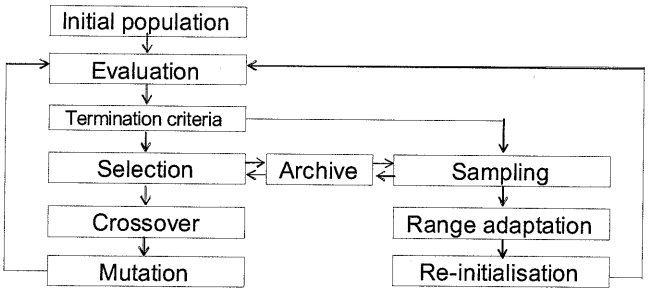
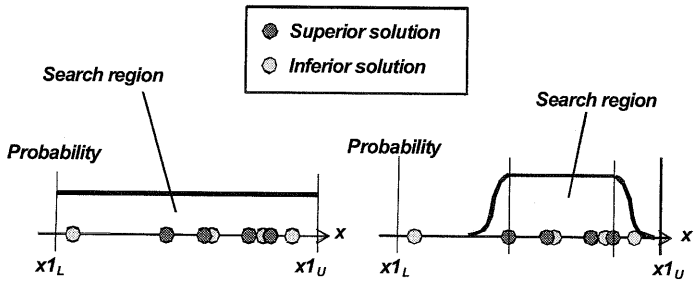


Fig. 9 Flowchart of ARMOGAs.



(a) Conventional MOEAs **(b) ARMOGAs**

Fig. 10 Sketch of search region.

The basic encoding system in ARMOGAs is the same as for real-coded ARGAs, but a straightforward extension is not suitable to preserve the diversity of the population. To better preserve the diversity of candidate solutions, the normal distribution for encoding has to be changed.

Figure 11 shows the search range with the distribution of probability. The search region is partitioned into three parts, I, II, and III. Regions I and III make use of the same encoding method as ARGAs. The real value of i-th design variable p_i is encoded to a real number r_i defined in (0,1). In contrast, region II adopts the conventional real-number encoding method. The plateau region (region II) is defined by the upper and lower design variables of chosen solutions. Then, the normal distribution is considered at both sides of the plateau determined by the average (μ_i) and the standard deviation (σ_i). This encoding system is controlled by the parameters α_r and fl_i, where α_r (<0.5) is the population ratio at region I and fl_i is half the length of the plateau at region II. The encoding is conducted at each region described below.

Region I ($p_i \le \mu_i - fl_i$, $0 \le r_i \le \alpha_r$):

$$r_i = \alpha_r \cdot r_i' \tag{13a}$$

$$r_i' = \int_{-\infty}^{p_{n,i}} N(0,1)(z)dz \tag{13b}$$

$$p_{n,i} = \frac{p_i - (\mu_i - fl_i)}{2\sigma_i} \tag{13c}$$

Region II ($\mu_i - fl_i < p_i < \mu_i + fl_i$, $\alpha_r < r_i < 1-\alpha_r$):

$$r_i = (1 - 2\alpha_r) \cdot r_i' + \alpha_r \tag{13d}$$

$$r_i' = \frac{p_i - (\mu_i - fl_i)}{2fl_i} \tag{13e}$$

Region III ($\mu_i + fl_i \le p_i$, $1-\alpha_r \le r_i \le 1$):

$$r_i = \alpha_r \cdot r_i' + (1 - \alpha_r) \tag{13f}$$

$$r_i' = \int_{-\infty}^{p_{n,i}} N(0,1)(z)dz \tag{13g}$$

$$p_{n,i} = \frac{p_i - (\mu_i + fl_i)}{2\sigma_i} \tag{13h}$$

III. Results and Discussions

ARMOGAs are evaluated by applying them to four different types of MO analytical problems. ARMOGAs are compared with another MOEA and two gradient-based methods: NSGA2 (a widely-used MOEA)[2], SQP (efficient gradient-based method)[19] and DHC (robust gradient-based method)[20]. SQP and DHC in SOFT[21] developed by Rolls-Royce plc. and UTCs are used. These gradient-based methods require the following utility function f to solve MO problems:

$$f = \alpha \cdot f_1 + \beta \cdot f_2 \tag{14}$$

where f_1 and f_2 represent objective-function values, and α and β represent weights. By changing the weights, the optimizer seeks different optimal solutions corresponding to the current utility function f. Therefore, trade-offs can be obtained by changing the weights. In the figure, the following name (SQP_2.0-1.0) is used to represent the

different utility function. SQP_2.0-1.0 means optimizer is SQP, α and β is set to 2.0 and 1.0, respectively. Initial searching point and optimal solution is also indicated in the figure (O) and the table.

