67. Preference-Based Multiobjective Particle Swarm Optimization for Airfoil Design

Robert Carrese, Xiaodong Li

A significant challenge to the application of evolutionary multiobjective optimization (EMO) for transonic airfoil design is the often excessive number of computational fluid dynamic (CFD) simulations required to ensure convergence. In this study, a multiobjective particle swarm optimization (MOPSO) framework is introduced, which incorporates designer preferences to provide further guidance in the search. A reference point is projected onto the Pareto landscape by the designer to guide the swarm towards solutions of interest. The framework is applied to a typical transonic airfoil design scenario for robust aerodynamic performance. Time-adaptive Kriging models are constructed based on a high-fidelity Reynolds-averaged Navier–Stokes (RANS) solver to assess the performance of the solutions. The successful integration of these design tools is facilitated through the reference point, which ensures that the swarm does not deviate from the preferred search trajectory. A comprehensive discussion on the proposed optimization framework is provided, highlighting its viability for the intended design application.

67.1 Airfoil Design

Airfoil design originates from an understanding of the fundamental physics of flight, where the aim is to identify or conform to the best possible shape for the given operating requirements. It has evolved from the use of wind tunnel catalogs and traditional *cut-and-try* methods to automated computational frameworks. While automated frameworks effectively simplify the design process, success is still largely dependent on the fidelity of the computational methods, as well as the experience of the designer in formulating the problem [67.1]. This section is devoted to a discussion of airfoil design optimization architecture. The concepts that are especially applicable to this study are introduced, laying the foundations for the proposed methodology.

67.1.1 Airfoil Design Architecture

The direct method of airfoil design, pioneered by the work of *Hicks* and *Henne* [67.2], refers to the philosophy of using mathematical optimization methods to identify the optimal shape that achieves the prescribed

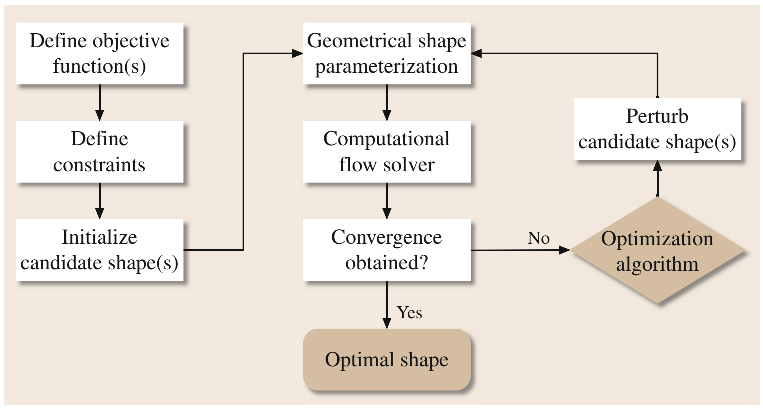

Define objective function(s)
Define constraints
Initialize candidate shape(s)
Geometrical shape parameterization
Computational flow solver
Convergence obtained?
No
Yes
Optimal shape
Optimization algorithm
Perturb candidate shape(s)

Fig. 67.1 Generalized process flowchart for direct airfoil shape optimization

design criteria. The generalized framework for an aerodynamic shape optimization process is demonstrated in Fig. 67.1. The success of the direct approach is essentially dependent on three main components within the design loop:

- *Shape parameterization.* All design strategies share the common requirement that the geometry is represented by a finite number of design variables. A method to mathematically parameterize shapes is, therefore, required so that modifications can be made via direct manipulation of the design variables. The number of design variables is directly proportional to the geometrical degrees of freedom and, therefore, governs the dimensionality of the problem.
- *Computational flow solver.* The objective function is obtained from the flow solver and it is, therefore, up to the discretion of the designer to appropriately formulate the objective and constraint functions, such that they reflect the design and operating requirements. The choice of the flow solver ultimately governs the overall fidelity and efficiency of the optimization process, since repeated evaluations of the objective function are required for each candidate shape.
- *Optimization algorithm.* The responsibility of the optimizer is to iteratively determine the shape modifications required to satisfy the objective, whilst adhering to any shape or performance constraints. The optimizer should be robust and applicable to a wide operational spectrum, yet efficient to guarantee convergence with the least computational expense.

The integration of high-fidelity flow solvers and flexible parameterization methods for numerical optimization is still a computationally challenging and intensive undertaking. The extension to multiple objectives leads to a more generalized problem formulation, yet significantly increases the computational cost of convergence. While all elements of the design loop influence the efficiency of the process, arguably the most important element is the optimizer itself. The following section introduces the optimization paradigm adopted in this study, derived from the field of computational swarm intelligence.

67.1.2 Intelligent Optimization: PSO

The formation of hierarchies within groups of animals is a naturally occurring phenomenon and is simple to comprehend. Even humans have the intuitive tendency to appoint leaders (e.g., political leaders, military generals, etc.). Another interesting phenomenon, which is more difficult to perceive, is the self-organized behavior of groups where a leader *cannot* be identified. This is known as swarming and is evident from the flocking behavior of birds or fish moving in unison. The increasingly cited field of computational swarm intelligence focuses on the artificial simulation of swarming behavior to model a wide range of applications, including optimization [67.3].

Particle swarm optimization (PSO) is the stochastic population-based technique described by *Kennedy* and *Eberhart* [67.4] in accordance with the principles of swarm intelligence. The PSO architecture was derived from a synthesis of the fields of social psychology and engineering optimization. As was eloquently stated by the authors in their original paper [67.4]:

Why is social behavior so ubiquitous in the animal kingdom? Because it optimizes. What is a good way *to solve engineering optimization problems? Modeling social behavior.*

The dynamics of the swarm are modeled on the social-psychological tendency of individuals to learn from previous experience and emulate the success of others. Similar to most evolutionary techniques, the swarm is initialized with a population of random individuals (particles) sampled over the design space. The particles navigate the multi-dimensional design space over a number of iterations or time steps. Each particle maintains knowledge of its current position in the design space. This is analogous to the *fitness* concept of conventional evolutionary algorithms (EAs). Each particle also records its personal best position, which is where the particle has experienced the greatest success. Aside from recording personal information, each particle also tracks the position of other members in the swarm. This level of social interaction between particles is coined the swarm *topology*. Particles may either be confined to share information only with their immediate neighbors, or they may be encouraged to share their experiences with the entire swarm. Utilizing this information, each particle adjusts its position in the design space by *accelerating* towards the successful areas of the design space. The absence of selection is compensated by this use of leaders to guide the swarm to converge to the most successful position. In this way, a solution which initially performs poorly may possibly be on the future road to success.

PSO has steadily gained popularity as a global optimization technique [67.3]. Its increasing use in the literature is due to its simple and straightforward implementation (despite its intricate origins) and its efficient and accurate convergence rates [67.5].

67.1.3 Multiobjective Optimization

Airfoil design problems are often characterized by several interacting or conflicting requirements, which must be satisfied simultaneously. In the case of an airfoil operating within the transonic regime, airfoil shape optimization is performed to limit shock and viscous drag (C_d) losses, and reduce shock-induced boundary layer instability at the design Mach number (M) and lift coefficient (C_l). This often occurs at the expense of excessive pitching moments (C_m) due to aft loading and performance degradation under off-design conditions. To facilitate adequate performance over a wide operational spectrum requires a search algorithm capable of handling multiple conflicting objectives.

Let $S \in \mathbb{R}^n$ denote the design space and let $\boldsymbol{x} = \{x_1, x_2, \dots, x_n\} \in S$ denote the decision vector with lower and upper bounds $\boldsymbol{x}_{\min}$ and $\boldsymbol{x}_{\max}$, respectively. The generic unconstrained multiobjective problem (MOP) is thus expressed as,

$$\min \boldsymbol{f}(\boldsymbol{x}) = \{f_1(\boldsymbol{x}), \dots, f_m(\boldsymbol{x})\} \ , \tag{67.1}$$

where $f_i(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ is the i-th component of the objective vector and m is the number of objectives. The definition of the optimum must be redefined since in the presence of conflicting objectives, improvement in one objective may cause a deterioration in another. It is often necessary to identify a set of trade-off solutions, which can all be considered equally optimal. A solution is termed non-dominated or Pareto optimal (after the nineteenth century Italian economist Vilfredo Pareto) if the value of any objective cannot be improved without deteriorating at least one other objective. The candidate solutions are defined as $\boldsymbol{a}$ and $\boldsymbol{b} \in S$. The candidate $\boldsymbol{a}$ *dominates* the candidate $\boldsymbol{b}$ (denoted by $\boldsymbol{a} \prec \boldsymbol{b}$) if,

$$\forall j = 1, \dots, m \quad f_j(\boldsymbol{a}) \le f_j(\boldsymbol{b}) \wedge \exists j : f_j(\boldsymbol{a}) < f_j(\boldsymbol{b}) \ . \tag{67.2}$$

The concept of dominance is illustrated in Fig. 67.2. The shaded area denotes the area of objective vectors dominated by $\boldsymbol{a}$. A decision vector $\boldsymbol{a^*}$ is, therefore, non-dominated or Pareto optimal if there is no other feasible decision vector $\boldsymbol{a} \neq \boldsymbol{a^*} \in S$ such that $f(\boldsymbol{a}) \prec f(\boldsymbol{a^*})$. The Pareto front is the set of objective vectors which correspond to all non-dominated solutions. Multiobjective algorithms aim to identify the closest approximation to the true Pareto front, while ensuring a diverse Pareto optimal set.

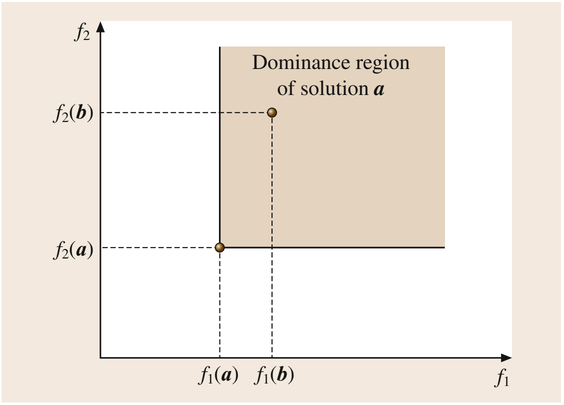


Fig. 67.2 Illustration of dominance on a two-objective landscape

Techniques for Solving MOP

From a design perspective, the primary aim of multiobjective optimization is to obtain Pareto optimal solutions which are in the preferred interests of the designer, or best suit the intended application. Methods for solving MOPs are, therefore, characterized by how the designer preferences are articulated. As suggested by *Fonseca* and *Fleming* [67.6], there are three generic classes of methods for solving multiobjective problems:

- A priori methods. The preferences of the designer are expressed by aggregating the objective functions into a single scalar through weights or bias, ultimately making the problem single objective.
- A posteriori methods. The algorithm first identifies a set of non-dominated solutions, subsequently providing the designer greater flexibility in selecting the most appropriate solution.
- *Interactive* methods. The decision making and optimization processes occur at interleaved steps, and the preferences of the designer are interactively refined.

