HYBRID COMPUTER OPTIMIZATION OF SYSTEMS WITH RANDOM PARAMETERS

by

Robert Cantey White, Jr.

Get DRF

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THE UNIVERSITY OF ARIZONA

NGL-03-002-024

1970

ACKNOWLEDGMENTS

The author wishes to thank Professor Granino A. Korn for his suggestion of the dissertation topic and for his helpful guidance during the research and writing. This project is part of a continuing study of hybrid analog-digital computation directed by Professor Korn and supported by the National Aeronautics and Space Administration under NASA Grant NGR-03-002-024. The author is also grateful for support of his Ph. D. program through a NASA Pre-doctoral Traineeship, a General Electric Foundation Graduate Fellowship, and a Graduate Associateship in Research in the Engineering Experiment Station of The University of Arizona.

STATEMENT BY AUTHOR

TABLE OF CONTENTS

TABLE OF CONTENTS--*Continued*

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS--Continued

LIST OF TABLES

ABSTRACT

This thesis presents a hybrid-computer Monte-Carlo method for the optimization of systems containing random parameters. In the design of a dynamical system, the values of a set of system parameters may be chosen so as to optimize a performance criterion. If, however, the manufacturing process results in production variations in these parameters, the optimal system becomes an idealization which cannot, in general, be realized by the systems actually manufactured. In this case it may be advantageous to treat the system parameters as random variables having, for example, Gaussian probability distributions. Then parameter mean values and variances can be chosen so as to optimize a criterion function which includes average system performance and also the cost of manufacturing systems with certain parameter variances.

In order to solve this type of problem, the dynamical system, including the random variations in the system parameters, is simulated on a fast repetitive analog computer (The University of Arizona's ASTRAC-II) and the average system performance is estimated by the Monte-Carlo method. A small digital computer (Digital Equipment Corporation PDP-9) controls the operation of the analog machine and implements an optimization algorithm for determining the optimal parameter means and variances.

Since an estimate of the average system performance is a random variable, the optimization algorithm must operate with noisy measurements of the criterion function. A review of the literature on parameter optimization led to the development of a creeping-random-search algorithm for optimization in the presence of noise. Incorporated in the optimization program are provisions for interaction between the operator and the algorithm by way of a cathode-ray-tube display console and the accumulator switches on the PDP-9.

The method is applied to the optimization of the means and variances of two guidance-unit parameters in a hypothetical radar-homing missile. With differential equation solution rates of approximately 500 runs per second, typical optimization times are on the order of 6-7 minutes. It is found that optimizations with lower bound constraints on the parameter variances result in optimal mean values different from those for the unconstrained case.

CHAPTER 1

INTRODUCTION

A common approach to the design of an engineering system is first to choose a general configuration in which the values of several parameters are left undetermined; these values are then chosen so as to optimize some criterion of performance. If many of these systems are to be manufactured, however, it may be difficult and/or costly to ensure that the parameter values are very close to the optimum. The system with optimum parameter values then becomes only an idealization which is not, in general, realized by the systems that are manufactured. In such a situation it may be advantageous to model the output of the manufacturing process as a statistical ensemble of systems with parameters having, for example, Gaussian probability distributions. Then, parameter mean values and variances could be chosen to optimize a criterion function which would include average system performance as well as the cost of manufacturing systems with certain parameter variances.

In this thesis, a hybrid-computer method employing a fast repetitive analog computer (ASTRAC-II) and a digital computer (PDP-9), is developed for the simulation and optimization of an ensemble of systems with random parameters. The method is applied to the simultaneous optimization of the means and variances of two parameters in a hypothetical radar-homing missile.

1.1 Problem Definition

Let us consider an ensemble of systems identical except for the values of a set of k system parameters $\underline{p} = (p_1, p_2, \ldots, p_k)^t$. These parameters are assumed to be statistically independent random variables. A sample system from the ensemble is defined by a specific ordered set $(p_1, p_2, \ldots, \mathring{p}_k)$. The situation is pictured in Fig. 1.1.

It is assumed that the type of probability distribution for each random parameter is specified, but that the constants which precisely define these distributions may be varied. These constants are termed distribution constants and are represented by $\underline{x} = (x_1, x_2, \ldots, x_n)^t$. For example, suppose the parameters p_j are Gaussian with respective means μ_j and standard deviations σ_j, where the μ_j's and σ_j's may be chosen by the design engineer. Then $\underline{x} = (\underline{\mu}, \underline{\sigma}) = (\mu_1, \ldots, \mu_k; \sigma_1, \ldots, \sigma_k)^t$ and $n = 2k$.

For any sample system in the ensemble, we define a measure of system performance, a performance index J, which is a function of the random parameters and is, therefore, a random variable:

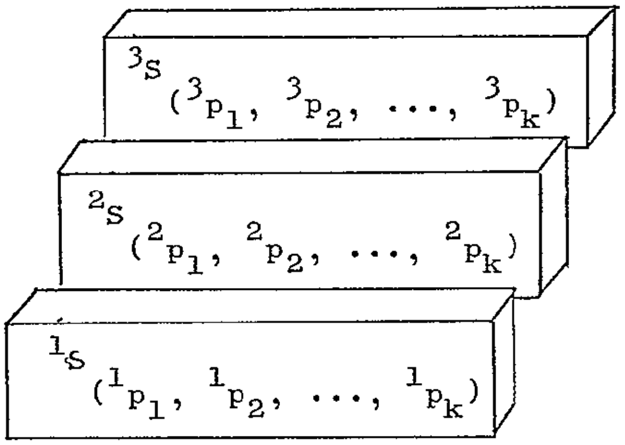

^3S
$(^3p_1,\ ^3p_2,\ \ldots,\ ^3p_k)$
^2S
$(^2p_1,\ ^2p_2,\ \ldots,\ ^2p_k)$
^1S
$(^1p_1,\ ^1p_2,\ \ldots,\ ^1p_k)$

Fig. 1.1 Three samples ^1S, ^2S, ^3S from an ensemble of systems.

$$J = J(p_1, p_2, \dots, p_k) \tag{1.1}$$

The ensemble average (expected value) of J is a measure of the average system performance. This expectation of J, $E\{J\} = \Psi$, termed the <u>average performance index</u>, is a function of the distribution constants of the random parameters:

$$E\{J\} = \Psi = \Psi(x_1, \dots, x_n) \tag{1.2}$$

For a given set of distribution constants we may also define a <u>cost function</u>, $C(\underline{x})$, as a measure of the cost associated with manufacturing systems with these distribution constants. Typically, $C(\underline{x})$ will depend specifically on the parameter tolerances, σ_i. The cost function and the average performance index are summed to form the <u>criterion function</u>, $F(\underline{x})$.

$$F(\underline{x}) = \Psi(\underline{x}) + C(\underline{x}) \tag{1.3}$$

The problem to be solved is that of optimizing (minimizing or maximizing) F with respect to $x_1, \dots, x_n$, subject to a set of m inequality constraints on the x_i's.

$$\phi_i(\underline{x}) \leq 0 \qquad i = 1, \dots, m \tag{1.4}$$

The inequality constraints may represent restrictions imposed by the design engineer or constraints required for proper definition of the distribution functions of the system parameters. For example, if p_i is Gaussian with mean μ_1 and variance σ_1^2, then the designer may require $-10 \leq \mu_1 \leq 5$, and we must have $\sigma_1 \geq 0$.

1.2 Previous Work

The problem of optimizing an ensemble of systems with respect to parameter variances as well as parameter mean values has been recognized for some time, but little work in this area has been accomplished. The exceedingly large number of system simulations necessary to evaluate and optimize the criterion function for a dynamic system with random parameters makes the solution to such problems impractical without the use of very fast hybrid computers, which have become available only in recent years. In 1959 McGhee and Levine (1964) employed Monte-Carlo simulation in the optimization of production tolerances for two Gaussian parameters in a radar-homing missile (this paper is discussed more thoroughly in Chapter 4). Parameter mean values were selected prior to the simulation, and the criterion function was then estimated for sixteen combinations of tolerance values. With a slow analog computer, approximately one week of computing time was required, demonstrating the need for a fast repetitive machine in solving a problem of any complexity. Korn (1966) has outlined the problem of hybrid-computer optimization of systems with parameters subject to production variations. Note that simultaneous optimization of mean values as well

<u>as variances will, in general, result in optimum mean values different from those for the case where all variances are set to zero</u>.

Recently Bohling and O'Neill (1970) have presented a hybrid-computer approach to parameter tolerance <u>analysis</u>. With the aid of an interactive display system, the operator can quickly evaluate the effects of parameter tolerances on system performance and reject unsatisfactory designs without waiting for the accumulation of large statistical samples. This type of operator-program interaction, which provides insight into system behavior as well as a saving in computer time, could be equally beneficial in parameter optimization.

1.3 Solution Approach

The solution of the parameter optimization problem outlined in Section 1.1 may be divided into two parts: evaluating the criterion function $F(\underline{x})$ and choosing the x_i's to optimize $F(\underline{x})$.

The main problem in evaluating F is the calculation of $\Psi = E\{J(p_1, p_2, \ldots, p_k)\}$. This expectation may be calculated analytically for only the simplest of systems and performance indices. For systems of any complexity, a natural method of calculating Ψ is to <u>estimate</u> it by Monte-Carlo simulation. With this approach, the mathematical model of the system is implemented by a computer. For a given set of distribution constants sample values of the random system parameters are obtained from noise generators, and the system is operated or "run" many times to obtain an estimate of the average performance index Ψ. For systems described by differential equations, this task is a natural one for a high-speed iterative analog computer, which is capable of solving differential equations much more quickly than a digital machine.

The job of optimizing the Monte-Carlo estimate of F is most easily handled by a digital computer, which can examine the performance index estimate and implement sophisticated strategies for locating the optimal parameter values. The main difficulty in solving the parameter optimization problem results from our inability to measure $\Psi(\underline{x})$ exactly. The estimate of Ψ from many analog computer runs will, in general, contain an error which can lead to a wrong decision in the search for the optimum parameters. For the reasons discussed in Chapter 3, a <u>creeping random search algorithm</u> was chosen for the optimization strategy.

The division of the problem into these two tasks, estimation of $\Psi(\underline{x})$ and optimization, suggests the use of a hybrid computer consisting of a small digital computer interfaced to a high-speed analog machine. Such a computing system is employed for the problem solved here. The digital computer is a Digital Equipment Corporation PDP-9, which has an 18-bit word length and 16K of core memory. The University of Arizona's ASTRAC-II is a $\pm$ 10. volt repetitive analog computer capable of differential-equation solution rates of 1000 runs per second.

A review of the literature on parameter optimization was undertaken in preparation for selecting an effective search strategy for noisy criterion functions. This survey is the subject of Chapter 2. The algorithms developed for the estimation of the criterion function and optimization are discussed in Chapter 3. Chapter 4 describes the application of the method to the radar-homing missile problem, and some general remarks and conclusions are given in Chapter 5.

CHAPTER 2

A SURVEY OF PARAMETER OPTIMIZATION TECHNIQUES

2.1 Introduction and Notation

During the past fifteen years the fields of optimum systems design and optimal control have produced a large number of parameter optimization techniques. This survey reviews the important techniques available and attempts to evaluate their relative worth. Since no one method is best for all situations, attention is focused on factors which determine the suitability of a method for a particular class of problems. These factors include the type of criterion function to be minimized, constraints on the parameters, errors in measuring the criterion function, and the computing equipment to be used. The techniques discussed have been chosen for their applicability to the wide range of criterion functions found in engineering problems. Thus, algorithms designed for rather specific functions are not treated here. Such methods include linear programming, Gauss's least squares, and geometric programming, which are discussed by Wilde and Beightler (1967).

There are several references which review or discuss parameter optimization methods in detail. The most comprehensive and thorough treatment is found in Wilde and Beightler (1967), which covers most of the methods mentioned here, with the exception of the creeping random techniques and stochastic approximation. The latter topic is discussed by Wilde (1964). Creeping random methods are treated by Rastrigin (1967), Korn (1966), and Bekey and Karplus (1968). McGhee (1967) gives an introduction to gradient methods. Techniques especially suitable for analog or hybrid computers are described by Korn and Korn (1964), Bekey (1964), and Bekey and Karplus (1968). A more mathematical treatment of parameter optimization, specifically of the nonlinear programming problem, may be found in Saaty and Bram (1964), which contains a full treatment of techniques for handling constraints. Some other general references with discussions of several parameter optimization methods are Leon (1964), Lavi and Vogl (1966), Carnahan (1966), Fleischer (1966), Kopp (1967), Hague and Glatt (1968), and Spang (1962). A bibliography of hybrid-computer parameter optimization methods is given by Gilbert (1967).

Formal definitions of the general parameter optimization (nonlinear programming) problem and related mathematical concepts are given by Korn and Korn (1968) and Saaty and Bram (1964). The notation to be used here is introduced in the following problem statement.

Determine the ordered set of n unknown <u>parameters</u> $\underline{x} \equiv (x_1, x_2, \ldots, x_n)^\tau$ which <u>optimizes</u> (minimizes or maximizes) the criterion function (objective function, performance index)

$$F(\underline{x}) \tag{2.1}$$

subject to the m inequality constraints

$$\phi_i(\underline{x}) \geq 0 \quad (\text{or} \leq) \quad (i=1, \ldots, m) \tag{2.2}$$

The optimal parameter values and associated criterion function value will be denoted by $\underline{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)^t$ and F^*. The set of all $\underline{x}$ satisfying the constraints (2.2) defines a region R called the feasible region. For convenience, all optimization problems are considered here as minimization problems. In some situations, constraints are not present or may be effectively eliminated (unconstrained optimization).

In the evaluation of optimization algorithms the notion of convergence is used to describe how quickly the search proceeds to the optimum point. In particular, some algorithms are said to exhibit quadratic convergence, which has been defined in several ways in the literature. Wilde and Beightler (1967) state that an algorithm capable of finding the minimum of a quadratic function of n variables after measuring n gradients is said to converge quadratically. McGhee (1967) defines quadratic convergence in the following way. Let $\Delta\underline{x}$ be the parameter step vector computed by the algorithm. Then quadratic convergence implies that as $\delta\underline{x} = \underline{x}^* - \underline{x}$ approaches zero, the ratios of the components of $\Delta\underline{x}$ and $\delta\underline{x}$, $\Delta x_i/\delta \underline{x}_j$, approach 1 for $i = j$ and approach zero for $i \neq j$. According to Box (1966) and Fletcher and Reeves (1964), an algorithm enjoying quadratic convergence will locate the minimum of a quadratic function in a finite number of steps. Unless otherwise stated, this last definition will be adopted for discussions here.

The notions of <u>quasi-quadratic functions</u> and <u>quasi-quadratic convergence</u> are used by Wilde and Beightler (1967). Let $F(\underline{x})$ be a quadratic function of $\underline{x}$, and let h be a monotonic function. Then

$$y(\underline{x}) = h[F(\underline{x})]$$

is said to be quasi-quadratic, and we shall describe an algorithm capable of minimizing a quasi-quadratic function in a finite number of steps as converging quasi-quadratically.

The optimization techniques described here have been grouped under the headings: gradient descent methods, conjugate search-direction methods, quadratic fit methods, direct search methods, random methods, and stochastic approximations. The discussions are carried out for the unconstrained case, Section 2.8 describes methods for handling constraints. Comparative evaluations of the methods on the basis of results from test functions and practical problems are given in Section 2.9.

2.2 Gradient Descent Methods

The techniques discussed in this section assume a smooth objective function and make use of first-order partial derivatives to determine the optimizing steps. These methods include steepest descent schemes and Partan (McGhee, 1967, Wilde, 1964).

2.2.1 Steepest Descent

A smooth function $F(\underline{x})$ may be represented locally about any point $\underline{x}^o$ by a Taylor series:

$$F(\underline{x}^o+\Delta\underline{x}) = F(\underline{x}^o) + \underline{\nabla F}(\underline{x}^o)^t\Delta\underline{x} + O(\Delta\underline{x}^2) \tag{2.3}$$

where

$$\underline{\nabla F}(\underline{x}^o) = \underline{g}(\underline{x}^o) = \underline{g}^o = \begin{bmatrix} \left.\frac{\partial F}{\partial x_1}\right|_{\underline{x}^o} \\ \left.\frac{\partial F}{\partial x_2}\right|_{\underline{x}^o} \\ \cdot \\ \cdot \\ \cdot \\ \left.\frac{\partial F}{\partial x_n}\right|_{\underline{x}^o} \end{bmatrix} \tag{2.4}$$

and $O(\Delta\underline{x}^2)$ indicates a remainder consisting of terms of second-order and higher in the Δx_i. For small $\Delta\underline{x}$, the term linear in $\Delta\underline{x}$ is dominant, and to make $F(\underline{x}^o + \Delta\underline{x}) < F(\underline{x}^o)$ we take a step in the direction $-g(\underline{x}^o)$. To show that

$F(\underline{x})$ can be decreased by such a step, we let $\Delta\underline{x} = -\alpha g(\underline{x}^o)$, $\alpha > 0$. Then,

$$F(\underline{x}^o+\Delta\underline{x}) - F(\underline{x}^o) = -\alpha \underline{g}(\underline{x}^o)^t \underline{g}(\underline{x}^o) + O[\left|-\alpha\underline{g}(\underline{x}^o)\right|^2]$$

$$< 0 \text{ for small } \alpha.$$

The choice of α is critical in determining the speed of convergence; for small α, convergence is slow, and too large an α may result in no convergence. While there are many schemes for choosing α, probably the most used are the Newton-Raphson and "optimum gradient" methods.

The Newton-Raphson technique (McGhee, 1967) uses the representation of Eq. (2.3) and, neglecting the higher order terms, finds $\alpha = \alpha_o$ such that $F(\underline{x}^o + \Delta\underline{x}) = 0$. Thus,

$$\alpha_o = \frac{F(\underline{x}^o)}{\underline{g}^{ot}\underline{g}^o} .$$

This step size may locate a point $\underline{x}^o + \Delta\underline{x}$ such that $F(\underline{x}^o + \Delta\underline{x}) > F(\underline{x}^o)$, and it is possible for the Newton-Raphson method never to converge, as shown in the example of Fig. 2.1. On the other hand, this technique can be effective in avoiding local minima (Fig. 2.2).

The problem of instability can be avoided by determining α by the optimum gradient method (McGhee, 1967; Bekey and McGhee, 1964). Since $F(\underline{x})$ is known to decrease in the negative gradient direction for some small α, there exists an α^* on $(o, \alpha_o]$ such that $F(\underline{x}^o + \alpha^*\Delta\underline{x}) \leq F(\underline{x}^o + \alpha\Delta\underline{x})$

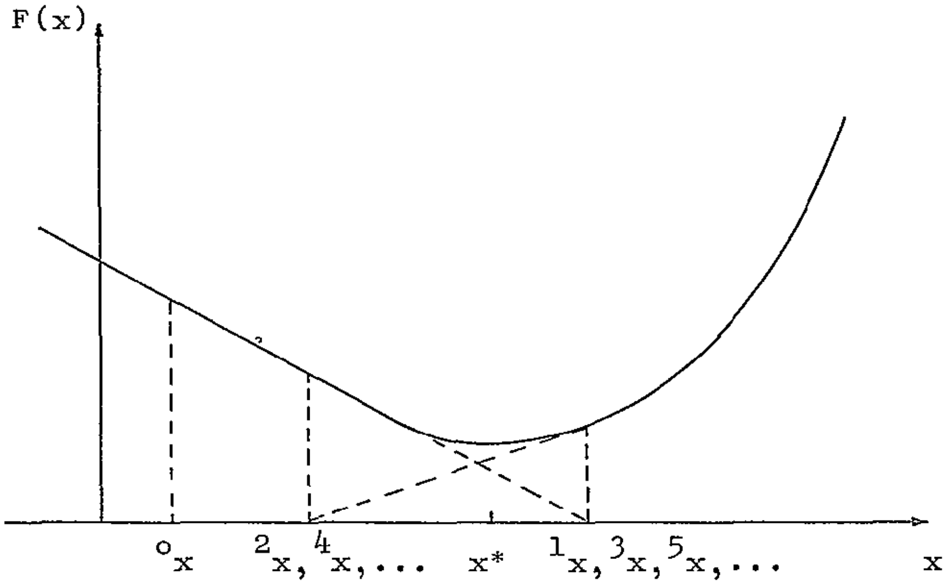


Fig. 2.1 An example of non-convergence with the Newton-Raphson method.

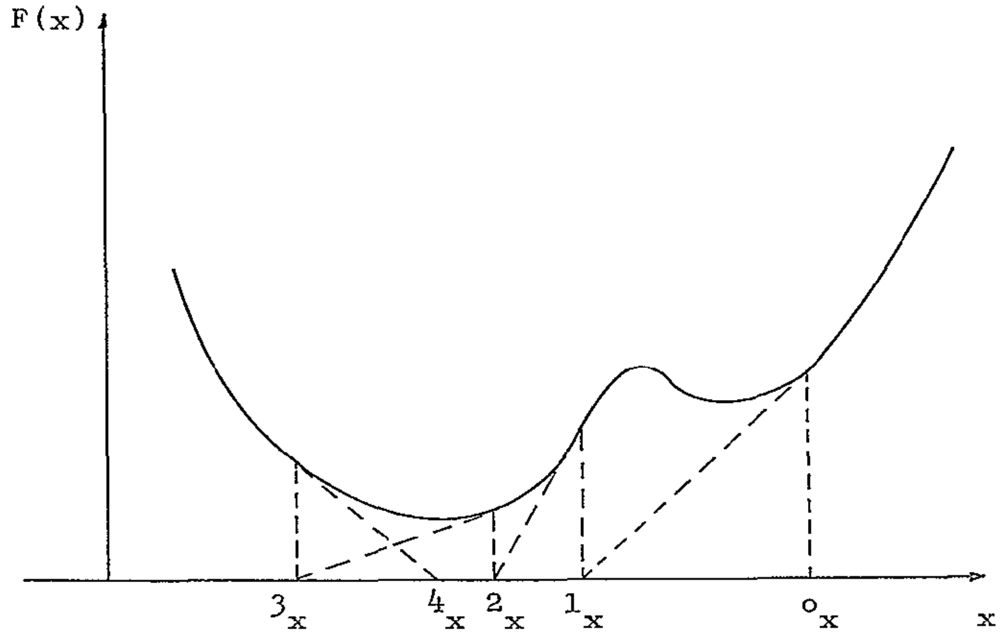


Fig. 2.2 An example showing avoidance of a local minimum with the Newton-Raphson method.

where α is any other scale factor on $(0,\alpha_o]$. The optimum gradient method uses a one-dimensional search to locate α^* for each step in the steepest-descent direction.

Steepest-descent methods were of the first to be used in optimization and have been applied successfully to many problems, especially in the initial stages of the search. Convergence, however, tends to be very slow near the optimum, and the method may fail altogether for functions with irregular parameter landscapes. In addition, since the direction of the gradient vector depends on the scaling of the parameters, x_i, the performance is strongly dependent on this scaling, problems with long, narrow contours will be more difficult to solve than ones with nearly circular contours. When gradient information is available, more modern methods such as Partan or the conjugate direction techniques are superior.

2.2.2 Parallel Tangents (Partan)

An attempt to speed up the convergence of gradient descent algorithms led to the method of parallel tangents (Partan), which was developed by Shah, Buehler, and Kempthorne (1964) after Forsythe and Motzkin's (1951) suggestion of a steepest descent acceleration technique in two dimensions. The two versions, steepest-descent (or gradient) Partan and general Partan, are discussed in detail by Buehler, Shah, and Kempthorne (1964), Shah et al. (1964), and Wilde (1964).

Steepest-descent Partan alternates steepest descent steps with acceleration steps as shown in Fig. 2.3. [In this discussion of Partan a "step" implies a minimization of $F(\underline{x})$ along a line.] For general Partan acceleration steps alternate with steps along lines parallel to planes which are tangent to $F(\underline{x})$ at previous even-numbered points ${}^{J}\underline{x}$ (Fig. 2.4. π_{J} = tangent plane at ${}^{J}\underline{x}$). General Partan has the property of scale invariance, which is usually considered an advantage in minimizing general functions. With either method a quasi-quadratic function of n variables is minimized in 2n or less steps. To carry on the algorithms for general functions after 2n steps either method can be restarted at the point ${}^{2n}\underline{x}$ (iterated Partan), or steepest descent Partan may simply be continued (continued Partan). The partial derivatives $\partial F/\partial x_{i}$ must be evaluated or approximated before alternate steps to obtain the gradient for steepest descent Partan or the tangent plane for general Partan. Harkins (1964) has found the very interesting result that convergence can be <u>improved</u> by inaccuracies in determining the minimum along a line. He suggested using only one to five points with a golden section search.

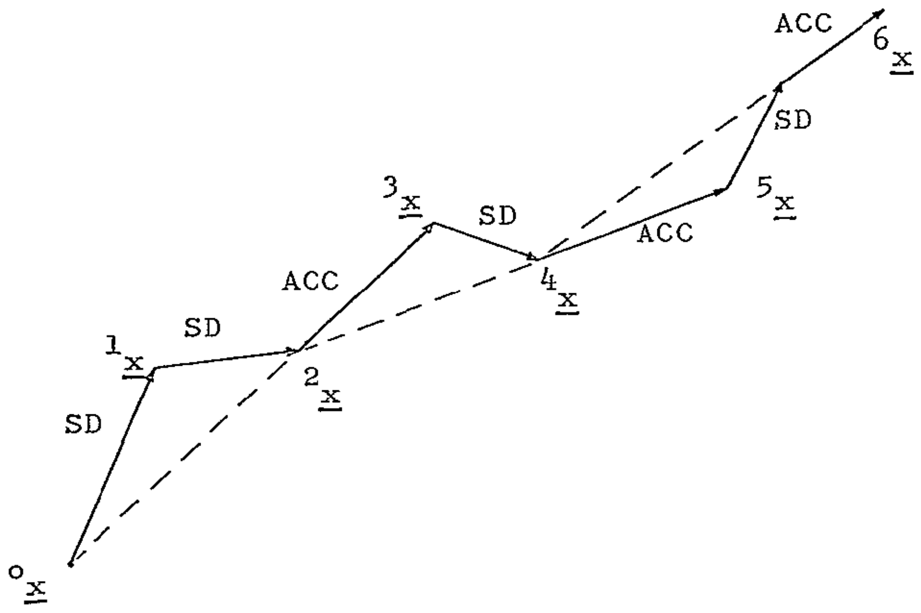


Fig. 2.3 Schematic diagram of steepest descent Partan.

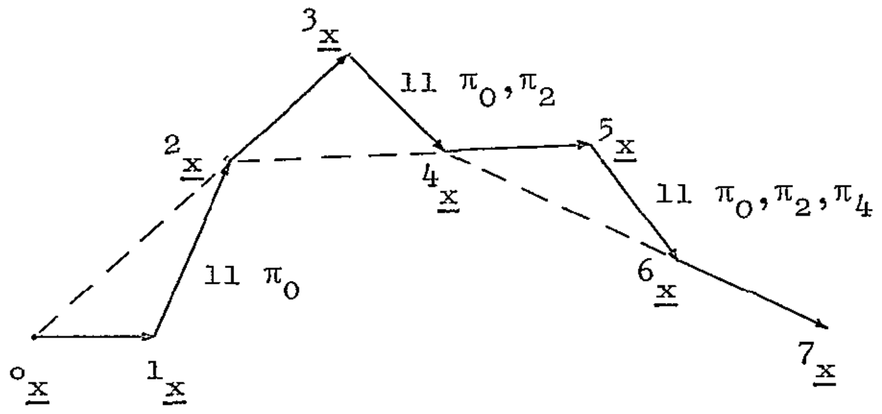


Fig. 2.4 Schematic diagram of general Partan.

2.3 Conjugate Search-Direction Methods

The techniques discussed in this section are designed to minimize a quadratic function by a series of one-dimensional minimizations along lines termed conjugate directions. For most of the methods, a quadratic function of n variables is minimized with n one-dimensional minimizations. To the extent that a non-quadratic function to be minimized can be approximately represented by a quadratic, these methods provide rapid convergence, especially in a region near the optimum, where the first- and second-order terms of a Taylor series expansion of a smooth function dominate. The conjugate-direction algorithms perform well on difficult test functions and have been used successfully in the solution of optimal control problems (Birta and Trushel, 1969, Lasdon, Mitter, and Waren, 1967). An introduction to some general properties of conjugate directions is followed by discussions of several algorithms.

Let $F(\underline{x})$ be a quadratic function of n variables x_i,

$$F(\underline{x}) = \frac{1}{2}\,\underline{x}^t A \underline{x} + \underline{b}^t \underline{x} + c \qquad (2.5)$$

with gradient

$$\underline{g}(\underline{x}) = A\underline{x} + \underline{b} \qquad (2.6)$$

where A is positive definite and symmetric.

<u>2.3.1 Conjugate Direction Properties</u>

A set of n independent directions ${}^{o}\underline{d}$, ${}^{1}\underline{d}$, ..., ${}^{n-1}\underline{d}$ are conjugate with respect to a positive semi-definite matrix B (B-conjugate) if

$$
\begin{aligned}
{}^{i}\underline{d}^{t} B\, {}^{j}\underline{d} &= 0 \qquad i \neq j \\
{}^{i}\underline{d}^{t} B\, {}^{i}\underline{d} &= 0
\end{aligned}
\tag{2.7}
$$

The importance of conjugate directions derives from the property that <u>n successive minimizations in the A-conjugate directions will locate the minimum of</u> F($\underline{x}$). To see this (Fletcher and Reeves, 1964), let ${}^{o}\underline{d}$, ${}^{1}\underline{d}$, ..., ${}^{n-1}\underline{d}$ be A-conjugate, and let a step from ${}^{i}\underline{x}$ to ${}^{i+1}\underline{x}$ be determined by

$$
{}^{i+1}\underline{x} = {}^{i}\underline{x} + {}^{i}\alpha\, {}^{i}\underline{d} \tag{2.8}
$$

where ${}^{i}\alpha$ is chosen such that

$$
\underline{g}^{t}({}^{i+1}\underline{x})\, {}^{i}\underline{d} = {}^{i+1}\underline{g}^{t}\ {}^{i}\underline{d} = 0 \tag{2.9}
$$

(i.e., ${}^{i}\alpha$ minimizes F along the direction ${}^{i}\underline{d}$). Iteration on (2.8) from ${}^{j+1}\underline{x}$ to ${}^{n}\underline{x}$ yields

$$
{}^{n}\underline{x} = {}^{j+1}\underline{x} + \sum_{i=j+1}^{n-1} {}^{i}\alpha\, {}^{i}\underline{d} \qquad (0 \leq j \leq n-1) \tag{2.10}
$$

As a convenience, let us assume a change of variables so that $\underline{b} = \underline{0}$. It then follows from (2.6) that

$$ {}^{n}\underline{g} = {}^{j+1}\underline{g} + \sum_{i=j+1}^{n-1} {}^{i}\alpha\, A\, {}^{i}\underline{d} \tag{2.11}$$

From (2.9) we have

$$ {}^{n}\underline{g}^{t}\, {}^{j}\underline{d} = \sum_{i=j+1}^{n-1} {}^{i}\alpha\, {}^{i}\underline{d}^{t} A\, {}^{j}\underline{d} \tag{2.12}$$

and as ${}^{o}\underline{d}$, ${}^{1}\underline{d}$, ..., ${}^{n-1}d$ are A-conjugate,

$$ {}^{n}\underline{g}^{t}\, {}^{j}\underline{d} = 0 \tag{2.13}$$

Since the n independent ${}^{j}\underline{d}$'s constitute a basis for E^n, ${}^{n}\underline{g} = \underline{0}$, which is the condition for ${}^{n}\underline{x}$ to be the minimum of (2.5).

This same property may be demonstrated in a slightly different way. Pearson (1968) shows that since ${}^{o}\underline{d}$, ${}^{1}\underline{d}$, ..., ${}^{n-1}\underline{d}$ constitute a basis, any $\underline{x}\epsilon E^n$ can be represented in terms of the ${}^{i}\underline{d}$'s, and $F(\underline{x}) = F({}^{o}\underline{d}, {}^{1}\underline{d}, \ldots, {}^{n-1}\underline{d})$ may be decomposed into n independent terms (each depending on only one ${}^{i}\underline{d}$) to be minimized separately.