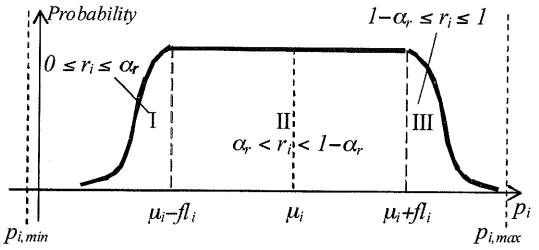
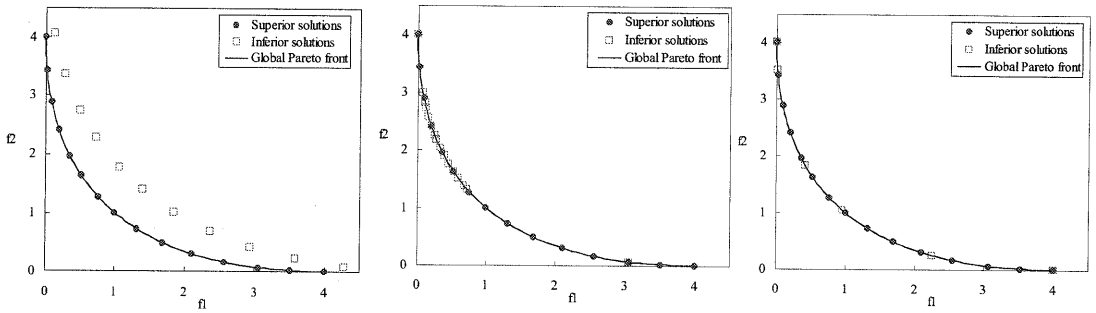


Fig. 11 Sketch of probability distribution of phenotype design variable p_i in ARMOGAs.



(a) Closeness (b) Spread (c) Sampling

Fig. 12 Superior and inferior non-dominated front considered in the present MO optimization.

The search performance of optimizers is evaluated in terms of closeness, reasonable spread, and many samplings in the Pareto front as shown in Fig. 12. These characteristics will help to understand the trade-off between objectives. ARMOGAs and NSGA2 use comparatively small number of evaluations for aerodynamic optimization problems requiring time-consuming CFD. ARMOGAs generate different candidate designs all the time to prevent wasting computational time evaluating the same design and also to use a Master-Slave type parallel processing of the evaluation tool. The population size (*pop*) and the generation (*gen*) are set to eight and 20, respectively. On the other hand, NSGA2 may create the same design if the number of design variables is small. To ensure all EAs are almost same number of function evaluations (*call*), the total number of generations is set to 30 in NSGA2. When the Master-Slave type parallelization of evaluation is adopted, ARMOGAs can obtain the result faster than NSGA2 in this case.

Both ARMOGAs and NSGA2 adopt crossover rate 1.0, SBX crossover with η_c=2.0, mutation rate 0.1, polynomial mutation with η_m=5.0. As GAs often depend on an initial population, three different initial populations are used for the comparison. In ARMOGAs, range adaptation starts at fifth generation (M_{sa}=5) and then the range adaptation occurs every five generations (M_{ra}=5). Three trials are performed by changing initial population because MOEAs are stochastic approach.

A. Convex Pareto Front Case

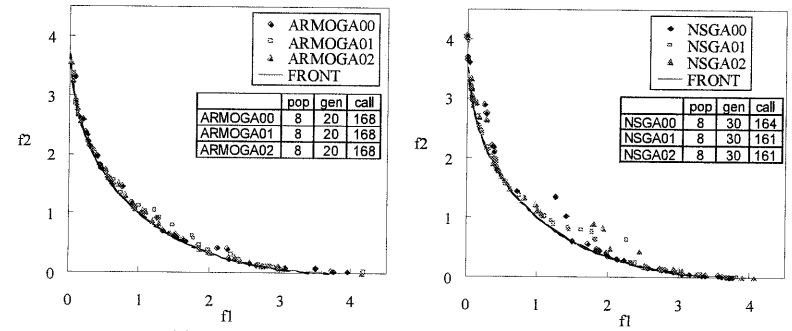
This problem has two objective functions to be minimized as formulated below[25]:

$$\text{Minimize } f_1(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i^2 \tag{15a}$$

$$\text{Minimize } f_2(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - 2) \tag{15b}$$

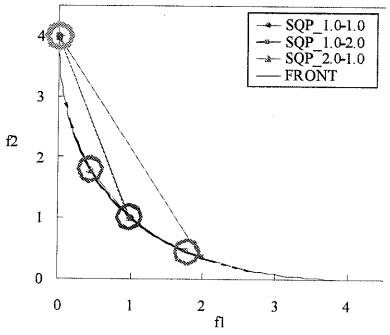
$$\text{subject to } -4 \le x_i \le 4, \quad i = 1, 2$$

