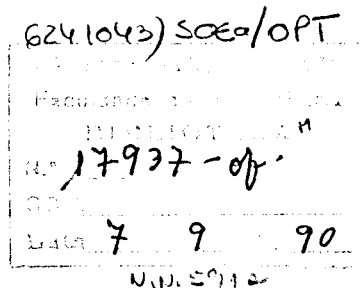


OPTIMIZATION OF REINFORCED
CONCRETE FRAMES USING
INTEGRATED ANALYSIS AND RELIABILITY

By

ALFREDO V. SOEIRO



A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1989

043D
8664
8

ACKNOWLEDGEMENTS

I want to express my sincerest gratitude to Dr. Marc Hoit for monitoring my research, for the inventive ideas and for his constant support. I am specially thankful to Dr. Fernando Fagundo for the productive discussions, his friendship and his vigorous encouragement. I owe to these two my best recollections from the University of Florida. My sincerest appreciation is extended to Dr. Prabhat Hajela for the teachings and the careful reading of my dissertation. My indebtment goes also to Dr. Clifford Hays and Dr. John Lybas for the useful conversations and their activity in my committee. I am also grateful to Dr. David Bloomquist for his support to initiate my geotechnical reliability research and his continuous disposition to help.

I would also like to acknowledge my appreciation to the Fulbright Comission and to the Department of Civil Engineering of the University of Florida for their financial aid and the opportunity to study the fascinating area of structural optimization. My thanks go also to the Department of Civil Engineering of the University of Porto, specially to Dr. Adão da Fonseca, for giving me the possibility to research and study in the USA.

My sincere appreciation and best remembrances go to my friends in the Gainesville Portuguese community and to my colleagues Jose, Joon, Lin and Prasit that helped smoothe the life contours created by the research work. Finally, my gratitude goes to my wife, Paula, for her work, her patience and her support throughout the whole period during which this dissertation was completed.

TABLE OF CONTENTS

	<u>Page</u>
ACKNOWLEDGEMENTS.....	ii
LIST OF TABLES.....	vi
LIST OF FIGURES.....	vii
ABSTRACT.....	viii
 CHAPTERS	
1 STRUCTURAL OPTIMIZATION.....	1
Introduction.....	1
Historical Background.....	3
Methods.....	6
Typical Applications.....	8
Study Objectives.....	15
Summary.....	17
2 INTEGRATED OPTIMIZATION OF LINEAR FRAMES.....	21
Original Research.....	21
Augmented Lagrangian Function.....	22
Unconstrained Minimization Techniques.....	25
Final Results.....	28
Further Improvements.....	32
3 NONLINEAR REINFORCED CONCRETE ELEMENT.....	37
Introduction.....	37
Element Modeling Survey.....	38
Beam Element with Inelastic Hinges.....	40
Beam Element Stiffness.....	49
4 STRUCTURAL ELEMENT RELIABILITY.....	54
Introduction.....	54
Two Dimensional Space Example.....	60
Reinforced Concrete Element Reliability.....	69
5 SYSTEM RELIABILITY.....	74
Introduction.....	74
System Reliability and Optimization.....	75
Methods.....	77
Generation of Failure Modes.....	82
Beta Unzipping Method.....	90

	<u>Page</u>
6 PROCEDURE IMPLEMENTATION.....	97
Introduction.....	97
Augmented Lagrangian Formulation.....	98
Generalized Reduced Gradient.....	108
Reliability.....	114
7 EXAMPLES.....	119
Introduction.....	119
Result Verification.....	120
Debug Frame.....	121
Compared Frame.....	131
Building Frame.....	136
8 CONCLUSIONS AND RECOMMENDATIONS.....	139
Linear Material Behavior.....	139
Nonlinear Material Behavior.....	141
Future Work.....	142
APPENDICES	
A AUGMENTED LAGRANGIAN SUBROUTINES.....	145
B GENERALIZED REDUCED GRADIENT EXAMPLE.....	189
C GENERALIZED REDUCED GRADIENT SUBROUTINES.....	195
REFERENCES.....	230
BIOGRAPHICAL SKETCH.....	236

LIST OF TABLES

<u>Table</u>	<u>Page</u>
7.1. Debug frame (GRG): linear version results.....	124
7.2. Debug frame: Augmented Lagrangian version.....	126
7.3. Debug frame (GRG): yielding stiffness results.....	127
7.4. Debug frame (GRG): secant stiffness results.....	129
7.5. Debug frame: element moments.....	130
7.6. Compared frame: initial steel area reinforcement.....	133
7.7. Compared frame results.....	135
7.8. Building frame results.....	138

LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
1.1. Implicit optimization.....	5
1.2. Element optimization.....	10
1.3. Truss optimization.....	11
1.4. System optimization.....	13
1.5. Geometry optimization.....	14
2.1. Pattern Search.....	27
2.2. Cantilever beam.....	29
2.3. One bay frame.....	31
2.4. Gradient method.....	34
3.1. Element model.....	41
3.2. Material behavior.....	43
3.3. Reinforced concrete section.....	45
3.4. Element deformation diagrams.....	48
3.5. Curvature integration.....	50
3.6. Secant spring stiffness.....	52
4.1. Design safety region.....	61
4.2. Probabilistic functions.....	64
4.3. Safety checks.....	66
4.4. Reliability index.....	68
5.1. System models.....	78
5.2. Failure graph.....	83
5.3. Element displacements definition.....	85
5.4. System failure modes.....	91
5.5. Combinatorial tree.....	96
6.1. Augmented lagrangian function plot.....	104
6.2. Augmented Lagrangian version flowchart.....	106
6.3. Generalized Reduced Gradient version flowchart...	113
6.4. Bilinear elastic-plastic diagram.....	117
7.1. Displacement verification.....	122
7.2. Debug frame.....	123
7.3. Compared frame.....	132
7.4. Building frame.....	137
B.1. Integrated optimization example.....	191

Abstract of the Dissertation Presented to the Graduate
School of the University of Florida in Partial Fulfillment
of the Requirements for the Degree of Doctor of Philosophy

OPTIMIZATION OF REINFORCED
CONCRETE FRAMES USING
INTEGRATED ANALYSIS AND RELIABILITY

By

ALFREDO V. SOEIRO

August 1989

Chairman: Dr. Marc I. Hoit
Cochairman: Dr. Fernando E. Fagundo
Major Department: Civil Engineering

Simultaneous analysis and design were considered in the optimization of reinforced concrete frames. Frame elements had rectangular cross sections with double steel reinforcement. Design variables were the section dimensions, the area of steel reinforcement and the structure global displacements. Equality constraints were the equilibrium equations and inequality constraints were generated by element reliability requirements, code reinforcement ratios and section dimension bounds. Optimization strategies were based on the Augmented Lagrangian formulation and on the Generalized Reduced Gradient method.

Reliability of the frames was considered at the element and system level. An element failure function was defined using moment forces and flexural strength. The random

variables considered were flexural strength of concrete and external loads. System reliability was evaluated at the mechanism level using combinations of the elementary failure mechanisms.

Optimization of the frames considering material nonlinear behavior was also investigated. Inclusion of this property was performed using a one-component model for the reinforced concrete element. Inelastic rotational springs were added to the ends of the linear elastic element. The element matrix was obtained by condensation of element elastic stiffness and secant spring stiffness.

Three frames were researched. Respective results using linear material behavior were discussed. In these three cases the optimal solutions were found. Element reliability constraints were active and system reliability was satisfied. The integrated formulation was validated in the linear behavior range. The nonlinear material behavior results were presented for the smaller frame.

CHAPTER 1

STRUCTURAL OPTIMIZATION

Introduction

Optimization is a state of mind that is always implicitly present in the structural engineering process. From experience engineers learn to recognize good initial dimension ratios so that their preliminary designs demand small changes through the iterative process and that elements are not overdesigned. The motivation behind this attitude is to create a structure that for given purposes is simultaneously useful and economic.

Structural optimization theory tries to rationalize this methodology for several reasons. The main one is to reduce the design time, specially for repetitive projects. It provides a systematized logical design procedure and yields some design improvement over conventional methods. It tries to avoid bias due to engineering intuition and experience. It also increases the possibility of obtaining improved designs and requires a minimal amount of human-machine interaction.

There are, however, some limitations and disadvantages when using design optimization techniques. The first one is the increase in computational time when the number of design variables becomes large. Another disadvantage is that the applicability of the specific analysis program that results from the optimization formulation is generally limited to the particular purpose to which it was developed. A common inconvenience is that conceptual errors and incomplete formulations are frequent. Another drawback is that most optimization algorithms have difficulty in dealing with nonlinear and discontinuous functions and, hence, caution must be exercised when formulating the design problem. Another factor of concern is that the optimization algorithm does not guarantee convergence to the global optimum design, yielding on most occasions local optimum points. These facts lead to the conclusion that optimization results may often be misleading and, therefore, should always be examined.

Therefore, some authors suggest that the word "optimization" in structural design should be replaced by "design improvement" as a better expression to materialize the root and outcome of this structural design activity (1). Nevertheless, there is an increasing recognition that it is a convenient and valuable tool to improve structural designs has been increasing among the designers community.

Historical Background

Throughout time there have been various attempts to address structural optimization. The earliest ideas of optimum design can be found in Galileo's works concerning the bending strength of beams. Other eminent scientists like Bernouilli, Lagrange, Young, worked on structural optimum design based on applied mechanics concepts (2). These pioneering attempts were based on a close relation to the thoughts and accomplishments of structural mechanics. They started with hypotheses of stress distribution in flexural elements and ended with material fatigue laws.

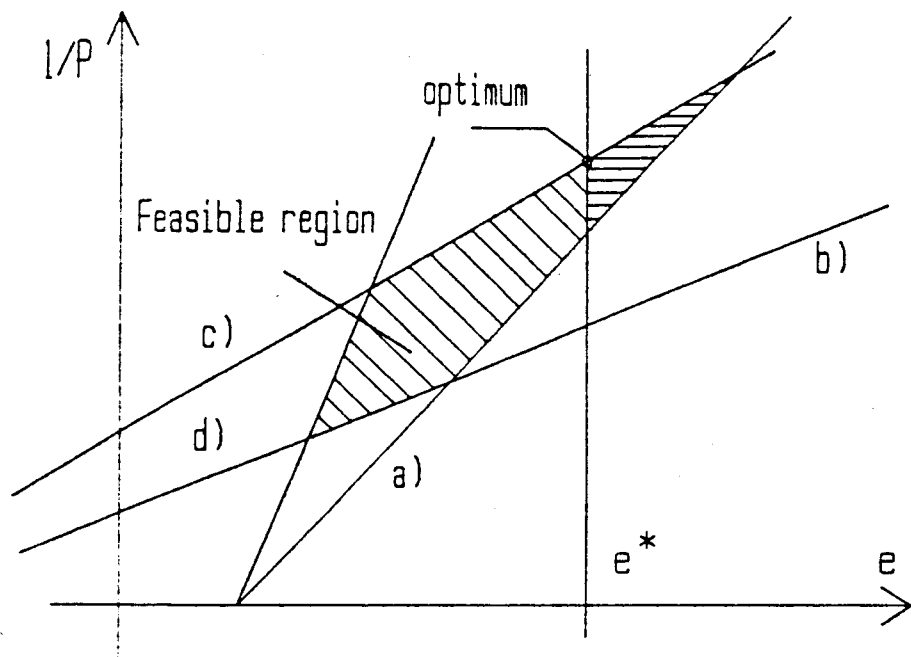
The accepted first work in structural optimization discusses layout theory, or structural topology. The paper focused on the grouping of truss bars that creates the minimum weight structure for a given set of loads and materials. The author of this primary achievement was Maxwell, in 1854, and Michell developed and publicized these concepts in 1904 (3). The practical application of these theorems was never accomplished since significant constraints were not included in the original works.

