Spring 1988

A feasible direction procedure for general multiple objective optimization

Wen-Tsia Liu
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

Order Number 8816822

A feasible direction procedure for general multiple objective optimization

Liu, Wen-Tsia, D.Eng.Sc.

New Jersey Institute of Technology, 1988

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark __✓__.

1. Glossy photographs or pages _____
2. Colored illustrations, paper or print _____
3. Photographs with dark background _____
4. Illustrations are poor copy _____
5. Pages with black marks, not original copy _____
6. Print shows through as there is text on both sides of page _____
7. Indistinct, broken or small print on several pages _____
8. Print exceeds margin requirements _____
9. Tightly bound copy with print lost in spine _____
10. Computer printout pages with indistinct print _____
11. Page(s) _____ lacking when material received, and not available from school or author.
12. Page(s) _____ seem to be missing in numbering only as text follows.
13. Two pages numbered _____. Text follows.
14. Curling and wrinkled pages ✓
15. Dissertation contains pages with print at a slant, filmed as received _____
16. Other _____

_____

_____

A FEASIBLE DIRECTION PROCEDURE FOR GENERAL MULTIPLE OBJECTIVE OPTIMIZATION

by

Wen-Tsia Liu

Dissertation submitted to the Faculty of the Graduate School of the New Jersey Institute of Technology in partial fulfillment of the requirements for the degree of Doctor of Engineering Science

1988

APPROVAL SHEET

Title of Thesis : A Feasible Direction Procedure for General Multiple Objective Optimization

Name of Candidate : Wen-Tsia Liu

Doctor of Engineering Science, 1988

Thesis and Abstract Approved :

Dr. Michael Pappas Date

Professor

Department of Mechanical Engineering

Signatures of Other Members of The Thesis Committee

Date

Date

Date

Date

VITA

Name : Wen-Tsia Liu

Degree and date to be conferred : D. Eng. Sc., 1988

secondary education :

College	dates	Degree	Date of Degree
Ming-Chi Institute of Technology	68-73	BSME	1973
New Jersey Institue of Technology	80-81	MSME	1982
New Jersey Institute of Technology	83-88	D. Eng. Sc.	1988

Major : Mechanical Engineering

ABSTRACT

Title of Thesis : A Feasible Direction Procedure for General Multiple Objective Optimization

Wen-Tsia Liu, Doctor of Engineering Science, 1988

Thesis directed by : Dr. Michael Pappas
Professor
Department of Mechanical Engineering

The Feasible Direction Finding Problem (DFP) of Zoutendijk is adapted to create a general Mathematical Programming (MP) algorithm for treating optimization problems with multiple objective functions. Classically such problems are reduced to standard MP form by converting them to single objective function problems by the use of weighting functions. Unfortunately not all practical problems can be so reduced. Consider the problem of maximizing the strength of a structure. Typically there are several, or even many, failure modes. All active failure modes must be included in the optimal search in such a way that resistance to one active mode can not be increased at the expense of another. Thus this problem can not be treated by reduction. The search must seek to increase resistance to all active modes.

The DFP formulation seeks to improve the objective function by including said function as a constraint in the DFP Linear Programming problem. Multiple objective

functions can be treated by simply including each such function as a constraint in the DFP. Thus the solution to such a DFP improves all the objective functions considered. There is no need to resort to reduction to a single objective function. An algorithm based on the DFP is described. This procedure locates a variable set where , at least locally, no further improvement in all objective functions is available (a Parato Optimum). A general multiple objective formulation is developed defining a wide range of optimization problems. It is shown that this formulation also includes the problem of locating the feasible region, either from an infeasible starting point, or for feasibility restoration during the search. Thus the method is of value in single objective function optimization.

The procedure is applied to a six variable problem with eleven constraints where the objective is to separate the two lowest natural frequencies of a stiffened thin shell. Four active frequencies are considered. Several two-variable, constrained and unconstrained, problems are also treated. The procedure was found to efficiently locate Parato Optima and was effective in feasible region location and restoration.

ACKNOWLEDGEMENT

I wish to express my sincere gratitude and thanks to my thesis advisor, Dr. Michael Pappas, for his valuable guidance and constructive criticisms throughout the whole process of this study. Without his guidance and encouragement, this work would not have been possible. I would also like to thank Dr. Koplik, Dr. Herman, Dr. Dave, and Dr. Sodhi for their critical reading of the manuscript and constructive suggestions.

TABLE OF CONTENTS

NOMENCLATURE

A_q = leveling function weighting factors

a_q = inequality constraint parameters

b_q = equality constraint parameters

B_j = behavior constraint lower limit

$C(x_i)$ = composite objective function

$C[f_q(x_i)]$ = multiple objective composite function

e_{j1} = behavior constraint band width parameter

e_{m1} = leveling function band width parameter

e_{j2} = active behavior constraint band width parameter

e_3 = optimality criteria parameter

e_4 = objective function convergence parameter

e_5 = minimum step size parameter

e_6 = leveling function convergence parameter

e_{j7} = excessive constraint violation band width parameter

$f_q(x_i)$ = objective functions

$f_q'(x_i)$ = alternate form of objective functions

$g_j(x_i)$ = behavior constraint functions

$h_k(x_i)$ = equality constraint function

I = number of variables

J = number of behavior constraint

J_p = number of potentially active behavior constraint

J_v = number of constraint violation

J_a = number of active constraint

K = number of equality constraint

K_1 = penalty function constant

$P(x_i)$ = penalty function

Q = number of objective functions

Q_A = number of active objective functions

S = move direction

S = move direction vector

x_i = design variables

x_i = optimal design variables

x = design variable vector

U_j = lower limit of behavior constraint

w_q = weighting factor

W = DFP weighting factor

α_i = step size

ϵ_m = level function

η = reduction attempt

σ = slack variable

$< \phi >$ = bracket function

∇ = gradient operator

ω = frequency

Subscripts and Superscripts

B = base point

C = comparison point

f = objective function index

i = variable index

j = behavior constraint index

k = equality constraint index

l = lower limit

m = leveling function number

q = objective function number

q' = smallest objective functions

r = iteration index

T = transpose of vector

u = upper limit

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Engineering design optimization procedures generally minimize an objective function subject to a set of constraints [1-3]. For example, in optimal structural design one often seeks to minimize the weight of the structure subject to constraints on its behavior. In many design situations, however, a single design objective does not adequately define the critical design goals. To treat this situation one needs to consider the simultaneous optimization of multiple objective functions. Unfortunately, there is in general no unique solution to the multiple objective problem, even locally.

During the last decade, considerable attention has been given to multiple objective design optimization [4-13]. Osyczka [4] and Goiccechea et al [5] survey the application of operations research methods to engineering problems. Mechanical Engineering multiple objective problems have been studied by; Bartel and Marks [6] who describe the use of trade-off relationships to deal with conflicting objectives; Charmicheal [7] who studies multiobjective optimal design of a simple truss; Rao and Hati [8] who apply game theory to multicriteria optimization of machanisms; Yoshimura, et al [9] who derive the conditions for the Parato optimum

solution (POS) set [10] for machine tool spindle design from the Kuhn-Tucker conditions; Metwalli, et al [11] who adopt an exponential weighting method to the multiobjective design of hydrodynamic bearings; Aldi [12] who uses a weighting method for multiobjective design of antisymetric, angle ply laminates; and Ito, et al [13] who adopt a weighting method for the development of multiobjective man-machine interactive optimal planning systems.

The general approach to multiple objective optimization is to transform the multiple objective functions into a single composite function. This implies a trade-off relationship among the objective functions. Usually this trade-off is defined by a set of weighting parameters and associated smooth, continuous functional relationship among the competing objective functions. Unfortunately such an approach is not applicable to the maximum performance problems often encountered in engineering [14-15] . Pappas describes a procedure for treating this multiobjective maximum performance problem based on the Feasible Direction Finding Problem (DFP) of Zoutendijk [16].

This thesis expands on the earlier work of Pappas by defining a more efficient and general formulation of the multiple objective function problem capable of treating both the maximum performance problem as well as the problems that can be treated by reduction to a single objective form.

Further, this formulation is sufficiently broad so as to also allow its use for location of the feasible region from an infeasible point. Equality constraints which severely restrict the feasible region, can be treated here in an explicit form. Because of their difficulty , these constraints are usually avoided in nonlinear problems or are eliminated by solving for one of the variables in terms of the others by reducing the constraint to implicit form. Unfortunately such a procedure usually increases the nonlinearity of the equations [1,17,18]. Furthermore, such an approach cannot treat complex equality constraints especially those with inexpressible functions. Therefore, this treatment is also valuable in general purpose optimization.

CHAPTER II

REVIEW OF SINGLE OBJECTIVE DESIGN OPTIMIZATION

2.1 Importance of Optimization

The concept of optimization is intrinsically tied to humanity's desire to excel. Though one may not conciously recognize it, this concept appears everywhere in life. Optimization is of great interest and utility in many fields including engineering, operations research, science, mathematics, military, industrial operations, and economics. Designers can apply optimization methods to engineering design problems to achieve the best results in terms of material, efficiency, weight, cost, manufacturing reliability, marketing, or combination of all or part of these.

With existing optimization procedures, an ordered approach is used for design decisions in situations where previously one relied heavily on intuition and experience. Among the various approaches to optimization, Mathematical Programming (MP) procedures appear to have the broadest range of application. MP procedures such as linear, nonlinear, quadratic, dynamic, geometric, and interger programming are flexible and easy to adapt.

Most optimization problems require use of iterative numerical procedures. A difficulty with these procedures is often that a vast amount of computation effort is needed to reach an optimum. Other difficulties include problem complexity, and existence of multiple or even numerous local optima. Still, recent dramatic improvements in computer computational speed and the development of efficient MP procedures have made the optimization methods quite pratical for many applications even in the face of these difficulties. In Mechanical Engineering, MP optimal design methods can be applied to all fields such as structural analysis, thermodynamics, fluid dynamics, heat transfer, biomechanics, and composite materials [1-3].

2.2 Problem Formulation

The MP optimization problem can be stated mathematically as follows :

Find those values $\bar{x}_i$ of the design variables x_i that minimize the objective function

$$f(\bar{x}_i) = \min f(x_i) \qquad i = 1,2,...I \qquad (2.1)$$

subject to the inequality constraints

$$g_j(x_i) \leq 0 \qquad j = 1,2,...J \qquad (2.2)$$

and/or the equality constraints

$$h_k(x_i) = 0 \qquad k = 1,2,...K \qquad (2.3)$$

A " side or regional " constraint form of inequality constraints

$$x_i^l \leq x_i \leq x_i^u \qquad (2.4)$$

are generally specified since these are simpler to treat than the general inequality constraints of Eq.(2.2). Here x_i^l and x_i^u are lower and upper limits on the design variable x_i respectively.

These functions may be explicit or implicit in x_i and may be evaluated by analytical or numerical techniques. Except for special classes of optimization problems which use special solution techniques, it is important that these functions are continuous and have continuous first derivatives in x_i [1].

Though current digital computers offer rapid calculation for design analysis, the CPU time required to achieve an optimal design solution, as demonstrated by the example of Ref.[1], may vary from a few CPU seconds to 3200 years or more. Therefore, a more rational approach to design automation is needed. Mathematical programming techniques offer a logical approach to design automation and many algorithms have been proposed in recent years.

For those problems where Eq.(2.1) through (2.3) are all linear, highly effective linear programming methods can reliably locate the global optimum in a finite number of steps [20]. This situation unfortunately does not exist in the case of general nonlinear problems. None of the many nonlinear methods proposed can guarantee a solution except in certain relatively restricted problem forms [21].

Nonlinear MP methods are essentially optimal search strategies. The basic unconstrained nonlinear problem is, by comparison with the general constrained linear problem, quite difficult. The difficulty is greatly compounded when constraints are used. Equality constraints are particularly troublesome since they severely restrict the feasible region. Although in most situations, equality constraints can be eliminated by using the equality to remove an independent variable, for many highly complex and nonexpressible nonlinear equations, such an elimination can not be accomplished. Furthermore, in certain situations with large number of equality constraint equations, the elimination method may inrease the complexity and nonlinearity of the design problem [17]. In this thesis, a simple approach using the power of the Direction Finding Problem (DFP) of Zoutendijk [16] will be demonstrated to treat equality constraints effectively.

2.3 Methods of Solution

Most of optimization algorithms require that the initial design variables , $\underset{\sim}{x}^{o}$, be specified. Beginning from this starting point, the design is updated iteratively until a termination criteria is satisfied. Probably the most common form of this iterative update is given by

$$\underset{\sim}{x}^{r} = \underset{\sim}{x}^{r-1} + \alpha^{*} \underset{\sim}{s}^{r} \qquad (2.5)$$

where r is the iteration number, and $\underset{\sim}{s}$ is a search direction vector in the design space. The scalar quantity α^{*} defines the distance that one wishes to move in direction $\underset{\sim}{s}$. A variety of methods involving ordinary and variational calculus, mathematical programming and optimal criteria, etc are available to search along the $\underset{\sim}{s}$ vector and define the scalar α^{*}. Among those methods, the MP procedures appear to have the broadest range of application [22]. Such methods are flexible, easy to adapt, and able to offer " automatic " optimal computer solutions [23].

Eason and Fenton [24] evaluated seventeen different optimization codes on ten constrained nonlinear problems. Later, Pappas [24,25] used this evaluation method for his CADOP5 series codes and the OPT code of Gabriele and Ragsdell [26]. These studies show CADOP5 which is based on the DFP to be more efficient than any other code on

problems with nonlinear constraints. This thesis will expand the use of the DFP to treat the highly nonlinear constrained multiobjective design problems. A new code CADOP8 based on the CADOP5 is developed for this purpose.

2.4 Existence and Uniqueness of An Optimal Solution

In the application of optimization techniqes, it is seldom possible to ensure that the global solution will be found. This may be due to the existence of local optimum solutions, or simply because numerical ill-conditioning in setting up the problem results in extremely slow convergence of the optimization algorithm.

It is possible to use mathematical methods such as the Kuhn-Tucker conditions [27] to define the necessary conditions for an optimum, and therefore, identify a local optimum. However, it is seldom possible in practical applications to know whether the sufficiency conditions for an optimum are met. Also, it is not practical to identify a global minimum. Thus, from a practical standpoint, the best approach is usually to start the optimization process from several different initial vectors. Moreover, if the optimization results in essentially the same final design, one can reasonably assume that this is the true global optimum. However, it should be noted that typical design problems have several or even many local optima. Thus

positive identification of the true global optimum may not be feasible. Multiple starting points however, will, fortunately, usually identify either a local optimum nearly equal to the global, or find the global optimum itself.

CHAPTER III

REVIEW OF MULTIPLE OBJECTIVE DESIGN OPTIMIZATION

3.1 Introduction

The modern optimization methods for engineering design generally assume that a scalar objective function such as cost, efficiency or weight can be defined so that standard computational algorithms from mathematical programming can be applied. The usefulness of those methods is seriously limited by the fact that the quality of a complex engineering design generally depends on a number of different and often conflicting objectives which can not be combined into a single design objective. Hence, the consideration of multiple objectives becomes important in the optimization of engineering design.

During the last decade, considerable attention has been given in the literature to multiobjective design optimization problems [4-13]. Examples include problems in structures [7,12], thermal systems [13], hydrodynamic journal bearings [6,11], mechanisms [8], and machine-tool spindle design[9]. Many theories of multiple objective optimization were developed in operations research, and many methods of solutions were applied in engineering applications to make decisions in situations, in which

several conflicting objectives are sought. In general, it is impossible to achieve the minima or maxima of all the objective functions in a single design. Hence, a compromise solution is usually chosen. Instead of using " minimize or maximize " , the word " optimize " will be used in multiple objective design optimization. Here " optimize " does not mean simply to find the minimum of the objective functions as it does for a single objective optimization problem. It means to find a " best " solution considering all the objective functions. The concept of Parato Optimal Solutions (POS) [10] is widely used to build up the sets of solution for decision making. In order to select the optimum design, a criterion is needed to rank the possible POS. This design criterion might result from a third design objective, experimental evidence, an expert's rating, or some other relationships among objectives [6]. Among many different decision making procedures, the class of Mathematical Programming methods have recently begun to receive much attention. In this class of problems, an optimization task is described by functions which refer to both constraints and objectives, and give a formal description of the task.

3.2 Problem Formulation

If possible, one would like to find those $\bar{x}_i$ such that

$$f_q(\bar{x}_i) = \text{Smin } f_q(x_i) \qquad q = 1,2,\ldots Q \tag{3.1}$$

where Smin $f_q(x_i)$ is the simultaneous minimum of the function $f_q(x_i)$ subject to appropriate constraints. Unfortunately, no such solution is generally possible since those values of x_i that minimize one function will not minimize another. Alternatively, one can define a Smin $f_q(x_i)$ where no function $f_q(x_i)$ can be further reduced in value without increasing another function value. This, of course, is the Parato Optimum [10] and generally not a unique point. Different Parato Optima generally have different values of the $f_q(x_i)$; the user is then faced with selecting one of these optima as " best " by involving some secondary consideration, usually some specified relationship among the variables. Thus, the multiple objective function problem is usually formulated as :

Find $\bar{x}_i$ such that

$$C[f_q(\bar{x}_i)] = \min C[f_q(x_i)] \qquad q = 1,2,...Q \qquad (3.2)$$

where $C[f_q(x_i)]$ is a compositive function of the $f_q(x_i)$, and Q is the number of objective functions.

3.3 The Conflict between Competing Objectives

Where the minimization of one objective function results in an increase in another objective function, these

design objectives are said to be competing. The conflict often occurs in multiple objective design optimization and is the basic difficulty associated with multiple objective design decision making. It is impossible to obtain the minima for all functions in a single design. Hence, any solution must include some compromise among the minima of the competing objectives. This situation will generally be true for unconstrained problems.

From Fig. 1, one can see that a conflict condition exists between the minimum of f_1 and the minimum of f_2. Although in certain regions one can reduce both f_1 and f_2 simultaneously, ultimately f_1 or f_2 can be reduced only at the expense of one of the others, and thus, these functions compete. Constrained problems can, however, restrict the feasible region such that the functions do not compete.

3.4 The Concept of Parato Optimality

The concept of Parato optimality was formulated by V. Parato [10]. It is still the most important part of multiple objective design optimization. Its physical meaning is easily understood. One defines a Parato Optimal Solution (POS) as follows : If the set $(\bar{x}_i)$ is a POS, then for any other (x_i) in the neighborhood at least one function increases compared to its value at $(\bar{x}_i)$. The Parato optimum defines a set of solutions called non-inferior or non-

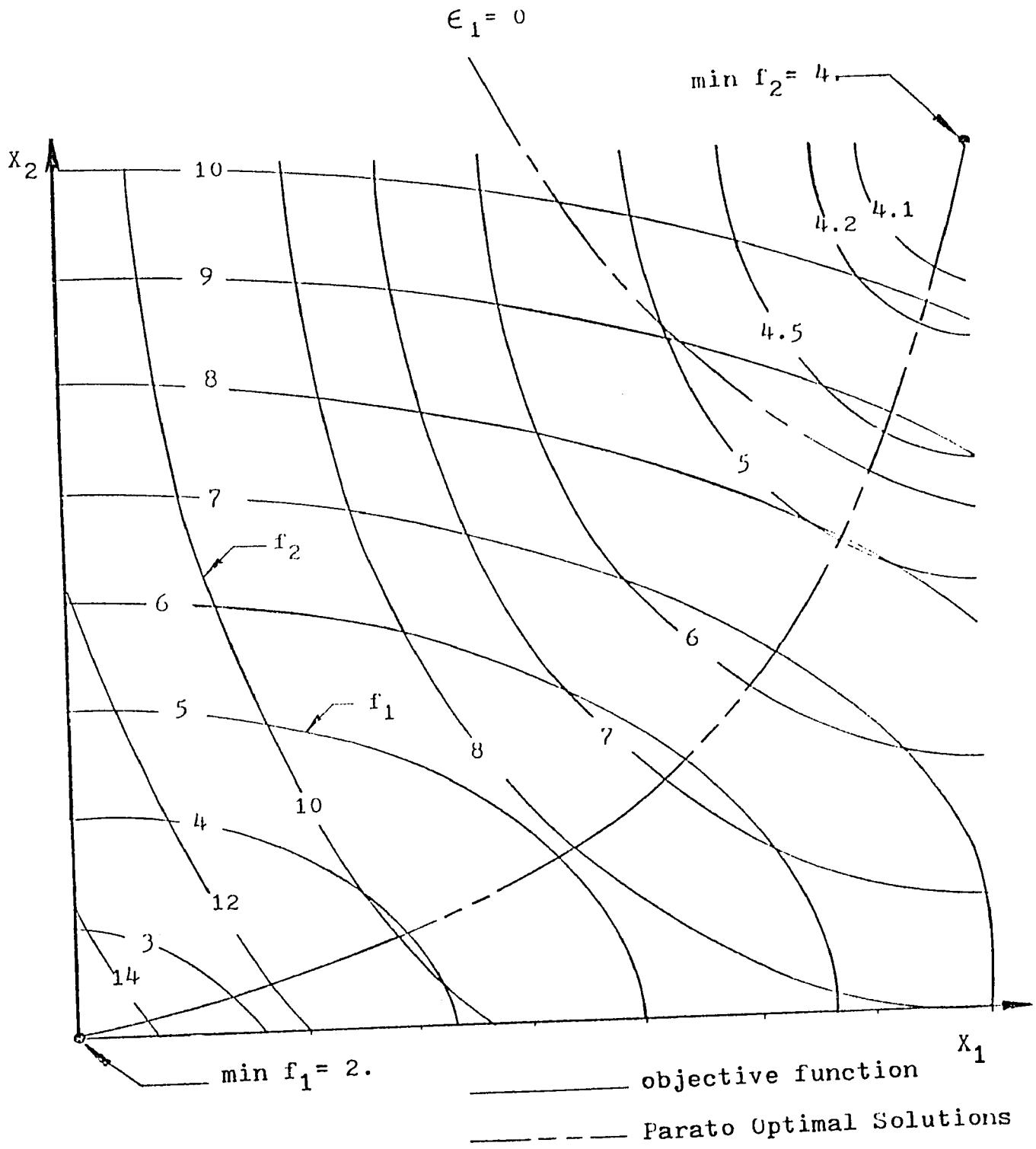


Fig. 1 Conflict condition between two objective functions

dominated solutions. All possible compromise solutions of competing objectives are defined in the POS set. On the basis of Parato optimum, many methods have been developed to search the POS set which supplies the possible choice for multiple objective optimal design [4-13]. In Fig. 1, the heavy phantom between the minima of f_1 and f_2 is the POS set for two objectives. On that POS line, nothing can be done to improve both f_1 and f_2 simultaneously since ∇f_1 and ∇f_2 are in opposite direction.

