CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

1 Optimization in Process Systems Engineering

1.1 Introduction

This book deals with mathematical programming techniques for process systems engineering (PSE) (Sargent, 2004). The main focus is on the theory, model formulation, and algorithms of mathematical optimization techniques. In this chapter, we provide a general overview of how optimization problems arise in process systems engineering (Biegler et al., 1997). A number of interesting and practical applications are increasingly being tackled in the chemical industry through modern optimization models and methods. The major areas that have been considered include the following:

– process design and synthesis
– planning and scheduling
– process control and operation.

Table 1.1 gives specific areas of application of optimization in process systems engineering (Biegler and Grossmann, 2004; Grossmann and Biegler, 2004). Major optimization techniques include linear programming (LP), nonlinear programming (NLP), mixed-integer linear programming (MILP), mixed-integer nonlinear programming (MINLP), and quadratic programming (QP) (Minoux, 1986).

Figure 1.1 provides the major steps that are involved in the formulation of optimization problems listed in Table 1.1. Given is either a process flowsheet for producing a particular chemical, or a production plan for manufacturing several different products. If we fix the configuration and equipment sizes of a process flowsheet, or the timing of the operations for the production plan, we resort to an analysis of the flowsheet or production plan in order to evaluate the economics (e.g., operating cost), or its performance (e.g., completion time), or its operability (e.g., flexibility or controllability). In order to improve the economics, performance, or operability, we can select new values of parameters that represent degrees of freedom (e.g., pressures, temperatures, reactor volumes, recycle ratios, production rates, etc.). If our aim is to optimize the value of these parameters, mathematically this corresponds to an optimization with continuous variables. We can go a step further and modify the structure of the flowsheet or of the production plan, which can be interpreted as a synthesis step since it gives rise to a new flowsheet structure or to a new production plan in terms of production sequence. If our aim is to optimize these decisions, mathematically this corresponds to an optimization problem with discrete variables.

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

Table 1.1 Applications of mathematical programming in process systems engineering

	MILP	MINLP	LP	QP	NLP
Design and synthesis					
HENS	x	x	x		x
MENS	x	x	x		x
Separations	x	x			
Reactors		x	x		x
Equipment design		x			x
Flowsheeting		x			x
Operations					
Scheduling	x	x	x		
Supply chain	x	x	x		
Real-time optimization			x	x	x
Linear MPC			x	x	
Nonlinear MPC					x
Hybrid	x				x

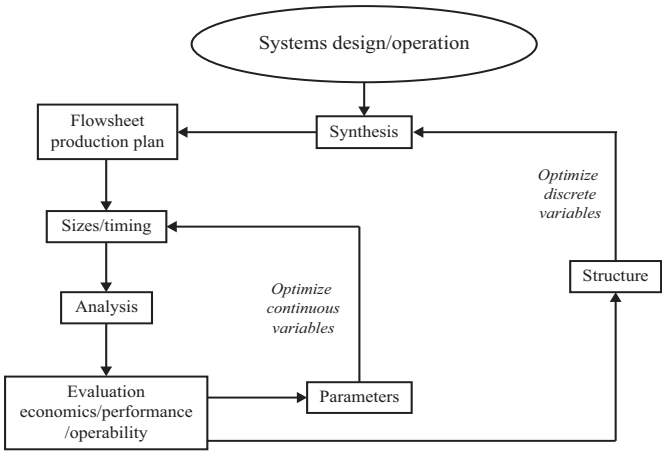

Figure 1.1 Major steps in the formulation of optimization problems in PSE.

In summary, from Fig. 1.1 we can conclude that the major steps involved in the design/operation of a process flowsheet or production plan are:

(a) analysis
(b) evaluation
(c) synthesis.

The important point to realize is that the improved design/operation of a flowsheet or production plan can be achieved through the optimization of continuous and discrete

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

variables, which is the main topic of this textbook. Therefore, the outline of this textbook is as follows.

