AIAA JOURNAL
Vol. 34, No. 5, May 1996

Genetic Optimization of Target Pressure Distributions for Inverse Design Methods

Shigeru Obayashi*
Tohoku University, Sendai 980-77, Japan
and
Susumu Takanashi†
National Aerospace Laboratory, Chofu 182, Japan

A genetic algorithm has been applied to optimize target pressure distributions for inverse design methods. Pressure distributions around airfoils are parameterized by B-spline polygons, and the airfoil drag is minimized under constraints on lift, airfoil thickness, and other design principles. Once target pressure distributions are obtained, corresponding airfoil/wing geometries can be computed by an inverse design code coupled with a Navier–Stokes solver. Successful design results were obtained for transonic cases with and without a shock wave.

Nomenclature

C_D, C_d	= drag coefficients for wing and airfoil, respectively
C_{dv}	= viscous drag coefficient
C_{dw}	= wave drag coefficient
C_L, C_l	= lift coefficients for wing and airfoil, respectively
C_p	= pressure coefficient
$C_{p,l}$, $C_{p,u}$	= pressure coefficients of lower and upper surfaces of airfoil, respectively
C_p^*	= pressure coefficient corresponding to local Mach number of 1.0
c	= airfoil chord length
M	= Mach number
M_∞	= freestream Mach number
$r, r1, r2, r3$	= uniform random variables in the interval [0, 1]
t	= airfoil thickness
U_e	= potential velocity
U_∞	= freestream velocity
x, z	= Cartesian coordinates in the chordwise and normal directions taking the origin at the leading edge
y	= coordinate representing pressure coefficient

Introduction

As an engineering tool, computational fluid dynamics (CFD) is expected to enhance the computational design environment. One area in which CFD might have significant impact on realistic applications is development of computational design methods.[1] The design efficiency will further be improved by applying numerical optimization.

In Ref. 2, aerodynamic numerical optimization methods are categorized into two classes: direct and inverse numerical optimization methods. The direct numerical optimization methods are formed by coupling aerodynamic analysis methods with numerical optimization algorithms. They minimize (or maximize) a given aerodynamic object function by iterating directly on the geometry. The geometry is represented by a general function, such as polynomial and cubic splines, by a linear combination of known airfoils, or by a basic shape plus a combination of typical geometry perturbations. Such procedures, however, become extremely expensive as the number of geometry parameters is increased. Unfortunately, flowfields are often very sensitive to the geometry, and thus those procedures do not seem practical even with the aid of the current supercomputers.

The inverse numerical optimization methods deal with pressure distributions rather than the geometry, to minimize, for example, drag under given lift and pitching moment. Once the target pressure distributions are optimized, a corresponding geometry can be determined by the inverse methods. This process will be repeated until the resulting geometry satisfies specific geometry constraints and certain off-design performances.

The inverse methods themselves form a class of powerful design tools. These methods solve the classical inverse problem of determining the aerodynamic shape that will produce given pressure distributions. However, they leave the user with the problem of translating the design goals into properly defined pressure distributions exhibiting the required aerodynamic characteristics.[1] Although skillful designers are capable of producing successful designs, the design efficiency can be improved by providing the designer with tools for target pressure specification. For this purpose, numerical optimization of target pressure distributions has been studied in Refs. 3 and 4. Those inverse numerical optimization methods avoid most of the limitations of the inverse methods while requiring considerably less computational effort than the direct numerical optimization approach.[2]

Whether the approach is direct or inverse, selection of the optimization algorithm is also important. Among the numerical optimization algorithms, gradient-based methods have been used widely. The optimum obtained from these methods will be a global optimum if the objective and constraints are differentiable and convex.[5] In practice, however, it is very difficult to prove these properties. One could only hope for a local optimum in a neighbor of the initial point, provided the gradient is well defined. Therefore, one must start the design from various initial points to see if one can obtain a consistent optimum and therefore have reasonable assurance that this is the true optimum. In this sense, the gradient-based methods are not robust. For example, Ref. 4 pointed out that the aerodynamic design problem was strongly nonlinear and exhibited discontinuous derivatives for the object as well as constraint functions. Robustness of the numerical optimization algorithm is a key factor for the development of aerodynamic optimization methods.

Genetic algorithms[6] (GAs) are known to be robust optimization algorithms and have been enjoying increasing popularity in the field of numerical optimization in recent years. GAs are search algorithms based on the mechanics of natural selection and natural genetics. One of the key features of GAs is that they search from a population of points, not a single point. In addition, they use objective function

Received May 1, 1995; presented as Paper 95-1649 at the AIAA 12th Computational Fluid Dynamics Conference, San Diego, CA, June 19–22, 1995; revision received Nov. 27, 1995; accepted for publication Dec. 5, 1995.

