A feasible direction procedure for general multiple objective optimization

Wen-Tsia Liu
New Jersey Institute of Technology

Follow this and additional works at: https://digitalcommons.njit.edu/dissertations

Recommended Citation
https://digitalcommons.njit.edu/dissertations/1219

This Dissertation is brought to you for free and open access by the Electronic Theses and Dissertations at Digital Commons @ NJIT. It has been accepted for inclusion in Dissertations by an authorized administrator of Digital Commons @ NJIT. For more information, please contact digitalcommons@njit.edu.
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
The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.
INFORMATION TO USERS

The most advanced technology has been used to photograph and reproduce this manuscript from the microfilm master. UMI films the original text directly from the copy submitted. Thus, some dissertation copies are in typewriter face, while others may be from a computer printer.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyrighted material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is available as one exposure on a standard 35 mm slide or as a 17" × 23" black and white photographic print for an additional charge.

Photographs included in the original manuscript have been reproduced xerographically in this copy. 35 mm slides or 6" × 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
A feasible direction procedure for general multiple objective optimization

New Jersey Institute of Technology, 1988
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______________________________________________

__________________________________________________________

__________________________________________________________

UMI
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
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
Professor
Department of Mechanical Engineering

Signatures of Other Members of The Thesis Committee

Date

Date

Date
Name: Wen-Tsia Liu

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

Secondary education:

<table>
<thead>
<tr>
<th>College</th>
<th>dates</th>
<th>Degree</th>
<th>Date of Degree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ming-Chi Institute of Technology</td>
<td>68-73</td>
<td>BSME</td>
<td>1973</td>
</tr>
<tr>
<td>New Jersey Institute of Technology</td>
<td>80-81</td>
<td>MSME</td>
<td>1982</td>
</tr>
<tr>
<td>New Jersey Institute of Technology</td>
<td>83-88</td>
<td>D. Eng. Sc.</td>
<td>1988</td>
</tr>
</tbody>
</table>

Major: Mechanical Engineering
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

TITLE i
APPROVAL SHEET ii
VITA iii
ABSTRACT iv
ACKNOWLEDGEMENT vi
TABLE OF CONTENTS vii
NOMENCLATURE viii
LIST OF FIGURES x
LIST OF TABLES xi

Chapter

I. INTRODUCTION 1

II. REVIEW OF SINGLE OBJECTIVE DESIGN OPTIMIZATION 4
   2.1 Importance of Optimization 4
   2.2 Problem Formulation 5
   2.3 Methods of Solution 8
   2.4 Existance and Uniqueness of An Optimal Solution 9

III. REVIEW OF MULTIPLE OBJECTIVE DESIGN OPTIMIZATION 11
   3.1 Introduction 11
   3.2 Problem Formulation 12
   3.3 The Conflict of Competing Objectives 13
   3.4 The Concept of Parato Optimality 14
   3.5 Methods of Solution 16
      3.5.1 Arithmetic Weighting Method 16
      3.5.2 Exponential Weighting Method 17
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5.3 Constraint Method</td>
<td>17</td>
</tr>
<tr>
<td>IV. THE FUNDAMENTAL LEVEL OBJECTIVE PROBLEM</td>
<td>19</td>
</tr>
<tr>
<td>4.1 Multiple Active Modes</td>
<td>19</td>
</tr>
<tr>
<td>4.2 Formulation Generalization</td>
<td>21</td>
</tr>
<tr>
<td>4.3 Application to Other Multiple Objective Problems</td>
<td>23</td>
</tr>
<tr>
<td>4.4 Application of the Fundamental Level Objective Formulation to Feasibility Restoration (FR)</td>
<td>24</td>
</tr>
<tr>
<td>V. GENERAL OPTIMIZATION PROCEDURE</td>
<td>26</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>26</td>
</tr>
<tr>
<td>5.2 Determination of Active Objective Functions</td>
<td>29</td>
</tr>
<tr>
<td>5.3 Parato Optimality Search</td>
<td>30</td>
</tr>
<tr>
<td>VI. SATISFACTION OF CONSTRAINTS</td>
<td>38</td>
</tr>
<tr>
<td>6.1 Satisfaction of the Leveling and Other Equality Constraints</td>
<td>38</td>
</tr>
<tr>
<td>6.2 Satisfaction of Inequality Constraints</td>
<td>42</td>
</tr>
<tr>
<td>6.3 General Procedure for Constraint Satisfaction</td>
<td>43</td>
</tr>
<tr>
<td>VII. ALGORITHM CONTROL PARAMETER SELECTION</td>
<td>44</td>
</tr>
<tr>
<td>7.1 Step Size and Reduction Attempt</td>
<td>44</td>
</tr>
<tr>
<td>7.2 Convergence Parameter</td>
<td>45</td>
</tr>
<tr>
<td>7.3 Constraint Linearity Band Width Parameter</td>
<td>46</td>
</tr>
<tr>
<td>7.4 Penalty Function Constant</td>
<td>46</td>
</tr>
<tr>
<td>7.5 Constraint Band Width Parameter</td>
<td>47</td>
</tr>
<tr>
<td>7.6 Weighting Parameter</td>
<td>48</td>
</tr>
<tr>
<td>VIII. EXAMPLES</td>
<td>50</td>
</tr>
<tr>
<td>8.1 Mathematical Test Problem</td>
<td>50</td>
</tr>
</tbody>
</table>
8.2 Design Example - Optimal Frequency Separation

Problem of "T" Ring Stiffened Cylindrical Shell

IX. DISUSSION OF RESULTS

9.1 Two Variable Mathematical Test Problem

9.2 The Six Variable Design Problem

9.3 General Observations

X. CONCLUSION

REFERENCES

APPENDIX A. USER INSTRUCTION FOR CADOP8

APPENDIX B. PROGRAM LISTING OF CADOP8
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

$\alpha_i =$ step size

$\epsilon_m =$ level function

$\eta =$ reduction attempt

$\sigma =$ slack variable

$< \phi > =$ bracket function

$\nabla =$ gradient operator

$\omega =$ 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
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I.</td>
<td>Conflict Condition between Two Objective Functions</td>
<td>15</td>
</tr>
<tr>
<td>II.</td>
<td>Parato Optimum Search for Four Different Procedures</td>
<td>27</td>
</tr>
<tr>
<td>III.</td>
<td>Fully Constrained Optimum of A Single Objective Function</td>
<td>35</td>
</tr>
<tr>
<td>IV.</td>
<td>Partially Constrained Optimum of A Single Objective Function</td>
<td>36</td>
</tr>
<tr>
<td>V.</td>
<td>Typical Shell Cross-Section</td>
<td>54</td>
</tr>
<tr>
<td>VI.</td>
<td>CADOP8 Search Path for the Unconstrained Two Objective Function Problem</td>
<td>57</td>
</tr>
<tr>
<td>VII.</td>
<td>CADOP5 Search Path for the Arithmetic Weighting Form</td>
<td>58</td>
</tr>
<tr>
<td>VIII.</td>
<td>CADOP8 Search Path for the Constrained Two Objective Function Problem</td>
<td>62</td>
</tr>
<tr>
<td>IX.</td>
<td>CADOP8 Search Path for the Unconstrained Three Objective Function Problem</td>
<td>63</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Arithmetic Weighting Method and Level Function Method Search Paths for the Two Variable Unconstrained Problem</td>
<td>56</td>
</tr>
<tr>
<td>II. CADOP8 Search Paths for Constrained Two Objective Function Problem and Unconstrained Three Objective Function Problem</td>
<td>61</td>
</tr>
<tr>
<td>III. SUBSHL7 and CADOP8 Search Paths from the Starting Point of Ref.[14]</td>
<td>65</td>
</tr>
<tr>
<td>IV. SUBSHL7 and CADOP8 Search Paths from the Starting Point Nearby the Optimum of Ref.[14], First Set</td>
<td>67</td>
</tr>
<tr>
<td>VI. SUBSHL7 and CADOP8 Search Paths from the Starting Point Nearby the Optimum of Ref.[14], Second Set</td>
<td>68</td>
</tr>
<tr>
<td>IV. Objective and Level Function values of CADOP8 Search from the Starting Point of Ref.[14]</td>
<td>70</td>
</tr>
</tbody>
</table>
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 Goicoechea 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 mechanisms; Yoshimura, et al [9] who derive the conditions for the Parato optimum

