

Assessment of Inverse and Direct Methods for Airfoil and Wing Design

Mengmeng Zhang and Arthur William Rizzi

Abstract The goal of aerodynamic design for airfoils and wings is to improve the performance of the lifting surfaces, e.g., by minimizing the drag. We consider here two approaches, the classical inverse design approach that finds the surface which produces desired pressure distributions, and the direct mathematical optimization based on local parameter searches, that is usually enabled by fast gradient computation, for example, by the adjoint method. The hybrid approach is to combine both of them. Each approach has its own pros and cons. In this chapter the approaches are assessed by application to the design of transonic RAE2822 airfoil and ONERA M6 wing.

Keywords Inverse design • Gradient-based optimization • Parametrization • Drag reduction • Wing surface curvature • Adjoint solver

MSC code: 76N25 (Flow control and optimization)

1 Introduction

Aircraft design activities are concerned with the determination of designs that meet a priori specified performance features of the vehicle. The specified design objectives are traditionally met through an iterative process of analyses, evaluations, and modifications of the design. In this sequential trial-and-error procedure, the designer must rely on experience, intuition, and ingenuity for every re-design, and this makes aircraft design an exciting creative discipline. In practice, however, designers are often forced to depend on tried concepts to cut a path through an incomprehensible number of feasible designs, historically characterizing the process as a slow gradual improvement of existing types of concepts.

The task of designing an aircraft is among the most complex in engineering. The complexity can be simplified by sequential decision that divides the design

M. Zhang (✉) • A.W. Rizzi
Royal Institute of Technology (KTH), 10044 Stockholm, Sweden
e-mail: mzha@kth.se; rizzi@kth.se

into the conceptual design, preliminary design, and the detailed design. While the greatest freedom to exploit potential trade-offs between aircraft subsystems for the optimization of the design occurs earliest in the design stage, since the decisions taken during the earlier stage commit up to 80% of the life-cycle costs, although, of course, the actual costs incurred appear on the books much later. Mistakes here must be avoided because they are very costly to remedy later and delay acceptance. Matters involving the interaction of aerodynamics with structures and controls are particularly prone to errors due to the low fidelity of the analysis methods traditionally used. If these complicated problems are not resolved in an integrated sense, the sub-optimal design might be led to in a global sense and become “myopic.”

1.1 MDO and Aerodynamic Design Approaches

Multidisciplinary Design Optimization, or MDO [1, 2], combines analysis and optimizations in several individual disciplines with those of the entire system concurrently through formal mathematical processes. It puts into place a formal integrated system design process for better product quality by effectively exploiting the synergism of interdisciplinary couplings. MDO, as a discipline, itself comprises of many areas of research. It is “a methodology for design of complex engineering systems that are governed by mutually interacting physical phenomena and made up of distinct interacting subsystems (suitable for systems for which) in their design, everything influences everything else” [1].

One of the significant factors holding back the widespread adoption of MDO is its computational cost when the number of design variables becomes very large (*the curse of dimensionality*). The use of high-fidelity models can raise the cost from merely expensive to unbearable. Parallel computing helps, but cannot overcome computational inefficiency. The revolution in computing speed and memory capacity of digital computers together with persistent systematization of design methodology has led to tools for computational aircraft design (e.g., MDO) that aim at automation of the conventional design process through integration of numerical methods for analysis, sensitivity analysis and mathematical programming so that the best design in terms of a pre-defined criterion can be determined. Traditionally the process of selecting design variations has been carried out by trial-and-error, relying on the intuition and experience of the designer, the engineer in the loop. Increasing the level of automation by computational means has reduced, but not eliminated, the engineer-in-the-loop activities. The overall success of the design process depends heavily not only on reliability and accuracy of the computational methods but also on how well the designer has set his goals.

1.1.1 Aerodynamic Design: Three Approaches from a User Perspective

Optimization of aircraft wings is not new. The thing that makes wings so hard to design is that their aerodynamics and structure are not just interdependent, they are

variable. Computational aerodynamic design, one of the disciplinary subsets of the aircraft design process, aims directly at determining the geometrical shape of the aircraft hull that produces certain specified aerodynamic properties, with or without constraints on the geometry. Usually termed aerodynamic shape optimization (ASO) [3–5], this is the subject of this chapter. ASO is a very attractive technology because it replaces workable designs with optimal ones, and cuts down design times, thus enabling faster responses to the economic pressure of the marketplace.

For wing design there are generally three approaches: the direct optimization design (mathematics-skill dependent) [6], the inverse design (engineering-skill dependent) [5, 7–9], and the hybrid approach which combines both of them. The direct approach requires the user to define the cost function (usually the drag) along with the constraints, and then seeks the solution to the constrained optimization problem by mathematical algorithms for non-linear optimization. When the algorithm involves gradient searches, the sensitivities that indicate how to change the geometry in order to reduce the cost function can be computed also for very many design parameters by solution of an adjoint to the flow problem. This approach is the most popular way of doing optimization nowadays as the computer capacities are continuously increasing. However, it is always trapped in a local optimum due to the limitation of the algorithms, and it may overexploit the flow localities. The second approach works by first finding a well-posed pressure distribution that fulfills the design requirements and then determining a geometry that yields this target pressure. One big issue of this approach is that it needs to formulate a good “target” pressure distribution first, which requires engineer in the loop. The last method can employ both approaches under user control, but very manual. One example is LINDOP optimizer [10, 11] in the MSES [12] package.

2 Introduction to Direct Optimization, Inverse Design, and Hybrid Approach

2.1 Direct Mathematical Optimization

A straightforward way to search for an optimal design is to construct a non-linear constrained optimization problem,

$$\begin{aligned}
 & \min : I = I(w, X) \\
 & \text{subject to :} \\
 & C_L(w, X) \geq C_L^0, \\
 & C_m(w, X) = C_m^0, \\
 & g_j(X_\Gamma) \leq 0, \quad 1 \leq j \leq m,
 \end{aligned} \tag{1}$$

where X is the mesh, X_Γ is the surface of the geometry, w is the flow-field variables, and g_j are the geometric constrains. The cost function I is selected by the designer, which might be the drag coefficient $I = C_D$, the drag to lift ratio $I = \frac{C_D}{C_L}$, or the pressure difference $I = \int (p - p_d)^2 d\Omega$ if an inverse design problem is being posed. Numerous optimization algorithms [6] are available for attempted solution of the mathematical problem. We have the mesh generation algorithms

$$M(X, X_\Gamma) = 0 \quad (2)$$

and the surface parametrization algorithm S

$$S(X_\Gamma, \ell) = 0 \quad (3)$$

where ℓ are the design variables which determine the surface X_Γ .

The change in I can be estimated by a small variation $\delta\ell$ to the parameter vector and recalculating the flow to obtain the change in I , thus approximating the directional derivative,

$$I(\ell + \delta\ell) = I(\ell) + \frac{dI}{d\ell} \cdot \delta\ell + O(\|\delta\ell\|^2). \quad (4)$$

Most optimization algorithms employ a line search along search direction d ,

$$\ell^{n+1} = \ell^n - \lambda d \quad (5)$$

with λ a step size parameter. The search direction d is composed of gradients $\frac{dI}{d\ell}$; quasi-Newton methods also compute approximations to the Hessian matrix of second derivatives, $H_{ij} = \frac{d^2 I}{d\ell_i d\ell_j}$, by differences of gradients in an updating scheme. When the parameter space is high-dimensional, this approach using the gradient itself entails high computational cost.

2.1.1 Gradients by Adjoint Equations

The adjoint method was originally applied to aerodynamics by Jameson [13] adapting ideas originally formulated by Lions [14] on optimal control of systems governed by partial differential equations. The adjoint equations can be conveniently formulated in a framework to calculate the sensitivity of a given objective function I to parameters ℓ which control the geometry. The derivation is easy when R , etc. below are interpreted as the finite dimensional discretization of the flow equations, objective functions, etc. The residual R of the governing equations for a given flight state(s) which expresses the dependence of flow variables w on the mesh X is:

$$R(w, X) = 0 \quad (6)$$

Thus a small change in X produces a small change δI to the cost function,

$$\delta I = \frac{\partial I}{\partial w} \delta w + \frac{\partial I}{\partial X} \delta X \quad (7)$$

and a small change δw to the flow w ,

$$\delta R = \frac{\partial R}{\partial w} \delta w + \frac{\partial R}{\partial X} \delta X = 0. \quad (8)$$

The mesh deformation δX is calculated from the corresponding displacements of the nodes that define the surface of the geometry X_Γ by parametrization \mathbf{S} .

Equation (8) is multiplied by a Lagrange multiplier vector Ψ , subtracted from Eq. (7) and the result re-arranged,

$$\delta I = \left(\frac{\partial I}{\partial w} - \Psi^T \frac{\partial R}{\partial w} \right) \delta w + \left(\frac{\partial I}{\partial X} - \Psi^T \frac{\partial R}{\partial X} \right) \delta X. \quad (9)$$

Choosing Ψ so that the first term on the right vanishes gives

$$\left[\frac{\partial R}{\partial w} \right]^T \Psi = \left(\frac{\partial I}{\partial w} \right)^T. \quad (10)$$

This is a linear PDE known as the adjoint equation. There are two main ways to characterize the adjoint approach, as a discrete method, in which the discretized governing equations are used to derive the adjoint equations, and as a continuous method, in which the adjoint equations are derived from the analytical PDEs [15]. The discrete and continuous approaches are found to have relative advantages and disadvantages over each other [16]. The discrete adjoint equations derived directly from the discrete flow equations become very complicated when the flow equations are discretized with higher order schemes using flux limiters. On the other hand it can provide an exact gradient of the inexact cost function which results from the discretization of the flow equations. In theory a discrete method can handle PDEs of arbitrary complexity without significant mathematical development and can treat arbitrary functionals I . In comparison, the continuous adjoint requires significant theoretical development but is better connected to the underlying physics and can be solved by a method independent of the flow solution scheme. However, it is more limited in the types of functionals and governing equations that can be treated, and the gradient calculated will differ more from that found by finite differencing. But as the mesh is refined, all three gradients, discrete, continuous, and finite difference, converge to the same limit.

A few words are needed here to explain why we can consider this (the discretization of) a linear PDE known as the adjoint to the flow equations: we see no derivatives operating on Ψ . The key here is the scalar product: for the continuous PDE formulation, an integral over the domain. The trick is to perform suitable

integrations by part in the integral to move derivatives from the primary variables (the flow variables) to the dual—the Lagrange parameters. Notice also that it is the linearized flow equation that appears. This implies that the adjoint of the Euler flow equations is very similar to the linearized Euler equations—they are almost self-adjoint, which in turn implies that the adjoint equation can be solved by much the same procedures as the primal (flow equations).

The total perturbation δI now depends only on the change of the mesh δX , but is independent of the flow solution perturbation δw . Unlike the gradient calculation by finite difference, for each optimization step, the gradient of I with respect to an arbitrary number of design variables (usually a large amount) can be determined without the need for additional flow-field evaluations. To solve the adjoint equation (10), it costs approximately as much as a flow solution. Note, however, that the boundary conditions in the adjoint PDE are usually chosen to eliminate boundary integral contributions rather than efficient expulsion of waves through the boundaries and this may hamper convergence of the numerical solution. Finite difference methods can also be used to find these sensitivities but are in general significantly more expensive, requiring at least one additional flow solution per parameter.