3.5 Methods of Solution

Before final decision making, the POS set must be obtained. Many methods have been developed and applied in engineering design problem [4-13] for the determination of this set. The methods based on function scalarization are most common [4]. Some of these approaches are briefly outlined below :

(1) Arithmetic Weighting Methods

An objective is constructed as a linear combination of the original objectives as in [4-6]. Here

$$C[f_q(x_i)] = \sum_{q=1}^{Q} w_q \, f_q(x_i) \qquad \begin{matrix} i = 1,2,....I \\ q = 1,2,....Q \end{matrix} \qquad (3.3)$$

where w_q are positive weighting factors denoting the

relative importances of the various objective functions for minimization. It is usually assumed that

$$\sum_{q=1}^{Q} w_q = 1 \qquad (3.4)$$

If one wishes w_q to reflect closely the importance of the objectives, all functions should have approximately the same numerical values [4].

(2) Exponential Weighting Method

In this method, the composite objective is constructed as :

$$C[f_q(x_i)] = \frac{(f_1)^{w_1}(f_2)^{w_2}.....(f_n)^{w_n}}{(f_{n+1})^{w_{n+1}}(f_{n+2})^{w_{n+2}}...(f_Q)^{w_Q}} \qquad (3.5)$$

where w_q are exponential weighting factors, f_q are to be minimized for q = 1,2,...n, and to be maximized for q = n+1, n+2,...Q [11].

(3) Constraint Method

One can generate the trade-off curves by the constrained method of Ref.[5] which states :

$$\text{minimize} \qquad f_1(x_i) \qquad\qquad i = 1,2,....I \quad (3.6)$$

where the constraints

$$f_q(x_i) \leq a_q \qquad\qquad q = 1,2,...Q$$

or

$$f_q(x_i) = b_q$$

are added to the constraint set.

Here $f_1(x_i)$ is chosen from $f_q(x_i)$, a_q and b_q assume a range of values.

Most of above methods are solved by numerical optimization methods. An analytical method using the Kuhn-Tucker conditions is described by Yoshimura [9] to solve the optimization of Eq. (3.3) and to define the range of w_q that will yield a POS.

CHAPTER IV

THE FUNDAMENTAL LEVEL OBJECTIVE PROBLEM

4.1 Multiple Active Modes

The fundamental level objective problem arises frequently in the maximum performance problem such as described in Refs.[14,28] where it is desired to maximize the minimum natural frequency or buckling resistance of a structure. In general structural design, this situation may also occur from design requirements limiting maximum stress.

The design objective in a maximum performance structural problem is typically associated with behavior modes, such as natural frequency, buckling, stress etc. Consider a case where it is desired to maximize the buckling resistance of a structure. If, as is usual, only the most critical mode is considered in formulating the redesign problem then after redesign a new mode which was not previously critical may become critcal. This can lead to a redesign with a lower rather than higher critical buckling load value since the new critical constraint is ignored.

Thus the redesign attempt to improve performance may fail causing algorithm failure. One needs to consider and maximize simultaneously all potentially active buckling modes. Thus one must solve a multiple objective problem to treat this case.

The minimum natural frequency problem of [14] seeks to maximize the lowest natural frequency $f_{q'} = \min f_q(x_i)$, which is the minimum among all of the natural frequencies f_q. To solve this problem, all potentially active f_q must be simultaneously maximized. This problem can be stated as follows :

Find those $\bar{x}_i$ such that

$$f_q(\bar{x}_i) = \max [\min f_q(x_i)] \qquad q \in Q_A \qquad (4.1)$$

where the max [min $f_q(x_i)$] is the maximum of the smallest of the set Q_A of active behavior modes q , and the f_q is the frequency associated with the q^{th} mode.

This will be called the fundamental level objective problem. The problem may be put into the form of Eq. (3.2) by noting that

$$C[f_q(\bar{x}_i)] = - \min f_q(x_i) \qquad (4.2)$$

This problem is a multiple objective problem. Unfortunately, conventional formulation of multiple objective problems as discussed in chapter 3 and associated solution methologies are not suitable for this problem form.

4.2 Formulation Generalization

One can write Eq.(4.1) in light of Eq.(4.2) as :

Find those $\bar{x}_i$ such that

$$f_q(\bar{x}_i) = \min [-\min f_q(x_i)] \quad . \tag{4.3}$$

Now for this to be the case in a region with conflicting objective functions then

$$f_{q'}(x_i) - f_q(x_i) \leq 0 \quad \text{for all } q \neq q' \tag{4.4}$$

Where q' is associated with the min $f_q(x_i)$.

Consider the case where Q = 2. Where f_1 is the smallest of f_1 and f_2 then

$$f_1 - f_2 \leq 0 \quad .$$

Where this is not true then f_2 must be the smallest and thus

$$f_2 - f_1 \leq 0 \quad .$$

Since one or the other must be true one can combine equations and state that

$$|f_1 - f_2| \leq 0 \quad .$$

But this is only true if

$$f_1 - f_2 = 0 \quad .$$

This argument can be generalized and thus one can replace Eq. (4.4) with

$$f_{q'} - f_q = 0 \qquad q = 2,3,\ldots Q \quad . \qquad (4.5)$$

Now Eq.(4.1) can also be written as ;

Find $\bar{x}_i$ such that

$$f_q(\bar{x}_i) = \min f_{q'}(x_i) \qquad (4.6)$$

subject to q-1 equality " leveling " constraints

$$\epsilon_{q-1} = f_{q'} - f_q = 0 \qquad q \neq q' \quad . \qquad (4.7)$$

To avoid notation confusion, one can replace Eq.(4.7) for convenience as :

$$\epsilon_m = |f_1 - f_{1+m}| = 0 \qquad m = 1,2,\ldots Q_A - 1 \qquad (4.8)$$

Here the functions are renumbered and f_1 replaces $f_{q'}$. Due to the existance of the absolute notation of Eq. (4.8), one can choose f_1 from f_q arbitrary without affecting the result.

4.3 Application to Other Multiple Objective Problems

The solution methology developed for the solution of Eqs. (4.1-4.2) may be applied to locate POS of a conventional multiple objective optimization problem by reformulating the fundamental level objective problem of Eqs. (4.1) and (4.2) simply by defining

$$C[f_q'(\bar{x}_i)] = \min [A_q f_q'(x_i)] \qquad q = 1,2,...Q \qquad (4.9)$$

By substituting $A_q f_q'$ for $f_q(x_i)$ of Eq. (4.1) through (4.2) where f_q' are the functions to be simultaneously minimized and where A_q is a weighting parameter.

Thus using the argument of section 4.2, the POS for a problem with Q objective functions can be found from the problem ;

Find $\bar{x}_i$ such that

$$f_q(\bar{x}_i) = \min f_q(x_i) \qquad (4.10)$$

subject to the constraints

$$\epsilon_m = \left| f_1(x_i) - (A_q/A_1)\ f_q(x_i) \right| = 0 \quad m = 1,2,..Q-1 \quad (4.11)$$

The minimization of Eq.(4.9) in a region where objective functions conflict means a POS set is obtained. A range of POS sets can be generated by solving Eq. (4.9) through a suitable range of A_q. Equation (4.9) is therefore a new weighting method which may be used to generate the POS sets.

4.4 Application of the Fundamental Level Objective Formulation to Feasibility Restoration (FR)

If the design initial point is in the infeasible region of a constrained problem, one can attempt to eliminate all constraint violations by decreasing the value of all violated g_j without causing some new violation or by reducing the magnitude of all $|h_k|$ to zero. Hence, one has a version of the maximum performance with critical constraint elimination as the goal.

Thus the FR problem can be formulated by equation of the form Eq. (4.1) where $f_q(x_i) = - g_j(x_i)$ or $f_q = - |h_k(x_i)|$. The FR problem can then be stated as :

Find those $\tilde{x}_i$ such that

$$g_j(x_i) = \min [\max g_j(x_i)] \qquad j \in J_v \qquad (4.12)$$

$$h_k(x_i) = \min [\max h_k(x_i)] \qquad (4.13)$$

The search for a solution to Eqs. (4.12) and (4.13) proceeds until $g_j \leq 0$ and all $h_k = 0$. A MP procedure designed to solve Eq. (4.1) can likewise treat the multiple objective and feasibility restoration problem both in single and multiple objective optimization.

CHAPTER V

GENERAL OPTIMIZATION PROCEDURE

5.1 Introduction

There are many ways to approach the problem of Eqs. (4.6-4.7) by use of the DFP. Four methods are illustrated in Fig. 2.

The first method is to first reduce all ϵ_m until all $\epsilon_m = 0$ and then to reduce all f_q along these equality constraints. In the first stage, all ϵ_m are minimized simultaneously. During these moves the f_q may be increased if necessary to reduce ϵ_m.

The second method is to reduce all ϵ_m and f_q simultaneously until reaching a POS of the f_q, and then to proceed as in the the first method.

The third method is to first reduce all f_q on one move then to reduce all ϵ_m on the next. The process is repeated until all f_q and all ϵ_m are minimized to the POS.

The fourth method is to first minimize all f_q simultaneously until reaching a POS of the f_q, then to reduce all ϵ_m simultaneously until $\epsilon_m = 0$. The process

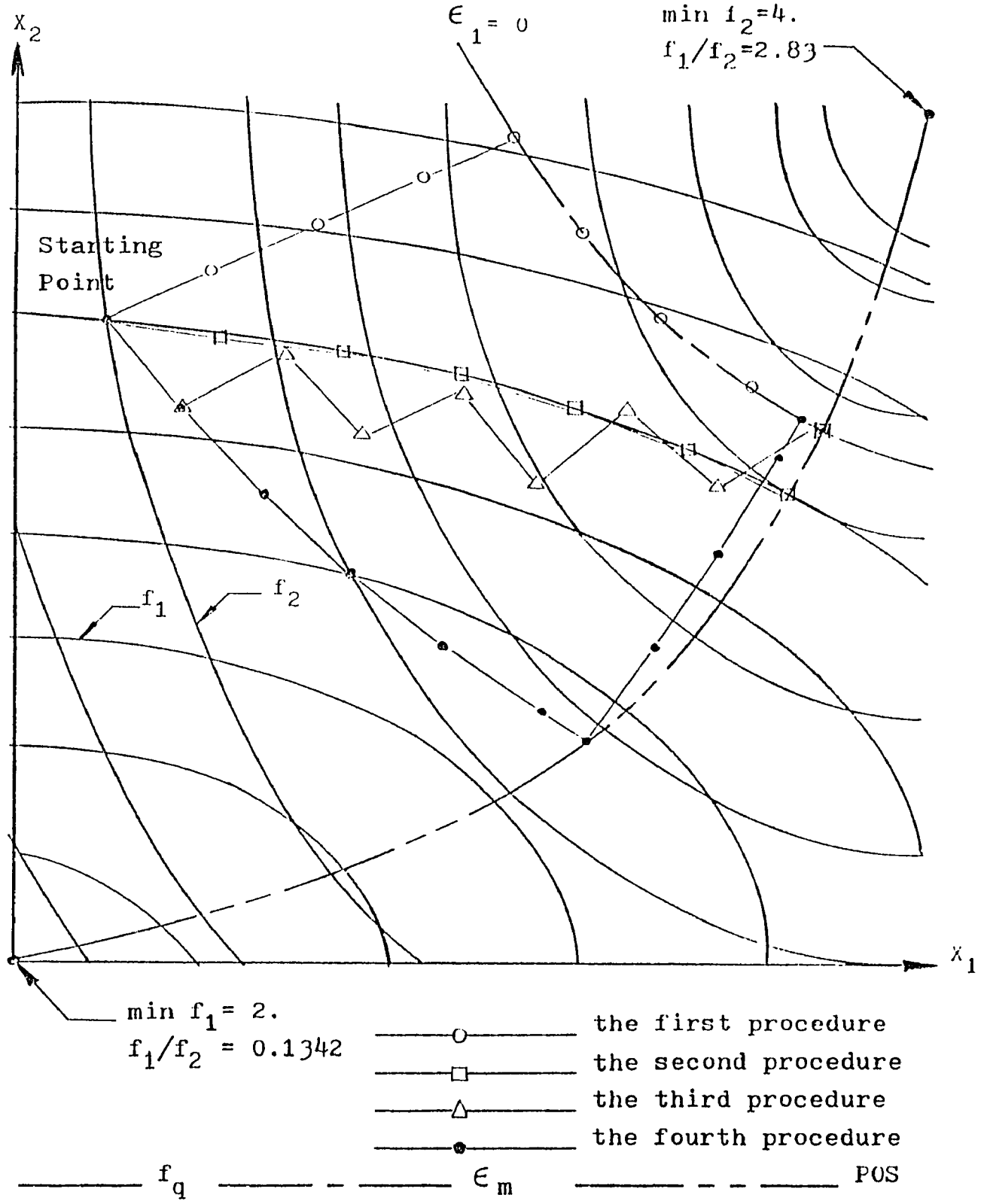


Fig 2. Parato optimum search of four different procedures

is repeated until no further minimization of the f_q is possible. In this procedure, if the f_q are minimized, the ϵ_m may be increased if necessary to reduce the f_q. Preliminary experimental studies of these four approaches undertaken as part of this research show the first to be the most effective and reliable method for problems where the active or competing objective can be identified. It is therefore adopted here for such problems.

For the general constrained problem, there are two tasks to be accomplished in solving Eqs. (4.6-4.7). First, the feasible region must be found, preferably without increasing, or with a minimum increase in, objective function values. Secondly, the objective function values must be minimized simultaneously along the feasible region boundary.

The idea presented in [14,16] is adapted to the procedure proposed here to treat the Multiple Objective Problem. The modification to the DFP suggested in [29] so as to greatly improve its convergence power is also used. The symmetric penalty method of [30,31] is utilized for the purpose of comparing feasible and infeasible points in the move strategy.

5.2 Determination of Active Objective Functions

The constraints of Eq. (4.7) can be imposed only for competing objectives. For unconstrained problems with Q objective functions there will be Q-1 such constraints. Thus there will be Q active objective functions. However, where the problem is constrained the feasible region may be limited such that not all Q functions are competing. For many maximum performance problems there may be many more possible objective functions than variables and no need to include all the possible functions in the problem formulation. Thus one has the problem of determining which, and how many, of the possible objective functions should be included (considered or active).

Consider two approaches. In the first approach one includes all possible functions in the problem formulation. In the second approach one includes only those functions that are competing at the point under evaluation.

An attempt to find a solution to the problem based on the first approach will overconstrain the problem since there is no need to specify a leveling constraint ϵ_m associated with functions which do not conflict in the region of search termination. One could, therefore, remove the noncompeting functions and restart the search. It is not clear however

that this procedure would converge to a POS if there is one. In the second approach leveling function constraints would be added as additional competing functions are identified or deleted as objecitves become noncompeting. Objective competition can be established from a comparison of functions and from the derivative information generally computed in the formulation of the DFP.

Clearly the second approach is superior and is adopted here as the generalized procedure. Where, however, one knows beforehand which objectives are competing at the POS then search efficiency can be improved by immediately including these objectives rather than waiting for them to compete. This later method is therefore used where possible.

5.3 Parato Optimality Search

The purpose of this procedure is to seek the POS along the leveling constraint Eq. (4.8). The procedure follows the method described by Pappas [14,19] and is given by the following steps :

1. For a near feasible base point $\underset{\sim}{x}_B{}^r$, evaluate the composite objective function value $C_q(\underset{\sim}{x}_B{}^r)$

where $$C_q(\underset{\sim}{x}) = f_q(\underset{\sim}{x}) + P_q(\underset{\sim}{x}) \qquad q = 1,2,....Q \qquad (5.1)$$

$$P_q(x) = \max \left| \lambda_{qj} \langle g_j(x) \rangle \text{ or } \lambda_{qm} \langle \epsilon_m(x) \rangle \right| \quad (5.2)$$

$$\lambda_{qm} = \begin{cases} 2|\nabla f_q|^2 \;/\; \nabla \epsilon_m \cdot \nabla f_q & \epsilon_m \le e_{m1} \\ K_1 & \epsilon_m > e_{m1} \end{cases} \quad (5.3)$$

$$\lambda_{qj} = \begin{cases} 2|\nabla f_q|^2 \;/\; \nabla g_j \cdot \nabla f_q & g_j \le e_{j1} \\ K_1 & g_j > e_{j1} \end{cases} \quad (5.4)$$

$$\begin{aligned} \langle \phi \rangle &= 0 \qquad \phi < 0 \\ \langle \phi \rangle &= \phi \qquad \phi > 0 \end{aligned} \quad (5.5)$$

Here $f_q(\underset{\sim}{x})$ is the objective function, $C_q(\underset{\sim}{x})$ is the compositive function for design comparison, $P_q(\underset{\sim}{x})$ is the penalty function, which is defined by Eq.(5.2) and will be described more clearly later, $|\underset{\sim}{A}|$ is the magnitude of vector $\underset{\sim}{A}$, $\nabla\phi$ is the gradient of the scalar function ϕ, K_1 is an arbitrary large positive number, e_{m1} and e_{j1} are band width parameter defining excessive constraint violation of equality constraint ϵ_m and inequality constraint g_j respectively [19].

3. At point $\underset{\sim}{x}_B^r$ find σ_q and S_i so as to :

$$\text{Maximize} \quad \sum_{q=1}^{Q} \sigma_q \qquad \sigma_q > 0 \quad (5.6)$$

subject to the conditions

$$(S_i)^T \cdot \nabla f_q(x_i) + \sigma_q \le 0 \qquad q = 1,2,\ldots.Q \quad (5.7)$$

$$(S_i)^T \nabla \epsilon_m(x_i) + \epsilon_m(x_i) = 0 \qquad m = 1,2,....Q-1 \tag{5.8}$$

$$(S_i)^T \nabla g_j(x_i) + g_j(x_i) \leq 0 \qquad j = 1,2,...J_a \tag{5.9}$$

$$, g_j(x_i) > -e_{j2}$$

$$(S_i)^l \leq S_i \leq (S_i)^u \tag{5.10}$$

$$S_i^l = \begin{cases} x_i^l - x_i \text{ if } x_i - x_i^l < \alpha_i^f \\ -\alpha_i^f \quad , \text{ otherwise} \end{cases} \tag{5.11}$$

$$S_i^u = \begin{cases} x_i^u - x_i \text{ if } x_i^u - x_i < \alpha_i^f \\ -\alpha_i^f \quad , \text{ otherwise} \end{cases} \tag{5.12}$$

where σ_q is a slack variable, α_i^f is the maximum step size in the direction of x_i, ϵ_m is the leveling function as defined by Eq. (4.8). The problem defined by Eqs. (5.6-5.12) is called the Direction Finding Problem (DFP). Based on the local linearization, equations (5.6-5.10) constitute a linear programming problem with the variables σ_q and S_i. The solution S_i can be obtained reliably and efficiently using any suitable linear programming method such as simplex procedure [20].

