Report no. 97/07

Aerospace design: a complex task

M. B. Giles

These lecture notes, prepared for the 1997 VKI Lecture Course on Inverse Design, identify some of the complexities inherent in the design of aeroengines and aircraft. It is argued that to handle the geometric complexities requires a computational representation which is hierarchical and based on parametric solids at the lowest level. On top of this can be built a tightly coupled two-level design system which uses an integrated set of multidisciplinary analysis packages which are also hierarchical, with differing levels of approximation and computational cost appropriate to preliminary design and detailed component design.

A range of different numerical approaches to design are outlined, and their strengths and weaknesses are compared. Other related topics such as distributed computing, risk management and strategic research planning are also discussed.

Key words and phrases: design, CAD/CAM, optimisation, sensitivity analysis, adjoint methods, genetic algorithms

This work was supported by Rolls-Royce plc and EPSRC.

Oxford University Computing Laboratory
Numerical Analysis Group
Wolfson Building
Parks Road
Oxford, England OX1 3QD
http: //www.comlab.ox.ac.uk
email: giles@comlab.oxford.ac.uk

September, 1997

1 Introduction

1.1 Design

The objective of engineering analysis, the mathematical and computational modelling of an engineering product, is not to determine the behaviour of a single product; if this were the case it would be simpler and cheaper to take the first manufactured product and test it extensively. The real objective of engineering analysis is to predict the behaviour of a product, and then use that information to design a better product. Often a whole sequence of improved designs is created, with very few being actually manufactured and tested experimentally. Given that experimental testing is often very time-consuming and expensive, the extensive use of computational engineering analysis can greatly reduce the time and cost of the design process.

Although design is the ultimate objective of engineering analysis, most academic research in the past 20 years has been directed towards the simpler task of modelling the behaviour of specific aspects of the engineering system. This research, coupled with enormous increases in computational power, has led to the development of three-dimensional modelling and numerical methods in CFD, combustion modelling, finite element structural analysis and computational electromagnetics. Despite limitations due to modelling approximations, such as turbulence and transition modelling, these methods are now capable of analysing the behaviour of many aspects of engineering systems with an accuracy which is acceptable for most engineering purposes. They are being widely used within industry, particularly in aeronautical engineering, and are responsible for an improvement in product quality and a shortening of the design cycle.

However, the use of current engineering analysis methods in designing complex systems relies very heavily on the knowledge and intuition of the designer. Individual analyses of a specific design reveal its performance; it is then up to the designer to note shortcomings in the performance, identify possible causes and then formulate remedial design changes. When designing very large complex engineering systems, the situation can be worse, because there is then a team of people responsible for the design, usually with different people responsible for different aspects of the performance of the system. Any design will involve trade-offs between the different aspects, but an intuitive understanding of the trade-offs becomes increasingly hard as the complexity of the system and its multidisciplinary nature increase.

Although there is still scope for significant improvement in the accuracy and capabilities of the individual analysis tools, my belief is that the greatest advance in engineering analysis in the next 20 years will come from the harnessing of the individual tools within integrated design systems. The aim of these is not to replace the designer by a 'black box' producing optimal designs, but to provide the designer with a flexible environment in which to define the design space (the parameters which define the design), explore the trade-offs, and investigate novel designs which may well lie outside the designer's range of intuition based on past experience. These design systems will also lead to a further shortening of the design cycle, reduce the risk, reduce costs (in part through more automation of the analysis process) and allow a better targetting of strategic research.

1.2 Aeronautical engineering

The systems being designed in aeronautical applications are among the most complex in engineering. Not counting the smallest components such as nuts, bolts and rivets, the aeroengines and the rest of an aircraft may have hundreds of thousands of components, with over a million important design parameters and many more which are less important. Furthermore, a complete unsteady simulation of the workings of an aeroengine including all fluid dynamics, structural stresses and vibration, heat transfer, etc., would take on the order of a year on on the computing resources of a typical large aerospace company. A complete simulation of an aircraft from take-off, to cruising at altitude and then landing would be similarly expensive.

It is this level of complexity in aeronautical engineering which rules out the general use of methods such as genetic algorithms for optimisation. Such methods have been very usefully applied to problems with very few design parameters and very low computational analysis costs. They may also be very appropriate for preliminary design optimisation, as will be discussed later, but their computational cost means they can not be applied to optimisation of the entire system using all of the design parameters and the full computational simulation. Instead, a hierarchical approach is needed, both in handling the design space (the set of design parameters) and in modelling the engineering system. The complexity of the design space also provides a strong argument for the use of a design system which keeps track of all of the design variables, allowing the designer to investigate the effect of certain 'active' design variables to try to improve the design while satisfying all of the many constraints that a design always has.

Another important features of aeronautical engineering is its strongly multidisciplinary nature. In wing design, the optimal thickness of the wing is a trade-off between aerodynamic performance (which decreases as the thickness increases) and penalties due to weight (which also decreases as the thickness increases because the bending moment can be sustained using an thinner airfoil skin). Similarly, the optimal trailing edge thickness of a turbine blade is a compromise between aerodynamic performance (which decreases as the thickness increases) and structural integrity due to both static and dynamic loading (which improves with thickness).

This multidisciplinary aspect of the design problem is a major challenge, because there are few engineers with the necessary knowledge of each of the disciplines, and fewer still with a good intuitive grasp of the various compromises to be reached in coming to a good overall design. Here again is where an integrated design system has much to offer, for example giving the designer the necessary knowledge about how variations in the thickness affect the aerodynamic and structural performance.

The development of these integrated design systems will not be an easy task. Much of it is a complex software engineering task, linking together different components modules from CAD/CAM, grid generation, CFD, structural analysis, etc. The difficulty comes from the lack of well accepted standards for the interfaces, together with the fact that those carrying out the software engineering may have difficulty appreciating the goals and challenges of engineering design. Academics (such as myself) may be able to help to define the long-term vision of what could be accomplished by such systems, as well as working on specific pieces of it such as genetic algorithms and adjoint methods. There is also a substantial body of academic research on engineering design, most of which seems oriented towards simple engineering systems, but may still be relevant to much larger systems.

However, much of the work to be done on engineering design systems for large complex applications is best done in industry, directed by those who best understand the realities of engineering design and how the current design processes may be improved. Many of the challenges to be faced are a direct consequence of the complexity of the engineering application, and it is not feasible to duplicate applications of this magnitude in academia; the only option is for academics to be closely involved in the research be carried on within industry.

1.3 These lecture notes

My objective is to explain what I see as being the sources of complexity in aeronautical engineering, and how this complexity can be tackled by the use of appropriate hierarchies in both EPD (electronic product definition) and analysis/design methods. I also survey the main numerical approaches to design, discussing their relative strengths and weaknesses.

As I write these notes, I am conscious of the fact that I do not have any actual experience of designing an engineering product. Therefore, I cannot pretend to be a designer or to understand all of the issues that concern a designer. However, I know a lot about mathematical modelling and computational engineering analysis, and I have the benefit of almost 20 years association with Rolls-Royce, developing CFD tools which are used by designers. Therefore, I will present my vision of the engineering design system of the future, knowing that it is quite possible I will omit some features that designers may consider crucial. I also recommend that readers consult Reference [17] for an industrial viewpoint on design and the limitations of some academic approaches to the subject.