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 consciously 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 integer 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 practical 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

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

subject to the inequality constraints

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

and/or the equality constraints
\[ h_k(x_i) = 0 \quad k = 1, 2, \ldots K \quad (2.3) \]

A "side or regional" constraint form of inequality constraints

\[ x_i^l \leq x_i \leq x_i^u \quad (2.4) \]

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 increase 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, $\mathbf{x}^0$, 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

$$\mathbf{x}^r = \mathbf{x}^{r-1} + \alpha^* S^r$$  \hspace{1cm} (2.5)

where $r$ is the iteration number, and $S$ is a search direction vector in the design space. The scalar quantity $\alpha^*$ defines the distance that one wishes to move in direction $S$. A variety of methods involving ordinary and variational calculus, mathematical programming and optimal criteria, etc are available to search along the $S$ vector and define the scalar $\alpha^*$. 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 techniques, 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 cannot 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

$$f_q(\bar{x}_i) = S\min f_q(x_i) \quad q = 1, 2, \ldots, Q$$

(3.1)
where \( S_{\text{min}} f_q(x_i) \) 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 \( S_{\text{min}} f_q(x_i) \) 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:

\[
\text{Find } \bar{x}_i \text{ such that }
\]

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

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 \( \nabla f_1 \) and \( \nabla f_2 \) 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

\[
C[f_q(x_i)] = \sum_{q=1}^{Q} w_q f_q(x_i) \quad i = 1, 2, \ldots, I \quad (3.3)
\]

\[
q = 1, 2, \ldots, Q
\]

where \( w_q \) are positive weighting factors denoting the
relative importances of the various objective functions for minimization. It is usually assumed that

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

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:

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

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

(3) Constraint Method

One can generate the trade-off curves by the constrained method of Ref.[5] which states:
minimize $f_1(x_i)$ \hspace{1cm} i = 1,2,\ldots,I \hspace{1cm} (3.6)

where the constraints

$$f_q(x_i) \leq a_q \hspace{1cm} q = 1,2,\ldots,Q$$

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 critical. 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 $x_i$ such that

$$f_q(x_i) = \max \left[ \min f_q(x_i) \right] \quad q \in QA \quad (4.1)$$

where the $\max \left[ \min f_q(x_i) \right]$ is the maximum of the smallest of the set $QA$ 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

$$C[f_q(x_i)] = -\min f_q(x_i) \quad (4.2)$$
This problem is a multiple objective problem. Unfortunately, conventional formulation of multiple objective problems as discussed in chapter 3 and associated solution methodologies 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

\[ f_q(\bar{x}_i) = \min \left[ -\min f_q(x_i) \right] \quad (4.3) \]

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

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

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

\[ |f_1 - f_2| \leq 0. \]

But this is only true if

\[ f_1 - f_2 = 0. \]

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

\[ f_{q'} - f_q = 0 \quad q = 2, 3, \ldots, Q \quad (4.5) \]

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

Find \( \bar{x}_i \) such that

\[ f_q(\bar{x}_i) = \min f_{q'}(x_i) \quad (4.6) \]

subject to \( q-1 \) equality "leveling" constraints

\[ \epsilon_{q-1} = f_{q'} - f_q = 0 \quad q \neq q' \quad (4.7) \]

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

\[ \epsilon_m = |f_1 - f_{1+m}| = 0 \quad m = 1, 2, \ldots, Q_{A-1} \quad (4.8) \]
Here the functions are renumbered and $f_1$ replaces $f_q'$. Due to the existence 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 methodology 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

$$C[f_q'(\bar{x}_i)] = \min \{ \lambda_q f_q'(x_i) \} \quad q = 1, 2, \ldots Q \quad (4.9)$$

By substituting $\lambda_q f_q'$ for $f_q(x_i)$ of Eq. (4.1) through (4.2) where $f_q'$ are the functions to be simultaneously minimized and where $\lambda_q$ 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

$$f_q(\bar{x}_i) = \min f_q(x_i) \quad (4.10)$$
subject to the constraints

\[ \varepsilon_m = |f_1(x_i) - \left(\frac{A_q}{A_1}\right)f_q(x_i)| = 0 \quad m = 1, 2, \ldots, Q-1 \quad (4.11) \]

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 \( \bar{x}_i \) such that
\[ g_j(x_i) = \min \left\{ \max g_j(x_i) \right\} \quad j \in J_v \quad (4.12) \]

\[ h_k(x_i) = \min \left\{ \max h_k(x_i) \right\} \quad (4.13) \]

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

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

The third method is to first reduce all $f_q$ on one move then to reduce all $\xi_m$ on the next. The process is repeated until all $f_q$ and all $\xi_m$ 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 $\xi_m$ simultaneously until $\xi_m = 0$. The process
Starting Point

\begin{align*}
\min f_1 &= 2. \\
f_1/f_2 &= 0.1342
\end{align*}

\begin{align*}
\min f_2 &= 4. \\
f_1/f_2 &= 2.83
\end{align*}

\begin{itemize}
  \item the first procedure
  \item the second procedure
  \item the third procedure
  \item the fourth procedure
\end{itemize}

\text{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 $\xi_m$ 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 $\xi_m$ 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 objectives 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 Pareto 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 \( x_B^r \), evaluate the composite objective function value \( C_q(x_B^r) \)

\[
C_q(x) = f_q(x) + p_q(x) \quad q = 1, 2, \ldots, Q \quad (5.1)
\]
$$\text{P}_q(x) = \max | \lambda_{qj} < g_j(x) > \ or \ \lambda_{qm} < \varepsilon_m(x) > | \ (5.2)$$

$$\lambda_{qm} = \left\{ \begin{array}{ll}
2 |\nabla f_q|^2 / \nabla \varepsilon_m \cdot \nabla f_q & \varepsilon_m \leq \varepsilon_{m1} \\
K_1 & \varepsilon_m > \varepsilon_{m1}
\end{array} \right. \ (5.3)$$

$$\lambda_{qj} = \left\{ \begin{array}{ll}
2 |\nabla f_q|^2 / \nabla g_j \cdot \nabla f_q & g_j \leq \varepsilon_{j1} \\
K_1 & g_j > \varepsilon_{j1}
\end{array} \right. \ (5.4)$$

$$\langle \phi \rangle = 0 \quad \phi < 0 \ (5.5)$$

$$\langle \phi \rangle = \phi \quad \phi > 0$$

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

3. At point \( x_B \) find \( \sigma_q \) and \( S_i \) so as to:

Maximize \[ \sum_{q=1}^{Q} \sigma_q \] \quad \sigma_q > 0 \ (5.6)

subject to the conditions

\[(S_i)^T \nabla f_q(x_i) + \sigma_q \leq 0 \quad q = 1, 2, \ldots, Q \ (5.7)\]
\[
(S_i)^T \nabla \xi_m(x_i) + \xi_m(x_i) = 0 \quad m = 1,2,\ldots,Q-1 \quad (5.8)
\]
\[
(S_i)^T \nabla g_j(x_i) + g_j(x_i) \leq 0 \quad j = 1,2,\ldots,J_a \quad (5.9)
\]
\[
(S_i)^1 \leq S_i \leq (S_i)^u \quad (5.10)
\]
\[
S_i^1 = \begin{cases} 
 x_i^1 - x_i & \text{if } x_i - x_i^1 < \alpha_i^f \\
 -\alpha_i^f & \text{otherwise}
\end{cases} \quad (5.11)
\]
\[
S_i^u = \begin{cases} 
 x_i^u - x_i & \text{if } x_i^u - x_i < \alpha_i^f \\
 -\alpha_i^f & \text{otherwise}
\end{cases} \quad (5.12)
\]

where \( \alpha_q \) is a slack variable, \( \alpha_i^f \) is the maximum step size in the direction of \( x_i \), \( \xi_m \) 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 \( \alpha_q \) 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

\[
|g^r| < e_3 \quad (5.13)
\]

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

\[ x_C^r = x_B^r + s_B^r \]  \hspace{1cm} (5.14)

5. If any \[ c_q(x_C^r) > c_q(x_B^r) \]  \hspace{1cm} (5.15)
call \[ x_B^{r+1} = x_B^r \], then increase \( r \) by 1 and repeat step 1 to 4 with \( \alpha_i^f \) halved.