The a priori strategy requires the designer to indicate the relative importance of each objective before performing the optimization. A notable method that falls into this category is the weighted aggregation method, which is a fairly popular choice for airfoil design applications due to its simplicity and capability of handling many flight conditions [67.7–9]. Despite its popularity, there are recognized deficiencies with this strategy [67.10]. The prior selection of weights does not necessarily guarantee that the final solution will faithfully reflect the preferred interests of the designer, and varying the weights continuously will not necessarily result in an even distribution of Pareto optimal solutions, nor a complete representation of the Pareto front [67.11].

Alternatively the a posteriori methods provide maximum flexibility to the designer to identify the most preferred solution, at the expense of greater computational effort. Generally, these methods involve explicitly solving each objective to obtain a set of non-dominated solutions, a concept which is ideal for population-based evolutionary algorithms [67.12–14]. While these methods are computationally more complex, researchers in aerodynamic design are realizing the benefits of evolutionary multiobjective optimization (EMO), especially if there is a certain ambiguity in selecting the final design [67.15–17]. However, it poses the challenge of identifying and exploiting the entire Pareto front, which may be impractical for design applications due to the excessive number of function evaluations.

While conventional EMO techniques may be computationally demanding, *Fonseca* and *Fleming* [67.12] argue that their most attractive aspect is the intermediate information generated, which can be exploited by the designer to refine preferences and improve convergence. These *interactive* methods involve the progressive articulation of preferences, which originates from the multicriteria decision making literature [67.18]. The optimization and decision making processes are interleaved, exploiting the intermediate information provided by the optimizer to refine preferences [67.6].

Handling Multiple Objectives with PSO

PSO has been demonstrated to be an effective tool for single-objective optimization problems due to its fast convergence [67.5]. It has also gained rapid popularity in the field of multi-objective optimization (MOO) [67.19]. Since PSO is a population-based technique, it could ideally be tailored to identify a number of trade-off solutions to a MOP in one single run, similar to EMO techniques. Comprehensive surveys on extending PSO to handle multiple objectives have been provided by *Engelbrecht* [67.20], and more recently by *Sierra* and *Coello Coello* [67.19]. It was established that the primary ambiguity in specifically tailoring PSO to handle multiple objectives was the selection of guides for each particle to avoid convergence to a single solution. The selection process for particle leaders must, therefore, be restructured, to encourage search diversity and to ensure that non-dominated solutions found during the search are maintained.

Initial attempts to design a multiobjective particle swarm optimization (MOPSO) algorithm were motivated by the archive strategy by [67.21]. *Coello Coello* and *Lechuga* [67.22] incorporated the concept of Pareto dominance in PSO by maintaining two independent populations: the particle swarm and the elitist archive. Non-dominated solutions are stored in the archive and subsequently used as neighborhood leaders. The objective space is separated into hypercubes, which serve as a particle anti-clustering mechanism. Solutions in sparsely populated hypercubes have a higher selection pressure to be leaders, and solutions in densely populated hypercubes are removed if the archive limit is exceeded. This initial approach was later extended by *Mostaghim* and *Teich* [67.23], who studied the concept of ϵ-dominance and compared it to existing clustering techniques for fixing the archive size, with favorable results.

Fieldsend and *Singh* [67.24] addressed the computational complexity of maintaining a restricted archive, by incorporating the *dominated tree* method. This data structure allows for an unrestricted archive size, which interacts with the population to define global leaders. A *turbulence* operator (similar to the concept of mutation in EA) was also implemented, where swarm members were randomly displaced on the design space to reduce the probability of premature stagnation. In the non-dominated sorting particle swarm optimization (NSPSO) algorithm of *Li* [67.25], the non-dominated sorting mechanisms of non-dominated sorting genetic algorithm (NSGA-II) are incorporated. The population and the personal best position of each particle are consolidated to form one single population, and the non-dominated sorting scheme is utilized to rank each solution. Global guides are selected based on particle clustering, where a niching or crowding distance metric is used to further classify non-dominated solutions. *Li* later proposed the maximinPSO algorithm [67.26], which does not use any niching method to maintain diversity.

Sierra and *Coello Coello* [67.27] proposed an elitist archive incorporating the ϵ-dominance strategy to maintain global leaders for the swarm. A crowding distance operator is employed to classify non-dominated solutions and maintain uniformity. The crowding distance operator is also used to limit the number of candidate leaders after each population update, simplifying the mechanism to control the set of candidate leaders. A turbulence operator is implemented to encourage diversity, whereby particles are randomly mutated. A similar approach by [67.28] was developed in parallel (although this method does not implement ϵ-dominance), where the crowding distance was used to both define the global guides and truncate the size of the archive. The proposed algorithm is primarily influenced by the two latter studies.

Preference-Based Optimization

The concept of interactive optimization has led to an increasing interest in coupling classical interactive methods to EMO as an intuitive way of reflecting the designer preferences and identifying solutions of interest to the designer. This has led to the development of the preference-based optimization philosophy, which provides the motivation for the current study. Comprehensive surveys on preference-based optimization are provided by *Coello Coello* [67.29] and *Rachmawati* and *Srinivasan* [67.30]. The first recorded attempt at incorporating preferences within an evolutionary multiobjective optimization framework was made by *Fonseca* and *Fleming* [67.31] using the goal programming approach. Goal programming [67.11] is an ideal approach to indicate desired levels of performance for each objective, since they closely relate to the final solution of the problem. Goals may either represent target or ideal values. *Fonseca* and *Fleming* later extended the approach where an online decision making strategy was proposed based on goal and priority information [67.6]. A goal programming mechanism for identifying preferred solutions for MOP was also proposed by [67.32]. While the reported frameworks draw on the preferred interests of the designer to aid the optimization process, the goal programming approach is computationally complex, and there is no means of specifying any relation or trade-off between the objectives [67.30].

Thiele et al. [67.33] proposed another variant of interactive evolutionary multiobjective optimization. A coarse representation of the Pareto front is initially presented to the designer. The most interesting regions are subsequently isolated, on which the algorithm continues to focus on exclusively. This proposal effectively removes the necessity to predefine target values for each objective and provides the designer with a means of isolating the preferred trade-offs. However, it is a two-stage approach requiring an initial approximation to the Pareto front, which may be unnecessarily expensive. The integration of other classical preference articulation methods has also been proposed in the literature. A reference point-based evolutionary multiobjective optimization framework was proposed by [67.34]. The crowding distance operator of the NSGA-II algorithm was modified to include the reference point information and the extent of the preferred region was controlled by ϵ-dominance. *Deb* and *Kumar* also experimented with the use of other classical preference methods, such as the reference direction method [67.35] and the light beam search method [67.36].

Recently, the use of interactive methods has also been integrated within PSO frameworks. *Wickramasinghe* and *Li* [67.37] integrated the reference point method to both the NSPSO [67.25] and maximinPSO [67.26] algorithms. Significant improvement in convergence efficiency was highlighted, and it was demonstrated that final solutions are of higher relevance to the designer. *Wickramasinghe* and *Li* [67.38] later extended their approach to handle MOP, by replacing the dominance criteria entirely with the simpler distance metric. It was conclusively demonstrated that without the use of the reference point, obtaining a final set of preferred solutions solely through conventional dominance-based techniques is improbable.

67.1.4 Surrogate Modeling

The most prohibiting factor of design optimization is the cost of evaluating the objective and constraint functions. For high-fidelity airfoil design, function evaluations may very well be measured in hours. This computational burden ultimately questions the practicality of performing an optimization study, and is often alleviated by simply reducing the level of sophistication of the solver. This consequently reduces the fidelity of the final design, which is undesirable. Another mitigating strategy which has steadily gained popularity in design is the use of inexpensive surrogates or metamodels [67.39]. These models emulate the response of the expensive function at an unobserved location, based on observations at other locations. Surrogate models are not specifically optimization methods, but rather they may ideally be used in lieu of the expensive function to extract information from the design space during the optimization process.

The insightful texts by *Keane* and *Nair* [67.39] and *Forrester* et al. [67.40] provide a detailed account of the use of surrogates in design. The most common use is to construct a *curve fit* of an expensive function landscape, which can be used to predict results without recourse to the original function. This is supported by the assumption that the inexpensive surrogate will still be usefully accurate when predicting sufficiently far from observed data points [67.40]. Figure 67.3 illustrates the use of a surrogate to fit the one-dimensional multi-modal function, based on four sample observations. It is important to note, however, that the original function landscape could potentially represent any deterministic quantity of the design space. Rather than exactly emulating the response of a high-fidelity flow solver, the surrogate may, in fact, be used to bridge the gap between flow solvers of varying fidelity [67.40]. Alternatively, a surrogate may be used to interpret or filter noisy landscapes, so as to eliminate the adverse effects of flow solver convergence or grid discretization. Surrogates may also be used for data mining and design space visualization. Such methodologies are applied to extract useful information about the relationship between the design space and the objective space, allowing informed decisions to be made, which could simplify a seemingly complex problem.

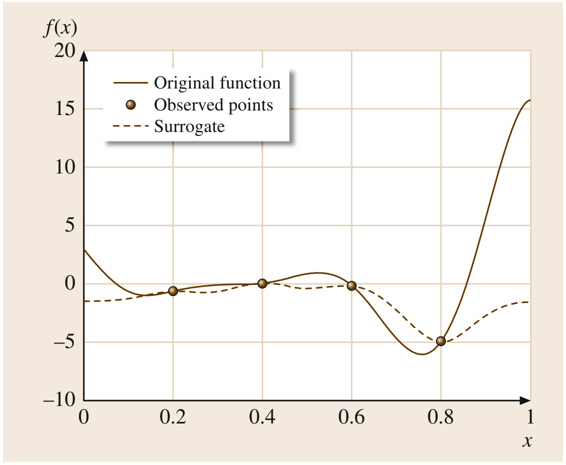


Fig. 67.3 Constructing an interpolation-based surrogate to fit a one-dimensional function

For the aforementioned uses of surrogate modeling, the common requirement is to replicate the function relationship between the variable inputs and the output quantity of interest. This is typically achieved by sampling the design space using the exact function to sufficiently model the underlying relationship within the allowable computational budget. Whether the aim is to locally model the design space surrounding an existing design or tune a surrogate to replicate the global design space is entirely dependent on the formation of the sampling plan [67.39]. The construction of a surrogate model in either case should ideally make use of a parallel computing structure. A suitable surrogate model $\hat{f}$ of the precise objective function f should then be constructed to fit the dataset.