Examples shown here of direct optimization design are computed by the SU2 [15] software suite from Stanford University: an open-source, integrated analysis and design tool for solving complex, multi-disciplinary problems on unstructured computational grids. The built-in optimizer is a Sequential Least Squares Programming (SLSQP) algorithm [6] from the *SciPy Python* scientific library. The gradient is calculated by continuous adjoint equations of the flow governing equations [15, 17]. SU2 is in continued development. Most examples pertain to inviscid flow but also RANS flow models with the Spalart–Allmaras and the Menter SST $k-\omega$ turbulence models can be treated.

2.2 Inverse Design

Inverse design is a classical way of designing airfoils and wings, which was popular several decades ago before the advent of high performance computing as a tool in aircraft design.¹ The method consists of *predictor* and *corrector* processes which require engineering know-how at the very beginning of the design stage. The predictor/corrector design approach systematically modifies a given geometry based on direct solutions for the flow around the airfoils or wings. The calculated pressure distribution is compared with a prescribed target distribution and the resulting differences are used by a geometry “corrector” module to modify the current geometry to a shape more likely to generate the desired pressure. The corrector module may be an optimization procedure such as LINDOP [10–12] described

¹This section is adapted from [5, 18].

below, or a design algorithm that directly relates pressure changes to geometry changes. Examples of the latter type include Barger and Brooks [9] methods for designing super-critical airfoils. The method was developed and coupled with several two- and three-dimensional transonic codes by Campbell [7].

Dulikravich [19] solved a similar 3D problem using a Fourier series method. Later Campbell [20] and Obayashi [21] raised some ideas on setting up the target pressure distribution and reasonable constraints for inverse design problem.

Inverse design approach has a long history but it is not out of date. Dealing with the surface curvatures is robust, and the aerodynamicist sees more physical properties of the wing. The approach was recently re-visited and improved by German Aerospace Center (DLR) [22] with good results on laminar wing design. Zhang developed the SCID toolbox with the resulting surface curvature [7, 20] inverse design method. The flow chart of SCID inverse design is shown in Fig. 1. It connects streamline curvatures on the wing surface with pressure changes to iteratively modify an initial shape. It is combined with under-relaxation chosen

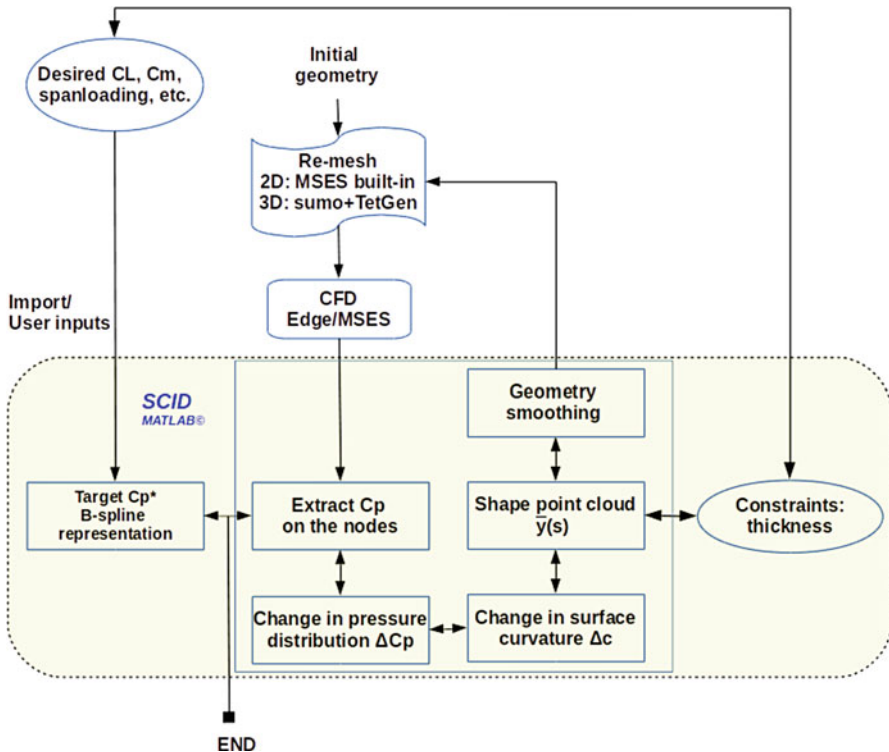


Fig. 1 Flow chart for wing inverse design using SCID algorithm, retrieved from [18]

to help convergence, and smoothing procedures to ensure a smooth surface and curvature. Examples shown here for inverse design are computed by SCID [18, 23]. In SCID the CFD code MSES [12] is used for airfoils and EDGE² is used for wings.

2.2.1 Pressure–Curvature Relations

The relation is derived from the normal component of the momentum equation for inviscid flow along the streamline on the wing surface as long as the flow is attached. The steady Euler equations with \mathbf{e}_s the unit vector along a streamline, so that the velocity vector is $\mathbf{u} = U\mathbf{e}_s$, $U = |\mathbf{u}|$,

$$UU_s\mathbf{e}_s + U^2\frac{d\mathbf{e}_s}{ds} + \nabla p/\rho = 0 \quad (11)$$

where s is the arc-length along the streamline. In the Darboux frame in Fig. 2, \mathbf{e}_n and \mathbf{e}_t are the surface unit normal, and the second unit normal $\mathbf{e}_s \times \mathbf{e}_n$ to the streamline, there holds

$$\frac{d^2\boldsymbol{\gamma}}{ds^2} = \frac{d\mathbf{e}_s}{ds} = c_n\mathbf{e}_n + c_g\mathbf{e}_t \quad (12)$$

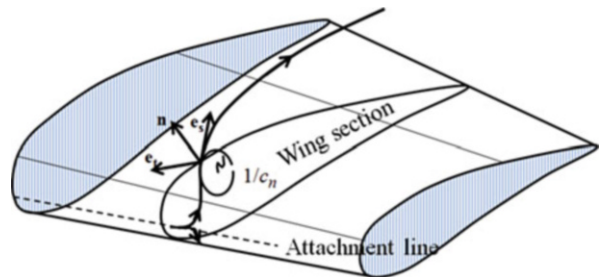
where c_n and c_g are the *normal* and *geodesic* curvatures. The normal component of the streamline Euler equation is:

$$0 + \rho U^2 c_n + \frac{\partial p}{\partial n} = 0 \quad (13)$$

from which we can derive a relation between the curvature and the pressure coefficient,

$$c_n + \frac{C_p}{2L(1 - C_p)} = 0$$

Fig. 2 Wing represented in the Darboux frame



²www.foi.se/en/Customer--Partners/.../Edge1/Edge/.

where L is a length scale for the pressure gradient $L \approx \frac{p-p_\infty}{\partial p/\partial n}$. Since L is unknown and varies along the streamline, we introduce an inverse length scale coefficient F . The relation used in the shape modification step is the proportionality between changes in normal curvature and pressure coefficient, for unit chord-length, so curvature becomes non-dimensional,

$$c_n = F \cdot C_p = \frac{d^2 \boldsymbol{\gamma}}{ds^2} \cdot \mathbf{e}_n - F \cdot C_p. \quad (14)$$

The F -coefficient was proposed by Barger and Brooks [9]. Campbell [7] suggests the c_n -dependence $F = A(1 + c_n^2)^B$ for perturbations ΔC_n , ΔC_p , to produce

$$\Delta c_n = A(1 + c_n^2)^B \Delta C_p \quad (15)$$

where A and B are adjustable constants.

There remains to relate the surface normal curvature change to geometry change itself. The surface analogue of the Frenet–Serret formulas for the surface coordinates $\boldsymbol{\gamma}(s)$ along the streamline is

$$\boldsymbol{\gamma}_{ss} = c_n \mathbf{e}_n + c_g \mathbf{e}_t. \quad (16)$$

For small change on the surface, it gives

$$\Delta \boldsymbol{\gamma}_{ss} = \Delta c_n \mathbf{e}_n + c_n \Delta \mathbf{e}_n + \Delta c_g \mathbf{e}_t + c_g \Delta \mathbf{e}_t \quad (17)$$

where only the first term on the right-hand side is kept. Note that the last two terms vanish for airfoils. Since only displacement normal to the surface will change the surface, it makes sense to so restrict the geometry change, say

$$\Delta \boldsymbol{\gamma} = h(s) \mathbf{e}_n$$

and then the normal projection of Eq. (17) gives precisely

$$h_{ss} = \Delta c_n.$$

In the shape modification step A is chosen as large as possible without creating divergence in the iteration. Reported values range from 0 to 0.5. Smaller values give slow convergence, larger values may cause divergence. The correct coefficients must be chosen as compromise between speed of convergence and robustness. An adjustment algorithm is applied to select A and B according to the status of convergence.

2.2.2 Shape Modification of Streamline Sections

The following applies to airfoils but is also used in the wing design, with the assumption of surface streamlines not deviating much from the surface traces of the sections used to build the wing. With Δc_n from Eq. (15), the new shape $\mathbf{y}(s)$ is computed from the two-point boundary value problem

$$\frac{d^2 \Delta \mathbf{y}}{ds^2} = \text{coeff} \Delta C_p \mathbf{e}_n, \quad \Delta \mathbf{y}(s_{0, \text{lower}}) = \Delta \mathbf{y}(s_{\text{max}, \text{upper}}) = 0. \quad (18)$$

The arc-length s starts from the trailing edge on the lower surface. The boundary conditions are applied to ensure a sharp and closed trailing edge. The section geometry is represented by point clouds $\mathbf{y}^i = (x, z), i = 1, 2, \dots, N, N$ is the number of total points of the airfoil.

2.3 Hybrid Design

This chapter describes a hybrid scheme which combines inverse design with optimization of the aerodynamic shape based on the above as shown in Fig. 3. The ‘‘hybrid’’ means we use mathematical optimization with gradients produced by the adjoint technique or by finite differences in a loop together with inverse design that integrates the streamline curvature to produce the shape associated with the target pressure distribution to find the shape (right). One key point is that as the iteration proceeds, the engineer, with some insight from the direct optimization, can modify the current target pressure to guide the design process [7, 21, 23].