6. Otherwise call

\[ x_B^{r+1} = x_C^r \]  \hspace{1cm} (5.16)

Now if for all objective functions

\[ \left| \left[ \frac{c_q(x_B^{r+1}) - c_q(x_B^r)}{c_q(x_B^r)} \right] \right| \]

or

\[ \left| c_q(x_B^{r+1}) - c_q(x_B^r) \right| \]

where \( e_4 \) is an arbitrary small objective function convergence parameter, then the design is considered a Pareto 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 Pareto
optimal search. A simplified approach can be used by considering only one of the $f_q$ in Eq. (5.7). The $\xi_m$ 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 $\sigma_q$ value of equation (5.6) is to be maximized. This means the values of $S_i^T \nabla f_q$ 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 $\alpha_i^f$ 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 $\alpha_i^0$ would produce convergence without any need for step size reduction and without needs for design comparisons. For a problem such
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 $\alpha_i$ 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 $\xi_m$ 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 $\alpha_i$ is insufficient to reduce the $\xi_m$ to zero, slack variables $\sigma_m$ are added. These slack variables must be minimized and ultimately vanish. The minimization of these variables must take precedence 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 \( x_B^R \), evaluate the composite objective function value \( C_m(x_B) \) where

\[
C_m(x) = \xi_m(x) + P_m(x) \quad m = 1, 2, \ldots, Q-1 \quad (6.1)
\]

\[
P_m(x) = \max \left( \lambda_m \langle g_j(x) \rangle \right) \quad (6.2)
\]

\[
\lambda_m = \begin{cases} 
2|\nabla \xi_m|^2/\sqrt{g_j \cdot \nabla \xi_m} & g_j \leq e_j1 \\
K_1 & g_j > e_j1 
\end{cases} \quad (6.3)
\]

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

2. At point \( x_B^R \) find \( \sigma_q, \sigma_m \) and \( S_i \) so as to

\[
\text{maximize} \quad \sum_{q=1}^{Q} \sigma_q - W_1 \sum_{m=1}^{Q-1} \sigma_m \quad (6.4)
\]
subject to the conditions

\[ (S_i)^T \nabla f_q(x_i) + \sigma_q \leq 0 \quad q = 1, 2, \ldots, Q \]  \hspace{1cm} \text{(6.5)}

\[ (S_i)^T \nabla \epsilon_m(x_i) - \sigma_m + \epsilon_m(x_i) = 0 \quad m = 1, 2, \ldots, Q-1 \]  \hspace{1cm} \text{(6.6)}

\[ (S_i)^T \nabla g_j(x_i) + g_j(x_i) \leq 0 \quad j \in J_a, \quad g_j(x_i) > -e_{j2} \]  \hspace{1cm} \text{(6.7)}

\[ \sigma_q \geq 0 \quad \text{or} \quad \sigma_q < 0 \]  \hspace{1cm} \text{(6.8)}

\[ 0 \leq \sigma_m < \epsilon_m \]  \hspace{1cm} \text{(6.9)}

\[ (S_i)^l \leq S_i \leq (S_i)^u \]  \hspace{1cm} \text{(6.10)}

Here \( \sigma_q \) and \( \sigma_m \) 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

\[ S_i^l = \begin{cases} x_i^l - x_i & \text{if } x_i - x_i^l < \alpha_i^\epsilon \\ - \alpha_i^\epsilon & \text{otherwise} \end{cases} \]  \hspace{1cm} \text{(6.11)}
\[ S_i^u = \begin{cases} x_i^u - x_i & \text{if } x_i^u - x_i < \alpha_i^\epsilon \\ \alpha_i^\epsilon & , \text{otherwise} \end{cases} \]

where \( \alpha_i^\epsilon \) is the step size for the reduction of \( \xi_m \). It is a specified maximum limit on the change in variables \( S_i \).

3. Define a comparison base

\[ X_c^r = x_B^r + S_c^r \quad (6.12) \]

4. if \([ c_m(x_c^r) ]_{\text{max}} > [ c_m(x_B^r) ]_{\text{max}}\) \hspace{1cm} (6.13)

call \( x_B^{r+1} = x_B^r \) and increase \( r \) by 1 and repeat step 2 to 4 with \( \alpha_i^\epsilon \) halved.

5. Otherwise call \( x_B^{r+1} = x_c^r \) \hspace{1cm} (6.14)

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

The DFP procedure of Eqs. (6.4-6.11) is formulated to reduce \( \xi_m \) 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 \( -(S_i)^T \cdot \nabla \xi_m(x_1) \) is the estimated amount reduction in \( \xi_m \) after a move in \( S_i \) direction. The smaller the value of \( \sigma_m \), the greater the
value of the reduction in $\varepsilon_m$. Thus, $\sigma_m$ must be minimized for a maximum reduction of $\varepsilon_m$. 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 $\sigma_q$ in Eqs. (6.4-6.5). Here, the maximization of $\sigma_q$ 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 $\varepsilon_m$ 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 $\varepsilon_m$ 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 $\varepsilon_m$ and $\sigma_m$ with the $h_k$ and with the $\sigma_k$.

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 reduction DFP. The search procedure of section 6.1 is then invoked until convergence 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 \( e_1, e_2, e_3, e_4, e_5, e_6, e_j, K_1, \alpha_i^f, \) and \( \alpha_i^\varepsilon \). 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 parameters warrants their adoption.

(1) Step size and reduction attempt

Step size \( \alpha_i^\varepsilon \) is a specified maximum limit on the change in variable \( x_i \). In this procedure, the initial step size \( \alpha_i^\varepsilon \) for boundary restoration search and the initial step size \( \alpha_i^f \) for Pareto optimality search are given by

\[
\alpha_i^\varepsilon = \frac{\varepsilon_m^*(x_B)}{\max(\varepsilon_m^*, i)} \left/ \frac{\nabla \varepsilon_m^*}{|\nabla \varepsilon_m^*|^2} \right. \quad (7.1)
\]

\[
\alpha_i^f = \frac{f_q^*(x_B)}{\max(f_q^*, i)} \left/ \frac{\nabla f_q^*}{|\nabla f_q^*|^2} \right. \quad (7.2)
\]

where \( \varepsilon_m^* \) is associated with the \( m \) producing \( \max \varepsilon_m \), \( f_q^* \) is associated with \( q \) producing \( \max f_q \), and \( \eta \) is arbitrary selected. Here \( \eta \) is the estimated fraction change in \( \varepsilon_m^* \) or \( f_q^* \) if a move were made in the \( \nabla \varepsilon_m^* \) or \( \nabla f_q^* \) direction with components limited to \( \alpha_i^\varepsilon \) or \( \alpha_i^f \) respectively. Thus
\( \eta \) 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 \( \eta \) is recommended.