*Associate Professor, Department of Aeronautics and Space Engineering. Senior Member AIAA.

†Chief, Computational Aerodynamics Section (deceased). Senior Member AIAA.

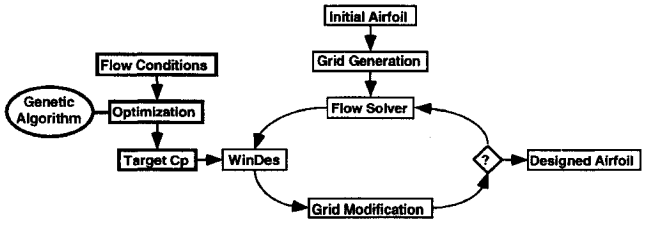

Fig. 1 Flowchart of the present design procedure.

information (fitness value), not derivatives or other auxiliary knowledge. These features make GAs attractive to practical engineering applications.

GAs have been applied to aeronautical problems in several ways, including parametric and conceptual design of aircraft,[7,8] preliminary design of turbines,[9] topological design of nonplanar wings,[10] and aerodynamic optimization using CFD.[11,12] Applicability to CFD problems, however, was limited because they were the direct numerical optimization methods. The direct approach requires CFD evaluation of each member of the population at every generation in GAs. As a result, it requires a tremendous amount of computational time. The inverse approach will therefore alleviate the computational time for engineering purposes.

In this paper, a GA is applied to an inverse numerical optimization method coupled with a Navier–Stokes (NS) solver for the airfoil/wing shape design. In the present GA, chordwise target pressure distributions are optimized for minimum drag without any CFD evaluation. Once optimized distributions are found, a corresponding airfoil shape is obtained by the inverse design code, WinDes.[13] WinDes is an iterative inverse solver that requires flowfield evaluation at each iteration. Since the code requires only a small number of iterations, the two-dimensional NS solver, LANS2D,[14] is coupled with WinDes for the airfoil design cases. The overall procedure is presented in the flowchart in Fig. 1. Three-dimensional results will be shown for a wing design based on the straight isobar pattern that is obtained from the same chordwise sectional pressure distributions at any spanwise station.

Approach

Coding

GAs simulate evolution by selection. Design candidates are considered as individuals in the population. An individual is characterized by genes represented as a string of parameters. In this paper, an individual is a pressure distribution. Therefore, a coding scheme is required to specify a pressure distribution in terms of a string of parameters.

One of the parameterization techniques recommended for ab initio designs is B-spline parameterization.[15] The B-spline curve can be constructed so that the first and last points coincide with those of the defining polygon. Thus, pressure distributions are split into two curves, corresponding to pressure distributions on the upper and lower surfaces of an airfoil, respectively. In this research, seven points are used to define a B-spline polygon (Fig. 2), specifying the stagnation pressure $C_p = 1$ at $x = 0$ (since the inverse method does not solve the stagnation point exactly) and the trailing-edge pressure $C_p = C_{p,te}$ (by user specification) at $x = 1$ (assuming the chord length unity).

Among 14 points to define upper and lower C_p curves, 10 points are free to move. The x and y ($y = C_p$) coordinates of those points are the design variables, and thus we have 20 design variables in total. Although standard GAs are characterized by the use of the binary coding for design variables, a real (decimal) number is used for simplicity.

Initial population is generated randomly in the region of $0 < x < 1$ and $-1.5 < y < 1$. Besides the leading and trailing edges, one more point of the seven-point B-spline polygon is initially confined on the y axis to describe a steep C_p drop near the leading-edge region, that is, to obtain a large leading-edge radius typical for a supercritical airfoil.[16] To create each individual, the lower C_p curve is first generated based on constraints mentioned later. Then, the upper C_p curve is generated so as to satisfy the rest of constraints

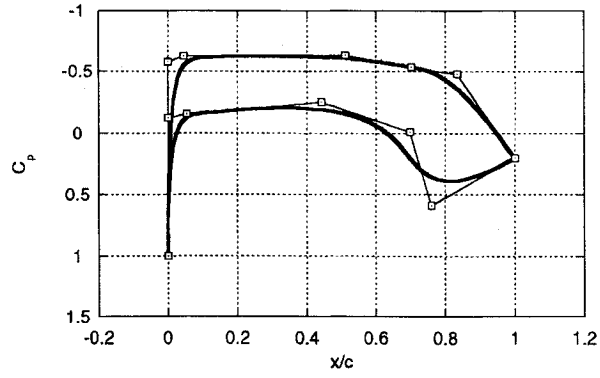

Fig. 2 B-spline polygons and corresponding pressure distributions: ——, target pressures and —⊡—, B-spline polygon.