4. If S_i^r is sufficiently small i.e. if

$$|\underset{\sim}{S}^r| < e_3 \tag{5.13}$$

where e_3 is an arbitrary small variable convergence parameter. Then the design is considered Parato optimum and

the procedure is terminated. Otherwise define a comparison base

$$\underset{\sim}{x}_C{}^r = \underset{\sim}{x}_B{}^r + \underset{\sim}{S}^r \tag{5.14}$$

5. If any $$C_q(\underset{\sim}{x}_C{}^r) > C_q(\underset{\sim}{x}_B{}^r) \tag{5.15}$$

call $\underset{\sim}{x}_B{}^{r+1} = \underset{\sim}{x}_B{}^r$, then increase r by 1 and repeat step 1 to 4 with $\alpha_i{}^f$ halved.

6. Otherwise call

$$\underset{\sim}{x}_B{}^{r+1} = \underset{\sim}{x}_C{}^r \tag{5.16}$$

Now if for all objective functions

$$\left.\begin{array}{l} \left|\left[\, C_q(\underset{\sim}{x}_B{}^{r+1}) - C_q(\underset{\sim}{x}_B{}^r) \,\right] / C_q(\underset{\sim}{x}_B{}^r) \,\right| \\ \text{or} \\ \left|C_q(\underset{\sim}{x}_B{}^{r+1}) - C_q(\underset{\sim}{x}_B{}^r)\right| \end{array}\right\} < e_4 \tag{5.17}$$

where e_4 is an arbitrary small objective function convergence parameter , then the design is considered a Parato optimum and the procedure is terminated. Otherwise, index r by one and repeat steps 1 to 6 . The procedure will be terminated by step 4 , or 6, or by a minimum step size criteria which terminate the procedure when $\alpha_i{}^f$ is smaller than a minimum value e_5.

The above is a general purpose procedure for Parato

optimal search. A simplified approach can be used by considering only one of the f_q in Eq. (5.7). The ϵ_m restrictions of Eq.(4.8) force the minimization of all f_q automatically while only one of f_q is used in Eq. (5.7).

The σ_q value of equation (5.6) is to be maximized. This means the values of $S_i^T \cdot \nabla f_q$ in equation (54) are to be reduced to the maximum amount $-\sigma_q$ simultaneously. Then S_i is the direction which minimizes all f_q simultaneously.

The Eqs.(5.8-5.9) differ from that of Zoutendijk [16] in that the former tend to drive the design as necessary to a location estimated to be on the constraint boundary, rather than tending to move parallel to, or deflected away, from this boundary [19].

The penalty function of Eq. (5.2) is needed to allow the comparison in Eq. (5.15) of the desirability of an infeasible point for the purpose to determine if it is necessary to reduce step size α_i^f so as to avoid excessively large moves or to prevent oscillation [19]. For example, a fully constrained problem as illustrated in Fig. 3 where the number of active constraints equals the number of design variables the initial α_i^o would produce convergence without any need for step size reduction and without needs for design comparisons. For a problem such

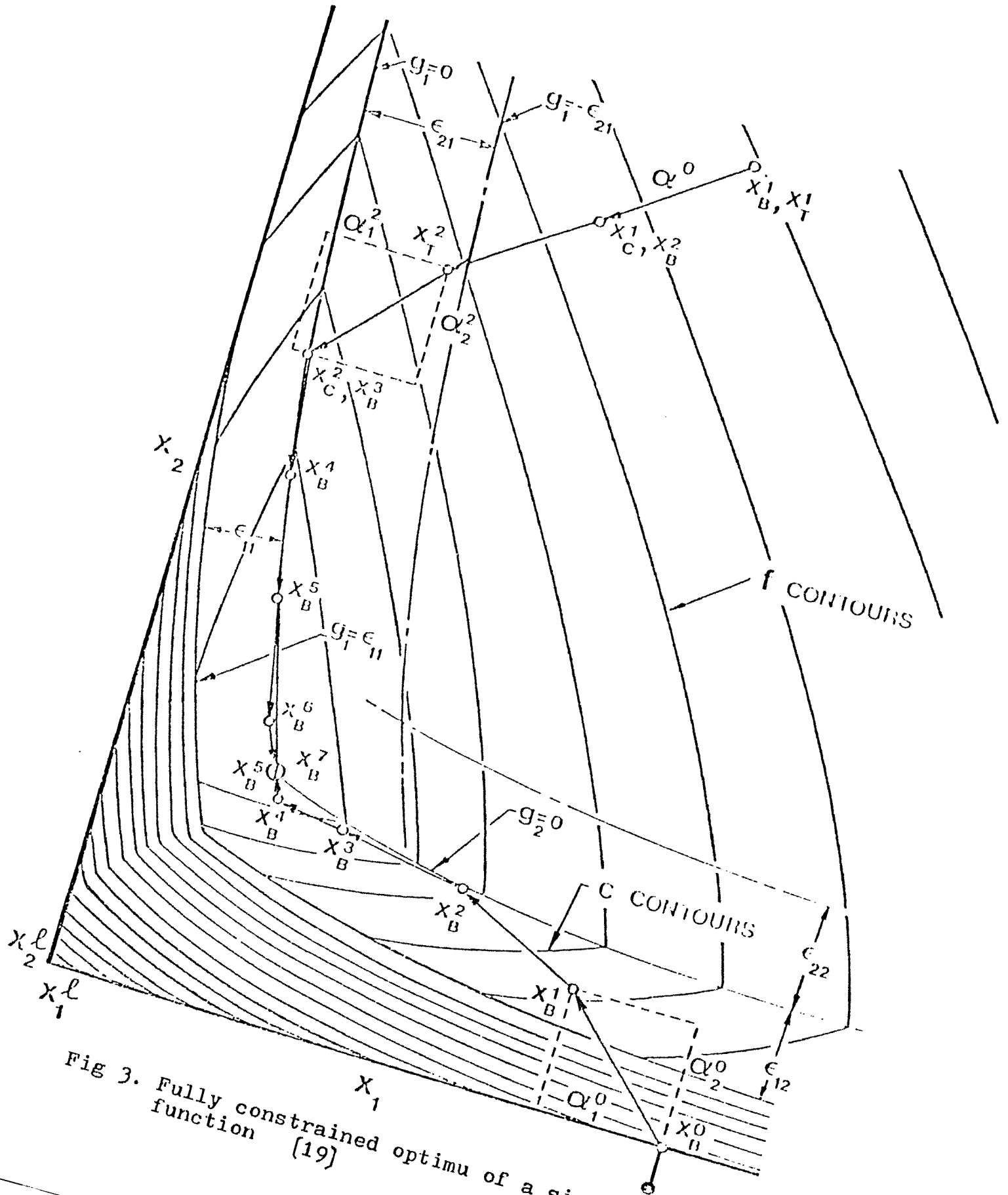


Fig 3. Fully constrained optimu of a single objective function (19)

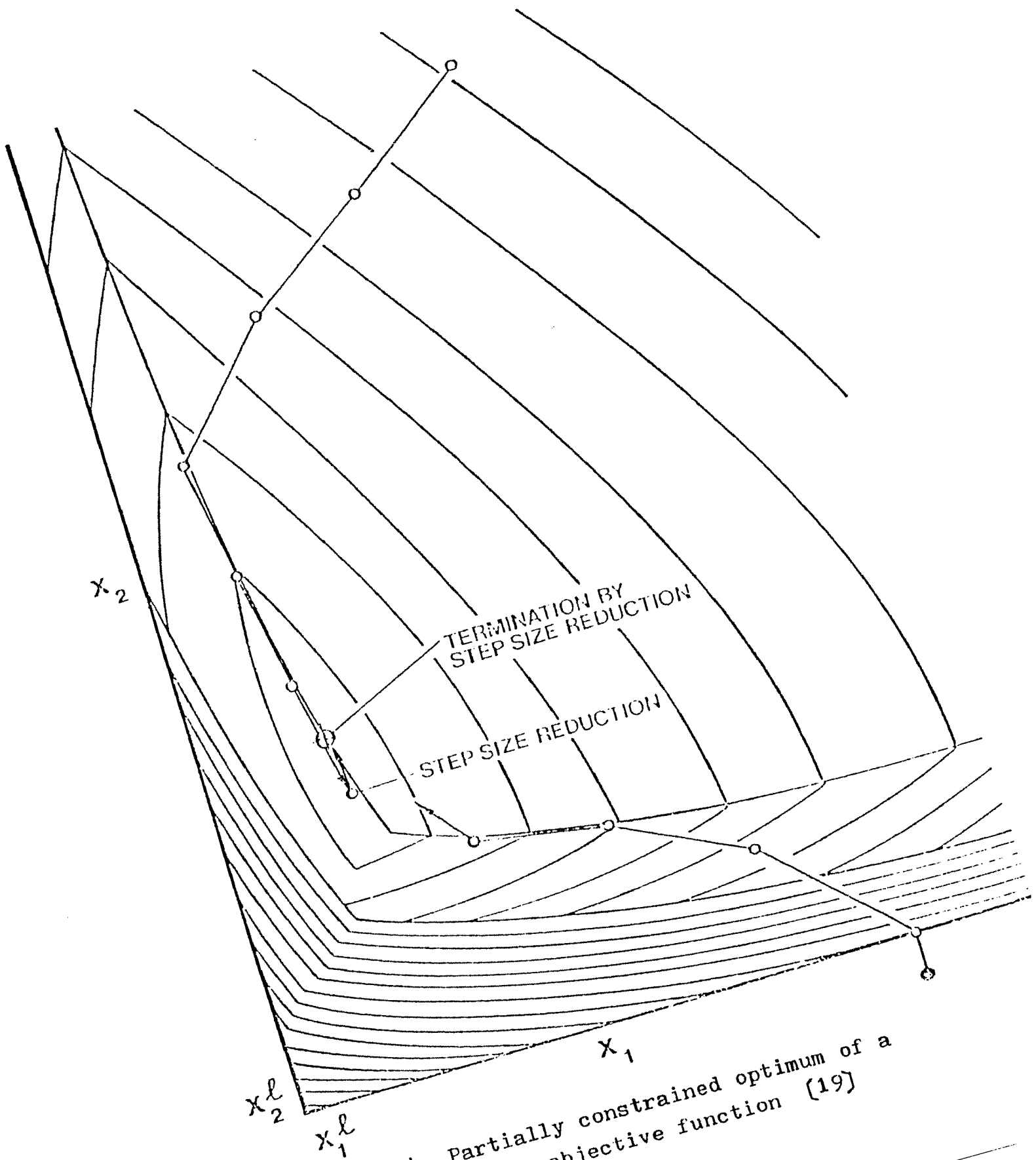


Fig 4. Partially constrained optimum of a single objective function [19]

as that illustrated in Fig. 4, however, oscillation about the optimum will result unless α_i is reduced. The penalty form is used in preference to the objective function alone for design comparison since a move which produces substantial constraint violation reduction with some increase in objective function value is generally more desirable than a move that produces the reverse situation. Furthermore, constraint violation must be considered in design comparison in methods which admit infeasible points if convergence to a feasible design is to be achieved.

CHAPTER VI

SATISFACTION OF CONSTRAINTS

The selection of a feasible starting point in inequality constrained problems, although often quite easy, can on occasion also be quite difficult and require much trial and error. Determining a near feasible starting point for problems with equality constraints is always difficult and can occasionally be essentially impossible. Thus an automatic procedure for the location of feasible points is of great utility, not only during the search process, but also for search initiation. The formulation of Chapter V can be utilized for such a procedure.

6.1 Satisfaction of Leveling and Other Equality Constraints

Satisfaction of the leveling constraints from a point that is near feasible with respect to the Behavior Constraints can be accomplished by solving a DFP where the ϵ_m are eliminated from Eq. (5.2) and replace the f_q of Eq. (5.1). To insure satisfaction of Eqs. (5.8) where the step size α_i is insufficient to reduce the ϵ_m to zero, slack variables σ_m are added. These slack variables must be minimized and ultimately vanish. The minimization of these variables must take procedence over any reduction in

the f_q as provided by Eq. (5.7) and thus a large weighting parameter W_1 is used with these slack variables so that their reduction dominates the DFP.

The search path of Chapter V can then be used to solve the resulting DFP. This solution will yield a point satisfying the Leveling Constraint equations.

The procedure is thus as follows :

1. For an infeasible base point $\underset{\sim}{x}_B{}^r$, evaluate the composite objective function value $C_m(\underset{\sim}{x}_B)$ where

$$C_m(\underset{\sim}{x}) = \epsilon_m(\underset{\sim}{x}) + P_m(\underset{\sim}{x}) \qquad m = 1,2,....Q-1 \tag{6.1}$$

$$P_m(\underset{\sim}{x}) = \max \left| \left(\lambda_{mj} \langle g_j(\underset{\sim}{x}) \rangle \right) \right| \tag{6.2}$$

$$\lambda_{mj} = \begin{cases} 2|\nabla\epsilon_m|^2/\nabla g_j \cdot \nabla\epsilon_m & g_j \le e_{j1} \\ K_1 & g_j > e_{j1} \end{cases} \tag{6.3}$$

all variables and bracket functions have the same definition as stated in section 5.3.

2. At point $\underset{\sim}{x}_B{}^r$ find σ_q, σ_m and S_i so as to

$$\text{maximize} \qquad \sum_{q=1}^{Q} \sigma_q - W_1 \sum_{m=1}^{Q-1} \sigma_m \tag{6.4}$$

subject to the conditions

$$(S_i)^T \cdot \nabla f_q(x_i) + \sigma_q \leq 0 \qquad q = 1,2,....Q \tag{6.5}$$

$$(S_i)^T \cdot \nabla \epsilon_m(x_i) - \sigma_m + \epsilon_m(x_i) = 0 \quad m = 1,2,..Q-1 \tag{6.6}$$

$$(S_i)^T \cdot \nabla g_j(x_i) + g_j(x_i) \leq 0 \qquad j \in J_a, \quad g_j(x_i) > -e_{j2} \tag{6.7}$$

$$\sigma_q \geq 0 \quad \text{or} \quad \sigma_q < 0 \tag{6.8}$$

$$0 \leq \sigma_m < \epsilon_m \tag{6.9}$$

$$(S_i)^l \leq S_i \leq (S_i)^u \tag{6.10}$$

Here σ_q and σ_m are slack variables, W_1 is a weighting factor, and the other variables are defined as chapter V. The set J_a contains the active constraints for all inequality constraints which are greater than " Near Constraint Band Width " $-e_{j2}$. The concept in [29] so as to greatly improve the converge power is used to choose e_{j2}. The S_i^l and S_i^u are lower and upper limits on S_i^r which are given by

$$S_i^l = \begin{cases} x_i^l - x_i \text{ if } x_i - x_i^l < \alpha_i^\epsilon \\ -\alpha_i^\epsilon \quad \text{, otherwise} \end{cases} \tag{6.11}$$

$$S_i^u = \begin{cases} x_i^u - x_i \text{ if } x_i^u - x_i < \alpha_i^\epsilon \\ \alpha_i^\epsilon \quad , \text{ otherwise} \end{cases}$$

where α_i^ϵ is the step size for the reduction of ϵ_m. It is a specified maximum limit on the change in variables S_i.

3. Define a comparison base

$$\underset{\sim}{x}_C^r = \underset{\sim}{x}_B^r + \underset{\sim}{S}^r \tag{6.12}$$

4. if $[\ C_m(\underset{\sim}{x}_C^r)\ ]_{max} > [\ C_m(\underset{\sim}{x}_B^r)\ ]_{max}$ (6.13)

call $\underset{\sim}{x}_B^{r+1} = \underset{\sim}{x}_B^r$ and increase r by 1 and repeat step 2 to 4 with α_i^ϵ halved.

5. Otherwise call $\underset{\sim}{x}_B^{r+1} = \underset{\sim}{x}_C^r$, (6.14)

Increase r by one and repeat step 2-5. Continue the process until all $\epsilon_m < e_6$ are satisfied.

The DFP procedure of Eqs. (6.4-6.11) is formulated to reduce ϵ_m to zero. During this procedure, the f_q may be also reduced, or may be increased with minimum amounts. In Eq. (6.6) of the DFP, the value of $-(S_i)^T \cdot \nabla \epsilon_m(x_i)$ is the estimated amount reduction in ϵ_m after a move in S_i direction. The smaller the value of σ_m, the greater the

value of the reduction in ϵ_m. Thus, σ_m must be minimized for a maximum reduction of ϵ_m. The second term of Eq.(6.4) is used for this purpose. The same principle can also be applied to explain the function of the σ_q in Eqs. (6.4-6.5). Here, the maximization of σ_q will maximize the reduction of f_q (if $\sigma_q > 0$), or minimize any increase of f_q (if $\sigma_q < 0$).

The Eq. (6.4) is used to minimize all ϵ_m simultaneously. A move with a large step size, however, may cause search failure since the assumption of local linearity may not be sufficiently valid. The step size reduction strategy can overcome this problem but at the expense of substantially increasing the number of iterations. One can avoid this difficulty by choosing the largest ϵ_m to make design comparison [19]. Thus, the Eq.(6.13) is using this approach.

The same procedure can, of course, be used for any equality constraint by simply replacing the ϵ_m and σ_m with the h_k and with the σ_k.

6.2 Satisfaction of Inequality Constraints

A similar approach can be used for locating near

feasible point ($g_j \leq e_{j7}$) from points with substantial ($g_j \geq e_{j7}$) violation of the inequality constraints. Here one simply replaces the f_q with those $g_j > e_{j7}$ and introduce the DFP. The procedure of section 6.1 can now be used. The procedure however can be terminated as soon as all constraints are within the excessive constraint violation band width, that is all

$$g_j \leq e_{j7}$$

since such a point is sufficiently near the Feasible-Infeasible Boundary to be considered a near feasible point.

6.3 General Procedure for Constraint Satisfaction

The procedure of Section 6.1 and 6.2 can be combined to locate a near feasible point satisfying the equality constraints. This is accomplished by including each substantially violated inequality constraint and all equality constraints in the constraint reduciton DFP. The search procedure of section 6.1 is then invoked until cnvergence is achieved with respect to the equality constraints and all the inequality constraints are within the excessive constraint violation band width.

CHAPTER VII

ALGORITHM CONTROL PARAMETER SELECTION

The performance of the previous procedure depends on the selection of the parameters e_{j1}, e_{j2}, e_3, e_4, e_5, e_6, e_{j7}, K_1, α_i^f, and α_i^ϵ. Some parameters, are designated by the user, are fixed, or are computed in the procedure. The successful experience of CADOP5 [19] in the selection of these parameter warrants their adoption.

(1) Step size and reduction attempt

Step size α_i is a specified maximum limit on the change in variable x_i. In this procedure, the initial step size α_i^ϵ for boundary restoration search and the initial step size α_i^f for Parato optimality search are given by

$$\alpha_i^\epsilon = \epsilon_m^*(\underset{\sim}{x}_B)\,[\,|\max(\epsilon_m^*,_i)|\,/\,|\nabla\epsilon_m^*|^2\,] \qquad (7.1)$$

$$\alpha_i^f = f_q^*(\underset{\sim}{x}_B)\,[\,|\max(f_q^*,_i)|\,/\,|\nabla f_q^*|^2\,] \qquad (7.2)$$

where ϵ_m^* is associated with the m producing max ϵ_m, f_q^* is associated with q producing max f_q, and η is arbitrary selected. Here η is the estimated fraction change in ϵ_m^* or f_q^* if a move where made the $\nabla\epsilon_m^*$ or ∇f_q^* direction with components limited to α_i^ϵ or α_i^f respectively. Thus

η may be thought of as attempted objective function reduction where $\eta = 0.5$ would be an attempt at a 50 % reduction. The actual reduction would usually be substantially less than estimated since the actual move would be deflected away from the function gradient direction by the active constraints. A value of 0.5 for η is recommended.

(2) Convergence parameters

In the boundary restoration search, all ϵ_m equal or near zero are assumed convergence. An abitrary small number e_6 is used to define this convergence. Experience shows that $e_6 = 0.001$ is small enough to allow the Parato optimality search to reach the POS at a level of all $\epsilon_m \leq 0.001$. In Parato optimality search, if the magnitute of S , the step size , or the objective reduction between step size reduction is sufficiently small, convergence to the optimum is assumed. An arbitrary small number e_3 is used to define the convergence for S, an arbitrary small number e_4 is used to define the convergence for the objectives, and another arbitrary small number e_5 is used to define the convergence for the minimum step size.