(2) Convergence parameters

In the boundary restoration search, all \( \xi_m \) equal or near zero are assumed convergence. An arbitrary 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 \( \xi_m \leq 0.001 \). In Parato optimality search, if the magnitude 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 $\xi_m$, 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,

$$g_j(x_i) = \frac{(B_j - U_j)}{U_j} U_j \leq 0$$ (7.3)

where $B_j$ represents the controlled behavior and $U_j$ the upper limit on behavior, even for highly nonlinear functions. For $\xi_m$ constraints in Pareto 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

$$e_{j2} = N \min \left( \frac{\alpha_i}{g_{j,i}(x_i)} \right) g_j^2 \quad (7.4)$$

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 $\nabla g_j$ 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 similarly 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 $\sigma_q$ and $\sigma_m$. This large value causes the $\sigma_q$ 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 $\varepsilon_m$ 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.

$$W_1 = M \left( \frac{\sum_{q=1}^{Q} \sum_{i=1}^{I} |\varepsilon_{f_q,i}|}{\sum_{m=1}^{Q-1} \sum_{i=1}^{I} |\alpha_{q,i} \varepsilon_{m,i}|} \right) \text{ or } \left( \sum_{m=1}^{Q-1} \sum_{i=1}^{I} |\varepsilon_{m,i}| \right)_{\text{min}} \quad (7.5)$$
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

\[ f_1(x_i) = \left(\frac{9}{25} x_1^2 + x_2^2 \right)^{0.5} + 2 \]  \hspace{1cm} (8.1)

\[ f_2(x_i) = \frac{(x_1 - 8)^2}{9} + \frac{(x_2 - 8)^2}{16} + 4 \]  \hspace{1cm} (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:

\[ C[f_q(x_i)] = \min \left[ f_1 + 2.0 f_2 \right] \]  \hspace{1cm} (8.3)

From the POS of Eq. (8.3), one finds \( f_1/f_2 = 1.8347 \). Therefore, the corresponding weighting factors of 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:

\[ C[f_q(x_i)] = \min \left[ f_1 , 1.8347 f_2 \right] \]  \hspace{1cm} (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

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

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

For unconstrained problem, the POS is located at a point where objectives 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

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

is added. This example shows how the Eq. (4.3) can be
solved with an arbitrary weighting factor set \( A_1 = 1.0 \), \( A_2 = 1.8347 \), 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:

\[
C[f_q(x_1)] = \min \left[ f_1, 1.8347 f_2, 1.0052 f_3 \right] \quad (8.7)
\]

The purpose of this example is to show how the feasibility of the equality constraint \( \epsilon_1 \) and \( \epsilon_2 \) 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 maximizes the separation between the lowest two natural frequencies of vibration for a "T" ring stiffened cylindrical shell under hydrostatic pressure. The coalescence 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 \( Q = 3 \).
Fig 5. Typical shell cross-section [14]
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.
<table>
<thead>
<tr>
<th>nₙ</th>
<th>nᵢ</th>
<th>C[f(x₁)]</th>
<th>x₁</th>
<th>x₂</th>
<th>nᵢ</th>
<th>f₁</th>
<th>f₂</th>
<th>x₁</th>
<th>x₂</th>
<th>ε₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6</td>
<td>27.419</td>
<td>1.</td>
<td>.</td>
<td>6</td>
<td>6.023</td>
<td>9.694</td>
<td>1.</td>
<td>6.</td>
<td>9.757</td>
</tr>
<tr>
<td>1</td>
<td>12</td>
<td>19.567</td>
<td>5.4</td>
<td>2.5</td>
<td>12</td>
<td>9.474</td>
<td>7.566</td>
<td>2.341</td>
<td>7.341</td>
<td>4.444</td>
</tr>
<tr>
<td>2</td>
<td>18</td>
<td>18.665</td>
<td>8.7</td>
<td>6.4</td>
<td>18</td>
<td>10.665</td>
<td>6.082</td>
<td>3.661</td>
<td>6.399</td>
<td>0.475</td>
</tr>
<tr>
<td>4</td>
<td>* 46</td>
<td>18.066</td>
<td>6.412</td>
<td>4.176</td>
<td>32</td>
<td>9.112</td>
<td>5.029</td>
<td>5.207</td>
<td>6.386</td>
<td>0.115</td>
</tr>
<tr>
<td>5</td>
<td>* 58</td>
<td>17.885</td>
<td>7.293</td>
<td>4.826</td>
<td>38</td>
<td>8.735</td>
<td>4.820</td>
<td>5.678</td>
<td>5.736</td>
<td>0.109</td>
</tr>
<tr>
<td>6</td>
<td>* 72</td>
<td>17.874</td>
<td>7.058</td>
<td>4.965</td>
<td>44</td>
<td>8.550</td>
<td>4.710</td>
<td>6.546</td>
<td>5.241</td>
<td>0.092</td>
</tr>
<tr>
<td>7</td>
<td>* 84</td>
<td>17.873</td>
<td>7.194</td>
<td>4.955</td>
<td>* 50</td>
<td>8.509</td>
<td>4.664</td>
<td>7.218</td>
<td>4.860</td>
<td>0.055</td>
</tr>
<tr>
<td>8</td>
<td>* 100</td>
<td>17.872</td>
<td>7.112</td>
<td>4.970</td>
<td>115</td>
<td>8.551</td>
<td>4.661</td>
<td>7.111</td>
<td>4.971</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>* 112</td>
<td>17.873</td>
<td>7.120</td>
<td>4.967</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>128</td>
<td>17.873</td>
<td>7.119</td>
<td>4.967</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

nₙ : Base number   nᵢ : 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 magnitude 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
<table>
<thead>
<tr>
<th>n_b</th>
<th>( \xi_1 )</th>
<th>( g_1 )</th>
<th>( f_1 )</th>
<th>( f_2 )</th>
<th>( x_1 )</th>
<th>( x_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>9.757</td>
<td>-33.00</td>
<td>8.023</td>
<td>9.694</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4.444</td>
<td>-40.13</td>
<td>9.474</td>
<td>7.586</td>
<td>2.341</td>
<td>7.341</td>
</tr>
<tr>
<td>2</td>
<td>0.475</td>
<td>-39.25</td>
<td>10.69</td>
<td>6.082</td>
<td>3.681</td>
<td>8.399</td>
</tr>
<tr>
<td>3*</td>
<td>0.311</td>
<td>-27.80</td>
<td>9.567</td>
<td>5.383</td>
<td>4.543</td>
<td>7.059</td>
</tr>
<tr>
<td>4</td>
<td>0.115</td>
<td>-18.25</td>
<td>9.112</td>
<td>5.029</td>
<td>5.207</td>
<td>6.388</td>
</tr>
<tr>
<td>5</td>
<td>0.109</td>
<td>-7.044</td>
<td>8.735</td>
<td>4.520</td>
<td>5.878</td>
<td>5.736</td>
</tr>
<tr>
<td>6*</td>
<td>0.022</td>
<td>-1.838</td>
<td>8.668</td>
<td>4.737</td>
<td>6.213</td>
<td>5.529</td>
</tr>
<tr>
<td>7</td>
<td>0.003</td>
<td>0.0183</td>
<td>8.648</td>
<td>4.715</td>
<td>6.213</td>
<td>5.529</td>
</tr>
<tr>
<td>8</td>
<td>0.000</td>
<td>0.0000</td>
<td>8.649</td>
<td>4.714</td>
<td>6.327</td>
<td>5.459</td>
</tr>
</tbody>
</table>

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

Table 2  CAD0P8 search paths for constrained two objective function problem and unconstrained three objective function problem
min \( f \)

\[
\frac{f_1}{f_2} = \frac{1.8347}{0.0} = 2.83
\]