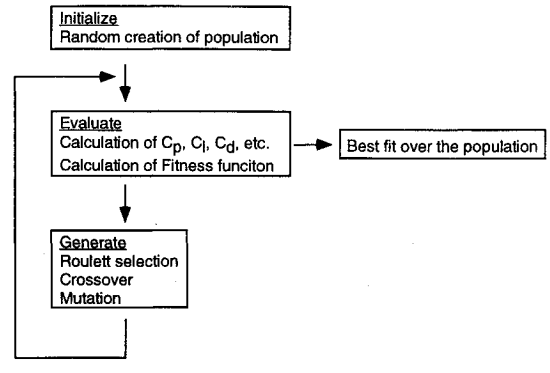

Fig. 3 Flowchart of the present GA.

approximately. If the resulting lift differs from the specified lift less than 10%, the corresponding string of parameters is assigned to an individual as genes. If it differs more than 10%, the string of parameters is discarded and regenerated. This process is repeated until 100 individuals are created.

GA

At each generation (iteration) of GA, the fitness value (object function value) of every individual is evaluated and used to specify its probability of reproduction. A new population is generated from selected parents by performing specific operators on their genes. These operators are briefly explained in the following.

A simple GA is composed of three operators: 1) reproduction, 2) crossover, and 3) mutation.[6] Reproduction is a process in which individual strings are copied according to their fitness values. This implies that strings with a higher value have a higher probability of contributing one or more offspring in the next generation. A typical reproduction operator is the roulette wheel method described in Ref. 6. The reproduction process produces a mating pool as a result. Then crossover proceeds in two steps. First, members in the mating pool are mated at random. Second, each pair of strings undergoes partial exchange of their strings at a random crossing site. This results in a pair of strings of a new generation. Mutation is a bit change of a string that will occur during the crossover process at a given mutation rate. Mutation implies a random walk through the string space, and it plays a secondary role in the simple GA. A flowchart of the present GA is illustrated in Fig. 3.

A simple crossover operator for real number strings is the average crossover[17] that would compute the arithmetic average of two real numbers provided by the mated pair. In this paper, a weighted average is used as

$$\begin{aligned} \text{Child1} &= r * \text{Parent1} + (1 - r) * \text{Parent2} \\ \text{Child2} &= (1 - r) * \text{Parent1} + r * \text{Parent2} \end{aligned} \tag{1}$$

where Child1,2 and Parent1,2 denote 20 design variables of the children (members of the new population) and parents (a mated pair of the old generation), respectively. The random number r is regenerated for every design variable. Because of Eq. (1), the number of the initial population is assumed to be even.

Mutation takes place at a probability of 10% (when a random number satisfies $r1 < 0.1$). Equations (1) will then be replaced by

$$\text{Child1} = r * \text{Parent1} + (1 - r) * \text{Parent2} + \frac{r2 - 0.5}{5}$$
$$\text{Child2} = (1 - r) * \text{Parent1} + r * \text{Parent2} + \frac{r3 - 0.5}{5} \quad (2)$$

Fitness Evaluation: Objective and Constraints

In this paper, we define the optimization problem as follows:
Minimize

$$C_d$$

subject to
1) C_l = specified
2) Airfoil thickness t = specified
3) $C_{p,l} < 0$ at $0.1 < x < 0.6$ and

$$\int_{0.1}^{0.6} |C_{p,l}|\,dx \geq 0.1$$

4) $\max_{0.6 < x < 1} C_{p,l} < 0.4$
5) $C_p|_{\text{suction peak}} \leq C_p^*$
6)

$$\left(\frac{dC_{p,u}}{dx}\right)\Bigg|_{0.1 < x < 0.5} \cong 0$$

7)

$$\frac{dC_{p,u}}{dx} \leq 2.5$$

8) Number of inflection points < 2

To specify airfoil thickness t approximately, a formula is taken from Ref. 4 as

$$t = -\frac{\sqrt{1 - M_\infty^2}}{2} \int_0^1 \frac{C_{p,u} + C_{p,l}}{2}\,dx \quad (3)$$

The constraints 5 and 6 aim to reproduce sonic plateau pressure distributions described in Ref. 16. The constraint 7 was taken from the observation of C_p plots in Ref. 16 to avoid a separation of the boundary layer.

Drag is calculated as a sum of viscous and wave drag. When the flow is attached, the profile drag can be calculated from a knowledge of the potential-flow pressure distributions and location of transition from laminar to turbulent flow. Locations of transition are left for user specification. When no shock wave is present, wave drag is ignored.