2.3.2 Conjugate Direction Algorithms

Pearson (1968) has presented a unified treatment of a class of conjugate-direction algorithms. One, the projected-gradient algorithm, is based on the fact that conjugate directions may be generated by requiring that successive steps, ${}^{i}\underline{s} = {}^{i+1}\underline{x} - {}^{i}\underline{x}$, be made orthogonal to previous gradient differences, i.e.,

$$({}^{j+1}\underline{g} - {}^{j}\underline{g})^{t}\ {}^{i}\underline{s} \equiv {}^{j}\underline{y}^{t}\ {}^{i}\underline{s} = 0 \quad (j \leq i - 1) \tag{2.14}$$

This leads to the following method. (Pearson's numbering of the algorithms is retained here.)

Algorithm 1--Projected Gradient

Choose an initial point ${}^{o}\underline{x}$ and an initial positive-definite symmetric matrix ${}^{o}H$. Set $i = 0$.

1. Compute the search direction

$$^{i}\underline{d} = - {}^{i}H\ {}^{i}\underline{g} \tag{2.15}$$

2. Locate the next point ${}^{i+1}\underline{x}$ by minimizing $F({}^{i}\underline{x} + {}^{i}\alpha{}^{i}\underline{d})$ with respect to ${}^{i}\alpha$ (${}^{i}\alpha > 0$).

$$\begin{aligned} {}^{i+1}\underline{x} &= {}^{i}\underline{x} + {}^{i}\alpha{}^{i}\underline{d} \\ &= {}^{i}\underline{x} + {}^{i}\underline{s} \end{aligned} \tag{2.16}$$

3. Update the matrix ${}^{i}H$ by

$${}^{i+1}H = {}^{i}H - \frac{{}^{i}H{}^{i}\underline{y}{}^{i}\underline{y}^{t}{}^{i}H}{{}^{i}\underline{y}^{t}{}^{i}H{}^{i}\underline{y}} \tag{2.17}$$

and return to step 1 with i replaced by $i+1$.

After not more than n iterations (each consisting of steps 1-3), ${}^{i}\underline{x} = \underline{x}^{*}$ and ${}^{i}H = 0$.

The other algorithms considered by Pearson, including the well-known Fletcher-Powell-Davidon variable metric method, are based on the following idea.

Let ${}^{i}S = [{}^{o}\underline{s}, {}^{1}\underline{s}, \ldots, {}^{i-1}\underline{s}]$ be a matrix whose columns are the search steps ${}^{j}\underline{s}$, and let ${}^{i}Y = [{}^{o}\underline{y}, {}^{1}\underline{y}, \ldots, {}^{i-1}\underline{y}]$ be a matrix whose columns are the gradient differences ${}^{j}\underline{y} = {}^{j+1}\underline{g} - {}^{j}\underline{g}$. Then if ${}^{o}\underline{s}, {}^{1}\underline{s}, \ldots, {}^{i-1}\underline{s}$ are independent, the next search step ${}^{i}\underline{s}$ will be A-conjugate to the ${}^{j}\underline{s}$ $(0 < j \leq i-1)$ if

$${}^{i}\underline{d} = {}^{i}H^{t}\ {}^{i}\underline{g} \tag{2.18}$$

where ${}^{i}H$ is chosen to satisfy

$${}^{i}H\ {}^{i}Y = {}^{i}S \tag{2.19}$$

Equation (2.19) has the following general solution for an arbitrary n x n matrix Z.

$${}^{i}H = {}^{i}S^{i}Y^{+} + Z(I-{}^{i}Y^{i}Y^{+}) \tag{2.20}$$

where ${}^{i}Y^{+}$ is the generalized inverse of ${}^{i}Y$. Different choices of Z in (2.20) yield didferent solutions for ${}^{i}H$ corresponding to different methods of choosing the A-conjugate directions ${}^{i}\underline{d}$. Pearson derives four algorithms in this way, three of which lead to readily computable formulas given below. Each algorithm proceeds from an initial point ${}^{o}\underline{x}$ according to steps 1-3 above with the proper formula for ${}^{i}H$ inserted for Eq. (2.17).

<u>Algorithm 2</u>

$$ {}^{i+1}H = {}^{i}H + \frac{({}^{i}\underline{s} - {}^{i}H\,{}^{i}\underline{y})\,{}^{i}\underline{s}^{t}}{{}^{i}\underline{s}^{t}\,{}^{i}\underline{y}} \qquad (2.21) $$

<u>Algorithm 3</u>

$$ {}^{i+1}H = {}^{i}H + \frac{({}^{i}\underline{s} - {}^{i}H\,{}^{i}\underline{y})\,({}^{i}H^{t}\,{}^{i}\underline{y})^{t}}{{}^{i}\underline{y}^{t}\,{}^{i}H\,{}^{i}\underline{y}} \qquad (2.22) $$

<u>Algorithm 4</u> Fletcher-Powell-Davidon (F-P-D)

$$ {}^{i+1}H = {}^{i}H + {}^{i}A + {}^{i}B \qquad (2.23) $$

where

$$ {}^{i}A = \frac{{}^{i}\underline{s}\,{}^{i}\underline{s}^{t}}{{}^{i}\underline{s}^{t}\,{}^{i}\underline{y}} $$

$$ {}^{i}B = \frac{-{}^{i}H\,{}^{i}\underline{y}\,{}^{i}\underline{y}^{t}\,{}^{i}H}{{}^{i}\underline{y}^{t}\,{}^{i}H\,{}^{i}\underline{y}} $$

For a quadratic $F(\underline{x})$ each algorithm converges in n steps or less to the optimum point $\underline{x}^*$, and this convergence is stable in the sense that $F({}^{i+1}\underline{x}) \leq F({}^{i}\underline{x})$. At $\underline{x}^*$, ${}^{n}H = A^{-1}$, the inverse (Hessian) matrix of the second partial derivatives of $F(\underline{x})$. This information can be helpful in practical design problems, since it indicates the sensitivity of $F(\underline{x})$ to small deviations of $\underline{x}$ from $\underline{x}^*$. Note that for each iteration the major computational effort consists of:

evaluating the gradient of $F(^1\underline{x})$, performing a linear search for the optimum scale factor $^1\alpha$, and updating ^1H. Algorithm 4 is the modification by Fletcher and Powell (1963) of Davidon's (1959) variable metric algorithm.

Fletcher and Reeves (1964) have investigated the conjugate gradient algorithm, which is a modification of a technique due to Hestenes and Stiefel (1952) for iteratively solving a set of n linear equations in n variables u_i.

$$B\,\underline{u} = \underline{k} \tag{2.24}$$

An excellent description of the conjugate-gradient method for solving Eq. (2.24) is given by Beckman (1960). The application to the problem of minimizing a quadratic function (2.5) is made clear by writing the condition for $\underline{x}$ to be the optimum point.

$$g(\underline{x}) = A\underline{x} + \underline{b} = 0$$

$$A\underline{x} = -\underline{b} \tag{2.25}$$

Thus, the problem of minimizing $F(\underline{x})$ is equivalent to solving the set of linear equations (2.25) when A and $\underline{b}$ are not known explicitly.

The Fletcher-Reeves (F-R) algorithm proceeds as follows. Choose a starting point $^o\underline{x}$ and initially let $^o\underline{d} = -^o\underline{g}$. Set i = 0.

1. Locate the next point by minimizing $F(^i\underline{x}+^i\alpha\,^i\underline{d})$ with respect to $^i\alpha$.

$$^{i+1}\underline{x} = {}^{i}\underline{x} + {}^{i}\alpha\ {}^{i}\underline{d}$$

$$= {}^{i}\underline{x} + {}^{i}\underline{s}$$

2. Compute the next search direction $^{i+1}\underline{d}$ by

$$^{i+1}\underline{d} = -{}^{i+1}\underline{g} + {}^{i}\beta\,{}^{i}\underline{d}$$

where

$$^{i}\beta = \frac{({}^{i+1}\underline{g})^2}{({}^{i}\underline{g})^2}$$

and return to step 1 with i+1 replacing i.

In the original method for linear equation solving, $^{i}\alpha$ and $^{i}\underline{g}$ are computed directly from B and $\underline{k}$, while here $^{i}\underline{g}$ must be evaluated by computing the partials of $F(\underline{x})$, and $^{i}\alpha$ is determined by the linear minimization of step 1. Convergence is stable, and for quadratic functions, the optimum point is obtained in at most n interations. Unlike in the Fletcher-Powell-Davidon method, A^{-1} is not explicitly available at the end of the search, but the computational effort for this algorithm is less.

Results of applying the algorithms to test functions have been published by Box (1966), Fletcher and Powell (1963), Fletcher and Reeves (1964), and Pearson (1968). Pearson found that in using these algorithms to minimize functions where $\underline{x}^*$ is located on the boundary of a constraint, convergence was improved by setting ${}^{1}H = {}^{o}H$ every n + 1 steps for Algorithms 1-4 and resetting ${}^{1}\underline{d} = -{}^{1}\underline{g}$ in the Fletcher-Reeves algorithm. (The constrained minimizations were performed using the created response-surface technique discussed in Section 2.8.) Acceleration of convergence by resetting ${}^{1}H$ in irregular parameter landscapes has also been reported by Huelsman (1968).

A conjugate-direction method for minimizing a function without calculating the gradient has been invented by Powell (1964). Beginning with an initial point ${}^{o}\underline{x}$ and n linearly independent directions ${}^{1}\underline{d}$, ${}^{2}\underline{d}$, ..., ${}^{n}\underline{d}$, his basic procedure minimizes $F(\underline{x})$ sequentially along ${}^{1}\underline{d}$, ${}^{2}\underline{d}$, ..., ${}^{n}\underline{d}$. Let ${}^{n}\underline{x}$ be the point determined by the last one-dimensional minimization. Then ${}^{n}\underline{x} - {}^{o}\underline{x}$ is taken as the direction for another one-dimensional minimization. For the next iteration of the procedure, ${}^{i}\underline{d}$ is replaced by ${}^{i+1}\underline{d}$ for i = 1, 2, ..., n-1, and ${}^{n}\underline{d}$ is replaced by $({}^{n}\underline{x} - {}^{o}\underline{x})$. Thus, at each iteration a new search direction is defined, and Powell proves that for a quadratic $F(\underline{x})$ these directions are conjugate. Thus, the minimum is located in n iterations. A difficulty with this method arises because, in discarding the old ${}^{1}\underline{d}$ at each iteration, the algorithm may be left with a new set of directions which does not span the parameter space. Powell's modification to eliminate this problem results in an algorithm requiring more than n iterations to minimize a quadratic.

Zangwill (1967) considered this same problem and proposed his own modification of Powell's basic procedure. His algorithm is shown to converge for the case of $F(\underline{x})$ strictly convex and to converge in n or less iterations (or in $2n^2$ or less one-dimensional minimizations) for a quadratic $F(\underline{x})$.

Powell's method has been applied to several test functions with good results (Box, 1966; Fletcher, 1965; Powell, 1964). Similar data for Zangwill's algorithm are not available, although the author has used it successfully in minimizing Rosenbrock's function (Section 3.1).

2.4 Quadratic Fit Methods

It is again assumed that the function to be minimized can be represented adequately by a quadratic (Eq. [2.5]) in the neighborhood of the optimum. Using Eq. (2.6) for the gradient of $F(\underline{x})$, we can solve for the parameter change $\Delta\underline{x} = \underline{x}^* - \underline{x}$ which yields $\underline{g}(\underline{x}) = \underline{0}$.

$$\Delta\underline{x} = -A^{-1}\underline{g}(\underline{x}) \tag{2.26}$$

Newton's method (Bekey and McGhee, 1964, McGhee, 1967) consists of evaluating $\underline{g}$ and A and computing the optimizing steps by Eq. (2.26). For problems which can be expressed in the framework of a least-squares regression, the Gauss-Newton method approximates A by a regression matrix, which requires only first-derivative information (McGhee, 1967).

Note that for either method considerable computational effort is required at each step--evaluating $\underline{g}$ and A (or its approximation) and inverting A. Furthermore, if $\underline{g}$ and A are calculated from perturbations, care must be taken in selecting the step size (Section 2.9). Although convergence may be very rapid with either method, a poor starting point may result in divergence. This disadvantage makes methods of this type more desirable when incorporated in a strategy including a more stable search method. The following technique may be more suited to this type of strategy.

Rather than evaluating $\underline{g}$ and A directly, we may fit a second-order regression surface to a set of N observations of $F(\underline{x})$. The regression surface is defined by

$$\Psi(\underline{x}) = \frac{1}{2}\, \underline{x}^t \Gamma \underline{x} + \underline{\beta}^t \underline{x} + \alpha \tag{2.27}$$

Performing the minimization

$$\min_{\gamma_{j,k},\beta_j,\alpha} \sum_{i=1}^{N} \left[F({}^i\underline{x}) - \Psi({}^i\underline{x}) \right]^2 \tag{2.28}$$

results in $(n^2 + 3n + 2)/2$ equations which determine Γ, $\underline{\beta}$, and α. The estimate for the optimum point is obtained by solving

$$\underline{\nabla}\Psi(\underline{x}) = \Gamma \underline{x} + \underline{\beta} = 0 \tag{2.29}$$

Since the N observations may be taken at any values of $\underline{x}$ (although they must be sufficient to define Γ, $\underline{\beta}$, and α), this technique could be combined with another climbing method, for example, a pattern search (Section 2.5) or a creeping random search (Section 2.6), i.e., observations made during the climbing method are also stored for use in Eq. (2.28).

2.5 Direct Search Methods

Gradient descent methods and the conjugate direction methods utilizing the gradient expend a large amount of effort in obtaining information (the gradient) at a single point; this information is extrapolated to search for a better point. Noting this considerable effort at one point and the inefficiency of steepest descent techniques on many problems, Hooke and Jeeves (1961) proposed making exploratory moves and always moving the base of the search when an improvement was found. Algorithms of this type have become known as direct search methods.

Hooke and Jeeves' pattern search is a direct method designed to follow a descent path to the optimum by searching in previously successful directions (pattern moves). (Explicit instructions for the algorithm are given by Wilde and Beightler [1967].) Following each pattern move, exploratory moves are made with each coordinate separately to detect changes of direction of the descent path. The programmer sets the exploratory move step length (which may be reduced later by the algorithm); the lengths of pattern moves are determined by exploratory step lengths and previous pattern-move lengths. Thus, there is no effort expended in minimizing along a search direction. The search is ended when successive failures lead to a reduction in the exploratory step length below a preset minimum. Note that the progress of a pattern search depends only on whether each function measurement is greater than or less than some previous observation, the magnitude of differences in function values are ignored. The fact that convergence does not depend on accurate measurements of function differences (as in the case of algorithms requiring gradients or linear minimizations) may be an advantage in problems with noisy observations of the criterion function. (The problem of optimizing in the presence of noise is discussed in Section 2.7.)

Rosenbrock's method of rotating coordinates (Rosenbrock, 1960, Wilde, 1964) and its alteration by Swann (Swann, 1964, Fletcher, 1965) are also designed to recognize a direction of descent and to search along it. However, the fixed-length steps of pattern search are replaced by successive linear minimizations in n orthogonal directions. The net progress in parameter space resulting from n such minimizations establishes a new search direction, which is analogous to a "pattern" direction. The remaining n-1 directions for the next series of minimizations are made orthogonal to the newly established one.

A unique approach to optimization was borrowed from the <u>sequential simplex</u> or <u>simplicial</u> method of Spendly, Hext, and Himsworth (1962) for locating a nearby optimum point and following it in the presence of noise. The method is begun by placing n+1 measurements at the vertices of an n-dimensional simplex (Fig. 2.5). The point on the simplex with the largest function value is determined, and a new point is located by reflecting this "worst" point through the center of the simplex. Thus, a new simplex is created, consisting of the old one, but with the new point replacing the previous worst one. This movement of the simplex tends to track the optimum point. In order to speed the progress of the search from a starting point far from the optimum, Nelder and Mead (1965) modified the original method to allow for expansion and contraction of the simplex. With this provision it was found that the initial size of the simplex did not greatly affect the speed of convergence. Since the movement of the search depends only on finding the worst point of the simplex, the method is not disturbed by small observation errors. Spendly, Hext, and Himsworth noted that the rate of advance was inversely proportional to the standard deviation of Gaussian measurement noise--an indication that averaging observations at a point would not be beneficial, since the standard deviation

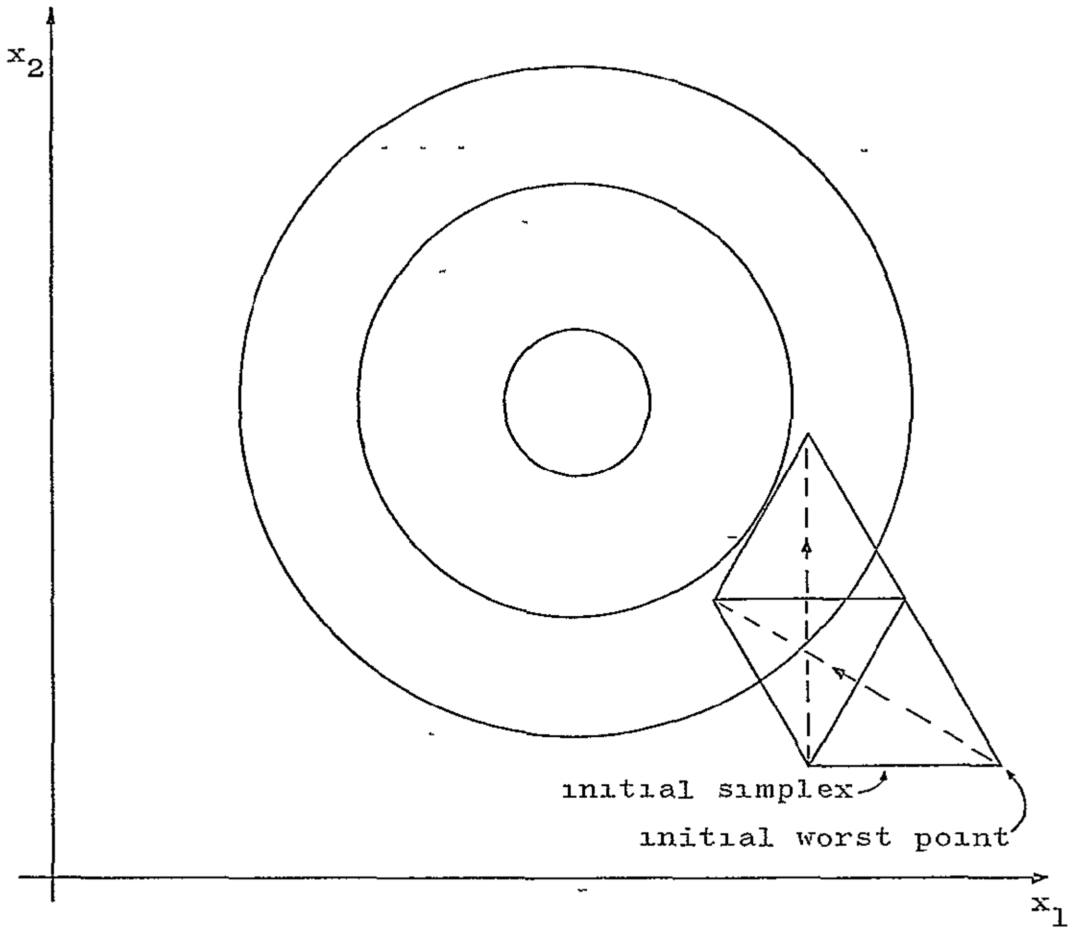


Fig. 2.5 Operation of the simplex method in two dimensions.

is reduced in proportion to the square root of the number of observations.

Data on the performance of the simplicial and rotating coordinates algorithms have been published by Fletcher (1965) and Box (1966). Similar data for pattern search are not known to the author, although it has been applied successfully to network-design optimization by Huelsman (1968). Wilde and Beightler (1967) report that for pattern search the number of function evaluations for optimization tends to be only a linear function of the number of parameters, n, rather than a quadratic or cubic function as for most other methods (another exception is the creeping-random search of Section 2.6).

The direct search methods are designed to find the best search directions and to proceed in these directions without wasting time evaluating derivatives. This tends to make their performance favorable in the early stages of the search. However, in the neighborhood of the optimum the derivative information acquired by the quadratically-convergent conjugate-direction algorithms accelerates their progress. This behavior was noticed by Fletcher in comparing the performance of Swann's version of the rotating coordinates method with the conjugate direction method of Powell (1964). The results of Box indicate that the simplicial and rotating coordinate methods become ineffective compared to the conjugate direction algorithms as n increases beyond 5.

2.6 Random Search Methods

The development of random search optimization was motivated mainly by the need for methods which were simple to program and effective in irregular parameter landscapes. Before the availability of true analog-digital hybrid computers simple random search algorithms could be implemented by hard-wired optimizers attached to analog machines. Random search methods are still especially attractive for hybrid computers consisting of high-speed repetitive analog machines capable of evaluating the criterion function quickly and small digital computers without the floating-point hardware necessary to make complicated algorithms fast enough to be advantageous. Furthermore, the complex, nonlinear dynamic systems which are most advantageously simulated on analog machines often have parameter landscapes with the sharp ridges, discontinuous first derivatives, etc., which can cause deterministic algorithms to fail. There is also evidence to suggest that random methods are superior in optimizing smooth functions of many parameters (Schumer and Steiglitz, 1968).

The literature reviewed here has been loosely grouped into the categories of theoretical developments and specific algorithms with applications.

2.6.1 Theoretical Developments

Brooks (1958) suggested choosing observation points from a uniform distribution over the entire parameter space. After N such points have been tested, the one with the smallest criterion function value is taken as the best approximation to the optimum. To evaluate the effectiveness of this method, let the parameter space be an n-dimensional hypercube with sides of unit length, and imagine the optimum point to be enclosed by a smaller hypercube with sides of length δ and volume $v = \delta^n$. We would like to ensure that the search will place at least one point in the smaller hypercube with a specified probability. Brooks showed that the number of trials necessary to have probability p of casting at least one point into the smaller hypercube is

$$N = \frac{\log(1-p)}{\log(1-v)} \tag{2.30}$$

Taking v to be constant in Eq. (2.30), it was concluded that the number of trials required for random search does not depend on the number of parameters. However, as pointed out by Hooke and Jeeves (1958) and Spang (1962), for v to remain constant, δ must increase exponentially, so that for a fixed number of trials the uncertainty in the parameter values, δ, increases exponentially with n. Spang showed that substitution of δ^n for v in Eq. (2.30) yields

$$N \approx -\log(1-p)/\delta^n$$
$$\approx 2.3/\delta^n \qquad (2.31)$$

for p = .9, whereas the number of points required for a deterministic grid test (points located equal distances δ apart) is $1/\delta^n$. Such a large number of trials obviates the use of either method as a means to locate the optimum accurately. But in the absence of any information regarding the location of the optimum, a grid search might be used to choose a starting point for some sequential search algorithm.

Rastrigin (1963) has studied the convergence properties of a fixed step-size, creeping random search algorithm (FSSRS). Beginning from a point ${}^i\underline{x}$, exploratory steps $\Delta\underline{x}$ are made with fixed length and random direction. When a point is found such that $F({}^i\underline{x} + \Delta\underline{x}) < F({}^i\underline{x})$, the corresponding increment is labeled ${}^{i+1}\Delta\underline{x}$ and the search is moved to the new base point

$${}^{i+1}\underline{x} = {}^i\underline{x} + {}^{i+1}\Delta\underline{x} \qquad (2.32)$$

(With this notation, i indexes only successful trials.) The algorithm was compared to a steepest descent method in which at each iteration a step of the same magnitude was made in the direction of the gradient at ${}^i\underline{x}$. Rastrigin introduced the concept of <u>search loss</u>, defined as the number of criterion function evaluations required for a displacement in the negative-gradient direction equal to the step length $\Delta\underline{x}$, or equivalently, the reciprocal of the average displacement in the negative-gradient direction per function evaluation. The search loss was computed for both algorithms applied to a linear test function and a distance function $F(\underline{x}) = (\sum_{i=1}^{n} x_i^2)^{1/2}$. For both functions it was found that as the number of parameters increased, the creeping random algorithm was superior to the steepest descent method on the basis of search loss. The limitations of this comparison might be noted here. The steepest descent algorithm is made very inefficient by requiring a gradient evaluation (n+1 function evaluations) at each iteration and allowing only constant step sizes. A more practical steepest descent program could make more efficient use of the gradient information (for example, the optimum gradient method of Section 2.2). Thus, in practice the relative advantage of the creeping random strategy might not be as great.

The convergence of the creeping random method in the presence of noise has been studied by Gurin and Rastrigin (1965). For a linear criterion function, measurements were corrupted by Gaussian noise with zero mean and variance σ^2. The random search algorithm used a "testing step" of fixed length α and random direction. When such a testing step resulted in an improvement in the measured value of $F(\underline{x})$, a step of length $\Delta\underline{x}$ was taken in the same direction. The progress of this algorithm was compared to that of a steepest descent method, which used $\underline{2n}$ perturbations of length α to determine the gradient and then took a working step of length $\Delta\underline{x}$ in the estimated negative-gradient direction. Comparisons were made on the basis of search loss, and as a function of the number of parameters n and a signal-to-noise ratio

$$\delta = \frac{|\underline{\nabla F}|\alpha}{\sigma\sqrt{2}} .$$

For any fixed value of δ search loss is a linear function of n for the random method. For $\delta = \infty$ (no noise) the gradient method has a search loss linear in n, but for $\delta = 1$ the search loss is greater than $c\, n\sqrt{n-1}$, where c is a constant. For $\delta = 1$ and $\delta = \infty$ the random search method was superior for $n \geq 6$. For $n = 6$ the increase of noise level from $\delta = \infty$ to $\delta = 1$ caused the search loss for both methods to increase from 12 to approximately 32 (function evaluations necessary for a net progress of $\Delta\underline{x}$ in the negative-gradient direction). Brooks and Mickey (1961) have studied the fixed step-size steepest-descent algorithm for a linear criterion function with Gaussian noise. Their results indicate that in order to minimize search loss, a minimum number of function evaluations should be expended on estimating the gradient. Thus, had Gurin and Rastrigin used n+1 steps (rather than 2n) to estimate $\underline{\nabla F}$, the relative advantage of the creeping random method over steepest descent might have been diminished.

Beginning with Rastrigin's fixed step-size random search (Eq. [2.32]), Schumer and Steiglitz (1968) developed an algorithm with adaptive step size. For the criterion function $F(\underline{x}) = \sum_{i=1}^{n} x_i^2 = \rho^2$, the expected improvement per step, normalized by the present value of F, was computed as a function of n and $\eta = s/\rho$, the ratio of the step size to the distance to the optimum, i.e.,

$$I(n,\eta) = \frac{-E\{\Delta F\}}{F} \tag{2.33}$$

$I(n,\eta)$ was maximized with respect to η, and the optimum $I(n)$ was evaluated for large n. This led to the result that the average number of function evaluations necessary to minimize F within a fixed accuracy is asymptotically linear in n. A practical algorithm, which attempts to adjust the step size to the optimum during the minimization process, was developed and compared to two deterministic algorithms. These were the simplicial method of Nelder and Mead (1965) and a second-order method which evaluates first and second partial derivatives at each iteration. Performances were compared on the basis of the average number of function evaluations required for minimization. For a quadratic function, the second-order method was superior for $n \leq 78$, but for the function $F(\underline{x}) = \sum_{i=1}^{n} x_i^4$ the adaptive random search algorithm was superior to the second order method for $n > 2$ and superior to the simplicial method for $n > 10$. The adaptive search was also tested for $F = \sum_{i=1}^{n} a_i x_i^2$ where the a_i were chosen from a probability distribution uniform on [.1,1.]. <u>For each of these three test functions the number of function evaluations required by the adaptive random search method was proportional to n</u>. This compares with results reported for pattern search (Section 2.5). For other methods, function evaluations are usually proportional to the second or third power of n.

Adaptation of a creeping random search with respect to search <u>direction</u> has been discussed at length by Rastrigin (1967). He has proposed several learning algorithms which adjust ${}^k p_i$, the probability of selecting a positive increment for the ith parameter at the kth step, as a function of past performance. Adjustment is accomplished by making ${}^k p_i = {}^k p_i ({}^k w_i)$, a monotonic, non-decreasing function of the <u>memory parameter</u> ${}^k w_i$. One example of Rastrigin's schemes for adjusting ${}^k w_i$ is the following algorithm.

$$ {}^{k+1} w_i = {}^k w_i - \delta \, {}^k \Delta x_i \, {}^k \Delta F \qquad (2.34)$$

where

$$ {}^k \Delta x_i = {}^k x_i - {}^{k-1} x_i $$

$$^{k}\Delta F = F(^{k}\underline{x}) - F(^{k-1}\underline{x})$$

and

$$c_1 \leq w_i \leq c_2.$$

The adjustment of $^{k}w_i$ is proportional to the magnitude of $^{k}\Delta F$, the step size causing $^{k}\Delta F$ and a positive coefficient, δ. For example, a positive $^{k}\Delta x_i$ causing an improvement ($^{k}\Delta F < 0$) brings about an increase in $^{k}w_i$ and thereby an increase in $^{k+1}p_i$, the probability of increasing x_i at the next step. Rastrigin introduces other algorithms similar to Eq. (2.34), which allow for a discarding information collected in the distant past ("forgetting") and which provide for better adaptation to the best of possible successful directions.

An interesting aspect of Rastrigin's work is his idea of separating the search algorithm from the learning algorithm. The learning algorithm (Eq. [2.34]) collects information on past performance and adjusts the directions for future exploratory steps. It is the function of the search algorithm to decide whether or not to actually move the center of the search as a result of an exploratory step. One possibility is to move only when such a step results in a reduction of F, e.g., Eq. (2.32). Rastrigin also suggests the possibility of moving the search with every exploratory step. This places the learning algorithm in complete control of the search. Such a policy might be beneficial in stepping over local minima or local flat regions and in problems with observation error.

2.6.2 Specific Algorithms and Applications

Experiments with creeping random search strategies on analog computers were reported as early as 1958-59. Favreau and Franks (1958) described a creeping random method for optimizing dynamic systems, and Munson and Rubin (1959) optimized a system of nonlinear algebraic equations by a continuous creeping random perturbation of parameters. A hard-wired creeping random optimizer, including provisions for expanding and reducing step size and correlating future trial-step directions with past successful directions, was built by Mitchell (1964) for use with a fast repetitive hybrid computer. This was employed by Maybach (1966a) to solve minimum-time bang-bang optimal control problems.

The availability of true analog-digital hybrid computers has made it possible to employ more sophisticated random search strategies than could be implemented by hardware optimizers attached to analog machines. Here we shall discuss alterations to the basic creeping random search which were introduced and applied chiefly by Bekey et al. (1966) and by Stewart, Kavanaugh, and Brocker (1967).

One modification concerns the classification of a trial step as a success or failure. Let $F({}^1\underline{x})$ be the current value of the criterion function and $F({}^1\underline{x} + \Delta\underline{x})$ the value at a trial step. Stewart et al. use a threshold strategy to define a success for a minimization problem:

$$F({}^1\underline{x} + \Delta\underline{x}) - F({}^1\underline{x}) \leq \eta\, F({}^1\underline{x}) \quad (\eta > 0) \tag{2.35}$$

In the beginning of the search, when $F({}^1\underline{x})$ is large, a relatively large improvement is required for a success, while near the end of the search smaller improvements are required. Stewart et al. found that the average number of steps required for solution could be reduced by approximately one-third for $\eta = .3$ and $\eta = .7$ as opposed to $\eta = 0$, while $\eta = 1$ resulted in a sharp increase in required steps. Another possibility is a constant threshold level:

$$F({}^1\underline{x} + \Delta\underline{x}) - F({}^1\underline{x}) \leq \varepsilon \tag{2.35}$$

For example, ε might be taken just large enough to overcome errors in measuring $F(\underline{x})$.

A vector-valued criterion function was employed by Stewart et al. in a creeping random algorithm to solve the two-point boundary value problem resulting from a Maximum-Principle optimization of an orbit transfer problem. Boundary conditions were to be matched for state variables representing displacement and velocity, $\underline{x}_d$ and $\underline{x}_v$, and adjoint variables, $\underline{p}$. The criterion function was defined as

$$\underline{F} = (F_d, F_v, F_p),$$

where each component of $\underline{F}$ is the sum of the errors in matching the boundary conditions for one class of variables. For a trial to be regarded as a success, it was required that all three components of $\underline{F}$ be reduced (the threshold strategy [Eq. (2.35)] was applied to each component). This more restrictive success criterion might be useful in avoiding a local minimum where only one or two components of F are small. Gonzalez (1969) employed a vector-valued function in a Maximum-Principle optimization of the same systems solved by Maybach (1966a). The number of evaluations required for convergence was reduced on the average, the most striking reductions being obtained for difficult starting points in the parameter space.