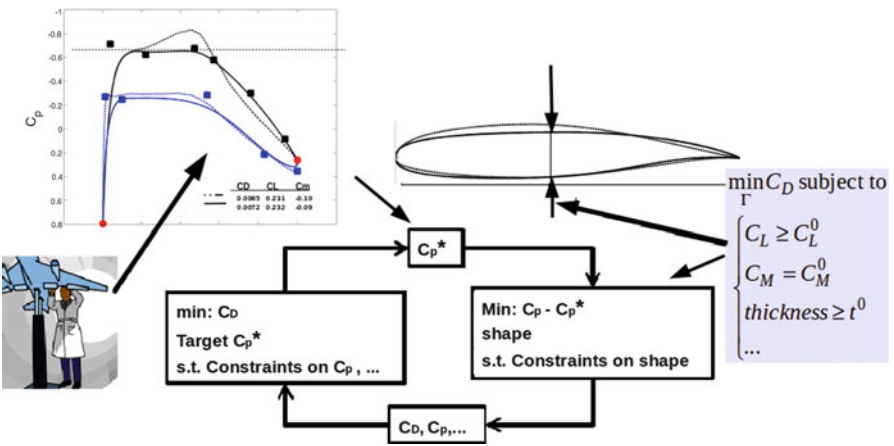


Fig. 3 The feedback design loop

The target pressure distribution on the wing planform is constructed by considering the span loads, isobar patterns, etc. [24], as well as other best practice guidelines provided by experience. Keeping the engineer in the loop emphasizes wing design rather than accurate solution of a (possibly not-so-well formulated) mathematical optimization task:

Engineer in loop to minimize drag by finding the best feasible target pressure distribution, for which a feasible shape can be found by inverse design.

LINDOP [10, 11] is an external code apart from MSES [12] which is used for airfoil optimization, it intelligently breaks down the evolution of design into reasonably small number of optimization cycles and allows the engineer in the design process during each cycle. MSES estimates the flow field by solving the Euler equations in the internal flow field coupled to thin boundary layer equations in the boundary layer [12, 25]. The overall equation system

$$R(\mathbf{w}; \alpha, AoA, M) = 0 \quad (19)$$

consists of the interior steady Euler equations, the boundary layer equations, and the necessary coupling and boundary conditions. The flow field is solved by Newton-based methods.

The optimization method used in LINDOP is also based on Newton iteration, with gradients easily available from the (exact) Jacobians employed in the flow solution. The designer can use the gradients to interactively try out various objective functions I (e.g., aerodynamic forces C_L , C_D , $\frac{C_L}{C_D}$) with respect to small perturbations in design parameters α , and flow parameters AoA and M with almost no additional cost. The Hessian matrix necessary for quasi-Newton optimization is approximated by the BFGS updating scheme [6].

There are two types of optimization problems defined in LINDOP:

- (I) Least-square problem (modal-inverse design): e.g., $I = \frac{1}{2} \int (f(s) - f_{\text{spec}}(s))^2 ds$, where $f(s)$ is usually the pressure distribution.
- (II) General optimization problem (direct design): e.g., $I = C_D$, always with some constraints, on, for example, lift, pitching moment, wing volume, or thickness, etc.

Indeed, those two problems are the most common ones among many design cases. The former one (2.3) is always solved by inverse design, if specified pressure distributions ($f_{\text{spec}}(s)$) are given. The latter one (2.3) is usually solved directly by optimization. The following chapter shows how to solve those two problems using hybrid design with MSES-LINDOP an exemplary tool. The designer is allowed to generate design-parameter changes in many ways regarding to different problem to be solved, it can be from direct keyboard inputs, or indirect posing & solving

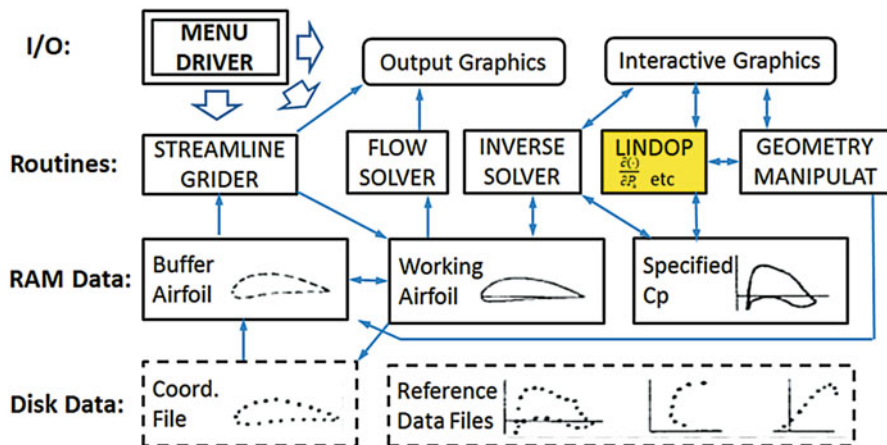


Fig. 4 LINDOP work flow, retrieved from Drela [11]

optimization problems [10]. The hybrid approach can be complicated but it is particularly useful for *complex* design problems, for example, multi-element airfoils with multi-point design [26].

Figure 4 spells out how LINDOP works. By specifying the target pressure distribution the user can interactively work through the feedback graphics where the flow solver and inverse solver are applied. The procedures require substantial user intervention. The engineer is in the loop to lead the design towards the correct direction in every small step, however it is also very manual and tedious.

3 Parametrization

There are many ways to parametrize a wing, to produce either the lofted wing surface, or the set of surface mesh points. For example the wing surface can be lofted through airfoil stacks (Fig. 5), or the geometry can be represented by modeling the perturbations of the “baseline” shape [27]. The latter technique can also perturb mesh points, or so-called mesh deformation [28]. This section shows several popular parametrization methods used in the test cases, and discusses the mesh update methods, namely, re-meshing and mesh deformation.

For the overall shape definition, mapping from surface mesh to volume mesh is usually done by “re-meshing,” i.e. re-creation of the complete grid for each shape to analyze. The generated grids will have well-formed cells, and usually mesh generation takes only a small fraction of the flow solution time. This allows loose coupling but also means that each flow solution must be done essentially from scratch since the number of flow variables is different from the previous calculation. However, if the CFD package supports interpolation between arbitrary grids it is possible to obtain a good initial guess for the flow which can speed up the solution.

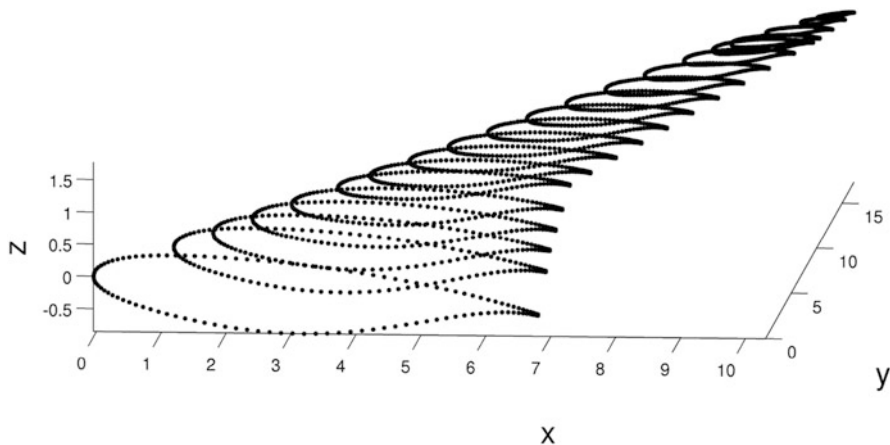


Fig. 5 Airfoil stacks

In the deformation approach, only the coordinates of grid points change, so no interpolation is necessary for initial guesses. A mesh deformation algorithm for propagating the deformation of surfaces through the whole grid is needed. The cheapest alternative is interpolation methods for arbitrary points, based on, e.g., radial basis functions or Kriging. Also PDE-based methods employing the equations of elasticity or the Laplace equations are in use. However, these techniques are most easily implemented closely coupled to the flow solver. The PDE methods provide some guard against creation of bad computational cells. But although they deal with deformations which are very small compared to wing dimensions, they can be large compared to mesh cell sizes, e.g., at a sharp trailing edge when the twist is changed.

3.1 Shape Definition

There are many ways to parameterize a wing, to produce either the lofted wing surface, or the set of surface mesh points. For example, the wing surface can be lofted through airfoil stacks, or the geometry can be represented by modeling the perturbations of a “baseline” shape [27]. The latter technique can also perturb off-surface mesh points, by so-called mesh deformation [28]. This section shows popular parametrization methods used in the test cases, and discusses the mesh update methods, namely, re-meshing and mesh deformation. Although the CAD-free parametrization techniques have been proposed [29, 30], we believe that the re-meshing technique has some advantages. Re-meshing is easy if a smooth geometry is provided. A reliable and fast meshing tool is a key. SCID uses sumo [31], a tool for rapid automatic Euler and RANS meshing. If re-meshing rather than mesh deformation is applied in finite difference approximation of derivatives,

changes to design variables cannot be too small, lest the unavoidable, random-looking, mesh changes resulting from the detailed workings of the generation algorithm hide the gradient information.

3.2 *Airfoil Stacks and Re-Meshing*

Airfoil sections are the most important building block of aerodynamic geometry. Vassberg and Jameson [3] state that: “Airfoils are used to define wings, pylons, nacelles, struts, winglets, features, horizontal stabilizers, verticals, propellers, turbomachinery blades and stators, cowls, blimps, sailboat sails, keels and ballast-bulbs, cascades, helicopter rotors, fins, chines, strakes, vertical/horizontal-axis wind turbines, flaps, frisbees, and boomerangs.” In most software systems for aircraft shape definition, the defining stations are chordwise cuts. The wing surface parametrization is decomposed into parameterization of n stations of airfoils. It is customary for the first defining station to be at the symmetry plane (wing root), and the last defining station to be at the wing’s theoretical tip. Each airfoil (defined as scaled to leading edge at the origin to trailing edge at $[1,0]$) is rotated by an incidence, translated to the defining station leading edge, then scaled to match the projected planform chord. The wing surface is usually lofted by Bézier/Bspline surfaces [32].

In SCID as well as many software systems for aircraft shape definition, the defining stations are spanwise cuts. The wing surface parametrization is decomposed into parametrization of n stations of airfoils. The geometry is updated (and smoothed) in every design cycle, then a *re-meshing* is carried out, as indicated in Fig. 1.

3.3 *Airfoil Shape Definition*

Some airfoil families are defined by a number of parameters with geometric interpretation, such as the NACA four, five, and six-digit families. But those families are of limited interest for the super-critical airfoils for transonic speeds, so more general schemes must be devised.

3.3.1 *Bézier/Bspline Curves*

Using Bézier/Bspline polynomials to parametrize the airfoil shape is simple and robust [32–34], it ensures geometrical properties including leading edge radius, trailing edge shape by solving a least-square fitting problem. It usually gives good representation of an airfoil (and smoothing) chosen by the number of control points. Melin et al. [35] developed a technique that uses four pieces of cubic Bézier

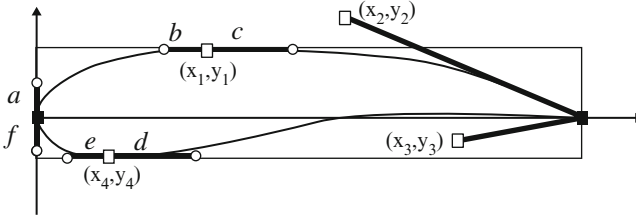
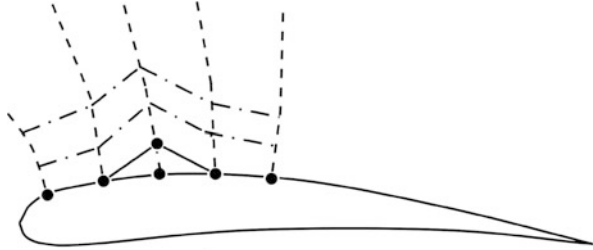


Fig. 6 Airfoil is parametrized by four pieces of cubic Bézier curves

Fig. 7 The surface mesh point at i, j is moved by $\delta X_{\Gamma_{i,j}}$



curves [33] to parametrize an airfoil within a reasonable error level. A similar parametrization is used in SCID [18, 23] for geometry update and smoothing purpose with wing represented by airfoil stacks (Fig. 6).

