CAMBRIDGE

Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

1 Introduction

Optimization is an integral part of engineering today. Engineers use optimization techniques to design civil structures, machine components, electrical circuits, plant layouts, chemical processes and so on. Indeed, one simply cannot make technological advances without optimization.

1.1 An Example of Optimization

While there are numerous examples of optimization, we will consider here a design problem posed in Figure 1.1 where a plate is subject to a uniform pressure loading on one face, and is fixed at the other. The plate can be modeled and analyzed using any of the popular finite element packages; we will discuss one such package, namely SOLIDWORKS ([1]), later in the text.

Based on finite element analysis (FEA), one can determine the stress distribution within the plate, as illustrated in Figure 1.2; the maximum stress happens to be around 515 MPa, and occurs on the periphery of the large hole, as expected.

A typical design objective now is to reduce the maximum stress, without increasing the mass of the plate. Further, the overall plate dimensions and the diameter of the larger hole cannot be modified. The location and size of the two smaller holes can, however, be modified. In this example, the engineer finds out, through trial-and-error, that the stress can be reduced by simply enlarging the two smaller holes. This is illustrated in Figure 1.3, where the maximum stress is reduced to 466 MPa; this is done by increasing the diameter of the two holes from 15 mm to 25 mm. The maximum stress now occurs at the periphery of one of the smaller holes. *Observe that, in the process of reducing the stress, the mass has also been reduced!*

One can now ask if the stress can be further reduced, i.e., *is there an optimal location and optimal diameter for the two smaller holes such that the maximum stress is minimized?* This is an example of shape optimization that we shall study later in the text.

1.2 Challenges

There are numerous such "simple" optimization examples in engineering. However, in the author's experience, while solving such problems engineers often run into several challenges: (1) *How does one translate the above problem*

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

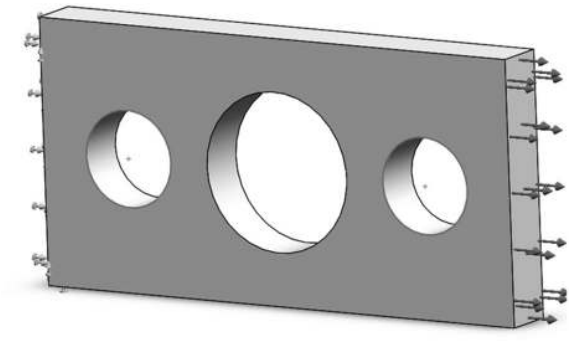

Figure 1.1 Design problem.

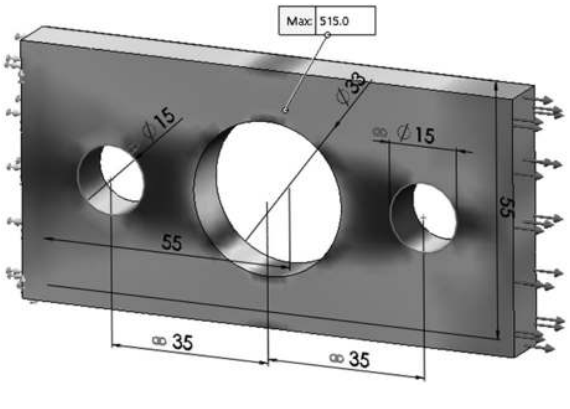

Figure 1.2 Stress plot based on finite element analysis (FEA).

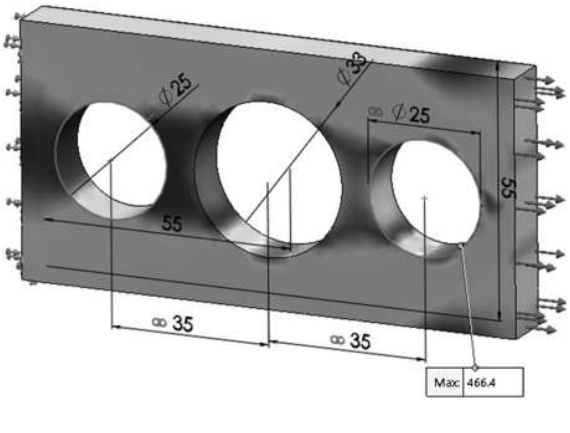

Figure 1.3 Reduced stress for a modified design.

into a formal optimization statement (with objective, constraints, feasible space, design variables, etc.)? (2) *What optimization method should one use and why?* (3) *What if the optimization method does not converge?* And so on.