A modification for directional adaptation is the introduction of absolute positive and negative biasing (Bekey et al., 1966) into the basic creeping random algorithm, which is repeated here.

$$^{i+1}\underline{x} = {}^{i}\underline{x} + {}^{i+1}\Delta\underline{x} \tag{2.37}$$

If the last increment resulted in a success, it is used again for the next trial step, i.e., $\Delta\underline{x} = {}^{i}\Delta\underline{x}$ (positive biasing). If the last increment $\Delta\underline{x}$ resulted in a failure,

$-\Delta \underline{x}$ is used for the next trial step (negative biasing). Of course, negative biasing is not used following two successive failures, or the algorithm will loop endlessly. Also, it is wasteful to use it after the first failure following a success. Bekey et al. reported that absolute biasing was effective in improving convergence. Stewart et al. used only positive biasing and found that it decreased the average number of steps required by approximately 40% compared to the search without biasing.

Another element of randomness may be introduced by using a random increment for each variable, rather than an increment of fixed length and random sign only. This results in a step $\Delta \underline{x}$ which is random in length and direction, and all directions are possible. For the algorithm with only random sign for each variable, only 2^n discrete directions are possible. The disadvantage of this can be seen in Fig. 2.6, where a zig-zag path must be followed from the initial point ${}^{o}\underline{x}$ to the optimum. Bekey et al. chose the increments Δx_i as independent Gaussian random variables with zero mean. Gonzalez (1969) chose the increments from a uniform distribution, which is usually easier to generate on a digital computer.

If random increments Δx_j are used, the average step size can be adjusted by changing the variance of the distribution of the increments. If the step size is small, a large proportion (asymptotic to 50%) of the trials result

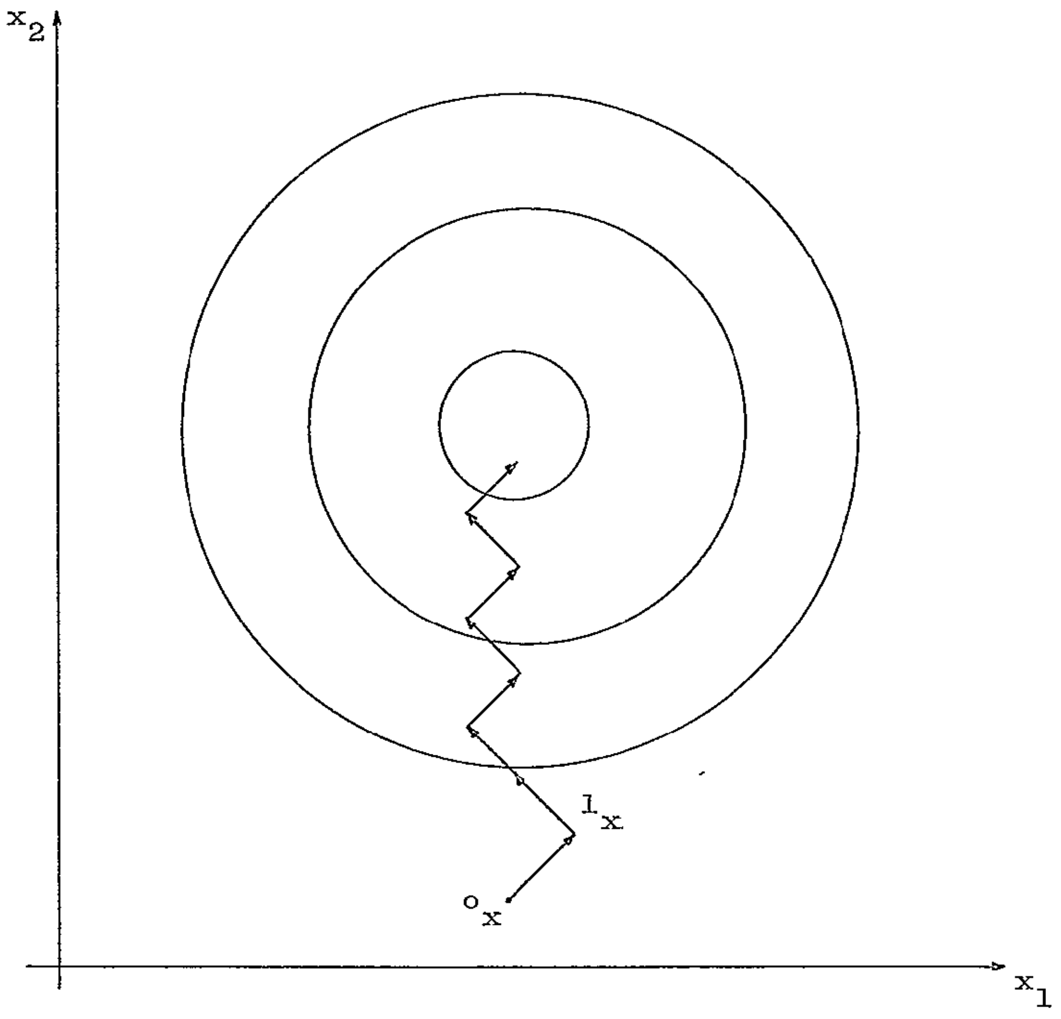


Fig. 2.6 The behavior of a "discrete-direction" random search algorithm.

in success (assuming no threshold strategy), but the average improvement per step is small. On the other hand, a large step size results in a small ratio of improvements to trial steps. Karnopp (1963) suggests increasing the variance if an improvement occurs within two trials and decreasing the variance if no improvement occurs within three trials. Stewart et al. provide for variance reduction by some factor after a number of consecutive failures. Bekey et al. used a constant variance of 4% of the range of each parameter during the entire local search. It was reported that their work and the results of a further study (Adams and Lew, 1966) failed to find a variance adjustment strategy yielding faster convergence than the constant variance method. This result is especially interesting when contrasted with the work of Schumer and Steiglitz (1968) on an adaptive step-size algorithm. It should be noted that the adaptive algorithm was developed for the criterion function $F = \sum_{i=1}^{n} x_i^2$ and was tested on other smooth functions, whereas the results of Bekey et al. are based on a nonlinear dynamic system with minimum-time and minimum-fuel criteria, which could lead to an irregular parameter landscape.

An algorithm for directional adaptation of the creeping random search has been proposed by Matyas (1965). From the point ${}^{i}\underline{x}$ a trial step ${}^{i+1}\Delta\underline{x}$ is taken.

$$^{i+1}\underline{x} = {}^{i}\underline{x} + {}^{i+1}\Delta\underline{x}$$

If $F({}^{i+1}\underline{x}) < F({}^{i}\underline{x})$, the center of the search is moved to the point $^{i+1}\underline{x}$. Otherwise the center of the search remains at $^{i}\underline{x}$, and another trial step is taken. The trial steps are given by

$$^{i+1}\Delta\underline{x} = {}^{i+1}\underline{d} + {}^{i+1}T \; {}^{i+1}\underline{\xi}$$

where $^{i+1}\underline{\xi}$ is an n-dimensional Gaussian random vector with zero mean and unit correlation matrix, $^{i+1}\underline{d}$ specifies the mean of $^{i+1}\Delta\underline{x}$, and ^{i+1}T is an n x n matrix. Adaptation is accomplished by adjusting $^{i+1}\underline{d}$ as a function of past trial steps and past successes and failures. Let

$$^{i+1}\underline{d} = c_o \, {}^{i}\underline{d} + c_1 \, {}^{i}\Delta\underline{x},$$

where c_o and c_1 satisfy the following conditions. If the last step $^{i}\Delta\underline{x}$ resulted in an improvement $[F({}^{i}\underline{x}) < F({}^{i-1}\underline{x})]$,

$$0 \leq c_o \leq 1, \; c_1 > 0, \; c_o + c_1 > 1.$$

Otherwise,

$$0 \leq c_o \leq 1, \; c_1 \leq 0, \; \left| c_o + c_1 \right| < 1.$$

Thus, the mean value for the next trial step is weighted positively by the present mean value and weighted positively or negatively by the last trial step. The transformation matrix ^{i+1}T might be used to introduce correlation between the trial step components $^{i+1}x_j$. But for a simple algorithm, Matyas specified ${}^{i+1}T$ by

$$ {}^{i+1}T = {}^{i+1}b \; I $$

where I is the identity matrix. The coefficient ${}^{i+1}b$ may be adjusted to control the variance of the trial steps.

A somewhat different approach to random search has been described by Rastrigin (1967) and is currently being investigated by Heydt (1969). A search is made about an initial point ${}^{o}\underline{x}$ for an improved point ${}^{1}\underline{x}$ $[F({}^{1}\underline{x}) < F({}^{o}\underline{x})]$. The line ${}^{1}\underline{x} - {}^{o}\underline{x}$ is used to determine the axis of symmetry of a hypercone in parameter space with focus at ${}^{o}\underline{x}$ (Fig. 2.7). The hypercone is constructed with angle θ and length h. Observations are made at points uniformly distributed inside the cone. The best of these $({}^{2}\underline{x})$ is selected, and the line ${}^{2}\underline{x} - {}^{1}\underline{x}$ defines the axis of symmetry of the next hypercone. Thus, past successes are used to determine the search direction. If no ${}^{i+1}\underline{x}$ with $F({}^{i+1}\underline{x}) < F({}^{i}\underline{x})$ is found in some number of observations, θ and h are increased to enlarge the search region. This method would seem to be effective in jumping over some local minima. On the other hand a hyperconical search region may make the algorithm inefficient in turning sharp corners, and Heydt has proposed experimenting with hyper-paraboloids and hyper-hyperboloids. His algorithm with the hyperconical search region was successful in optimizing a satellite attitude acquisition problem, which was solved by

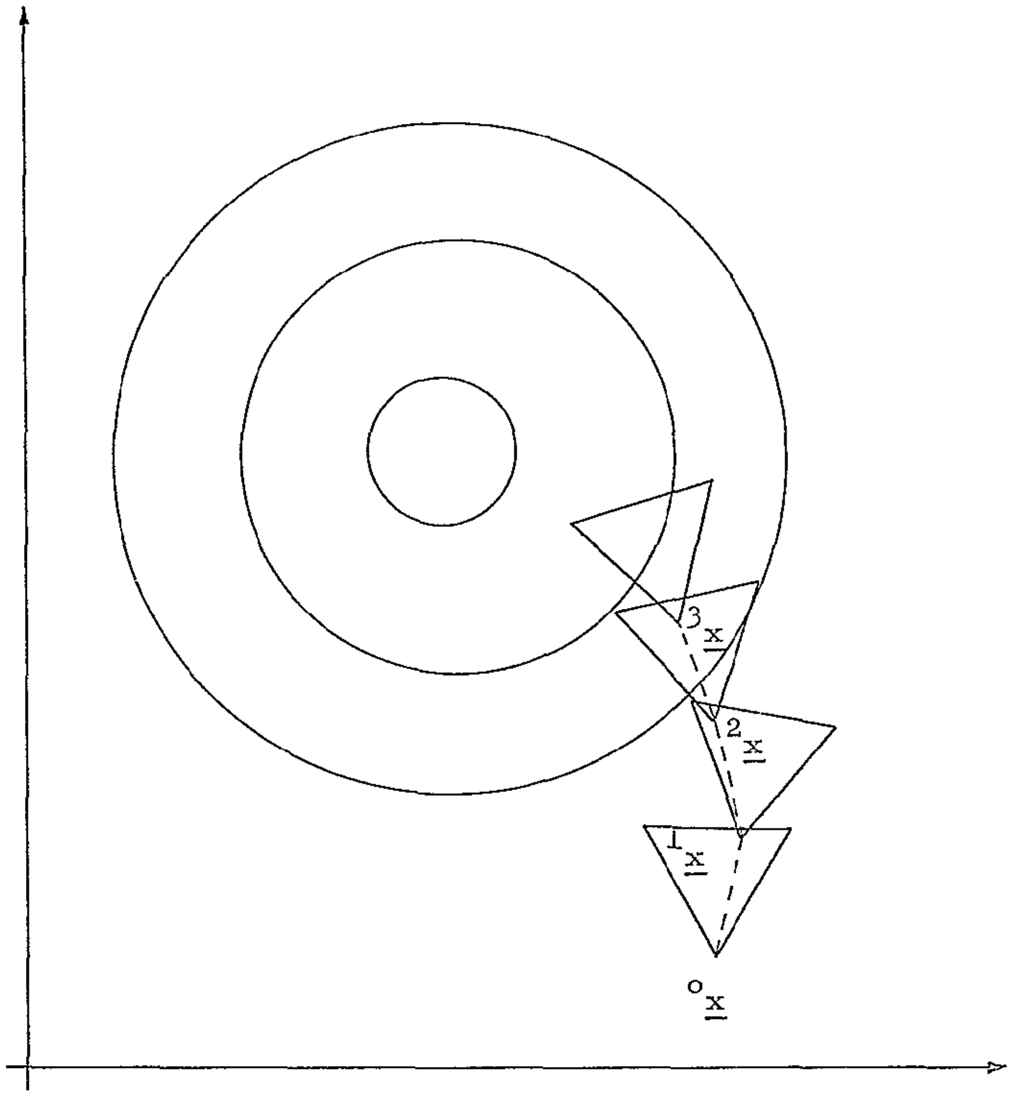


Fig. 2.7 Creeping random search with hyperconical search regions.

Kavanaugh, Stewart, and Brocker (1968) with the creeping random algorithm described by Stewart et al. (1967).

2.7 Stochastic Approximation

Most of the optimization techniques discussed in previous sections assume that the criterion function is evaluated without error. If error or "noise" is present, these methods are reduced in efficiency or may fail altogether. Stochastic approximation is a technique for optimization in the presence of noise.

Let us assume that the observations $f(\underline{x})$ of a unimodal criterion function are contaminated by additive noise:

$$f(\underline{x}) = F(\underline{x}) + v, \qquad (2.38)$$

where the random variable v has zero mean and finite variance. A stochastic approximation minimization algorithm (satisfying certain conditions discussed below) will converge to the optimum, $\underline{x}^*$, in mean square and with probability 1 as the number of observations, i, of $f(\underline{x})$ tends to infinity. Since the existing theorems of stochastic approximation guarantee convergence only as $i \to \infty$, it is necessary to refer to specific applications for speed of convergence. Unfortunately, published experimental results obtained with these algorithms are few.

The mathematical requirements which the algorithm must satisfy in order to converge were discovered and developed mainly by Robbins and Munro (1951), Kiefer and Wolfowitz (1952), Blum (1952), and Dvoretsky (1956). Chapter 6 of Wilde (1964) contains a lucid introduction to stochastic approximation; other readable treatments are given by Hampton (1968) and Chang (1961). In this section we shall discuss briefly the algorithm of Kiefer and Wolfowitz, the general theorem given by Dvoretsky, and some practical algorithms with applications.

The Kiefer-Wolfowitz (K-W) algorithm described here is for a function of one variable; the extension to the multidimensional case is straightforward. The technique is similar to a deterministic steepest descent. From a point ${}^{i}x$, the noisy objective function is evaluated at two points ${}^{i}x + {}^{i}c$ and ${}^{i}x - {}^{i}c$ to obtain an estimate of the slope of $F({}^{i}x)$.

$$\frac{f({}^{i}x + {}^{i}c) - f({}^{i}x - {}^{i}c)}{2\,{}^{i}c} \qquad (2.39)$$

Then a working step is taken according to this estimate and a step-size factor, $2\,{}^{i}a$

$${}^{i+1}x = {}^{i}x + \frac{{}^{i}a[f({}^{i}x + {}^{i}c) - f({}^{i}x - {}^{i}c)]}{{}^{i}c} \qquad (2.40)$$

${}^{i}a$ and ${}^{i}c$ are elements of sequences of real numbers which must satisfy the following conditions in order that Eq. (2.40) converges to the minimum of F(x) as $i \to \infty$.

$$\lim_{i\to\infty} {}^{i}a = 0 \tag{2.41a}$$

$$\lim_{i\to\infty} {}^{i}c = 0 \tag{2.41b}$$

$$\sum_{i=1}^{\infty} {}^{i}a = \infty \tag{2.41c}$$

$$\sum_{i=1}^{\infty} \left(\frac{{}^{i}a}{{}^{i}c}\right)^2 < \infty \tag{2.41d}$$

Note that, as with all stochastic approximation schemes, convergence is guaranteed only as i approaches infinity, the movement toward the optimum may be very slow. It may be seen from Eq. (2.40) that if the true differences $[F({}^{i}x + {}^{i}c) - F({}^{i}x - {}^{i}c)]$ are not large compared to the noise variance, many steps will be taken in the wrong direction.

Dvoretsky (1956) has treated stochastic approximation from the point of view of a very general algorithm, which includes those of Robbins and Munro and Kieffer and Wolfowitz as special cases and from which other specific methods have been developed. His basic algorithm is represented as the sum of a deterministic term $T({}^{1}x, {}^{2}x, \ldots, {}^{i}x)$ and a random term ${}^{i}r$, which includes the effects of noise:

$${}^{i+1}\underline{x} = T({}^{1}\underline{x},\ {}^{2}\underline{x},\ \ldots,\ {}^{i}\underline{x}) + {}^{i}r. \qquad (2.42)$$

Equation (2.40) could be expressed in this form by writing $f({}^{i}x) = F({}^{i}x) + {}^{i}v$ and separating out the terms containing ${}^{i}v$. It may be noted that the algorithm allows ${}^{i+1}\underline{x}$ to be a function of all previous $\underline{x}$'s. Although Dvoretsky's theorem is important in the mathematical development of stochastic approximation, it is not stated here, as it provides no specific algorithm for optimization.

The problem of optimization in the presence of noise has been investigated by Kushner (1963), who used the K-W algorithm as a basis for several search procedures. A feature of Kushner's methods is the use of information obtained during the search to estimate the "best" sequences $\{{}^{i}a\}$ and $\{{}^{i}c\}$, whose optimum values depend on the unknown function to be minimized. This information is extracted from the sequence of angles ${}^{i}\phi$ formed by successive steps in the parameter space, as illustrated in Figs. 2.8 and 2.9. In Fig. 2.8 there is a sequence of predominantly large angles, indicating that the ratio of the step size to the distance from the optimum is small. In Fig. 2.9 the angles are small, indicating that the process is overshooting the optimum. This information is used to adapt $\{{}^{i}a\}$ and $\{{}^{i}c\}$ to the local behavior of the objective function.

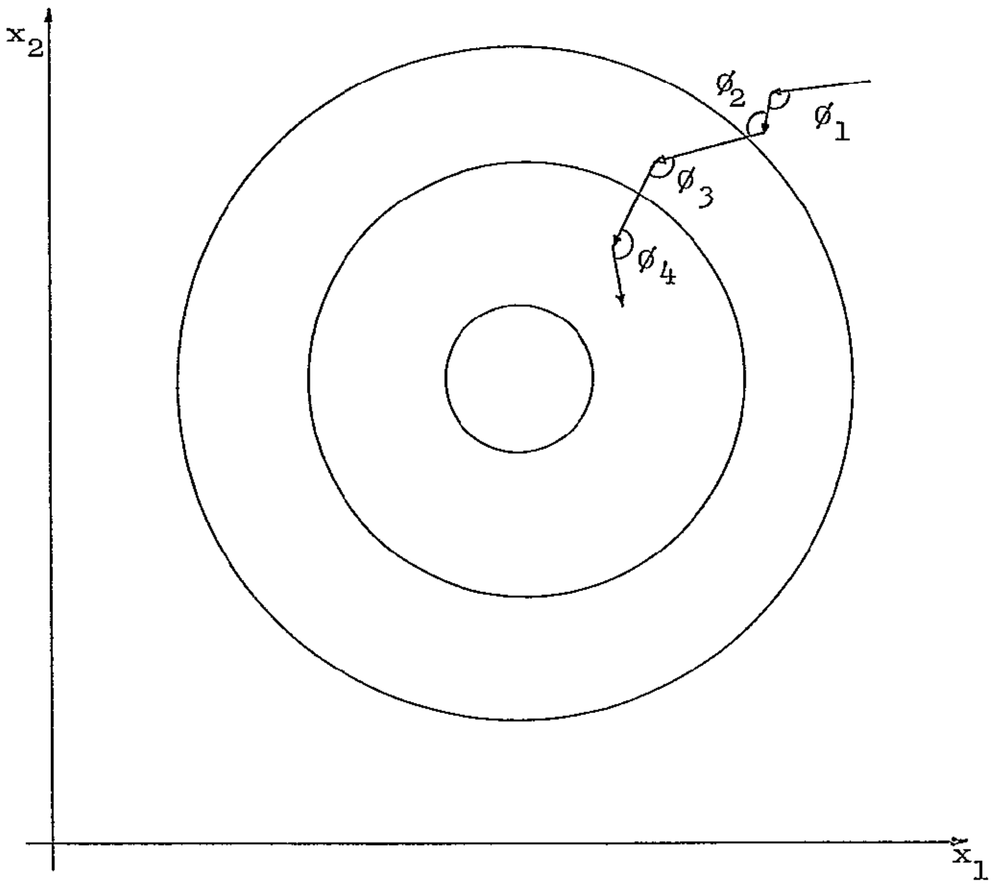


Fig. 2.8 A stochastic approximation algorithm with a small step size.

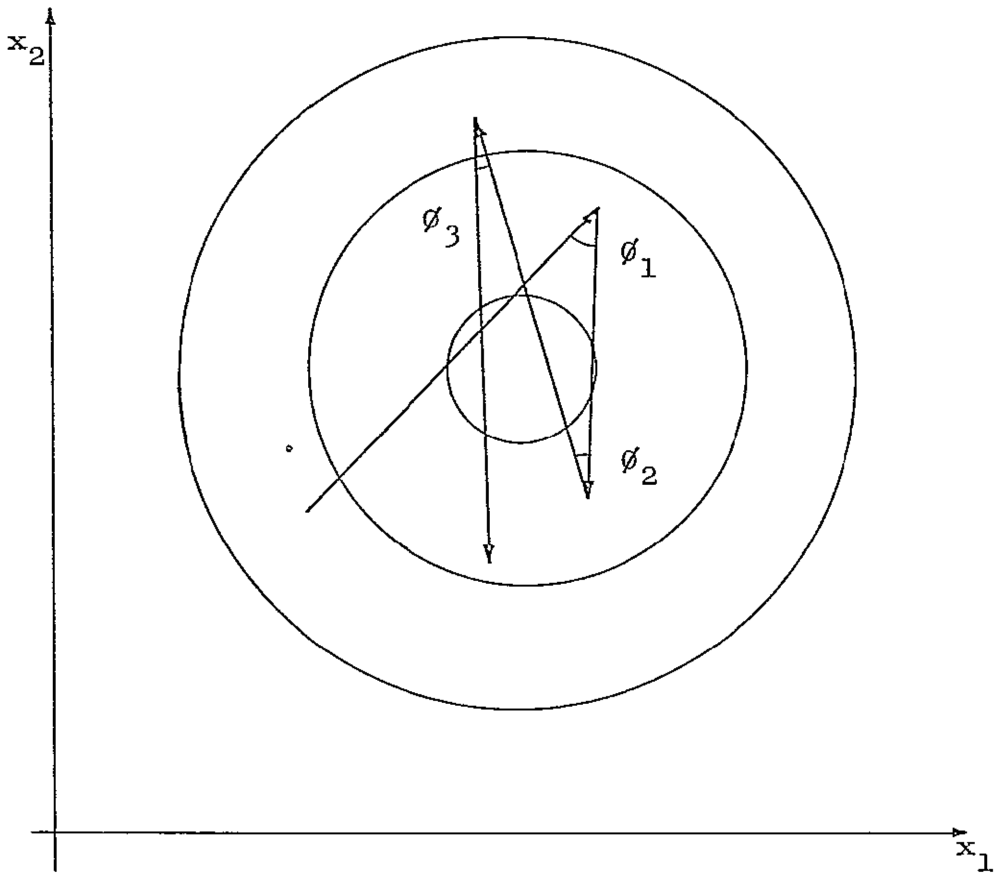


Fig. 2.9 A stochastic approximation algorithm with a large step size.

Five search procedures were investigated, each incorporating the K-W algorithm with adaptive coefficient sequences. The first (a) is the basic n-dimensional K-W algorithm, which estimates the gradient for every working step. The other four procedures sequentially choose single directions in parameter space and apply the one-dimensional K-W algorithm to search for a minimum along these lines. For these four methods the search directions are selected as follows.

(b) the coordinate directions

(c) the estimated gradient direction

(d) a randomly chosen direction

(e) the direction determined by the current point and the point corresponding to the lowest objective function measurement for a number of local, randomly placed observations.

Method (b) was suggested as an improvement over (a), since a pair of sequences $\{{}^1a_j\}$ and $\{{}^1c_j\}$ ($j=1, \ldots, n$) can be assigned to and adapted for each coordinate direction. However, the efficiencies of both (a) and (b) were thought to decrease rapidly as the number of parameters is increased. Methods (c), (d), and (e) are attempts to increase efficiency for problems with many parameters, especially in the initial stages of the search. Methods (c) and (e) were found superior to (d) for quadratic objective functions with additive, uniformly-distributed noise whenever the true function value is large compared to the noise variance. This advantage is greatly reduced close to the optimum, where the signal-to-noise ratio becomes small. Close to the optimum, attempts at consistently choosing profitable search directions are unsuccessful, but the properties of the stochastic approximation algorithm ensure convergence, although it may be very slow.

Janac (1967, 1969) has proposed an algorithm consisting of the basic K-W formula with two modifications:

$$ {}^{i+1}\underline{x} = {}^{i}\underline{x} - \frac{{}^{i}a}{{}^{i}c}\left({}^{i}h - 1\right) \underline{w}({}^{i}\underline{Y}) \qquad (2.43) $$

where.

${}^{i}h$ is an integer equal to the first unsuccessful "working" step in the estimated gradient direction, subject to $({}^{i}h-1) \geq 1$,

$${}^{i}Y_j = f({}^{i}x_1, \ldots, {}^{i}x_j + {}^{i}c, \ldots, {}^{i}x_n) - f({}^{i}\underline{x}),$$
$$(j=1, \ldots, n);$$

$\underline{w}({}^{i}\underline{Y})$ is a vector with the same direction as ${}^{i}\underline{Y}$ and a magnitude function illustrated in Fig. 2.10; and the sequences $\{{}^{i}a\}$ and $\{{}^{i}c\}$ satisfy

$${}^{i}a,\ {}^{i}c > 0$$

$$\lim_{i\to\infty} {}^{i}c = 0$$

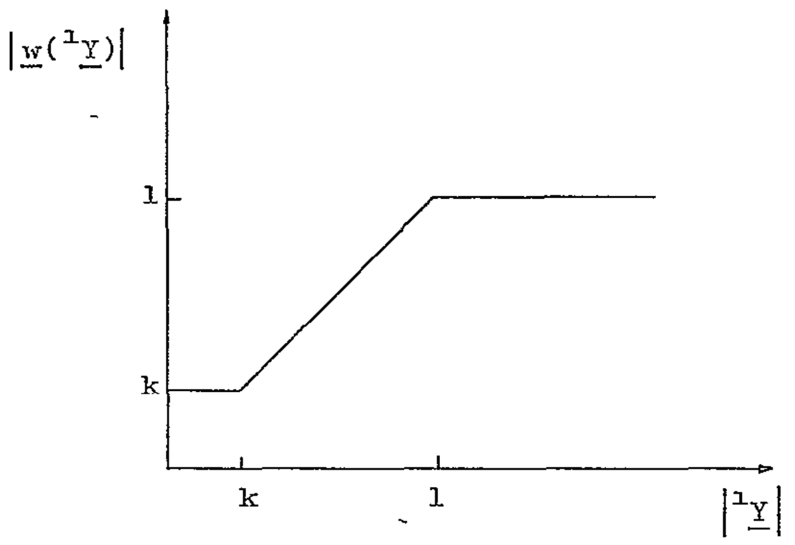


Fig. 2.10 A function for specifying the step size in a stochastic approximation algorithm.

$$\sum_{i=1}^{\infty} {}^{i}a = \infty$$

$$\sum_{i=1}^{\infty} {}^{i}a \, {}^{i}c < \infty$$

$$\sum_{i=1}^{\infty} \left(\frac{{}^{i}a}{{}^{i}c} \right)^2 < \infty$$

In Eq. (2.43) ${}^{i}\underline{x}$ and ${}^{i+1}\underline{x}$ are points at which a new gradient is measured. Following a gradient estimate, steps are taken in the negative-gradient direction until such a step results in an increase of the criterion function measurement. This strategy is designed to make maximum use of each gradient estimate. The nonlinear function w constrains the step size α by $\frac{k\,{}^{i}a}{{}^{i}c} \leq \alpha \leq \frac{{}^{i}a}{{}^{i}c}$. This algorithm was applied to a 4-parameter optimization of the suspension system of a trailer truck riding on a random road surface (Janac, 1969). While the optimization was completed in only 30 working steps (not including function evaluations for gradient estimates), it is impossible to judge the value of the algorithm, because no information is given concerning the variance of the noise.

Stochastic approximation is an attractive approach to the noisy optimization problem, because convergence is guaranteed as $i \to \infty$ under very general conditions. However, it may be that other methods are more effective in reaching a small neighborhood of the optimum in a finite number of steps--a more practical type of convergence to seek.

2.8 Constraints

The optimization techniques which have been discussed here are suitable for unconstrained problems or problems where the optimum is located far enough from the constraint boundaries so that the search procedure does not encounter them. But for many engineering problems the optimum may lie on or close to a constraint boundary. Most of the methods discussed above must be altered to allow for this possibility.

The problem of minimizing $F(\underline{x})$ subject to inequality constraints includes the nonlinear programming problem

Minimize the criterion function

$$F(x_1, x_2, \ldots, x_n) \qquad (2.44)$$

subject to the m inequality constraints

$$\phi_i(\underline{x}) \leq 0 \qquad (i=1, \ldots, m) \qquad (2.45)$$

and

$$x_j \geq 0 \qquad (j=1, \ldots, n) \qquad (2.46)$$

Elegant methods for solving this problem are described by Saaty and Bram (1964) and Wilde and Beightler (1967). Most of these require assumptions such as the convexity of $F(\underline{x})$ and of the $\phi_i(\underline{x})$ and many are designed for a quadratic

$F(\underline{x})$ and/or linear constraints. The methods described here are applicable to less restrictive cases, and do not require the conditions (2.46).

2.8.1 Gradient Projection Method

The gradient projection method (Rosen, 1960, 1961, Saaty and Bram, 1964; Wilde and Beightler, 1967) alter the gradient of $F(\underline{x})$ at constraint boundaries, so that a modified steepest-descent minimization can be employed. The constraints are only required to be convex.

When the search reaches the boundary of a nonlinear constraint, the negative gradient vector is projected onto a plane tangent to the constraint boundary at that point (Fig. 2.11). A move in this negative projected-gradient direction results in an infeasible point which must be moved onto the constraint boundary. For linear constraints the gradient projection is onto the constraint boundary itself, a modified steepest-descent move results in a feasible point. For the case of simple range constraints,

$$a_i \leq x_i \leq b_i \qquad (i=1, \ldots, n) \qquad (2.47)$$

there is a simplified method for obtaining the projected gradient. This has been incorporated into the optimum gradient procedure and is described by McGhee (1967).