Table 1: Hierarchical definition of a turbine vane

Level 1	number of blades, hub/tip radius, throat area, inflow/outflow angles, mass flow
Level 2	camber/thickness distribution, cooling mass flow
Level 3	geometry of fillets at hub and tip junctions
Level 4	details of film cooling holes and slots, temperature of inflow and cooling flow
Level 5	alloy type and thermal properties

Table 2: Hierarchical definition of an aircraft

Level 1	aircraft weight, wingspan, cruising speed
Level 2	wing/fuselage geometry
Level 3	engines, tail, winglets
Level 4	high-lift flaps & slats, take-off climb rate
Level 5	control surfaces, fairings, desired roll rate

2 Managing complexity

2.1 Hierarchical EPD

As explained in the Introduction, the problem with electronic product definition (EPD) in aeronautical engineering is the magnitude of the definition. There may be hundreds of thousands of components, and well over a million important design parameters. This is far too large a number to be handled easily by the designer.

Furthermore, the complete product definition contains a level of detail which is undesirable for much engineering analysis. An example of this in turbomachinery is the cooling holes in a high pressure turbine. The geometry of these must be contained in the EPD database for manufacturing purposes, but when computing the viscous flow in the blade passage and the resulting heat transfer to the blade it is usual to ignore the details of the cooling holes and simply model the coolant injection through a transpiration boundary condition. An even more extreme example for aircraft is the rivets in the skin of the wing. Again, the details of these must be contained in the EPD database for manufacturing purposes, but when generating an unstructured grid for a CFD calculation it would be ridiculous to resolve the details of the flow over each rivet head.

The solution to these problems is to use a hierarchical EPD database in which every component is defined at a number of different levels of detail. Tables 1 and 2 illustrate possible hierarchical representations of a high pressure turbine vane and a complete aircraft. In each case, the level 1 representation is the most basic, with higher levels adding successively more detail. Neither table is complete; the rivets on the aircraft may not appear until level 20! Note also that both tables contain data which is not geometric; variables such as operating conditions, material types and cooling flow rates are important design parameters in addition to the more obvious geometric parameters.

An engineering analysis tool will interface to whichever level of the EPD is most appropriate. In the case of the turbine vane, a CFD code might interface at level 2, treating the blade/hub and blade/tip junction as sharp corners, and modelling the film cooling as a distributed mass source. A stress analysis package would need to interface at level 3 or higher since the fillet geometry is needed to calculate the correct stresses in the corners. A thermal analysis of the solid would need to go to level 5.

In coupled multidisciplinary applications, such as aeroelasticity or combined aerothermal analysis, it is possible that the applications being coupled may be working with different representations of the component. For example, suppose one wishes to perform a combined aerothermal analysis of the turbine vane, performing a Navier-Stokes CFD analysis of the flow in the vane passage and coupling this (through matching the surface temperature and heat flux) to a thermal analysis of the vane. If the CFD code is working with the level 2 definition while the solid thermal analysis package is working with level 5, there will be subtle differences in the surface geometry because of the absence or presence of the fillets. Moreover, the computational grids used for each analysis are likely to be quite different. Both of these problems raise the question of how to link the two packages. The solution is to do so parametrically. An arbitrary point on the surface of the vane can be referenced parametrically by (ξ_1, ξ_2) where ξ_1 is the fractional distance from hub to tip and ξ_2 is the fractional distance around the vane. The interface conditions between the two codes can then be defined as matching temperature and heat flux at corresponding (ξ_1, ξ_2) points. This will still require interpolation from one grid to another but this is a relatively straightforward task.

An important aspect of a hierarchical EPD is that each level is defined relative to the ones below, so that design changes at one level are ‘inherited’ by those above. A good example of this is the cooling holes in the high pressure turbine vane. If the designer changes the blade profile in level 2, then the cooling holes in level 4 are automatically adjusted accordingly. In practice, this may be accomplished by defining the (ξ_1, ξ_2) location of each hole together with its diameter and its angle relative to the surface. Its length would be implicitly defined by its intersection with the internal cooling plenum, which would itself need to be defined relative to the blade profile. This ‘inheritance’ feature of a hierarchical EPD database means that all of the parameters at each level of representation are design parameters which may perhaps be altered by the design system.

2.2 Hierarchical design

The hierarchical EPD approach is a convenient way to organise the very large number of design parameters in a typical aeronautical engineering application. However, on its own it does not address the other problem identified in the Introduction, the huge computational cost of analysing the entire engineering system and investigating the effect of all of the design parameters.

To tackle this requires a hierarchical approach to the design process as well. In most aerospace companies design is carried out at two levels, preliminary design and detailed component design. The preliminary design group considers the engine or aircraft as an entire system, thinking about the customers' requirements, sizing the major components, deciding which subsystems to retain from previous products, aiming to maximise profit over the lifetime of the entire project. When trying to optimise the overall configuration during the preliminary design process the system is modelled at a very approximate level using a considerable amount of empiricism based on previous experience. It may even assume future advances in component technology. This approximate modelling, working with the lowest levels in the hierarchical EPD database with a strictly limited number of fundamental design parameters means that the cost of simulating the entire system is reduced to a few minutes, at most. This allows a thorough investigation of the global trade-offs influencing the overall system configuration. At the conclusion of the preliminary design process, many crucial design decisions have been made. In the case of aeroengine design, this would include engine thrust, mass flow, fan radius, the number of compressor and turbine stages, and the approximate size and pressure ratio of each blade row. In the case of aircraft design, it would include the aircraft weight and cruising speed, number of engines and their thrust, ing span and aspect ratio, approximate sizes of control surfaces and high-lift flaps and slats.

The second stage in the design process is component design. A number of different design teams are given responsibility for the design of the components within a particular subsystem. For example, the Turbine Group would be responsible for the development of the turbine stages in an aeroengine. Design responsibility may be further broken down so that just three people, an aerodynamicist, a structural analyst and an expert in blade cooling may design the high pressure turbine vane, under the eye of the designer with overall responsibility for the turbine. At this level of design, the design intent for each component has been fairly tightly specified by the preliminary design group, and many constraints have been imposed. The task of the component design team is to fulfil the design intent as well as possible (good aerodynamic performance, good structural behaviour, low weight, low cost of manufacture and maintenance, etc.) subject to the constraints. To a large extent, this is a matter of shape optimisation, the non-geometric design parameters having been set in preliminary design.

The analysis tools which are used at this level are usually three-dimensional

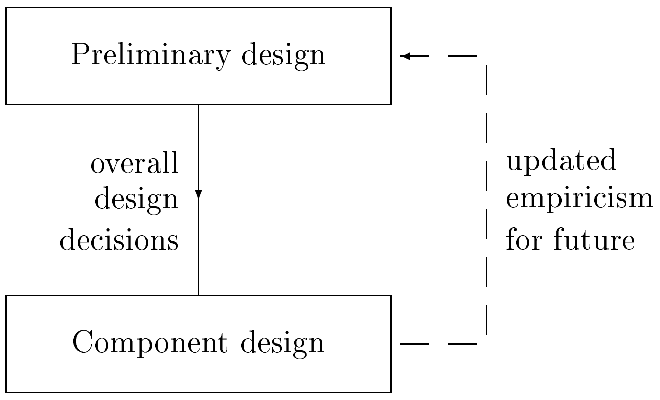

Figure 1: Current sequential two-level design process

CFD and structural analysis packages, with execution times ranging from a few minutes to many hours. These packages will work with the higher levels of the EPD database, using the much more complete definitions of each component. Sometimes, more approximate analysis methods may be used, such as 2D analyses, inviscid flow methods, even simple panel methods. The advantage of these methods is their very low computational cost, often measured in seconds rather than minutes, but this comes at the expense of reduced modelling accuracy so the designer must decide on the level of accuracy required. In some cases, the accuracy of the more approximate methods can be improved by the addition of ‘corrections’ (e.g. loss or drag, deviation angles) derived from the more accurate methods. This third form of hierarchy, hierarchical analysis, is already well-established in industrial practice and will not be discussed further.

