AC-CR-75-002

PARAMETRIC OPTIMAL DESIGN

Part 2: Second Order Algorithm and Hybrid Method

by B.M. KWAK, K. RIM and J.S. ARORA

MARCH 1975

Contract no
DAAA09-74-M-2015

prepared by
University of Iowa
Division of Materials Engineering
College of Engineering
Iowa City, Iowa 52242

prepared for
U.S. Army Armament Command
Rock Island, Illinois 61201

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AC-CR-75-002

2. GOVT ACCESSION NO.:

3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle):
PARAMETRIC OPTIMAL DESIGN
PART II: SECOND ORDER ALGORITHM AND HYBRID METHOD

5. TYPE OF REPORT & PERIOD COVERED:
Final Technical Report
May, 1974 to Dec., 1974

6. PERFORMING ORG. REPORT NUMBER:
14

7. AUTHOR(s):
B. M. Kwak, K. Rim, and J. S. Arora

8. CONTRACT OR GRANT NUMBER(s):
DAAA09-74-M-2015

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Department of Mechanics and Hydraulics
College of Engineering, University of Iowa
Iowa City, Iowa 52242

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:

11. CONTROLLING OFFICE NAME AND ADDRESS:
U.S. Army Armament Command
AMSAR-RDT
Rock Island, Illinois 61201

12. REPORT DATE:
March, 1975

13. NUMBER OF PAGES:
88

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):

15. SECURITY CLASS. (of this report):
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report):
Distribution of this report is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report):

18. SUPPLEMENTARY NOTES:

19. KEY WORDS (Continue on reverse side if necessary and identify by block number):
Parametric optimal design; Chebyshev approximation problem; Sensitivity analysis with error compensation; Second order algorithm; Hybrid second and first order method

20. ABSTRACT (Continue on reverse side if necessary and identify by block number):
In the Report Number 13, a first order algorithm for parametric optimal design has been presented. In this report, a second order algorithm and a hybrid second and first order method are developed using sensitivity analysis with error compensation. Test problems are solved and compared with the results obtained by first order method. Also presented is the implementation of the developed algorithms for computer calculations.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TECHNICAL REPORT #14

PARAMETRIC OPTIMAL DESIGN

PART II: SECOND ORDER ALGORITHM AND HYBRID METHOD

by

B. M. Kwak, K. Rim, and J. S. Arora

March, 1975

Project Title: OPTIMAL STRUCTURAL DESIGN WITH VARIABLE GEOMETRIC AND LOAD CONDITIONS

Contract No.: DAAA09-74-2015

This report is a part of the Ph.D. thesis of Byung Man Kwak done under the supervision of Professor Kwan Rim and Professor Edward J. Haug, Jr.

TABLE OF CONTENTS

PART I: FIRST ORDER ALGORITHM

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

A	free parameter (α) constraint set; partial derivatives of constraint function with respect to free parameter vector (α)
A(x)	answering set at x of a function $\phi(x,\alpha)$
$\overline{AA}$	second partial derivative of constraint function with respect to free parameter vector (α)
B	a Banach space; partial derivatives of constraint function with respect to design variable vector (b)
B*	the normed dual of B
$\overline{BA}$	cross partial derivative of constraint function with respect to design variable and free parameter vector
$\overline{BB}$	second partial derivative of constraint function with respect to design variable vector (b)
b	design variable vector
-C	a vector in the first order feedback law compensating the error in current maximum point
C(A)	a normed function space with domain A
D	constraint region
-D	sensitivity coefficient of the maximum point, as given in the first order feedback law
E	Young's modulus
e	direction vector
$F(\cdot), f(\cdot)$	functions
$G(\cdot), g_1(\cdot)$	functions

$h_1(\cdot)$	function associated with state equations
I	identity matrix; moment of inertia of a beam when used after E; index set
I_ε	ε-active index set
J	objective function
K^ϕ	constant matrix for constraint functions due to the vector C
L^J	second derivative of objective function with respect to design variable vector
ℓ^J	first derivative of objective function with respect to design variable vector
L^ϕ	matrix of second derivatives of constraint functions with respect to design variable vector
ℓ^ϕ	matrix of first derivatives of constraint function with respect to design variable vector
M	tangent subspace
M(x)	set of support functionals to a function at x; moment along the beam axis
M, M_q, M_{qq}	various matrices associated with free parameter constraint set
$M_{\phi\phi}$	matrix associated with design variable constraint set
m	dimension of state variable vector (z)
n	dimension of design variable vector (b)
O(x)	$O(x)/\|\|x\|\| \leq K$ for small $\|\|x\|\|$ and some constant K
p	dimension of free parameter vector (α)
Q	matrix of gradients for free parameter constraint equations
$q_1(\alpha)$	function associated with free parameter constraint set

R^n	n dimensional Euclidean space
r	number of free parameter constraint equations
$S(\cdot)$	algorithm
U_i	neighborhood
$U_i(x)$	answering set at x on U_i for $\phi(x,\alpha)$
W	positive definite weighting matrix
W_b	weighting matrix associated with design variables
W_α	weighting matrix associated with free parameters
x,y	vectors in a Euclidean space
z	state variable vector
α	free parameter vector
β	small parameter for implementation of algorithm
γ	Lagrange multiplier for design variable step size constraint
Γ_x	set of feasible directions at x
$\delta(\cdot)$	variation of $(\cdot)$
$\Delta(\cdot)$	desired reduction in $(\cdot)$
$\varepsilon, \varepsilon_O, \varepsilon_A, \varepsilon_B, \varepsilon_S$	small parameters for implementation of algorithm
η	constant for design variable stepsize restriction
κ	adjoint variable vector to the linearized state equations
λ,μ	Lagrange multiplier vectors
q_λ	Lagrange multiplier for inequalities in second order approximate problem
$\mu(\cdot)$	function

ν	Lagrange multiplier for free parameter step size constraint
ξ	constant for free parameter stepsize restriction
ρ	density
σ	stress
$\sigma_Y, \sigma_T, \sigma_C$	yield stresses
$\phi(\cdot)$	function
Φ	modified matrix of second partial derivatives of parametric constraint functions with respect to free parameter vector
ω	excitation frequency

Superscripts

$(i), i$	i-th iterate
1,2	decomposition of a vector by two components
T	transpose of a matrix

Subscripts

i	i-th component of a vector

Miscellaneous Symbols

max	maximize
min	minimize
NLP	nonlinear programming
POD	parametric optimal design
sup	supremum
ϵ	belong to
$\nabla(\cdot)$	gradient

$(\bar{\cdot})$	solution vector; a quantity calculated at the solution
$\|\|\cdot\|\|$	norm (usually Euclidean norm)
$\cup$	union
[ ]	denotes reference numbers listed in REFERENCES

INTRODUCTION

This report is a continuation of Report Number 13, "Parametric Optimal Design Part I: First Order Algorithm". In this report, two new methods of parametric optimal design (POD) are developed. The first method is called a second order algorithm, and the second one is called a hybrid algorithm. The hybrid algorithm is a combination of the first and the second order methods. Chapter 4 presents a design sensitivity analysis with error compensation which makes second order algorithms possible. Hybrid methods are discussed in Chapter 5. Several example problems are solved, using both the second order algorithm and the hybrid method, and a comparison of results is made. In Chapter 6, detailed procedures for implementation of various algorithms is presented. Chapter 7 provides a summary of the main contributions and comparison of various methods alongwith the difficulties involved in applications of the developed algorithms.

CHAPTER 4

SECOND ORDER ALGORITHM

4.1 Introduction

The first order algorithm described in the previous chapter is equivalent to alternate maximization and minimization. This is a feasible method and has been used in solving the classical minimax problem. It is apparent, however, that during the maximization procedure in the subproblem, if one can account for higher order changes in the design variables, one will have a better estimate of $\bar{\alpha}$ for the next iteration, without a separate maximization procedure. In other words, one wishes to use a higher order approximation to predict the effect on the maximum point of the subproblem, considering the change in the design variable δb. Furthermore, if the current maximum point $\bar{\alpha}$ is not exact, then this error should be corrected in the next iteration. The algorithm developed in this section is based on this idea.

The procedure to be employed is divided into two major steps, (1) Sensitivity analysis for the subproblem and (2) a minimization procedure for the outer problem, using sensitivity information.

4.2 Sensitivity Analysis with Error Compensation

Conventionally, sensitivity analysis consists of determination of sensitivity of the objective function with respect to a change in constraint level, and determination of the sensitivity of the solution point to a change in the constraint.

A more general sensitivity analysis includes the sensitivity of the solution point with respect to a set of parameters that appear in the problem. This last type is of concern for POD, and will be referred to as sensitivity analysis unless otherwise specified. As explained in the introduction to this chapter, if one obtains a functional relation for the solution point of subproblem, $\bar{\alpha} = \bar{\alpha}(b)$, the POD problem can be reduced to a classical nonlinear programming problem by substitution of this relation for each subproblem. This relation may be considered a closed loop solution of the subproblem, or a feedback law for the maximum point. Theoretically, then, the POD problem can be solved by obtaining the closed loop solution. However, the closed loop solution can not be found in most practical problems.

The approach that is persued here involves solution for a linear approximation to the feedback law, in the neighborhood of the current solution point. The sensitivity of the maximum point $\bar{\alpha}$, with respect to a change in b,

is then given by the first order feedback law and is expressed by $\frac{\partial \alpha}{\partial b}$.

Sensitivity analysis for a general nonlinear programming problem is considered by Fiacco [42]. The problem treated is to find the partial derivatives of the coordinates of the solution point and the objective function with respect to the free parameters. These derivatives are called the first order sensitivity of a second order local solution. It is shown in [42] that under the assumptions of a certian degree of differentiability and some regularity conditions, the derivatives are well defined.

The results of [42] might be used in obtaining explicit sensitivity information for the problem at hand, provided the solution point and the Lagrange multipliers are known. However, when there are many state equations, this sensitivity analysis requires excessive computer storage and computer time. Therefore, in subsequent papers [43,44] the same authors use an algorithm of direct calculation of the components of the sensitivity derivatives, using finite differences.

In this section, an explicit expression for the first order feedback law is obtained by using a gradient projection method with constraint error compensation. This method is derived by use of a second order approximation of the

objective function of the subproblem Eq. (2.3) and of the state equations (2.2). A first order approximation of the α-constraint equations (2.5) is employed. The second order approximation is essential, since with a linear approximation, the maximum point $\bar{\alpha}$ is independent of b and the sensitivity is zero.

The first order feedback law is now obtained by developing an explicit feedback law for the second order approximate problem.

The inner problem is of the following type: Choose α to maximize

$$g(z,b,\alpha) \tag{4.1}$$

subject to

$$h(z,b,\alpha) = 0, \tag{4.2}$$

$$q(\alpha) \leq 0. \tag{4.3}$$

Since z is considered as a function of b and α, one can express the variation of g in terms of variations of b and α, as follows;

$$\delta g = B^T \delta b + A^T \delta\alpha + \frac{1}{2}\delta b^T \overline{BB}\delta b + \delta b^T \overline{BA}\delta\alpha + \frac{1}{2}\delta\alpha^T \overline{AA}\delta\alpha. \tag{4.4}$$

Also,

$$\delta\tilde{q} = \frac{\partial\tilde{q}}{\partial\alpha}\delta\alpha \equiv \tilde{Q}^T\delta\alpha \leq \Delta\tilde{q}, \tag{4.5}$$

where $\tilde{()}$ denotes the constraints that are tight or violated and $\Delta\tilde{q}$ is the desired reduction in constraint violation, usually taken $\Delta\tilde{q} = -\tilde{q}(\bar{\alpha})$. The various matrices in the

expression for δg are given in Appendix D. In Eq. (4.5), only a linear approximation of the α constraints has been used. In many problems, $q(\alpha)$ is linear in α.

The approximate problem is now to maximize δg of Eq. (4.4) with respect to $\delta\alpha$, subject to the constraint of Eq. (4.5). To insure the validity of the approximations, a restriction is placed on $\delta\alpha$ such that

$$\delta\alpha^T W_\alpha \delta\alpha \leq \xi^2 \tag{4.6}$$

where W_α is a positive definite weighting matrix, and is predetermined in the same way as explained for W_b in the previous chapter.

Following Section 2.4.4, the Kuhn-Tucker necessary conditions for the problem (4.4), (4.5), and (4.6), where $\delta\alpha$ is now the design variable as an NLP, are solved to obtain,

$$\delta\alpha = -D\delta b - C, \tag{4.7}$$

where

$$D = \Phi^{-1}\{I - M_{qq}\}\cdot\overline{BA}^T, \tag{4.8}$$

$$C = \Phi^{-1}[\{I - M_{qq}\}\cdot A - M_q\Delta\tilde{q}], \tag{4.9}$$

and

$$\mu = M^{-1}\tilde{Q}^T\Phi^{-1}(A + \overline{BA}^T\delta b) + M^{-1}\Delta\tilde{q}, \tag{4.10}$$

where

$$\Phi = \overline{AA} - 2\nu W_\alpha, \tag{4.11}$$

$$M_q = \tilde{Q}M^{-1},$$

$$M = \tilde{Q}^T\Phi^{-1}\tilde{Q}, \tag{4.12}$$

$$M_{qq} = M_q \tilde{Q}^T \Phi^{-1}.$$

The constants ν and μ are multipliers to Eqs. (4.6) and (4.5), respectively.

Eq. (4.7) is called the first order feedback law with error compensation, where $-C$ is the compensating term for the error in the nominal solution to the inner problem. A necessary condition for the nominal solution to be the solution of the subproblem is that $C=0$ and Φ is negative semi-definite, since Φ is the Hessian of the Lagrangian functional to problem (4.4) and (4.5) (Section 2.4.2). Hence, in a neighborhood of the maximum point, the feedback law is given by $\delta a = -D\delta b$ and the sensitivity $\frac{\partial a}{\partial b}$ is given by $-D$.