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

After reading through this text, and completing the exercises, the author sincerely believes that the reader will be able to answer such questions confidently, and also extend the concepts to his/her field of study.

Study of optimization can be fun and enriching. However, for most effective learning, engineers should study optimization within the context of a relevant and familiar application. Learning to code sophisticated optimization methods, without understanding why these different methods exist in the first place, is both meaningless and counterproductive. Therefore, this text introduces fundamental optimization concepts through concrete applications.

For example, to introduce the important concept of *numerical scaling*, we will consider optimizing a truss system. We will observe that, without numerical scaling, even the best optimization method will not converge correctly. By considering a variety of such applications, fundamental optimization concepts can be assimilated easily.

No prior background in optimization is assumed in this text; basic undergraduate-level mathematics is sufficient. Prior experience in programming, especially MATLAB programming [2], is helpful. For this text, we will rely entirely on MATLAB programming, basics of which are covered in Chapters 3, 6 and 7.

1.3 MATLAB Code

The MATLAB code accompanying this text is an integral part of student learning, and can be downloaded as a zip-file from the author's website at www .ersl.wisc.edu under the "Research" tab. The code is organized chapter-wise; the use of this code is discussed in Chapter 3, where MATLAB is introduced.

1.4 Organization of Text

This text covers three complementary topics in engineering optimization: (1) the underlying mathematics, (2) numerical methods and nuances and (3) engineering applications; see Figure 1.4.

The text is organized as in Table 1.1; each chapter will introduce a critical mathematical concept/numerical method (identified by rows in the table) by considering a specific engineering application (identified by columns in the table).

Chapter 2 introduces the critical concept of *modeling*, i.e., the art of translating a loosely worded optimization problem into a formal mathematical statement. The notions of objective and constraint are introduced by considering various applications. This chapter will serve as an overview for the remainder of this text.

Chapter 3 is a short introduction to MATLAB. It is by no means an exhaustive review. The reader will be introduced to basic computing and plotting

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

Table 1.1 Text organization across various chapters.

Applications → *Concepts/Tools*↓	Analytical optimization	Truss optimization	Shape optimization
Modeling	Chapter 2	Chapter 2	Chapter 2
Basic MATLAB programming	Chapter 3		
Unconstrained theory	Chapter 4		
Basic algorithms	Chapter 5		
MATLAB optimization toolbox	Chapter 6		
Constrained theory	Chapter 7		
Specialized problems	Chapter 8		
Structural analysis		Chapter 9	
Numerical scaling		Chapter 10	
Gradient computation		Chapter 11	
Finite element analysis (2D)			Chapters 12, 13
Finite element analysis (3D)			Chapters 14, 15, 16

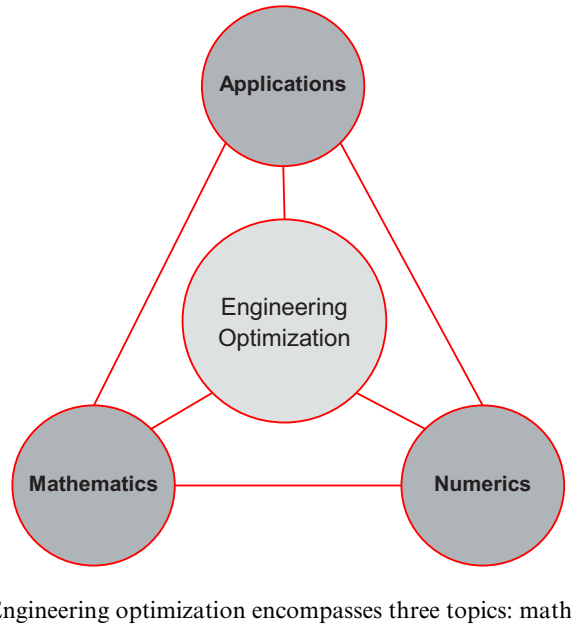

Figure 1.4 Engineering optimization encompasses three topics: mathematics, numerics and applications.

routines available within MATLAB. Various programming constructs such as the for-loop and if-then-else will be discussed. The concept of object-oriented programming will be reviewed through examples.