As described above, and illustrated in Figure 1, the current hierarchical design approach is sequential, preliminary design followed by component design. Except in exceptional circumstances, the decisions made in preliminary design are not changed during component design. This is due to preliminary design being firmly based on empiricism from past experience, so major surprises are unlikely to arise during the component design process. Indeed, at the end of the component design process the empiricism in the preliminary design modelling should be updated to improve the preliminary design for the next major project.

There are two drawbacks to this sequential design process. The first is that its success depends on the new design not being too different from past designs, so that the empiricism in the modelling remains valid. This makes it very difficult to develop radically new designs, but it may be argued that in both aeroengine and aircraft design it is unlikely that a radically new design will be significantly better than existing designs. Therefore, the less risky nature of the current evolutionary process makes it commercially desirable. The second drawback of the current process is perhaps more mundane, but it is also probably more important. The empiricism in the preliminary design system represents the collective experience of past projects, but no two projects are ever identical. Even if the customer requirements are identical, technological advances mean that the best engine or aircraft of today would be different from that designed twenty years ago. To some extent this technological progress can be accounted for in the empiricism, but inevitably it remains true that the preliminary design process carries out its overall system optimisation and makes crucial design decisions based on only an approximate model of the system.

In the future I believe there will be a shift to a more tightly-coupled two-level design system, as illustrated in Figure 2. The overall system design will begin, as now, with a preliminary design based on past empiricism. This will then provide the starting point for the detailed component design. The change from the current sequential design process is that at this point data will be fed back into the overall system design, updating its empiricism based on the results of the detailed engineering analyses performed during the component design. This will allow further refinement of the overall system performance by fine-tuning some of the global trade-offs. Ideally, this design cycle would be repeated a number of times, with the component design responsible for the shape optimisation of specific components from a 'local' viewpoint, while the system design is concerned with overall optimisation of the major sub-systems based on a global viewpoint. Further in the future it is even possible that additional levels of optimisation may be introduced [1]. For example, in designing turbine blades, 2D profile optimisation to minimise boundary layer growth may be added as an additional level of optimisation after the basic 3D shape has already been optimised. [1]

The main reason a tightly coupled design system is not used today is time. The design time for an engine or aircraft project is strictly limited. There are very strong commercial pressures to bring a product to market as quickly as possible, even if this involves sacrificing a certain amount of performance because of the lack of time to investigate all design options. Spending more time on refining a design also has manpower and experimental testing costs; these have to be weighed against the possible benefits to be gained. The key to the successful adoption of a tightly coupled design system in the future lies in software engineering and ever increasing computational power. Good software engineering will minimise the time spent by designers in the coupling between system design and component design. The continuing doubling of computational power every 18-24 months will ensure decreasing execution times for the more expensive analysis tools, allowing more cycles of the coupled design process to be completed within a given time.

[1]Readers familiar with numerical methods might like to note that this hierarchical design is similar to multigrid in that the very approximate representation of the system (the 'coarse grid') is used for global tradeoffs and the design of the overall system, while the fully detailed representation of each component (the 'fine grid') is used for the detailed design of each individual component.

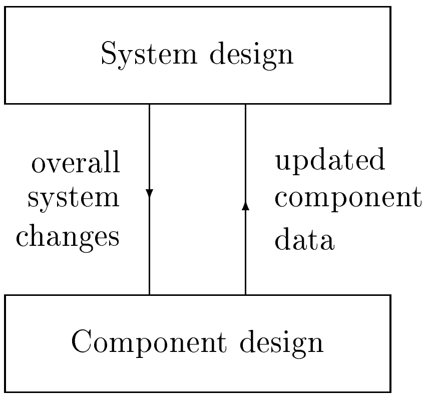

Figure 2: Future tightly-coupled two-level design process

2.3 Parametric variations and grid generation

In the design system, geometric design parameters will be handled by considering changes to their values and determining, in a nonlinear or linear manner, the effect of those changes on the engineering system. To properly appreciate the later discussion of the different numerical approaches, it is important to understand at a more detailed level how these parametric variations change the geometry of the component and how those geometric changes change the resulting computational grids used by the analysis packages.

In CAD systems, objects used to be defined by a collection of surfaces. For example, a turbine vane and the hub and tip annuli to which it is attached could be defined by a set of bi-cubic spline patches. However, in practice there were problems with this approach because at intersections, such as at the root of the vane, there could be slight gaps between the two surfaces which would cause enormous difficulties, especially for grid generation packages.

At the heart of most CAD systems these days is a solids modelling package (e.g. ParaSolids, Pro Engineer [6, 13]), which defines each object as a composite built from simpler solids using rules of union, intersection and exclusion. For example, a turbine vane is a solid which is a union of three parts, the blade itself and the 'fir trees' which attach it to the hub and tip annuli. The blade part can be defined as an intersection between two solids, one a body of revolution corresponding to the volume between the hub and tip annuli, and the other a parametric cylinder part of whose cylindrical parametric surface corresponds to the vane surface. The hub fir tree part is an intersection between a hub solid of revolution, and a standard fir tree solid (possibly defined as a parametric cube) whose location depends on the minimum radius along the line of the hub/blade intersection. This may seem quite complex, but it is the task of modern CAD systems to make the definition and handling of such objects as simple as possible.

Once the designer has input the solid definition, the CAD system can determine its surfaces, lines of intersection (such as the vane/hub root) and points of intersection (such as where the trailing edge line intersects the hub and tip annuli). The CAD system can output these as cubic spline patches (for surfaces), or cubic splines (for lines) of the requisite accuracy for use by grid generators [6]). Alternatively, it is becoming more common to be able to access the EPD database directly through a programming interface, so that the grid generator can directly interrogate the EPD database to find, to a specified accuracy, the location of points on the surfaces and along surface lines.

If the designer, or design system, then changes one or more design parameters in the definition of the solids in the EPD database, the CAD system can compute the new surfaces and lines of intersection. The grid generator can then use the new information to construct an entirely new grid. Alternatively, the grid generator can determine from the CAD system the perturbations to the surfaces and lines of intersection, and use these to perturb the surface grid points of the original grid so that they lie on the surfaces and lines of intersection of the new geometry. Having done so, the grid generator can then perturb the interior grid points to produce a valid perturbed grid with the same topology as the original grid. For reasons to be discussed in the next section, this second approach is preferable in many cases even though it involves additional programming in the grid generation package.

3 Design approaches

3.1 Preliminaries

Before starting to discuss specific design methods, it is necessary to define the design task and some of the nomenclature which will be used.

The design parameters are labelled $\alpha_1, \alpha_2, \alpha_3, \ldots$ and are referred to collectively as $\boldsymbol{\alpha}$. $\boldsymbol{e}_k$ is defined to be a vector whose elements are zero except for its k^{th} element which has value unity. Therefore, $\Delta\boldsymbol{\alpha} = \epsilon_k \boldsymbol{e}_k$ represents a perturbation of magnitude ϵ_k to the k^{th} design parameter, keeping the others unchanged.

