Chapter 15
Multidisciplinary Design Optimization: Designed by Computer

Cees Bil

Abstract Multidisciplinary design optimization (MDO) has been a field of research for 25 years. It refers to the formulation of the design problem in mathematical models and applying optimization techniques to find the minimum or maximum of a predefined objective function, possibly subject to a set of constraints. MDO has become an important tool in concurrent engineering (CE), with the ability to handle many design variables (DV) across various disciplines. Advances in computer technologies and software engineering have facilitated the practical application of MDO in industry, including aerospace, automotive, shipbuilding, etc. However, active research and development in MDO continues. The creative input of the human designer to the design process is critical and must be integrated in the MDO process. For MDO to be effective in the design of modern complex systems it must also incorporate non-technical disciplines, such as finance, environment, operational support, etc. It remains a challenge to do model them with adequate fidelity. Simulations and analytical models have imbedded assumptions, inaccuracies and approximations. How do we deal with these in an MDO environment? This chapter gives an introduction to MDO with an historical review, a discussion on available numerical optimization methods each with their specific features, the various MDO architectures and decompositions and two case studies where MDO has been applied successfully.

Keywords Multidiscplinary design optimization · Complex systems · Concurrent engineering · MDO architectures · Optimization methods

C. Bil (✉)
RMIT University, GPO Box 2476, Melbourne, VIC 3001, Australia
e-mail: cees.bil@rmit.edu.au

J. Stjepandić et al. (eds.), *Concurrent Engineering in the 21st Century*,
DOI 10.1007/978-3-319-13776-6_15

15.1 Introduction

The whole is more than the sum of its parts—Aristotle

When the Wright brothers made their historical flight in 1903, their objective was to achieve powered and controlled flight. In the twenty first century, achieving powered and controlled flight is hardly the challenge anymore, the question is how well will it fly and will it meet the user's needs. The user's needs are not necessarily focused on hardware, but on a total business solution, including maintenance, support, upgrades, etc., that achieve a certain objective over the life-cycle of the system. Since the industrial revolution, engineers have invested their ingenuity in developing increasingly complex machines, but perhaps the most striking development in terms of rapid technical progression and complexity is the aerospace domain (Fig. 15.1).

The current design environment of complex systems is defined by a rapid turnaround of cost effective solutions, involving all operational and business aspects. The concurrent engineering (CE) approach considers all technical and business aspects simultaneously, rather than sequentially as in the traditional design approach. A sequential design approach does not guarantee that an overall optimum design is found. Figure 15.2 shows a typical aircraft design problem. In the sequential design process, the aerodynamics group determines the best aspect ratio (AR), for example, for maximum range P, subject to performance requirements (design 01). Unfortunately, the structures group cannot comply with the flutter requirement and needs to increase the wing weight $W_{\min}$ (design 02). For design 02 all requirements are met, but it is not the optimum design (design 03). Considering

Fig. 15.1 Evolution of engineering complexity in the past century

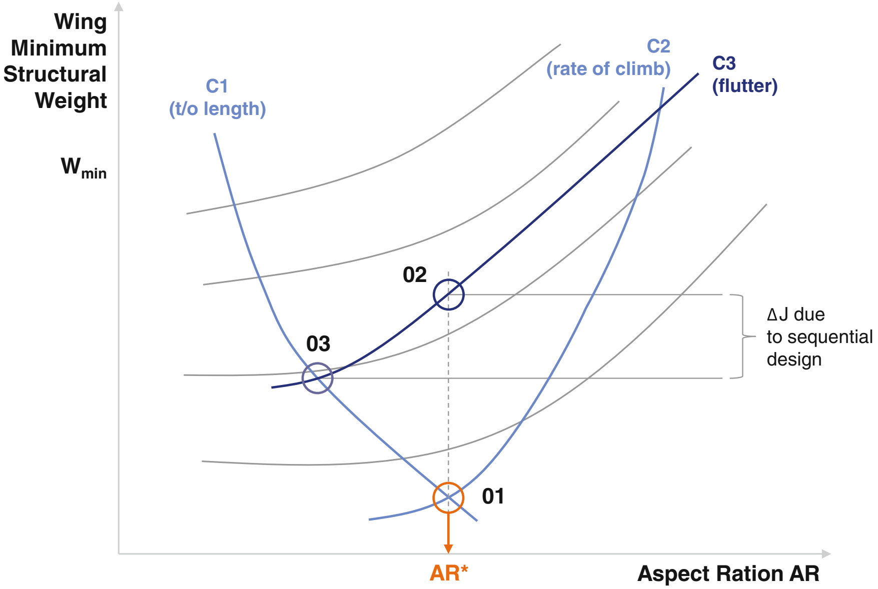


Fig. 15.2 Sequential versus concurrent design process [1]

aerodynamics, structures and performance at the same time, i.e. concurrently, would have resulted in an improved design.

This chapter introduces the multidisciplinary design optimization (MDO) approach that represents a modelling and simulation environment where numerical optimization techniques are applied to drive the optimization process. This chapter gives an overview of MDO applications to complex systems design. Section 15.2 provides the motivation for using MDO and its potential benefits in reduced lead times and improved design quality. Section 15.3 gives an historical background to MDO development. Section 15.4 discusses a range of numerical optimization methodologies, their classification and specific features. Section 15.5 cover more specifically nonlinear optimization methodologies, including the gradient-based methods such as SQL and GRM, and the genetic algorithms (GA) which have gained recent popularity as they do not rely on gradient information and are able to find a global optimum. Section 15.6 discusses MDO techniques for cases which are multi-modal or have multiple objectives. Section 15.7 gives an overview of various MDO architectures and the opportunity to decompose the optimization problem into different levels and coupling of variables, which avoids redundant computations and can speed up the process considerably. Section 15.8 presents two case studies where MDO was applied successfully in the structural design of a car body and of an aircraft wing. Section 15.9 concludes with a discussion some of the impediments in MDO application and focus areas for future research and development.

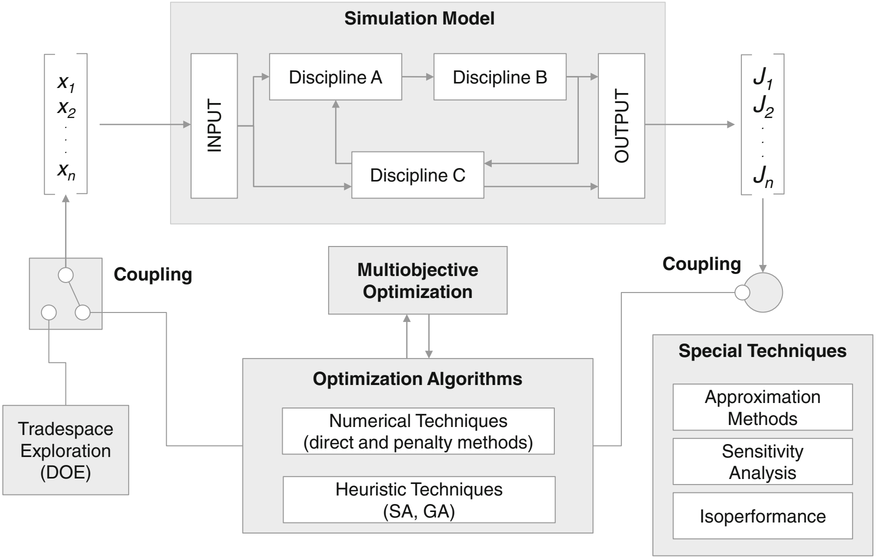


Fig. 15.3 Generic MDO framework [1]

15.2 Multidisciplinary Design Optimization (MDO)

A new approach that has gained much interest in the past two decades in assisting design teams is MDO [2, 3]. MDO is a sub-field of computational engineering and proposes an environment where all the relevant analysis tools, or simulation models, are coupled and a numerical optimization algorithm is applied to search for the optimum design as defined by a given objective function and subject to design constraints (Fig. 15.3).

There are a number of advantages to the MDO approach, such as:

- Reduction in design time
- Systematic, logical design procedure
- Handles wide variety of design variables (DV) and constraints concurrently
- Not biased by prejudice

These potential benefits have motivated many researchers, scientists and engineers to develop MDO frameworks for a range of different application [4–10].

15.3 Historical Background

The existence of optimization methods is as old as calculus and can be traced to the days of Newton, Lagrange and Cauchy [11]. The development of differential calculus methods of optimization was possible because of the contributions of

Newton and Leibnitz. The foundations of calculus of variations were laid by Bernoulli, Euler, Lagrange and Weierstrass. The optimization of constrained problems, which involves the addition of unknown multipliers, became known by the name of its inventor Lagrange. Cauchy made the first application of the steepest descent method to solve unconstrained minimization problems. In spite of these early contributions, very little progress was made until the middle of the 20th century, when high-speed digital computers made the implementation of the optimization procedures possible and stimulated further research in new methods.

The first step in the application of optimization was in structural design in the 1960s when Schmit [12] proposed a rather general new approach, which served as the conceptual foundation for the development of many modern structural optimization methods. It introduced the idea and indicated the feasibility of coupling finite element structural analysis and non-linear mathematical programming to create automated optimum design capabilities for a rather broad class of structural design problems.

An alternative, analytical form of structural optimization was offered by Prager and in numerical form by Venkayya in 1968 [13]. This concept became known as the optimality criteria. In the design of statically determinate structures, each member is fully stressed under at least one loading condition. The strength of the two methods suggested a natural separation of the design problem, where optimality criteria would deal with a large number of DV and mathematical programming would solve the component-design problem. This approach was pursued by Sobieski et al. in 1972 in the design of fuselage structures.

For practical MDO applications there are two important issues. The first is the selection of the models and analysis methods. As mathematical optimization relies only on the analysis methods provided; these methods must not only be accurate, but also correctly reflect the sensitivity to variations in the selected DV. The choice of analysis methods will depend on the design phase. It is usually not appropriate to use a Navier-Stokes CFD code in conceptual design as design is still very flexible and not accurately defined yet. Instead statistical/empirical methods as found in are more appropriate in the early design stages. A number of computer-based design synthesis systems have been developed for aircraft configuration design, such as ACSYNT, ADAS, RDS, SOCCER and AAA. Note that statistical/empirical methods are not based on engineering science and are therefore only applicable in a narrow range of applications and are not necessarily correctly sensitive to the selected DV.

The second issue is an acceptable computing time required to determine the optimum solution. This depends on the available computing power, sophistication of the analysis methods and the efficiency of the optimization method its and implementation. Investigations into using approximation methods as a mechanism to improve the efficiency of mathematical programming techniques started in the 1970s. This hybrid method uses approximations to find the optimum solution and then applies a more sophisticated analysis method to the approximate optimum design. The final optimum design is obtained iteratively. A form of this approach is known as surrogate models or response function techniques [14, 15].