Chapter 2 Solving Nonlinear Equations
Chapter 3 Basic Theoretical Concepts in Optimization
Chapter 4 Nonlinear Programming Algorithms
Chapter 5 Linear Programming
Chapter 6 Mixed-Integer Programming Models
Chapter 7 Systematic Modeling of Constraints with Logic
Chapter 8 Mixed-Integer Linear Programming
Chapter 9 Mixed-Integer Nonlinear Programming
Chapter 10 Generalized Disjunctive Programming
Chapter 11 Constraint Programming
Chapter 12 Nonconvex Optimization
Chapter 13 Lagrangean Decomposition
Chapter 14 Stochastic Programming
Chapter 15 Flexibility Analysis
Appendix A Modeling Systems and Optimization Software
Appendix B Optimization Models for Process Systems Engineering

1.2 Classification of Optimization Models

Optimization problems can generally be formulated with mathematical models that involve continuous and discrete variables (Grossmann, 2002). These variables must be selected to satisfy equations and inequality constraints, while optimizing a given objective function as expressed with the model below:

$$\begin{aligned} &\min\ f(x,y) \\ &\text{s.t.}\ \ h(x,y)=0 \\ &\qquad g(x,y)\le 0 \\ &\qquad x\in X, y\in Y. \end{aligned} \tag{1.1}$$

The continuous variables are represented by x and the discrete variables by y, both with arbitrary dimensionality. The feasible region for the variables x and y is defined by the following constraints: $h(x,y)=0$, which are equations describing the performance of a system (e.g., mass, energy balances, design equations); and $g(x,y)\le 0$, which are inequalities that are related to specifications (e.g., minimum purity, maximum throughput). In addition, the variables can be further confined to sets X (typically lower and upper bounds) and Y (specific integers). The former typically specifies the ranges of values for the continuous variables (e.g., positive flows, minimum/maximum pressures for a chemical reactor), while the latter specifies the discrete choices (e.g., only 0-1 choices for the selection or not of

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

a processing unit, or an integer number for number of plates in a distillation column). It should be noted that the model in (1.1) can in practice also be posed as a maximization problem, or with inequalities that must be greater than or equal to zero. The form in (1.1), however, is used for presentation purposes. Also, any problem can be converted into that form because maximizing a function is equivalent to minimizing its negative value. Inequalities that are greater than or equal to zero can also easily be converted to less than or equal to zero.

The mathematical model in (1.1) is technically known as a "mathematical program" and an extensive body of literature exists on this subject (see Williams, 1985, for an introduction to models; and Minoux, 1986, for a general review of methods). From a practical standpoint, the important feature of model (1.1) is that it provides a powerful framework for modeling many optimization problems, including design, scheduling, and control problems. More recently, machine learning models are also being solved through formal optimization methods (Bottou et al., 2018). Depending on the application at hand and the level of detail in the equations, the objective function and constraints can be given in explicit or implicit form. Explicit equations are given by mathematical expressions that are often used for simplified models. In this case, the time to perform an evaluation of the model may be very fast even though it may involve thousands of variables and constraints. Implicit equations arise when detailed and complex calculations must be performed as procedures, as with, for instance, process simulators or differential equation models. In these cases, the dimensionality of the optimization model can be greatly reduced, although the time for performing an evaluation of one trial point may require a significant length of time.

Formulating a given decision problem as a mathematical program in (1.1) requires three major assumptions: (a) the criterion for optimal selection can be expressed through a single objective function, (b) the constraints must be exactly satisfied, and (c) the parameters (input data) are deterministic in nature. These, of course, represent oversimplifications of real-world problems. However, it should be noted that extensions of the model in (1.1) are available for addressing some of these issues. For instance, it is possible to relax the assumptions in (a) and (b) through multiobjective optimization methods (Audet et al., 2008), while the assumption in (c) can be addressed through stochastic optimization methods (Sahinidis, 2004) or multiparametric optimization methods (Pistikopoulos et al., 2007).

If one assumes no special structure to the problem in (1.1) and to its various particular cases (e.g., only discrete or only continuous variables), then direct search techniques are often the easiest methods to apply but also the most time consuming (Rios and Sahinidis, 2013). Here, either a systematic or random selection of trial points is chosen to evaluate and improve the objective function. Satisfaction of constraints can also be accomplished, but frequently with some difficulty (e.g., using penalty functions in which a weighted violation of constraints is added to the original objective function). Perhaps the most popular direct search method that has emerged recently in process systems engineering is simulated annealing. This method is based on analogies with free-energy minimization in statistical mechanics. This method is, in principle, easy to apply to problems with simple constraints, and is likely to find solutions that are close to the global optimum. However, aside from the

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

fact that it often requires many thousands of function evaluations before the likely optimum is found, its performance tends to be highly dependent on the selection of the parameters of the algorithm.

On the other hand, the most prevalent approach that has been taken in optimization is to consider particular problem classes of (1.1) depending on the form of the objective function and constraints with which efficient solution methods can be derived to exploit special structures.