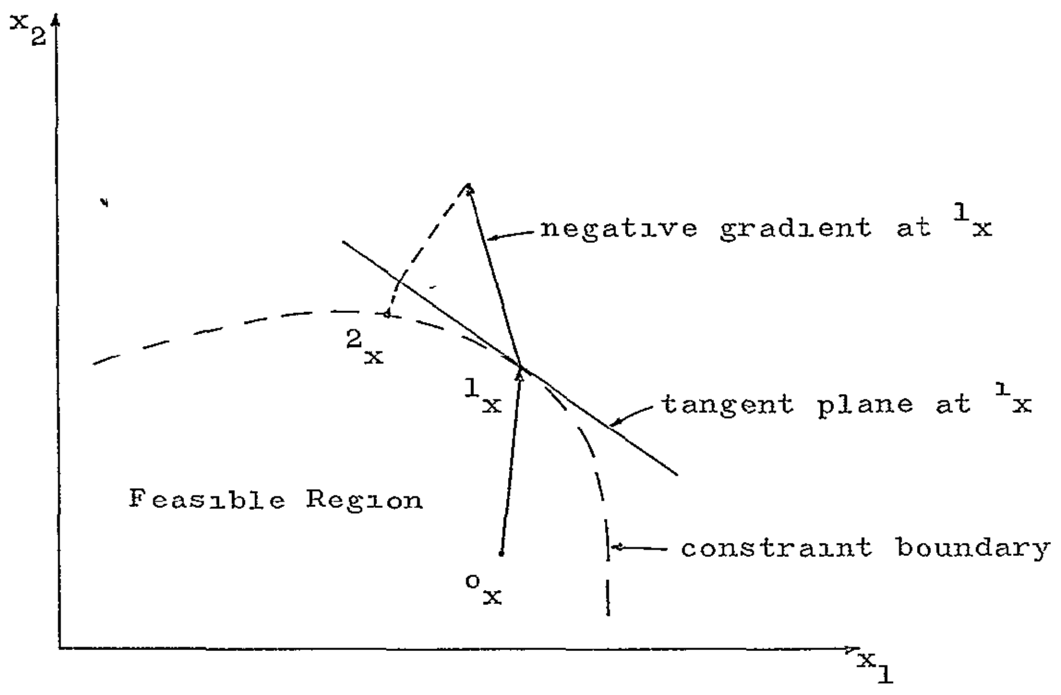


Fig. 2.11 The gradient projection method at the boundary of a nonlinear constraint.

2.8.2 Created Response-Surface Method

The created response-surface method (Fiacco and McCormick, 1964, 1968, Saaty and Bram, 1964; Wilde and Beightler, 1967) is based on the definition of a modified objective function:

$$\Phi(\underline{x},r) = F(\underline{x}) + r \sum_{i=1}^{m} 1/\emptyset_i(\underline{x}) \quad (r > 0) \tag{2.48}$$

Note that for any $r > 0$, $\Phi(\underline{x},r)$ increases rapidly as $\underline{x}$ moves toward a constraint boundary $\left\{\emptyset_i(\underline{x}) \to 0\right\}$. The technique selects values of r from a monotone decreasing sequence and optimizes $\Phi(\underline{x},r)$ for each value of r. Thus, the constrained minimization problem is converted into a sequence of unconstrained minimizations. If the optimum point of $F(\underline{x})$ is on the boundary, the minimum of $\Phi(\underline{x},r)$ approaches the boundary as $r \to 0$. In order to have $\Phi(\underline{x},r)$ well-behaved near the boundary, it is required that $F(\underline{x})$ and each of $\emptyset_i(\underline{x})$ be continuously twice differentiable and $\Phi(\underline{x},r)$ be strictly convex for each r.

Fiacco and McCormick (1964) have used the created response-surface technique with the optimum gradient method (Section 2.2) and Newton's method (Section 2.4) for minimizing $\Phi(\underline{x},r)$. Box (1965) reports that the Fletcher-Powell-Davidon method (Section 2.3) also has been employed successfully with this technique.

2.8.3 Penalty Functions

An optimization problem subject to constraints can be converted into a single unconstrained one by modifying the criterion function with the addition of penalty functions, $p_i(\phi_i)$ (Korn and Korn, 1968).

$$\Phi(\underline{x}) = F(\underline{x}) + \sum_{i=1}^{m} c_i \, p_i(\phi_i) \tag{2.49}$$

where $c_i > 0$ and

$$p_i(\phi_i) = \begin{cases} h_i(\phi_i) & \phi_i > 0 \\ 0 & \phi_i \leq 0 \end{cases}$$

and where $h_i(\phi_i)$ is a strictly monotone increasing function of ϕ_i. For $\underline{x}$ in the feasible region R, $\Phi(\underline{x}) = F(\underline{x})$, but as $\underline{x}$ moves outside R, $\Phi(\underline{x})$ is made to increase rapidly. During the optimization $\underline{x}$ is allowed to violate the constraints, but such a move is penalized by a large value of the modified criterion functions. Note that here, in contrast to the created response-surface method, the minimum of $\Phi(\underline{x})$ is found only once. The simplicity of this approach is offset by disadvantages in certain situations. It may be that $F(\underline{x})$ is undefined for $\underline{x}$ outside R--for example, $x_i < 0$ where x_i is a length or a spring constant. In such a case we might redefine $\Phi(\underline{x})$

$$\Phi(\underline{x}) = \begin{cases} F(\underline{x}) & \underline{x} \in R \\ K + \sum_{i=1}^{m} c_i \, p_i(\phi_i) & \underline{x} \notin R \end{cases} \qquad (2.50)$$

where $K \geq F(\underline{x})$ for $\underline{x}$ on the constraint boundary. In either case, unless $F(\underline{x})$ is known analytically and $h_i(\phi_i)$ can be chosen carefully, $\Phi(\underline{x})$ and/or its derivatives will be discontinuous at the boundaries. This is detrimental to search algorithms, such as the conjugate direction methods, with quadratic convergence properties.

2.8.4 Restrict to Feasible Region

For direct search methods and random methods inequality constraints may be handled by simply restricting $\underline{x}$ to the feasible region R. Before any step $\Delta\underline{x}$ is made, the values of the proposed new point $\underline{x}$ are checked, and if any constraints are violated, a different point is chosen. The search can be made to move very close to the boundary if the step size $\Delta\underline{x}$ is reduced until no constraint is violated. The simplicity of this scheme makes direct and random search methods attractive for problems where the optimum may lie close to or on a constraint boundary.

2.9 A Comparison of Methods and Some Remarks

While most of the techniques discussed in this chapter are designed to locate local minimum points, the engineer is usually seeking the best of these, the global optimum. If the value of $F(\underline{x})$ at the global optimum, F^*, is known, the optimizing algorithm can automatically escape from local minima with $F > F^*$ by expanding the search about the local minimum or jumping to a new starting point. For the more difficult case in which F^* is unknown, local minima must be detected and the values of $F(\underline{x})$ compared. Automatic search for the global optimum may be inefficient, and interaction by the operator could be valuable.

The value of easy interaction between the operator and the algorithm has been recognized by Bohling and Chernak (1965) and Carlson (1967). Displays of the performance of the system being optimized and information concerning the progress of the search help the engineer to gain insight into the behavior of the system. With this information and his own experience he may be able to help guide the search toward a solution, by changing parameters of the optimization strategy or selecting different starting points. Bohling and Chernak point out that <u>information about the system gained during the optimization may be more valuable than the final solution</u>. The opportunity for this kind of interaction has made hybrid computation attractive for optimization. However, display systems interfaced with small digital computers or time-shared computers are making easy interaction possible with all-digital optimizations as well (Korn, 1969).

The choice of a parameter optimization method for a specific problem should be guided by the computing equipment available, what is expected about the nature of the criterion function--a smooth or irregular landscape, noisy or noise-free--and the number of parameters. If the time to measure the criterion function is relatively long, then the computational effort required by complex methods will not increase the optimization time appreciably. But if the time to evaluate $F(\underline{x})$ is small compared to the time for calculations, as in the case of a high-speed analog machine interfaced to a minicomputer without floating-point hardware, then a simple direct search method or random search may be faster, even though it is less efficient in terms of function evaluations. For very irregular criterion functions derivatives may not exist at some points, and the choice of a perturbation step size for derivative measurement is difficult. Too large a step size gives a poor approximation for a derivative at a point; a step size too small may cause problems due to accuracy limitations in computing $F(\underline{x})$. Noise can cause large errors in derivative measurement. For problems with many parameters the results of Rastrigin (1963), Gurin and Rastrigin (1965), and Schumer and Steiglitz (1968) indicate that the creeping random methods are likely to require fewer function evaluations. In addition, for large n the computation times for creeping random search methods do not increase as rapidly as for algorithms requiring matrix manipulations. Korn and Kosako (1970) have successfully employed a creeping random algorithm in a 200-parameter functional-optimization problem.

If the criterion function is smooth, or if derivatives can be obtained without using the perturbation method, the conjugate direction algorithms appear to be the most efficient and most reliable. The extremely rapid convergence of Newton's method [$\Delta\underline{x} = -A^{-1}g\ (\underline{x})$] from favorable starting points is offset by the computational effort for calculation of A^{-1} and the tendency of the algorithm to diverge. When gradient measurements are easily obtained, the Fletcher-Powell-Davidon algorithm is superior. This conclusion is based on the results of Box (1966) for a series of test functions and the results of Birta and Trushel (1969), who found the F-P-D algorithm more efficient than the Fletcher-Reeves algorithm in solving optimal control problems via the Maximum Principle. Lasdon et al. (1967) found the F-R algorithm far superior to a steepest descent scheme for similar optimal control problems. The calculations for the F-R algorithm are simpler than for the F-P-D method, while the latter requires fewer function evaluations. A comparison of Partan with the conjugate direction algorithms is difficult, because there is a lack of published data for the performance of Partan on test functions and practical problems which have been solved by the conjugate-direction algorithms. Wilde and Beightler (1967) found the F-P-D algorithm more efficient in minimizing Rosenbrock's function. If the gradient of F is not readily obtained, Powell's conjugate-direction method without gradients appears to be the most efficient (Fletcher, 1965; Box, 1966). Although no published data have appeared for Zangwill's modification of Powell's method, the author has found the two to be roughly equivalent in minimizing Rosenbrock's function.

For the case of irregular criterion functions with discontinuous derivatives and possibly measurement noise, direct search methods, creeping random search and stochastic approximation are more practical. The direct-search and creeping random search algorithms decide on the next step $\Delta\underline{x}$ simply by comparing function values at different points rather than using function differences to calculate precise search directions and step sizes. Again, there is a lack of comparative data from which to judge the relative merits of the various direct-search and creeping random algorithms. But the theoretical and practical results obtained for the creeping random algorithms make a strong case for this method as an efficient and reliable technique. For noisy criterion functions the stochastic approximation algorithms have the attractive feature of convergence as the number of steps tends to infinity, but more results for test functions and practical problems are required to indicate how quickly they reach a reasonable neighborhood of the optimum.

The most obvious conclusion from a study of parameter optimization methods is that no one technique is best suited for all types of problems. An algorithm designed to be capable of minimizing all types of criterion functions will probably be inefficient for the majority of individual functions. It seems necessary to be armed with a variety of techniques in order to attack efficiently a problem with a completely unknown criterion function. Some optimization software packages, including AESOP (Hague and Glatt, 1968) and GOSPEL (Huelsman, 1968), have been developed. Such a battery of algorithms, coupled with a computer system having easy operator-machine interaction, could comprise a fruitful approach to the solution of a variety of parameter optimization problems.

CHAPTER 3

OPTIMIZATION IN THE PRESENCE OF NOISE

The problems of optimizing a noisy criterion function have been pointed out in Chapter 2. This chapter considers the evidence from the literature and some experimental results leading to the development of a strategy for optimizing noisy criterion functions (Sections 3.1 and 3.2). Constraints and the problem of estimating the criterion function are discussed in Sections 3.3 and 3.4. A specific optimization algorithm for the example problem treated in this study is described in Chapter 4.

3.1 The Choice of a Strategy

From the discussion of search methods in Chapter 2, the strategies best suited for noisy optimization appear to be stochastic approximation, direct search, and random search. However, the powerful convergence properties of the conjugate-direction methods also seem to warrant an investigation of their efficiency in the presence of noise. The only results known for gradient algorithms on a noisy function are those of Gurin and Rastrigin (1965), who concluded that a steepest-descent method is inferior to a creeping-random-search algorithm. It was felt that a conjugate-direction algorithm involving no gradient measurements but only linear minimizations might still perform well in the presence of noise.

The algorithm developed by Zangwill (1967) combined with a quadratic-interpolation method for the linear minimizations was programmed in FORTRAN for the PDP-9 computer. Gaussian noise was added to the criterion function, with the standard deviation chosen as a fraction of the value of $F(\underline{x})$. The observed function values are

$$f(\underline{x}) = F(\underline{x}) + [\sigma \cdot F(\underline{x})]z \tag{3.1}$$

where z is a Gaussian random variable with zero mean and unit variance, and σ is the coefficient specifying the standard deviation of the noise added to $F(\underline{x})$. The algorithm was applied to two test functions with given starting points, as follows

$$F(\underline{x}) = \sum_{i=1}^{n} x_i^2, \quad {}^{o}\underline{x} = (1, 1, \ldots 1) \tag{3.2}$$

$$F(\underline{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2, \quad {}^{o}\underline{x} = (-1.2, 1) \tag{3.3}$$

After each linear minimization the (noisy) function value is compared to $F_{min} = 10^{-4}$; if $f(\underline{x}) < F_{min}$, the search is ended. Zangwill's algorithm also terminates the search if n successive minimizations in the coordinate directions lead to no improvement--an indication that the gradient is zero.

To minimize the quadratic function (3.2) for $\sigma = 0.$, 9 and 20 function evaluations were required for n=2 and n=4, respectively. For $\sigma = 0.1$ and n=2, 5 of 10 trials failed to converge, because observation errors resulted in a false indication of zero gradient. For the 5 successful trials, an average of 18 function evaluations were required. For n=4, there were no failures in 10 trials, and an average of 74 function evaluations were required. For large n, there is less chance of noisy observations leading to n successive coordinate minimizations with no improvement.

For Rosenbrock's function (3.3), 135 evaluations were required to converge for $F_{min} = 10^{-4}$ and $\sigma = 0$. However, no convergence could be obtained for noise levels as small as $\sigma = .01$. Again, the algorithm terminated prematurely due to a zero-gradient indication.

<u>These results indicate that for Zangwill's algorithm to be effective in the presence of noise, the premature terminations due to false zero-gradient indications would have to be eliminated, and/or the linear-minimization algorithm would have to be improved</u>. One possibility would be to use a stochastic-approximation algorithm, such as Kushner's (1963), for the linear search.

Modification of this algorithm was abandoned, and <u>a creeping-random-search strategy was chosen for this study</u>. The reasons for this selection are summarized here.

1. Unless the variance of the measurement noise is very small, estimating the gradient of $F(\underline{x})$ by small perturbations is impractical. Gurin and Rastrigin (1965) have shown a random search algorithm to be more effective than steepest descent in the presence of noise.
2. While stochastic approximation algorithms have the attractive feature of guaranteed convergence as the number of optimizing steps tends to infinity, actual progress toward the optimum may be slow.
3. Creeping random search has been found effective in optimizing very "irregular" parameter landscapes (Maybach, 1966a, Stewart et al., 1967; Kavanaugh et al., 1968, Bekey et al., 1966).
4. The results of Rastrigin (1963) and Schumer and Steiglitz (1968) indicate that creeping random search is especially effective for problems with many parameters.
5. Creeping-random-search algorithms permit the use of a "vector-valued" criterion function (Stewart et al., 1967).
6. Constraints are easily handled by simply restricting trial steps to the feasible region of parameter space.
7. The comparatively modest computations required for random search algorithms can be programmed easily

in assembly language (instead of FORTRAN) for a small digital computer. This results in a fast digital program, which is better suited for operation with a high-speed analog machine.

3.2 A Random-Search Algorithm for Noisy Criterion Functions

For the class of problems considered in this study, the parameter vector $\underline{x}$ consists of the distribution constants introduced in Section 1.1. It is assumed that each system parameter p_i is Gaussian with mean μ_i and variance σ_i^2, so that $\underline{x}$ appears as a column vector $(\underline{\mu}, \underline{\sigma})$.

$$\underline{x} \equiv (\underline{\mu}, \underline{\sigma}) \tag{3.4}$$

We also assume constraints of the form

$$a_i \leq p_i \leq b_i \tag{3.5}$$

$$\sigma_i \geq c_i \left| \mu_i \right| \geq 0 \tag{3.6}$$

$$(i = 1, 2, \ldots, n/2)$$

These are discussed in Section 3.3.

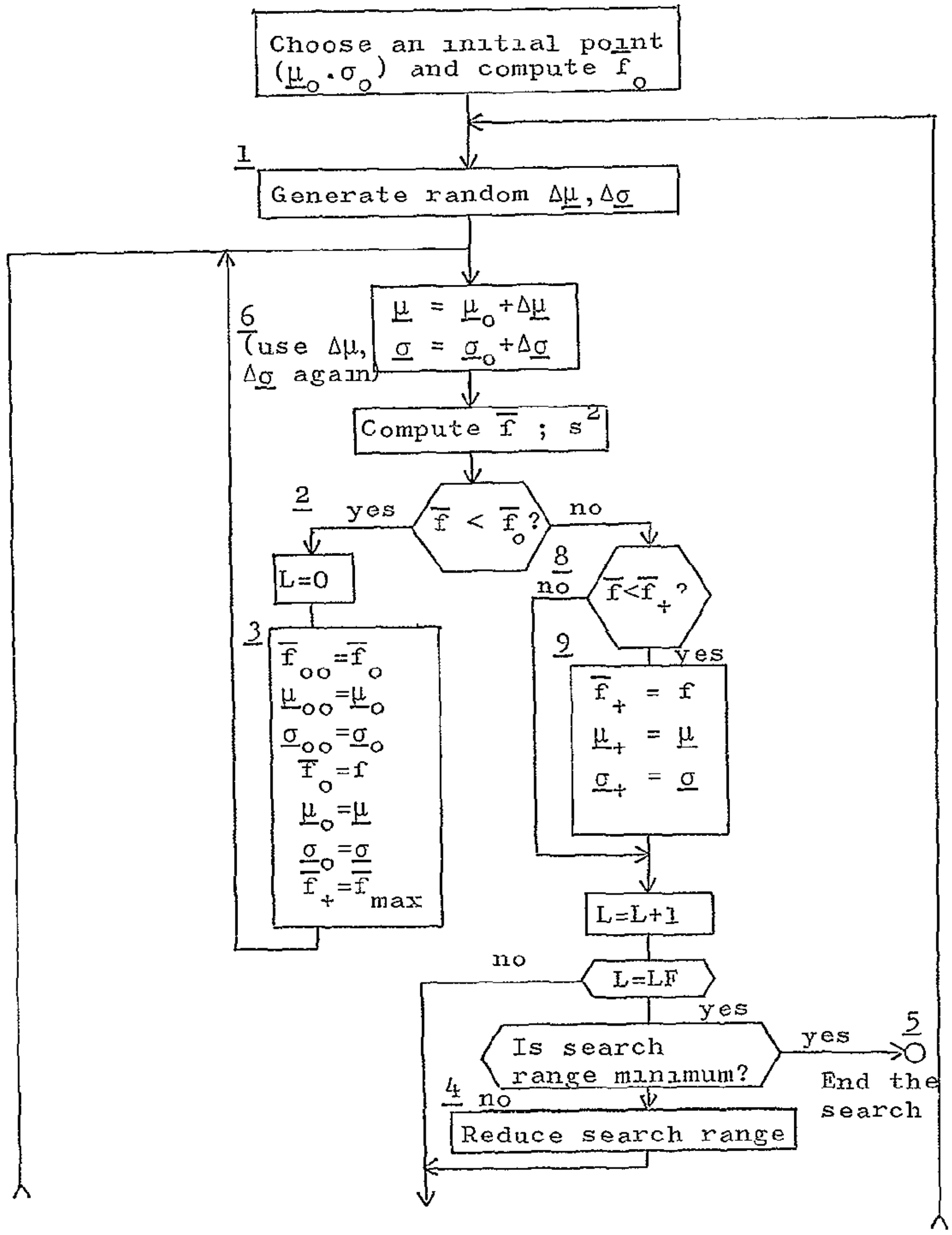
To arrive at an effective search procedure for noisy criterion functions, a basic creeping-random-search algorithm is combined with a strategy for averaging measurements of the criterion function so as to reduce the noise variance. The observations of the criterion function are represented by

$$f(\underline{x}) = F(\underline{x}) + v \tag{3.7}$$

where $F(\underline{x})$ is the true value of the criterion function, and v is a zero-mean random variable. Our estimate of $F(\underline{x})$ will be denoted by $\overline{f}(\underline{x})$. At a point $\underline{x}$ observations of the function are averaged until the variance of $\overline{f}(\underline{x})$, denoted by s^2, is less than some specified value. (A sequential estimation scheme for this is described in Section 3.4.) Before discussing the strategy for choosing s^2, the creeping random algorithm is described.

Figure 3.1 illustrates a basic creeping-random-search algorithm, which searches for a local minimum. (In the following paragraphs, numbers enclosed by brackets, [], refer to corresponding numbers in the flow diagrams.) Exploratory steps, $\Delta\underline{\mu}$ and $\Delta\underline{\sigma}$ [1], are random in magnitude and direction. When the criterion function estimate $\overline{f}$ at an exploratory point is an improvement over the current optimum value ($\overline{f} < \overline{f}_o$) [2], the center of the search is moved to the corresponding new point [3]. Following some integral number, LF, of consecutive failures, the search range is then reduced [4], and after LF failures with minimum search range, the algorithm is finished [5]. Other features of the algorithm may be noted:

1. The random parameter perturbations, $\Delta\mu_i$ and $\Delta\sigma_i$, are chosen from a uniform distribution, which has a variance proportional to the current optimum value of μ_i. For most problems this method of choosing the variance of the perturbations appears



Choose an initial point $(\underline{\mu}_o, \sigma_o)$ and compute f_o
1
Generate random $\Delta\underline{\mu}, \Delta\underline{\sigma}$
$\underline{\mu} = \underline{\mu}_o + \Delta\underline{\mu}$
$\underline{\sigma} = \underline{\sigma}_o + \Delta\underline{\sigma}$
6
(use $\Delta\mu$, $\Delta\underline{\sigma}$ again)
Compute $\overline{f}$; s^2
2
yes
$\overline{f} < \overline{f}_o$?
no
L=0
3
$\overline{f}_{oo} = \overline{f}_o$
$\underline{\mu}_{oo} = \underline{\mu}_o$
$\underline{\sigma}_{oo} = \underline{\sigma}_o$
$\overline{f}_o = f$
$\underline{\mu}_o = \underline{\mu}$
$\underline{\sigma}_o = \underline{\sigma}$
$\overline{f}_+ = \overline{f}_{max}$
8
no
$\overline{f} < \overline{f}_+$?
9
yes
$\overline{f}_+ = f$
$\underline{\mu}_+ = \underline{\mu}$
$\underline{\sigma}_+ = \underline{\sigma}$
L=L+1
no
L=LF
yes
Is search range minimum?
yes
5
End the search
4
no
Reduce search range

Fig. 3.1 Flow diagram for the basic creeping-random-search algorithm.

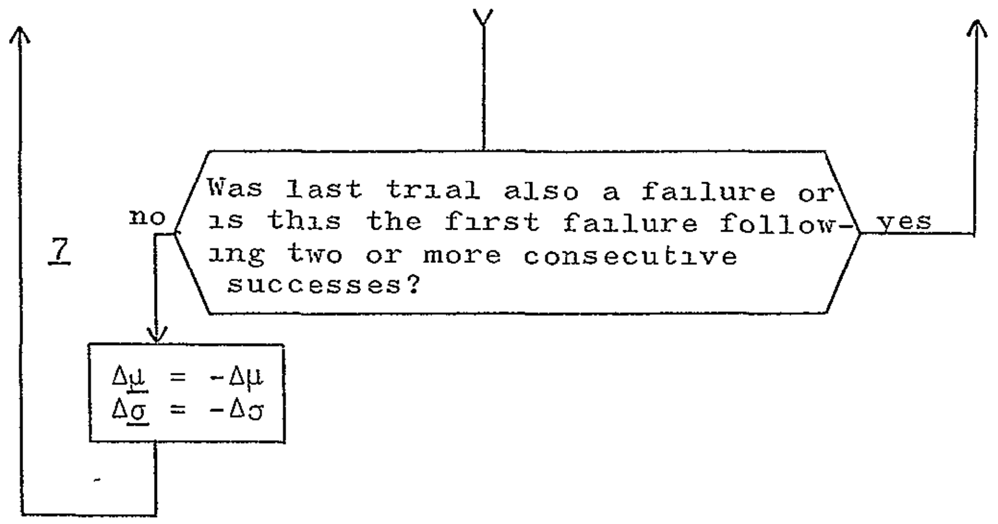

Was last trial also a failure or is this the first failure following two or more consecutive successes?
no
yes
7
$\Delta\underline{\mu} = -\Delta\mu$
$\Delta\underline{\sigma} = -\Delta\sigma$

Fig. 3.1--Continued

more logical than having a fixed parameter-perturbation variance. In the latter case, the same perturbation, $\Delta\mu_i$, can represent a very large or a very small percentage change in the parameter value, depending on the current optimal value of μ_i. For the same reason, the standard deviations σ_i are expressed in the program as percentages of the corresponding mean values.

2. The algorithm employs absolute positive biasing [6] and absolute negative biasing [7] as described in Section 2.6.

3. During the optimization the program keeps track of both the previous optimum point $(\underline{\mu}_{oo}, \underline{\sigma}_{oo}; \overline{f}_{oo})$ [3] and the point with the smallest function value since the last success $(\underline{\mu}_+, \underline{\sigma}_+; \overline{f}_+ : \overline{f}_o < \overline{f}_+ < \overline{f}_i$, where i indexes all of the other failure points since the last success) [8 and 9]. Saving these values has no effect on the basic creeping-random-search algorithm, but they will be used in the overall strategy described below.

<u>To implement the creeping-random-search method, the variance s^2 allowed in the estimate $\overline{f}$ of F, must be specified</u>. Let us assume that the optimization must be accomplished with some number N of criterion function observations. If s^2 is chosen to be small, then there will

be few errors in deciding whether a trial step is a success or failure, in spite of the noise, but we will be able to take only a small number of trial steps. For a large value of s^2, more trial steps are possible, but many of our success-failure decisions are likely to be erroneous. In particular, the situation pictured in Fig. 3.2 may result. The optimization has proceeded to the point ${}^1x$, at which the estimate $\overline{f}({}^1x)$ is unusually noisy. From this point, it is difficult to find a successful step; either another unusually noisy observation will have to occur, which may require many trials, or else a large trial step toward x^* must be generated. These considerations suggest that <u>the choice of the variance</u> s^2 <u>is an important one in determining the success of the optimization</u>.

If the starting point for the search is in a "smooth" region of the criterion surface where there is an appreciable gradient, the creeping-random-search algorithm can progress well, even when the variance of $\overline{f}$ is large. Thus, in the initial search our estimates of F are allowed to be rather coarse. If many exploratory-step failures occur consecutively, indicating that the search has encountered a ridge, entered a region of small gradient, or (later in the search) approached the optimum, then the estimation algorithm should be made to reduce the variance of $\overline{f}$.

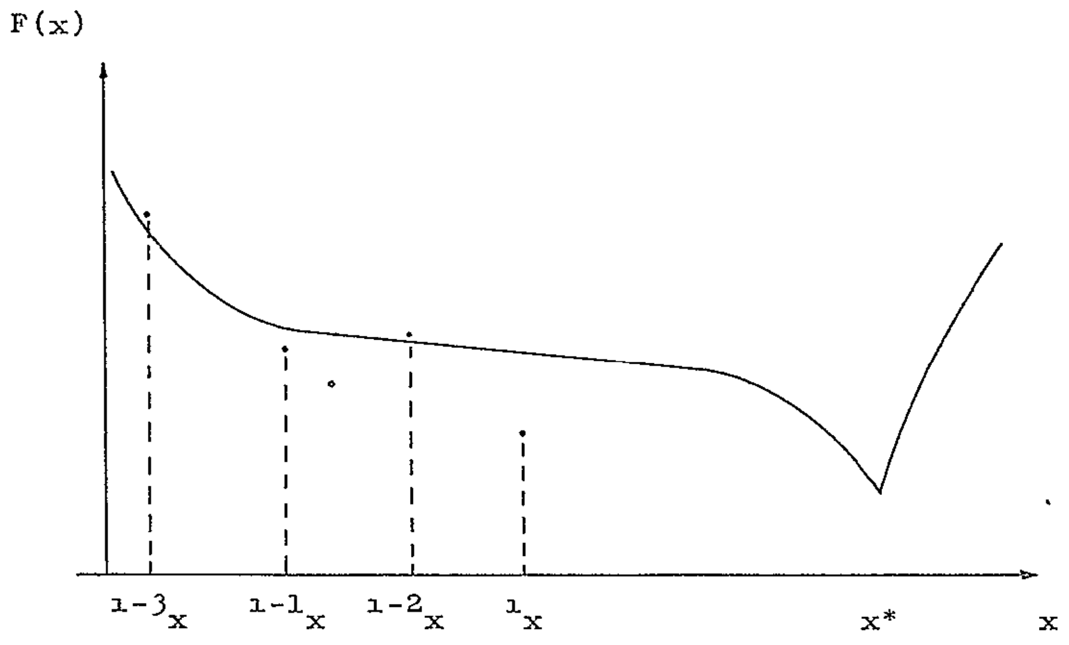


Fig. 3.2 A creeping random search in a region of small gradient for a noisy criterion function.