The multiplier ν is determined as a stepsize, instead of calculating it as the Lagrange multiplier for the stepsize restriction of Eq. (4.6). This approach is employed, since the stepsize restriction contains another constant ξ that has little more physical significance than ν itself. The choice of ν will be discussed in a later section. By choosing $\nu>0$, Φ can be made negative definite and inversion of Φ is always feasible. This will be recognized as a great advantage to numerical computation, even for the case in which $\overline{AA}$ is nearly singular. For a system without constraint violation, $C=\Phi^{-1}A$ is simply the correction obtained by Newton's method, with a stepsize restriction. The

weighting matrix W_α changes the local scales in design space. A wise choice of this matrix can improve the convergence of the overall iterative method. These points are discussed in Chapter 6 when the Newton's method is derived.

It can be shown by direct multiplication [34] that $(I-M_{qq})$ is a projection matrix and that it projects the α-space onto the tangent subspace determined by the $\tilde{q}$ constraint set. Thus, C is a vector from A, projected onto the tangent subspace and then adjusted by a local metric W_α. Similarly, the sensitivity D is the matrix transformed from the cross derivative $\overline{BA}^T$ by the projection matrix and the local metric W_α.

In the follwoing section, Eq. (4.7) is utilized to obtain a second order approximation of the POD problem, in terms of δb, and to solve the outer problem.

4.3 Solution of the Second Order Approximate Problem

The minimization of δf, subject to the reduced constraints, without the environmental parameter, will be termed the outer procedure for the POD problem. By direct substitution from the preceding section, the POD problem can be approximated explicitly in terms of δb as: Minimize

$$\delta f = \ell^{J^T} \delta b + \tfrac{1}{2}\delta b^T L^J \delta b, \qquad (4.13)$$

subject to

$$\delta \tilde{g} = \ell^{\phi^T} \delta b + \tfrac{1}{2}\delta b^T L^{\phi} \delta b - K^{\phi} \leq \Delta \tilde{g}, \qquad (4.14)$$

where

$$\ell_j^{\phi} = (B-D^T A-\overline{BA}\cdot C+D^T\cdot\overline{AA}\cdot C)_j,\quad \ell^{J^T} = \frac{\partial f}{\partial b}, \tag{4.15}$$

$$L_j^{\phi} = [\overline{BB}+D^T\cdot\overline{AA}\cdot D-(\overline{BA}\cdot D+D^T\cdot\overline{BA}^T)]_j,\quad L^J = \frac{\partial^2 f}{\partial b^2}, \tag{4.16}$$

$$K_j^{\phi} = (A^T C-\tfrac{1}{2}C^T\cdot\overline{AA}\cdot C)_j,\quad j\in I_{\varepsilon}, \tag{4.17}$$

and $\Delta\tilde{g}= -\tilde{g}$ is the intended reduction in the current constraint violation. The superscript ϕ denotes the inclusion of all the ε-active constraints whose indices are in I_{ε}. It is noted that the vector ℓ^{ϕ} in Eq. (4.14) is a formal expression to the total derivative of g with respect to b, when C is assumed zero. If the optimality condition for the subproblem is utilized, the expression for ℓ^{ϕ} is simplified and K^{ϕ} is zero, since C=0 in this case. But for numerical calculation, this optimality condition is satisfied only approximately. Hence, it is necessary to retain C.

To solve the second order approximate problem, the Newton-Raphson technique is employed to solve the Kuhn-Tucker necessary conditions. To assure that the approximate problem is close to the original problem, an additional stepsize restriction is imposed on δb. From Eq. (4.7) $||\delta a||' \leq ||\phi^{-1}||\cdot||I-M_{qq}||\cdot||\overline{BA}^T||\cdot||\delta b|| + ||C||'$, where $||\phi||$ denotes a matrix norm, defined by $||\phi||= \sup_{||x||=1} ||\phi x||'$, and $||\cdot||$ and $||\cdot||'$ are norms on R^n and R^p. For a normal problem, ϕ is negative definite and the spectral

norm of the projection matrix $(I-M_{qq})$ is less than or equal to 1. Therefore,

$$||\delta\alpha||' \leq K||\delta b|| + ||C||', \tag{4.18}$$

where $K=||\phi^{-1}||\cdot||I-M_{qq}||\cdot||\overline{BA^T}||$ is finite. Since $||C||'\to 0$ as the nominal solution approaches the solution to the subproblem, if $||\delta b||$ is small, then $||\delta\alpha||'$ is small; but not necessarily vice versa. Hence, the stepsize restriction on $||\delta\alpha||'$, given by Eq. (4.6), is not sufficient and the following stepsize restriction is imposed on δb:

$$\delta b^T W_b \delta b \leq \eta^2 \tag{4.19}$$

where W_b is a positive definite design variable weighting matrix, predetermined as described in Section 6.3.2.

The Kuhn-Tucker conditions for the approximate problem are, using summation notation,

$$\ell_i^J + L_{ij}^J \delta b_j + \ell_{ij}^\phi \lambda_j + L_{ijk}^\phi \delta b_j \lambda_k + 2\gamma W_{b_{ij}} \delta b_j = 0, \tag{4.20}$$

$$\lambda_i(\ell_{ji}^\phi \delta b_j + \tfrac{1}{2}\delta b_k L_{kji}^\phi \delta b_j - K_i^\phi - \Delta g_i) = 0, \tag{4.21}$$

$$\gamma(\delta b_i W_{b_{ij}} \delta b_j - \eta^2) = 0, \tag{4.22}$$

where $\lambda_i \geq 0$, and $\gamma \geq 0$ are multipliers.

The system of Eqs. (4.20), (4.21), and (4.22) is highly nonlinear in the unknowns δb, λ, and γ. The existence of a solution is difficult to prove. Even when a solution exists for a physically well formulated problem, it may not be unique. Further, because of the sign restriction on multipliers, the proper solution is difficult to obtain,

in practice. If a solution of the system of equations yields a negative multiplier, the corresponding constraint is discarded and the computation is repeated. To solve the resulting system, the Newton-Raphson method is used, with the following derivative matrix:

$$\begin{bmatrix} L^J_{ij}+\lambda_k L^\phi_{ijk}+2\gamma W_{b_{ij}}, & \ell^\phi_{ij}+L^\phi_{ikj}\delta b_k, & 2W_{b_{ij}}\delta b_j \\ \lambda_i(\ell^\phi_{ji}+L^\phi_{kji}\delta b_k) & \delta_{ij}F_{n+i} & 0 \\ \quad (i \text{ not summed}) & & \\ 2\gamma\delta b_i W_{b_{ij}} & 0 & \delta b^T W_b \delta b - \eta^2 \end{bmatrix}, \tag{4.23}$$

where

$$F_{n+i} = \ell^\phi_{ji}\delta b_j + \frac{1}{2}\,\delta b_k L^\phi_{kji}\delta b_j - K^\phi_i - \Delta g_i.$$

This derivative matrix is nonsymmetric† and nonsingular, as long as the system is normal.

The success of the Newton-Raphson method depends largely on the initial estimate. Since the solution for δb is expected to be near the origin of design space, $\delta b = 0$ is found to be effective as the initial estimate.

The initial choice of λ and γ is obtained by the algorithm given in Section 2.4.4, with ℓ^J and ℓ^ϕ interpreted as given in the present section and Δg replaced by $\Delta g + K^\phi$.

---

†If equality is presumed to hold in Eqs. (4.14), and (4.19) instead of Eqs. (4.21), and (4.22), the derivative matrix would be symmetric. But the solution set from this system of equations may be too restrictive. It is only a subset of the solution set of the original system, (4.20), (4.21), and (4.22).

The vectors δb and λ are given as:

$$\delta b = -\frac{\delta b^1}{2\gamma} + \delta b^2$$

$$= -\frac{W_b^{-1}}{2\gamma}(\ell + \ell^{\phi}\lambda^1) + W_b^{-1}\ell^{\phi}\lambda^2, \qquad (4.24)$$

$$\lambda = \lambda^1 - 2\gamma\lambda^2$$

where

$$(\ell^{\phi^T} W_b^{-1} \ell^{\phi})\lambda^1 = -\ell^{\phi^T} W_b^{-1} \ell^J, \qquad (4.25)$$

$$(\ell^{\phi^T} W_b^{-1} \ell^{\phi})\lambda^2 = K^{\phi} + \Delta\tilde{g}. \qquad (4.26)$$

It is noted that this is different from the result of the pure first order algorithm, as given in Section 3.2, since the expression of ℓ^{ϕ} in this section is different from $\ell^{\phi}=\{B_j^T,\ j\varepsilon I_{\varepsilon}\}$ in the first order algorithm. Also, in Section 3.2, $K^{\phi}=0$.

It is shown in Appendix E that the Lagrange multiplier for the first order approximate problem is the same as for the original problem. The Lagrange multiplier for the second order approximate problem is obtained as

$$q_{\lambda}{}^1 = -M_{\phi\phi}^{-1}(\ell^{\phi^T} + \overline{\delta b}^T L^{\phi})\bar{W}_b^{-1}(\ell^J + L^{J^T}\overline{\delta b}), \qquad (4.27)$$

$$q_{\lambda}{}^2 = -M_{\phi\phi}^{-1}(\bar{K}^{\phi} + \Delta\tilde{g}), \qquad (4.28)$$

where

$$M_{\phi\phi} = (\ell^{\phi^T} + \overline{\delta b}^T L^{\phi})W_b^{-1}(\ell^{\phi} + L^{\phi^T}\overline{\delta b}), \qquad (4.29)$$

$$\bar{K}^{\phi} = \delta\tilde{g}\Big|_{\overline{\delta b}}, \qquad (4.30)$$

and ℓ^{ϕ}, L^{ϕ}, ℓ^J, and L^J are claculated at the solution of the approximate problem, $\overline{\delta b}$. Comparing with the above first order result, Eq. (4.25), $q_{\lambda}{}^1$ is equal to λ^1 when $\overline{\delta b} = 0$. In the limit, the solution of the second order

approximate problem is zero. During iteration, however, $\overline{\delta b}$ is nonzero and ${}^{q}\lambda^{1} \neq \lambda^{1}$, ${}^{q}\lambda^{2} \neq \lambda^{2}$. Therefore, if the initial estimate for the Newton-Raphson method, $\delta b^{(o)}$, is zero, λ^{1} and λ^{2} from Eqs. (4.25) and (4.26) are the proper estimates for ${}^{q}\lambda$. As each outer porcedure converges, the solution of the second order approximate problem must converge to zero. As the iterate of POD converges, ${}^{q}\lambda^{1} = \lambda^{1}$ and ${}^{q}\lambda^{2} = \lambda^{2}$, since $\overline{\delta b} \to 0$. Further, as shown in Appendix E, λ^{1} and λ^{2}, from Eqs (4.25) and (4.26), approach the Lagrange multipliers of the POD problem.

4.4 Numerical Examples

The weapons allocation problem described in Section 2.3.1 is solved by the second order method of this chapter. With the same initial estimates as given in Section 3.3, the solution is obtained only after 8 iterations (Table 7); $b = [0.0, 0.18186, 0.27273]^{T}$, with a cost of -2.2340 and the worst case $\alpha = [0.0, 0.1751, 0.4654]^{T}$. As compared with the behaviour of iterates in the first order method (Table 1), Table 7 shows that convergence is rapid as the iterates approach the solution. Since the problem is convex, a quadratic rate of convergence can be expected. The other feature is that the maximization procedure for the inner problem, which is an essential part of the first order algorithm, is not necessary; since the second order method accounts for the change in maximum points, as well as

compensating any current error in the nominal solution of inner problems.

The sensitivity coefficient, -D in Eq. (4.7), at the solution, is found to be

$$\begin{bmatrix} 3.73(-8) & 2.61(-8) & 1.86(-8) \\ 9.06(-2) & 4.55(-1) & 7.70(-2) \\ 9.74(-2) & 7.70(-2) & 3.58(-1) \end{bmatrix}$$

Tables 8 and 9 show the solutions of the vibration isolator design problem (Section 2.3.3), with frequency ranges of [0.5, 1.5] and [0.9, 1.25], respectively. The initial estimate for case (i) is rather good and the method converges rapidly. On the other hand, the initial estimate for case (ii) is rather bad. Since the artificial design variable has been used, the first 6 iterations only adjust the artificial design variable. After iteration number 6, rather good convergence is obtained. It is noted that case (ii) behaves badly, with regard to the system of Kuhn-Tucker equations, (4.20), (4.21), and (4.22). After fixing $\gamma>0$, determined by the method described in Section 6.3.2, the Newton's method converges very rapidly. This may limit the convergence rate of outer procedure, but the precise affect is not known. The computer time per iteration on an IBM 360/65 was about 0.24 seconds, in FORTRAN(H).

Sensitivity coefficients are obtained as: For case (i), -D = [0.835, 0.197] at α=0.848, and -D =

[-1.103, 0.300] at α=1.059. For case (ii) -D = [1.36(-9), -8.05(-8)] at α=0.9, and -D = [-0.574, 0.657] at α=1.119.

McMunn's problem [32], described in Section 2.3.5, is solved, with results in Table 10. This problem has a saddle point solution. As pointed out earlier, however, since one is not sure, at the beginning, how many maxima there are for a given range of free parameters, a certain amount of intuition is necessary. Four initial estimates of maximum points are used here. Table 10 shows that they merge into one point, meaning one maximum point at the optimum design. The same initial design estimates, as given in [32], are used. As in the vibration isolator design case (ii) (Table 9), the first 7 iterations are spent compensating the constraint violations. After the 7-th iteration, it is seen that the rate of convergence is quadratic. The final solution is $b = [2.099, 0.9133, 1.145]^T$, with J = 1.490 attained at α=1.505. The computer time was about 1 second per iteration. McMunn's solution was $b = [2.1160, 0.9149, 1.0695]^T$, and J = 1.494 at α=1.503. Note that part of the difference may come from the round-off error in the numerical values of the matrices M, D, and K. A sensitivity coefficient of -D = [0.976, 5.414, 0.447] is obtained during the solution procedure.