(3) Constraint linearity band width parameter

As penalty function, $P(x_i)$, is based on the assumption of local linearity [31], thus a limit (e_{j1}) must be set, depending on the nonlinearity of g_j or ϵ_m, on applying the symmetric penalty function $P(x_i)$. A small number for e_{j1} (or e_m) = 0.1 has been found satisfactory after years of use code CADOP5 [19] where the constraints are given in a non-dimensional form,

$$g_j(x_i) = (B_j - U_j)/ U_j \quad U_j \neq 0 \qquad (7.3)$$

where B_j represents the controlled behavior and U_j the upper limit on behavior, even for highly nonlinear functions. For ϵ_m constraints in Parato optimality search, the value of e_6 as mentioned in section 7.2 defines the constraint violation.

(4) Penalty function constant

The K_1 parameter is used to assign penalty that is proportional to the degree of constraint violation to points outside the Near Satisfaction Band Width. The penalty must be large enough so that a move which reduces constraint violation will always produce a lower value of the composite objective function even where such a move increases the f_q. Thus K_1 is made arbitrarily large. A value of 10^4 has been

found to be satisfactory for almost all problems during extensive use of CADOP5. A larger value may be necessary if the value of any objective function begins to approach the value of K_1. This parameter should be at least an order of magnitude larger than the largest magnitude of the objective functions.

(5) Constraint band width parameter

A band parameter width e_{j2} is required to reduce the DFP computational effort by excluding obviously inactive constraints. The e_{j2} can be defined by

$$e_{j2} = N \min_{j \in J_p} \left(\alpha_i^r / g_{j,i}(x_i) \right) \; g_j^{\,2} \qquad (7.4)$$

This band width, based on the assumption of local linearity, is defined such that all constraints which could be violated by a move in the ∇g_j direction are included in the DFP. All constraints not in the potentially active set J_p are ignored with respect to the DFP. N > 1 can be used to help account for nonlinearity. The potentially active constraints are all those within a band width double the largest band width of the constraints in the previous DFP. Initially the potentially active band width is arbitrarily selected with a value equal to 2 where the g_j are given in the form of Eq.(7.3) and is as defined above. A value for

N=1 has been found to be satisfactory and is recommended since the band width will usually avoid violation of inactive constraints and since infeasible designs are admissible.

The value of e_{j7} determines the condition of the excessive constraint violation. The FR procedure for inequality constraint should be started for a point outside this range. The value of e_{j7} can be established similarily as e_{j2} except that here a value of $N = 1/N_v$ where N_v is the number of constraints violated is recommended [19].

(6) Weighting parameter

It has been suggested in Ref.[1] that a value of $W_1 \geq 10^5$ will be satisfactory to define the relative importance between σ_q and σ_m. This large value cuases the σ_q term in Eq.(6.4) to be ignored due to the round off error and causes programming inefficiency. Because the search direction is unpredictable, one cannot estimate this value exactly. This thesis adopts a initial calculation and then adaptively modifies it from experience. Further investigation is need to better define user parameter. The value of W_1 is computed from Eq. (7.5). Examining this equation it may be seen that W_1 is M times the estimated

ratio of the greatest possible reduction of the ϵ_m to the greatest possible reduction of the f_q. The constant M is used to adjust this calculation if needed. Experience shows that M = 2, or 3 works well. An inadequate weighting factor will fail to locate the feasible region and a larger magnification factor M will be needed.

$$W_1 = M \left(\sum_{q=1}^{Q} \left| \sum_{i=1}^{I} \alpha_i^{\epsilon} f_{q,i} \right| \right) / \left(\sum_{m=1}^{Q-1} \left| \sum_{i=1}^{I} \alpha_i^{\epsilon} \epsilon_{m,i} \right| \text{ or } \sum_{m=1}^{Q-1} \epsilon_m \right)_{min} \quad (7.5)$$

CHAPTER VIII

EXAMPLES

8.1 Mathematical Test Problem

(a) Unconstrained problem with two objective functions

This example shows how Eq. (4.9) can generate the same POS as the arithmetic weighting method of section 3.5 and how the leveling constraint of Eq. (4.11) is used. The objective functions are

$$f_1(x_i) = (9/25\ x_1^2 + x_2^2\ )^{0.5} + 2 \qquad (8.1)$$

$$f_2(x_i) = (x_1 - 8\ )^2/\ 9 + (x_2 - 8\ )^2\ /\ 16 + \ \ 4 \qquad (8.2)$$

The weighting factors for the arithmetic weighting method are chosen as $w_1 = 1.0$ and $w_2 = 2.0$. Thus the composite function is :

$$C[f_q(x_i)] = \min\ [\ f_1 + 2.0\ f_2\ ] \qquad (8.3)$$

From the POS of Eq. (8.3), one finds $f_1/f_2 = 1.8347$. Therefore, the corresponding weighting factors of the the level function formulation of Eq. (4.9) are chosen as $A_1 = 1.0$ and $A_2 = 1.8347$ to illustrate the procedure of this thesis and to compare it with the arithmetic weighting method. Thus the composite function is :

$$C[f_q(x_i)] = \min\ [\ f_1\ ,\ 1.8347\ f_2\ ] \qquad (8.4)$$

Equation (8.3) will be solved by CADOP5 as a single unconstrained function and Eq. (8.4) by the new computer code CADOP8, based on the procedure presented in this thesis.

(b) Example problem with two objective and one constraint functions

In this example , the two objectives f_1 and f_2 of example 8.1 (a) are used , but a constraint equation

$$g_1(x_i) = (x_1)^2 + (x_2 - 10)^2 - 49 \leq 0 \qquad (8.5)$$

is added. The equation to be minimized is the same as Eq. (8.3).

For unconstrained problem, the POS is located at a point whereobjectives compete. In constrained problem, however, the POS is frequently on an active constraint [6]. This example illustrates the latter case

(c) Unconstrained problem with three objective functions

In addition f_1 and f_2 above, a third objective function

$$f_3(x_i) = [(x_1 - 8)^2 + x_2^2)]^{0.5} + 4 \qquad (8.6)$$

is added. This example shows how the Eq. (4.3) can be

solved with an arbitrary weighting factor set $A_1 = 1.$, $A_2 = 1.8347$, and $A_3 = 1.0052$. These weighting factors are chosen from a range of values that will produce a POS as described in chapter 9. The compositive form will be :

$$C[f_q(x_i)] = \min [\ f_1\ ,\ 1.8347\ f_2\ ,\ 1.0052\ f_3\ ] \qquad (8.7)$$

The purpose of this example is to show how the feasibility of the equality constraint ϵ_1 and ϵ_2 is established by the procedure developed herein.

8.2 Design Example - Optimal Frequency Separation Problem of " T " Ring Stiffened Cylindrical Shells

Bronowicki et al [32] pose a frequency separation problem which maximimizes the separation between the lowest two natural frequencies of vibration for a " T " ring stiffened cylindrical shell under hydrostatic pressure. The coalesence of vibration modes as the optimum is approached requires simultaneous separation of several frequencies [14]. Thus this problem is a typical maximum performance type of level function problem.

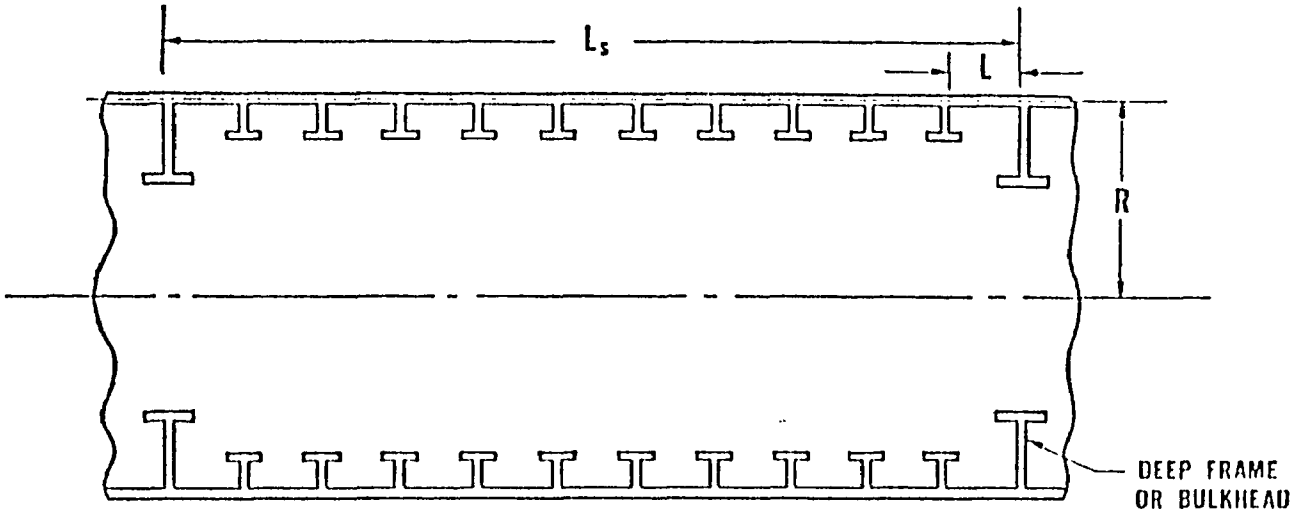
This problem is of the form of Eq. (4.1) or Eq. (4.2) and can be stated as :

Maximize $C[{}_q f(x_i)] = (\omega_{1+q} - \omega_1)$ $q = 1,2,....Q$ (8.8)

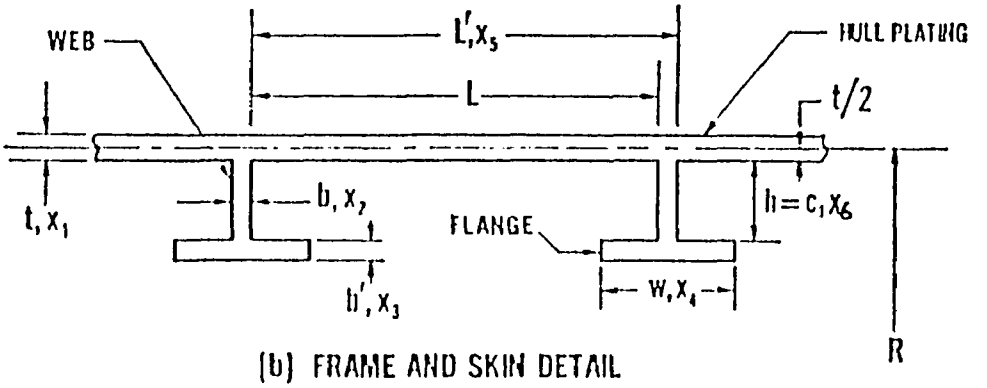
subject to the behavior constraints of [14,32].

This problem is the type A problem of Ref.[14], which has the same equations as the type (II) problem of Ref.[32]. The structure of this cylindrical shell is shown as Fig. 5 . The objective function here is given by equation (8.8). The constraints used are : g_1 = gross (general) buckling, g_2 = shell (internal) buckling, g_3 = shell yielding, g_4 = stiffener yielding, g_5 = stiffener flange buckling, g_6 = maximum flange thickness, g_7 = minimum flange width, g_8 = minimum internal or maximum external radius, g_9 = minimum natural frequency, g_{10} = maximum weight, and g_{11} = web buckling. The constraint equations used are the same as reference [14,32]. The six design variables are shown in Fig. 5 .

The behavior subroutines of SBSHL7 program used in Ref.[14] are called by the computer code CADOP8 to compute the frequency and constraint values. Based on the experience of Ref.[14] , four frequencies (three objectives) are nearly active. Thus, one can test this example with Q = 3.



(a) HULL SEGMENT CROSS-SECTION



(b) FRAME AND SKIN DETAIL

Fig 5. Typical shell cross-section [14]

CHAPTER IX

DISCUSSION OF RESULTS

A new computer program CADOP8 is used to test the algorithms presented here on the two types of test problems of Chapter VIII. The first problem type consists of two variable mathematical functions. For these examples, one can search and map the functions in two dimensional space and thus readily observe algorithm performance. The second problem type is a six variable engineering design problem [14] using complex equations subjected to many complex constraints. The complex multiple active objective functions and constraints of this problem present a rigorous test for the algorithm described herein.

9.1 Two Variable Mathematical Test Problem

The search path for the unconstrained two variable problem generated by CADOP8 and CADOP5 is given in Table 1 and plotted in Fig.6 and Fig. 7 respectively . In comparing the search paths it may be seen that in both instances convergence to a point where the objective function or functions are within 1% of the optimum occurs rapidly. Most of the search effort is associated with oscillation about the optimum. Oscillation is less pronounced with the level function method due to the use of the leveling constraint. This constraint tends to restrict movement to an one dimensional search along the $\epsilon_1 = 0$ line

	Arithmetic Weighting Method (Solved by CADOP5)				Level Function Method (Solved by CADOP8)					
n_b	n_f	$C[f(x_i)]$	x_1	x_2	n_f	f_1	f_2	x_1	x_2	ϵ_1
0	6	27.419	1.	6.	6	8.023	9.694	1.	6.	9.757
1	12	19.567	5.4	2.5	12	9.474	7.586	2.341	7.341	4.444
2	18	18.685	8.7	6.4	18	10.685	6.082	3.681	8.399	0.475
3	* 34	18.076	7.622	6.	* 24	9.657	5.383	4.543	7.059	0.311
4	* 46	18.066	6.412	4.176	32	9.112	5.029	5.207	6.388	0.115
5	* 58	17.885	7.293	4.826	38	8.735	4.820	5.878	5.738	0.109
6	* 72	17.874	7.058	4.965	44	8.550	4.710	6.548	5.241	0.092
7	* 84	17.873	7.194	4.955	* 50	8.509	4.604	7.218	4.860	0.085
8	* 100	17.872	7.112	4.970	115	8.551	4.661	7.111	4.971	0.000
9	* 112	17.873	7.120	4.967						
10	128	17.873	7.119	4.967						

n_b : Base number n_f : number of objective function evaluation

* : Reduced step size

Table 1 Arithmetic weighting method and level function method search paths for the two variable unconstrained problem

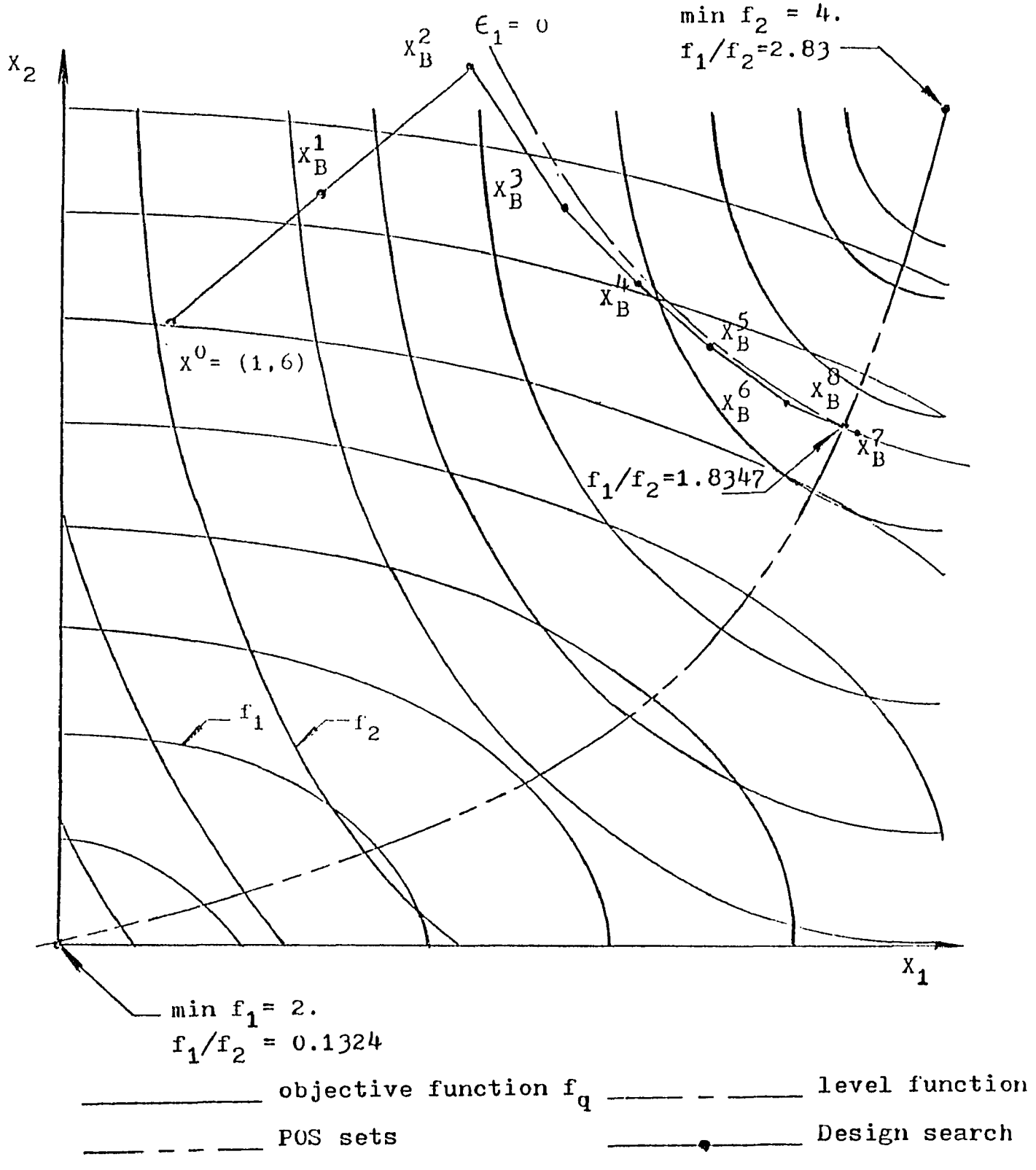


Fig. 6 CADOP8 search path for the unconstrained two objective function problem

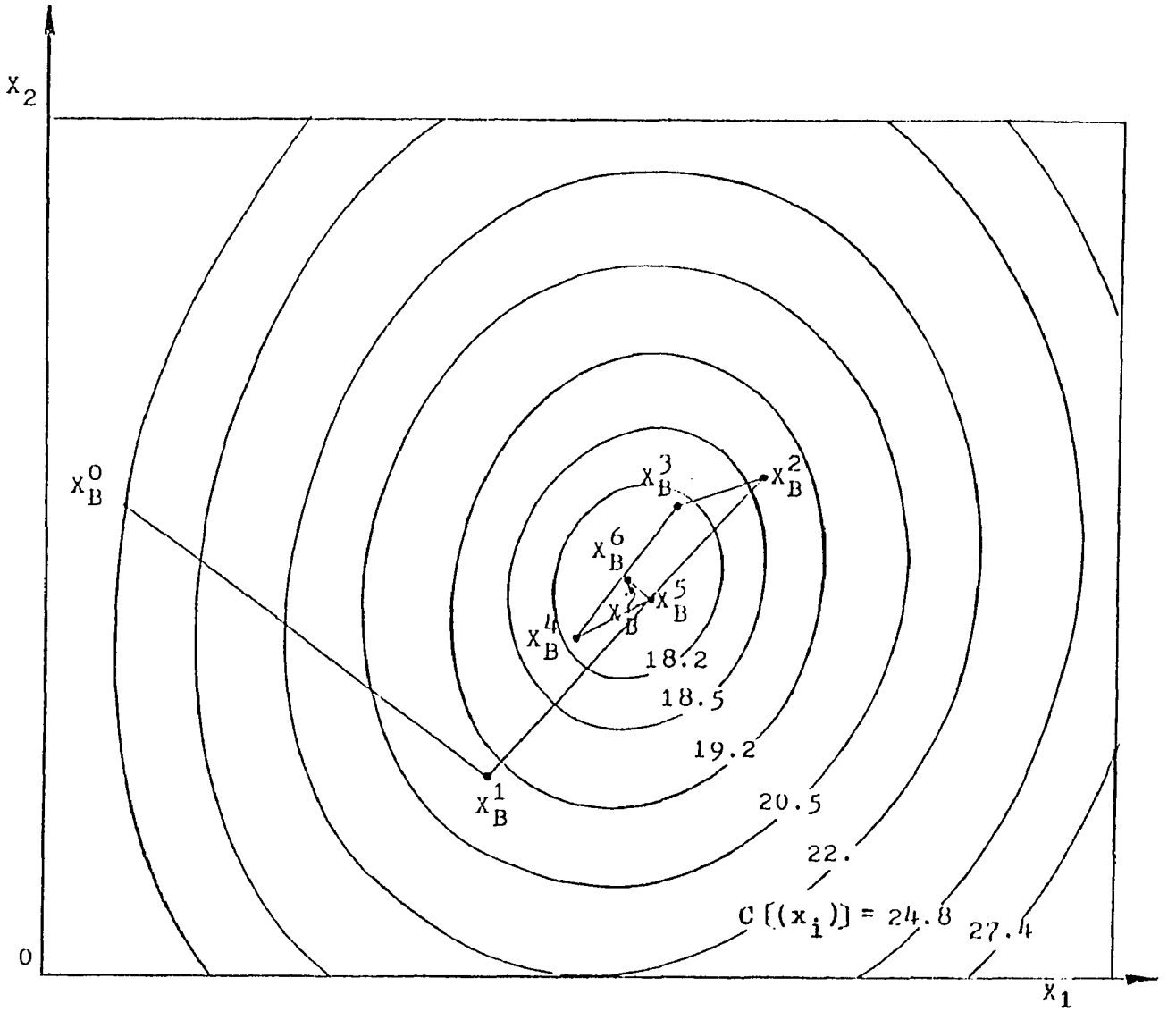


Fig.7 CADOP5 search path of arithmetic weighting form

thus producing a more direct search path. The total computational effort associated with both methods is essentially the same as indicated by the number of objective function evaluations required for convergence. The level function method requires fewer steps but more computational effort at each step.

