Chapter 2
Real-Coded Adaptive Range Genetic Algorithm

2.1. Introduction

Finding a global optimum in the continuous domain is challenging for Genetic Algorithms (GAs). Traditional GAs use the binary representation that evenly discretizes a real design space. Although such binary-coded GAs have been successfully applied to a wide range of design optimization problems, they suffer from disadvantages, when applied to the real-world problems involving a large number of real design variables. Since binary substrings representing each parameter with the desired precision are concatenated to represent an individual, the resulting string encoding a large number of design variables would wind up a huge string length. For example, for 100 variables with a precision of six digits, the string length is about 2000. GAs would perform poorly for such design problems. Previous applications have been kept away from this problem by sacrificing precision or narrowing down the search regions prior to the optimization. However, such approaches might exclude the region that actually has the global optimum.

Another drawback of the binary-coded GAs applied to parameter optimization problems in continuous domains comes from discrepancy between the binary representation space and the actual problem space. For example, two points close to each other in the representation space might be far in the binary represented problem space. It is still an open question to construct an efficient crossover operator that suits to such a modified problem space.

A simple solution to these problems is the use of the floating-point representation of parameters. In these real-coded GAs, an individual is coded as a vector of real numbers corresponding to the design variables. The real-coded GAs are robust, accurate, and efficient because the floating-point representation is conceptually closest to the real design space, and moreover, the string length reduces to the number of design variables. However, even the real-coded GAs would lead to premature convergence when applied to aerodynamic shape designs with a large number of design variables.

A more sophisticated approach is to dynamically alter the coarseness of the search space referred to as dynamic coding. In [1], Krishnakumar *et al.* presented Stochastic Genetic Algorithms (Stochastic GAs) to efficiently solve problems with a large number of real design parameters. The key features of Stochastic GAs are:

1. Each binary number represents a region of the real space instead of a single point to maintaining good precision with the small string length.
2. Those regions adapt during the optimization process according to the 1/5th success rule as Evolutionary Strategies (ES) to improve efficiency and robustness.

The Stochastic GAs have been successfully applied to Integrated Flight Propulsion Controller designs [1] and air combat tactics optimization [2]. As they explained, the Stochastic GAs bridge the gap between ES and GAs to handle large design problems.

Adaptive Range Genetic Algorithms (ARGAs) are a new approach using dynamic coding proposed by Arakawa and Hagiwara [3] for binary-coded GAs to treat continuous design space. The essence of their idea is to adapt the population toward promising design regions during the optimization process, which enables efficient and robust search in good precision while keeping the string length small. Moreover, ARGAs eliminate prior definition of boundaries of the search regions since ARGAs distribute design candidates according to the normal distributions of the design variables in the present population. In [4], ARGAs have been applied to pressure vessel designs and outperformed other optimization algorithms.

The objective of this chapter is to develop robust and efficient GAs applicable to aerodynamic shape designs. To achieve this goal, the idea of the dynamic coding is incorporated with the used of the floating-point representation. Since the ideas of the Stochastic GAs and the use of the floating-point representation are incompatible, ARGAs for the floating-point representation are developed. The real-coded ARGAs are expected to possess both advantages of the binary-coded ARGAs and the floating-point representation to overcome the problems of having a large search space that requires continuous sampling. First, to display advantages of the present approach, the proposed approach is applied to a test function optimization problem. Then, an aerodynamic airfoil shape optimization is demonstrated to ensure the feasibility of the proposed approach in aerodynamic design problems.

2.2. Adaptive Range Genetic Algorithms

2.2.1 ARGAs for Binary Representation

When conventional binary-coded GAs are applied to real-number optimization problems, discrete values of real design variables p_i are given by evenly discretizing prior-defined search regions for each design variable [$p_{i,min}$, $p_{i,max}$] according to the length of the binary substring $b_{i,l}$ as

$$p_i = (p_{i,\max} - p_{i,\min})\frac{c_i}{2^{sl} - 1} + p_{i,\min} \qquad (2.1)$$

where

$$c_i = \sum_{l=1}^{sl} (b_{i,l} \cdot 2^{l-1}) \qquad (2.2)$$

In binary-coded ARGAs, decoding rules for the offspring are given by the following normal distributions:

$$\begin{aligned} N'(m_i, s_i^{\,2})(p_i) &= \sqrt{2p}\,s_i \cdot N(m_i, s_i^{\,2})(p_i) \\ &= \exp(-\frac{(p_i - m_i)^2}{2s_i^{\,2}}) \end{aligned} \qquad (2.3)$$

where the average m_i and the standard deviation s_i of each design variable are determined by the population statistics. Those values are recomputed in every generation. Then, mapping from a binary string into a real number is given so that the region between N'_{UB} and N'_{LB} in Fig. 2.1 is divided into equal size regions according to the binary bit size as

$$p_i = \begin{cases} m_i - \sqrt{-2s_i^{\,2} \cdot \ln(N'_{LB} + (N'_{UB} - N'_{LB})\frac{c_i}{2^{sl-1} - 1})} & \text{for} \quad c_i \le 2^{sl-1} - 1 \\ m_i + \sqrt{-2s_i^{\,2} \cdot \ln(N'_{UB} - (N'_{UB} - N'_{LB})\frac{c_i - 2^{sl-1}}{2^{sl-1} - 1})} & \text{for} \quad c_i \ge 2^{sl-1} \end{cases} \qquad (2.4)$$

where N'_{UB} and N'_{LB} are additional system parameters defined in [0,1]. In the ARGAs, genes of design candidates represent relative locations in the updated range of the design space. Therefore, the offspring are supposed to represent likely a range of an optimal value of design variables.

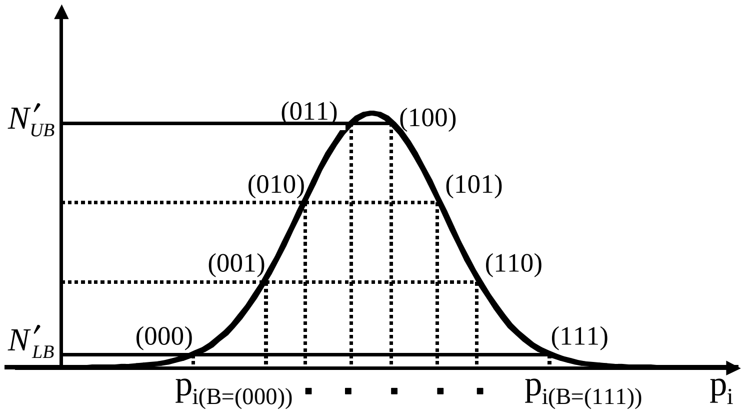

Fig 2.1 Decoding for binary-coded ARGAs

Although the original ARGAs have been successfully applied to real parameter optimizations, there is still room for improvements. The first one is how to select the system parameters N'_{UB} and N'_{LB} on which robustness and efficiency of ARGAs largely depend. The second one is the use of constant intervals even near the center of the normal distributions. Third one is that since genes represent relative locations, the offsprings become constantly away from the centers of the normal distributions when the distributions are updated. Therefore, the actual population statistics does not coincide with the updated population statistics.

2.2.2 ARGAs for Floating-Point Representation

In real-coded GAs, real values of design variable are directly coded as a real string r_i:

$$p_i = r_i \qquad (2.5)$$

where

$$p_{i,\min} \le r_i \le p_{i,\max}$$

Or, sometimes normalized values of the design variables are used as:

$$p_i = (p_{i,\max} - p_{i,\min}) \cdot r_i + p_{i,\min} \qquad (2.6)$$

where

$$0 \le r_i \le 1$$

To employ the floating-point representation for ARGAs, the real-valued design variable p_i is rewritten here by a real number r_i defined in [0,1] so that integral of the probability distribution of the normal distribution from $-\infty$ to pn_i is equal to r_i as:

$$p_i = s_i \cdot pn_i + m_i \qquad (2.7)$$

$$r_i = \int_{-\infty}^{pn_i} N(0,1)(z)dz \qquad (2.8)$$

where the average m_i and the standard deviation s_i of each design variable are calculated by the top half of the present population. Schematic view of this coding is illustrated in Fig. 2.2. It should be noted that the real-coded ARGAs resolve drawbacks of the original ARGAs; no need for selecting N'_{UB} and N'_{LB} as well as arbitrary resolution near the average. To prevent inconsistency between the actual and updated population statistics, the present ARGAs update μ and σ every N generations and then the population is reinitialized. Flowchart of the resulting ARGA is shown in Fig.