In the notation of Section 2.2.2, the following Chebyshev approximation problem is taken from [19];

$$f(\alpha) = \frac{1}{1 + \alpha} ,$$

and

$$F(b,\alpha) = b_1 e^{b_3 \alpha} + b_2 e^{b_4 \alpha}, \quad \alpha \in [0., 1.0].$$

Table 11 shows the results. In [19], the solution is obtained as b={0.713835, 0.285966, -0.407327, -2.442953}, with the square of maximum error in between 0.23(-7) and 0.76(-7). There is some difference in the solution, although $||\delta b^1||$ is quite small, satisfying any practical stopping criterion. It was found that even though the iterations were continued no significant improvement was obtained. This may be caused by round-off errors. The computer time was about 0.33 seconds per iteration.

Table 7 Weapons Allocation Problem

(Second Order Algorithm)

iter	b_i			obj.	$\|\|\delta b^1\|\|$	a_i			$\|\|\delta a^1\|\|$	tight g_i, q_i
					0.1	0.1	0.2	0.3		
0	0.1	0.2	0.3	-2.10	0.795	0.0	0.246	0.563	(4)0.002	q_1
1	-0.579	0.323	0.481	-2.48	0.23	0.0	0.219	0.533		g_2,q_1
2	0.0	0.309	0.468	-2.014	0.91	0.0	0.219	0.533	(0)0.15	"
3	0.0	0.193	0.277	-2.233	1.0	0.0	0.193	0.466	(2)0.004	"
4	0.0	0.193	0.277	-2.233	0.14	0.0	0.194	0.482	(0)0.003	"
5	0.0	0.178	0.270	-2.235	0.02	0.0	0.183	0.473	(0)0.008	"
6	0.0	0.181	0.271	-2.234	0.01	0.0	0.176	0.467	(0)0.001	"
7	0.0	0.182	0.272	-2.234	0.003	0.0	0.175	0.465	(1)0.0002	"
8	0.0	0.18186	0.27273	-2.2340	0.0002	0.0	0.1751	0.4654	(0)0.2(-4)	"

Table 8 Vibration Isolator Design, Case (i)

(Second Order Algorithm)

iter	b_i		obj.	$\|\|\delta b^1\|\|$	$\|\|\delta a^1\|\|_{max}$	tight constraint, $g_i(a)$
0	0.15	1.0	58.4	22.35	(5)0.0017	$g_1(.865)$
1	0.1416	0.9713	43.42	22.34	(0)0.0002	$g_1(.857)$
2	0.1207	0.8998	29.69	22.30	(0)0.0075	$g_1(1.087)$
3	0.1182	0.9389	33.94	19.24	(1)0.004	$g_1(.837)$
4	0.1385	0.9187	24.06	15.66	(0)0.03	$g_1(.837), g_1(1.113)$
5	0.1621	0.9084	21.40	3.61	(1)0.0001	$g_1(.843), g_1(1.065)$
6	0.1686	0.9090	21.06	0.015	(0)0.0004	$g_1(.848), g_1(1.059)$
7	0.16860	0.90906	21.060	0.0008	(0)0.5(-6)	$g_1(.848), g_1(1.059)$

Table 9 Vibration Isolator Design, Case (ii)

(Second Order Algorithm, γ fixed)

iter	b_i		obj.	$\|\|\delta b^1\|\|$	$\|\|\delta a^1\|\|_{max}$	tight constraint, $g_i(a)$
0	0.15	1.0	37.8	19.99	(4)0.0015	$g_1(.9)$
1	0.15	1.0	37.8	19.99	(0)0.2(-4)	$g_1(.9)$
2	0.15	1.0	38.24	19.99	(0)0.9(-6)	$g_1(.9)$
3	0.15	1.0	38.24	19.99	(0)0.7(-6)	$g_1(.9)$
4	0.15	1.0	38.25	19.99	(0)0.5(-6)	$g_1(.9)$
5	0.15	1.0	38.25	19.99	(0)0.5(-6)	$g_1(.9)$
6	0.15	1.0	38.25	0.90	(0)0.5(-6)	$g_1(.9), g_1(1.13)$
7	0.1575	0.9537	22.84	9.75	(1)0.001	$g_1(.9), g_1(1.092)$
8	0.1380	0.9479	17.98	6.12	(1)0.002	$g_1(.9), g_1(1.108)$
9	0.1299	0.9485	16.65	3.53	(0)0.6(-4)	$g_1(.9), g_1(1.112)$
10	0.1263	0.9516	16.61	1.30	(0)0.5(-4)	$g_1(.9), g_1(1.116)$
11	0.1250	0.9527	16.61	0.49	(0)0.6(-5)	$g_1(.9), g_1(1.118)$
12	0.1246	0.9531	16.61	0.18	(0)0.3(-6)	$g_1(.9), g_1(1.118)$
13	0.1244	0.9532	16.61	0.064	(0)0.2(-6)	$g_1(.9), g_1(1.119)$
14	0.12432	0.95329	16.605	0.008	(0)0.5(-6)g	$g_1(.9), g_1(1.119)$

Table 10 Five Degree-of-Freedom Vehicle Problem

(Second Order Algorithm)

iter	b_1			obj.	$\|\|\delta b^1\|\|$	maximizing a's				$\|\|\delta a^1\|\|_{max}$	violated inner g_i
0	0.2	0.2	0.2	16.741	1.0	0.5	1.2	3.5	7.0		g_1
1	0.288	0.309	0.216	8.259	1.0	1.01	2.0	5.5		(4)0.15	g_1
2	0.394	0.401	0.236	5.191	0.99	0.99	1.3	4.8		(4)0.15	g_1, g_2
3	0.527	0.502	0.268	3.554	0.99	0.98	4.0			(4)0.15	g_1
4	0.688	0.619	0.315	2.637	0.95	0.96	3.3			(4)0.15	g_1
5	0.873	0.753	0.385	2.107	0.85	0.96	2.5			(4)0.15	g_1
6	1.067	0.893	0.475	1.804	0.66	0.97	1.8			(4)0.15	g_1, g_2
7	1.250	1.023	0.572	1.636	0.47	1.03	1.08			(4)0.05	g_1, g_2
8	1.512	1.185	0.749	1.516	0.11	1.429				(3)0.2(-3)	g_1
9	1.742	0.973	0.885	1.496	0.19	1.470				(0)0.12	g_1
10	1.968	0.938	1.027	1.491	0.06	1.435				(1)0.001	g_1
11	2.048	0.926	1.104	1.490	0.004	1.505				(0)0.004	g_1
12	2.097	0.913	1.141	1.490	0.004	1.505				(1)0.6(-5)	g_1
13	2.097	0.914	1.141	1.490	0.5(-3)	1.058				(0)0.003	g_1
14	2.0985	0.9133	1.1448	1.4898	0.2(-4)	1.5053				(0)0.1(-4)	g_1

Table 11 Chebyshev Approximation Problem

(Second Order Algorithm)

iter	b_i				obj.	$\|\|\delta b^1\|\|$	$\|\|\delta a^1\|\|_{max}$	violated $g_i(a)$
0	0.7	0.3	-0.4	-2.4	0.13(-4)	0.6(-2)	(5)0.02	$g_1(1.0)$
1	0.7	0.3	-0.3928	-2.393	0.74(-6)	0.3(-12)	(3)0.02	$g_1(0), g_1(0.42)$
2	0.700	0.300	-0.3927	-2.393	0.71(-6)	0.9(-23)	(5)0.02	$g_1(0), g_1(0.417)$ $g_1(1.0)$
3	0.700	0.300	-0.3928	-2.393	0.72(-6)	0.3(-36)	(2)0.7(-4)	$g_1(0), g_1(0.417)$ $g_1(1.0)$

4.5 Conclusions

In the present chapter, the sensitivity analysis that gives a first order feedback law and the method of solving the approximate problem, obtained from the PCD problem, have been described.

Several examples are solved and compared with the solutions obtained by the first order algorithm of Chapter 3. The second order method exhibits rapid convergence, once the iterates are near the solution. Since the second order method accounts for the change in the maximum point, due to small changes in the design and compensates any error in the current maximum point, the method does not require separate solution of the inner problems.

The disadvantage of the method lies in the determination of a good initial estimate and solution of the Kuhn-Tucker equations. The method is not reliable unless the problems under consideration have nice properties such as convexity, at least locally, or unless the initial estimates are close enough to the solution.

To make use of the advantages of the first and second order algorithms, some variations will be considered in the following chapter.

CHAPTER 5

HYBRID METHODS

5.1 Introduction

In the previous chapters, purely first and second order algorithms were discussed. It has been observed that one numerical approach is not always effective for all problems. There may be a variation on these approaches that is better suited to a given problem. It is the purpose of this chapter to discuss some possible variations of the algorithms described in the preceding chapters.

Once one defines the effectiveness of an algorithm, in terms of rate of convergence and the effort of constructing and running a program, a best combination may be possible for a given problem. In this sense, the selection of approach is an optimization process, which is critical in practical problems. In the following, possible alternatives are given for the POD algorithm and a comparison of computational effort is shown, based on the number of numerical operations required to implement the algorithm.

5.2 Review of Alternatives

As stated earlier, the first order algorithm was developed under the assumption that the subproblem is solved as an integral part of each iteration. Hence, one has freedom to choose any solution method in solving the subproblem. In the second order algorithm, the solution of subproblem is adjusted through use of sensitivity factors, simultaneously with the minimization of the objective function. Also, a first order feedback law was developed, with a correction term for any error in the solution of the subproblem. A full second order approximation in the design variable was necessary to maintain the order of the approximation. From a numerical point of view, however, a first order method such as steepest descent has been found to be effective for the solution of NLP. It is natural to attempt a first order outer procedure, with sensitivity information obtained from a second order approximation of the inner problem.

Variations in methods fall into two categories: One is essentially a first order algorithm with a higher order method of solution of the subproblem. The other is a variation from the second order algorithm and uses a first order approximation in the outer procedure. Since this method, then, uses the first order total derivatives with respect to b as noted in Section 4.3, this method is,

in reality, first order. Including the original first and second order algorithms, four alternative methods are possible. They are:

(1) first order maximization + first order outer procedure,

(2) second order maximization + first order outer procedure,

(3) sensitivity with second order maximization + first order outer procedure, and

(4) sensitivity with second order maximization + second order outer procedure.

The two alternative methods, (1) and (2), have their background in the analysis given in Chapter 3, where it is shown that the first order algorithm for the solution of the POD problem is essentially an alternation of maximization and minimization, as is frequently used in min-max problems. The alternative methods (3) and (4) do not require a separate procedure for soving inner problems, since the sensitivity analysis with error compensation, presented in Chapter 4, gives the necessary change in the solution point. Being first order in nature, method (3) is expected to have sufficient reliability, while maintaining the accuracy of maximum points. On the other hand, the full second order method (4) is too complicated to solve the resulting approximate problem, requiring a Newton's procedure.

5.3 Second Order Method of Maximization

If one puts $\delta b=0$ in the sensitivity analysis of Chapter 4, one has a second order method of maximization. In this section, this approach is examined more closely, since it can be implemented for NLP and can be used in method (2).

Since it has been assumed that the α-constraints are linear in α, the problem considered here is to maximize

$$A^T\delta\alpha + \frac{1}{2}\delta\alpha^T\,\overline{AA}\,\delta\alpha \qquad (5.1)$$

subject to

$$Q^T\delta\alpha \leq \Delta q, \qquad (5.2)$$

and

$$\delta\alpha^T W_\alpha \delta\alpha \leq \xi^2. \qquad (5.3)$$

From Section 2.4, the solution of the Kuhn-Tucker necessary conditions is obtained as

$$\delta\alpha = -\delta\alpha^1 + \delta\alpha^2, \qquad (5.4)$$

where

$$\delta\alpha^1 = \Phi^{-1}(I-M_{qq})A, \qquad (5.5)$$

$$\delta\alpha^2 = \Phi^{-1}M_q\Delta q, \qquad (5.6)$$

and

$$\Phi = \overline{AA} - 2\nu W_\alpha, \qquad (5.7)$$

$$M_q = Q(Q^T\Phi^{-1}Q)^{-1}, \qquad (5.8)$$

$$M_{qq} = M_q Q^T\Phi^{-1}. \qquad (5.9)$$

As direct multiplication shows, M_{qq} is a projection matrix; i.e., $M_{qq}\,M_{qq}=M_{qq}$. To interpret this as in the

gradient projection method, the following relations can be verified:

$$(1)\quad \delta\alpha^{1T}\Phi\delta\alpha^{2} = 0, \tag{5.10}$$

$$(2)\quad Q^{T}\delta\alpha^{2} = \Delta q, \tag{5.11}$$

$$(3)\quad Q^{T}\delta\alpha^{1} = 0, \tag{5.12}$$

$$(4)\quad A^{T}\delta\alpha^{1} + \frac{1}{2}\delta\alpha^{1T}\,\overline{AA}\,\delta\alpha^{1} \geq 0, \tag{5.13}$$

where Φ is negative semi-definite.

Property (4), combined with the yet undetermined constant ξ, is valuable in practical computation. By choosing $\nu>0$ large enough such that Φ is negative definite, $\delta\alpha^{1}$ is always in the direction of increasing the objective function (5.1). Hence, one can generate a sequence of points converging to a local maximum point. This may lead, however, to a false impression of convergence, if one chooses very large ν.