The Squire–Young relation is an empirical relation between drag based on the momentum thickness of the boundary layer and the potential velocity.[18] The momentum thickness can be estimated from an integral equation of the turbulent boundary layer. The viscous drag is estimated as

$$C_{dv} = 2\theta_{te}\left(\frac{U_{e,te}}{U_\infty}\right)^{3.2} \quad (4)$$

where θ_{te} is the momentum thickness at the trailing edge and $U_{e,te}$ is the potential velocity at the trailing edge. (See Ref. 18 for more detail.)

All of the constraints were combined into a single fitness function with certain penalties. The exact form of the fitness function will be described in the Appendix. The optimization took about 3 min on an SGI Indy workstation using 1000 successive generations with 100 individuals in the population. It is far less than the computational time necessary for the inverse design as mentioned later.

Treatment of Shock Wave

Shock wave is automatically generated in the upper C_p curve at the smallest x coordinate satisfying $x > 0.5$, $M > 1.1$, and $dC_p/dx > 0.8$. A typical pressure jump across the shock wave on supercritical airfoils becomes about 70% of the Rankine–Hugoniot jump.[4] In this paper, the pressure jump is initially set to 60% as

$$\Delta C_p \equiv C_{p_{ds}} - C_{p_{us}} = 0.6\left(C_{p_{us}} + \frac{1}{0.7M_\infty^2}\right)\frac{7\left(M_{us}^2 - 1\right)}{6} \quad (5)$$

where ds and us denote downstream and upstream of the shock wave, respectively, and the ratio of specific heats is assumed to be 1.4. After the shock wave, the B-spline polygon is modified by adding ΔC_p to each y coordinate of the vertices except the first and last points.

A pressure plateau behind the shock wave is also necessary to stabilize the boundary layer.[16] Thus after computing a C_p curve with Eq. (5), a backward substitution is taken as

$$C_{p_{ds}} \equiv C_p(m_{\text{shock}} + 1) = C_p(m_{\text{shock}} + 2) \quad (6)$$

where m is the integer index of the B-spline curve (7 vertices provide 4 B-spline portions, and each portion is calculated by 10 parameters; in total, 40 points are used to represent a B-spline curve from the leading edge to the trailing edge) and m_{shock} is the index at the shock wave. Because of this substitution, ΔC_p finally becomes roughly 70% of the Rankine–Hugoniot jump. Wave drag caused by a shock wave is estimated as

$$C_{dw} = K\left(M_{us}^2 - 1\right)^{4.4} \quad (7)$$

where a constant $K = 0.02$.[19]

When a shock wave is generated, the constraints 5 and 6 are modified as

$$5' \cdot C_p|_{\text{suction peak}} \leq 2.2C_p^*$$

and between suction peak and shock wave

$$6' \cdot \frac{dC_{p,u}}{dx} > 0$$

The modified constraints aim to reproduce generalized design pressure distributions, as described in Ref. 16.

Inverse Design Cycle

Once the present GA finds optimum target pressure distributions, a corresponding airfoil geometry can be obtained by an inverse design method. Here the inverse design code WinDes is used.[13] The code can solve both two- and three-dimensional problems.

WinDes uses the following iterative procedure. Suppose the initial geometry and surface pressure distributions obtained from any CFD analysis code are given. First, pressure differences are calculated from the given initial and target pressure distributions. From these pressure differences, corresponding geometry corrections can be computed from the integral equations discretized at the panels on the geometry. An improved geometry is then obtained from the initial geometry and the computed geometry corrections. Finally, the CFD code is used again to check how close the resulting pressure distributions are to the target distributions. If the differences are still large, the process will be iterated. In practice, 10–20 iterations are sufficient to obtain the final geometry.

The advantage of this method is that the required analysis code is arbitrary and any type of analysis method, even experiment, can be used. In this paper, two- and three-dimensional NS codes, LANS2D[14] and LANS3D,[20] were used. The latest version of these codes uses the third-order upwind in the right-hand side.

In the present inverse design method, grid generation around the modified geometry is required at every iteration. To automate the inverse design loop, a grid generator has to be robust and efficient. An algebraic grid generation code described in Ref. 14 is used in this paper because of its robustness and efficiency. The two-dimensional C-type mesh contains 131×51 grid points in the chordwise and normal (to the surface) directions, respectively. In the three-dimensional case, the C-H topology is used, applying the two-dimensional grid generation at each spanwise section. The three-dimensional grid contains 30 sections in the spanwise direction.

Use of WinDes, LANS2D/3D, and the algebraic grid generator constructs an automated loop for the inverse design with reasonable computational requirements. These codes were implemented on a Cray C90/161024 supercomputer at the Institute of Fluid Science, Tohoku University. The inverse design for 10 cycles required about 30 min on the C90 using single processor.