Some procedures widely used by structural designers are nothing more than techniques of structural optimization. A well known example is the so-called Magnel's diagram (4). It is used to find the optimal eccentricity of the cable that leads to the smallest prestressing force without exceeding the limits imposed on the stresses in prestressed

concrete beams with excess capacity. This is a typical maximization problem in a linear design space, where the design variables are the eccentricity and the inverse of the cable prestressing force. The objective function is the value of the inverse of the cable prestressing force, and is to be maximized. The constraints represent the allowable stresses in tension and compression at the top and bottom of the cross-section. The problem is solved using a graphic representation of the problem, as shown in Figure 1.1, but could be solved numerically using the Simplex method.

Numerical optimization methods and techniques have been widely researched and used in the operations research area, commonly known as Mathematical Programming. The practical application of these theories has been carried out in several areas for some decades like management, economic analysis, warfare, and industrial production. Lucien Schmit was the first to use nonlinear programming techniques in structural engineering design (5). The main purpose of structural optimization methods was to supply an automated tool to help the designer distribute scanty resources. Presently, anyone who wants to consider optimum structural design must become familiar with recent synthesis approaches as well as with accepted analysis procedures.

Magnel's Diagram Optimum Pair P-e



P - Initial prestressing force;
 e - Eccentricity of cable;
 e^* - maximum cable eccentricity;
 a).b) - minimum $1/P$;
 c).d) - maximum $1/P$.

Figure 1.1 Implicit optimization.

Methods

In the last twenty years researchers have made considerable advances in developing techniques of optimum design. Research and exploration of these methods were mainly developed in the aeronautical and mechanical industries, where the need for more economical and efficient final products was extremely important. More recently, with the availability of increasing computer capabilities, civil engineering researchers and designers have increased their participation in structural optimization following the lines defined by the other engineering disciplines. Optimization methods are, nevertheless, common to these different engineering design areas and are mainly divided in two groups. These are commonly known by the names Optimality Criteria and Mathematical Programming (6). Another area in structural optimization researched by a few scientists is based on duality theory concepts, and is an attempt to unify the two basic methodologies (7).

Optimality Criteria methods are based on an iterative approach where the conditions for an optimum solution are previously defined. The concept can be used as the basis for the selection of a structure with minimum volume. This methodology derives from the extreme principles of structural mechanics and has been limited to simple structural forms and loading conditions. The formulation can be mathematically expressed as follows:

$$\underline{x}^{k+1} = \varphi (\underline{x}^k, \underline{u}^{k+1})$$

where \underline{x} is the vector of design variables, \underline{u}^{k+1} is an estimative of lagrangian multipliers and φ is an adequate recurrence relation. Estimation of the lagrangian multipliers is made using the active constraints, those inequality or equality constraints with value close to zero. Recurrence relation φ and lagrangian multipliers represent the necessary conditions for optimality known as Kuhn-Tucker conditions.

On the other hand, the Mathematical Programming approach establishes an iterative method that updates the search direction. It seeks the maximum or minimum of multivariable function subject to limitations expressed by constraint functions. The iterative procedure may be defined as follows:

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

where α^k is the step size and \underline{d}^k is the search direction. The search direction is obtained through an analysis of the optimization problem and the step size depends on the one-dimensional search along that direction. Methods of the second class may be divided in two areas. These areas are transformation methods, like penalty functions, barrier functions and method of multipliers, and primal methods, such as sequential linear and quadratic programming,

gradient projection method, generalized reduced gradient and method of feasible directions.

Typical Applications

In structural optimal design applications there are several types of problems. They address different targets in structural design such as the best configuration for a truss or the cross sections of a prestressed concrete beam. There are four main properties of any structure that may be focused by structural optimization. These are mechanical or physical properties of the material, topology of the structure, geometric layout of the structure and cross-sectional dimensions. Main types of applications are optimization of elements, truss bars, flexural systems, continuum systems, geometry and topology (8).

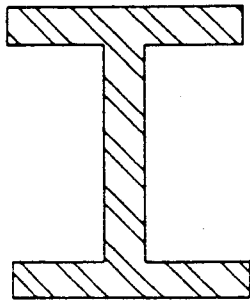
In the case dealing with element optimization, the search is done with a reduced number of variables and the use of code provisions transformed adequately to the optimization formulation. Element forces are found, element cross section is optimized, updated element forces are computed and the process is repeated until there is convergence. For instance, the optimal design of steel wide flange sections may have as design variables the width and thickness of the web and flanges. Constraints may be obtained in an explicit form, as the evaluation of the objective and constraint functions does not require matrix

structural analysis. The minimization technique may be chosen as any one of the available direct search methods (9). Examples of design variables in element optimization are presented in Figure 1.2.

Optimization of truss bar sections has been thoroughly studied due to the simplicity of truss structural optimization problems. There is a decline of interest since they are now rarely used in present structural engineering. Each bar is represented by one variable and the global stiffness matrix terms are linear functions of these variables. Of the various improvement techniques one is based on variable linking, consequently reducing the size of the problem. Another technique to decrease the size is based on constraint deletion, where inactive constraints are temporarily kept out of the optimization process. There are various formulations for the analysis model based on plastic analysis, force or displacement method (10). An example of the formulation used for truss optimization is presented in Figure 1.3.

The problem of system optimization is commonly addressed using design sensitivity analysis and explicit approximations of constraint functions. The intent is to improve the performance of the chosen algorithm. Design sensitivity analysis is the calculation of the analytical derivatives of the objective and constraint functions with respect to the design variables. This information about the change in the value of a constraint related to the changes

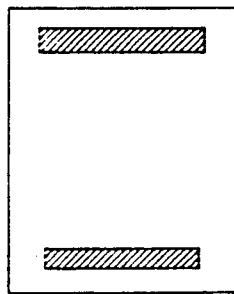
STEEL SECTION



DESIGN VARIABLES

Flange width
Flange thickness
Web height
Web thickness

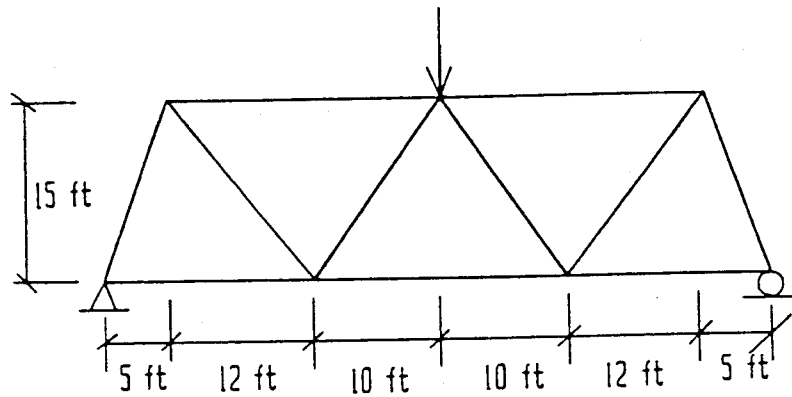
CONCRETE SECTION



DESIGN VARIABLES

Width
Height
Top reinforcement
Bottom reinforcement

Figure 1.2. Element optimization.



Minimize $\sum L_i A_i$
 subject to

$$F_i < F_c$$

$$F_i < F_t$$

where

L_i - length of truss bar i

A_i - area of truss bar i

F_i - stress in truss bar i

F_c - allowable compressive stress

F_t - allowable tensile stress

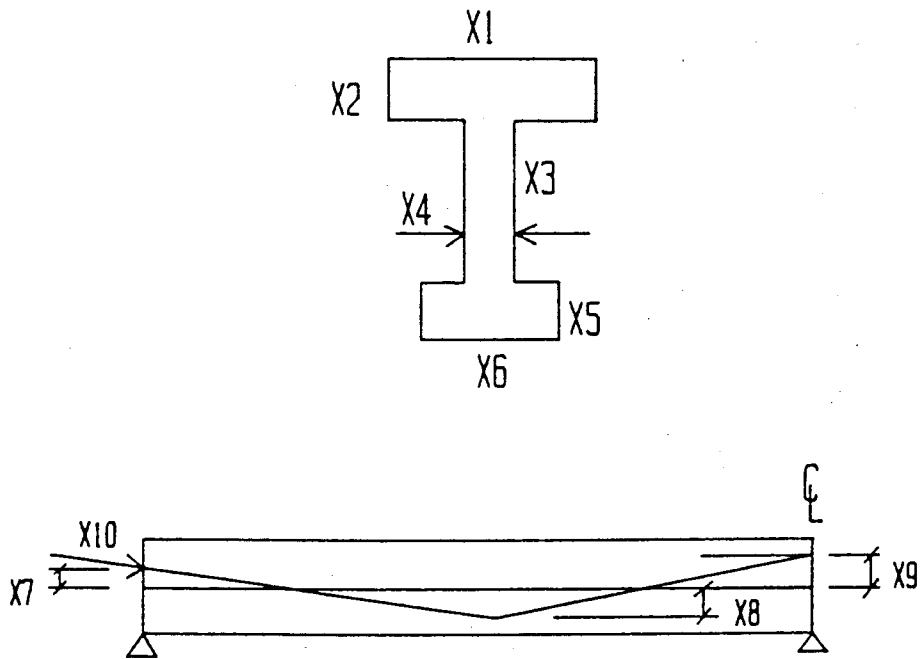
Figure 1.3. Truss optimization.

in the design variables, contributes to the reduction of the exact analyses required during the optimization process. Explicit approximations of the constraint functions using a first order Taylor series expansion are widely used in Optimality Criteria and Mathematical Programming methods. In large and continuum systems some other techniques are used. For example, the sequential optimization of substructures or decomposition using model coordination techniques are used to improve the performance (11). An example of a type of system optimization is illustrated in Figure 1.4.

Geometric and topologic optimization creates geometric design variables that are, for instance, the coordinates of nodes in a finite element mesh or the pier location for a continuous bridge. In certain cases where the areas of the elements have zero as lower bounds, the unnecessary elements can be eliminated by the optimization algorithm. Sometimes the concept of separate design spaces, one for joint coordinates and the other for cross sectional element sizes, is used when trying to reduce the size of the design space considered at any stage (12). An example of optimal configuration is presented in Figure 1.5.

In large optimization problems it is usual to use multilevel optimization techniques where the structural designer has to coordinate and optimize at several levels of the design process. This technique is also useful when the main goal is to find the optimum geometry besides optimizing

Optimization of a Two span prestressed beam



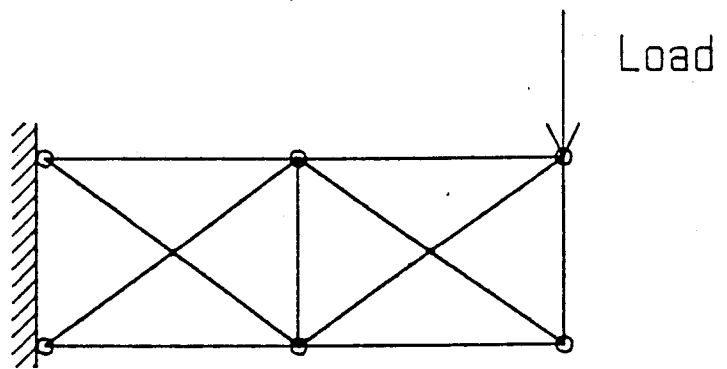
$X1$ to $X6$ - Section geometry

$X7$ to $X9$ - Eccentricities of draped cable

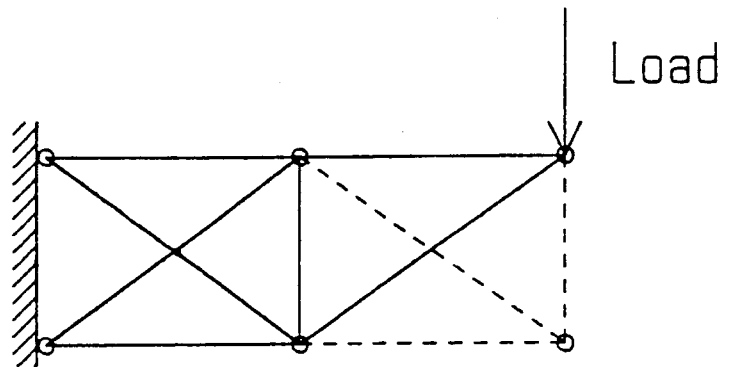
$X10$ - Prestressing force

Figure 1.4. System optimization.