Pareto-optimal solutions have x_i in [0,2] and the corresponding Pareto front is convex. The optimization was conducted by ARMOGAs, NSGA2, SQP (2cases) and DHC (2cases). Figures 13 (a) and (b) show the optimization results of ARMOGAs and NSGA2, respectively. Both MOEAs could obtain good spread of Pareto solutions. Regarding the gradient-based method, weights of utility function [α and β in Eq. (14)] were changed to obtain trade-offs as described in Table 1. Two different initial points were used for comparison. Table1 shows the numbers of function calls, initial points, and optimal solutions. Figures 13 (c) and (d) show the search histories of SQP-1 and DHC-1. Because this convex problem is easy to solve by gradient-based methods, simple gradient-based method, SQP, could find the optimal solutions rapidly. These two algorithms obtained same final Pareto solutions according to the utility function by changing the initial points as indicated in Table 1. The difference between two gradient-based methods is the number of evaluations. SQP could obtain final Pareto solutions rapidly, but DHC required a large number of evaluations similar to MOEAs.

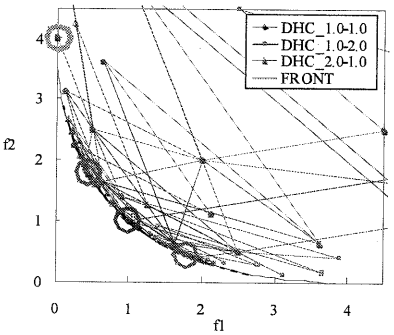


(a) ARMOGAs

(b) NSGAs



(c) SQP-1



(d) DHC-1

Fig. 13 Comparison of optimization results shown in the objective function space for convex Pareto front case.

Table 1 Optimization summary of gradient-based methods for convex Pareto front case

(a) SQP-1

weight	call	initial	optimal
1.0-1.0	6	(0.0, 4.0)	**(1.00, 1.00)**
1.0-2.0	9	(0.0, 4.0)	**(1.78, 0.44)**
2.0-1.0	9	(0.0, 4.0)	**(0.44, 1.77)**

(b) DHC-1

weight	call	initial	optimal
1.0-1.0	48	(0.0, 4.0)	**(1.00, 1.00)**
1.0-2.0	145	(0.0, 4.0)	**(1.78, 0.44)**
2.0-1.0	132	(0.0, 4.0)	**(0.44, 1.78)**

(c) SQP-2

weight	call	initial	optimal
1.0-1.0	9	(9.0, 25.0)	**(1.00, 1.00)**
1.0-2.0	9	(9.0, 25.0)	**(1.78, 0.44)**
2.0-1.0	9	(9.0, 25.0)	**(0.44, 1.77)**

(d) DHC-2

weight	call	initial	optimal
1.0-1.0	39	(9.0, 25.0)	**(1.00, 1.00)**
1.0-2.0	139	(9.0, 25.0)	**(1.78, 0.44)**
2.0-1.0	121	(9.0, 25.0)	**(0.44, 1.78)**

B. Concave Pareto Front Case

This problem has a concave Pareto front[26]: The problem is formulated as follows:

$$\text{Minimize } f_1(\mathbf{x}) = 1 - \exp\left(-\sum_{i=1}^{2}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right) \tag{16a}$$

$$\text{Minimize } f_2(\mathbf{x}) = 1 - \exp\left(-\sum_{i=1}^{2}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \tag{16b}$$

$$\text{subject to } -4 \le x_i \le 4, \quad i = 1, 2$$

The same six cases of optimization as the previous section were conducted. Figures 14 (a) and (b) show the non-dominated front of ARMOGAs and NSGA2. Both GAs could obtain approximate Pareto solutions with reasonable spread. All the results of gradient-based method is summarized in Table 2. SQP-1 could obtain Pareto solutions at three cases, but SQP-2 could not obtain Pareto solutions, which started from different initial points. DHC-2 started from the same point as SQP-2, but it was able to find the global Pareto solutions because DHC is a more robust approach. The gradient-based methods with weight function could find global optima, but it was difficult to obtain trade-offs because a final optimal solution always reaches the extreme Pareto solution as shown in Figs 14 (c) and (d). Figures 15 (a) and (b) show contours of the objective-function f_1 and f_2 for design variables x_1 and x_2. Figure 15 (c) shows contour of utility function f with α=2.0 and β=1.0. Even if the utility function is changed, only the optima of either objective function f_1 or f_2 is finally obtained. When the utility function is used to identify trade-offs of concave Pareto front cases, optimizers can only obtain the extreme optimal solutions.