Results

Case 1—Airfoil Design: $C_l = 0.5$

In the first test case, the flow condition was set to the freestream Mach number of 0.75 and the Reynolds number of 10×10^6. In the NS computation, the Baldwin–Lomax turbulence model[21] was used. An angle of attack was set to zero. Locations of transition were fixed at 5 and 10% chord for upper and lower surfaces, respectively. The lift was specified as 0.5 and the trailing-edge pressure coefficient was set to 0.15. Figure 4 shows the optimization history of the present GA. The optimum was obtained after about 450 generations.

Figure 5 shows the target pressure distributions obtained from the present GA, the designed geometry obtained from the inverse method, and the corresponding pressure distributions obtained from the NS computation. The optimized pressure distributions have a sonic plateau to avoid a shock wave on the upper surface of the airfoil and a rear loading region typical for the supercritical airfoils. In the inverse design cycle, the initial geometry was chosen as the NACA0012 airfoil. In total, nine iterations were required to obtain the final geometry.

In Fig. 5, the attached table shows a comparison of the aerodynamic performances among the target (indicated as GA, estimated), the inverse design result (indicated as NS, designed) and a modified supercritical airfoil (indicated as NS, SC(2)-0610′). The original SC(2)-0610 airfoil is taken from Ref. 16. Although it has a blunt trailing edge, the geometry is modified to have a sharp trailing edge for a comparison. The drag obtained from the target optimization was underestimated due to the simplified calculation of the viscous drag. Comparing the NS results, the present design shows higher lift-to-drag L/D ratio than the SC(2)-0610′ airfoil.

Case 2—Airfoil Design: $C_l = 0.7$

In the second test case, the freestream Mach number was increased to 0.8 and the trailing-edge pressure coefficient was reduced to 0.1. The lift was specified as 0.7. Other conditions were the same as those in case 1.

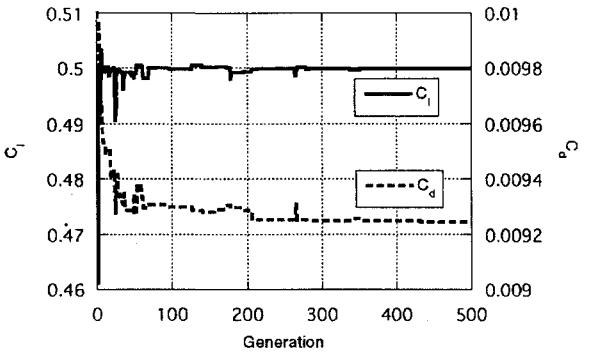

Fig. 4 Optimization history of lift and drag of the best fit in generation.

	GA, estimated	NS, designed	NS, SC(2)-0610′
C_l	0.500	0.502	0.446
C_d	9.29E-3	1.17E-2	1.17E-2

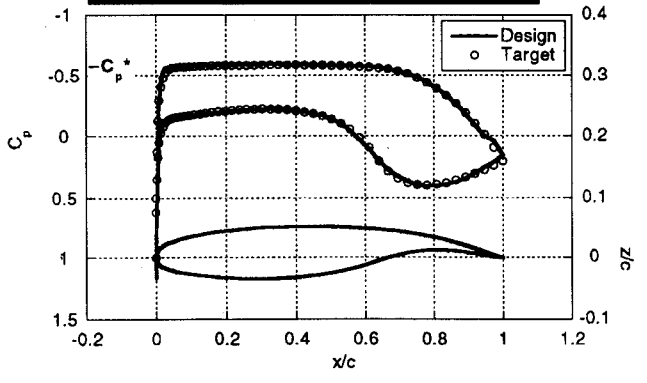

Fig. 5 Design result and comparison of aerodynamic performance for case 1.

Figure 6 shows the target pressure distributions obtained from the present GA and the design result obtained from the inverse method. Because of the higher lift specified, the design result contains a shock wave. Because of the presence of the shock wave, the inverse design required 17 iterations. Although there is a minor discrepancy near the leading edge, target and design pressure distributions agree well, including the shock jump. The resulting airfoil has a typical supercritical airfoil shape. The L/D ratio reaches 43.3.

The present optimization problem has an apparent discontinuity when a shock wave appears. This design result indicates the robustness of the GA since the optimum is sought among pressure distributions with and without a shock wave. On the other hand, it will be difficult to apply the conventional gradient-based approach to this kind of problem.

As reported in Ref. 16, use of the blunt trailing edge improves the aerodynamic performance. NS analysis of such flowfields will be studied to obtain better airfoil design.

Case 3—Wing Design

By using the optimized pressure distributions in case 1, the inverse design of a transonic transport wing was performed. A simple design technique to have the straight isobar pattern is used to give spanwise pressure distributions.