There are a multitude of popular techniques for constructing surrogates in the literature. For a comprehensive review of different methods, the reader is referred to (among others) [67.39–42]. The selection of the surrogate model is dependent on the information that the designer is attempting to extract from the design space. Polynomial response surfaces and radial basis functions are fairly popular techniques for constructing local surrogates, especially if some level of regression is desirable. Techniques such as Kriging or support vector machines are more ideally suited to global optimization studies, since they offer greater flexibility in tuning model parameters and provide a confidence interval of the predicted output. Neural networks require extensive training and validation, yet have also been a popular technique for design applications, notably in aerodynamic modeling [67.43] and visualization techniques [67.44, 45].

67.2 Shape Parameterization and Flow Solver

It was established in Sect. 67.1.1 that the shape parameterization method essentially governs the dimensionality of the problem and the attainable shapes, whereas the objective flow solver dictates the overall fidelity of the optimum design. In this section, we present a discussion on these elements of the design loop to be used in conjunction with the developed optimizer for the subsequent design process.

67.2.1 The PARSEC Parameterization Method

Geometry manipulation is of particular importance in aerodynamic design. The selection of the shape parameterization method is an important contributing factor, since it will effectively define the objective landscape and the topology of the design space [67.46]. If the aim of the optimization process is to improve on an established design, then perhaps local parameterization methods, which offer a greater number of geometrical degrees of freedom, are desirable. However, the large number of variables may cause the convergence rate for global design applications to deteriorate. The development of efficient parameterization models has, therefore, been given significant attention, to increase the flexibility of geometrical control with a minimum number of design variables.

For certain applications, it is possible to make use of fundamental aerodynamic theory to refine the parameterization method, such that the design variables relate to important aerodynamic or geometric quantities. A common method for airfoil shape parameterization is the PARSEC method [67.47]. It has the advantage of strict control over important aerodynamic features, and it allows independent control over the airfoil geometry for imposing shape constraints. The methodology is characterized by 11 design variables (Fig. 67.4), including leading edge radius (r_{LE}), upper and lower thickness locations ($x_{\mathrm{UP}}, z_{\mathrm{UP}}, x_{\mathrm{LO}}, z_{\mathrm{LO}}$) and curvatures ($z_{xx_{\mathrm{UP}}}, z_{xx_{\mathrm{LO}}}$), trailing edge direction (α_{TE}) and wedge angle (β_{TE}), and trailing edge coordinate (z_{TE}) and thickness (Δz_{TE}). The shape function is modeled via a sixth-order polynomial function

$$z_k = \sum_{n=1}^{6} a_{n,k} \cdot x_k^{n-\frac{1}{2}} , \tag{67.3}$$

where (x, z) are the shape coordinates and k denotes either the upper (suction) or lower (pressure) airfoil surface. The coefficients a_n are determined from the geometric parameters. A modification by *Jahangirian* and *Shahrokhi* [67.48] was introduced to provide additional control over the trailing edge curvature. For supercritical transonic airfoils, this is beneficial to reduce the probability of downstream boundary layer separation, which results in increased drag values. A new variable $\Delta\alpha_{\mathrm{TE}}$ was introduced, which directly influences the additional curvature of the trailing edge. The modification decouples the trailing edge parameterization by first defining a smoother upper surface contour and then constraining the lower surface to intersect the trailing edge coordinate. Figure 67.5 illustrates the modification to the trailing edge curvature. The modification is applied to the upper and lower surfaces as follows

$$\delta_z = \frac{L \cdot \tan \Delta\alpha_{\mathrm{TE}}}{2\mu\tau} \left[1 + \eta \cdot x^{\tau} - \left(1 - x^{\tau}\right)^{\mu} \right] , \tag{67.4}$$

where the constants η, μ, and τ are set to 0.8, 2, and 6, respectively. The modification is applied over the entire

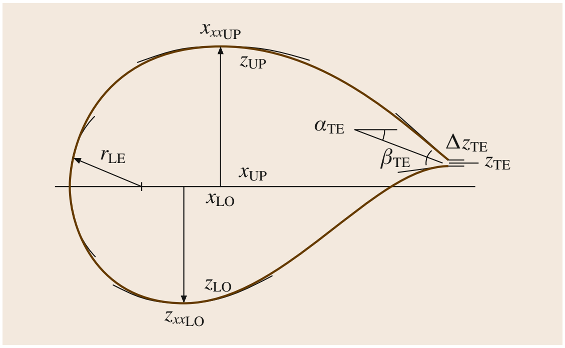


Fig. 67.4 Airfoil representation via the PARSEC method

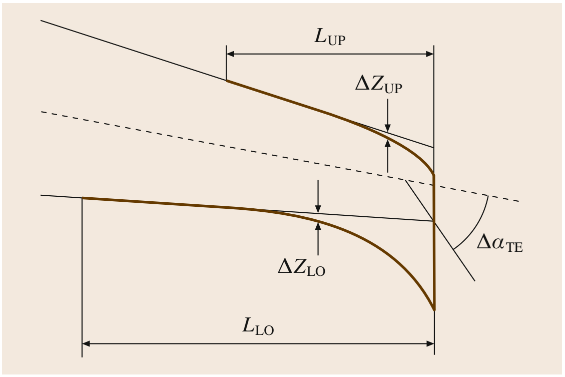


Fig. 67.5 Additional trailing edge curvature via the modified PARSEC method

Table 67.1 PARSEC parameter ranges for transonic optimization

Description	Variable	Lower bound	Upper bound
Leading edge radius	r_{LE}	0.0063	0.0151
Trailing edge direction	α_{TE}	0.2405(−)	0.0026(−)
Trailing edge wedge angle	β_{TE}	0.0655	0.2618
Upper-crest abscissa	x_{UP}	0.3170	0.5250
Upper-crest ordinate	z_{UP}	0.0497	0.0683
Upper-crest curvature	z_{xxUP}	0.5135(−)	0.2393(−)
Lower-crest abscissa	x_{LO}	0.2835	0.3418
Lower-crest ordinate	z_{LO}	0.0603(−)	0.0478(−)
Lower-crest curvature	z_{xxLO}	0.2535	0.8405
Trailing edge curvature	$\delta_{\alpha TE}$	0.0080(−)	0.3696

surface, such that $L_{UP} = L_{LO} = c$, where c is the airfoil chord length.

Table 67.1 presents the upper and lower boundaries for the subsequent optimization case study. These boundaries have been selected based on a thorough screening study involving a statistical sample of a number of benchmark airfoils.

67.2.2 Transonic Flow Solver

The optimization process is ultimately dependent on the selection of the flow solver, since it is the most computationally expensive component, and repeated evaluations of the objective and constraint functions are required for each candidate shape. However if the flow solver is not sufficiently accurate, the optimization process will converge to shapes that exploit the numerical errors or limitations, rather than the fundamental physics of the problem. For this reason, it is desirable to maintain the correct balance between solution accuracy and computational expense, which is dictated by the flow regime. For certain problems where the aerodynamic flow field is well behaved, it may be sufficient to consider more robust linear solvers. However for high-fidelity design it is prudent to consider non-linear and more computationally demanding solvers, to ensure that optimized shapes provide the anticipated performance requirements in flight.

The general purpose finite volume code ANSYS Fluent is adopted in this study. A pressure-based numerical procedure is adopted with third-order spatial discretization to capture the occurring flow phenomena. The momentum equations and pressure-based continuity equation are solved concurrently, with the Courant–Friedrichs–Lewy number set at 200. The one-equation

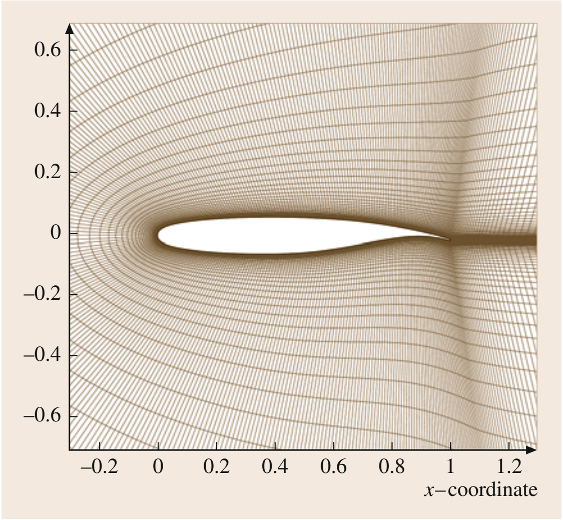


Fig. 67.6 C-type grid for transonic simulation

Spalart–Allmaras turbulence model [67.49] is selected, and turbulent flow is modeled over the entire airfoil surface. The C-type grid (as represented in Fig. 67.6) stretches 25 chord lengths aft and normal of the airfoil section. Resolution of the C-grid is 460×65, providing an affordable mesh size of approximately 30 000 elements. The first grid point is located 2.5×10^{-4} units normal to the airfoil surface, resulting in an average y-plus value of 120. In the interest of robust and efficient convergence rates, a full multi-grid (FMG) initialization scheme is employed, with coarsening of the grid to 30 cells. In the FMG initialization process, the Euler equations are solved using a first-order discretization to obtain a flow field approximation before submission to the full iterative calculation.

67.3 Optimization Algorithm

The proposed algorithm was primarily motivated by the studies of *Wickramasinghe* and *Li* [67.38]. The principal argument is that for most design applications, to explore the entire Pareto front is often unnecessary, and the computational burden can be alleviated by considering the immediate interests of the designer. In Sect. 67.1.3, a discussion on the benefits of preference-based optimization was provided. Drawing on these concepts, a preference-based algorithm is proposed, where a designer-driven distance metric is used to scalar quantify the success of a solution. The multiobjective search effort is coordinated via a MOPSO algorithm. The swarm is guided by a reference point, which is an intuitive means of articulating the preferences of the designer and can ideally be based on an existing or target design. This section provides a comprehensive discussion on the proposed algorithm, highlighting its viability for the intended domain of application.

67.3.1 The Reference Point Method

In this research, the swarm is guided by a reference point to confine its search focus exclusively on the preferred region of the Pareto front as dictated by the preferences of the designer. Introducing the preferred region provides the designer flexibility to explore other interesting alternatives. This hybrid methodology is advantageous for navigating high-dimensional and multimodal landscapes, which are typical of aerodynamic design problems. Furthermore, inherently considering the preferences of the designer provides a feasible means of quantifying the practicality of a design.

