Optimization Problem and Model Formulation

Introduction

In the previous lecture we studied the history of evolution of optimization methods and their engineering applications. A brief introduction was also given to the art of modeling. In this lecture we will study the Optimization problem, its various components and its formulation as a mathematical programming problem.

Basic components of an optimization problem:

An **objective function** expresses the main aim of the model which is either to be minimized or maximized. For example, in a manufacturing process, the aim may be to maximize the profit or minimize the cost. In comparing the data prescribed by a user-defined model with the observed data, the aim is minimizing the total deviation of the predictions based on the model from the observed data. In designing a bridge pier, the goal is to maximize the strength and minimize size.

A set of **unknowns** or **variables** control the value of the objective function. In the manufacturing problem, the variables may include the amounts of different resources used or the time spent on each activity. In fitting-the-data problem, the unknowns are the parameters of the model. In the pier design problem, the variables are the shape and dimensions of the pier.

A set of **constraints** are those which allow the unknowns to take on certain values but exclude others. In the manufacturing problem, one cannot spend negative amount of time on any activity, so one constraint is that the "time" variables are to be non-negative. In the pier design problem, one would probably want to limit the breadth of the base and to constrain its size.

The optimization problem is then to find values of the variables that minimize or maximize the objective function while satisfying the constraints.

Objective Function

As already stated, the objective function is the mathematical function one wants to maximize or minimize, subject to certain constraints. Many all optimization problems have a single
objective function. (When they don't they can often be reformulated so that they do) The two exceptions are:

- **No objective function.** In some cases (for example, design of integrated circuit layouts), the goal is to find a set of variables that satisfies the constraints of the model. The user does not particularly want to optimize anything and so there is no reason to define an objective function. This type of problems is usually called a *feasibility problem*.

- **Multiple objective functions.** In some cases, the user may like to optimize a number of different objectives concurrently. For instance, in the panel design problem, it would be nice to **minimize weight** and **maximize strength** simultaneously. Usually, the different objectives are not compatible; the variables that optimize one objective may be far from optimal for the others. In practice, problems with multiple objectives are reformulated as single-objective problems by either forming a weighted combination of the different objectives or by treating some of the objectives as constraints.

### Statement of an optimization problem

An optimization or a mathematical programming problem can be stated as follows:

To find \( \mathbf{X} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \) which minimizes \( f(\mathbf{X}) \) \hspace{1cm} (1.1)

Subject to the constraints

\[ g_i(\mathbf{X}) \leq 0, \quad i = 1, 2, \ldots, m \]

\[ l_j(\mathbf{X}) = 0, \quad j = 1, 2, \ldots, p \]

where \( \mathbf{X} \) is an \( n \)-dimensional vector called the design vector, \( f(\mathbf{X}) \) is called the *objective function*, and \( g_i(\mathbf{X}) \) and \( l_j(\mathbf{X}) \) are known as inequality and equality constraints, respectively. The number of variables \( n \) and the number of constraints \( m \) and/or \( p \) need not be related in any way. This type problem is called a *constrained optimization problem*.

If the locus of all points satisfying \( f(\mathbf{X}) = \) a constant \( c \), is considered, it can form a family of surfaces in the design space called the *objective function surfaces*. When drawn with the
constraint surfaces as shown in Fig 1 we can identify the optimum point (maxima). This is possible graphically only when the number of design variables is two. When we have three or more design variables because of complexity in the objective function surface, we have to solve the problem as a mathematical problem and this visualization is not possible.

![Diagram showing optimization](image)

**Fig 1**

Optimization problems can be defined without any constraints as well.

To find $X = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ which minimizes $f(X)$

\begin{equation}
C_1 > C_2 > C_3 > C_4 \ldots > C_n
\end{equation}

Such problems are called *unconstrained optimization problems*. The field of unconstrained optimization is quite a large and prominent one, for which a lot of algorithms and software are available.
Constraints

Constraints are not essential. It's been argued that almost all problems really do have constraints. For example, any variable denoting the "number of objects" in a system can only be useful if it is less than the number of elementary particles in the known universe! In practice though, answers that make good sense in terms of the underlying physical or economic criteria can often be obtained without putting constraints on the variables.

Variables

These are essential. If there are no variables, we cannot define the objective function and the problem constraints.