The dependent variables are labelled $\boldsymbol{U}$. In the original mathematical modelling, this would be a collection of continuous variables in various regions. For example, depending on the application, this may include flow variables, structural displacements and stresses, temperatures, etc. For simplicity, however, we will take $\boldsymbol{U}$ to be a set of discrete values of these quantities related to the computational grids being used for the engineering analyses. Accordingly, these dependent variables are related to the design parameters through a set of nonlinear discrete equations of the form

$$\boldsymbol{F}(\boldsymbol{U}, \boldsymbol{\alpha}) = 0.$$

A design usually has to satisfy a number of constraints. Some of these will be equality constraints (e.g. fixed thrust) and will be written collectively as

$$\boldsymbol{E}(\boldsymbol{U}, \boldsymbol{\alpha}) = 0.$$

Others will be in the form of inequalities (e.g. maximum fan diameter or wing span) and will be written as

$$\boldsymbol{C}(\boldsymbol{U}, \boldsymbol{\alpha}) \geq 0.$$

A constraint is said to be *violated* if it is not satisfied. A constraint is said to be *active* if it is either an equality constraint or an inequality constraint which is currently satisfied through strict equality. The number of active constraints should not exceed the total number of design parameters; otherwise the problem is over-constrained and will not have a solution in general.

3.2 Optimisation and the role of the designer

Before talking about specific numerical approaches to design, it is useful to first discuss the aims of design and the role of the designer within the design process.

At the outset, the designer must specify the design space, the parameters in the parametric design which are to be varied to improve the design. Since the computational cost of some of the numerical methods discussed later is proportional to the number of design parameters, it is very important that the designer uses his expert judgement to limit the number of design parameters to those which are most important.

The designer must also specify the design constraints. As mentioned above, some of these will be inequality constraints (minimum and maximum values for the design parameters, minimum blade thickness, etc.) while some will be equalities (specified lift for a wing, or pressure ratio for a compressor).

There are then three possible design scenarios. In the first, the designer is able to define a single scalar function $I(\boldsymbol{U}, \boldsymbol{\alpha})$ (known as the *objective* function) to be optimised subject to all of the constraints. In preliminary design this objective function may be overall fuel efficiency, or even the financial return on investment of the operating airline [11]. The computational design system will then attempt to find the optimal solution to the problem, subject to the constraints, using the most appropriate optimisation technique.

In component design, the definition of a suitable objective function can be trickier. Ideally, the designer might wish to minimise drag (for an aircraft) or loss (for a compressor). However, due to limitations in turbulence and transition modelling, the time-averaged treatment of unsteadiness such as vortex shedding and wake/rotor interaction, and numerical effects due to grids which do not yet fully resolve all features in three-dimensional flows, CFD methods are often not able to predict drag or loss with sufficient accuracy for the purposes of design optimisation. Therefore, it is much more common for the designer to choose an

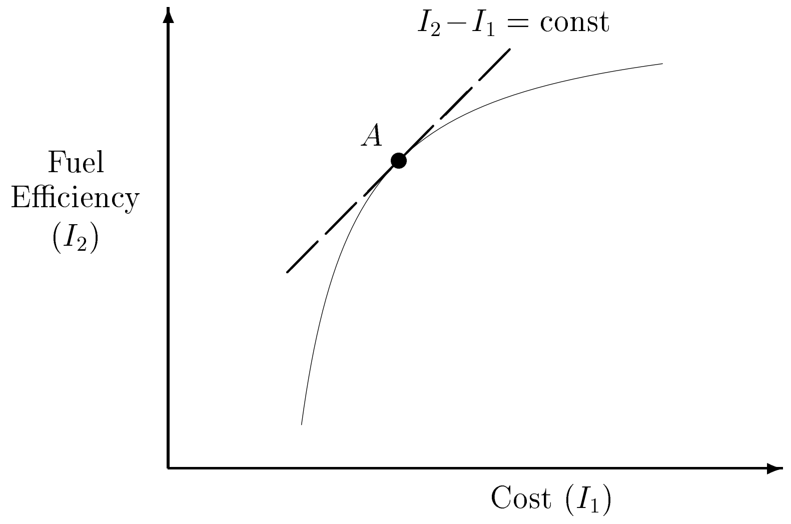

Figure 3: An example of a tradeoff between two different objective functions

alternative objective function, which if optimised will lead to an improved design with lower drag or loss. A common choice is the root-mean-square deviation from a target pressure distribution [10, 9, 14, 15, 16, 20]. CFD methods usually do well in predicting pressure distributions, and so should be able to optimise such an objective function quite reliably. A knowledgeable designer is able to specify a target pressure distribution which in the case of wing design will lead to low boundary layer growth, and in the case of turbomachinery blade design will also reduce the secondary flow and hence the losses. Thus, the designer plays a critical role in formulating a well-behaved objective function which can be reliably optimised by the available computational analysis tools.

During the optimisation process, the designer's task is to monitor the evolution of the design parameters, making sure that the design remains sensible. This may prove to be a much harder task than it appears. Aerospace design is very multidisciplinary and highly constrained. Initially, one might ignore a large number of inequality constraints, believing them to be unimportant because they will not be active in the final design, and wishing to minimise the computational cost of each step in the design process. One may even forget a large number of 'obvious' constraints (such as minimum blade thicknesses). The designer must therefore watch the evolution of the design to see if new constraints should be added [17]. In component design, it may also be necessary to examine the detailed results from the engineering analyses to ensure that the design does not produce flow fields or other features which violate basic modelling assumptions inherent in the analyses. For example, the use of potential flow modelling would no longer be appropriate if the design led to the presence of strong shocks.

In the second design scenario, it is appropriate to work with more than one objective function. For example, in preliminary aeroengine design one might choose to perform a design which optimises fuel efficiency at a fixed manufacturing cost. However, the airlines considering purchasing the engine will be considering their own profit optimisation, taking into account the running costs due to both fuel consumption and interest payments on the capital costs. Therefore, in preliminary design it may be more helpful to plot the trade-off between fuel efficiency and unit engine cost, as illustrated in Figure 3. Each point on this curve represents the optimum fuel efficiency for a fixed cost, and the optimum cost for a fixed fuel efficiency. The designer would have the responsibility of studying this tradeoff curve and deciding on the optimum tradeoff.

Similarly, at the component design level, the thickness of a compressor blade is a tradeoff between aerodynamic performance, which decreases with increasing thickness, and structural integrity which improves with increasing thickness. Rather than fixing one and optimising the other, the designer may prefer to study the tradeoff between the two before making a judgement about the best compromise. This need to assess multidisciplinary tradeoffs is emphasised in Reference [17] in the context of aircraft design.

If the relative importance of the two objective functions is known beforehand, then a single composite objective function of the form $I_2+\lambda I_1$ can be created. Referring to Figure 3, optimising I_2-I_1 corresponds to finding the point A on the curve which has the maximum value of I_2-I_1, and for which there is a tangent line of the form $I_2-I_1 = \text{const}$. The drawback of this approach is that the designer may not have a good idea of the appropriate value of λ, and optimising in this fashion would give no information about how the optimum would change if the value of λ were changed.

In the first two design scenarios, the engineering design system was responsible for some, or all, of the optimisation of the design, with the designer monitoring the design evolution in the first, and making some critical design decisions in the second. In the third design scenario, the designer performs the optimisation, with the design system supplying the designer with sensitivity information about the consequences of design changes. This assumes that there is an existing design and the objective is to improve upon it. The designer specifies the active design parameters and the constraint functions and objective functions he considers important. The design system returns the sensitivity of each of the functions to changes in each of the parameters, and invites the designer to decide upon suitable parameter changes. The design system may also aid the designer by ensuring that the changes are compatible with the constraints in the problem.

This scenario gives the designer the greatest flexibility, allowing the designer to take into account other factors and constraints which may be hard to specify in a computerised design system [17]. In particular, during the development of an integrated design system, when not all of the analysis modules have been developed or integrated into the system, this third approach may be the only feasible option.