2.3.

To improve robustness of the present ARGAs further, relaxation factors w_m and w_s are introduced to update the average and standard deviation as

$$m_{new} = m_{present} + w_m (m_{sampling} - m_{present}) \tag{2.9}$$

$$s_{new} = s_{present} + w_s (s_{sampling} - s_{present}) \tag{2.10}$$

where w lower than 1 contributes to improve robustness of the ARGAs. $m_{sampling}$ and $s_{sampling}$ are determined by sampling the top half of the present population.

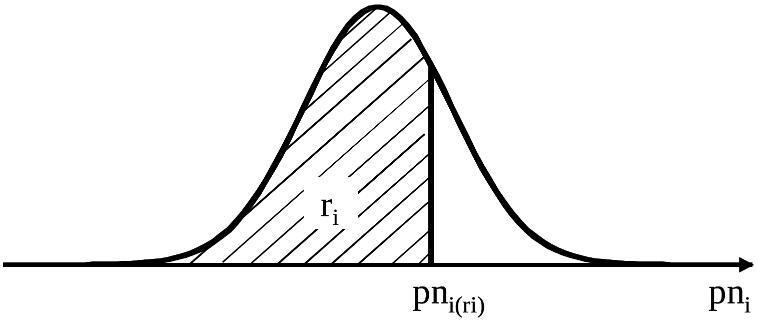

Fig. 2.2 Decoding for real-coded ARGAs

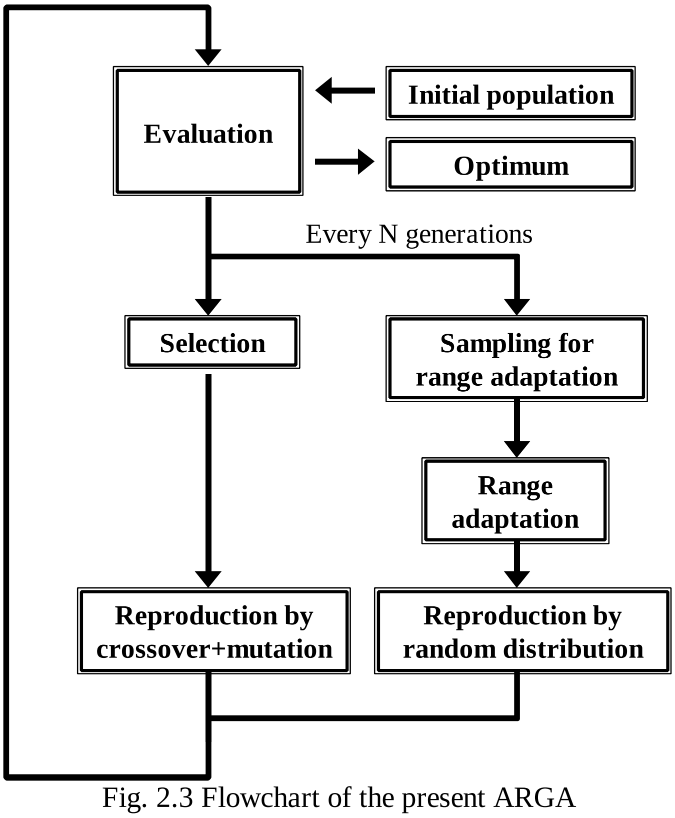
Evaluation
Initial population
Optimum
Every N generations
Selection
Sampling for range adaptation
Range adaptation
Reproduction by crossover+mutation
Reproduction by random distribution

Fig. 2.3 Flowchart of the present ARGA

2.3. Results

In this section, a real-coded ARGA is applied to two test problems: (1) a test function optimization problem, (2) an aerodynamic airfoil shape optimization. In both calculations, the present ARGA uses non-overlapping system coupled with elitist strategy where the best and the second best individuals are copied into the next generation, parental selection by SUS coupled with ranking using Michalewicz's nonlinear function, one-point crossover, uniform mutation that takes place at a probability of 0.1 and then adds a random disturbances to the design variable in the amount up to ±10% of the domain. User-defined parameters of the real-coded ARGA, w_m, w_s and N, are set to 1, 0.5 and 4, respectively. These parameters are determined by a parametric study using a simple test function not shown here. Unbiased initial population is generated by randomly spreading solutions over the entire design space in consideration.

