1

INTRODUCTION TO STRUCTURAL DESIGN

1.1 Elements of Structural Design

The design of a structural system has two categories: designing a new structure and improving the existing structure to perform better. The design engineer's experience and creative ideas are required in the development of a new structure, since it is very difficult to quantify a new design using mathematical measures. Recently, limited in-roads have been made in the creative work of the structural design using mathematical tools [1]. However, the latter evolutionary process is encountered much more frequently in engineering designs. For example, how many times does an automotive company design a new car using a completely different concept? The majority of a design engineer's work concentrates on improving the existing vehicle so that the new car can be more comfortable, more durable, and safer. In this text, we will focus on a design's evolutionary process by using mathematical models and computational tools.

Structural design is a procedure to improve or enhance the performance of a structure by changing its parameters. A *performance measure* can be quite general in engineering fields, and can include: the weight, stiffness, and compliance of a structure; the fatigue life of a mechanical component; the noise in the passenger compartment; the vibration level; the safety of a vehicle in a crash, etc. However, this text does not address such aesthetic measures as whether a car or a structural design is attractive to customers. All performance measures are presumed to be measurable quantities. System parameters are variables that a design engineer can change during the design process. For example, the thickness of a vehicle body panel can be changed to improve vehicle performance. The cross-section of a beam can be changed in designing bridge structures. System parameters that can be changed during the design process are called *design variables*, even including the geometry of the structure.

Recently, the simulation-based design process has emerged as the future tool of the product development and manufacturing process, since it allows one to achieve a higher quality product, through a reduction in development time in introducing new products to the market, a reduction in testing cycles, and a reduction in total development costs. As noted in the scholarly treatment of product performance by Clark and Fujimoto [2], essentially all development activities prior to the operation of a vehicle are simulations. In this sense, simulation can involve either mathematical models, or physical experiments that are created to emulate environments and conditions experienced by the product in its actual use.

Great strides have been made during the past decade in computer-aided design (CAD) and computer-aided engineering (CAE) tools for mechanical system development. Discipline-oriented simulation capabilities in structures, mechanical system dynamics, aerodynamics, control systems, and numerous related fields are now being used to support a broad range of mechanical system design applications. Integration of these tools to create a robust simulation-based design capability, however, remains a challenge. Based on their extensive survey of the automotive industry in the mid 1980s, Clark and Fujimoto [2] concluded that simulation tools in support of vehicle development were on the horizon, but not yet ready for pervasive application. The explosion in computer, software, and modeling and simulation technology that has occurred since the mid 1980s suggests that high fidelity tools for simulation-based design are now at hand. Properly integrated, they can resolve uncertainties and significantly impact mechanical system design.

An example of an integrated concurrent engineering environment for development of large-scale wheeled and tracked vehicle systems is illustrated in Fig. 1.1.1 [3]. It comprises simulation and modeling tools and an integration infrastructure to: (1) support design analysis, supportability analysis, operation analysis, and development process control; (2) establish connectivity between all application tools, with tool interactions transparent to the user; (3) refine product requirements; and (4) conduct trade-off analyses and make informed decisions to yield a robust optimal design. The integrated test-bed shown supports concurrent design, operator-in-the-loop driving simulation, dynamic performance analysis, durability prediction, structural design sensitivity analysis and optimization, maintainability analysis, and design process management. The test-bed permits all members of the development team to simulate the performance and effectiveness of product and process designs, at a level of fidelity comparable to that, which would be achieved in physical prototyping.

Using the test-bed, an integrated simulation-based design process for the fatigue life of vehicle components can be developed [3], as shown in Fig. 1.1.2. The process includes CAD-modeling, dynamic analysis, fatigue analysis, design sensitivity analysis, and design optimization. The CAD-based design model is critically important for multidisciplinary analysis and design optimization. The process allows engineers to create a CAD model of a vehicle system and automatically translate the CAD model into a dynamics model. Dynamic simulations of the vehicle model are then carried out over typical road profiles to obtain load histories at selected components. In the meantime, CAD models of the selected vehicle components are created for design parameterization and translated into finite element analysis (FEA) models. The computation of the fatigue life of a component consists of two parts: dynamic stress computation and fatigue life prediction. The dynamic stress can be obtained either from experiments (mounting sensors or transducers on a physical component) or from simulation. Fatigue analysis is performed using the low cycle fatigue approach. Once the fatigue life of the vehicle components is obtained, design sensitivity analysis with respect to shape design variables defined in the CAD model is performed. With the design sensitivity information, design optimization can be carried out to obtain an optimum design.

As shown in Figs. 1.1.1 and 1.1.2, modern developments of structural design are closely related to concurrent engineering environments by which multidisciplinary simulation, design, and manufacturing are possible. Even though concurrent engineering is not the focus of this text, we want to emphasize structural design as a component of concurrent engineering. Figure 1.1.1 shows an example of concurrent engineering environments used in structural design. An important feature of Fig. 1.1.1 is database management using the CAD tool. Structural modeling and most interfaces are achieved using the CAD tool. Thus, design parameterization and structural model updates have to be carried out in the CAD model. Through the design parameterization, CAD, CAE, and CAM procedures are interrelated to form a concurrent engineering environment.

The engineering design of the structural system in the simulation-based design process consists of: structural modeling, design parameterization, structural analysis, design problem definition, design sensitivity analysis, and design optimization. Figure 1.1.3 is a flow chart of the structural design process in which computational analysis and mathematical programming play essential roles in the design. The success of the system-level, simulation-based design process shown in Fig. 1.1.2 strongly depends on a consistent design parameterization, an accurate structural and design sensitivity analysis, and an efficient mathematical programming algorithm.

A design engineer simplifies the physical engineering problem into a mathematical model that can represent the physical problem up to the desired level of accuracy. A mathematical model has parameters that are related to the system parameters of the physical problem. A design engineer identifies those design variables to be used during the design process. *Design parameterization*, which allows the design engineer to define the geometric properties for each design component of the structural system being designed, is one of the most important steps in the structural design process. The principal role of design parameterization is to define the geometric parameters that characterize the structural model

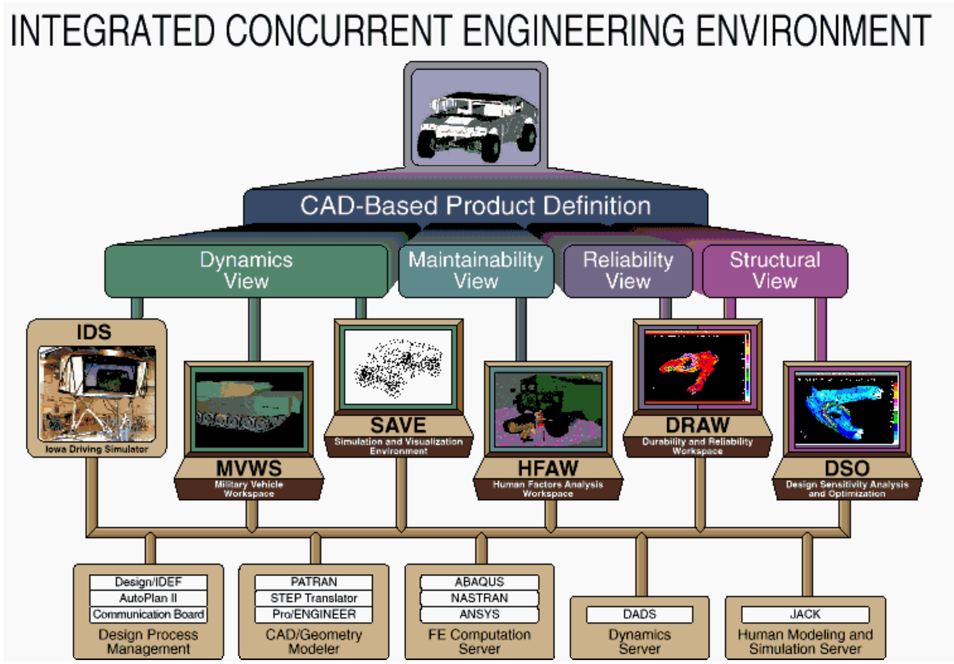

Fig. 1.1.1. Integrated Concurrent Engineering Environment

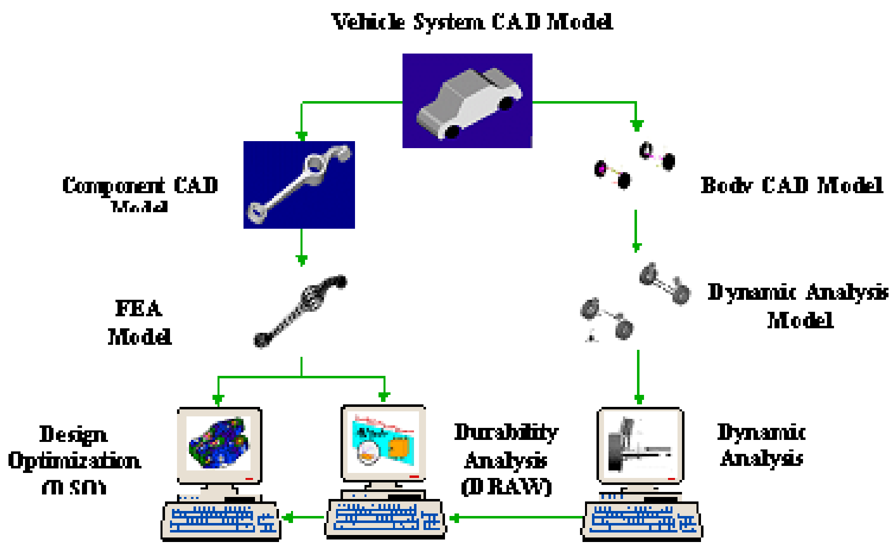

Fig. 1.1.2. Simulation-Based Design Optimization Process for Fatigue Life

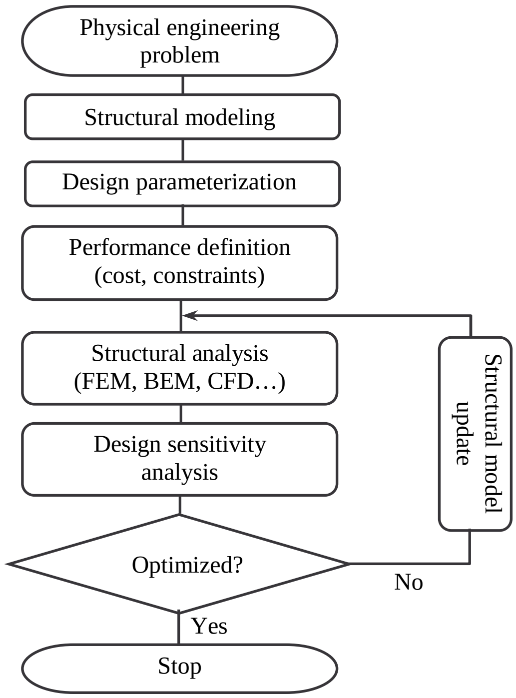

Fig. 1.1.3. Structural Design Process

and to collect a subset of the geometric parameters as design variables. Design parameterization forces engineering teams in design, analysis, and manufacturing to interact at an early design stage, and supports a unified design variable set to be used as the common ground to carry out all analysis, design, and manufacturing processes. Only proper design parameterization will yield a good optimum design, since the optimization algorithm will search within a design space that is defined for the design problem. The design space is defined by the type, number, and range of design variables. Depending on whether it is a concept or detailed design, selected design variables could be non-CAD based parameters. An example of such a design variable is a tire stiffness characteristic in vehicle dynamics during an early vehicle design stage.

Structural analysis can be carried out using experiments in actual or reduced scale, which is a straightforward and still prevalent method for industrial applications. However, the expense and the inefficiency involved in fabricating prototypes make this approach difficult to apply. The analytical method may resolve these difficulties, since it approximates the structural problem as a mathematical model and solves it in a simplified form. In this text, a mathematical model is used to evaluate the performance measures of a structural problem. However, the analytical method has limitations even for very simple structural problems.

With the emergence of various computational capabilities, most analytical approaches to mathematical problems have been converted to numerical approaches, which are able to solve very complicated, real engineering applications. Finite element analysis (FEA), boundary element analysis (BEA), and meshfree analysis are a short list of mathematical tools used in structural analysis. The development of FEA is one of the most remarkable successes in structural analysis. The governing differential equation of the structural problem is converted to its integral form and then solved using FEA. Vast amounts of literature are published regarding FEA; for example, refer to [4] and references therein. The complex structural domain is discretized by a set of non-overlapping, simple-shaped finite elements, and an equilibrium condition is imposed on each element. By solving a linear system of matrix equations, the performance measures of a structure are computed in the approximated domain. The accuracy of the approximated solution can be improved by reducing the size of finite elements and/or increasing the order of approximation within an element.