**Objective function** \( f \)  

**Behavior constraint** \( g \)

Design search

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 objective 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 SUBSHL7 locates a design with a greater frequency separation CADOP8 has greatly superior convergence properties. The CADOP8 code requires in excess 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.
<table>
<thead>
<tr>
<th></th>
<th>Initial Design</th>
<th>SBSSL7 Termination</th>
<th>CADOP8 Termination</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1.000</td>
<td>1.519</td>
<td>1.188</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1.000</td>
<td>0.523</td>
<td>0.325</td>
</tr>
<tr>
<td>$x_3$</td>
<td>1.000</td>
<td>1.498</td>
<td>1.188</td>
</tr>
<tr>
<td>$x_4$</td>
<td>10.000</td>
<td>4.996</td>
<td>15.038</td>
</tr>
<tr>
<td>$x_5$</td>
<td>20.000</td>
<td>55.169</td>
<td>30.616</td>
</tr>
<tr>
<td>$x_6$</td>
<td>10.000</td>
<td>30.758</td>
<td>18.268</td>
</tr>
<tr>
<td>$g_1$</td>
<td>-0.539</td>
<td>-0.790</td>
<td>-0.661</td>
</tr>
<tr>
<td>$g_2$</td>
<td>-0.472</td>
<td>-0.306</td>
<td>-0.383</td>
</tr>
<tr>
<td>$g_3$</td>
<td>0.001</td>
<td>-0.116</td>
<td>0.000</td>
</tr>
<tr>
<td>$g_4$</td>
<td>-0.389</td>
<td>-0.467</td>
<td>-0.416</td>
</tr>
<tr>
<td>$g_5$</td>
<td>-0.947</td>
<td>-0.995</td>
<td>-0.904</td>
</tr>
<tr>
<td>$g_6$</td>
<td>0.000</td>
<td>-0.014</td>
<td>0.000</td>
</tr>
<tr>
<td>$g_7$</td>
<td>-10.000</td>
<td>-4.996</td>
<td>-15.038</td>
</tr>
<tr>
<td>$g_8$</td>
<td>-0.598</td>
<td>-0.545</td>
<td>-0.579</td>
</tr>
<tr>
<td>$g_9$</td>
<td>-0.534</td>
<td>-0.717</td>
<td>-0.676</td>
</tr>
<tr>
<td>$g_{10}$</td>
<td>0.035</td>
<td>-0.001</td>
<td>0.00000016</td>
</tr>
<tr>
<td>$g_{11}$</td>
<td>-0.967</td>
<td>-0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>$\omega_1(n_1 m_1)$</td>
<td>25.76(2,1)</td>
<td>30.458(2,1)</td>
<td>37.065(2,1)</td>
</tr>
<tr>
<td>$\omega_2(n_2 m_2)$</td>
<td>33.04(3,1)</td>
<td>72.897(3,1)</td>
<td>64.363(3,1)</td>
</tr>
<tr>
<td>$\omega_3(n_3 m_3)$</td>
<td>44.29(3,2)</td>
<td>72.897(1,1)</td>
<td>64.363(2,2)</td>
</tr>
<tr>
<td>$\omega_4(n_4 m_4)$</td>
<td>57.00(2,2)</td>
<td>72.924(13,1)#</td>
<td>64.363(1,1)</td>
</tr>
<tr>
<td>$f_1$</td>
<td>7.279</td>
<td>30.458</td>
<td>27.298</td>
</tr>
<tr>
<td>$f_2$</td>
<td>18.531</td>
<td>30.458</td>
<td>27.298</td>
</tr>
<tr>
<td>$f_3$</td>
<td>31.242</td>
<td>30.485</td>
<td>27.298</td>
</tr>
<tr>
<td>$e_1$</td>
<td>11.252</td>
<td>0.00006</td>
<td>0.00021</td>
</tr>
<tr>
<td>$e_2$</td>
<td>23.963</td>
<td>0.02659</td>
<td>0.00023</td>
</tr>
<tr>
<td>$n_i$</td>
<td>0</td>
<td>1379</td>
<td>117</td>
</tr>
<tr>
<td>$n_f$</td>
<td>1</td>
<td>27934</td>
<td>844</td>
</tr>
<tr>
<td>$n_g$</td>
<td>11</td>
<td>59723</td>
<td>4532</td>
</tr>
</tbody>
</table>

$n_i$ : 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. SBSSL7 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
<table>
<thead>
<tr>
<th>Initial Design</th>
<th>SBSHL7 Termination</th>
<th>CADOP8 Termination</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1.5203</td>
<td>1.5237</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0.5464</td>
<td>0.5467</td>
</tr>
<tr>
<td>$x_3$</td>
<td>1.5203</td>
<td>1.5237</td>
</tr>
<tr>
<td>$x_4$</td>
<td>3.7510</td>
<td>3.8528</td>
</tr>
<tr>
<td>$x_5$</td>
<td>54.6339</td>
<td>55.5451</td>
</tr>
<tr>
<td>$x_6$</td>
<td>32.0743</td>
<td>32.1279</td>
</tr>
<tr>
<td>$g_1$</td>
<td>-0.7911</td>
<td>-0.7919</td>
</tr>
<tr>
<td>$g_2$</td>
<td>-0.3161</td>
<td>-0.3506</td>
</tr>
<tr>
<td>$g_3$</td>
<td>-0.1190</td>
<td>-0.1191</td>
</tr>
<tr>
<td>$g_4$</td>
<td>-0.4642</td>
<td>-0.4670</td>
</tr>
<tr>
<td>$g_5$</td>
<td>-0.9975</td>
<td>-0.9973</td>
</tr>
<tr>
<td>$g_6$</td>
<td>-0.0000</td>
<td>-0.0000</td>
</tr>
<tr>
<td>$g_7$</td>
<td>-3.7510</td>
<td>-3.8528</td>
</tr>
<tr>
<td>$g_8$</td>
<td>-0.5417</td>
<td>-0.5415</td>
</tr>
<tr>
<td>$g_9$</td>
<td>-0.7612</td>
<td>-0.7167</td>
</tr>
<tr>
<td>$g_{10}$</td>
<td>0.0000</td>
<td>-0.0001</td>
</tr>
<tr>
<td>$g_{11}$</td>
<td>0.0000</td>
<td>0.0002</td>
</tr>
<tr>
<td>$\omega_1(n_1m_1)$</td>
<td>42.2877(2,1)</td>
<td>42.3573(2,1)</td>
</tr>
<tr>
<td>$\omega_2(n_2m_2)$</td>
<td>72.4108(3,1)</td>
<td>72.5438(13,1)*</td>
</tr>
<tr>
<td>$\omega_3(n_3m_3)$</td>
<td>72.9524(1,1)</td>
<td>72.5506(3,1)</td>
</tr>
<tr>
<td>$\omega_4(n_4m_4)$</td>
<td>74.8315(2,2)</td>
<td>73.0272(1,1)</td>
</tr>
<tr>
<td>$f_1$</td>
<td>30.1231</td>
<td>30.1865</td>
</tr>
<tr>
<td>$f_2$</td>
<td>30.6647</td>
<td>30.1933</td>
</tr>
<tr>
<td>$f_3$</td>
<td>32.5438</td>
<td>30.6699</td>
</tr>
<tr>
<td>$\varepsilon_1$</td>
<td>0.5415</td>
<td>0.0068</td>
</tr>
<tr>
<td>$\varepsilon_2$</td>
<td>2.4207</td>
<td>0.4834</td>
</tr>
<tr>
<td>$n_i$</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>$n_{o}$</td>
<td>1</td>
<td>294</td>
</tr>
<tr>
<td>$n_g$</td>
<td>11</td>
<td>637</td>
</tr>
</tbody>
</table>