Much literature has been published on Newton's method and its variations [45,46]. Most of this literature treats unconstrained problems. When constraints are absent, the method presented here is another version of Newton's method. Same techniques have been studied in the literature [45,47], from a different point of view. Although the reliability of the method can be easily guaranteed, convergence properties are largely dependent on the optimal choice of ν [45]. An estimate of ν and numerical implementation are discussed in Section 6.3.

5.4 A Computational Modification for Alternative Method (3)

As has been discussed in Section 5.2, method (3) is rather efficient. In this section, a computationally efficient modification is described for the case in which no α-constraints are present, i.e., $M_q=0=M_{qq}$ in Eq. (4.7).

In the case of the first order outer procedure, the inversion of Φ for the expression (4.14) can be avoided in the following way. From Eq.(4.7), multiplying by Φ,

$$\Phi\delta\alpha = -D'\delta b - C' \tag{5.14}$$

where,

$$\Phi = \overline{AA} - 2\nu W_{\alpha},$$
$$D' = \overline{BA}^T,$$
$$C' = A.$$

Method (3) utilized a first order expansion in Eq. (4.4),

$$\delta g = B^T\delta b + A^T\delta\alpha. \tag{5.15}$$

To eliminate the second term, let κ be determined as the solution of

$$\Phi^T\kappa = A. \tag{5.16}$$

Post-multiplying by $\delta\alpha$, after taking the transpose, one obtains

$$\kappa^T\Phi\delta\alpha = A^T\delta\alpha. \tag{5.17}$$

Hence,

$$\delta g = (B^T - \kappa^T D')\delta b - \kappa^T C'. \tag{5.18}$$

Therefore, the first order approximate problem, using

sensitivity information, is obtained as: Minimize

$$\delta f = {}_{\ell}J^{T}\delta b \tag{5.19}$$

subject to

$$\delta \tilde{g} = {}_{\ell}\phi^{T}\delta b - K^{\phi} \leq \Delta\tilde{g} \tag{5.20}$$

where

$${}_{\ell}J = \frac{\partial f^{T}}{\partial b}, \tag{5.21}$$

$${}_{\ell}\phi_{j} = (B - D'^{T}\kappa)_{j}, \tag{5.22}$$

$$K^{\phi}_{j} = -(\kappa^{T}C')_{j},\ j \in I_{\varepsilon}. \tag{5.23}$$

Here, the subscript j denotes the quantity calculated for the j-th inner problem. After solving this approximate problem for δb, using the gradient projection method described in Section 2.4, $\delta\alpha$ is obtained from Eq. (5.14).

It is noted that the inversion of Φ has been replaced by two solutions of a system of linear equations (5.16) and (5.14), which is a great computational advantage when the dimension of α is large.

5.5 An Application to the Optimal Design Problem with State Equations Given Implicitly by a Minimum Principle

The type of problems under consideration has been introduced in Section 2.2.3, and can be treated by the method discussed in Section 5.4. In the present section, the case of a function H(b,z) in Eq. (2.15), which is quadratic with respect to z, is considered. Comparisons are possible, since this results in a common optimal design

problem. Most of the problems in mechanics belong to this category, where H(b,z) usually represents total potential energy of the system. The problem is stated as: Minimize

$$f(b,z) \tag{5.24}$$

subject to

$$g(b,z) \leq 0, \tag{5.25}$$

where z maximizes the quadratic function,

$$-H(b,z) = \frac{1}{2}(2z^T F - z^T K(b)z), \tag{5.26}$$

K(b) is a positive definite matrix, and F is a known vector.

The sensitivity analysis of Section 4.2, with the modification given in the previous section, is applied to the present problem. Noting that the state variable z corresponds to the free parameter α, one has

$$\Phi \delta z = -D'\delta b - C' \tag{5.27}$$

where,

$$\Phi = -K - 2\nu W, \tag{5.28}$$

$$D' = -\frac{\partial}{\partial b}[K(b)z], \tag{5.29}$$

$$C' = F - Kz. \tag{5.30}$$

With these interpretations, the optimal design problem becomes, from the linearized expressions of Eqs. (5.24) and (5.25), minimize

$$\delta f = \ell^{J^T} \delta b \tag{5.31}$$

subject to

$$\delta \tilde{g} = \ell^{\Phi^T} \delta b - K^{\Phi} \leq \Delta \tilde{g}, \tag{5.32}$$

where

$$\ell^J = \frac{\partial f^T}{\partial b} - D'^T_\kappa J, \tag{5.33}$$

$$\ell^\phi = \frac{\partial g^T}{\partial b} - D'^T_\kappa \phi, \tag{5.34}$$

$$K^\phi = -\kappa^\phi C', \tag{5.35}$$

and κ^J and κ^ϕ are solutions of the systems

$$\phi^T_\kappa J = \frac{\partial f^T}{\partial z}, \tag{5.36}$$

$$\phi^T_\kappa \phi = \frac{\partial g^T}{\partial z}. \tag{5.37}$$

This linearized approximate problem can be solved by the gradient projection method described in Section 2.4. Then, the state variable z is predicted as $z+\delta z$, where δz is solved from Eq. (5.27).

Note that the state equations implicitly given by Eq. (5.26) can be obtained explicitly as

$$K(b)z - F = 0. \tag{5.38}$$

The final linearized approximate problem of Eqs. (5.31) and (5.32) is essentially the same as the one obtained directly using Eq. (5.38) [33]. In the present approach, the problem contains new terms $-2\nu W$ and K^ϕ, which give an additional feature to the method. The factor ν, acting as damping, stabilizes the convergence of the method, as explained in Section 5.3, and K^ϕ compensates the error involved in the current solution z.

In the case of a quadratic function H(b,z), little computational efficiency is gained but in the general case of state relations given implicitly by a nonlinear programming problem, no other method seems known.

5.6 Numerical Examples

The weapons allocation problem, solved in previous chapters by the first and second order methods, is solved by the hybrid method (3). Table 12 shows that the hybrid method has the advantage of the second order algorithm, in automatic adjustment of maximum points, and it has the reliability of the first order algorithm. Because the maximum points are self-adjusted, better convergence is obtained than the pure first order algorithm. The computer time for this problem was about 0.1 seconds per iteration.

Application of the hybrid method (3) to the vibration isolator design also shows that the method can be superior to the other two. The computer time per iteration was 0.1 seconds.

The solution of the 5 degree-of-freedom vehicle problem is shown in Table 15. Although at the beginning the convergence was rapid, it was slow near the solution. The computer time in this case was 0.7 seconds.

Results for the Chebyshev problem in Table 16 show a similar trend of rapid convergence for the first few iterations. Although the mathematical solution is difficult to attain by the hybrid method, since this method is essentially first order, the solution obtained seems good for engineering purposes. The computer time per iteration was 0.23 seconds, and is about 70% of that for the second order method.

Table 12 Weapons Allocation Problem

(Hybrid Method)

iter	b_1			obj.	$\|\delta b^1\|$	a_1			$\|\delta a^1\|$	tight g_i, q_i
						0.1	0.2	0.3		
0	0.1	0.2	0.3	-2.10	0.795	0.0	0.246	0.563	(4)0.002	q_1
1	0.099	0.200	0.300	-2.10	0.792	0.0	0.244	0.567	(0)0.0006	q_1
2	-0.004	0.145	0.245	-2.22	0.465	0.0	0.127	0.444	(0)0.02	g_2, q_1
3	0.0	0.327	0.351	-2.03	0.870	0.0	0.305	0.536	(4)0.007	"
4	0.0	5.194	0.245	-2.230	0.180	0.0	0.194	0.445	(5)0.004	"
5	0.0	0.181	0.288	-2.235	0.077	0.0	0.183	0.454	(0)0.014	"
6	0.0	0.172	0.281	-2.234	0.062	0.0	0.167	0.471	(3)0.002	"
7	0.0	0.179	0.279	-2.234	0.034	0.0	0.169	0.471	(0)0.002	"
8	0.0	0.182	0.274	-2.234	0.016	0.0	0.172	0.470	(0)0.001	"
9	0.0	0.183	0.273	-2.234	0.01	0.0	0.173	0.469	(0)0.001	"
10	0.0	0.183	0.272	-2.234	0.006	0.0	0.174	0.468	(0)0.001	"
11	0.0	0.18277	0.27174	-2.2340	0.0025	0.0	0.1749	0.4663	(1)0.0003	"

Table 13 Vibration Isolator Design, Case (i)

(Hybrid Method)

iter	b_i		obj.	$\|\|\delta b^l\|\|$	$\|\|\delta a^l\|\|_{max}$	tight constraint, $g(\alpha)$
0	0.15	1.0	58.4	22.35	(5)0.0017	$g_1(.865)$
1	0.1507	0.9447	30.84	22.31	(0)0.006	$g_1(.849)$
2	0.1511	0.9057	22.78	9.04	(1)0.0018	$g_1(.835), g_1(1.074)$
3	0.1636	0.9084	21.18	3.44	(0)0.01	$g_1(.835), g_1(1.074)$
4	0.1761	0.9082	21.19	4.38	(1)0.0001	$g_1(.851), g_1(1.054)$
5	0.1636	0.9095	21.12	2.88	(0)0.0016	$g_1(.843), g_1(1.065)$
6	0.1685	0.9091	21.07	0.07	(0)0.0004	$g_1(.848), g_1(1.059)$
7	0.1686	0.9091	21.06	0.03	(0)0.2(-5)	$g_1(.848), g_1(1.059)$
8	0.16857	0.90906	21.060	0.02	(0)0.8(-6)	$g_1(.848), g_1(1.059)$

Table 14 Vibration Isolator Design, Case (ii)

(Hybrid Method)

iter	b_i		obj.	$\|\|\delta b^1\|\|$	$\|\|\delta a^1\|\|_{max}$	tight constraint, $g_i(a)$
0	0.15	1.0	37.8	19.99	(4)0.0015	$g_1(.9)$
1	0.1494	0.9655	24.27	19.95	(0)0.0015	$g_1(.9)$
2	0.1487	0.9443	19.25	19.89	(0)0.0003	$g_1(.9)$
3	0.1480	0.9309	17.89	11.53	(0)0.0002	$g_1(.9), g_1(1.088)$
4	0.1310	0.9472	16.75	4.28	(0)0.0015	$g_1(.9), g_1(1.112)$
5	0.1161	0.9602	16.73	5.63	(0)0.0007	$g_1(.9), g_1(1.13), g_2(.94)$
6	0.1310	0.9475	16.69	4.20	(0)0.001	$g_1(.9), g_1(1.112)$
7	0.1223	0.9550	16.62	1.34	(0)0.0002	$g_1(.9), g_1(1.12), g_2(.94)$
8	0.1271	0.9509	16.62	1.85	(0)0.0001	$g_1(.9), g_1(1.115)$
9	0.1251	0.9527	16.61	0.50	(0)0.1(-4)	$g_1(.9), g_1(1.118)$
10	0.1247	0.9530	16.61	0.27	(0)0.1(-6)	$g_1(.9), g_1(1.118)$
11	0.1244	0.9532	16.61	0.09	(0)0.4(-6)	$g_1(.9), g_1(1.118)$
12	0.1244	0.9533	16.61	0.03	(0)0.7(-6)	$g_1(.9), g_1(1.119)$
13	0.12432	0.95329	16.605	0.009	(0)0.6(-6)	$g_1(.9), g_1(1.119)$

Table 15 Five Degree-of-Freedom Vehicle Problem

(Hybrid Method)

iter	b_i			obj.	$\|\|\delta b^1\|\|$	$\|\|\delta a^1\|\|_{max}$	violated $g_i(a)$
0	0.2	0.2	0.2	16.741	1.0	(4)0.15	$g_1(0.5)$
1	0.288	0.309	0.216	8.259	1.0	(4)0.15	$g_1(1.01)$
2	0.394	0.401	0.236	5.191	0.99	(4)0.15	$g_1(0.99), g_1(1.3)$
3	0.527	0.502	0.268	3.554	0.99	(4)0.15	$g_1(0.98)$
4	0.688	0.619	0.315	2.637	0.95	(4)0.15	$g_1(0.96)$
5	0.873	0.753	0.385	2.107	0.85	(4)0.15	$g_1(0.96)$
6	1.068	0.894	0.475	1.803	0.66	(4)0.002	$g_1(0.97)$
7	1.251	1.024	0.572	1.635	0.47	(1)0.003	$g_1(1.027)$
8	1.396	1.124	0.657	1.551	0.31	(0)0.047	$g_1(1.098)$
9	1.493	1.185	0.720	1.520	0.15	(2)0.3(-3)	$g_1(1.238)$
10	1.536	1.195	0.755	1.514	0.096	(0)0.06	$g_1(1.496)$
15	1.067	1.110	0.838	1.504	0.062	(0)0.002	$g_1(1.484)$
20	1.740	1.052	0.942	1.497	0.14	(0)0.036	$g_1(1.593)$
25	1.775	0.857	0.974	1.466	0.08	(5)0.04	$g_1(1.633)$
30	1.838	0.989	1.006	1.492	0.031	(0)0.016	$g_1(1.493)$

Table 16 Chebyshev Approximation Problem

(Hybrid Method)

iter	b_i				obj.	$\|\|\delta b^1\|\|$	$\|\|\delta a^1\|\|_{max}$	violated $g_i(a)$
0	0.7	0.3	-0.4	-2.4	0.13(-4)	0.6(-2)	(5)0.02	$g_1(1.0)$
1	0.7	0.3	-0.4	-2.4	0.13(-4)	0.6(-2)	(5)0.5(-4)	$g_1(1.0)$
2	0.7326	0.3044	-0.377	-2.399	0.14(-2)	0.3(-2)	(0)0.2(-8)	$g_1(0.),g_1(1.0)$
3	0.7038	0.2993	-0.4046	-2.399	0.10(-4)	0.3(-2)	(0)0.1(-8)	$g_1(0.),g_1(1.0)$
4	0.7060	0.2930	-0.3978	-2.399	0.11(-5)	0.8(-3)	(0)0.2(-8)	$g_1(0.),g_1(1.0)$
5	0.7058	0.2937	-0.3988	-2.399	0.27(-6)	0.6(-3)	(0)0.0	$g_1(0.),g_1(1.0)$

5.7 Comparison of Computational Effort

In this section, the effect of the dimension on the overall number of arithmatic operations will be given, in an approximate way. The count given denotes the number of mulitplications. Nearly the same number of operations of addition or subtraction is necessary. Operations such as division and logic are not included here.