Selection of a design space and an analysis method must be carefully determined since the analysis, both in terms of accuracy and efficiency, must be able to handle all possible designs in the chosen design space. That is, the larger the design space, the more sophisticated the analysis capability must be. For example, if larger shape design changes are expected during design optimization, mesh distortion in FEA could be a serious problem and a finite element model that can handle large shape design changes must be used.

A performance measure in a simulation-based design is the result of structural analysis. Based on the evaluation of analysis results, such engineering concerns as high stress, clearance, natural frequency, or mass can be identified as performance measures for design improvement. Typical examples of performance measures are mass, volume, displacement, stress, compliance, buckling, natural frequency, noise, fatigue life, and crashworthiness. A definition of performance measures permits the design engineer to specify the structural performance from which the sensitivity information can be computed.

Cost and constraints can be defined by combining certain performance measures with appropriate constraint bounds for interactive design optimization. *Cost function*, sometimes called the *objective function*, is minimized (or maximized) during optimization. Selection of a proper cost function is an important decision in the design process. A valid cost function has to be influenced by the design variables of the problem; otherwise, it is not possible to reduce the cost by changing the design. In many situations, an obvious cost function can be identified. In other situations, the cost function is a combination of different structural performance measures. This is called a multi-objective cost function.

Constraint functions are the criteria that the system has to satisfy for each feasible design. Among all design ranges, those that satisfy the constraint functions are candidates for the optimum design. For example, a design engineer may want to design a bridge whose weight is minimized and whose maximum stress is less than the yield stress. In this case, the cost function, or weight, is the most important criterion to be minimized. However, as long as stress, or constraint, is less than the yield stress, the stress level is not important.

Design sensitivity analysis, which is a main focus of this text, is used to compute the sensitivity of performance measures with respect to design variables. This is one of the most expensive and complicated procedures in the structural optimization process. Structural design sensitivity analysis is concerned with the relationship between design variables available to the engineer and the structural response determined by the laws of mechanics. Design sensitivity information provides a quantitative estimate of desirable design change, even if a systematic design optimization method is not used. Based on the design sensitivity results, a design engineer can decide on the direction and amount of design change needed to improve the performance measures. In addition, design sensitivity information can provide answers to 'what if' questions by predicting performance measure perturbations when the perturbations of design variables are provided.

Substantial literature has emerged in the field of structural design sensitivity analysis [5]. Design sensitivity analysis of structural systems and machine components has emerged as a much needed design tool, not only because of its role in optimization algorithms, but also because design sensitivity information can be used in a computer–aided engineering environment for early product trade–off in a concurrent design process.

Recently, the advent of powerful graphics–based engineering workstations with increasing computational power has created an ideal environment for making interactive design optimization a viable alternative to more monolithic batch–based design optimization. This environment integrates design processes by letting the design engineer create a geometrical model, build a finite element model, parameterize the geometric model, perform FEA, visualize FEA results, characterize performance measures, and carry out design sensitivity analysis and optimization.

Design sensitivity information can be used during a post–processing of the interactive design process. The principal objective of the post–processing design stage is to utilize the design sensitivity information to improve the design. Figure 1.1.4 shows the four–step interactive design process: (1) to visually display design sensitivity information, (2) to carry out what–if studies, (3) to make trade–off determinations, and (4) to execute interactive design optimization. The first three design steps, which are interactive modes of the design process, help the design engineer improve the design by providing structural behavior information at the current design stage. The last design step, which could be either interactive or a batch mode of the design process, launches a mathematical programming algorithm to perform design optimization. Depending on the design problem, the design engineer could use some or all of the four design steps to improve the design at each iterative step. As a result, new designs could be obtained from what–if, trade–off, or interactive optimization design steps.

For the purposes of design optimization, a mathematical programming technique is often used to find an optimum design that can best improve the cost function within a feasible region. Mathematical programming generates a set of design variables that require performance values from structural analysis and sensitivity information from design sensitivity analysis to find an optimum design. Thus, the structural model has to be updated for a different set of design variables supplied by mathematical programming. If the cost function reaches a minimum with all constraint requirements satisfied, then an optimum design is obtained.

In the following sections, each element of the design process is discussed in detail.

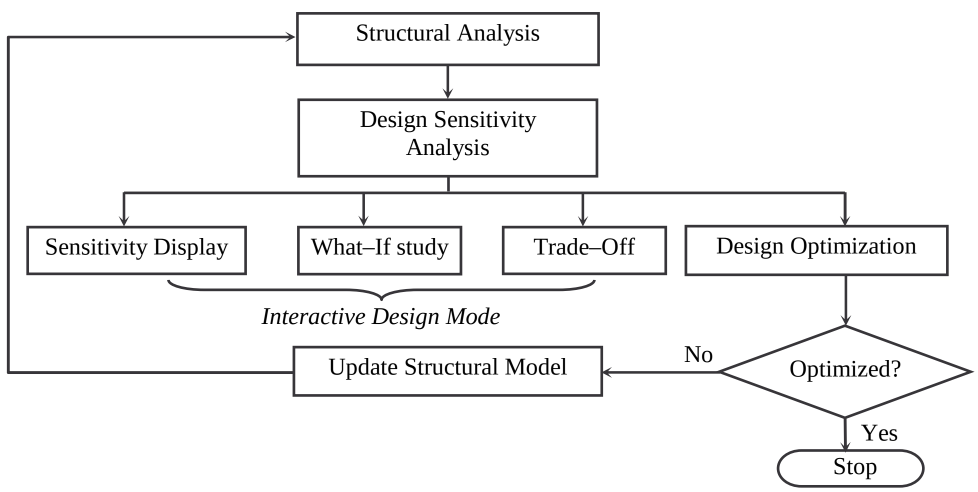

Fig. 1.1.4. Post–processing Design Stage

1.2 Structural Modeling and Design Parameterization

The first step in the design process is structural modeling and design parameterization. The physical engineering problem is converted to a mathematical model and the parameters that define the mathematical model have to be identified. Then, the goal of the design process is to find the proper set of design variables to produce the desired performance.

1.2.1 Structural Modeling

When engineers analyze a structural problem, they need to convert the physical problem into a mathematical representation. Many analysis tools can be used to solve this ideal mathematical problem. After arriving at a solution of the mathematical problem, the meaning of the solution has to be correctly interpreted in its physical sense. Thus, if there is an error in the mathematical representation of the physical problem, then it is impossible to properly analyze the physical problem no matter what analysis tools are used. This mathematical representation of the structural problem is called *structural modeling*.

The reliability of the analysis results strongly depends on the assumptions and idealization used in structural modeling. However, a too–complex representation of the physical problem may make it difficult to solve the mathematical problem. For example, when an engineer wants to determine the height and width of a bridge, it may not be important to model every bolt, because the desired results will consist of global flexibility and the bridge's maximum degree of deflection. If each bolt is modeled for the entire bridge structure, then the analysis cost dramatically increases. It may be presumed that sections are constructed continuously without any breaks. However, after determining the size of the bridge, the engineer may want to design the number and size of each section of the bridge. The maximum magnitude of load carried by each bolt would then be of major concern. In this case, the size and distance between bolts would be important and structural modeling would need to include bolt strength. Consequently, different concerns require different structural models. It is the engineer's responsibility to find an appropriate trade–off between accuracy and computational costs of analysis.

1.2.2 Design Parameterization

In structural modeling, the physical problem is represented by mathematical expressions, which contain parameters for defining that problem. For example, the cantilever beam in Fig. 1.2.1 has parameters including the length l, the radius of cross-section r, and Young's modulus E. These parameters define the system and are called *design variables*. If design variables are determined, then the structural problem can be analyzed. Obviously, different design variable values usually yield different analysis results. The aim of the structural design process is to find the values of design variables that satisfy all requirements.

All design variables must satisfy the physical requirements of the problem. For example, length l of the cantilever beam in Fig. 1.2.1 cannot have a negative value. Physical requirements define the design variable bounds. Valid design variables may have to take into account various manufacturing requirements. For example, the radius of a cantilever beam satisfies its physical requirement if r is a positive number. However, in real applications, the circular cross-sectional beam may not be manufactured if its radius is bigger than r^0. Thus, the range of feasible design can be stated as $0 < r \leq r^0$. [1] In addition, the design engineer may want to impose certain design constraints on the problem. For example, the maximum stress of the beam may not exceed σ^0 and the maximum tip displacement of the beam must not be greater than z^0. A set of design variables that satisfy the constraints is called a *feasible design*, while a set that does not satisfy constraints is called an *infeasible design*. It is difficult to determine whether a current design is feasible, unless the structural problem is analyzed. For complicated structural problems, it may not be simple to choose the appropriate design constraints so that the feasible region is not empty.

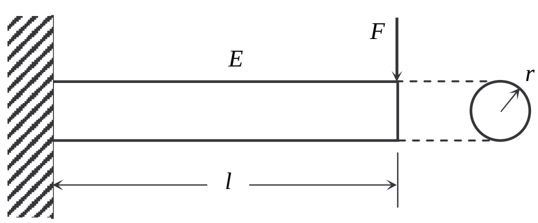

Fig. 1.2.1. Parameters Defining Circular Cross-sectional Cantilever Beam

There are two types of design variables: continuous and discrete. Many design optimization algorithms consider design variables to be continuous. In this text, we presume that all design variables are

[1] In general the bounds of design variables are denoted as $r^L \leq r \leq r^U$ where r^L is called the lower bound and r^U is called the upper bound, respectively.

continuous within their lower and upper bound limits. However, discrete design problems often appear in real engineering problems. For example, due to manufacturing limitations, the structural components of many engineering systems are only available in fixed shapes and sizes. Discrete design variables can be thought of as continuous design variables with constraints. As a result, it is more expensive to obtain an optimum design for a problem with discrete design variables. It is possible, however, to solve the problem assuming continuous design variables. After obtaining an optimum solution for the design problem, the nearest discrete values of the optimum design variables can be tested for feasibility. If the nearest discrete design variables are not feasible, then several iterations can be carried out to find the nearest feasible design.

It is convenient to classify design variables according to their characteristics. In the design of structural systems made of truss, beam, membrane, shell, and elastic solid members, there are five kinds of design variables: material property design variables such as Young's modulus; sizing design variables such as thickness and cross-sectional area; shape design variables such as length and geometric shape; configuration design variables such as orientation and location of structural components; and topological design variables.

Material Property Design Variable

In structural modeling, the material property is used as a parameter of the structural problem. Young's modulus and Poisson's ratio, for example, are required in the linear elastic problem. If these material properties are subject to change, then they are called *material propertydesign variables*. These kinds of design variables do not appear in regular design problems, since in most cases material properties are presumed to be constant. Analysis using such a constant material property is called the deterministic approach. Another approach uses probability and assumes that material properties are not constant but distributed within certain ranges. This is called the probabilistic approach and is more practical, since a number of experiments will usually yield a number of different test results. In this case, material properties are no longer considered to be constant and can therefore be used as design variables.

Sizing Design Variable

Sizing design variables are related to the geometric parameter of the structure. For example, most automotive and airplane parts are made from plate/shell components. It is natural that a design engineer wants to change the thickness (or gauge) of the plate/shell structure in order to reduce the weight of the vehicle. For a structural model, plate thickness is considered a parameter. However, the global geometry of the structure does not change. Plate thickness can be considered a *sizingdesign variable*. The sizing design variable is similar to the material property design variable in the sense that both variables change the parameters of the structural problem.

Another important type of sizing design variable is the cross-sectional geometry of the beam and truss. Figure 1.2.2 provides some examples of the shapes and parameters that define these cross-sections. In the structural analysis of truss, for example, the cross-sectional area is required as a parameter of the problem. If a rectangular cross-section is used, then the area would be defined as $A = b \times h$. Thus, without any loss of generality, b and h can be considered design variables of the design problem. Detailed discussions of sizing design problems are discussed in Chapter 5 using the distributed parameter approach.