The particular case of (1.1) that is best-known is the linear programming (LP) problem, in which the objective function and all the constraints are linear, and all the variables are continuous (Saigal, 1995; Vanderbei, 2020). As will be discussed in Chapter 5, LP problems have the property that the optimum solution lies at a vertex of the feasible space. Also, any local optimal solution corresponds to the global optimum. These problems have been successfully solved for many years with computer codes that are based on the Simplex algorithm, which is rooted in linear algebra methods. Major changes that have taken place over the past ten years at the level of solution methods are the development of interior-point algorithms that rely on nonlinear transformations and whose computational requirements are theoretically bounded by a polynomial expressed in terms of the problem size (Karmarkar, 1984; Marsten et al., 1990). Interestingly, this property is not shared by the Simplex algorithm, which theoretically may require exponential time. Since this performance is rarely observed in practice, further significant advances have taken place for solving large-scale problems with the Simplex algorithm. With this algorithm, problems with up to 50 000–100 000 constraints can be solved quite efficiently, while interior-point methods tend to perform better in problems with up to 500 000–1000 000 constraints. It should also be pointed out that the mathematical structure of very specialized cases of LP problems (e.g., network flows in assignment or transportation problems) has been greatly exploited, which has made possible the development of codes that can be applied to problems involving millions of variables (Bertsekas, 1998).

The extension of the LP model that involves discrete variables is known as a mixed-integer linear program (MILP) (Nemhauser and Wolsey, 1988; Wolsey, 1998). As discussed in Chapter 6, this model greatly expands the capabilities of formulating real-world problems since one can include logical decisions with 0-1 variables, or account for discrete amounts. As discussed in Chapter 8, the most common method for MILP is the branch and bound search, which consists of solving a subset of LP subproblems while searching within a decision tree of the discrete variables. The other common approach relies on the use of cutting planes that attempt to make the MILP solvable as an LP with the addition of special constraints. Owing to the combinatorial nature that is introduced by the discrete variables in MILP problems, they have proven to be very hard to solve. In fact, theoretically, one can show that this class of problems are NP-hard; that is, there is no known algorithm whose computational requirements do not exceed a polynomial increase in terms of problem size. Nevertheless, recent advances that are based on combining branch and bound methods with cutting planes, and which have been coupled with advances in LP technology, are providing rigorous optimal solutions to problems that were regarded as unsolvable ten years ago.

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

For the case when all or at least some of the functions are nonlinear, and only continuous variables are involved, problem (1.1) gives rise to nonlinear programming problems (NLP) (Bazaraa et al., 2006; Biegler, 2010). For the case when the objective and constraint functions are differentiable, local optima can be defined by optimality conditions known as the Karush–Kuhn–Tucker (KKT) conditions, as discussed in Chapter 3. These are perhaps the most common type of models that arise in process systems engineering. While ten years ago, problems involving 1000 variables for NLP were regarded as being large, nowadays the solution of problems with up to one million variables is quite common. Reduced-gradient, successive-quadratic programming and interior-point methods, which can be derived by applying Newton's method to the Karush–Kuhn–Tucker conditions, have emerged as the major algorithms for NLP, as will be discussed in Chapter 4. The reduced-gradient method is, in general, better suited for problems with mostly linear constraints, successive-quadratic programming tends to be the method of choice for highly nonlinear problems, while the interior-point method is best suited for very large-scale NLP problems. A limitation of these methods is that they are only guaranteed to converge to a local optimum. For problems that involve a convex objective function and a convex feasible region, this is not a difficulty since these exhibit only one local optimum, which therefore corresponds to the global optimum. In practice, proving convexity in a nonlinear problem is often not possible, and therefore finding any local optimum is often regarded as a satisfactory solution, especially if this yields a significant improvement. On the other hand, there are applications in which finding the global optimum to nonconvex problems is a major issue. Over the past few years, significant progress has been made in developing rigorous methods for globally optimizing problems with special structures (e.g., bilinear functions), as will be discussed in Chapter 12 (Horst and Tuy, 1996; Tawarmalani and Sahinidis, 2004).

The extension of nonlinear programming for handling discrete variables yields a mixed-integer nonlinear programming (MINLP) problem that in its general form is identical to problem (1.1) (Grossmann, 2002; Trespalacios and Grossmann, 2014). MINLP problems were regarded as essentially unsolvable twenty years ago. Algorithms such as the outer-approximation method and extensions of the generalized Benders decomposition method, which are discussed in Chapter 9, have emerged as major methods. These methods assume differentiability in the functions, and consist of solving an alternating sequence of NLP subproblems and MILP master problems. The former optimizes the continuous variables, while the latter optimizes the discrete variables. As in the NLP case, global-optimum solutions can be guaranteed only for convex problems. Solution of problems with typically up to 1000 binary variables and 100 000 continuous variables and constraints has been reported with these methods. Major difficulties encountered in MINLP include those encountered in MILP (combinatorial nature requiring large computations) and in NLP (nonconvexities yielding local solutions).