The computational effort of one iteration, for a full second order algorithm, is proportional to

s with a factor of $\{n^2m^2 + nm^2p + n^2p^2\cdots\}$,

plus {STATE, G, MATINV(p), GAUSS(p), Q, MATINV($\tilde{g}$)},

n^2 with a factor of $\{sm^2 + 4\tilde{g}\cdots\}$,

m^2 with a factor of $\{s(n^2 + np + p^2)\cdots\}$,

p^3 with a factor of $\{s(n + \tilde{q})\cdots\}$,

p^2 with a factor of $\{s(n^2 + m^2 + \tilde{q}^2)\cdots\}$,

and $\tilde{g}^2$ with a factor of $\{s + 2n\cdots\}$. Also, GAUSS($n+\tilde{g}$) is necessary as many times as the iterations of Newton's method.

In the above, n, m, and p denote the number of design variables, state variables, and free parameters, respectively. The numbers s, $\tilde{g}$, and $\tilde{q}$ are the number of inner problems, active design variable constraints, and active free parameter constraints, respectively; STATE solves the state equations and calculates necessary derivatives; G and Q denote the function evaluations of g's and q's;

MATINV(p) denotes a p-dimensional matrix inversion; and GAUSS(p) denotes a p-dimensional linear equation solver. The notation $\cdots$ implies lower order terms are neglected, where $n > m > p$ is assumed.

In the case of the first order algorithm, the computational effort of one iteration is proportional to s, with a factor of $\{nm + (m+2n)p + \tilde{p}^2\cdots\}$ plus {STATE, G, Q, GAUSS($\tilde{p}$)}, and $\tilde{p}^2$ with a factor of $\{s + 2n\cdots\}$.

It is noted that the increase of computational effort for the second order algorithm, compared with that of first order, is enormous and it depends on the convergence of the Newton's method. For the hybrid method (3), the only increase in the computational effort will appear in the solutions of inner problems.

5.8 Conclusions

In this chapter, modifications of the algorithms described in previous chapters have been presented. The hybrid method (3), which is essentially of first order, has been found quite effective. This is because it makes use of both the advantage of sensitivity information and the reliability of first order outer procedure.

A second order method of solution for a nonlinear programming problem has been discussed in detail, which has appeared in the sensitivity analysis in Section 4.2. Also an application of sensitivity analysis to an optimal design

problem, where state equations are given implicitly by a minimum principle, is discussed.

CHAPTER 6

IMPLEMENTATION AND PROGRAMMING

6.1 Introduction

In this chapter, the description of a computer program, based on the second order algorithm developed in Chapter 4, is given. The program for the first order algorithm can be obtained by removing all the statements involving second order derivatives. A schematic flow chart of the second order algorithm follows a general description of the program. In the following, Newton's method may be replaced by the gradient projection method for the first order outer procedure, i.e., methods (1), (2), and (3). The other modifications are self-evident from Chapter 5.

The algorithms discussed in the previous chapters are rather complicated. This is due partly to the complexity of the problem formulation, but also to the complexity of the underlying method, i.e., the gradient projection method which needs delicate computational skills.

In this chapter, the second order algorithm is conveniently divided into several subalgorithms, or subprocedures. Unfortunately, most of subprocedures work nicely only for problems that have special properties.

In the present thesis, the algorithm is motivated by engineering purpose and is numerical solution oriented. Hence, the major contents of the present chapter are based on numerical experience with a number of engineering problems and contains the description of parameter choice for implementation, a discussion of difficulties involved, and an inclusion of a heuristic step for certain types of problems. As is true with the rather well established methods of solving nonlinear programming problems, such as the gradient projection method, method of feasible directions, and various other steepest descent techniques, there is a trade-off between being mathematically precise and reducing the amount of computational and programming effort.

6.2 Description of Computer Program

6.2.1 Simplified Block Diagram of POD Algorithm

As shown in Figure 7, the whole program consists of five major steps. With the estimated designs and parameters, derivatives and function values are calculated by subroutines. After checking the constraints, various matrices involved in the theoretical algorithm of the previous chapter are constructed. If necessary, a maximization procedure is executed, using the matrices calculated

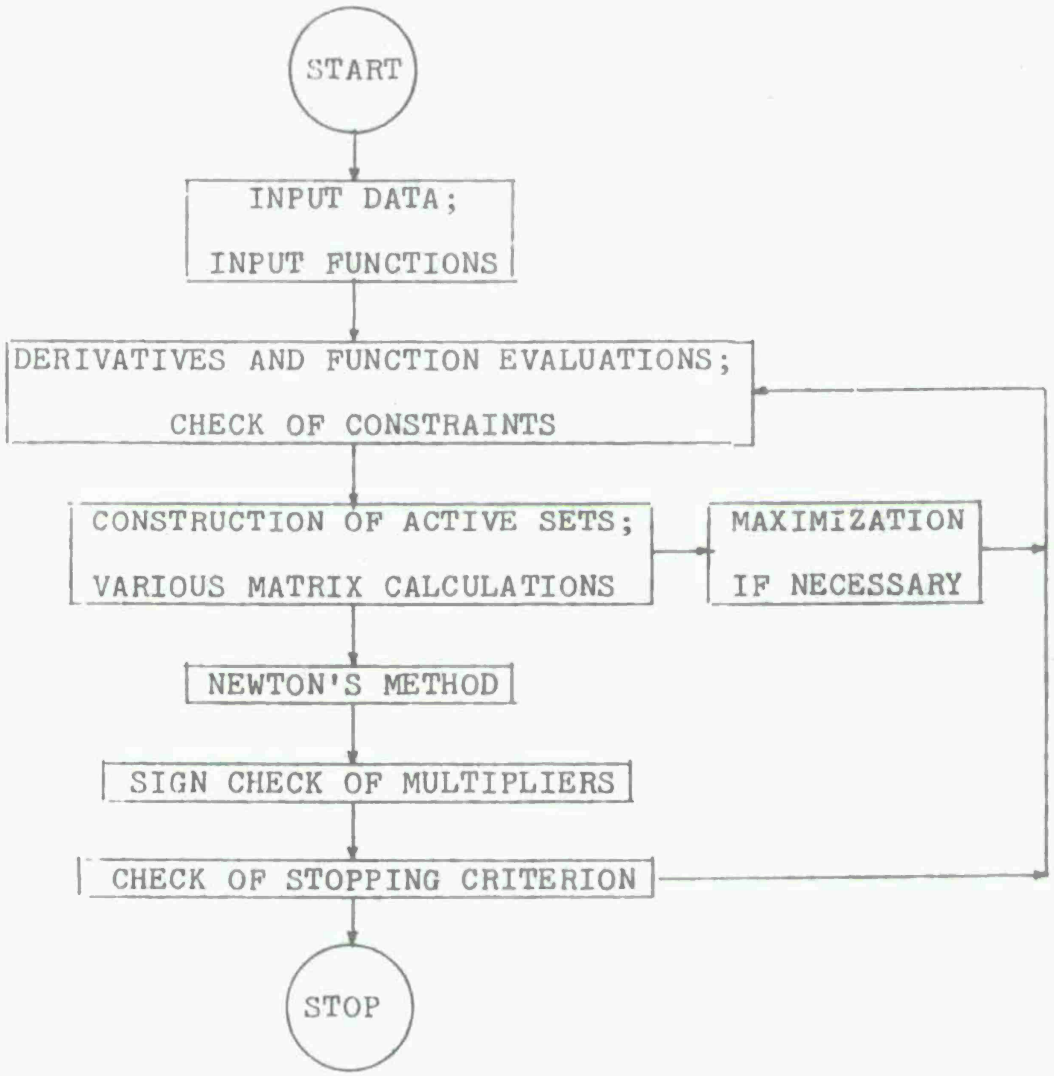


Figure 7 Simplified Block Diagram of POD

above. The Newton's method is applied to solve for design modifications and Lagrange multipliers. As is the case in the algorithm developed in the previous chapter, signs of the multipliers corresponding to inequality constraints must be checked. If some are negative, associated constraints are eliminated and the active constraint set is redefined. A stopping criteria is checked and iteration is terminated, or is continued with improved designs and free parameters. In the following sections, these major steps are described in detail.

6.2.2 Explanation of the Major Steps

(1) If state equations are involved, a subroutine (STATE) is constructed to solve for the values of state variables and the matrices described in Appendix D. Subroutines are constructed for the functionals of subproblems, in order to construct the necessary derivatives and functional values. At this stage, the constraint functionals are checked for violations. Even though a parametric constraint is strictly satisfied, it is necessary to retain all derivatives required to adjust the worst case, since the maximum point is going to be changed in later steps, and may become an active constraint. It is observed in nonlinear programming that if the iterates are near the solution, the active set remains the same during the iterations. In this case the inactive set can be dropped,

eliminating unnecessary computations. Caution is necessary to retain the global maximum of the subproblem, when it is not a concave programming problem.

(2) With the return of derivatives and function values from the appropriate subroutines, the various matrices in Eqs. (4.15), (4.16), (4.17), and (4.7) are constructed. In constructing the matrices ℓ^{ϕ} and L^{ϕ} in Eq. (4.14), the active constraints are defined through use of an ε-active index set, introduced in Section 2.4.2. By removing all the inequalities, $g_i + \ell_i^{\phi^T}\delta b + \frac{1}{2}\delta b^T L_i^{\phi}\delta b - K_i^{\phi} \leq 0$, for which $g_i < 0$, a zigzagging phenomenon would take place. To reduce this phenomenon, constraints with sufficiently negative values are removed, keeping weakly negative and strictly violated constraints in the active set. The same idea can be used for α-constraints.

With these index sets defined, the derivative matrix of Eq. (4.23) is constructed.

(3) In solving the algebraic Eqs. (4.20), (4.21), and (4.22) for δb_i, λ_i, and γ by the Newton's method, an initial estimate is made according to the suggestion in Section 4.3. From Eq. (4.24), $||\delta b^1||_{W_b} = \delta b^{1T} W_b \delta b^1$ is calculated, and used as a stopping criterion. Because of the nonlinearity of the equations, no unique solution is expected. Among those solutions that exist, the solution giving the smallest value of $\delta b^T W_b \delta b$ is chosen.

(4) With the designs fixed, a "sufficiently" accurate maximum point for each subproblem is often desired, during the iterations as well as at the beginning of the optimization process, even though the second order algorithm achieves maximization, as Eq. (4.7) shows. For this purpose one may go through the outer iterations to compute the vector C of Eq. (4.7), without calculating irrelevant matrices. This step is denoted as the maximization procedure in Figure 7. The expression for the vector C in Eq. (4.7) is shown to be different from that obtained from the first order gradient projection method (See Section 2.4), only in the matrix Φ of Eq. (4.11). This similarity arises from the fact that the α-constraint was expanded only up to first order terms, in Eq. (4.5). The algebraic signs of the multipliers associated with the α-constraints, given in Eq. (4.10), must be checked.

The estimation of the undetermined constant ν, which determines stepsize, will be considered in the next section.

(5) The solution of the algebraic equations (4.20), (4.21), and (4.22), may not be the desired solution of the approximate problem. One has to satisfy the positivity conditions of the multipliers. Hence, the next step is a check of the signs of these multipliers, including the multipliers corresponding to the α-constraints.

Since these multipliers are implicitly related, one is not sure whether the removal of any α-constraints corresponding to negative multipliers can make the negative multipliers corresponding to the b-constraints positive. At this stage, the question is not settled.

In the present program, the multipliers corresponding to the α-constraints, μ, are checked first. If some are negative, removal of the α-constraints corresponding to these negative multipliers follows. At the same time, the sign of multiplier, λ, corresponding to b-constraints is checked. Next, if γ is negative, the stepsize constraint is removed.

(6) With all multipliers nonnegative, the design and free parameters are updated by adding the increment formed from the Newton procedure, and Eq. (4.7).

By the solution $\bar{b}$ and $\bar{\alpha}_i$ of POD, one means that for fixed $\bar{b}$, $\bar{\alpha}_i$ is the maximum point of g_i, under the α-constraints. Hence, $||\delta\alpha^1||$ should be zero (See Section 2.4). For fixed $\bar{\alpha}_i$, $\bar{b}$ is the minimum point of the objective function subject to the b-constraints, satisfying $||\delta b^1|| = 0$.

6.3 Various Other Devices

6.3.1 The Choice of the Multiplier ν

In Eq. (4.11), the constant ν must be known to obtain the matrices D and C. It is known that in the first order gradient projection method, the multiplier associated with the stepsize restriction is inversely related to the stepsize in the direction of steepest descent [33]. Like the other multipliers, ν could be determined in the first order method by using the stepsize restriction, considered as equality instead of inequality. The stepsize restriction, however, has an undetermined constant ξ^2, yet to be decided. In practice, one chooses the stepsize ν and lets ξ^2 be determined accordingly.

In the first order gradient projection method, since ν is the inverse of the stepsize, it is sometimes easy to estimate a value for ν based on physical considerations. For the present formulation, however, since ν does not have an explicit physical interpretation, one must choose ξ^2 in Eq. (4.6) and have ν determined correspondingly. The situation here does not allow one to compute ν exactly. An approximate estimation is suggested.