Shape Design Variable

While material property and the sizing design variables are related to the parameters of the structural problem, the shape design variable is related to the structure's geometry. The shape of the structure does not explicitly appear as a parameter in the structural formulation. Although the design variables in Fig. 1.2.2 determine the cross-sectional shape, they are not shape design variables, since these cross-sectional shapes are considered parameters in the structural problem. However, the length of the truss or

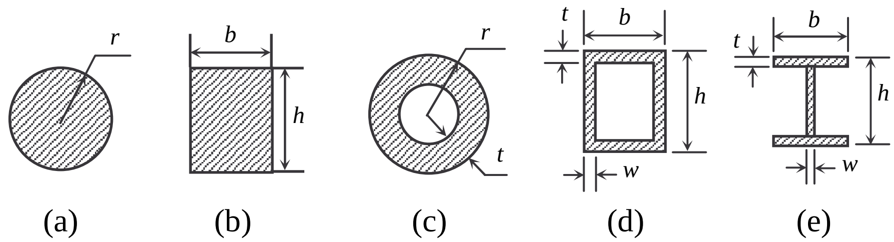

Fig. 1.2.2. Sizing Design Variables for Cross-sectional Areas of Truss and Beam. (a) Solid Circular; (b) Rectangular; (c) Circular Tube; (d) Rectangular tube; (e) I–section

beam should be treated as a shape design variable. Usually, the shape design variable defines the domain of integration in structural analysis. Thus, it is not possible to extract shape design variables from a structural model and to use them as sizing design variables.

Consider a rectangular block with a slot, as presented in Fig. 1.2.3. The location and size of the slot is determined by the geometric values of C_x, C_y, D_y, r_1, and r_2, which are shape design variables. Different values of shape design variables yield different structural shapes. However, these shape design variables do not explicitly appear in the structural problem. If the finite element method is used to perform structural analysis, then integration is carried out over the structural domain (the gray area), which is the shape design variable. Since shape design variables do not explicitly appear in the structural problem, the shape design problem is more difficult to solve than the sizing design problem. Detailed discussions of the shape design problem are presented in Chapter 6 using the material derivative concept of continuum mechanics.

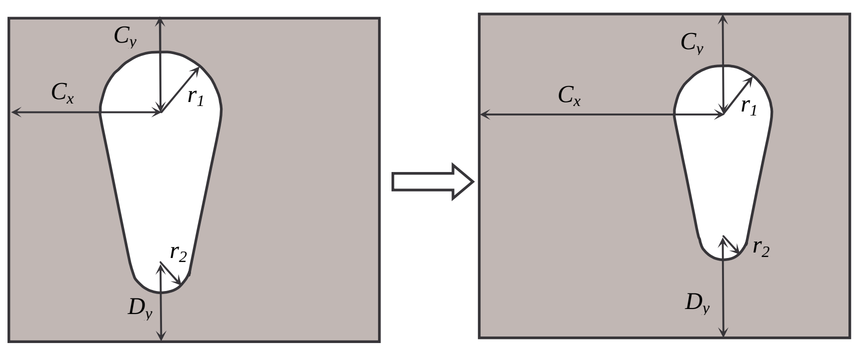

Fig. 1.2.3. Shape Design Variables

Configuration Design Variable

For those built-up structures made of truss, beam, and shell components, there is another type of design variable in addition to shape design called the configuration design variable, which is related to the structural component's orientation. These components have local coordinate systems fixed on the body of the structure, and state variables of the problem are described in local coordinate systems. If several different components are connected together for the built-up structure, the state variables described in the local coordinate system are transformed to the global coordinate system. If the structural components change their orientation in space, the transformation between the local and global coordinates also changes. Thus, this transformation can be considered the configuration design variable. Since configuration design variables are defined for built-up structures, they are inherently coupled with shape design variables. That is, in order to allow one member of the built-up structure to rotate, another member's shape needs to be changed. The configuration design variable is not applicable to solid components in which all rotations can be expressed in terms of shape changes. A simple configuration design variable will be explained using the example of a three-bar truss in Section 1.2.3. More detailed discussions of the configuration design problem are presented in Chapter 7 using the material derivative concept in continuum mechanics.

Topology Design Variable

If shape and configuration design variables represent changes in structural geometry and orientation, then topology design determines the structure's layout. For example, in Fig. 1.2.3, shape design can change the size and location of the slot within the block. However, shape design cannot completely remove the slot from the block, or introduce a new slot. Topology design determines whether the slot can be removed or an additional slot is required.

The choice of the topology design variable is nontrivial compared to other design variables. Which parameter is capable of representing the birth or death of the structural layout? Early developments in topology design focused on truss structures. For a given set of points in space, design engineers tried to connect these points using truss structures, in order to find the best layout to support the largest load. Thus, the on–off types of topology design variables are used. These kinds of designs, however, could turn out to be discontinuous and unstable.

Recent developments in topology design are strongly related to finite element analysis. The candidate design domain is modeled using finite elements, and then the material property of each element is controlled. If it is necessary to remove a certain region, then the material property value (e.g., Young's modulus) will approach zero, such that there will be no structural contribution from the removed region. Thus, material property design variables could be used for the purpose of topology design variables. The on–off type of design variable can be approximated by using continuous polynomials in order to remove the difficulties associated with discrete design variables.

In many applications, topology design is used at the concept design stage such that the layout of the structure is determined. After the layout is determined, sizing and shape designs are used to determine the detailed geometry of the structure.

A final comment on design parameterization: it is desirable to have a linearly independent set of design variables. If one does not, then relations between design variables must be imposed as constraints, which may make the design optimization process expensive, as the number of design variables and constraints increase. Furthermore, if design variable constraints are not properly established, meaningless design results will be obtained after an extensive amount of computational effort. As mentioned before, this problem is strongly related to structural modeling, since a well–defined structural model should have an independent set of parameters to define the entire system. Even if defining a good model is not an easy task for a complicated design problem, the design engineer nevertheless has to define a proper and independent set of parameters as much as possible in the structural modeling stage.

1.2.3 Three–Bar Truss Example

In this section, a simple example is introduced to discuss design parameterization, which includes material property, sizing, shape, and configuration design variables. This example will be used repeatedly in subsequent sections to explain the structural analysis and design process. The three–bar truss consists of three truss components, as shown in Fig. 1.2.3.

For truss components, only one material parameter is involved, which is Young's modulus E. Thus, the material design parameter is $\boldsymbol{u} = [E]$. On the other hand, the cross-sectional area of each component can be chosen to represent the sizing design variables, stated as $\boldsymbol{u} = [b_1,\ b_2,\ b_3]^T$. As explained in Fig. 1.2.2, the dimensions that determine the cross-sectional shape can be represented as a sizing design. However, for general truss structures, it is purely a matter of convenience whether the cross-sectional dimensions or the cross-sectional area is chosen as the sizing design. As far as analysis is concerned, only cross-sectional area information is required. For example, if the cross-section of component 1 is a solid circular shape with radius r, then the relation between the cross-sectional area and the radius would be $b_1 = \pi r^2$

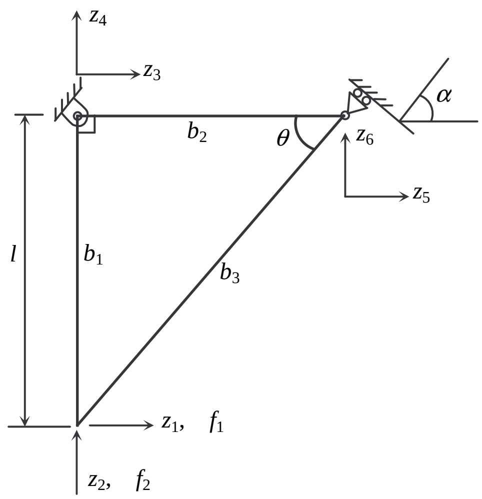

Fig. 1.2.4. Three–bar Truss Structure

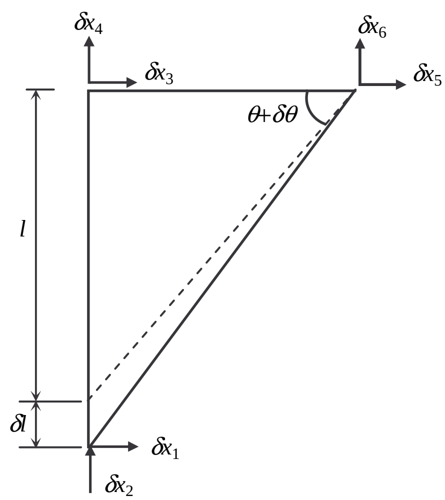

Fig. 1.2.5. Shape and Configuration Design Variable of Truss

As previously pointed out, shape and configuration design variables are closely coupled in the three–bar truss structure. The shape and configuration design variable is

$$\delta \mathbf{x} = [\delta x_1,\ \delta x_2,\ \delta x_3,\ \delta x_4,\ \delta x_5,\ \delta x_6]^T \tag{1.2.1}$$

In Fig. 1.2.3, for example, if the length of member 1 is changed from l to $l + \delta l$, then the integration domain, which is the shape design, would change. However, since all the members are interconnected, the third member's length and orientation must be modified to satisfy geometric requirements

(Fig. 1.2.5). The change of length involves the shape design, while the rotation of a member affects the configuration design.

1.3 Structural Analysis

Structural analysis is solving the mathematical model of the physical problem. In this section, a *variational method* or *energy method* is introduced by using a simple truss structure. The structural equilibrium is viewed as a stationary condition of the total potential energy. For the positive definite quadratic energy, this condition becomes the global minimum condition. Combined with finite element discretization, this method is one of the most popular approaches in structural analysis.

Consider a one–dimensional truss structure under a distributed load $f(x)$ and with point loads F_1 at x = 0 and F_2 at $x = l$, as shown in Fig. 1.3.1. In this text, $z(x)$ denotes a displacement function; the more common notation u used for displacement is reserved as the design variable. For the moment, let the cross-sectional area $u = A(x)$ vary along the truss structural component.

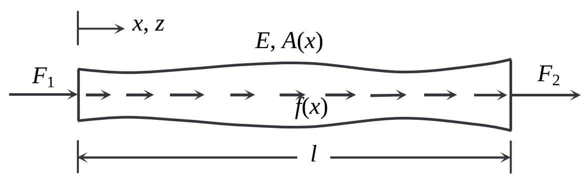

Fig. 1.3.1. Truss Structural Component

For given x, we assume that stress σ is constant over the cross-section $A(x)$. Thus, stress is a function of x alone. The same assumption is given to strain ε such that $\varepsilon(x) = dz/dx$. The stress–strain relation is linear elastic, stated as

$$\sigma(x) = E\varepsilon(x) \tag{1.3.1}$$

where E is Young's modulus.

As loads are applied to the structure, the structure deforms to resist them. If all loads are removed, then the structure recovers its original shape. Thus, energy is stored in the deformed structure and is called the *strain energy*, defined as

$$U \equiv \frac{1}{2}\int_0^l A\sigma\varepsilon\, dx = \frac{1}{2}\int_0^l EA\left(\frac{dz}{dx}\right)^2 dx \tag{1.3.2}$$

If the applied load accompanies the deformation, then work is done to the structure, and we can define it as

$$W \equiv \int_0^l fz\, dx + F_1 z(0) + F_2 z(l) \tag{1.3.3}$$

We can define the total potential energy of the structure as the difference between U and W, given as

$$\Pi = U - W = \frac{1}{2}\int_0^l EA\left(\frac{dz}{dx}\right)^2 dx - \int_0^l fz\, dx - F_1 z(0) - F_2 z(l) \tag{1.3.4}$$

When the structure is in equilibrium, the forces generated by the structural deformation are the same as the externally applied loads. It is equally true that the total potential energy in Eq. (1.3.4) becomes stationary, which means that the first variation vanishes, so that

$$\delta\Pi = \int_0^l EA\left(\frac{dz}{dx}\right)\left(\frac{d\bar{z}}{dx}\right)dx - \int_0^l f\,\bar{z}\,dx - F_1\bar{z}(0) - F_2\bar{z}(l) = 0 \tag{1.3.5}$$

where $\bar{z}$ is the first–order variation of displacement z. A detailed explanation of the variation is provided in Chapter 2. For the moment, $\bar{z}$ can be thought of as a small, arbitrary perturbation of z. If z is fixed at a point x, then $\bar{z}$ vanishes at the same point. The structural problem is to solve for z in a way that satisfies Eq. (1.3.5) for all arbitrary $\bar{z}$.

If the kinematic boundary conditions are given at some point for displacement z, then the possible candidates for the solution are limited to those that satisfy the displacement boundary conditions. For example, if the truss structure in Fig. 1.3.1 is fixed at x = 0, then the candidates for the solution belong to the following solution space:

$$Z = \left\{z \in H^1(0,l) \mid \ z(0) = 0\right\} \tag{1.3.6}$$

where H^1 is a Sobolev space [6] of order one whose elements are continuous functions, and the first derivative of the function is square integrable in the domain. For a more detailed discussion of basic function spaces, refer to Appendix A.2. For the moment, readers can think of H^1 as a space of smooth functions. In addition, the displacement variation $\bar{z}$ should satisfy Eq. (1.3.6). Thus, the term $F_1\bar{z}(0)$ in Eq. (1.3.5) would vanish. Physically, if the displacement is fixed, then there would be no work done to the structure by the applied load.