Table 2 Optimization summary of gradient-based methods for concave Pareto front case

(a) SQP-1

weight	call	initial	optimal
1.0-1.0	3	(0.63, 0.63)	
1.0-2.0	17	(0.63, 0.63)	**(0.98, 0.00)**
2.0-1.0	17	(0.63, 0.63)	**(0.00, 0.98)**
1.0-1.2	26	(0.63, 0.63)	**(0.98, 0.00)**

(b) DHC-1

weight	Call	initial	optimal
1.0-1.0	99	(0.63, 0.63)	**(0.00, 0.98)**
1.0-2.0	113	(0.63, 0.63)	**(0.98, 0.00)**
2.0-1.0	136	(0.63, 0.63)	**(0.00, 0.98)**

(c) SQP-2

weight	call	Initial	optimal
1.0-1.0	6	(1.0,1.0)	
1.0-2.0	3	(1.0,1.0)	
2.0-1.0	3	(1.0,1.0)	

(d) DHC-2

weight	Call	initial	optimal
1.0-1.0	104	(9.0, 25.0)	**(0.98, 0.00)**
1.0-2.0	100	(9.0, 25.0)	**(0.98, 0.00)**
2.0-1.0	128	(9.0, 25.0)	**(0.00, 0.98)**

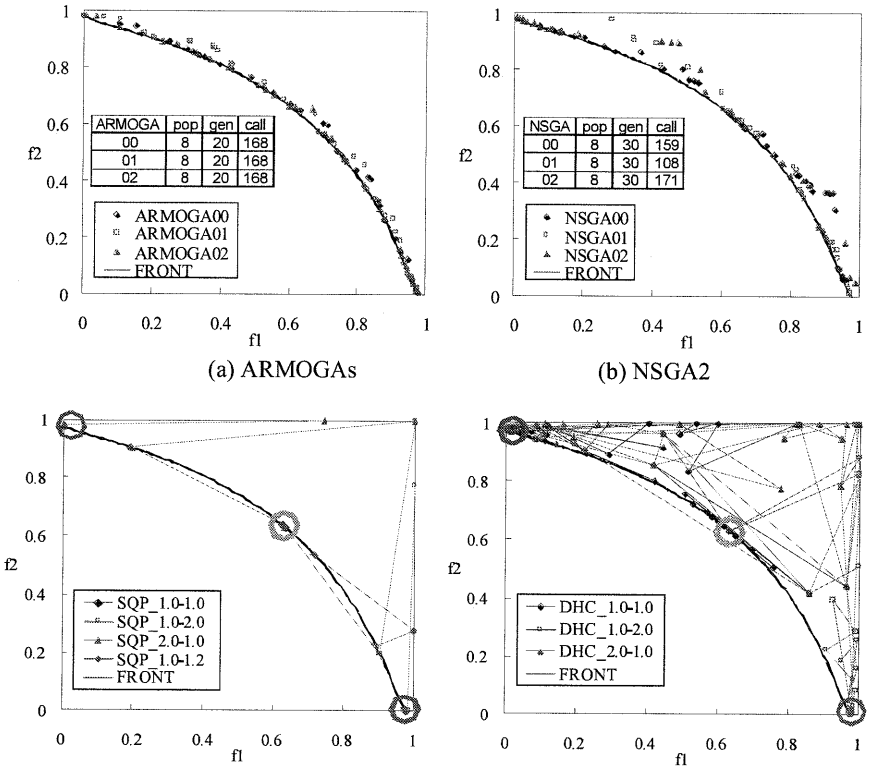
C. Discontinuous Pareto Front Case

This problem has a nonconvex as well as disconnected Pareto-optimal set, composed of three disconnected Pareto-optimal fronts and single point (-20,0).[27] The formulation is as follows:

$$\text{Minimize } f_1(\mathbf{x}) = \sum_{i=1}^{2}\left[-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right] \tag{17a}$$

$$\text{Minimize } f_2(\mathbf{x}) = \sum_{i=1}^{3}|x_i|^{0.8} + 5\sin\left(x_i^3\right) \tag{17b}$$

$$\text{subject to } -5 \le x_i \le 5, \quad i = 1, 2, 3$$



(a) ARMOGAs (b) NSGA2

(c) SQP-1 (d) DHC-1

Fig. 14 Comparison of optimization results shown in the objective function space for concave Pareto front case.

ARMOGAs could obtain better spread in non-dominated front compared to NSGA2 as shown in Fig. 16. As the number of evaluations is small, only the overview of the trade-offs were identified. Figures 16 (c) – (f) show the search history of gradient-based methods. Table 3 summarizes the optimization results. DHC obtained similar optima even by changing utility functions. SQP could not search Pareto solutions, and it could only find a local optima of the utility function. These figures show the danger in the use of utility function because it is possible to misunderstand the trade-off between objectives.