$n_i$: number of design iterations  
$n_{o}$: 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
<table>
<thead>
<tr>
<th></th>
<th>Initial Design</th>
<th>SBSHL7 Termination</th>
<th>CADOP8 Termination</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1.518</td>
<td>1.5182</td>
<td>1.5208</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0.539</td>
<td>0.5459</td>
<td>0.5474</td>
</tr>
<tr>
<td>$x_3$</td>
<td>1.331</td>
<td>1.5125</td>
<td>1.5208</td>
</tr>
<tr>
<td>$x_4$</td>
<td>4.826</td>
<td>3.9533</td>
<td>3.8646</td>
</tr>
<tr>
<td>$x_5$</td>
<td>55.040</td>
<td>55.0233</td>
<td>55.2650</td>
</tr>
<tr>
<td>$x_6$</td>
<td>31.690</td>
<td>32.0522</td>
<td>32.2157</td>
</tr>
<tr>
<td>$e_1$</td>
<td>-0.7904</td>
<td>-0.7906</td>
<td>-0.7913</td>
</tr>
<tr>
<td>$e_2$</td>
<td>-0.3070</td>
<td>-0.3076</td>
<td>-0.3067</td>
</tr>
<tr>
<td>$e_3$</td>
<td>-0.1163</td>
<td>-0.1165</td>
<td>-0.1176</td>
</tr>
<tr>
<td>$e_4$</td>
<td>-0.4664</td>
<td>-0.4661</td>
<td>-0.4670</td>
</tr>
<tr>
<td>$e_5$</td>
<td>-0.9941</td>
<td>-0.9971</td>
<td>-0.9973</td>
</tr>
<tr>
<td>$e_6$</td>
<td>-0.1232</td>
<td>-0.0037</td>
<td>0.00000000</td>
</tr>
<tr>
<td>$e_7$</td>
<td>-4.8260</td>
<td>-3.9533</td>
<td>-3.8646</td>
</tr>
<tr>
<td>$e_8$</td>
<td>-0.5433</td>
<td>-0.5418</td>
<td>-0.5413</td>
</tr>
<tr>
<td>$e_9$</td>
<td>-0.7172</td>
<td>-0.7173</td>
<td>-0.7174</td>
</tr>
<tr>
<td>$e_{10}$</td>
<td>-0.0007</td>
<td>-0.0007</td>
<td>0.000000105</td>
</tr>
<tr>
<td>$e_{11}$</td>
<td>-0.0008</td>
<td>-0.0032</td>
<td>-0.0000149</td>
</tr>
</tbody>
</table>

- $\omega_1(n_{1m_1})$: 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)*

- $r_1$: 30.4610 30.4658 30.4784
- $r_2$: 30.4615 30.4660 30.4789
- $r_3$: 30.7654 30.8380 30.4790
- $e_1$: 0.000552 0.00016 0.000510
- $e_2$: 0.304444 0.37202 0.000575
- $n_i$: 0 484 12
- $n_{i^*}$: 1 8803 79
- $n_{eg}$: 11 19167 500

$n_i$: number of design iterations

$n_{i^*}$: number of objective function evaluations

$n_{eg}$: 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. SUBSHEL7 and CADOP8 search paths from the starting point nearby the optimum of Ref. [14], second set

68
restoration worked well in all cases. From Figs. 6, 8 and 9, for example, one can see that the procedure moves quickly near the $\xi_m = 0$. 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 $\xi_m = 0$. constraint occurs by the 68th 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.
<table>
<thead>
<tr>
<th>Base No.</th>
<th>Objective Function</th>
<th>Level Function</th>
<th>Active Constraint Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( f_1 )</td>
<td>( f_2 )</td>
<td>( f_3 )</td>
</tr>
<tr>
<td>0(^a)</td>
<td>7.279</td>
<td>18.531</td>
<td>31.242</td>
</tr>
<tr>
<td>1</td>
<td>8.747</td>
<td>19.619</td>
<td>31.333</td>
</tr>
<tr>
<td>2</td>
<td>10.311</td>
<td>20.414</td>
<td>30.617</td>
</tr>
<tr>
<td>3</td>
<td>11.855</td>
<td>21.254</td>
<td>29.929</td>
</tr>
<tr>
<td>4</td>
<td>13.385</td>
<td>22.116</td>
<td>29.220</td>
</tr>
<tr>
<td>5</td>
<td>14.902</td>
<td>22.999</td>
<td>28.494</td>
</tr>
<tr>
<td>6</td>
<td>16.406</td>
<td>23.901</td>
<td>27.753</td>
</tr>
<tr>
<td>8(^*)</td>
<td>19.373</td>
<td>25.750</td>
<td>26.229</td>
</tr>
<tr>
<td>12</td>
<td>20.494</td>
<td>25.631</td>
<td>26.483</td>
</tr>
<tr>
<td>15</td>
<td>21.221</td>
<td>25.252</td>
<td>26.642</td>
</tr>
<tr>
<td>19</td>
<td>22.159</td>
<td>24.758</td>
<td>26.004</td>
</tr>
<tr>
<td>21</td>
<td>22.614</td>
<td>24.519</td>
<td>25.689</td>
</tr>
<tr>
<td>22</td>
<td>22.837</td>
<td>24.400</td>
<td>25.533</td>
</tr>
<tr>
<td>24</td>
<td>23.275</td>
<td>24.168</td>
<td>25.223</td>
</tr>
<tr>
<td>25</td>
<td>23.491</td>
<td>24.054</td>
<td>25.071</td>
</tr>
<tr>
<td>26</td>
<td>23.703</td>
<td>23.941</td>
<td>24.919</td>
</tr>
<tr>
<td>27</td>
<td>23.918</td>
<td>23.920</td>
<td>24.856</td>
</tr>
<tr>
<td>28</td>
<td>24.047</td>
<td>24.050</td>
<td>24.948</td>
</tr>
<tr>
<td>29</td>
<td>24.174</td>
<td>24.176</td>
<td>25.037</td>
</tr>
<tr>
<td>30</td>
<td>24.297</td>
<td>24.299</td>
<td>25.124</td>
</tr>
<tr>
<td>31</td>
<td>24.417</td>
<td>24.419</td>
<td>25.208</td>
</tr>
<tr>
<td>32</td>
<td>24.534</td>
<td>24.536</td>
<td>25.289</td>
</tr>
<tr>
<td>33</td>
<td>24.648</td>
<td>24.650</td>
<td>25.368</td>
</tr>
<tr>
<td>34</td>
<td>24.760</td>
<td>24.762</td>
<td>25.446</td>
</tr>
<tr>
<td>35</td>
<td>24.870</td>
<td>24.871</td>
<td>25.520</td>
</tr>
<tr>
<td>36</td>
<td>24.976</td>
<td>24.978</td>
<td>25.593</td>
</tr>
<tr>
<td>37</td>
<td>25.080</td>
<td>25.082</td>
<td>25.664</td>
</tr>
<tr>
<td>38</td>
<td>25.182</td>
<td>25.184</td>
<td>25.733</td>
</tr>
<tr>
<td>39</td>
<td>25.282</td>
<td>25.284</td>
<td>25.780</td>
</tr>
<tr>
<td>40</td>
<td>25.380</td>
<td>25.381</td>
<td>25.865</td>
</tr>
</tbody>
</table>