Figure 7 shows wing planform and computed pressure distributions for the initial and designed wing geometries. The wing has a given planform with a leading-edge sweep of 20.4 deg, an aspect ratio of 8.22, and a trailing-edge kink at the 36% semispan section. Although the NS grid had 22 points on the wing surface in the spanwise direction, the inverse design was performed only at 6 sections in the spanwise direction: the root, 18, 36, 54, 72, and 90% semispan sections. For the NS grid, the modification of wing geometry was

	GA, estimated	NS, designed
C_l	0.700	0.644
C_d	1.10E-2	1.49E-2

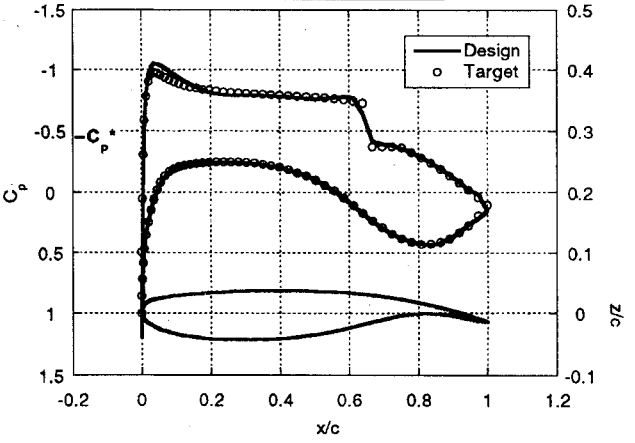

Fig. 6 Design result and comparison of aerodynamic performance for case 2.

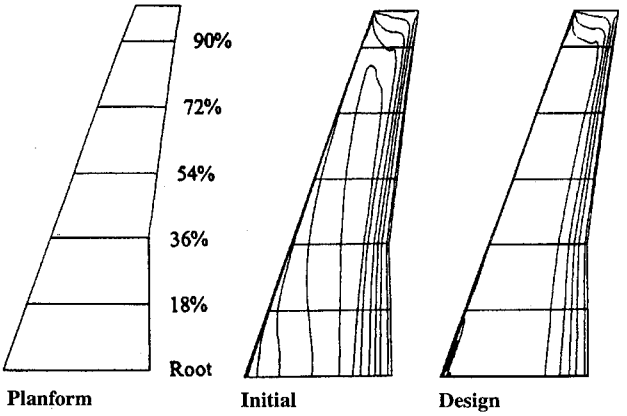

Fig. 7 Wing planform and pressure distributions.

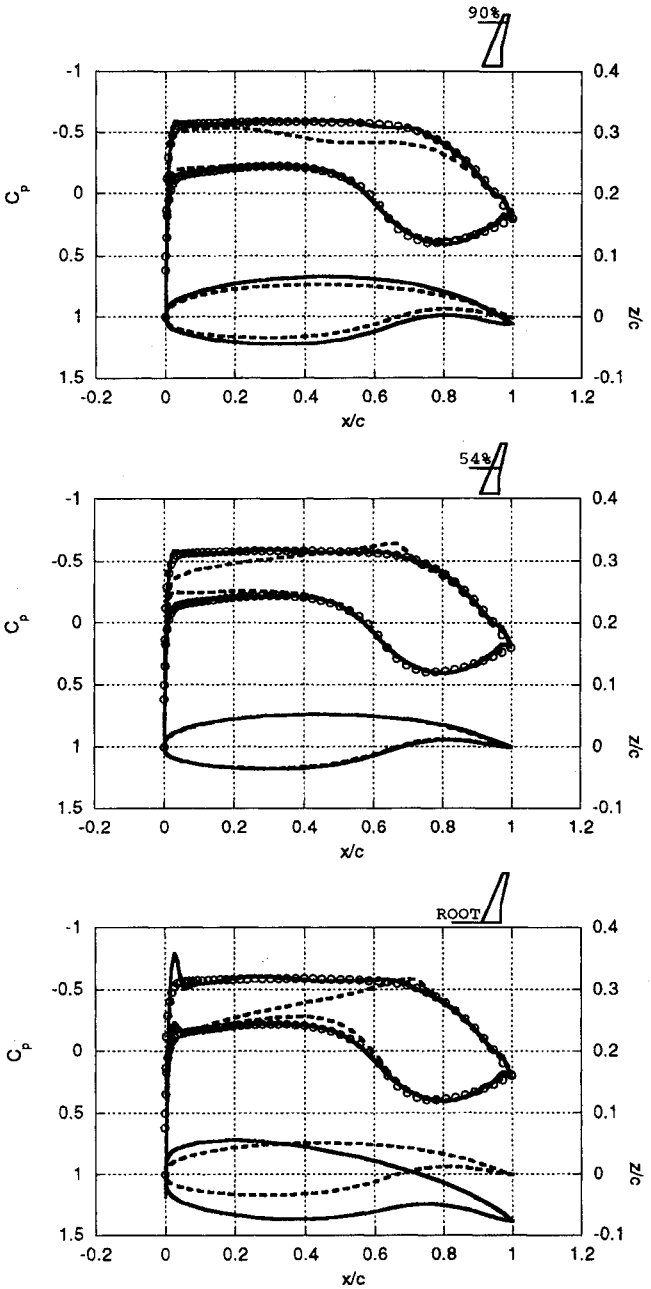

Fig. 8 Initial and designed wing geometries and corresponding pressure distributions: ——, design; - - - -, initial; and ○, target.