Consider the second order maximization described in Section 5.3. From Eq. (5.4), whatever value of ν is chosen (assuming that this can assure a small step, $\delta\alpha$), the term compensating the constraint error, $\Phi^{-1}M_q\Delta q$, is determined

accordingly. By this observation, let the value of ν be determined with the α-constraints neglected. Even in this unconstrained case, the optimality of this multiplier as related to stepsize is conditional, as long as the choice of ξ^2 is not optimal. The value of this multiplier, however, can indicate the order of magnitude of the multiplier for the subproblem with the α-constraints included.

A procedure for finding the multiplier for the subproblem with α-constraints neglected follows. With $\delta b=0$, this procedure is exactly that of determining the constant for the Newton's method of maximization, described in Section 5.3. It was noted that the approach there gives a different point of view from those in the literature [45, 34]. In this literature, ν is determined so that the combined matrix Φ has the property of definiteness, for convergence. Hence, an eigenvalue analysis or decomposition of matrices is necessary for the determination of the constant. In Section 5.3, it has been shown that ν can be interpreted as the Lagrange multiplier for the stepsize restriction. Hence, in the following, ν will be determined in terms of ξ as a Lagrange multiplier.

For problems that are linear in $\delta\alpha$, one has from Eq. (5.1), to maximize

$$A^T\delta\alpha \tag{6.1}$$

subject to

$$\delta\alpha^T W_\alpha \delta\alpha \leq \xi^2. \tag{6.2}$$

Since the objective is linear, the inequality constraint can be replaced by an equality. Putting

$$\delta\alpha = \frac{W_\alpha^{-1}}{2} A, \tag{6.3}$$

one may determine

$$\nu = \frac{A^T W_\alpha^{-1} A}{2}. \tag{6.4}$$

For problems that are quadratic on $\delta\alpha$, one has, from Eq. (5.1), to maximize

$$A^T\delta\alpha + \frac{1}{2}\delta\alpha^T \overline{AA}\delta\alpha, \tag{6.5}$$

subject to

$$\delta\alpha^T W_\alpha \delta\alpha \leq \xi^2. \tag{6.6}$$

From Eq. (5.4), one finds

$$\delta\alpha = -(\overline{AA} - 2\nu W_\alpha)^{-1} A. \tag{6.7}$$

To find ν for this problem, an iterative scheme is employed. Let

$$h(\nu) = -(\overline{AA} - 2\nu W_\alpha)^{-1} A. \tag{6.8}$$

Assume a ν^o to get the direction $h \equiv h(\nu^o)$ and maximize the objective along h to find $\delta\alpha^o = \beta h(\nu^o)$, where $\beta > 0$. That is, maximize

$$\delta g \equiv \beta A^T h + \frac{\beta^2}{2} h^T \overline{AA} \cdot h, \tag{6.9}$$

subject to

$$\beta^2 h^T W_\alpha h \leq \xi^2. \tag{6.10}$$

Now, for the equality case

$$\beta = \frac{\xi}{\sqrt{h^T W_\alpha h}}. \tag{6.11}$$

For the inequality case,

$$A^T h + \beta h^T \overline{AA} \cdot h = 0,$$

so

$$\beta = - \frac{A^T h}{h^T \overline{AA} \cdot h} . \qquad (6.12)$$

By choosing the smaller, nonnegative β from the above, one has

$$\beta h(\nu^o) = -(\overline{AA} - 2\nu W_\alpha)^{-1} A,$$

or solving for ν,

$$\nu = \frac{\beta h^T \overline{AA} \cdot h + h^T A}{2\beta h^T W_\alpha \cdot h}, \qquad (6.13)$$

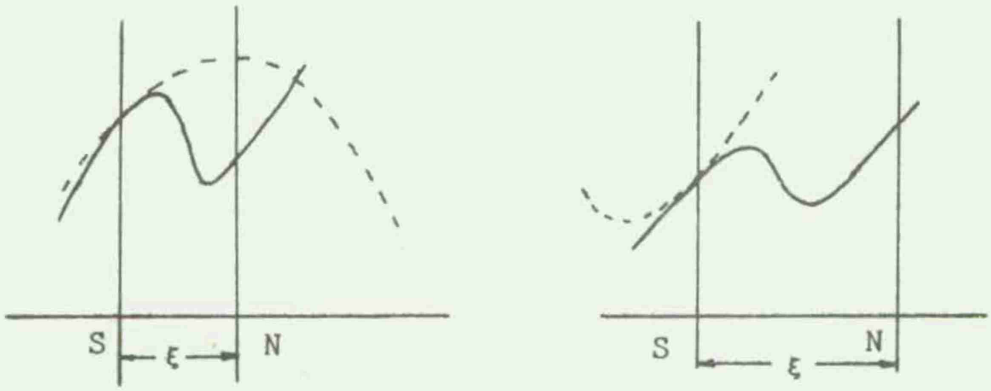
which is the improved value of ν. Defining the improved direction $h=h(\nu)$, the procedure is repeated until the change of ν is within a prescribed limit.

For the POD problem, the value of ν chosen in this way is only an approximation, due to the presence of the term, $\overline{BA}^T \delta b$, in Eq. (4.7), where δb is still unknown. As noted earlier, in the presence of α-constraints, the error from the optimal ν would be increased. This latter error, however, would not be significant as long as one could still move on the tangent plane of the active constraints.

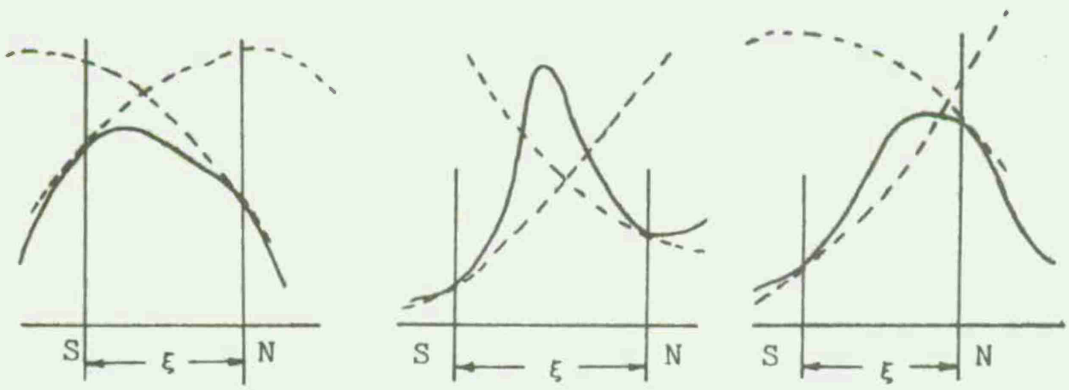
Difficulties can be encountered when ξ^2 is too large. Such problems in the one dimensional case are shown in Figure 8. If ξ^2 is small enough, these difficulties can be avoided, but this may require excessive computing.

For the first case in Figure 8, there seems to be no

(i) Missing completely



(ii) Oscillating indefinitely



S ≡ starting point

N ≡ next point

Figure 8 Difficulties in the Choice of Stepsize

solution except to use sufficiently small ξ^2 from the start. For the second case, one can reduce ξ^2 when oscillation is observed.

It is evident that, when oscillation occurs between two points, there exists a local maximum between the point points. To discover this oscillation, the inner product $-\delta\alpha^{(i-1)T}\cdot\delta\alpha^{(i)}/||\delta\alpha^{(i-1)}||\cdot||\delta\alpha^{(i)}||$ is checked. If it is larger than a prescribed number, say 0.9, then oscillation is clearly occurring.

As an example of these difficulties, one may have a practical situation of the following form of constraints (which is the subproblem of POD):

$$a|z_1(\alpha)| \leq z_2(\alpha), \text{ for all } \alpha \in A. \qquad (6.14)$$

When $a\geq 0$, there is no difficulty. When $a<0$ however, at a point where $z_1(\alpha)=0$, one may have a local maximum point, where the derivatives are discontinuous (not satisfying the assumptions of POD). This will give an oscillatory behaviour around that maximum point.

6.3.2 A Suggestion on the Choice of γ and W_b

In the gradient projection method with constraint error compensation, described in Section 2.4, there is an undetermined constant η^2 in the stepsize restriction given by Eq. (2.54). From Eq. (2.55), it is easily seen that the multiplier γ is directly related to the stepsize. Since no optimal choice of η^2 is known, the precise

determination of γ has little meaning (Compare with the discussion in the previous section). Hence it might be possible to make γ nonzero by assuming that η^2 were adjusted such that $\delta b^T W_b \delta b = \eta^2$, still satisfying the condition $\gamma(\delta b^T W_b \delta b - \eta^2) = 0$, from the Kuhn-Tucker theorem. In reality, one can think that the stepsize restriction is always active, meaning that γ should be positive. Hence, in the following, $\gamma > 0$ will be assumed. Then, from Eqs. (2.54) and (2.62),

$$\eta^2 \geq \delta b^T W_b \delta b = \frac{\delta b^{1T} W_b \delta b^1}{4\gamma^2} + \delta b^{2T} W_b \delta b^2, \qquad (6.15)$$

since $\delta b^{1T} W_b \delta b^2 = 0$. This allows an explicit lower bound for γ such that

$$\gamma \geq \frac{\delta b^{1T} W_b \delta b^2}{2\sqrt{\eta^2 - \delta b^{2T} W_b \delta b^2}}, \qquad (6.16)$$

where $\eta^2 > \delta b^{2T} W_b \delta b^2$ is assumed. Otherwise, a compensation procedure, $\delta b = \delta b^2$, may first be necessary.

From computational experience, W_b is usually taken as a diagonal matrix such that

$$\tilde{b}_i W_{b_i} \tilde{b}_i = 1, \quad i = 1, \cdots, n, \qquad (6.17)$$

where $\tilde{b}_i$ denotes the magnitude of the estimated design component and W_{b_i} denotes the diagonal element of W_b. Then, η is taken between 0.01 and 0.1, at the beginning. Adjustment of γ during iterations often leads to a better convergence.

6.3.3 Preliminary Elimination Procedure

In some cases, the vectors ℓ_j^{ϕ}, $j \in I_\varepsilon$, are not linearly independent, giving a singular matrix in Eq. (4.23). This usually happens when several candidates for local maximum points are assumed at the beginning, for the same g_i. With poor initial estimates for designs and free parameters, there can be too many constraints violated. To prevent this from happening, a preliminary elimination procedure may be applied.

According to the theorem due to Fritz John (See Section 2.4), there can be at most n(=dimension of the design variable vector) tight constraints, which are enough to characterize the solution point $\bar{b}$. Therefore, if there were more constraints violated than the number n, it might be useful to consider only the more "significant" constraints. Suppose g_i is violated and if $\nabla g_i \nabla f^T = \ell_i^{\phi T} \ell^J > 0$, one may expect that after an iteration this violation would be compensated for, even though g_i is removed from the active set. Such constraints are, thus, deleted.

6.3.4 Merge Test

A "merge test" is given as often as required to check whether the assumed local candidate maximum points for g_i are near enough to give a singular matrix. Since one has to include every local maximum point in the effort to find

the global maximum point for g_1, there is a tendency to have more inner problems than are really necessary. The merge test is an important step to reduce the amount of computation by eliminating unnecessary inner problems, as well as preventing linear dependency of the gradients of g_1's. For a check of nearness of two vectors, one may use the maximum norm of the difference vector.

6.4 Integration of the Algorithm

6.4.1 Introduction

Most of the steps described above need to be repeated iteratively to obtain accurate results. Consider the construction of a sequence b^i converging to the limit point $\bar{b}$ by using another sequence $\delta b_1^i, \delta b_2^i, \cdots$, that converges to $\overline{\delta b^i}$. One then puts $b^{i+1} = b^i + \overline{\delta b^i}$. This latter construction may be termed a subalgorithm.

Theoretically, one can never compute the exact limit point. In practice, therefore, one has to truncate the construction of the sequence. If one lets the subsequence run too long for each (outer) iteration, one may be using more computer time than is really needed. On the other hand, if one truncates too soon, he may degrade convergence in the outer iteration. The best truncation procedure is not known. An approach to introducing a truncation procedure is described in [36], and is used in the present work.

Since one can measure the closeness of the current iterate to the limit point by a scalar quantity, one relates this quantity to the subalgorithm, such that a correspondingly accurate iterate for the subsequence is obtained. In reality, this quantity is not known until the limit point is obtained. In the algorithm model shown below, one reduces a scalar quantity from a given value by a predetermined ratio, until the value is less than a prescribed small number. Such an algorithm is cited in Figure 9 from [36], where $S(\varepsilon, b^i)$ denotes the subalgorithm, $c(b^i)$ is a scalar quantity that relates the closeness of the current b^i to $\bar{b}$, $\alpha>0$, $\beta\varepsilon(0,1)$, $\varepsilon_o>0$ and $\varepsilon_s\varepsilon(0,\varepsilon_o)$ are predetermined numbers. In certain cases, the regular reducing of ε by fixed ratio β would result in slow convergence. In this case, the algorithm can be modified by choosing β differently after each iteration. Such a modification is included in the present program.

The truncation procedure, as explained in this section using a decreasing ε, will be termed an ε-procedure. A convergence property for this algorithm is described in the same reference, under assumptions originally due to Zangwill [48].