In many practical problems, one cannot choose the design variable arbitrarily. They have to satisfy certain specified functional and other requirements. Design constraints are restrictions that must be satisfied to produce an acceptable design.

Constraints can be broadly classified as:

1) Behavioral or Functional constraints: These represent limitations on the behavior performance of the system.

2) Geometric or Side constraints: These represent physical limitations on design variables such as availability, fabricability, and transportability.

For example, for the retaining wall design shown in the Fig 2, the base width W cannot be taken smaller than a certain value due to stability requirements. The depth D below the ground level depends on the soil pressure coefficients $K_a$ and $K_p$. Since these constraints depend on the performance of the retaining wall they are called behavioral constraints. The number of anchors provided along a cross section $N_i$ cannot be any real number but has to be a whole number. Similarly thickness of reinforcement used is controlled by supplies from the manufacturer. Hence this is a side constraint.
Consider the optimization problem presented in eq. 1.1 with only the inequality constraint $g_i(X) \leq 0$. The set of values of $X$ that satisfy the equation $g_i(X) \leq 0$ forms a boundary surface in the design space called a constraint surface. This will be a $(n-1)$ dimensional subspace where $n$ is the number of design variables. The constraint surface divides the design space into two regions: one with $g_i(X) < 0$ (feasible region) and the other in which $g_i(X) > 0$ (infeasible region). The points lying on the hyper surface will satisfy $g_i(X) = 0$. The collection of all the constraint surfaces $g_i(X) = 0$, $j = 1, 2, \ldots, m$, which separates the acceptable region is called the composite constraint surface.

Fig 3 shows a hypothetical two-dimensional design space where the feasible region is denoted by hatched lines. The two-dimensional design space is bounded by straight lines as shown in the figure. This is the case when the constraints are linear. However, constraints may be nonlinear as well and the design space will be bounded by curves in that case. A design point that lies on more than one constraint surface is called a bound point, and the associated constraint is called an active constraint. Free points are those that do not lie on any constraint surface. The design points that lie in the acceptable or unacceptable regions can be classified as following:

1. Free and acceptable point
2. Free and unacceptable point
3. Bound and acceptable point
4. Bound and unacceptable point.

Examples of each case are shown in Fig. 3.

**Formulation of design problems as mathematical programming problems**

In mathematics, the term **optimization**, or **mathematical programming**, refers to the study of problems in which one seeks to minimize or maximize a real function by systematically choosing the values of real or integer variables from within an allowed set. This problem can be represented in the following way

*Given:* a function \( f : A \rightarrow \mathbb{R} \) from some set \( A \) to the real numbers

*Sought:* an element \( x_0 \) in \( A \) such that \( f(x_0) \leq f(x) \) for all \( x \) in \( A \) ("minimization") or such that \( f(x_0) \geq f(x) \) for all \( x \) in \( A \) ("maximization").

Such a formulation is called an optimization problem or a **mathematical programming problem** (a term not directly related to computer programming, but still in use for example...*)
Many real-world and theoretical problems may be modeled in this general framework. Typically, \( A \) is some subset of the Euclidean space \( \mathbb{R}^n \), often specified by a set of constraints, equalities or inequalities that the members of \( A \) have to satisfy. The elements of \( A \) are called candidate solutions or feasible solutions. The function \( f \) is called an objective function, or cost function. A feasible solution that minimizes (or maximizes, if that is the goal) the objective function is called an optimal solution. The domain \( A \) of \( f \) is called the search space.

Generally, when the feasible region or the objective function of the problem does not present convexity (refer module 2), there may be several local minima and maxima, where a local minimum \( x^* \) is defined as a point for which there exists some \( \delta > 0 \) so that for all \( x \) such that
\[
\| x - x^* \| \leq \delta,
\]
and
\[
f(x^*) \leq f(x)
\]
that is to say, on some region around \( x^* \) all the function values are greater than or equal to the value at that point. Local maxima are defined similarly.

A large number of algorithms proposed for solving non-convex problems – including the majority of commercially available solvers – are not capable of making a distinction between local optimal solutions and rigorous optimal solutions, and will treat the former as the actual solutions to the original problem. The branch of applied mathematics and numerical analysis that is concerned with the development of deterministic algorithms that are capable of guaranteeing convergence in finite time to the actual optimal solution of a non-convex problem is called global optimization.

**Problem formulation**

Problem formulation is normally the most difficult part of the process. It is the selection of design variables, constraints, objective function(s), and models of the discipline/design.