The large initial oscillation associated with the arithmetic weighting method is the result of an excessively large initial step size selected by CADOP5. This code selects a step size essentially inversely proportional to the magnitute of the objective function gradient. The relative flatness of the composite objective function of the arithmetic weighting problem form produces the large initial step size.

At the optimum of the arithmetic weighting function problem $f_1/f_2 = 1.8347$. Thus $A_1 = 1.$ and $A_2 = 1.8347$ were chosen as the weighting parameters of the level objective problem so that the optimal points are the same for both problems. Thus a more direct comparison is possible and to demonstrate the ability of the level objective function formulation and CADOP8 to locate a POS.

It may be seen from Fig. 6 that the POS runs from a point where f_1 is a minimum (min f_1) to a point where f_2 is a minimum (min f_2). At min f_1 the ratio of f_1 to f_2 is

0.1324 and at min f_2 this ratio is 2.83. Thus one can search for POS sets using the level function method by selecting A_2/A_1 in the range from 0.1324 to 2.83. Unfortunately one must know this range if one is to locate a POS. Values of A_1 and A_2 selected outside this range will not produce a POS. This property is a limitation of the level function formulation in solving conventional multiple objective function problems. Where one wishes to map the POS region this limitation is not serious since the POS region end points and thus appropriate A_{re} are easily determined. The primary consideration with respect to this property is that the solution of a level function problem is not in general a POS.

The search paths for the level function formulation of the constrained, two objective function problem and the unconstrained three objective function problem are given in Table 2 and illustrated in Figs. 8 and 9 respectively. The initial path of constrained, two objective problem is essentially the same as the unconstrained form of this problem. The presence of the additional constraint however results in a fully constrained problem and thus rapid convergence without oscillation. The three objective problem is also fully constrained by virtue of the two function leveling constraints and thus similar rapid convergence to the optimum occurs. This convergence is much faster than the arithmetic weighting form on these problems where

	Constrained Two Objective Function Problem						Unconstrained (**) Three Objective Function Problem						
n_b	ϵ_1	g_1	f_1	f_2	x_1	x_2	ϵ_1	ϵ_2	I_1	I_2	f_3	x_1	x_2
0	9.757	-33.00	8.023	9.694	1.	6.	9.757	5.259	8.023	9.694	13.22	1.	6.
1	4.444	-40.13	9.474	7.586	2.341	7.341	1.543	2.894	9.565	6.064	12.41	3.727	7.245
2	0.475	-39.25	10.69	6.082	3.681	8.399	1.609	1.106	7.828	5.144	8.888	6.135	4.581
3*	0.311	-27.80	9.567	5.383	4.543	7.059	0.369	0.164	8.395	4.786	8.541	7.581	4.494
4	0.115	-18.25	9.112	5.029	5.207	6.388	0.030	0.016	8.633	4.721	8.603	7.958	4.603
5	0.109	-7.044	8.735	4.820	5.878	5.738	0.000	0.000	8.656	4.718	8.611	7.999	4.611
6*	0.022	-1.838	8.668	4.737	6.213	5.529							
7	0.003	0.0183	8.648	4.715	6.213	5.529							
8	0.000	0.0000	8.649	4.714	6.327	5.459							

* : Reduced step size ** : No step size change

Table 2 CADOP8 search paths for constrained two objectivefunction problem and unconstrained three objective function problem

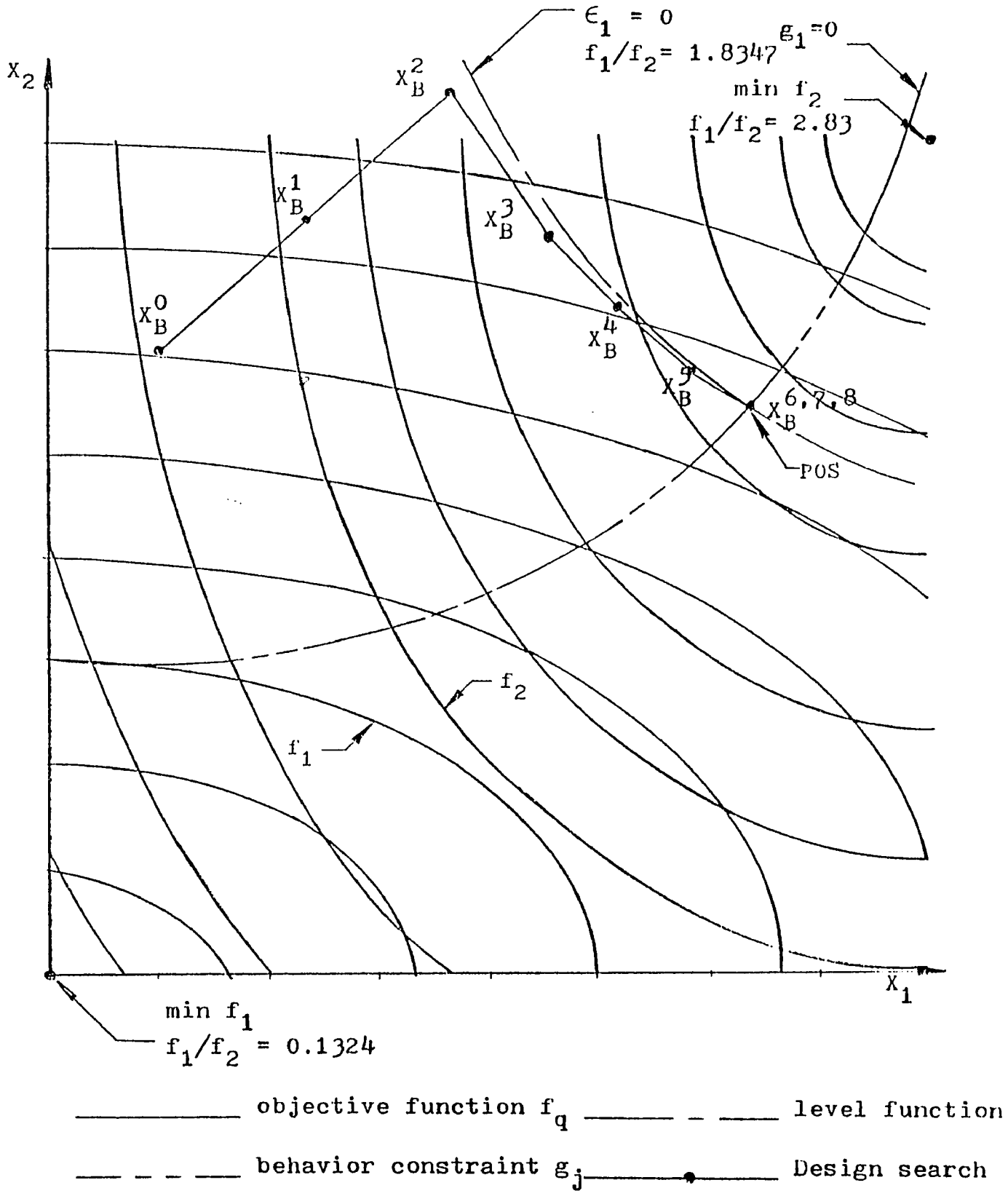


Fig. 8 CADOP8 search path for the constrained two objective function problem

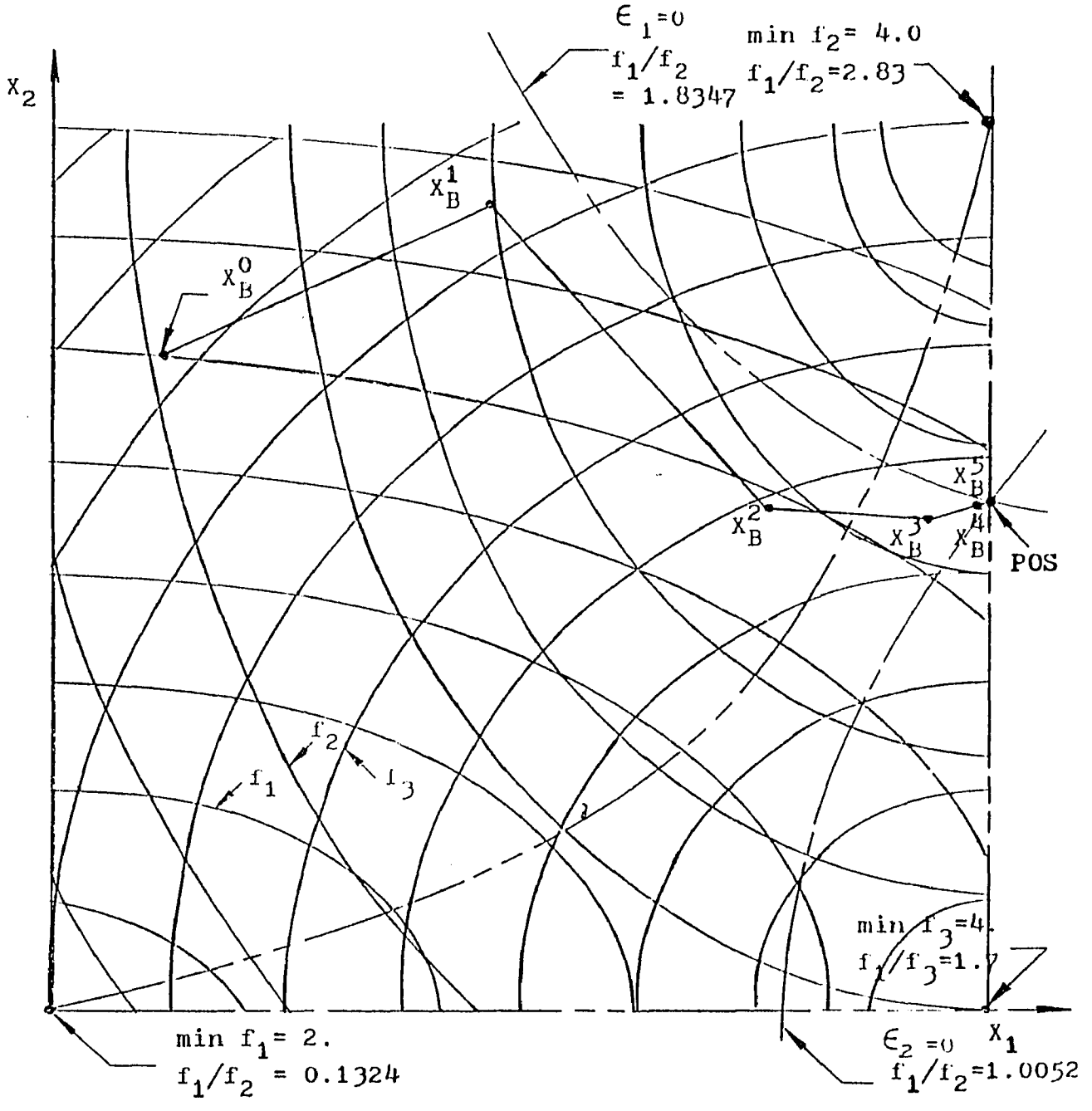


Fig. 9 CADOP8 search path for the unconstrained three objective function problem

oscillation about the optimum occurs.

9.2 The Six Variable Design Problem

Table 3 compares the results of optimization runs of the type A six variable problem discussed in chapter VIII using the SUBSHL7 program of Ref.[14] and CADOP8. The CADOP8 program for this trial utilizes the behavior subroutines of SUBSHL7 to compute the frequency and constraint values . Since SUBSHL7 only considers the four lowest frequencies, the level objecitve function formulation of this problem employees three frequency separation objective functions involving the separation of the first and the second, first and third, and first and fourth frequencies.

Although SUSHL7 locates a design with a greater frequency separation CADOP8 has greatly superior convergence properties. The CADOP8 code requires in exess of an order of magnitude less computational effort in achieving convergence superior to SUBSHL7. At the termination of the CADOP8 run, the second, third, and fourth frequencies are identical while using SUBSHL7 the fourth frequency is slightly higher than the second and third. Furthermore, at termination of the CADOP8 run there are four active behavioral constraints and two active objective function leveling constraints indicating a fully constrained solution

	Initial Design	SBSHL7 Termination	CADOP8 Termination
x_1	1.000	1.519	1.188
x_2	1.000	0.523	0.325
x_3	1.000	1.498	1.188
x_4	10.000	4.996	15.038
x_5	20.000	55.169	30.616
x_6	10.000	30.758	18.268
g_1	-0.539	-0.790	-0.661
g_2	-0.472	-0.306	-0.383
g_3	0.001	-0.116	-0.000
g_4	-0.389	-0.467	-0.416
g_5	-0.947	-0.995	-0.904
g_6	0.000	-0.014	-0.000
g_7	-10.000	-4.996	-15.038
g_8	-0.598	-0.545	-0.579
g_9	-0.534	-0.717	-0.676
g_{10}	0.035	-0.001	0.0000016
g_{11}	-0.967	-0.000	-0.000
$\omega_1(n_1m_1)$	25.76(2,1)	30.458(2,1)	37.065(2,1)
$\omega_2(n_2m_2)$	33.04(3,1)	72.897(3,1)	64.363(3,1)
$\omega_3(n_3m_3)$	44.29(3,2)	72.897(1,1)	64.363(2,2)
$\omega_4(n_4m_4)$	57.00(2,2)	72.924(13,1)*	64.363(1,1)
f_1	7.279	30.458	27.298
f_2	18.531	30.458	27.298
f_3	31.242	30.485	27.298
ϵ_1	11.252	0.00006	0.00021
ϵ_2	23.963	0.02659	0.00023
n_i	0	1379	117
n_f	1	27934	844
n_g	11	59723	4532

n_i : number of design iterations
n_f : number of objective function evaluations
n_g : number of constraint function evaluations
n : number of circumferential full waves
m : number of axial half waves
* : mode associated with shell (inter-ring) vibration

Table 3. SUBSHL7 and CADOP8 search paths from the starting point of Ref. [14]

and therefore complete convergence to a local optimum. Unfortunately, this local optimum is inferior in frequency separation to that located by the earlier code. At SUBSHL7 termination there are only two active behavioral constraints indicating incomplete convergence.

To further demonstrate the superior convergence of the level function approach on this problem additional runs were made with SUBSHL7 and CADOP8 from two points near the best located by the initial SUSHL7 run. The results of these runs are shown in table 4 and 5. SUBSHL7 terminated without significant improvement in convergence. CADOP8 on the otherhand quickly generated designs with improved convergence.

The failure of CADOP8 to move to a fully constrained design in these latter runs is the result of the failure to consider a fifth natural frequency $\omega(2,2)$ which is also active. This leads to the search failure due to switching of critical modes as discussed in chapter IV. The SUSHL7 behavior subroutines must be modified to consider all active frequencies in order to fully solve this problem.

9.3 General Observations

In observing the behavior of CADOP8 on all the test problems it should be noted that the procedure for boundary

	Initial Design	SBSHL7 Termination	CADOP8 Termination
x_1	1.5203	1.5237	1.5064
x_2	0.5464	0.5467	0.5480
x_3	1.5203	1.5237	1.5064
x_4	3.7510	3.8528	3.8080
x_5	54.6339	55.5451	54.5654
x_6	32.0743	32.1279	32.1228
g_1	-0.7911	-0.7919	-0.7875
g_2	-0.3161	-0.3506	-0.3009
g_3	-0.1190	-0.1191	-0.1099
g_4	-0.4642	-0.4670	-0.4628
g_5	-0.9975	-0.9973	-0.9973
g_6	-0.0000	-0.0000	-0.0000
g_7	-3.7510	-3.8528	-3.8080
g_8	-0.5417	-0.5415	-0.5416
g_9	-0.7612	-0.7167	-0.7169
g_{10}	-0.0000	-0.0001	-0.0063
g_{11}	0.0000	-0.0028	0.0000
$\omega_1(n_1 m_1)$	42.2877(2,1)	42.3573(2,1)	42.3882(2,1)
$\omega_2(n_2 m_2)$	72.4108(3,1)	72.5438(13,1)*	72.8315(13,1)*
$\omega_3(n_3 m_3)$	72.9524(1,1)	72.5506(3,1)	72.8315(1,1)
$\omega_4(n_4 m_4)$	74.8315(2,2)	73.0272(1,1)	72.8315(3,1)
f_1	30.1231	30.1865	30.4433
f_2	30.6647	30.1933	30.4433
f_3	32.5438	30.6699	30.4433
ϵ_1	0.5415	0.0068	0.00000490
ϵ_2	2.4207	0.4834	0.00000668
n_i	0	10	15
n_f	1	294	94
n_g	11	637	472

n_i : number of design iterations
n_f : number of objective function evaluations
n_g : number of constraint function evaluations
n : number of circumferential full waves
m : number of axial half waves
* : mode associated with shell (inter-ring) vibration

Table 4. SUBSHL7 and CADOP8 search paths from the starting point nearby the optimum of Ref.[14], first set

	Initial Design	SBSHL7 Termination	CADOP8 Termination
x_1	1.518	1.5182	1.5208
x_2	0.539	0.5459	0.5474
x_3	1.331	1.5125	1.5208
x_4	4.826	3.9533	3.8646
x_5	55.040	55.0233	55.2650
x_6	31.690	32.0522	32.2157
g_1	-0.7904	-0.7906	-0.7913
g_2	-0.3070	-0.3076	-0.3067
g_3	-0.1163	-0.1165	-0.1176
g_4	-0.4664	-0.4661	-0.4670
g_5	-0.9941	-0.9971	-0.9973
g_6	-0.1232	-0.0037	0.00000000
g_7	-4.8260	-3.9533	-3.8646
g_8	-0.5433	-0.5418	-0.5413
g_9	-0.7172	-0.7173	-0.7174
g_{10}	-0.0007	-0.0007	0.00000105
g_{11}	-0.0008	-0.0032	-0.00000149
$\omega_1(n_1m_1)$	42.4387(2,1)	42.4444(2,1)	42.4680(2,1)
$\omega_2(n_2m_2)$	72.8996(1,1)	72.9103(3,1)	72.9464(3,1)
$\omega_3(n_3m_3)$	72.9002(3,1)	72.9104(1,1)	72.9469(1,1)
$\omega_4(n_4m_4)$	73.2041(13,1)*	73.2824(13,1)*	72.9470(13,1)*
f_1	30.4610	30.4658	30.4784
f_2	30.4615	30.4660	30.4789
f_3	30.7654	30.8380	30.4790
ϵ_1	0.000552	0.00016	0.000510
ϵ_2	0.304444	0.37202	0.000575
n_i	0	484	12
n_f	1	8803	79
n_g	11	19167	500

n_i : number of design iterations
n_f : number of objective function evaluations
n_g : number of constraint function evaluations
n : number of circumferential full waves
m : number of axial half waves
* : mode associated with shell (inter-ring) vibration

Table 5. SUBSHL7 and CADOP8 search paths from the starting point nearby the optimum of Ref. [14], second set

restoration worked well in all cases. From Figs. 6, 8 and 9, for example, one can see that the procedure moves quickly near the $\epsilon_m = 0.$ line(s), the feasible region, and then essentially along that line, or lines, to the optimum. The same performance can be observed from Table 6 where movement to the $\epsilon_m = 0.$ constraint occurs by the 68^{th} iteration (base No. 66). Thus the procedure seems effective in locating the feasible region from an infeasible starting point even where difficult nonlinear equality constraints are used.