OPTIMIZATION
OF
TRUSS GEOMETRY



Initial Configuration



Optimal Configuration

Figure 1.5. Geometry optimization.

the structural elements. Design variables that control the geometry are often handled better when considered separately from the set of sizing variables (13).

Study Objectives

The main objectives of the present work are to combine adequately optimization and reliability concepts and to test the performance of the integrated approach to reinforced concrete frames. Reliability requirements are imposed at the element and the system level. At element level a maximum probability of failure is imposed for each element and at the system level a minimum reliability index is imposed for the failure mechanisms.

The material behavior of the reinforced concrete elements is separated in two phases. The first considers linear material behavior and the second includes the concrete and steel nonlinear behavior.

Structural frame optimization problems have usually been formulated based on the cycling between two distinct phases, analysis and optimal design. This methodology is the classical approach in structural optimization. The first phase consists in an initial sizing or structure definition. In the second phase, a structural analysis is performed and in the third phase, the structure is resized or redefined using Mathematical Programming or Optimality

Criteria methods. The cycling between phases two and three is interrupted when the termination criteria are met.

The research option summarized here combines phases two and three into one only stage. This is accomplished by the addition of the global displacements to the set of design variables. This addition implies that the equilibrium equations, solved explicitly in the cycling approach, are added to the set of constraints as equality constraints. These new equality constraints will be solved iteratively while in the cycling approach the solution is obtained using a Gauss type decomposition. The main objective behind the adoption of this strategy was to experiment this formulation where the variables related with element stiffness definition and the displacement variables are in the same design space. For that reason the simultaneous optimization and iterative solution of equilibrium equations could be more efficient than the classical nested approach.

The application of this formulation was initially performed with elastic linear frames subjected to static loading. The constraints consisted of limiting the global displacements and the element stresses, besides the additional set of equalities representing the equilibrium constraints. The optimization method used consisted of unconstrained minimization of an augmented lagrangian function of the initial objective function and the equality and inequality constraints (14).

Summary

Results obtained with the integrated approach were encouraging and proved that the method was acceptable for elastic design purposes with displacement and stress constraints. Despite the fact that optimum values were obtained there was however an increase in the size of the problem. This modification of the problem size was due to the fact that the number of variables and the set of constraints augmented.

The final type of optimization problem considered in this work was the minimization of the total cost of a reinforced concrete plane frame submitted to static loading considering the actual stress-strain diagram for concrete and the elastic-plastic behavior of the reinforcing steel. A typical element had a constant rectangular section and doubly reinforced with equal amount of flexural steel on both sides. Width and height of the cross sections had prescribed lower bounds, representing practical requirements and an adequate ratio between the height and the width. The amount of steel was limited by lower and upper bounds dictated by the minimum and maximum reinforcing steel ratios requested by the Building Code Requirements for Reinforced Concrete, commonly known as ACI 318-83.

Inequality constraints considered included maximum values for the global displacements and a minimum reliability index for the element flexural failure function.

Displacements allowed were based on serviceability requirements like cracking and relative story drift. The reliability indices were based on usual values of probability of failure used in design codes. Only the flexural behavior of the frames was analyzed since it is the most important for usual structures and the members were modeled as beam elements.

Inelastic behavior of the structure due to the material nonlinearities imposes a change of the global stiffness terms independently of those dictated by the alterations of the dimensions during the optimization search. For that reason, the reinforced concrete element was modeled as a linear elastic beam with nonlinear rotational springs at each end. Rotational spring stiffness was considered infinite when the moment was below the yielding moment. Above that value the element stiffness was inverted to its flexibility and the inverse of the secant spring stiffness value was added to the corresponding diagonal terms. Spring stiffness was calculated using the secant value of the bilinear moment-rotation diagram corresponding to the current global rotation. Values of the yielding and ultimate moments were obtained by integrating the actual stress-strain diagram for the compressive force in the concrete. The corresponding rotation at a hinge was calculated by integrating the curvature diagram along the element.

Element reliability was evaluated using a Level 2 method, i.e., an approximation to the evaluation of the exact probability of failure. The statistical variables considered were those assumed to have greater influence on the final result. These were the compressive strength of concrete and the external loads, assumed as normal distributed variables. The corresponding reliability index was calculated for constraint evaluation using the ultimate moment obtained from the integration of the respective strain diagram.

Optimization techniques tested were based on the Augmented Lagrangian and the Generalized Reduced Gradient methods. The optimization problem was run, and after termination, the structure probability of failure was compared with the assigned value. If the result was not satisfactory, the process was restarted with updated values of the element reliability indices for the members involved in the most probable collapse mechanism.

Evaluation of the system reliability was divided in two phases. First phase consisted of the identification of the elementary collapse mechanisms. In the second phase these elementary mechanisms were linearly combined to generate all significant mechanisms. System reliability was calculated considering the frame as a series system where each element is one of these mechanisms with higher probability of failure.

Generation of the fundamental collapse mechanisms was made using Watwood's method (15). The automatic procedure consisted of using the geometric configuration of the frame and external loading to find all the one degree of freedom failure mechanisms. The reliabilities of these mechanisms was calculated using the corresponding failure functions

System reliability was evaluated using the Beta unzipping method (16). The elementary mechanisms were linearly combined to obtain other failure mechanisms. The corresponding failure functions were created and the associated reliability indices calculated. In each set of combinations only those in the closeness of the minimum were considered for the next combination (17).

CHAPTER 2

INTEGRATED OPTIMIZATION OF LINEAR FRAMES

Original Research

Integrated formulations for structural optimization problems has received little attention in the published literature. The works of L. Schmit and R. L. Fox are considered the pioneering work as applied to integrated structural optimization (18). The concept of this structural synthesis problem is to combine the design variables with the behavioral variables.

The immediate consequences of this concept are that the problem has a larger number of design variables and the traditional nested analysis-optimization process is avoided. This approach has not been popular since past performance was not comparable to the iterative techniques based on Optimality Criteria and Mathematical Programming concepts. In the integrated formulation the equilibrium constraints generate a large additional number of equality constraints.

Several researchers have recently adopted the integrated approach with encouraging results. These recent attempts have been motivated by new solution procedures

considered more adequate for this type of formulation and by computer hardware development. An example is the optimization of elements with stiffness and strength properties proportional to the transverse size of the elements with linearization of the displacement constraints (19). Another algorithm uses the incremental load approach and conjugate gradient methods to optimize a structure subjected to nonlinear collapse constraints (20). In this case the stiffness matrix is approximated using the element-by-element technique (21). A more recent work uses a new solution technique based on Geometric Programming theory (22). In this formulation the equilibrium constraints are the sum of geometric terms that are function of the design variables.

This chapter describes research that was conducted to study the integrated analysis approach for portal frames with linear behavior and static loading (23-26). The initial phase addressed only constraints on the displacements. Stress constraints were added on a second phase. Throughout this part of the study the frame elements had continuous prismatic rectangular cross section.

Augmented Lagrangian Function

The optimization technique of cycling unconstrained minimizations of a penalty function, based on an pseudo-objective augmented lagrangian function, was chosen as the

solution scheme (27). The design variables were the areas and inertias of each element and the global displacements. Since it is a planar frame there are three degrees of freedom for each joint in the structure.

The merit function used was the volume of the structure. In frames made with one material, volume is generally considered to be proportional to the structure cost. This value was calculated as the sum, for all elements, of the product of the element area times the respective length. The set of inequality constraints was generated by the structure physical behavior and material properties. Limits were imposed on the global displacements and, in the final stage, the element stresses were also bounded.

The compatibility and equilibrium requirements were guaranteed by the additional group of equality constraints. This set was given by the product of the stiffness matrix and the vector of global displacements from which the vector of external global loads was subtracted.

A brief description of the problem variables and respective formulation for a typical planar frame is the following:

Structural parameters

- n structural elements;
- m number of global degrees of freedom;

- \underline{R} vector of static external loads;
- \underline{D} vector of bounds of \underline{m} ;

Design variables

$x_k, k=1,3,\dots,2n-1$ --- area of element $(k+1)/2$;

$x_j, j=2,4,\dots,2n$ --- inertia of element $j/2$;

$x_i, i=2n+1,2n+2,\dots,2n+m$ --- global displacements;

Objective Function

$$f(\underline{x}) = \sum l_p x_k, \quad p=1,n$$

where

l_p - length of element p ;

Equality Constraints

$$\underline{H}(\underline{x}) = \underline{K} \underline{x}^* - \underline{R}$$

where

\underline{K} - global stiffness matrix;

\underline{x}^* - displacement vector;

Inequality Constraints

$$\underline{G}(\underline{x}) = \underline{x}^* - \underline{D} \leq 0$$

Augmented Lagrangian Function

$$L(\underline{x}, \underline{u}, \underline{v}) = f(\underline{x}) + \underline{u} \underline{H} + P \underline{H} \underline{H} + \underline{v} \underline{G}' + P \underline{G}' \underline{G}'$$

where

$\underline{u}, \underline{v}$ - lagrangian multipliers;

P - penalty factor;

\underline{G}' - maximum of $\{\underline{G}, -\underline{v}/2P\}$.

The optimization procedure consists of several cycles of unconstrained minimization of the pseudo-objective function. The values of the lagrangian multipliers are kept constant during each cycle of the unconstrained minimization. At the end of an unconstrained minimization cycle, the multipliers are updated using an appropriate rule (12). The procedure is repeated for successive cycles until there is no significant change of the objective function. At this point the primal and dual optima have been found and the algorithm stops.

Unconstrained Minimization Techniques

Initially the technique used for the unconstrained minimization of the augmented lagrangian function was a zero order method referred to as the Hooke and Jeeves method or Pattern Search. The classification as a zero order method means that it does not utilize any information about the form or shape of the function. After the phase when stress constraints were added, a first order method, Steepest Descent, was tested as an improvement in the algorithm's

performance (27). The technique is based on the gradient of the function that indicates the direction with the highest slope at a given point. Second order methods were determined inappropriate because the pseudo inequality constraints, g' , have discontinuous second derivatives.

Hooke and Jeeves method is an iterative procedure where each step may involve two kinds of moves. The first type of moves explores the local configuration of the pseudo-objective function along the directions of the design variables. The investigation is done within a prescribed step size from the current temporary design point. Each variable is investigated one at a time. The value of the step size is increased or decreased with success or failure in the exploration. This search along the coordinate directions will eventually lead to a smaller value of the pseudo-objective function. Otherwise the optimum has been reached and the exploration stops.

Once all variables have been searched, a pattern move is attempted. The pattern direction is defined by the starting and final points of the variable search and a move is made along that direction. The process of exploration and pattern moves is repeated until there is no significant improvement of the pseudo-objective function. A graphic example is presented in Figure 2.1. The initial point of the variable search, 1, and the final point of that cycle, 4, define a pattern direction that yields a better design point, 5.