**Selection of design variables**

A design variable, that takes a numeric or binary value, is controllable from the point of view of the designer. For instance, the thickness of a structural member can be considered a design variable. Design variables can be continuous (such as the length of a cantilever beam),
discrete (such as the number of reinforcement bars used in a beam), or Boolean. Design problems with continuous variables are normally solved more easily.

Design variables are often bounded, that is, they have maximum and minimum values. Depending on the adopted method, these bounds can be treated as constraints or separately.

**Selection of constraints**

A constraint is a condition that must be satisfied to render the design to be feasible. An example of a constraint in beam design is that the resistance offered by the beam at points of loading must be equal to or greater than the weight of structural member and the load supported. In addition to physical laws, constraints can reflect resource limitations, user requirements, or bounds on the validity of the analysis models. Constraints can be used explicitly by the solution algorithm or can be incorporated into the objective, by using Lagrange multipliers.

**Objectives**

An objective is a numerical value that is to be maximized or minimized. For example, a designer may wish to maximize profit or minimize weight. Many solution methods work only with single objectives. When using these methods, the designer normally weights the various objectives and sums them to form a single objective. Other methods allow multi-objective optimization (module 8), such as the calculation of a Pareto front.

**Models**

The designer has to also choose models to relate the constraints and the objectives to the design variables. These models are dependent on the discipline involved. They may be empirical models, such as a regression analysis of aircraft prices, theoretical models, such as from computational fluid dynamics, or reduced-order models of either of these. In choosing the models the designer must trade off fidelity with the time required for analysis.

The multidisciplinary nature of most design problems complicates model choice and implementation. Often several iterations are necessary between the disciplines’ analyses in order to find the values of the objectives and constraints. As an example, the aerodynamic loads on a bridge affect the structural deformation of the supporting structure. The structural deformation in turn changes the shape of the bridge and hence the aerodynamic loads. Therefore, in analyzing a bridge, the aerodynamic and structural analyses must be run a number of times in turn until the loads and deformation converge.
Representation in standard form

Once the design variables, constraints, objectives, and the relationships between them have been chosen, the problem can be expressed as shown in equation 1.1

Maximization problems can be converted to minimization problems by multiplying the objective by -1. Constraints can be reversed in a similar manner. Equality constraints can be replaced by two inequality constraints.

Problem solution

The problem is normally solved choosing the appropriate techniques from those available in the field of optimization. These include gradient-based algorithms, population-based algorithms, or others. Very simple problems can sometimes be expressed linearly; in that case the techniques of linear programming are applicable.

Gradient-based methods

- Newton's method
- Steepest descent
- Conjugate gradient
- Sequential quadratic programming

Population-based methods

- Genetic algorithms
- Particle swarm optimization

Other methods

- Random search
- Grid search
- Simulated annealing

Most of these techniques require large number of evaluations of the objectives and the constraints. The disciplinary models are often very complex and can take significant amount of time for a single evaluation. The solution can therefore be extremely time-consuming. Many of the optimization techniques are adaptable to parallel computing. Much of the current research is focused on methods of decreasing the computation time.
Optimization Methods: Introduction and Basic concepts

The following steps summarize the general procedure used to formulate and solve optimization problems. Some problems may not require that the engineer follow the steps in the exact order, but each of the steps should be considered in the process.

1) Analyze the process itself to identify the process variables and specific characteristics of interest, i.e., make a list of all the variables.

2) Determine the criterion for optimization and specify the objective function in terms of the above variables together with coefficients.

3) Develop via mathematical expressions a valid process model that relates the input-output variables of the process and associated coefficients. Include both equality and inequality constraints. Use well known physical principles such as mass balances, energy balance, empirical relations, implicit concepts and external restrictions. Identify the independent and dependent variables to get the number of degrees of freedom.

4) If the problem formulation is too large in scope:
   - break it up into manageable parts, or
   - simplify the objective function and the model

5) Apply a suitable optimization technique for mathematical statement of the problem.

6) Examine the sensitivity of the result, to changes in the values of the parameters in the problem and the assumptions.
Basic Problem Statement

Minimize \( f(x_1, \ldots, x_n) \)  

Subject to  
\( g_1(x_1, \ldots, x_n) \leq 0 \)  
\( \quad \vdots \)  
\( g_m(x_1, \ldots, x_n) = 0 \)

Alternatively, one may want to maximize a function subject to constraints.
Types of Optimization Problems

If the objective function and all constraint functions are linear, we have a Linear Programming (LP) problem.