Base No.	Objective Function			Level Function		Active Constraint Number
	f_1	f_2	f_3	ϵ_1	ϵ_2	
0[a]	7.279	18.531	31.242	11.252	23.963	3,6,10
1	8.747	19.619	31.333	10.871	22.587	
2	10.311	20.414	30.617	10.102	20.306	
3	11.855	21.254	29.929	9.399	18.074	
4	13.385	22.116	29.220	8.731	15.835	
5	14.902	22.999	28.494	8.097	13.592	
6	16.406	23.901	27.753	7.494	11.347	
7	17.896	24.818	26.997	6.922	9.101	
8*	19.373	25.750	26.229	6.377	6.856	
9*	19.736	25.978	26.025	6.242	6.289	
10	19.999	25.888	26.151	5.889	6.153	
11	20.247	25.759	26.318	5.511	6.070	
12	20.494	25.631	26.483	5.137	5.989	3,6,10,11
13	20.738	25.503	26.649	4.765	5.910	
14	20.981	25.377	26.803	4.396	5.822	
15	21.221	25.252	26.642	4.030	5.421	
16	21.459	25.126	26.482	3.667	5.022	
17	21.695	25.002	26.322	3.397	4.627	
18	21.928	24.880	26.163	2.952	4.234	
19	22.159	24.758	26.004	2.599	3.845	
20	22.388	24.638	25.846	2.250	3.459	
21	22.614	24.519	25.689	1.905	3.076	
22	22.837	24.400	25.533	1.564	2.697	
23	23.057	24.284	25.378	1.226	2.321	
24	23.275	24.168	25.223	0.893	1.949	
25	23.491	24.054	25.071	0.563	1.580	
26	23.703	23.941	24.919	0.238	1.216	
27	23.918	23.920	24.856	0.002	0.938	
28	24.047	24.050	24.948	0.002	0.901	
29	24.174	24.176	25.037	0.002	0.864	10,11
30	24.297	24.299	25.124	0.002	0.827	
31	24.417	24.419	25.208	0.002	0.791	
32	24.534	24.536	25.289	0.002	0.755	
33	24.648	24.650	25.368	0.002	0.720	
34	24.760	24.762	25.446	0.002	0.685	
35	24.870	24.871	25.520	0.002	0.651	
36	24.976	24.978	25.593	0.002	0.617	
37	25.080	25.082	25.664	0.002	0.584	
38	25.182	25.184	25.733	0.002	0.551	
39	25.282	25.284	25.780	0.002	0.518	
40	25.380	25.381	25.865	0.002	0.486	

a : Initial step size is 0.446
* : Reduced step size

Table 6. Objective and level function values of CADOP8 run from starting point of Ref.[14]

Base No.	Objective Function			Level Function		Active Constraint Number
	f_1	f_2	f_3	ϵ_1	ϵ_2	
41	25.474	25.476	25.929	0.002	0.454	10,11
42	25.567	25.569	25.990	0.002	0.423	
43	25.659	25.661	26.050	0.002	0.392	
44	25.748	25.750	26.109	0.002	0.361	
45	25.835	25.837	26.166	0.002	0.331	
46	25.920	25.922	26.222	0.002	0.301	
47	26.004	26.006	26.276	0.002	0.272	
48	26.086	26.088	26.328	0.002	0.243	
49	26.166	26.168	26.380	0.002	0.214	
50	26.244	26.246	26.430	0.002	0.186	
51	26.321	26.323	26.479	0.002	0.158	
52	26.396	26.398	26.526	0.002	0.130	
53	26.470	26.472	26.573	0.002	0.103	
54	26.542	26.544	26.618	0.002	0.076	
55	26.612	26.614	26.662	0.002	0.050	
56	26.682	26.684	26.705	0.002	0.024	
57	26.742	26.743	26.744	0.002	0.002	
58	26.777	26.778	26.779	0.001	0.001	11
59	26.812	26.813	26.814	0.001	0.001	
60	26.844	26.845	26.846	0.001	0.001	
61	26.874	26.875	26.875	0.001	0.001	
62	26.901	26.902	26.902	0.001	0.001	3,11
63	26.906	26.907	26.907	0.001	0.001	
64	26.932	26.933	26.933	0.001	0.001	
65	26.943	26.944	26.944	0.001	0.001	
66[b]	26.958	26.958	26.958	0.000	0.000	3,6,11
67[b]	26.975	26.977	26.977	0.002	0.002	
68	26.993	26.995	26.995	0.002	0.002	
69	27.010	27.012	27.012	0.002	0.002	
70	27.026	27.028	27.028	0.002	0.002	
71	27.042	27.043	27.043	0.002	0.002	
72	27.056	27.058	27.058	0.002	0.002	
73	27.071	27.072	27.072	0.002	0.002	
74	27.084	27.086	27.086	0.001	0.002	
75	27.097	27.098	27.099	0.001	0.002	
76	27.109	27.111	27.111	0.001	0.002	
77	27.121	27.122	27.123	0.001	0.001	
78	27.132	27.134	27.134	0.001	0.001	
79	27.143	27.144	27.144	0.001	0.001	
80	27.153	27.154	27.155	0.001	0.001	

b : Step size switched to 0.2268 for optimality search

Base No.	Objective Function			Level Function		Active Constraint Number
	f_1	f_2	f_3	ϵ_1	ϵ_2	
81	27.163	27.164	27.164	0.001	0.001	3,6,11
82	27.172	27.173	27.173	0.001	0.001	
83	27.181	27.182	27.182	0.001	0.001	
84	27.189	27.190	27.191	0.001	0.001	
85	27.197	27.198	27.198	0.001	0.001	
86	27.205	27.206	27.206	0.001	0.001	
87	27.212	27.213	27.213	0.001	0.001	
88	27.219	27.220	27.220	0.001	0.001	
89	27.225	27.226	27.226	0.001	0.001	
90	27.231	27.232	27.232	0.001	0.001	
91	27.237	27.238	27.238	0.001	0.001	
92	27.242	27.243	27.243	0.001	0.001	
93	27.247	27.248	27.248	0.001	0.001	
94	27.252	27.253	27.253	0.001	0.001	
95	27.257	27.257	27.257	0.001	0.001	
96	27.261	27.262	27.262	0.001	0.001	
97	27.265	27.265	27.266	0.001	0.001	
98	27.268	27.269	27.269	0.001	0.001	
99	27.272	27.272	27.273	0.001	0.001	
100	27.275	27.276	27.276	0.001	0.001	
101	27.278	27.278	27.279	0.001	0.001	
102	27.280	27.281	27.281	0.001	0.001	
103	27.283	27.284	27.284	0.001	0.001	
104	27.285	27.286	27.286	0.001	0.001	
105	27.287	27.288	27.288	0.001	0.001	
106	27.289	27.290	27.290	0.001	0.001	
107	27.291	27.291	27.291	0.001	0.001	
108	27.294	27.294	27.294	0.001	0.001	
109	27.295	27.295	27.295	0.001	0.001	
110	27.296	27.296	27.296	0.001	0.001	
111	27.296	27.297	27.297	0.001	0.001	
112	27.297	27.298	27.298	0.000	0.001	
113	27.298	27.298	27.298	0.000	0.001	
114	27.298	27.298	27.298	0.00021	0.00023	3,6,10,11

CHAPTER X

CONCLUSION

The level objective function formulation and associated solution methodology provides a basis for effectively treating multiple objective function problems that cannot be reduced to a single objective problem through the use of weighting function as well as those that can. The approach is thus significantly more flexible than earlier methods. It is a new tool for the treatment of multiple objective problems. The new procedure is of importance primarily in the maximum performance problem. It however also provides an alternative approach to conventional multiple objective problems providing a new way of comparing objectives and locating POS.

The evaluation of the new procedure shows that it can be, and should be, much more efficient than the procedure of Pappas [14] on maximum performance problems. However the ability to adapt to an arbitrary, initially unknown, number of objective functions such as demonstrated by Nashanian and Pappas [28] has yet to be fully developed and tested. Thus the new procedure needs further development before its potential for the treatment of the general maximum performance problem can be exploited.

The new procedure appears to possess characteristics which make it quite useful in treating conventional multiple objective problems. The use of leveling constraints tends to reduce problem dimensionality but adds computational effort associated with constraint management. The effect is that the number of iterations needed for convergence is decreased but the computational effort per iteration is increased. On the two objective unconstrained test problem the total effort required for the new procedure was about the same as required for the solution of the problem formulated by conventional arithmetic weighting using a reasonably efficient single objective function optimizer. For more highly constrained problems with greater number of objective functions the benefits of iteration reduction should increase faster than the added computational effort per iteration. Thus the new procedure should be particularly effective for constrained problems and/or problems with a large number of objective functions. Further work is needed to more fully understand how this procedure compares in effectiveness to existing methods of treating conventional multiple objective problems. It should be noted however that no comparison of multiple procedures has been yet published and such a comparison is a major undertaking.

The major contribution of this new procedure to the treating of conventional multiple objective problems is

however not its efficiency potential but rather the additional formulation option it provides. Since there is no unique solution to the fundamental multiple objective function problem one seeks to examine available solutions (POS) and to select from them by the use of auxiliary conditions, which are often subjective. The new level function formulation provides an additional way of examining and locating POS thus allowing greater options for selection.

The difficulties associated with determining if a solution is a POS are not unique to the procedure presented here. The difficulty is also present in conventional methods. Similarly the methods for identifying a POS by use of Kuhn-Tucker concepts can likewise be applied to the new procedure. The conditions for identifying a POS for the level objective formulation have, however, yet to be formulated and tested.

The limited testing indicates that the procedure is quite effective in locating the feasible region from an infeasible point even where difficult nonlinear equality constraints are used. Thus the procedure also seems to be a useful tool for single objective function optimization.

In summary the procedures presented here appear to be a an effective tool for the treatment of an expanded range of

multiple objective optimization, as well as single objective function problems. Further work is needed to develop its potential, particularly the treatment of an arbitrary number of initially unknown objectives to allow its use for maximum performance problems.

REFERENCES

[1] Vanderplates, G. N., "Numerical Optimizations for Engineering Design with Application", McGraw Hill Inc, 1984.

[2] Fox, R. L., "Optimization Methods for Engineering Design", Addison-Wesley Publishing Co., Mass 1971

[3] Wilde, D. J., and Beightler, C. S., " Foundations of Optimization", Prentice-Hall, New Jersey, 1967

[4] Osyczka, A., " Multictiterion Optimization in Engineering", Ellis Horwood Limited, England, 1984

[5] Goicoechea, A., Hansen, R., and Duckstein, L., "Mutiobjective Decision Analysis with Engineering and Business Applications", John Wiley & Sons Inc., 1982.

[6] Bartel, D. J. and Marks, R. W., "Optimum Design of Mechanical Systems with Competing Design Objectives", ASME Trans, Journal of Engineering for Industry, Feb 1974, pp171-178.

[7] Carmichael, D. G., "Computation of Parato Optima in Structural Design", J Num Meths Eng., Vol 15, 1980 pp925-929.

[8] Rao,S. S., and Hati, S. K., "Game Theory Approach in Multicriteria Optimization of Function Generating Mechanisms", ASME Design Engineering Technical Conference, Sep 1978, paper No. 78-DET-87.

[9] Yoshimura, M., et al, " Multiobjective Design

Optimization of Machine-Tool Spindels", ASME Design and Production Engineering Technical Conference, Sep 1983, paper No. 83-DET-30

[10] Parato. V., Cours d'Economie Politique, Rouge, Lausanne. Switzerland, 1896.

[11] Metwalli, S. M., et al., "Multiple Design Objectives in Hydrodynamic Journal Bearing Optimization", ASME Design and Production Engineering Technical Conference, Sep 1983, paper No. 83-DET-41.

[12] Adali, A.," Multiobjective Design of an Antisymmetric Angle-ply Laminate by Nonlinear Programming", Trans ASME, Journal of Mechanisms, Transmission, and Automation in Design, Vol. 105, June 1983, pp214-219

[13] Ito, K., Akagi, S., and Ohta, M., "A Multiobjective Optimal Design of Thermal Systems", ASME Design and Production Engineering Technical Conference, Sep 1983, paper No. 83-DET-3.

[14] Pappas, M., "Optimal Frequency Separation of Cylindrical Shells", AIAA Journal, Vol6, Sep 1978, pp999-1001

[15]Pappas, M., "General Procedure for Numerical Design Optimization", NJIT Repoat NV-18, March 1982.

[16] Zoutendijk, G., Methods of Feasible Directions, Elsevier, Amsterdam, 1960.

[17] Bartel, D. L., Haug, E. J., and Rim, K., "The Optimum Design of Spatial Frames Using The Method of

Constrainted Steepest Descent with State Equations", Trans ASME, Journal of Engineering for Industry, Nov 1971, pp1261-1267.

[18] Pappas, M., and Moradi, J. Y., " An Improved Direct Search Mathematical Programming Algorithm ", Journal of Engineering for Industry, Trans ASME, Vol.97, No.4, Nov 1975, pp1305- 1310.

[19] Pappas, M., "A General Procedure and Program [CADOP5] for Numerical Design Optimization", NJIT Report No. NV19, March 1983.

[20] Dantzig, G., Linear Programming and Extensions , Princeton University Press, Princeton, NJ, 1963.

[21] Zangwill, W. T., "Nonlinear Programming : A Unified Approach", Prentice-Hall, Englewood Cliffs, NJ 1969.

[22] Sheu, C. Y., and Prager, W., "Recent Developments in Optimization Structural Design", Applied Mechanis Review, Vol.21, No.10, Oct 1968, pp985-992.

[23] Schmit, L. A., " Automated Design ", International Science and Technology, No.54, June 1966, pp63-78 and 115-117.

[24] Eason, E. D., and Fenton, R. G., "A Comparison of Numerical Optimization Methods for Engineering Design", Trans ASME, Ser B, Journal of Engineering for Industry, Vol.96, Feb 1974, pp196-200.

[25]Pappas, M.," Performance of the Direct Search Design

Algorithm as a Numerical Optimization Methods", AIAA Journal, Vol.13, No.6, June 1975, pp827-829.

[26] Gabriele, G. A., and Ragsdell, K. M., "The Generalized Reduced Gradient Methods : A reliable Tool for Optimal Design", Journal of Engineering for Industry, Trans ASME, Ser B, Vol.99, No.4, Nov 1977, pp1305-1311.

[27] Kuhn, H. W., and Tucher, A. W., " Nonlinear Programming", Proc of Second Berkely Symposium on Mathematical Statistics and Probability, 1951, pp481-4923.

[28] Nshanian, Y. S., and Pappas, M., "Optimal Laminated Composite Shells for Buckling and Vibration ", AIAA Journal, Vol.21, No.3, March 1983, pp430-437.

[29] Pappas, M., "Improved Methods for Large Structural Synthesis", AIAA Journal, Vol.19, No.9, Sep 1981, pp1227-1233.

[30] Pappas, M., and Amba-Rao, C. L. "A Direct Search Algorithm for Automated Optimal Design", AIAA Journal, Vol.9, No.3, March 1971, pp387-393.

[31] Pappas, M., "Use of Direct Search in Automated Optimal Design", Journal of Engineering for Industry, Trans ASME, Ser B, Vol.94, No.2, May 1972, pp395-401.

[32] Bronowicki, A. J., Nelson, R. B., Felton, L. P., and Schmit, L. A., "Optimization of Ring Stiffened Cylindrical Shells", AIAA Journal, Vol.13, Oct 1975, pp1319-1325.

APPENDIX A

USER INSTRUCTION FOR CADOP8

1. Introduction

The computer code CADOP8 treats the multiple objective program by means of a gradient based mathematical programming method. The method starts from a user specified starting or initial point and generates a sequence of better points until, hopefully, an POS, or near POS, point is reached. Because of the possibility of multiple local POS, numerical difficulties, or algorithm failure, nonlinear mathematical programming methods generally can not quarantee an optimal solution. It is desirable therefore to use repeated runs starting with different, widely separated starting point to confirm the achievement of a global optimum to determine the presence of local optima.

This instruction shows user how to plug in all equations for conventional multiple objective optimization. For maximum performance problem, the Q value may be unknown initially thus user must choose some strategies as described in section 5.2 to define the Q value. A separate subroutine to define the Q value must be prepared by the user to link with CADOP8. All equations can be likewise plug in without further description herein.

This code is composed of six subroutine and one function subprogram. The follows are the description of these subroutines.

(1) Main program

To input data of design variables (starting, lower, and upper), parameters, and etc., To print the results.

(2) CONVRG

This subroutine computes the convergence of objective function values. The termination criteria for convergence of algorithm are coded in this subroutines.

(3) LINPRO

This is used to solve the linear programming problem set up in SOLVE.

(4) SOLVE

This subroutine sets up the DFP and computes the move size. It decides whether the move size satisfies the termination criteria or not.

(5) OPTSRH

This is the optimization subroutine. It calls FIND, SOLVE, and CONVRG. Most of the optimization procedures are coded and thus computed herein. It prints most of the computation process and the results.

(6) FIND

This computes the value of objective functions, penalty functions by calling function subroutine (OBJ), and composite objective functions. It decides the activity and linearity of constraint and computes the initial value of step size in the automated process.

(7) DERIVE

This numerically differentiates the objective and constraint functions using a forward difference procedure.

(8) FUNCTION OBJ

The designer-defined program of design problem is attached herein. It transfers the value ofobjective and constraint function to FIND. This function subroutine counts the number of objective function evaluations and the number of constraint function evaluations.

2. Problem Formulation

Find the optimal design values x_i of the design variables x_i , i = 1,2,....IP, and the parameters P_k, k = 1,2,....KP which result in the minimization of the objective functions $f_q(x_i, P_k)$ subject to specified constraints. That is to find

$$C[f_q(x_i)] = \min A_q f_q(P_k, x_i) \quad q = 1,2,...Q \tag{A1}$$

while also satistying the behavior constraints

$$g_j(P_k, x_i) \leq 0 \quad j = 1,2,....JP \tag{A2}$$

and the regional constraints

$$x_i^l \leq x_i \leq x_i^u \tag{A3}$$

where A_q are weighting factors, q = 1,2,....Q

The CADOP8 program treats problems with $2 \leq IP \leq 10$, $0 \leq JP \leq 10$, $0 \leq KP \leq 100$ and $1 < Q \leq 5$. Further expansion is possible by changing the arrays in Main and all subroutines as stated in section A.5 .

3. Program Coding of Problem

The subroutine FUNCTION OBJ(J) is essentially a dummy

subprogram created to accept the user's FORTRAN program statement defining the objective and behavior constraints. Behavior constraints are expressed as a normal form of

$$B_j(x_i, P_k) \leq U_j(x_i, P_k) \qquad \begin{matrix} i = 1,2,...IP \\ j = 1,2,...JP \\ k = 1,2,...KP \end{matrix} \qquad (A4)$$

where B_j can be thought of as the behavior and U_j the upper limit of behavior. The objective and constraint functions are defined immediately after the end of FUNCTION OBJ as follows

```
        GO TO (11,12,....,1Q),j*            [ * : here j = q ]
1j      OBJ = expression defining f_q using P(k), X(I)
        GO TO 201
        :
        :
201     IF = IF + 1
        RETURN
1000    GO TO (1,2,...,JP),j
j       B = expression defining B_j
        U = expression defining U_j
        GO TO 101
        :
        :
101     OBJ = B - U
        IF ((B.NE.0) .AND. (U.NE.0)) OBJ = OBJ/DABS(U)
```

```
        IG = IG + 1
        RETURN
        END
```

If no constraint is used, statements after first RETURN arenot required. The last objective group statements and the last constraint statements need no use the final GO TO statement. An three objectives and three constraints problem can be written as follows

```
        GO TO (11, 12, 13), j
11      OBJ = (9/25 x1**2 + x2**2 )**0.5 + 2
        GO TO 201
12      OBJ = (x1 - 8 )**2/ 9 + (x2 -8 )**2 / 16 + 4
        GO TO 201
13      OBJ = [(x1 - 8)**2 + x2**2)]**0.5 + 4
201     IF = IF +1
        RETURN
10000   GO TO (1, 2, 3), j
1       B = (x1)**2 + (x2 - 10 )**2
        U = 49
        GO TO 101
2       B = x
        U = 1.
        GO TO 101
3       B = x2
        U = x
```

```
101     B = B- U
        IG = IG +1
        IF ((B.NE.0).AND.(U.NE.0)) OBJ = OBJ/DABS(U)
        RETURN
        END
```

The constraint values at the optimum are printed in the form as

$$g_j = \begin{cases} (B_j - U_j)/\ U_j & U_j \neq 0 \text{ and } B_j \neq 0 \\ (B_j - U_j) & U_j = 0 \text{ or } B_j = 0 \end{cases} \quad (A5)$$

thus a negative value of g_j indicates the constraint is satisfied.