15.4 Numerical Optimization Methods

Optimization is an important tool in decision science and in the analysis of physical systems. To use this methodology, we must first identify an objective, a quantitative measure of the quality of the system, for example profit, time, potential energy, or any quantity or combination of quantities that can be represented by a single numeric. The objective depends on certain characteristics of the system, called DV. The aim is to find the values for the DV that maximizes and minimizes the objective function. Often the range of values for the variables is constrained. The process of defining the relationship between the objective function, DV, and constraints for a given problem is known as modeling. Construction of an appropriate model is the first step—sometimes the most important step—in an optimization process. If the model is too simple, it will not give useful insights into the practical problem. If it is too complex, it may be too difficult to solve.

Once the model has been formulated, an optimization algorithm can be used to find its numerical solution. A variety of optimization algorithms exists, each tailored to a particular type of optimization problem. The responsibility of choosing the algorithm that is appropriate for a specific application often falls on the user. This choice is an important one, as it may determine whether the problem is solved rapidly or slowly and, indeed, whether the solution is found at all. After the optimization process has been completed, we must be able to recognize whether it has succeeded in its task of finding an optimum solution. In many cases, there are elegant mathematical expressions known as optimality conditions for checking that the current set of values for the DV is indeed the optimum solution of the problem. If the optimality conditions are not satisfied, they may still give useful information on how the current estimate of the solution can be improved. The model may be improved by applying techniques such as sensitivity analysis, which reveals the sensitivity of the solution to changes in the model and data. Interpretation of the solution may also suggest ways in which the model can be refined or improved (or corrected). If any changes are made to the model, the optimization problem is solved anew, and the process repeats.

15.4.1 Mathematical Formulation

In a mathematical context, optimization is the minimization or maximization of a function subject to constraints on its variables [16, 17]. We use the following notation:

- x is the vector of variables, also called unknowns or parameters;
- f is the objective function, a (scalar) function of x to be maximized or minimized;
- c_i are constraint functions, which are scalar functions of x that define certain equalities and inequalities that the unknown vector x must satisfy.

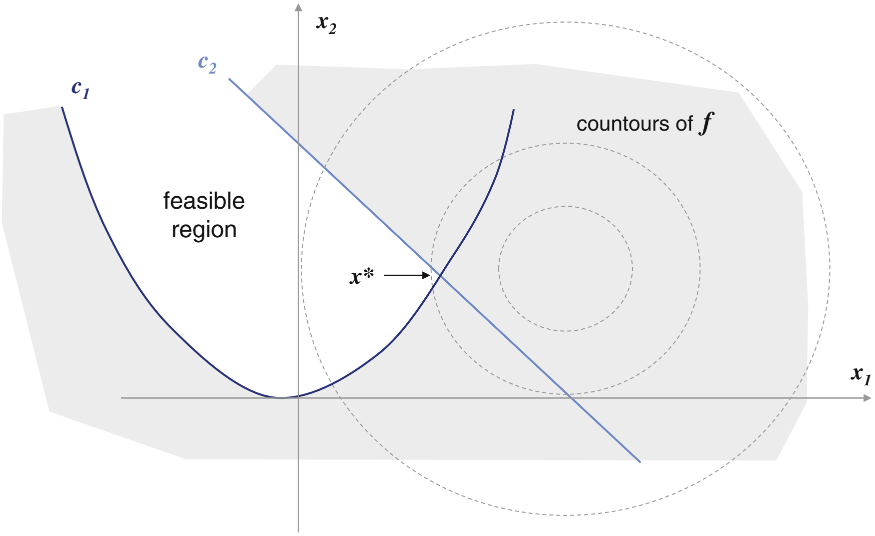


Fig. 15.4 Optimization problem with two design variables [18]

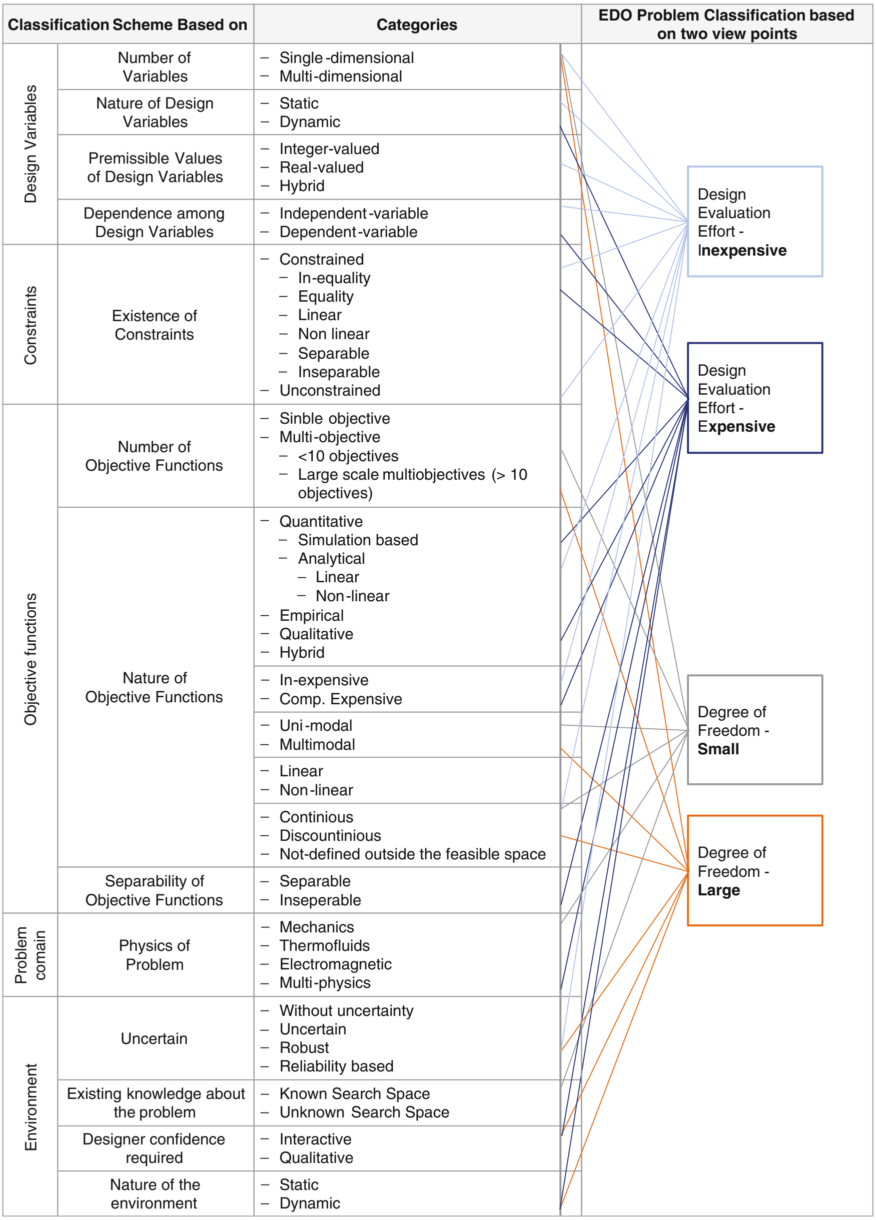
Using this notation, the optimization problem can be written as follows:

$$\min_{x \in R^n} f(x), \quad \text{subject to} \begin{array}{l} c_i(x) = 0, i \in n_e \\ c_i(x) \geq 0, i \in n_i \end{array} \tag{15.1}$$

Figure 15.4 shows the contours of the objective function, that is, the set of points for which $f(x)$ has a constant value [18]. It also illustrates the feasible region, which is the set of points satisfying all the constraints (the area between the two constraint boundaries), and the point x^*, which is the solution of the problem. The "infeasible side" of the inequality constraints is shaded. Classification of the engineering design optimization problem is necessary to select the right approach for a given problem [18, 19]. A classification is presented in Fig. 15.5. In the next sections different categories of optimization methods are discussed with their specific features and capabilities.

15.4.2 Constrained and Unconstrained Optimization

Problems with the general form of Eq. (15.1) can be classified according to the nature of the objective function and constraints (linear, nonlinear, convex), the number of variables, large or small, the smoothness of the functions, differentiable or non-differentiable, and so on. An important distinction is between problems that have constraints on the variables and those that do not. Unconstrained optimization problems, for which we have $n_e = n_i = 0$ in Eq. (15.1), arise in many practical



Classification Scheme Based on		Categories	EDO Problem Classification based on two view points
Design Variables	Number of Variables	– Single-dimensional – Multi-dimensional	
	Nature of Design Variables	– Static – Dynamic	
	Premissible Values of Design Variables	– Integer-valued – Real-valued – Hybrid	
	Dependence among Design Variables	– Independent-variable – Dependent-variable	
Constraints	Existence of Constraints	– Constrained – In-equality – Equality – Linear – Non linear – Separable – Inseparable – Unconstrained	
Objective functions	Number of Objective Functions	– Sinble objective – Multi-objective – <10 objectives – Large scale multiobjectives (> 10 objectives)	
	Nature of Objective Functions	– Quantitative – Simulation based – Analytical – Linear – Non-linear – Empirical – Qualitative – Hybrid	
		– In-expensive – Comp. Expensive	
		– Uni-modal – Multimodal	
		– Linear – Non-linear	
		– Continious – Discountinious – Not-defined outside the feasible space	
	Separability of Objective Functions	– Separable – Inseperable	
Problem comain	Physics of Problem	– Mechanics – Thermofluids – Electromagnetic – Multi-physics	
Environment	Uncertain	– Without uncertainty – Uncertain – Robust – Reliability based	
	Existing knowledge about the problem	– Known Search Space – Unknown Search Space	
	Designer confidence required	– Interactive – Qualitative	
	Nature of the environment	– Static – Dynamic	

Design Evaluation Effort - **Inexpensive**
Design Evaluation Effort - **Expensive**
Degree of Freedom - **Small**
Degree of Freedom - **Large**

Fig. 15.5 Classification of optimization methods [18]

applications. Even for some problems with natural constraints on the variables, it may be appropriate to disregard them if they do not affect the solution and do not interfere with the algorithms. Unconstrained problems arise also as reformulations of constrained optimization problems, in which the constraints are replaced by penalization terms added to objective function that have the effect of discouraging constraint violations. Constrained optimization problems arise from models in which constraints play an essential role, for example in imposing budgetary constraints in an economic problem or shape constraints in a design problem. These constraints may be simple bounds, more general linear constraints, or nonlinear inequalities that represent complex relationships among the variables.

When the objective function and all the constraints are linear functions of x, the problem is a linear programming problem. Problems of this type are probably the most widely formulated and solved of all optimization problems, particularly in management, financial, and economic applications. Nonlinear programming problems, in which at least some of the constraints or the objective function is nonlinear, tend to arise naturally in the physical sciences and engineering, and are becoming more widely used in management and economic sciences as well [20, 21].