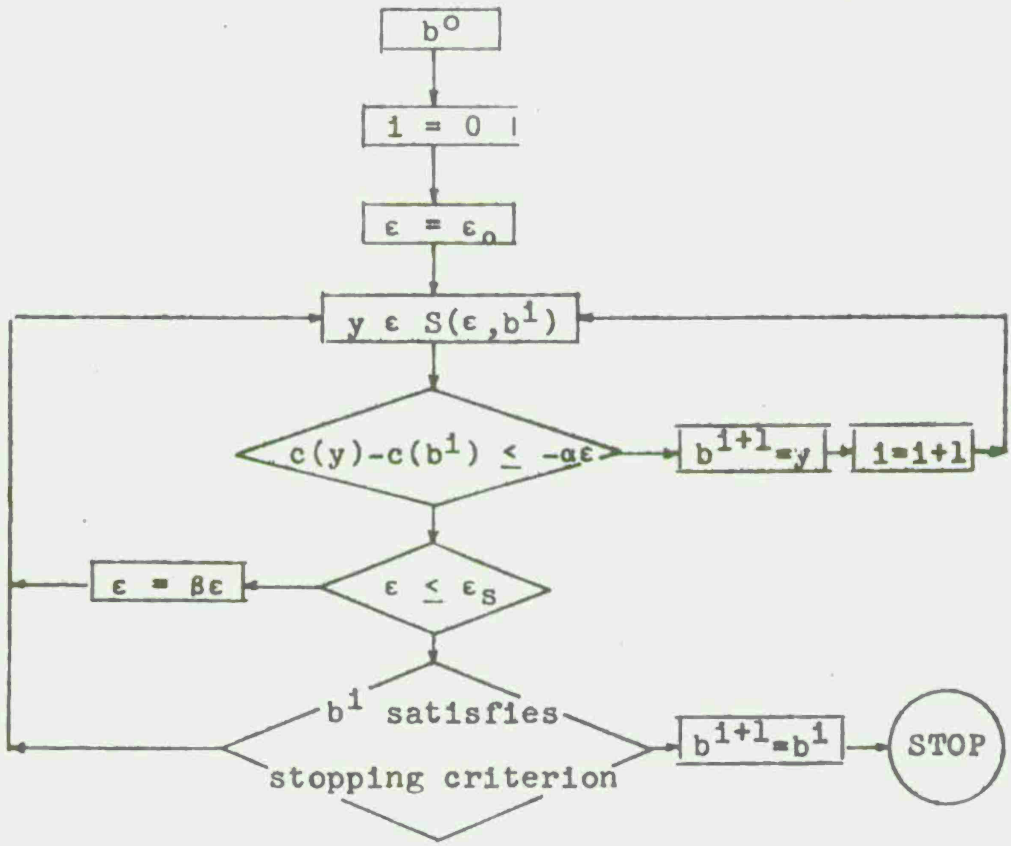


Figure 9 An ε-procedure for an Algorithm $S(\varepsilon, b^i)$

6.4.2 Application of the ε-procedure to POD

For the present problem, the Newton's method employed to generate $b+\delta b$ will be identified as a subalgorithm $S(\varepsilon, b)$, with the ε involved in the stopping criterion for the Newton's method. The scalar quantity $c(b^1)$ will be identified as $||\delta b^1||$. For a normal problem, the Newton's method should generate points such that $||\delta b^1||$ is reduced. But for POD, several other procedures are involved and often some elements b^1 of the sequence, generated during the iterations, are not even feasible points, obscuring the convergence. Convergence is further obscured due to the convergence property of the Newton's method itself, especially for global convergence.

The same type of ε-procedure as applied to the Newton's method above can be used in the truncation of the sequence generated by the maximization procedure and in the sign check of the various multipliers corresponding to ε-active constraints. For the latter procedure, the subalgorithm is similar to that described in [36], except that for the present computer program the Lagrange multipliers and δb are calculated by another subprocedure, the Newton's method. Thus the overall algorithm for POD contains several subprocedures, interwoven and nested in such a way that it is much more complex than conventional algorithms for nonlinear programming.

The essential part of the flow chart of the POD program is shown schematically in Figure 10. Major subalgorithms are as follows. Loop (1) denotes a Newton's method, whose number of iterations depends on the parameter ε_A, which, in turn, is reduced by a factor of $\beta\varepsilon(0,1)$ when b^i is near the solution. Loop (2) denotes the sign check of various mulitpliers. Here, also, a slight negative value of multipliers, by an amount of ε_A, is considered to satisfy the positivity condition. Whenever the accuracy of the maximum point, compared with that of design variable is poor, the maximization procedure denoted by Loop (3) is executed until an accuracy of ε_B is obtained. In this case, the branching variable MAX is set equal to one. In some situations, a regular decrease of ε_A by a constant factor β is inefficient, resulting in too much time in Loop (4), which decreases ε_A to an acceptable value ε_S. To check whether this is the case, a rapid decrease in ε_A is inserted in Loop (4) by use of the branching variable IEADJ. When this rapid decrease in ε_A does not stop the program, the original value of ε_A stored in ε_A' is recovered. The variables, ITNEW, INNER, IMAX, and ITER register the number of iterations for the subalgorithms, and are supposed not to exceed given allowable numbers, although not explicitly expressed in the flow chart.

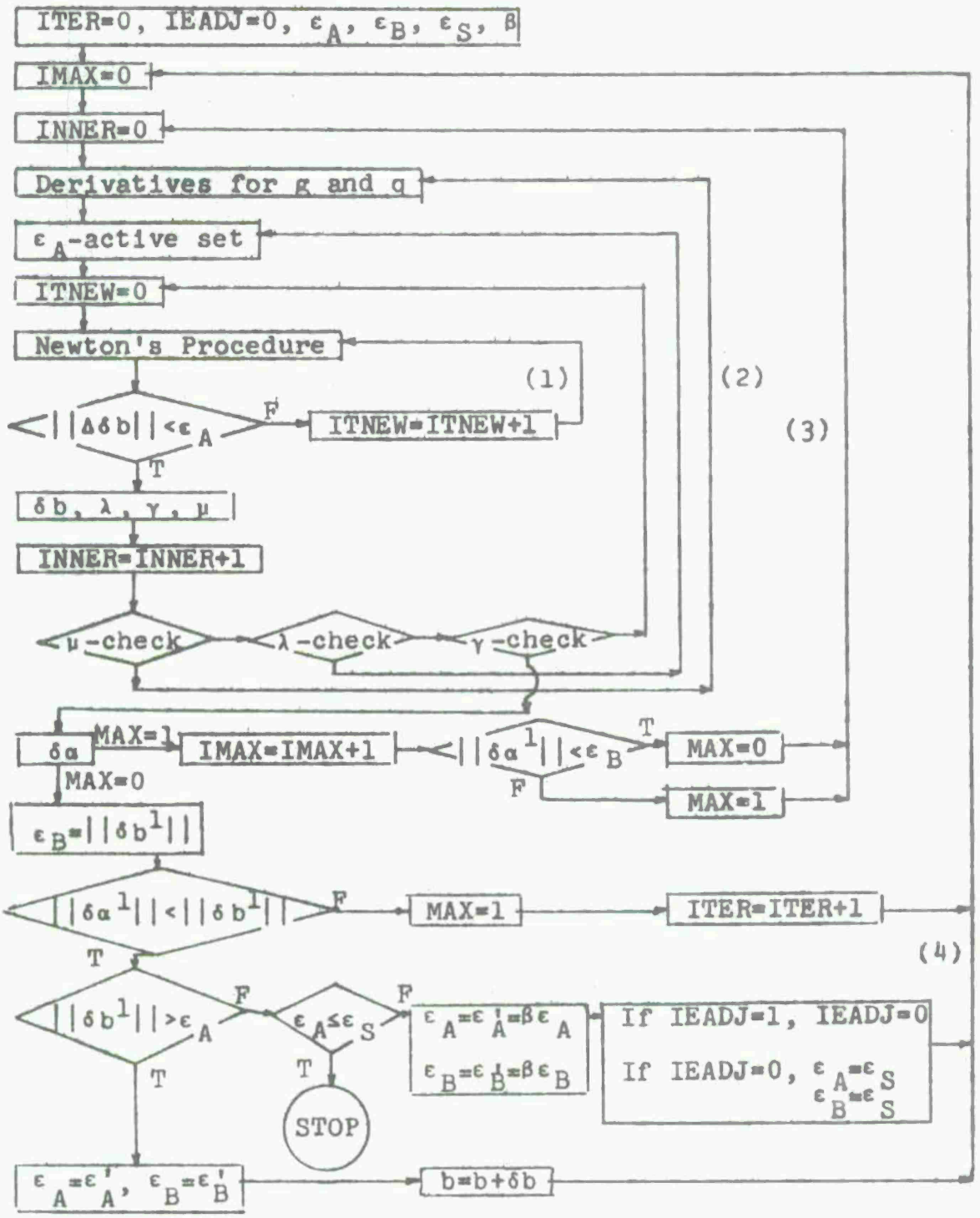


Figure 10 Schematic Flow Chart of POD

6.5 Conclusions

A simplified block diagram of the algorithm developed in Chapter 4 has been described. Several suggestions regarding selection of parameters in the algorithm have been given, based on numerical experience. It is pointed out that the conclusions of the present chapter draw on experimental results. Mathematical investigation of the algorithms developed, thus, must be a subject of future study.

CHAPTER 7

SUMMARY AND CONCLUSIONS

In this thesis, parametric optimal design problems, motivated from the concept of environmental parameters, have been defined. Solution algorithms have been developed and applied to the solution of a series of approximate problems.

The basic theoretical tool employed is the implicit function theorem, applied to the state equations and the expansion procedure described in Sections 2.1.3 and 2.4.6. Algorithms are generated, based on different precision of approximations. In Chapter 3, a first order algorithm is developed, which is equivalent to the alternating maximization and minimization technique for min-max problems. Numerical examples show that, although convergence is slow near the solution, the method is reliable.

In Chapter 4, a second order approximation has been utilized. The inclusion of second order terms is motivated from the desire to adjust the maximum points in accordance with the change in the design variable. This latter idea leads directly to sensitivity analysis for the maximum point. It has been pointed out that this analysis gives a first order feedback law for the inner problem.

This relation is essential to developing various other types of alternative methods. The numerical results given in Chapters 4 and 5 show that, near the solution, this procedure converges rapidly. During the sensitivity analysis in Section 4.2, a compensating term for the error in the nominal maximum point is found to be the same as that obtained from a second order method of solving a nonlinear programming problem. This method of maximization gives a different point of view for the damping factor, as appeared in a damped Newton's method [45,47]. In Chapter 5, the possibility of some other variations to the first and second order algorithms has been discussed, with numerical solutions of example problems presented.

Numerical experimentation with the algorithms shows that their success is problem dependent. That is, the structure of the specific problem and the nature of the functions involved are determining factors. Also, it has been shown in Chapter 6 that the choice of parameters in the algorithm, for a given problem, is crucial.

Table 17 summarizes the general features of the various algorithms, based on the numerical experimentation. In general, the algorithms work well, although there is room for further development. It is hoped that the present development of algorithms; based on the idea of expansion procedures, sensitivity analysis with error compensation, and second order methods of maximization; will

motivate theoretical as well as practical investigation of the parametric optimal design problem.

Table 17 Comparison of Algorithms

	1-st order algorithm	Hybrid method	2-nd order algorithm
1. reliability	excellent	excellent	dependent on initial estimates and problem structure
2. convergence*	slow	good	fast
3. programming difficulty	simplest	simple	complicated
4. computational effort	small	reasonable	large
5. underlying methods	gradient projection	2-nd order maximization, sensitivity analysis, gradient projection	2-nd order maximization, sensitivity analysis, Newton's method
6. separate maximizations	essential	not essential	not essential
7. overall justification†	good	better	problem dependent

*The rate of convergence appears to approach that of the first order method (steepest descent) and second order method (Newton's method) of NLP, as the nominal design approaches to the solution. This is because the solutions of inner problems do not change very much and the POD problem behaves like a NLP problem.

†In general, quite problem dependent.

APPENDIX D

THE VARIATION OF g IN TERMS OF δb AND δa

Since it is assumed that the system of state equations is normal, i.e., $\frac{\partial h}{\partial z} \neq 0$, $z = z(b,a)$, $g(z,b,a) = G(b,a)$. It is necessary, in Section 4.2, to calculate the derivatives of $g(z(b,a),b,a)$ with respect to b and a. By use of chain rule of differentiation,

$$\frac{\partial G}{\partial b} = \frac{\partial g}{\partial b} + \frac{\partial g}{\partial z}\frac{\partial z}{\partial b}$$

$$\frac{\partial G}{\partial a} = \frac{\partial g}{\partial a} + \frac{\partial g}{\partial z}\frac{\partial z}{\partial a}$$

$$\frac{\partial^2 G}{\partial b^2} = \frac{\partial^2 g}{\partial b^2} + \frac{\partial^2 g}{\partial b \partial z}\frac{\partial z}{\partial b} + \frac{\partial z^T}{\partial b}\frac{\partial^2 g}{\partial z \partial b} + \frac{\partial z^T}{\partial b}\frac{\partial^2 g}{\partial z^2}\frac{\partial z}{\partial b} + \frac{\partial g}{\partial z}\frac{\partial^2 z}{\partial b^2}$$

$$\frac{\partial^2 G}{\partial a^2} = \frac{\partial^2 g}{\partial a^2} + \frac{\partial^2 g}{\partial a \partial z}\frac{\partial z}{\partial a} + \frac{\partial z^T}{\partial a}\frac{\partial^2 g}{\partial z \partial a} + \frac{\partial z^T}{\partial a}\frac{\partial^2 g}{\partial z^2}\frac{\partial z}{\partial a} + \frac{\partial g}{\partial z}\frac{\partial^2 z}{\partial a^2}$$

$$\frac{\partial^2 G}{\partial b \partial a} = \frac{\partial^2 g}{\partial b \partial a} + \frac{\partial^2 g}{\partial b \partial z}\frac{\partial z}{\partial a} + \frac{\partial z^T}{\partial b}\frac{\partial^2 g}{\partial z \partial a} + \frac{\partial z^T}{\partial b}\frac{\partial^2 g}{\partial z^2}\frac{\partial z}{\partial a} + \frac{\partial g}{\partial z}\frac{\partial^2 z}{\partial b \partial a}$$

where

$$\left(\frac{\partial g}{\partial z}\frac{\partial^2 z}{\partial b^2}\right)_{ij} \equiv \sum_k \frac{\partial g}{\partial z_k}\frac{\partial^2 z_k}{\partial b_i \partial b_j}, \text{ etc.}$$

Then, to second order terms

$$\delta g = \delta G$$

$$= \frac{\partial G}{\partial b}\delta b + \frac{\partial G}{\partial a}\delta a + \frac{1}{2}\delta b^T \frac{\partial^2 G}{\partial b^2}\delta b + \delta b^T \frac{\partial^2 G}{\partial b \partial a}\delta a + \frac{1}{2}\delta a^T \frac{\partial^2 G}{\partial a^2}\delta a,$$

which is the desired expression. Comparing this formula with Eq. (4.4), the matrices B, A, $\overline{BB}$, $\overline{AA}$, and $\overline{BA}$ are

identified.

