Chapter 1

Introduction

There is scarcely a modern journal, whether of engineering, economics, management, mathematics, physics or the social sciences, in which the concept *optimization* is missing from the subject index. If one abstracts from all specialist points of view, the recurring problem is to select a better or best (Leibniz, 1710; eventually, he introduced the term *optimal*) alternative from among a number of possible states of affairs. However, if one were to follow the hypothesis of Leibniz, as presented in his *Theodicee*, that our world is the best of all possible worlds, one could justifiably sink into passive fatalism. There would be nothing to improve or to optimize.

Biology, especially since Darwin, has replaced the static world picture of Leibniz' time by a dynamic one, that of the more or less gradual development of the species culminating in the appearance of man. Paleontology is providing an increasingly complete picture of *organic evolution*. So-called *missing links* repeatedly turn out to be not missing, but rather hitherto undiscovered stages of this process. Very much older than the recognition that man is the result (or better, intermediate state) of a *meliorization* process is the seldom-questioned assumption that he is a perfect end product, the "pinnacle of creation." Furthermore, long before man conceived of himself as an active participant in the development of things, he had unconsciously influenced this evolution. There can be no doubt that his ability and efforts to make the environment meet his needs raised him above other forms of life and have enabled him, despite physical inferiority, to find, to hold, and to extend his place in the world–so far at least. As long as mankind has existed on our planet, spaceship earth, we, together with other species have mutually influenced and changed our environment. Has this always been done in the sense of meliorization?

In 1759, the French philosopher Voltaire (1759), dissatisfied with the conditions of his age, was already taking up arms against Leibniz' philosophical *optimism* and calling for conscious effort to change the state of affairs. In the same way today, when we optimize we find that we are both the subject and object of the history of development. In the desire to improve an object, a process, or a system, Wilde and Beightler (1967) see an expression of the human striving for *perfection*. Whether such a lofty goal can be attained depends on many conditions.

It is not possible to optimize when there is only one way to carry out a task–then one has no alternative. If it is not even known whether the problem at hand is soluble, the situation calls for an *invention* or *discovery* and not, at that stage, for any optimization. But wherever two or more solutions exist and one must decide upon one of them, one should choose the best, that is to say optimize. Those independent features that distinguish the results from one another are called (independent) *variables* or *parameters* of the object or system under consideration; they may be represented as binary, integer, otherwise discrete, or real values. A *rational decision* between the real or imagined variants presupposes a value judgement, which requires a scale of values, a quantitative *criterion of merit*, according to which one solution can be classified as better, another as worse. This dependent variable is usually called an *objective (function)* because it depends on the objective of the system–the goal to be attained with it–and is functionally related to the parameters. There may even exist several objectives at the same time–the normal case in living systems where the mix of objectives also changes over time and may, in fact, be induced by the actual course of the evolutionary paths themselves.

Sometimes the hardest part of optimization is to define clearly an objective function. For instance, if *several subgoals* are aimed at, a relative weight must be attached to each of the individual criteria. If these are contradictory one only can hope to find a compromise on a trade-off *subset of non-dominated solutions*. Variability and distinct order of merit are the unavoidable conditions of any optimization. One may sometimes also think one has found the right objective for a subsystem, only to realize later that, in doing so, one has provoked unwanted side effects, the ramifications of which have worsened the disregarded total objective function. We are just now experiencing how narrow-minded scales of value can steer us into dangerous plights, and how it is sometimes necessary to consider the whole Earth as a system, even if this is where differences of opinion about value criteria are the greatest.

The second difficulty in optimization, particularly of multiparameter objectives or processes, lies in the choice or design of a suitable *strategy* for proceeding. Even when the objective has been sufficiently clearly defined, indeed even when the functional dependence on the independent variables has been mathematically (or computationally) formulated, it often remains difficult enough, if not completely impossible, to find the optimum, especially in the time available.

The uninitiated often think that it must be an easy matter to solve a problem expressed in the language of mathematics, that most exact of all sciences. Far from it: The problem of how to solve problems is unsolved–and mathematicians have been working on it for centuries. For giving exact answers to questions of extremal values and corresponding positions (or conditions) we are indebted, for example, to the *differential* and *variational calculus*, of which the development in the 18th century is associated with such illustrious names as Newton, Euler, Lagrange, and Bernoulli. These constitute the foundations of the present methods referred to as classical, and of the further developments in the *theory of optimization*. Still, there is often a long way from the theory, which is concerned with establishing *necessary (and sufficient) conditions for the existence of minima and maxima*, to the practice, the determination of these most desirable conditions. Practically significant solutions of *optimization problems* first became possible with the arrival of (large and) fast programmable computers in the mid-20th century. Since then the flood of publications on the subject of optimization has been steadily rising in volume; it is a simple matter to collect several thousand published articles about optimization methods.

Even an interested party finds it difficult to keep pace nowadays with the development that is going on. It seems far from being over, for there still exists no all-embracing theory of optimization, nor is there any universal method of solution. Thus it is appropriate, in Chapter 2, to give a general survey of optimization problems and methods. The special rôle of direct, static, non-discrete, and non-stochastic *parameter optimization* emerges here, for many of these methods can be transferred to other fields; the converse is less often possible. In Chapter 3, some of these strategies are presented in more depth, principally those that extract the information they require only from values of the objective function, that is to say without recourse to analytic partial derivatives (*derivative-free methods)*. Methods of a probabilistic nature are omitted here.

Methods which use *chance* as an aid to decision making, are treated separately in Chapter 4. In *numerical optimization*, chance is simulated deterministically by means of a *pseudorandom* number generator able to produce some kind of *deterministic chaos* only.

One of the *random strategies* proves to be extremely promising. It imitates, in a highly simplified manner, the *mutation-selection* game of nature. This concept, a two membered *evolution strategy*, is formulated in a manner suitable for numerical optimization in Chapter 5, Section 5.1. Following the hypothesis put forward by Rechenberg, that biological evolution is, or possesses, an especially advantageous optimization process and is therefore worthy of imitation, an extended multimembered scheme that imitates the *population principle* of evolution is introduced in Chapter 5, Section 5.2. It permits a more natural as well as more effective specification of the step lengths than the two membered scheme and actually invites the addition of further *evolutionary principles*, such as, for example, sexual propagation and recombination. An approximate theory of the rate of convergence can also be set up for the $(1, \lambda)$ evolution strategy, in which only the best of λ descendants of a generation become parents of the following one.

A short excursion, new to this edition, introduces nearly concurrent developments that the author was unaware of when compiling his dissertation in the early 1970s, i.e., *genetic algorithms*, *simulated annealing*, and *tabu search.*

Chapter 6 then makes a *comparison* of the evolution strategies with the direct search methods of zero, first, and second order, which were treated in detail in Chapter 3. Since the predictive power of theoretical proofs of convergence and statements of rates of convergence is limited to simple problem structures, the comparison includes mainly numerical tests employing various model objective functions. The results are evaluated from two points of view:

- *Efficiency*, or speed of approach to the objective
- *Effectivity*, or reliability under varying conditions

The evolution strategies are highly successful in the test of effectivity or *robustness.* Contrary to the widely held opinion that biological evolution is a very wasteful method of optimization, the *convergence rate* test shows that, in this respect too, the evolution methods can hold their own and are sometimes even more efficient than many purely deterministic methods. The circle is closed in Chapter 7, where the analogy between iterative optimization and evolution is raised once again for discussion, with a look at some natural improvements and extensions of the concept of the evolution strategy.

The list of test problems that were used can be found in Appendix A, and FORTRAN codes of the evolution strategies, with detailed guidance for users, are in Appendix B. Finally, Appendix C explains how to use the C and FORTRAN programs on the floppy disk.

Chapter 2

Problems and Methods of Optimization

2.1 General Statement of the Problems

According to whether one emphasizes the theoretical aspect (existence conditions of optimal solutions) or the practical (procedures for reaching optima), *optimization* nowadays is classified as a branch of applied or *numerical mathematics*, *operations research*, or of computer-assisted *systems (engineering) design*. In fact many optimization methods are based on principles which were developed in linear and non-linear algebra. Whereas for equations, or systems of equations, the problem is to determine a quantity or set of quantities such that functions which depend on them have specified values, in the case of an optimization problem, an initially unknown extremal value is sought. Many of the current methods of solution of systems of linear equations start with an approximation and successively improve it by minimizing the deviation from the required value. For non-linear equations and for incomplete or overdetermined systems this way of proceeding is actually essential (Ortega and Rheinboldt, 1970). Thus many seemingly quite different and apparently unrelated problems turn out, after a suitable reformulation, to be optimization problems.

Into this class come, for example, the solution of differential equations (boundary and initial value problems) and eigenvalue problems, as well as problems of observational calculus, adaptation, and *approximation* (Stiefel, 1965; Schwarz, Rutishauser, and Stiefel, 1968; Collatz and Wetterling, 1971). In the first case, the basic problem again is to solve equations; in the second, the problem is often reduced to minimize deviations in the Gaussian sense (sum of squares of residues) or the Tchebycheff sense (maximum of the absolute residues). Even *game theory* (Vogelsang, 1963) and *pattern or shape recognition* as a branch of information theory (Andrews, 1972; Niemann, 1974) have features in common with the theory of optimization. In one case, from among a stored set of idealized types, a pattern will be sought that has the maximum similarity to the one presented; in another case, the search will be for optimal courses of action in conflict situations. Here, two or more interests are competing. Each player tries to maximize his chance of winning with regard to the way in which his opponent supposedly plays. Most optimization problems, however, are characterized by a single interest, to reach an objective that is not influenced by others.

The engineering aspect of optimization has manifested itself especially clearly with the design of learning robots, which have to *adapt* their operation to the prevailing conditions (see for example Feldbaum, 1962; Zypkin, 1970). The feedback between the environment and the behavior of the robot is effected here by a program, a *strategy*, which can perhaps even alter itself. Wiener (1963) goes even further and considers self-reproducing machines, thus arriving at a consideration of robots that are similar to living beings. *Computers* are often regarded as the most highly developed robots, and it is therefore tempting to make comparisons with the human brain and its neurons and synapses (von Neumann, 1960, 1966; Marfeld, 1970; Steinbuch, 1971). They are nowadays the most important aid to optimization, and many problems are intractable without them.

2.2 Particular Problems and Methods of Solution

The *lack of a universal method of optimization* has led to the present availability of numerous procedures that each have only limited application to special cases. No attempt will be made here to list them all. A short survey should help to distinguish the *parameter optimization* strategies, treated in detail later, from the other procedures, but while at the same time exhibiting some features they have in common. The chosen scheme of presentation is to discuss two opposing concepts together.

2.2.1 Experimental Versus Mathematical Optimization

If the functional relation between the variables and the objective function is unknown, one is forced to experiment either on the real object or on a scale model. To do so one must be as free as possible to vary the independent variables and have access to measuring instruments with which the dependent variable, the quality, can be measured. Systematic investigation of all possible states of the system will be too costly if there are many variables, and random sampling of various combinations is too unreliable for achieving the desired result. A procedure must be significantly more effective if it systematically exploits whatever information is retained about preceding attempts. Such a plan is also called a *strategy.* The concept originated in game theory and was formulated by von Neumann and Morgenstern (1961).

Many of the search strategies of mathematical optimization to be discussed later were also applied under experimental conditions–not always successfully. An important characteristic of the experiment is the unavoidable effect of (stochastic) disturbances on the measured results. A good *experimental optimization* strategy has to take account of this fact and approach the desired extremum with the least possible cost in attempts. Two methods in particular are most frequently mentioned in this connection:

- The EVOP (evolutionary operation) method proposed by G. E. P. Box (1957), a development of the experimental gradient method of Box and Wilson (1951)

- The strategy of artificial evolution designed by Rechenberg (1964)

The algorithm of Rechenberg's *evolution strategy* will be treated in detail in Chapter 5. In the experimental field it has often been applied successfully: for example, to the solution of multiparameter shaping problems (Rechenberg, 1964; Schwefel, 1968; Klockgether and Schwefel, 1970). All variables are simultaneously changed by a small random amount. The changes are (binomially or) normally distributed. The expected value of the random vector is zero (for all components). Failures leave the starting condition unchanged, only successes are adopted. Stochastic disturbances or perturbations, brought about by errors of measurement, do not affect the reliability but influence the speed of convergence according to their magnitude. Rechenberg (1973) gives rules for the optimal choice of a *common variance* of the probability density distribution of the random changes for both the unperturbed and the perturbed cases.

The *EVOP method* of G. E. P. Box changes only two or three parameters at a time–if possible those which have the strongest influence. A square or cube is constructed with an initial condition at its midpoint; its $2^2 = 4$ or $2^3 = 8$ corners represent the points in a cycle of trials. These deterministically established states are tested sequentially, several times if perturbations are acting. The state with the best result becomes the midpoint of the next pattern of points. Under some conditions, one also changes the scaling of the variables or exchanges one or more parameters for others. Details of this altogether heuristic way of proceeding are described by Box and Draper (1969, 1987). The method has mainly been applied to the dynamic optimization of chemical processes. Experiments are performed on the real system, sometimes over a period of several years.

The counterpart to experimental optimization is *mathematical optimization*. The functional relation between the criterion of merit or quality and the variables is known, at least approximately; to put it another way, a more or less simplified *mathematical model* of the object, process or system is available. In place of experiments there appears the manipulation of variables and the objective function. It is sometimes easy to set up a mathematical model, for example if the laws governing the behavior of the physical processes involved are known. If, however, these are largely unresearched, as is often the case for ecological or economic processes, the work of model building can far exceed that of the subsequent optimization.

Depending on what deliberate influence one can have on the process, one is either restricted to the collection of available data or one can uncover the relationships between independent and dependent variables by judiciously planning and interpreting tests. Such methods (Cochran and Cox, 1950; Kempthorne, 1952; Davies, 1954; Cox, 1958; Fisher, 1966; Vajda, 1967; Yates, 1967; John, 1971) were first applied only to agricultural problems, but later spread into industry. Since the analyst is intent on building the best possible model with the fewest possible tests, such an analysis itself constitutes an optimization problem, just as does the synthesis that follows it. Wald (1966) therefore recommends proceeding sequentially, that is to construct a model as a hypothesis from initial experiments or given a priori information, and then to improve it in a stepwise fashion by a further series of tests, or, alternatively, to sometimes reject the model completely. The fitting of model parameters to the measured data can be considered as an optimization problem insofar as the expected error or maximum risk is to be minimized. This is a special case of optimization, called *calculus of observations* , which involves statistical tests like *regression and variance analyses* on data subject to errors, for which the principle of maximum likelihood or minimum χ^2 is used (see Heinhold and Gaede, 1972).

The cost of constructing a model of large systems with many variables, or of very complicated objects, can become so enormous that it is preferable to get to the desired optimal condition by direct variation of the parameters of the process, in other words to optimize experimentally. The fact that one tries to analyze the behavior of a model or system at all is founded on the hope of understanding the processes more fully and of being able to solve the synthesis problem in a more general way than is possible in the case of experimental optimization, which is tied to a particular situation.

If one has succeeded in setting up a mathematical model of the system under consideration, then the optimization problem can be expressed mathematically as follows:

$$F(x) = F(x_1, x_2, \ldots, x_n) \to \text{extr}$$

The round brackets symbolize the functional relationship between the n independent variables

$$\{x_i \, , i \; = \; 1(1)n\}^1$$

and the dependent variable F, the quality or objective function. In the following it is always a scalar quantity. The variables can be scalars or functions of one or more parameters. Whether a maximum or a minimum is sought for is of no consequence for the method of optimization because of the relation

$$\max\{F(x)\} = -\min\{-F(x)\}$$

Without loss of generality one can concentrate on one of the types of problem; usually the minimum problem is considered. Restrictions do arise, insofar as in many practical problems the variables cannot be chosen arbitrarily. They are called *constraints.* The simplest of these are the non-negative conditions:

$$x_i \geq 0 \, , \qquad \text{for all } i \; = \; 1(1)n$$

They are formulated more generally like the objective function:

$$G_j(x) = G_j(x_1, x_2, \ldots, x_n) \left\{ \begin{array}{c} \leq \\ = \\ \geq \end{array} \right\} 0 \, , \qquad \text{for all } j \; = \; 1(1)m$$

The notation chosen here follows the convention of parameter optimization. One distinguishes between equalities and inequalities. Each equality constraint reduces the number of true variables of the problem by one. Inequalities, on the other hand, simply reduce the size of the allowed space of solutions without altering its dimensionality. The sense of the inequality is not critical. Like the interchanging of minimum and maximum problems, one can transform one type into the other by reversing the signs of the terms. It is sufficient to limit consideration to one formulation. For minimum problems this is

---

1The term 1(1)n stands for 1,2,3,...,n.

normally the type $G_j(x) \geq 0$. Points on the edge of the (closed) allowed space are thereby permitted. A different situation arises if the constraint is given as a strict inequality of the form $G_j(x) > 0$. Then the allowed space can be open if $G_j(x)$ is continuous. If for $G_j(x) \geq 0$, with other conditions the same, the minimum lies on the border $G_j(x) = 0$, then for $G_j(x) > 0$, there is no true minimum. One refers here to an *infimum*, the greatest lower bound, at which actually $G_j(x) = 0$. In analogous fashion one distinguishes between maxima and *suprema* (smallest upper bounds). Optimization in the following means always to find a maximum or a minimum, perhaps under inequality constraints.

2.2.2 Static Versus Dynamic Optimization

The term *static optimization* means that the optimum is time invariant or stationary. It is sufficient to determine its position and size once and for all. Once the location of the extremum has been found, the search is over. In many cases one cannot control all the variables that influence the objective function. Then it can happen that these uncontrollable variables change with time and displace the optimum (non-stationary case).

The goal of *dynamic optimization*[2] is therefore to maintain an optimal condition in the face of varying conditions of the environment. The search for the extremum becomes a more or less continuous process. According to the speed of movement of the optimum, it may be necessary, instead of making the slow adjustment of the independent variables by hand–as for example in the EVOP method (see Chap. 2, Sect. 2.2.1), to give the task to a *robot* or *automaton.*

The automaton and the process together form a control loop. However, unlike conventional control loops this one is not required to maintain a desired value of a quantity but to discover the most favorable value of an unknown and time-dependent quantity. Feldbaum (1962), Frankovič et al. (1970), and Zach (1974) investigate in detail such automatic optimization systems, known as extreme value controllers or optimizers. In each case they are built around a search process. For only one variable (adjustable setting) a variety of schemes can be designed. It is significantly more complicated for an optimal value loop when several parameters have to be adjusted.

Many of the search methods are so very costly because there is no a priori information about the process to be controlled. Hence nowadays one tries to build *adaptive* control systems that use information gathered over a period of time to set up an internal model of the system, or that, in a sense, learn. Oldenburger (1966) and, in more detail, Zypkin (1970) tackle the problems of learning and self-learning robots. Adaptation is said to take place if the change in the control characteristics is made on the basis of measurements of those input quantities to the process that cannot be altered–also known as disturbing variables. If the output quantities themselves are used (here the objective function) to adjust the control system, the process is called *self-learning* or *self-adaptation.* The latter possibility is more reliable but, because of the time lag, slower. Cybernetic engineering is concerned with learning processes in a more general form and always sees or even seeks links with natural analogues.

An example of a robot that adapts itself to the environment is the *homeostat* of Ashby

[2]Some authors use the term dynamic optimization in a different way than is done here.

(1960). Nowadays, however, one does not build one's own optimizer every time there is a given problem to be solved. Rather one makes use of so-called *process computers*, which for a new task only need another special program. They can handle large and complicated problems and are coupled to the process by sensors and transducers in a closed loop (on-line) (Levine and Vilis, 1973; McGrew and Haimes, 1974). The actual computer usually works digitally, so that analogue-digital and digital-analogue converters are required for input and output. Process computers are employed for keeping process quantities constant and maintaining required profiles as well as for optimization. In the latter case an internal model (a computer program) usually serves to determine the optimal process parameters, taking account of the latest measured data values in the calculation.

If the position of the optimum in a dynamic process is shifting very rapidly, the manner in which the search process follows the extremum takes on a greater significance for the overall quality. In this case one has to go about setting up a dynamic model and specifying all variables, including the controllable ones, as functions of time. The original parameter optimization goes over to functional optimization.

2.2.3 Parameter Versus Functional Optimization

The case when not only the objective function but also the independent variables are scalar quantities is called *parameter optimization*. Numerical values

$$\{x_i^*, i = 1(1)n\}$$

of the variables or parameters are sought for which the value of the objective function

$$F^* = F(x^*) = \mathrm{extr}\{F(x)\}$$

is an optimum. The number of parameters describing a state of the object or system is finite. In the simplest case of only one variable ($n = 1$), the behavior of the objective function is readily visualized on a diagram with two orthogonal axes. The value of the parameter is plotted on the abscissa and that of the objective function on the ordinate. The functional dependence appears as a curve. For $n = 2$ a three dimensional Cartesian coordinate system is required. The state of the system is represented as a point in the horizontal plane and the value of the objective function as the vertical height above it. A mountain range is obtained, the surface of which expresses the relation of dependent to independent variables. To further simplify the representation, the curves of intersection between the mountain range and parallel horizontal planes are projected onto the base plane, which provides a contour diagram of the objective function. From this three dimensional picture and its two dimensional projection, concepts like peak, plateau, valley, ridge, and contour line are readily transferred to the n-dimensional case, which is otherwise beyond our powers of description and visualization.

In *functional optimization*, instead of optimal points in three dimensional Euclidean space, optimal trajectories in function spaces (such as Banach or Hilbert space) are to be determined. Thus one refers also to infinite dimensional optimization as opposed to the finite dimensional parameter optimization. Since the variables to be determined are themselves functions of one or more parameters, the objective function is a function of a function, or a functional.

A classical problem is to determine the smooth curve down which a point mass will slide between two points in the shortest time, acted upon by the force of gravity and without friction. Known as the *brachistochrone problem*, it can be solved by means of the ordinary variational calculus (Courant and Hilbert, 1968a,b; Denn, 1969; Clegg, 1970). If the functions to be determined depend on several variables it is a multidimensional variational problem (Klötzler, 1970). In many cases the time t appears as the only parameter. The objective function is commonly an integral, in the integrand of which will appear not only the independent variables

$$x(t) = \{x_1(t), x_2(t), \ldots, x_n(t)\}$$

but also their derivatives $\dot{x}_i(t) = \partial x_i / \partial t$ and sometimes also the parameter t itself:

$$F(x(t)) = \int_{t_1}^{t_2} f(x(t), \dot{x}(t), t)\, dt \rightarrow \text{extr}$$

Such problems are typical in control theory, where one has to find optimal controlling functions for control processes (e.g., Chang, 1961; Lee, 1964; Leitmann, 1964; Hestenes, 1966; Balakrishnan and Neustadt, 1967; Karreman, 1968; Demyanov and Rubinov, 1970).

Whereas the *variational calculus* and its extensions provide the mathematical basis of functional optimization (in the language of control engineering: optimization with distributed parameters), parameter optimization (with localized parameters) is based on the *theory of maxima and minima* from the elementary differential calculus. Consequently both branches have followed independent paths of development and become almost separate disciplines. The functional analysis theory of Dubovitskii and Milyutin (see Girsanov, 1972) has bridged the gap between the problems by allowing them to be treated as special cases of one fundamental problem, and it could thus lead to a general theory of optimization. However different their theoretical bases, in cases of practical significance the problems must be solved on a computer, and the *iterative methods* employed are then broadly the same.

One of these iterative methods is the *dynamic programming* or stepwise optimization of Bellman (1967). It was originally conceived for the solution of economic problems, in which time-dependent variables are changed in a stepwise way at fixed points in time. The method is a discrete form of functional optimization in which the trajectory sought appears as a steplike function. At each step a decision is taken, the sequence of which is called a *policy.* Assuming that the state at a given step depends only on the decision at that step and on the preceding state–i.e., there is *no feedback*–, then dynamic programming can be applied. The Bellman *optimum principle* implies that each piece of the optimal trajectory that includes the end point is also optimal. Thus one begins by optimizing the final decision at the transition from the last-but-one to the last step. Nowadays dynamic programming is frequently applied to solving discrete problems of optimal control and regulation (Gessner and Spremann, 1972; Lerner and Rosenman, 1973). Its advantage compared to other, analytic methods is that its algorithm can be formulated as a program suitable for digital computers, allowing fairly large problems to be tackled (Gessner and

Wacker, 1972). Bellman's optimum principle can, however, also be expressed in differential form and applied to an area of continuous functional optimization (Jacobson and Mayne, 1970).

The principle of stepwise optimization can be applied to problems of parameter optimization, if the objective function is *separable* (Hadley, 1969): that is, it must be expressible as a sum of partial objective functions in which just one or a very few variables appear at a time. The number of steps (k) corresponds to the number of the partial functions; at each step a decision is made only on the (ℓ) variables in the partial objective. They are also called control or decision variables. Subsidiary conditions (number m) in the form of inequalities can be taken into account. The constraint functions, like the variables, are allowed to take a finite number (b) of discrete values and are called state variables. The recursion formula for the stepwise optimization will not be discussed here. Only the number of required operations (N) in the calculation will be mentioned, which is of the order

$$N \sim k\, b^{m+\ell}$$

For this reason the usefulness of dynamic programming is mainly restricted to the case $\ell = 1$, $k = n$, and $m = 1$. Then at each of the n steps, just one control variable is specified with respect to one subsidiary condition. In the other limiting case where all variables have to be determined at one step, the normal case of parameter optimization, the process goes over to a *grid method* (complete enumeration) with a computational requirement of order $O(b^{(n+m)})$. Herein lies its capability for locating global optima, even of complicated *multimodal* objective functions. However, it is only especially advantageous if the structure of the objective function permits the enumeration to be limited to a small part of the allowed region.

Digital computers are poorly suited to solving continuous problems because they cannot operate directly with functions. Numerical integration procedures are possible, but costly. *Analogue computers* are more suitable because they can directly imitate dynamic processes. Compared to digital computers, however, they have a small numerical range and low accuracy and are not so easily programmable. Thus sometimes digital and analogue computers are coupled for certain tasks as so-called *hybrid computers*. With such systems a set of differential equations can be tackled to the same extent as a problem in functional optimization (Volz, 1965, 1973). The digital computer takes care of the iteration control, while on the analogue computer the differentiation and integration operations are carried out according to the parameters supplied by the digital computer. Korn and Korn (1964), and Bekey and Karplus (1971), describe the operations involved in trajectory optimization and the solution of differential equations by means of hybrid computers. The fact that random methods are often used for such problems has to do with the computational imprecision of the analogue part, with which deterministic processes usually fail to cope. If requirements for accuracy are very high, however, purely digital computation has to take over, with the consequent greater cost in computation time.

2.2.4 Direct (Numerical) Versus Indirect (Analytic) Optimization

The classification of mathematical methods of optimization into direct and indirect procedures is attributed to Edelbaum (1962). Especially if one has a computer model of a system, with which one can perform simulation experiments, the search for a certain set of exogenous parameters to generate excellent results asks for robust direct optimization methods. Direct or numerical methods are those that approach the solution in a stepwise manner (iteratively), at each step (hopefully) improving the value of the objective function. If this cannot be guaranteed, a trial and error process results.