D. Constrained Test Case

It is common that there are many constraints when solving industrial optimization problems. In this section, a constrained analytical test problem solved by MOEAs is described. This problem is formulated as follows[1]:

$$f_1(\mathbf{x}) = x_1 \tag{18a}$$

$$f_1(\mathbf{x}) = \frac{1 + x_2}{x_1} \tag{18b}$$

$$\text{subject to } g_1(\mathbf{x}) = 6 - x_2 - 9x_1 \le 0 \tag{18c}$$

$$g_2(\mathbf{x}) = 1 + x_2 - 9x_1 \le 0 \tag{18d}$$

$$0.1 \le x_1 \le 1 \; 0 \le x_2 \le 5$$

This problem has two objective functions and two constraints. The real Pareto front is cut by the constraints as shown in Fig. 17. Two different constraint-handling techniques are used in ARMOGAs. The constrained Pareto ranking method is used in ARMOGA-CP. On the other hand, ARMOGA-CG always evaluates feasible designs by preventing the generation of infeasible designs. Briefly, all new individuals are generated repeatedly until they satisfy the two constraints. Table 4 describes the optimization conditions for ARMOGAs and NSGA2. Figure 18 shows the non-dominated solutions of the three methods. Both ARMOGAs could obtain the approximate Pareto front, which is well-dispersed along the real Pareto front. In contrast, NSGA2 could obtain many non-dominated solutions just around the lower region, but failed to obtain the Pareto front cut by the constraint. When the results of both ARMOGAs are compared, the non-dominated solutions of ARMOGA-CP are closer to the Pareto front than these of ARMOGA-CG. This is because ARMOGA-CG could not create design candidates that were located close to the constraints because the method generated new designs only in the feasible region. In terms of obtaining trade-offs, ARMOGA-CG was able to find reasonable non-dominated solutions. The method would be practical for constrained multi-objective aerodynamic optimization problems to avoid CFD computation of hopeless designs that are infeasible.

Table 3 Optimization summary of gradient-based methods for discontinuous Pareto front case

(a) SQP-1

weight	call	initial	optimal
1.0-1.0	14	(-20.0, 0.0)	
1.0-2.0	14	(-20.0, 0.0)	
2.0-1.0	14	(-20.0, 0.0)	
1.0-50.0	97	(-20.0, 0.0)	(-9.75, -8.44)

(b) DHC-1

weight	call	initial	optimal
1.0-1.0	115	(-20.0, 0.0)	**(-14.52, -11.58)**
1.0-2.0	160	(-20.0, 0.0)	**(-14.48, -11.62)**
2.0-1.0	33	(-20.0, 0.0)	
1.0-50.0	138	(-20.0, 0.0)	**(-14.44, -11.63)**

(c) SQP-2

weight	call	initial	optimal
1.0-1.0	43	(-8.6, 21.6)	(-7.26, -4.58)
1.0-2.0	45	(-8.6, 21.6)	(-7.47, -7.57)
2.0-1.0	49	(-8.6, 21.6)	(-8.40, -7.41)
1.0-50.0	59	(-8.6, 21.6)	(-7.87, -6.91)

(d) DHC-2

weight	call	initial	optimal
1.0-1.0	170	(-8.6, 21.6)	**(-14.52, -11.58)**
1.0-2.0	163	(-8.6, 21.6)	**(-14.48, -11.62)**
2.0-1.0	152	(-8.6, 21.6)	(-11.64, -9.64)
1.0-50.0	170	(-8.6, 21.6)	(-11.56, -9.72))

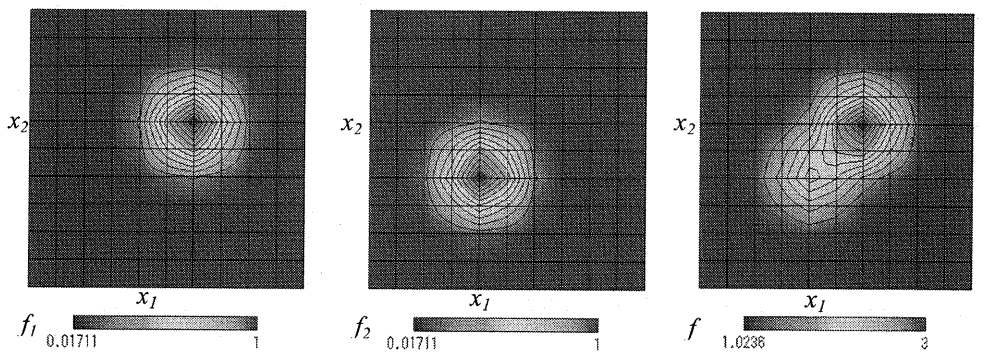
Table 4 Conditions of optimization for constrained test problem