HOOKE and JEEVES

1 - Initial Point 4/5 - Pattern Move
6 - Final Point

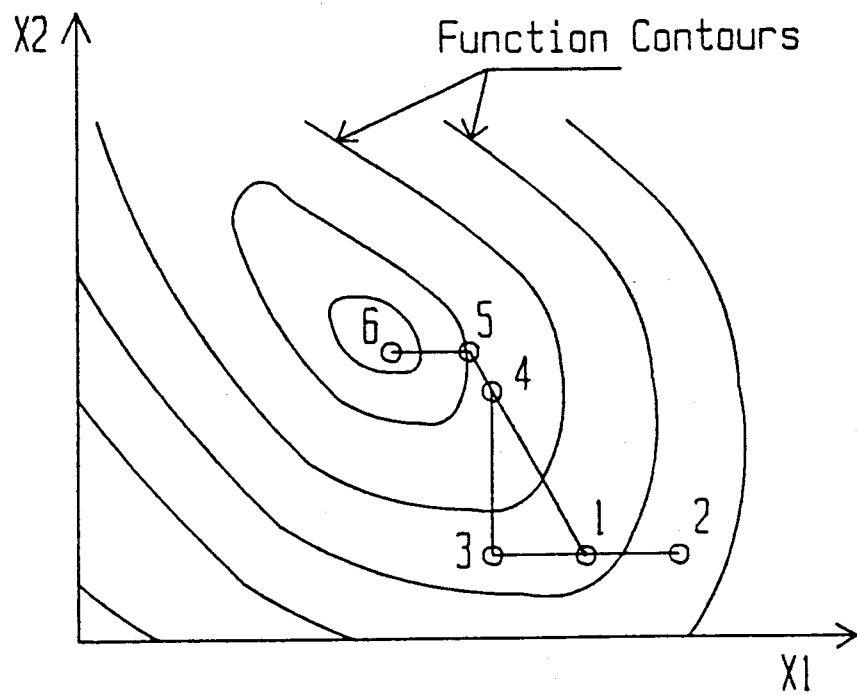


Figure 2.1. Pattern Search.

A computer program was written in accordance with the previous statements and discussions. The structure of the program was conceived by taking into account future inclusions of other types of constraints, changes in the minimization techniques, element replacements and extension to nonlinear and dynamic problems. Hence the program was divided into separate subprograms for the independent tasks (26).

Final Results

The performance and accuracy of the formulation described above was evaluated. Test examples for that purpose were structures with an explicit optimal configuration or simple frames. In the isostatic examples the optimal explicit solutions could be obtained and compared to the computer results. For the other structures, several runs were made with different initial design points and the optimal configuration was determined.

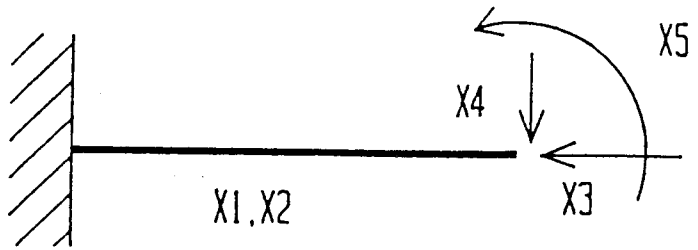
Minimum values were imposed for the dimensions of the cross sections, represented by lower bounds of the areas and moments of inertia. The optimization results show the final values of the displacement variables as the exact solutions for the equilibrium equations. The final area and moment of inertia are also the expected optimal values. Results of a cantilever beam are presented in Figure 2.2.

A computer program was written in accordance with the previous statements and discussions. The structure of the program was conceived by taking into account future inclusions of other types of constraints, changes in the minimization techniques, element replacements and extension to nonlinear and dynamic problems. Hence the program was divided into separate subprograms for the independent tasks (26).

Final Results

The performance and accuracy of the formulation described above was evaluated. Test examples for that purpose were structures with an explicit optimal configuration or simple frames. In the isostatic examples the optimal explicit solutions could be obtained and compared to the computer results. For the other structures, several runs were made with different initial design points and the optimal configuration was determined.

Minimum values were imposed for the dimensions of the cross sections, represented by lower bounds of the areas and moments of inertia. The optimization results show the final values of the displacement variables as the exact solutions for the equilibrium equations. The final area and moment of inertia are also the expected optimal values. Results of a cantilever beam are presented in Figure 2.2.



X1 - area of beam
 X2 - inertia of beam
 X3 - horizontal tip displacement
 X4 - vertical tip displacement
 X5 - tip rotation

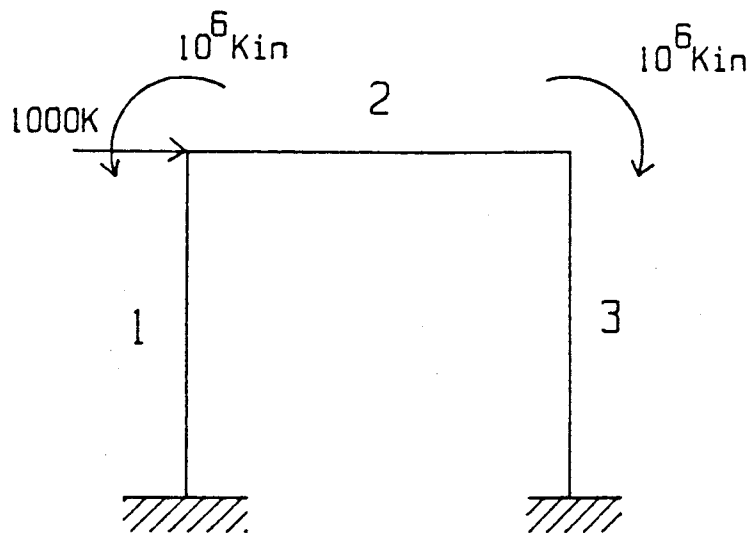
VARIABLE	INITIAL	FINAL
X1 (in ²)	1.0	6.65
X2 (in ⁴)	1.0	78625
X3 (in)	0.4	0.500
X4 (in)	0.4	0.353
X5 (rad)	0.4	0.006

Figure 2.2. Cantilever beam.

Penalty factors used in these runs were of an order of magnitude greater than that of the objective function and constraints. They were kept constant during each optimization cycle. Scaling was also mandatory since the various terms of augmented lagrangian function have different orders of magnitude. The adopted scaling method consisted of using the inverse of the initial value of the expressions concerned. Initial guesses for the design variables were also important for the algorithm performance. The closer these initial designs were to the optimum, the faster the convergence rate.

An updated version of this algorithm was created with the addition of stress constraints. The results of the structures used to test this addition illustrated the adequacy of the method for this type of problems. Again, for the cantilever beam with the explicit solution, the optimum results were obtained. For the frame, the final answer corresponded to what was expected and convergence was obtained. Final mass distribution resembles that previously attained just with displacement constraints. The geometry and related values are presented in Figure 2.3.

A tapered cantilever loaded at the tip was compared with the results obtained using a recursive relation between the dimensions and displacements (12). The two sets of results, those from the reference and those from the program run, are very close. The maximum absolute difference



ELEMENT		INITIAL	FINAL
1	Area (in2)	1.0	25.4
	Inertia (in4)	1.0	120224
2	Area (in2)	1.0	179
	Inertia (in4)	1.0	5912
3	Area (in2)	1.0	35.1
	Inertia (in4)	1.0	17058

Figure 2.3. One bay frame.

between the correspondent section dimensions is less than five percent.

Further Improvements

In subsequent developments, some other improvements were added to the algorithm that used the augmented lagrangian formulation. The first consisted of eliminating from the search those constraints that were inactive. Those constraints whose value did not show a change when the line search was along one of the design variable, were skipped from recalculation. This savings in computational effort allowed a reduction of forty per cent of the total run-time. This feature was discarded when the gradient search method was implemented. With this technique the changes in the design variables were done simultaneously, all constraints were altered and selective recalculation was no longer possible.

Another significant improvement was achieved by starting the solution with feasible displacements. The displacement variables were calculated at the beginning of the program corresponding to the initial loading and frame dimensions. This led to the situation where the equality constraints were exactly satisfied at the start of the iteration procedure. This addition was kept in the version using the gradient search. Work was also done on selecting the initial cross section dimensions. Rules of thumb were

found to expedite calculations to obtain acceptable initial values.

The method of steepest descent makes use of the gradient of the pseudo-objective function. The gradient vector represents the line along which there is the highest variation of the pseudo-objective function at the actual design point. Moving in the direction defined by the negative of the gradient vector is expected to decrease the value of the pseudo-objective function. This direction is called the steepest descent. A graphical representation of the method is displayed in Figure 2.4. Since the explicit formulation of the gradient of the pseudo-objective function was not practical to obtain, the gradient vector was obtained using a finite difference technique. To obtain the minimum point along the gradient direction another design point along that line is found such that it has a higher pseudo-objective function value. Then, the optimum value should lie in this interval and a line search is performed using the golden section method.

The gradient vector was normalized to avoid numerical ill-conditioning. For the same reason, constraints and the design variables were also scaled. Numerical difficulties are predictable if just one of the constraint function, or the objective function, is of different magnitude than the rest of the terms or its rate of change is considerably different from the others. Scaling factors for each constraint were evaluated as the ratio between the gradient

STEEPEST DESCENT

1-Initial Point
4-Final Point

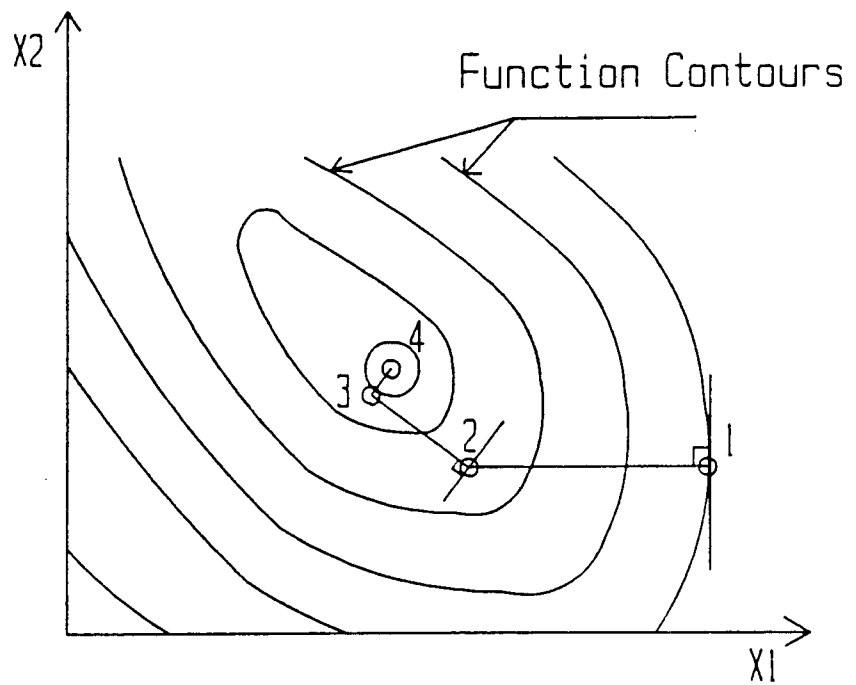


Figure 2.4. Gradient method.

of the objective function and the gradient of that constraint. Scaling of the design variables was also tried. The normalization of the design variables consisted of applying scaling factors that reduced them to a single order of magnitude.

The results obtained with this unconstrained minimization technique were inferior to those using the Hooke and Jeeves method. The apparent reason was the shape of the surface generated by the augmented lagrangian function. Around a relative local optimal point, where the equality constraints are satisfied, the variation of the augmented lagrangian function was very abrupt. Consequently, any line search performed starting at a relative optimal point would invariably return to the same initial point.

When using a set of design variables that was not a relative local optimum, the gradient search would still not converge. The reason for this lack of convergence was the numerical error created by the steep slope of the function. This fact could not be avoided despite the several combinations of the constraint and variable scalings aimed at smoothing the shape of the augmented lagrangian function.