The flow diagram for the algorithm is shown in Fig. 3.3. The "initial search" is executed with coarse estimates of $F(\underline{\mu},\underline{\sigma})$. When LF consecutive trial steps result in no improvement, the algorithm proceeds to the "final search" [1]. At this point, a better estimate of $F(\underline{\mu}_o,\underline{\sigma}_o)$ is computed [2]. More observations are taken at the point $(\underline{\mu}_o,\underline{\sigma}_o)$ until the variance of the estimate $\overline{f}_o$ is less than or equal to s_o^2 $(s_o^2 < s^2)$. The recalculation of $\overline{f}_o$ is designed to avoid the type of difficulty illustrated in Fig. 3.2. In general, note that whenever a more accurate estimate of $F(\underline{\mu},\underline{\sigma})$ is computed, previous observations at $(\underline{\mu},\underline{\sigma})$ are utilized, thus saving computer time. Following the recalculation of $\overline{f}_o$, the algorithm proceeds by the following steps:

1. $\overline{f}_o$ is compared to the previous optimum $\overline{f}_{oo}$, in case the move from $(\underline{\mu}_{oo},\underline{\sigma}_{oo})$ to $(\underline{\mu}_o,\underline{\sigma}_o)$ was erroneous [3].
2. The minimum of $\overline{f}_o$ and $\overline{f}_{oo}$ is also compared to the "best" of the failures $(\overline{f}_+)$, in case a very noisy observation at $(\underline{\mu}_o,\underline{\sigma}_o)$ had resulted in rejecting an improved point [4].
3. After the minimum of $\overline{f}_o$, $\overline{f}_{oo}$, and $\overline{f}_+$ is determined and labeled $\overline{f}_o$, the creeping random algorithm is continued [5]. For trial steps $(\underline{\mu}_o + \Delta\underline{\mu},\ \underline{\sigma}_o + \Delta\underline{\sigma})$ the variance of $\overline{f}$ is still only required to be less than s^2. But if an improvement is indicated,

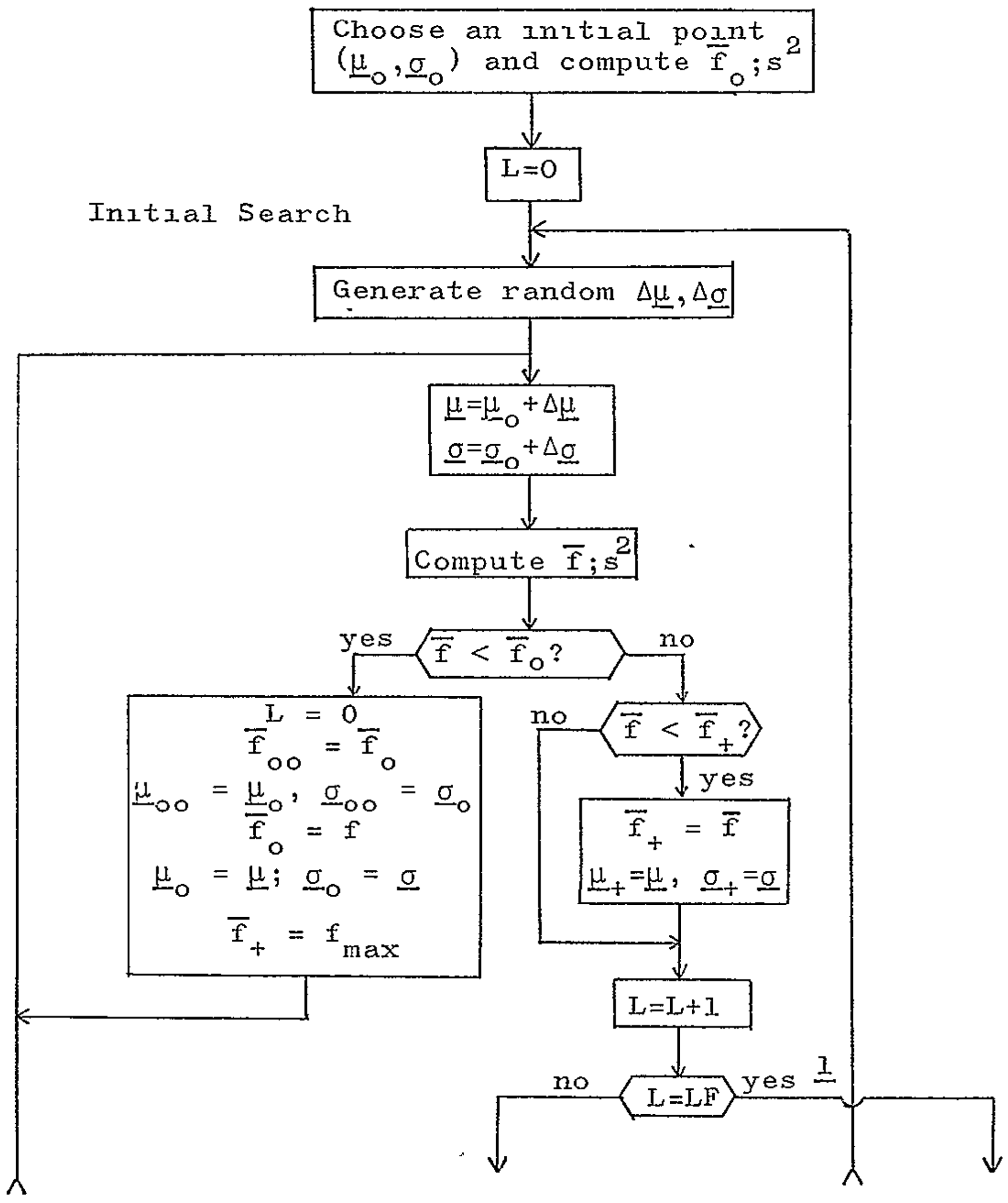
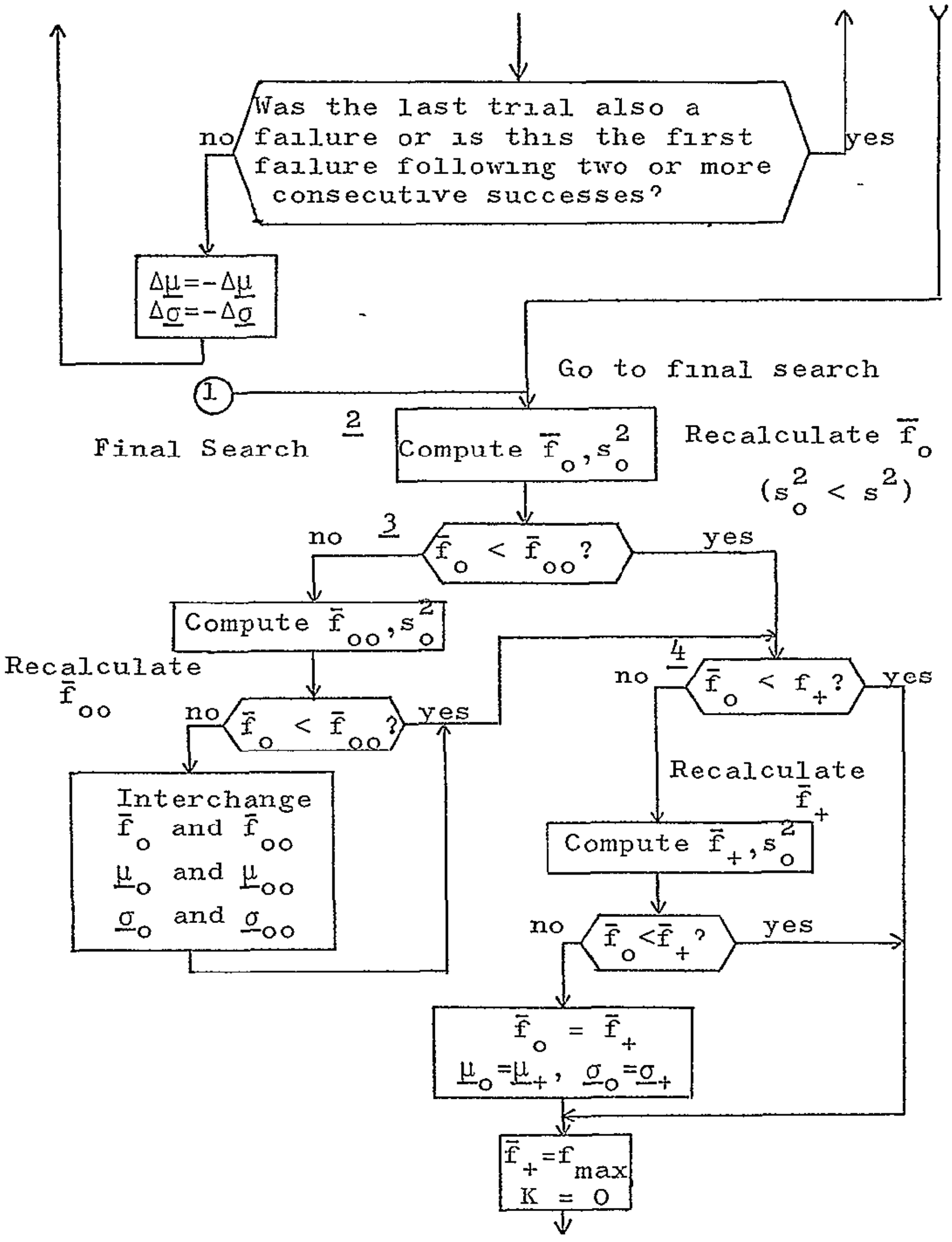

Choose an initial point $(\underline{\mu}_o, \underline{\sigma}_o)$ and compute $\bar{f}_o; s^2$
L=0
Initial Search
Generate random $\Delta\underline{\mu}, \Delta\underline{\sigma}$
$\underline{\mu} = \underline{\mu}_o + \Delta\underline{\mu}$
$\underline{\sigma} = \underline{\sigma}_o + \Delta\underline{\sigma}$
Compute $\bar{f}; s^2$
$\bar{f} < \bar{f}_o$?
yes
no
L = 0
$\bar{f}_{oo} = \bar{f}_o$
$\underline{\mu}_{oo} = \underline{\mu}_o$, $\underline{\sigma}_{oo} = \underline{\sigma}_o$
$\bar{f}_o = f$
$\underline{\mu}_o = \underline{\mu}$; $\underline{\sigma}_o = \underline{\sigma}$
$\bar{f}_+ = f_{max}$
no
$\bar{f} < \bar{f}_+$?
yes
$\bar{f}_+ = \bar{f}$
$\underline{\mu}_+ = \underline{\mu}$, $\underline{\sigma}_+ = \underline{\sigma}$
L=L+1
no
L=LF
yes
1

Fig. 3.3 A flow diagram for the optimization algorithm.



Was the last trial also a failure or is this the first failure following two or more consecutive successes?
no
yes
$\Delta\mu=-\Delta\mu$
$\Delta\sigma=-\Delta\sigma$
1
Go to final search
Final Search
2
Compute $\bar{f}_o, s_o^2$
Recalculate $\bar{f}_o$
$(s_o^2 < s^2)$
3
no
yes
$\bar{f}_o < \bar{f}_{oo}$?
Compute $\bar{f}_{oo}, s_o^2$
Recalculate $\bar{f}_{oo}$
no
yes
$\bar{f}_o < \bar{f}_{oo}$?
Interchange $\bar{f}_o$ and $\bar{f}_{oo}$, μ_o and μ_{oo}, σ_o and σ_{oo}
4
no
yes
$\bar{f}_o < f_+$?
Recalculate $\bar{f}_+$
Compute $\bar{f}_+, s_o^2$
no
yes
$\bar{f}_o<\bar{f}_+$?
$\bar{f}_o = \bar{f}_+$
$\mu_o=\mu_+$, $\sigma_o=\sigma_+$
$\bar{f}_+=f_{max}$
K = 0

Fig. 3.3--Continued A flow diagram for the optimization algorithm.

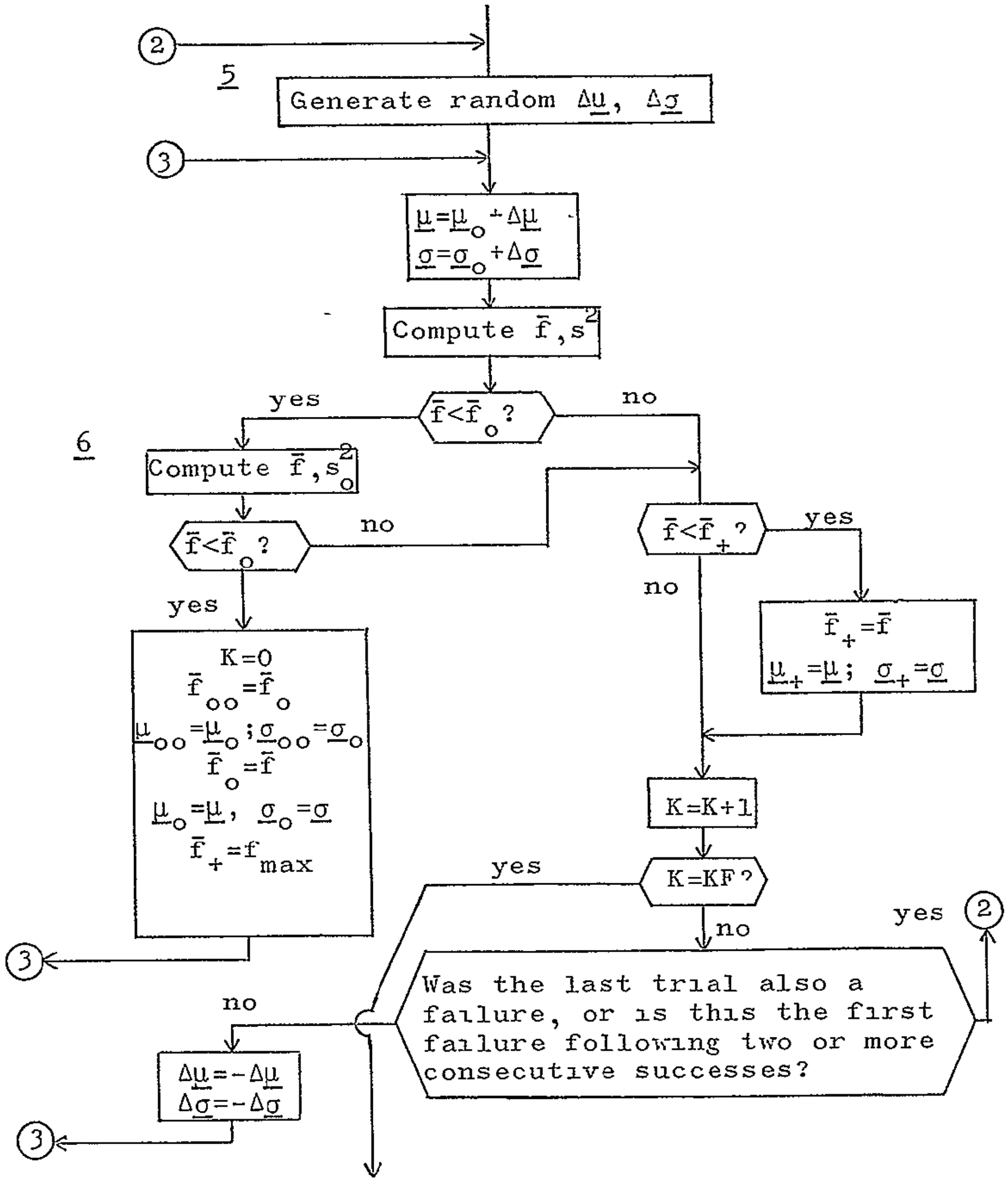


Fig. 3.3--Continued A flow diagram for the optimization algorithm.

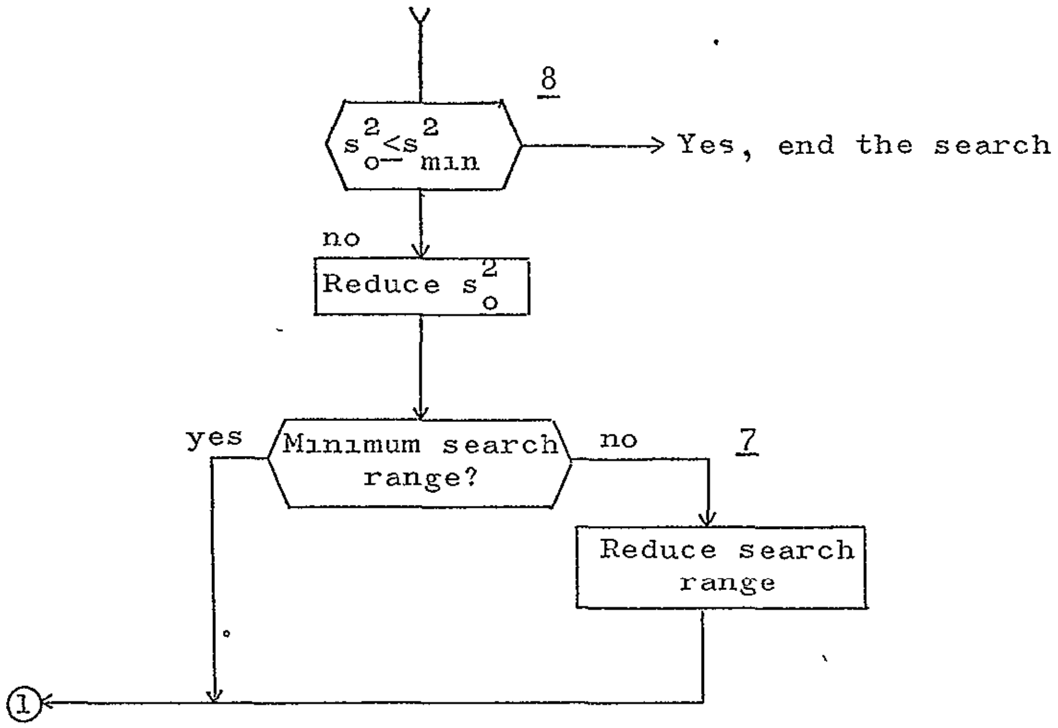


Fig. 3.3--Continued A flow diagram for the optimization algorithm.

$(\bar{f} < \bar{f}_o)$, $\bar{f}$ is recalculated and again compared to $\bar{f}_o$ [6]. This strategy allows for a greater number of trial steps to be taken.

4. Following KF successive failures in the final search, the search range is reduced [7], and the algorithm returns to recalculate $\bar{f}_o$ again.
5. When KF consecutive failures occur with s_o^2 at its minimum value (s^2_{min}), the search is terminated [8].

In order to use this random-search method, a starting point $(\underline{\mu}_o, \sigma_o)$, starting and minimum values of the search range, and values for LF, KF, s^2, s_o^2, and s^2_{min} must be specified. In the absence of any prior knowledge of the nature of the criterion function, it is likely that initial choices for these values may result in an inefficient search. It is felt that a solution to this problem lies in a provision for communication between the search algorithm and the operator. Such a facility for interaction with the computing system employed for this study is described in Section 4.3.

3.3 Constraints, Modeling the Distributions of the System Parameters

The constraints on the system parameters p_i and the distribution constants σ_i are specified by the inequalities (3.5) and (3.6). The constraints on p_i may arise from design limits set by the engineer or from considerations of realizability of the physical system being modeled. For example, if p_i is the mass of a flywheel, the design engineer may place an upper limit on p_i, and physical realizability requires $p_i \geq 0$. The constraints on σ_i may be necessary in a situation where it is known that production tolerances cannot be held below a certain percentage of the design values μ_i.

In the form of (3.5), the constraints on the p_i's are inconvenient to enforce. After values of $\underline{\mu}$ and $\underline{\sigma}$ for a trial step are selected by the optimization program, many values of p_i are generated in order to estimate $F(\underline{\mu},\underline{\sigma})$. Checking each value of p_i is time consuming. Furthermore, if, after many observations of the criterion function, a value of p_i violates the constraints, new values of $\underline{\mu}$ and $\underline{\sigma}$ must be selected and the estimation of $F(\underline{\mu},\underline{\sigma})$ begun again. To avoid this waste of computer time, the constraints of (3.5) are replaced by

$$\mu_i - r\,\sigma_i \geq a_i$$

$$\mu_i + r\,\sigma_i \leq b_i \tag{3.8}$$

For r=3, only 0.27% of the sample values of a Gaussian random variable will violate the constraints of (3.5) when $\mu_i - r\sigma_i = a_i$ and $\mu_i + r\sigma_i = b_i$. With this form of the constraints, feasible values of $\underline{\mu}$ and $\underline{\sigma}$ may be selected before the estimation of $F(\underline{\mu},\underline{\sigma})$ is begun.

Pseudo-Gaussian samples for the random variables p_i are generated by adding and normalizing ten uniformly-distributed random numbers from a hardware random-noise generator interfaced to the digital computer (Belt, 1969). This provides for deviations from the mean as large as 5.5 σ. We introduce negative correlation into our random-parameter sample (Korn, 1966) by selecting ${}^{2}p_i$, ${}^{4}p_i$, ..., ${}^{n}p_i$ with deviations about μ_i which are equal and opposite to the deviations of ${}^{1}p_i$, ${}^{3}p_i$, ..., ${}^{n-1}p_i$, i.e.,

$$ {}^{k+1}p_i = \mu_i - ({}^{k}p_i - \mu_i) \qquad (k = 1, 3, 5, \ldots, n-1) $$

This ensures that the sample mean is equal to μ_i, and time is saved, since only n/2 pseudo-Gaussian random numbers are generated.

Although the inequalities (3.8) are a practical way of enforcing constraints on almost all of the p_i's, values of p_i which violate (3.8) must still be accommodated by the analog machine used to estimate $F(\underline{\mu},\underline{\sigma})$. Thus, all values of p_i are limited to the range of the analog computer to produce a new random variable p_i'.

$$ p_i' = \begin{cases} 1 \text{ m.u. if } p_i > 1 \text{ m.u.} \\ -1 \text{ m.u. if } p_i < -1 \text{ m.u.} \\ p_i \text{ otherwise} \end{cases} \qquad (3.9) $$

where 1 m.u. denotes one machine unit for the analog computer. In general, the limited random variable p_i' will have a new mean μ_i' and standard deviation σ_i' different from μ_i and σ_i. The effect on μ_i is most severe if p_i is limited on only one side of the distribution. If b_i corresponds to 1 machine unit, as shown in Fig. 3.4, all values of $p_i > \mu_i + r\sigma_i$ will be set equal to 1 machine unit. The effect on σ_i is greatest when p_i is limited on both sides of the distribution, a_i and b_i correspond to -1 m.u. and 1 m.u. respectively (Fig. 3.5). The effects of these two cases of limiting are calculated in Appendix A, and results are shown in Table 3.1. For the problem described in Chapter 4, $a_i = 0$ and $b_i = 1$ m.u. The values of μ_i and σ_i (satisfying the constraints) which result in maximum σ_i are $\mu_i = 0.5$ m.u. and $\sigma_i = .5/3$ m.u. For this worst case and for $r = 3$, $\mu_i' = \mu_i - .00051$ m.u. $= \mu_i - .0051$ volts for the $\pm$ 10 volt range of ASTRAC-II. This 5 mv. worst case error is approximately equal to the accuracy of setting the values of p_i' by the digital-analog converters on ASTRAC-II. The worst case error in the standard deviation σ_i is approximately 0.13%.

3.4 Sequential Estimation of $F(\mu,\sigma)$

The criterion function $F(\underline{\mu},\underline{\sigma})$ is estimated from observations denoted by $f(\underline{\mu},\underline{\sigma}) = F(\underline{\mu},\underline{\sigma}) + v$, where v is a zero-mean random variable. An unbiased estimate of F based on n observations is

$$^{n}\overline{f} = \frac{1}{n} \sum_{i=1}^{n} {}^{i}f \qquad (3.10)$$

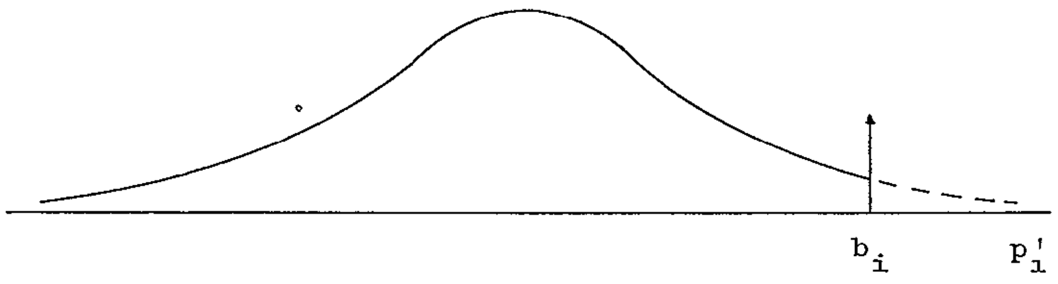


Fig. 3.4 The Gaussian density function limited at one end.

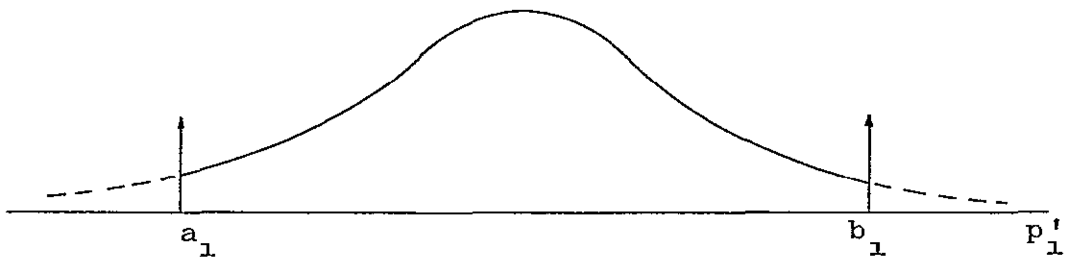


Fig. 3.5 The Gaussian density function limited at both ends.

Table 3.1 The effects of limiting a Gaussian random variable.

r	μ'	σ'
	<u>Limiting at $r\sigma$ at one end of a Gaussian distribution</u>	
1	$\mu - .0833\sigma$	$.8667\sigma$
2	$\mu - .0312\sigma$	$.9794\sigma$
3	$\mu - .00308\sigma$	$.9987\sigma$
4	$\mu - .00010\sigma$	$.99997\sigma$
	<u>Limiting at $r\sigma$ at both ends of a Gaussian distribution</u>	
1		$.7183\sigma$
2		$.9594\sigma$
3		$.9975\sigma$
4		$.9999\sigma$

The sample variance

$$^{n}S^{2} = \frac{1}{n-1} \sum_{i=1}^{n} ({}^{i}f - {}^{n}\overline{f})^{2} \tag{3.11}$$

is an unbiased estimate of Var{f}. If f has a Gaussian distribution then

$$\frac{({}^{n}\overline{f} - F)\sqrt{n}}{{}^{n}S} = t_{n-1} \tag{3.12}$$

where t_{n-1} has the Student-t distribution with n-1 degrees of freedom. This allows us to make a confidence statement about our estimate of F. Before sampling, we can state that the probability that our estimate $^{n}\overline{f}$ will differ from F by some amount less than d is given by

$$P[\,|{}^{n}\overline{f} - F| \leq d] = 1 - \alpha \tag{3.13}$$

where

$$d = \frac{{}^{n}S \; t_{n-1;\alpha/2}}{\sqrt{n}} \tag{3.14}$$

and $t_{n-1;\alpha/2}$ is the value of the Student-t variable such that

$$\int_{t_{n-1;\alpha/2}}^{\infty} \phi(z)dz = \alpha/2$$

(ϕ is the density function for the Student-t variable with n-1 degrees of freedom). To use this statement in deciding the number of observations to make for our estimate, samples of f are taken until ${}^{n}S/\sqrt{n}$ is small enough so that $({}^{n}S\ t_{n-1;\alpha/2})/\sqrt{n} \leq d$.

In order to implement such a <u>sequential estimation</u> scheme, it is convenient to have <u>recursive estimates</u> of ${}^{n}\overline{f}$ and ${}^{n}S^2$ rather than performing the summations of Eqns. (3.10) and (3.11) after each observation. Recurrence relations are given by Korn (1966). Let ${}^{n}s^2 = \frac{n-1}{n}\,{}^{n}S^2$.

$$ {}^{n}\overline{f} = {}^{n-1}\overline{f} + \frac{1}{n}\left({}^{n}f - {}^{n-1}\overline{f}\right) \tag{3.15} $$

$$ {}^{n}s^2 \approx {}^{n-1}s^2 + \frac{1}{n}\left[\left({}^{n}f - {}^{n}\overline{f}\right)^2 - {}^{n-1}s^2\right] \tag{3.16} $$

Note that for large n, ${}^{n}s^2 \approx {}^{n}S^2$. Updating ${}^{n}\overline{f}$ and ${}^{n}s^2$ with these relations requires a division by n, which may be time consuming when floating-point or double-precision fixed-point arithmetic is necessary to obtain accurate estimates. Deardorff and Trimble (1968) replace the division by n by a division by a power of two to obtain the so-called "stable-averaging" algorithm.

$$ {}^{n}\hat{f} = {}^{n-1}\hat{f} + \frac{1}{2^{N_n}}\left({}^{n}f - {}^{n-1}\hat{f}\right) \quad \left(2^{N_n - 1} < n \leq 2^{N_n}\right) \tag{3.17} $$

This algorithm is considerably faster than Eq. (3.15), because the division can be accomplished by a simple shift operation in a binary computer. However, $2^{N_n} \leq n$ for all n, so that the variance of ${}^{n}\hat{f}$ is greater than the variance of ${}^{n}\bar{f}$ (the minimum-variance linear unbiased estimate of the expected value of f).

The variance of the "stable-averaging" estimate can be reduced by modifying the choice of N_n so that 2^{N_n} is more nearly equal to n (White, 1970). The modified estimate is defined by

$$ {}^{n}\tilde{f} = {}^{n-1}\tilde{f} + \frac{1}{2^{M_n}} ({}^{n}f - {}^{n-1}\tilde{f}) \quad (2^{M_n - 1} \leq n \leq 2^{M_n}) \qquad (3.18) $$

The method of uniquely determining M_n is most easily shown by a flow diagram (Fig. 3.6). Table 3.2 lists the resulting sequences $\{n\}$, $\{2^{N_n}\}$, $\{2^{M_n}\}$. It is seen that this method of choosing the power of two in Eq. (3.18) yields a divisor closer to the ideal value n than Eq. (3.17), and the increased time needed to generate M_n rather than N_n is small. White (1970) shows that for $n \geq 100$ only about 5% more observations are required with the modified algorithm (3.18) to reduce the standard deviation of ${}^{n}\tilde{f}$ to that of ${}^{n}\bar{f}$ in Eq. (3.15). This should be compared with 15-20 per cent more observations required with the "stable averaging" estimate ${}^{n}\hat{f}$. The improvement appears modest, but represents a very substantial saving in cases where $E\{f\}$ must be estimated many times at best possible speed.

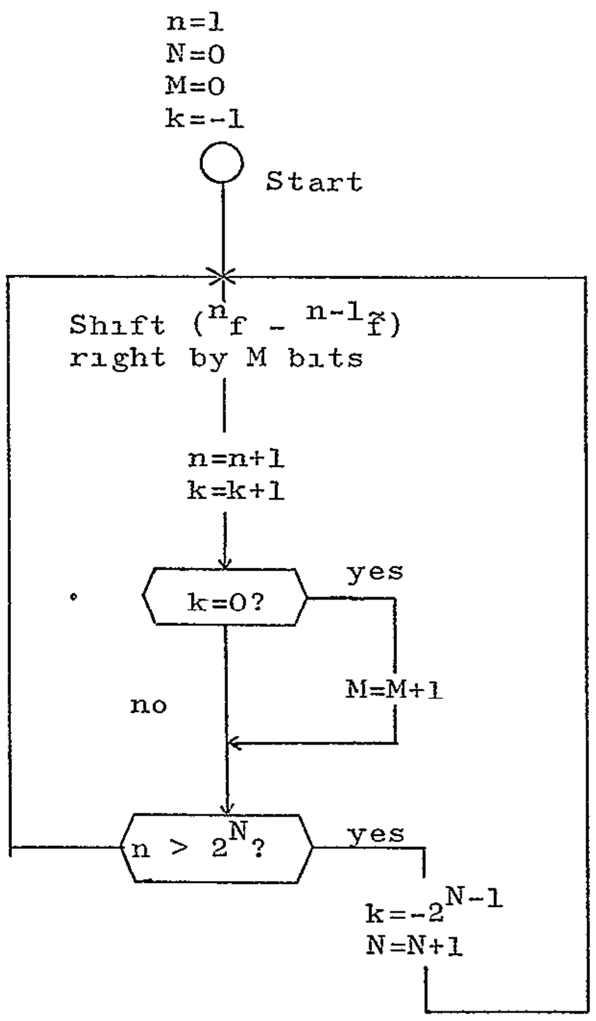


Fig. 3.6 Flow diagram for the shift operation in Eq. (3.18).

Table 3.2 Divisors used in the three recursive estimation algorithms.

n	2^{N_n}	2^{M_n}
1	1	1
2	2	2
3	4	2
4	4	4
5	8	4
6	8	4
7	8	8
8	8	8
9	16	8
.	.	.
.	.	.
.	.	.
12	16	8
13	16	16
.	.	.
.	.	.
.	.	.
17	32	16
.	.	.
.	.	.
.	.	.

CHAPTER 4

AN OPTIMIZATION EXPERIMENT

In 1959, McGhee and Levine (1964) studied the problem of the determination of optimum production tolerances for a hypothetical radar-homing missile. Their experiment was performed before fast analog-digital hybrid computers were generally available. An analog computer was used to simulate flights of missiles having production variations in two guidance-unit parameters, a gain K and a time constant τ, which were modeled as Gaussian random variables with means μ_K and μ_τ and variances σ_K^2 and σ_τ^2. Values of μ_K and μ_τ were selected prior to the simulation and were held constant during their experiments. For sixteen combinations of values of σ_K and σ_τ, an average performance index (the probability of hitting a target) was estimated by Monte-Carlo simulation. A digital computer then performed a quadratic regression analysis on these data in order to arrive at an expression for the hit probability as a function of σ_K and σ_τ. Surprisingly, it was found that for σ_τ equal to 20% or 30% of μ_τ, increasing σ_K from 10% of μ_K to 20% caused an <u>increase</u> in the hit probability. Thus, <u>the popular assumption that performance</u>

is degraded by increasing production tolerances is not always valid.

This chapter discusses the simulation and optimization of a similar missile system. In this experiment the mean values, μ_K and μ_T, are optimized simultaneously with the variances, σ_K^2 and σ_T^2.