An *indirect or analytic* procedure attempts to reach the optimum in a single (calculation) step, without tests or trials. It is based on the analysis of the special properties of the objective function at the position of the extremum. In the simplest case, parameter optimization without constraints, one proceeds on the assumption that the tangent plane at the optimum is horizontal, i.e., the first partial derivatives of the objective function exist and vanish in x^*:

$$\left.\frac{\partial F}{\partial x_i}\right|_{x=x^*} = 0\,, \qquad \text{for all } i \;=\; 1(1)n \tag{2.1}$$

This *system of equations* can be expressed with the so-called Nabla operator (∇) as a single vector equation for the stationary point x^*:

$$\nabla F(x^*) = 0 \tag{2.2}$$

Equation (2.1) or (2.2) transforms the original optimization problem into a problem of solving a set of, perhaps non-linear, simultaneous equations. If $F(x)$ or one or more of its derivatives are not continuous, there may be extrema that do not satisfy the otherwise necessary conditions. On the other hand not every point in $\mathbb{R}^n$–the n-dimensional space of real variables– that satisfies conditions (2.1) need be a minimum; it could also be a maximum or a saddle point. Equation (2.2) is referred to as a *necessary condition* for the existence of a local minimum.

To give *sufficient conditions* requires further processes of differentiation. In fact, differentiations must be carried out until the determinant of the matrix of the second or higher partial derivatives at the point x^* is non-zero. Things remain simple in the case of only one variable, when it is required that the lowest order non-vanishing derivative is positive and of even order. Then and only then is there a minimum. If the derivative is negative, x^* represents a maximum. A saddle point exists if the order is odd.

For n variables, at least the $\frac{n}{2}(n+1)$ second partial derivatives

$$\frac{\partial^2 F(x)}{\partial x_i\, \partial x_j}\,, \qquad \text{for all } i,j \;=\; 1(1)n$$

must exist at the point x^*. The determinant of the *Hessian matrix* $\nabla^2 F(x^*)$ must be positive, as well as the further $n - 1$ principle subdeterminants of this matrix. While MacLaurin had already completely formulated the sufficient conditions for the existence of minima and maxima of one parameter functions in 1742, the corresponding theory for functions of several variables was only completed nearly 150 years later by Scheeffer (1886) and Stolz (1893) (see also Hancock, 1960).

Sufficient conditions can only be applied to check a solution that was obtained from the necessary conditions. The analytic path thus always leads the original optimization problem back to the problem of solving a system of simultaneous equations (Equation (2.2)). If the objective function is of second order, one is dealing with a linear system, which can be solved with the aid of one of the usual methods of linear algebra. Even if non-iterative procedures are used, such as the *Gaussian elimination* algorithm or the *matrix decomposition* method of Cholesky, this cannot be done with a single-step calculation. Rather the number of operations grows as $O(n^3)$. With fast digital computers it is certainly a routine matter to solve systems of equations with even thousands of variables; however, the inevitable rounding errors mean that complete accuracy is never achieved (Broyden, 1973).

One can normally be satisfied with a sufficiently good approximation. Here *relaxation methods*, which are iterative, show themselves to be comparable or superior. It depends in detail on the structure of the coefficient matrix. Starting from an initial approximation, the error as measured by the residues of the equations is minimized. Relaxation procedures are therefore basically optimization methods but of a special kind, since the value of the objective function at the optimum is known beforehand. This a priori information can be exploited to make savings in the computations, as can the fact that each component of the residue vector must individually go to zero (e.g., Traub, 1964; Wilkinson and Reinsch, 1971; Hestenes, 1973; Hestenes and Stein, 1973).

Objective functions having terms or members of higher than second order lead to non-linear equations as the necessary conditions for the existence of extrema. In this case, the stepwise approach to the null position is essential, e.g., with the *interpolation method*, which was conceived in its original form by Newton (Chap. 3, Sect. 3.1.2.3.2). The equations are linearized about the current approximation point. Linear relations for the correcting terms are then obtained. In this way a complete system of n linear equations has to be solved at each step of the iteration. Occasionally a more convenient approach is to search for the minimum of the function

$$\tilde{F}(x) = \sum_{i=1}^{n} \left(\frac{\partial F}{\partial x_i} \right)^2$$

with the help of a *direct optimization* method. Besides the fact that $\tilde{F}(x)$ goes to zero, not only at the sought for minimum of $F(x)$ but also at its maxima and saddle points, it can sometimes yield non-zero minima of no interest for the solution of the original problem. Thus it is often preferable not to proceed via the conditions of Equation (2.2) but to minimize $F(x)$ directly. Only in special cases do indirect methods lead to faster, more elegant solutions than direct methods. Such is, for example, the case if the necessary existence condition for minima with one variable leads to an algebraic equation, and *sectioning algorithms* like the computational scheme of Horner can be used; or if objective functions are in the form of so-called *posynomes*, for which Duffin, Peterson, and Zener (1967) devised *geometric programming*, an entirely indirect method.

Subsidiary conditions, or constraints, complicate matters. In rare cases equality constraints can be expressed as equations in one variable, that can be eliminated from the objective function, or constraints in the form of inequalities can be made superfluous by a transformation of the variables. Otherwise there are the methods of bounded variation and *Lagrange multipliers*, in addition to penalty functions and the procedures of mathematical programming.

The situation is very similar for functional optimization, except that here the indirect methods are still dominant even today. The *variational calculus* provides as conditions for optima differential instead of ordinary equations–actually ordinary differential equations (Euler-Lagrange) or partial differential equations (Hamilton-Jacobi). In only a few cases can such a system be solved in a straightforward way for the unknown functions. One must usually resort again to the help of a computer. Whether it is advantageous to use a digital or an analogue computer depends on the problem. It is a matter of speed versus accuracy. A hybrid system often turns out to be especially useful. If, however, the solution cannot be found by a purely analytic route, why not choose from the start the direct procedure also for functional optimization? In fact with the increasing complexity of practical problems in numerical optimization, this field is becoming more important, as illustrated by the work of Daniel (1969), who takes over methods *without derivatives* from parameter optimization and applies them to the optimization of functionals. An important point in this is the discretization or *parameterization* of the originally continuous problem, which can be achieved in at least two ways:

- By approximation of the desired functions using a sum of suitable known functions or polynomials, so that only the coefficients of these remain to be determined (Sirisena, 1973)

- By approximation of the desired functions using step functions or sides of polygons, so that only heights and positions of the connecting points remain to be determined

Recasting a functional into a parameter optimization problem has the great advantage that a digital computer can be used straightaway to find the solution numerically. The disadvantage that the result only represents a suboptimum is often not serious in practice, because the assumed values of parameters of the process are themselves not exactly known (Dixon, 1972a). The experimentally determined numbers are prone to errors or to statistical uncertainties. In any case, large and complicated functional optimization problems cannot be completely solved by the indirect route.

The direct procedure can either start directly with the functional to be minimized, if the integration over the substituted function can be carried out (Rayleigh-Ritz method); or with the necessary conditions, the differential equations, which specify the optimum. In the latter case the integral is replaced by a finite sum of terms (Beveridge and Schechter, 1970). In this situation *gradient methods* are readily applied (Kelley, 1962; Klessig and Polak, 1973). The detailed way to proceed depends very much on the subsidiary conditions or constraints of the problem.

2.2.5 Constrained Versus Unconstrained Optimization

Special techniques have been developed for handling problems of optimization with constraints. In parameter optimization these are the methods of *penalty functions* and *mathematical programming.* In the first case a modified objective function is set up, which

- For the minimum problem takes the value $F(x) = +\infty$ in the forbidden region, but which remains unchanged in the allowed (feasible) region (*barrier method*; e.g., used within the evolution strategies, see Chap. 5)
- Only near the boundary inside the allowed region, yields values different from $F(x)$ and thus keeps the search at a distance from the edge (*partial penalty function*; e.g., used within Rosenbrock's strategy, see Chap. 3, Sect. 3.2.1.3)
- Differs from $F(x)$ over the whole space spanned by the variables (*global penalty function*)

This last is the most common way of treating constraints in the form of inequalities. The main ideas here are due to Carroll (1961; *created response surface technique*) and to Fiacco and McCormick (1964, 1990; SUMT, *sequential unconstrained minimization technique*).

For the problem

$$\begin{aligned} F(x) &\rightarrow \min \\ G_j(x) &\geq 0\,, \quad \text{for all } j = 1(1)m \\ H_k(x) &= 0\,, \quad \text{for all } k = 1(1)\ell \end{aligned}$$

the penalty function is of the form (with $r, v_j, w_k > 0$, and $G_j > 0$)

$$\tilde{F}(x) = F(x) + r \sum_{j=1}^{m} \frac{v_j}{G_j(x)} + \frac{1}{r} \sum_{k=1}^{\ell} w_k \, [H_k(x)]^2$$

The coefficients v_j and w_k are weighting factors for the individual constraints and r is a free parameter. The optimum of $\tilde{F}(x)$ will depend on the choice of r, so it is necessary to alter r in a stepwise way. The original extreme value problem is thereby solved by a sequence of optimizations in which r is gradually reduced to zero. One can hope in this way at least to find good approximations for the required minimum problem within a finite sequence of optimizations.

The choice of suitable values for r is not, however, easy. Fiacco (1974) and Fiacco and McCormick (1968, 1990) give some indications, and also suggest further possibilities for penalty functions. These procedures are usually applied in conjunction with gradient methods. The *hemstitching* method and the *riding the constraints* method of Roberts and Lyvers (1961) work by changing the chosen direction whenever a constraint is violated, without using a modified objective function. They orient themselves with respect to the gradient of the objective and the derivatives of the constraint functions (*Jacobian matrix*). In hemstitching, there is always a return into the feasible region, while in riding the constraints the search runs along the active constraint boundaries. The variables are reset into the allowed region by the *complex method* of M. J. Box (1965) (a direct search strategy) whenever explicit bounds are crossed. Implicit constraints on the other hand are treated as barriers (see Chap. 3, Sect. 3.2.1.6).

The methods of *mathematical programming*, both linear and non-linear, treat the constraints as the main aspect of the problem. They were specially evolved for operations research (Müller-Merbach, 1971) and assume that all variables must always be positive. Such non-negativity conditions allow special solution procedures to be developed. The simplest models of economic processes are linear. There are often no better ones available. For this purpose Dantzig (1966) developed the simplex method of *linear programming* (see also Krelle and Künzi, 1958; Hadley, 1962; Weber, 1972).

The linear constraints, together with the condition on the signs of the variables, span the feasible region in the form of a polygon (for $n = 2$) or a polyhedron, sometimes called simplex. Since the objective function is also linear, except in special cases, the desired extremum must lie in a corner of the polyhedron. It is therefore sufficient just to examine the corners. The *simplex method* of Dantzig does this in a particularly economic way, since only those corners are considered in which the objective function has progressively better values. It can even be thought of as a gradient method along the edges of the polyhedron. It can be applied in a straightforward way to many hundreds, even thousands, of variables and constraints. For very large problems, which may have a particular structure, special methods have also been developed (Künzi and Tan, 1966; Künzi, 1967). Into this category come the revised and the dual simplex methods, the multiphase and duplex methods, and decomposition algorithms. An unpleasant property of linear programs is that sometimes just small changes of the coefficients in the objective function or the constraints can cause a big alteration in the solution. To reveal such dependencies, methods of *parametric* linear programming and *sensitivity analysis* have been developed (Dinkelbach, 1969).

Most strategies of *non-linear programming* resemble the simplex method or use it as a subprogram (Abadie, 1972). This is the case in particular for the techniques of quadratic programming, which are conceived for quadratic objective functions and linear constraints. The theory of non-linear programming is based on the optimality conditions developed by Kuhn and Tucker (1951), an extension of the theory of maxima and minima to problems with constraints in the form of inequalities. These can be expressed geometrically as follows: at the optimum (in a corner of the allowed region) the gradient of the objective function lies within the cone formed by the gradients of the active constraints. To start with, this is only a necessary condition. It becomes sufficient under certain assumptions concerning the structure of the objective and constraint functions. For minimum problems, the objective function and the feasible region must be convex, that is the constraints must be concave. Such a problem is also called a *convex program.* Finally the Kuhn-Tucker theorem transforms a convex program into an equivalent saddle point problem (Arrow and Hurwicz, 1956), just as the Lagrange multiplier method does for constraints in the form of equalities. A complete theory of equality constraints is due to Apostol (1957).

Non-linear programming is therefore only applicable to convex optimization, in which, to be precise, one must distinguish at least seven *types of convexity* (Ponstein, 1967). In addition, all the functions are usually required to be continuously differentiable, with an analytic specification of their partial derivatives. There is an extensive literature on this subject, of which the books by Arrow, Hurwicz, and Uzawa (1958), Zoutendijk (1960), Vajda (1961), Künzi, Krelle, and Oettli (1962), Künzi, Tzschach, and Zehnder (1966, 1970), Künzi and Krelle (1969), Zangwill (1969), Suchowitzki and Awdejewa (1969), Mangasarian (1969), Stoer and Witzgall (1970), Whittle (1971), Luenberger (1973), and Varga (1974) are but a small sample. Kappler (1967) considers some of the procedures from the point of view of gradient methods. Künzi and Oettli (1969) give a survey of the more extended procedures together with an extensive bibliography. FORTRAN programs are to be found in McMillan (1970), Kuester and Mize (1973), and Land and Powell (1973).

Of special importance in *control theory* are optimization problems in which the constraints are partly specified as differential equations. They are also called *non-holonomous constraints*. Pontrjagin et al. (1967) have given necessary conditions for the existence of optima in these problems. Their trick was to distinguish between the free control functions to be determined and the local or state functions which are bound by constraints. Although the theory has given a strong foothold to the analytic treatment of optimal control processes, it must be regarded as a case of good luck if a practical problem can be made to yield an exact solution in this way. One must usually resort in the end to numerical approximation methods in order to obtain the desired optimum (e.g., Balakrishnan and Neustadt, 1964, 1967; Rosen, 1966; Leitmann, 1967; Kopp, 1967; Mufti, 1970; Tabak, 1970; Canon, Cullum, and Polak, 1970; Tolle, 1971; Unbehauen, 1971; Boltjanski, 1972; Luenberger, 1972; Polak, 1973).

2.3 Other Special Cases

According to the type of variables there are still other special areas of mathematical optimization. In parameter optimization for example the variables can sometimes be restricted to discrete or integer values. The extreme case is if a parameter may only take two distinct values, zero and unity. Mixed variable types can also appear in the same problem; hence the terms *discrete, integer, binary* (or *zero-one*), and *mixed-integer programming*. Most of the solution procedures that have been worked out deal with linear integer problems (e.g., those proposed by Gomory, Balas, and Beale). An important class of methods, the *branch and bound methods*, is described for example by Weinberg (1968). They are classed together with dynamic programming as *decision tree strategies*. For the general non-linear case, a last resort can be to try out all possibilities. This kind of optimization is referred to as *complete enumeration*. Since the cost of such a procedure is usually prohibitive, heuristic approaches are also tried, with which usable, not necessarily optimal, solutions can be found (Weinberg and Zehnder, 1969). More clever ways of proceeding in special cases, for example by applying non-integer techniques of linear and non-linear programming, can be found in Korbut and Finkelstein (1971), Greenberg (1971), Plane and McMillan (1971), Burkard (1972), Hu (1972), and Garfinkel and Nemhauser (1972, 1973).

By *stochastic programming* is meant the solution of problems with objective functions, and sometimes also constraints, that are subject to statistical perturbations (Faber, 1970). It is simplest if such problems can be reduced to deterministic ones, for example by working with *expectation values.* However, there are some problems in which the probability distributions significantly influence the optimal solution. Operational methods at first only existed for special cases such as, for example, warehouse problems (Beckmann, 1971). Their numbers as well as the fields of application are growing steadily (Hammer, 1984; Ermoliev and Wets, 1988; Ermakov, 1992). In general, one has to make a clear distinction between deterministic solution methods for more or less noisy or stochastic situations and *stochastic methods* for deterministic but difficult situations like multimodal or fractal topologies. Here we refer to the former; in Chapter 4 we will do so for the latter, especially under the aspect of *global optimization.*

In a rather new branch within the mathematical programming field, called *non-smooth* or *non-differentiable optimization*, more or less classical gradient-type methods for finding solutions still persist (e.g., Balinski and Wolfe, 1975; Lemarechal and Mifflin, 1978; Nurminski, 1982; Kiwiel, 1985).

For successively approaching the zero or extremum of a function if the measured values are subject to uncertainties, a familiar strategy is that of *stochastic approximation* (Wasan, 1969). The original concept is due to Robbins and Monro (1951). Kiefer and Wolfowitz (1952) have adapted it for problems in which the maximum of a unimodal regression function is sought. Blum (1954a) has proved that the method is certain to converge. It distinguishes between test or trial steps and work steps. With one variable, starting at the point $x^{(k)}$, the value of the objective function is obtained at the two positions $x^{(k)} \pm c^{(k)}$. The slope is then calculated as

$$y^{(k)} = \frac{F(x^{(k)} + c^{(k)}) - F(x^{(k)} - c^{(k)})}{2c^{(k)}}$$

A work step follows from the recursion formula (for minimum searches)

$$x^{(k+1)} = x^{(k)} - 2a^{(k)} y^{(k)}$$

The choice of the positive sequences $c^{(k)}$ and $a^{(k)}$ is important for convergence of the process. These should satisfy the relations

$$\begin{aligned} \lim_{k\to\infty} c^{(k)} &\to 0 \\ \sum_{k=1}^{\infty} a^{(k)} &= \infty \\ \sum_{k=1}^{\infty} a^{(k)} c^{(k)} &< \infty \\ \sum_{k=1}^{\infty} \left(\frac{a^{(k)}}{c^{(k)}}\right)^2 &< \infty \end{aligned}$$

One chooses for example the sequences

$$\begin{aligned} a^{(k)} = \frac{a^{(0)}}{k}, \quad a^{(0)} &> 0 \\ c^{(k)} = \frac{c^{(0)}}{\sqrt[4]{k}}, \quad c^{(0)} &> 0, \quad k > 0 \end{aligned}$$

This means that the work step length goes to zero very much faster than the test step length, in order to compensate for the growing influence of the perturbations.

Blum (1954b) and Dvoretzky (1956) describe how to apply this process to multidimensional problems. The increment in the objective function, hence an approximation to the gradient vector, is obtained from $n + 1$ observations. Sacks (1958) uses $2\,n$ trial steps. The stochastic approximation can thus be regarded, in a sense, as a particular gradient method.

Yet other basic strategies have been proposed; these adopt only the choice of step lengths from the stochastic approximation, while the directions are governed by other criteria. Thomas and Wilde (1964) for example, combine the stochastic approximation with the relaxation method of Southwell (1940, 1946). Kushner (1963) and Schmitt (1969) even take *random directions* into consideration. All the proofs of convergence of the stochastic approximation assume unimodal objective functions. A further disadvantage is that stability against perturbations is bought at a very high cost, especially if the number of variables is large. How many steps are required to achieve a given accuracy can only be stated if the probability density distribution of the stochastic perturbations is known. Many authors have tried to devise methods in which the basic procedure can be accelerated: e.g., Kesten (1958), who only reduces the step lengths after a change in direction of the search, or Odell (1961), who makes the lengths of the work steps dependent on measured values of the objective function. Other attempts are directed towards reducing the effect of the perturbations (Venter, 1967; Fabian, 1967), for example by making only the direction and not the size of the gradients determine the step lengths. Bertram (1960) describes various examples of applications. More of such work is that of Krasulina (1972) and Engelhardt (1973).

In this introduction many classes of possible or practically occurring optimization problems and methods have been sketched briefly, but the coverage is far from complete. No mention has been made, for example, of *broken rational programming*, nor of *graphical methods* of solution. In operations research especially (Henn and Künzi, 1968) there are many special techniques for solving *transport, allocation, routing, queuing*, and *warehouse problems*, such as network planning and other graph theoretical methods. This excursion into the vast realm of optimization problems was undertaken because some of the algorithms to be studied in more depth in what follows, especially the random methods of Chapter 4, owe their origin and nomenclature to other fields. It should also be seen to what extent methods of direct parameter optimization permeate the other branches of the subject, and how they are related to each other. An overall scheme of how the various branches are interrelated can be found in Saaty (1970).

If there are two or more objectives at the same time and occasion, and especially if these are not conflict-free, single solution points in the decision variable space can no longer give the full answer to an optimization question, not even in the otherwise simplest situation. How to look for the whole subset of efficient, non-dominated, or *Pareto-optimal* solutions can be found under keywords like *vector optimization*, *polyoptimization* or *multiple criteria decision making* (MCDM) (e.g., Bell, Keeney, and Raiffa, 1977; Hwang and Masud, 1979; Peschel, 1980; Grauer, Lewandowski, and Wierzbicki, 1982; Steuer, 1986). *Game theory* comes into play when several decision makers have access to different parts of the decision variable set only (e.g., Luce and Raiffa, 1957; Maynard Smith, 1982; Axelrod, 1984; Sigmund, 1993). No consideration is given here to these special fields.

Chapter 3

Hill climbing Strategies

In this chapter some of the direct, mathematical parameter optimization methods will be treated in more detail for static, non-discrete, non-stochastic, mostly unconstrained functions. They come under the general heading of *hill climbing strategies* because their manner of searching for a maximum corresponds closely to the intuitive way a sightless climber might feel his way from a valley up to the highest peak of a mountain. For minimum problems the sense of the displacements is simply reversed, otherwise *uphill or ascent* and *downhill or descent* methods (Bach, 1969) are identical. Whereas methods of mathematical programming are dominant in operations research and the special methods of functional optimization in control theory, the hill climbing strategies are most frequently applied in *engineering design*. Analytic methods often prove unsuitable in this field

- Because the assumptions are not satisfied under which necessary conditions for extrema can be stated (e.g., continuity of the objective function and its derivatives)
- Because there are difficulties in carrying out the necessary differentiations
- Because a solution of the equations describing the conditions does not always lead to the desired optimum (it can be a local minimum, maximum, or saddle point)
- Because the equations describing the conditions, in general a system of simultaneous non-linear equations, are not immediately soluble

To what extent hill climbing strategies take care of these particular characteristics depends on the individual method. Very thorough presentations covering some topics can be found in Wilde (1964), Rosenbrock and Storey (1966), Wilde and Beightler (1967), Kowalik and Osborne (1968), Box, Davies, and Swann (1969), Pierre (1969), Pun (1969), Converse (1970), Cooper and Steinberg (1970), Hoffmann and Hofmann (1970), Beveridge and Schechter (1970), Aoki (1971), Zahradnik (1971), Fox (1971), Céa (1971), Daniel (1971), Himmelblau (1972b), Dixon (1972a), Jacoby, Kowalik and Pizzo (1972), Stark and Nicholls (1972), Brent (1973), Gottfried and Weisman (1973), Vanderplaats (1984), and Papageorgiou (1991). More variations or theoretical and numerical studies of older methods can be found as individual publications in a wide variety of journals, or in the volumes of collected articles such as Graves and Wolfe (1963), Blakemore and Davis (1964), Lavi and Vogl (1966), Klerer and Korn (1967), Abadie (1967, 1970), Fletcher (1969a), Rosen, Mangasarian, and Ritter (1970), Geoffrion (1972), Murray (1972a), Lootsma (1972a), Szegö (1972), and Sebastian and Tammer (1990).

Formulated as a minimum problem without constraints, the task can be stated as follows:

$$\min_x \{F(x) \mid x \in \mathbb{R}^n\} \tag{3.1}$$

The column vector x^* (at the extreme position) is required

$$x^* = \begin{bmatrix} x_1^* \\ x_2^* \\ \vdots \\ x_n^* \end{bmatrix} = (x_1^*, x_2^*, \ldots, x_n^*)^T$$

and the associated extreme value $F^* = F(x^*)$ of the objective function $F(x)$, in this case the minimum. The expression $x \in \mathbb{R}^n$ means that the variables are allowed to take all real values; x can thus be represented by any point in an n-dimensional Euclidean space $\mathbb{R}^n$. Different types of minima are distinguished: strong and weak, local and global.

For a local minimum the following relationship holds:

$$F(x^*) \leq F(x) \tag{3.2}$$

for

$$0 \leq \quad \|x - x^*\| = \sqrt{\sum_{i=1}^{n} (x_i - x_i^*)^2} \quad \leq \varepsilon$$

and

$$x \in \mathbb{R}^n$$

This means that in the neighborhood of x^* defined by the size of ε there is no vector x for which $F(x)$ is smaller than $F(x^*)$. If the equality sign in Equation (3.2) only applies when $x = x^*$, the minimum is called *strong*, otherwise it is *weak*. An objective function that only displays one minimum (or maximum) is referred to as *unimodal*. In many cases, however, $F(x)$ has several *local* minima (and maxima), which may be of different heights. The smallest, absolute or *global* minimum (minimum minimorum) of a multimodal objective function satisfies the stronger condition

$$F(x^*) \leq F(x)\,, \quad \text{for all } x \in \mathbb{R}^n \tag{3.3}$$

This is always the preferred object of the search.

If there are also constraints, in the form of inequalities

$$G_j(x) \geq 0\,, \quad \text{for all } j = 1(1)m \tag{3.4}$$

or equalities

$$H_k(x) = 0\,, \quad \text{for all } k = 1(1)\ell \tag{3.5}$$

then $\mathbb{R}^n$ in Equations (3.1) to (3.3) must either be replaced by the hopefully non-empty subset $M \in \mathbb{R}^n$ to represent the feasible region in $\mathbb{R}^n$ defined by Equation (3.4), or by $\mathbb{R}^{n-\ell}$, the subspace of lower dimensionality spanned by the variables that now depend on each other according to Equation (3.5). If solutions at infinity are excluded, then the theorem of Weierstrass holds (see for example Rothe, 1959): "In a closed compact region $a \leq x \leq b$ every function which is continuous there has at least one (i.e., an absolute) minimum and maximum." This can lie inside or on the boundary. In the case of discontinuous functions, every point of discontinuity is also a potential candidate for the position of an extremum.

3.1 One Dimensional Strategies

The search for a minimum is especially easy if the objective function only depends on one variable.

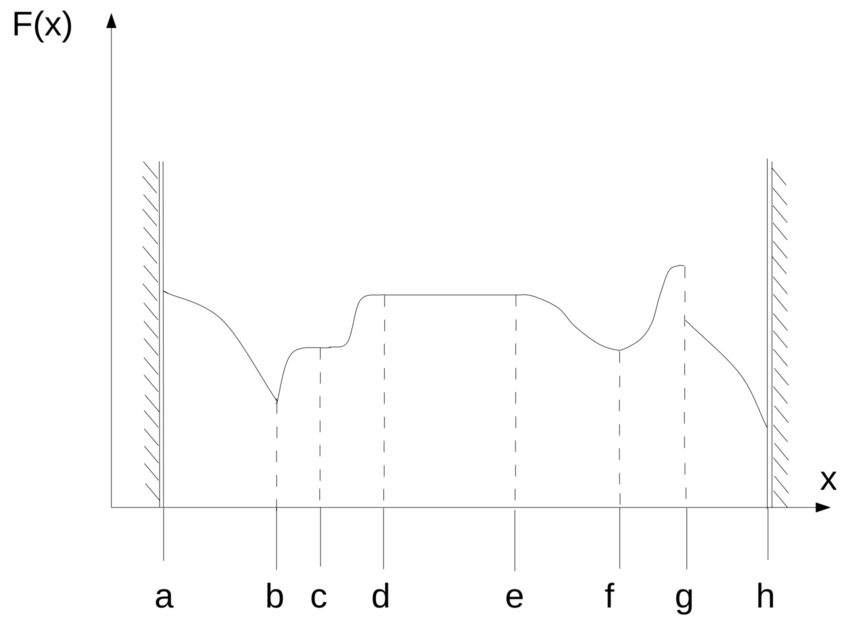


Figure 3.1: Special points of a function of one variable

a:	local maximum at the boundary
b:	local minimum at a point of discontinuity of $F_x(x)$
c:	saddle point, or point of inflection
d-e:	weak local maximum
f:	local minimum
g:	maximum (may be global) at a point of discontinuity of $F(x)$
h:	global minimum at the boundary

This problem would be of little practical interest, however, were it not for the fact that many of the multidimensional strategies make use of one dimensional minimizations in selected directions, referred to as *line searches.* Figure 3.1 shows some possible ways minima and other special points can arise in the one dimensional case.