It should also be noted that all the above methods assume that the problem in (1.1) is expressed through algebraic equations. Very often, however, these models may involve differential equations as constraints. This gives rise to problems known as optimal control problems or the optimization of differential algebraic systems of equations. The major

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

approach that has emerged here is to approximate the differential equations by algebraic equations that yield nonlinear programming problems (Biegler, 2010). The alternate approach is to solve the differential model in a routine that is then treated by the optimizer as a procedure (or implicit function) (Vassiliadis et al., 1994).

Finally, many optimization problems require anticipating the effect of uncertain parameters (Sahinidis, 2004). The most general form of this problem is known as stochastic programming (Birge and Louveaux, 2011), discussed in Chapter 14, in which the basic idea is to optimize over a set of scenarios that arise from discretizing probability distribution functions. Since these problems can give rise to very large-scale problems they are usually solved with decomposition methods. In particular, Benders decomposition is usually used for two-stage stochastic programming models (stage 1: here and now decisions; stage 2: wait and see decisions), while Lagrangean decomposition, which is covered in Chapter 13, is used for multistage stochastic programs by dualizing the so-called "nonanticipativity" constraints. Robust optimization is the other major approach for optimization under uncertainty (Bertsimas et al., 2011), but it is covered only briefly in Chapter 14 as a semi-infinite programming problem. Finally, Chapter 15 examines flexibility analysis, which deals with the problem of determining feasibility in the space of uncertain parameters, while accounting for the fact that corrective action can be taken to regain feasibility.

1.3 Outline of the Book

Having given a general overview of optimization in this chapter, we now provide a general outline of the book to explain the rationale of the order of the chapters, which in part is somewhat nonconventional.

First, we cover Newton's method for nonlinear equation solving in Chapter 2, as it is the basis of deriving the different nonlinear programming algorithms. We also discuss in that chapter quasi-Newton methods that approximate the Jacobian matrix. In Chapter 3 we provide basic concepts on convexity and necessary and sufficient conditions for optimality for unconstrained, equality constrained, and both equality and inequality constrained problems, which give rise to the Karush–Kuhn–Tucker (KKT) conditions. In Chapter 4, we discuss nonlinear programming algorithms to emphasize the fact that NLP algorithms represent the application of Newton's method to the Karush–Kuhn–Tucker conditions. In Chapter 5, we provide some theoretical properties of LP models, and use them as a basis for deriving the Simplex algorithms for which we view it as a special case of nonlinear programming by making connections with the Karush–Kuhn–Tucker conditions and the reduced-gradient method covered in Chapter 4.

Second, we introduce, in Chapter 6, modeling of mixed-integer linear programming problems, stressing the importance of linear 0-1 inequalities. We also introduce several classical combinatorial problems: assignment, facility location, knapsack, set covering, and traveling salesman. Next, in Chapter 7, we introduce the use of propositional logic as a higher-level modeling framework from which linear 0-1 inequalities can be systematically

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

derived. We also introduce disjunctions on continuous constraints, and its two major reformulations as mixed-integer linear constraints big-M and hull reformulations.

Third, we cover in Chapter 8, the major methods for solving mixed-integer linear programs: branch and bound, and cutting planes. In Chapter 9 we describe methods for solving mixed-integer nonlinear programming models, namely branch and bound, outer-approximation (OA) and generalized Benders decomposition (GDB). Although GBD was developed before outer approximation, we show it is easier to derive GDB from OA, and hence the proposed order.

Fourth, we introduce, in Chapter 10, the concept of generalized disjunctive programming, as a high-level representation of optimization problems in terms of continuous and Boolean variables with algebraic, logic, and disjunctive constraints. Aside from showing that the GDP can be used as a modeling framework to derive algebraic mixed-integer linear and nonlinear problems, we also briefly describe disjunctive branch and bound and logic-based outer-approximation algorithms.

Finally, the last five chapters deal with more specialized topics. Chapter 11 describes constraint programming, an alternative logic-based optimization modeling and solution approach based on concepts of logic inference. Chapter 12 discusses global optimization of nonconvex NLP and MINLP models, stressing the use of convex envelopes within spatial branch and bound methods. Chapter 13 describes Lagrangean relaxation and Lagrangean decomposition methods, Chapter 14 describes methods aimed at solving mixed-integer linear stochastic programming models, and Chapter 15 describes formulations and solution methods for flexibility analysis which are relevant for establishing feasibility of constraints under uncertainty with recourse actions.