1.4 Finite Element Analysis

The analytical solution to Eq. (1.3.5) is non-trivial, even for a simple built-up structure. For a general shaped structure, an approximation of Eq. (1.3.5) is required by using finite elements. The finite element method approximates the domain of the structure as a simple geometry set, and then establishes the equilibrium conditions for each finite element. By combining all finite elements, a global system of matrix equations is obtained.

Consider a truss finite element with a constant cross-sectional area as shown in Fig. 1.4.1. For simplicity, the distributed load is removed. Displacement of the truss element is represented by two end–displacements, namely, z_1 and z_2.

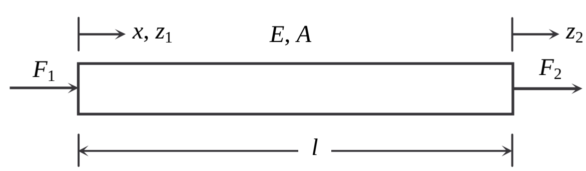

Fig. 1.4.1. Truss Finite Element

The displacement at a point x in the domain (0, l) is interpolated by using z_1 and z_2 as

$$\begin{aligned} z(x) &= N_1(x)z_1 + N_2(x)z_2 \\ N_1(x) &= \frac{l-x}{l}, \quad N_2(x) = \frac{x}{l} \end{aligned} \tag{1.4.1}$$

where $N_1(x)$ and $N_2(x)$ are called the *shape functions* corresponding to nodes 1 and 2, respectively. Note that $z(0) = z_1$ and $z(l) = z_2$. The derivative of displacement, or strain, can be directly obtained from Eq. (1.4.1) as

$$\begin{aligned} \frac{dz}{dx} &= \frac{1}{l}(z_2 - z_1) \\ \frac{d\bar{z}}{dx} &= \frac{1}{l}(\bar{z}_2 - \bar{z}_1) \end{aligned} \tag{1.4.2}$$

in which the second equation is the derivative of the displacement variation. By using Eqs. (1.4.1) and (1.4.2), the variation of the total potential energy in Eq. (1.3.5) can be written as

$$\delta\Pi = \frac{EA}{l}(z_2 - z_1)(\bar{z}_2 - \bar{z}_1) - F_1\bar{z}_1 - F_2\bar{z}_2 = 0 \tag{1.4.3}$$

for all $\bar{z}_1$ and $\bar{z}_2$ in Z. To express Eq. (1.4.3) systematically, it is necessary to define the following vectors

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \quad \bar{\mathbf{z}} = \begin{bmatrix} \bar{z}_1 \\ \bar{z}_2 \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}, \quad \mathbf{k} = \frac{EA}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{1.4.4}$$

where $\mathbf{z}$ is the nodal displacement vector and $\bar{\mathbf{z}}$ its variation; $\mathbf{f}$ is the nodal force vector; and $\mathbf{k}$ is the element stiffness matrix. The variational equation (1.4.3) for the truss element can be written as

$$\bar{\mathbf{z}}^T\mathbf{k}\mathbf{z} = \bar{\mathbf{z}}^T\mathbf{f} \tag{1.4.5}$$

for all $\bar{\mathbf{z}}$ in Z. Since Eq. (1.4.5) is satisfied for all $\bar{\mathbf{z}}$ in Z, it is equivalent to solving the following matrix equation:

$$\mathbf{k}\mathbf{z} = \mathbf{f} \tag{1.4.6}$$

which is called the *local finite element equation*. Equation (1.4.6) is applicable to one finite element. However, for a built-up structure, many truss elements are connected together to make a complete structure. In this case, the local finite element equation in Eq. (1.4.6) has to be combined to construct the global finite element equation, and this process is called *assembly*.

To see the assembly and solution processes of the truss element, consider the three–bar truss example with a multipoint boundary condition, as shown in Fig. 1.2.3. The displacement and load vectors can be written in the global coordinate system as

$$\mathbf{z}_g = [\, z_1 \;\, z_2 \;\, z_3 \;\, z_4 \;\, z_5 \;\, z_6 \,]^T \tag{1.4.7}$$

$$\mathbf{F}_g = [\, f_1 \;\, f_2 \;\, 0 \; 0 \; 0 \; 0 \,]^T \tag{1.4.8}$$

The element stiffness matrix in the body-fixed local coordinate system must transformed to the global coordinate system. For this purpose, let $\mathbf{d}^i$ denote the element local coordinate, and $\mathbf{q}^i$ represent the globally oriented element coordinate, as illustrated in Fig. 1.4.2. In each element, the transformation between the body-fixed and the globally oriented coordinate is

$$\begin{aligned} \mathbf{d}^1 &= \mathbf{q}^1, \\ \mathbf{d}^2 &= \mathbf{q}^2, \\ \mathbf{d}^3 &= \begin{bmatrix} c & s & 0 & 0 \\ -s & c & 0 & 0 \\ 0 & 0 & c & s \\ 0 & 0 & -s & c \end{bmatrix}\mathbf{q}^3 \end{aligned}$$

where c= cosθ and s= sinθ. In addition, the globally oriented coordinates are transformed into the global coordinate $\mathbf{z}_g$ by using Boolean matrices as

$$\mathbf{q}^1 = \begin{bmatrix} 0&1&0&0&0&0 \\ 1&0&0&0&0&0 \\ 0&0&0&1&0&0 \\ 0&0&1&0&0&0 \end{bmatrix} \mathbf{z}_g$$

$$\mathbf{q}^2 = \begin{bmatrix} 0&0&1&0&0&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix} \mathbf{z}_g$$

$$\mathbf{q}^3 = \begin{bmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix} \mathbf{z}_g$$

After the assembly process, the generalized global stiffness matrix is expressed in terms of the global displacement coordinate $\mathbf{z}_g$ as

$$\mathbf{K}_g(\mathbf{u}) = \frac{E}{l} \begin{bmatrix} b_3c^2s & b_3cs^2 & 0 & 0 & -b_3c^2s & -b_3cs^2 \\ b_3cs^2 & b_1 + b_3s^3 & 0 & -b_1 & -b_3cs^2 & -b_3s^3 \\ 0 & 0 & b_2s/c & 0 & -b_2s/c & 0 \\ 0 & -b_1 & 0 & b_1 & 0 & 0 \\ -b_3c^2s & -b_3cs^2 & -b_2s/c & 0 & b_2s/c + b_3c^2s & b_3cs^2 \\ -b_3cs^2 & -b_3s^3 & 0 & 0 & b_3cs^2 & b_3s^3 \end{bmatrix} \tag{1.4.9}$$

In Eq. (1.4.9), $\mathbf{u} = [b_1, b_2, b_3]^T$ denotes the design variable vector, which is the cross-sectional area of each truss element. Note that the global stiffness matrix $\mathbf{K}_g$ is singular, since it has a rigid body motion that can be removed by applying boundary conditions. As shown in Fig. 1.2.3, the displacement variables z_3 and z_4 are fixed. In addition, z_5 and z_6 are dependent on each other. Solution candidates must satisfy these conditions. In this problem, space Z of kinematically admissible displacements is

$$Z = \left\{ \mathbf{z}_g \in R^6 : \quad z_3 = z_4 = 0, \quad z_5 \cos\alpha + z_6 \sin\alpha = 0 \right\} \tag{1.4.10}$$

and $\mathbf{K}_g(\mathbf{u})$ is the positive definite in Z, although it is not positive definite in all of R^6. Thus, the global variational equation of three–bar truss example is obtained as

$$\bar{\mathbf{z}}_g^T \mathbf{K}_g \mathbf{z}_g = \bar{\mathbf{z}}_g^T \mathbf{F}_g, \qquad \forall \bar{\mathbf{z}}_g \in Z \tag{1.4.11}$$

where $\forall \bar{\mathbf{z}}_g \in Z$ denotes for "all $\bar{\mathbf{z}}_g$ in Z."

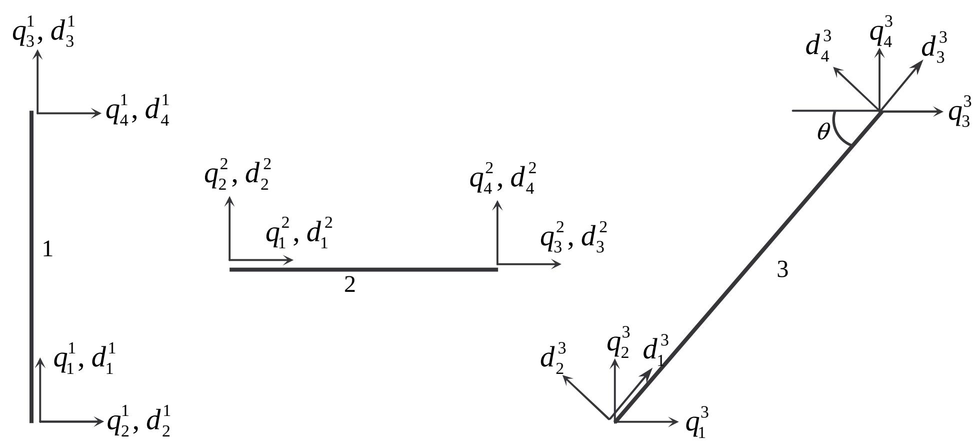

Fig. 1.4.2. Body-Fixed and Globally Oriented Element Coordinate System

In actual computation, the global variational equation is modified to explicitly eliminate the boundary condition. However, this is not the general case. In many FEA codes, the size of $\mathbf{K}_g$ is retained. Instead of explicitly removing rows and columns corresponding to the boundary conditions, equivalent relations are substituted to make $\mathbf{K}_g$ positive definite. Since z_3 and z_4 are prescribed, they can be eliminated from the variational equation. In addition, since z_5 and z_6 has a relation, z_6 can be expressed in terms of z_5, as in Eq. (1.4.10).

Consider the case in which $\theta = 45^o$ and $\alpha = 30^o$, and define the reduced global displacement vector as $\mathbf{z} = [z_1,\ z_2,\ z_5]^T$. Accordingly, the reduced global load vector is defined as $\mathbf{F} = [f_1,\ f_2,\ 0]^T$. By removing the third and the fourth rows and columns of $\mathbf{K}_g$, and by substituting the relation $z_6 = -\sqrt{3}z_5$, the reduced stiffness matrix in this example would be

$$\mathbf{K}(\mathbf{u}) = \frac{E}{2\sqrt{2}l}\begin{bmatrix} b_3 & b_3 & (\sqrt{3}-1)b_3 \\ b_3 & 2\sqrt{2}b_1 + b_3 & (\sqrt{3}-1)b_3 \\ (\sqrt{3}-1)b_3 & (\sqrt{3}-1)b_3 & 2\sqrt{2}b_2 + (4-2\sqrt{3})b_3 \end{bmatrix} \tag{1.4.12}$$

Thus, the reduced global matrix equation is written as

$$\mathbf{Kz} = \mathbf{F} \tag{1.4.13}$$

If $f_1 = f_2 = 1$ and $l = 1$, then the solution of the reduced global matrix Eq. (1.4.13) is obtained as

$$\mathbf{z} = \begin{bmatrix} \frac{4-2\sqrt{3}}{Eb_2} + \frac{2\sqrt{2}}{Eb_3} & 0 & \frac{1-\sqrt{3}}{Eb_2} \end{bmatrix}^T \tag{1.4.14}$$

The solution method in Eq. (1.4.13) is easier than that of Eq. (1.4.11). However, for general, complex kinematic constraints, it may not be easy to explicitly construct the reduced matrix $\mathbf{K}$. In addition, many FEA codes do not generate a reduced matrix $\mathbf{K}$ during the solution procedure. Thus, the use of Eq. (1.4.11) is clear. In the next section, we will discuss how solution $\mathbf{z}$ in Eq. (1.4.14) can be changed as a function of design vector $\mathbf{b}$.

1.5 Structural Design Sensitivity Analysis

Design sensitivity analysis is used to compute the rate of performance measure change with respect to design variable changes. Obviously, the performance measure is presumed to be a differentiable function of the design, at least in the neighborhood of the current design point. For complex engineering applications, it is not simple to prove a performance measure's differentiability with respect to the design. Consequently, the question of differentiability will be postponed until Chapters 5 and 6. For most problems in this text, one can assume that the performance measure is differentiable with respect to the design.

In general, a structural performance measure depends on the design. For example, a change in the cross-sectional area of a beam would affect the structural weight. This type of dependence is simple if the expression of weight in terms of the design variables is known. For example, the weight of a straight beam with a circular cross-section can be expressed as

$$W(r) = \pi r^2 l$$

where $u = r$ is the radius and l is the length of the beam. If the radius is a design variable, then the design sensitivity of W with respect to r would be

$$\frac{dW}{dr} = 2\pi r l$$

This type of function is *explicitly dependent* on the design, since the function can be explicitly written in terms of that design. Consequently, only algebraic manipulation is involved and no finite element analysis is required to obtain the design sensitivity of an explicitly dependent performance measure.