3.1.1 Simultaneous Methods

One possible way of discovering the minimum of a function with one parameter is to determine the value of the objective function at a number of points and then to declare the point with the smallest value the minimum. Since in principle all trials can be carried out at the same time, this procedure is referred to as *simultaneous optimization.* How closely the true minimum is approached depends on the choice of the number and location of the trial points. The more trials are made, the more accurate the solution can be. One will be concerned, however, to obtain a result at the lowest cost in time and computation (or material). The two requirements of high accuracy and lowest cost are contradictory; thus an optimum compromise must be sought.

The effectiveness of a search method is judged by the size of the largest remaining interval of uncertainty (in the least favorable case) relative to the position of the minimum for a given number of trials (the so-called *minimax concept*, see Wilde, 1964; Beamer and Wilde, 1973). Assuming that the points in the series of trials are so densely distributed that several at a time are in the neighborhood of a local minimum, then the length of the interval of uncertainty is the same as the distance between the two points in the neighborhood of the smallest value of $F(x)$. The number of necessary trials can thus get very large unless one has at least some idea of whereabouts the desired minimum is situated. In practice one must limit investigation of the objective function to a finite interval $[a, b]$. It is obvious, and it can be proved theoretically, that the optimal choice of all simultaneous search methods is the one in which the trial points are evenly distributed over the interval $[a, b]$ (Boas, 1962, 1963a–d).

If N equidistant points are used, the interval of uncertainty is of length

$$\ell_N = \frac{2}{N+1}\,(b-a)$$

and the effectiveness takes the value

$$\eta = \frac{2}{N+1}$$

Put another way: To be sure of achieving an accuracy of $\varepsilon > 0$, the equidistant search (also called *lattice, grid,* or *tabulation method*) requires N trials, where

$$\frac{2(b-a)}{\varepsilon} - 1 < N \leq \frac{2(b-a)}{\varepsilon}\,, \qquad N \text{ integer} \tag{3.6}$$

Apart from the requirements that the chosen interval $[a, b]$ should contain the absolute minimum being sought and that N should be big enough in relation to the "wavyness" of the objective function, no further conditions need to be fulfilled in order for the grid method to succeed.

Even more effective search schemes can be devised if the objective function is unimodal in the interval $[a, b]$. Wilde and Beightler (1967) describe a procedure, using evenly distributed pairs of points, which is also referred to as a simultaneous *dichotomous search.* The distance δ between two points of a pair must be chosen to be sufficiently large that their objective function values are different. As $\delta \rightarrow 0$ the dichotomous search with an even number of trials (*even block search*) is the best. The number of trials required is

$$\frac{2(b-a)}{\varepsilon} - 2 < N \leq \frac{2(b-a)}{\varepsilon} - 1\,, \qquad N \text{ integer} \tag{3.7}$$

This is one less than for equidistant searches with the same accuracy requirement. Such a scheme is referred to as optimal in the sense of the *δ-minimax* concept. Investigations of arrangements of trials also for uneven numbers (*odd block search*), with non-vanishing δ, can be found in Wilde (1966) and Wilde and Beightler (1967).

The one dimensional search procedures dealt with so far, which strive to reduce the interval of uncertainty, are called by Wilde (1964) *direct elimination procedures.* As a second category one can consider the various *interpolation methods.* These can be much more effective, but only when the objective function $F(x)$ satisfies certain "smoothness" conditions, or can be sufficiently well approximated in the interval under consideration by a polynomial $P(x)$. In this case, instead of the minimum of $F(x)$, a zero of $P_x(x) = dP(x)/dx$ is determined. The number of points at which the objective function must be investigated depends on the order of the chosen polynomial and on the type of information at the points that determine it. Besides the values of the objective function itself, consideration is given to its first, second, and, less frequently, higher derivatives. In general, no exact statements can be made regarding the quality of the approximation to the desired minimum for a given number of trials. Details of the various methods of locating real zeros of rational functions of one variable, such as regula falsi, Lagrangian and Newtonian interpolation, can be found in books on practical or numerical mathematics under the heading of non-linear equations: e.g., Booth (1967), Faddejew and Faddejewa (1973), Saaty and Bram (1964), Traub (1964), Zurmühl (1965), Walsh (1966), Ralston and Wilf (1967, 1969), Householder (1970), Ortega and Rheinboldt (1970), and Brent (1973). Although from a fundamental point of view interpolation methods represent indirect optimization procedures, they are of interest here as *line search* strategies, especially when they are applied iteratively and obtain information about derivatives from function values.

3.1.2 Sequential Methods

If the trials for determining a minimum can be made sequentially, the intermediate results retained at a given time can be used to locate the next trial of the sequence more favorably than would be possible without this information. With the digital computers usually available nowadays, which work in a serial way, one is actually obliged to execute all steps one after the other.

Sequential methods, in which the solution is approached in a stepwise, or *iterative*, manner, are advantageous here. The evaluation of the intermediate results and prediction of favorable conditions for the next trial presupposes a more or less precise *internal model* of the objective function; the better the model corresponds to reality, the better will be the results of the interpolation and extrapolation processes. The simplest assumption is that the objective function is unimodal, which means that local minima also always represent global minima. On this basis a number of sequential interval-dividing procedures have been constructed (Sect. 3.1.2.2). Iterative interpolation methods demand more "smoothness" of the objective function (Sect. 3.1.2.3). In the former case it is necessary, in the latter useful, to determine at the outset a suitable interval, $[a^{(0)}, b^{(0)}]$, in which the desired extremum lies (Sect. 3.1.2.1).

3.1.2.1 Boxing in the Minimum

If there are no clues as to whereabouts the desired minimum might be situated, one can start with two points $x^{(0)}$ and $x^{(1)} = x^{(0)} + s$ and determine the objective function there. If $F(x^{(1)}) < F(x^{(0)})$ one proceeds in the chosen direction keeping the same step length:

$$x^{(k+1)} = x^{(k)} + s$$

until

$$F(x^{(k+1)}) > F(x^{(k)})$$

If, however, $F(x^{(1)}) > F(x^{(0)})$, one chooses the opposite direction:

$$x^{(2)} = x^{(0)} - s$$

and

$$x^{(k+1)} = x^{(k)} - s\,, \qquad \text{for } k \geq 2$$

similarly, until a step past the minimum is taken, one has thus determined the minimum of the unimodal function to within an uncertainty interval of length $2\,s$ (Beveridge and Schechter, 1970).

In numerical optimization problems the values of the variables often run through several powers of 10, or alternatively they must be precisely determined at many points. In this case the boxing-in method with a very small fixed step length is too costly. Box, Davies, and Swann (1969) therefore suggest starting with an initial step length $s^{(0)}$ and doubling it at each successful step. Their recursion formula is as follows:

$$x^{(k+1)} = x^{(0)} + 2^k\, s^{(0)}$$

It is applied as long as $F(x^{(k+1)}) \leq F(x^{(k)})$ holds. As soon as $F(x^{(k+1)}) > F(x^{(k)})$ is registered, however, $b^{(0)} = x^{(k+1)}$ is set as the upper bound to the interval and the starting point $x^{(0)}$ is returned to. The lower bound $a^{(0)}$ is found by a corresponding process with negative step lengths going in the opposite direction. In this way a starting interval $[a^{(0)}, b^{(0)}]$ is obtained for the one dimensional search procedure to be described below. It can happen, because of the convention for equality of two function values, that the search for a bound to the interval does not end if the objective function reaches a constant horizontal level. It is therefore useful to specify a maximum step length that may not be exceeded.

The boxing-in method has also been proposed occasionally as a one dimensional optimization strategy (Rosenbrock, 1960; Berman, 1966) in its own right. In order not to waste too many trials far from the target when the accuracy requirement is very high, it is useful to start with relatively large steps. Each time a loop ends with a failure the step length is reduced by a factor less than 0.5, e.g., 0.25. If the above rules for increasing and reducing the step lengths are combined, a very flexible procedure is obtained. Dixon (1972a) calls it the *success/failure routine*. If a starting interval $[a^{(0)}, b^{(0)}]$ is already at hand, however, there are significantly better strategies for successively reducing the size of the interval.

3.1.2.2 Interval Division Methods

If an equidistant division method is applied repeatedly, the interval of uncertainty is reduced at each step by the same factor α, and thus for k steps by α^k. This exponential progression is considerably stronger than the linear dependence of the value of α on the number of trials per step. Thus as few simultaneous trials as possible would be used. A comparison of two schemes, with two and three simultaneous trials, shows that except in the first loop, only two new objective function values must be obtained at a time in both cases, since of three trial points in one step, one coincides with a point of the previous step. The total number of trials required with sequential application of the equidistant three point scheme is

$$1 + \frac{2 \log\left(\frac{b-a}{\varepsilon}\right)}{\log 2} < N \le 3 + \frac{2 \log\left(\frac{b-a}{\varepsilon}\right)}{\log 2}, \qquad N \text{ odd} \tag{3.8}$$

Even better results are provided by the sequential *dichotomous search* with one pair per step. For the limiting case $\delta \to 0$ one obtains

$$\frac{2 \log\left(\frac{b-a}{\varepsilon}\right)}{\log 2} < N \le 2 + \frac{2 \log\left(\frac{b-a}{\varepsilon}\right)}{\log 2}, \qquad N \text{ even} \tag{3.9}$$

Detailed investigations of the influence of δ on various equidistant and dichotomous search schemes can be found in Avriel and Wilde (1966b), and Beamer and Wilde (1970). Of greater interest are the two purely sequential elimination methods described in the following chapters, which only require a single objective function value per step. They require that the objective function be unimodal, otherwise they only guarantee that a local minimum or maximum will be found. Shubert (1972) describes an interval dividing procedure that is able to locate all local extrema, including the global optimum. To use it, however, an upper bound to the slope of the function must be known. The method is rather costly, especially with regard to the storage space required.

3.1.2.2.1 Fibonacci Division. This interval division strategy was introduced by Kiefer (1953). It operates with a series due to Leonardo of Pisa, which bears his pseudonym *Fibonacci*:

$$f_0 = f_1 = 1$$

$$f_k = f_{k-1} + f_{k-2}\,, \qquad \text{for } k \geq 2$$

An initial interval $[a^{(0)}, b^{(0)}]$ is required, containing the extremum together with a number N, which represents the total number of intended interval divisions. If the general interval is called $[a^{(k)}, b^{(k)}]$, the lengths

$$s^{(k)} = t^{(k)}\,(b^{(k)} - a^{(k)}) \; = \; (b^{(k+1)} - a^{(k+1)})$$

are subtracted from its ends, with the reduction factor

$$t^{(k)} = \frac{f_{N-k-1}}{f_{N-k}} \tag{3.10}$$

giving

$$c^{(k)} = a^{(k)} + s^{(k)}$$
$$d^{(k)} = b^{(k)} - s^{(k)}$$

The values of the objective function at $c^{(k)}$ and $d^{(k)}$ are compared and whichever subinterval contains the better (in a minimum search, lower) value is taken as defining the interval for the next step.
If

$$F(d^{(k)}) < F(c^{(k)})$$

then

$$a^{(k+1)} = a^{(k)}$$

and

$$b^{(k+1)} = c^{(k)}$$

If

$$F(d^{(k)}) > F(c^{(k)})$$

then

$$a^{(k+1)} = d^{(k)}$$

and

$$b^{(k+1)} = b^{(k)}$$

A consequence of the Fibonacci series is that, except for the first interval division, at all of the following steps one of the two new points $c^{(k+1)}$ and $d^{(k+1)}$ is always already known.
If

$$F(d^{(k)}) < F(c^{(k)})$$

then

$$c^{(k+1)} = d^{(k)}$$

and if

$$F(d^{(k)}) > F(c^{(k)})$$

then

$$d^{(k+1)} = c^{(k)}$$

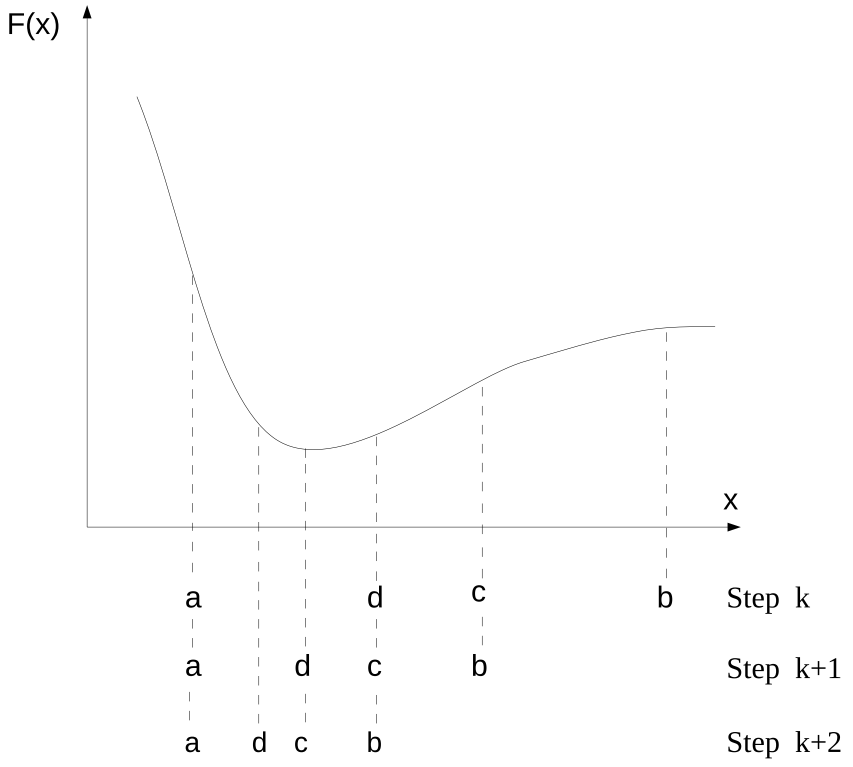


Figure 3.2: Interval division in the Fibonacci search

so that each time only one new value of the objective function needs to be obtained.

Figure 3.2 illustrates two steps of the procedure. The process is continued until $k = N - 2$. At the next division, because $f_2 = 2\,f_1$, $d^{(k)}$ and $c^{(k)}$ coincide. A further interval reduction can only be achieved by slightly displacing one of the test points. The displacement δ must be at least big enough for the two objective function values to still be distinguishable. Then the remaining interval after N trials is of length

$$\ell_N = \frac{1}{f_N}\,(b^{(0)} - a^{(0)}) + \delta$$

As $\delta \rightarrow 0$ the effectiveness tends to f_N^{-1}. Johnson (1956) and Kiefer (1957) show that this value is optimal in the sense of the δ-minimax concept, according to which the Fibonacci search is the best of all sequential interval division procedures. However, by taking account of the δ displacement, not only at the last but at all the steps, Oliver and Wilde (1964) give a recursion formula that for the same number of trials yields a slightly smaller residual interval. Avriel and Wilde (1966a) provide a proof of optimality. If one has a priori information about the structure of the objective function it can be exploited to advantage (Gal, 1971) in order to reduce further the number of trials. Overholt (1967a, 1973) suggests that in general there is no a priori information available to fix δ suitably, and it is therefore better to omit the final division using a displacement rule and to choose N one bigger from the start. In order to obtain the minimum with accuracy $\varepsilon > 0$ one should choose N such that

$$f_N > \frac{b^{(0)} - a^{(0)}}{\varepsilon} \geq f_{N-1}$$

Then the effectiveness of the procedure becomes

$$\eta = \frac{2}{f_{N+1}}$$

and since (Lucas, 1876)

$$f_N = \frac{1}{\sqrt{5}} \left[\left(\frac{1+\sqrt{5}}{2} \right)^{N+1} - \left(\frac{1-\sqrt{5}}{2} \right)^{N+1} \right] \simeq \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^{N+1}$$

the number of trials is approximately

$$N \simeq \frac{\log \frac{b^{(0)}-a^{(0)}}{\varepsilon} + \log \sqrt{5}}{\log \frac{1+\sqrt{5}}{2}} \quad \sim \quad \log \frac{b^{(0)} - a^{(0)}}{\varepsilon} \tag{3.11}$$

Overholt (1965) shows by means of numerical tests that the procedure must often be terminated prematurely as $F(d^{(k+1)})$ becomes equal to $F(c^{(k+1)})$, for example because of computing with a finite number of significant figures. Further divisions of the interval of uncertainty are then pointless.

For the boxing-in method of determining the initial interval one would fix an initial step length of about $10\,\varepsilon$ and a maximum step length of about $5 \cdot 10^9\,\varepsilon$, so that for a 36-bit computer the number range of integers is not exceeded by the largest required Fibonacci number. Finally, two further applications of the Fibonacci procedure may be mentioned. By reversing the scheme, Wilde and Beightler (1967) obtain a method of boxing in the minimum. Kiefer (1957) shows how to proceed if values of the objective function can only be obtained at discrete, not necessarily equidistant, points. More about such *lattice search* problems can be found in Wilde (1964), and Beveridge and Schechter (1970).

3.1.2.2.2 The Golden Section. It can sometimes be inconvenient to have to specify in advance the number of interval divisions. In this case Kiefer (1953) and Johnson (1956) propose, instead of the reduction factor $t^{(k)}$, which varies with the iteration number in the Fibonacci search, a constant factor

$$t = \frac{2}{1+\sqrt{5}} \simeq 0.618\,, \quad \text{(positive root of: } t^2 + t = 1\text{)} \tag{3.12}$$

For large $N - k$, $t^{(k)}$ reduces to t. In addition, t is identical to the ratio of lengths a to b, which is obtained by dividing a total length of $a + b$ into two pieces such that the smaller, a, has the same ratio to the larger, b, as the larger to the total. This *harmonic division* (after Euclid) is also known as the *golden section*, which gave the procedure its name (Wilde, 1964). After N function calls the uncertainty interval is of length

$$\ell_N = t^{N-1}\,(b^{(0)} - a^{(0)})$$

For the limiting case $N \rightarrow \infty$, since

$$\lim_{N\rightarrow\infty} (t^{N-1} f_N) = 1.17$$

the number of trials compared to the δ Fibonacci procedure is about 17% higher. Compared to the Fibonacci search without δ displacement, since

$$\lim_{N\rightarrow\infty} \left(\frac{1}{2}\, t^{N-1} f_{N+1}\right) \simeq 0.95$$

the number of trials is about 5% lower. It should further be noted that, when using the Fibonacci method on digital computers, the Fibonacci numbers must first be generated, or a sufficient number of them must be provided and stored. The number of trials needed for a sequential golden section is

$$N = \left\lceil \frac{\log \frac{b^{(0)}-a^{(0)}}{\varepsilon}}{\log\ t} \right\rceil - 1 \quad \sim \quad \log \frac{b^{(0)}-a^{(0)}}{\varepsilon} \tag{3.13}$$

Other properties of the iteration sequence, including the criterion for termination at equal function values, are the same as those of the method of interval division according to Fibonacci numbers. Further details can be found, for example, in Avriel and Wilde (1968). Complete programs for the interval division procedures have been published by Pike and Pixner (1965), and Overholt (1967b,c) (see also Boothroyd, 1965; Pike, Hill, and James, 1967; Overholt, 1967a).

3.1.2.3 Interpolation Methods

In many cases one is dealing with a continuous function, the minimum of which is to be determined. If, in addition to the value of the objective function, its slope can be specified everywhere, many methods can be derived that may converge faster than the optimal elimination methods. One of the oldest schemes is based on the procedure named after Bolzano for determining the zeros of a function. Assuming that one has two points at which the slopes of the objective function have opposite signs, one bisects the interval between them and determines the slope at the midpoint. This replaces the interval end point, which has a slope of the same sign. The procedure can then be repeated iteratively. At each trial the interval is halved. If the slope has to be calculated from the difference of two objective function values, the *bisection or midpoint strategy* becomes the sequential *dichotomous search.* Avriel and Wilde (1966b) propose, as a variant of the Bolzano search, evaluating the slope at two points in the interval so as to increase the reduction factor. They show that their *diblock strategy* is slightly superior to the dichotomous search.

If derivatives of the objective function are available, or at least if it can be assumed that these exist, i.e., the function $F(x)$ is continuous and differentiable, far better strategies for the minimum search can be devised. They determine analytically the minimum of a trial function that coincides with the objective function, and possibly also its derivatives, at selected argument values. One distinguishes linear, quadratic, and cubic models according to the order of the trial polynomial. Polynomials of higher order are virtually never used.

They require too much information about the function $F(x)$. Furthermore, it turns out that in contrast to all the methods referred to so far such strategies do not always converge, for reasons other than rounding error.

3.1.2.3.1 Regula Falsi Iteration. Given two points $a^{(k)}$ and $b^{(k)}$, with their function values $F(a^{(k)})$ and $F(b^{(k)})$, the simplest approximation formula for a zero $c^{(k)}$ of $F(x)$ is

$$c^{(k)} = a^{(k)} - F(a^{(k)})\,\frac{b^{(k)} - a^{(k)}}{F(b^{(k)}) - F(a^{(k)})}$$

This technique, known as *regula falsi* or *regula falsorum*, predicts the position of the zero correctly if $F(x)$ depends linearly on x. For one dimensional minimization it can be applied to find a zero of $F_x(x) = dF(x)/dx$:

$$c^{(k)} = a^{(k)} - F_x(a^{(k)})\,\frac{b^{(k)} - a^{(k)}}{F_x(b^{(k)}) - F_x(a^{(k)})} \tag{3.14}$$

The underlying model here is a second order polynomial with linear slope. If $F_x(a^{(k)})$ and $F_x(b^{(k)})$ have opposite sign, $c^{(k)}$ lies between $a^{(k)}$ and $b^{(k)}$. If $F_x(c^{(k)}) \neq 0$, the procedure can be continued iteratively by using the reduced interval $[a^{(k+1)}, b^{(k+1)}] = [a^{(k)}, c^{(k)}]$ if $F_x(c^{(k)})$ and $F_x(b^{(k)})$ have the same sign, or using $[a^{(k+1)}, b^{(k+1)}] = [c^{(k)}, b^{(k)}]$ if $F_x(c^{(k)})$ and $F_x(a^{(k)})$ have the same sign. If $F_x(a^{(k)})$ and $F_x(b^{(k)})$ have the same sign, $c^{(k)}$ must lie outside $[a^{(k)}, b^{(k)}]$. If $F_x(c^{(k)})$ has the same sign again, $c^{(k)}$ replaces the argument value at which $|F_x|$ is greatest. This extrapolation is also called the *secant method.* If $F_x(c^{(k)})$ has the opposite sign, one can continue using regula falsi to interpolate iteratively. As a termination criterion one can apply $F_x(c^{(k)}) = 0$ or $|F_x(c^{(k)})| \leq \varepsilon$, $\varepsilon > 0$. A minimum can only be found reliably in this way if the starting point of the search lies in its neighborhood. Otherwise the iteration sequence can also converge to a maximum, at which, of course, the slope also goes to zero if $F_x(x)$ is continuous.

Whereas in the Bolzano interval bisection method only the sign of the function whose zero is sought needs to be known at the argument values, the regula falsi method also makes use of the magnitude of the function. This extra information should enable it to converge more rapidly. As Ostrowski (1966) and Jarratt (1967, 1968) show, for example, this is only the case if the function corresponds closely enough to the assumed model. The simpler bisection method is better, even optimal (as a zero method in the minimax sense), if the function has opposite signs at the two starting points, is not linear and not convex. In this case the linear interpolation sometimes converges very slowly. According to Stanton (1969), a cubic interpolation as a line search in the eccentric quadratic case often yields even worse results. Dixon (1972a) names two variants of the regula falsi recursion formula, but it is not known whether they lead to better convergence. Fox (1971) proposes a combination of the Bolzano method with the linear interpolation. Dekker (1969) (see also Forsythe, 1969) accredits this procedure with better than linear convergence. Even greater reliability and speed is attributed to the algorithm of Brent (1971), which follows Dekker's method by a quadratic interpolation process as soon as the latter promises to be successful.

It is inconvenient when dealing with minimization problems that the derivatives of the function are required. If the slopes are obtained from function values by a difference method, difficulties can arise from the finite accuracy of such a process. For this reason Brent (1973) combines regula falsi iteration with division according to the golden section. Further variations can be found in Schmidt and Trinkaus (1966), Dowell and Jarratt (1972), King (1973), and Anderson and Björck (1973).

3.1.2.3.2 Newton-Raphson Iteration. Newton's interpolation formula for improving an approximate solution $x^{(k)}$ to the equation $F(x) = 0$ (see for example Madsen, 1973)

$$x^{(k+1)} = x^{(k)} - \frac{F(x^{(k)})}{F_x(x^{(k)})}$$

uses only one argument value, but requires the value of the derivative of the function as well as the function itself. If $F(x)$ is linear in x, the zero is correctly predicted here, otherwise an improved approximation is obtained at best, and the process must be repeated. Like regula falsi, Newton's recursion formula can also be applied to determining $F_x(x) = 0$, with of course the reservations already stated. The so-called *Newton-Raphson rule* is then

$$x^{(k+1)} = x^{(k)} - \frac{F_x(x^{(k)})}{F_{xx}(x^{(k)})} \tag{3.15}$$

If $F(x)$ is not quadratic, the necessary number of iterations must be made until a *termination criterion* is satisfied. Dixon (1972a) for example uses the condition $|x^{(k+1)} - x^{(k)}| < \varepsilon$. To set against the advantages that only one argument value is required, and, for quadratic objective functions, one iteration is sufficient to find a point where $F_x(x) = 0$, there are several disadvantages:

- If the derivatives F_x and F_{xx} are obtained approximately by numerical differentiation, the efficiency of the procedure is worsened not only by rounding errors but also by inaccuracies in the approximation. This is especially true in the neighborhood of the minimum being sought, since F_x becomes vanishingly small there.
- Minima, maxima, and saddle points are not distinguished. The starting point $x^{(0)}$ must already be located as close as possible to the minimum being sought.
- If the objective function is of higher than second order, the Newton-Raphson iteration can diverge. The condition for convergence towards a minimum is that $F_{xx}(x) > 0$ for all x.

3.1.2.3.3 Lagrangian Interpolation. Whereas regula falsi and Newton-Raphson iteration attempt to approximate the minimum using information about the derivatives of the objective function at the argument values and can therefore be classified as indirect optimization methods, in Lagrangian interpolation only values of the objective function itself are required. In its general form the procedure consists of *fitting* a pth order polynomial through $p + 1$ points (Zurmühl, 1965). One usually uses three argument values and a parabola as the model function (*quadratic interpolation*). Assuming that the three points are $a^{(k)} < b^{(k)} < c^{(k)}$, with the objective function values $F(a^{(k)}), F(b^{(k)})$, and $F(c^{(k)})$, the trial parabola $P(x)$ has a vanishing first derivative at the point

$$d^{(k)} = \frac{1}{2} \frac{[(b^{(k)})^2 - (c^{(k)})^2]\, F(a^{(k)}) + [(c^{(k)})^2 - (a^{(k)})^2]\, F(b^{(k)}) + [(a^{(k)})^2 - (b^{(k)})^2]\, F(c^{(k)})}{[b^{(k)} - c^{(k)}]\, F(a^{(k)}) + [c^{(k)} - a^{(k)}]\, F(b^{(k)}) + [a^{(k)} - b^{(k)}]\, F(c^{(k)})} \tag{3.16}$$

This point is a minimum only if the denominator is positive. Otherwise $d^{(k)}$ represents a maximum or a saddle point. In the case of a minimum, $d^{(k)}$ is introduced as a new argument value and one of the old ones is deleted:

$$\left.\begin{array}{l} a^{(k+1)} = a^{(k)} \\ b^{(k+1)} = d^{(k)} \\ c^{(k+1)} = b^{(k)} \end{array}\right\} \qquad \left\{\begin{array}{l} \text{if } a^{(k)} < d^{(k)} < b^{(k)} \\ \text{and } F(d^{(k)}) < F(b^{(k)}) \end{array}\right.$$

$$\left.\begin{array}{l} a^{(k+1)} = d^{(k)} \\ b^{(k+1)} = b^{(k)} \\ c^{(k+1)} = c^{(k)} \end{array}\right\} \qquad \left\{\begin{array}{l} \text{if } a^{(k)} < d^{(k)} < b^{(k)} \\ \text{and } F(d^{(k)}) > F(b^{(k)}) \end{array}\right.$$

$$\left.\begin{array}{l} a^{(k+1)} = b^{(k)} \\ b^{(k+1)} = d^{(k)} \\ c^{(k+1)} = c^{(k)} \end{array}\right\} \qquad \left\{\begin{array}{l} \text{if } b^{(k)} < d^{(k)} < c^{(k)} \\ \text{and } F(d^{(k)}) < F(b^{(k)}) \end{array}\right.$$

$$\left.\begin{array}{l} a^{(k+1)} = a^{(k)} \\ b^{(k+1)} = b^{(k)} \\ c^{(k+1)} = d^{(k)} \end{array}\right\} \qquad \left\{\begin{array}{l} \text{if } b^{(k)} < d^{(k)} < c^{(k)} \\ \text{and } F(d^{(k)}) > F(b^{(k)}) \end{array}\right. \tag{3.17}$$

Figure 3.3 shows one of the possible cases.