The Reference Point Distance Metric

The reference point method has been integrated into MOO algorithms, notably by *Deb* and *Sundar* [67.34] and *Wickramasinghe* and *Li* [67.37, 38]. These studies highlight the benefits of incorporating preference information via the reference point in terms of convergence. Guided by the information provided by the reference point, the swarm can simultaneously identify multiple solutions in the preferred region. This provides the designer flexibility to explore several *preferred* designs, while alleviating the computational burden of identifying the entire Pareto front. A reference point distance metric following the work of *Wickramasinghe* and *Li* [67.37] is proposed. This metric provides an intuitive criterion to select global leaders and assists the swarm to identify only solutions of interest to the designer. The distance of a particle $\boldsymbol{x}$ to the reference point $\bar{z}$ is defined as

$$d_z(\boldsymbol{x}) = \max_{i=1:m} \{(f_i(\boldsymbol{x}) - \bar{z}_i)\} \ . \qquad (67.5)$$

A solution $\boldsymbol{a}$ is, therefore, *preferred* to solution $\boldsymbol{b}$ if $d_z(\boldsymbol{a}) < d_z(\boldsymbol{b})$. This condition is an extension of the condition $f(\boldsymbol{a}) \prec f(\boldsymbol{b})$, therefore, the distance metric may, in fact, substitute the dominance criteria entirely [67.38]. Using this distance metric, the swarm is guided to preferred regions of the Pareto optimal front. Figure 67.7 illustrates the search directions of the algorithm when guided by a reference point, and the corresponding preferred design as a direct result of minimizing the distance metric d_z.

The distinguishing feature of the reference point distance metric over the mathematical Euclidean distance is that solutions do not converge to the reference point, but on the preferred region of the Pareto front as dictated by the search direction. This is illustrated in Fig. 67.8. All solutions are non-dominated and lie on the circular arc surrounding the reference point $\bar{z}$, and thus the Euclidean distance to the reference point is equal. However, since solution i has the smallest maximum translational distance to the reference point compared to any other solution, it is considered preferred. The definition of the reference point distance also suggests negative values. If the distance of the preferred solution $d_z(z') < 0$, then it can simply be considered that the reference point is dominated or $z' \prec \bar{z}$. Since the designer generally has no prior knowledge of the topology and location of the Pareto front,

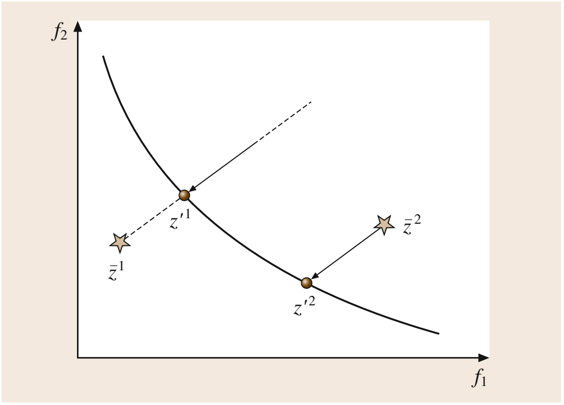


Fig. 67.7 Illustration of the search direction governed by the reference point

a reference point may be ideally placed in any feasible or infeasible region, as is shown in Fig. 67.7. It is, therefore, the consensus that the reference point draws on the experience of the designer to express the preferred compromise, rather than specific target values or goals. Similarly, the reference point distance metric ranks or assesses the success of a particle as one single scalar, instead of an array of objective values.

Defining the Preferred Region

As is demonstrated in Fig. 67.7, if there is no control over the solution spread the swarm will explore the preferred search direction and converge to the single solution z' as dictated by the reference point $\bar{z}$. The advantage of maintaining a population of particles provides the designer the possibility to explore a range of interesting alternatives within a preferred region of the Pareto front. The aim is, therefore, to identify a set of solutions surrounding the intersection point z'. A threshold parameter $\delta > 0$ is defined, such that a solution $\boldsymbol{x}$ is within the preferred region if the following conditional statement is true

$$d_z(\boldsymbol{x}) \le d_z(z') + \delta \,. \tag{67.6}$$

Figure 67.9 illustrates the preferred region for a bi-objective problem. The extent of the solution spread is proportional to δ and evidently as $\delta \to 0$, the designer is interested in determining only the most preferred solution z'. Conversely, as $\delta \to \infty$, the designer is interested in determining all solutions along the Pareto front, and thus the influence of the reference point location diminishes.

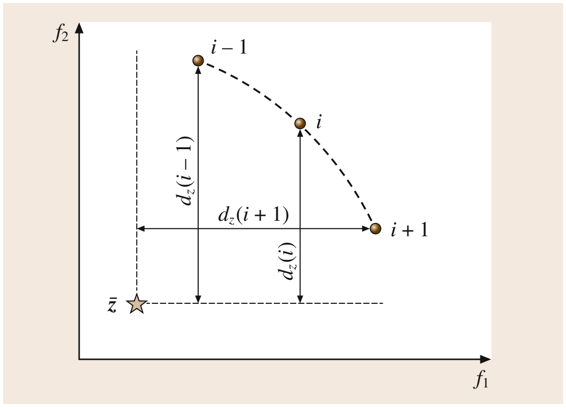


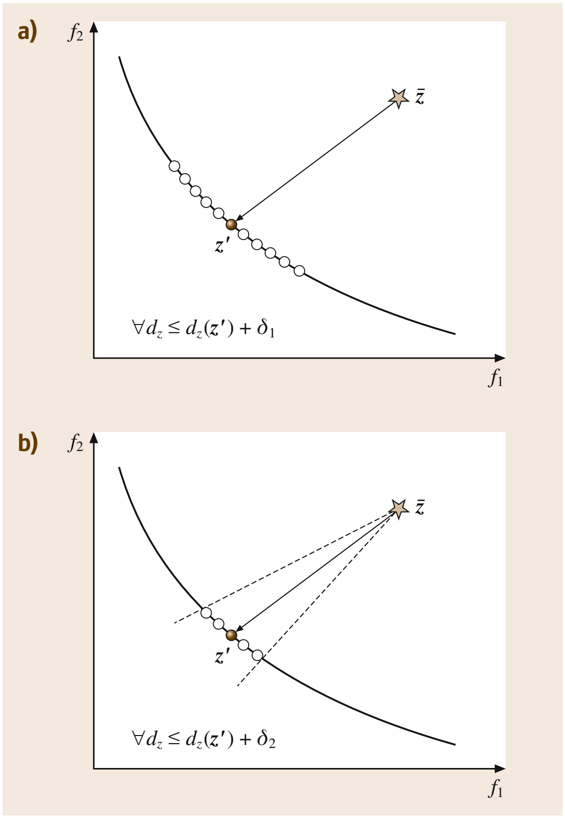
Fig. 67.8 Illustration of the reference point distance for solutions with equal Euclidean distance

67.3.2 User-Preference Multiobjective PSO: UPMOPSO

The proposed algorithm combines the searching proficiency of PSO and the guidance of the reference point method. The swarm is guided by the user-defined reference point to confine its search to focus exclusively on the identified preferred region of the Pareto front. While the concept of the reference point is fairly intuitive, ensuring that the swarm is guided by this information to identify preferred solutions is more ambiguous. The algorithm function is consolidated in Algorithm 67.1 and further described in the subsequent steps.

Algorithm 67.1 The UPMOPSO algorithm

```
1: OBTAIN user-defined preferences
2: INITIALIZE swarm
3: EVALUATE fitness and distance metric
4: ASSIGN personal best
```



a)
f_2
$\bar{z}$
z'
$\forall d_z \le d_z(z') + \delta_1$
f_1
b)
f_2
$\bar{z}$
z'
$\forall d_z \le d_z(z') + \delta_2$
f_1

Fig. 67.9a,b Definition of the preferred region via the parameter δ. **(a)** $\delta_1 = 0.01$, **(b)** $\delta_2 = 0.001$

```
 5: CONSTRUCT archive
 6: $t = 1$
 7: **repeat**
 8:     SELECT global leaders
 9:     UPDATE particle velocity
10:     UPDATE particle position
11:     ADJUST boundary violation
12:     EVALUATE fitness and distance metric
13:     UPDATE personal best
14:     UPDATE archive
15:     $t = t + 1$
16: **until** $t = t_{\max}$ OR $f_{\max}$
```

OBTAIN User-Defined Preferences

The designer stipulates the reference point $\bar{z}$ and the corresponding solution spread δ to define the location and extent of the preferred region. For airfoil design applications, designers can exploit the existing domain knowledge to determine the most feasible performance compromise for the desired operating conditions.

INITIALIZE the Particles

A swarm of N particles is required to navigate the design space S bounded by $\boldsymbol{x}_{\min}$ and $\boldsymbol{x}_{\max}$. To safeguard against magnitude and scaling issues, all variables are normalized into the unit cube, such that $S = [0, 1]^n$. The i-th particle in the swarm is characterized by the n-dimensional vectors $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$, which are the particle position and velocity, respectively. These vectors are randomly initialized within the unit cube at time $t = 0$. The particle personal best position is recorded as the particle position, such that $\boldsymbol{p}_i = \boldsymbol{x}_i$. The particles are then evaluated with the objective functions and fitness is assigned. The reference point distance metric is computed for each particle to measure the individual preference value.

UPDATE Archive and SELECT Global Leaders

A secondary population of non-dominated solutions in the form of an elitist archive is maintained at time, t. The non-dominated solutions identified by the particles are appended to the archive. A non-dominated sorting procedure is applied, where all members pertaining to local inferior fronts are omitted. The archive serves as a mutually accessible memory bank for the particles of the swarm. Each member is a potential candidate for global leadership of the particles during the subsequent velocity update.

Defining the global leaders ultimately governs the direction of the search. The swarm should efficiently navigate the design space such that the search effort is locally focused within the preferred region and provides a uniform spread of solutions. Since all members of the archive are mutually non-dominated, a ranking procedure is necessary to distinguish the most appropriate candidates for leadership from the remaining members. At each time step t, the most preferred solution $z'(t)$ is recorded. The subset of members $\boldsymbol{X}_g(t)$ selected for global leadership satisfy the condition of (67.6), such that

$$\boldsymbol{X}_g(t) \in d_z(\boldsymbol{Q}(t)) \leq d_z(z'(t)) + \delta \,. \tag{67.7}$$

Since not every member will initially satisfy this condition, the number of candidate leaders may fluctuate over time. This condition provides the necessary selection pressure for particles to locally focus the search effort within the preferred region, avoiding the unnecessary computational effort of exploring undesired regions of the design space. Each swarm particle is randomly assigned a leader to promote diversity in the search. In the case where all non-dominated solutions satisfy the condition of (67.7), additional guidance through a crowding distance metric (as described in [67.27]) is provided to promote a uniform spread.

As the particles are guided to converge to the preferred region, the number of identified non-dominated solutions will steadily increase. To avoid this number escalating unnecessarily and to maintain high competitiveness within the archive, there is a restriction (denoted by $K_{\max}$) on the number of solutions permitted for entry. If the number of members $K > K_{\max}$, the newest solution is permitted entry, and the existing least preferred member is removed. If all archive members exist within the preferred region, the most crowded solutions are removed. This ensures that solutions in densely populated regions are removed in favor of solutions which exploit sparsely populated regions, to further promote a uniform spread.