However, in most cases, a structural performance measure does not explicitly depend on the design. For example, when the stress of a beam is considered as a performance measure, there is no simple way to express the design sensitivity of stress explicitly in terms of the design variable r. In the linear elastic problem, the stress of the structure is determined from the displacement, which is a solution to the finite element analysis. Thus, the sensitivity of stress $\sigma(z)$ can be written as

$$\frac{d\sigma}{dr} = \frac{d\sigma}{d\mathbf{z}}^T \frac{d\mathbf{z}}{dr} \tag{1.5.1}$$

where $\mathbf{z}$ is the displacement of the beam. Since the expression of stress as a function of displacement is known, $d\sigma/d\mathbf{z}$ can be easily obtained. The only difficulty is the computation of $d\mathbf{z}/dr$, which is the state variable (displacement) sensitivity with respect to the design variable r.

When a design engineer wants to compute the design sensitivity of performance measures such as stress $\sigma(z)$ in Eq. (1.5.1), structural analysis (finite element analysis, for example) has presumably already been carried out. Assume that the structural problem is governed by the following linear algebraic equation

$$\mathbf{K}(u)\mathbf{z} = \mathbf{f}(u) \tag{1.5.2}$$

Equation (1.5.2) is a matrix equation of finite elements if $\mathbf{K}$ and $\mathbf{f}$ are understood to be the stiffness matrix and load vector, respectively. Suppose the explicit expressions of $\mathbf{K}(u)$ and $\mathbf{f}(u)$ are known and differentiable with respect to u. Since the stiffness matrix $\mathbf{K}(u)$ and load vector $\mathbf{f}(u)$ depend on the design u, solution $\mathbf{z}$ also depends on the design u. However, it is important to note that this dependency is implicit, which is why we need to develop a design sensitivity analysis methodology. As shown in Eq. (1.5.1), $d\mathbf{z}/du$ must be computed using the governing equation of Eq. (1.5.2). This can be achieved by differentiating Eq. (1.5.2) with respect to u, as

$$\mathbf{K}(u)\frac{d\mathbf{z}}{du} = \frac{d\mathbf{f}}{du} - \frac{d\mathbf{K}}{du}\mathbf{z} \tag{1.5.3}$$

Assuming that the explicit expressions of $\mathbf{K}(u)$ and $\mathbf{f}(u)$ are known, $d\mathbf{K}/du$ and $d\mathbf{f}/du$ can be evaluated. Thus, if solution $\mathbf{z}$ in Eq. (1.5.2) is known, then $d\mathbf{z}/du$ can be computed from Eq. (1.5.3), which can then be substituted into Eq. (1.5.1) to compute $d\sigma/du$. Note that the stress performance measure is *implicitly dependent* on the design through state variable $\mathbf{z}$.

In this text, it is assumed that the general performance measure ψ depends on the design explicitly and implicitly. That is, the performance measure ψ is presumed to be a function of design u, and state variable $\mathbf{z}(u)$, as

$$\psi = \psi\left(\mathbf{z}(u), u\right) \tag{1.5.4}$$

The sensitivity of ψ can thus be expressed as

$$\frac{d\psi(\mathbf{z}(u),u)}{du} = \left.\frac{\partial\psi}{\partial u}\right|_{\mathbf{z}=const} + \left.\frac{\partial\psi}{\partial\mathbf{z}}\right|_{u=const}^{T} \frac{d\mathbf{z}}{du} \tag{1.5.5}$$

The only unknown term in Eq. (1.5.5) is $d\mathbf{z}/du$. Various computational methods to obtain $d\mathbf{z}/du$ are introduced in the following sub–sections.

1.5.1 Methods of Structural Design Sensitivity Analysis

Various methods employed in design sensitivity analysis are listed in Fig. 1.5.1. Three approaches are used to obtain the design sensitivity: the approximation, discrete, and continuum approach. In the approximation approach, design sensitivity is obtained by either the *forward finite difference* or the *central finite difference method*. In the discrete method, design sensitivity is obtained by taking design derivatives of the discrete governing equation. For this process, it is necessary to take the design derivative of the stiffness matrix. If this derivative is obtained analytically using the explicit expression of the stiffness matrix with respect to the design variable, it is an *analytical method*, since the analytical expressions of $\mathbf{K}(u)$ and $\mathbf{f}(u)$ are used. However, if the derivative is obtained using a finite difference method, the method is called a *semi-analytical method*. In the continuum approach, the design derivative of the variational equation is taken before it is discretized. If the structural problem and sensitivity equations are solved as a continuum problem, then it is called the *continuum–continuum method*. However, only very simple, classical problems can be solved analytically. Thus, the continuum sensitivity equation is solved by discretization in the same way that structural problems are solved. Since differentiation is taken at the continuum domain and is then followed by discretization, this method is called the *continuum–discrete method*. These methods will be explained in detail in the following sections.

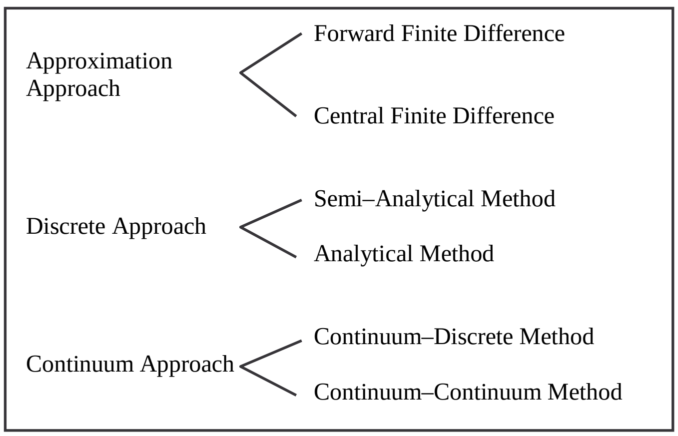

Fig. 1.5.1. Approaches to Design Sensitivity Analysis

1.5.2 Finite Difference Method

The easiest way to compute sensitivity information of the performance measure is by using the finite difference method. Different designs yield different analysis results and, thus, different performance values. The finite difference method actually computes design sensitivity of performance by evaluating performance measures at different stages in the design process. If u is the current design, then the analysis results provide the value of performance measure $\psi(u)$. In addition, if the design is perturbed to $u + \Delta u$, where Δu represents a small change in the design, then the sensitivity of $\psi(u)$ can be approximated as

$$\frac{d\psi}{du} \approx \frac{\psi(u + \Delta u) - \psi(u)}{\Delta u} \tag{1.5.6}$$

Equation (1.5.6) is called the *forward difference method* since the design is perturbed in the direction of +Δu. If –Δu is substituted in Eq. (1.5.6) for Δu, then the equation is defined as the *backward difference method*. Additionally, if the design is perturbed in both directions, such that the design sensitivity is approximated by

$$\frac{d\psi}{du} \approx \frac{\psi(u + \Delta u) - \psi(u - \Delta u)}{2\Delta u} \tag{1.5.7}$$

then the equation is defined as the *central difference method*.

The advantage of the finite difference method is obvious. If structural analysis can be performed and the performance measure can be obtained as a result of structural analysis, then the expressions in Eqs.

(1.5.6) and (1.5.7) are virtually independent of the problem types considered. Consequently, this method is still popular in engineering design.

However, sensitivity computation costs become the dominant concern in the design process. If n represents the number of designs, then n+1 analyses have to be carried out for either the forward or backward difference method, and $2n$+1 analyses are required for the central difference method. For modern, practical engineering applications, the cost of structural analysis is rather expensive. Thus, this method is infeasible for large-scale problems containing many design variables.

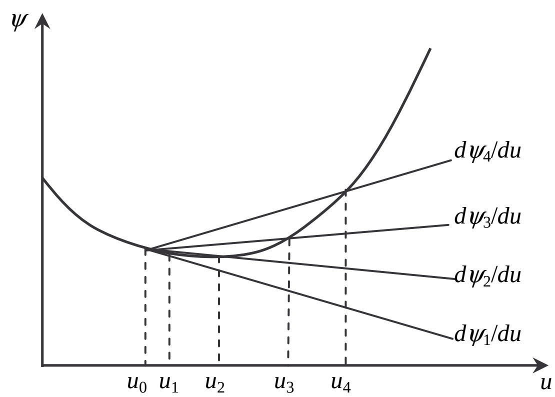

Fig. 1.5.2. Influence of Step–size in Forward Finite Difference Method

Another major disadvantage of the finite difference method is the accuracy of its sensitivity results. In Eq. (1.5.6), accurate results can be expected when Δu approaches zero. Figure 1.5.2 shows some sensitivity results using the finite difference method. The tangential slope of the curve at u_0 is the exact sensitivity value. Depending on perturbation size, we can see that sensitivity results are quite different. For a mildly nonlinear performance measure, relatively large perturbation provides a reasonable estimation of sensitivity results. However, for highly nonlinear performances, a large perturbation yields completely inaccurate results. Thus, the determination of perturbation size greatly affects the sensitivity result. And even though it may be necessary to choose a very small perturbation, numerical noise becomes dominant for a too small perturbation size. That is, with a too small perturbation, no reliable difference can be found in the analysis results. For example, if up to five digits of significant numbers are valid in a structural analysis, then any design perturbation in the finite difference that is smaller than the first five significant digits cannot provide meaningful results. As a result, it is very difficult to determine design perturbation sizes that work for all problems.

Example 1.5.1 Three–bar Truss (Finite Difference Method)

Consider the three–bar truss example shown in Fig. 1.2.3. In this example, the finite element matrix equation, Eq. (1.4.13), can be solved analytically with the solution given in Eq. (1.4.14) as a function of the design variable vector $\mathbf{u} = [b_1, b_2, b_3]^T$, which is the cross-sectional area of the truss elements. For simplicity, if the current value of the design is $\mathbf{u} = [1, 1, 1]^T$, and $E = 1$, then the solution becomes

$$\mathbf{z} = \left[4 - 2\sqrt{3} + 2\sqrt{2}, \quad 0, \quad 1 - \sqrt{3}\right]^T \tag{1.5.8}$$

Let us compute the design sensitivity of z_1 by using the finite difference method. Since the dependence of z_1 on the design is explicitly given, z_1 can be straightforwardly computed at different design stages. Table 1.5.1 shows the sensitivities of z_1 with different perturbation sizes. As the perturbation size decreases, the sensitivity value using finite difference method approaches an exact sensitivity value. In many cases, the central finite difference method is more accurate than the forward/backward finite difference method, as shown in Table 1.5.1, although for the central finite difference method, two performance measure evaluations are involved. Note that, in the latter case, since Eq. (1.4.14) is the exact solution of the matrix equation in Eq. (1.4.13), there is no concern with numerical noise. That is, as the design perturbation size decreases, the finite difference results will converge to the exact design sensitivities.

Table 1.5.1. Sensitivity Results of Finite Difference Method

Design	Forward FDM			Central FDM			Exact
	Δb=0.5	Δb=0.1	Δb=0.01	Δb=0.5	Δb=0.1	Δb=0.01	
b_1	0	0	0	0	0	0	0
b_2	–0.35727	–0.48718	–0.53059	–0.71453	–0.54131	–0.53595	–0.53590
b_3	–1.88561	–2.57130	–2.8004	–3.77124	–2.85700	–2.82871	–2.82843

1.5.3 Discrete Method

A structural problem is often discretized in finite dimensional space in order to solve complex problems, as shown with the finite element method in Section 1.4. The discrete method computes the performance design sensitivity of the discretized problem, where the governing equation is a system of linear equations, as in Eq. (1.5.2). If the explicit form of the stiffness matrix $\mathbf{K}(u)$ and the load vector $\mathbf{f}(u)$ are known, and if solution $\mathbf{z}$ of matrix equation $\mathbf{Kz} = \mathbf{f}$ is obtained, then the design sensitivity of the displacement vector can also be obtained, as

$$\mathbf{K}(u)\frac{d\mathbf{z}}{du} = \frac{d\mathbf{f}}{du} - \frac{d\mathbf{K}}{du}\mathbf{z} \tag{1.5.9}$$

This is a discrete approach to the *analytical method*, since the explicit expressions of $\mathbf{K}(u)$ and $\mathbf{f}(u)$ are used to obtain design derivatives of the stiffness matrix and load vector. Even if the expression of Eq. (1.5.9) is in the global system matrix, actual computation of these derivatives can still be carried out on the element level in order to avoid a massive amount of calculation related to global stiffness matrix $\mathbf{K}$. An in–depth discussion of this method is presented in Chapter 4.