It is useful to determine the ends of the interval $a^{(0)}$ and $c^{(0)}$ at the start of the one dimensional search by a procedure such as one of those described in Section 3.1.2.1. The third argument value is best positioned at the center of the interval. It can be seen from the recursion formula (Equation (3.17)) that only one value of the objective function needs to be obtained at each iteration, at the current position of $d^{(k)}$. The three points clearly do not in general remain equidistant. When the interpolation formula (Equation (3.16)) predicts a minimum that coincides to the desired accuracy ε with one of the argument values, the search can be terminated. As the result, one will select the smallest objective function value from among the argument values at hand and the computed minimum. There is little prospect of success if the minimum is predicted to lie outside the interval $[a^{(k)}, c^{(k)}]$, at any rate when a bounding procedure of the type described in Section 3.1.2.1 has been applied initially. In this case too the procedure is best terminated. The same holds if a negative denominator in Equation (3.16) indicates a maximum, or if a point of inflection is expected because the denominator vanishes. If the measured objective function values are subject to error, for example in experimental optimization, special precautions must be taken. Hotelling (1941) treats this problem in detail.

How often the interpolation must be repeated until the minimum is sufficiently well approximated cannot be predicted in general; it depends on the level of agreement between

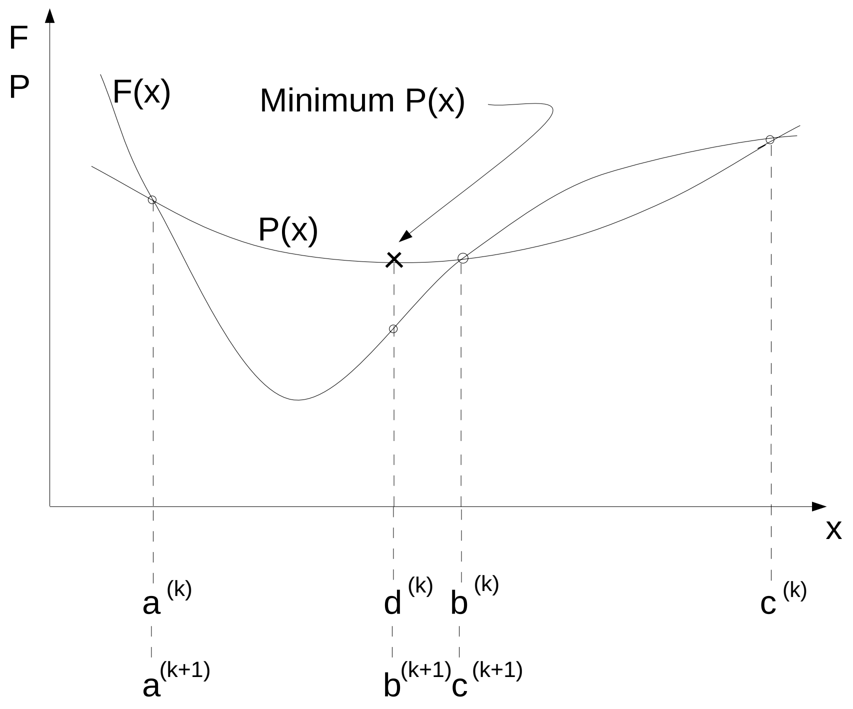


Figure 3.3: Lagrangian quadratic interpolation

the objective function and the trial function. In the most favorable case the objective function is also quadratic. Then one iteration is sufficient. This is why it can be advantageous to use an interpolation method rather than an interval division method such as the optimal Fibonacci search. Dijkhuis (1971) describes a variant of the basic procedure in which four argument values are taken. The two inner ones and each of the outer ones in turn are used for two separate quadratic interpolations. The weighted mean of the two results yields a new iteration point. This procedure is claimed to increase the reliability of the minimum search for non-quadratic objective functions.

3.1.2.3.4 Hermitian Interpolation. If one chooses, instead of a parabola, a third order polynomial as a test function, more information is needed to make it agree with the objective function. Beveridge and Schechter (1970) describe such a *cubic interpolation* procedure. In place of four argument values and associated objective function values, two points $a^{(k)}$ and $b^{(k)}$ are enough, if, in addition to the values of the objective function, values of its slope, i.e., the first order differentials, are available. This Hermitian interpolation is mainly used in conjunction with gradient or quasi-Newton methods, because in any case they require the partial derivatives of the objective function, or they approximate them using finite difference methods.

The interpolation formula is:

$$c^{(k)} = a^{(k)} + (b^{(k)} - a^{(k)}) \frac{w - F_x(a^{(k)}) - z}{2\,w + F_x(b^{(k)}) - F_x(a^{(k)})}$$

where

$$z = \frac{3\,[F(a^{(k)}) - F(b^{(k)})]}{(a^{(k)} - b^{(k)})} - F_x(a^{(k)}) - F_x(b^{(k)}) \tag{3.18}$$

and

$$w = +\sqrt{z^2 - F_x(a^{(k)})\, F_x(b^{(k)})}$$

Recursive exchange of the argument values takes place according to the sign of $F_x(c^{(k)})$ in a similar way to the Bolzano method. It should also be verified here that $a^{(k)}$ and $b^{(k)}$ always bound the minimum. Fletcher and Reeves (1964) use Hermitian interpolation in their conjugate gradient method as a subroutine to approximate a *relative* minimum in specified directions. They terminate the iteration as soon as $|a^{(k)} - b^{(k)}| < \varepsilon$.

As in all interpolation procedures, the speed and reliability of convergence depend on the degree of agreement between the model function and the objective function. Pearson (1969), for example, reports that the Fibonacci search is superior to Hermitian interpolation if the objective function is logarithmic or a polynomial of high order. Guilfoyle, Johnson, and Wheatley (1967) propose a combination of cubic interpolation and either the Fibonacci search or the golden section.

3.2 Multidimensional Strategies

There have been frequent attempts to extend the basic ideas of one dimensional optimization procedures to several dimensions. The *equidistant grid strategy*, also known in the experimental field as the *method of factorial design*, places an evenly meshed grid over the space under consideration and evaluates the objective function at all the nodes.

If n is the dimension of the space under consideration and $N_i\,(i = 1, 2, \ldots, n)$ is the number of discrete values that the variable x_i can take, then the number of combinations to be tested is given by the product

$$N = \prod_{i=1}^{n} N_i \tag{3.19}$$

If $N_i = N_1$ for all $i = 1(1)n$, one obtains $N = N_1^n$. This exponential increase in the number of trials and the computational requirements is what provoked Bellman's now famous *curse of dimensions* (see Wilde and Beightler, 1967). On a traditional computer, which works sequentially, the trials must all be carried out one after another. The computation time therefore increases as $O(c^n)$, in which the constant c depends on the required accuracy and the size of the interval to be investigated.

Proceeding sequentially brought considerable advantages in the one dimensional case if it could only be assumed that the objective function was unimodal. Krolak and Cooper (1963) (see also Krolak, 1968) and Sugie (1964) have given an extension to several dimensions of the Fibonacci search scheme. For $n = 2$, two points are chosen on one of the two coordinate axes within the given interval in the same way as for the usual one dimensional Fibonacci search. The values of this variable are first held constant while two complete Fibonacci searches are made to find the relative optima with respect to the second variable. Both end results are then used to reject one of the values of the first variable that were held constant, and to reduce the size of the interval with respect to this parameter. By analogy, a three dimensional minimization consists of a recursive sequence of two dimensional Fibonacci searches. If the number of function calls to reduce the uncertainty interval $[a_i, b_i]$ sufficiently with respect to the variable x_i is N_i, then the total number N also obeys Equation (3.19). The advantage compared to the grid method is simply that N_i depends logarithmically on the ratio of initial interval size to accuracy (see Equation (3.11)). Aside from the fact that each variable must be suitably fixed in advance, and that the unimodality requirement of the objective function only guarantees that local minima are approached, there is furthermore no guarantee that a desired accuracy will be reached within a finite number of objective function calls (Kaupe, 1964).

Other elimination procedures have been extended in a similar way to the multivariable case, such as, for example, the dichotomous search (Wilde, 1965) and a sequential boxing-in method (Berman, 1969). In each case the effort rises exponentially with the number of variables. Another elimination concept for the multidimensional case, the method of *contour tangents*, is due to Wilde (1963) (see also Beamer and Wilde, 1969). It requires, however, the determination of gradient vectors. Newman (1965) indicates how to proceed in the two dimensional case, and also for discrete values of the variables (lattice search). He requires that $F(x)$ be convex and unimodal. Then the cost should only increase linearly with the number of variables. For $n \geq 3$, however, no applications of the contour tangent method are as yet known.

Transferring interpolation methods to the n-dimensional case means transforming the original minimum problem into a series of problems, in the form of a set of equations to be solved. As non-linear equations can only be solved iteratively, this procedure is limited to the special case of linear interpolation with quadratic objective functions. Practical algorithms based on the regula falsi iteration can be found in Schmidt and Schwetlick (1968) and Schwetlick (1970). The procedure is not widely used as a minimization method (Schmidt and Vetters, 1970). The slopes of the objective function that it requires are implicitly calculated from function values. The *secant method* described by Wolfe (1959b) for solving a system of non-linear equations also works without derivatives of the functions. From $n + 1$ current argument values, it extracts the required information about the structure of the n equations.

Just as the transition from simultaneous to sequential one dimensional search methods reduces the effort required at the expense of global convergence, so each further acceleration in the multidimensional case is bought by a reduction in reliability. High convergence rates are achieved by gathering more information and interpreting it in the form of a model of the objective function. If assumptions and reality agree, then this procedure is successful; if they do not agree, then extrapolations lead to worse predictions and possibly even to abandoning an optimization strategy. Figure 3.4 shows the contour diagram of a smooth two parameter objective function.

All the strategies to be described assume a degree of smoothness in the objective function. They do not converge with certainty to the global minimum but at best to one of the local minima, or sometimes only to a saddle point.

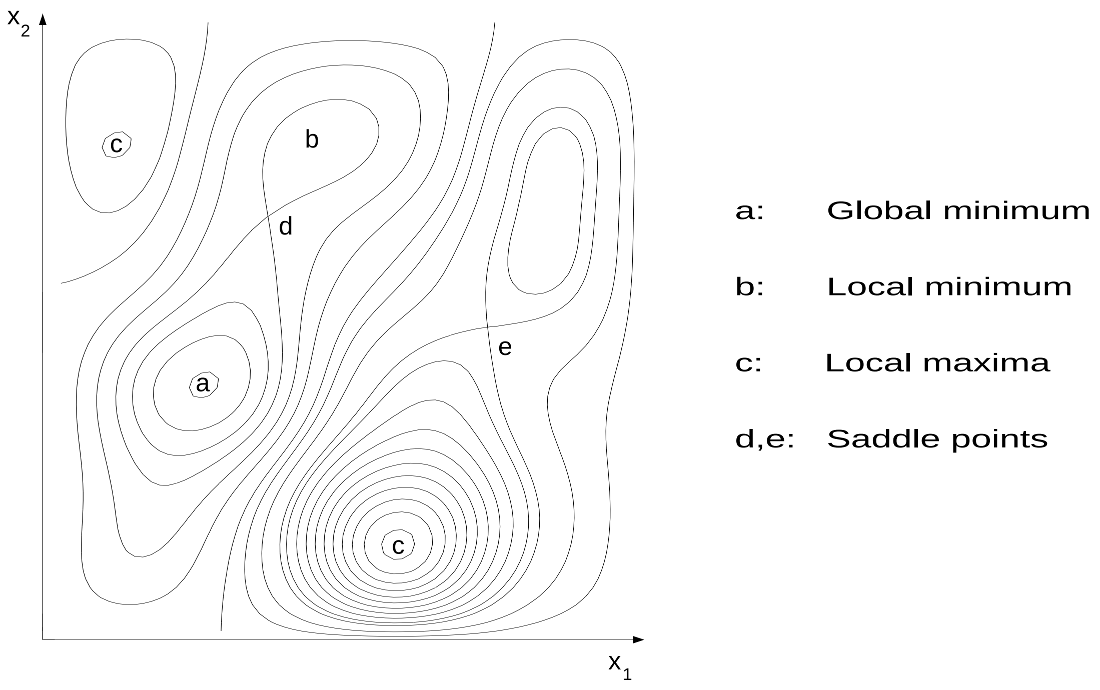


Figure 3.4: Contour lines of a two parameter function $F(x_1, x_2)$

Various methods are distinguished according to the kind of information they need, namely:

- *Direct search methods*, which only need objective function values $F(x)$
- *Gradient methods*, which also use the first partial derivatives $\nabla F(x)$ (first order strategies)
- *Newton methods*, which in addition make use of the second partial derivatives $\nabla^2 F(x)$ (second order strategies)

The emphasis here will be placed on derivative-free strategies, that is on direct search methods, and on such higher order procedures as glean their required information about derivatives from a sequence of function values. The *recursion scheme* of most multidimensional strategies is based on the formula:

$$x^{(k+1)} = x^{(k)} + s^{(k)} \, v^{(k)} \tag{3.20}$$

They differ from each other with regard to the choice of *step length* $s^{(k)}$ and *search direction* $v^{(k)}$, the former being a scalar and the latter a vector of unit length.

3.2.1 Direct Search Strategies

Direct search strategies do without constructing a model of the objective function. Instead, the directions, and to some extent also step lengths, are fixed heuristically, or by a scheme of some sort, not always in an optimal way under the assumption of a specified internal model. Thus the risk is run of not being able to improve the objective function value at each step. Failures must accordingly be planned for, if something can also be "learned" from them. This trial character of search strategies has earned them the name of *trial-and-error methods*. The most important of them that are still in current use will be presented in the following chapters. Their attraction lies not in theoretical proofs of convergence and rates of convergence, but in their simplicity and the fact that they have proved themselves in practice. In the case of convex or quadratic unimodal objective functions, however, they are generally inferior to the first and second order strategies to be described later.

3.2.1.1 Coordinate Strategy

The oldest of multidimensional search procedures trades under a variety of names (e.g., successive variation of the variables, relaxation, parallel axis search, univariate or univariant search, one-variable-at-a-time method, axial iteration technique, cyclic coordinate ascent method, alternating variable search, sectioning method, Gauss-Seidel strategy) and manifests itself in a large number of variations.

The basic idea of the *coordinate strategy*, as it will be called here, comes from linear algebra and was first put into practice by Gauss and Seidel in the single step relaxation method of solving systems of linear equations (see Ortega and Rockoff, 1966; Ortega and Rheinboldt, 1967; VanNorton, 1967; Schwarz, Rutishauser, and Stiefel, 1968). As an optimization strategy it is attributed to Southwell (1940, 1946) or Friedmann and Savage (1947) (see also D'Esopo, 1959; Zangwill, 1969; Zadeh, 1970; Schechter, 1970).

The parameters in the iteration formula (3.20) are varied in turn individually, i.e., the search directions are fixed by the rule:

$$v^{(k)} = e_\ell \,, \qquad \text{with } \ell = \begin{cases} n \,, & \text{if } k = p\,n \,,\ p \text{ integer} \\ k \,(\mathrm{mod}\ n) \,, & \text{otherwise} \end{cases}$$

where e_ℓ is the unit vector whose components have the value zero for all $i \neq \ell$, and unity for $i = \ell$. In its simplest form the coordinate strategy uses a constant step length $s^{(k)}$. Since, however, the direction to the minimum is unknown, both positive and negative values of $s^{(k)}$ must be tried. In a first and easy improvement on the basic procedure, a successful step is followed by further steps in the same direction, until a worsening of the objective function is noted. It is clear that the choice of step length strongly influences the number of trials required on the one hand and the accuracy that can be achieved in the approximation on the other.

One can avoid the problem of the choice of step length most effectively by using a *line search* method each time to locate the *relative* optimum in the chosen direction. Besides the interval division methods, the Fibonacci search and the golden section, Lagrangian interpolation can also be used, since all these procedures work without knowledge of the partial derivatives of the objective function. A further strategy for boxing in the minimum must be added, in order to establish a suitable starting interval for each one dimensional minimization.

The algorithm can be described as follows:

Step 0: (Initialization)
Establish a starting point $x^{(0,0)}$ and choose an accuracy bound $\varepsilon > 0$ for the one dimensional search.
Set $k = 0$ and $i = 1$.

Step 1: (Boxing in the minimum)
Starting from $x^{(k,i-1)}$ with an initial step length $s = s_{min}$ (e.g., $s_{min} = 10\,\varepsilon$), box in the minimum in the direction $\pm e_i$.
Double the step length at each successful trial, as long as $s < s_{max}$ (e.g., $s_{max} = 5 \cdot 10^9\,\varepsilon$).
Define the interval limits $a_i^{(k)}$ and $b_i^{(k)}$.

Step 2: (Line search)
By varying x_i within the interval $[a_i^{(k)}, b_i^{(k)}]$ search for the relative minimum x' with the required accuracy ε (line search with an interval division procedure or an iterative interpolation method):
$F(x') = \min\limits_{s}\{F(x^{(k,i-1)} + s\,e_i)\}$

Step 3: (Check for improvement)
If $F(x') \leq F(x^{(k,i-1)})$, then set $x^{(k,i)} = x'$;
otherwise set $x^{(k,i)} = x^{(k,i-1)}$.

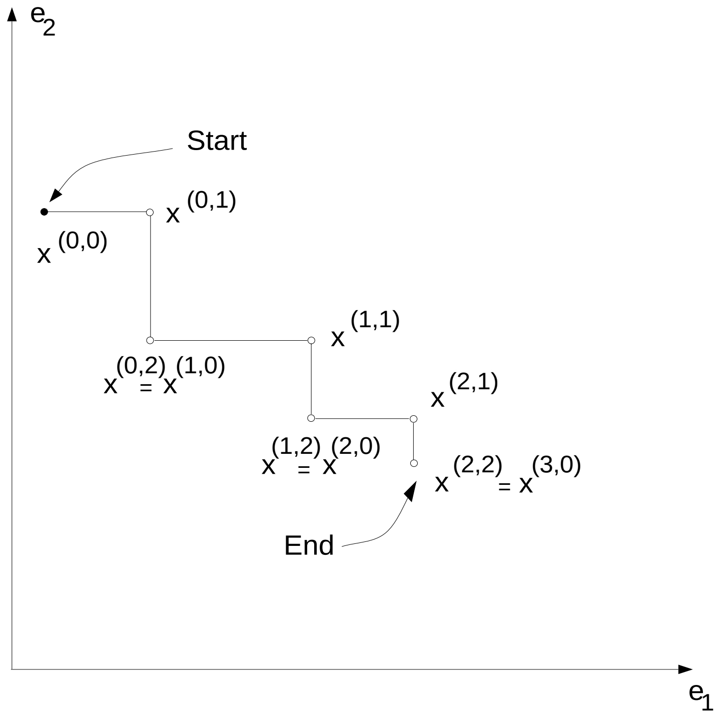
Step 4: (Inner loop over all coordinates)
If $i < n$, increase $i \leftarrow i + 1$ and go to step 1.

Step 5: (Termination criterion, outer iteration loop)
Set $x^{(k+1,0)} = x^{(k,n)}$.
If all $|x_i^{(k+1,0)} - x_i^{(k,0)}| \leq 5\,\varepsilon$ for all $i = 1(1)n$, then end the search;
otherwise increase $k \leftarrow k + 1$, set $i = 1$, and go to step 1.

Figure 3.5 shows a typical sequence of iteration points for $n = 2$ variables.

In theory the coordinate strategy always converges if $F(x)$ has continuous partial derivatives and the line searches are carried out exactly (Kowalik and Osborne, 1968; Schechter, 1968; Ortega and Rheinboldt, 1970). Rapid convergence is only achieved, however, if the contour surfaces $F(x) = const.$ are approximately concentric surfaces of a hypersphere, or, in the case of elliptic contours, if the principle axes almost coincide with the coordinate axes.

If the significant system parameters influence each other (non-separable objective function), then the distances covered by each line search are small without the minimum being within striking distance. This is especially true when the number of variables is large. At discontinuities in the objective function it may happen that improvements in the objective function can only be made in directions that are not along a coordinate axis. In this case the coordinate strategy fails. There is a similar outcome at steep "valleys" of a continuously differentiable function, i.e., if the step lengths that would enable successful



Numbering	Iteration index k	Direction index i	Variable values x_1	x_2
(0)	0	0	0	9
(1)	0	1	3	9
(2)	0	2	3	5
		carried over		
(2)	1	0	3	5
(3)	1	1	7	5
(4)	1	2	7	3
		carried over		
(4)	2	0	7	3
(5)	2	1	9	3
(6)	2	2	9	2
		carried over		
(6)	3	0	9	2

Figure 3.5: Coordinate strategy

convergence are so small that the number of significant figures to which data are handled by the computer is insufficient for the variables to be significantly altered.

Numerical tests with the coordinate strategy show that an exact determination of the relative minima is unnecessary, at least at distances far from the objective. It can even happen that one inaccurate line search can make the next one particularly effective. This phenomenon is exploited in the procedures known as *under- or overrelaxation* (Engeli, Ginsburg, Rutishauser, and Stiefel, 1959; Varga, 1962; Schechter, 1962, 1968; Cryer, 1971). Although the relative optimum is determined as before, either an increment is added on in the same direction or an iteration point is defined on the route between the start and finish of the one dimensional search. The choice of the under- or overrelaxation factor requires assumptions about the structure of the problem. The necessary information is available for the problem of solving systems of linear equations with a positive definite matrix of coefficients, but not for general optimization problems.

Further possible variations of the coordinate strategy are obtained if the sequence of searches parallel to the axes is not made to follow the cyclic scheme. Southwell (1946), for example, always selects either the direction in which the slope of the objective function

$$F_{x_i}(x) = \frac{\partial F(x)}{\partial x_i}$$

is maximum, or the direction in which the largest step can be taken. To evaluate the choice of direction, Synge (1944) uses the ratio $F_{x_i}/F_{x_ix_i}$ of first to second partial derivatives at the point $x^{(k)}$. Whether or not the additional effort for this scheme is worthwhile depends on the particular topology of the contour surface. Adding directions other than parallel to the axes is also often found to accelerate the convergence (Pinsker and Tseitlin, 1962; Elkin, 1968).

Its great simplicity has always made the coordinate strategy attractive, despite its sometimes slow convergence. Rules for handling constraints–not counting here penalty function methods–have been devised, for example, by Singer (1962), Murata (1963), and Mugele (1961, 1962, 1966). Singer's *maze method* departs from the coordinate directions as soon as a constraint is violated and progresses into the feasible region or along the boundary. For this, however, the gradient of the active constraints must be known. Mugele's *poor man's optimizer*, a discrete coordinate strategy without line searches, not only handles active constraints, but can also cope with narrow valleys that do not run parallel to the coordinate axes. In this case diagonal steps are permitted. Similar to this strategy is the direct search method of Hooke and Jeeves, which because it has become very widely used will be treated in detail in the following chapter.

3.2.1.2 Strategy of Hooke and Jeeves: Pattern Search

The *direct pattern search* of Hooke and Jeeves (1961) was originally devised as an automatic experimental strategy (see Hooke, 1957; Hooke and VanNice, 1959). It is nowadays much more widely used as a numerical parameter optimization procedure.

The method by which the direct pattern search works is characterized by two types of move. At each iteration there is an *exploratory move*, which represents a simplified Gauss-Seidel variation with one discrete step per coordinate direction. No line searches are made. On the assumption that the line joining the first and last points of the exploratory move represents an especially favorable direction, an extrapolation is made along it (*pattern move*) before the variables are varied again individually. The extrapolations do not necessarily lead to an improvement in the objective function value. The success of the iteration is only checked after the following exploratory move. The length of the pattern step is thereby increased each time, while the optimal search direction only changes gradually. This pays off to most advantage where there are narrow valleys. An ALGOL implementation of the strategy is due to Kaupe (1963). It was improved by Bell and Pike (1966), as well as by Smith (1969) (see also DeVogelaere, 1968; Tomlin and Smith, 1969). In the first case, the sequence of plus and minus exploratory steps in the coordinate directions is modified to suit the conditions at any instant. The second improvement aims at permitting a retrospective scaling of the variables as the step lengths can be chosen individually to be different from each other.

The algorithm runs as follows:

Step 0: (Initialization)
Choose a starting point $x^{(0,0)} = x^{(-1,n)}$, an accuracy bound $\varepsilon > 0$, and initial step lengths $s_i^{(0)} \neq 0$ for all $i = 1(1)n$ (e.g., $s_1^{(0)} = 1$ if no more plausible values are at hand).
Set $k = 0$ and $i = 1$.

```
Step 1:  (Exploratory move)
         Construct $x' = x^{(k,i-1)} + s_i^{(k)} e_i$   (discrete step in positive direction).
         If $F(x') < F(x^{(k,i-1)})$, go to step 2   (successful first trial);
         otherwise replace $x' \leftarrow x' - 2\, s_i^{(k)} e_i$   (discrete step in negative direction).
         If $F(x') < F(x^{(k,i-1)})$, go to step 2   (success);
         otherwise replace $x' \leftarrow x' + s_i^{(k)} e_i$   (back to original situation).

Step 2:  (Retention and switch to next coordinate)
         Set $x^{(k,i)} = x'$.
         If $i < n$, increase $i \leftarrow i + 1$ and go to step 1.

Step 3:  (Test for total failure in all directions)
         If $F(x^{(k,n)}) \geq F(x^{(k,0)})$, set $x^{(k+1,0)} = x^{(k,0)}$ and go to step 9.

Step 4:  (Pattern move)
         Set $x^{(k+1,0)} = 2\, x^{(k,n)} - x^{(k-1,n)}$   (extrapolation),
         and $s_i^{(k+1)} = s_i^{(k)} \,\mathrm{sign}(x_i^{(k,n)} - x_i^{(k-1,n)})$ for all $i = 1(1)n$.
         (This may change the sequence of positive and negative directions
         in the next exploratory move.)
         Increase $k \leftarrow k + 1$ and set $i = 1$.
         (Observe: There is no success control of the pattern move so far.)

Step 5:  (Exploration after extrapolation)
         Construct $x' = x^{(k,i-1)} + s_i^{(k)} e_i$.
         If $F(x') < F(x^{(k,i-1)})$, go to step 6;
         otherwise replace $x' \leftarrow x' - 2\, s_i^{(k)} e_i$.
         If $F(x') < F(x^{(k,i-1)})$, go to step 6;
         otherwise replace $x' \leftarrow x' + s_i^{(k)} e_i$.

Step 6:  (Inner loop over coordinates)
         Set $x^{(k,i)} = x'$.
         If $i < n$, increase $i \leftarrow i + 1$ and go to step 5.

Step 7:  (Test for failure of pattern move)
         If $F(x^{(k,n)}) \geq F(x^{(k-1,n)})$   (back to position before pattern move),
         set $x^{(k+1,0)} = x^{(k-1,n)}$, $s_i^{(k+1)} = s_i^{(k)}$ for all $i = 1(1)n$, and go to step 10.

Step 8:  (After successful pattern move, retention and first termination test)
         If $|x_i^{(k,n)} - x_i^{(k-1,n)}| \leq \frac{1}{2}\,|s_i^{(k)}|$ for all $i = 1(1)n$,
         set $x^{(k+1,0)} = x^{(k-1,n)}$ and go to step 9;
         otherwise go to step 4 (for another pattern move).

Step 9:  (Step size reduction and termination test)
         If $|s_i^{(k)}| \leq \varepsilon$ for all $i = 1(1)n$, end the search with result $x^{(k,0)}$;
         otherwise set $s_i^{(k+1)} = \frac{1}{2}\, s_i^{(k)}$ for all $i = 1(1)n$.
```

Step 10: (Iteration loop)
Increase $k \leftarrow k + 1$, set $i = 1$, and go to step 1.