4. Data Input

The first data set is to input , in order, the number of parameters (KP), variables (IP), behavior constraints (JP), linear functions (NLIN), and objective functions (Q). The data is entered on 5I10 format.

If and only if , linear functions are used (NLIN > 0) , a second data set is entered. If the objective functions is linear, a digit 1 is entered in the q^{th} column of the card. If the constraint g_j is linear, a digit 1 is entered in column j+Q.

The third data set is entered in the data control card and is used to specified whether : 1) new parameters are to be used (ICNTR2), 2)a new initial point is to be used (ICNTR3), and 3) regional limits or new regional limits are to be used (ICNTR4). The entries are made in 3I10 format. If regional limits are used, an entry of any none zero digit is made in the third field (col 21-30).

If, and only if, the number of parameters specified is greater than zero (KP > 0) a fourth data set is used to enter the problem parameters P_k. The entries are made 5 to a card in F15.8 format in order.

The fifth data set is to input initial variables x_i. The entries is made 8 to a card in F15.5 format.

The sixth data set is to input the weighting factors A_q. The entries are made 5 to a card in F15.8 format.

If and only if, an entry is made in the third field of the data control card (ICNTR4) = 0), a seventh data set defining the lower limits is entered, 8 to a card, in F15.5 format followed by an upper limit set with similar format.

For every additional run, an additional data control card is added followed by parameter and /or initial point

and/or regional limit sets. The need for a new set is indicated by an appropriate entry (ICNTR2 = 0) on the data control card.

5.Change of Problem Size

To change the maximun number of variable to (IP), or the maximum number of constraints to (JP), or maximum number of objectives (Q) the program can treat, the arrays in Main and all subroutines must be changed as follows :

(1) Change all D, X, DL, DU, and SS arrays in COMMON/REAL to D(IP)

(2) Change the G(R in SOLVE), and B(C in SOLVE and OPSRH, F in FIND) and BL arrays in COMMON/REAL and IA, IC arrays in COMMON/INT to G(JP) etc, as they occur.

(3) Change the F, CQ, EO, EO20, TO, SB, SUMF in COMMON/REAL to F(Q) etc, and change the ST, STE, SUME, TDIF(5), EOBASE, in DIMENSION of each subroutine to ST(Q) etc.

(4) Change A(32,78) arrays as they occur to A(M,L) and SSS(L) in COMMON/REAL, where

$M = JP + 3N + IP + 1$

$L = 2JP + 9N + 2IP + 3$

(5) Change the LIN and ABASE arrays in COMMON/REAL as they occur to LIN(JP+Q) and ABASE(JP+Q,IP).

(6) Change all DUMMY arrays such as DUM, IDUM etc., to equalize COMMON/REAL and COMMON/INT sizes and to property place all arrays in these common statements.

(7) In the SOLVE subprogram DIMENSION statement, change AT to AT(L) etc, [L as defined in (4)], DELTA, DELTAU to DELTA(IP) etc.

(8)In the OPTSRH subprogram DIMENSION statement, change XTEMP, XB, IAB, to XTEMP(IP), XB(IP) and IAB(JP).

(9) In the FIND subroutine DIMENSION statement, change HTEMP and SUMG to HTEMP(JP) and SUMG(JP).

(10) In the DERIV subprogram DIMENSION statement, change SUMG to SUMG(JP).