It is not difficult to compute $d\mathbf{f}/du$, since the applied force is usually either independent of the design, or it has a simple expression. However, the computation of $d\mathbf{K}/du$ in Eq. (1.5.9) depends on the type of problem. In addition, modern advances in the finite element method use numerical integration in the computation of $\mathbf{K}$. In this case, the explicit expression of $\mathbf{K}$ in terms of u may not be available. Moreover, in the case of the shape design variable, computation of the analytical derivative of the stiffness matrix is quite costly. Because of this, the semi–analytical method is a popular choice for discrete shape design sensitivity analysis approaches. However, Barthelemy and Haftka [7] show that the semi–analytical method can have serious accuracy problems for shape design variables in structures modeled by beam, plate, truss, frame, and solid elements. They found that accuracy problems occur even for a simple cantilever beam. Moreover, errors in the early stage of approximation multiply during the matrix equation solution phase. As a remedy, Olhoff *et al*. [8] proposed an exact numerical differentiation method when the analytical form of the element stiffness matrix is available.

Example 1.5.2 Three–bar Truss (Discrete Method)

To obtain discrete design sensitivity with respect to sizing design variables, consider the three bar truss problem. More complicated design variables, namely, shape and configuration, will be considered in Chapters 6 and 7. Let us begin with the reduced global stiffness matrix given in Eq. (1.4.12). Since the explicit form of $\mathbf{K}(\mathbf{u})$ is given as a function of design variable $\mathbf{u}$, its derivative can be obtained as

$$\frac{d\mathbf{K}}{db_1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \frac{d\mathbf{K}}{db_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \frac{d\mathbf{K}}{db_3} = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & \sqrt{3}-1 \\ 1 & 1 & \sqrt{3}-1 \\ \sqrt{3}-1 & \sqrt{3}-1 & 4-2\sqrt{3} \end{bmatrix} \tag{1.5.10}$$

and load vector **f** is independent of the design, i.e., $d\mathbf{f}/du$ = 0. The right side of Eq. (1.5.9) can be obtained by multiplying Eq. (1.5.10) with Eq. (1.5.8) for each design variable. If $\mathbf{F}^u$ denotes the right side of Eq. (1.5.9), then its explicit expression would be

$$\mathbf{F}^{b_1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{F}^{b_2} = \begin{bmatrix} 0 \\ 0 \\ \sqrt{3}-1 \end{bmatrix}, \quad \mathbf{F}^{b_3} = \begin{bmatrix} -1 \\ -1 \\ 1-\sqrt{3} \end{bmatrix} \tag{1.5.11}$$

Since the right side, corresponding to the first design variable, vanishes, the sensitivity of displacement with respect to b_1 also vanishes. By solving Eq. (1.5.9) with respect to $d\mathbf{z}/du$, we obtain

$$\frac{d\mathbf{z}}{db_1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \frac{d\mathbf{z}}{db_2} = \begin{bmatrix} 2\sqrt{3}-4 \\ 0 \\ \sqrt{3}-1 \end{bmatrix}, \quad \frac{d\mathbf{z}}{db_3} = \begin{bmatrix} -2\sqrt{2} \\ 0 \\ 0 \end{bmatrix} \tag{1.5.12}$$

which is the same result as the direct computation of sensitivity in Eq. (1.4.14).

Example 1.5.3 Cantilever Beam (Discrete Method)

Consider the cantilever beam in Fig. 1.5.3 with point load p at $x = l$. If one finite element is used to discretize the structure, as shown in Fig. 1.5.3(b), then displacement $z(x)$ can be approximated as

$$z(x) = \mathbf{N}^T\mathbf{z}_g = [\,N_1 \; N_2 \; N_3 \; N_4\,] \begin{bmatrix} z_1 \\ \theta_1 \\ z_2 \\ \theta_2 \end{bmatrix} \tag{1.5.13}$$

where z_1 and z_2 are nodal displacement, θ_1 and θ_2 are nodal rotations, and N_i's are corresponding shape functions of the approximation, defined as

$$\begin{aligned} N_1 &= 1 - \frac{3x^2}{l^2} + \frac{2x^3}{l^3}, \qquad N_2 = x - \frac{2x^2}{l} + \frac{x^3}{l^2} \\ N_3 &= \frac{3x^2}{l^2} - \frac{2x^3}{l^3}, \qquad N_4 = -\frac{x^2}{l} + \frac{x^3}{l^2} \end{aligned} \tag{1.5.14}$$

The variational equation of the beam bending problem in a continuum model can be written as

$$\int_0^l EIz_{,xx}\bar{z}_{,xx}\,dx = \int_0^l p\delta(x-l)\bar{z}\,dx, \qquad \forall \bar{z} \in Z \tag{1.5.15}$$

where E is Young's modulus, $I = bu^3/12$ is the second moment of inertia, $\delta(x-l)$ is the Dirac delta measure which has a value of infinity at $x = l$ and zero otherwise, and Z is the space of kinematically admissible displacements. For the moment, Z can be thought of as the space of smooth functions that satisfy the boundary condition $z(0) = \theta(0) = 0$.

To obtain a finite element equation, the approximation in Eq. (1.5.13) is substituted into variational Eq. (1.5.15) to obtain

$$\bar{\mathbf{z}}_g^T\mathbf{K}_g\mathbf{z}_g = \bar{\mathbf{z}}_g^T\mathbf{F}_g, \qquad \forall \bar{\mathbf{z}}_g \in Z_h \tag{1.5.16}$$

where $Z_h = \{\mathbf{z} \in R^4 | z_1(0) = \theta_1(0) = 0\}$ is the discretized version of Z. $\mathbf{K}_g$ and $\mathbf{F}_g$ are given as

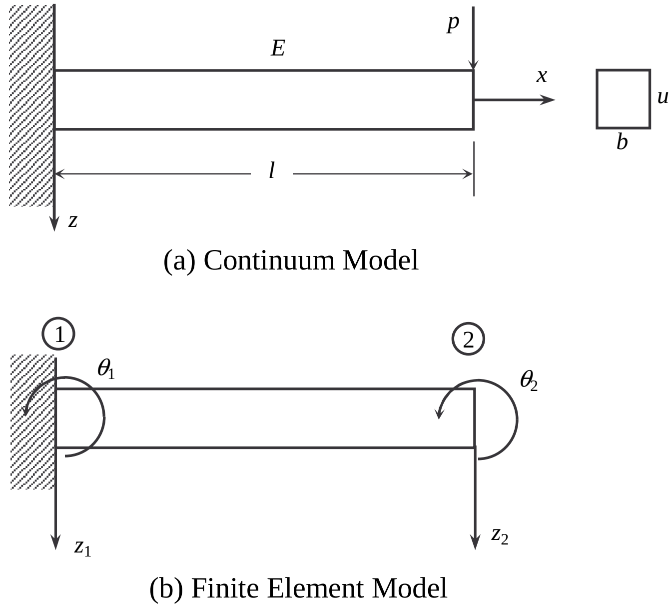

Fig. 1.5.3. Cantilever Beam

$$\mathbf{K}_g = \frac{EI}{\ell^3}\begin{bmatrix} 12 & 6\ell & -12 & 6\ell \\ 6\ell & 4\ell^2 & -6\ell & 2\ell^2 \\ -12 & -6\ell & 12 & -6\ell \\ 6\ell & 2\ell^2 & -6\ell & 4\ell^2 \end{bmatrix} \tag{1.5.17}$$
$$\mathbf{F}_g = \begin{bmatrix} 0 \; 0 \; p \; 0 \end{bmatrix}^T$$

For the cantilever beam shown in Fig. 1.5.3, the boundary condition is given such that $z_1(0) = \theta_1(0) = 0$. Thus, matrix $\mathbf{K}_g$ and vector $\mathbf{F}_g$ can be reduced only for z_2 and θ_2, as

$$\mathbf{Kz} \equiv \frac{EI}{\ell^3}\begin{bmatrix} 12 & -6\ell \\ -6\ell & 4\ell^2 \end{bmatrix}\begin{bmatrix} z_2 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} p \\ 0 \end{bmatrix} \equiv \mathbf{F} \tag{1.5.18}$$

The solution $\mathbf{z}$, and thus $\mathbf{z}_g$, can be obtained by solving Eq. (1.5.18) as

$$\mathbf{z}_g = \begin{bmatrix} 0 \; 0 \; \frac{p\ell^3}{3EI} \; \frac{p\ell^2}{2EI} \end{bmatrix}^T \tag{1.5.19}$$

and by using approximation in Eq. (1.5.13), displacement function $z(x)$ can be obtained as

$$z(x) = \mathbf{N}^T\mathbf{z}_g = \frac{p}{6EI}(-x^3 + 3\ell x^2) \tag{1.5.20}$$

Note that $z(x)$ in Eq. (1.5.20) is the exact solution in this special example.

The discrete method of design sensitivity can be obtained by differentiating the finite element matrix Eq. (1.5.18) with respect to the design. Consider height u of the cross-sectional dimension as a design variable. The design sensitivity equation can be obtained from Eq. (1.5.18) using a procedure similar to that in Eq. (1.5.9), as

$$\mathbf{K}(u)\frac{d\mathbf{z}}{du} = \frac{d\mathbf{F}}{du} - \frac{d\mathbf{K}}{du}\mathbf{z} \equiv \mathbf{F}^u \tag{1.5.21}$$

where $d\mathbf{F}/du = \mathbf{0}$, since $\mathbf{F}$ is independent of the design, and $d\mathbf{K}/du$ is calculated from Eq. (1.5.18) as

$$\frac{d\mathbf{K}}{du} = \frac{3EI}{\ell^3 u}\begin{bmatrix} 12 & -6\ell \\ -6\ell & 4\ell^2 \end{bmatrix} \tag{1.5.22}$$

Thus, the right side of Eq. (1.5.21) can be computed as

$$\mathbf{F}^u = \frac{d\mathbf{F}}{du} - \frac{d\mathbf{K}}{du}\mathbf{z} = -\frac{3EI}{\ell^3 u}\begin{bmatrix} 12 & -6\ell \\ -6\ell & 4\ell^2 \end{bmatrix}\begin{bmatrix} \frac{p\ell^3}{3EI} \\ \frac{p\ell^2}{2EI} \end{bmatrix} = \begin{bmatrix} -\frac{3p}{u} \\ 0 \end{bmatrix} \tag{1.5.23}$$

and the design sensitivity of the displacement vector can be solved from Eq. (1.5.21) as

$$\frac{d\mathbf{z}}{du} = \frac{\ell}{12EI}\begin{bmatrix} 4\ell^2 & 6\ell \\ 6\ell & 12 \end{bmatrix}\begin{bmatrix} -\frac{3p}{u} \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{p\ell^3}{EIu} \\ -\frac{3p\ell^2}{2EIu} \end{bmatrix} \tag{1.5.24}$$

Since the shape function of the finite element approximation is independent of the design, the interpolation in Eq. (1.5.13) is valid for displacement sensitivity. Thus, the displacement function sensitivity can be obtained as

$$\frac{dz(x)}{du} = \mathbf{N}^T\frac{d\mathbf{z}}{du} = \frac{p}{2EIu}x^2(x-3\ell) \tag{1.5.25}$$

Equation (1.5.25) can be verified by directly differentiating the exact solution in Eq. (1.5.20).

1.5.4 Continuum Method

In the continuum method, the design derivative of the variational equation (the continuum model of the structure) is taken before discretization. Since differentiation is taken before any discretization takes place, this method provides more accurate results than the discrete approach. In addition, profound mathematical proofs are available regarding the existence and uniqueness of the design sensitivity. Most discussions in this text focus on the continuum method, in which analytical expressions of design sensitivity are obtained in the continuum setting.

Sizing design variables are distributed parameters of the continuum equation. For shape design variables, the material derivative concept of continuum mechanics is used to relate variations in structural shape to the structural performance measures [5]. Using the continuum design sensitivity analysis approach, design sensitivity expressions are obtained in the form of integrals, with integrands written in terms of such physical quantities as displacement, stress, strain, and domain shape change. If exact solutions to the continuum equations are used to evaluate these design sensitivity expressions, then this procedure is referred to as the continuum–continuum method. On the other hand, if approximation methods such as the finite element, boundary element, or meshfree method are used to evaluate these terms, then this procedure is called the continuum–discrete method. The continuum–continuum method provides the exact design sensitivity of the exact model, whereas the continuum–discrete method provides an approximate design sensitivity of the exact model. When FEA is used to evaluate the structural response, then the same discretization method as structural analysis has to be used to compute the design sensitivity of performance measures in the continuum–discrete method.