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

In Chapter 4, we will address optimization theory, focusing on the *unconstrained optimization* problems discussed in Chapter 2. Some of the questions raised in Chapter 2 will be answered. Critical concepts such as global/local minima and stationary points will be introduced.

Chapter 5 complements Chapter 4 in that we will *implement a few basic algorithms* to appreciate the nuances of numerical optimization. While engineers do not typically implement optimization algorithms "from scratch," numerical implementation will provide a good understanding of how algorithms work, and sometimes fail!

In Chapter 6, we will delve into MATLAB's *optimization toolbox* and study some of the algorithms for solving unconstrained problems. Using several test cases, we will observe the behavior of these methods, and correlate them to the observations made in the previous chapter.

Chapter 7 addresses the basic theory behind *constrained optimization*. Critical concepts such as Lagrange multipliers will be introduced within the context of engineering applications. Physical and mathematical interpretations of Lagrange multipliers will be discussed. We will also study MATLAB-supported algorithms for solving constrained problems.

Chapter 8 covers certain special types of optimization problems, including linear-programming problems, least-squares problems and multi-objective problems, that are not addressed in the previous chapters. One of the primary reasons for treating them separately is that they often require the use of specialized numerical methods.

In Chapter 9, trusses are analyzed through force balance and potential energy principles. The main concept emphasized is that "elastic structures, such as truss systems, when subject to an external load, reach a stable configuration when their potential energy reaches a local minimum." In other words, *physical systems behave in an optimal fashion*! This observation is both fascinating and important in engineering.

Chapter 10 will build on Chapter 9 by considering the *size optimization of truss systems*, where the goal is to find the optimal size (diameter) of truss members such that a given objective (say, the mass of the truss) is minimized, subject to certain constraints (such as deflection and stresses). MATLAB optimization methods introduced in Chapter 7 will be deployed to find optimal solutions to such problems. We will find that the algorithms do not always converge to the correct answer (even for simple problems). This will motivate the need for *numerical scaling* – a critical concept in numerical optimization! This concept will be covered and illustrated through several examples.

Chapter 11 will cover an equally important concept of gradient (i.e., sensitivity) computation. Gradient computation is critical if first-order methods of optimization are employed. This chapter covers the pitfalls of finite-difference-based gradient computation and provides alternative methods.

In Chapters 12 and 13, we consider FEA and optimization of 2D elastic problems. Conceptually, this is a direct extension of truss analysis and

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

optimization, discussed in Chapters 9 and 10 respectively. However, new concepts such as shape parameters and finite element discretization enter the picture.

Chapter 14 is an extension of Chapter 12 to 3D FEA. Here, we will rely on SOLIDWORKS for modeling and analysis. It is assumed that the reader is familiar with the process of creating 3D models and carrying out basic FEA within SOLIDWORKS. The objective of this chapter is to develop a foundation for parametric study and optimization to be pursued in the next chapter.

Chapter 15 introduces SOLIDLAB, an interface between SOLIDWORKS and MATLAB. SOLIDLAB was developed by the author and his graduate students. This chapter will illustrate the use of SOLIDLAB to query and analyze SOLIDWORKS models, from within the comfort of MATLAB.

Chapter 16 addresses 3D shape optimization by combining SOLIDWORKS, MATLAB and SOLIDLAB. The fundamental difference between compliance and stress minimization is highlighted.

The Appendix covers additional mathematical concepts and proofs.

Finally, it goes without saying that no textbook is ever complete or comprehensive. There are several excellent textbooks on engineering optimization (see references [3], [4], [5], [6], [7]) and numerical optimization (see references [8], [9]) that complement this text. The reader is strongly encouraged to consult these for topics not covered here.

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE

Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

2 Modeling

Highlights

1. This chapter discusses *modeling*, the first step in optimization. Modeling is the process of converting a loosely worded optimization problem into a formal mathematical statement.
2. Several modeling examples from geometry and structural mechanics are considered, and each example is converted into the standard formulation.
3. Through these examples, relevant optimization terminology is also explained.
4. Finally, important observations are made about modeling; these include: (1) introducing appropriate design variables, (2) exploiting optimality criteria for simplification and (3) the iterative nature of modeling.