15.4.3 Continuous Versus Discrete Optimization

In some optimization problems the variables make sense only if they take on integer values. For example, a variable x could represent the number of power plants that should be constructed by an electricity provider during the next 5 years, or it could indicate whether or not a particular factory should be located in a particular city. The mathematical formulation of such problems includes integrality constraints or binary constraints, in addition to algebraic constraints like those appearing in Eq. (15.1). Problems of this type are called integer programming problems. If some of the variables in the problem are not restricted to be integer or binary variables, they are called mixed integer programming (MIP) problems. Integer programming problems are a type of a discrete optimization problem. Generally, discrete optimization problems may contain not only integers and binary variables, but also more abstract variable objects such as permutations of an ordered set. The defining feature of a discrete optimization problem is that the unknown x is drawn from a finite, but often very large, set. By contrast, the feasible set for continuous optimization problems is usually infinite, as when the components of x are allowed to be real numbers.

Continuous optimization problems are usually easier to solve because the smoothness of the functions makes it possible to use objective and constraint information at a particular point x to deduce information about the function's behavior at all points close to x. In discrete problems, by contrast, the behavior of the objective and constraints may change significantly as we move from one feasible point to another, even if the two points are "close" by some measure. The feasible sets for discrete optimization problems can be thought of as exhibiting an extreme form of non-convexity, as a convex combination of two feasible points is in general not feasible. Continuous optimization techniques often play an important role in solving discrete optimization problems. For instance, the branch-and-bound method for integer linear programming problems requires the repeated solution of linear programming "relaxations," in which some of the integer variables are fixed at integer values, while for other integer variables the integrality constraints are temporarily ignored. These sub-problems are usually solved by the simplex method.

15.4.4 Global and Local Optimization

Many algorithms for nonlinear optimization problems seek only a local solution, a point at which the objective function is smaller than at all other feasible nearby points. They do not always find the global solution, which is the point with lowest function value among all feasible points. Global solutions are needed in some applications, but for many problems they are difficult to recognize and even more difficult to locate. For convex programming problems, and more particularly for linear programs, local solutions are also global solutions. General nonlinear problems, both constrained and unconstrained, may possess local solutions that are not global solutions.

15.4.5 Stochastic and Deterministic Optimization

In some optimization problems, the model cannot be fully specified because it depends on quantities that are unknown at the time of formulation. This characteristic is shared by many economic and financial planning models, which may depend for example on future interest rates, future demands for a product, or future commodity prices, but uncertainty can arise naturally in almost any type of application.

Rather than just use a "best guess" for the uncertain quantities, more useful solutions may be obtained by incorporating additional knowledge about these quantities into the model. For example, they may know a number of possible scenarios for the uncertain demand, along with estimates of the probabilities of each scenario. Stochastic optimization algorithms use these quantifications of the uncertainty to produce solutions that optimize the expected performance of the model. Related paradigms for dealing with uncertain data in the model include chance constrained optimization, in which we ensure that the variables x satisfy the given constraints to some specified probability, and robust optimization, in which certain constraints are required to hold for all possible values of the uncertain data.

Many algorithms for stochastic optimization do, however, proceed by formulating one or more deterministic sub-problems, each of which can be solved by the aforementioned techniques. Stochastic and robust optimization are seeing a great deal of recent research activity.

15.4.6 Convexity

The concept of convexity is fundamental in optimization. Many practical problems possess this property, which generally makes them easier to solve both in theory and practice. If the objective function in the optimization problem (1) and the feasible region are both convex, then any local solution of the problem is in fact a global solution. The term convex programming is used to describe a special case of the general constrained optimization problem in which:

- Objective function is convex,
- Equality constraint functions ci $(\cdot)$, $i \in E$, are linear, and
- Inequality constraint functions ci $(\cdot)$, $i \in I$, are concave.

Optimization algorithms are iterative: they begin with an initial guess of the variable x and generate a sequence of improved estimates (called “iterates”) until they terminate, hopefully at a solution. The strategy used to move from one iterate to the next distinguishes one algorithm from another. Most strategies make use of the values of the objective function f, the constraint functions c_i, and possibly the first and second derivatives of these functions.

Some algorithms accumulate information gathered at previous iterations, while others use only local information obtained at the current point. Regardless of these specifics, good algorithms should possess the following properties:

- Robustness: they should perform well on a wide variety of problems in their class, for all reasonable values of the starting point.
- Efficiency: they should not require excessive computer time or storage.
- Accuracy: they should be able to identify a solution with precision, without being overly sensitive to errors in the data or to the arithmetic rounding errors that occur when the algorithm is implemented on a computer.

These goals may conflict. For example, a rapidly convergent method for a large unconstrained nonlinear problem may require too much computer memory. On the other hand, a robust method may also be the slowest. Tradeoffs between convergence rate and storage requirements, and between robustness and speed, and so on, are central issues in numerical optimization.

The mathematical theory of optimization is used both to characterize optimal points and to provide the basis for most algorithms. It is not possible to have a good understanding of numerical optimization without a firm grasp of the supporting theory. Accordingly, this chapter gives a solid, though not comprehensive, treatment of optimality conditions, as well as convergence analysis that reveals the strengths and weaknesses of some of the most important algorithms.

15.5 Nonlinear Programming Techniques

Most MDO systems for complex engineering design will have to assume the general case that at least the objective function or one of the constraint functions are nonlinear. In that case a nonlinear optimization technique is used. In this category there are gradient-based methods that rely on first and second derivatives of the objective function and constraint functions to determine the search direction and update the DV. If they cannot be calculated implicitly, these derivatives can be approximated using a finite-difference method. Stochastic methods such as GAs do not require gradients and have therefore gained significant interest over the past few years.

15.5.1 Sequential Quadratic Programming

Sequential quadratic programming (SQP) methods are iterative methods that solve at the *k*th iteration a quadratic sub-problem (QP) of the form QP [22, 23]:

$$\text{Minimise}: \quad \min d^t H_k d + \nabla f(x_k)^t d \tag{15.2}$$

subject to

$$\begin{aligned} &\nabla h_i(x_k)^t d + h_i(x_k) = 0, \quad i = 1, \ldots, p, \\ &\nabla g_j(x_k)^t d + g_j(x_k) \leq 0, \quad j = p+1, \ldots, q \end{aligned}$$

where d is the search direction and H_k is a positive definite approximation to the Hessian matrix of Lagrangian function of problem (P). The Lagrangian function is given by:

$$L(x,u,v) = f(x) + \sum_{i=1}^{p} u_i h_i(x) + \sum_{j=p+1}^{q} v_j g_j(x) \tag{15.3}$$

where u_i and v_j are the Lagrangian multipliers. The sub-problem (QP) can be solved by using the active set strategy. The solution d_k is used to generate a new iterate:

$$x_{k+1} = x_k + \alpha_k d_k \tag{15.4}$$

where the step-length parameter $\alpha_k \in (0,1]$ depends on some line search techniques. At each iteration, the matrix H_k is updated according to any of the quasi-Newton method. The most preferable method to update H_k is Broyden-Fletcher-Goldfarb-Shanno (BFGS) method, where H_k is initially set to the identity matrix I and updated using the equation:

$$H_{k+1} = H_k + \frac{y_k y_k^t}{s_k y_k^t} - \frac{H_k s_k s_k^t H_k}{s_k^t H_k s_k} \tag{15.5}$$

where

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla L(x_{k+1}, u_{k+1}, v_{k+1}) - \nabla L(x_k, u_k, v_k) \tag{15.6}$$

15.5.2 Generalized Reduced Gradient

The generalized reduced gradient (GRG) transforms inequality constraints into equality constraints by introducing slack variables [24]. Hence all the constraints in (P) are of equality form and can be represented as follows:

$$h_i(x) = 0, \quad i = 1, \ldots, q \tag{15.7}$$

where x contains both original variables and slacks. Variables are divided into dependent, x_D, and independent, x_I, variables (or basic and nonbasic, resp.):

$$x = \begin{bmatrix} x_D \\ \cdots \\ x_I \end{bmatrix} \tag{15.8}$$

The names of basic and nonbasic variables are from linear programming. Similarly, the gradient of the objective function bounds and the Jacobian matrix J may be partitioned as follows:

$$\begin{gathered} a = \begin{bmatrix} a_D \\ \cdots \\ a_I \end{bmatrix}, \quad b = \begin{bmatrix} b_D \\ \cdots \\ b_I \end{bmatrix}, \quad \nabla f(x) = \begin{bmatrix} \nabla_D f(x) \\ \cdots \\ \nabla_I f(x) \end{bmatrix} \\ J(x) = \begin{bmatrix} \nabla_D h_1(x) & \cdots & \nabla_I h_1(x) \\ \cdots & \cdots & \cdots \\ \nabla_D h_q(x) & \cdots & \nabla_I h_q(x) \end{bmatrix} \end{gathered} \tag{15.9}$$

Let x^0 be an initial feasible solution, which satisfies equality constraints and bound constraints. Note that basic variables must be selected so that $J_D(x^0)$ is nonsingular. The reduced gradient vector is determined as follows:

$$g_I = \nabla_I f(x^0) - \nabla_D f(x^0)(J_D(x^0))^{-1} J_I(x^0) \tag{15.10}$$

The search directions for the independent and the dependent variables are given by:

$$d_j = \begin{cases} 0, & \text{if } x_i^0 = a_i,\ g_i > 0 \\ 0, & \text{if } x_i^0 = b_i,\ g_i < 0 \\ -g_i, & \text{otherwise} \end{cases} \tag{15.11}$$

$$d_D = -(J_D(x^0))^{-1} J_I(x^0) d_t \tag{15.12}$$

A line search is performed to find the step length $f¿$ as the solution to the following problem:

$$\min f\left(x^0 + \alpha\, d\right) \tag{15.13}$$

Subject to: $0 \le \alpha \le \alpha_{\max}$, where $\alpha_{\max} = \sup\left\{\frac{\alpha}{a} \le x^0 \le x^0 + \alpha d \le b\right\}$.

The optimal solution α^* to the problem gives the next solution: $x^1 = x^0 + \alpha \cdot d$.

15.5.3 Genetic Algorithms

In the computer science field of artificial intelligence, a GA, evolutionary algorithms (EA) and particle swarm optimization (PSO) include a search heuristic that mimics the process of natural selection (biology-mimicking) [25–29]. This heuristic, also sometimes called a meta-heuristic, is routinely used to generate useful solutions to optimization and search problems [30]. GAs belong to the larger class of EA, which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. GAs are stochastic optimization algorithms based upon the principles of evolution observed in nature. Because of their power and ease of implementation, the use of GAs has noticeably increased in recent years. Unlike the gradient methods, they have no requirements on convexity, differentiability, and continuity of the objective, and constraint functions. These significant characteristics of GAs increase their popularity in applications. The basic GA can be summarized by the following steps:

1. Generate an initial population of possible solution (chromosomes) randomly,
2. Evaluate the fitness of each chromosome in the initial population,
3. Select chromosomes that will have their information passed on to the next generation,
4. Cross over the selected chromosomes to produce new offspring chromosomes,
5. Mutate the genes of the offspring chromosomes,
6. Repeat steps (3) through (5) until a new population of chromosomes is created,
7. Evaluate each of the chromosomes in the new population,
8. Go back to step (3) unless some predefined termination condition is satisfied.