Figure 3.6, together with the following table, presents a possible sequence of iteration points. From the starting point (0), a successful step (1) and (3) is taken in each coordinate direction. Since the end point of this exploratory move is better than the starting point, it serves as a basis for the first extrapolation. This leads to (4). It is not checked here whether or not any improvement over (3) has occurred. At the next exploratory move, from (4) to (5), the objective function value can only be improved in one coordinate direction. It is now checked whether the condition (5) is better than that of point (3). This is the case. The next extrapolation step, to (8), has a changed direction because of the partial failure of the exploration, but maintains its increased length. Now it will be assumed that, starting from (8) with the hitherto constant exploratory step length, no success will be scored in any coordinate direction compared to (8). The comparison with (5) shows that a reduction in the value of the objective function has nevertheless occurred. Thus the next extrapolation to (13) remains the same as the previous one with respect to direction and step length. The next exploratory move leads to a point (15), which although better than (13) is worse than (8). Now there is a return to (8). Only after the exploration again has no success here, are the step lengths halved in order to make further progress possible. The fact that at some points in this case the objective function was tested several times is not typical for $n > 2$.

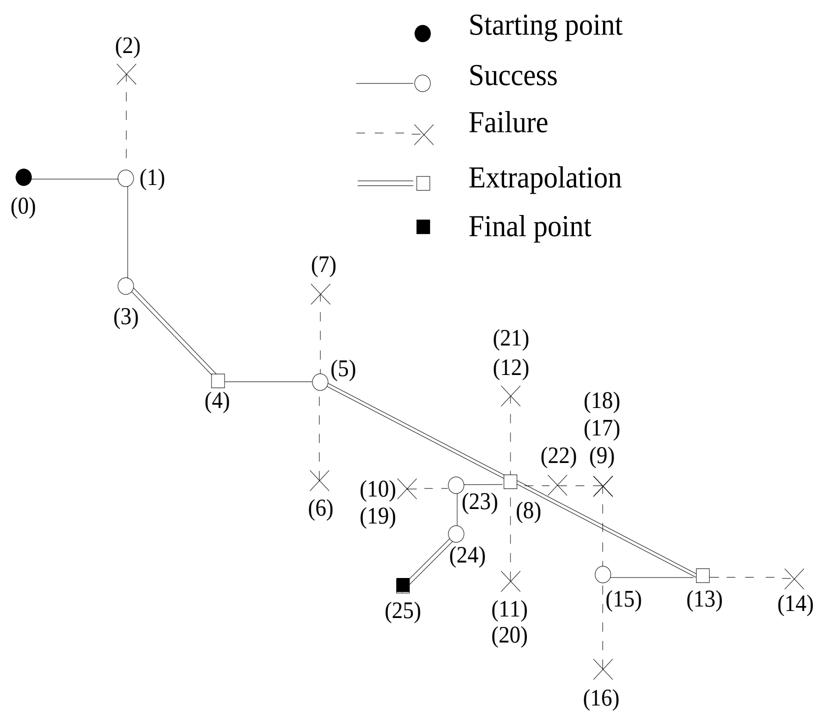


Figure 3.6: Strategy of Hooke and Jeeves

Numbering	Iteration index k	Direction index i	Variable values x_1	x_2	Comparison point	Step lengths s_1	s_2	Remarks
(0)	0	0	0	9	-	2	2	starting point
(1)	0	1	2	9	(0)			success
(2)	0	2	2	11	(1)			failure
(3)	0	2	2	7	(1)			success
(4)	1	0	4	5	-	2	-2	extrapolation
(5)	1	1	6	5	(4),(3)			success, success
(6)	1	2	6	3	(5)			failure
(7)	1	2	6	7	(5)			failure
(8)	2	0	10	3	-,(5)	2	-2	extrapolation, success
(9)	2	1	12	3	(8)			failure
(10)	2	1	8	3	(8)			failure
(11)	2	2	10	1	(8)			failure
(12)	2	2	10	5	(8)			failure
(13)	3	0	14	1	-	2	-2	extrapolation
(14)	3	1	16	1	(13)			failure
(15)	3	1	12	1	(13),(8)			success, failure
(16)	3	2	12	-1	(15)			failure
(17)	3	2	12	3	(15)			failure
(8)	4	0	10	3	-	2	-2	return
(18)	4	1	12	3	(8)			failure
(19)	4	1	8	3	(8)			failure
(20)	4	2	10	1	(8)			failure
(21)	4	2	10	5	(8)			failure
(8)	5	0	10	3	-	1	-1	step lengths halved
(22)	5	1	11	3	(8)			failure
(23)	5	1	9	3	(8)			success
(24)	5	2	9	2	(23),(8)			success, success
(25)	6	0	8	1	-	-1	-1	extrapolation

A proof of convergence of the direct search of Hooke and Jeeves has been derived by Céa (1971); it is valid under the condition that the objective function $F(x)$ is strictly convex and continuously differentiable. The computational operations are very simple and even in unforeseen circumstances cannot lead to invalid arithmetical manipulations such as, for example, division by zero. A further advantage of the strategy is its small storage requirement. It is of order $O(n)$. The selected pattern accelerates the search in valleys, provided they are not sharply bent. The extrapolation steps follow, in an approximate way, the gradient trajectory. However, the limitation of the trial steps to coordinate directions can also lead to a premature termination of the search here, as in the coordinate strategy.

Further variations on the method, which have not achieved such popularity, are due to, among others, Wood (1960, 1962, 1965; see also Weisman and Wood, 1966; Weisman, Wood, and Rivlin, 1965), Emery and O'Hagan (1966; *spider method*), Fend and Chandler (1961; *moment rosetta search*), Bandler and MacDonald (1969; *razor search*; see also Bandler, 1969a,b), Pierre (1969; *bunny-hop search*), Erlicki and Appelbaum (1970), and Houston and Huffman (1971). A more detailed enumeration of older methods can be found in Lavi and Vogl (1966). Some of these modifications allow constraints in the form of inequalities to be taken into account directly. Similar to them is a program designed by M. Schneider (see Drenick, 1967). Aside from the fact that in order to use it one must specify which of the variables enter the individual constraints, it does not appear to work very effectively. Excessively long computation times and inaccurate results, especially with many variables, made it seem reasonable to omit M. Schneider's procedure from the strategy comparison (see Chap. 6). The problem of how to take into account constraints in a direct search has also been investigated by Klingman and Himmelblau (1964) and Glass and Cooper (1965). The resulting methods, to a greater or lesser extent, transform the original problem. They have nowadays been superseded by the general penalty function methods. Automatic "optimizators" for on-line optimization of chemical processes, which once were well known under the names *Opcon* (Bernard and Sonderquist, 1959) and *Optimat* (Weiss, Archer, and Burt, 1961), also apply modified versions of the direct search method. Another application is described by Sawaragi at al. (1971).

3.2.1.3 Strategy of Rosenbrock: Rotating Coordinates

Rosenbrock's idea (1960) was to remove the limitation on the number of search directions in the coordinate strategy so that the search steps can move parallel to the axes of a coordinate system that can rotate in the space $\mathbb{R}^n$. One of the axes is set to point in the direction that appears most favorable. For this purpose the experience of successes and failures gathered in the course of the iterations is used in the manner of Hooke and Jeeves' direct search. The remaining directions are fixed normal to the first and mutually orthogonal.

To start with, the search directions comprise the unit vectors

$$v_i^{(0)} = e_i \,, \quad \text{for all } i = 1(1)n$$

Starting from the point $x^{(0,0)}$, a trial is made in each direction with the discrete initial step lengths $s_i^{(0,0)}$ for all $i = 1(1)n$. When a success is scored (including equality of the objective function values), the changed variable vector is retained and the step length is multiplied by a positive factor $\alpha > 1$; for a failure, the vector of variables is left unchanged and the step length is multiplied by a negative factor $-1 < \beta < 0$. Rosenbrock proposes the choice $\alpha = 3$ and $\beta = -0.5$. This process is repeated until at least one success followed (not necessarily immediately) by a failure is registered in each direction. As a rule several cycles must be run through, because if there is a failure in any particular direction the opposite direction is not tested in the same cycle. Following this first part of the search, the coordinate axes are rotated. Rosenbrock uses for this the *orthogonalization* procedure of Gram and Schmidt (see, for example, Birkhoff and MacLane, 1953; Rutishauser, 1966; Nake, 1966). The recursion formulae are as follows:

$$v_i^{(k+1)} = \frac{w_i}{\|w_i\|} \,, \quad \text{for all } i = 1(1)n$$

where

$$w_i = \begin{cases} a_i \,, & \text{for } i = 1 \\ a_i - \sum\limits_{j=1}^{i-1} (a_i^T \, v_j^{(k+1)}) \, v_j^{(k+1)} \,, & \text{for } i = 2(1)n \end{cases} \tag{3.21}$$

and

$$a_i = \sum_{j=i}^{n} d_j^{(k)} \, v_j^{(k)} \,, \quad \text{for all } i = 1(1)n$$

A scalar $d_i^{(k)}$ represents the distance covered in direction $v_i^{(k)}$ in the kth iteration. Thus $v_1^{(k+1)}$ points in the overall successful direction of the step k. It is expected that a particularly large search step can be taken in this direction at the next iteration. The requirement of waiting for at least one success in each direction has the effect that no direction is lost, and the $v_i^{(k)}$ always span the full n-dimensional Euclidean space. The termination rule or convergence criterion is determined by the lengths of the vectors $a_1^{(k)}$ and $a_2^{(k)}$. Before each orthonormalization there is a test whether $\|a_1^{(k)}\| < \varepsilon$ and $\|a_2^{(k)}\| > 0.3 \, \|a_1^{(k)}\|$. When this condition is satisfied in six consecutive iterations, the search is ended. The second condition is designed to ensure that a premature termination of the search does not occur just because the distances covered have become small. More significantly, the requirement is also that the main success direction changes sufficiently rapidly; something that Rosenbrock regards as a sure sign of the proximity of a minimum. As the strategy comparison will show (see Chap. 6), this requirement is often too strong. It even hinders the ending of the procedure in many cases.

In his original publication Rosenbrock has already given detailed rules as to how inequality constraints can be treated. His procedure for doing this can be viewed as a *partial penalty function* method, since the objective function is only altered in the neighborhood of the boundaries. Immediately after each variation of the variables, the objective function value is tested. If the comparison is unfavorable, a failure is registered as in the unconstrained case. For equality or an improvement, however, if the iteration point lies near a boundary of the region, the success criterion changes. For example, for constraints of the form $G_j(x) \geq 0$ for all $j = 1(1)m$, the extended objective function $\tilde{F}(x)$ takes the form (this is one of several suggestions of Rosenbrock):

$$\tilde{F}(x) = F(x) + \sum_{j=1}^{m} \varphi_j(x) \, (f_j - F(x))$$

in which

$$\varphi_j(x) = \begin{cases} 0 \,, & \text{if} \quad G_j(x) \geq \delta \\ 3\,\eta - 4\,\eta^2 + 2\,\eta^3 \,, & \text{if} \quad 0 < G_j(x) < \delta \\ 1 \,, & \text{if} \quad G_j(x) \leq 0 \end{cases} \tag{3.22}$$

and

$$\eta = 1 - \frac{1}{\delta} \, G_j(x)$$

The auxiliary item f_j is the value of the objective function belonging to the last success of the search that did not fall in the region of the jth boundary. As a reasonable value for the boundary zone one can take $\delta = 10^{-4}$. (Rosenbrock sets $\delta = 10^{-4} \, [b_j(x^{(k)}) - a_j(x^{(k)})]$

for constraints of the form $a_j(x) \leq G_j(x) \leq b_j(x)$; this kind of double sided bounding is not always given however). The basis of the procedure is fully described in Rosenbrock and Storey (1966).

Using the notations

x_i	object variables
s_i	step sizes
v_i	direction components
d_i	distances travelled
ν_i	success/failure indications

the extended algorithm of the strategy runs as follows:

```
Step 0: (Initialization)
        Choose a starting point $x^{(0,0)}$ such that $G_j(x^{(0,0)}) \geq \delta > 0$ for all $j = 1(1)m$.
        Choose an accuracy parameter $\varepsilon > 0$
        (Rosenbrock takes $\varepsilon = 10^{-4}$, $\delta = 10^{-4}$).
        Set $v_i^{(0)} = e_i$ for all $i = 1(1)n$.
        Set $k = 0$   (outer loop counter),
            $\mu = 0$   (inner loop counter).
        If there are constraints ($m > 0$),
        set $f_j = F(x^{(0,0)})$ for all $j = 1(1)m$.

Step 1: (Initialization of step sizes, distances travelled, and indicators)
        Set $s_i^{(k,0)} = 0.1$,
            $d_i^{(k)} = 0$, and
            $\nu_i^{(k)} = -1$ for all $i = 1(1)n$.
        Set $\ell = 0$ and $i = 1$.

Step 2: (Trial step)
        Construct $x' = x^{(k,n\ell+i-1)} + s_i^{(k,\ell)}\, v_i^{(k)}$.
        If   $F(x') > F(x^{(k,n\ell+i-1)})$, go to step 6;
        otherwise
        $\text{if } m \begin{cases} = 0, & \text{go to step 5} \\ \neq 0, & \text{set } \tilde{F} = F(x') \text{ and } j = 1. \end{cases}$

Step 3: (Test of feasibility)
        $\text{If } G_j(x') \begin{cases} \leq 0, & \text{go to step 7} \\ \geq \delta, & \text{set } f_j = F(x') \text{ and go to step 4;} \\ \text{otherwise,} & \text{replace } \tilde{F} \leftarrow \tilde{F} + \varphi_j(x')(f_j - \tilde{F}) \text{ as Equation (3.22).} \\ & \text{If } \tilde{F} > F(x^{(k,n\ell+i-1)}), \text{ go to step 6.} \end{cases}$

Step 4: (Constraints loop)
        If $j < m$, increase $j \leftarrow j + 1$ and go to step 3.
```

Step 5: (Store the success and update the internal memory)
Set $x^{(k,n\ell+i)} = x'$, $s_i^{(k,\ell+1)} = 3\,s_i^{(k,\ell)}$, and replace $d_i^{(k)} \leftarrow d_i^{(k)} + s_i^{(k,\ell)}$.
If $\nu_i^{(k)} = -1$, set $\nu_i^{(k)} = 0$.
Go to step 7.

Step 6: (Internal memory update in case of failure)
Set $x^{(k,n\ell+i)} = x^{(k,n\ell+i-1)}$,
$s_i^{(k,\ell+1)} = -\frac{1}{2}\,s_i^{(k,\ell)}$.
If $\nu_i^{(k)} = 0$, set $\nu_i^{(k)} = 1$.

Step 7: (Main loop)
If $\nu_j^{(k)} = 1$ for all $j = 1(1)n$, go to step 8;
otherwise

$$\text{if } i \begin{cases} < n, & \text{increase } i \leftarrow i + 1 \\ = n, & \text{increase } \ell \leftarrow \ell + 1 \text{ and set } i = 1. \end{cases}$$

Go to step 2.

Step 8: (Preparation for the orthogonalization and check for termination)
Set $x^{(k+1,0)} = x^{(k,n\ell+i)} = x^{(k,0)} + \sum_{j=1}^{n} d_j^{(k)}\,v_j^{(k)}$.
Construct the vectors $a_i^{(k)} = \sum_{j=i}^{n} d_j^{(k)}\,v_j^{(k)}$ for all $i = 1(1)n$
(a_1 is the total progress during the loop just finished).
If $\|a_1^{(k)}\| < \varepsilon$ and (if $n > 1$)$\|a_2^{(k)}\| > 0.3\,\|a_1^{(k)}\|$, increase $\mu \leftarrow \mu + 1$;
otherwise set $\mu = 0$.
If $\mu = 6$, end the search.

Step 9: (Orthogonalization)
If $n > 1$,
construct new direction vectors $v_i^{(k+1)}$ for $i = 1(1)n$
according to the recursion formula (Equation (3.21)) of the
Gram-Schmidt orthogonalization.
Increase $k \leftarrow k + 1$ and go to step 1.

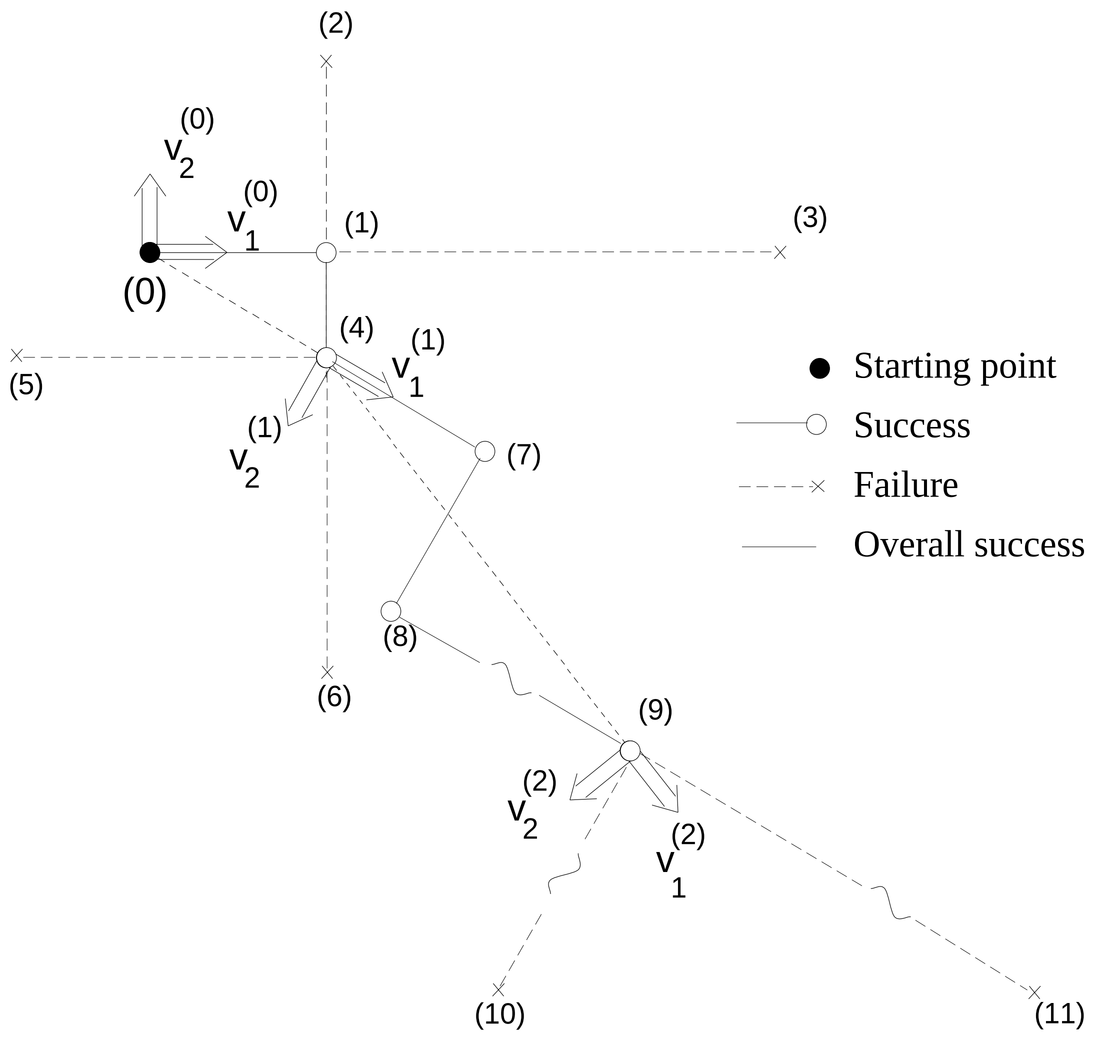


Figure 3.7: Strategy of Rosenbrock

Numbering	Iteration index k	Test index/ iteration $nl+i$	Variable values x_1	x_2	Step lengths s_1	s_2	Remarks
(0)	0	0	0	9	2	2	starting point
(1)	0	1	2	9	2	-	success
(2)	0	2	2	11	-	2	failure
(3)	0	3	8	9	6	-	failure
(4)	0	4	2	8	-	-1	success
(5)	0	5	-1	8	-3	-	failure
(6)	0	6	2	5	-	-3	failure
(4)	1	0	2	8	2	2	transformation and orthogo-nalization
(7)	1	1	3.8	7.1	2	-	success
(8)	1	2	2.9	5.3	-	2	success
(9)	1	3	8.3	2.6	6	-	success
(10)	1	4	5.6	-2.7	-	6	failure
(11)	1	5	24.4	-5.4	18	-	failure
(9)	2	0	8.3	2.6	2	2	transformation and orthogo-nalization

In Figure 3.7, including the following table, a few iterations of the Rosenbrock strategy for $n = 2$ are represented geometrically. At the starting point $x^{(0,0)}$ the search directions are the same as the unit vectors. After three runs through (6 trials), the trial steps in each direction have led to a success followed by a failure. At the best condition thus attained, (4) at $x^{(0,4)} = x^{(1,0)}$, new direction vectors $v_1^{(1)}$ and $v_2^{(1)}$ are generated. Five further trials lead to the best point, (9) at $x^{(1,3)} = x^{(2,0)}$, of the second iteration, at which a new choice of directions is again made. The complete sequence of steps can be followed, if desired, with the help of the accompanying table.

Numerical experiments show that within a few iterations the rotating coordinates become oriented such that one of the axes points along the gradient direction. The strategy thus allows sharp valleys in the topology of the objective function to be followed. Like the method of Hooke and Jeeves, Rosenbrock's procedure needs no information about partial derivatives and uses no line search method for exact location of relative minima. This makes it very robust. It has, however, one disadvantage compared to the direct pattern search: The orthogonalization procedure of Gram and Schmidt is very costly. It requires storage space of order $O(n^2)$ for the matrices $A = \{a_{ij}\}$ and $V = \{v_{ij}\}$, and the number of computational operations even increases with $O(n^3)$. At least in cases where the objective function call costs relatively little, the computation time for the orthogonalization with many variables becomes highly significant. Besides this, the number of parameters is in any case limited by the high storage space requirement.

If there are constraints, care must be taken to ensure that the starting point is inside the allowed region and sufficiently far from the boundaries. Examples of the application of

Rosenbrock's strategy can be found in Storey (1962), and Storey and Rosenbrock (1964). Among them is also a discretized functional optimization problem. For unconstrained problems there exists the code of Machura and Mulawa (1973). The Gram-Schmidt orthogonalization has been programmed, for example, by Clayton (1971).

Lange-Nielsen and Lance (1972) have proposed, on the basis of numerical experiments, two improvements in the Rosenbrock strategy. The first involves not setting constant step lengths at the beginning of a cycle or after each orthogonalization, but rather modifying them and simultaneously scaling them according to the successes and failures during the preceding cycle. The second improvement concerns the termination criterion. Rosenbrock's original version is replaced by the simpler condition that, according to the achievable computational accuracy, several consecutive trials yield the same value of the objective function.

3.2.1.4 Strategy of Davies, Swann, and Campey (DSC)

A combination of the Rosenbrock idea of rotating coordinates with one dimensional search methods is due to Swann (1964). It has become known under the name *Davies-Swann-Campey* (abbreviated DSC) strategy. The description of the procedure given by Box, Davies, and Swann (1969) differs somewhat from that in Swann, and so several versions of the strategy have arisen in the subsequent literature. Preference is given here to the original concept of Swann, which exhibits some features in common with the method of conjugate directions of Smith (1962) (see also Sect. 3.2.2). Starting from $x^{(0,0)}$, a line search is made in each of the unit directions $v_i^{(0)} = e_i$ for all $i = 1(1)n$. This process is followed by a one dimensional minimization in the direction of the overall success so far achieved

$$v_{n+1}^{(0)} = \frac{x^{(0,n)} - x^{(0,0)}}{\|x^{(0,n)} - x^{(0,0)}\|}$$

with the result $x^{(0,n+1)}$.

The orthogonalization follows this, e.g., by the Gram-Schmidt method. If one of the line searches was unsuccessful the new set of directions would no longer span the complete parameter space. Therefore only those old direction vectors along which a prescribed minimum distance has been moved are included in the orthogonalization process. The other directions remain unchanged. The DSC method, however, places a further hurdle before the coordinate rotation. If the distance covered in one iteration is smaller than the step length used in the line search, the latter is reduced by a factor 10, and the next iteration is carried out with the old set of directions.

After an orthogonalization, one of the new directions (the first) coincides with that of the $(n+1)$-th line search of the previous step. This can therefore also be interpreted as the first minimization in the new coordinate system. Only n more one dimensional searches need be made to finish the iteration. As a termination criterion the DSC strategy uses the length of the total vector between the starting point and end point of an iteration. The search is ended when it is less than a prescribed accuracy bound.

The algorithm runs as follows:

Step 0: (Initialization)
Specify a starting point $x^{(0,0)}$ and an initial step length $s^{(0)}$ (the same for all directions).
Define an accuracy requirement $\varepsilon > 0$.
Choose as a first set of directions $v_i^{(0)} = e_i$ for all $i = 1(1)n$.
Set $k = 0$ and $i = 1$.

Step 1: (Line search)
Starting from $x^{(k,i-1)}$, seek the relative minimum $x^{(k,i)}$ in the direction $v_i^{(k)}$ such that
$F(x^{(k,i)}) = F(x^{(k,i-1)} + d_i^{(k)}\, v_i^{(k)}) = \min\limits_d \{F(x^{(k,i-1)} + d\, v_i^{(k)})\}$.

Step 2: (Main loop)

$$\text{If } i \begin{cases} < n, & \text{increase } i \leftarrow i + 1, \text{ and go to step 1} \\ = n, & \text{go to step 3} \\ = n + 1, & \text{go to step 4.} \end{cases}$$

Step 3: (Eventually one more line search)
Construct $z = x^{(k,n)} - x^{(k,0)}$.
If $\|z\| > 0$, set $v_{n+1}^{(k)} = z \,/\, \|z\|\,, i = n + 1$, and go to step 1;
otherwise set $x^{(k,n+1)} = x^{(k,n)}$, $d_{n+1}^{(k)} = 0$, and go to step 5.

Step 4: (Check appropriateness of step length)
If $\|x^{(k,n+1)} - x^{(k,0)}\| \geq s^{(k)}$, go to step 6.