\( a \) : 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]
<table>
<thead>
<tr>
<th>Base No.</th>
<th>Objective Function</th>
<th>Level Function</th>
<th>Active Constraint Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f_1$</td>
<td>$f_2$</td>
<td>$f_3$</td>
</tr>
<tr>
<td>41</td>
<td>25.474</td>
<td>25.476</td>
<td>25.229</td>
</tr>
<tr>
<td>42</td>
<td>25.567</td>
<td>25.569</td>
<td>25.990</td>
</tr>
<tr>
<td>43</td>
<td>25.659</td>
<td>25.661</td>
<td>26.050</td>
</tr>
<tr>
<td>44</td>
<td>25.748</td>
<td>25.750</td>
<td>26.109</td>
</tr>
<tr>
<td>45</td>
<td>25.835</td>
<td>25.837</td>
<td>26.166</td>
</tr>
<tr>
<td>46</td>
<td>25.920</td>
<td>25.922</td>
<td>26.222</td>
</tr>
<tr>
<td>47</td>
<td>26.004</td>
<td>26.006</td>
<td>26.276</td>
</tr>
<tr>
<td>48</td>
<td>26.086</td>
<td>26.088</td>
<td>26.328</td>
</tr>
<tr>
<td>49</td>
<td>26.166</td>
<td>26.168</td>
<td>26.380</td>
</tr>
<tr>
<td>50</td>
<td>26.244</td>
<td>26.246</td>
<td>26.430</td>
</tr>
<tr>
<td>51</td>
<td>26.321</td>
<td>26.323</td>
<td>26.479</td>
</tr>
<tr>
<td>52</td>
<td>26.396</td>
<td>26.398</td>
<td>26.526</td>
</tr>
<tr>
<td>53</td>
<td>26.470</td>
<td>26.472</td>
<td>26.573</td>
</tr>
<tr>
<td>54</td>
<td>26.542</td>
<td>26.544</td>
<td>26.618</td>
</tr>
<tr>
<td>55</td>
<td>26.612</td>
<td>26.614</td>
<td>26.662</td>
</tr>
<tr>
<td>56</td>
<td>26.682</td>
<td>26.684</td>
<td>26.705</td>
</tr>
<tr>
<td>57</td>
<td>26.742</td>
<td>26.743</td>
<td>26.744</td>
</tr>
<tr>
<td>58</td>
<td>26.777</td>
<td>26.778</td>
<td>26.779</td>
</tr>
<tr>
<td>59</td>
<td>26.812</td>
<td>26.813</td>
<td>26.814</td>
</tr>
<tr>
<td>60</td>
<td>26.844</td>
<td>26.845</td>
<td>26.846</td>
</tr>
<tr>
<td>61</td>
<td>26.874</td>
<td>26.875</td>
<td>26.875</td>
</tr>
<tr>
<td>62</td>
<td>26.901</td>
<td>26.902</td>
<td>26.902</td>
</tr>
<tr>
<td>63</td>
<td>26.906</td>
<td>26.907</td>
<td>26.907</td>
</tr>
<tr>
<td>64</td>
<td>26.932</td>
<td>26.933</td>
<td>26.933</td>
</tr>
<tr>
<td>65</td>
<td>26.943</td>
<td>26.944</td>
<td>26.944</td>
</tr>
<tr>
<td>66b</td>
<td>26.958</td>
<td>26.958</td>
<td>26.958</td>
</tr>
<tr>
<td>67b</td>
<td>26.975</td>
<td>26.977</td>
<td>26.977</td>
</tr>
<tr>
<td>68</td>
<td>26.993</td>
<td>26.995</td>
<td>26.995</td>
</tr>
<tr>
<td>69</td>
<td>27.010</td>
<td>27.012</td>
<td>27.012</td>
</tr>
<tr>
<td>70</td>
<td>27.026</td>
<td>27.028</td>
<td>27.028</td>
</tr>
<tr>
<td>71</td>
<td>27.042</td>
<td>27.043</td>
<td>27.043</td>
</tr>
<tr>
<td>72</td>
<td>27.056</td>
<td>27.058</td>
<td>27.058</td>
</tr>
<tr>
<td>73</td>
<td>27.071</td>
<td>27.072</td>
<td>27.072</td>
</tr>
<tr>
<td>74</td>
<td>27.084</td>
<td>27.086</td>
<td>27.086</td>
</tr>
<tr>
<td>75</td>
<td>27.097</td>
<td>27.098</td>
<td>27.099</td>
</tr>
<tr>
<td>76</td>
<td>27.109</td>
<td>27.111</td>
<td>27.111</td>
</tr>
<tr>
<td>77</td>
<td>27.121</td>
<td>27.122</td>
<td>27.123</td>
</tr>
<tr>
<td>78</td>
<td>27.132</td>
<td>27.134</td>
<td>27.134</td>
</tr>
<tr>
<td>79</td>
<td>27.143</td>
<td>27.144</td>
<td>27.144</td>
</tr>
<tr>
<td>80</td>
<td>27.153</td>
<td>27.154</td>
<td>27.155</td>
</tr>
</tbody>
</table>

b : Step size switched to 0.2268 for optimality search
<table>
<thead>
<tr>
<th>Base No.</th>
<th>Objective Function</th>
<th>Level Function</th>
<th>Active Constraint Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f_1$</td>
<td>$f_2$</td>
<td>$f_3$</td>
</tr>
<tr>
<td>81</td>
<td>27.163</td>
<td>27.164</td>
<td>27.164</td>
</tr>
<tr>
<td>82</td>
<td>27.172</td>
<td>27.173</td>
<td>27.173</td>
</tr>
<tr>
<td>83</td>
<td>27.181</td>
<td>27.182</td>
<td>27.182</td>
</tr>
<tr>
<td>84</td>
<td>27.189</td>
<td>27.190</td>
<td>27.191</td>
</tr>
<tr>
<td>85</td>
<td>27.197</td>
<td>27.198</td>
<td>27.198</td>
</tr>
<tr>
<td>86</td>
<td>27.205</td>
<td>27.206</td>
<td>27.206</td>
</tr>
<tr>
<td>87</td>
<td>27.212</td>
<td>27.213</td>
<td>27.213</td>
</tr>
<tr>
<td>88</td>
<td>27.219</td>
<td>27.220</td>
<td>27.220</td>
</tr>
<tr>
<td>89</td>
<td>27.225</td>
<td>27.226</td>
<td>27.226</td>
</tr>
<tr>
<td>90</td>
<td>27.231</td>
<td>27.232</td>
<td>27.232</td>
</tr>
<tr>
<td>91</td>
<td>27.237</td>
<td>27.238</td>
<td>27.238</td>
</tr>
<tr>
<td>92</td>
<td>27.242</td>
<td>27.243</td>
<td>27.243</td>
</tr>
<tr>
<td>93</td>
<td>27.247</td>
<td>27.248</td>
<td>27.248</td>
</tr>
<tr>
<td>94</td>
<td>27.252</td>
<td>27.253</td>
<td>27.253</td>
</tr>
<tr>
<td>95</td>
<td>27.257</td>
<td>27.257</td>
<td>27.257</td>
</tr>
<tr>
<td>96</td>
<td>27.261</td>
<td>27.262</td>
<td>27.262</td>
</tr>
<tr>
<td>97</td>
<td>27.265</td>
<td>27.265</td>
<td>27.266</td>
</tr>
<tr>
<td>98</td>
<td>27.268</td>
<td>27.269</td>
<td>27.269</td>
</tr>
<tr>
<td>99</td>
<td>27.272</td>
<td>27.272</td>
<td>27.273</td>
</tr>
<tr>
<td>100</td>
<td>27.275</td>
<td>27.276</td>
<td>27.276</td>
</tr>
<tr>
<td>101</td>
<td>27.278</td>
<td>27.278</td>
<td>27.279</td>
</tr>
<tr>
<td>102</td>
<td>27.280</td>
<td>27.281</td>
<td>27.281</td>
</tr>
<tr>
<td>103</td>
<td>27.283</td>
<td>27.284</td>
<td>27.284</td>
</tr>
<tr>
<td>104</td>
<td>27.285</td>
<td>27.286</td>
<td>27.286</td>
</tr>
<tr>
<td>105</td>
<td>27.287</td>
<td>27.288</td>
<td>27.288</td>
</tr>
<tr>
<td>106</td>
<td>27.289</td>
<td>27.290</td>
<td>27.290</td>
</tr>
<tr>
<td>107</td>
<td>27.291</td>
<td>27.291</td>
<td>27.291</td>
</tr>
<tr>
<td>108</td>
<td>27.294</td>
<td>27.294</td>
<td>27.294</td>
</tr>
<tr>
<td>109</td>
<td>27.295</td>
<td>27.295</td>
<td>27.295</td>
</tr>
<tr>
<td>110</td>
<td>27.296</td>
<td>27.296</td>
<td>27.296</td>
</tr>
<tr>
<td>111</td>
<td>27.296</td>
<td>27.297</td>
<td>27.297</td>
</tr>
<tr>
<td>112</td>
<td>27.297</td>
<td>27.298</td>
<td>27.298</td>
</tr>
<tr>
<td>113</td>
<td>27.298</td>
<td>27.298</td>
<td>27.298</td>
</tr>
<tr>
<td>114</td>
<td>27.298</td>
<td>27.298</td>
<td>27.298</td>
</tr>
</tbody>
</table>
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


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


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 guarantee 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 of objective 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 \( x_i, i = 1,2,\ldots, IP \), and the parameters \( P_k, k = 1,2,\ldots, KP \) which result in the minimization of the objective functions \( f_q(x_i, P_k) \) subject to specified constraints. That is to find

\[
C[f_q(x_i)] = \min A_q f_q(P_k, x_i) \quad q = 1,2,\ldots,Q \quad (A1)
\]

while also satisfying the behavior constraints

\[
g_j(P_k, x_i) \leq 0 \quad j = 1,2,\ldots,JP \quad (A2)
\]

and the regional constraints

\[
x_{i1} \leq x_i \leq x_{iu} \quad (A3)
\]

where \( A_q \) are weighting factors, \( q = 1,2,\ldots,Q \).