UPDATE Particle Position

The update equations of PSO adjust the position of the i-th particle from time t to $t+1$. In this algorithm, the constriction *type 1* framework of *Clerc* and *Kennedy* [67.50] is adopted. In their studies, the authors studied particle behavior from an eigenvalue analysis of swarm dynamics. The velocity update of the i-th particle is a function of acceleration components to both the personal best position, $\boldsymbol{p}_i$, and the global best position, $\boldsymbol{p}_g$. The updated velocity vector is given by the expression,

$$\boldsymbol{v}_i(t+1) = \chi\{\boldsymbol{v}_i(t) + \boldsymbol{R}_1[0,\varphi_1] \otimes (\boldsymbol{p}_i(t) - \boldsymbol{x}_i(t)) + \boldsymbol{R}_2[0,\varphi_2] \otimes (\boldsymbol{p}_g(t) - \boldsymbol{x}_i(t))\} . \quad (67.8)$$

The velocity update of (67.8) is quite complex and is composed of many quantities that affect certain search characteristics. The previous velocity $\boldsymbol{v}_i(t)$ serves as a memory of the previous flight direction and prevents the particle from drastically changing direction and is referred to as the inertia component. The cognitive component of the update equation $(\boldsymbol{p}_i(t) - \boldsymbol{x}_i(t))$ quantifies the performance of the i-th particle relative to past performances. The effect of this term is that particles are drawn back to their own best positions, which resembles the tendency of individuals to return to situations where they experienced most success. The social component $(\boldsymbol{p}_g(t) - \boldsymbol{x}_i(t))$ quantifies the performance of the i-th particle relative to the global (or neighborhood) best position. This resembles the tendency of individuals to emulate the success of others.

The two functions $\boldsymbol{R}_1[0,\varphi_1]$ and $\boldsymbol{R}_2[0,\varphi_2]$ return a vector of uniform random numbers in the range $[0,\varphi_1]$ and $[0,\varphi_2]$, respectively. The constants φ_1 and φ_2 are equal to $\varphi/2$ where $\varphi = 4.1$. This randomly affects the magnitude of both the social and cognitive components. The constriction factor χ applies a dampening effect as to how far the particle explores within the search space and is given by

$$\chi = 2/|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}| . \quad (67.9)$$

Once the particle velocity is calculated, the particle is displaced by adding the velocity vector (over the unit time step) to the current position,

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1) . \quad (67.10)$$

Particle flight should ideally be confined to the feasible design space. However, it may occur during flight that a particle involuntarily violates the boundaries of the design space. While it is suggested that particles which leave the confines of the design space should simply be ignored [67.51], the violated dimension is restricted such that the particle remains within the feasible design space without affecting the flight trajectory.

UPDATE Personal Best

The ambiguity in updating the personal best using the dominance criteria lies in the treatment of the case when the personal best solution $\boldsymbol{p}_i(t)$ is mutually nondominated with the solution $\boldsymbol{x}_i(t+1)$. The introduction of the reference point distance metric elegantly deals with this ambiguity. If the particle position $\boldsymbol{x}_i(t+1)$ is preferred to the existing personal best $\boldsymbol{p}_i(t)$, then the personal best is replaced. Otherwise the personal best is remained unchanged.

67.3.3 Kriging Modeling

Airfoil design optimization problems benefit from the construction of inexpensive surrogate models that emulate the response of exact functions. This section presents a novel development in the field of preference-based optimization. Adaptive Kriging models are incorporated within the swarm framework to efficiently navigate design spaces restricted by a computational budget. The successful integration of these design tools is facilitated through the reference point distance metric, which provides an intuitive criterion to update the Kriging models during the search.

In most engineering problems, to construct a globally accurate surrogate of the original objective landscape is improbable due to the weakly correlated design space. It is more common to locally update the prediction accuracy of the surrogate as the search progresses towards promising areas of the design space [67.40]. For this purpose, the Kriging method has received much interest, because it inherently considers confidence intervals of the predicted outputs. For a complete derivation of the Kriging method, readers are encouraged to follow the work of *Jones* [67.41] and *Forrester* et al. [67.40]. We provide a very brief introduction to the ordinary Kriging method, which expresses the unknown function $y(\boldsymbol{x})$ as,

$$y(\boldsymbol{x}) = \beta + z(\boldsymbol{x}) , \quad (67.11)$$

where $\boldsymbol{x} = [x_1, \ldots, x_n]$ is the data location, β is a constant global mean value, and $z(\boldsymbol{x})$ represents a local deviation at the data location $\boldsymbol{x}$ based on a stochastic process with zero-mean and variance σ^2 following the Gaussian distribution. The approximation $\hat{y}(\boldsymbol{x})$ is obtained from

$$\hat{y}(\boldsymbol{x}) = \hat{\beta} + \boldsymbol{r}^T \boldsymbol{R}^{-1}(\boldsymbol{Y} - \boldsymbol{1}\hat{\beta}) , \quad (67.12)$$

where $\hat{\beta}$ is the approximation of β, $\boldsymbol{R}$ is the correlation matrix, $\boldsymbol{r}$ is the correlation vector, $\boldsymbol{Y}$ is the training dataset of N observed samples at location $\boldsymbol{X}$, and $\boldsymbol{1}$ is a column vector of N elements of 1. The correlation matrix is a modification of the Gaussian basis function,

$$R(\boldsymbol{x}^i, \boldsymbol{x}^j) = \exp\left(-\sum_{k=1}^{n} \boldsymbol{\theta}_k |x_k^i - x_k^j|^2\right) , \tag{67.13}$$

where $\theta_k > 0$ is the k-th element of the correlation parameter $\boldsymbol{\theta}$. Following the work of *Jones* [67.41], the correlation parameter $\boldsymbol{\theta}$ (and hence the approximations $\hat{\beta}$ and $\hat{\sigma}^2$) are estimated by maximizing the concentrated ln-likelihood of the dataset $\boldsymbol{Y}$, which is an n-variable single-objective optimization problem, solved using a pattern search method. The accuracy of the prediction $\hat{y}$ at the unobserved location $\boldsymbol{x}$ depends on the correlation distance with sample points $\mathbf{X}$. The closer the location of $\boldsymbol{x}$ to the sample points, the more confidence in the prediction $\hat{y}(\boldsymbol{x})$. The measure of uncertainty in the prediction is estimated as

$$\hat{s}^2(\boldsymbol{x}) = \hat{\sigma}^2 \left[1 - \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r} + \frac{(1 - \mathbf{1}^T \boldsymbol{R}^{-1} \boldsymbol{r})^2}{\mathbf{1}^T \boldsymbol{R}^{-1} \mathbf{1}}\right] \tag{67.14}$$

if $\boldsymbol{x} \subset \boldsymbol{X}$, it is observed from (67.14) that $\hat{s}(\boldsymbol{x})$ reduces to zero.

67.3.4 Reference Point Screening Criterion

Training a Kriging model from a training dataset is time consuming and is of the order $O(N^3)$. Stratified sampling using a maximin Latin hypercube (LHS [67.52] is used to construct a global Kriging approximation $[\boldsymbol{X},\ \boldsymbol{Y}]$. The non-dominated subset of $\boldsymbol{Y}$ is then stored within the elitist archive. This ensures that candidates for global leadership have been precisely evaluated (or with negligible prediction error) and, therefore, offer no false guidance to other particles. Adopting the concept of individual-based control [67.42], Kriging predictions are then used to pre-screen each candidate particle after the population update (or after mutation) and subsequently flag them for precise evaluation or rejection. The Kriging model estimates a lower-confidence bound for the objective array as

$$\{\hat{f}_1(\boldsymbol{x}), \ldots, \hat{f}_m(\boldsymbol{x})\}_{lb} = [\{\hat{y}_1(\boldsymbol{x}) - \omega \hat{s}_1(\boldsymbol{x})\}, \ldots, \{\hat{y}_m(\boldsymbol{x}) - \omega \hat{s}_m(\boldsymbol{x})\}] , \tag{67.15}$$

where $\omega = 2$ provides a 97% probability for $\hat{f}_{lb}(\boldsymbol{x})$ to be the lower bound value of $\hat{f}(\boldsymbol{x})$. An approximation to the reference point distance, $\hat{d}_z(\boldsymbol{x})$, can thus be obtained using (67.5). This value, whilst providing a means of ranking each solution as a single scalar, also gives an estimate to the improvement that is expected from the solution. At time t, the archive member with the highest ranking according to (67.5) is recorded as $d_{\min}$. The candidate $\boldsymbol{x}$ may then be accepted for precise evaluation, and subsequent admission into the archive if $\hat{d}_z(\boldsymbol{x}) < d_{\min}$. Particles will thus be attracted towards the areas of the design space that provide the greatest resemblance to $\bar{z}$, and the direction of the search will remain consistent.

As the search begins in the explorative phase and the prediction accuracy of the surrogate model(s) is low, depending on the deceptivity of the objective landscape(s) there will initially be a large percentage of the swarm that is flagged for precise evaluation. Subsequently, as the particles begin to identify the preferred region and the prediction accuracy of the surrogate model(s) gradually increases, the screening criterion becomes increasingly difficult to satisfy, thereby reducing the number of flagged particles at each time step. To restrict saturation of the dataset used to train the Kriging models, a limit is imposed of $N = 200$ sample points where lowest ranked solutions according to (67.5) are removed.

67.4 Case Study: Airfoil Shape Optimization

The parameterization method and transonic flow solver described in the preceding section are now integrated within the Kriging-assisted UPMOPSO algorithm for an efficient airfoil design framework. The framework is applied to the re-design of the NASA-SC(2)0410 airfoil for robust aerodynamic performance. A three-objective constrained optimization problem is formulated, with $f_1 = C_d$ and $f_2 = -C_m$ for $M = 0.79$, $C_l = 0.4$, and $f_3 = \partial C_d / \partial M$ for the design range $M = [0.79, 0.82]$, $C_l = 0.4$. The lift constraint is satisfied internally within the solver, by allowing Fluent to determine the angle of incidence required. A constraint is imposed on the allowable thickness, which is defined through the parameter ranges (see Table 67.2) as approximately 9.75% chord. The reference point is logically selected as the NASA-SC(2)0410, in an attempt to improve on the performance characteristics of the airfoil, whilst still maintaining a similar level of compromise between the design objectives. The solution variance is controlled by $\delta = 5 \times 10^{-3}$.

The design application is segregated into three phases: pre-optimization and variable screening; optimization and; post-optimization and trade-off screening.

67.4.1 Pre-Optimization and Variable Screening

Global Kriging models are constructed for the aerodynamic coefficients from a stratified sample of $N = 100$ design points based on a Latin hypercube design. This sampling plan size is considered sufficient in order to obtain sufficient confidence in the results of the subsequent design variable screening analysis. Whilst a larger sampling plan is essential to obtain fairly accurate correlation, the interest here is to quantify the elementary effect of each variable to the objective landscapes. The global Kriging models are initially trained via cross-validation. The cross-validation curves for the Kriging models are illustrated in Fig. 67.10. The subscripts to the aerodynamic coefficients refer to the respective angle of incidence.