4.1 A Radar-Homing Missile Problem

Figure 4.1 illustrates the motion in one plane of a hypothetical radar-homing missile. In the diagram we consider a small change $\underline{\delta v}$ in the missile velocity vector $\underline{v}$. For a small angle δr, $|\underline{v} + \underline{\delta v}| \approx |\underline{v}| = v_m$. The acceleration normal to $\underline{v}$ is

$$v_m \frac{dr}{dt} = v_m \dot{r} \tag{4.1}$$

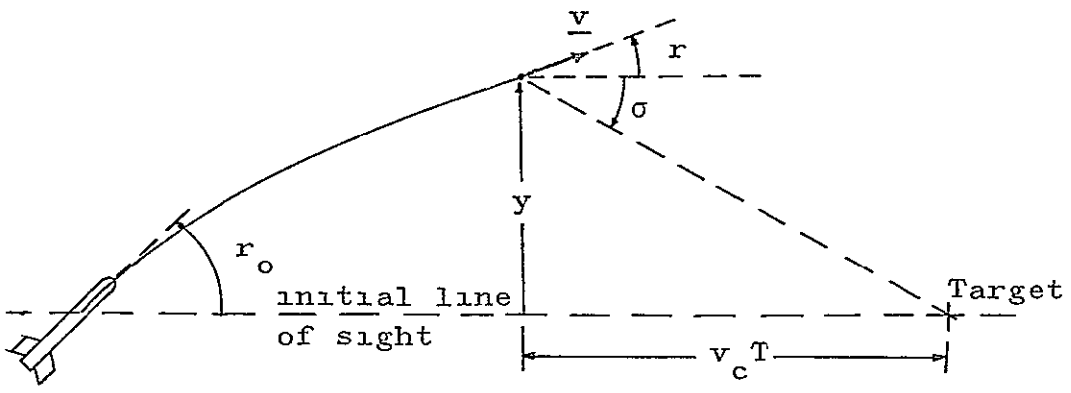
or

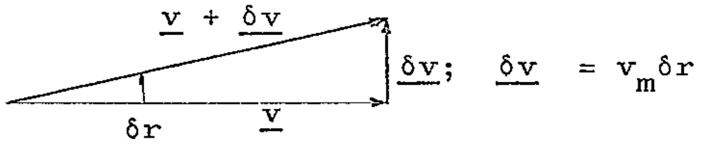
$$\ddot{y} = v_m \dot{r}$$

$$\dot{y}(t) = \int_0^t v_m \dot{r}(s) ds - v_m r_o \tag{4.2}$$

Equation (4.2) describes the kinematics of the missile. For a small angle σ,

$$\sigma = \arctan \frac{y}{v_c T} \approx \frac{y}{v_c T} \tag{4.3}$$





$\underline{v} + \underline{\delta v}$
$\underline{\delta v}$; $\underline{\delta v} = v_m \delta r$
δr
$\underline{v}$

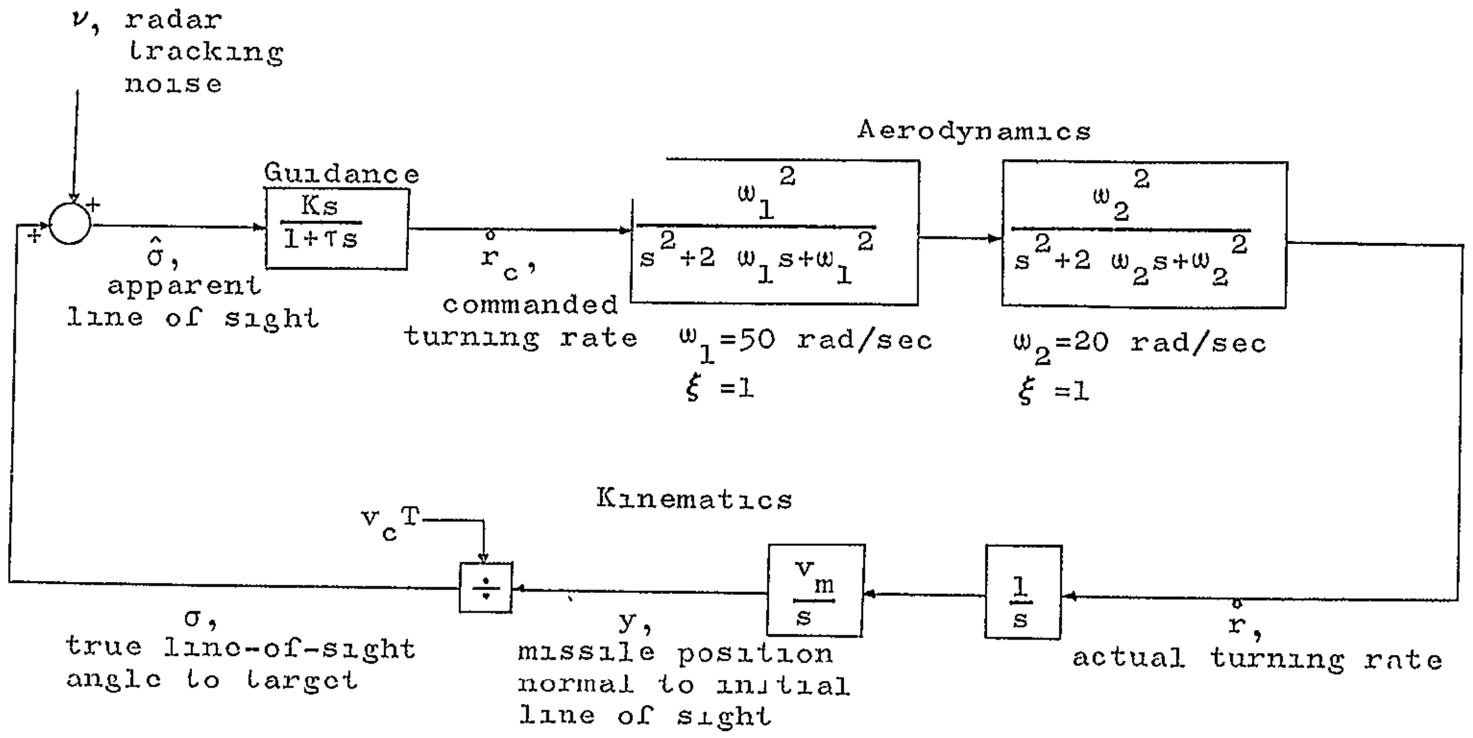
y	=	missile position normal to initial line of sight (ft.)
v_c	=	missile-to-target closing velocity (ft./sec.)
T	=	time to go until impact (sec.)
r	=	angle between missile velocity vector and initial line of sight (rad.)
σ	=	true line of sight angle
$\underline{v}$	=	missile velocity vector; $\lvert\underline{v}\rvert = v_m$ (ft./sec.)

Fig. 4.1 The motion of the missile in a plane.

The line-of-sight angle σ is used by the guidance unit to steer the missile toward the target, as shown in the block diagram of Fig. 4.2. The guidance-unit output is a commanded turning rate $\dot{r}_c$, and the missile aerodynamics produce an actual turning rate $\dot{r}$. At time t=0, the missile is given a random heading angle, $r(0) = r_o$, which is chosen from a Gaussian distribution with zero mean and a standard deviation of 0.1 radians. At $t = t_f = 7.$ sec, the missile position normal to the initial line of sight, y, is measured. If $|y(t_f)| \leq = 30.$ ft., we say that the missile has hit the target. The line-of-sight angle σ is corrupted by wideband radar-tracking noise with $\Phi(0) = .0155$ deg.^{2}/Hz, where $\Phi(\omega)$ is the two-sided power spectral density. The navigation gain, K, and the principal missile filtering time constant, τ, are assumed to be Gaussian with mean μ_K and variance σ_K^2, and Gaussian with mean μ_τ and variance σ_τ^2, respectively. <u>The problem is to choose the values of μ_K, μ_τ, σ_K, and σ_τ which maximize the probability of hitting the target</u>.

In the notation of Chapter 1, the system parameters are $\underline{p} \equiv (K,\tau)$, and the distribution constants are $\underline{x} = (\mu_K, \mu_\tau, \sigma_K, \sigma_\tau)$. The performance index is given by:

$$J = \begin{cases} 1 & \text{if } |y(t_f)| \leq 30 \text{ ft. (hit)} \\ 0 & \text{otherwise (miss)} \end{cases}$$



ν, radar tracking noise
+
+
Guidance
$\frac{Ks}{1+\tau s}$
$\hat{\sigma}$, apparent line of sight
$\dot{r}_c$, commanded turning rate
Aerodynamics
$\frac{\omega_1^2}{s^2+2\,\omega_1 s+\omega_1^2}$
$\omega_1=50$ rad/sec
$\xi=1$
$\frac{\omega_2^2}{s^2+2\,\omega_2 s+\omega_2^2}$
$\omega_2=20$ rad/sec
$\xi=1$
Kinematics
$v_c T$
$\div$
$\frac{v_m}{s}$
$\frac{1}{s}$
σ, true line-of-sight angle to target
y, missile position normal to initial line of sight
$\dot{r}$, actual turning rate

v_c, missile-to-target closing velocity = 2913 ft/sec

v_m, missile velocity = 1942 ft/sec

T, time to go until impact = 7. sec initially

Fig. 4.2 Radar-homing missile navigating in a plane.

The average performance index is the <u>hit probability</u>

$$\Psi = E\{J\} = \text{probability of a hit.}$$

Since a cost function is not included for this problem, the criterion function simply equals the average performance index

$$F(\underline{x}) = \Psi = \text{probability of a hit.}$$

The inequality constraints are:

$$K \geq 0$$
$$\tau \geq 0$$
$$\sigma_K \geq c_K \left|\mu_K\right| \geq 0$$
$$\sigma_\tau \geq c_\tau \left|\mu_\tau\right| \geq 0.$$

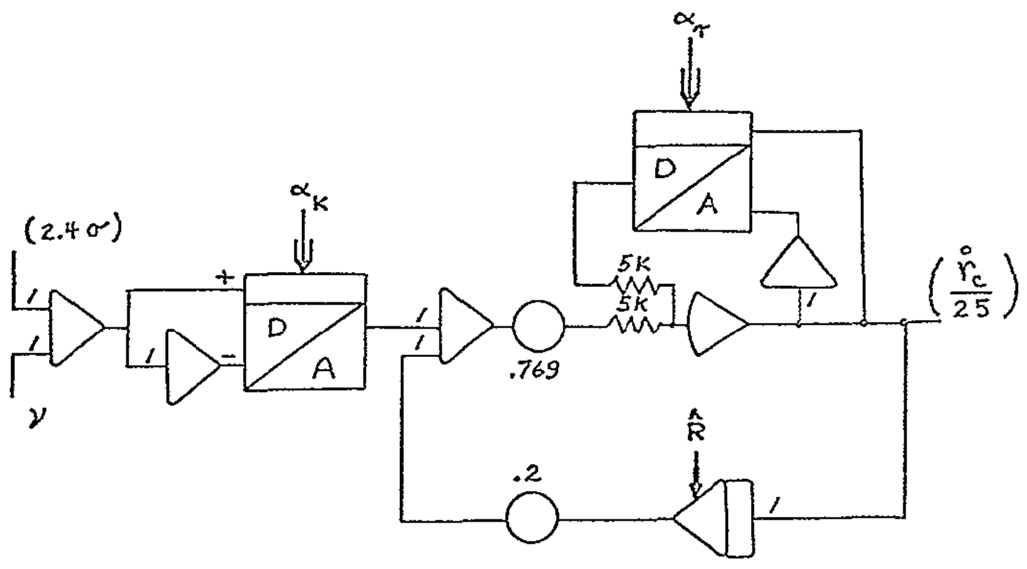
K and τ must be greater than zero for the system to be stable. Positive values of c_K and/or c_τ may be used to determine the best performance obtainable when production variations are allowed in K and/or τ.

<u>4.2 The Simulation</u>

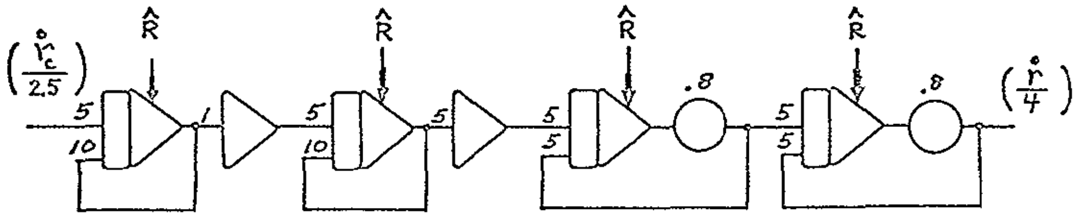
Figures 4.3 and 4.4 show the analog computer diagram and control logic for the simulation. The time scale is given by

$$t = \frac{1}{250 \times 10^{-6}} t'$$

where t is the problem time ($0 \leq t \leq t_f = 7.$ sec) and t' is the computer time ($0 \leq t' \leq 1.75$ msec). <u>This allows for</u>



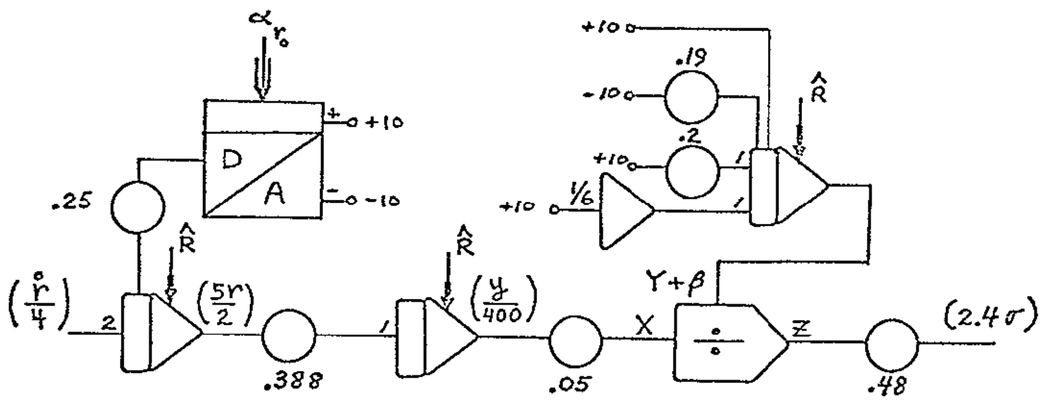
Guidance


$\left(\frac{\mathring{r}_c}{25}\right)$
$\hat{R}$
5
10
1
5
10
5
5
5
.8
5
5
.8
$\left(\frac{\mathring{r}}{4}\right)$

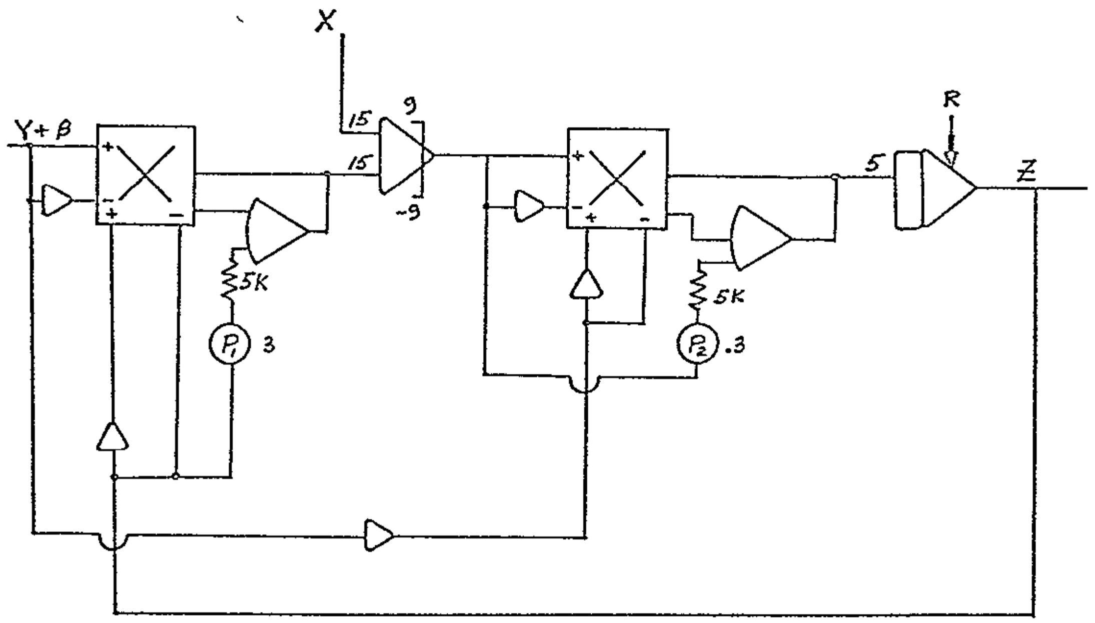
Aerodynamics

Fig. 4.3 Analog computer diagram of the radar-homing missile simulation.

3

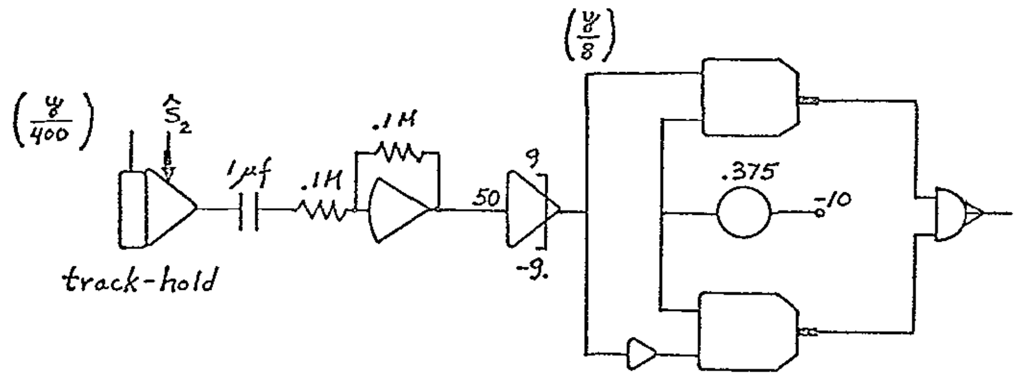

α_{r_0}
D
A
+10
-10
.25
$\hat{R}$
$\left(\frac{\dot{r}}{4}\right)$
2
$\left(\frac{5r}{2}\right)$
.388
1
$\left(\frac{y}{400}\right)$
.05
X
$Y+\beta$
Z
.48
(2.4σ)
+10
-10
.19
.2
1/6
1

Kinematics

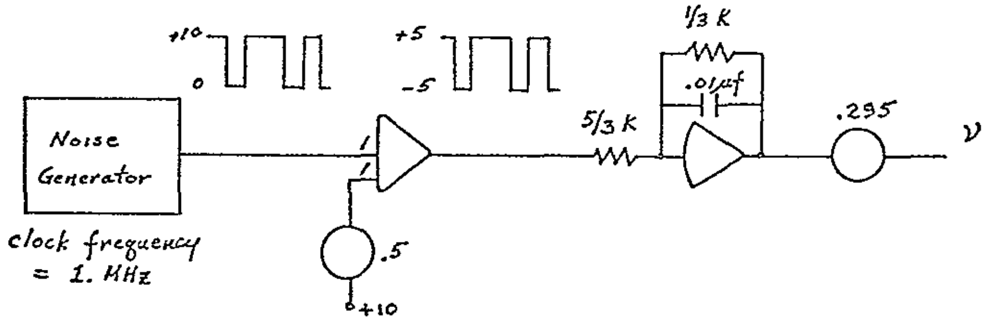


Steepest-descent division circuit

Fig. 4.3--Continued Analog computer diagram of the radar-homing missile simulation.



Hit-miss decision circuit



Radar tracking-noise generation

Fig. 4 3--Continued Analog computer diagram of the radar-homing missile simulation.

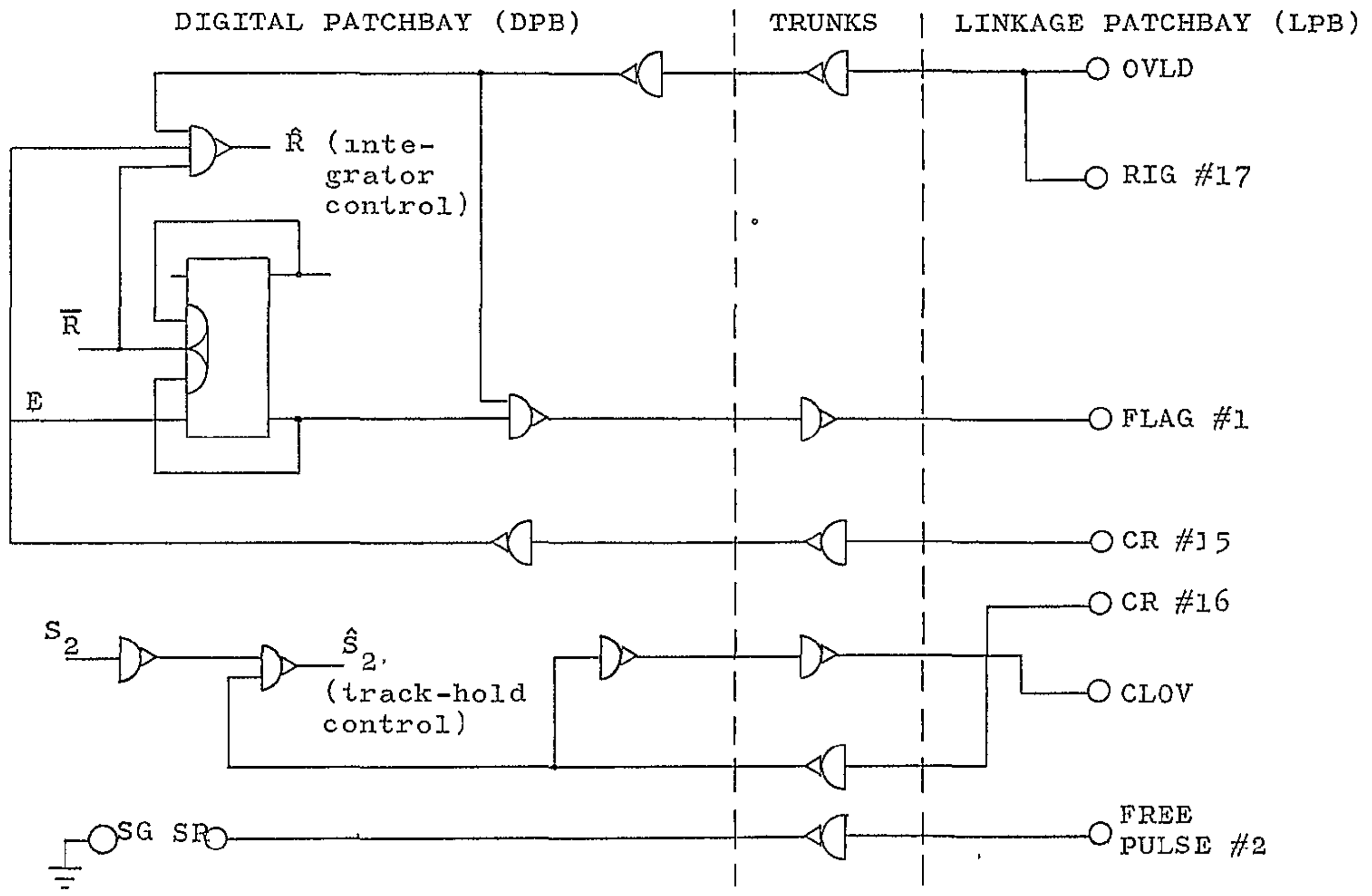


Fig. 4.4 Control logic for the simulation.

solution rates of 500 runs per second. The digital inputs, α_K and α_τ, to the multiplying D/A converters provide the following ranges for K and τ.

$$0 \leq K \leq 6.$$

$$0 \leq \tau \leq 1.3 \text{ sec}$$

The awkward division by $v_c T$, which approaches zero together with the numerator $y(t)$ as t approaches t_f, is implemented by a very fast steepest-descent circuit (Maybach, 1966b). Potentiometers p_1 and p_2 compensate for the fact that the actual divisor is $v_c T + \beta$, where $\beta = 3.$ volts. This constant is added to maintain a reasonably large input to the quarter-square multipliers, which are less accurate for small inputs.

A missile-firing simulation is begun with a random initial condition r_o. At $t = t_f$, the track-hold circuit holds $y(t_f)$, which is compared to $\pm$ d by the two comparators. The 1 μf capacitor and the summing amplifier constitute a d.c. blocking circuit for filtering out drift voltages. The comparator outputs are gated and applied to a read-in gate on the analog-digital interface for hit-miss detection by the digital computer. The integrators are controlled by a logic signal $\hat{R}$ (Fig. 4.4). This is essentially the normal compute-reset signal (R) modified for automatic resetting at the occurrence of an overload or upon a command from PDP-9 by way of the control register. The track-hold logic signal $\hat{S}_2$ is S_2 augmented by a provision for specifying the track mode with PDP-9 at an overload condition and during idle periods. Simulations are initiated by Free Pulse #2. The end of a simulation is signaled by Flag 1, which is raised upon the occurrence of an overload or at the completion of the 1.75 msec COMPUTE period. If an analog computer overload occurs during a simulation, that simulation is regarded as a miss.

Usually overloads occur for parameter values and/or an initial condition which would result in a miss. It is possible, however, for an overload to occur even during a simulation which would result in a hit; in this case, assigning a miss is erroneous. If ℓ such errors are made in a hit-probability estimate of N simulations, the error in probability is $\Delta p = - 2\ell/N$. The optimization program allows three overloads per 1024 simulations before voiding the estimate of the hit probability. Thus, the worst-case error is given by $\Delta p = - .0059$.

4.3 The Optimization

The basic optimization strategy has been discussed in Section 3.2. A modification and some additional features are described here.

Since the criterion function for the example problem is a probability p, and separate runs are considered to be statistically independent, the variance of an estimate of p is known a priori. Let our estimate of p be given by

$$\bar{f} = \frac{1}{N} \sum_{i=1}^{N} {}^{i}f \qquad (4.4)$$

where

$${}^{i}f = \begin{cases} 1 & \text{if } \left| y(t_f) \right| \leq 30.\ \text{ft.} \\ 0 & \text{otherwise} \end{cases}$$

$\bar{f}$ has a binomial distribution with mean p and variance p(1-p)/N. For Np and N(1-p) both at least 5, the distribution may be approximated reasonably as Gaussian (Hahn and Shapiro, 1967). Then, we can make the following probability statement concerning our estimate of p:

$$P\left[\left| \bar{f} - p \right| \leq \sqrt{\frac{p(1-p)}{N}}\, z_{\alpha/2}\right] = 1 - \alpha,$$

where

$$\int_{z_{\alpha/2}}^{\infty} \phi(z)\,dz = \alpha/2$$

and $\phi(z)$ is the standardized Gaussian density function (zero mean, unit variance). Table 4.1 lists values of the confidence-interval half-width as a function of p and N for $\alpha = 0.05$. In the optimization program for the example problem (Fig. 4.5), the variance of our estimate of p is controlled by adjusting N. Otherwise, the strategy is the same as discussed in Section 3.2. In order to estimate the

Table 4.1 Confidence-interval half-widths, $\sqrt{\frac{p(1-p)}{N}}\, z_{\alpha/2}$, for $\alpha = .05$

	Hit probability, p		
Number of simulations, N	0.1	0.25	0.5
128	.0520	.0750	.0866
256	.0367	.0530	.0613
512	.0260	.0375	.0433
1024	.0184	.0255	.0306
2048	.0130	.0187	.0216
4096	.00919	.0133	.0153
8192	.00649	.00915	.0108

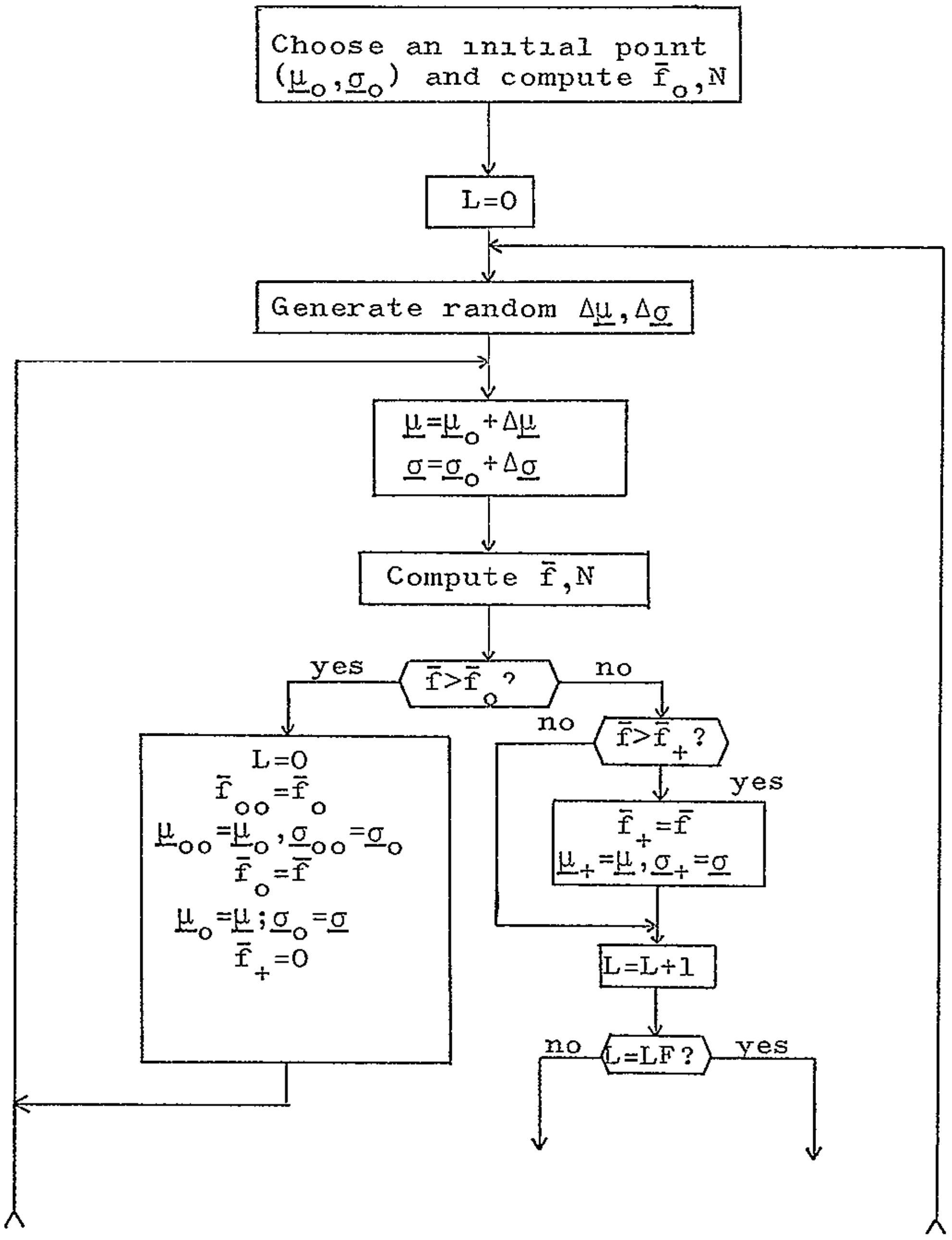
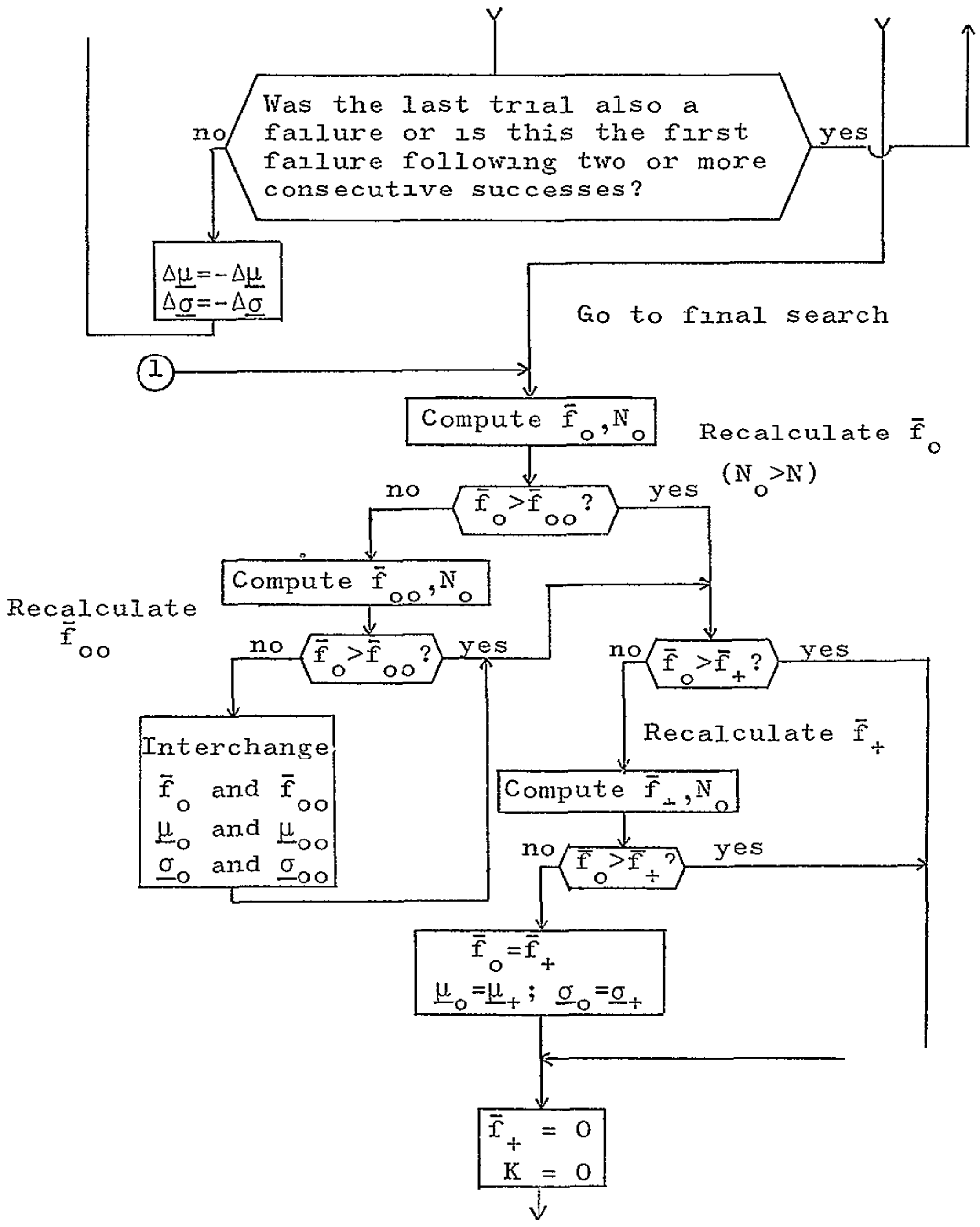
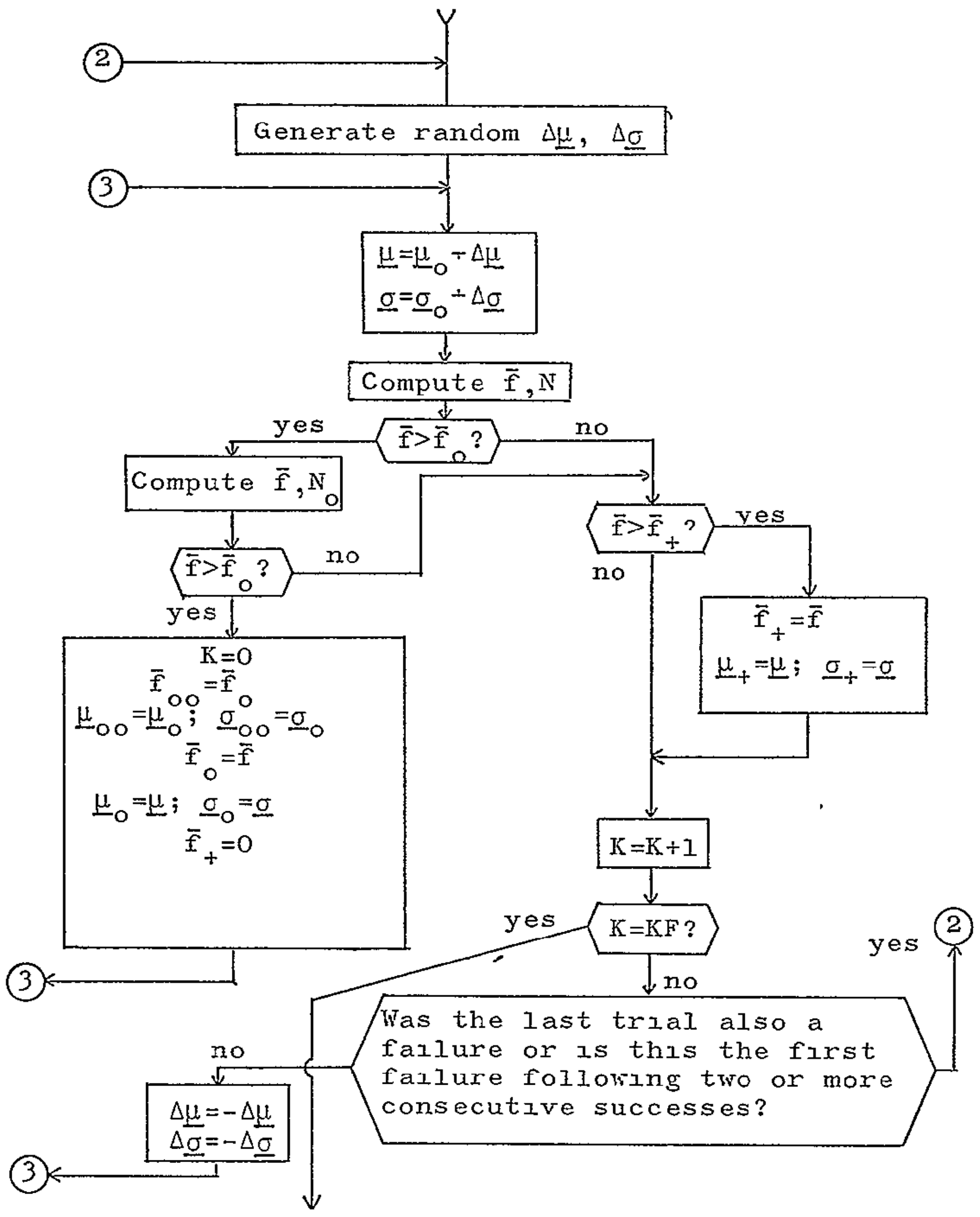