(a) ARMOGA-CP and ARMOGA-CG

	pop	generation	call
ARMOGA-1	8	20	168
ARMOGA-2	8	20	168
ARMOGA-3	8	20	168

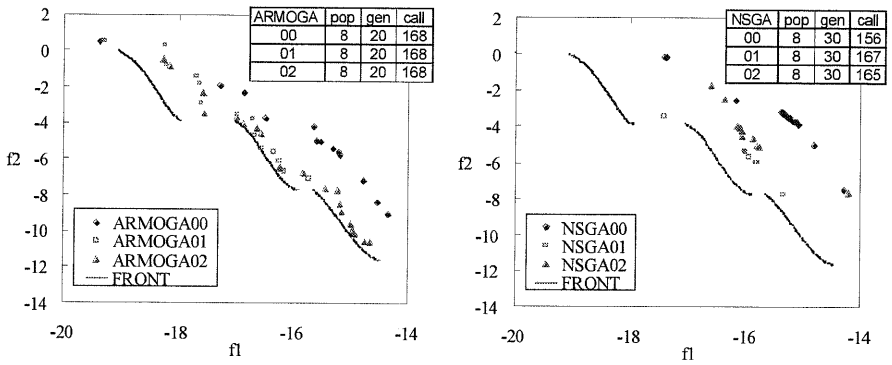
(b) NSGA2

	pop	generation	call
NSGA2-1	8	30	157
NSGA2-2	8	30	142
NSGA2-3	8	30	165

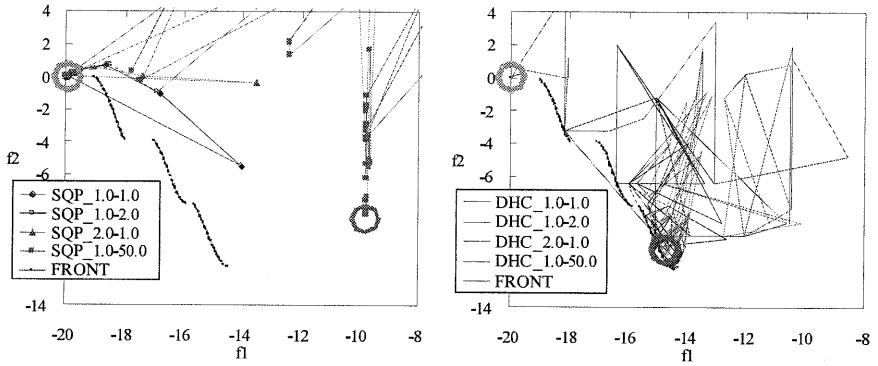


(a) Objective function f_1 **(b) Objective function f_2** **(c) Utility function f (α=2.0, β=1.0)**

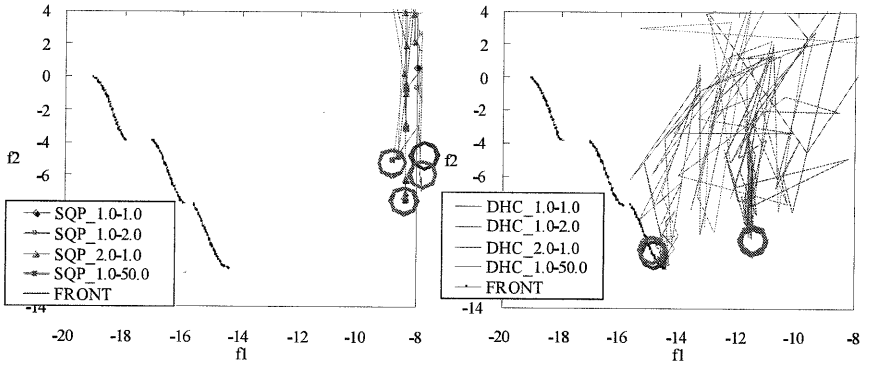
Fig. 15 Contours of objective functions f_1, f_2 and utility function f for design variable x_1, x_2.



(a) ARMOGAs (b) NSGA2



(c) SQP-1 (d) DHC-1



(e) SQP-2 (f) DHC-2

Fig. 16 Comparison of optimization results shown in the objective function space for discontinuous Pareto front case.