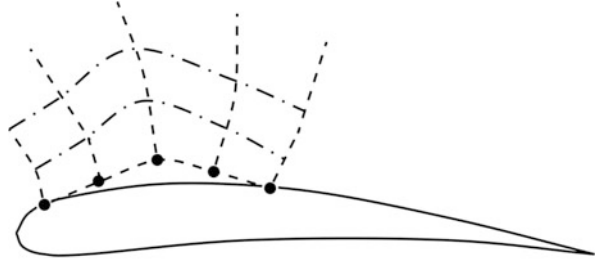
3.4 Parameterization of Shape Perturbations

3.4.1 Mesh Points and Mesh Deformation

When the mesh points are used to represent the surface, the design variables are the coordinates of the mesh points. The main advantage of parameterizing a shape with mesh points is that there is no restriction on the attainable geometry. Also, this parametrization technique can be easily implemented in any design problem. However, the use of mesh points does present some difficulties. First, the independent displacement of points may create non-smooth surfaces which are unsuitable as lifting surfaces and give the flow solver a hard time. Second, if all surface mesh points are used, the method is very costly for 3D problem since we will deal with a large number of design variables (i.e., surface mesh points). Both difficulties can be easily resolved by using a set of smooth functions to perturb the initial mesh so that the surface mesh points are *mapped* from a limited number of design variables, such as in MSES-LINDOP [10–12], SU2 [15]. Figure 7 shows the surface mesh point at i, j which is moved by $\delta X_{\Gamma_{i,j}}$, Fig. 8 shows the surface mesh points moved by Hicks–Henne bumps.

Deforming the computational mesh is an efficient alternative to re-meshing and it enables a smooth *mapping* from the design parameters to the cost function. One issue for mesh deformation is that by deforming the surface boundary of the mesh

Fig. 8 The surface mesh points are moved by Hicks–Henne bumps [37]



points, the rest of the grids must be deformed accordingly. SU2 uses the linear elasticity equations [15, 36] to compute the volume mesh displacement from the displacement of the perturbed surface. If the computational cells are small, this prevents creation of negative volume cells by deformation. In certain circumstances, further mesh smoothing [28] will be required.

3.4.2 Hicks–Henne Bumps

A single Hicks–Henne (HH) bump function [37] perturbs the airfoil shape (y coordinates) by a “bumps,” so that with a sequence of HH bumps there obtains a perturbation of airfoil shape,

$$\Delta y(x) = \sum_{k=1}^N \alpha_k \sin \left(\pi x \frac{\log 0.5}{\log x_k} \right)^t \tag{20}$$

with the x -locations of max-points are $x_k, k = 1, 2, \dots, N$, and the coefficients α_k are design variables. Figure 9 shows an example of the fourth order bumps ($t = 4$) with $N = 10$, x_k is equally distributed over $[0.5/N, 1 - 0.5/N]$.

3.4.3 Free-Form Deformation

Free-form deformations (FFD) provide a method of deforming an object by adjusting the control points of a lattice. The technique was first described by Sederberg and Parry in 1986 [38] and its effect is used in computer animation. In 2D the shape perturbations are simply modeled by Bézier/Bspline/NURB control points [33, 34, 38, 39]

$$d\mathcal{Y}(x, y) = \sum_{i,j=0}^{nx-1,ny-1} dCP_{i,j} B_i^{nx}(u) B_j^{ny}(v) \tag{21}$$

$$x = x_{\min} + u(x_{\max} - x_{\min})$$

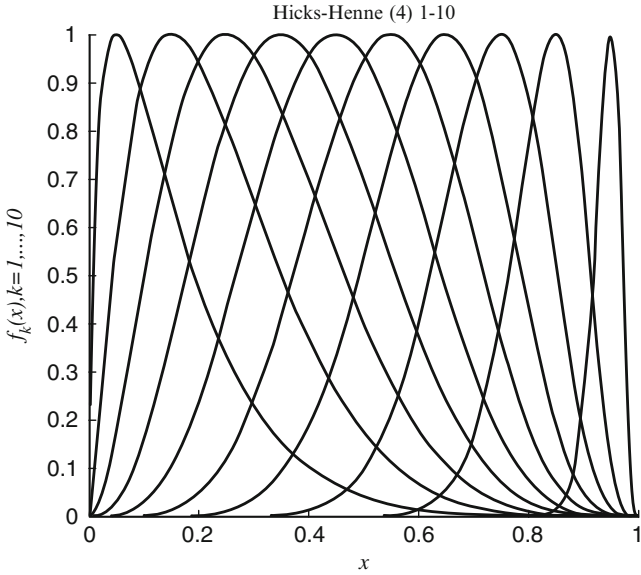


Fig. 9 The Hicks–Henne bump functions, $t = 4$, with $N = 10$, x_k is equally distributed over $[0.5/N, 1 - 0.5/N]$

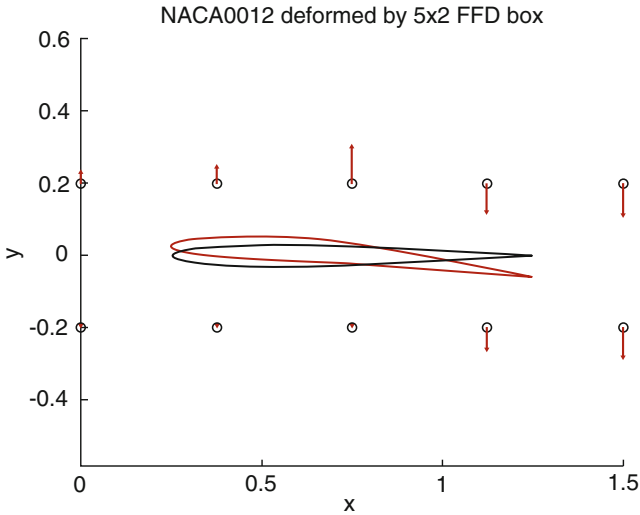


Fig. 10 NACA-0012 is deformed by a 5×2 FFD Bézier box

where n_x, n_y are the degrees of the FFD functions, $u, v \in [0, 1]$ are the parametric coordinates, $CP_{i,j}$ is the $n_x \times n_y$ array of control points, B_s are the Bernstein polynomials [32]. Fig. 10 shows an example that deforms an NACA-0012 airfoil by a 5×2 FFD Bézier box. In SU2 deformation of the baseline wing is done by a 3D FFD Bézier box [15, 38] in a similar way.

4 Test Cases Statement

4.1 Case I: RAE-2822 Airfoil in Transonic Viscous Flow

The drag should be minimized at Mach number 0.734 and lift coefficient of 0.824, and the cross section area must exceed or equal that of the baseline. Initial angle of attack is 2.79° . The flow is viscous with Reynolds number $Re = 6.5 \times 10^6$. The optimization problem is

$$\begin{aligned}
 \min : & \quad c_d \\
 \text{subjectto :} & \quad c_\ell = 0.824 \\
 & \quad c_m \geq -0.092 \\
 & \quad A \geq A_{baseline}
 \end{aligned} \tag{22}$$

where c_l , c_d , and c_m are the lift, drag, and pitch moment coefficients, A is the airfoil cross section area.

4.2 Case II: ONERA M6 Wing Optimization in Transonic Inviscid Flow

The ONERA M6 wing is a classic computational fluid dynamics (CFD) validation case for external flows because of its simple geometry combined with complexities of transonic flow (i.e., local supersonic flow, shocks, etc.). It has almost become a standard for CFD codes because of its inclusion as a validation case in numerous CFD papers over the years [40]. In the proceedings of a single conference, the 14th AIAA CFD Conference³ (1999), the ONERA M6 wing was included in 10 of the approximately 130 papers. This wing configuration is used here as a baseline for drag minimization. The drag should be minimized at Mach number 0.8395 and the flow is assumed to be inviscid. The maximum thickness t of each section should be preserved to a specified value. Initial angle of attack is 3.06° .

$$\begin{aligned}
 \min : & \quad c_d \\
 \text{subjectto :} & \quad c_\ell = 0.2864 \\
 & \quad t_{i,max} = t_{i,specified}
 \end{aligned} \tag{23}$$

³<http://www.aiaa.org/>.

5 Results from Direct Optimization SU2

5.1 Test Case I

The Spalart–Allmaras turbulent model [41] is used in this test case. Figure 11 shows the airfoil grids with 140,573 nodes. The mesh is perturbed by Hicks–Henne bump functions [37] with 19 design variables. Table 1 shows the optimization solution table for RAE 2822 airfoil (Fig. 12). The KKT condition [6] is met after 35 design cycles. The shock at around 55 % chord is weakened, with a drag benefit of 70 drag counts.⁴ Figure 13 shows the pressure distribution and airfoil shape for both baseline and optimized airfoils. The C_p of the optimized shape has wiggles in between 0.5 and 0.6 chord, that the shock starts to re-build a little. Figure 14 shows the Mach contours of both RAE 2822 and its optimized shape, with weakened shock on the optimized shape. Note that the Mach is re-developed between 0.5 and 0.6 chord.