linearly interpolated between those sections. In the tip region, the same airfoil section was used outside of the 90% section, whereas the wing twist was linearly extrapolated. The tip region is usually designed by other means and thus the optimization of this region is not considered here.

The initial wing geometry was constructed by simply giving the same two-dimensional airfoil section designed in case 1. The freestream Mach number was set to 0.8 in this wing case because of the leading-edge sweep. The Reynolds number based on the root chord was set to 10×10^6. The resulting pressure contours do not show the isobar pattern at all as shown in the middle of Fig. 7. The initial wing has $C_L = 0.385$ and $C_D = 0.0157$.

After 15 iterations in the inverse design, the straight isobar pattern was materialized as shown in the right of Fig. 7. A minor oscillation can be seen at the leading edge near the root section and will be discussed later. The resulting wing has $C_L = 0.492$ and $C_D = 0.0212$.

Figure 8 shows comparisons of the airfoil geometries and chordwise pressure distributions of the initial and designed wings at the root, 54 and 90% semispan sections. The spanwise variation of the initial pressure distributions is apparent when the same airfoil shape is given over the span. In contrast, there is no spanwise variation in the pressures on the designed wing except at the leading edge of the root section.

On the other hand, the airfoil geometries of the designed wing vary drastically in the spanwise direction. The root section has a thicker airfoil and a high angle of attack. Toward the midspan, the thickness is reduced and the wing is twisted down. The airfoil geometry at the 54% section is similar to the two-dimensional design.

In particular, the geometric change between the root and 18% sections is very large. This probably causes the pressure oscillation near the leading edge since we do not have enough grid points to represent a large geometric change. Grid refinement may be necessary to overcome the oscillation in this region.

Compared with the computational time required for the NS solver, the computational time required for the inverse design and grid generation codes is negligible. In the present case, the total computational time was about 15 h using a single processor of a Cray C90 supercomputer. By using multiple processors for the NS solver, we can probably reduce the design time to only a few hours.

Concluding Remarks

An efficient and robust optimization technique has been developed to specify target pressure distributions for the inverse design method. A GA is applied to minimize the airfoil drag under constraints on lift, airfoil thickness, and other design principles. Since GA is robust, it allows discontinuities in the objective as well as constraints. Thus, a weak shock wave is automatically introduced, if necessary, for the specified lift and flow conditions. The present optimization is so efficient that it can be used as preprocessing for the inverse design on a standard workstation.

Once target pressure distributions are obtained, corresponding airfoil/wing geometries can be computed by an inverse design code coupled with a NS solver. Successful design results were obtained for transonic cases with and without a shock wave. Three-dimensional wing design was demonstrated by specifying the straight isobar pattern in the spanwise direction.

Appendix: Construction of Fitness Function

GAs require the definition of an objective to be maximized. An inverse of the drag coefficient is taken as the objective here. Then all of the constraints are required to be combined with the objective. There are eight constraints in the fitness evaluation section. The constraints 1 and 2 are equality constraints and the constraints 3–8 are inequality constraints. The equality constraints are multiplied to the objective as an exponential function to reject the individuals that do not satisfy the specified lift and airfoil thickness. The inequality constraints are expressed so as to increase their values when violated, and the inverse of their sum is added to the objective with penalty.

The final form of the objective is given as

$$\text{Fitness} = \left(\frac{0.4}{C_d^2} + \frac{2 \times 10^7}{IC^2}\right) \cdot \exp(-100 \times EC)$$

where IC and EC denote the sum of inequality constraints and equality constraints, respectively. For IC, the constraints 3–8 are represented as

$$IC = 10{,}000 \times \left[\min\left(\int_{0.1}^{0.6} |C_p| dx, 0.1\right) - 0.1\right]^2$$

$$+ 5 \times \left[\max_{0.6 < x < 1} (C_{p,l}, 0.4) - 0.4\right]^2$$

$$+ 3 \times \left[\max\left(\frac{C_p|_{\text{suction peak}}}{C_p^*}, 1.0\right) - 1.0\right]^2$$