GAs are directly applicable only to unconstrained problems. In the application of GAs to constrained nonlinear programming problems, chromosomes in the initial population or those generated by genetic operators during the evolutionary process generally violate the constraints, resulting in infeasible chromosomes. During the past few years, several methods were proposed for handling constraints, grouped into the following four categories:

- Preserving feasibility of solutions,
- Penalty functions,
- Search for feasible solutions,
- Hybrid methods.

Penalty function methods are the most popular methods used in the GAs for constrained optimization problems. These methods transform a constrained problem into an unconstrained one by penalizing infeasible solutions. Penalty is imposed by adding to the objective function $f(x)$ a positive quantity to reduce fitness values of such infeasible solutions:

$$\hat{f}(x) = \begin{cases} f(x) & \text{if } x \in F \\ f(x) + p(x) & \text{otherwise} \end{cases} \tag{15.14}$$

where $\hat{f}(x)$ is the fitness function and $p(x)$ is the penalty function whose value is positive. The design of the penalty function $p(x)$ is the main difficulty of penalty function methods. Several forms of penalty functions are available in the literature.

15.6 Multi-modal and Multi-objective Design Optimization

Optimization problems are often multi-modal: they possess multiple good solutions. They could all be globally good (same cost function value) or there could be a mix of globally good and locally good solutions. Obtaining all (or at least some of) the multiple solutions is the goal of a multi-modal optimizer [31–39].

Classical optimization techniques due to their iterative approach do not perform satisfactorily when they are used to obtain multiple solutions, since it is not guaranteed that different solutions will be obtained even with different starting points in multiple runs of the algorithm. EA however are very popular approaches to obtain multiple solutions in a multi-modal optimization task.

Real life engineering designs often have more than one conflicting objective functions thus requiring a multi-objective optimization approach. The multi-objective optimization becomes more difficult with increasing number of objectives and it has been shown in that existing multi-objective optimization algorithms do not perform well with more than five objectives. The optimization identifies several solutions that are good considering the objective functions. These are called Pareto solutions.

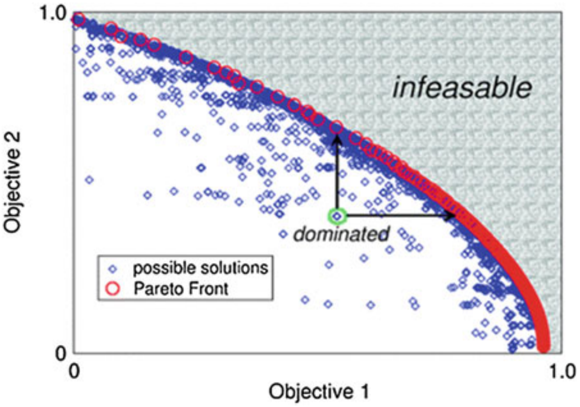


Fig. 15.6 Pareto front with two objective functions

Figure 15.6 shows a Pareto front defining the solutions for a two objective (F_1 and F_2) problem. Multi-objective optimization has been applied in many fields of science, including engineering, economics and logistics where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives.

For a nontrivial multi-objective optimization problem, there does not exist a single solution that simultaneously optimizes each objective. In that case, the objective functions are said to be conflicting, and there exists a, possibly infinite number of, Pareto optimal solutions. A solution is called non-dominated, Pareto optimal, Pareto efficient or non-inferior, if none of the objective functions can be improved in value without degrading some of the other objective values. Without additional subjective preference information, all Pareto optimal solutions are considered equally good (as vectors cannot be ordered completely). Researchers study multi-objective optimization problems from different viewpoints and, thus, there exist different solution philosophies and goals when setting and solving them. The goal may be to find a representative set of Pareto optimal solutions, and/or quantify the trade-offs in satisfying the different objectives, and/or finding a single solution that satisfies the subjective preferences of a human decision maker (DM).

15.7 MDO Architectures

It is a fact of physics that in an engineering system such as a road vehicle there are interactions among the physical phenomena and the vehicle hardware parts. These interactions make the vehicle a synergistic whole that is greater than the sum of its parts. Taking advantage of that synergy is the mark of a good design but the web of interactions is difficult to untangle. That difficulty combined with the need to partition the work into subtasks executed simultaneously to compress the project time gave rise to the practice of dividing the detailed design work into specialty areas, each area centered on a physical phenomenon, e.g. stress and strain, or on a hardware subsystem, e.g. the car suspension. The above practice has achieved its purpose of developing a broad work front and compressing project time but on the downside it impeded trade-offs across the subtasks boundaries making the design of the vehicle fall somewhat short of optimal.

The MDO has evolved as a new discipline that provides a body of methods and techniques to assist engineers in moving engineering system design closer to the global optimum. Parallel to the development of these methodologies, a number of software packages have been created to facilitate integration of codes, data, and user interface. These packages, such as FIDO, iSIGHT, LMS Optimus, and DAKOTA, are often referred to as frameworks [40].

The key concept in several of these MDO methods is a decomposition of the design task into subtasks performed independently in each of the modules, and a system-level or coordination task giving rise to a two-level optimization. In general, decomposition was motivated by the obvious need to distribute work over many people and computers to compress the task calendar time. Equally important benefit from the decomposition is granting autonomy to the groups of engineers responsible for each particular subtask in choosing their methods and tools for the subtask execution. As an additional advantage, the concurrent execution of the subtasks fits well the technology of massively concurrent processing that is now becoming available (see Chap. 4).

Several requirements exist for a framework to provide an easy-to-use and robust MDO environment:

- Provide for quick and easy linking of analysis tools. The set of analysis tools to be linked could involve such tools as COTS software (CAD, CAE, CAM), legacy (in house) codes, spreadsheets, databases, and tools to capture user's knowledge.
- Provide effective support for geographically distributed modelling and optimization, through CORBA client-server compliancy of the software tools and models, facilitating both tight and loose collaboration, ranging from OEMs, customers, suppliers and consultants.
- Access to efficient parametric study capability such as design of experiments (DOE) based procedures, including full factorial designs, fractional factorial designs (orthogonal arrays), central composite designs and Latin hypercube designs.
- Access to a full range of optimization search strategies ranging from gradient based numerical optimization, simulated annealing and GAs and most importantly, an optimization advisor that can appropriately recommend the optimization algorithm or a combination of algorithms (hybrid optimization plan) to be used for solution of the user problem.
- Access to a full range of model approximation techniques such as polynomial, Kriging, or neural networks based response surfaces, sensitivity based Taylor series linearization, and variable complexity models.

- Provide the ability to perform trade-off studies between different design responses.
- Provide support to easy description and set up of MDO problems using formal, decomposition based MDO methods such as global sensitivity equations (GSE) based Optimization, collaborative optimization (CO), and bi-level integrated system synthesis (BLISS).
- Provide the ability to account for uncertainties in design using probabilistic constraints and robust design formulations.
- Framework should provide support for parallel computing, including parallel invocations of simulation codes as well as subsystem optimizations and intelligent load balancing [41].
- Provide effective support of visualization of design data both at runtime and post-processing stages.
- Provide effective support for database management through structured query language (SQL) interface for data storage/access/manipulation both at the local (subsystem) and global (system) levels.
- The framework should be easy to use in terms of user interface for MDO, extensible for user addition of optimization solvers, scalable for large scale problem solving and provide for robust performance [42].

A brief description of some of the formal MDO architecture used to solve the system optimization problem is provided in the following sub-sections [43–58].

15.7.1 Multidisciplinary Design Feasible (MDF)

The All-in-One (A-i-O) method, also referred to as multidisciplinary feasibility (MDF), is the most common way of approaching the solution of MDO problems. In this method, the vector of DV x is provided to the coupled system of analysis disciplines and a complete multidisciplinary analysis (MDA) is performed via a fixed-point iteration with that value of x to obtain the system MDA output variable $y(x)$ that is then used in evaluating the objective $f(x, y(x))$ and the constraints $c(x, y(x))$. The optimization problem is:

$$\min f(z, x, y(x, y, z)) \tag{15.15}$$

With respect to: z, x and subject to: $c(z, x, y(x, y, z)) \geq 0$

If a gradient-based method is used to solve the above problem, then a complete MDA is necessary not just at each iteration but at every point where the derivatives are to be evaluated. Thus, attaining multidisciplinary compatibility can be prohibitively expensive in realistic application. Figure 15.7 shows the data flow in an A-i-O analysis and optimization. The different disciplines are considered as a single monolithic analysis. This is conceptually very simple, and once all disciplines are coupled to form one single MDA module, one can use the same techniques that are

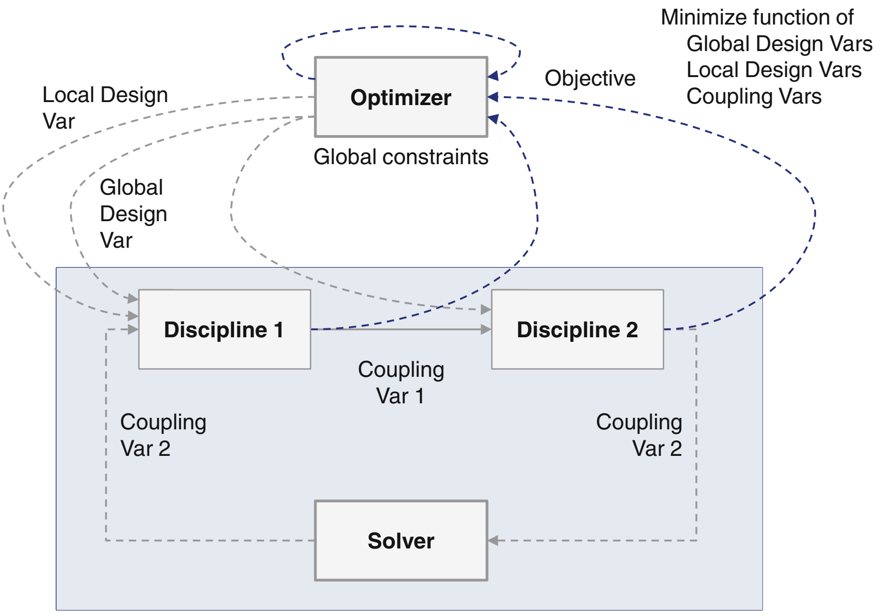


Fig. 15.7 MDF architecture

used in single discipline optimization. One of the disadvantages of this approach is that the solution of the one system might be very costly and does not exploit the potentially weak coupling between some of the disciplines that would enable the division into different analyses modules that might run in parallel. The only opportunity for parallelizing the optimization procedure would be the use of different processes for each member of the population when using a GA or running the analyses for different design points when calculating gradients by finite differencing or when evaluating the points for a response surface.