APPENDIX B

```
C ************ CADOP8 ************************
C
C This program is modified from CADOP5 [19] by Wen-Tsia Liu
C
        IMPLICIT REAL*8(A-H,O-Z)
      COMMON/REAL/D(10),P(100),X(10),G(10),T(10),B(10),DL(10
),DU(10),E4,
     1V,VMIN,F(5),DUM(2796),CQ(5),EO(4),WE
      COMMON/INT/IP,JP,LP,KL,IKF,IKG,IDUM(13),LIN(19),IDUMM(21)
     &   ,KQ,IEMOVE
1      FORMAT(5F15.8)
  2    FORMAT(5I10)
  25   FORMAT(3I10,3F10.5)
3      FORMAT(2F10.5)
101    FORMAT(8F10.5)
       READ2,KP,IP,JP,NLIN,LP
       IF(LP.EQ.0)LP=1
       IF(NLIN.EQ.0) GO TO 45
       K=JP+LP+LP-1
       READ 14,(LIN(J),J=1,K)
14     FORMAT(80I1)
   45 DO 21 I=1,IP
       DL(I)=-1.E+40
 21    DU(I)=1.E+40
       READ25,ICNTR2,ICNTR3,ICNTR4,E4,V,VMIN
       IF(KP.EQ.0) GO TO 12
       READ1,(P(I),I=1,KP)
 12    READ101,(D(I),I=1,IP)
 18    FORMAT(1H0,' STARTING DESIGN VARIABLE VALUES' )
       READ1,(CQ(I),I=1,LP)
       GO TO 23
  24   READ(5,25,END=33)ICNTR2,ICNTR3,ICNTR4,E4,V,VMIN
       IF(ICNTR2+ICNTR3+ICNTR4.EQ.0)STOP
       IF(ICNTR2.NE.0)READ1,(P(I),I=1,KP)
       IF(ICNTR3.NE.0)READ101,(D(I),I=1,IP)
 23    IF(ICNTR4.EQ.0)GO TO 22
       READ 101,(DL(I),I=1,IP)
 17    FORMAT(1H0,' LOWER LIMITS OF DESIGN VARIABLES')
       READ 101,(DU(I),I=1,IP)
 19    FORMAT(1H0,' UPPER LIMITS OF DESIGN VARIABLES')
22     IF(KP.LE.0) GO TO 104
       PRINT 6
       PRINT7,(K,P(K),K=1,KP)
104    PRINT 18
       PRINT101,(D(I),I=1,IP)
       PRINT 17
       PRINT101,(DL(I),I=1,IP)
       PRINT 19
       PRINT101,(DU(I),I=1,IP)
```

```
      IF(V.EQ.0.)V=.5
      IF(E4.EQ.0.)E4=1.E-06
      IF(VMIN.EQ.0.)VMIN=E4/10.
      PRINT 20,V,VMIN,E4
   20  FORMAT(' REDUCTION ATTEMPT=',F10.7,'MINIMUM ATTEMPT=',
     1F10.7,' CONVERGENCE SPEC=',E16.8/)
      CALL OPTSRH
      PRINT 4,KL,IKF,IKG
   4   FORMAT('1NO.OF REDESIGN CYCLES=',I5,' NO.OF OBJ
EVAL=',
     1I5,' NO.OF CONSRT EVAL=',I5/)
      DO13L=1,LP
13    F(L)=OBJ(L,0)
      PRINT305,(CQ(I),I=1,KQ)
305   FORMAT(' OBJECTIVE FUNCTION RATIOS=',5F15.8)
      PRINT5,(F(L),L=1,LP)
   5  FORMAT(' OPTIMUM OBJECTIVE FUNCTION VALUE(S)=',5E16.8)
      LPE=LP-1
      WRITE(6,905)(EO(L),L=1,LPE)
905   FORMAT(' EO(L)= ',5F15.8)
6     FORMAT( ' DESIGN PARAMETERS'/)
7     FORMAT(' P',I2,'=',F14.4)
      PRINT8
8     FORMAT( ' DESIGN VARIABLE VALUES'/)
      PRINT9,(K,X(K),K=1,IP)
9     FORMAT(' X',I1,'=',F15.8)
      IF(JP.EQ.0) GO TO 24
      PRINT10
10    FORMAT( ' NEARNESS TO CONSTRAINTS'/)
      PRINT11,(K,T(K),K=1,JP)
11    FORMAT(' G',I1,'=',F13.8)
 100  GO TO 24
33    STOP
      END
```

```
      SUBROUTINE CONVRG(ICODE,CONLMT,I,LP,F,FLAST,TDIF)
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION DIF(5),F(5),FLAST(5),TDIF(5)
      DO 3 L=1,LP
3     DIF(L)=DABS((FLAST(L)-F(L))/F(L))
      DO 5 L=1,LP
      IF(DIF(L).GT.CONLMT)GO TO 1
5     CONTINUE
      I=I+1
      IF(I.LT.2)GO TO 2
      DO 6 L=1,LP
      IF(DIF(L).GE.TDIF(L))GO TO 1
6     CONTINUE
      ICODE=0
      RETURN
    1 I=0
2     DO 4 L=1,LP
      FLAST(L)=F(L)
4     TDIF(L)=DIF(L)
      ICODE=1
      RETURN
      END
```

```
      SUBROUTINE LINPRO(K2)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/REAL/DUM(477),A(32,78),Q,CQ(5),EO(4),WE
      COMMON/INT/IDUM(49),Z,E,G,BB,W,B,H,N,M,L,KQ,IEMOVE
      INTEGER Z,E,G,BB,W,B,H,R,C
      M=M-1
      NN=6
560   LL=M+2
      DO 580 K=2,LL
 570  A(K-1,N+G+K-1)=1
 580  A(K-1,BB)=K+N+G-1
 600  IF(G.NE.0) GO TO 620
 610  IF(E.EQ.0) GO TO 780
 611  GO TO 650
620   KK=L+E+2
      LL=M+2
      DO 630 K=KK,LL
 630  A(K-1,K+N-L-E-1)=-1
 650  W=W+1
 660  Q=0
670   LL=N+G
      DO 760 J=1,LL
 680  S=0
690   LL1=M-G-E+2
      KK1=M+1
      DO 700 I=LL1,KK1
 700  S=S+A(I,J)
 720  A(W+1,J)=-S
 730  IF(A(W+1,J).GT.Q) GO TO 760
 740  Q=A(W+1,J)
 750  C=J
760   CONTINUE
 761  S=0
762   LL=M-G-E+2
      KK=M+1
      DO 763 J=LL,KK
 763  S=S+A(J,B)
 765  A(W+1,B)=-S
 790  IF(G.EQ.0) GO TO 810
      LL=N+1
      KK=N+G
 810  IF(L.EQ.0) GO TO 830
      LL=N+G+1
      KK=N+G+L
      IQ=1
 830  IF(G+E.EQ.0) GO TO 2000
831   LL=N+G+L+1
      KK=B-1
      IQ=1
 860  GO TO 2000
 895  IF(Q.EQ.99999) GO TO 1230
 900  IF(Q.EQ.0) GO TO 1330
 910  GO TO 1400
 920  H=H+1
```

```
 930   Q=.1E39
 940   R=-1
       LL=M+1
 950   DO 1000 I=1,LL
 960   IF(A(I,C).LE.0) GO TO 1000
 970   IF(A(I,B)/A(I,C).GT.Q) GO TO 1000
 980   Q=A(I,B)/A(I,C)
 990   R=I
1000   CONTINUE
1010   IF(FLOAT(R).GE.-.5) GO TO 1050
1020   CONTINUE
       IQ=3
1030   GO TO 2000
1050   P=A(R,C)
1060   A(R,BB)=C
1070   DO 1080 J=1,B
1080   A(R,J)=A(R,J)/P
       LL=W+1
1100   DO 1180 I=1,LL
1110   IF(I.EQ.R) GO TO 1180
1120   DO 1170 J=1,B
1130   IF(J.EQ.C) GO TO 1170
1140   A(I,J)=A(I,J)-A(R,J)*A(I,C)
1150   IF(ABS(A(I,J)).GT..1E-4) GO TO 1170
1160   A(I,J)=0
1170   CONTINUE
1180   CONTINUE
       LL=W+1
1190   DO 1200 I=1,LL
1200   A(I,C)=0
1210   CONTINUE
1220   A(R,C)=1
1230   Q=0
1240   LL=N+G+L
       DO 1280 J=1,LL
1250   IF(A(W+1,J).GT.Q)GO TO 1280
1260   Q=A(W+1,J)
1270   C=J
1280   CONTINUE
1290   GO TO 900
1330   IF(W.EQ.M+1) GO TO 1360
1340   W=W-1
1350   IF(A(W+2,B).LT..1E-5) GO TO 1353
       K2=0
1352   RETURN
1353   LL=M+1
       DO 1358 I=1,LL
1354   IF(INT(A(I,BB)).LE.N+G+L) GO TO 1358
1355   DO 1356 J=1,B
1356   A(1,J)=0
1358   CONTINUE
1359   GO TO 1230
1360 CONTINUE
1400   IF(Q.EQ.0) GO TO 1420
```

```
1420  CONTINUE
      LL=M+1
1430  DO 1460 I=1,LL
1440  IF(INT(A(I,BB)).EQ.0) GO TO 1460
1460  CONTINUE
1470  IF(Q.NE.0) GO TO 920
      XJJ=-Z*A(W+1,B)
      LL=H-1
      IQ=3
1550  GO TO 2000
2000  LL=H-1
      LL=W+1
2030  DO 5000 I=1,LL
5000  CONTINUE
2091  CONTINUE
      GO TO(895,1050,999),IQ
999   RETURN
      END
```

```
      SUBROUTINE SOLVE(SUM,K2,JK,KF)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/REAL/D(10),P(100),X(10),R(10),T(10),C(10),DL(10
 ),DU(10),E4,
     1V,VMIN,TO(5),BL(10),ALP,SB(5),SUMF(5),SS(10),SSS(78),
     2ABASE(19,10),A(32,78),Q,CQ(5),EO(4),WE
      COMMON/INT/IP,JP,LP,KL,IKF,IKG,NACT,NACL,IA(10),KX,IDM(30),
     1Z,E,G,BB,W,B,H,N,M,L,KQ,IEMOVE
      INTEGER Z,E,G,BB,W,B,H,BT
      DIMENSION AT(78),DELTA(10),DELTU(10),
ATT(32,78),SIGMAE(4)
      JK=1
      K2=1
      LPE=LP-1
      MM=NACT+LP+LPE+LPE+IP
      IF(KX.EQ.1.OR.IEMOVE.EQ.0)MM=MM-LPE
      B=MM+IP+2*LP+LPE+1
      IF(IEMOVE.EQ.0)B=MM+IP+1+1
      IF(KX.EQ.1)B=MM+IP+2*LPE+NACT+1
      BB=B+MM+1-IP
      DO 103 I=1,BB
  103 SSS(I)=0.
      H=1
      Z=-1
      E=0
      G=0
      Q=99999
3     DO 4 K=1,IP
      DELTA(K)=(D(K)-DL(K))
      DELTU(K)=(DU(K)-D(K))
      IF(DELTU(K).GT.ALP)DELTU(K)=ALP
      IF(DELTA(K).GT.ALP)DELTA(K)=ALP
    4 CONTINUE
      LX=MM+2
      ML=NACT+LP+1+LPE
      IPP=IP+1
      DO 350 I=ML,LX
      DO 350 J=1,BB
  350 A(I,J)=0.
      ML=ML-1
      DO 360 I=1,ML
      DO 360 J=IPP,BB
  360 A(I,J)=0.
      IF(KX.EQ.0)GO TO 5370
      DO 5360 I=1,LP
      DO 5360 J=1,IP
5360  A(I,J)=0.
5370  CONTINUE
      J=LP+NACT+LPE+LPE
      IF(KX.EQ.1.OR.IEMOVE.EQ.0)J=J-LPE
      W=J+IP
      N=IP+1
      IF(KX.EQ.1)GO TO 6400
      IF(IEMOVE.EQ.0)GO TO 6200
```

```
      DO 6928 L=1,LPE
      SIGMAE(L)=0.
      DO 6948 I=1,IP
6948  SIGMAE(L)=SIGMAE(L)+DABS(A(LP+NACT+L,I))*ALP
      SIGMAE(L)=EO(L)-0.5*SIGMAE(L)
      IF(EO(L)*0.8.GT.SIGMAE(L))SIGMAE(L)=EO(L)*0.8
6928  CONTINUE
      DO 6028 I=1,LP
      A(I,N)=1.
      A(W+1,N)=1.
      N=N+1
      A(I,N)=-1.
      A(W+1,N)=-1.
6028  N=N+1
      LX=LP+NACT+1
      LY=J-LPE
      DO 6029 I=LX,LY
      A(I,N)=-1.
      A(I+LPE,N)=1.
      A(W+1,N)=-WE
6029  N=N+1
      GO TO 6500
6200  DO 6228 I=1,LP
6228  A(I,N)=1.
      A(W+1,N)=1.
      N=N+1
      GO TO 6500
6400  III=LP+1
      NNN=LP+NACT
      DO 6428 I =III,NNN
      A(I,N)=-1.
      A(W+1,N)=-1.
6428  N=N+1
      III=LP+NACT+1
      NNN=LP+NACT+2*LPE
      DO 6429 I=III,NNN
      A(I,N)=1.
      A(W+1,N)=1.
      A(I,N+1)=-1.
      A(W+1,N+1)=-1.
6429  N=N+1
6500  CONTINUE
      N=N-1
      IF(KX.EQ.1.OR.IEMOVE.EQ.0)GO TO 8730
      LY=LY+1
      DO 8720 I=LY,J
8720  A(I,B)=SIGMAE(LY-LP-NACT-LPE)
8730  CONTINUE
      DO 27 I=1,IP
      J=J+1
      A(J,I)=1.
  27  A(J,B)=DELTA(I)+DELTU(I)
      W=J
      MM=LP+NACT+LPE
```

```
      DO 9 I=1,MM
      DO 20 K=1,IP
      A(I,B)=A(I,B)+DELTA(K)*A(I,K)
20    CONTINUE
      IF(I.LE.LP)GO TO 9
      IF(I.GT.LP+NACT)GO TO 1090
      K3=IA(I-LP)
      A(I,B)=A(I,B)-T(K3)
      IF(I.LE.LP+NACT)GO TO 9
      IF(KX.EQ.1)GO TO 9
1090  A(I,B)=A(I,B)-EO(I-LP-NACT)
    9 CONTINUE
      IG=W
      G=0
      MX=W+1
      L=0
      E=LPE
      DO 3400 I=1,W
      IF(I.GT.LP+NACT.AND.I.LE.LP+NACT+LPE)GO TO 3400
      IF(A(I,B).GE.0.)L=L+1
      IF(A(I,B).LT.0.)G=G+1
3400  CONTINUE
      LT=0
      GT=0
      ET=0
      DO 520 I=1,W
      IF(I.GT.LP+NACT.AND.I.LE.LP+NACT+LPE)GO TO 580
      IF(A(I,B).GE.0.)GO TO 540
      GT=GT+1
      DO 550 K=1,B
550   ATT(L+E+GT,K)=-A(I,K)
      GO TO 520
540   LT=LT+1
      DO 570 K=1,B
570   ATT(LT,K)=A(I,K)
      GO TO 520
580   ET=ET+1
      IF(A(I,B).LT.0.)GO TO 585
      DO 5599 K=1,B
5599  ATT(L+ET,K)=A(I,K)
      GO TO 520
585   DO 595 K=1,B
595   ATT(L+ET,K)=-A(I,K)
520   CONTINUE
      DO 530 I=1,W
      DO 530 K=1,B
530   A(I,K)=ATT(I,K)
      DO 510 I=1,N
510   A(W+1,I)=Z*A(W+1,I)
      M=W
      BT=B
      B=B+G
      BB=B+1
      IF(G.EQ.0)GO TO 26
```

```
      DO 590 I=1,M
      ATEMP=A(I,BT)
      A(I,BT)=0.
590    A(I,B)=ATEMP
  26  JK=1
      CALL LINPRO(K2)
      LL=M+1
      DO 1460 I=1,LL
      J=A(I,BB)
       SSS(J)=A(I,B)
1460  CONTINUE
      DO 19 I=1,IP
       SS(I)=SSS(I)
19     CONTINUE
      DO 13 I=1,IP
       SS(I)=SS(I)-DELTA(I)
13    CONTINUE
 113  SUM=0.0
      DO 115I=1,IP
115    SUM=SUM+SS(I)*SS(I)
       SUM=DSQRT(SUM)
      IF(SUM.LT.E4.AND.SUM.LT.ALP)JK=0
      RETURN
      END
```

```
      SUBROUTINE OPTSRH
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/REAL/D(10),P(100),X(10),G(10),T(10),C(10),DL(10
),DU(10),E4,
     1V,VMIN,TO(5),BL(10),ALP,SB(5),SUMF(5),SS(10),SSS(78),
     2ABASE(19,10),A(32,78),Q,CQ(5),EO(4),WE
      COMMON/INT/IP,JP,LP,KL,IKF,IKG,NACT,NACL,IA(10),KX,LIN
(19),IC(10),
     1IL,IDUM(10),KQ,IEMOVE
      DIMENSION
ST(5),TDIF(5),XTEMP(10),XB(10),SBLST(10),IAB(10)
     1   ,XACT(10),TT(10),BLB(10),STE(4),EOBASE(4)
      KL=0
      KF=0
      NACTB=0
      NACLB=0
      IKF=0
      IKG=0
      ICO=0
      IEMOVE=1
      INT=1
      VF=.125
      NOPTN=0
      MV=0
      ED=0.1
      LPE=LP-1
      ALP=1.
      DO 44 J=1,JP
44    BL(J)=2.*V
46    DO 37 L=1,LPE
      SBLST(L)=1.E+40
      SB(L)=SBLST(L)
37    TDIF(L)=SB(L)
      MOVE=0
1     DO 47 I=1,IP
      IF(D(I).LT.DL(I))D(I)=DL(I)
      IF(D(I).GT.DU(I))D(I)=DU(I)
47    CONTINUE
      IF(IEMOVE.EQ.0)VF=V
      IF(IEMOVE.EQ.0)ALPF=ALP
      IF(IEMOVE.EQ.1)VE=V
      IF(IEMOVE.EQ.1)ALPE=ALP
      CALL FIND(ST,KD,STE,INT,MV,NOPTN,ALPF,KF,ED)
      IF(IEMOVE.EQ.1.AND.INT.EQ.0)GO TO 405
      IF(IEMOVE.EQ.1.AND.INT.EQ.1)GO TO 3405
      IF(IEMOVE.EQ.0.AND.INT.EQ.0)GO TO 405
      DO 437 L=1,LP
      SBLST(L)=1.E+40
437   TDIF(L)=SBLST(L)
      GO TO 405
3405  DO 3406 L=1,LPE
      SBLST(L)=1.E+40
3406  TDIF(L)=SBLST(L)
405   CONTINUE
```

```
      IF(KL.EQ.0)ALPE=ALP
   69 KL=KL+1
   32 IF(KD.NE.0.AND.MOVE.NE.2)GO TO 9
      IF(KD.NE.0) GO TO 48
      KXBASE=KX
      WEBASE=WE
      KF=0
      LPFE=LP
      IF(IEMOVE.EQ.0)GO TO 440
      LPFE=LP-1
      DO 434 L=1,LPFE
434   SB(L)=STE(L)
      IF(INT.EQ.0)GO TO 412
      ALP=ALPE
      V=VE
      GO TO 412
440   DO 34 L=1,LPFE
34    SB(L)=ST(L)
      IF(INT.EQ.0)GO TO 412
      ALP=ALPF
      V=VF
412   CONTINUE
      DO 634 L=1,LPE
634   EOBASE(L)=EO(L)
      CALL CONVRG(ICONDE,E4,ICO,LPFE,SB,SBLST,TDIF)
      IF(ICONDE.EQ.0)GO TO 24
   12 IF(JP.EQ.0)GO TO 6
      DO 7 J=1,JP
   7  T(J)=G(J)
      DO 53 K=1,NACT
      J=IA(K)
      BLB(J)=BL(J)
53    IAB(K)=IA(K)
   6  NACTB=NACT
      NACLB=NACL
      MM=NACTB+LP
      IF(MOVE.EQ.2) GO TO 55
      DO 56 I=1,IP
56    XB(I)=D(I)
55    DO 8 I=1,IP
8     X(I)=D(I)
      DO 65 L=1,LP
      DO 65 I=1,IP
   65  ABASE(L,I)=A(L,I)
      NN=LP+1
      IF(NACT.EQ.0)GO TO 67
      DO 66 K=NN,MM
      J=IA(K-LP)+LP
      DO 66 I=1,IP
66    ABASE(J,I)=A(K,I)
67    DO 366 L=1,LPE
      DO 366 I=1,IP
366   ABASE(LP+JP+L,I)=A(LP+NACT+L,I)
      IF(MOVE.EQ.2)GO TO 48
```

```
      IF((NACT.NE.0.AND.KX.EQ.0).OR.MOVE.NE.1.OR.IEMOVE.EQ.0
     & .OR.NOPTN.EQ.1)GO TO 48
      MV=0
51    SUMT=0.
      KF=1
      DO 4 I=1,IP
      D(I)=2.*D(I)-XTEMP(I)
4     SUMT=SUMT+(D(I)-XTEMP(I))**2
      SUMT=DSQRT(SUMT)
      MOVE=3
      IF(SUMT.LT.SUM/2.)GO TO 9
      MOVE=2
      GO TO 1
48    DO 50 I=1,IP
50    XTEMP(I)=XB(I)
  3   CONTINUE
      IF(KF.NE.1.OR.JP.EQ.0)GO TO 3760
      DO 3770 J=1,JP
      TT(J)=T(J)
3770  T(J)=G(J)
3760  CONTINUE
      WRITE(6,15)(SB(L),L=1,LPFE)
      WRITE(6,315)(EO(L),L=1,LPE)
      WRITE(6,38)ALP,V ,IEMOVE
      WRITE(6,13)(D(I),I=1,IP)
      IF(JP.EQ.0)GO TO 45
      WRITE(6,14)(G(J),J=1,JP)
  45  CALL SOLVE(SUM,K2,JK,KF)
      IF(K2.EQ.0)GO TO 9
      IF(JK.EQ.0)GO TO 23
      AMULT=1.
  25  TEST=0.
      DO 18 I=1,IP
      IF(DABS(SS(I)).GT.TEST)TEST=DABS(SS(I))
  18  D(I)=D(I)+SS(I)*AMULT
      SUMT=0.
      DO 74 I=1,IP
74      SUMT=SUMT+(D(I)-XTEMP(I))**2
      SUMT=DSQRT(SUMT)
      MOVE=3
      IF(SUMT.LT.SUM/2.)GO TO 9
      MOVE=1
      GO TO 1
  9   DO 19 I=1,IP
  19  D(I)=X(I)
      DO 719 L=1,LPE
719   EO(L)=EOBASE(L)
      IF(NACTB.EQ.0)GO TO 20
      DO 54 K=1,NACTB
54    IA(K)=IAB(K)
      DO 22 K=NN,MM
      J=IA(K-LP)+LP
      DO 22 I=1,IP
  22  A(K,I)=ABASE(J,I)
```

```
20      DO 68 K=1,LP
        DO 68 I=1,IP
68      A(K,I)=ABASE(K,I)
        DO 422 L=1,LPE
        DO 422 I=1,IP
422     A(LP+NACTB+L,I)=ABASE(LP+JP+L,I)
        NACL=NACLB
        NACT=NACTB
        IF(KF.NE.1)GO TO 33
        KF=0
        DO 49 I=1,IP
49      XB(I)=X(I)
        DO 3750 J=1,JP
3750    T(J)=TT(J)
        GO TO 48
  33    V=V/2.
        ALP=ALP/2.
        KX=KXBASE
        WE=WEBASE
        IF(NACT.EQ.0)GO TO 3800
        NACT=0
        DO 3800 K=1,NACTB
        J=IA(K)
        BL(J)=BLB(J)*0.5
        BLB(J)=BL(J)
        IF(T(J).LT.-BL(J))GO TO 3800
        NACT=NACT+1
        IA(NACT)=J
        IF(NACT.EQ.K)GO TO 3800
        DO 3820 M=1,IP
3820    A(NACT+LP,M)=A(K+LP,M)
3800    CONTINUE
        DO 3840 L=1,LPE
        DO 3840 I=1,IP
3840    A(LP+NACT+L,I)=A(LP+NACTB+L,I)
        IF(V.LT.VMIN)GO TO 28
        IF(MOVE.EQ.0.OR.KF.GE.1)GO TO 35
        IF(MOVE.EQ.3)GO TO 3
        CALL CONVRG(ICONDE,E4,ICO,LPFE,SB,SBLST,TDIF)
        IF(ICONDE.EQ.0)GO TO 24
  35    MOVE=0
        KF=2
        IF(TEST.LT.ALP)GO TO 33
        IF(NACT.NE.0.OR.NACL.NE.0.OR.IEMOVE.EQ.1)GO TO 3
        AMULT=AMULT/2.
        SUM=SUM/2.
        GO TO 25
  28    WRITE(6,29)
29      FORMAT(' TERMINATION BY MINIMUM STEP SIZE CRITERIA')
        RETURN
  24    WRITE(6,27)
13      FORMAT(' BASE VARIABLES=',5E16.8)
        RETURN
14      FORMAT(' BASE CONSTRAINT VALUES=',5E16.8)
```

```
  15   FORMAT(' BASE OBJECTIVE FUNCTION VALUE(S)=',5E16.8)
315    FORMAT(' EO(L)=',5F16.8)
38     FORMAT(' STEP SIZE=',E16.8,' RED ATTEMP=',F10.7,'
IEMOVE=',I3)
26     FORMAT(' TERMINATION BY SATISFACTION OF OPTIMALITY
CRITERIA')
27     FORMAT(' TERMINATION BY SATISFACTION OF CONVERGENCE
CRITERIA')
  23   WRITE(6,26)
       RETURN
       END
```

```
      SUBROUTINE FIND(ST,KODE,STE,INT,MV,NOPTN,ALPF,KF,ED)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/REAL/D(10),P(100),X(10),G(10),T(10),F(10),DL(10
     ),DU(10),E4,
     1V,VMIN,TO(5),BL(10),ALP,SB(5),SUMF(5),SS(10),SSS(78),
     2ABASE(19,10),A(32,78),Q,CQ(5),EO(4),WE
      COMMON/INT/IP,JP,LP,KL,IKF,IKG,NACT,NACL,IA(10),KX,LIN
     (19),IC(10),
     1IL,IDUM(10),KQ,IEMOVE
      DIMENSION HTEMP(10),SUMG(10),ST(10) ,STE(4),SUME(4)
      IEMOV=IEMOVE
      KODE=2
      LPE=LP-1
      DO 11 L=1,LP
      TO(L)=OBJ(L,0)
      IF(L.GT.1)EO(L-1)=DABS(TO(1)-TO(L)*CQ(L))
      IF(TO(L).GT.SB(L).AND.KX.EQ.0.AND.IEMOVE.EQ.0.AND.KF.NE.1)RETU
     IF(L.EQ.1)GO TO 11
     IF(EO(L-1).GT.SB(L-
1).AND.KX.EQ.0.AND.IEMOVE.EQ.1.AND.KF.NE.1
     &  .AND.EO(L-1).GT.ED) RETURN
     STE(L-1)=EO(L-1)
  11 ST(L)=TO(L)
     IF(IEMOVE.EQ.0)GO TO 1200
2080 DO 1080 L=1,LPE
     IF(EO(L).GT.ED)GO TO 1200
1080 CONTINUE
     ED=ED*0.1
     GO TO 2080
     IF(IEMOV.EQ.1)ED1=ED
1200 CONTINUE
     INT=0
     ME=0
     DO 211 L=1,LPE
     IF(EO(L).LT.0.1)NOPTN=1
211  IF(EO(L).GE.0.001)ME=1
     IF((IEMOVE.EQ.0.AND.ME.EQ.1).OR.(IEMOVE.EQ.1.AND.ME.EQ.0))INT=1
     IEMOVE=ME
     IF(IEMOVE.EQ.0)ED=0.1
     IV=0
     KX=0
     IL=0
     NACT=0
     KODE=0
     IF(JP.EQ.0)GO TO 1
     DO 2 J=1,JP
     G(J)=OBJ(J,1)
     IF(G(J).LT.-2.*BL(J))GO TO 2
     IL=IL+1
     IC(IL)=J
  2  CONTINUE
   1 CALL DERIV(SUMG,ALP,SUME)
     IF(KF.NE.1)GO TO 1060
     KODE=1
```

```
      DO 1050 L=1,LPE
1050   IF(EO(L).GT.SB(L))RETURN
      KODE=0
1060  CONTINUE
      MX=LP+IL
      IF(KL.EQ.0)GO TO 300
      IF(IEMOVE.EQ.0.AND.MV.EQ.0)GO TO 30
32    NACL=0
      DO 20 K=1,IP
      IF(D(K)-DL(K).LE.ALP)NACL=NACL+1
      IF(DU(K)-D(K).LT.ALP)NACL=NACL+1
  20  CONTINUE
      IF(IL.EQ.0)RETURN
      LPFE=LP-1
      IF(IEMOV.EQ.0)LPFE=LP
      DO 15 L=1,LPFE
      HTEMP(L)=0.
      DO 15 K=1,IL
      SUM=0.
      J=IC(K)
      IF(G(J).LE.0.)GO TO 15
      IV=IV+1
      IF(G(J).LT..1)GO TO 17
  19  E=10000.*G(J)
      GO TO 18
17    IF(IEMOV.EQ.1)GO TO 317
      DO 7 M=1,IP
   7  SUM=SUM+A(L,M)*A(K+LP,M)
      IF(DABS(SUM).LT.1.E-15)GO TO 19
      E=DABS(G(J)*SUMF(L)/SUM*2.)
      GO TO 18
317   DO 307 M=1,IP
307   SUM=SUM+A(LP+IL+L,M)*A(K+LP,M)
      IF(DABS(SUM).LT.1.E-15)GO TO 19
      E=DABS(G(J)*SUMF(L)/SUM*2.)
   18 IF(E.GT.HTEMP(L))HTEMP(L)=E
  15  CONTINUE
      KODE=1
      DO 5 K=1,IL
      J=IC(K)
   6  TEMP=1.E+48
      DO 9 M=1,IP
      QUAN=DABS(A(K+LP,M))
      IF(QUAN.LT.1.E-06)GO TO 9
      CHECK=1./QUAN
      IF( CHECK.LT.TEMP)TEMP=CHECK
   9  CONTINUE
      XIV=IV+1.E-10
      DSUM=SUMG(J)**0.5
      IF(DSUM.LT.0.000001)DSUM=0.000001
      IF(G(J)/DSUM.GT. ALP/XIV)KX=1
      BL(J)=TEMP*SUMG(J)*ALP
 55   IF(G(J).LT.-BL(J))GO TO 5
      NACT=NACT+1
```

```
      IA(NACT)=J
      IF(NACT.EQ.K)GO TO 5
      DO 10 M=1,IP
   10 A(NACT+LP,M)=A(K+LP,M)
    5 CONTINUE
      DO 410 L=1,LPE
      DO 410 I=1,IP
  410 A(LP+NACT+L,I)=A(LP+IL+L,I)
      MX=LP+NACT
      IF(IEMOV.EQ.0)GO TO 316
      DO 314 L=1,LPFE
      STE(L)=EO(L)+HTEMP(L)
      IF(STE(L).LT.ED)GO TO 314
      IF(STE(L).GT.SB(L))IEMOVE=IEMOV
      IF(IEMOVE.EQ.1)ED=ED1
      IF(IEMOVE.EQ.1)INT=0
      IF(STE(L).GT.SB(L))RETURN
  314 CONTINUE
      GO TO 318
  316 DO 14 L=1,LP
      ST(L)=TO(L)+HTEMP(L)
      IF(ST(L).GT.SB(L))IEMOVE=IEMOV
      IF(IEMOVE.EQ.0)INT=0
      IF(ST(L).GT.SB(L))RETURN
   14 CONTINUE
  318 CONTINUE
      IF(KX.EQ.1.AND.KL.EQ.0)GO TO 35
   36 CONTINUE
      KODE=0
      RETURN
  300 TEMP=0.
      CHECK=1.E+40
      DO 343 L=1,LPE
      IF(SUME(L).GE.CHECK)GO TO 343
      IQ=L
      CHECK=SUME(L)
  343 CONTINUE
      DO 331 I=1,IP
      IF(DABS(A(IQ+LP+IL,I)).GT.TEMP)TEMP=DABS(A(IQ+LP+IL,I))
  331 CONTINUE
      ALP=V*DABS(EO(IQ))*TEMP/SUME(IQ)
      GO TO 350
   30 TEMP=0.
      CHECK=1.E+40
      DO 43 L=1,LP
      IF(SUMF(L).GE.CHECK)GO TO 43
      IQ =L
      CHECK =SUMF(L)
   43 CONTINUE
      DO 31 I=1,1P
      IF(DABS(A(IQ,I)).GT.TEMP)TEMP=DABS(A(IQ,I))
   31 CONTINUE
      V=.125
      ALPF=V*DABS(TO(IQ))*TEMP/SUMF(IQ)
```

```
      MV=1
      GO TO 32
350   CONTINUE
      DO 50 I=1,IP
50     X(I)=D(I)
      IF(JP.EQ.0)GO TO 32
      DO 49 J=1,JP
49     T(J)=G(J)
      GO TO 32
  35  TEMP=0.
      CHECK=1.E+40
      DO 34 L=1,NACT
      J=IA(L)
      IF(G(J).GE.CHECK)GO TO 34
      IQ =L
      CHECK=G(J)
34    CONTINUE
      DO 37 I=1,IP
      IF(DABS(A(IQ+LP,I)).GT.TEMP)TEMP=DABS(A(IQ+LP,I))
37    CONTINUE
      J=IA(IQ)
      ALPT=2*V*G(J)*TEMP/SUMG(J)
      IF(ALPT.LE.ALP)GO TO 36
      DO 38 J=1,JP
38     BL(J)=BL(J)*ALPT/ALP
      ALP=ALPT
      GO TO 36
      END
```

```
      SUBROUTINE  DERIV(SUMG,ALP,SUME)
        IMPLICIT  REAL*8(A-H,O-Z)
      COMMON/REAL/D(10),DUM1(110),G(10),DUM2(43),TO(5),DUM3(16),
     1SUMF(5),DUM4(88),ABASE(19,10),A(32,78),DUM5,CQ(5),EO(4),WE
      COMMON/INT/IP,JP,LP,KL,IKF,IKG,NACT,NACL,IA(10),KX,LIN
(19),IC(10),
     1IL,IDUM(10),KQ,IEMOVE
      DIMENSION SUMG(10),EODEL(4),TODEL(5),SUME(4)
      DEL=.0001
      LPE=LP-1
      DO 1001 L=1,LPE
1001   IF(EO(L).LT.0.05)DEL=0.0000001
      IF(DEL.GT.ALP/10.)DEL=ALP/10.
  1    DO 12 L=1,LP
  12   SUMF(L)=0.
       DO 13 J=1,JP
  13   SUMG(J)=0.
       DO 14 L=1,LPE
14     SUME(L)=0.
       DO 3 K=1,IP
       DT=D(K)
       D(K)=D(K)+DEL
       DO 8 L=1,LP
       TODEL(L)=OBJ(L,0)
       IF(LIN(L).LE.1)A(L,K)=(TODEL(L)-TO(L))/DEL
       IF(LIN(L).GT.1)A(L,K)=ABASE(L,K)
       IF(LIN(L).EQ.1)ABASE(L,K)=A(L,K)
8      SUMF(L)=SUMF(L)+A(L,K)**2.
       IF(IL.EQ.0)GO TO 33
       DO 4 L=1,IL
       J=IC(L)
       IF(LIN(J+LP).LE.1)A(LP+L,K)=(OBJ(J,1)-G(J))/DEL
       IF(LIN(J+LP).GT.1)A(L+LP,K)=ABASE(J+LP,K)
       IF(LIN(J+LP).EQ.1)ABASE(J+LP,K)=A(L+LP,K)
  4    SUMG(J)=SUMG(J)+A(LP+L,K)**2
33      CONTINUE
       DO 18 L=1,LPE
       EODEL(L)=DABS(TODEL(1)-TODEL(1+L)*CQ(1+L))
       IF(LIN(LP+JP+L).LE.1)A(LP+IL+L,K)=(EODEL(L)-EO(L))/DEL
       IF(LIN(LP+JP+L).GT.1)A(LP+IL+L,K)=ABASE(LP+JP+L,K)
       IF(LIN(LP+JP+L).EQ.1)ABASE(LP+JP+L,K)=A(LP+IL+L,K)
18     SUME(L)=SUME(L)+A(LP+IL+L,K)**2.
3      D(K)=DT
       AF=0.
       DO 918 L=1,LP
       DO 918 K=1,IP
918    AF=AF+DABS(A(L,K))
       AE=0.
       DO 919 L=1,LPE
       DO 919 K=1,IP
919    AE=AE+DABS(A(LP+IL+L,K))
       WE=(AF/AE)*0.5
       IF(LPE.GT.1)WE=100./WE
       DO 5 L=1,LP
```

```
      IF(LIN(L).EQ.1)LIN(L)=2
5     CONTINUE
      DO 7 L=1,LPE
      IF(LIN(LP+JP+L).EQ.1)LIN(LP+JP+L)=2
7     CONTINUE
      DO 6 L=1,IL
      J=IC(L)
      IF(LIN(J+LP).EQ.1)LIN(J+LP)=2
6     CONTINUE
      RETURN
      END
```

```
REAL FUNCTION OBJ*8(J,KODE)
  IMPLICIT REAL*8(A-H,O-Z)
COMMON/REAL/X(10),P(100),DUM(2874)
COMMON/INT/IDUM(4),IF,IG,IDUMM(55)
IF(KODE.NE.0)GO TO 1000
```