It is observed in Fig. 67.10 that the Kriging models constructed for the aerodynamic coefficients are able to reproduce the training samples with sufficient confidence, recording error margin values between 2 to 4%. It is hence concluded that the Kriging method is very adept at modeling complex landscapes represented by a limited number of precise observations.

To investigate the elementary effect of each design variable on the metamodeled objective landscapes, we present a quantitative design space visualization technique. A popular method for designing preliminary experiments for design space visualization is the screening method developed by *Morris* [67.53]. This algorithm calculates the elementary effect of a variable x_i and establishes its correlation with the objective space f as:

a) Negligible
b) Linear and additive
c) Nonlinear
d) Nonlinear and/or involved in interactions with x_j.

Table 67.2 NASA-SC(2)0410 airfoil results for the formulated objectives

Airfoil	Mach number, M	f_1	f_2	f_3
NASA-SC(2)0410	0.79	0.008708	0.1024	0.189625

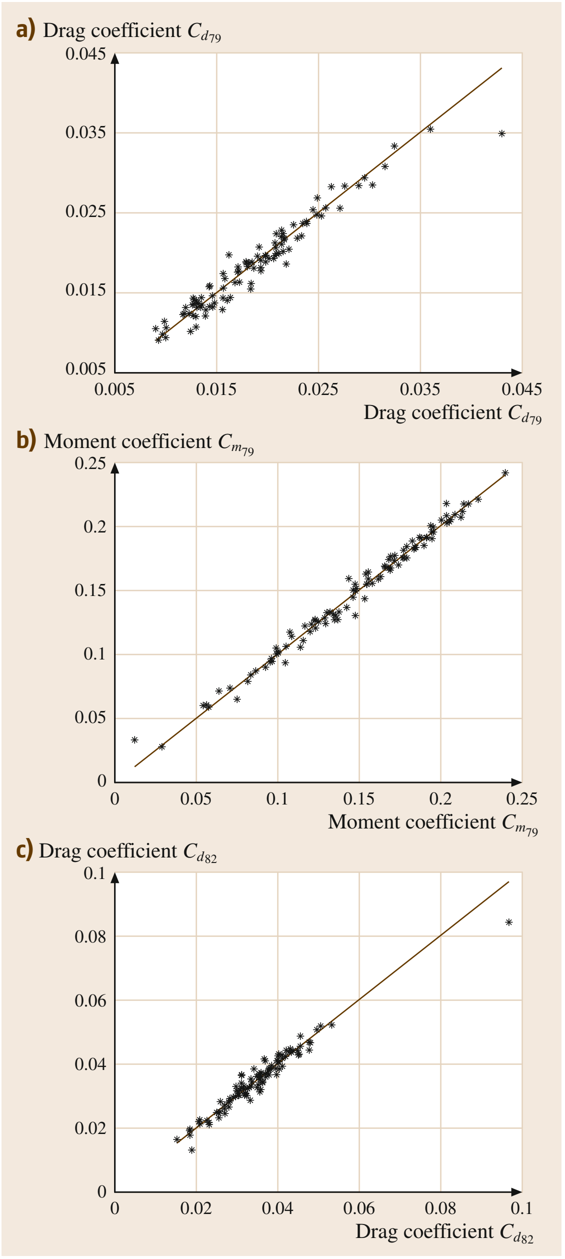


Fig. 67.10a–c Cross-validation curves for the constructed Kriging models. **(a)** Training sample for C_d at $M = 0.79$. **(b)** Training sample for C_m at $M = 0.79$. **(c)** Training sample for C_d at $M = 0.82$

In plain terminology, the Morris algorithm measures the sensitivity of the i-th variable to the objective landscape f. For a detailed discussion on the Morris al-

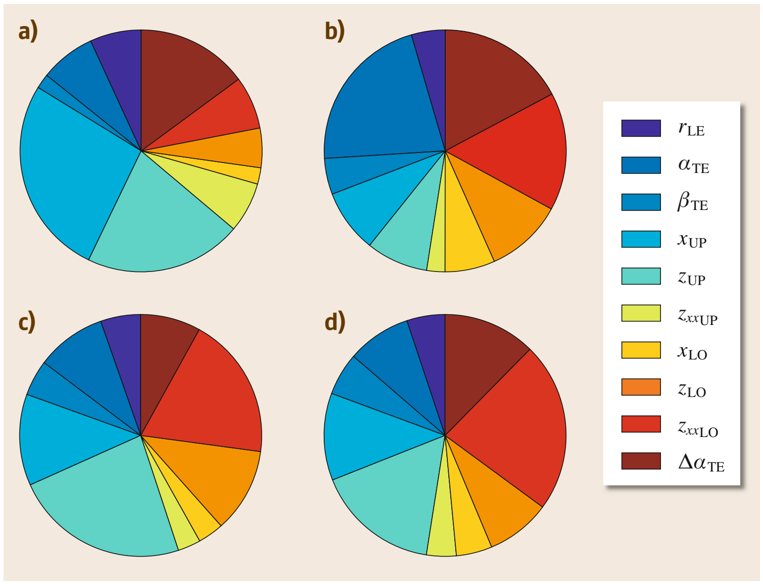


Fig. 67.11a–d Variable influence on aerodynamic coefficients (subscripts refer to the operating Mach number). **(a)** Drag $C_{d_{79}}$. **(b)** Moment $C_{m_{79}}$. **(c)** Drag $C_{d_{82}}$. **(d)** d_z

gorithm the reader is referred to *Forrester* et al. [67.40] and *Campolongo* et al. [67.54]. Presented here are the results of the variable screening analysis using the Morris algorithm for the proposed design application.

Figure 67.11 graphically shows the results obtained from the design variable screening study. It is immediately observed that the upper thickness coordinates have a relatively large influence on the drag coefficient for both design conditions. At higher Mach numbers the effect of the lower surface curvature $z_{xx\mathrm{LO}}$ is also significant. It is demonstrated, however, that the variables $z_{xx\mathrm{LO}}$ and α_{TE} have the largest effect on the moment coefficient – variables which directly influence the aft camber (and hence the aft camber) on the airfoil. These variables will no doubt shift the loading on the airfoil forward and aft, resulting in highly fluctuating moment values.

Similar deductions can be made by examining the variable influence on d_z shown in Fig. 67.11d. The variable influence on d_z is case specific and entirely dependent on the reference point chosen for the proposed optimization study. Since the value of d_z is a means of ranking the success of a multiobjective solution as one single scalar, variables may be ranked by influence, which is otherwise not possible when considering a multiobjective array. Preliminary conclusions to the priority weighting of the objectives to the reference point compromise can also be made. It is observed that the variable influence on d_z most closely resembles the plots of the drag coefficients $C_{d_{79}}$ and $C_{d_{82}}$, suggesting that the moment coefficient is of least priority for the preferred compromise. It is interesting to see that the trailing edge modification variable $\Delta\alpha_{\mathrm{TE}}$ is of particular importance for all design coefficients, which validates its inclusion in the subsequent optimization study.

67.4.2 Optimization Results

A swarm population of $N_{\mathrm{s}} = 100$ particles is flown to solve the optimization problem. The objective space is normalized for the computation of the reference point distance by $f_{\max} - f_{\min}$. Instead of specifying a maximum number of time steps, a computational budget of 250 evaluations is imposed. A stratified sample of $N = 100$ design points using an LHS methodology was used to construct the initial global Kriging approximations for each objective. A further 150 precise updates were performed over $t \approx 100$ time steps until the computational budget was breached. As is shown in Fig. 67.12a, the largest number of update points was recorded during the initial explorative phase. As the preferred region becomes populated and $\hat{s} \to 0$, the algorithm triggers exploitation, and the number of update points steadily reduces.

The UPMOPSO algorithm has proven to be very capable for this specific problem. Figure 67.12b features the progress of the highest ranked solution (i. e., $d_{\min}$) as the number of precise evaluations increase. The reference point criterion is shown to be proficient in filtering out poorer solutions during exploration, since it

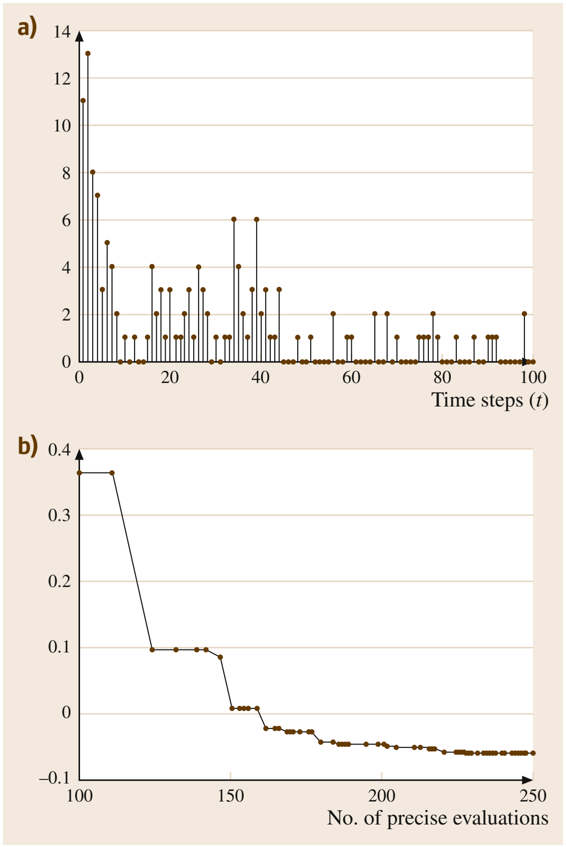


Fig. 67.12a,b UPMOPSO performance for transonic airfoil shape optimization. **(a)** History of precise updates. **(b)** Progress of most preferred solution

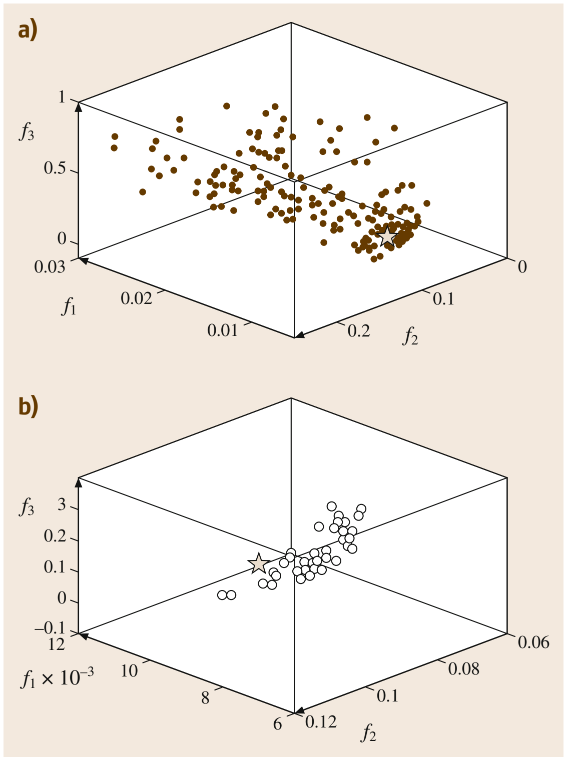


Fig. 67.13a,b Precise evaluations performed and the resulting non-dominated solutions. **(a)** Scatter plot of all precise evaluations. **(b)** Preferred region 250 evaluations

is only required to reach 50 update evaluations within 15% of $d_{\min}$, and to reach a further 50 evaluations within 3%. Furthermore, no needless evaluations as a result of the lower-bound prediction are performed during the exploitation phase. This conclusion is further complemented by Fig. 67.13a, as a distinct attraction to the preferred region is clearly visible. A total of 30 non-dominated solutions were identified in the preferred region, which are shown in Fig. 67.13b.