$$+ 40{,}000 \times \left[1.1 - \min\left(\frac{\text{slope} + |\text{slope}|}{2}, 1.1\right)\right]$$

$$+ 0.001 \times \left[\max\left(\frac{dC_{p,u}}{dx}, 2.5\right) - 2.5\right]^2$$

$$+ 0.01 \times \exp[\text{number of inflection points}]$$

where

$$\text{slope} = \frac{C_{p,u}|_{x=0.5} - C_{p,u}|_{x=0.1}}{0.4} + 1$$

For EC the constraints 1 and 2 are represented as

$$EC = \max\left(|t_{\text{specified}} - t_{\text{calculated}}|, 10^{-4}\right) + \max\left(|C_{l\,\text{specified}} - C_{l\,\text{calculated}}|, 10^{-4}\right) - 0.01$$

where the differences below 10^{-4} are ignored for the optimization.

Acknowledgments

The first author's work was partially supported by Sumitomo Foundation Grant 94-103-361. The authors would like to thank Kisa Matsushima of Fujitsu Limited, Chiba, Japan, for her help in running the two-dimensional case.

References

[1]Labrujére, T. E., and Slooff, J. W., "Computational Methods for the Aerodynamic Design of Aircraft Components," *Annual Review of Fluid Mechanics*, Vol. 25, 1993, pp. 183–214.

[2]van den Dam, R. F., van Egmond, J. A., and Slooff, J. W., "Optimization of Target Pressure Distributions," Special Course on Inverse Methods for Airfoil Design for Aeronautical and Turbomachinery Applications, AGARD Rept. 780, Ref. 3, Nov. 1990.

[3]van den Dam, R. F., "Constrained Spanload Optimization for Minimum Drag of Multi-Lifting Surface Configurations," Computational Methods for Aerodynamic Design (Inverse) and Optimization, AGARD CP 463, Ref. 16, March 1990.

[4]van Egmond, J. A., "Numerical Optimization of Target Pressure Distributions for Subsonic and Transonic Airfoil Design," Computational Methods for Aerodynamic Design (Inverse) and Optimization, AGARD CP 463, Ref. 17, March 1990.

[5]Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design: With Applications*, McGraw–Hill, New York, 1984, pp. 12–19.

[6]Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison–Wesley, Reading, MA, 1989.

[7]Bramlette, M. F., and Cusic, R., "A Comparative Evaluation of Search Methods Applied to the Parametric Design of Aircraft," *Proceedings of the 3rd International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, 1989, pp. 213–218.

[8]Crispin, Y., "Aircraft Conceptual Optimization Using Simulated Evolution," AIAA Paper 94-0092, Jan. 1994.

[9]Powell, D. J., Tong, S. S., and Skolnick, M. M., "EnGENEous Domain Independent, Machine Learning for Design Optimization," *Proceedings of the 3rd International Conference of Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, 1989, pp. 151–159.

[10]Gage, P., and Kroo, I., "A Role for Genetic Algorithms in a Preliminary Design Environment," AIAA Paper 93-3933, Aug. 1993.

[11]Gregg, R. D., and Misegades, K. P., "Transonic Wing Optimization Using Evolution Theory," AIAA Paper 87-0520, Jan. 1987.

[12]Quagliarella, D., and Cioppa, A. D., "Genetic Algorithms Applied to the Aerodynamic Design of Transonic Airfoils," AIAA Paper 94-1896, June 1994.

[13]Takanashi, S., "Iterative Three-Dimensional Transonic Wing Design Using Integral Equations," *Journal of Aircraft*, Vol. 22, No. 8, 1985, pp. 655–660.

[14]Matsushima, K., Obayashi, S., and Fujii, K., "Navier–Stokes Computations of Transonic Flow Using the LU-ADI Method," AIAA Paper 87-0421, Jan. 1987.

[15]Rogers, D. F., and Adams, J. A., *Mathematical Elements for Computer Graphics*, 2nd ed., McGraw–Hill, New York, 1990.

[16]Harris, C. D., "NASA Supercritical Airfoils—A Matrix of Family-Related Airfoils," NASA TP-2969, March 1990.

[17]Davis, L., *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1990.

[18]Young, A. D., *Boundary Layers*, AIAA Education Series, AIAA, Washington, DC, 1989.

[19]Inger, G. R., "Application of Oswatitsch's Theorem to Supercritical Airfoil Drag Calculation," *Journal of Aircraft*, Vol. 30, No. 3, 1993, pp. 415, 416.

[20]Fujii, K., and Obayashi, S., "Navier–Stokes Simulations of Transonic Flows over a Practical Wing Configuration," *AIAA Journal*, Vol. 25, No. 3, 1987, pp. 369, 370.

[21]Baldwin, B. S., and Lomax, H., "Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, Jan. 1978.