Step 5: (Termination criterion)
Set $s^{(k+1)} = 0.1\, s^{(k)}$.
If $s^{(k+1)} \leq \varepsilon$ end the search;
otherwise set $x^{(k+1,0)} = x^{(k,n+1)}$,
increase $k \leftarrow k + 1$, set $i = 1$, and go to step 1.

Step 6: (Check appropriateness of orthogonalization)
Reorder the directions $v_i^{(k)}$ and associated distances $d_i^{(k)}$ such that

$$|d_i^{(k)}| \begin{cases} > \varepsilon \text{ for all } i = 1(1)p, \\ \leq \varepsilon \text{ for all } i = p+1\,(1)n. \end{cases}$$

If $p < 2$, thus always for $n = 1$, go to step 5.

Step 7: (Orthogonalization)
Construct new direction vectors $v_i^{(k+1)}$ for $i = 1(1)p$ by means of the orthogonalization process of Gram and Schmidt (Equation (3.21)).
Set $s^{(k+1)} = s^{(k)}$, $d_1^{(k+1)} = d_{n+1}^{(k)}$,
and $x^{(k+1,0)} = x^{(k,n)}$, $x^{(k+1,1)} = x^{(k,n+1)}$.
Increase $k \leftarrow k + 1$, set $i = 2$, and go to step 1.

No geometric representation has been attempted here, since the fine deviations from the Rosenbrock method would hardly be apparent on a simple diagram.

The line search procedure of the DSC method has been described in detail by Box, Davies, and Swann (1969). It boxes in the minimum in the chosen direction using three equidistant points and then applies a single Lagrangian quadratic interpolation. The authors state that, in their experience, this is more economical with regard to the number of objective function calls than an exact line search with a sequence of interpolations. The algorithm of the line search is:

```
Step 0:  (Initialization)
         Specify a starting point $x_0$, a step length $s$, and a direction $v$
         (all given from the main program).

Step 1:  (Step forward)
         Construct $x = x_0 + s\,v$.
         If $F(x) \leq F(x_0)$, go to step 3.

Step 2:  (Step backward)
         Replace $x \leftarrow x - 2\,s\,v$ and $s \leftarrow -s$.
         If $F(x) \leq F(x_0)$, go to step 3;
         otherwise (both first trials without success) go to step 5.

Step 3:  (Further steps)
         Replace $s \leftarrow 2\,s$ and set $x_0 = x$.
         Construct $x = x_0 + s\,v$.
         If $F(x) \leq F(x_0)$, repeat step 3.

Step 4:  (Prepare interpolation)
         Replace $s \leftarrow 0.5\,s$.
         Construct $x = x_0 + s\,v$.
         Of the four points just generated, $x_0 - s$, $x_0$, $x_0 + s$, and $x_0 + 2\,s$, reject
         the one which is furthest from the point that has the smallest value of the
         objective function.

Step 5:  (Interpolation)
         Define the three available equidistant points $x_1 < x_2 < x_3$ and the associated
         function values $F_1$, $F_2$, and $F_3$ (this can be done in the course of the trial steps
         to box in the minimum). Fit a trial parabola through the three points and
         solve the necessary condition for its minimum. Because the argument values
         are equidistant, Equation (3.16) for the Lagrangian quadratic interpolation
         simplifies to
             $x = x_2 + \frac{s\,(F_1 - F_3)}{2\,(F_1 - 2\,F_2 + F_3)}$
         If the denominator vanishes or if it turns out that $F(x) > F_2$, then the line
         search ends with the result $x_0' = x_2$, $\tilde{F}_0 = F_2$;
         otherwise with the result $x_0' = x$ and $\tilde{F}_0 = F(x)$.
```

A numerical strategy comparison by M. J. Box (1966) shows the method to be a very effective optimization procedure, in general superior both to the Hooke and Jeeves and the Rosenbrock methods. However, the tests only refer to smooth objective functions with few variables. If the number of parameters is large, the costly orthogonalization process makes its inconvenient presence felt also in the DSC strategy.

Several suggestions have been made to date as to how to simplify the Gram-Schmidt procedure and to reduce its susceptibility to numerical rounding error (Rice, 1966; Powell, 1968a; Palmer, 1969; Golub and Saunders, 1970, Householder method).

Palmer replaces the conditions of Equation (3.21) by:

$$v_i^{(k+1)} = \begin{cases} v_i^{(k)}, & \text{if } \sum\limits_{j=1}^{n} d_j^2 = 0; \text{ otherwise} \\ \sum\limits_{j=1}^{n} d_j\, v_j^{(k)} \Big/ \sqrt{\sum\limits_{j=1}^{n} d_j^2}, & \text{for } i = 1 \\ \left(d_{i-1} \sum\limits_{j=i}^{n} d_j\, v_j^{(k)} - v_{i-1}^{(k)} \sum\limits_{j=i}^{n} d_j^2 \right) \Big/ \sqrt{\sum\limits_{j=i}^{n} d_j^2 \sum\limits_{j=i-1}^{n} d_j^2}, & \text{for } i = 2(1)n \end{cases}$$

He shows that even if no success was obtained in one of the directions $v_i^{(k)}$, that is $d_i = 0$, the new vectors $v_i^{(k+1)}$ for all $i = 1(1)n$ still span the complete parameter space, because $v_{i+1}^{(k+1)}$ is set equal to $-v_i^{(k)}$. Thus the algorithm does not need to be restricted to directions for which $d_i > \varepsilon$, as happens in the algorithm with Gram-Schmidt orthogonalization.

The significant advantage of the revised procedure lies in the fact that the number of computational operations remains only of the order $O(n^2)$. The storage requirement is also somewhat less since one $n \times n$ matrix as an intermediate storage area is omitted. For problems with linear constraints (equalities and inequalities) Box, Davies, and Swann (1969) recommend a modification of the orthogonalization procedure that works in a similar way to the method of *projected gradients* of Rosen (1960, 1961) (see also Davies, 1968). Non-linear constraints (inequalities) can be handled with the *created response surface* technique devised by Carroll (1961), which is one of the penalty function methods.

Further publications on the DSC strategy, also with comparison tests, are those of Swann (1969), Davies and Swann (1969), Davies (1970), and Swann (1972). Hoshino (1971) observes that in a narrow valley the search causes zigzag movements. His remedy for this is to add a further search, again in direction $v_1^{(k)}$, after each set of n line searches. With the help of two examples, for $n = 2$ and $n = 3$, he shows the accelerating effect of this measure.

3.2.1.5 Simplex Strategy of Nelder and Mead

There are a group of methods called *simplex* strategies that work quite differently to the direct search methods described so far. In spite of their common name they have nothing to do with the simplex method of linear programming of Dantzig (1966). The idea (Spendley, Hext, and Himsworth, 1962) originates in an attempt to reduce, as much as possible, the number of simultaneous trials in the experimental identification procedure of factorial design (see for example Davies, 1954). The minimum number according to Brooks and Mickey (1961) is $n+1$. Thus instead of a single starting point, $n+1$ vertices are used. They are arranged so as to be equidistant from each other: for $n = 2$ in an equilateral triangle; for $n = 3$ a *tetrahedron*; and in general a *polyhedron*, also referred to as a *simplex*. The objective function is evaluated at all the vertices. The iteration rule is: Replace the vertex with the largest objective function value by a new one situated at its *reflection* in the midpoint of the other n vertices. This rule aims to locate the new point at an especially promising place. If one lands near a minimum, the newest vertex can also be the worst. In this case the second worst vertex should be reflected. If the edge length of the polyhedron is not changed, the search eventually stagnates. The polyhedra rotate about the vertex with the best objective function value. A closer approximation to the optimum can only be achieved by halving the edge lengths of the simplex. Spendley, Hext, and Himsworth suggest doing this whenever a vertex is common to more than $1.65\,n + 0.05\,n^2$ consecutive polyhedra. Himsworth (1962) holds that this strategy is especially advantageous when the number of variables is large and the determination of the objective function prone to error.

To this basic procedure, various modifications have been proposed by, among others, Nelder and Mead (1965), Box (1965), Ward, Nag, and Dixon (1969), and Dambrauskas (1970, 1972). Richardson and Kuester (1973) have provided a complete program. The most common version is that of Nelder and Mead, in which the main difference from the basic procedure is that the size and shape of the simplex is modified during the run to suit the conditions at each stage.

The algorithm, with an extension by O'Neill (1971), runs as follows:

Step 0: (Initialization)
Choose a starting point $x^{(0,0)}$, initial step lengths $s_i^{(0)}$ for all $i = 1(1)n$ (if no better scaling is known, $s_i^{(0)} = 1$), and an accuracy parameter $\varepsilon > 0$ (e.g., $\varepsilon = 10^{-8}$). Set $c = 1$ and $k = 0$.

Step 1: (Establish the initial simplex)
$x^{(k,\nu)} = x^{(k,0)} + c\,s_\nu^{(0)}\,e_\nu$ for all $\nu = 1(1)n$.

Step 2: (Determine worst and best points for the normal reflection)
Determine the indices w (worst point) and b (best point) such that
$F(x^{(k,w)}) = \max_\nu\{F(x^{(k,\nu)}), \nu = 0(1)n\}$
$F(x^{(k,b)}) = \min_\nu\{F(x^{(k,\nu)}), \nu = 0(1)n\}$
Construct $\bar{x} = \frac{1}{n}\sum\limits_{\nu=0,\nu\neq w}^{n} x^{(k,\nu)}$ and $x' = 2\,\bar{x} - x^{(k,w)}$ (normal reflection).

If $F(x') < F(x^{(k,b)})$, go to step 4.

Step 3: (Compare trial with other vertices)
Determine the number μ for which $F(x') \le F(x^{(k,\nu)})$ holds for all $\nu = 0(1)n$.

$$\text{If} \quad \mu \begin{cases} > 1, & \text{set } x^{(k+1,w)} = x' \text{ and go to step 8} \\ = 1, & \text{go to step 5} \\ = 0, & \text{go to step 6.} \end{cases}$$

Step 4: (Expansion)
Construct $x'' = 2\,x' - \bar{x}$.
If $F(x'') < F(x^{(k,b)})$, set $x^{(k+1,w)} = x''$;
otherwise set $x^{(k+1,w)} = x'$.
Go to step 8.

Step 5: (Partial outside contraction)
Construct $x'' = 0.5\,(\bar{x} + x')$.
If $F(x'') \le F(x')$, set $x^{(k+1,w)} = x''$ and go to step 8;
otherwise go to step 7.

Step 6: (Partial inside contraction)
Construct $x'' = 0.5\,(\bar{x} + x^{(k,w)})$.
If $F(x'') \le F(x^{(k,w)})$, set $x^{(k+1,w)} = x''$ and go to step 8.

Step 7: (Total contraction)
Construct
$x^{(k+1,\nu)} = 0.5\,(x^{(k,b)} + x^{(k,\nu)})$ for all $\nu = 0(1)n$.
Go to step 9.

Step 8: (Normal iteration loop)
Assign $x^{(k+1,\nu)} = x^{(k,\nu)}$ for all $\nu = 0(1)n$ except $\nu = w$.

Step 9: (Termination criterion)
Increase $k \leftarrow k + 1$.
If $\frac{1}{n}\left(\sum\limits_{\nu=0}^{n} F^2(x^{(k,\nu)}) - \frac{1}{n+1}\left[\sum\limits_{\nu=0}^{n} F(x^{(k,\nu)})\right]^2\right) < \varepsilon^2$, go to step 10;
otherwise go to step 2.

Step 10: (Restart test; note that index w points to the new vertex)
Test whether any vector
$x = x^{(k,w)} \pm 0.001\,s_i^{(0)}\,e_i$ for all $i = 1(1)n$ exists, such that $F(x) < F(x^{(k,w)})$.
If so, set $x^{(k,0)} = x, c = 0.001$, and go to step 1 (restart);
otherwise end the search with result $x^{(k,w)}$.

The criterion for ending the minimum search is based on testing whether the variance of the objective function values at the vertices of the simplex is less than a prescribed limit. A few hypothetical iterations of the procedure for two variables are shown in Figure 3.8 including the following table. The sequence of reflections, expansions, and contractions is taken from the accompanying table, in which the simplex vertices are numbered sequentially and sorted at each stage in order of their objective function values.

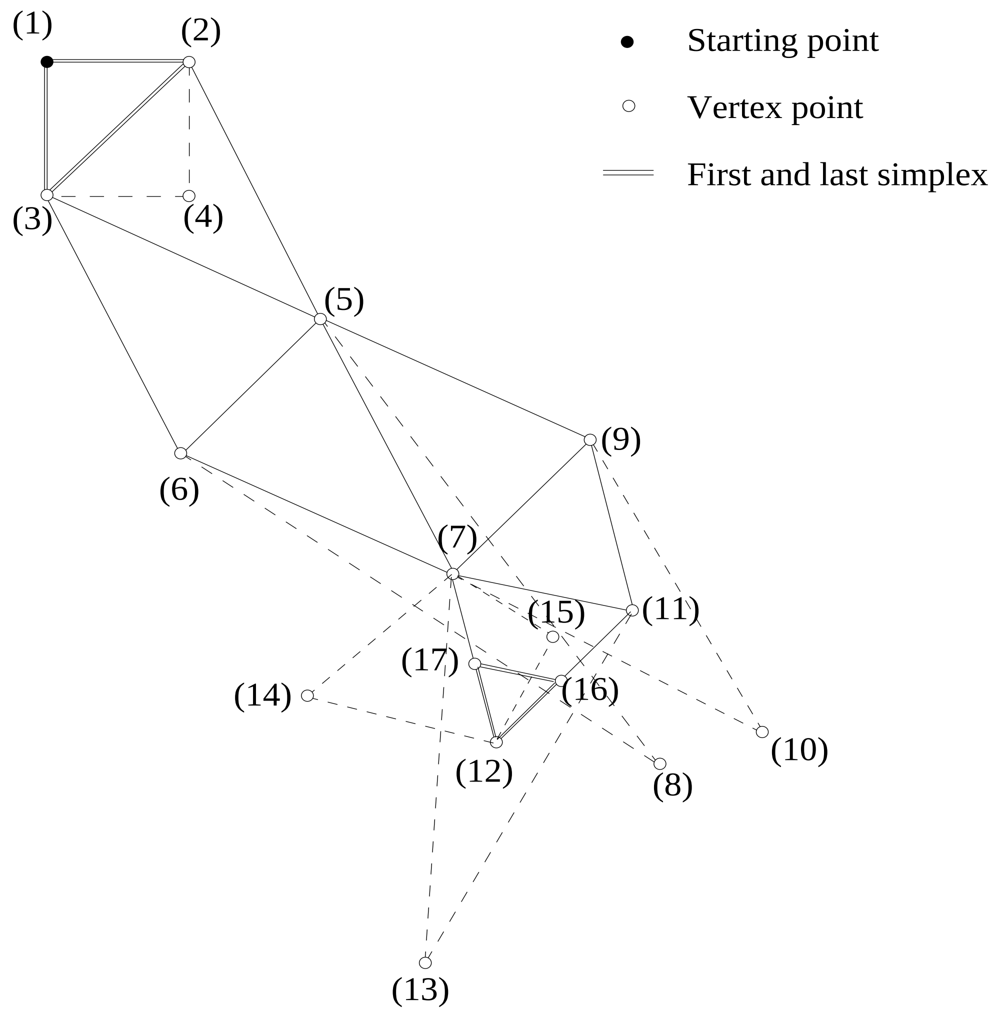


Figure 3.8: Simplex strategy of Nelder and Mead

Iteration				Simplex vertices				
index		worst				best		Remarks
0		1		2		3		start simplex
				2		3	4	reflection
				2		3	5	expansion (successful)
1		2		3		5		
				3	6	5		reflection
2		3		6		5		
				6		5	7	reflection
				6	8	5		expansion (unsuccessful)
3		6		5		7		
				5	9	7		reflection
4		5		9		7		
			10	9		7		reflection
				9	11	7		partial outside contraction
5		9		11		7		
				11		7	12	reflection
				11	13	7		expansion (unsuccessful)
6		11		7		12		
	14			7		12		reflection
	15			7		12		partial inside contraction
			17		16	12		total contraction
7		17		16		12		

The main difference between this program and the original strategy of Nelder and Mead is that after a normal ending of the minimization there is an attempt to construct a new starting simplex. To this end, small trial steps are taken in each coordinate direction. If just one of these tests is successful, the search is started again but with a simplex of considerably reduced edge lengths. This restart procedure recommends itself because, especially for a large number of variables, the simplex tends to no longer span the complete parameter space, i.e., to collapse, without reaching the minimum.

For few variables the simplex method is known to be robust and reliable, but also to be relatively costly. There are $n+1$ parameter vectors to be stored and the reflection requires a number of computational operations of order $O(n^2)$. According to Nelder and Mead, the number of function calls increases approximately as $O(n^{2.11})$; however, this empirical value is based only on test results with up to 10 variables. Parkinson and Hutchinson (1972a,b) describe a variant of the strategy in which the real storage requirement can be reduced by about half (see also Spendley, 1969). Masters and Drucker (1971) recommend altering the expansion or contraction factor after consecutive successes or failures respectively.

3.2.1.6 Complex Strategy of Box

M. J. Box (1965) calls his modification of the polyhedron strategy the *complex method*, an abbreviation for constrained simplex, since he conceived it also for problems with inequality constraints. The starting point of the search does not need to lie in the feasible region. For this case Box suggests locating an allowed point by minimizing the function

$$\tilde{F}(x) = -\sum_{j=1}^{m} G_j(x)\,\delta_j(x)$$

with

$$\delta_j(x) = \begin{cases} 0\,, & \text{if} \quad G_j(x) \geq 0 \\ 1\,, & \text{otherwise} \end{cases} \tag{3.23}$$

until

$$\tilde{F}(x) = 0$$

The two most important differences from the Nelder-Mead strategy are the use of more vertices and the expansion of the polyhedron at each normal reflection. Both measures are intended to prevent the complex from eventually spanning only a subspace of reduced dimensionality, especially at active constraints. If an allowed starting point is given or has been found, it defines one of the $n + 1 \leq N \leq 2\,n$ vertices of the polyhedron. The remaining vertex points are fixed by a random process in which each vector inside the closed region defined by the explicit constraints has an equal probability of selection. If an implicit constraint is violated, the new point is displaced stepwise towards the midpoint of the allowed vertices that have already been defined until it satisfies all the constraints. Implicit constraints $G_j(x) \geq 0$ are dealt with similarly during the course of the minimum search. If an explicit boundary is crossed, $x_i \geq a_i$, the offending variable is simply set back in the allowed region to a value near the boundary.

The details of the algorithm are as follows:

Step 0: (Initialization)

Choose a starting point $x^{(0)}$ and a number of vertices $N \geq n + 1$ (e.g., $N = 2\,n$). Number the constraints such that the first $j \leq m_1$ each depend only on one variable, $x_{\ell j}$ ($G_j(x_{\ell j})$, explicit form).

Test whether $x^{(0)}$ satisfies all the constraints.

If not, then construct a substitute objective function according to Equation (3.23).

Set up the initial complex as follows:

$x^{(0,1)} = x^{(0)}$

and $x^{(0,\nu)} = x^{(0)} + \sum\limits_{i=1}^{n} z_i\,e_i$ for $\nu = 2(1)N$,

where the z_i are uniformly distributed random numbers from the range

$$\begin{cases} [a_i, b_i], \text{ if constraints are given in the form } a_i \leq x_i \leq b_i; \\ \text{otherwise } \left[x_i^{(0)} - 0.5\,s, x_i^{(0)} + 0.5\,s\right], \text{where, e.g., } s = 1. \end{cases}$$

If $G_j(x^{(0,\nu)}) < 0$ for any $j \leq m_1, \nu > 1$,

replace $x_{\ell j}^{(0,\nu)} \leftarrow 2\,x_{\ell j}^{(0,1)} - x_{\ell j}^{(0,\nu)}$.

If $G_j(x^{(0,\nu)}) < 0$ for any $j \leq m, \nu > 1$,

replace $x^{(0,\nu)} \leftarrow 0.5\,[x^{(0,\nu)} + \frac{1}{\nu-1}\sum\limits_{\mu=1}^{\nu-1} x^{(0,\mu)}]$.

(If necessary repeat this process until $G_j(x^{(0,\nu)}) \geq 0$ for all $j = 1(1)m$.)
Set $k = 0$.

Step 1: (Reflection)
Determine the index w (worst vertex) such that
$F(x^{(k,w)}) = \max_{\nu}\{F(x^{(k,\nu)}), \nu = 1(1)N\}$.

Construct $\bar{x} = \frac{1}{N-1} \sum\limits_{\substack{\nu=1 \\ \nu \neq w}}^{N} x^{(k,\nu)}$

and $x' = \bar{x} + \alpha(\bar{x} - x^{(k,w)})$ (over-reflection factor $\alpha = 1.3$).

Step 2: (Check for constraints)
If $m = 0$, go to step 7; otherwise set $j = 1$.
If $m_1 = 0$, go to step 5.

Step 3: (Set vertex back into bounds for explicit constraints)
Obtain $g = G_j(x') = G_j(x'_{\ell j})$.
If $g \geq 0$, go to step 4;
otherwise replace $x'_{\ell j} \leftarrow x'_{\ell j} + g + \varepsilon$ (backwards length $\varepsilon = 10^{-6}$).
If $G_j(x') < 0$, replace $x'_{\ell j} \leftarrow x'_{\ell j} - 2(g + \varepsilon)$.

Step 4: (Explicit constraints loop)
Increase $j \leftarrow j + 1$.

$$\text{If} \begin{cases} j \leq m_1, & \text{go to step 3} \\ m_1 < j \leq m, & \text{go to step 5} \\ j > m, & \text{go to step 7.} \end{cases}$$

Step 5: (Check implicit constraints)
If $G_j(x') \geq 0$, go to step 6;
otherwise go to step 8, unless the same constraint caused a failure five times in a row without its function value $G_j(x')$ being changed. In this case go to step 9.

Step 6: (Implicit constraints loop)
If $j < m$, increase $j \leftarrow j + 1$ and go to step 5.

Step 7: (Check for improvement)
If $F(x') < F(x^{(k,\nu)})$ for at least one $\nu = 1(1)N$ except $\nu = w$,

$$\text{set } x^{(k+1,\nu)} = \begin{cases} x^{(k,\nu)}, & \text{for all } \nu = 1(1)N \text{ except } \nu = w, \\ x', & \text{for } \nu = w, \end{cases}$$

increase $k \leftarrow k + 1$ and go to step 1;
otherwise go to step 8, unless a failure occurred five times in a row with no change in the objective function value $F(x')$. In this case go to step 9.

Step 8: (Contraction)
Replace $x' \leftarrow 0.5(\bar{x} + x')$.
Go to step 2.

Step 9: (Termination)
Determine the index b (best vertex) such that
$F(x^{(k,b)}) = \min_{\nu}\{F(x^{(k,\nu)}), \nu = 1(1)N\}$.
End the search with the result $x^{(k,b)}$ and $F(x^{(k,b)})$.

Box himself reports that in numerical tests his complex strategy gives similar results to the simplex method of Nelder and Mead, but both are inferior to the method of Rosenbrock with regard to the number of objective function calls. He actually uses his own modification of the Rosenbrock method. Investigation of the effect of the number of vertices of the complex and the expansion factor (in this case $2n$ and 1.3 respectively) lead him to the conclusion that neither value has a significant effect on the efficiency of the strategy. For $n > 5$ he considers that a number of vertices $N = 2n$ is unnecessarily high, especially when there are no constraints.

The convergence criterion appears very reliable. While Nelder and Mead require that the standard deviation of all objective function values at the polyhedron vertices, referred to its midpoint, must be less than a prescribed size, the complex search is only ended when several consecutive values of the objective function are the same to computational accuracy.

Because of the larger number of polyhedron vertices the complex method needs even more storage space than the simplex strategy. The order of magnitude, $O(n^2)$, remains the same. No investigations are known of the computational effort in the case of many variables. Modifications of the strategy are due to Guin (1968), Mitchell and Kaplan (1968), and Dambrauskas (1970, 1972). Guin defines a contraction rule with which an allowed point can be generated even if the allowed region is not convex. This is not always the case in the original method because the midpoint to which the worst vertex is reflected is not tested for feasibility.

Mitchell finds that the initial configuration of the complex influences the results obtained. It is therefore better to place the vertices in a deterministic way rather than to make a random choice. Dambrauskas combines the complex method with the step length rule of the stochastic approximation. He requires that the step lengths or edge lengths of the polyhedron go to zero in the limit of an infinite number of iterations, while their sum tends to infinity. This measure may well increase the reliability of convergence; however, it also increases the cost. Beveridge and Schechter (1970) describe how the iteration rules must be changed if the variables can take only discrete values. A practical application, in which a process has to be optimized dynamically, is described by Tazaki, Shindo, and Umeda (1970); this is the original problem for which Spendley, Hext, and Himsworth (1962) conceived their simplex EVOP (*evolutionary operation*) procedure.

Compared to other numerical optimization procedures the *polyhedra strategies* have the disadvantage that in the closing phase, near the optimum, they converge rather slowly and sometimes even stagnate. The direction of progress selected by the reflection then no longer coincides at all with the gradient direction. To remove this difficulty it has been suggested that information about the topology of the objective function, as given by function values at the vertices of the polyhedron, be exploited to carry out a quadratic interpolation. Such *surface fitting* is familiar from the related methods of test planning and evaluation (*lattice search, factorial design*), in which the task is to set up mathematical models of physical or other processes. This territory is entered for example by G. E. P. Box (1957), Box and Wilson (1951), Box and Hunter (1957), Box and Behnken (1960), Box and Draper (1969, 1987), Box et al. (1973), and Beveridge and Schechter (1970). It will not be covered in any more detail here.

3.2.2 Gradient Strategies

The Gauss-Seidel strategy very straightforwardly uses only directions parallel to the coordinate axes to successively improve the objective function value. All other direct search methods strive to advance more rapidly by taking steps in other directions. To do so they exploit the knowledge about the topology of the objective function gleaned from the successes and failures of previous iterations. Directions are viewed as most promising in which the objective function decreases rapidly (for minimization) or increases rapidly (for maximization). Southwell (1946), for example, improves the relaxation by choosing the coordinate directions, not cyclically, but in order of the size of the local gradient in them. If the restriction of parallel axes is removed, the local best direction is given by the (negative) *gradient vector*

$$\nabla F(x) = (F_{x_1}(x), F_{x_2}(x), \ldots, F_{x_n}(x))^T$$

with

$$F_{x_i}(x) = \frac{\partial F}{\partial x_i}(x)\,, \qquad \text{for all } i \,=\, 1(1)n$$

at the point $x^{(0)}$. All hill climbing procedures that orient their choice of search directions $v^{(0)}$ according to the first partial derivatives of the objective function are called *gradient strategies*. They can be thought of as analogues of the *total step procedure* of Jacobi for solving systems of linear equations (see Schwarz, Rutishauser, and Stiefel, 1968).

So great is the number of methods of this type which have been suggested or applied up to the present, that merely to list them all would be difficult. The reason lies in the fact that the gradient represents a *local* property of a function. To follow the path of the gradient exactly would mean determining in general a *curved trajectory* in the n-dimensional space. This problem is only approximately soluble numerically and is more difficult than the original optimization problem. With the help of analogue computers continuous gradient methods have actually been implemented (Bekey and McGhee, 1964; Levine, 1964). They consider the trajectory $x(t)$ as a function of time and obtain it as the solution of a system of first order differential equations.