67.4.3 Post-Optimization and Trade-Off Visualization

The reference point distance also provides a feasible means of selecting the most appropriate solutions. For example, solutions may be ranked according to how well they represent the reference point compromise. To illustrate this concept, self-organizing maps (SOMs) [67.44] are utilized to visualize the interaction of the objectives with the reference point compromise. Clustering SOM techniques are based on a technique of unsupervised artificial neural networks that can classify, organize, and visualize large sets of data from a high to low-dimensional space [67.45]. A neuron used in this SOM analysis is associated with the weighted vector of m inputs. Each neuron is connected to its adjacent neurons by a neighborhood relation and forms a two-dimensional hexagonal topology [67.45]. The SOM learning algorithm will attempt to increase the correlation between neighboring neurons to provide a global representation of all solutions and their corresponding resemblance to the reference point compromise.

Each input objective acts as a neuron to the SOM. The corresponding output measures the reference point distance (i. e., the resemblance to the reference point compromise). A two-dimensional representation of the data is presented in Fig. 67.14, organized by six SOM-ward clusters. Solutions that yield negative d_z values

Table 67.3 Preferred airfoil objective values with measure of improvement

Airfoil	f_1	f_2	f_3
NASA-SC(2)0410	0.008708	0.1024	0.189625
Preferred design	0.008106	0.0933	0.168809
% Improvement	6.9	8.8	10.9

indicate success in the improvement over each aspiration value. Solutions with positive d_z values do not surpass each aspiration value but provide significant improvement in at least one other objective. Each of the node values represent one possible Pareto-optimal solution that the designer may select. The SOM chart colored by d_z is a measure of how far a solution deviates from the preferred compromise. However, the concept of the preferred region ensures that only solutions that slightly deviate from the compromise dictated by $\bar{z}$ are identified. Following the SOM charts, it is possible to

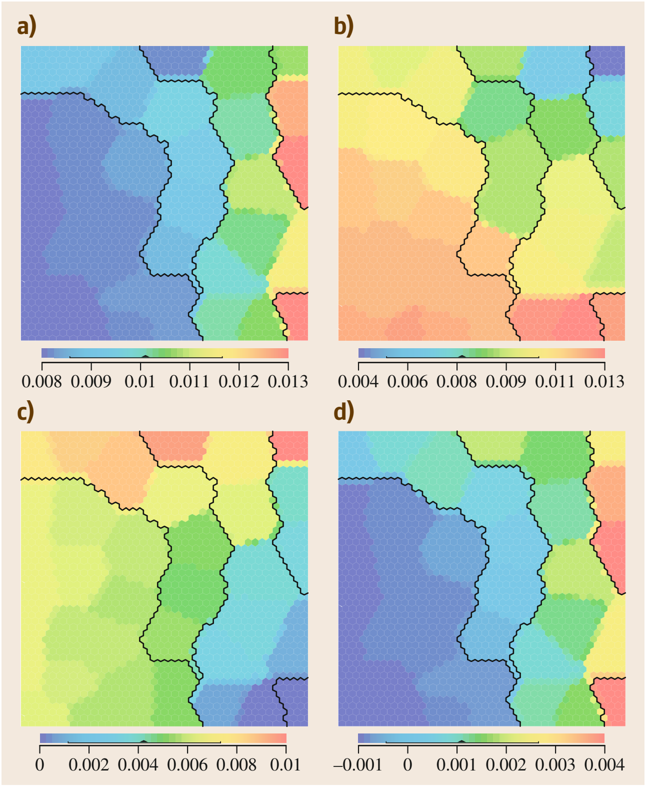


Fig. 67.14 SOM charts to visualize optimal trade-offs between the design objectives. **(a)** f_1, **(b)** f_2, **(c)** f_3, **(d)** d_z

visualize the preferred compromise between the design objectives that is obtained. The chart of d_z closely follows the f_1 chart, which suggests that this objective has the highest priority. If the designer were slightly more inclined towards another specific design objective, then solutions that perhaps place more emphasis on the other objectives should be considered. In this study, the most preferred solution is ideally selected as the highest ranked solution according to (67.5).

67.4.4 Final Designs

Table 67.3 shows the objective comparisons with the NASA-SC(2)0410. Of interest to note is that the most active objective is f_1, since the solution which provides the minimum d_z values also provides the minimum f_1 value. This implies that the reference point was sit-

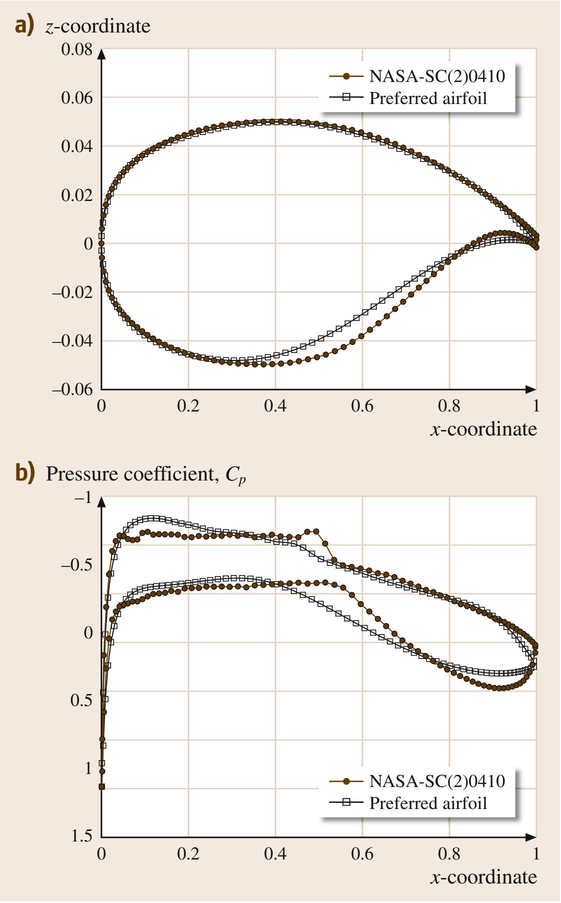


Fig. 67.15a,b The most preferred solution observed by the UPMOPSO algorithm. **(a)** Preferred airfoil geometry. **(b)** C_p distributions for $M = 0.79$, $C_l = 0.4$ ◀

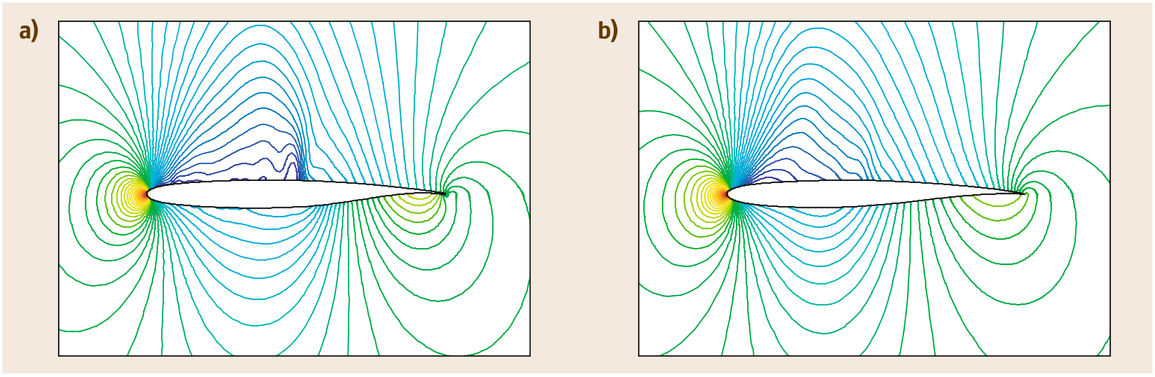


Fig. 67.16a,b Pressure contours for design condition of $M = 0.79$, $C_l = 0.4$. **(a)** NASA-SC(2)0410, **(b)** Preferred airfoil

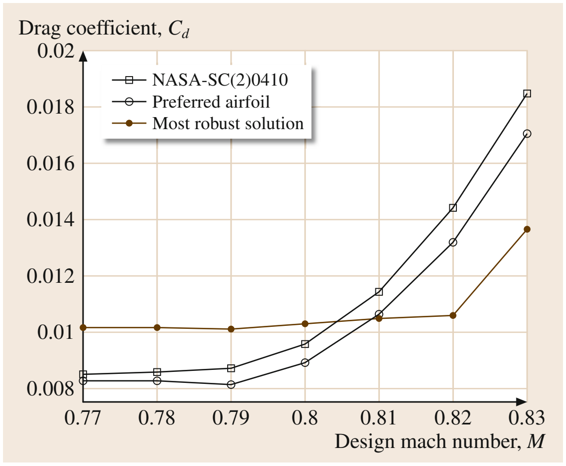


Fig. 67.17 Drag rise curves for $C_l = 0.4$

uated near the f_1 Pareto boundary. Of the identified set of Pareto-optimal solutions, the largest improvements obtained in objectives f_2 and f_3 are 36.4 and 91.6%, respectively, over the reference point. The preferred airfoil geometry is shown in Fig. 67.15a in comparison with the NASA-SC(2)0410. The preferred airfoil has a thickness of 9.76% chord and maintains a moderate curvature over the upper surface. A relatively small aft curvature is used to generate the required lift, whilst reducing the pitching moment.

Performance comparisons between the NASA-SC(2)0410 and the preferred airfoil at the design condition of $M = 0.79$ can be made from the static pressure contour output in Fig. 67.16, and the surface pressure distribution of Fig. 67.15b. The reduction in C_d is attributed to the significantly weaker shock that appears slightly upstream of the supercritical shock position. The reduction in the pitching moment is clearly visible from the reduced aft loading. Along with the improvement at the required design condition, the preferred airfoil exhibits a lower drag rise by comparison, as is shown in Fig. 67.17. There is a notable decline in the drag rise at the design condition of $M = 0.79$, and the drag is recorded as lower than the NASA-SC(2)0410, even beyond the design range. Also visible is the solution that provides the most robust design (i. e., min f_3). The most robust design is clearly not obtained at the expense of poor performance at the design condition, due to the compromising influence of $\bar{z}$. If the designer were interested in obtaining further alternative solutions which provide greater improvement in either objective, it would be sufficient to re-commence (at the current time step) the optimization process with a larger value of δ, or by relaxing one or more of the aspiration values, $\bar{z}_i$.