Another phase of research consisted in using a mixed method for the search. In a first phase, Hooke and Jeeves was used to obtain a better second point than the starting design point. This second point was then used to apply the gradient search. The procedure was repeated with the

consequent updates of the lagrangian multipliers. This mixed method did not present any improvement over the Hooke and Jeeves method. The important conclusion from the results of this mixed strategy was that convergence could only be obtained when enough iterations of the Hooke and Jeeves phase were completed. Consequently, the adopted unconstrained minimization method for the optimization of the augmented lagrangian function in the linear static formulation was the Hooke and Jeeves method.

CHAPTER 3

NONLINEAR REINFORCED CONCRETE ELEMENT

Introduction

Reinforced concrete elements are made of two different materials, concrete and steel. Concrete is the massive component, has a high compressive strength and fails easily when submitted to tension. Steel is embedded whenever tensile strength is required. For that reason the additional steel bars are commonly designated as reinforcing steel.

Adequate combination of these two materials originates a symbiotic composite material that has been widely used (28). These elements are designed with bending, compression and torsion requirements for code and safety compliance. In some cases tension is also allowed.

Concrete and steel have nonlinear stress-strain diagrams. Consequently, when material nonlinearities are included, modeling of the behavior of any composite element is very difficult (29-30). When loads produce a tensile stress greater than the maximum allowable value for the concrete cracking results. When reinforcing steel stress

reaches the yielding value there is a large strain and section curvature increase. Geometric nonlinearities are then created by extra rotations of flexural elements from the cracking and steel yielding.

A basic assumption in nonlinear analysis of reinforced concrete frames is that the element rotations with relation to the line defined by the nodes, chord rotations, are small and the theory for straight elements may be applied with some adaptations. The most popular analysis techniques are based on incremental loadings of the structure and are known by the initial stiffness and tangent stiffness methods. A technique based on the application of the entire load at a single step is known by the secant stiffness method. This last technique was chosen for the analysis of the structure since it is more adequate to the optimization formulation.

Element Modeling Survey

In the last three decades there have been many attempts to create a simplified beam model of the inelastic reinforced concrete element (31-33). The main objective for this research has been to advance a solution providing precise results within reasonable computational and memory storage limits. The study has a significant importance for the analysis of reinforced concrete structures submitted to dynamic loads (34-35). In these examples the moments at the ends are close to the ultimate allowable values. This

closeness implies that the concrete and steel stresses are in the nonlinear intervals of the stress-strain diagrams. The frame behaves as if inelastic plastic hinges have formed due to concrete cracking and steel yielding.

Initial studies in this area addressed simple structures with moment-rotation relationship conditioned by the moments at the beam extremities. This produced the one-component model with nonlinear rotational springs at the ends. Later, another theory assumed a bilinear moment resistance with two parallel elements, one to simulate yielding and the other to represent strain hardening. Several variations of these two theories have been developed and experimentally tested (36).

Recent improvements in computer software led to sophisticated modeling of reinforced concrete elements using nontraditional finite element techniques. A simple approach to this type of problem is based on the theory of damage mechanics (37). The beam element is modeled as a macroelement divided in models with explicit and accurate behavior. The behavior of the whole structure is then extrapolated from the small elements.

These types of models have been tested thoroughly to ascertain its reliability and accuracy (38). These evaluations, made mostly by comparison of computer program results with experimental test data, provided a great deal of information for further enhancements and refinements. The option for this study had to fall on a element model

that is a compromise between the accuracy required and the cyclic nature of the optimization process (39). Repeated evaluation of the element stiffness due to the changes of the physical properties of the elements is required. For this reason it is highly desirable to choose a model with low computational requirements.

Beam Element with Inelastic Hinges

Given the available solutions for the model of the reinforced concrete element, the one-component model was chosen as shown in Figure 3.1. It is a simple idealization that doesn't increase the total number of elements of the structure. This model has shown to accurately model the nonlinear behavior of reinforced concrete, even for dynamic loadings (40). Some basic assumptions and simplifications were made for the definition of the model. For example, the fact that concrete cracks under tensile loading, causing local nonlinear behavior, was not accounted for. Time dependent properties of the concrete were not considered. Shear effects were not included in this formulation. The loads were considered applied at the nodes and elements with loads in the span can be approximated by a discrete number of elements with nodal loads.

The unique element internal action considered was flexure. Yielding of the reinforcing steel may only take place in the hinges at the element ends. Strain hardening

One-Component Model
Reinforced Concrete Element

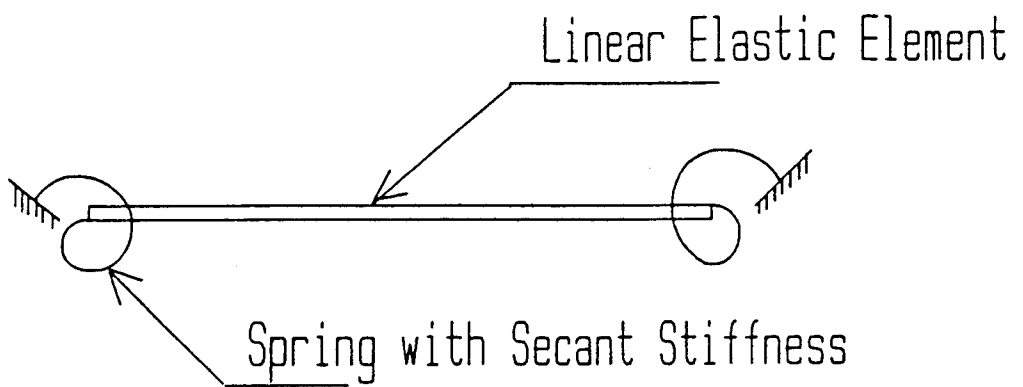
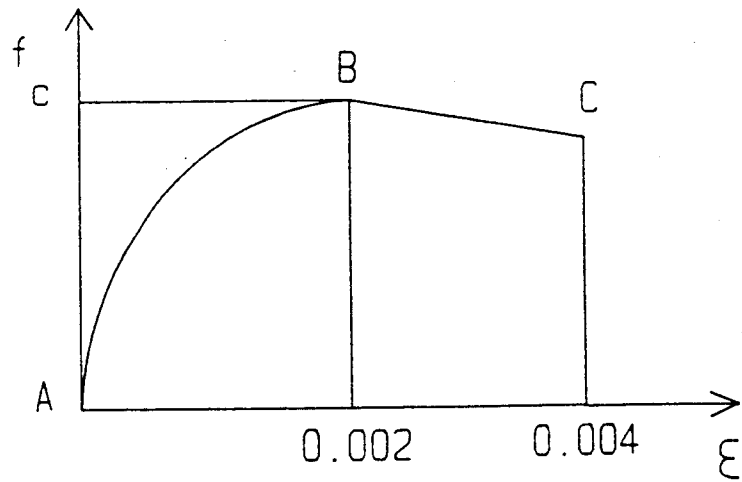


Figure 3.1. Element model.

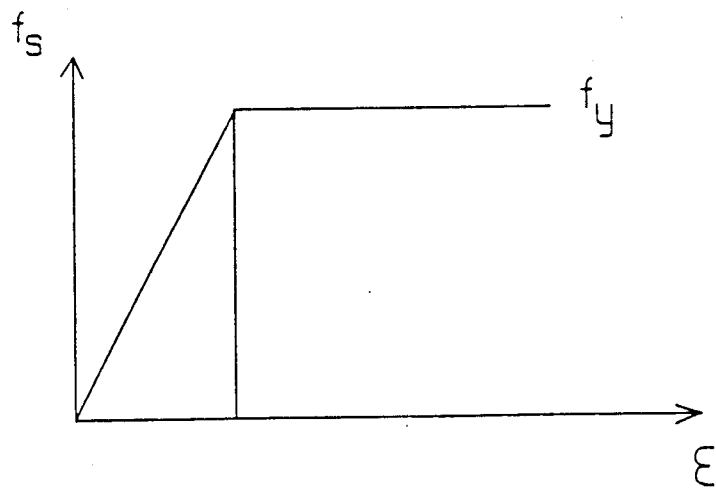
and related altered element stiffness are simulated by the linear element with nonlinear rotational springs at the extremities. Inelastic rotations of reinforced concrete hinges at the element ends are determined as a function of the respective moment-curvature relationship for each element. These curves are redefined every time any element sectional properties changes during the optimization process since the ultimate and yielding moments also change.

A typical moment-curvature diagram for reinforced concrete elements is bilinear. It is obtained assuming material stress-strain curves that are parabolic-linear for the concrete and bilinear for the reinforcing steel as shown in Figure 3.2 (28). The stress in the concrete is designated by f_c and the stress in the steel reinforcement is represented by f_s . The algorithm used to compute the moment corresponding to a certain strain diagram is an iterative Newton based iteration that determines the depth of the neutral axis guaranteeing equilibrium of the internal forces. Then, after determining the internal coupled forces the related moment is computed.

All reinforced concrete elements are doubly reinforced with equal areas of steel on both sides. This assumption is valid for columns and acceptable for beams since in continuous frames there are moments of different sign along the beams. Evaluation of the moments for each reinforced concrete section was based on the exact internal equilibrium equations as follows:



Concrete Stress-Strain Diagram



Steel Stress-Strain Diagram

Figure 3.2. Material behavior.

$$C_C + C_S = T_S$$

where

C_C - compressive force in the concrete and is equal to the area under stress-strain curve corresponding to concrete strain ϵ_C ;

C_S - compressive force in the steel area A_S corresponding to steel strain ϵ_{CS} ;

T_S - tensile force in the steel area A_S corresponding to steel strain ϵ_S ($\epsilon_S \leq \epsilon_Y$).

Typical element moments necessary to define the bilinear moment-curvature diagram were the yielding and ultimate values. These characteristic values were determined considering the corresponding section strain distribution, the stress-strain diagrams for concrete and steel, the location of neutral axis and the moment of the internal forces as shown in Figure 3.3. The compressive force of the concrete is given at any time by

$$C_C = \alpha f_{cm} b kd$$

where

$$\alpha = \int_0^{\epsilon_{ca}} f_c / (f_{cm} \epsilon_{ca}) d\epsilon_c;$$

f_{cm} - maximum flexural concrete stress;

ϵ_{ca} - concrete strain at the top compression fiber;

b - element cross section base;

kd - distance of neutral axis from top compressed fiber.