All the numerical variants of the gradient method differ in the lengths of the discrete steps and thereby also with regard to how exactly they follow the gradient trajectory. The iteration rule is generally

$$x^{(k+1)} = x^{(k)} - s^{(k)} \, \frac{\nabla F(x^{(k)})}{\|\nabla F(x^{(k)})\|}$$

It assumes that the partial derivatives everywhere exist and are unique. If $F(x)$ is continuously differentiable then the partial derivatives exist and $F(x)$ is continuous.

A distinction is sometimes drawn between *short step methods*, which evaluate the gradients again after a small step in the direction $\nabla F(x^{(k)})$ (for maximization) or $-\nabla F(x^{(k)})$ (for minimization), and their equivalent *long step methods*. Since for finite step lengths $s^{(k)}$ it is not certain whether the new variable vector is really better than the old, after the step the value of the objective function must be tested again. Working with small steps increases the number of objective function calls and gradient evaluations. Besides $F(x)$, n partial derivatives must be evaluated. Even if the slopes can be obtained analytically and can be specified as functions, there is no reason to suppose that the number of computational operations per function call is much less than for the objective function itself. Except in special cases, the total cost increases roughly as the weighting factor $(n + 1)$ and the number of objective function calls. This also holds if the partial derivatives are approximated by differential quotients obtained by means of trial steps

$$F_{x_i}(x) = \frac{\partial F(x)}{\partial x_i} = \frac{F(x + \delta\, e_i) - F(x)}{\delta} + O(\delta^2)\,, \qquad \text{for all } i \,=\, 1(1)n$$

Additional difficulties arise here since for values of δ that are too small the subtraction is subject to rounding error, while for trial steps that are too large the neglect of terms $O(\delta^2)$ leads to incorrect values. The choice of suitable deviations δ requires special care in all cases (Hildebrand, 1956; Curtis and Reid, 1974).

Cauchy (1847), Kantorovich (1940, 1945), Levenberg (1944), and Curry (1944) are the originators of the gradient strategy, which started life as a method of solving equations and systems of equations. It is first referred to as an aid to solving variational problems by Hadamard (1908) and Courant (1943). Whereas Cauchy works with fixed step lengths $s^{(k)}$, Curry tries to determine the distance covered in the (not normalized) direction $v^{(k)} \,=\, -\nabla F(x^{(k)})$ so as to reach a relative minimum (see also Brown, 1959). In principle, any one of the one dimensional search methods of Section 3.1 can be called upon to find the optimal value for $s^{(k)}$:

$$F(x^{(k)} + s^{(k)}\, v^{(k)}) = \min_s \{F(x^{(k)} + s\, v^{(k)})\}$$

This variant of the basic strategy could thus be called a *longest step procedure*. It is better known however under the name *optimum gradient method*, or *method of steepest descent* (for maximization, *ascent*). Theoretical investigations of convergence and rate of convergence of the method can be found, e.g., in Akaike (1960), Goldstein (1962), Ostrowski (1967), Forsythe (1968), Elkin (1968), Zangwill (1969), and Wolfe (1969, 1970, 1971). Zangwill proves convergence based on the assumptions that the line searches are exact and the objective function is continuously twice differentiable. Exactness of the one dimensional minimization is not, however, a necessary assumption (Wolfe, 1969). It is significant that one can only establish theoretically that a stationary point will be reached ($\nabla F(x) \,=\, 0$) or approached ($\|\nabla F(x)\| \,<\, \varepsilon\,, \varepsilon \,>\, 0$). The stationary point is a minimum, only if $F(x)$ is convex and three times differentiable (Akaike, 1960). Zellnik, Sondak, and Davis (1962), however, show that saddle points are in practice an obstacle, only if the search is started at one, or on a straight gradient trajectory passing through one. In other cases numerical rounding errors ensure that the path to a saddle point is unstable.

The gradient strategy, however, cannot distinguish global from local minima. The optimum at which it aims depends only on the choice of the starting point for the search. The only chance of finding absolute extrema is to start sufficiently often from various initial values of the variables and to iterate each time until the convergence criterion is satisfied (Jacoby, Kowalik, and Pizzo, 1972). The termination rules usually recommended for gradient methods are that the absolute value of the vector

$$\|\nabla F(x^{(k)})\| < \varepsilon_1, \quad \varepsilon_1 \geq 0$$

or the difference

$$F(x^{(k-1)}) - F(x^{(k)}) < \varepsilon_2, \quad \varepsilon_2 \geq 0$$

vanishes or falls below a given limit.

The rate of convergence of the strategy of steepest descent depends on the structure of the objective function, but is no better than first order apart from exceptional cases like contours that are concentric, spherically symmetric hypersurfaces (Forsythe, 1968). In the general quadratic case

$$F(x) = \frac{1}{2}\, x^T A\, x + x^T b + c \tag{3.24}$$

with elliptic contours, i.e., positive definite matrix A, the convergence rate depends on the ratio of smallest to greatest eigenvalue of A, or geometrically expressed, on the oblateness of the ellipsoid. It can be extremely small (Curry, 1944; Akaike, 1960; Goldstein, 1962) and is in principle no better than the coordinate strategy with line searches (Elkin, 1968). In narrow valleys both procedures execute zigzag movements with very small changes in the variable values in relation to the distance from the objective. The individual steps can even become too small to effect any change in the objective function value if this is defined with a finite number of decimal places. Then the search ends before reaching the desired optimum. To obviate this difficulty, Booth (1949, 1955) has suggested only going 90% of the way to the relative minimum in each line search (see also Stein, 1952; Kantorovich, 1952; Faddejew and Faddejewa, 1973). In fact, one often obtains much better results with this kind of underrelaxation. Even more advantageous is a modification due to Forsythe and Motzkin (1951). It is based on the observation that the search movements in the minimization of quadratic objective functions oscillate between two asymptotic directions (Stiefel, 1952; Forsythe, 1968). Forsythe and Motzkin therefore from time to time include a line search in the direction

$$v^{(k)} = x^{(k)} - x^{(k-2)}\,, \quad \text{for } k \geq 2$$

in order to accelerate the convergence. For $n = 2$ the gradient method is thereby greatly improved; with many variables the efficiency advantage is lost again. Similar proposals for increasing the rate of convergence have been made by Baer (1962), Humphrey and Cottrell (1962, 1966), Witte and Holst (1964), Schinzinger (1966), and McCormick (1969). The *Partan* (acronym for parallel tangents) methods of Buehler, Shah, and Kempthorne (1961, 1964) have attracted particular attention. One of these, *continued gradient Partan*, alternates between gradient directions

$$v^{(k)} = -\nabla F(x^{(k)})\,, \quad \text{for } k = 0 \text{ as well as } k \geq 1, \text{ odd}$$

and those derived from previous iteration points

$$v^{(k)} = x^{(k)} - x^{(k-3)}\,, \qquad \text{for } k \geq 2, \text{ even (with } x^{(-1)} = x^{(0)})$$

For quadratic functions the minimum is reached after at most $2n - 1$ line searches (Shah, Buehler, and Kempthorne, 1964). This desirable property of converging after a finite number of iterations, that is also called quadratic convergence, is only shared by strategies that apply conjugate gradients, of which the Partan methods can be regarded as forerunners (Pierre, 1969; Sorenson, 1969).

In the fifties, simple gradient strategies were very popular, especially the method of steepest descent. Today they are usually only to be found as components of program packages together with other hill climbing methods, e.g., in GROPE of Flood and Leon (1966), in AID of Casey and Rustay (1966), in AESOP of Hague and Glatt (1968), and in GOSPEL of Huelsman (1968). McGhee (1967) presents a detailed flow diagram. Wasscher (1963a,b) has published two ALGOL codings (see also Haubrich, 1963; Wallack, 1964; Varah, 1965; Wasscher, 1965). The partial derivatives are obtained numerically. A comprehensive bibliography by Leon (1966b) names most of the older versions of strategies and gives many examples of their application. Numerical comparison tests have been carried out by Fletcher (1965), Box (1966), Leon (1966a), Colville (1968, 1970), and Kowalik and Osborne (1968). They show the superiority of first (and second) order methods over direct search strategies for objective functions with smooth topology. Gradient methods for solving systems of differential equations are described for example by Talkin (1964). For such problems, as well as for functional optimization problems, analogue and hybrid computers have often been applied (Rybashov, 1965a,b, 1969; Sydow, 1968; Fogarty and Howe, 1968, 1970). A literature survey on this subject has been compiled by Gilbert (1967). For the treatment of variational problems see Kelley (1962), Altman (1966), Miele (1969), Bryson and Ho (1969), Céa (1971), Daniel (1971), and Tolle (1971).

In the experimental field, there are considerable difficulties in determining the partial derivatives. Errors in the values of the objective function can cause the predicted direction of steepest descent to lie almost perpendicular to the true gradient vector (Kowalik and Osborne, 1968). Box and Wilson (1951) attempt to compensate for the perturbations by repeating the trial steps or increasing their number above the necessary minimum of $(n + 1)$. With 2^n trials, for example, a complete factorial design can be constructed (e.g., Davies, 1954). The slope in one direction is obtained by averaging the function value differences over 2^{n-1} pairs of points (Lapidus et al., 1961). Another possibility is to determine the coefficients of a linear polynomial such that the sum of the squares of the errors between measured and model function values at $N \geq n + 1$ points is a minimum. The linear function then represents the tangent plane of the objective function at the point under consideration. The cost of obtaining the gradients when there are many variables is too great for practical application, and only justified if the aim is rather to set up a mathematical model of the system than simply to perform the optimization.

In the EVOP (acronym for evolutionary operation) scheme, G. E. P. Box (1957) has presented a practical simplification of this gradient method. It actually counts as a direct search strategy because it does not obtain the direction of the gradient but only one of a finite number of especially good directions. Spendley, Hext, and Himsworth (1962) have devised a variant of the procedure (see also Sections 3.2.1.5 and 3.2.1.6). Lowe (1964) has gathered together the various schemes of trial steps for the EVOP strategy. The philosophy of the EVOP strategy is treated in detail by Box and Draper (1969). Some examples of applications are given by Kenworthy (1967). The efficiency of methods of determining the gradient in the case of stochastic perturbations is dealt with by Mlynski (1964a,b, 1966a,b), Sergiyevskiy and Ter-Saakov (1970), and others.

3.2.2.1 Strategy of Powell: Conjugate Directions

The most important idea for overcoming the convergence difficulties of the gradient strategy is due to Hestenes and Stiefel (1952), and again comes from the field of linear algebra (see also Ginsburg, 1963; Beckman, 1967). It trades under the names *conjugate directions* or *conjugate gradients*. The directions $\{v_i\,, i = 1(1)n\}$ are said to be conjugate with respect to a positive definite $n \times n$ matrix A if (Hestenes, 1956)

$$v_i^T\, A\, v_j = 0\,, \qquad \text{for all } i, j = 1(1)n\,, i \neq j$$

A further property of conjugate directions is their linear independence, i.e.,

$$\sum_{i=1}^{n} \alpha_i\, v_i = 0$$

only holds if all the constants $\{\alpha_i\,, i = 1(1)n\}$ are zero. If A is replaced by the unit matrix, $A = I$, then the v_i are mutually orthogonal. With $A = \nabla^2 F(x)$ (*Hessian matrix*) the minimum of a quadratic function is obtained exactly in n line searches in the directions v_i. This is a factor two better than the gradient Partan method. For general non-linear problems the convergence rate cannot be specified. As it is frequently assumed, however, that many problems behave roughly quadratically near the optimum, it seems worthwhile to use conjugate directions. The quadratic convergence of the search with conjugate directions comes about because second order properties of the objective function are taken into account. In this respect it is not, in fact, a first order gradient method, but a second order procedure. If all the n first and $\frac{n}{2}(n + 1)$ second partial derivatives are available, the conjugate directions can be generated in one process corresponding to the Gram-Schmidt orthogonalization (Kowalik and Osborne, 1968). It calls for expensive matrix operations. Conjugate directions can, however, be constructed without knowledge of the second derivatives: for example, from the changes in the gradient vector in the course of the iterations (Fletcher and Reeves, 1964). Because of this implicit exploitation of second order properties, conjugate directions has been classified as a gradient method.

The *conjugate gradients method* of Fletcher and Reeves consists of a sequence of line searches with Hermitian interpolation (see Sect. 3.1.2.3.4). As a first search direction $v^{(0)}$ at the starting point $x^{(0)}$, the simple gradient direction

$$v^{(0)} = -\nabla F(x^{(0)})$$

is used. The recursion formula for the subsequent iterations is

$$v^{(k)} = \alpha^{(k)}\, v^{(k-1)} - \nabla F(x^{(k)})\,, \qquad \text{for all } k = 1(1)n \tag{3.25}$$

with the correction factor

$$\alpha^{(k)} = \frac{\nabla F(x^{(k)})^T \, \nabla F(x^{(k)})}{\nabla F(x^{(k-1)})^T \, \nabla F(x^{(k-1)})}$$

For a quadratic objective function with a positive definite Hessian matrix, conjugate directions are generated in this way and the minimum is found with n line searches. Since at any time only the last direction needs to be stored, the storage requirement increases linearly with the number of variables. This often signifies a great advantage over other strategies. In the general, non-linear, non-quadratic case more than n iterations must be carried out, for which the method of Fletcher and Reeves must be modified. Continued application of the recursion formula (Equation (3.25)) can lead to linear dependence of the search directions. For this reason it seems necessary to forget from time to time the accumulated information and to start afresh with the simple gradient direction (Crowder and Wolfe, 1972). Various suggestions have been made for the frequency of this restart rule (Fletcher, 1972a). Absolute reliability of convergence in the general case is still not guaranteed by this approach. If the Hessian matrix of second partial derivatives has points of singularity, then the conjugate gradient strategy can fail. The exactness of the line searches also has an important effect on the convergence rate (Kawamura and Volz, 1973). Polak (1971) defines conditions under which the method of Fletcher and Reeves achieves greater than linear convergence. Fletcher (1972c) himself has written a FORTRAN program.

Other conjugate gradient methods have been proposed by Powell (1962), Polak and Ribière (1969) (see also Klessig and Polak, 1972), Hestenes (1969), and Zoutendijk (1970). Schley (1968) has published a complete FORTRAN program. Conjugate directions are also produced by the *projected gradient methods* (Myers, 1968; Pearson, 1969; Sorenson, 1969; Cornick and Michel, 1972) and the *memory gradient methods* (Miele and Cantrell, 1969, 1970; see also Cantrell, 1969; Cragg and Levy, 1969; Miele, 1969; Miele, Huang, and Heidemann, 1969; Miele, Levy, and Cragg, 1971; Miele, Tietze, and Levy, 1972; Miele et al., 1974). Relevant theoretical investigations have been made by, among others, Greenstadt (1967a), Daniel (1967a, 1970, 1973), Huang (1970), Beale (1972), and Cohen (1972).

Conjugate gradient methods are encountered especially frequently in the fields of functional optimization and optimal control problems (Daniel, 1967b, 1971; Pagurek and Woodside, 1968; Nenonen and Pagurek, 1969; Roberts and Davis, 1969; Polyak, 1969; Lasdon, 1970; Kelley and Speyer, 1970; Kelley and Myers, 1971; Speyer et al., 1971; Kammerer and Nashed, 1972; Szegö and Treccani, 1972; Polak, 1972; McCormick and Ritter, 1974). *Variable metric strategies* are also sometimes classified as conjugate gradient procedures, but more usually as *quasi-Newton methods*. For quadratic objective functions they generate the same sequence of points as the Fletcher-Reeves strategy and its modifications (Myers, 1968; Huang, 1970). In the non-quadratic case, however, the search directions are different. With the variable metric, but not with conjugate directions, Newton directions are approximated.

For many practical problems it is very difficult if not impossible to specify the partial derivatives as functions. The sensitivity of most conjugate gradient methods to imprecise specification of the gradient directions makes it seem inadvisable to apply finite difference methods to approximate the slopes of the objective function. This is taken into account by some procedures that attempt to construct conjugate directions without knowledge of the derivatives. The oldest of these was devised by Smith (1962). On the basis of numerical tests by Fletcher (1965), however, the version of Powell (1964) has proved to be better. It will be briefly presented here. It is arguable whether it should be counted as a gradient strategy. Its intermediate position between direct search methods that only use function values, and Newton methods that make use of second order properties of the objective function (if only implicitly), nevertheless makes it come close to this category.

The strategy of conjugate directions is based on the observation that a line through the minimum of a quadratic objective function cuts all contours at the same angle. Powell's idea is then to construct such special directions by a sequence of line searches. The unit vectors are taken as initial directions for the first n line searches. After these, a minimization is carried out in the direction of the overall result. Then the first of the old direction vectors is eliminated, the indices of the remainder are reduced by one and the direction that was generated and used last is put in the place freed by the nth vector. As shown by Powell, after n cycles, each of $n + 1$ line searches, a set of conjugate directions is obtained provided the objective function is quadratic and the line searches are carried out exactly.

Zangwill (1967) indicates how this scheme might fail. If no success is obtained in one of the search directions, i.e., the distance covered becomes zero, then the direction vectors are linearly dependent and no longer span the complete parameter space. The same phenomenon can be provoked by computational inaccuracy. To prevent this, Powell has modified the basic algorithm. First of all, he designs the scheme of exchanging directions to be more flexible, actually by maximizing the determinant of the normalized direction vectors. It can be shown that, assuming a quadratic objective function, it is most favorable to eliminate the direction in which the largest distance was covered (see Dixon, 1972a). Powell would also sometimes leave the set of directions unchanged. This depends on how the value of the determinant would change under exchange of the search directions. The objective function is here tested at the position given by doubling the distance covered in the cycle just completed. Powell makes the termination of the search depend on all variables having changed by less than $0.1\,\varepsilon$ within an iteration, where ε represents the required accuracy. Besides this first convergence criterion, he offers a second, stricter one, according to which the state reached at the end of the normal procedure is slightly displaced and the minimization repeated until the termination conditions are again fulfilled. This is followed by a line search in the direction of the difference vector between the last two endpoints. The optimization is only finally ended when the result agrees with those previously obtained to within the allowed deviation of $0.1\,\varepsilon$ for each component.

The algorithm of Powell runs as follows:

Step 0: (Initialization)
Specify a starting point $x^{(0)}$
and accuracy requirements $\varepsilon_i > 0$ for all $i = 1(1)n$.

```
Step 1:  (Specify first set of directions)
         Set $v_i^{(0)} = e_i$ for all $i = 1(1)n$
         and set $k = 0$.

Step 2:  (Start outer loop)
         Set $x^{(k,0)} = x^{(k)}$ and $i = 1$.

Step 3:  (Line search)
         Determine $x^{(k,i)}$ such that
         $F(x^{(k,i)}) = \min_s\{F(x^{(k,i-1)} + s\, v_i^{(k)})\}$.

Step 4:  (Inner loop)
         If $i < n$, increase $i \leftarrow i + 1$ and go to step 3.

Step 5:  (First termination criterion)
         Increase $k \leftarrow k + 1$.
         Set $x^{(k)} = x^{(k-1,n)}$ and $v_i^{(k)} = v_i^{(k-1)}$ for all $i = 1(1)n$.
         If $|x_i^{(k)} - x_i^{(k-1)}| < 0.1\,\varepsilon_i$ for all $i = 1(1)n$, go to step 9.

Step 6:  (First test for direction exchange)
         Determine $\tilde{F} = F(2\,x^{(k)} - x^{(k-1)})$.
         If $\tilde{F} \geq F(x^{(k-1)})$, go to step 2.

Step 7:  (Second test for direction exchange)
         Determine the index $\ell, 1 \leq \ell \leq n$, such that
         $\varphi_\ell = \max_i\{\varphi_i, i = 1(1)n\}$, where $\varphi_i = F(x^{(k-1,i-1)}) - F(x^{(k-1,i)})$.
         If $[F(x^{(k-1)}) - 2\,F(x^{(k)}) + \tilde{F}]\,[F(x^{(k-1)}) - F(x^{(k)}) - \varphi_\ell]^2 \geq$
         $\geq \frac{1}{2}\,\varphi_\ell\,[F(x^{(k-1)}) - \tilde{F}]^2$, go to step 2.

Step 8:  (New direction set and additional line search)
         Eliminate $v_\ell^{(k)}$ from the list of directions so that $v_n^{(k)}$ becomes free.
         Set $v_n^{(k)} = x^{(k)} - x^{(k-1)} = x^{(k-1,n)} - x^{(k-1,0)}$.
         Determine a new $x^{(k)}$ such that
         $F(x^{(k)}) = \min_s\{F(x^{(k-1,n)} + s\,v_n^{(k)})\}$.
         Go to step 2.

Step 9:  (Second termination criterion)
         Set $y^{(1)} = x^{(k)}$ and replace $x^{(0)} \leftarrow y^{(1)} + \sum_{i=1}^{n} 10\,\varepsilon\,e_i$.
         Repeat steps 1 to 8 until the convergence criterion (step 5) is fulfilled again
         and call the result $y^{(2)}$.
         Determine $y^{(3)}$ such that
         $F(y^{(3)}) = \min_s\{F(y^{(2)} + s\,(y^{(2)} - y^{(1)}))\}$.
         If $|y_i^{(3)} - y_i^{(2)}| < 0.1\,\varepsilon_i$ for all $i = 1(1)n$
         and $|y_i^{(3)} - y_i^{(1)}| < 0.1\,\varepsilon_i$ for all $i = 1(1)n$,
         then end the search with the result $y^{(3)}$ and $F(y^{(3)})$;
```

otherwise set $x^{(0)} = y^{(3)}, v_1^{(0)} = y^{(3)} - y^{(1)}$,
$v_i^{(0)} = v_i^{(k)}$ for $i = 2(1)n$, k = 0, and go to step 2.

Figure 3.9 illustrates a few iterations for a hypothetical two parameter function. Each of the first loops consists of $n + 1 = 3$ line searches and leads to the adoption of a new search direction. If the objective function had been of second order, the minimum would certainly have been found by the last line search of the second loop. In the third and fourth loops it has been assumed that the trial steps have led to a decision not to exchange directions, thus the old direction vectors, numbered v_3 and v_4 are retained. Further loops, e.g., according to step 9, are omitted.

The quality of the line searches has a strong influence on the construction of the conjugate directions. Powell uses a sequence of Lagrangian quadratic interpolations. It is terminated as soon as the required accuracy is reached. For the first minimization within an iteration three points and Equation (3.16) are used. The argument values taken in direction v_i are: x (the starting point), $x + s_i\, v_i$, and either $x + 2\, s_i\, v_i$ or $x - s_i\, v_i$, according to whether $F(x + s_i\, v_i) < F(x)$ or not. The step length s_i is given initially by the associated accuracy ε_i multiplied by a maximum factor and is later adjusted in the course of the iterations.

In the direction constructed from the successful results of n one dimensional searches, the argument values are called $x^{(k,0)}, x^{(k,n)}$, and $2\, x^{(k,n)} - x^{(k,0)}$. With three points $(a < b < c)$ and associated objective function values (F_a, F_b, F_c), not only the minimum but also the second derivative of the quadratic trial function $P(x)$ can be specified. In the notation of Section 3.1.2.3.3, the formula for the curvature ρ_i in the direction v_i is

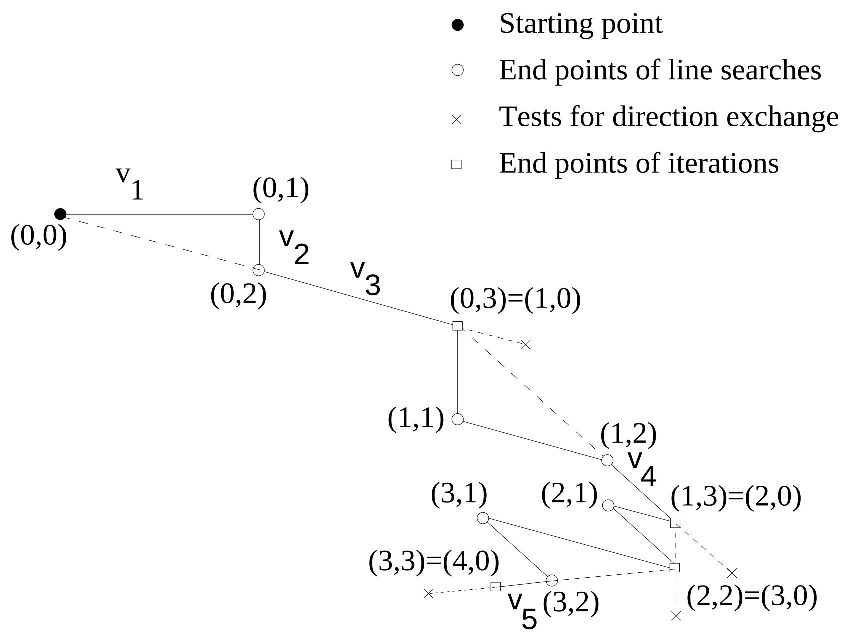


Figure 3.9: Strategy of Powell, conjugate directions

$$\rho_i = \frac{\partial^2}{\partial s^2}(P(x + s\,v_i)) = -2\,\frac{(b-c)\,F_a + (c-a)\,F_b + (a-b)\,F_c}{(b-c)\,(c-a)\,(a-b)}$$

Powell uses this quantity ρ_i for all subsequent interpolations in the direction v_i as a scale for the second partial derivative of the objective function. He scales the directions v_i, which in his case are not normalized, by $1/\sqrt{\rho_i}$. This allows the possibility of subsequently carrying out a simplified interpolation with only two argument values, x and $x + s_i\,v_i$. It is a worthwhile procedure, since each direction is used several times. The predicted minimum, assuming that the second partial derivatives have value unity, is then

$$x' = x + \left(\frac{1}{2}\,s_i - \frac{1}{s_i}\,[F(x + s_i\,v_i) - F(x)]\right)\,v_i$$

For the trial step lengths s_i, Powell uses the empirical recursion formula

$$s_i^{(k)} = 0.4\,\sqrt{F(x^{(k)}) - F(x^{(k-1)})}$$

Because of the scaling, all the step lengths actually become the same. A more detailed justification can be found in Hoffmann and Hofmann (1970).

Contrary to most other optimization procedures, Powell's strategy is available as a precise algorithm in a tested code (Powell, 1970f). As Fletcher (1965) reports, this method of conjugate directions is superior for the case of a few variables both to the DSC method and to a strategy of Smith, especially in the neighborhood of minima. For many variables, however, the strict criterion for adopting a new direction more frequently causes the old set of directions to be retained and the procedure then converges slowly. A problem which had a singular Hessian matrix at the minimum made the DSC strategy look better. In a later article, Fletcher (1972a) defines a limit of $n = 10$ to 20, above which the Powell strategy should no longer be applied. This is confirmed by the test results presented in Chapter 6. Zangwill (1967) combines the basic idea of Powell with relaxation steps in order to avoid linear dependence of the search directions. Some results of Rhead (1971) lead to the conclusion that Powell's improved concept is superior to Zangwill's. Brent (1973) also presents a variant of the strategy without derivatives, derived from Powell's basic idea, which is designed to prevent the occurrence of linear dependence of the search directions without endangering the quadratic convergence. After every $n + 1$ iterations the set of directions is replaced by an orthogonal set of vectors. So as not to lose all the information, however, the unit vectors are not chosen. For quadratic objective functions the new directions remain conjugate to each other. This procedure requires $O(n^3)$ computational operations to determine orthogonal eigenvectors. As, however, they are only performed every $O(n^2)$ line searches, the extra cost is $O(n)$ per function call and is thus of the same order as the cost of evaluating the objective function itself. Results of tests by Brent confirm the usefulness of the strategy.