5.2 Test Case II

It is an unstructured mesh with 36,454 tetrahedral cells, half geometry with a symmetric plane at $y = 0$. The wing tip is capped. The wing is parametrized by FFD Bézier box [38] as discussed in previous section by 176 design variables, with root section unchanged. The optimized wing is obtained after 14 design cycles, the drag coefficient C_D is reduced by around 17.9 *counts*, while the lift coefficient

Fig. 11 RAE-2822 airfoil mesh with 140,573 nodes

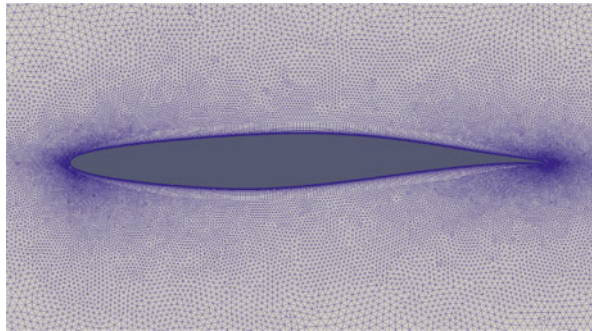
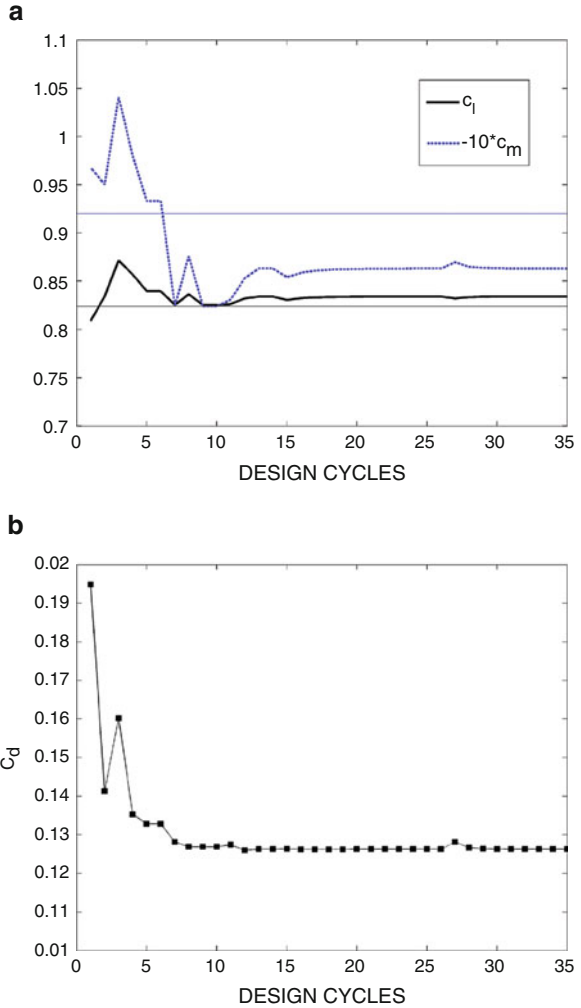


Table 1 RAE 2822 optimization results table

Airfoil	c_l	c_d	c_m
Baseline RAE2822	0.8092	0.01949	-0.09679
Optimized	0.8431	0.01263	-0.08631

⁴1 drag count is defined as 10^4 drag coefficient; 1 lift count is defined as 10^3 lift coefficient.

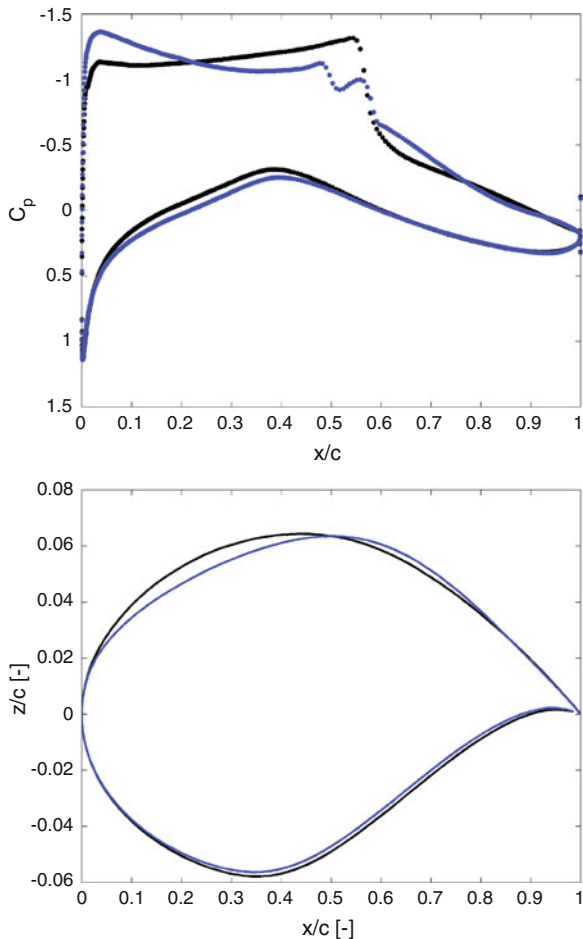
Fig. 12 The design cycles history for RAE 2822 with 19 design variables on the 140,573 nodes mesh. (a) Convergence of constraints on c_l and $-10 * c_m$ (b) Convergence of cost function c_d



C_L is increased by 4% as it can be seen in Table 2. Figure 15 shows the pressure coefficient comparisons for M6 baseline and its optimized wing shape from SU2. The shock is reduced such that 17 drag counts benefit is obtained.

The wing section profiles are studied at five stations from root (0 m) to tip (1.1963 m) on both baseline and optimized geometries. The maximum thickness varies from section to section, and its maximum locations even shifted a bit forward on inboard sections, see Figures 16a, b. The optimizer changes twist by less than 1 degree, and maximum chamber by less than 1% to arrive at a point believed to be a local optimum, using 176 design variables. This indicates that the wing is hard to improve on.

Fig. 13 Pressure distributions and airfoil shapes for RAE 2822, *black*: baseline; *blue*: optimized



5.3 Assessment: Direction Optimization

The direct optimization gives a rapid indication of possible directions for improvement when traditional inverse design and/or geometric cut-and-try are impractical. It also provides the possible design improvement paths when unusual non-aerodynamic design variables are present, for example, the r.m.s. strain constraints. Due to the fact that it minimizes the cost function and the cost function can be defined in multi-points, it is suitable to handle multi-point design problems [42], whereas single-point design is better handled with traditional inverse design. For the 2D problem, the optimized result in the test case was obtained after only 35 iterations. This approach is clear to extend to 3D (as the ONERA M6 wing), but needs even longer computation time.

Fig. 14 Mach contour for RAE 2822 and its optimized shape at Mach 0.734, $c_l = 0.824$. (a) Baseline RAE 2822 Airfoil (b) Optimized shape

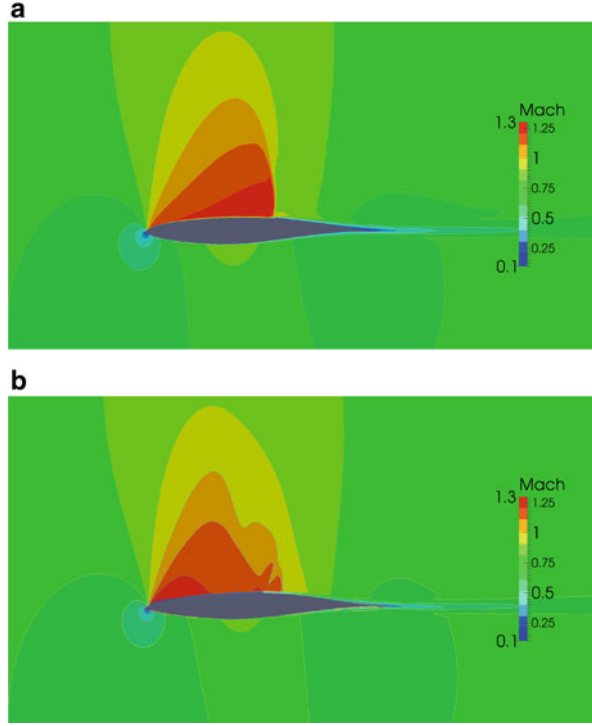


Table 2 M6 optimization results table

Shape	C_L	C_D
Baseline (ONERA M6)	0.28641	0.0117909
Optimized	0.298087	0.010016

The cons are also considerable. First of all, it requires tough learning curve, and one cell with high surface gradient can influence the overall search direction. For example, SU2 has an option that removes the sharp edge sensitivities from the gradient calculation to guarantee a descent direction in optimization [17]. This treatment to sharp trailing edge is easier to find the mean gradient. However, the drawback is that it removes the gradient from trailing edge, resulting little geometric changes around trailing edge, if we recalled the ONERA M6 wing case, the optimized wing has little twists and cambers compared with the baseline configuration, see Figure 16.

Setting up the optimization problem requires engineering skill as well. The cost function and the constraints should be well defined to ensure convergence. Palacios et al. [17] claimed a “sequential way” to apply constraints when designing a simple wing-body configuration in transonic viscous flow using SU2 otherwise the optimizer would fail. The gradient is sensitive to the mesh deformation method/strategy, if the gradient is not on the order of a meaningful dimensional perturbation of the design variables (control points), the first step of the optimizer will cause the mesh deformation to fail due to too large of a step being taken.

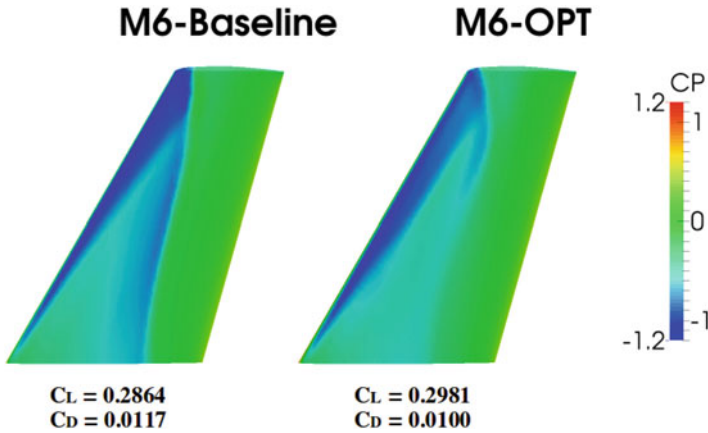


Fig. 15 Upper surface C_p for ONERA M6 baseline and optimized wing from SU2 optimization

This approach is always too “myopic,” it is trapped in the local optimum rather than finding the global optimum, exploiting the smallest significant physical scales. For 3D problem, it requires long computing time on big computers, although the adjoint approach reduces the gradient calculation time a lot.

6 Results from Inverse Design SCID

6.1 Test Case I

The target pressure is defined as a pressure distribution with weakened shocks. The target is found perfectly with 50 iterations by SCID-inviscid mode to get quick convergence, see Figure 17. Due to zero pressure gradient through boundary layer we would rather use SCID-inviscid mode to compute once the target pressure is given. However there are form drag and skin friction drag that SU2 can give while SCID-inviscid mode cannot. A compromise is made by re-running the solution from inviscid SCID in MSES [12] viscous mode, see Figure 18. The drag is reduced from 170 counts to 115 counts, with c_m constraint perfectly held (Table 3).