An LP in which the variables can only take integer values is an Integer Programming Problem.

A LP in which some of the variables are restricted to be integers is called a mixed Integer Programming Problem.

If the objective function and/or constraint are non-linear we have a non-linear programming problem (NLP).
In mathematics and computer science, an **optimization problem** is the problem of finding the best solution from all feasible solutions. Optimization problems can be divided into two categories depending on whether the variables are continuous or discrete. An optimization problem with discrete variables is known as a **combinatorial optimization problem**. In a combinatorial optimization problem, we are looking for an object such as an integer, permutation or graph from a finite (or possibly countable infinite) set. Problems with continuous variables include constrained problems and multimodal problems.

---

**Contents** [hide]

1 Continuous optimization problem
2 Combinatorial optimization problem
   2.1 NP optimization problem
3 References
4 See also

---

**Continuous optimization problem** [ edit ]

The **standard form** of a (continuous) optimization problem is\(^1\)

\[
\begin{align*}
\text{minimize} \quad & f(x) \\
\text{subject to} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m \\
& h_i(x) = 0, \quad i = 1, \ldots, p
\end{align*}
\]

where

- \(f(x) : \mathbb{R}^n \to \mathbb{R}\) is the **objective function** to be minimized over the variable \(x\),
- \(g_i(x) \leq 0\) are called **inequality constraints**, and
- \(h_i(x) = 0\) are called **equality constraints**.

By convention, the standard form defines a **minimization problem**. A **maximization problem** can be treated by negating the objective function.
Convex sets and functions

Conv. set

Set $c$ is convex if $\forall x, y \in c$

$\lambda x + (1-\lambda)y \in c \quad 0 < \lambda < 1$

Non-conv. set
Convex function

Convex

Non-convex
**Convex function**

Definition: A function with convex domain is convex if for all \( x, y \)

\[ f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \quad 0 < \alpha < 1 \]

strictly convex - when inequality is strict
l0-Norm, l1-Norm, l2-Norm, ...

I'm working on things related to norm a lot lately and it is time to talk about it. In this post we are going to discuss about a whole family of norm.

What is a norm?

Mathematically a norm is a total size or length of all vectors in a vector space or matrices. For simplicity, we can say that the higher the norm is, the bigger the (value in) matrix or vector is. Norm may come in many forms and many names, including these popular name: Euclidean distance, Mean-squared Error, etc.

Most of the time you will see the norm appears in a equation like this:

\[ \|x\| \text{ where } x \text{ can be a vector or a matrix.} \]

For example, a Euclidean norm of a vector \( a = \begin{bmatrix} 3 \\ -2 \\ 1 \end{bmatrix} \) is

\[ \|a\|_2 = \sqrt{3^2 + (-2)^2 + 1^2} = 3.742 \text{ which is the size of vector } a \]

The above example shows how to compute a Euclidean norm, or formally called an \( l_2 \)-norm. There are many other types of norm that beyond our explanation here, actually for every single real number, there is a norm correspond to it (Notice the emphasised word real number, that means it not limited to only integer.)

Formally the \( l_p \)-norm of \( x \) is defined as:

\[ \|x\|_p = \sqrt[p]{\sum |x_i|^p} \text{ where } p \in \mathbb{R} \]

That’s it! A \( p \)-th-root of a summation of all elements to the \( p \)-th power is what we call a norm.

The interesting point is even though every \( l_p \)-norm is all look very similar to each other, their mathematical properties are very different and thus their application are dramatically different too. Hereby we are going to look into some of these norms in details.
Definition

$x \in C$ is a local min of $f$ over set $C$ if $\exists \varepsilon > 0$: $f(x) \leq f(y)$ for all $y \in B_\varepsilon(x) \cap C$,

where $B_\varepsilon(x)$ - $\varepsilon$-ball around $x$:

$$B_\varepsilon(x) = \{ z : \| z - x \| \leq \varepsilon \}$$

Local & Global minima

Let $C \subseteq \mathbb{R}^n$; $f: C \to \mathbb{R}$

Definition

$x \in C$ is a global min of $f$ over set $C$ if $f(x) \leq f(y)$ for all $y \in C$. 

Global min

Local min