2.3.1 Test Function Minimization

Prior to attack the aerodynamic shape optimization, the real-coded ARGAs were applied to a test function optimization problem. The test function in [5] is a dynamic control as

$$\min(x_N{}^2 + \sum_{k=0}^{N-1}(x_k{}^2 + u_k{}^2)) \qquad (2.11)$$

subject to

$$x_{k+1} = x_k + u_k \ , \ k = 0,1,\ldots,N-1,$$

where, x_k is a state, and $\vec{\boldsymbol{u}} = (\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{N-1})$ is the control vector. The search domain is [-200,200] for each u_k. The initial state x_0 and the size of the control vector are given by 100 and 45, respectively. Here, 500 generations were allowed with a population size of 200. Five trials were run for each GA changing seeds for random numbers to give different initial populations. Figure 2.4 compares the optimization histories for both GAs. The bold line indicates the analytically obtained optimum value. While all the trials using the conventional GA lead to premature convergence, the ARGA find the global optimum at every trial.

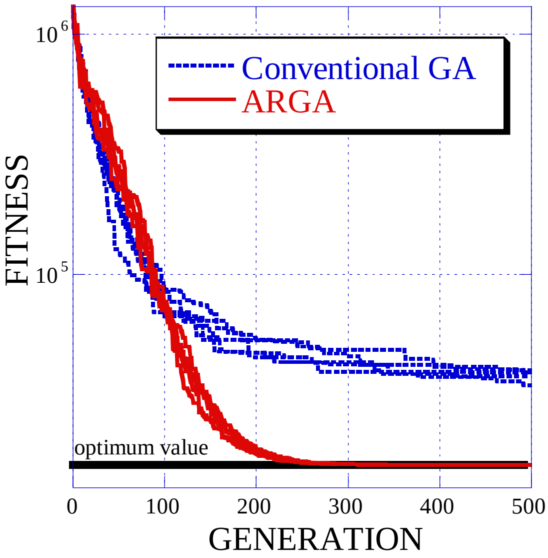

Fig. 2.4 Optimization histories for the test function optimization problem

2.3.2 Aerodynamic Airfoil Shape Optimization

To demonstrate performance of the real-coded ARGAs in comparison with the conventional real-coded GAs, aerodynamic airfoil shape optimizations were carried out. The objective function was the lift-to-drag ratio to be maximized where the free stream Mach number and the angle of attack were set to 0.8 and 2 degrees, respectively. The airfoil thickness was constrained so that the maximum thickness was greater than 12% of the chord length. The aerodynamic performance of each design was evaluated by the Navier-Stokes solver described in section 4.4. PARSEC airfoil (see subsection 3.2.1.5) was used with the trailing-edge thickness and its ordinate (at *X*=1) frozen to 0.

Figure 2.5 compares optimization histories of three trials using the real-coded ARGA and the conventional real-coded GA. Both of the population size and the number of generations were 100. The present ARGA outperformed the conventional GA starting from all initial populations.