6.2 Case II Variation

A variation design of test case II is carried out, which is a similar exercise as Jameson did [43] by the adjoint code for inverse design. The wing planform is ONERA-M6 and the initial geometry was made up of NACA 0012 sections and

Fig. 16 Geometric comparison for M6 baseline vs. optimized results on five spanwise stations. (a) Maximum thickness (b) Maximum thickness locations (c) Local twist distributions

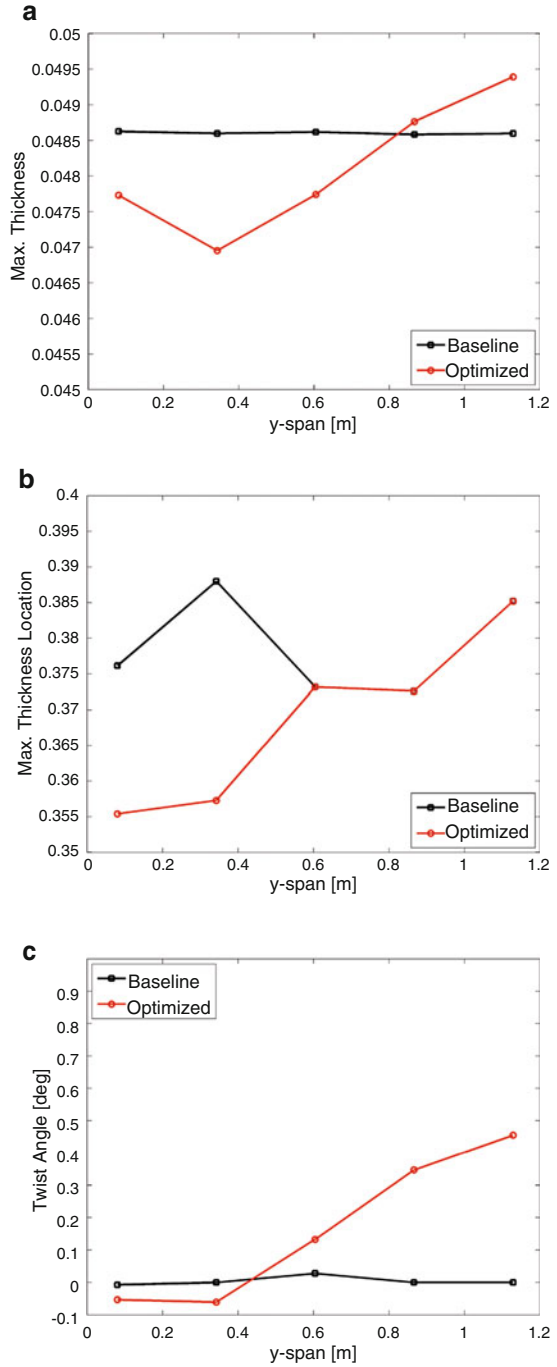
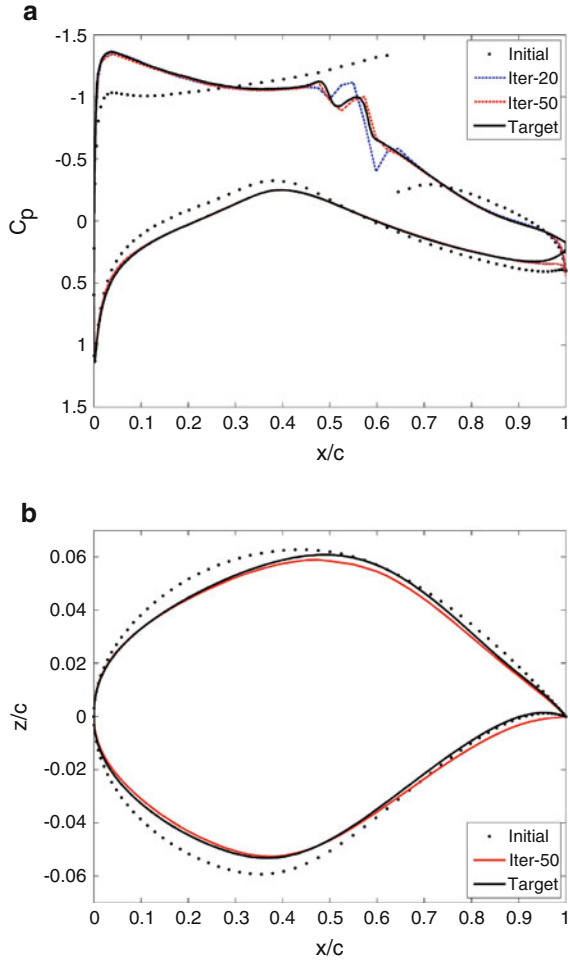


Fig. 17 The resulting pressure distributions and the airfoil shape from SCID-inviscid; baseline/initial shape is RAE-2822 airfoil. (a) Pressure distributions computed from SCID-inviscid (b) Resulting airfoil shapes computed from SCID-inviscid



the target pressure distribution was the pressure distribution over the ONERA-M6 wing. The target pressure distribution was computed by *SU2* in inviscid flow with the same mesh and under the same conditions as computed the ONERA M6 baseline in the previous section, namely, unstructured grids with 36,454 cells (rather coarse), Mach number 0.8395, angle of attack is fixed at 3.06° . Eight equally spaced sections are designed for half wing, from 0% of the wing semi-span (root) to 94.32% of the wing semi-span. Figures 19 and 20 show the pressure distribution and the section geometries over the initial NACA 0012 airfoil wing and the final design by *SCID*. The final design was achieved by 110 designs. Note that after 40 designs the target pressure distribution was already almost found, with only slightly deviations. More design cycles would not make significant difference. Smoothing plays an important

Fig. 18 Re-run the solution Iter-50 in viscous mode

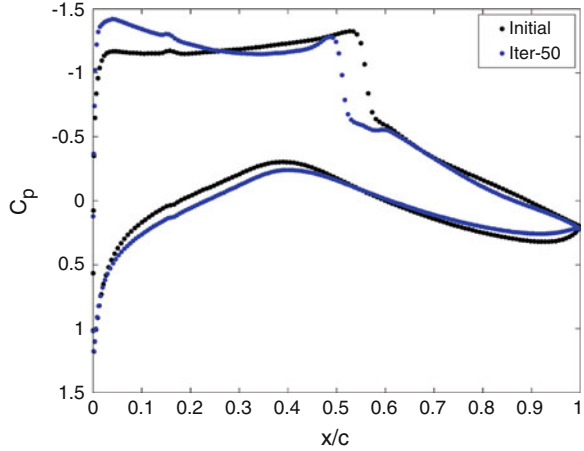


Table 3 RAE 2822 design by re-running the baseline and the optimized solution from inviscid SCID in MSES viscous mode

Airfoil	c_l	c_d	c_m
Baseline RAE2822 -vis	0.824	0.016987	-0.09394
Iter-50 -vis	0.824	0.011512	-0.05896

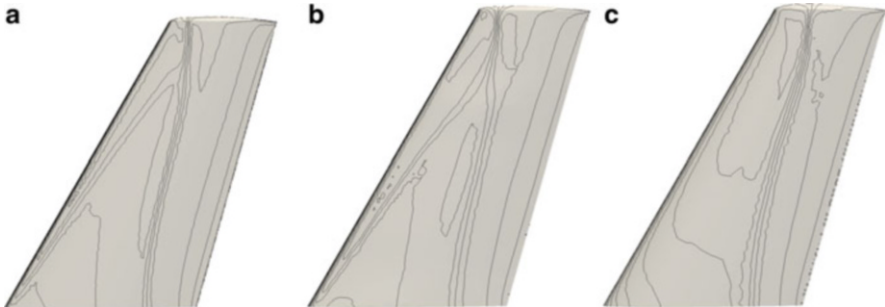


Fig. 19 Initial and Final C_p contours comparisons of M6 wing planform. (a) Target pressure distribution contours over an M6 wing computed by SU2; (b) Final pressure distribution contours obtained by SCID-inviscid mode; (c) Initial pressure distribution contours over an M6 wing with NACA 0012 profile

role, the different smoothing technique would lead different pressure distributions especially for the last design cycles. As Jameson [43] claimed, this is a particularly challenging test, because it calls for the recovery of a smooth symmetric profile from an asymmetric pressure distribution containing a triangular pattern of shock waves, (Table 4, Fig. 21).

Fig. 20 Initial and Final pressure distribution and modified section geometries along the wing, computed by SCID-inviscid mode, compared with target pressure distribution computed by SU2 Root. (a) section, 0% semi-span (b) Mid section: 54.26% semi-span (c) Tip section: 94.32% semi-span

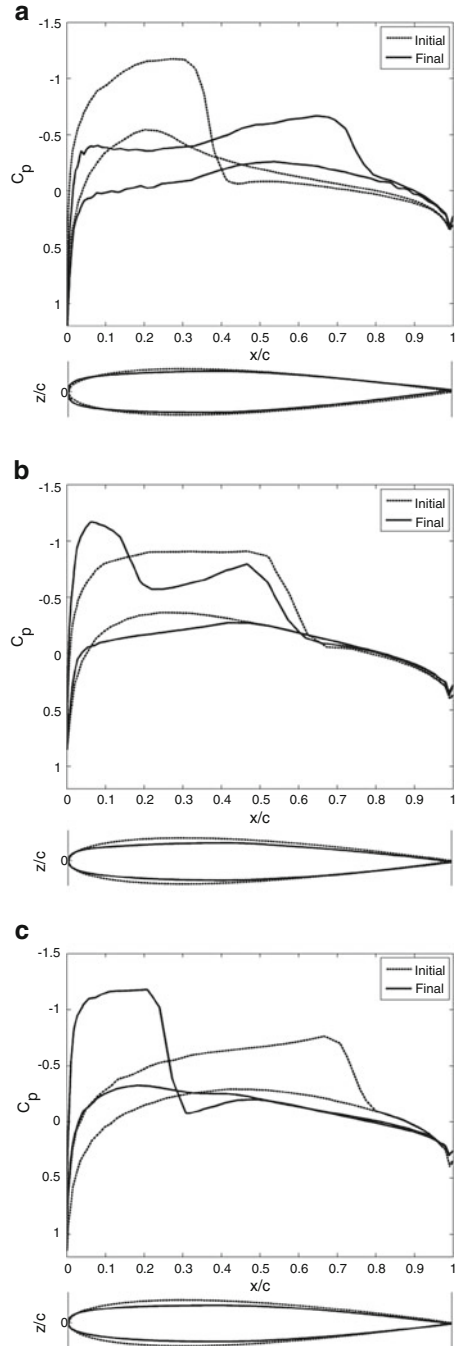


Table 4 Wing design for ONERA M6 planform by SCID

ONERA M6 Planform	C_L	C_D	C_m
Initial SCID	0.29828	0.020906	-0.133737
Final SCID	0.28634	0.01540	-0.11941
Target SU2	0.28641	0.0117909	-0.120047

6.3 Assessment: Inverse Design

The significant advantages of inverse design are that the engineer applies her knowledge and (thus) a realistic airfoil is obtained at every iteration (e.g., MSES inverse design mode [12]). The pressure gradient is zero through boundary layer [44] so that it can be chosen to maintain laminar flow. The changes in streamline curvature–pressure relationship are robust that it ensures fast convergence to target pressure with good quality. The engineer in the loop is seeking favorable properties of the pressure distributions such as:

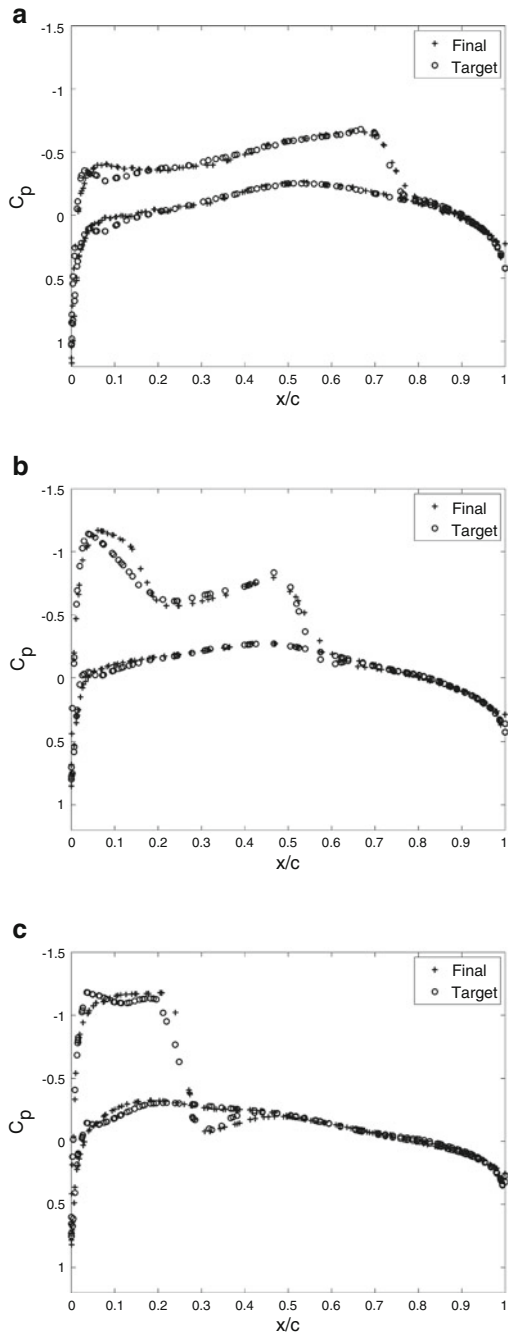
1. the flattened leading edge pressure peak on the upper surface which avoids leading edge flow separation;
2. weakened or eliminated shock waves which reduce the wave drag;
3. monotonic trailing edge pressure recovery that avoids boundary layer separation.

However, using the inverse method, the engineer must be well experienced and knowledgeable to know how to set the target pressure. Because this method works on pressure distributions rather than the lift or drag coefficients, it may not reach a true optimum, at best only the target pressure. Moreover, the streamline curvature–pressure relation used in SCID is more tricky and complex for transonic flow in 3D since the shocks and cross-flow are introduced [18].

7 Results from Hybrid Design: Case I

Figure 22 shows the pressure distributions on the baseline airfoil calculated in MSES inviscid mode, and the target pressure distributions (solid). The target pressure is obtained from the optimized solution of RAE 2822 airfoil in inviscid flow using EDGE adjoint solver, which is out of scope here. Directly driving the C_p towards the “target” pressure distributions causes divergence. What we need to do in LINDOP is to make the design *iteratively*. There are many options that user can interact, for example, the search direction, the search step, and quasi-Newton toggle. Figures 23 and 24 show the gradient-inverse design procedures in LINDOP inviscid mode. In the former figure only the upper surface is modified, and the C_p value at the trailing edge (right endpoint of freewall segment) is fixed. In the latter figure only the lower surface is modified, and the stagnation C_p value (left endpoint of freewall segment) is fixed.

Fig. 21 Final and target pressure distributions along the wing using inverse design. (a) Root section, 0 % semi-span (b) Mid section: 54.26 % semi-span (c) Tip section: 94.32 % semi-span



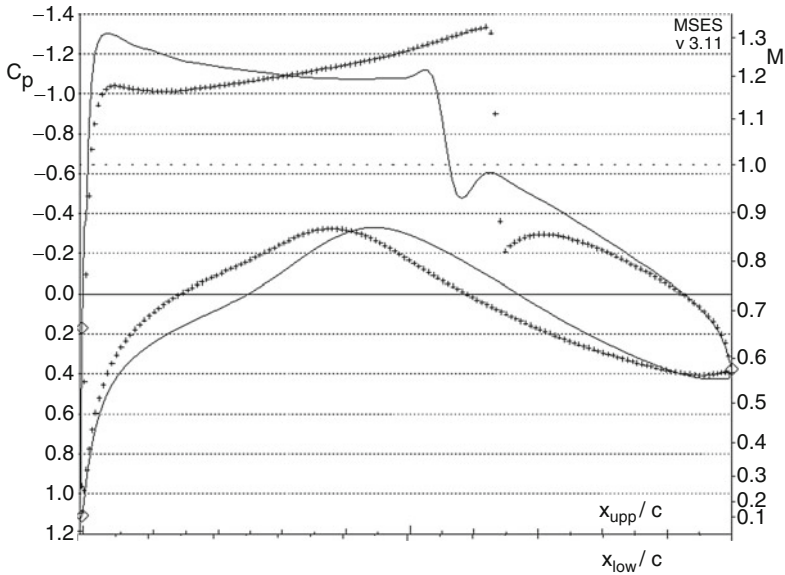


Fig. 22 The pressure for baseline airfoil RAE2822 (*markers*) calculated in MSES inviscid mode and the target pressure distributions (*solid*)

Figure 25 shows the optimized results from RAE 2822 airfoil, inviscid flow solutions. It is found that in inviscid flow, the drag coefficient c_d is reduced by 74 %, the lift coefficient is maintained, and the c_m constraint is held (Table 5).

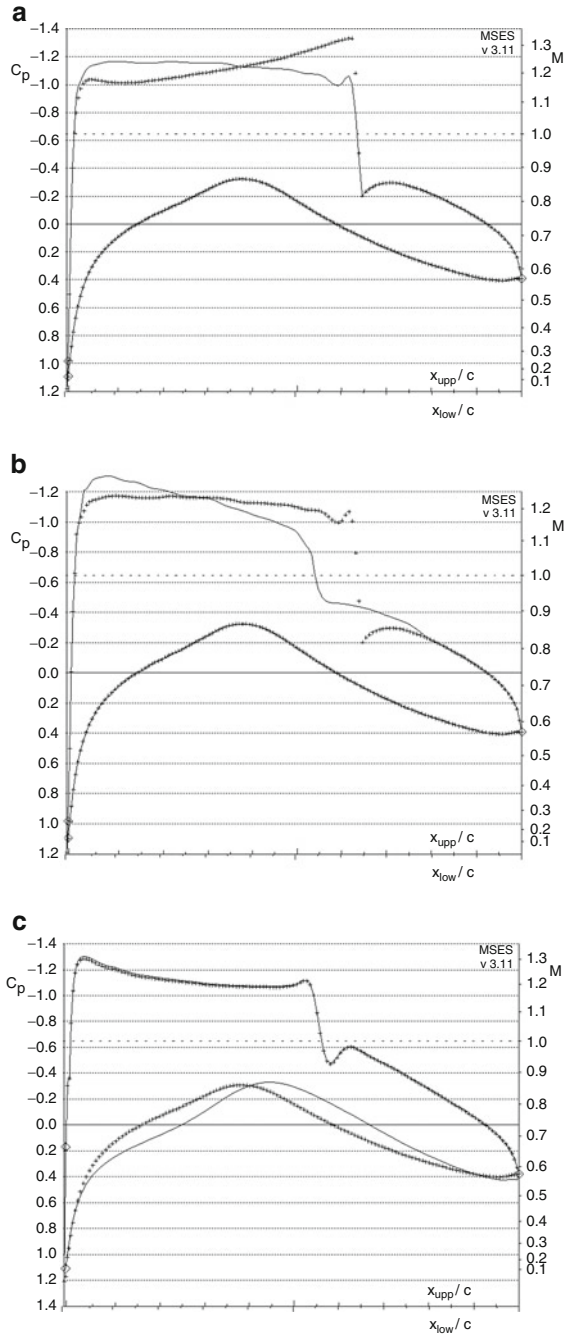
7.1 Assessment: Hybrid Design

The pros of hybrid design are:

- it can systematically set constraints and cost functions, thus can obtain benefits of both inverse design and direct optimization;
- it offers the engineer know-how advantages (c.f. long list of user options in LINDOP menu);
- iterations with visualized feedback and (thus) a realistic airfoil is obtained at every iteration.

However, there are a number of cons for hybrid design approach. It also has a tough learning curve for *users*, and it is too manual to set/determine too many options, especially for new users. There is no guarantee for convergence, unless the target pressure is a “small” modification of the initial one (e.g., Figures 23, 24). There are some tricks to get convergent solutions. First of all, make small pressure changes (ΔC_p) for each design cycle; second, modify/re-design one surface each

Fig. 23 The design procedures for RAE 2822 in LINDOP inviscid mode, modify upper surface *only*, to be continued. (a) Fix right endpoint segment (b) Fix right endpoint segment (c) Fix right endpoint segment



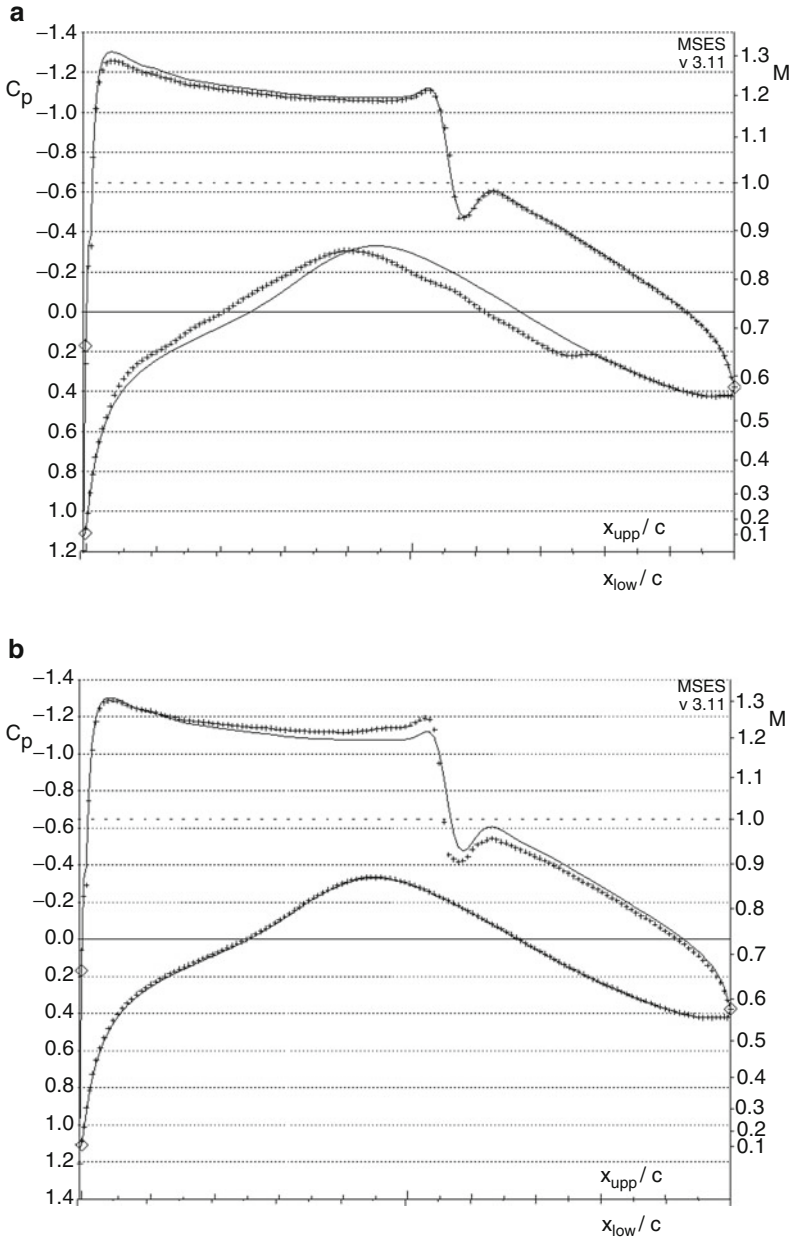


Fig. 24 The design procedures for RAE 2822 LINDOP inviscid mode, modify lower surface *only*, completed. (a) Fix left endpoint segment (b) Fix left endpoint segment