Sensitivity analysis is a crucial component of the tightly coupled two-level

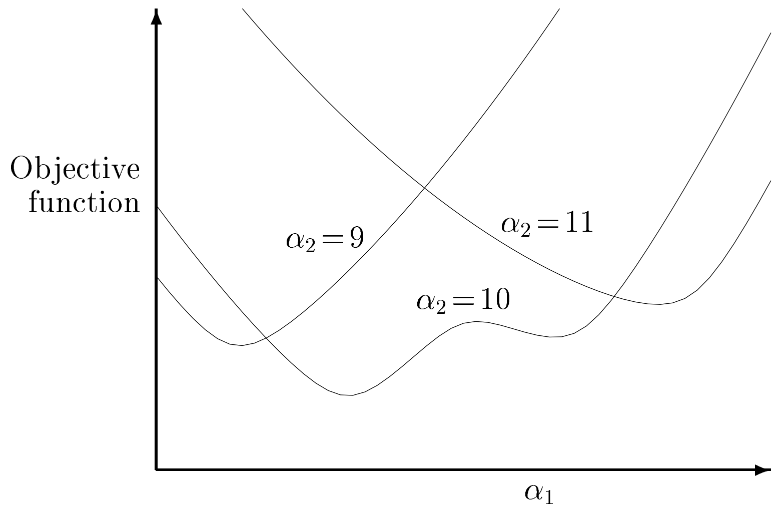

Figure 4: An example of an objective function depending on one continuous parameter, α_1, and one integer parameter, α_2.

design system described earlier. It is unlikely that at the component level one would simultaneously optimise the size and shape of different blade rows in an engine. However, sensitivity analysis at the component level could determine, for example, the change in the aerodynamic efficiency due to a change in chord length. With this information, the system level design could consider tradeoffs, increasing the chord of one blade row while simultaneously decreasing the chord length of another to retain a fixed overall engine size.

The whole approach of sensitivity analysis also has spinoff benefits in other areas, such as risk management and strategic research planning; this is discussed later.

3.3 Methods for global optimisation

Global optimisation is particularly appropriate to the preliminary design task of finding the optimum configuration of the overall system. The objective function to be optimised will usually be something relatively simple of clear engineering importance, such as aiming to maximise fuel efficiency. In addition to constraints such as fixed thrust, there may also be inequality constraints to ensure that the design remains within the design space for which the empiricism is thought to be valid.

The aim then is to find the optimal solution over the entire permissible design space. Figure 4 illustrates two of the important features of this problem. The first is that some of the design parameters may be integers. In the case of aeroengine design, the number of stages in the compressor, the number of blades in each blade row and the material type of different parts of the engine are all examples of integer design variables. In aircraft design, the number of engines, the number of high-lift flap and slat components and the material type (metal or composite) of various components are all integer parameters. The problem with integer parameters is that for each set of values for the integer parameters there will be an optimal solution for certain values of the real (i.e. continuous) parameters. How does the optimisation procedure select the correct global optimum? The second feature, illustrated by the curve for $\alpha_2 = 10$, is that even when there are only continuous parameters there is the possibility that the objective function may have multiple local optima. This again raises the problem of how to find the global optimum.

Because the optimisation procedure must to some extent search the entire design space, it is vital that the computational cost of determining $\boldsymbol{U}$, and hence $\boldsymbol{E}, \boldsymbol{C}$ and I, for a given set of design parameters $\boldsymbol{\alpha}$ is quite low. This is why global optimisation is best suited to preliminary design using inexpensive empirical modelling. Assuming this low computational cost, genetic algorithms and other stochastic evolutionary methods are a particularly appropriate choice to find the global optimum [7, 11]. With genetic algorithms, one starts with an initial family of solutions, some of which may not be feasible because they violate one or more of the constraints. The genetic algorithm then produces new families of solutions in a way that gives a preference towards solutions which are feasible and more optimal. The process can handle integer variables as well as continuous variables, and with a large initial family one gets effective cover of the entire design space making it likely that the process will converge to the global optimum rather than an incorrect local optimum [18]. Another good feature of genetic optimisation is that the computational cost increases rather slowly as the number of design parameters increases so there is little need for the designer to prune the design space.

3.4 Local Optimisation

In local optimisation, within either system or component design, one has an initial design which one wishes to improve upon through evolution, looking at the neighbourhood of the current design within the design space.

There are three broad categories of optimisation methods:

stochastic Like genetic algorithms and related evolutionary algorithms, these methods introduce a random element in the evolution of the design. They evaluate whether or not the new designs are better and feasible, allowing some infeasible and poorer solutions in the short term but aiming for feasibility and optimality in the long term.

gradient-based There are a huge number of methods designed to optimise a smooth function $I(\boldsymbol{\alpha})$ given the ability to evaluate both $I(\boldsymbol{\alpha})$ and its vector gradient $\nabla_\alpha I$ whose k^{th} element is $\frac{\partial I}{\partial \alpha_k}$.

In unconstrained optimisation, the simplest method is to choose $\nabla_\alpha I$ as a search direction and use a simple one-dimensional optimisation method to find a local optimum, and then repeat with the new gradient. More efficient methods approximate the Hessian matrix of second derivatives, $\frac{\partial^2 I}{\partial \alpha_j \partial \alpha_k}$, effectively constructing a local quadratic approximation to $I(\boldsymbol{\alpha})$.

There are modifications to the standard unconstrained optimisation methods to handle constraints, often through the use of Lagrange multipliers. However, when there are a large number of constraints, other approaches may be preferable. In particular, when the number of active constraints is equal to the number of active design parameters, the best approach may be to use a linear approximation to the objective function and constraints. The optimal solution to this linear programming problem can be found using the simplex method, and then the process is repeated using the new solution and gradient information.

other There are other optimisation methods which are deterministic, but do not utilise gradient information. One of the most popular is, confusingly, also called the simplex method (or sometimes 'the other simplex method' to distinguish it from the linear programming problem. In a N-dimensional design space this method uses a N-dimensional simplex having $N+1$ vertices. This is a triangle in 2D, a tetrahedron in 3D, and a generalisation of this in higher dimensions. At each step of the procedure, all of the vertices are held fixed apart from the one which is least optimal which is reflected in the opposing face. If the new vertex is better than the poorest of the rest the procedure is repeated; if not, it indicates the simplex is close to the optimum and so the size of the simplex is reduced.

In general, when dealing with a small number of design parameters the gradient-based methods are computationally more efficient than the other methods, but they can also be less robust, getting confused much more easily when the objective function is not very smooth, and getting stuck in local optima. When there is a larger number of design parameters, and a large number of constraints, the stochastic methods are more competitive. Their randomness gives them the ability to escape small local optima en route to a global optimum. Therefore, for local optimisation within system level design I would tend to prefer stochastic optimisation methods; even if the number of active design parameters is small the computational cost of the system-level analysis is so small that the computational expense of using the stochastic methods is perfectly acceptable.

For local optimisation within component design the choice is harder. If there are many active design variables the cost of almost any of the optimisation methods will be substantial. As explained later, the use of adjoint methods to compute the linear sensitivities can greatly decrease the cost, but only if there are very few active constraints. In practice, the designer should attempt to minimise the

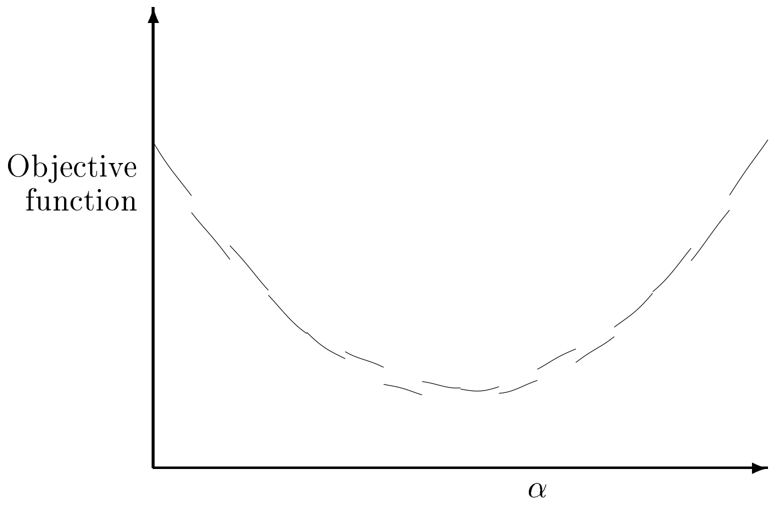

Figure 5: Discontinuities and ripples in an objective function due to discretisation effects

number of active design variables by careful selection of the most appropriate design parameters for a particular optimisation.