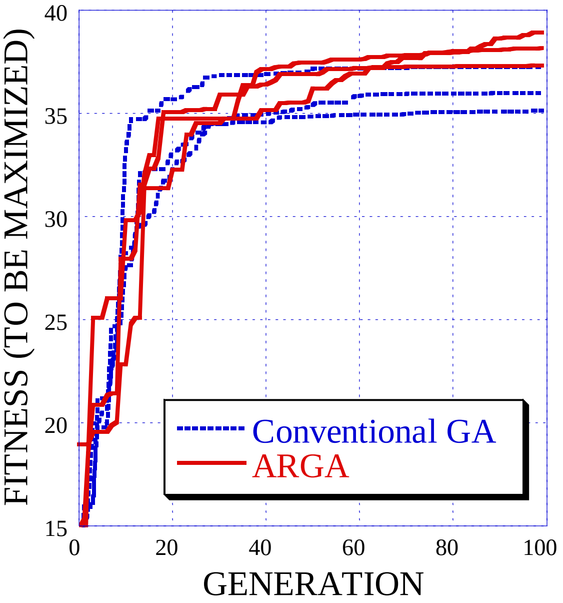

Fig. 2.5 Optimization histories for the aerodynamic airfoil shape optimization

Figure 2.6 compares the best airfoil shapes designed by the conventional GA and ARGA and the corresponding pressure coefficient distributions. The surface pressure distribution of the design optimized by ARGA is almost identical to that of NASA supercritical airfoils. This indicates the feasibility of the present ARGA in aerodynamic designs.

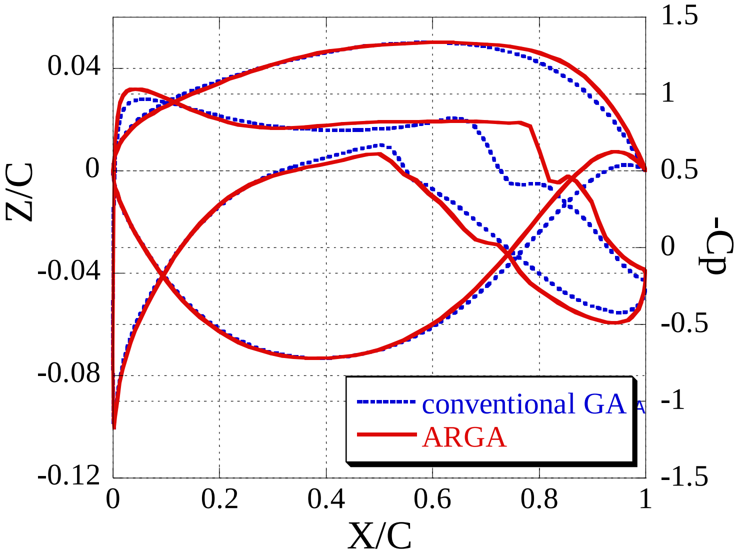

Fig. 2.6 Designed airfoil shape and the corresponding pressure distribution

2.4. Summary

To develop robust and efficient EAs applicable to aerodynamic shape designs, the real-coded ARGAs have been developed by incorporating the idea of the binary-coded ARGAs with the use of the floating-point representation. The resulting real-coded ARGAs are expected to possess both advantages of the binary-coded ARGAs and the use of the floating-point representation to overcome the problems of having a large search space that requires continuous sampling. First, the efficiency and the robustness of the proposed approach have been demonstrated by using a typical test function. Then the proposed approach has been applied to an aerodynamic airfoil shape optimization problem. The results confirm that the real-coded ARGAs consistently find better solutions than the conventional real-coded GAs do. The design result is considered to be the global optimal and thus ensures the feasibility of the real-coded ARGAs in aerodynamic designs.

References

[1] Krishnakumar, K., Swaminathan, R., Garg, S. and Narayanaswamy, S., “Solving Large Parameter Optimization Problems Using Genetic Algorithms,” Proceedings of the Guidance, Navigation, and Control Conference, Baltimore, MD, 1995.

[2] Mulgund, S., Harper, K., Krishnakumar, K. and Zacharias. G., “Air Combat Tactics Optimization Using Stochastic Genetic Algorithms,” Proceedings of 1998 IEEE International Conference on Systems, Man, and Cybernetics, San Diego, CA, Oct. 1998, pp.3136-3141.

[3] 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.

[4] Arakawa, M. and Hagiwara, H., “Nonlinear Integer, Discrete and Continuous Optimization Using Adaptive Range Genetic Algorithms,” in CD-ROM Proceedings of the 34th Design Automation Conference, Anaheim, California, Jun.1997.

[5] Janikow, C. Z. and Michalewicz, Z., “An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms,” *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1991, pp.31-36.