Modeling is the process of converting a loosely worded "optimization" problem into a mathematically precise and standard formulation. For example, converting the stress minimization problem discussed in Section 1.1 into a formal optimization statement would be a modeling effort. Accurate modeling is crucial; an inaccurate model will lead to erroneous conclusions. To illustrate the concept of modeling, we consider several examples in this chapter. However, first, the standard optimization formulation is presented, together with an explanation of the terminology.

2.1 Standard Optimization Formulation

Almost all optimization problems considered in this text will be posed in the following standard form ("s.t." is short for "such that"):

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}}\ & f(\mathbf{x}) \\ \text{s.t.}\quad & h_i(\mathbf{x}) = 0;\ i = 1, 2, \ldots \\ & g_j(\mathbf{x}) \leq 0;\ j = 1, 2, \ldots \\ & \mathbf{x}^{\min} \leq \mathbf{x} \leq \mathbf{x}^{\max} \\ & \mathbf{x} = \{x_1 \quad x_2 \quad \ldots \quad x_N\} \end{aligned} \tag{2.1}$$

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

where

- $f(\mathbf{x})$ is the single *objective* that we are trying to minimize. Given a loosely worded problem statement, the first task is to identify and *quantify* the objective. In structural design problems, the objective is typically the volume or weight of a part, or the maximum deflection, or the maximum stress, and so on. Two points are worth noting here: (1) Often, an engineer may be interested in multiple objectives (for example, minimize weight and minimize stress); such *multi-objective* problems are briefly considered in Section 6.7, but are not the main focus of this text; further, it is often possible to interpret one or more of these objectives as constraints. (2) If we desire to *maximize* an objective (for example, maximize the stiffness of a part), it is easy to convert this into a minimization problem in multiple ways, as discussed later on.
- $\mathbf{x}$ are the *optimization or design variables*. These are the free parameters that can be modified to meet the objective. In structural design problems, these could be geometric parameters (thickness of a truss member, the location and size of a hole, topology), or material properties (Young's modulus, yield strength), and so on. In this text, we will assume that the optimization variables are *continuously varying* (such as the radius of a hole). Discrete variables, such as the number of holes in a design, are not explicitly treated in this text; however, references and examples are provided on how such integer problems can be handled.
- $\mathbf{x}^{\min}$ and $\mathbf{x}^{\max}$ are the lower and upper bounds on the optimization variables; for example, a lower bound and/or upper bound on a hole radius. It is not essential for optimization variables to exhibit lower and upper bounds. However, if such bounds exist, it is important to include them in the problem statement.
- $h_i(\mathbf{x})$ are the *equality constraints*; for example, "the diameter of truss member A must be exactly equal to three times the diameter of truss member B." While we are not distinguishing here between linear and non-linear constraints in the formulation, such differences can be important in numerical analysis. They are highlighted later on.
- $g_j(\mathbf{x})$ are the *inequality constraints*; for example, "the diameter of truss member A must be less than three times the diameter of truss member B." Again, we are not distinguishing here between linear and non-linear inequality constraints, but such differences can be important in numerical analysis.

In the remainder of this chapter, we shall study several optimization examples and convert them into the standard form shown in Equation (2.1). We will observe that there are several special cases of the standard form; for example, when the constraints are absent, we obtain an *unconstrained minimization* problem, which is mathematically and numerically easy to analyze:

$$\underset{\mathbf{x}}{\text{minimize}}\ f(\mathbf{x}) \tag{2.2}$$

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

where $\mathbf{x} = \{x_1 \quad x_2 \quad \dots \quad x_N\}$. Further, when there is precisely one optimization variable, this reduces to a *single-variable unconstrained minimization*, which is one of the simplest optimization problems that one can pose:

$$\underset{x}{\text{minimize}} \; f(x) \tag{2.3}$$

2.2 Illustrative Examples

Before moving to modeling, we consider here several examples of the standard form. Consider

$$\underset{x}{\text{minimize}} \; x^2 \tag{2.4}$$

This is a single-variable unconstrained minimization problem, with a trivial solution of $x = 0$. On the other hand, the problem