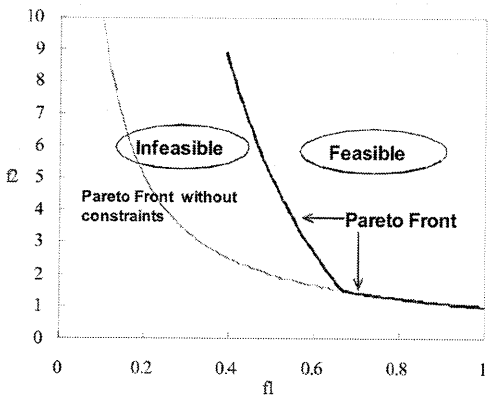
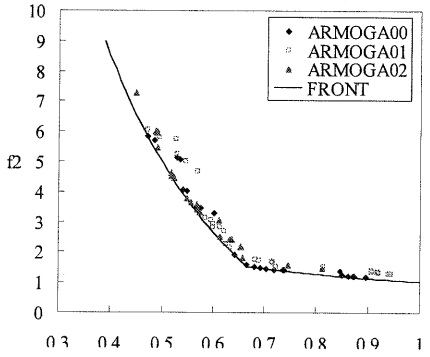
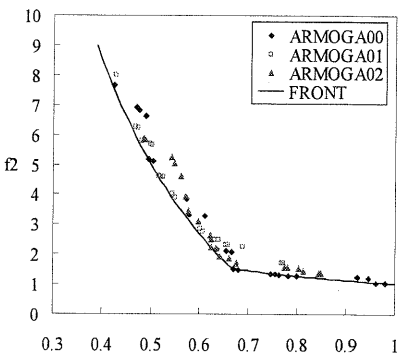


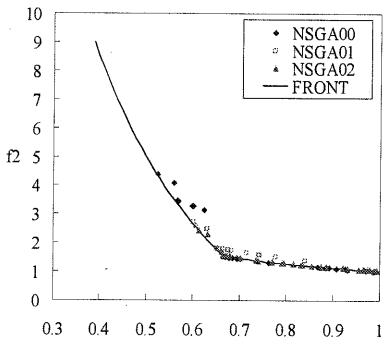
Fig. 17 Pareto front of constrained problem.



(a) ARMOGA-CP



(b)ARMOGA-CG



(c) NSGA2

Fig. 18 Comparison of optimization results in the objective-function space for constrained test case.

IV. Conclusion

In this study, real-coded ARMOGAs, which aims to find non-dominated solutions efficiently, were presented. ARMOGAs were developed to solve multi-objective optimization problems based on real-coded ARGAs. The sophisticated encoding system composed of the normal distribution and the plateau region is adopted to maintain diversity of population. To determine the new search region by range adaptation, designs are sampled based on the archiving technique. Constraint-handling techniques are also introduced in the sampling procedure.

The performance of ARMOGAs was examined using four analytical test problems. ARMOGAs showed reasonable search performance in all cases. These test problems were also solved by NSGA2 and two gradient-based methods (SQP and DHC) for comparison. ARMOGAs were able to find a reasonable quality non-dominated front using a small number of function evaluations comparable to DHC. On the other hand, gradient-based methods were not suitable for obtaining trade-offs, although DHC was slightly more robust than SQP. The performance of NSGA2 was compared to ARMOGAs for unconstrained test problems. However, NSGA2 could not find well-dispersed non-dominated solutions of a constrained test problem. Therefore, ARMOGAs will be useful for multi-objective and/or multi-disciplinary aerodynamic optimization with time-consuming high-fidelity CFD.

Acknowledgments

The authors would like to thank Dr. Shahpar, who is an aerothermal design specialist at the Aerothermal Methods Group, Rolls-Royce plc, Derby, United Kingdom. This research was performed during the first author's stay at Rolls-Royce plc under the industrial trainee program.

References

[1]Deb, K., *Multi-Objective Optimization using Evolutionary Algorithms*, John Wiley & Sons, Ltd., Chichester, 2001.

[2]Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II," *Proceedings of the Parallel Problem Solving from Nature VI Conference*, 2000.

[3]Fonseca, C. M., and Fleming, P. J., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1993, pp. 416-423.

[4]Obayashi, S., Takahashi, S., and Takeguchi, Y., "Niching and Elitist Models for MOGAs," *Proceedings of Parallel Problem Solving from Nature – PPSN V*, Lecture Notes in Computer Science 1498, Springer, Berlin, 1998, pp. 260-269.

[5]Obayashi, S., Tsukahara, T., and Nakamura, T., "Multiobjective Genetic Algorithm Applied to Aerodynamic Design of Cascade Airfoils," *IEEE Transactions on Industrial Electronics*, Vol. 47, No. 1, 2000, pp. 211-216.

[6]Obayashi, S., Sasaki, D., Takeguchi, Y., and Hirose, N., "Multiobjective Evolutionary Computation for Supersonic Wing-Shape Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 4, No. 2, 2000, pp. 182-187.