Once the number of design parameters is reduced, the gradient-based methods become the natural choice, but here one encounters another problem, the fact that the objective function is often not smooth. In fact, for many engineering analyses, it is not even continuous. The origin of the problem is not in the mathematical modelling but in the numerical discretisation of the partial differential equations in the fluid and structural modelling. There are in fact two problems. The first is that as the design changes the grids produced by the grid generator will usually not vary continuously. The number of grid points will often change, and in the case of unstructured grids the grid connectivity will also change. Both of these will produce small discontinuities in all objective and constraint functions. The second problem is discontinuities in the solutions (such as shocks) or very rapid variations which are inadequately resolved (such as wakes and free shear layers). The position of these features relative to grid points will vary as the design parameters changes, and this variation in position relative to the grid can produce a small 'ripple' in the objective and constraint functions. The magnitude of the ripple is small, but its 'wavelength' corresponds to the variation in the design parameter necessary for the feature to move one grid point. Thus the wavelength is small and so the variation in the gradient of all of the functions may be appreciable. Both of these features are illustrated in Figure 5 which shows a function of a single design variable.

In practice, the difficulties associated with these two problems may not become significant until one is very close to the optimum solution, in which case one might reasonably argue that it is not important since the solution one obtains may be sufficiently close to optimal. However, it may also be true that certain objective functions (such as drag and loss) are inherently more sensitive than others (such as lift, deviation from a target pressure distribution, etc.)

One solution to this problem of 'noise' superimposed upon the objective function is to construct a smooth function (sometimes called a 'response surface' or 'regression surface' [5, 12]) which is a good approximation to the noisy objective function. At its simplest, this involves a least-squares fitting of a quadratic function to the noisy data. The smooth approximating function can then be optimised reliably by standard gradient-based methods.

Another problem of local optimisation within component design is that iterative methods are often used to solve the finite difference equations that arise from the discretisation of the partial differential equations. The iterative process is almost always terminated before the solution has converged to the level of machine accuracy, and so there is a residual error corresponding to the fact that the nonlinear discrete equations are not satisfied exactly. This residual error will mean corresponding errors in all of the objective and constraint functions. When the design process has come close to the optimum, these residual errors may become significant and one may need to use a tighter convergence criterion at the expense of increased computational costs. In practice, however, I suspect that this is not an important problem.

3.5 Sensitivity analysis

3.5.1 Nonlinear analysis

In nonlinear sensitivity analysis, one obtains approximate linear sensitivities by simple finite differencing of the solutions from a number of nonlinear computations [22, 23, 21]. For each set of design parameters $\boldsymbol{\alpha}$, the discrete equations

$$\boldsymbol{F}(\boldsymbol{U}, \boldsymbol{\alpha}) = 0,$$

can be solved to implicitly obtain $\boldsymbol{U}$ as a function of $\boldsymbol{\alpha}$. Using simple one-sided differencing, we can define the approximate sensitivity of the flow solution to variations in the k^{th} design parameter as

$$\frac{d\boldsymbol{U}}{d\alpha_k} \approx \frac{\boldsymbol{U}(\boldsymbol{\alpha}+\epsilon_k \boldsymbol{e}_k) - \boldsymbol{U}(\boldsymbol{\alpha}))}{\epsilon_k}.$$

Similarly, the approximate gradient of an objective function $I(\boldsymbol{U}, \boldsymbol{\alpha})$ is given by

$$\frac{dI}{d\alpha_k} \approx \frac{I(\boldsymbol{U}(\boldsymbol{\alpha}+\epsilon_k \boldsymbol{e}_k), \boldsymbol{\alpha}+\epsilon_k \boldsymbol{e}_k) - I(\boldsymbol{U}(\boldsymbol{\alpha}), \boldsymbol{\alpha})}{\epsilon_k}.$$

A question that arises is what should be the magnitude of the perturbations ϵ_k? Since the finite difference corresponds to a secant approximation to the gradient, it is intuitively clear from Figure 5 that a large value of ϵ_k has the advantage of minimising the errors due to discontinuities and ripple in the function. However, a large value of ϵ_k would have a significant error due to the one-sided nature of the differencing and so it might be necessary to use central differencing,

$$\frac{dI}{d\alpha_k} \approx \frac{I(\boldsymbol{U}(\boldsymbol{\alpha}+\epsilon_k\boldsymbol{e}_k), \boldsymbol{\alpha}+\epsilon_k\boldsymbol{e}_k) - I(\boldsymbol{U}(\boldsymbol{\alpha}-\epsilon_k\boldsymbol{e}_k), \boldsymbol{\alpha}-\epsilon_k\boldsymbol{e}_k)}{2\epsilon_k}$$

doubling the computational cost.

Using an extremely small value of ϵ_k would theoretically lead to a finite difference value approaching the analytic derivative, but in practice there is the problem of rounding errors due to finite precision floating point arithmetic if the ratio

$$\frac{I(\boldsymbol{U}(\boldsymbol{\alpha}+\epsilon_k\boldsymbol{e}_k), \boldsymbol{\alpha}+\epsilon_k\boldsymbol{e}_k) - I(\boldsymbol{U}(\boldsymbol{\alpha}), \boldsymbol{\alpha})}{I(\boldsymbol{U}(\boldsymbol{\alpha}), \boldsymbol{\alpha})}$$

becomes too small. However, 64-bit arithmetic is being used increasingly, and for this the rounding errors would be acceptable for values of ϵ_k for which the above ratio is larger than 10^{-10}.

There is still the problem of the 'ripple'. If this is a major problem, the best solution may be to use an intermediate value for ϵ_k, one which is small but not very small. The danger in this is that differencing across a discontinuity in the objective function could produce a very large error. To avoid this, it would be necessary to use a perturbed grid of the same size and topology as the base grid. The construction of such a grid was discussed in Section 2.3.

The main advantage of the nonlinear sensitivity approach is its simplicity. There are no major new analysis codes to be written, just a small amount of programming to evaluate the objective and constraint functions. With the appropriate design software to manage the construction of the approximate sensitivities it is then possible to build the analysis codes into a design system very rapidly.

The direct sensitivity approach also has a big advantage when the objective function comes from a least-squares minimisation problem, in which case it has the form

$$I = \tfrac{1}{2} \sum_n \left(f_n(\boldsymbol{U}) - f_{n0}\right)^2,$$

where $f_n(\boldsymbol{U})$ is some nonlinear function of the flow variables, such as the pressure at a particular surface point, and f_{n0} is the target value for the design.