67.5 Conclusion

In this chapter, an optimization framework has been introduced and applied to the aerodynamic design of transonic airfoils. A surrogate-driven multiobjective particle swarm optimization algorithm is applied to navigate the design space to identify and exploit preferred regions of the Pareto frontier. The integration of all components of the optimization framework is entirely achieved through the use of a reference point distance metric which provides a scalar measure of the preferred interests of the designer. This effectively allows for the scale of the design space to be reduced, confining it to the interests reflected by the designer.

The developmental effort that is reported on here is to reduce the often prohibitive computational cost of multiobjective optimization to the level of practical affordability in computational aerodynamic design. A concise parameterization model was considered to perform the necessary shape modifications in conjunction with a Reynolds-averaged Navier–Stokes flow solver. Kriging models were constructed based on a stratified sample of the design space. A pre-optimization visualization tool was then applied to screen variable elementary influence and quantify its relative influence to the preferred interests of the designer. Initial design drivers were easily identified and an insight to the optimization landscape was obtained. Optimization was achieved by driving a surrogate-assisted particle swarm towards a sector of special interest on the Pareto front, which is shown to be an effective and efficient mechanism. It is observed that there is a distinct attraction towards the preferred region dictated by the reference point, which implies that the reference point criterion is adept at filtering out solutions that will disrupt or deviate from the optimal search path.

Non-dominated solutions that provide significant improvement over the reference geometry were identified within the computational budget imposed and are clearly reflective of the preferred interest. A post-optimization data mining tool was finally applied to facilitate a qualitative trade-off visualization study. This analysis provides an insight into the relative priority of each objective and their influence on the preferred compromise.

References

67.1 T.E. Labrujère, J.W. Sloof: Computational methods for the aerodynamic design of aircraft components, Annu. Rev. Fluid Mech. **51**, 183–214 (1993)

67.2 R.M. Hicks, P.A. Henne: Wing design by numerical optimization, J. Aircr. **15**(7), 407–412 (1978)

67.3 J. Kennedy, R.C. Eberhart, Y. Shi: *Swarm Intelligence* (Morgan Kaufmann, San Francisco 2001)

67.4 J. Kennedy, R.C. Eberhart: Particle swarm optimization, Proc. IEEE Intl. Conf. Neural Netw. (1995) pp. 1942–1948

67.5 I.C. Trelea: The particle swarm optimization algorithm: Convergence analysis and parameter selection, Inform. Proces. Lett. **85**(6), 317–325 (2003)

67.6 C.M. Fonseca, P.J. Fleming: Multiobjective optimization and multiple constraint handling with evolutionary algorithms. A unified formulation, IEEE Trans. Syst. Man Cybern. A **28**(1), 26–37 (1998)

67.7 M. Drela: Pros and cons of airfoil optimization, Front. Comput. Fluid Dynam. **19**, 1–19 (1998)

67.8 W. Li, L. Huyse, S. Padula: Robust airfoil optimization to achieve drag reduction over a range of mach numbers, Struct. Multidiscip. Optim. **24**, 38–50 (2002)

67.9 M. Nemec, D.W. Zingg, T.H. Pulliam: Multipoint and multi-objective aerodynamic shape optimization, AIAA J. **42**(6), 1057–1065 (2004)

67.10 I. Das, J.E. Dennis: A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems, Struct. Optim. **14**, 63–69 (1997)

67.11 R.T. Marler, J.S. Arora: Survey of multi-objective optimization methods for engineering, Struct. Multidiscip. Optim. **26**, 369–395 (2004)

67.12 C.M. Fonseca, P.J. Fleming: An overview of evolutionary algorithms in multiobjective optimization, Evol. Comput. **3**, 1–16 (1995)

67.13 K. Deb: *Multi-Objective Optimization Using Evolutionary Algorithms* (Wiley, New york 2001)

67.14 C.A. Coello Coello: *Evolutionary Algorithms for Solving Multi-Objective Problems* (Springer, Berlin, Heidelberg 2007)

67.15 S. Obayashi, D. Sasaki, Y. Takeguchi, N. Hirose: Multiobjective evolutionary computation for supersonic wing-shape optimization, IEEE Trans. Evol. Comput. **4**(2), 182–187 (2000)

67.16 A. Vicini, D. Quagliarella: Airfoil and wing design through hybrid optimization strategies, AIAA J. **37**(5), 634–641 (1999)

67.17 A. Ray, H.M. Tsai: Swarm algorithm for single and multiobjective airfoil design optimization, AIAA J. **42**(2), 366–373 (2004)

67.18 M. Ehrgott, X. Gandibleux: *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys* (Kluwer, Boston 2002)

67.19 M.R. Sierra, C.A. Coello Coello: Multi-objective particle swarm optimizers: A survey of the state-of-the-art, Int. J. Comput. Intell. Res. **2**(3), 287–308 (2006)

67.20 A.P. Engelbrecht: *Fundamentals of Computational Swarm Intelligence* (Wiley, New York 2005)

67.21 J. Knowles, D. Corne: Approximating the nondominated front using the Pareto archived evolution strategy, Evol. Comput. **8**(2), 149–172 (2000)

67.22 C.A. Coello Coello, M.S. Lechuga: Mopso: A proposal for multiple objective particle swarm optimization, IEEE Cong. Evol. Comput. (2002) pp. 1051–1056

67.23 S. Mostaghim, J. Teich: The role of ϵ-dominance in multi-objective particle swarm optimization methods, IEEE Cong. Evol. Comput. (2003) pp. 1764–1771

67.24 J.E. Fieldsend, S. Singh: A multi-objective algorithm based upon particle swarm optimisation, an efficient data structure and turbulence, U.K Workshop Comput. Intell. (2002) pp. 37–44

67.25 X. Li: A non-dominated sorting particle swarm optimizer for multiobjective optimization, Genet. Evol. Comput. Conf. (2003) pp. 37–48

67.26 X. Li: Better spread and convergence: Particle swarm multiobjective optimization using the maximin fitness function, Genet. Evol. Comput. Conf. (2004) pp. 117–128

67.27 M.R. Sierra, C.A. Coello Coello: Improving pso-based multi-objective optimization using crowding, mutation and ϵ-dominance, Lect. Notes Comput. Sci. **3410**, 505–519 (2005)

67.28 C.R. Raquel, P.C. Naval: An effective use of crowding distance in multiobjective particle swarm optimization, Genet. Evol. Comput. Conf. (2005) pp. 257–264

67.29 C.A. Coello Coello: Handling preferences in evolutionary multiobjective optimization: A survey, IEEE Cong. Evol. Comput. (2000) pp. 30–37

67.30 L. Rachmawati, D. Srinivasan: Preference incorporation in multi-objective evolutionary algorithms: A survey, IEEE Cong. Evol. Comput. (2006) pp. 962–968

67.31 C.M. Fonseca, P.J. Fleming: Handling preferences in evolutionary multiobjective optimization: A survey, Proc. IEEE 5th Int. Conf. Genet. Algorithm (1993) pp. 416–423

67.32 K. Deb: Solving goal programming problems using multi-objective genetic algorithms, IEEE Cong. Evol. Comput. (1999) pp. 77–84

67.33 L. Thiele, P. Miettinen, P.J. Korhonen, J. Molina: A preference-based evolutionary algorithm for multobjective optimization, Evol. Comput. **17**(3), 411–436 (2009)

67.34 K. Deb, J. Sundar: Reference point based multi-objective optimization using evolutionary algorithms, Genet. Evol. Comput. Conf. (2006) pp. 635–642

67.35 K. Deb, A. Kumar: Interactive evolutionary multi-objective optimization and decision-making using reference direction method, Genet. Evol. Comput. Conf. (2007) pp. 781–788

67.36 K. Deb, A. Kumar: Light beam search based multi-objective optimization using evolutionary algorithms, Genet. Evol. Comput. Conf. (2007) pp. 2125–2132

67.37 U.K. Wickramasinghe, X. Li: Integrating user preferences with particle swarms for multi-objective optimization, Genet. Evol. Comput. Conf. (2008) pp. 745–752

67.38 U.K. Wickramasinghe, X. Li: Using a distance metric to guide pso algorithms for many-objective optimization, Genet. Evol. Comput. Conf. (2009) pp. 667–674

67.39 A.J. Keane, P.B. Nair: *Computational Approaches for Aerospace Design: The Pursuit of Excellence* (Wiley, New York 2005)

67.40 A. Forrester, A. Sóbester, A.J. Keane: *Engineering Design Via Surrogate Modelling: A Practical Guide* (Wiley, New York 2008)

67.41 D.R. Jones: A taxomony of global optimization methods based on response surfaces, J. Glob. Optim. **21**, 345–383 (2001)

67.42 Y. Jin: A comprehensive survey of fitness approximation in evolutionary computation, Soft Comput. **9**(1), 3–12 (2005)

67.43 R.M. Greenman, K.R. Roth: High-lift optimization design using neural networks on a multi-element airfoil, J. Fluids Eng. **121**(2), 434–440 (1999)

67.44 T. Kohonen: *Self-Organizing Maps* (Springer, Berlin, Heidelberg 1995)

67.45 S. Jeong, K. Chiba, S. Obayashi: Data mining for aerodynamic design space, J. Aerosp. Comput. Inform. Commun. **2**, 452–469 (2005)

67.46 W. Song, A.J. Keane: A study of shape parameterisation methods for airfoil optimisation, Proc. 10th AIAA/ISSMO Multidiscip. Anal. Optim. Conf. (2004)

67.47 H. Sobjieczky: Parametric airfoils and wings, Notes Numer. Fluid Mech. **68**, 71–88 (1998)

67.48 A. Jahangirian, A. Shahrokhi: Inverse design of transonic airfoils using genetic algorithms and a new parametric shape model, Invers. Probl. Sci. Eng. **17**(5), 681–699 (2009)

67.49 P.R. Spalart, S.R. Allmaras: A one-equation turbulence model for aerodynamic flows, Rech. Aerosp. **1**, 5–21 (1992)

67.50 M. Clerc, J. Kennedy: The particle swarm – explosion, stability, and convergence in a multidimensional complex space, IEEE Trans. Evol. Comput. **6**(1), 58–73 (2002)

67.51 D. Bratton, J. Kennedy: Defining a standard for particle swarm optimization, IEEE Swarm Intell. Symp. (2007) pp. 120–127

67.52 M.D. Mckay, R.J. Beckman, W.J. Conover: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Invers. Prob. Sci. Eng. **21**(2), 239–245 (1979)

67.53 M.D. Morris: Factorial sampling plans for preliminary computational experiments, Technometrics **33**(2), 161–174 (1991)

67.54 F. Campolongo, A. Saltelli, S. Tarantola, M. Ratto: *Sensitivity Analysis in Practice* (Wiley, New York 2004)