[7]Sasaki, D., Obayashi, S., and Nakahashi, K., "Navier-Stokes Optimization of Supersonic Wings with Four Objectives Using Evolutionary Algorithm," *Journal of Aircraft*, Vol. 39, No. 4, 2002, pp. 621-629.

[8]Sasaki, D., Yang, G., and Obayashi, S., "Automated Aerodynamic Optimization System for SST Wing-Body Configuration," Transactions of the Japan Society for Aeronautical and Space Sciences, Vol. 46, No. 154, 2004, pp. 230-237.

[9]Sasaki, D., Shahpar, S., and Obayashi, S., "Multi-Objective Optimization of Low Pressure Compression System," *Proceedings of the 24th of International Congress of the International Council of the Aeronautical Sciences* (CD-ROM), ICAS 2004-6.2.2, 2004.

[10]Oyama, A., Liou M.-S., "Multiobjective Optimization of Rocket Engine Pumps Using Evolutionary Algorithm," *AIAA Journal of Propulsion and Power*, Vol. 18, No. 3, pp. 528-535, 2003.

[11]Kanazaki, M., Obayashi, S., and Nakahashi, K., "Exhaust Manifold Design with Tapered Pipes using Divided Range MOGA," *Engineering Optimization*, Vol. 36, No. 2, pp. 149-163, 2004.

[12]Chiba, K., Obayashi, S., Nakahashi, K., and Morino, H., "Multidisciplinary Design Optimization of Wing Shape for Regional Jet," *Proceedings of 4th International Symposium on Advanced Fluid Information*, 2004.

[13]Mäkinen, R. A. E., Périaux, J., and Toivanen, J., "Multidisciplinary Shape Optimization in Aerodynamics and Electromagnetics using Genetic Algorithms," *International Journal for Numerical Methods in Fluids*, Vol. 30, 1999, pp. 149-159.

[14]Poloni, C., Pediroda, V., and Bucchieri, L, "Multi Objective Optimization of Turbine Blades by TASCFlow and FRONTIER on a Linux Cluster," *CFX USERS MEETING*, 2000.

[15]Padovan, L., Pediroda, V., and Poloni, C., "Multi Objective Robust Design Optimization of Airfoils in Transonic Fields (M.O.R.D.O.)," *Proceedings of Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems* (CD-ROM), CIMNE, 2003.

[16]Arakawa, M., and Hagiwara, I., "Development of Adaptive Real Range (ARRange) Genetic Algorithms," *JSME International Journal*, Series C, Vol. 41, No. 4, 1998, pp. 969-977.

[17]Arakawa, M., and Hagiwara, I., "Nonlinear Integer, Discrete and Continuous Optimization Using Adaptive Range Genetic Algorithms," *Proceedings of 1997 ASME Design Engineering Technical Conferences*, 1997.

[18]Oyama, A., Obayashi, S., and Nakamura, T., "Real-Coded Adaptive Range Genetic Algorithm Applied to Transonic Wing Optimization," *Applied Soft Computing*, Vol. 1, No. 3, 2001, pp. 179-187.

[19]*DOT USERS MANUAL*, Vanderplaats, R&W, Inc., Colorado Springs, CO, 1999.

[20]Yuret, D., and de la Maza, M., "Dynamic Hill Climbing: Overcoming the Limitations of Optimization Techniques," *Proceedings of the Second Turkish Symposium on Artificial Intelligence and Neural Networks*, 1993, pp. 208-212.

[21]Shahpar, S., "SOFT: A New Design and Optimization Tool for Turbomachinery," *Proceedings of Evolutionary Methods for Design, Optimization and Control*, CIMNE, 2002.

[22]Bäck, T., Fogel, D. B., and Michalewicz, Z, *Handbook of Evolutionary Computation*, IOP Publishing Ltd. and Oxford Univ. Press, 1997.

[23]Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, Inc., Reading, MA, 1989.

[24]Baker, J. E., "Reducing Bias and Inefficiency in the Selection Algorithm," *Proceedings of the Second International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1987, pp.14-21.

[25]Yaochu, J., Okabe, T., and Sendhoff, B., "Adapting Weighted Aggregation for Multiobjective Evolution Strategies," *Proceedings of Evolutionary Multi-Criterion Optimization 2001*, Lecture Notes in Computer Science 1993, 2001, pp. 96-110.

[26]Fonseca, C. M., and Fleming, P. J., "An Overview of Evolutionary Algorithms in Multi-Objective Optimization," *Evolutionary Computation Journal*, Vol. 3, No. 1, 1995, pp. 1-16.

[27]Kursawe, F., "A Variant of Evolution Strategies for Vector Optimization," *Proceedings of Parallel Problem Solving from Nature I (PPSN-I)*, 1990, pp. 193-197.