Linearising about a base solution $\boldsymbol{U}_0$ gives

$$I \approx \tfrac{1}{2} \sum_n \left(f_n(\boldsymbol{U}_0) - f_{n0}\right)^2 + \sum_n \left(f_n(\boldsymbol{U}_0) - f_{n0}\right) \frac{\partial f_n}{\partial \boldsymbol{U}} \widetilde{\boldsymbol{U}} + \tfrac{1}{2} \sum_n \left(\frac{\partial f_n}{\partial \boldsymbol{U}} \widetilde{\boldsymbol{U}}\right)^2,$$

where $\widetilde{\boldsymbol{U}}$ is the flow perturbation. Putting

$$\widetilde{\boldsymbol{U}} = \sum_i \frac{d\boldsymbol{U}}{d\alpha_i} \widetilde{\alpha}_i,$$

then gives

$$I \approx I_0 + \sum_i \frac{\partial I}{\partial \alpha_i}\,\widetilde{\alpha}_i + \tfrac{1}{2}\sum_{i,j} \frac{\partial^2 I}{\partial \alpha_i \partial \alpha_j}\,\widetilde{\alpha}_i \widetilde{\alpha}_j,$$

where

$$\frac{\partial I}{\partial \alpha_i} = \sum_n \left(f_n(\boldsymbol{U}_0) - f_{n0}\right) \frac{\partial f_n}{\partial \boldsymbol{U}} \frac{d\boldsymbol{U}}{d\alpha_i},$$

and

$$\frac{\partial^2 I}{\partial \alpha_i \partial \alpha_j} = \sum_n \left(\frac{\partial f_n}{\partial \boldsymbol{U}} \frac{d\boldsymbol{U}}{d\alpha_i}\right) \left(\frac{\partial f_n}{\partial \boldsymbol{U}} \frac{d\boldsymbol{U}}{d\alpha_j}\right).$$

Thus, the direct sensitivity approach gives both the gradient and the approximate Hessian of the least-squares objective function [8, 21, 22, 23], enabling the approximate quadratic function to be minimised by solving the linear equations

$$\sum_j \frac{\partial^2 I}{\partial \alpha_i \partial \alpha_j}\,\widetilde{\alpha}_j \quad + \quad \frac{\partial I}{\partial \alpha_i} = 0.$$

The main disadvantage of the nonlinear approach is its cost when the number of design parameters is large. This is why it is important that the designer exercises good judgement in limiting the number of active design parameters.

Another way of reducing the computational cost is to use coarse computational grids for the purpose of evaluating approximate sensitivities. Starting with the fine grid solution $\boldsymbol{U}$, a coarse grid solution $\boldsymbol{U}_{c0}$ could be defined by interpolation onto the coarse grid. Defining the coarse grid residual vector as

$$\boldsymbol{R}_c = \boldsymbol{F}_c(\boldsymbol{U}_{c0}, \boldsymbol{\alpha}),$$

the coarse grid equations defining perturbed solutions would be

$$\boldsymbol{F}_c(\boldsymbol{U}_c(\boldsymbol{\alpha}+\Delta\boldsymbol{\alpha}), \boldsymbol{\alpha}+\Delta\boldsymbol{\alpha}) = \boldsymbol{R}_c.$$

The purpose of the residual vector on the r.h.s. of this equation is to ensure that $\boldsymbol{U}_c = \boldsymbol{U}_{c0}$, when $\Delta\boldsymbol{\alpha} = 0$.

The approximate sensitivities can then be obtained with the same one-sided approximation as before,

$$\frac{dI}{d\alpha_k} \approx \frac{I\left(\boldsymbol{U}_c(\boldsymbol{\alpha}+\epsilon_k \boldsymbol{e}_k), \boldsymbol{\alpha}+\epsilon_k \boldsymbol{e}_k\right) - I\left(\boldsymbol{U}_c(\boldsymbol{\alpha}), \boldsymbol{\alpha}\right)}{\epsilon_k}.$$

3.5.2 Linear analysis

Mathematically, the simplest form of linear analysis is equivalent to the nonlinear analysis in the limit as $\epsilon_k \to 0$. If we define $\widetilde{\boldsymbol{U}}_k$ to be the sensitivity of $\boldsymbol{U}$ to changes in the k^{th} design parameter, then linearising the nonlinear discrete equations yields

$$\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}\,\widetilde{\boldsymbol{U}}_k + \frac{\partial \boldsymbol{F}}{\partial \alpha_k} = 0.$$

This can be solved, directly or iteratively, to obtain $\widetilde{\boldsymbol{U}}_k$ for each design parameter. The total derivative of an objective function with respect to the k^{th} design parameter is then given by

$$\frac{dI}{d\alpha_k} = \frac{\partial I}{\partial \boldsymbol{U}} \widetilde{\boldsymbol{U}}_k + \frac{\partial I}{\partial \alpha_k}.$$

There are also linear methods in which the original nonlinear partial differential equation is linearised first, and then discretised. These do not have any particular advantages over the approach above, starting with the nonlinear discretisation and then linearising. Indeed, there are some disadvantages in that it can be harder to understand how best to treat the perturbations to the boundary conditions.

The big disadvantage of the linear methods compared to the nonlinear methods is the need to develop an entirely new analysis code to solve the linear perturbation problem. In CFD applications, the task of linearising the discrete Navier-Stokes equations together with the turbulence modelling is at best tedious and error-prone, although automatic differentiation software may be very helpful.

In CFD applications, the cost of solving the linear system of equations is comparable to the cost of solving the nonlinear system, so there are no computational savings from using the linear approach. The one exception to this is an unusual turbomachinery application in which one designs a blade row with a sinusoidal circumferential variation in camber with the aim of producing a corresponding pressure variation cancelling that produced by a single large pylon. In this case the nonlinear analysis must be performed for the full annulus whereas the linear analysis can be performed using complex variables on a single blade passage with a complex phase shift between its two periodic boundaries [21, 22, 23].

In structural applications the linear approach may be much cheaper, particularly if the nonlinear equations are solved by a Newton-Raphson method which involves the computation of the LU decomposition of the matrix $\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}$.

As with the nonlinear sensitivity approach, it is possible to reduce the computational cost of the linear analysis by using coarser grids.

3.5.3 Adjoint methods

The simplest form of the adjoint approach starts from the linear equations above, and then eliminates $\widetilde{\boldsymbol{U}}$ to obtain [10, 9]

$$\frac{dI}{d\alpha_k} = -\frac{\partial I}{\partial \boldsymbol{U}} \left(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}} \right)^{-1} \frac{\partial \boldsymbol{F}}{\partial \alpha_k} + \frac{\partial I}{\partial \alpha_k}.$$

This can then be written as

$$\frac{dI}{d\alpha_k} = \boldsymbol{V}^T \frac{\partial \boldsymbol{F}}{\partial \alpha_k} + \frac{\partial I}{\partial \alpha_k},$$

where the vector $\boldsymbol{V}$ satisfies the equation

$$\left(\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}\right)^T \boldsymbol{V} + \left(\frac{\partial I}{\partial \boldsymbol{U}}\right)^T = 0.$$

The great advantage of this approach is that one only needs to solve a single finite difference equation to get the sensitivities of I with respect to all of the design parameters. This is because the same solution $\boldsymbol{V}$ is used for each value of k. The only additional cost for each design parameter is the computation of $\frac{\partial \boldsymbol{F}}{\partial \alpha_k}$ and $\frac{\partial I}{\partial \alpha_k}$, which is inexpensive, and the dot product $\boldsymbol{V}^T \frac{\partial \boldsymbol{F}}{\partial \alpha_k}$ which is even cheaper.

The main drawback of the adjoint approach is that a separate adjoint equation must be solved for each objective function or constraint function. Hence, in a highly constrained design in which the number of active constraints is comparable with the number of active design parameters, there is little to be gained from the adjoint approach.