Choose an initial point ($\underline{\mu}_o$,$\underline{\sigma}_o$) and compute $\bar{f}_o$,N
L=0
Generate random $\Delta\underline{\mu}$,$\Delta\underline{\sigma}$
$\underline{\mu}=\underline{\mu}_o+\Delta\underline{\mu}$
$\underline{\sigma}=\underline{\sigma}_o+\Delta\underline{\sigma}$
Compute $\bar{f}$,N
$\bar{f}>\bar{f}_o$?
yes
no
L=0
$\bar{f}_{oo}=\bar{f}_o$
$\underline{\mu}_{oo}=\underline{\mu}_o$,$\underline{\sigma}_{oo}=\underline{\sigma}_o$
$\bar{f}_o=\bar{f}$
$\underline{\mu}_o=\underline{\mu}$;$\underline{\sigma}_o=\underline{\sigma}$
$\bar{f}_+=0$
$\bar{f}>\bar{f}_+$?
no
yes
$\bar{f}_+=\bar{f}$
$\underline{\mu}_+=\underline{\mu}$,$\underline{\sigma}_+=\underline{\sigma}$
L=L+1
L=LF?
no
yes

Fig. 4.5 A flow diagram for the optimization algorithm.



Was the last trial also a failure or is this the first failure following two or more consecutive successes?
no
yes
$\Delta\underline{\mu}=-\Delta\underline{\mu}$
$\Delta\underline{\sigma}=-\Delta\underline{\sigma}$
Go to final search
1
Compute $\bar{f}_o, N_o$
Recalculate $\bar{f}_o$ $(N_o>N)$
$\bar{f}_o>\bar{f}_{oo}$?
Compute $\bar{f}_{oo}, N_o$
Recalculate $\bar{f}_{oo}$
Interchange $\bar{f}_o$ and $\bar{f}_{oo}$ $\underline{\mu}_o$ and $\underline{\mu}_{oo}$ $\underline{\sigma}_o$ and $\underline{\sigma}_{oo}$
$\bar{f}_o>\bar{f}_+$?
Recalculate $\bar{f}_+$
Compute $\bar{f}_+, N_o$
$\bar{f}_o=\bar{f}_+$
$\underline{\mu}_o=\underline{\mu}_+$; $\underline{\sigma}_o=\underline{\sigma}_+$
$\bar{f}_+ = 0$
$K = 0$

Fig. 4.5--Continued A flow diagram for the optimization algorithm.



2
Generate random $\Delta\underline{\mu}$, $\Delta\underline{\sigma}$
3
$\underline{\mu}=\underline{\mu}_o+\Delta\underline{\mu}$
$\underline{\sigma}=\underline{\sigma}_o+\Delta\underline{\sigma}$
Compute $\bar{f}$, N
$\bar{f}>\bar{f}_o$?
yes
no
Compute $\bar{f}$, N_o
$\bar{f}>\bar{f}_o$?
no
yes
K=0
$\bar{f}_{oo}=\bar{f}_o$
$\underline{\mu}_{oo}=\underline{\mu}_o$; $\underline{\sigma}_{oo}=\underline{\sigma}_o$
$\bar{f}_o=\bar{f}$
$\underline{\mu}_o=\underline{\mu}$; $\underline{\sigma}_o=\underline{\sigma}$
$\bar{f}_+=0$
3
$\bar{f}>\bar{f}_+$?
yes
no
$\bar{f}_+=\bar{f}$
$\underline{\mu}_+=\underline{\mu}$; $\underline{\sigma}_+=\underline{\sigma}$
K=K+1
K=KF?
yes
no
Was the last trial also a failure or is this the first failure following two or more consecutive successes?
yes
2
no
$\Delta\underline{\mu}=-\Delta\underline{\mu}$
$\Delta\underline{\sigma}=-\Delta\underline{\sigma}$
3

Fig. 4.5--Continued A flow diagram for the optimization algorithm.

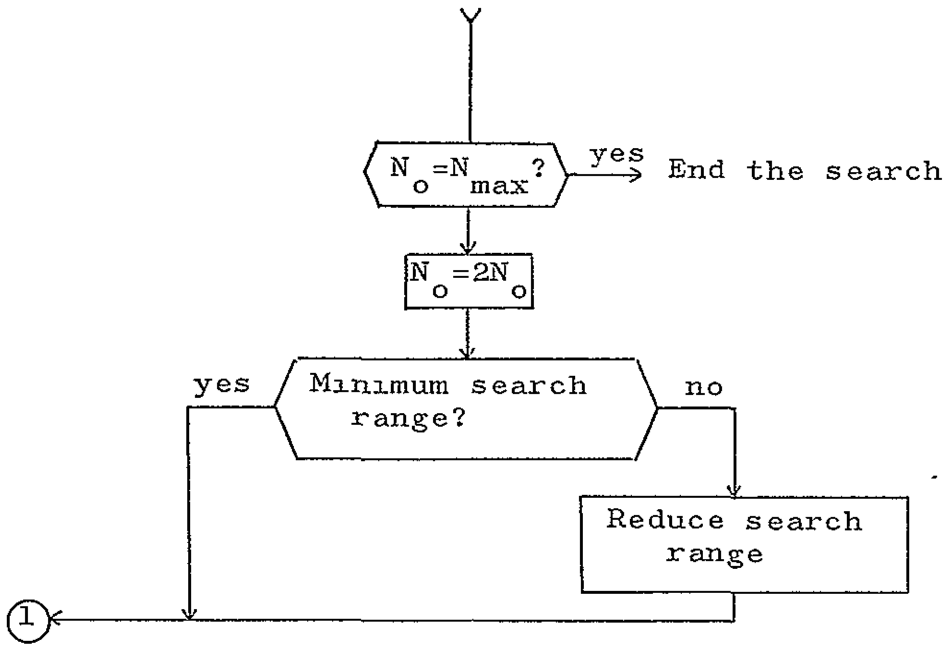


Fig. 4.5--Continued A flow diagram for the optimization algorithm.

hit probability for the final optimal parameters within approximately $\pm$ .01 for the worst case of $p_o = 0.5$, the maximum number of simulations per estimate (N_{max} in Fig. 4.5) was chosen as 8192.

In order to find a reasonable starting point for the creeping random algorithm, an initial pure-random search is provided. The criterion function is estimated at some specified number of points chosen from a distribution which is uniform over the entire parameter space. The point with the largest estimate of the function is returned for use as a starting point for the creeping random search. Alternatively, the operator may specify any starting point himself.

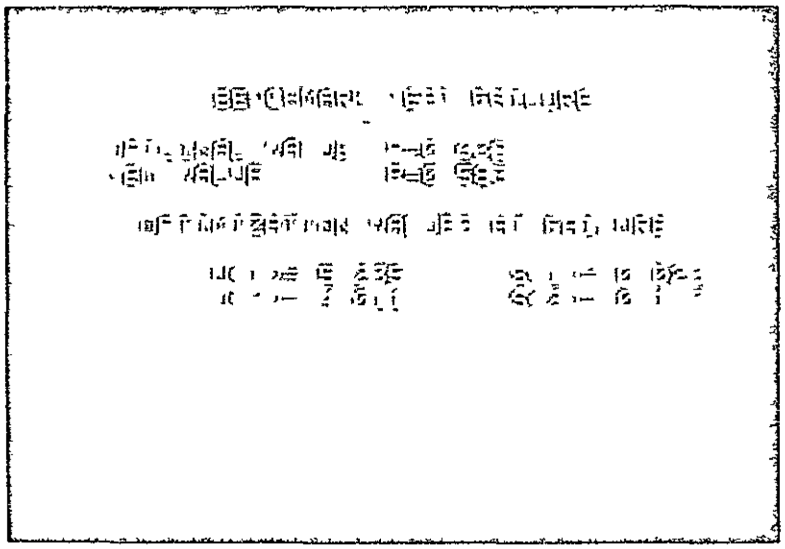
An optimization study involving searches from several starting points, each requiring five to ten minutes, may take an hour or more of computing time in spite of the fast analog computations. In this case, up to two million computer runs could be made. For this reason, malfunction or drift of an analog computer component should be detected before a large amount of spurious data is collected. For this purpose a "benchmark test" is included in the optimization program. Upon loading the program and beginning an optimization, the criterion function is measured at a point $(\underline{\mu}_B, \underline{\sigma}_B)$. During subsequent optimizations, the program periodically returns to the same point and reevaluates the criterion function. If an estimate $\overline{f}(\underline{\mu}_B, \underline{\sigma}_B)$ differs from the original measurement by an amount which causes the rejection of the hypothesis that the criterion function is unchanged, the operator is notified by a message on a cathode-ray-tube display console (CRT), as shown in Fig. 4.6. For the benchmark tests, 8192 simulations are used to estimate the criterion function. Let $\overline{f}_1$ be the estimate of the hit probability p_1 at the initial benchmark evaluation, and let $\overline{f}_2$ and p_2 be the estimate and the hit probability at some later test. We want to test the hypothesis H_o: $p_1 = p_2 = p$. $\overline{f}_1$ and $\overline{f}_2$ are approximately Gaussian with mean p_i and variance $p_i(1-p_i)/n$, for $i = 1,2$ and $n = 8192$. Under the hypothesis H_o, the distribution of $\overline{f}_1 - \overline{f}_2$ is approximately Gaussian with zero mean and variance $2p(1-p)/n$, and the following probability statement applies:

$$P[\,|\overline{f}_1 - \overline{f}_2| \le \sqrt{\frac{2p(1-p)}{n}}\, z_{\alpha/2}] = 1 - \alpha. \tag{4.5}$$

Since p is unknown, the variance $2p(1-p)/n$ is replaced by the sample variance.

$$P[\,|\overline{f}_1 - \overline{f}_2| \le \sqrt{\frac{\overline{f}_1(1-\overline{f}_1) + \overline{f}_2(1-\overline{f}_2)}{n}}\, z_{\alpha/2}] = 1 - \alpha. \tag{4.6}$$

(The new statistic has a Student-t distribution, but is approximately Gaussian for large n.) Equation (4.6) is used to test the hypothesis H_o at the 0.95 level of significance.



```
BENCHMARK TEST FAILURE

[illegible]
[illegible]

[illegible] FAILURE

[illegible]
[illegible]
```

Fig. 4.6 CRT output for a benchmark test failure.

4.4 Operation of the Optimization Program

This section briefly describes the procedure for performing an optimization and the facility for operator-program interaction.

The differential equations for the simulation are patched on ASTRAC-II's analog and digital patchbays. ASTRAC-II is placed in the SINGLE RUN mode, which allows for initiation of compute periods on command from the PDP-9 by way of the linkage patchbay.

After the digital program is loaded from magnetic tape into core memory, the program enters a "command mode," and the following index is displayed on the CRT·

1. Read input data
2. Display input data
3. Begin optimization

The operator can select the desired mode of operation by typing the corresponding index number on the CRT keyboard. Typing a "1" results in a display of an index to the program variables which must be assigned values by the operator:

1. M, the number of system parameters.
2. MODE, a number specifying one of three operating modes: 0--a single evaluation of the criterion function for specified parameter values; 1--the creeping-random-search algorithm; 2--the uniform-random search.

3. NSHIFT, a number specifying the initial search-range for the creeping-random search.

4. MAXS, a number specifying the minimum search-range for the creeping-random search.

5. NRAN, the number of criterion function evaluations for the uniform-random search.

6. LF, the number of consecutive failures allowed in the initial search (Fig. 4.5).

7. KF, the number of consecutive failures allowed in the final search (Fig. 4.5).

8. N, the number of ASTRAC-II runs per function evaluation for trial steps.

9. MAXN, the maximum number of runs per function evaluation in the final search (Fig. 4.5).

10. NPRINT, the number of trial steps between CRT printouts of the progress of the optimization.

11. PMIN(I), PMAX(I), the minimum and maximum allowable values for the system parameters (a_i and b_i in Eq. [3.5]).

    SLIM(I), the lower bound on the percentage standard deviations of the parameters (c_i in Eq. [3.6]).

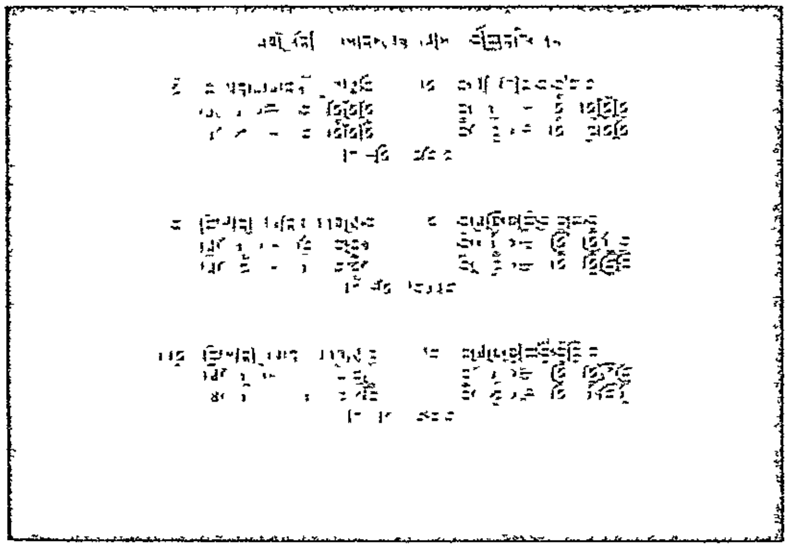
12. U(I), S(I), initial values of μ_i and σ_i.

13. UB(I), SB(I), values for the "benchmark" parameters.

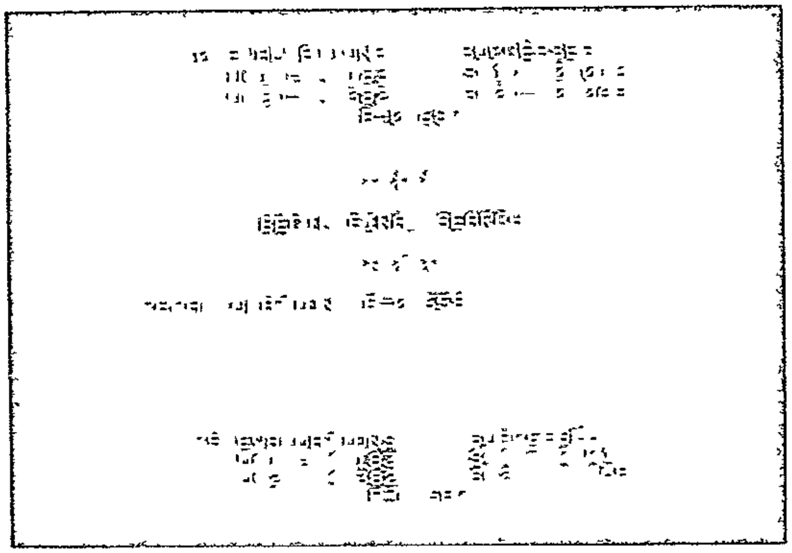
Displayed on the CRT screen below the index is a request for the operator to type the number corresponding to the input variable he wishes to enter. When the number is typed, the screen is cleared, and the input variable name followed by an "equal" sign is displayed. The operator then types in the value for the input variable. When the value is read by the computer, the input data index is displayed again. After the input data have been entered, the operator may return to the command mode by typing a special-code S ($\uparrow$S). For verification of the input data, the operator can type a "2" while in command mode to obtain a CRT display of the data. Typing a "3" in the command mode initiates an optimization according to the specified value of MODE (No. 2 above).

As the optimization proceeds, the CRT displays the number of steps taken, the number of these steps resulting in an improvement of the criterion function, and the parameter values and criterion function value at the current optimal point (Fig. 4.7). A summary of the optimization is displayed upon completion (Fig. 4.7).

The operator can affect the course of the optimization by communicating with the algorithm through accumulator switches. <u>While the search proceeds</u>, he can control the search range, hold any parameters constant while the program continues to optimize with respect to the other parameters, suppress the failure counters (K or L) in order to remain in one part of the search, request any CRT output duplicated in hard copy by a Teletype, or request a termination of the search. <u>This kind of algorithm-operator</u>

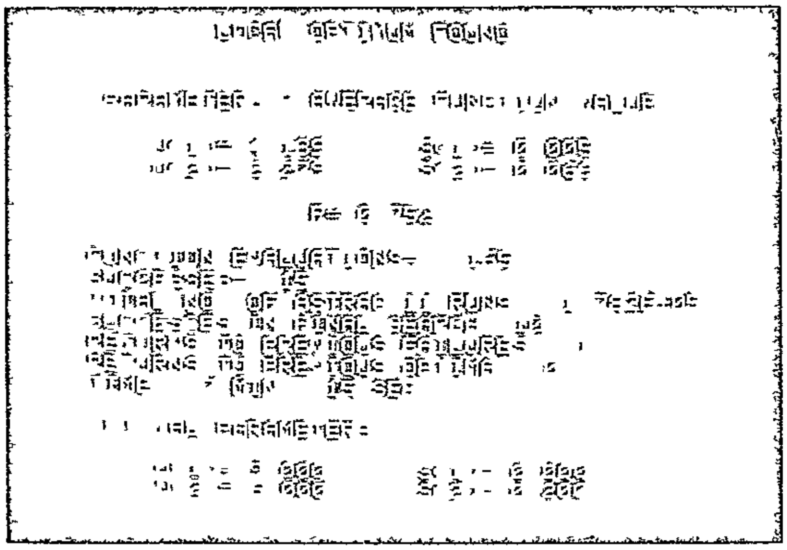


a. Initial search.



b. Beginning of the final search.

Fig. 4.7 CRT displays during the optimization.



[illegible]

c. End of the search.

Fig. 4.7--Continued

interaction can provide the engineer with insight into the behavior of the system and might enable him to speed the search for the optimum.

In the interest of execution speed, the programs for estimating the criterion function and for the optimization were written in MACRO-9, the PDP-9 assembly language. Input-output routines were programmed in FORTRAN. The program-interrupt facility enables efficient use of computing time by allowing the digital computer to perform computations during ASTRAC-II's compute period. While one simulation is under way, the PDP-9 averages the results of the previous simulation and selects the random parameter values and initial conditions for the next simulation.

4.5 Experiments and Results

Contours of constant hit probability are shown in Figs. 4.8 and 4.9. Results are expressed in terms of scaled parameter values, $K' = K/6.$ and $\tau' = \tau/1.3$, which are in the range (0,1). In Fig. 4.8, contours are plotted as a function of the scaled parameter mean values for $\sigma_{K'} = \sigma_{\tau'} = 0$. The maximum hit probability, $p_o \approx .750$, occurs at approximately $(\mu_{K'}, \mu_{\tau'}, \sigma_{K'}, \sigma_{\tau'}) = (0.42, 0.22, 0.0, 0.0)$. In Fig. 4.9, the contours are plotted against the dispersions $\sigma_{K'}$ and $\sigma_{\tau'}$ for $\mu_{K'} = 0.42$ and $\mu_{\tau'} = 0.22$. For the optimal parameter values, Fig. 4.10 shows sample trajectories with and without the radar-tracking noise.

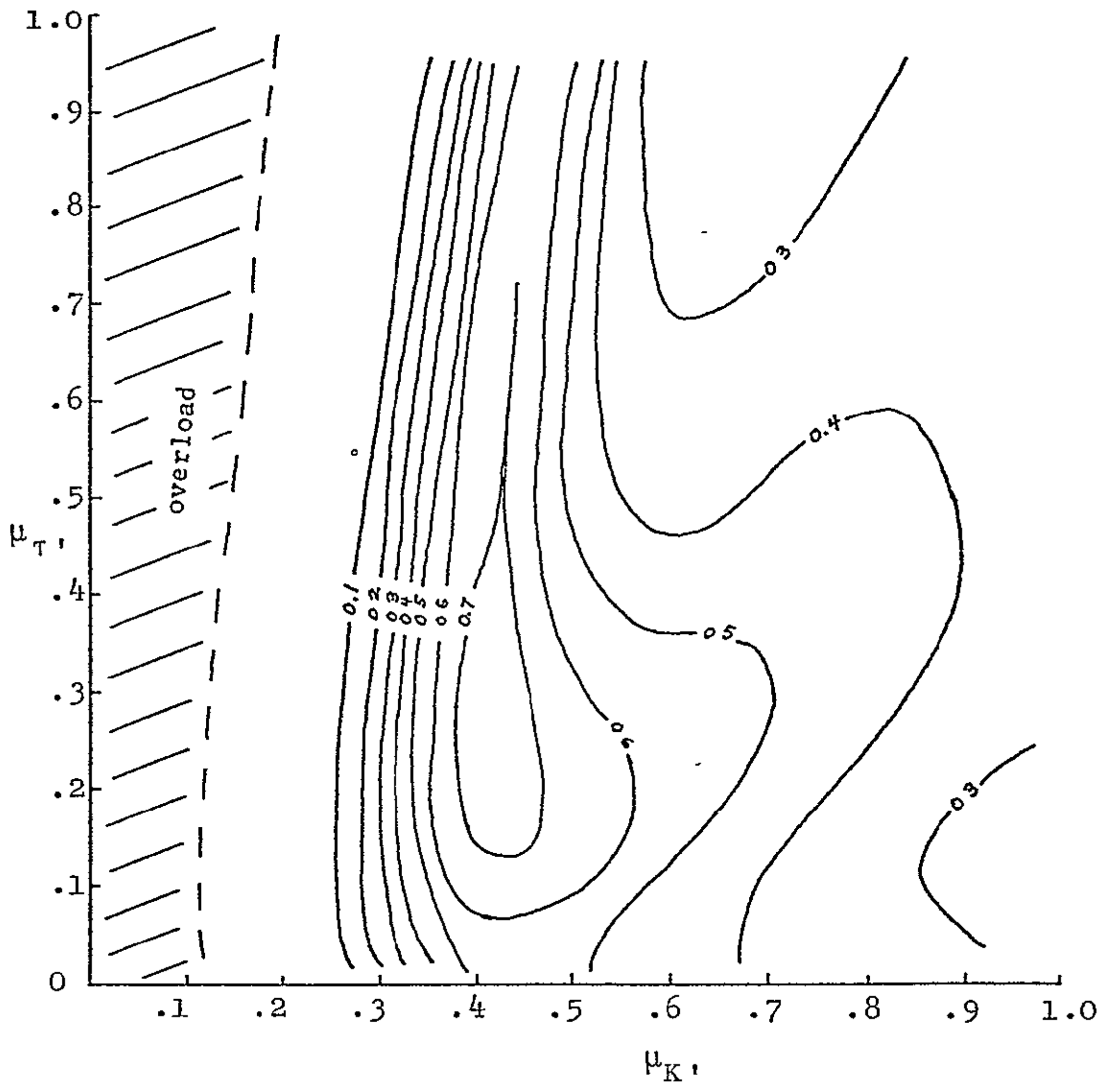


Fig. 4.8 Contours of constant hit probability as a function of $\mu_{K'}$ and $\mu_{T'}$ (mean values of unknown parameters).

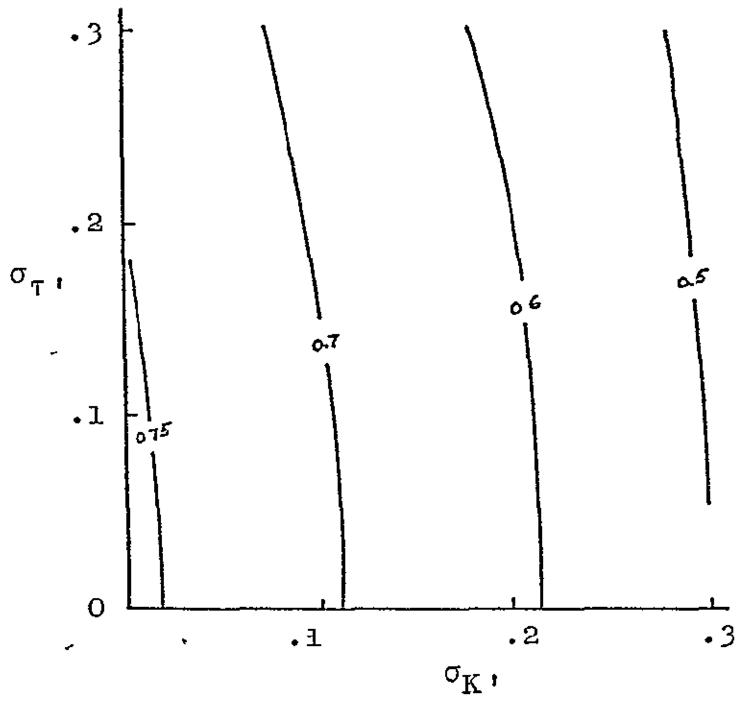
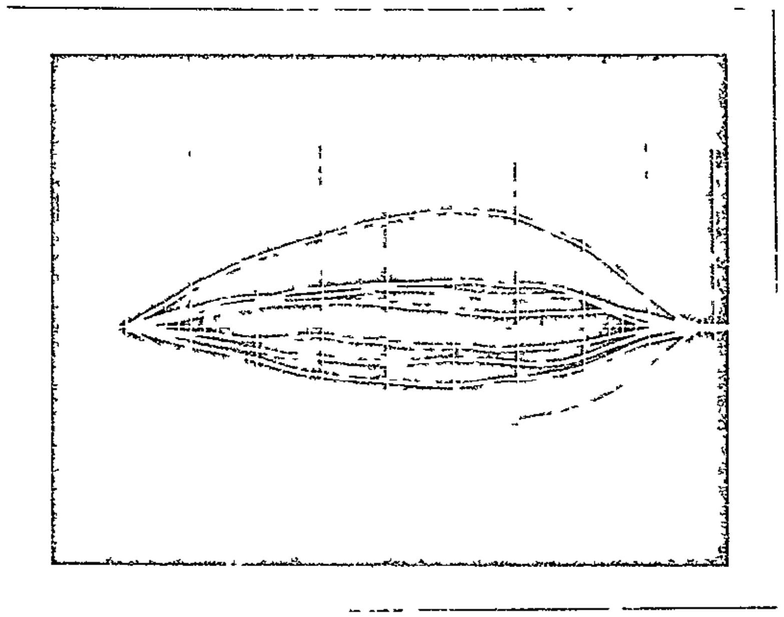
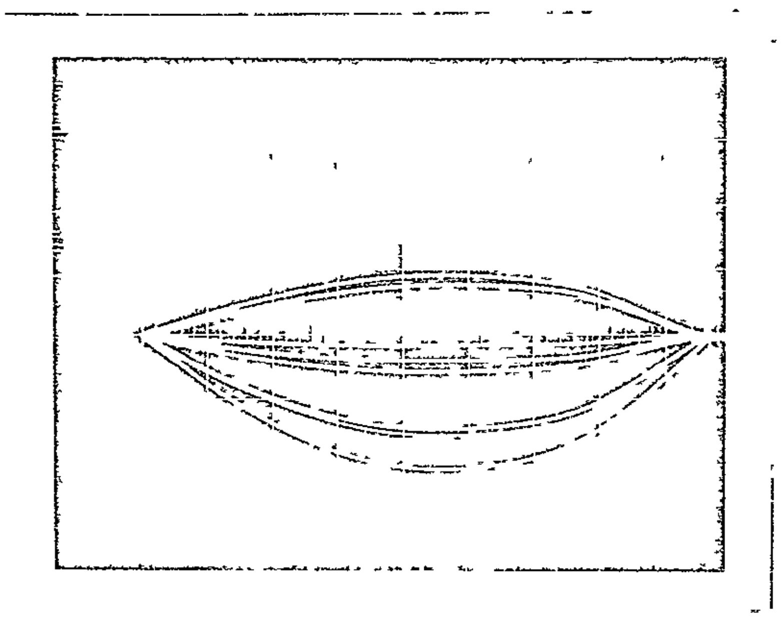


Fig. 4.9 Contours of constant hit probability as a function of $\sigma_{K'}$ and $\sigma_{T'}$ (dispersions of unknown parameters).



a. Trajectories with noise.



b. Trajectories without noise.