15.7.2 Individual Discipline Feasible (IDF)

The IDF formulation provides a way to avoid a complete MDA at optimization. IDF maintains individual discipline feasibility, while allowing the optimizer to drive the individual disciplines to MDF and optimality by controlling the interdisciplinary coupling variables. In IDF, the specific analysis variables that represent communication, or coupling, between analysis disciplines are treated as optimization variables and are in fact indistinguishable from DV from the point of view of a single analysis discipline solver. The IDF architecture is shown in Fig. 15.8.

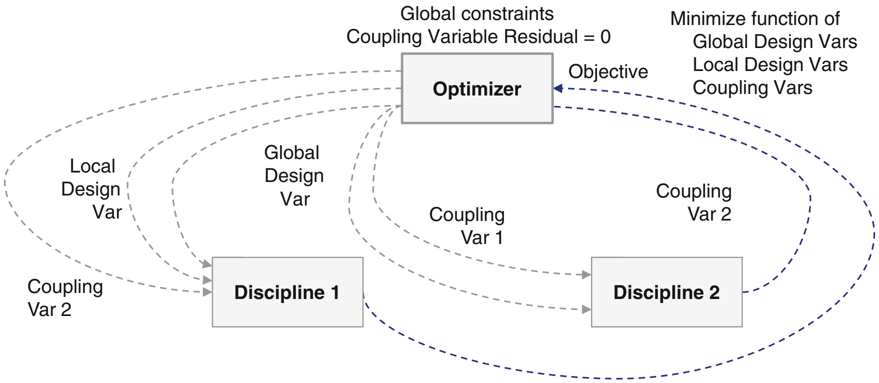


Fig. 15.8 IDF architecture

15.7.3 Simultaneous Analysis and Design (SAND)

This approach optimizes the design and solves the governing equations at the same time by posing the problem as:

$$\min f(z,x,y) \tag{15.16}$$

With respect to: x, y, z, subject to: $c(z,x,y(x,y,z)) \geq 0, \quad R(x,y,z) = 0$

SAND is not inherently multidisciplinary and can also be used for single discipline optimization problems. It can be very efficient since we solve the whole problem at once, but if a very efficient analysis is already in place, it is usually not worthwhile to use SAND. To implement SAND, one needs to calculate the residual of each governing equation.

15.7.4 Optimizer-Based Decomposition (OBD)

The main idea of this method is to use the optimizer to enforce inter-disciplinary compatibility. Instead of iterating the MDA to converge the coupling variables y, these coupling variables are given by the optimizer as a guess, or target, y^t. The new optimization problem can be written as:

$$\min f(z,x,y(x,y^t,z)) \tag{15.17}$$

With respect to: x, y^t, z, subject to: $c(z,x,y(x,y^t,z)) \leq 0, \quad y_i^t - y_i(x,y^t,z) = 0$

The number of DV has increased, and is equal to the number of original DV plus the number of coupling variables. This increases the size of the optimization problem, but conveniently decouples all the analyses, which can now be solved in parallel without intercommunication. Note that when using gradient-based optimization, the gradients $\partial f/\partial y^t$ and $\partial c/\partial y^t$ must also be calculated.

15.7.5 Collaborative Optimization (CO)

The CO architecture, shown in Fig. 15.9, is designed to promote disciplinary autonomy while achieving interdisciplinary compatibility. The optimization problem is decomposed into optimization subproblems corresponding to the different disciplines. Each subproblem is given control over its own set of local DV, is responsible for satisfying its own set of local constraints and does not know about the other disciplines' DV or constraints. The objective of each sub-problem is to agree on the values of the coupling variables with the other disciplines. A system-level optimizer is used to coordinate this process while minimizing the overall objective. The system level optimization problem can be stated as:

$$\min f(z^t, y^t) \tag{15.18}$$

With respect to: z^t, y^t, subject to: $j_i^*(z_i^t, z_i^*, y^t, y_i^*(x_i^*, y^t, z_i^*)) = 0, \quad i = 1, \ldots, N$ where N is the number of disciplines, and the subscript * represents the results from the solution of the i^h discipline optimization sub-problem:

$$\min j_i(z_i^t, z_i, y^t, y(x_i, y^t, z_i)) = \Sigma\left(1 - \frac{z_i}{z_i^t}\right)^2 + \Sigma\left(1 - \frac{y_i}{y_i^t}\right)^2 \tag{15.19}$$

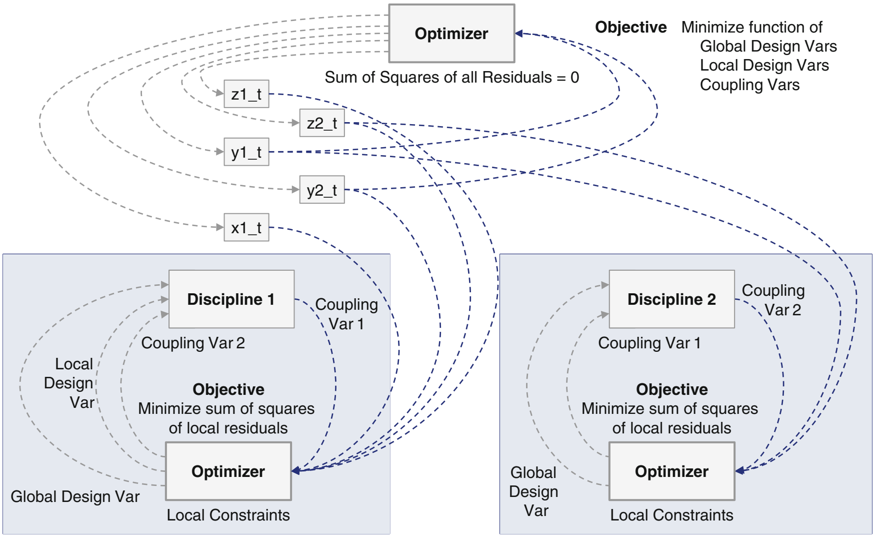


Fig. 15.9 Collaborative optimization architecture

With respect to: z_i, x_i, subject to: $c_i(x_i, z_i, y_i(x_i, y^t, z_i)) \geq 0$, where c is the vector of constraints for the i^h discipline. J is a measure of interdisciplinary discrepancy that we want to drive to zero at the system level. The solution of this sub-problem returns j_i^*. Note that post-optimality sensitivities are needed.

15.7.6 Concurrent Subspace Optimization (CSSO)

The CSSO method is also a decomposition-based strategy that allows for the disciplines to run decoupled from each other. Again, the multiple subspace optimization problems are driven by a system-level optimizer that provides overall coordination. Each sub-problem in CSSO uses approximations to non-local disciplinary coupling variables to estimate the influence of these variables on the system-level objective and constraints. The subspace optimization problem for the i^h discipline is given by:

$$\min f(z, x, \tilde{y}_j(z_i, x_i), y_i(x_i, \tilde{y}_j, z)) \tag{15.20}$$

With respect to: z_i, x_i, subject to: $c(x_i, z, \tilde{y}_j, (z_i, x_i), y_i(z_i, x_i, \tilde{y}_j)) \leq 0$, where $j \neq i$ and $y_i = (z, x_j)$ are the approximations to the other disciplines' coupling variables, or states. These approximations can be made using response surfaces. The system-level optimizer solves the following problem:

$$\min f(z, x, \tilde{y}(z, x)) \tag{15.21}$$

With respect to: z, x, subject to: $c(z, c, \tilde{y}(z, x)) \leq 0$.

After each iteration of the system-level optimizer, a MDA is performed to update the model which gives the approximate response of all coupling variables $\tilde{y}$.

15.7.7 Bi-Level Integrated System Synthesis (BLISS)

The recently introduced BLISS method uses a gradient-guided path to reach the improved system design, alternating between the set of modular design subspaces (disciplinary problems) and the system level design space. BLISS is an A-i-O like method in that a complete system analysis performed to maintain MDF at the beginning of each cycle of the path. With BLISS, the general system optimization problem is decomposed into a set of local optimizations dealing with a large number of detailed local DV (X) and a system level optimization dealing with a relatively small number of global variables (Z) in comparison with the other MDO methods. In optimization it is useful to distinguish between X and Z because:

- The X variables are associated with individual components and, therefore, they tend to be clustered. Also, the constraints they govern directly, e.g. the stringer buckling in built-up, thin-walled structures typical of aerospace vehicles, tend to be highly nonlinear. The total number of the X variables in a typical airframe is in thousands but their number in an individual substructure is likely to be quite small.
- The number of Z variables is much smaller than the total number of X variables.
- Nonlinearity of the overall behavior constraints, such as displacements, with respect to X and Z tends to be weaker than that of the local strength and buckling constraints.

With BLISS, the solution of the system level problem is obtained using either (i) the optimum sensitivity derivatives of the behavior/state (Y) variables with respect to system level DV (Z) and the Lagrange multipliers of the constraints obtained at the solution of the disciplinary optimizations, or (ii) a response surface constructed using either the system analysis solutions or the subsystem optimum solutions.