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

\[ B_j(x_i, P_k) \leq U_j(x_i, P_k) \quad i = 1, 2, \ldots, IP \quad (A4) \]
\[ j = 1, 2, \ldots, JP \]
\[ k = 1, 2, \ldots, KP \]

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), \( j^* \) \quad [ * : 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
::

101OBJ = 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 are not 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 = \((9/25 \times x_1^2 + x_2^2)^{0.5} + 2\)
GO TO 201
12 OBJ = \((x_1 - 8)^2/9 + (x_2 - 8)^2/16 + 4\)
GO TO 201
13 OBJ = \([(x_1 - 8)^2 + x_2^2])^{0.5} + 4\)
201 IF = IF +1
RETURN
10000 GO TO (1, 2, 3), j
1 B = \((x_1)^2 + (x_2 - 10)^2\)
U = 49
GO TO 101
2 B = x
U = 1.
GO TO 101
3 B = x_2
U = x
101 \quad 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 \\
\[ g_j = \begin{cases} 
\frac{(B_j - U_j)}{U_j} & U_j \neq 0 \text{ and } B_j \neq 0 \\
(B_j - U_j) & U_j = 0 \text{ or } B_j = 0 
\end{cases} \] (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 \( q^\text{th} \) 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 maximum number of variables to (IP), or the maximum number of constraints to (JP), or the 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 properly 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 This program is modified from CADOP5 [19] by Wen-Tsia Liu

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,
IV,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)

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)

READ101,(D(I),I=1,IP)
17 FORMAT(1H0,' LOWER LIMITS OF DESIGN VARIABLES' )
READ1,(CQ(I),I=1,LP)
GO TO 23

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)
19 FORMAT(1H0,' UPPER LIMITS OF DESIGN VARIABLES' )

22 IF(KP.LE.0) GO TO 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(' 1 NO. 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.O) GO TO 620
610 IF(E.EQ.O) 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.O) GO TO 810
810 LL=N+1
KK=N+G
810 IF(L.EQ.O) GO TO 830
830 IF(G+E.EQ.O) 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.O) 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
1040  P=A(R,C)
1050  A(R,BB)=C
1060  DO 1080 J=1,B
1070  A(R,J)=A(R,J)/P
   LL=W+1
1080  DO 1180 1=1,LL
1090  IF(I.EQ.R) GO TO 1180
1100  DO 1170 J=1,B
1110  IF(J.EQ.C) GO TO 1170
1120  A(I,J)=A(I,J)-A(R,J)*A(I,C)
1130  IF(ABS(A(I,J)).GT..1E-4) GO TO 1170
1140  A(I,J)=0
1150 CONTINUE
1160 CONTINUE
1170 CONTINUE
   LL=W+1
1180  DO 1200 I=1,LL
1190  A(I,C)=0
1200 CONTINUE
1210  A(R,C)=1
1220  Q=0
1230  LL=N+G+L
   DO 1280 J=1,LL
1240  IF(A(W+1,J).GT.Q) GO TO 1280
1250  Q=A(W+1,J)
1260  C=J
1270 CONTINUE
1280 CONTINUE
1290 GO TO 900
1300 IF(W.EQ.M+1) GO TO 1360
1310 W=W-1
1320 IF(A(W+2,B).LT..1E-5) GO TO 1353
   K2=0
1330 RETURN
1340 CONTINUE
1350 IF(A(W+1,J).LT..1E-5) GO TO 1353
1360 DO 1358 I=1,LL
1370 IF(INT(A(I,BB)).LE.N+G+L) GO TO 1358
1380 DO 1356 J=1,B
1390  A(I,J)=0
1400 CONTINUE
1410 GO TO 1230
1420 CONTINUE
1430 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,
IV,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,NAACL,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 ATT(78),DELT(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+1P+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.ALPE)DELTU(K)=ALP
IF(DELTA(K).GT.ALPE)DELTA(K)=ALP
4 CONTINUE
LX=MM+2
ML=NACT+LP+LPE+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
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
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)
CONTINUE
DO 27 I=1,IP
J=J+1
A(J,I)=1.
27 A(J,B)=DELT(A(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=1A(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(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(A(I,B).GE.0.) GO TO 540
GT=GT+1
DO 550 K=1,B
ATT(L+ET+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
   SUM=0.0
   DO 115 I=1,IP
       SUM=SUM+SS(I)*SS(I)
       SUM=DSQRT(SUM)
   115 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),
1L,L,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
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)VF=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
101
IF(KL.EQ.0) ALPE=ALP
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)
DO 68 K=1,LP
   DO 68 I=1,IP
   A(K,I)=ABASE(K,I)
   DO 422 L=1,LPE
   DO 422 I=1,IP
   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
   XB(I)=X(I)
   DO 3750 J=1,JP
   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
   NACT=0
   DO 3820 M=1,IP
   A(NACT+LP,M)=A(K+LP,M)
   CONTINUE
   DO 3840 L=1,LPE
   DO 3840 I=1,IP
   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,LPE,SB,SBLST,TDIF)
   IF(ICONDE.EQ.0)GO TO 24
   35   MOVE=0
   KF=2
   IF(TEST.LT.ALPIGO 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 ATTEMPT=',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)RETURN
IF(L.EQ.1)GO TO 11
11 ST(L)=TO(L)
IF(IEMOVE.EQ.0)GO TO 1200
2080 DO 1080 L=LPE,1
IF(EO(L).GT.ED)GO TO 1200
1080 CONTINUE
LD=ED*0.1
GO TO 2080
1200 CONTINUE
INT=0
ME=0
DO 211 L=1,LPE
IF(EO(L).LT.0.1)NOPTN=L
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
IEMOV=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=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,IL
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(IEMOVE.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(IEMOVE.EQ.1) GO TO 317
DO 7 M=1,IP
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
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

107
IA(NACT) = J
IF (NACT.EQ.K) GO TO 5
DO 10 M = 1, IP
10 A(NACT + LP, M) = A(K + LP, M)
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
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
CONTINUE
GO TO 318
318 CONTINUE
IF (KK.EQ.1.AND.KL.EQ.0) GO TO 35
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)
CONTINUE
DO 331 I = 1, IP
IF (DABS(A(IQ + LP + IL, I)).GT.TEMP) TEMP = DABS(A(IQ + LP + IL, I))
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)
CONTINUE
DO 31 I = 1, IP
IF (DABS(A(IQ, I)).GT.TEMP) TEMP = DABS(A(IQ, I))
CONTINUE
V = .125
ALPF = V*DABS(EO(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),
1 SUMF(5), DUM4(88), 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),
1 IL, 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(LP+L, K)=ABASE(J+LP, K)
IF (LIN(J+LP) .EQ. 1) ABASE(J+LP, K)=A(LP+L, K)
4 SUMG(J)=SUMG(J)+A(LP+L, K)**2.
3 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
5 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
110
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