SECTION CHARACTERISTICS

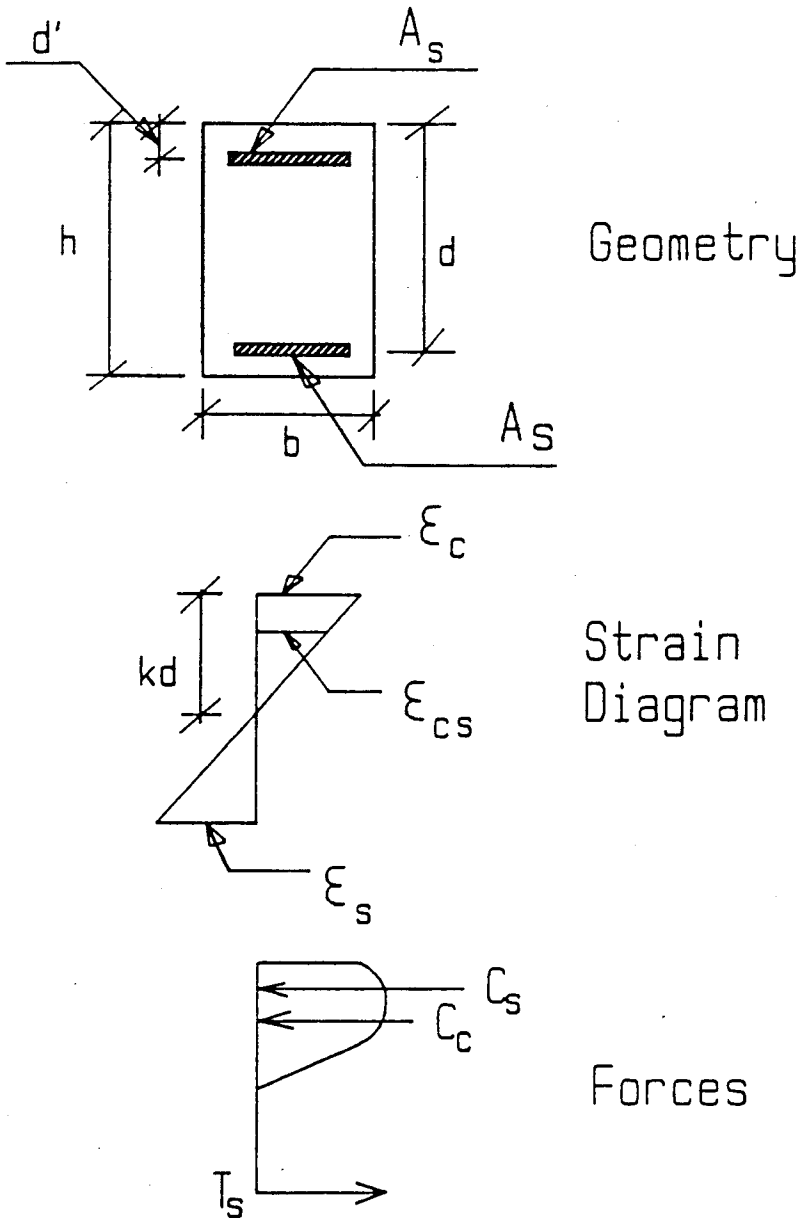


Figure 3.3. Reinforced concrete section.

The force in the compressed steel is given by

$$C_S = A_S f_{CS}$$

where

A_S - steel area;

f_{CS} - stress in compressed steel.

The force in the steel under tension is determined by

$$T_S = A_S f_y$$

where

f_y - yielding steel stress.

For instance, the internal ultimate moment is given by the moments of these three internal forces about the top compressed fiber. For that reason a parameter Ω , that defines the centroid of the concrete compressive stress diagram, is introduced as

$$\Omega = 1 - \int_0^{\epsilon_{ca}} \epsilon_c f_c d\epsilon_c / (\epsilon_{ca} \int_0^{\epsilon_{ca}} f_c d\epsilon_c)$$

These parameters, α and Ω , when the ultimate concrete strain is defined as $\epsilon_c = 0.004$, become

$$\alpha = 2/3 \text{ (region AB)}$$

$$\Omega = 3/8 \text{ (region AB)}$$

$$\alpha = 0.9 \text{ (region BC)}$$

$$\Omega = 0.51851 \text{ (region BC)}$$

where the regions AB and BC are defined in Figure 3.2. The section flexural strength, M_i , may be defined as

$$M_i = C_S d' + C_C \Omega kd - T_S d$$

where

d' - distance of C_S to top compressed fiber;

d - distance of T_S to top compressed fiber.

Element curvatures corresponding to these yielding and ultimate moments are obtained assuming that plane sections remain plane after deformation and there is no strain hardening of the reinforcing steel. These formulas are as follows:

$$\phi_y = (\epsilon_y + \epsilon_{ca})/d$$

$$\phi_u = (\epsilon_{sa} + \epsilon_{cu})/d$$

where

ϕ_y - yielding curvature;

ϵ_y - E_S / f_y ;

E_S - 29×10^6 psi;

f_y - yielding stress of reinforcing steel;

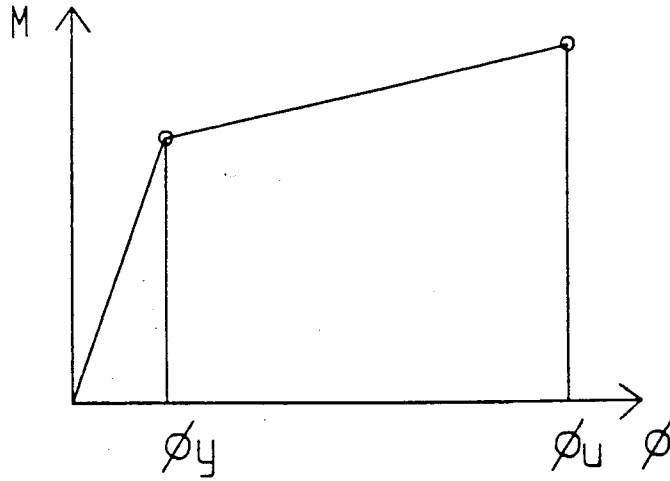
ϵ_{ca} - maximum concrete compressive strain;

ϕ_u - ultimate curvature;

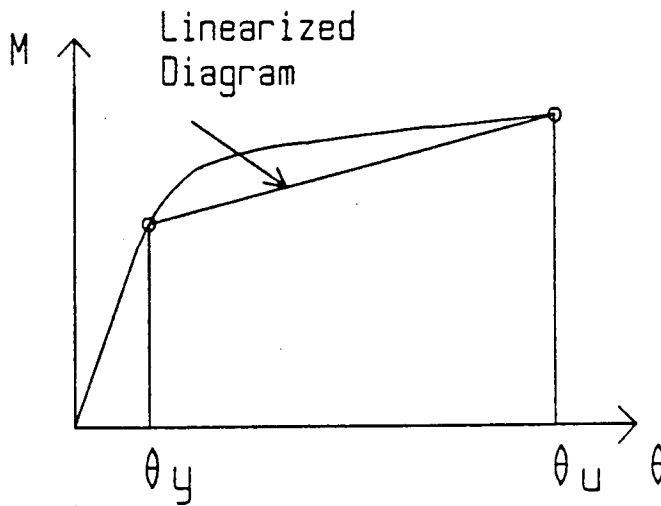
ϵ_{sa} - actual tensile strain of steel;

ϵ_{cu} - ultimate compressive strain of concrete.

These section characteristics define section diagrams as shown on Figure 3.4. The value of the ultimate rotation was given by the integration of the curvature along the element. Two types of curvature diagrams were considered



Moment Curvature Diagram



Moment Rotation Diagram

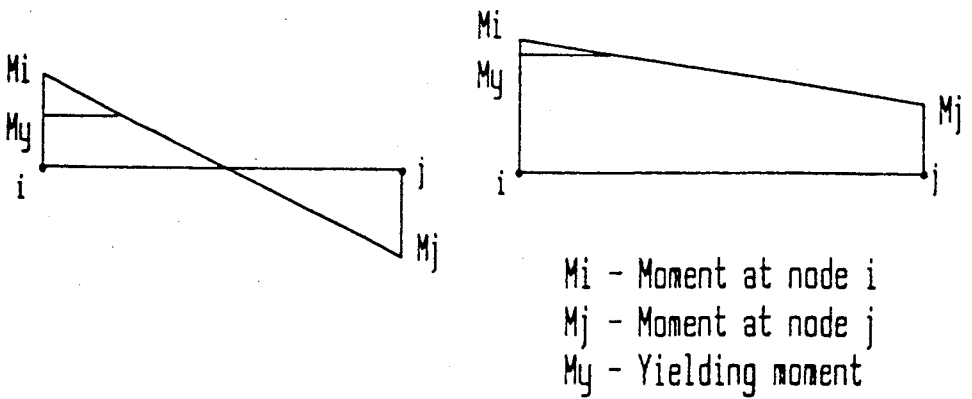
Figure 3.4. Element deformation diagrams.

for integration. The first one was when moments at element ends had the same rotational direction and the second when the rotational directions were opposite. In both cases a simplified method was used to integrate the curvature along the element to find the corresponding rotation since the moments at the other end were kept constant. Yielding rotation for any node of the element was calculated assuming the yielding moment at that node and keeping the other moment unchanged. The same method was applied for the calculation of the ultimate rotation where a modified curvature diagram was used as schematically exemplified in Figure 3.5.

Beam Element Stiffness

The elastic element chosen has a stiffness derived in classical terms. End rotational springs had variable stiffness depending on element moments at the nodes. A large value was assigned to the secant spring stiffness when moments were below the yielding value assured a linear behavior. The secant stiffness value obtained from the moment rotation diagram was used for moment values above yielding. The strain hardening ratio of the linearized moment rotation diagram was computed as the difference between ultimate and yielding moments divided by the difference between the ultimate and yielding rotations. A

MOMENT DIAGRAM



CURVATURE DIAGRAM

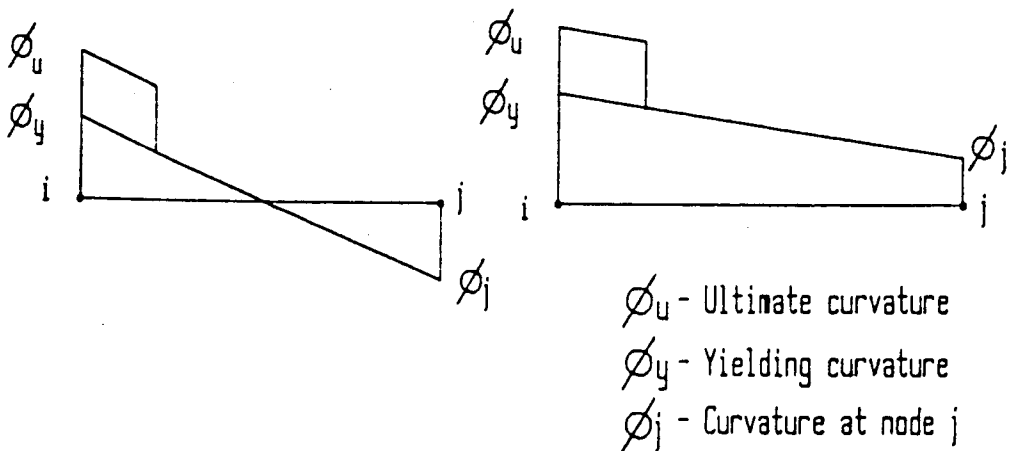


Figure 3.5. Curvature integration.

graphical description of these definitions is presented in Figure 3.6.