15.8 Case Studies in Multidisciplinary Design Optimization

15.8.1 Optimization of Automotive Structures Under Multiple Crash and Vibration Design Criteria

This design problem is aimed at reducing the overall mass of a vehicle by focusing on a group of structural components that are influential in both energy absorption (crashworthiness) and vehicle stiffness (vibration) [9]. Through a preliminary analysis, 22 components were selected as highlighted in Fig. 15.10. These components have a combined mass of 105.25 kg for 8 % of the crash-model mass at 1,333 kg and approximately 45 % of the vibration-model mass at 233 kg. Due to the vehicle model symmetry, the 22 components are represented by 15 wall-thickness DV denoted by $x1$ through $x15$. The 22 components contribute to 42, 27 and 36 % of the total energy absorbed in full frontal impact (FFI), offset frontal impact (OFI) and side impact (SI), respectively. In this study, the scope to sizing optimization focused on a subset of components that show considerable influence on both crash and vibration characteristics of the vehicle. The design optimization problem is formulated as:

$$\min f(x) \tag{15.22}$$

Subject to: $g_i(x) = R_i(x) - R_i^b(x) \le 0$ with $i = 1, \ldots, 8$, $g_i(x) = R_i^b(x) - R^b(x) \le 0$, with $i = 9, \ldots, 14$, $0.5x_j^b \le x_j \le 1.5x_j^b$ with $j = 1, \ldots, 15$

Where the objective function $f(x)$ represents the total mass of the selected components shown in Fig. 15.13. In the first group of design constraints, R_i, $i = 1, 8$ represent Toeboard Intrusion, Dash Intrusion for FFI and OFI, Door Intrusion for SI

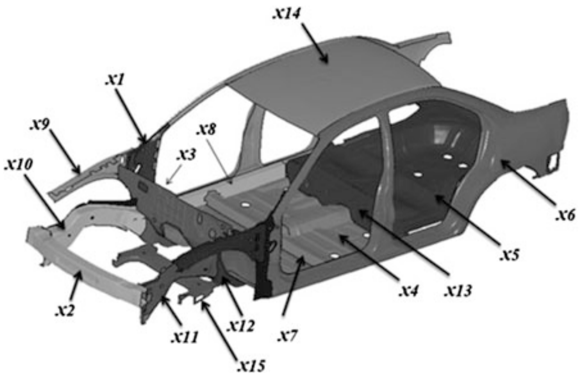


Fig. 15.10 Selected vehicle components and associated design variables [9]

in all three scenarios; all of these responses are required to be no greater than the corresponding values in the baseline model denoted by R_i^b, $i = 1, 8$. In the second group, R_i, $i = 9, 11$ represent the internal energy absorbed by the 22 components combined in the three crash scenarios whereas R_i, $i = 12, 14$ represent the three selected natural frequencies of the vibration model, with all required to be no less than the corresponding values in the baseline model. The design space is defined by 15 DV that represent the wall thicknesses of the components, with each bounded to within ±50 % of the respective baseline value. With the response surrogate models developed, the optimization problem was solved using SQP. Given the gradient-based search approach in SQP and the non-convex nature of the combined crash–vibration vehicle optimization problem, the problem was solved using 15 randomly selected initial design points with the best result corresponding to the optimum design defined in Table 15.1.

The objective function history showed 16 iterations for finding the optimum design point. A complete iteration refers to solution of the direction finding QP and step size associated with SQP. The optimization took a total of 163 analysis calls and approximately 20 min for the process to complete. The optimum mass was 101.49 kg for the 22 selected components in comparison to the baseline mass of 105.25 kg for a reduction of approximately 3.6 %.

Table 15.2 shows that the optimum design based on crashworthiness requirements alone reduces the overall vehicle stiffness as indicated by the frequency reduction of 6.4 % in the first mode, 5.7 % in second mode and 3.9 % in third mode. Frequencies of the current optimised design are the same as those in the baseline design. Out of 15 DV in the crash–vibration vehicle optimum, nine have increased and six have decreased relative to the respective baseline values with design variable five reaching its lower bound.

The general assessment of the results found in this study is that the crash and vibration responses are in competition. Vehicle components have to change thickness in such a way that both criteria are satisfied while weight is minimised. This is evident by the significant difference in optimised mass of the designs using

Table 15.1 Design variable bounds and optimum values [9]

Component	Lower bound (mm)	Baseline (mm)	Upper bound (mm)	Optimum (mm)
1 A-Pillar	0.806	1.611	2.417	1.471
2 Front bumper	0.978	1.956	2.934	2.169
3 Firewall	0.368	0.735	1.103	0.913
4 Front floor panel	0.353	0.705	1.058	0.560
5 Rear cabin floor	0.353	0.706	1.059	0.387
6 Outer cabin	0.415	0.829	1.244	0.897
7 Seat reinforcement	0.341	0.682	1.023	1.009
8 Cabin mid-rail	0.525	1.050	1.575	1.287
9 Shotgun	0.762	1.524	2.286	1.670
10 Inner side rail	0.948	1.895	2.843	1.694
11 Outer side rail	0.761	1.522	2.283	1.654
12 Side rail extension	0.948	1.895	2.843	1.952
13 Rear plate	0.355	0.710	1.065	0.668
14 Roof	0.351	0.702	1.053	0.815
15 Suspension frame	1.303	2.606	3.909	1.923

Table 15.2 Comparison of the baseline and optimum model [9]

Response	Baseline	Optimum	Diff (%)
FFI toe int (mm)	157.07	160.29	2.05
FFI dash int (mm)	122.06	118.30	−3.08
FFI accel (g)	63.51	59.12	−6.91
FFI int eng (kJ)	62.31	62.35	0.06
SI door int (mm)	313.93	311.09	−0.90
SI accel (g)	47.88	47.71	−0.36
SI Int eng (kJ)	22.37	23.51	5.10
OFI toe int (mm)	273.48	229.29	−16.16
OFI dash int (mm)	246.94	200.52	−18.80
OFI accel (g)	35.02	33.91	−3.17
OFI int eng (kJ)	39.42	41.46	5.17
Frq1 (Hz)	35.39	35.39	0.00
Frq2 (Hz)	36.23	36.23	0.00
Frq3 (Hz)	38.37	38.37	0.00
Mass (kg)	105.25	101.49	−3.60

crashworthiness and vibration, 101.49 kg, and crashworthiness alone, 88 kg. Adding vibration considerations to the optimization problem produced a design with less weight reduction but without sacrificing structural rigidity.

15.8.2 *Multidisciplinary Design Optimization of a Regional Aircraft Wing Box*

The structural design of an airframe is determined by multidisciplinary criteria (stress, fatigue, buckling, control surface effectiveness, flutter and weight etc.) [59]. Several thousands of structural sizes of stringers, panels, ribs etc. have to be determined considering hundreds of thousands of requirements to find an optimum solution, i.e. a design fulfilling all requirements with a minimum weight or minimum cost respectively. MDO techniques were successfully applied in sizing the wing boxes of the newly developed regional jet family. Figure 15.11 shows how the MDO process has been organized based on MSC Nastran SOL 200. Before the numerical optimization loop can be started, the design must be parameterized and all disciplines must make available their analysis models and design criteria. The wing box sizes can be parameterized by simply assigning DV to the FE-properties (cross-sections, thicknesses). The linking scheme between FE-properties and the independent DV is represented by the Design Model and it is based on constructive, manufacturing as well as numerical considerations.

Structural Analysis provides all relevant structural responses based on the analysis models and the current set of DV. The Sensitivity Analysis calculates the first derivatives of all responses with respect to the independent DV. A very important feature of MSC NASTRAN is the External Server, which allows the integration of user-defined design criteria described by Fortran routines. It therefore can be used to integrate various detailed design constraints, which are dependent on NASTRAN responses (stresses, displacements etc.). All detailed wing buckling

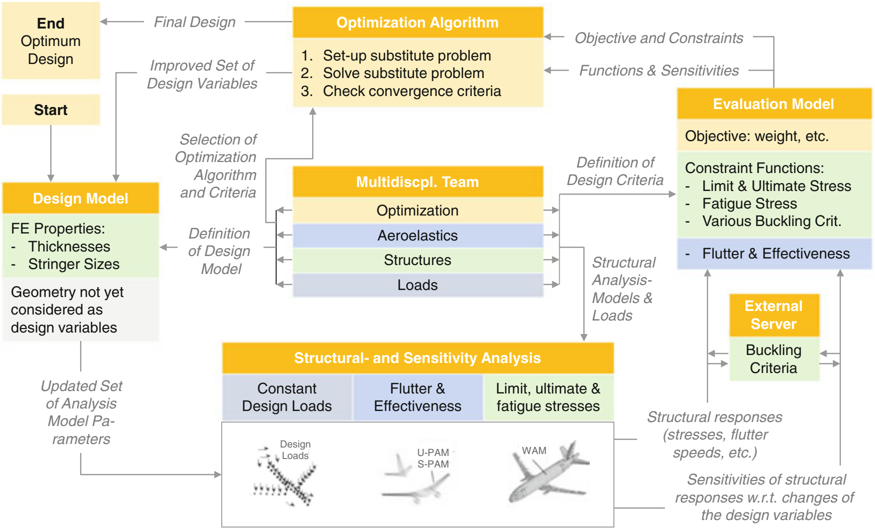


Fig. 15.11 Wing structural design process with multidisciplinary design optimization [59]

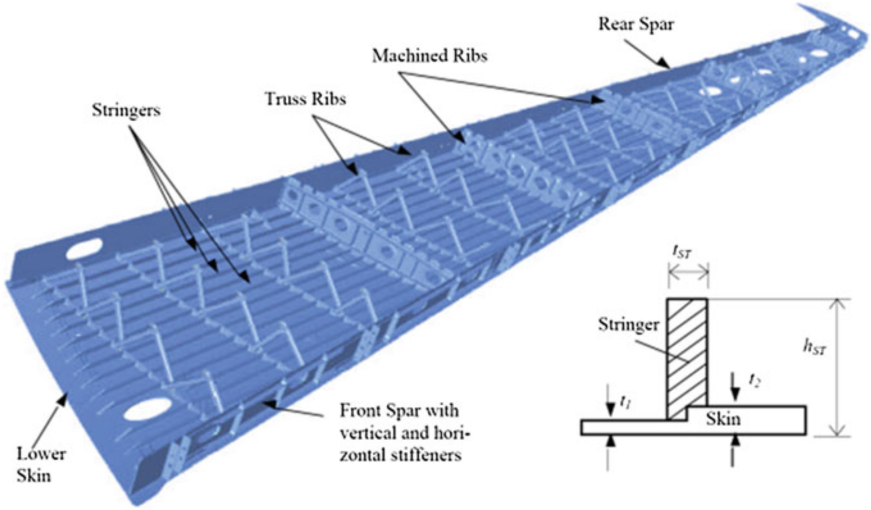


Fig. 15.12 General layout of the outer wing box [59]

criteria (skin, stringer, and column buckling and stringer crippling) have been implemented within this External Server. The objective function and all constraints are mathematically defined in the Evaluation Model based on structural responses. They are then transferred to the optimization algorithm to find an improved set of DV. This set is converted into a new set of FE-Properties in order to initiate the next cycle. As a result of the non-linear relationship between the constraints and DV, the full process must be repeated several times until an optimum design is found.

Figure 15.12 shows the lower panel, the spars and the internal ribs of the outer wing box. The panels consist of a skin stiffened by rectangular stringers. The number of stringers decreases from inboard to outboard due to wing taper. Ribs are connected both to spars and panels. The panels and spars carry global bending and torsional loads, whilst the primary function of ribs is to stabilize the whole structure and transfer the local air load into the wing box. Since the panels and the spars are machined from solids, the sizes of skin and stringers can change between each pocket surrounded by two stringers and two ribs. It is even possible to have a varying skin thickness or varying stringer height within a pocket to provide the locally required strength and stiffness with a minimum weight. This results in several thousands of independent parameters defining the whole wing box design.

The level of meshing detail of the wing model is shown in Fig. 15.13. This model is the same finite element model that is typically used for sizing by traditional methods. The wing box model mainly consists of Shell and Beam elements representing skin and stringers/stiffeners, respectively. Combining the wing box with fuselage and empennage FE models results in a Whole Aircraft Shell FE-Model (WAM) of approximately 250,000 degrees of freedom.