To eliminate various derivatives of z in the most general case, differentiation of $h(z,b,\alpha)=0$ yields

$$\frac{\partial h}{\partial z}\frac{\partial z}{\partial b} + \frac{\partial h}{\partial b} = 0,$$

from which $\frac{\partial z}{\partial b}$ is obtained, either by solving sets of linear equations or computing $\frac{\partial z}{\partial b} = -(\frac{\partial h}{\partial z})^{-1}\frac{\partial h}{\partial b}$, if the inverse is available. Similarly,

$$\frac{\partial h}{\partial z}\frac{\partial z}{\partial \alpha} + \frac{\partial h}{\partial \alpha} = 0, \text{ or } \frac{\partial z}{\partial \alpha} = -(\frac{\partial h}{\partial z})^{-1}\frac{\partial h}{\partial \alpha}.$$

Differentiating again,

$$(\frac{\partial z^T}{\partial b}\frac{\partial^2 h}{\partial z^2}+\frac{\partial^2 h}{\partial b \partial z})\frac{\partial z}{\partial b} + \frac{\partial h}{\partial z}\frac{\partial^2 z}{\partial b^2} + \frac{\partial^2 h}{\partial b^2} + \frac{\partial z^T}{\partial b}\frac{\partial^2 h}{\partial z \partial b} = 0.$$

Hence,

$$\frac{\partial h}{\partial z}\frac{\partial^2 z}{\partial b^2} = -\frac{\partial^2 h}{\partial b^2} - (\frac{\partial^2 h}{\partial b \partial z}\frac{\partial z}{\partial b}+\frac{\partial z^T}{\partial b}\frac{\partial^2 h}{\partial z \partial b}) - \frac{\partial z^T}{\partial b}\frac{\partial^2 h}{\partial z^2}\frac{\partial z}{\partial b}.$$

Similarly,

$$\frac{\partial h}{\partial z}\frac{\partial^2 z}{\partial \alpha^2} = -\frac{\partial^2 h}{\partial \alpha^2} - (\frac{\partial^2 h}{\partial \alpha \partial z}\frac{\partial z}{\partial \alpha}+\frac{\partial z^T}{\partial \alpha}\frac{\partial^2 h}{\partial z \partial \alpha}) - \frac{\partial z^T}{\partial \alpha}\frac{\partial^2 h}{\partial z^2}\frac{\partial z}{\partial \alpha},$$

and

$$\frac{\partial h}{\partial z}\frac{\partial^2 z}{\partial b \partial \alpha} = -\frac{\partial^2 h}{\partial b \partial \alpha} - (\frac{\partial^2 h}{\partial b \partial z}\frac{\partial z}{\partial \alpha}+\frac{\partial z^T}{\partial b}\frac{\partial^2 h}{\partial z \partial \alpha}) - \frac{\partial z^T}{\partial b}\frac{\partial^2 h}{\partial z^2}\frac{\partial z}{\partial \alpha}.$$

It is observed that if the dimension of the state variable z is small, it may be computationally more profitable to compute $(\frac{\partial h}{\partial z})^{-1}$. In general, however, the computation is very time-consuming. As a close approximation, if the state equations are linearized as follows,

$$\frac{\partial h}{\partial z}\delta z + \frac{\partial h}{\partial b}\delta b + \frac{\partial h}{\partial \alpha}\delta \alpha = 0,$$

then, since the second derivatives of z are zero, the computation is enormously reduced.

APPENDIX E

EQUIVALENCE OF LAGRANGE MULTIPLIERS OF NLP AND THOSE OF ITS FIRST ORDER APPROXIMATE PROBLEM

The Lagrange multipliers for the approximate problem are given by Eq.(2.56). At the solution $\bar{b}$, $\Delta g = 0$. Hence,

$$\lambda = -M_{\phi\phi}^{-1} \, \ell^{\phi T} W^{-1} \ell^{J}, \tag{E.1}$$

where $M_{\phi\phi} = \ell^{\phi T} W^{-1} \ell^{\phi}$, and ℓ^{J} and ℓ^{ϕ} are calculated at $\bar{b}$. The subscript ϕ denotes the active constraints at $\bar{b}$.

Now consider the original NLP given by Eqs. (2.43) and (2.44) without equality constraints. At the solution $\bar{b}$, Kuhn-Tucker theorem assures the existence of multipliers such that

$$\frac{\partial H}{\partial b} = 0 = \frac{\partial f}{\partial b} + \lambda^{T} \frac{\partial \tilde{g}}{\partial b}, \tag{E.2}$$

where $H = f + \lambda^{T} \tilde{g}$, and the only possibly nonzero elements are kept for λ. That is, if $g_j < 0$, $\lambda_j = 0$ so the nonzero elements of λ correspond to the active constraints only. In the notation of ℓ^{J} and ℓ^{ϕ} calculated at $\bar{b}$,

$$0 = \ell^{JT} + \lambda^{T} \ell^{\phi T}. \tag{E.3}$$

Since the existence of λ is insured, postmultiplying through by $W^{-1} \ell^{\phi}$ and transposing, one has

$$(\ell^{\phi T} W^{-1} \ell^{\phi}) \lambda = -\ell^{\phi T} W^{-1} \ell^{J}, \tag{E.4}$$

which gives multipliers λ, same as obtained in Eq. (E.1).

REFERENCES

1. John, F., "Extremum Problems with Inequalities as Subsidiary Conditions," in Studies and Essays, Courant Anniversary Volume, edited by K. O. Friedricks, O. E. Neugebauer, and J. J. Stoker, Interscience, New York, 1948, pp. 187-204.

2. Gehner, K. R., "Optimization Problems with an Infinite Number of Constraints and Applications to Constrained Approximation Problems," Ph.D. Thesis, The University of Wisconsin, 1971.

3. McLinden, L., "Minimax Problems, Saddle Functions and Duality," MRC Technical Summary Report #1190, Mathematics Research Center, The University of Wisconsin, March 1972.

4. Barry, P. E., "Optimal Control with Minimax Cost," Ph.D. Thesis, State University of New York, September 1969.

5. Danskin, J. M., The Theory of Max-Min, Springer-Verlag New York Inc., 1967.

6. Pshenichnyi, B. N., Necessary Conditions for an Extremum, English Translation, Marcel Dekker, Inc., New York, 1971.

7. Heller, J. E., "A Gradient Algorithm for Minimax Design," Report R-406, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, January 1969.

8. Medanic, J., "On some Theoretical and Computational Aspects of the Minimax Problem," Report R-423, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, July 1969.

9. Bracken, J., and McGill, T., "Mathematical Programs with Optimization Problems in the Constraints," Operations Research, Vol. 21, No. 1, 1973, pp. 37-44.

10. Kwak, B. M., Rim, K., and Arora, J. S., "Dynamic

Analysis and Optimal Design Formulation of a Weapon-Vehicle System," Report #50, Project Themis, The University of Iowa, Iowa City, Iowa, April 1974.

11. Owen, G., Game Theory, W. B. Saunders Company, 1968.

12. Bryson, A. E., Jr., and Ho, Y. C., Applied Optimal Control, Ginn and Company, 1969.

13. Parthasarathy, T., and Raghavan, T. E. S., Some Topics in Two-Person Games, American Elsevier Publishing Company, Inc., 1971.

14. Garabedian, H. L., ed., Approximation of Functions, Elsevier Publishing Company, 1965.

15. Berezin, I. S., and Zhidkov, N. P., Computing Methods, Volume 1, Pergamon Press, 1965.

16. Ralston, A., A First Course in Numerical Analysis, McGraw-Hill Book Company, 1965.

17. Handscomb, D. C., ed., Methods of Numerical Approximation, Pergamon Press, 1966.

18. Cheney, E. W., Introduction to Approximation Theory, McGraw-Hill, New York, 1966.

19. Meinardus, G., Approximation of Functions: Theory and Numerical Methods, Springer-Verlag New York Inc., 1967.

20 Schoenberg, I. J., ed., Approximations with Special Emphasis on Spline Functions, Academic Press, 1969.

21. Voronovskaja, E. V., The Functional Method and Its Applications, American Mathematical Society, 1970.

22. Young, D. M., and Gregory, R. T., A Survey of Numerical Mathematics, Volume I, Addison-Wesley Publishing Company, 1972.

23. Collatz, L., "Applications of Nonlinear Optimization to Approximation Problems," in Integer and Nonlinear Programming, edited by J. Abadie, North-Holland Publishing Company, 1970, pp. 285-308.

24. Aoki, M., "Minimum Norm Problems and some other Control System Optimization Techniques," in Modern Control Systems Theory, edited by C. T. Leondes,

McGraw-Hill Book Company, Inc., 1965, pp. 319-354.

25. Luenberger, G. G., Optimization by Vector Space Methods, John Wiley & Sons, Inc., 1969.

26. Schmit, L. A., et al., Structural Synthesis, Vol. 1, Summer Course Notes, Case Institute of Technology, 1965.

27. Sved, G., and Ginos, Z., "Structural Optimization under Multiple Loading," International Journal of Mechanical Sciences, Vol. 10, 1968, pp. 803-805.

28. Haug, E. J., Jr., and Arora, J. S., "Structural Optimization via Steepest Descent and Interactive Computation," Proceedings, Third Conference on Matrix Methods in Structural Mechanics, Wright-Patterson AFB, Ohio, 1971.

29. Fox, R. L., Optimization Methods for Engineering Design, Addison-Wesley Publishing Company, 1971

30. Den Hartog, J. P., Mechanical Vibration, fourth ed., McGraw-Hill Book Company, Inc., New York, 1956.

31. Kwak, B. M., Arora, J. S., and Haug, E. J., Jr., "Optimum Design of Damped Vibration Absorbers," to be published in AIAA Journal, 1974

32. McMunn, J., "Multi-Parameter Optimum Damping in Linear Dynamical Systems," Ph.D. Thesis, The University of Minnesota, 1967.

33. Haug, E. J., Jr., Engineering Design Handbook, Computer Aided Design of Mechanical Systems, Systems Research Division, RD&E Directorate, US Army Armament Command, Rock Island, Illinois, 1973

34. Luenberger, D. G., Introduction to Linear and Nonlinear Programming, Addison-Wesley Publishing Company, 1973.

35. Fiacco, A. V., and McCormick, G. P., Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley and Sons, Inc., 1968.

36. Polak, E., Computational Methods in Optimization, A Unified Approach, Academic Press, Inc., 1971.

37. Beltrami, E. J., An Algorithmic Approach to Nonlinear Analysis and Optimization, Academic Press, Inc., 1970.

38. Girsanov, I. V., "Lectures on Mathematical Theory of Extremum Problems," Vol. 67, Lecture Notes in Economics and Mathematical Systems, edited by M. Beckmann, et al., Springer-Verlag, 1972.

39. Haug, E. J., Jr., Pan, K. C., and Streeter, T. D., "A Computational Method for Optimal Structural Design. I. Piecewise Uniform Structures," International Journal for Numerical Methods in Engineering, Vol. 5, 1972, pp. 171-184.

40. Melts, I. O., "Nonlinear Programming Methods for Optimizing Dynamical Systems in Function Space," Automation and Remote Control, No. 1, January 1968, pp. 68-73.

41. Rockafellar, R. T., Convex Analysis, Princeton University, Princeton, New Jersey, 1970.

42. Fiacco, A. V., "Sensitivity Analysis for Nonlinear Programming Using Penalty Methods," Technical Paper T-275, The Institute for Management Science and Engineering, The George Washington University, Washington, D.C., 1973.

43. Armacost, R. L., and Fiacco, A. V., "Computational Experience in Sensitivity Analysis for Nonlinear Programming," Technical Paper T-276, The Institute for Management Science and Engineering, The George Washington University, Washington, D.C., 1973

44. Armacost, R. L., and Mylander, W. C., "A Guide to a SUMT-Version 4 Computer Subroutine for Implementing Sensitivity Analysis in Nonlinear Programming," Technical Paper T-287, The Institute for Management and Engineering, The George Washington University, Washington, D.C., 1973

45. Ortega, J. M., and Rheinboldt, W. C., Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, Inc., 1970.

46. Murray, W., ed., Numerical Methods for Unconstrained Optimization, Academic Press, Inc., 1972.

47. Levenberg, K., "A Method for the Solution of Certain Non-linear Problems in Least Squares," Quarterly of Applied Mathematics, Vol. 2, No. 2, 1944, pp. 164-168.

48. Zangwill, W. I., Nonlinear Programming: A Unified Approach, Prentice-Hall, Englewood Cliffs, New Jersey, 1969.

49. Crandall, S. H., and Dahl, N. C., ed., An Introduction to the Mechanics of Solids, McGraw-Hill Book Company, Inc., 1959.

50. Arora, J. S., "Optimal Design of Elastic Structures under Multiple Constraint Conditions," Ph.D. Dissertation, The University of Iowa, 1971.

51. Simmons, G. F., Introduction to Topology and Modern Analysis, McGraw-Hill Book Company, Inc., 1963.