The element modified stiffness was derived from the condensation of elastic stiffness matrices of the linear elastic element and the rotational spring elements. To condense the two matrices the first step consisted of inverting the sum of the corresponding flexibility matrices concerning the independent element rotational degrees of freedom. The next step was the expansion of this element stiffness to include the axial displacements, uncoupled from the spring rotations, and the other dependent element degrees of freedom. The main steps of this step are the following:

$$[K_S] = \left[\begin{bmatrix} 1/K_{Si} & 0 \\ 0 & 1/K_{Sj} \end{bmatrix}^{-1} + 3EI/L \begin{bmatrix} 1/3 & -1/6 \\ -1/6 & 1/3 \end{bmatrix} \right]^{-1}$$

$$[a] = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1/L & 1 & 0 & -1/L & 0 \\ 0 & 1/L & 0 & 0 & -1/L & 1 \end{bmatrix}$$

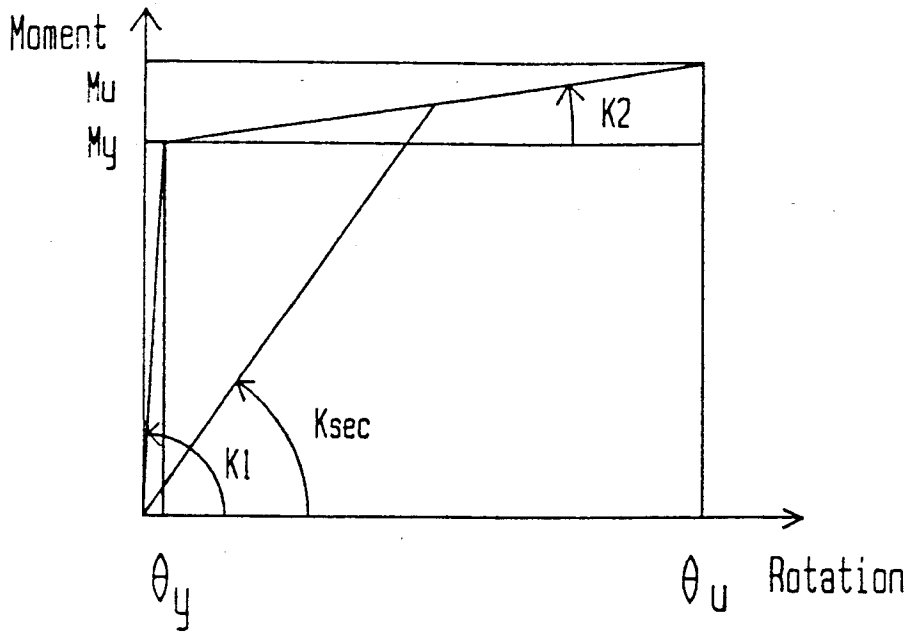
where $[K_{mod}] = [a]^t [K_S^*] [a]$

K_S - secant stiffness matrix with element rotations;

K_S^* - expanded secant stiffness matrix with
uncoupled axial stiffness;

K_{Si} - stiffness of spring at node i;

K_{Sj} - stiffness of spring at node j;



Spring Moment-Rotation Diagram

M_u - Ultimate moment

M_y - Yielding moment

K_1 - $10e30$

K_2 - $(M_u - M_y)/(\theta_u - \theta_y)$

K_{sec} - Spring stiffness for
 $M > M_y$

Figure 3.6. Secant spring stiffness.

E - element modulus of elasticity;
I - element moment of inertia;
L - element length;
a - expansion matrix;
 K_{mod} - modified element matrix.

After evaluating the modified element stiffness matrix it was transformed from the local coordinates to the global coordinates by the use of the corresponding rotation matrix. The values of the terms of this element stiffness matrix were then used to compute the corresponding updated equality constraint values. The process was similar to assembling a structure global matrix using a location matrix relating the element degrees of freedom with the structure global degrees of freedom.

CHAPTER 4

STRUCTURAL ELEMENT RELIABILITY

Introduction

Design and checking of structures in the field of Civil Engineering has been traditionally based on deterministic analysis. Adequate dimensions, material properties and loads are assumed and an analysis is carried out to obtain the required evaluation. Nevertheless, variations of all these parameters and questions related to the structural model may impose a different behavior than expected (41). It must be emphasized that if there were no uncertainties related to the prediction of loads, materials and structure modeling, then the respective safety would be more easily guaranteed.

For these reasons the use of probabilistic principles and methodologies in structural design has been increasing. Design for safety and performance should consider the conflict between safety and risk. The objective of probability concepts and methods is to develop a framework where the effects of these uncertainties are considered. Structural reliability has received the attention of several

researchers and, consequently, it is introduced into almost all recent structural codes worldwide.

It is a relatively young structural science that evolved in the same way as other new areas where theoretical studies dictate the general principles for systematic treatment of problems. There are however practical difficulties in obtaining enough statistical data and handling the sophistication of the probabilistic methods. For these reasons the analytical processes involved in the determination of structural reliability were grouped in different working levels (42). These working levels depend on the problem considered and the desired accuracy for the reliability evaluation. There are three basic levels and the classification increases with the sophistication of the method used and the amount of statistical data that is manipulated.

Level 1 uses a methodology that provides a structural member with an adequate structural reliability by the specification of partial safety factors and characteristic values of design variables. This is the method currently used in structural design codes (43). Level 2 includes all methods that control the probability of failure at certain points on the failure boundary defined by a limit state equation (44). Level 3 groups all techniques that perform a complete and exact analysis of the structure taking into account the joint probability function of all the variables involved (45).

In this chapter, the technique used to analyze the structural reliability of each reinforced concrete beam element is described. Due to the nature of the problem, where optimization and reliability evaluation are performed simultaneously at the element level, a Level 2 method was chosen. Since the concepts of limit state design and probability of failure are intimately connected with structural reliability, a brief description is also included.

Concept of limit state may be described as that state beyond which a structure, or part of it, can no longer fulfill the functions or satisfy the conditions for which it was designed. Namely, the structure is said to reach a limit state when a specific response parameter attains a threshold value. Examples of ultimate limit states are the loss of equilibrium of a part or the whole of the structure considered as a rigid body, failure or excessive plasticity of critical sections due to static actions, transformation of the structure into a mechanism, buckling due to elastic or plastic instability, fatigue, excessive deflections and abundant cracking.

Modern codes divide limit states into two main groups. Ultimate limit states, corresponding to the maximum load-carrying capacity, and serviceability limit states, related to the criteria governing normal use and durability (46). For each of these groups the importance of damage is

different and is represented by the adopted respective probability of failure.

For instance, in reinforced and prestressed concrete, code checks for the ultimate limit states are based on element forces, except in the plastic analysis where the design variables are the loads. In cases where fatigue is involved, stresses are also the control variables. The service limit states are the cracking limit state and the deformation limit state. In this work only the ultimate flexural limit state and the global deformation limit state are addressed since they are the more relevant for the optimization study.

Acceptable risks of failure for any structure are affected by the nature of the structure itself and its expected application. These are dependent on social and local variations. It is common for structural engineers to balance the contradiction between the economy and safety of the structure. This particular aspect is the main reason why it is so appealing to combine reliability and optimization in structural design.

Probabilities of failure used in limit state designs vary with the risk of loss of human lives, the number of lives affected and economic consequences. In ultimate limit states the range of probability of failure adopted is between 10^{-4} and 10^{-7} over a 50 year expected design life. In serviceability limit states the probability of failure varies between 10^{-1} and 10^{-3} .

A criterion proposed is as follows (41):

$$p_f = 10^{-5} U T / L$$

where

- U - 0.005 Places of public assembly, dams;
- 0.05 Domestic, office, industry, travel;
- 0.5 Bridges;
- 5 Towers, masts, offshore structures;
- T - life period of the structure(years);
- L - number of people involved.

These values must be interpreted carefully. For example, the value of 10^{-3} means theoretically that, on the average, out of 1000 nominally identical buildings, one will crack or deform excessively. It is evident that in civil engineering 1000 identical buildings rarely occur, even neglecting the fact that a statistically significant number require samples at least 10 to 20 times larger.

Moreover, the determination of these low probabilities requires extrapolations of statistical properties that are experimentally known only around the mean values of the random quantities. For these reasons, the probabilities of failure in civil engineering have no real statistical significance and they must be considered not as deterministic quantities but just as conventional comparative values.

In consequence of the above considerations, the differences between the methods used in each of the three levels are rather operational than conceptual. There are no rigid boundaries between them. They are used in accordance with the required accuracy and the nature of the problem to be studied.

Level 3 methods require a complete analysis of the problem and also the integration of the joint distribution density of the random variables extended over the safety domain. They remain in the field of research and are used to check the validity of approximations, idealizations and simplifications performed in the other two levels.

Level 2 methods use random variables characterized by their known or assumed distribution functions, defined in terms of important parameters as means and variances. This avoids the multidimensional integration of the previous method. These methods may be used by engineers to solve problems of special technical and economical importance. Code committees engaged in drafting and revising standard codes of practice use them to evaluate the partial safety factors. It is possible that computational developments in the near future will allow for such methods to be more commonly used by the practicing engineer. The probabilistic aspect of the problem in the Level 1 methods is represented by characteristic values of the random variables involved. With these characteristic values partial safety factors are derived using Level 2 methods. They are used by most

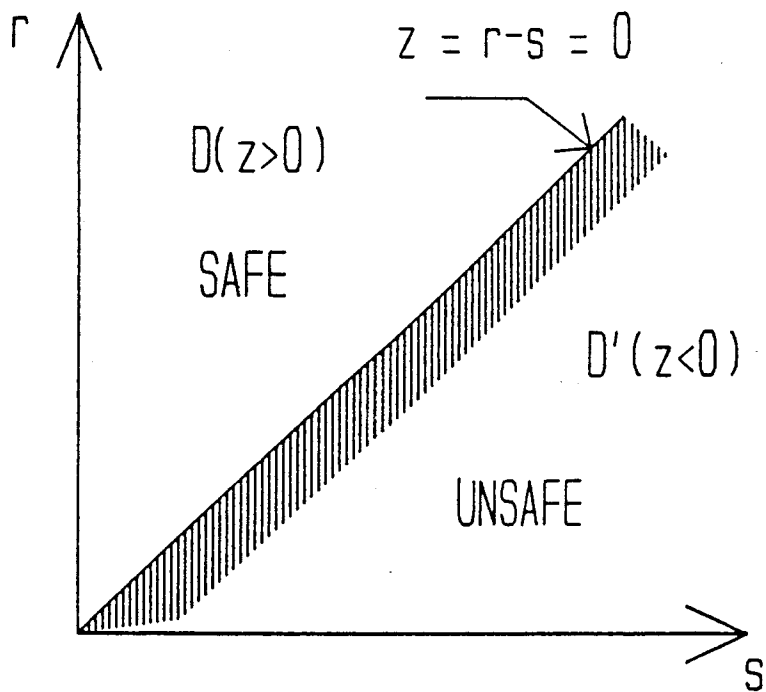
engineers where reliability theory and probabilistic methods are the basis of their code provisions.

These Level 1 methods could be replaced by the Level 2 methods if an agreement was obtained in the following issues: selection of basic random variables for each specific problem, their distribution types and relative statistical parameters; form of the various limit state equations and choice of models; operational reliability levels to be adopted in different design situations.

It must be emphasized that the advantage of Level 1 schemes over Level 2 are their great operational simplicity due to the use of fixed and constant partial safety factors for a given class of design situations. The main disadvantage of Level 1 is the selection of partial safety factors for a given structural class in such a way that the efficiency of the method proposed is satisfactory. It must assure that the deviation of the reliability of a design made on the basis of the adopted coefficients from the desired reliability level laid down in the code is acceptable.

Two Dimensional Space Example

Let R and S be two random variables, where R defines strength and S the load. Then the limit state function z shown in Figure 4.1 is defined as



Safe and Unsafe Design Regions

Figure 4.1. Design safety region.

$$z = r - s$$

where

r - resistance function;

s - load function.

The domain D ($z > 0$) is the safe domain and D' ($z \leq 0$) is the failure domain. The probability of failure, p_f , is the probability that a point (R, S) belongs to D' . Once the statistical distributions of the random variables R and S are known, the numerical solution of the corresponding equation will determine p_f . Assuming that both variables R and S have a Gaussian distribution, and further defining r^m and s^m as the mean values, and σ_R and σ_S as standard deviations of R and S , respectively, the random variable Z will also be normal and its statistical parameters are defined as

$$z^m = r^m - s^m$$

$$\sigma_z = (\sigma_R^2 + \sigma_S^2)^{\frac{1}{2}}$$

where

z^m - mean value function;

σ_z - standard deviation function.