Example 1.5.4 Cantilever Beam (Continuum Method)

Continuum–based design sensitivity analysis is used to differentiate the variational Eq. (1.5.15) for the cantilever beam discussed in Example 1.5.3. As with the discrete method, the right side of Eq. (1.5.15) is independent of the design. Let us define differentiation or variation as

$$z' = \frac{dz}{du}\delta u \tag{1.5.26}$$

where δu is the amount of perturbation. The left side of Eq. (1.5.15) can be differentiated with respect to design u as

$$\frac{d}{du}\left[\int_0^l EIz_{,xx}\bar{z}_{,xx}\,dx\right]\delta u = \int_0^l EIz'_{,xx}\bar{z}_{,xx}\,dx + \int_0^l \frac{3EI}{u}z_{,xx}\bar{z}_{,xx}\delta u\,dx \tag{1.5.27}$$

Thus, the continuum–based design sensitivity equation is obtained as, with δu= 1,

$$\int_0^l EIz'_{,xx}\bar{z}_{,xx}\,dx = -\int_0^l \frac{3EI}{u}z_{,xx}\bar{z}_{,xx}\,dx, \qquad \forall \bar{z} \in Z \tag{1.5.28}$$

which yields the solution $z' = dz/du$. The continuum–continuum method solves Eq. (1.5.28) to obtain z' directly, whereas the continuum–discrete method first discretizes Eq. (1.5.28), following the same procedure as the finite element method. If the same approximation is used for z' as displacement function z in Eq. (1.5.13), then the left side of Eq. (1.5.28) becomes equivalent to Eq. (1.5.15) by considering $\mathbf{z}_g$ as $\mathbf{z}'_g$. The discretized design sensitivity equation therefore becomes

$$\bar{\mathbf{z}}_g\mathbf{K}_g\mathbf{z'}_g = \bar{\mathbf{z}}_g\mathbf{F}_g^f, \qquad \forall \bar{\mathbf{z}}_g \in Z_h \tag{1.5.29}$$

which can be solved using the same procedure as in Example 1.5.3.

Note that in the continuum method it is neither necessary to differentiate the stiffness matrix $d\mathbf{K}/du$, nor to use any matrix multiplication procedure to calculate $(d\mathbf{K}/du)\cdot\mathbf{z}$, which involves a large amount of additional computational cost.

One frequently asked question is: "Are the discrete and continuum–discrete methods equivalent?" To answer this question, four conditions have to be given. First, the same discretization (shape function) used in the FEA method must be used for continuum design sensitivity analysis. Second, an exact integration (instead of a numerical integration) must be used in the generation of the stiffness matrix and in the evaluation of continuum–based design sensitivity expressions. Third, the exact solution (and not a numerical solution) of the finite element matrix equation and the adjoint equation should be used to compare these two methods. Fourth, the movement of discrete grid points must be consistent with the design parameterization method used in the continuum method. For the sizing design variable, it is shown in [5] that the discrete and continuum–discrete methods are equivalent under the conditions given above, using a beam as the structural component. It has also been argued that the discrete and continuum–discrete methods are equivalent in shape design problems under the conditions given above [9]. One point to note is that these four conditions are not easy to satisfy; in many cases, numerical integration is used and exact solutions of the FE matrix equation cannot be obtained.

1.5.5 Summary of Design Sensitivity Analysis Approaches

As explained in previous sections, the design sensitivity analysis method has been developed along two fundamentally different paths, as shown in Fig. 1.5.1. In the discrete method, design derivatives of a discretized structural FEA equation are taken to obtain design sensitivity information. In the continuum method, design derivatives of the variational governing equation are taken to obtain explicit design sensitivity expressions in integral form with integrands written in terms of the following variations: material property, sizing, shape, and configuration design variables, and such natural physical quantities as displacement, stress, and strain [5]. The explicit design sensitivity expressions are then numerically evaluated using the analysis results of FEA codes. Unlike the finite difference method, the continuum method provides accurate design sensitivity information without recourse to the uncertainties of perturbation size. In addition, the continuum method does not require the derivatives of stiffness, mass, and damping matrices, as with the discrete method shown in Fig. 1.5.4. Another advantage of the continuum method is that it provides unified, structural design sensitivity analysis capability, so that it is possible to develop one design sensitivity analysis software system that works with a number of well–established analysis methods, such as FEA, the boundary element method, the p-method of FEA, and the meshfree method.

One important advantage of the continuum–based design sensitivity analysis method is that it is possible to compute the results of design sensitivity analysis of the established FEA/BEA/Meshfree

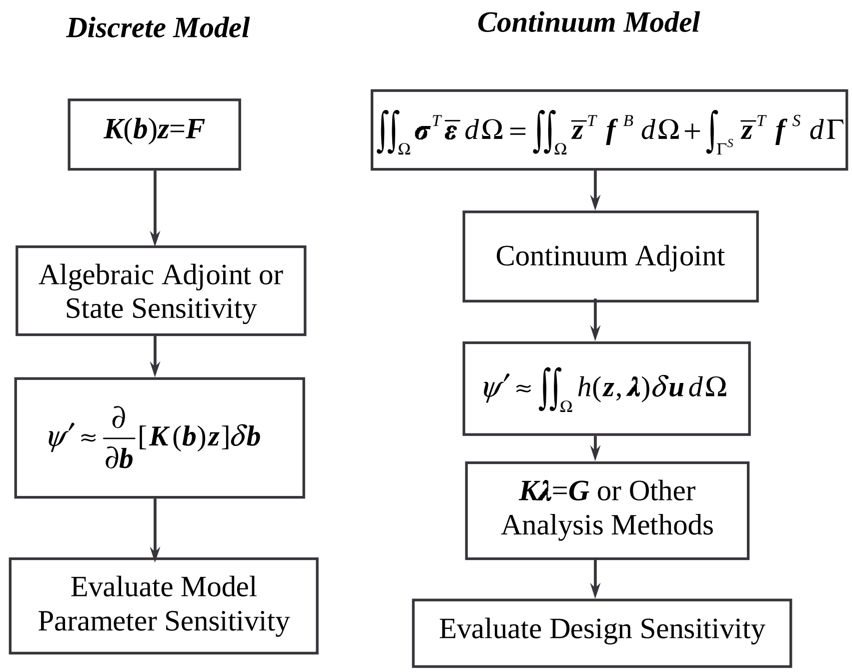

Fig. 1.5.4. Design Sensitivity Analysis Methods

codes with respect to the geometric design variables employed by CAD tools. For example, connections to CAD tools can be made by providing the design sensitivity of performance measures with respect to those design variables defined on the CAD tool. Using the same CAD-based design parameters in manufacturing tools lays the foundation for concurrent engineering. And once models are based on the same CAD tool, an integrated CAD–FEA–DSA system can be used to develop a design tool for a concurrent engineering method, such that design and manufacturing engineers can perform trade–off analysis in the early stages of the design process. A connection can also be made to multi-body dynamic simulation, computational fluid dynamics, and other computer–aided engineering (CAE) tools, if these tools use the same geometric CAD modeler as explained in Figs. 1.1.1 and 1.1.2.

1.6 Second–order Design Sensitivity Analysis

First–order design sensitivity analysis, which was introduced in the previous section, is a linear approximation of the performance measure in terms of the design. However, if a higher–order approximation is used, then the accuracy of the approximation obviously increases. Second–order design sensitivity analysis uses a quadratic formula in order to approximate the performance change. Let us consider a Taylor series expansion of $\psi(\mathbf{u})$ up to the quadratic terms, as

$$\psi(\mathbf{u} + \Delta\mathbf{u}) \approx \psi(\mathbf{u}) + \left(\frac{d\psi}{d\mathbf{u}}\right)^T \Delta\mathbf{u} + \frac{1}{2}\Delta\mathbf{u}^T \frac{d^2\psi}{d\mathbf{u}^2}\Delta\mathbf{u} \tag{1.6.1}$$

This approximation is exact if ψ is a quadratic function of $\mathbf{u}$. For general nonlinear performance, the quadratic approximation in Eq. (1.6.1) is much more accurate than linear approximation. In Eq. (1.6.1) the term $d^2\psi/d\mathbf{u}^2$ is called the *second–order design sensitivity* of ψ. For a general n dimensional design vector $\mathbf{u}$, the second–order design sensitivity becomes a $n \times n$ symmetric Hessian matrix, which involves $n(n$+1)/2 calculations.

Example 1.6.1 Three–bar Truss

Consider the three–bar truss example given in the previous section. For second–order design sensitivity, the design derivative is taken from the first–order sensitivity equation in Eq. (1.5.3), to obtain

$$\mathbf{K}\frac{d^2\mathbf{z}}{du^2} = \frac{d^2\mathbf{f}}{du^2} - \frac{d^2\mathbf{K}}{du^2}\mathbf{z} - 2\frac{d\mathbf{K}}{du}\frac{d\mathbf{z}}{du} \tag{1.6.2}$$

The derivatives of stiffness matrix with respect to design in Eq. (1.5.10) are constants. Thus, the second-order derivative of stiffness matrix $d^2\mathbf{K}/du^2$ vanishes, along with $d^2\mathbf{f}/du^2$, and Eq. (1.6.2) is simplified to

$$\mathbf{K}\frac{d^2\mathbf{z}}{du^2} = -2\frac{d\mathbf{K}}{du}\frac{d\mathbf{z}}{du} \tag{1.6.3}$$

By using Eqs. (1.5.10) and (1.5.12), Eq. (1.6.3) can be solved for $d^2\mathbf{z}/du^2$, yielding

$$\frac{d^2\mathbf{z}}{db_2^2} = \begin{bmatrix} 4(2-\sqrt{3}) \\ 0 \\ 2(1-\sqrt{3}) \end{bmatrix}, \quad \frac{d^2\mathbf{z}}{db_3^2} = \begin{bmatrix} 4\sqrt{2} \\ 0 \\ 0 \end{bmatrix} \tag{1.6.4}$$

which can be verified by differentiating the explicit expression of displacement in Eq. (1.4.14) twice with respect to the design. All other terms are zero.

Second–order sensitivity information is very useful for an optimization algorithm, since quadratic convergence can be achieved near the solution points if the Hessian information is available. However, computing second–order sensitivity results in quite large computational costs. Thus, the design engineer has to decide between computational cost and the optimization algorithm's efficiency.

1.7 Design Optimization

The purpose of many structural design problems is to find the best design among many possible candidates. For this reason, the design engineer has to specify the best possible design as well as the best possible candidates. As discussed in Section 1.2, a possible candidate must exist within a feasible design region to satisfy problem constraints. Every design in the feasible region is an acceptable design, even if it is not the best one. The best design is usually the one that minimizes (or maximizes) the cost function of the design problem. Thus, the goal of the design optimization problem is to find a design that minimizes the cost function among all feasible designs. Although design sensitivity analysis is the main focus of this text, because many optimized designs will be presented, design optimization algorithms are briefly introduced in this section. However, this brief discussion is by no means a complete treatment of optimization methods. For a more detailed treatment, refer to [10, 11, 12].

Most gradient-based optimization algorithms are based on the mathematical programming method, which requires the function values and sensitivity information at given design variables. For a given design variable that defines the structural model, structural analysis provides the values of the cost and constraint functions for the algorithm. Design sensitivities of the cost and constraint functions must also be supplied to the optimization algorithm. Then, the optimization algorithms, discussed in this section, calculate the best possible design of the problem. Each algorithm has its own advantages and disadvantages. The performance of an optimization algorithm critically depends on the characteristics of the design problem and the types of cost and constraint functions.

1.7.1 Linear Programming Method

The linear programming method can be used when cost and constraints are linear functions of the design variables [13]. Most structural design problems, however, are nonlinear with respect to their design variables. Thus, the linear programming method is not of much use for structural problems. However, a nonlinear problem can be solved by approximating a sequence of linear problems, which will be discussed in Section 1.7.3. The standard form of a linear programming problem is

$$\begin{aligned} \text{minimize} \quad & f = \mathbf{c}^T \mathbf{u} \\ \text{subject to} \quad & \mathbf{A}\mathbf{u} = \mathbf{b} \\ & \mathbf{u} \geq 0 \end{aligned} \tag{1.7.1}$$

where $\mathbf{c}$ = $[c_1, c_2, \ldots, c_n]^T$ is the coefficient of the cost function, $\mathbf{A}$ is the $m \times n$ matrix, and $\mathbf{b}$ is the $m \times 1$ vector. Inequality constraints can be treated as equality constraints by introducing slack variables. Since all functions are linear, the feasible regions defined by linear equalities are convex, along with the cost function.. Thus, if any optimum solution of Eq. (1.7.1) exists, then it is a global minimum solution of the problem. The reason for introducing the linear problem here is that a very efficient method exists for solving linear programming problems, namely *the simplex method*. A positive feature of a linear programming problem is that the solution always lies on the boundary of the feasible region. Thus, the simplex method finds a solution by moving each corner point of the convex boundary.