Vertical: missile position normal to line of sight, y/400, 1 volt/cm; horizontal: problem time, t', 0.2 msec/cm.

Fig. 4.10 The effect of radar-tracking noise on the missile trajectories.

In order to study the effectiveness of the optimization strategy, searches were begun from pre-selected starting points as well as from points chosen by the pure-random search. The results of these searches are summarized in Tables 4.2-4.5. With $N_{max} = 8192$, a 95% confidence-interval half-width for the hit-probability estimate is approximately $\pm$ .01. Thus, for optimizations without constraints on σ_K or σ_T, searches yielding an "optimal" point with $\overline{f} < .74$ are considered failures. An asterisk precedes the data for these searches.

Searches were begun from the point $(\mu_{K'},\mu_{T'},\sigma_{K'},\sigma_{T'}) = (0.9,0.6,0.0,0.2)$ with LF = 20 and KF = 40 in order to study the behavior of the algorithm as a function of N, the number of simulations used to estimate p at trial points (Table 4.2).

This starting point is in a region where the gradient of the criterion function is small; noise in the estimates of the hit probability can easily obscure the gradient. Note that for the successful searches, the ranges of the final values of $\mu_{T'}$ and $\sigma_{T'}$ are much larger than the ranges of $\mu_{K'}$ and $\sigma_{K'}$. This behavior is to be expected from the shape of the contours in Figs. 4.8 and 4.9. In general, as N decreases, so do the average number of simulations and the computer time per optimization while the number of unsuccessful searches increases. An exception is the case of N = 64, where the average number of

Table 4.2 Data for automatic optimizations from the starting point $(\mu_{K'},\mu_{T'},\sigma_{K'},\sigma_{T'}) = (0.9,0.6,0.0,0.2)$.

LF = 20 KF = 40 N_o = 4096 N_{max} = 8192

N	$\bar{f}_o$	$\mu_{K'}$	$\mu_{T'}$	$\sigma_{K'}$	$\sigma_{T'}$	Trial steps	ASTRAC-II runs
64	.741	.450	.169	.005	.067	174	234000
	.750	.421	.253	.028	.128	187	177000
	.744	.406	.268	.015	.172	174	213000
	*.715	.429	.371	.024	.038	146	133000
	*.739	.438	.217	.010	.227	151	248000
	*.706	.431	.393	.023	.234	117	119000
	*.716	.412	.367	.011	.148	161	237000
	.742	.421	.265	.014	.121	208	375000
					Average:	165	217000

Average time = 7 min 23 sec

N	$\bar{f}_o$	$\mu_{K'}$	$\mu_{T'}$	$\sigma_{K'}$	$\sigma_{T'}$	Trial steps	ASTRAC-II runs
128	.748	.407	.212	.014	.010	240	267000
	*.707	.428	.440	.006	.120	157	118000
	.751	.427	.259	.020	.111	132	136000
	*.672	.457	.784	.010	.073	115	117000
	*.738	.427	.247	.029	.130	148	165000
	*.733	.424	.300	.017	.066	127	135000
	.754	.422	.203	.011	.113	174	120000
	*.688	.436	.513	.026	.193	152	105000
					Average:	156	145000

Average time = 5 min 0 sec

N	$\bar{f}_o$	$\mu_{K'}$	$\mu_{T'}$	$\sigma_{K'}$	$\sigma_{T'}$	Trial steps	ASTRAC-II runs
256	.752	.425	.239	.019	.054	167	189000,
	.747	.431	.269	.003	.020	173	211000
	*.721	.424	.393	.024	.034	149	176000
	.740	.420	.277	.035	.141	140	109000
	*.719	.429	.367	.029	.220	119	117000
	*.705	.419	.402	.018	.151	142	114000
	.756	.422	.194	.005	.056	229	345000
	.748	.413	.226	.040	.109	195	497000
					Average:	140	198000

Average time = 6 min 36 sec

Table 4.2--Continued

512	.748	.416	.208	.039	.077	210	216000
	.749	.411	.245	.015	.145	162	212000
	*.695	.424	.504	.004	.061	120	146000
	*.720	.413	.317	.009	.296	141	182000
	.755	.435	.209	.021	.152	222	229000
	.758	.421	.214	.009	.043	202	220000
	.754	.422	.254	.026	.121	154	183000
	.744	.437	.214	.046	.020	159	165000
					Average:	171	185000
Average time = 6 min 20 sec							
1024	.745	.425	.286	.028	.052	111	152000
	.747	.435	.199	.017	.112	137	186000
	*.739	.421	.282	.028	.195	125	201000
	*.732	.422	.309	.028	.169	169	304000
	.745	.426	.307	.007	.176	226	393000
	.758	.414	.245	.010	.034	238	359000
	.740	.412	.189	.009	.198	125	200000
	.743	.421	.323	.020	.166	169	230000
					Average:	162	246000
Average time = 8 min 23 sec							
2048	.745	.419	.305	.003	.223	257	650000
	.761	.422	.267	.002	.065	133	331000
	*.730	.424	.363	.005	.298	112	237000
	.756	.420	.231	.002	.075	209	424000
					Average.	178	447000
Average time = 15 min 3 sec							

simulations and computer time increases. This is caused by the relatively large variance in the estimates of hit probability for trial steps in the final search. The "noisy" estimates lead to many false indications of improvements in the hit probability; each indication of an improvement is followed by a reevaluation requiring many simulations. From the data of Table 4.2, it was decided that the best compromise between performance of the algorithm and computer time occurred for N = 512. This value was used for the remainder of the study.

Table 4.3 shows results for searches begun from the point $(\mu_{K'},\mu_{T'},\sigma_{K'},\sigma_{T'}) = (0.5,0.9,0.3,0.0)$. This is a particularly difficult starting point, because here the search must climb a narrow ridge, which has steep sides and a very small slope in the direction of the optimum. In order to have the search reach the optimum, it was necessary to increase LF and KF, the number of consecutive failures allowed in the initial and final searches.

To illustrate a more practical method for locating the optimum, the algorithm was next started from the best point chosen from the pure-random search described above. Estimates of p based on 512 simulations were calculated for 45 random points. Data for the creeping random searches are listed in Table 4.4. Note that the two unsuccessful searches stopped at points on the ridge.

Table 4.3 Data for automatic optimizations from the starting point $(\mu_{K'},\mu_{\tau'},\sigma_{K'},\sigma_{\tau'}) = (0.5, 0.95, 0.3, 0.0)$.

$N = 512 \quad N_o = 4096 \quad N_{max} = 8192$

	$\bar{f}_o$	$\mu_{K'}$	$\mu_{\tau'}$	$\sigma_{K'}$	$\sigma_{\tau'}$	Trial steps	ASTRAC-II runs
LF=20 KF=40							
	*.694	.421	.519	.004	.107	133	117000
	*.698	.415	.497	.019	.030	140	161000
	*.681	.447	.706	.026	.059	129	131000
	*.681	.449	.675	.015	.046	139	155000
					Average:	135	147000
Average time = 5 min 4 sec							
LF=40 KF=40							
	.751	.419	.222	.010	.173	215	249000
	*.720	.430	.434	.005	.243	144	142000
	.753	.409	.245	.022	.016	187	208000
	*.725	.434	.374	.001	.215	204	171000
					Average:	187	197000
Average time = 6 min 35 sec							
LF=60 KF=30							
	*.739	.419	.255	.025	.268	187	224000
	.764	.427	.209	.005	.005	294	287000
	*.684	.460	.752	.003	.043	202	205000
	.762	.415	.239	.013	.044	160	140000
	*.705	.424	.511	.025	.004	265	238000
					Average:	222	219000
Average time = 7 min 28 sec							

Table 4.4 Data for automatic optimizations from starting points chosen by the pure-random search.

LF=20 KF=40 N=512 N_o=4096 N_{max}=8192

$\bar{f}_o$	$\mu_{K'}$	$\mu_{T'}$	$\sigma_{K'}$	$\sigma_{T'}$	Trial steps	ASTRAC-II runs
.748	.420	.223	.048	.003	121	143000
*.721	.424	.423	.014	.165	112	129000
.754	.419	.205	.001	.048	123	175000
.747	.432	.201	.019	.143	111	160000
.747	.416	.187	.002	.022	154	165000
.756	.422	.223	.012	.138	213	266000
*.723	.425	.381	.001	.258	147	163000
.753	.424	.228	.014	.106	181	255000
				Average:	143	172000

Average time = 5 min 55 sec

A pure-random search followed by the creeping-random-search algorithm was applied to optimizations with lower-bound constraints on $\sigma_{K'}$ and $\sigma_{T'}$ (Table 4.5). This is intended to model a situation where it is known that holding production tolerances below a certain level is very difficult and/or costly. Note that a lower bound on $\sigma_{K'}$ causes an increase in the optimal mean value $\mu_{K'}$. For the case of $\sigma_{K'} \geq .2$ and $\sigma_{T'} \geq .2$, the maximum hit probability appears to be about 0.625. This should be compared with a value of 0.600 for the point $(\mu_{K'},\mu_{T'},\sigma_{K'},\sigma_{T'}) = (0.42,0.22,0.2,0.2)$ in Fig. 4.9.

Optimizations were performed with several other combinations of lower-bound constraints on $\sigma_{K'}$ and $\sigma_{T'}$ as well as with equality constraints on $\sigma_{K'}$ and $\sigma_{T'}$. In no case, however, was it observed that increasing $\sigma_{K'}$ or $\sigma_{T'}$ resulted in an increase in the hit probability. It is believed that McGhee and Levine's observation of the hit probability increasing as $\sigma_{K'}$ is increased is a result of holding $\mu_{K'}$ and $\mu_{T'}$ constant, instead of locating new optimal values.

From the results for this example problem, it could be concluded that production variations in the gain K will have a significant effect on the hit probability, while large variations in τ degrade the performance only slightly. Also, for lower bounds on $\sigma_{K'}$ and $\sigma_{T'}$ the mean value $\mu_{K'}$ must be increased to obtain optimal performance. Note that

Table 4.5 Data for automatic optimizations with lower bound constraints on $\sigma_{K'}$ and $\sigma_{T'}$.

LF=20 KF=40 N=512 N_o=4096 N_{max}=8192

	$\bar{f}_o$	$\mu_{K'}$	$\mu_{T'}$	$\sigma_{K'}$	$\sigma_{T'}$	Trial steps	ASTRAC-II runs
$\sigma_{K'} \geq 0.1$							
	.700	.441	.268	.113	.243	120	164000
	.708	.473	.172	.101	.004	150	232000
	.707	.432	.282	.101	.043	131	179000
	.706	.457	.187	.104	.094	148	198000
					Average:	137	193000

Average time = 6 min 26 sec

	$\bar{f}_o$	$\mu_{K'}$	$\mu_{T'}$	$\sigma_{K'}$	$\sigma_{T'}$	Trial steps	ASTRAC-II runs
$\sigma_{K'} \geq .2$ $\sigma_{T'} \geq .2$							
	.610	.458	.174	.217	.233	123	154000
	.613	.479	.182	.208	.297	115	176000
	.623	.461	.206	.206	.236	149	195000
	.618	.488	.228	.208	.248	149	191000
	.628	.487	.194	.203	.222	175	224000
	.624	.501	.221	.205	.200	125	198000
	.631	.451	.186	.205	.212	158	212000
	.631	.456	.187	.201	.251	199	215000
					Average:	149	191000

Average time = 6 min 32 sec

this latter effect is revealed by the simulation of relatively large random variations in K; it would not be predicted from a small perturbation analysis.

CHAPTER 5

CONCLUSIONS AND DISCUSSION

In spite of the very large number of system simulations required for the optimization of the example problem, it is believed that this hybrid-computer approach to the optimization of systems with random parameters is a feasible one if a fast digitally controlled analog computer is available. Certainly, the large number of simulations demonstrates that an all-digital optimization of a dynamical system with random parameters by the Monte Carlo method would be impractical at this time.

It might be noted that <u>the type of criterion function optimized in the example (a probability) is one requiring a very large number of simulations in order to obtain a reasonable criterion-function estimate</u>. For example, if the hit probability is 0.5, our estimate is approximately Gaussian with mean 0.5 and standard deviation $1/2\sqrt{n}$, where n is the number of simulations used for the estimate of p. Then 100 simulations are required just to obtain an estimate with a standard deviation which is 10% of the mean. Criterion-function measurements for other types of problems may well have a more favorable signal-to-noise ratio.

With a computing speed of approximately 500 simulations per second, typical optimization times were on the order of 6-7 minutes for the 4-parameter example problem simulated on ASTRAC-II. For a commercially-available machine capable of about 200 simulations per second, a typical optimization time of about 16 minutes does not appear prohibitive. The results of Schumer and Steiglitz (1968) indicate that function evaluations (and computer time) should be expected to increase linearly as a function of the dimension of the parameter space.

The data presented in Chapter 4 were for completely automatic optimization in order to evaluate the effectiveness of the search algorithm. Operator-program interaction can save much computer time and provide more insight into the nature of the system. The automatic search is, however, the most important factor in devising an effective optimization system.

Parameter optimization in the presence of noise is surely an area requiring further research. The creeping-random-search algorithm described here is effective but wants improvement. The addition of a scheme for biasing the search in the direction of past successful steps should speed the progress along a ridge (Mitchell, 1964; Matyas, 1965, Rastrigin, 1967).

Two other approaches to "noisy" parameter optimization might be investigated. The digital computer is idle for much of the time during the integration of the differential equations on the analog machine. For the problem solved here, some of this time was used to generate parameter values for the next simulation. The opportunity for using this "idle time" would be increased with commercially-available hybrid computers, which have analog machines with slower computing speeds than ASTRAC-II and, typically, faster floating-point arithmetic than the PDP-9. During this time, the digital machine might make use of previous criterion-function measurements in order to fit a second-order regression surface to the criterion function, as briefly described in Section 2.4. If a measurement of the criterion function at the minimum point of the regression surface is an improvement over the current best point obtained by the creeping random search, the center of the search could be placed at the new point. Computing the regression surface and solving for the minimum point would, practically speaking, require floating-point computations.

Another possible approach to optimizing noisy criterion functions is to combine the conjugate-gradient algorithm of Powell (1964) or Zangwill (1967) with a stochastic-approximation method for the one-dimensional minimizations. It may not be necessary to locate the minima along the search directions with great accuracy; Harkins (1964) has noted that with the Partan method, convergence could be improved by inaccuracies in determining these minima.

APPENDIX A

THE EFFECTS OF LIMITING A GAUSSIAN RANDOM VARIABLE

Let the random variable X be Gaussian with mean μ and variance σ^2. The distribution function for X is given by

$$F(x) = \int_{-\infty}^{x} f(z)dz$$

$$= \int_{-\infty}^{x} \frac{1}{\sigma\sqrt{2\pi}} \exp[-(z-\mu)^2/2\sigma^2]dz$$

We "limit" the random variable X at $+r\sigma$ and at $\pm\, r\sigma (r>0)$ and show how the mean and variance of X are changed by these two limiting operations.

A.1 Limiting at $+r\sigma$

Without loss of generality, we can assume $\mu = 0$. Let the new random variable U be given by

$$U = \begin{cases} X & \text{for } X \leq r\sigma \\ r\sigma & \text{for } X > r\sigma \end{cases}$$

The distribution function for U is given by

$$G(u) = \begin{cases} F(u) & \text{for } u < r\sigma \\ 1 & \text{for } u \geq r\sigma \end{cases}$$

The expected value of U is

$$E\{U\} = \int_{-\infty}^{\infty} u dG(u)$$

$$= \int_{-\infty}^{r\sigma} uf(u)du + r\sigma \int_{r\sigma}^{\infty} f(u)du$$

$$= \frac{-\sigma \exp(-r^2/2)}{\sqrt{2\pi}} + r\sigma \int_{r\sigma}^{\infty} f(u)du \qquad (A.1)$$

The variance of U is

$$\mathrm{Var}\{U\} = E\{(U-E\{U\})^2\}$$

$$= E\{U^2\} - [E\{U\}]^2$$

$$= \int_{-\infty}^{r\sigma} u^2 f(u)du + (r\sigma)^2 \int_{r\sigma}^{\infty} f(u)du - [E\{U\}]^2 \qquad (A.2)$$

A.2 Limiting at $\pm r\sigma$

Again we assume $\mu = 0$. Let V be given by

$$V = \begin{cases} -r\sigma & \text{for } X < -r\sigma \\ X & \text{for } -r\sigma \leq X \leq r\sigma \\ r\sigma & \text{for } X > r\sigma \end{cases}$$

The distribution function for V is given by

$$H(v) = \begin{cases} 0 & \text{for } v < -r\sigma \\ F(v) & \text{for } -r\sigma \leq v < r\sigma \\ 1 & \text{for } v \geq r\sigma \end{cases}$$

Since the limiting operation is symmetric about the mean,

$$E\{V\} = E\{X\} \tag{A.3}$$

The variance of V is

$$\begin{aligned} \operatorname{Var}\{V\} &= E\{V^2\} - [E\{V\}]^2 \\ &= E\{V^2\} \\ &= (r\sigma)^2 \int_{-\infty}^{-r\sigma} f(v)dv + \int_{-r\sigma}^{r\sigma} v^2 f(v)dv + (r\sigma)^2 \int_{r\sigma}^{\infty} f(v)dv \\ &= \int_{-r\sigma}^{r\sigma} v^2 f(v)dv + 2(r\sigma)^2 \int_{r\sigma}^{\infty} f(v)dv \\ &= \sigma^2\{1 + 2[(r^2-1)\int_{r\sigma}^{\infty} f(v)dv - \frac{r\exp(-r^2/2)}{\sqrt{2\pi}}]\} \end{aligned} \tag{A.4}$$

Equations (A.1)-(A.4) give the means and variances of the limited random variables U and V as functions of r and σ. Table 3.1 lists the numerical values for r=1, 2, 3, and 4.

REFERENCES

Adams, R. J., and A. Y. Lew

1966 "Modified Sequential Random Search Using a Hybrid Computer," University of Southern California, Electrical Engineering Department Report, May, 1966.

Beckman, F. S.

1960 "The Solution of Linear Equations by the Conjugate Gradient Method," Mathematical Methods for Digital Computers, A. Ralston and H. F. Wilf (Eds.), Wiley, New York, 1960.

Bekey, G. A.

1964 "Optimization of Multiparameter Systems by Hybrid Computer Techniques," Simulation, vol. 3, no. 2 and no. 3, 1964.

Bekey, G. A., and W. J. Karplus

1968 Hybrid Computation, Wiley, New York, 1968.

Bekey, G. A., and R. B. McGhee

1964 "Gradient Methods for the Optimization of Dynamic System Parameters by Hybrid Computation," Computing Methods in Optimization Problems, A. V. Balakrishnan and L. W. Neustadt (Eds.), Academic, New York, 1964.

Bekey, G. A., M. H. Gran, A. E. Sabroff, and A. Wong

1966 "Parameter Optimization by Random Search Using Hybrid Computer Techniques," AFIPS Conference Proceedings, vol. 29, 1966.

Belt, J. E.

1969 "A Random Noise Generator for a Digital Computer," M.S. Thesis, Department of Electrical Engineering, University of Arizona, 1969.

Birta, L. G., and P. J. Trushel

1969 "A Comparative Study of Four Implementations of a Dynamic Optimization Scheme," Simulation, vol. 13, no. 2, 1969.

Blum, J. R.

1952 "Multidimensional Stochastic Approximation Methods," Annals of Mathematical Statistics, vol. 23, pp. 462-466, 1952.

Bohling, D., and J. Chernak

1965 "A Hybrid Computer Technique for Optimization," Simulation, vol. 5, no. 4, 1965.

Bohling, D., and L. A. O'Neill

1970 "An Interactive Approach to Tolerance Analysis," IEEE-TC, vol. C-19, no. 1, 1970.

Box, M. J.

1965 "A New Method of Constrained Optimization and a Comparison with Other Methods," The Computer Journal, vol. 8, no. 1, 1965.

1966 "A Comparison of Several Current Optimization Methods, and the Use of Transformations in Constrained Problems," The Computer Journal, vol. 9, no. 1, 1966.

Brooks, S. H.

1958 "A Discussion of Random Methods for Seeking Maxima," The Computer Journal, vol. 6, no. 2, 1958.

Brooks, S. H., and M. R. Mickey

1961 "Optimum Estimation of Gradient Direction in Steepest Ascent Experiments," Biometrics, vol. 17, no. 1, 1961.

Buehler, R. J., B. V. Shah, and O. Kempthorne

1964 "Methods of Parallel Tangents," Optimization Techniques, Chemical Engineering Progress Symposium Series, vol. 60, 1964.

Carlson, A. M.

1967 "A Partan Optimization Program," PCC Report, Princeton Computation Center, Electronics Associates, Inc., Princeton, New Jersey, August 18, 1967.

Carnahan, B.

1966 "Optimization Methods--A Review and Some Example Applications," Computers in Engineering Design Education, University of Michigan, March, 1966.

Carroll, C. W.

1961 "The Created Response Surface Technique for Optimizing Nonlinear Restrained Systems," Operations Research, vol. 9, no. 2, 1961.

Chang, S. S. L.

1961 Synthesis of Optimum Control Systems, McGraw-Hill, 1961.

Davidon, W. C.

1959 "Variable Metric Method for Minimization," AEC Research and Development Report Anl-5990, December, 1959.

Deardorff, J. E., and C. R. Trimble

1968 "Calibrated Real-Time Signal Averaging," Hewlett-Packard Journal, April, 1968.

Dvoretsky, A.

1956 "On Stochastic Approximation," Proceedings Third Berkeley Symposium on Mathematical Statistics and Probability, J. Neyman (Ed.), University of California Press, 1956.

Favreau, R. R., and R. G. Franks

1958 "Statistical Optimization," Proceedings Second International Analog Computer Conference, 1958.

Fiacco, A. V., and G. P. McCormick

1964 "Computational Algorithm for the Sequential Unconstrained Minimization Technique for Nonlinear Programming," Management Science, vol. 10, pp. 360-366, 1964.

1968 Nonlinear Programming: Sequential Unconstrained Minimization Techniques, Wiley, New York, 1968.

Fleischer, P. E.

1966 "Optimization Techniques," Chapter 6 in System Analysis by Digital Computer, F. F. Kuo and J. F. Kaiser (Eds.), Wiley, New York, 1966.

Fletcher, R.

1965 "Function Minimization without Derivatives--A Review," The Computer Journal, vol. 8, pp. 33-41, 1965.

Fletcher, R., and M. J. D. Powell

1963 "A Rapidly Convergent Descent Method for Minimization," The Computer Journal, vol. 6, no. 2, July, 1963.

Fletcher, R., and C. M. Reeves

1964 "Function Minimization by Conjugate Gradients," The Computer Journal, vol. 7, pp. 149-154, 1964.

Forsythe, G. E., and T. S. Motzkin

1951 "Acceleration of the Optimum Gradient Method," Bulletin of the American Mathematical Society, vol. 47, pp. 304-315, 1951.

Gilbert, E. G.

1967 "A Selected Bibliography on Parameter Optimization Methods Suitable for Hybrid Computation," Simulation, vol. 8, no. 6, 1967.

Gonzalez, R. S.

1969 "An Optimization Study on a Hybrid Computer," M.S. Thesis, Department of Electrical Engineering, The University of Arizona, 1969.

Gurin, L. S., and L. A. Rastrigin

1965 "Convergence of the Random Search Method in the Presence of Noise," Automation and Remote Control, vol. 26, pp. 1505-1511, 1965.

Hague, D. S., and C. R. Glatt

1968 "An Introduction to Multivariable Search Techniques for Parameter Optimization (and Program AESOP)," NASA CR-73200, April, 1968.

Hahn, G. J., and S. S. Shapiro

1967 Statistical Models in Engineering, Wiley, New York, 1967.

Hampton, R. L. T.

1968 "Survey of Stochastic Approximation and Its Applications," Term Paper, Department of Electrical Engineering, The University of Arizona, January, 1968.

Harkins, Alvin

1964 "The Use of Parallel Tangents in Optimization," Optimization Techniques, Chemical Engineering Symposium Series, vol. 60, 1964.

Hestenes, M. R., and E. Stiefel

1952 "Method of Conjugate Gradients for Solving Linear Systems," Journal of Research, National Bureau of Standards, vol. 59, pp. 409-436, 1952.

Heydt, G. T.

1969 "Random Search Using Hyperconical Search Regions," Ph. D. Thesis Proposal, Purdue University, February, 1969.

Hooke, R., and T. A. Jeeves

1958 "Comments on Brooks' Discussion of Random Methods," Operations Research, vol. 6, no. 6, 1958.

1961 "Direct Search Solution of Numerical and Statistical Problems," Journal of the Association of Computing Machinery, vol. 8, no. 2, 1961.

Huelsman, L. P.

1968 "GOSPEL--A General Optimization Software Package for Electrical Network Design," Dept. of Electrical Engineering, The University of Arizona, 1968.

Janac, Karel

1967 "Parameter Optimization of Dynamic Systems," Presented at the Fifth International Conference AICA, Lausanne, September, 1967.

1969 "Adaptive Stochastic Approximations," Electronic Associates, Inc., Princeton, New Jersey, 1969.

Karnopp, D. C.

1963 "Random Search Techniques for Optimization Problems," <u>Automatica</u>, vol. 1, pp. 111-121, 1963.

Kavanaugh, W. P., E. C. Stewart, and D. H. Brocker

1968 "Optimal Control of Satellite Attitude Acquisition by a Random Search Algorithm on a Hybrid Computer," <u>Proceedings Spring Joint Computer Conference</u>, 1968.

Kiefer, J., and J. Wolfowitz

1952 "Stochastic Estimation of the Maximum of a Regression Function," <u>Annals of Mathematical Statistics</u>, vol. 23, pp. 462-466, 1952.

Kopp, R. E.

1967 "Computational Algorithms in Optimal Control," Research Department, Grumman Aircraft Engineering Corp. Bethpage, New York, 1967.

Korn, G. A.

1966 <u>Random-Process Simulation and Measurements</u>, McGraw-Hill, New York, 1966.

1969 "Project DARE Differential Analyzer REplacement by On-line Digital Simulation," <u>Proceedings Fall Joint Computer Conference</u>, 1969.

Korn, G. A., and T. M. Korn

1964 Electronic Analog and Hybrid Computers, McGraw-Hill, New York, 1964.

1968 Mathematical Handbook for Scientists and Engineers, McGraw-Hill, New York, 1968.

Korn, G. A., and H. Kosako

1970 "A Proposed Hybrid-Computer Method for Functional Optimization," IEEE-TC, vol. C-19, no. 2, 1970.

Kushner, H. J.

1963 "Hill Climbing Methods for the Optimization of Multiparameter Noise Disturbed Systems," Journal of Basic Engineering (Transactions of the ASME), vol. 85, series D, no. 2, 1963.

Lasdon, L. S., S. K. Mitter, and A. D. Waren

1967 "The Conjugate Gradient Method for Optimal Control Problems," IEEE-TAC, vol. AC-12, no. 2, 1967.

Lavi, A., and T. P. Vogl (Eds.)

1966 Recent Advances in Optimization Techniques, Wiley, New York, 1966.

Leon, A.

1964 "A Comparison Among Eight Known Optimization Procedures," Internal Working Paper No. 20, Space Sciences Laboratory, University of California, Berkeley, August, 1964.

Matyas, J.

1965 "Random Optimization," Automation and Remote Control, vol. 26, no. 2, 1965.

Maybach, R. L.

1966a "Solution of Optimal Control Problems on a High-Speed Analog Computer," Simulation, vol. 7, no. 5, 1966.

Maybach, R. L.

1966b "Generation of Inverse Functions by the Method of Steepest Descent," Annales de l'Association Internationale pour le Calcul Analogique, vol. VIII, no. 4, 1966.

McGhee, R. B.

1967 "Some Parameter Optimization Techniques," Chapter 4.8 in Digital Computer User's Handbook, Melvin Klerer and G. A. Korn (Eds.), McGraw-Hill, 1967.

McGhee, R. B., and A. Levine

1964 "Determination of Optimum Production Tolerances by Combined Analog-Digital Computation," Simulation, vol. 3, no. 5, 1964.

Mitchell, B. A.

1964 "A Hybrid Analog-Digital Parameter Optimizer for ASTRAC-II," Proceedings Spring Joint Computer Conference, 1964.

Munson, J. K., and A. I. Rubin

1959 "Optimization by Random Search on the Analog Computer," IRE-TEC, vol. EC-8, no. 2, 1959.

Nelder, J. A., and R. Mead

1965 "A Simplex Method for Function Minimization," The Computer Journal, vol. 7, no. 4, 1965.

Pearson, J. D.

1968 "On Variable Metric Methods of Minimization," RAC-TP-302, Research Analysis Corp., McClean, Virginia, May, 1968.

Powell, M. J. D.

1964 "An Efficient Method of Finding the Minimum of a Function of Several Variables Without Calculating Derivatives," The Computer Journal, vol. 7, pp. 155-162, 1964.

Rastrigin, L. A.

1963 "The Convergence of the Random Search Method in the Extremal Control of a Many Parameter System," Automation and Remote Control, vol. 24, pp. 1337-1342, 1963.

1967 Random Search in Optimization Problems for Multiparameter Systems, Air Force Systems Command, Foreign Technology Division, August, 1967.

Robbins, H., and S. Munro

1951 "A Stochastic Approximation Method," Annals of Mathematical Statistics, vol. 22, pp. 400-407, 1951.

Rosen, J. B.

1960 "The Gradient Projection Method for Nonlinear Programming, Part I, Linear Constraints," SIAM Journal, vol. VIII, no. 1, 1960.

1961 "The Gradient Projection Method for Nonlinear Programming, Part II, Nonlinear Constraints," SIAM Journal, vol. IX, no. 4, 1961.

Rosenbrock, H. H.

1960 "An Automatic Method for Finding the Greatest or Least Value of a Function," The Computer Journal, vol. 3, no. 3, 1960.

Saaty, T. L., and J. Bram

1964 Nonlinear Mathematics, McGraw-Hill, New York, 1964.

Schumer, M. A., and K. Steiglitz

1968 "Adaptive Step Size Random Search," IEEE-TAC, vol. AC-13, no. 3, 1968.

Shah, B. V., R. J. Buehler, and O. Kempthorne

1964 "Some Algorithms for Minimizing a Function of Several Variables," Journal of SIAM, vol. 12, pp. 74-92, 1964.

Spang, H. A., III

1962 "A Review of Minimization Techniques for Non-linear Functions," SIAM Review, vol. 4, no. 4, 1962.

Spendly, W., G. R. Hext, and F. R. Himsworth

1962 "Sequential Applications of Simplex Designs in Optimization and Evolutionary Operation," Technometrics, vol. 4, no. 4, 1962.

Stewart, E. C., W. P. Kavanaugh, and D. H. Brocker

1967 "Study of a Global Search Algorithm for Optimal Control," Presented at the Fifth International Congress AICA, Lausanne, 1967.

Swann, W. H.

1964 "Report on the Development of a New Direct Search Method of Optimization," Imperial Chemical Industries Ltd., Central Instrument Laboratory Research Note 64/3, 1964.

White, R. C.

1970 "A Fast Digital-Computer Method for Recursive Estimation of the Mean," to be published, IEEE-TC, 1970.

Wilde, D. J.

1964 Optimum Seeking Methods, Prentice-Hall, Englewood Cliffs, New Jersey, 1964.

Wilde, D. J., and C. S. Beightler

1967 Foundations of Optimization, Prentice-Hall, Englewood Cliffs, New Jersey, 1967.

Zangwill, W. I.

1967 "Minimizing a Function Without Calculating Derivatives," The Computer Journal, vol. 10, no. 3, 1967.