3.2.3 Newton Strategies

Newton strategies exploit the fact that, if a function can be differentiated any number of times, its value at the point $x^{(k+1)}$ can be represented by a series of terms constructed at another point $x^{(k)}$:

$$F(x^{(k+1)}) = F(x^{(k)}) + h^T \nabla F(x^{(k)}) + \frac{1}{2} h^T \nabla^2 F(x^{(k)}) h + \ldots \tag{3.26}$$

where

$$h = x^{(k+1)} - x^{(k)}$$

In this *Taylor series*, as it is called, all the terms of higher than second order are zero if $F(x)$ is quadratic. Differentiating Equation (3.26) with respect to h and setting the derivative equal to zero, one obtains a condition for the stationary points of a second order function:

$$\nabla F(x^{(k+1)}) = \nabla F(x^{(k)}) + \nabla^2 F(x^{(k)}) (x^{(k+1)} - x^{(k)}) = 0$$

or

$$x^{(k+1)} = x^{(k)} - [\nabla^2 F(x^{(k)})]^{-1} \nabla F(x^{(k)}) \tag{3.27}$$

If $F(x)$ is quadratic and $\nabla^2 F(x^{(0)})$ is positive-definite, Equation (3.27) yields the solution $x^{(1)}$ in a single step from any starting point $x^{(0)}$ without needing a line search. If Equation (3.27) is taken as the iteration rule in the general case it represents the extension of the *Newton-Raphson method* to functions of several variables (Householder, 1953). It is also sometimes called a *second order gradient method* with the choice of direction and step length (Crockett and Chernoff, 1955)

$$\begin{aligned} v^{(k)} &= -[\nabla^2 F(x^{(k)})]^{-1} \nabla F(x^{(k)}) \\ s^{(k)} &= 1 \end{aligned} \tag{3.28}$$

The real length of the iteration step is hidden in the non-normalized Newton direction $v^{(k)}$. Since no explicit value of the objective function is required, but only its derivatives, the Newton-Raphson strategy is classified as an indirect or analytic optimization method. Its ability to predict the minimum of a quadratic function in a single calculation at first sight looks very attractive. This single step, however, requires a considerable effort. Apart from the necessity of evaluating n first and $\frac{n}{2}(n+1)$ second partial derivatives, the *Hessian matrix* $\nabla^2 F(x^{(k)})$ must be inverted. This corresponds to the problem of solving a system of linear equations

$$\nabla^2 F(x^{(k)}) \; \triangle\, x^{(k)} = -\nabla F(x^{(k)}) \tag{3.29}$$

for the unknown quantities $\triangle x^{(k)}$. All the standard methods of linear algebra, e.g., Gaussian elimination (Brown and Dennis, 1968; Brown, 1969) and the matrix decomposition method of Cholesky (Wilkinson, 1965), need $O(n^3)$ computational operations for this (see Schwarz, Rutishauser, and Stiefel, 1968). For the same cost, the strategies of conjugate directions and conjugate gradients can execute $O(n)$ steps. Thus, in principle, the Newton-Raphson iteration offers no advantage in the quadratic case.

If the objective function is not quadratic, then

- $v^{(0)}$ does not in general point towards a minimum. The iteration rule (Equation (3.27)) must be applied repeatedly.

- $s^{(k)} = 1$ may lead to a point with a worse value of the objective function. The search diverges, e.g., when $\nabla^2 F(x^{(k)})$ is not positive-definite.
- It can happen that $\nabla^2 F(x^{(k)})$ is singular or almost singular. The Hessian matrix cannot be inverted.

Furthermore, it depends on the starting point $x^{(0)}$ whether a minimum, a maximum, or a saddle point is approached, or the whole iteration diverges. The strategy itself does not distinguish the stationary points with regard to type.

If the method does converge, then the convergence is better than of linear order (Goldstein, 1965). Under certain, very strict conditions on the structure of the objective function and its derivatives even *second order convergence* can be achieved (e.g., Polak, 1971); that is, the number of exact significant figures in the approximation to the minimum solution doubles from iteration to iteration. This phenomenon is exhibited in the solution of some test problems, particularly in the neighborhood of the desired extremum.

All the variations of the basic procedure to be described are aimed at increasing the reliability of the Newton iteration, without sacrificing the high convergence rate. A distinction is made here between *quasi-Newton strategies*, which do not evaluate the Hessian matrix explicitly, and *modified Newton methods*, for which first and second derivatives must be provided at each point. The only strategy presently known which makes use of higher than second order properties of the objective function is due to Biggs (1971, 1973).

The simplest modification of the Newton-Raphson scheme consists of determining the step length $s^{(k)}$ by a line search in the Newton direction $v^{(k)}$ (Equation (3.28)) until the relative optimum is reached (e.g., Dixon, 1972a):

$$F(x^{(k)} + s^{(k)}\, v^{(k)}) = \min_s \{F(x^{(k)} + s\, v^{(k)})\} \tag{3.30}$$

To save computational operations, the second partial derivatives can be redetermined less frequently and used for several iterations. Care must always be taken, however, that $v^{(k)}$ always points "downhill," i.e., the angle between $v^{(k)}$ and $-\nabla F(x^{(k)})$ is less than 90^0. The Hessian matrix must also be positive-definite. If the eigenvalues of the matrix are calculated when it is inverted, their signs show whether this condition is fulfilled. If a negative eigenvalue appears, Pearson (1969) suggests proceeding in the direction of the associated eigenvector until a point is reached with positive-definite $\nabla^2 F(x)$. Greenstadt (1967a) simply replaces negative eigenvalues by their absolute value and vanishing eigenvalues by unity. Other proposals have been made to keep the Hessian matrix positive-definite by addition of a correction matrix (Goldfeld, Quandt, and Trotter, 1966, 1968; Shanno, 1970a) or to include simple gradient steps in the iteration scheme (Dixon and Biggs, 1972). Further modifications, which operate on the matrix inversion procedure itself, have been suggested by Goldstein and Price (1967), Fiacco and McCormick (1968), and Matthews and Davies (1971). A good survey has been given by Murray (1972b).

Very few algorithms exist that determine the first and second partial derivatives numerically from trial step operations (Whitley, 1962; see also Wasscher, 1963c; Wegge, 1966). The inevitable approximation errors too easily cancel out the advantages of the Newton directions.

3.2.3.1 DFP: Davidon-Fletcher-Powell Method (Quasi-Newton Strategy, Variable Metric Strategy)

Much greater interest has been shown for a group of second order gradient methods that attempt to approximate the Hessian matrix and its inverse during the iterations only from first order data. This now extensive class of quasi-Newton strategies has grown out of the work of Davidon (1959). Fletcher and Powell (1963) improved and translated it into a practical procedure. The *Davidon-Fletcher-Powell* or *DFP* method and some variants of it are also known as *variable metric strategies.* They are sometimes also regarded as conjugate gradient methods, because in the quadratic case they generate conjugate directions. For higher order objective functions this is no longer so. Whereas the variable metric concept is to approximate Newton directions, this is not the case for conjugate gradient methods. The basic recursion formula for the DFP method is

$$x^{(k+1)} = x^{(k)} + s^{(k)}\, v^{(k)}$$

with

$$v^{(k)} = -H^{(k)T}\, \nabla F(x^{(k)})$$
$$H^{(0)} = I$$

and

$$H^{(k+1)} = H^{(k)} + A^{(k)}$$

The correction $A^{(k)}$ to the approximation for the inverse Hessian matrix, $H^{(k)}$, is derived from information collected during the last iteration; thus from the change in the variable vector

$$y^{(k)} = x^{(k+1)} - x^{(k)} = s^{(k)}\, v^{(k)}$$

and the change in the gradient vector

$$z^{(k)} = \nabla F(x^{(k+1)}) - \nabla F(x^{(k)})$$

it is given by

$$A^{(k)} = \frac{y^{(k)}\, y^{(k)T}}{y^{(k)T}\, z^{(k)}} - \frac{H^{(k)}\, z^{(k)}\, (H^{(k)}\, z^{(k)})^T}{z^{(k)T}\, H^{(k)}\, z^{(k)}} \tag{3.31}$$

The step length $s^{(k)}$ is obtained by a line search along $v^{(k)}$ (Equation (3.30)). Since the first partial derivatives are needed in any case they can be made use of in the one dimensional minimization. Fletcher and Powell do so in the context of a cubic Hermitian interpolation (see Sect. 3.1.2.3.4). A corresponding ALGOL program has been published by Wells (1965) (for corrections see Fletcher, 1966; Hamilton and Boothroyd, 1969; House, 1971). The first derivatives must be specified as functions, which is usually inconvenient and often impossible. The convergence properties of the DFP method have been thoroughly investigated, e.g., by Broyden (1970b,c), Adachi (1971), Polak (1971), and Powell (1971, 1972a,b,c). Numerous suggestions have thereby been made for improvements. Convergence is achieved if $F(x)$ is convex. Under stricter conditions it can be proved that the convergence rate is greater than linear and the sequence of iterations *converges quadratically*, i.e., after a finite number (maximum n) of steps the minimum of a quadratic function is located. Myers (1968) and Huang (1970) show that, if the same starting point is chosen and the objective function is of second order, the DFP algorithm generates the same iteration points as the conjugate gradient method of Fletcher and Reeves.

All these observations are based on the assumption that the computational operations, including the line searches, are carried out exactly. Then $H^{(k)}$ always remains positive-definite if $H^{(0)}$ was positive-definite and the minimum search is stable, i.e., the objective function is improved at each iteration. Numerical tests (e.g., Pearson, 1969; Tabak, 1969; Huang and Levy, 1970; Murtagh and Sargent, 1970; Himmelblau, 1972a,b), and theoretical considerations (Bard, 1968; Dixon, 1972b) show that rounding errors and especially inaccuracies in the one dimensional minimization frequently cause stability problems; the matrix $H^{(k)}$ can easily lose its positive-definiteness without this being due to a singularity in the inverse Hessian matrix. The simplest remedy for a singular matrix $H^{(k)}$, or one of *reduced rank*, is to forget from time to time all the experience stored within $H^{(k)}$ and to begin again with the unit matrix and simple gradient directions (Bard, 1968; McCormick and Pearson, 1969). To do so certainly increases the number of necessary iterations, but in optimization as in other activities it is wise to put safety before speed. Stewart (1967) makes use of this procedure. His algorithm is of very great practical interest since he obtains the required information about the first partial derivatives from function values alone by means of a cleverly constructed difference scheme.

3.2.3.2 Strategy of Stewart: Derivative-free Variable Metric Method

Stewart (1967) focuses his attention on choosing the length of the trial step $d_i^{(k)}$ for the approximation

$$g_i^{(k)} \simeq F_{x_i}\left(x^{(k)}\right) = \left.\frac{\partial F(x)}{\partial x_i}\right|_{x^{(k)}}$$

to the first partial derivatives in such a way as to minimize the influence of rounding errors on the actual iteration process. Two difference schemes are available:

$$g_i^{(k)} = \frac{1}{d_i^{(k)}}\left[F(x^{(k)} + d_i^{(k)}\, e_i) - F(x^{(k)})\right] \quad \text{(forward difference)}$$

and

$$g_i^{(k)} = \frac{1}{2\, d_i^{(k)}}\left[F(x^{(k)} + d_i^{(k)}\, e_i) - F(x^{(k)} - d_i^{(k)}\, e_i)\right] \quad \text{(central difference)} \quad (3.32)$$

Application of the one sided (forward) difference (Equation (3.32)) is preferred, since it only involves one extra function evaluation. To simplify the computation, Stewart introduces the vector $h^{(k)}$, which contains the diagonal elements of the matrix $(H^{(k)})^{-1}$ representing information about the curvature of the objective function in the coordinate directions.

The algorithm for determining the $g_i^{(k)}$, $i = 1(1)n$, runs as follows:

Step 0:

Set $\eta = \max\left\{\varepsilon_b\,, \varepsilon_c\,\frac{|g_i^{(k-1)}|\,|x_i^{(k)}|}{F(x^{(k)})}\right\}$

(ε_b represents an estimate of the error in the calculation of $F(x)$. Stewart sets $\varepsilon_b = 10^{-10}$ and $\varepsilon_c = 5 \cdot 10^{-13}$.)

Step 1:

If $\left(g_i^{(k-1)}\right)^2 \geq \eta\,\left|h_i^{(k-1)}\,F(x^{(k)})\right|$,

define $\delta_i' = 2\sqrt{\eta\,\frac{\left|F(x^{(k)})\right|}{\left|h_i^{(k-1)}\right|}}$ and $\delta_i = \delta_i'\left(1 - \frac{\delta_i'\,\left|h_i^{(k-1)}\right|}{3\,\delta_i'\,\left|h_i^{(k-1)}\right| + 4\,\left|g_i^{(k-1)}\right|}\right)$;

otherwise define

$$\delta_i' = 2\sqrt[3]{\eta\,\frac{\left|F(x^{(k)})\,g_i^{(k-1)}\right|}{(h_i^{(k-1)})^2}} \text{ and } \delta_i = \delta_i'\left(1 - \frac{2\,\left|g_i^{(k-1)}\right|}{3\,\delta_i'\,\left|h_i^{(k-1)}\right| + 4\,\left|g_i^{(k-1)}\right|}\right).$$

Step 2:

Set $d_i'^{(k)} = \delta_i\,\mathrm{sign}(h_i^{(k-1)})\,\mathrm{sign}(g_i^{(k-1)})$

and $d_i^{(k)} = \begin{cases} d_i'^{(k)}, & \text{if } d_i'^{(k)} \neq 0 \\ d_i^{(k-1)}, & \text{if } d_i'^{(k)} = 0. \end{cases}$

If $\left|\frac{h_i^{(k-1)}\,d_i^{(k)}}{2\,g_i^{(k-1)}}\right| \leq 10^{-\mu}$, use Equation (3.32); otherwise

replace $d_i^{(k)} \leftarrow \frac{1}{\left|h_i^{(k-1)}\right|}\left(-\left|g_i^{(k-1)}\right| + \sqrt{(g_i^{(k-1)})^2 + 2 \cdot 10^{\mu}\,\eta\,\left|F(x^{(k)})\,h_i^{(k-1)}\right|}\right)$

and use Equation (3.33). (Stewart chooses $\mu = 2$.)

Stewart's main algorithm takes the following form:

Step 0: (Initialization)
Choose an initial value $x^{(0)}$, accuracy requirements $\varepsilon_{a_i} > 0, i = 1(1)n$, and initial step lengths $d_i^{(0)}$ for the gradient determination, e.g.,

$$d_i^{(0)} = \begin{cases} 0.05\left|x_i^{(0)}\right|, & \text{if } x_i^{(0)} \neq 0 \\ 0.05, & \text{if } x_i^{(0)} = 0. \end{cases}$$

Calculate the vector $g^{(0)}$ from Equation (3.32) using the step lengths $d_i^{(0)}$.
Set $H^{(0)} = I, h_i^{(0)} = 1$ for all $i = 1(1)n$, and $k = 0$.

Step 1: (Prepare for line search)
Determine $v^{(k)} = -H^{(k)}\,g^{(k)}$.
If $k = 0$, go to step 3.
If $g^{(k)T}\,v^{(k)} < 0$, go to step 3.
If $h_i^{(k)} > 0$ for all $i = 1(1)n$, go to step 3.

Step 2: (Forget second order information)
Replace $H^{(k)} \leftarrow H^{(0)} = I$,
$h_i^{(k)} \leftarrow h_i^{(0)} = 1$ for all $i = 1(1)n$,
and $v^{(k)} \leftarrow -g^{(k)}$.

Step 3: (Line search and eventual break-off)
Determine $x^{(k+1)}$ such that
$F(x^{(k+1)}) = \min_s \{F(x^{(k)} + s\, v^{(k)})\}$.
If $F(x^{(k+1)}) \geq F(x^{(k)})$, end the search with result $x^{(k)}$ and $F(x^{(k)})$.

Step 4: (Prepare for inverse Hessian update)
Determine $g^{(k+1)}$ by the above difference scheme.
Construct $y^{(k)} = x^{(k+1)} - x^{(k)}$ and $z^{(k)} = g^{(k+1)} - g^{(k)}$.
If $k > n$ and $\left|v_i^{(k)}\right| < \varepsilon_{a_i}$ and $\left|y_i^{(k)}\right| < \varepsilon_{a_i}$ for all $i = 1(1)n$,
end the search with result $x^{(k+1)}$, $F(x^{(k+1)})$.

Step 5: (Update inverse Hessian)
Construct $H^{(k+1)} = H^{(k)} + A^{(k)}$ using Equation (3.31) and
$h_i^{(k+1)} = h_i^{(k)} + \frac{z_i^{(k)}}{v^{(k)T}\, z^{(k)}} \left(z_i^{(k)} \, [1 - \frac{s^{(k)}\, g^{(k)T}\, v^{(k)}}{v^{(k)T}\, z^{(k)}}] + 2\, s^{(k)}\, g_i^{(k)} \right)$
for all $i = 1(1)n$.

Step 6: (Main loop / termination criterion)
If the denominators are non-zero, increase $k \leftarrow k + 1$, and go to step 1;
otherwise terminate with result $x^{(k+1)}$, $F(x^{(k+1)})$.

In place of the cubic Hermitian interpolation, Stewart includes a quadratic Lagrangian interpolation as used by Powell in his conjugate directions strategy. Gradient calculations at the argument values are thereby avoided. One point, $x^{(k)}$, is given each time by the initial vector of the line search. The second, $x^{(k)} + s\, v^{(k)}$, is situated in the direction $v^{(k)}$ at a distance

$$s = \min \left\{ 1 \, , - \frac{2\, (F(x^{(k)}) - F_m)}{g^{(k)T}\, v^{(k)}} \right\}$$

F_m is an estimate of the value of the objective function at the minimum being sought. It must be specified beforehand. $s = 1$ is an upper bound corresponding to the length of a Newton-Raphson step. The third argument value is to be calculated from knowledge of the points $x^{(k)}$ and $x^{(k)} + s\, v^{(k)}$, their associated objective function values, and $g^{(k)T}\, v^{(k)}$, the derivative of the objective function at point $x^{(k)}$ in direction $v^{(k)}$. The sequence of Lagrangian interpolations is broken off if, at any time, the predicted minimum worsens the situation or lies at such a distance outside the interval that it is more than twice as far from the next point as the latter is from the midpoint.

Lill (1970, 1971) (see also Kovács and Lill, 1971) has published a complete ALGOL program for the derivative-free DFP strategy of Stewart; it differs slightly from the original only in the line search. Fletcher (1969b) reports tests that demonstrate the superiority of Stewart's algorithm to Powell's as the number of variables increases.

Brown and Dennis (1972) and Gill and Murray (1972) have suggested other schemes for obtaining the partial derivatives numerically from values of the objective function. Stewart himself reports tests that show the usefulness of his rules insofar as the results are completely comparable to others obtained with the help of analytically specified derivatives. This may be simply because rounding errors are in any case more significant here, due to the matrix operations, than for example in conjugate gradient methods. Kelley and Myers (1971), therefore, recommend carrying out the matrix operations with double precision.

3.2.3.3 Further Extensions

The ability of the quasi-Newton strategy of Davidon, Fletcher, and Powell (DFP) to construct Newton directions without needing explicit second partial derivatives makes it very attractive from a computational point of view. All efforts in the further rapid and intensive development of the concept have been directed to modifying the correction Equation (3.31) so as to reduce the tendency to instability because of rounding errors and inexact line searches while retaining as far as possible the quadratic convergence. There has been a spate of corresponding proposals and both theoretical and experimental investigations on the subject up to about 1973, for example:

Adachi (1973a,b)
Bass (1972)
Broyden (1967, 1970a,b,c, 1972)
Broyden, Dennis, and Moré (1973)
Broyden and Johnson (1972)
Davidon (1968, 1969)
Dennis (1970)
Dixon (1972a,b,c, 1973)
Fiacco and McCormick (1968)
Fletcher (1969a,b, 1970b, 1972b,d)
Gill and Murray (1972)
Goldfarb (1969, 1970)
Goldstein and Price (1967)
Greenstadt (1970)
Hestenes (1969)
Himmelblau (1972a,b)
Hoshino (1971)
Huang (1970, 1974)
Huang and Chambliss (1973, 1974)
Huang and Levy (1970)
Jones (1973)
Lootsma (1972a,b)
Mamen and Mayne (1972)
Matthews and Davies (1971)
McCormick and Pearson (1969)

McCormick and Ritter (1972, 1974)
Murray (1972a,b)
Murtagh (1970)
Murtagh and Sargent (1970)
Oi, Sayama, and Takamatsu (1973)
Oren (1973)
Ortega and Rheinboldt (1972)
Pierson and Rajtora (1970)
Powell (1969, 1970a,b,c,g, 1971, 1972a,b,c,d)
Rauch (1973)
Ribière (1970)
Sargent and Sebastian (1972, 1973)
Shanno and Kettler (1970a,b)
Spedicato (1973)
Tabak (1969)
Tokumaru, Adachi, and Goto (1970)
Werner (1974)
Wolfe (1967, 1969, 1971)

Many of the differently sophisticated strategies, e.g., the classes or families of similar methods defined by Broyden (1970b,c) and Huang (1970), are *theoretically equivalent.* They generate the same conjugate directions $v^{(k)}$ and, with an exact line search, the same sequence $x^{(k)}$ of iteration points if $F(x)$ is quadratic. Dixon (1972c) even proves this identity for more general objective functions under the condition that no term of the sequence $H^{(k)}$ is singular.

The important finding that under certain assumptions convergence can also be achieved without line searches is attributed to Wolfe (1967). A recursion formula satisfying these conditions is as follows:

$$H^{(k+1)} = H^{(k)} + B^{(k)}$$

where

$$B^{(k)} = \frac{(y^{(k)} - H^{(k)}\, z^{(k)})\,(y^{(k)} - H^{(k)}\, z^{(k)})^T}{(y^{(k)} - H^{(k)}\, z^{(k)})\, z^{(k)T}} \tag{3.33}$$

The formula was proposed independently by Broyden (1967), Davidon (1968, 1969), Pearson (1969), and Murtagh and Sargent (1970) (see Powell, 1970a). The correction matrix $B^{(k)}$ has rank one, while $A^{(k)}$ in Equation (3.31) is of rank two. *Rank one methods*, also called *variance methods* by Davidon, cannot guarantee that $H^{(k)}$ remains positive-definite. It can also happen, even in the quadratic case, that $H^{(k)}$ becomes singular or $B^{(k)}$ increases without bound. Hence in order to make methods of this type useful in practice a number of additional precautions must be taken (Powell, 1970a; Murray, 1972c). The following compromise proposal

$$H^{(k+1)} = H^{(k)} + A^{(k)} + \alpha^{(k)}\, B^{(k)} \tag{3.34}$$

where the scalar parameter $\alpha^{(k)} > 0$ can be freely chosen, is intended to exploit the advantages of both concepts while avoiding their disadvantages (e.g., Fletcher, 1970b). Broyden

(1970b,c), Shanno (1970a,b), and Shanno and Kettler (1970) give criteria for choosing suitable $\alpha^{(k)}$. However, the mixed correction, also known as *BFS* or *Broyden-Fletcher-Shanno formula*, cannot guarantee quadratic convergence either unless line searches are carried out. It can be proved that there will merely be a monotonic decrease in the eigenvalues of the matrix $H^{(k)}$. From numerical tests, however, it turns out that the increased number of iterations is usually more than compensated for by the saving in function calls made by dropping the one dimensional optimizations (Fletcher, 1970a). Fielding (1970) has designed an ALGOL program following Broyden's work with line searches (Broyden, 1965). With regard to the number of function calls it is usually inferior to the DFP method but it sometimes also converges where the variable metric method fails. Dixon (1973) defines a correction to the chosen directions,

$$v^{(k)} = -H^{(k)}\,\nabla F(x^{(k)}) + w^{(k)}$$

where

$$w^{(0)} = 0$$

and

$$w^{(k+1)} = w^{(k)} + \frac{(x^{(k+1)} - x^{(k)})^T\,\nabla F(x^{(k+1)})}{(x^{(k+1)} - x^{(k)})^T\,z^{(k)}}\,(x^{(k+1)} - x^{(k)})$$

by which, together with a matrix correction as given by Equation (3.35), quadratic convergence can be achieved without line searches. He shows that at most $n + 2$ function calls and gradient calculations are required each time if, after arriving at $v^{(k)} = 0$, an iteration

$$x^{(k+1)} = x^{(k)} - H^{(k)}\,\nabla F(x^{(k)})$$

is included. Nearly all the procedures defined assume that at least the first partial derivatives are specified as functions of the variables and are therefore exact to the significant figure accuracy of the computer used. The more costly matrix computations should wherever possible be executed with double precision in order to keep down the effect of rounding errors.

Just two more suggestions for derivative-free quasi-Newton methods will be mentioned here: those of Greenstadt (1972) and of Cullum (1972). While Cullum's algorithm, like Stewart's, approximates the gradient vector by function value differences, Greenstadt attempts to get away from this. Analogously to Davidon's idea of approximating the Hessian matrix during the course of the iterations from knowledge of the gradients, Greenstadt proposes approximating the gradients by using information from objective function values over several subiterations. Only at the starting point must a difference scheme for the first partial derivatives be applied. Another interesting variable metric technique described by Elliott and Sworder (1969a,b, 1970) combines the concept of the stochastic approximation for the sequence of step lengths with the direction algorithms of the quasi-Newton strategy.

Quasi-Newton strategies of degree one are especially suitable if the objective function is a sum of squares (Bard, 1970). Problems of minimizing a sum of squares arise for example from the problem of solving systems of simultaneous, non-linear equations, or determining the parameters of a mathematical model from experimental data (non-linear *regression* and *curve fitting*). Such objective functions are easier to handle because Newton directions can be constructed straight away without second partial derivatives, as long as the Jacobian matrix of first derivatives of each term of the objective function is given. The oldest iteration procedure constructed on this basis is variously known as the *Gauss-Newton* (Gauss, 1809) method, *generalized least squares* method, or *Taylor series method.* It has all the advantages and disadvantages of the Newton-Raphson strategy. Improvements on the basic procedure are given by Levenberg (1944) and Marquardt (1963). Wolfe's *secant method* (Wolfe, 1959b; see also Jeeves, 1958) is the forerunner of many variants which do not require the Jacobian matrix to be specified at the start but construct it in the course of the iterations. Further details will not be described here; the reader is referred to the specialist literature, again up to 1973:

Barnes, J.G.P. (1965)
Bauer, F.L. (1965)
Beale (1970)
Brown and Dennis (1972)
Broyden (1965, 1969, 1971)
Davies and Whitting (1972)
Dennis (1971, 1972)
Fletcher (1968, 1971)
Golub (1965)
Jarratt (1970)
Jones (1970)
Kowalik and Osborne (1968)
Morrison (1968)
Ortega and Rheinboldt (1970)
Osborne (1972)
Peckham (1970)
Powell (1965, 1966, 1968b, 1970d,e, 1972a)
Powell and MacDonald (1972)
Rabinowitz (1970)
Ross (1971)
Smith and Shanno (1971)
Späth (1967) (see also Silverman, 1969)
Stewart (1973)
Vitale and Taylor (1968)
Zeleznik (1968)

Brent (1973) gives further references. Peckham's strategy is perhaps of particular interest. It represents a modification of the simplex method of Nelder and Mead (1965) and Spendley (1969) and in tests it proves to be superior to Powell's strategy (1965) with regard to the number of function calls. It should be mentioned here at least that non-linear regression, where parameters of a model that enter the model in a non-linear way (e.g., as exponents) have to be estimated, in general requires a global optimization method because the squared sum of residuals defines a multimodal function.

Reference has been made to a number of publications in this and the preceding chapter in which strategies are described that can hardly be called genuine hill climbing methods; they would fall more naturally under the headings of mathematical programming or functional optimization. It was not, however, the intention to give an introduction to the basic principles of these two very wide subjects. The interested reader will easily find out that although a nearly exponentially increasing number of new books and journals have become available during the last three decades, she or he will look in vain for new direct search strategies in that realm. Such methods form the core of this book.