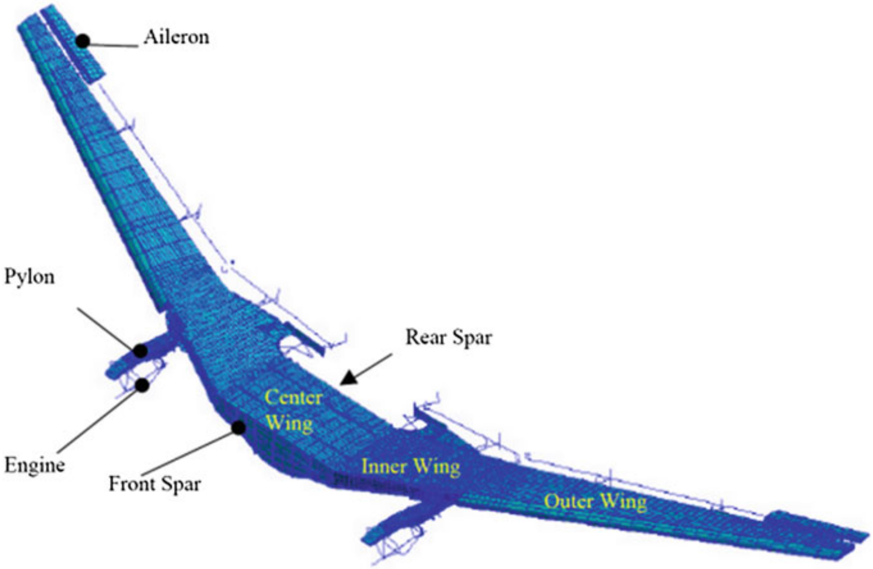


Fig. 15.13 FE-Model of the wing (93,000 DOF) [59]

The most important structural sizes of the wing box comprise the skin thickness and the stringer height and thickness. This applies to the panels as well as to the spars. Linear equations define the relationship between the independent DV and the FE-Properties representing skin and stringers sizes. For the purpose of applying buckling constraints, the upper and lower surfaces of the wing are subdivided into so called Buckling Fields. Each buckling field consists of the finite element mesh between two adjacent span wise ribs and two chord wise adjacent sets of stringers. Mechanically speaking, this corresponds to each stiffened sub-panel on the wing. The skin elements within each buckling field were linked together and represented by a single design variable.

The same applies to the stringer properties. The stringer offset and the second moment of inertia are updated after the optimization before the analysis of the new sizes takes place. The overall design model of the whole wing was structured corresponding to the major wing sections. Each of these components was subdivided again into upper and lower panels, front and rear spar, as well as skin and stringers. With this arrangement the total number of DV reached 2,515. Minimum and maximum sizes due to manufacturing or lightning protection were considered as lower and upper bounds for the FE-Properties. Special PATRAN command language (PCL) tools were developed to automate the creation and update of all corresponding design model input data for Nastran SOL 200.

The mathematical objective of the optimization process is to find a minimum feasible weight. All relevant wing box sizing criteria comprising of limit, ultimate and fatigue stresses, buckling criteria, manufacturing requirements, control surface

Table 15.3 Wing box design constraints [59]

Structure	Constraint type	Center	Inner	Outer	Load cases	Constraints
Skin elements	von-Mises stress	416	1,132	562	96 ultimate	202,560
Stringer and horizontal stiffener elements	Axial, tension and compression stress	476	985	622	96 ultimate	199,488
Spar web elements	Shear stress	148	525	280	96 ultimate	91,488
Buckling field skin	Panel buckling	147	251	364	96 ultimate	75,552
Buckling field skin	Crippling	147	251	364	96 ultimate	75,552
BF stringers	Stringer buckling	147	251	364	96 ultimate	75,552
BF skin and stringer	Euler buckling	147	251	364	96 ultimate	75,552
Lower panel skin	Principle stress	384	1042	508	3 fatigue	5,502
Panel joints	Principle stress	20	108	42	3 fatigue	510
Spar web elements	Principle stress		408		3 fatigue	1,224
Height of adjacent stringers	Maximum step size	120	199	115		434
Stringer thickness to height ration	Minimum ration	431	995	538		1,964
Outer wing box skin	Aileron effectiveness	3 times cases (zero aileron effectiveness)				3
Inner wing box skin	Lowest flutter speed	1 flutter speed limit				1
Total number of constraints						805,402

effectiveness and flutter criteria were applied in the form of in-equality constraints. The buckling constraints were communicated to NASTRAN during the optimization process by the External Server. Fatigue stress constraints were applied to all fatigue sensitive areas of the wing box. These areas included the lower skin panels, major wing box joints (inner and outer wing joint, lower front and rear panel joints), front spar web at the pylon attachment and rear spar web at the landing gear attachment. Due to manufacturing requirements, a minimum stringer thickness to height ratio had to be adhered to. Furthermore, the relative step size of the stringer height was limited in spanwise direction to prevent excessive out-of-plane bending stresses. Table 15.3 gives an overview of all constraints.

The aileron effectiveness constraint is incorporated via a roll performance criterion which is required to be greater than or equal to zero at maximum True Air Speed. A set of three trim cases, i.e. pairs of Mach number and dynamic pressure, were defined from which, on an empirical basis, the zero effectiveness curve can be extrapolated to maximum true air speed by a 2nd order polynomial.

The flutter constraint is defined such that the lowest flutter speed, i.e. a flutter mode with zero damping, must not be lower than a prescribed limit velocity which depends on the flight altitude. All normal modes up to 50 Hz are taken into account in the flutter analysis using the PK-method. The range of air speeds used for the flutter response is limited to a minimum required set. Because of the high computational effort required for flutter optimization, a pre-selection of very few critical flutter cases is indispensable. In order to get an indication for these cases, a

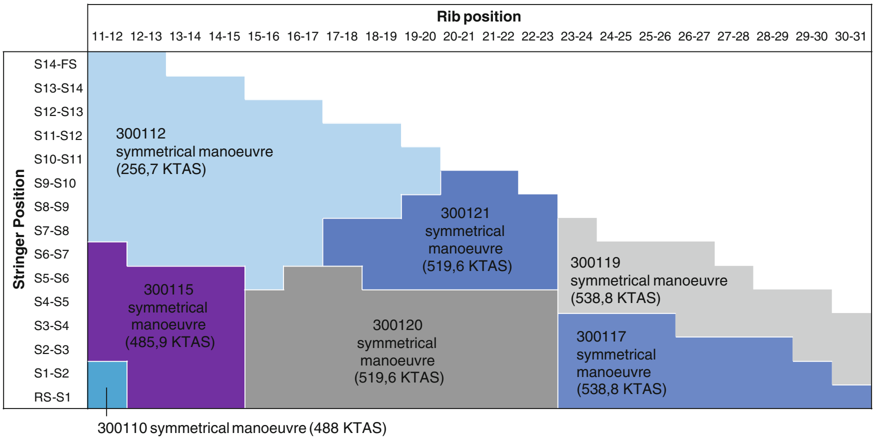


Fig. 15.14 Critical load cases, outer wing upper panels, column buckling criteria [59]

comprehensive flutter check covering the entire flight regime (i.e. a systematic variation of payload mass, fuel mass and flight level) is performed preceding the optimization runs.

A valuable means of displaying the results is shown in Fig. 15.14. In this figure, the driving load cases that design a given section with respect to column buckling of the outer wing are displayed. The driving cases are resulting from symmetrical maneuvers at different speeds, altitudes, flap settings etc. Similar plots for other wing sections and other buckling criteria are also produced. In order to satisfy the aileron reversal constraint the stiffness of the outer wing was locally increased. The skin thicknesses obtained from static optimization were taken as lower bounds. Significant changes are essentially restricted to a zone reaching diagonally from the aileron attachment area inboard to the leading edge, close to the inner wing connection. Similar results were obtained for the lower skin.

15.9 Discussion and Conclusions

MDO is at a crossroad. The focus of MDO has shifted dramatically over the past 25 years as researchers are finding new ways to use MDO methods and tools on a wide array of problems. The potential of MDO has been illustrated in this chapter with a few case studies. A strong research focus in MDO remains to resolve a number of issues that remain an impediment in implementing MDO in all levels of design an development. The major challenges in MDO integration are [60–64]:

- Integrating the designers' skills and experience in the design process. This makes the optimization task difficult to model in an algorithmic form.
- Companies have their own legacy and embedded design improvement processes and tend to resist the implementation of new optimization systems.

- Acquisition and maintenance of hardware and software can be costly.
- Handling large scale qualitative design spaces. It would be ideal to handle quantitative and qualitative information together within one framework.
- Interfaces between feature-based parametric CAD models and optimization models with automatic bi-directional conversions do not exist at present.
- Recently there is interest in design optimization within a dynamic environment. Research is required to extend this to multi-objective design optimization.
- Stochastic optimization, like GAs, is contradictory to conventional deterministic thinking, so how can the user select the most effective technique?
- Scalability is a major challenge for complex systems design optimization. Large-scale design optimization must deal with the complexity.
- There is a lack of understanding about the interaction between components and their behaviors. This may lead to results that cannot be explained.
- Uncertainty is another major challenge for complex systems design optimization. Robust design optimizations are addressing this issue.

There are three major areas of improvement when it comes to use of computing to address engineering design optimization: improve efficiency and speed of optimization and effective use of human knowledge. Large-scale optimization will require more research in topology design, computational power and efficient optimization algorithms. Emergent computing techniques such as grid computing, swarm intelligence and quantum computing improve efficiency and speed of the optimization. Future success of MDO is in application of expert knowledge with existing and emergent algorithmic and computing approaches to large-scale designs, supported by education on optimization.