$$\underset{x}{\text{minimize}} \; 3\sin x - x + 0.1x^2 + 0.1\cos(2x) \tag{2.5}$$

is also a single-variable unconstrained minimization problem, but its solution will require numerical analysis. The problem

$$\underset{\{u,v\}}{\text{minimize}} \begin{Bmatrix} \frac{1}{2}100\left(\sqrt{u^2+(1+v)^2}-1\right)^2 + \frac{1}{2}50\left(\sqrt{u^2+(1-v)^2}-1\right)^2 \\ -(10u+8v) \end{Bmatrix} \tag{2.6}$$

is a two-variable unconstrained minimization problem. The physical interpretation of the above problem is discussed later in this chapter. One can add upper and lower bounds to the above problem, leading to

$$\begin{aligned} \underset{\{u,v\}}{\text{minimize}} \; & \begin{Bmatrix} \frac{1}{2}100\left(\sqrt{u^2+(1+v)^2}-1\right)^2 + \frac{1}{2}50\left(\sqrt{u^2+(1-v)^2}-1\right)^2 \\ -(10u+8v) \end{Bmatrix} \\ \text{s.t.} \quad & 0 \le u \le 1.0 \\ & 0 \le v \end{aligned} \tag{2.7}$$

One can also impose an equality constraint linking the two variables:

$$\begin{aligned} \underset{\{u,v\}}{\text{minimize}} \; & \begin{Bmatrix} \frac{1}{2}100\left(\sqrt{u^2+(1+v)^2}-1\right)^2 + \frac{1}{2}50\left(\sqrt{u^2+(1-v)^2}-1\right)^2 \\ -(10u+8v) \end{Bmatrix} \\ \text{s.t.} \quad & v - u^3 = 0 \end{aligned} \tag{2.8}$$

The physical meaning behind such constraints is discussed later in this chapter, and such constraints can completely change the complexity of the problem.

© in this web service Cambridge University Press www.cambridge.org

CAMBRIDGE
Cambridge University Press
978-1-108-49160-0 — Design Optimization using MATLAB and SOLIDWORKS
Krishnan Suresh
Excerpt
More Information

2.3 Geometry Problems

We now consider problems in geometry; the primary goal is to translate each of these problems into the standard form presented in Equation (2.1).

Example 2.1 Closest-Point Computation

Problem: Given a point p on a plane, and a straight line passing through the origin, find the closest point on the straight line to the given point p.

Modeling: Let the straight line passing through the origin be denoted by $y = mx$ and the given point be $p = (x_0, y_0)$. The task is to find the point q on the straight line that is closest to p; see Figure 2.1.
Let $q = (\alpha, m\alpha)$ be the point on the straight line. Observe that q is defined such that it always lies on the straight line $y = mx$. Introducing such intermediate variables that satisfy the problem constraints is one of the most important steps in modeling. Observe that the only unknown now is α.

Finding a point q closest to p is equivalent to minimizing the distance between them, where the distance is given by

$$D(\alpha) = \sqrt{(x_0 - \alpha)^2 + (y_0 - m\alpha)^2} \tag{2.9}$$

Thus, the optimization problem is posed as

$$\underset{\alpha}{\text{minimize}} \;\; D(\alpha) = \sqrt{(x_0 - \alpha)^2 + (y_0 - m\alpha)^2} \tag{2.10}$$

Equation (2.10) is interpreted as "Find α that minimizes $D(\alpha)$."

Classification: The problem posed in Equation (2.10) is *a single-objective, single-variable, unconstrained* optimization problem since: (1) there is only one objective namely, the distance D, that must be minimized, (2) there is only one continuous (unknown) variable α and (3) there are no constraints.

Solution: In the chapters that follow, we shall discuss various numerical methods to solve optimization problems. However, for now, recall from basic calculus that, if a continuous *differentiable* function takes a minimum at a point, then the derivative of

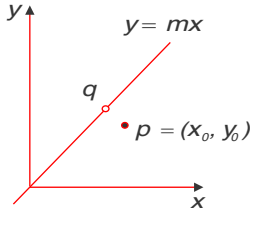

Figure 2.1 Closest point on a straight line.

© in this web service Cambridge University Press www.cambridge.org