A second weakness of the adjoint approach is that there is no simple way in which to compute the Hessian matrix $\partial^2 I/\partial\alpha_i\partial\alpha_j$ even when the objective function comes from a least-squares minimisation problem. Instead, the gradient-based optimisation methods must construct an approximation to the Hessian matrix using information about the variation in the gradient at different points in the design space. In addition, such methods usually determine a search direction and then find the optimum along this direction using a line search algorithm. Both of these aspects result in more steps in the optimisation procedure than are required when for the direct sensitivity approach using its approximate Hessian.

The label 'adjoint' comes from the alternative treatment in which one starts with the linearised partial differential equation and converts the linear sensitivity of the objective function into an equivalent form involving the solution of the adjoint partial differential equation with appropriate boundary conditions [19]. This can then be discretised and solved numerically [2, 3, 4, 14, 15, 16, 20, 24].

4 Other related topics

4.1 Manufacturing considerations

Reducing the cost of manufacturing is a major concern in the aerospace industry today. Many years ago it might have been the case that a design team would aim to design the best possible product, and then a manufacturing team would face the task of trying to fabricate the product. Nowadays, the design team is much more aware of the manufacturing implications of their design decisions.

In preliminary design, the cost/weight/strength tradeoffs of using different materials may be considered through integer optimisation by assigning each material a unique integer value.

In detailed component design, sensitivity analysis could reveal the probable performance deterioration due to manufacturing tolerances. This information could then be used in considering the cost/benefit tradeoff in using more expensive manufacturing techniques to reduce the manufacturing tolerances.

4.2 Risk management

Aerospace companies invest huge amounts of money in developing new aircraft and new aeroengines. They are also competing fiercely to reduce the product development cycle and deliver a new product to customers as quickly as possible. In this climate, project managers are very concerned with risk management, trying to minimise the risks in developing a new product, avoiding problems which might seriously delay the project. However, at the same time there is still the goal of developing improved products and so some level of risk is inevitable.

In deciding where to focus development effort, sensitivity analysis can reveal the improvement in overall system performance due to a possible improvement in a particular component. Engineering judgement gives the likely development cost to achieve this improvement in the component, and the probability that, despite everyone's best efforts, the improvement cannot be achieved. With this information, the project management can focus development effort on those areas which offer the greatest potential benefit for the lowest development cost and with the lowest level of risk.

4.3 Strategic research planning

Large aerospace companies have large research budgets, and an important problem is deciding on the relative importance of different research areas. One consideration is the relative costs and benefits of technological advances in different areas. For example, in military aircraft one might consider the merits of reducing the weight of the avionics by 50% compared to the introduction of a new composite material weighing 10% less than the current one. In the turbomachinery context, one might consider the benefits of improved CFD modelling leading to improved turbine film cooling, compared to the development of a new alloy for the turbine blade which would also allow an increase in the flow temperature. In each case, sensitivity analysis can give the overall system benefits from the possible technological advance. Engineering judgement is needed to assess the likelihood of the technological advance, and its cost. From this, rational decisions can be made about how best to spend one's research budget.

4.4 Distributed and parallel computing

Distributed computing on multiple workstations, and parallel computing on shared-memory and distributed-memory multiprocessors, is the underlying technology which makes an integrated design system possible by providing the computational resources necessary to achieve acceptable execution times. Determination of approximate linear sensitivities through the computation of multiple perturbed nonlinear solutions is an ideal task-parallel application, in that each calculation is independent of the others and so they can all be executed in parallel with one on each workstation.

If some calculations using very fine computational grids will take too long if executed on a workstation, or require too much memory, then they are good candidates for execution on a multiprocessor system using a data-parallel approach in which the computational domain is divided into a number of pieces, each running on a different processor with extensive communication between all of the processors.

Companies with very large distributed-memory parallel computers (such as a 128-processor CRAY-T3D) might well choose to pursue both approaches at the same time, executing 16 sensitivity calculations simultaneously with each one using 8 processors. This would probably be much more efficient than carrying out the 16 sensitivity calculations one after another, with each one using all 128 processors.

References

[1] N. Alexandrov. Multilevel and multiobjective optimization in multidisciplinary design. AIAA Paper 96-4170-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[2] W.K. Anderson and V. Venkatakrishnan. Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation. AIAA Paper 97-0643, 1997.

[3] O. Baysal and M. Eleshaky. Aerodynamic design optimization using sensitivity analysis and computational fluid dynamics. *Journal of the American Institute on Aeronautics and Astronautics*, 30(3):718–725, 1992.

[4] O. Baysal and M.E. Eleshaky. Aerodynamic sensitivity analysis methods for the compressible Euler equations. *Journal of Fluids Engineering*, 113:681–688, 1991.

[5] D. Bos. Multidisciplinary design optimization of a supersonic transport aircraft using a hybrid genetic/gradient-based algorithm. AIAA Paper 96-4055-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[6] S. Chen and D. Tortorelli. Three-dimensional shape optimization with variational geometry. AIAA Paper 96-3992-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[7] D. Doorly, J. Peiro, and J-P. Oesterle. Optimization of aerodynamic-structural design using parallel genetic algorithms. AIAA Paper 96-4027-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[8] M. Drela. Design and optimization method for multi-element airfoils. AIAA Paper 93-0969, 1993.

[9] J. Elliot and J. Peraire. Practical 3D aerodynamic design and optimization using unstructured grids. AIAA Paper 96-4122-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[10] J. Elliott and J. Peraire. Aerodynamic design using unstructured meshes. AIAA Paper 96-1941, 1996.

[11] M. Ewing and K. Downs. Conceptual aircraft design with genetic search based on financial return on investment. AIAA Paper 96-4106-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[12] A. Giunta, V. Balabanov, D. Haim, B. Grossman, W. Mason, L. Watson, and R. Haftka. Wing design for a high-speed civil transport using a design of experiments methodology. AIAA Paper 96-4001-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[13] E. Hardee, K-H. Chang, K. Choi, X. Yu, and I. Grindeanu. A CAD-based design sensitivity analysis and optimization for structural shape design applications. AIAA Paper 96-3990-CP, 1996. Proceedings of 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.

[14] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3:233–260, 1988.

[15] A. Jameson. Optimum aerodynamic design using the control theory. *Computational Fluid Dynamics Review*, pages 495–528, 1995.

[16] A. Jameson, N.A. Pierce, and L. Martinelli. Optimum aerodynamic design using the Navier–Stokes equations. AIAA Paper 97-0101, 1997.

[17] W. Jou, W. Huffmann, D. Young, and R. Melvin. Practical considerations in aerodynamic design optimization. AIAA Paper 95-1730, 1995.

[18] A.J. Keane. Genetic algorithm optimization of multi-peak problems: studies in convergence and robustness. *Artificial Intelligence in Engineering*, 9(2):75–83, 1995.

[19] J.L. Lions. *Optimal Control of Systems Governed by Partial Differential Equations*. Springer-Verlag, 1971. Translated by S.K Mitter.

[20] J. Reuther and A. Jameson. Control based airfoil design using the Euler equations. AIAA Paper 94-4272-CP, 1994.

[21] G.N. Shrinivas. *Three-dimensional design methods for turbomachinery applications*. PhD thesis, Oxford University Computing Laboratory, 1996.

[22] G.N. Shrinivas and M.B. Giles. Application of sensitivity analysis to the redesign of OGV's. In *Proceedings of the IMECE Conference*, 1995.

[23] G.N. Shrinivas and M.B. Giles. OGV tailoring to alleviate pylon-OGV-fan interaction. In *Proceedings of the IGTI Turbo Expo*, 1995. ASME paper 95-GT-198.

[24] S. Ta'asan, G. Kuruvila, and M.D. Salas. Aerodynamic design and optimization in one shot. AIAA Paper 92-0025, 1992.