References

1. de Weck O, Willcox K (2010) Multidisciplinary system design optimization. MIT ESD.77/16.888
2. Bloebaum CL (2013) The role of MDO in the design for complex engineered systems (DCES). In: 9th MDO specialist conference, 11 Apr 2013
3. Schuhmacher G (2008) Numerical optimization methods in the aerospace design process—civil and military applications and benefits. In: 2nd European hyperWorks technology conference, Strasbourg, France, 30 Sept–1 Oct 2008
4. Guo J, Guadagni L (2012) Multidisciplinary design optimization for concurrent engineering of space systems. In: SECESA2012, Lisbon, 17–19 Oct 2012
5. Neufeld D, Chung J, Kamaran B (2008) Development of a flexible MDO architecture for aircraft conceptual design. In: EngOpt 2008—international conference on engineering optimization, Rio de Janeiro, Brazil, 1–5 June 2008
6. Campana EF et al (2009) New global optimization methods for ship design problems. Optim Eng (2009) 10:533–555. doi:10.1007/s11081-009-9085-3
7. Di Pasquale E, Gielczynski G (2010) Multi-disciplinary optimization of railways systems. In: International conference on integrated design and manufacturing in mechanical engineering, Bordeaux, France, 20–22 Oct 2010

8. Breitkopf P, Filomeno Coelho R (eds) (2010) Multidisciplinary design optimization in computational Mechanics. Wiley, New York, ISBN 978-1-84821-138-4
9. Kiani M et al (2013) Surrogate-based optimization of automotive structures under multiple crash and vibration design criteria. Int J Crashworthiness 18(5):473–482
10. Bérend N, Bertrand S (2009) MDO Approach for early design of aerobraking orbital transfer vehicles. Acta Astronaut 65:1668–1678
11. Schoofs AJG (1993) Structural optimization history and state-of-the-art, topics in applied mechanics. Kluwer Academic Publishers, Dordrecht, pp 339–345
12. Schmit LA (1960) Structural design by systematic synthesis. In: Proceedings of the 2nd conference on electronic computation, ASCE, New York, pp 105–132
13. Venkayya VB (1978) Structural optimization: a review and some recommendations. Int J Numer Meth Eng 13:203–228. doi:10.1002/nme.1620130202
14. Serna1 A, Bucher C (2009) Advanced surrogate models for multidisciplinary design optimization. In: 8th Weimar Optimization and Stochastic Days
15. Boussouf L (2011) Surrogate based optimization for multidisciplinary design. In: SAE aerospace technology conference, Toulouse, France, 18–21 Oct 2011
16. Nocedal J, Wright SJ (2006) Numerical optimization, 2nd edn. Springer Science + Business Media, New York, ISBN-13: 978-0387-30303-1
17. Venter G (2010) Review of optimization techniques, encyclopedia of aerospace engineering. Wiley Ltd, Hoboken
18. Roy R, Hinduja SH, Teti R (2008) Recent advances in engineering design optimization: challenges and future trends. CIRP Ann Manufact Technol 57(2008):697–715
19. Waziruddin S, Brogan DC, Reynolds Jr PF (2004) Coercion through optimization: a classification of optimization techniques. In: 2004 fall simulation interoperability workshop, Orlando
20. Moldoveanu G, Abaluta O (2009) Multidisciplinary optimization in urban services management. Theor Empirical Res Urban Manage 1(10)
21. Lee HJ, Lee JW, Lee JO (2009) Development of web services-based multidisciplinary design optimization framework. Adv Eng Softw 40:176–183
22. Schittkowski K, Yuan Y (2007) Sequential quadratic programming methods. Wiley Encyclopedia of Operations Research and Management Science. doi:10.1002/9780470400531.eorms0984
23. Boggs PT, Tolle JW (1995) Sequential quadratic programming. Acta Numer 4:1–51
24. Lasdon LS, Fox RL, Ratner MW (1973) Nonlinear optimization using the generalized reduced gradient method. In: Office of naval research, technical memorandum no. 32
25. Michalewicz Z, Janikow CZ (1991) Genetic algorithms for numerical optimization. Stat Comput 1(2):75–91
26. Fernandes de Oliveira R et al (2008) Genetic optimization applied in conceptual and preliminary aircraft design. In: XVII Congresso e Exposição Internacionais da Tecnologia da Mobilidade, São Paulo, Brasil, 7–9 Oct 2008
27. Hart CG, Vlahopoulos N (2009) Integrating a particle swarm optimizer in a multi-discipline design optimization environment for conceptual ship design. In: SAE world congress and exhibition, Detroit, MI, 20–23 Apr 2009
28. Schutte JF et al (2004) Parallel global optimization with the particle swarm algorithm. Int J Numer Meth Eng 61(13):2183–2387
29. Peri D, Fasanoy G, Dessi D, Campana EF (2008) Global optimization algorithms in multidisciplinary design optimization. In: 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, Canada, 10–12 Sept 2008
30. Strobbe T, Pauwels P, Verstraeten R, De Meyer R (2011) Metaheuristics in architecture. Int J Sustain Constr Des 2(2):190, ISSN 2032-7471
31. Gonzalez LF, Periaux J, Srinivas K, Whitney EJ (2004) Evolutionary optimization tools for multi-objective design in aerospace engineering: from theory to MDO applications. In: Evolutionary algorithms and intelligent tools in engineering optimization, CIMNE, Barcelona

32. Depince P, Guedas B, Picard J (2007) Multidisciplinary and multiobjective optimization: comparison of several methods. In: 7th world congress on structural and multidisciplinary optimization, Seoul, 21–25 May 2007
33. Kesseler E, van Kan WJ (2006) Multidisciplinary design analysis and multi-objective optimization applied to aircraft wing. In: National aerospace laboratory NLR, NLR-TP-2006-748
34. Marler RT, Arora JS (2004) Survey of multi-objective optimization methods for engineering. J Struct Multi Optim 26:369–395
35. Li M (2007) Robust optimization and sensitivity analysis with multi-objective genetic algorithms: single- and multi-disciplinary applications. PhD thesis, University of Maryland
36. Zitzler E, Laumanns M, Bleuler S (2004) A tutorial on evolutionary multiobjective optimization. In: Metaheuristics for multiobjective optimization, vol 535. Springer, Berlin. doi:0.1007/978-3-642-17144-4_1
37. Yang F, Bouchlaghem D (2010) Genetic algorithm-based multiobjective optimization for building design. Archit Eng Des Manage 6:68–82. doi:10.3763/aedm.2008.0077
38. Fantini P (2007) Effective multiobjective MDO for conceptual aircraft design—an aircraft design perspective. PhD thesis, Cranfield University
39. Pierret S (2005) Multi-objective and multi-disciplinary optimization of three-dimensional turbomachinery blades. In: 6th world congresses of structural and multidisciplinary optimization, Rio de Janeiro, Brazil, 30 May–3 June 2005
40. Salas AO, Townsend JC (1998) Framework requirements for MDO application development. In: 7th AIAA/USAF/NASA/ISSMO symposium on multidisciplinary analysis and optimization, St. Louis, MO, 2–4 Sept 1998
41. Manolache F, Costiner S (2002) Parallel processing approaches for multi-disciplinary optimization algorithms. Carnegie Mellon University, Department of Mathematical Sciences, Center for Nonlinear Analysis
42. Gould N, Orban D, Toint P (2005) Numerical methods for large-scale nonlinear optimization. Acta Numer 14:299–361
43. Martins JRRA, Lambe AB (2013) Multidisciplinary design optimization: a survey of architectures. AIAA J 51(9):2049–2075
44. Cramer EJ et al (1993) Problem formulation for multidisciplinary optimization. In: AIAA symposium of multidisciplanary design optimization
45. Mainini L, Maggiore P (2012) Multidisciplinary integrated framework for the optimal design of a jet aircraft wing. Int J Aerosp Eng 2012(750642):9, doi:10.1155/2012/750642
46. Ajmera HC, Mujumdar PM, Sudhakar K (2004) MDO architectures for coupled aerodynamic and structural optimization of a flexible wing. In: 45th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference, Palm Springs, CA, 19–22 Apr 2004
47. Sobieszczanski-Sobieski J, Barthelemy JF (1985) Improving Engineering system design by formal decomposition, sensitivity analysis, and optimization. In: NASA technical memorandum 86377
48. Kodiyalam S, Sobieszczanski-Sobieski J (2001) multidisciplinary design optimization—some formal methods, framework requirements, and application to vehicle design. Int J Veh Des 25 (1/2):3–22
49. Sobieszczanski-Sobieski J (1990) Sensitivity analysis and multidisciplinary optimization for aircraft design: recent advances and results. AIAA J Aircr 27(12):993–1001
50. Kroo I, Manning V (2000) Collaborative optimization: status and directions. In: 8th AIAA/NASA/ISSMO symposium on multidisciplinary analysis and optimization, Long Beach, CA, 6–8 Sept 2000
51. Braun RD, Kroo IM (1995) Development and application of the collaborative optimization architecture in a multidisciplinary design environment. In: Proceedings of the ICASE/NASA langley workshop on multidisciplinary design optimization, Hampton, VA, 13–16 Mar 1995
52. Du X, Chen W (2001) A hierarchical approach to collaborative multidisciplinary robust design, Department of Mechanical Engineering, University of Illinois at Chicago, Chicago

53. Braun RD, Gage P, Kroo I, Sobieski I (1996) Implementation and performance issues of collaborative optimization. NASA/TM-2004-213192
54. Simpson TW, Martins JRRA (2011) Multidisciplinary design optimization for complex engineered systems: report from a national science foundation workshop. J Mech Des 133 (10):101002. doi:10.1115/1.4004465
55. Ryan A (2007) A Multidisciplinary approach to complex systems design. In: PhD thesis, University of Adelaide
56. Sobieski J (2010) A Perspective on the state of multidisciplinary design optimization (MDO). In: Workshop on MDO, Fort Worth, 16 Sept 2010
57. Sobieszczanski-Sobieski J, Haftka RT (1995) Multidisciplinary aerospace design optimization: survey of recent developments. In: 34th aerospace sciences meeting and exhibition, Reno, NV, 15–18 Jan 1995
58. Peri D, Fasanoy G, Dessi D, Campana EF (2008) Global optimization algorithms in multidisciplinary design optimization. In: 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, Canada, 10–12 Sept 2008
59. Schuhmacher G, Murra I, Wang L, Laxander A et al. (2002) Multidisciplinary design optimization of a regional aircraft wing box. In: 9th AIAA/ISSMO symposium on multidisciplinary analysis and optimization, Atlanta, GA, 4–6 Sept 2002
60. Simpson TW, Martins JRRA (2012) Advancing the design of complex engineered systems through multidisciplinary design optimization. In: NSF workshop, 53rd AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference, Honolulu, Hawaii, 23–26 Apr 2012
61. Kolonay RM (2014) A physics-based distributed collaborative design process for military aerospace vehicle development and technology assessment. Int J Agile Syst Manage 7(3/4):242–260
62. van Tooren M, La Rocca G (2008) Systems engineering and multi-disciplinary design optimization. In: Curran R et al (eds) Collaborative product and service life cycle management for a sustainable world. Proceedings of the 15th ISPE international conference on concurrent engineering. Springer, London, pp 401–415
63. Sobolewski M (2014) Unifying front-end and back-end federated services for integrated product development. In: Cha J et al (eds) Moving integrated product development to service clouds in global economy. Proceedings of the 21st ISPE Inc. international conference on concurrent engineering. IOS Press, Amsterdam, pp 3–16
64. Ottino A, Ghodous P, Ladjal H, Shariat B, Figay N (2014) Interoperability of simulation applications for dynamic network enterprises based on cloud computing—aeronautics application. In: Cha J et al (eds) Moving integrated product development to service clouds in global economy. Proceedings of the 21st ISPE Inc. international conference on concurrent engineering. IOS Press, Amsterdam, pp 597–606