1.7.2 Unconstrained Optimization Problems

When cost and/or constraints are nonlinear functions of the design, the design problem is called a *nonlinear programming method*, as contrasted to the linear programming method discussed in the previous section. Most engineering problems fall into the former category. Because the properties of nonlinear programming are nonlinear, this method is frequently solved using the numerical, rather than the analytical, method.

When there are no constraints on the design problem, it is referred to as an *unconstrainedoptimization problem*. Even if most engineering problems have constraints, these problems can be transformed into unconstrained ones by using the penalty method, or the Lagrange multiplier method. The unconstrained optimization problem sometimes contains the lower and upper limits of a design variable, since this type of constraint can be treated in simple way. The standard form of an unconstrained optimization problem can be written as

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{u}) \\ \text{subject to} \quad & u_k^L \leq u_k \leq u_k^U, \qquad k = 1, \cdots, n \end{aligned} \tag{1.7.2}$$

In the following sub–sections, numerical methods for solving Eq. (1.7.2) are discussed.

Steepest Descent Method

The numerical procedure for solving Eq. (1.7.2) is an iterative update of design $\mathbf{u}$. If $\mathbf{u}^k$ is the value of the design at the k-th iteration, then the new design at the (k+1)-th iteration can be obtained by

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \alpha \mathbf{d}^{k+1} \tag{1.7.3}$$

where $\mathbf{d}^{k+1}$ is called the descent direction and α is a step size, used to determine the amount of movement in the direction of $\mathbf{d}^{k+1}$. If the descent direction is given, then parameter α is determined by using the line search procedure to find the minimum value of a cost function in the descent direction. The steepest descent method uses the gradient of the cost function as the descent direction, such that

$$\mathbf{d}^{k+1} = -\frac{\partial f(\mathbf{u}^k)}{\partial \mathbf{u}^k} = -\nabla f(\mathbf{u}^k) \tag{1.7.4}$$

which is the design sensitivity of the cost function. This method suffers from a slow convergence near the optimum design, since it does not use any information from the previous design, and only first order information of the function is used. Note that $\mathbf{d}^k$ and $\mathbf{d}^{k+1}$ are always orthogonal, such that a zigzagging pattern appears in the optimization process.

Conjugate Gradient Method

The conjugate gradient method developed by Fletcher and Reeves [14] improves the rate of slow convergence in the steepest descent method by using gradient information from the previous iteration. The difference in this method is the computation of $\mathbf{d}^{k+1}$ in Eq. (1.7.3). The new descent direction is computed by

$$\mathbf{d}^{k+1} = -\nabla f(\mathbf{u}^k) + \beta_k^2 \mathbf{d}^{k-1} \tag{1.7.5}$$

where

$$\beta_k = \frac{\left\|\nabla f(\mathbf{u}^k)\right\|}{\left\|\nabla f(\mathbf{u}^{k-1})\right\|} \tag{1.7.6}$$

and where the first iteration is the same as Eq. (1.7.3). This method tends to select the descent direction as a diagonal of two orthogonal steepest descent directions, such that a zigzagging pattern can be eliminated. This method always has better convergence than the steepest descent method.

Newton Method

The previous methods we have examined use first-order information (first-order design sensitivity) of the cost function to find the optimum design, which is called linear approximation. The Newton method uses second–order information (second-order design sensitivity) to approximate the cost function as a quadratic function of the design. The major concern is how to compute the second-order design sensitivity (or Hessian) matrix. Let us define the Hessian matrix as second-order design sensitivity, defined in Eq. (1.6.1) as

$$\mathbf{H}(\mathbf{u}^k) \equiv \left[\frac{\partial f(\mathbf{u}^k)}{\partial u_i^k \partial u_j^k}\right], \quad i, j = 1, \cdots, n \tag{1.7.7}$$

The new design can then be determined, as

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta\mathbf{u}^{k+1} \tag{1.7.8}$$

where

$$\Delta\mathbf{u}^{k+1} = -\mathbf{H}^{-1}(\mathbf{u}^k)\nabla f(\mathbf{u}^k) \tag{1.7.9}$$

If the current estimated design $\mathbf{u}^k$ is sufficiently close to the optimum design, then Newton's method will show a quadratic convergence. However, the greater the number of design variables, the greater the cost of computing $\mathbf{H}(\mathbf{u}^k)$ in Eq. (1.7.7). In addition, Newton's method does not guarantee a convergence. Thus, several modifications are available. For example, the design update algorithm in Eq. (1.7.8) can be modified to include a step size by using a line search, as in Eq. (1.7.3).

Quasi–Newton Method

Although Newton's method has a quadratic convergence, the cost of computing the Hessian matrix, and the lack of a guaranteed convergence, are drawbacks to this method. The quasi–Newton method has an advantage over the steepest descent method and Newton's method: it only requires first–order sensitivity information, and it approximates the Hessian matrix to speed up the convergence.

The DFP (Davidon–Fletcher–Powell [14]) method approximates the inverse of the Hessian matrix using first–order sensitivity information. By initially assuming that the inverse of the Hessian is the identity matrix, this method updates the inverse of the Hessian matrix during design iteration. A nice feature of this method is that the positive definiteness of the Hessian matrix is preserved.

The BFGS (Broydon–Fletcher–Goldfarb–Shanno [15]) method updates the Hessian matrix directly, rather than updating its inverse as with the DFP method. Starting from the identity matrix, the Hessian matrix remains positive definite if an exact line search is used.

1.7.3 Constrained Optimization Problems

Most engineering problems have constraints that must be satisfied during the design optimization process. These two types of constraints are handled separately: equality and inequality constraints. The standard form of the design optimization problem in constrained optimization can be written as

$$\begin{aligned} &\text{minimize} && f(\mathbf{u}) \\ &\text{subject to} && h_i(\mathbf{u}) = 0, \qquad i = 1, \cdots, p \\ &&& g_j(\mathbf{u}) \le 0, \qquad j = 1, \cdots, m \\ &&& u_l^L \le u_l \le u_l^U, l = 1, \cdots, n \end{aligned} \tag{1.7.10}$$

The computational method to find a solution to Eq. (1.7.10) has two phases: first, to find a direction $\mathbf{d}$ that can reduce the cost $f(\mathbf{u})$, while correcting for any constraint violations that are violated; and second, to determine the step size of movement αgn the direction of $\mathbf{d}$.

Sequential Linear Programming (SLP)

The SLP method approximates the nonlinear problem as a sequence of linear programming problems such that the simplex method in Section 1.7.1 may be used to find the solution to each iteration. By using function values and sensitivity information, the nonlinear problem in Eq. (1.7.10) is linearized in a similar way as Taylor's expansion method in the first order, as

$$\begin{aligned} &\text{minimize} && f(\mathbf{u}^k) + \nabla f^T \Delta\mathbf{u}^k \\ &\text{subject to} && h_i(\mathbf{u}^k) + \nabla h_i^T \Delta\mathbf{u}^k = 0, \qquad i = 1, \cdots, p \\ &&& g_i(\mathbf{u}^k) + \nabla g_i^T \Delta\mathbf{u}^k \le 0, \qquad j = 1, \cdots, m \\ &&& u_l^L \le u_l \le u_l^U, \qquad l = 1, \cdots, n \end{aligned} \tag{1.7.11}$$

Since all functions and their sensitivities at $\mathbf{u}^k$ are known, the linear programming problem in Eq. (1.7.11) can be solved using the simplex method for $\Delta\mathbf{u}^k$. Even if the sensitivity information is not used to solve a linear programming problem, design sensitivity information is required in order to approximate the nonlinear problem as a linear one with SLP. In solving Eq. (1.7.11) for $\Delta\mathbf{u}^k$, the move limit $\Delta\mathbf{u}_L^k \le \Delta\mathbf{u}^k \le \Delta\mathbf{u}_U^k$ is critically important for convergence.

Sequential Quadratic Programming (SQP)

Compared to previous methods, which use first–order sensitivity information to determine the search direction $\mathbf{d}$, SQP solves a quadratic sub–problem to find that search direction, which has both quadratic cost and linear constraints.

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{u}^k) + \nabla f^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T \mathbf{H}\mathbf{d} \\ \text{subject to} \quad & h_i(\mathbf{u}^k) + \nabla h_i^T \mathbf{d} = 0, \qquad i = 1, \cdots, p \\ & g_j(\mathbf{u}^k) + \nabla g_j^T \mathbf{d} \leq 0, \qquad j = 1, \cdots, m \end{aligned} \tag{1.7.12}$$

This special form of the quadratic problem can be effectively solved, for example, by using the Kuhn–Tucker condition and the simplex method. Starting from the identity matrix, the Hessian matrix $\mathbf{H}$ is updated at each iteration by using the aforementioned methods in unconstrained optimization algorithms. The advantage of solving Eq. (1.7.12) in this way is that for positive definite $\mathbf{H}$ the problem is convex and the solution is unique. Moreover, this method does not require the move limit as in SLP.

Constrained Steepest Descent Method

In the unconstrained optimization process detailed in Section 1.7.2, the descent direction $\mathbf{d}$ is obtained from the cost function sensitivity. When constraints exist, this descent direction has to be modified in order to include their effect. If constraints are violated, then these constraints are added to the cost function using a penalty method. Design sensitivity of the penalized cost function combines the effects of the original cost function and the violated constraint functions.

Constrained Quasi–Newton Method

If the linear approximation of constraints in SQP is substituted for a quadratic approximation, then the convergence rate of Eq. (1.7.12) will be improved. However, solving the optimization problem for quadratic cost and constraints is not an easy process. The constrained quasi–Newton method combines the Hessian information of constraints with the cost function by using the Lagrange multiplier method. Nevertheless, it is still necessary to compute the constraint function Hessian. The main purpose of the constrained quasi–Newton method is to approximate the Hessian matrix by using first–order sensitivity information. The extended cost function is

$$L(\mathbf{u}, \mathbf{v}, \mathbf{w}) = f(\mathbf{u}) + \sum_{i=1}^{p} v_i h_i(\mathbf{u}) + \sum_{j=1}^{m} w_i g_i(\mathbf{u}) \tag{1.7.13}$$

where both $\mathbf{v}$ = $[v_1, v_2, \ldots, v_p]^T$ and $\mathbf{w}$ = $[w_1, w_2, \ldots, w_m]^T$ are the Lagrange multipliers for equality and inequality constraints, respectively. Note that $\mathbf{w} > \mathbf{0}$. Let the second–order design sensitivity of L be $\nabla^2 L$. The extended quadratic programming problem of Eq. (1.7.12) thus becomes

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{u}^k) + \nabla f^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T \nabla^2 L \mathbf{d} \\ \text{subject to} \quad & h_i(\mathbf{u}^k) + \nabla h_i^T \mathbf{d} = 0, \qquad i = 1, \cdots, p \\ & g_j(\mathbf{u}^k) + \nabla g_j^T \mathbf{d} \leq 0, \qquad j = 1, \cdots, m \\ & w_l \geq 0, \qquad l = 1, \cdots, m \end{aligned} \tag{1.7.14}$$

Feasible Direction Method

The feasible direction method is designed to allow design movement within the feasible region in each iteration. Based on the previous design, the updated design reduces the cost function and remains in the feasible region. Since all designs are feasible, a design at any iteration can be used, even if it is not an optimum design. Since this method uses the linearization of functions as in SLP, it is difficult to maintain nonlinear equality constraints. Thus, this approach is used exclusively for inequality constraints. Search direction $\mathbf{d}$ can be found by solving the following linear sub–problem:

$$\begin{aligned} \text{minimize} \quad & \beta \\ \text{subject to} \quad & \nabla f^T \mathbf{d} \le \beta \\ & \nabla g_i^T \mathbf{d} \le \beta, \quad i = 1, \cdots, m_{active} \\ & -1 \le d_j \le 1, \quad j = 1, \cdots, n \end{aligned} \tag{1.7.15}$$

where m_{active} is the number of active inequality constraints. After finding a direction $\mathbf{d}$ that can reduce cost function and maintain feasibility, a line search is used to determine step size α.

Gradient Projection Method

The feasible direction method solves the linear programming problem to find the direction of the design change. The gradient projection method, however, uses a simpler method for computing this direction. The direction obtained by the steepest descent method is projected on the constraint boundary, such that the new design can move along the constraint boundary. Thus, the direction of the design change reduces the cost function while maintaining the constraint along its boundary. For a general nonlinear constraint, however, a small movement along the tangent line of the boundary will violate this constraint. Thus, in actual implementation, a correction algorithm has to be followed. The gradient projection method behaves well when the constraint boundary is moderately nonlinear.