Appendix A provides information and links to modeling systems such as GAMS, AIMMS, AMPL, Pyomo, and Julia, and to optimization software for LP, NLP, MILP, MINLP, and GDP problems. Appendix B provides links to libraries of problems that contain many application problems in process systems engineering.

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

2 Solving Nonlinear Equations

2.1 Process Modeling Approaches

In order to perform the analysis of a process system there are two major approaches: equation oriented and sequential modular (Westerberg et al., 1979). Examples of computer software implementing these approaches are gPROMS for the former, and Aspen Plus for the latter.

The basic idea in the equation-oriented approach is to define explicitly all the variables x and assemble all the corresponding equations describing a process or system. Mathematically, this is represented by the system of equations

$$f(x) = 0, \tag{2.1}$$

where f is the vector of equations and x is the vector of variables. Assuming that the equations are independent, the dimensionality of both is the same, i.e. $\dim(f) = \dim(x)$, meaning there are no degrees of freedom. In contrast, in the sequential-modular approach the equations are implemented in a set of modules like the ones presented in Fig. 2.1.

Given an input to module 1 and a guess of the recycle variables x, modules 1, 2, and 3 are executed in sequence so that, given their inputs, the outputs are computed. The execution of module 3 yields the calculated value in the recycle stream, which is denoted by the implicit function $g(x)$. In order to converge to the solution, it is required that the following equation holds true:

$$x = g(x). \tag{2.2}$$

This equation is also known as the fixed-point problem. The existence of the solution to (2.2) is established through the contraction mapping theorem (Ortega and Rheinboldt, 1970). It should be noted that (2.2) can be rearranged in the form of (2.1) by defining $f(x)$ as follows:

$$f(x) = x - g(x). \tag{2.3}$$

Thus, we will consider the solution of problem (2.1) as it represents both modeling approaches. We should note that usually the existence of the solution of (2.1) is established on physical grounds.

2.2 Newton's Method

A general solution method for solving the system of equations in (2.1) is Newton's method (Ortega and Rheinboldt, 1970; Sargent, 1975). Below we present a brief derivation of this method.

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

CAMBRIDGE
Cambridge University Press & Assessment
978-1-108-83165-9 — Advanced Optimization for Process Systems Engineering
Ignacio E. Grossmann
Excerpt
More Information

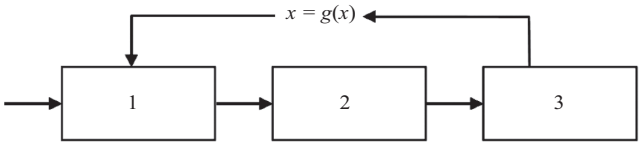

Figure 2.1 Example of execution of blocks in the sequential-modular approach.

Consider the linear expansion of $f(x)$ at x^k

$$f(x^k + p^k) = f(x^k) + J^k p^k, \tag{2.4}$$

where the computed step p^k is given by

$$p^k = x^{k+1} - x^k \tag{2.5}$$

and the Jacobian matrix $J^k = J^k(x^k)$, by the matrix of partial derivatives, is

$$J^k = \left(\frac{\partial f}{\partial x}\right)_{x^k} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \\ \frac{\partial f_n}{\partial x_1} & \cdots & & \frac{\partial f_n}{\partial x_n} \end{bmatrix}. \tag{2.6}$$

Notice that each row of J^k represents the gradient of the function $f_i(x)$, i.e. the ith element of the vector $f(x)$. Assuming the solution at $f(x^k + p^k) = 0$, this gives rise from (2.4) the following system of linear equations,

$$J^k p^k = -f(x^k), \tag{2.7}$$

which in turn yields the predicted step

$$p^k = -(J^k)^{-1} f(x^k), \tag{2.8}$$

and hence the new predicted points x^{k+1} are given by

$$x^{k+1} = x^k - (J^k)^{-1} f(x^k) \quad k = 0, 1, 2, \ldots, \tag{2.9}$$

where x^0 is the initial guess. Note that, because the inverse of the Jacobian $(J^k)^{-1}$ destroys the sparsity of the Jacobian, it is preferable to solve the linear equations in (2.7) in the computation of the step p^k. Hence, the main steps of Newton's algorithm are as follows.

Newton's Algorithm

(1) Guess x^0, set $k = 0$, and select tolerances $\varepsilon_1, \varepsilon_2$.
(2) Calculate $f(x^k), J^k = J^k(x^k)$.
(3) Solve linear equations $J^k p^k = -f(x^k)$ (sparse matrix).

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