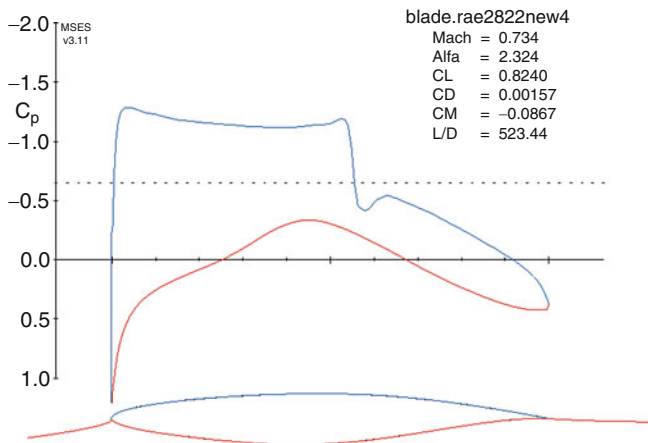


Fig. 25 The optimized design of RAE 2822 in LINDOP inviscid mode

Table 5 RAE 2822 optimization results table, computed in LINDOP inviscid mode

Airfoil	C_l	C_d	C_m
Baseline RAE2822 - invis	0.824	0.00648	-0.1294
Iter-50 -invis	0.824	0.00157	-0.0867

time; third, carefully choose order of mixed inverse prescribed shape function and global constraints; fourth, fix the left/right endpoint of freewall segment each time; finally, introduce the target pressure C_p^* from the last cycles when the current C_p is close to C_p^* . All of the tricks and the options which should be determined by users make the hybrid design method tough for new users.

8 Conclusions

This chapter assesses three different design methods for aerodynamic shape design by two test cases. They are not isolated to each other, the “hybrid” design is to combine the first two approaches. It is difficult to say one is superior than the other, each of them has pros and cons. What we can do is to understand the strong and weak points of each method, and use the appropriate and/or combined methods to a specified design problem. Van der Velden called it “cocktails” or combinations of optimizers [45] under the control of the engineer in the loop. This is also stressed in Zhang’s PhD thesis [5].

Acknowledgements Part of the computations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at PDC Centre for High Performance Computing (PDC-HPC)

References

1. Sobieszczanski-Sobieski, J.: *Multidisciplinary Design Optimization: An Emerging New Engineering Discipline*. Kluwer Academic, Boston (1995)
2. Kroo, I.M.: *Multidisciplinary Design Optimization: State-of-the-Art*, chapter MDO for Large-Scale Design, pp. 22–44. SIAM, Philadelphia (1997)
3. Vassberg, J.C., Jameson, A.: Influence of shape parametrization on aerodynamic shape optimization. Technical report, Brussels, Belgium (2014) Von Karman Institute Lecture-III
4. Castonguay, P., Nadarajah, S.K.: Effect of shape parameterization on aerodynamic shape optimization. In: 45th AIAA Aerospace Sciences Meeting and Exhibit, 2007-59, Reno, Nevada (2007)
5. Zhang, M.: *Contributions to Variable Fidelity MDO Framework for Collaborative and Integrated Aircraft Design*. Doctoral thesis, Department. Aeronautics, KTH Kungl. Tekniska Högskolan (2015)
6. Griva, I., Nash, S.G., Sofer, A.: *Linear and Nonlinear Optimization*, 2nd edn. Society for Industrial Applied Mathematics, Philadelphia (2009)
7. Campbell, R.L., Smith, L.A.: A hybrid algorithm for transonic airfoil and wing design. In: AIAA Aerospace Sciences Meeting, 87-2552 (1987)
8. Lamar, J.E.: A vortex-lattice method for the mean camber shapes of trimmed non-coplanar planforms with minimum vortex drag. Technical report, Washington, DC (1976). NASA TN D-8090
9. Barger, R.L., Jr. Brooks, C.W.: A streamline curvature method for design of supercritical and subcritical airfoils. Technical report (1974) NASA TN D-7770
10. Drela, M.: A user's guide to lindop v2.50. Technical report (1996)
11. Drela, M.: Lindop optimization procedures. Technical report, Computational Aerospace Sciences Laboratory, MIT Department of Aeronautics and Astronautics (1993). Technical Report
12. Drela, M.: Newton solution of coupled viscous/inviscid multielement airfoil flows. In: AIAA Aerospace Sciences Meeting, 90-1470 (1990)
13. Jameson, A.: Aerodynamic design via control theory. *J. Sci. Comput.* **3**, 233–260 (1998)
14. Lions, J.L.: *Optimal Control of Systems Governed by Partial Differential Equations*. Springer, New York (1971)
15. Palacios, F., Colonno, M.R., Aranake, A.C., Campos, A., Copeland, S.R., Economon, T.D., Lonkar, A.K., Lukaczyk, T.W., Taylor, T.W.R., Alonso, J.: Stanford University Unstructured (SU2): an open-source integrated computational environment for multi-physics simulation and design. In: 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, AIAA 2013-0287, Grapevine, 7–10 Jan (2013)
16. Nadarajah, S.K., Jameson, A.: A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization. In: 38th Aerospace Sciences Meeting, AIAA 2000-0667 (2000)
17. Palacios, F., Economon, T.D., Wendorff, A.D., Alonso, J.: Large-scale aircraft design using SU2. In: 53st AIAA Aerospace Sciences Meeting, AIAA 2015-1946, Kissimmee, Florida, 5–9 Jan (2015)
18. Zhang, M., Rizzi, A., Nangia, R.: Transonic airfoil and wing design using inverse and direct methods. In: AIAA SciTech, AIAA 2015-1934, Kissimmee, Florida, 5–9 Jan (2015)
19. Dulikravich, G.S., Baker, D.P.: Aerodynamic shape inverse design using a Fourier series method. In: AIAA Aerospace Sciences Meeting, AIAA 90-0185 (1990)
20. Campbell, R.L.: An approach to constrained aerodynamic design with application to airfoils. Technical report
21. Obayashi, S.: Genetic optimization of target pressure distributions for inverse design methods. In: AIAA Aerospace Sciences Meeting, AIAA-95-1649-CP (1995)

22. Streit, T., Wichmann, G., Von Knoblauch Zu Hatzbach, F., Campbell, R.: Implications of conical flow for laminar wing design and analysis. In: 29th AIAA Applied Aerodynamics Conference, AIAA 2011-3808 (2011)
23. Zhang, M., Wang, C., Rizzi, A., Nangia, R.: Hybrid feedback design for subsonic and transonic airfoils and wings. In: AIAA SciTech, AIAA 2014-0414, National Harbor, Maryland, 13–17 Jan (2014)
24. Takanashi, S.: An iterative procedure for three-dimensional transonic wing design by the integral equation method. In: AIAA 2nd Applied Aerodynamic Conference, AIAA-84-2155, Seattle, Washington, 21–23 August 1984
25. Mark Drela, M., Giles, M.B.: Viscous-inviscid analysis of transonic and low Reynolds number airfoils. *AIAA J.* **25**(10), 1347–1355 (1987)
26. Drela, M.: Design and optimization method for multi-element airfoils. In: AIAA Aerospace Sciences Meeting, 93-0969 (1993)
27. Amoignon, O., Navratil, J., Hradil, J.: Study of parametrization in the project CEDESA. In: AIAA SciTech, 2014-0570, National Harbor, Maryland, (2014)
28. Jakobsson, S., Amoignon, O.: Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization. *Comput. Fluids* **36**(6), 1119–1136 (2007)
29. Mohammadi, B., Molho, J.I., Santiago, J.G.: Design of minimal dispersion fluidic channels in a CAD-free framework. In: Center for Turbulence Research, Proceedings of the Summer Program 2000
30. Kenway, G.K.W., Kennedy, G.J., Martins, J.R.R.A.: A CAD-free approach to high-fidelity aerostructural optimization. In: 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference, AIAA 2010-9231, Fort Worth, Texas, 13–15 Sept (2010)
31. Tomac, M., Eller, D.: From geometry to CFD grids. an automated approach for conceptual design. *Prog. Aerosp. Sci.* **47**(11), 589–596 (2011). doi: 10.1016/j.paerosci.2011.08.005
32. Gallier, J.: Curves and surfaces in geometric modeling: Theory and algorithms. Technical report, Department of Computer and Information Science, University of Pennsylvania (2013)
33. Farin, G.E., Hoschek, J., Kim, M.-S.: Handbook of Computer Aided Geometric Design. Elsevier, New York (2002)
34. de Boor, C.: A practical guide to splines. Springer, New York (1978)
35. Melin, T., Amadori, K., Krus, P.: Parametric wing profile description for conceptual design. In: CEAS Meeting, Venice, Italy (2011)
36. Dwight, R.P.: Robust mesh deformation using the linear elasticity equations. In: Proceedings of the Fourth International Conference on Computational Fluid Dynamics, ICCFD, Ghent, 10–14 July 2006
37. Hicks, R., Henne, P.A.: Wing design by numerical optimization. In: AIAA, AIAA 77-1247, Seattle, Washington, 22–24 Aug (1977)
38. Sederberg, T.W., Parry, S.R.: Free-form deformation of solid geometric models. *ACM Trans. Math. Softw.* **20**(4), 151–160 (1986)
39. Les, P., Wayne, T.: The NURBS Book, 2nd edn. Springer, New York (1997)
40. Schmitt, V., Charpin, F.: Pressure distributions on the ONERA-M6_Wing at transonic Mach numbers, AGARD AR138 (1979)
41. Piquet, J.: Turbulent Flows: Models and Physics. Springer, New York (2001)
42. Lyu, Z., Kenway, G.K.W., Martins, J.R.R.A.: RANS-based aerodynamic shape optimization investigation of the common research model wing. In: 52nd Aerospace Sciences Meeting, AIAA 2014-0567, National Harbor, Maryland, 13–17 Jan 2014. doi: 10.2514/6.2014-0567
43. Jameson, A.: Aerodynamic shape optimization using the adjoint method. Technical report, Brussels, Belgium (2003) Von Karman Institute Lecture-III
44. Anderson, J.D.: Computational Fluid Dynamics: The Basic with Applications, International edition, McGraw-Hill, New York (1995)
45. Van der Velden, A.: The global aircraft shape. In: AGARD-FDP-VKI Special Course at VKI, pp. 9–1–9–11, Rhode-Saint-Genese, April (1994)