Defining F_z as the cumulative normal distribution function, the probability of failure may be calculated as

$$p_f = P\{Z < 0\} = F_z(0)$$

A graphic representation of these functions is presented in Figure 4.2. Introducing the standardized variable u and the reliability index as

$$u = (z - z^m) / \sigma_z$$

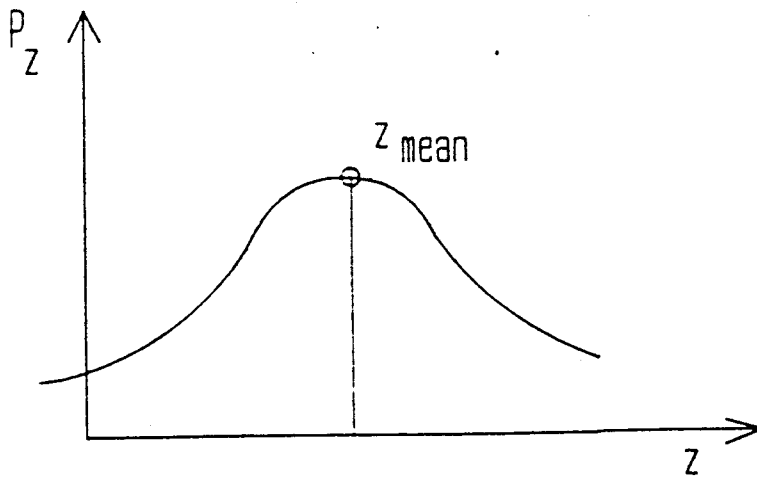
$$\beta = z^m / \sigma_z = (r^m - s^m) / (\sigma_R^2 + \sigma_S^2)^{1/2}$$

then the probability of failure may be expressed as

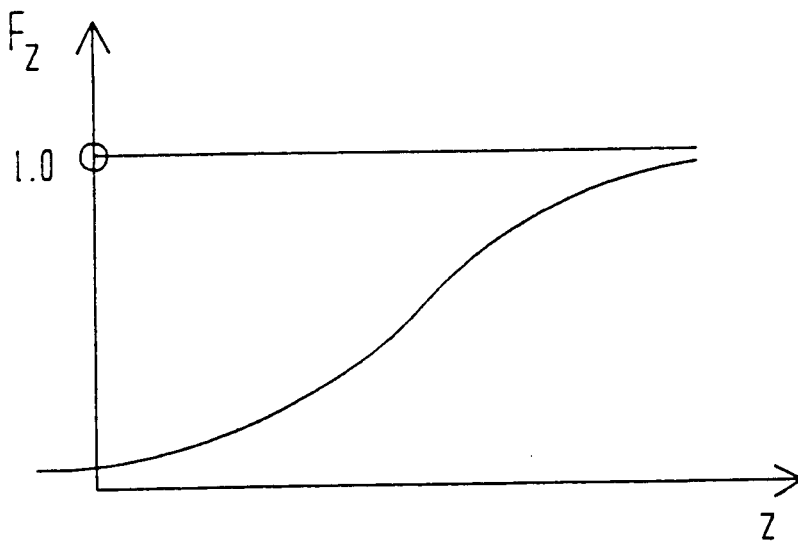
$$p_f = F_u(-z^m / \sigma_z) = F_u(-\beta)$$

An important concept widely used in structural safety when considering random variables is the Central Safety Factor. It relates the mean values and coefficients of variation of R and S to determine a probabilistic safety factor (44). It is a simplistic way of establishing some influence on the design variables of the respective random characteristics.

To consider a more detailed study a Level 2 method is applied in the element reliability evaluation. In this method safety checks are made at a finite number of points of the failure boundary. A graphic representation in a two dimensional space is presented in Figure 4.3. In the case where this check is made at only one point, the parameter to be determined is the minimum distance between the origin of the system of the standardized variables to the boundary of the safety domain.



Probability Density Function



Cumulative Density Function

Figure 4.2. Probabilistic functions.

It is possible to associate this distance with a precise meaning in terms of reliability. A technique derived from this concept is the Lind-Hasofer Minimum Distance method illustrated in Figure 4.3 (47).

Let \underline{X} (X_1, X_2, \dots, X_n) be the vector of the basic random variables of a given structural problem that may be assumed to be statistically uncorrelated, involved in a given structural problem. Let $z = g(x_1, x_2, \dots, x_n) = 0$ be the boundary of the safety domain. The values of \underline{X} belonging to the failure domain will satisfy the inequality

$$z = g(\underline{X}) < 0$$

The method consists in projecting the function z in the space of standardized variables defined as

$$u_i = (x_i - x_i^m) / \sigma x_i$$

Measuring, in this space, the minimum distance β of the transformed surface $g(u_1, u_2, \dots, u_n)$ from the origin of the axes. A design is regarded reliable if $\beta \geq \beta^*$, where β^* is prescribed by an appropriate code provision.

In geometrical terms, the hypersphere having radius β^* and with center at the origin of the axes u_i is required to lie within the transformed safety domain. The justification for such a method is that most of the joint probability density of the variables involved will be concentrated in

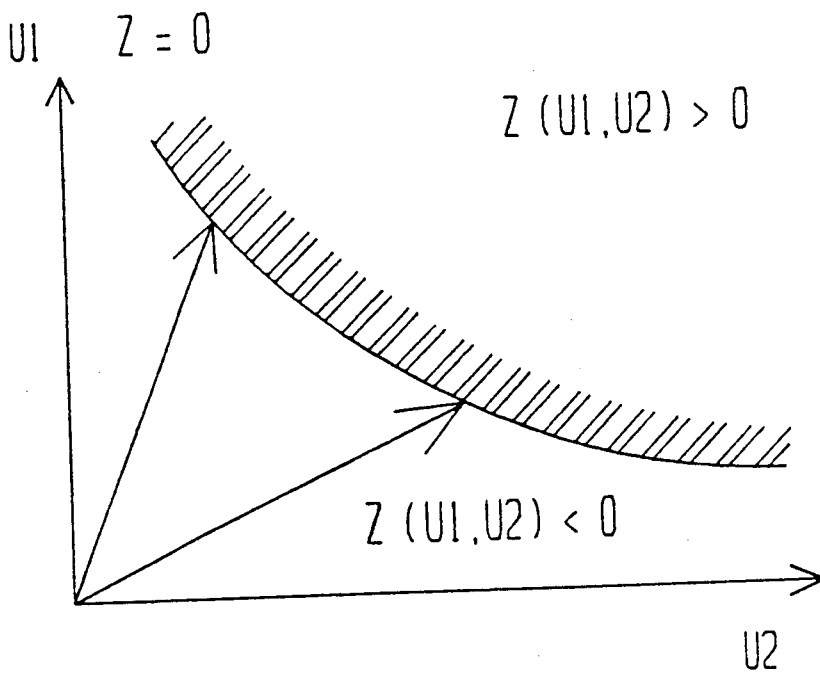


Figure 4.3. Safety checks.

the hypersphere having radius β^* , and that consequently it will be associated with values of vector X belonging to the safety domain. Mathematically, the problem to be solved is to find

$$\beta = \min (\sum u_i^2)^{\frac{1}{2}}$$

In a great number of cases the safety boundary domain is linear, and one can write an expression for z as follows:

$$z = g(x_1, x_2, \dots, x_n) = b + \sum a_i x_i$$

Then, β can be immediately determined as follows

$$g(u_1, u_2, \dots, u_n) = b + \sum a_i x_i^m + \sum a_i \sigma x_i u_i = 0$$

and the distance of this hyperplane to the origin is

$$\beta = \sum (a_i \cdot x_i^m + b) / (\sum a_i^2 \sigma x_i^2)^{\frac{1}{2}}$$

Expressing in terms of the standardized variables is equivalent to replacing the hypersurface by the hyperplane passing through P^* , the point of minimum distance between the two geometric elements. A graphical illustration of this approximation in a two dimensional space is presented in Figure 4.4. Finally, the probability of failure, p_f , and

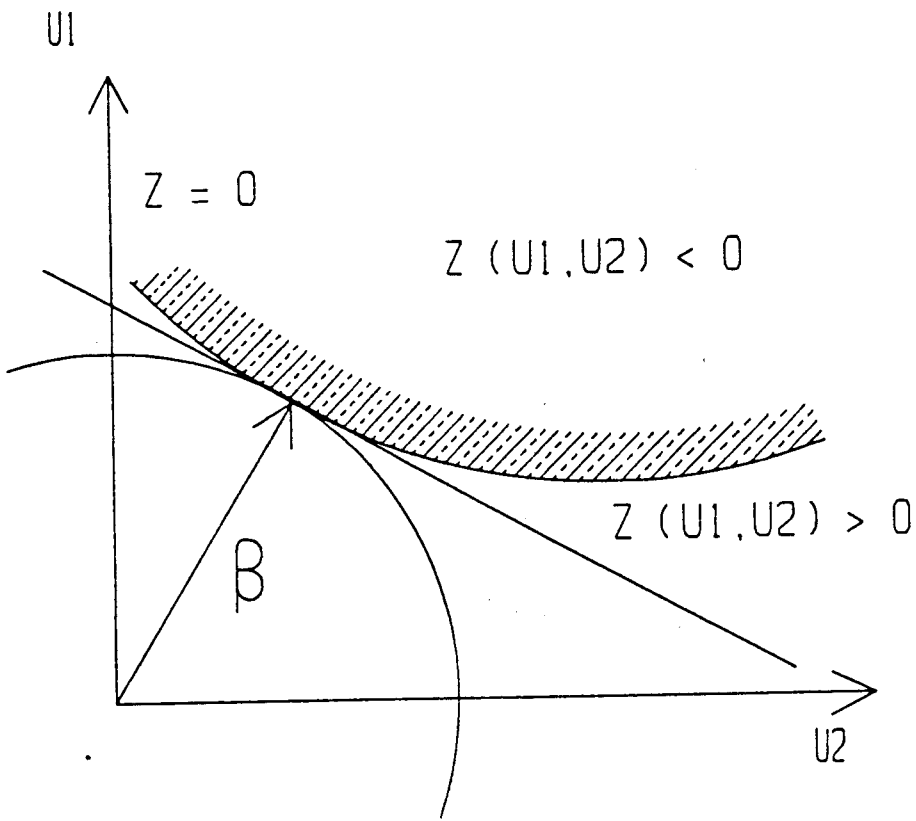


Figure 4.4. Reliability index.

the reliability index, β , are within certain approximations related by

$$p_f = 1 - \varphi(\beta)$$

where φ is the function of standardized cumulative normal distribution.

Reinforced Concrete Element Reliability

The element actions considered in analysis are only the moments at the member ends. These are the points of maximum value since only concentrated nodal loading is considered. The failure function z is then defined as

$$z = r - s = M_i - M_e$$

where

M_i - ultimate internal resisting moment;

M_e - maximum external element moment.

The external moment at the section is obtained from the element displacements using the condensed element stiffness matrix defined in the previous chapter. The expressions to obtain the value of M_i were defined in the previous chapter. The random values chosen in this study were the characteristic strength of the concrete, f'_c , and the maximum external moment in the element, M_e . All other

variables of the expression defining M_i could be taken as random but concrete strength was chosen due to the high coefficient of variation. Thus, the flexural failure function is linear and the respective reliability of failure can be easily calculated.

Compressive strength of concrete is influenced by a large number of factors grouped in three main categories, namely materials, production and testing. Material variability depend on the cement quality, moisture content, mineral composition, physical properties and particle shape of aggregates. The production factors involve the type of batching, transportation procedure and workmanship. Testing includes sampling techniques, test methodology, specimen preparation and curing (48).

It is difficult to evaluate correctly the importance of these three groups of factors. Their importance is certain to vary for different regions and construction projects. It has been found that the distribution of concrete compressive strengths can be approximated by the normal (Gaussian) distribution (49-50). Characteristic concrete compressive strengths obtained from a sampling of test data leads to a conclusion that for strength levels between 3,000 and 4,000 psi, the coefficient of variation is constant. For strengths beyond that range the standard deviation is constant (51). Since the values in reinforced concrete frames used are generally within the first interval the statistical value considered was the variance of f'_c . The

average standard variation for 68 a good quality control testing at the construction site is 550 psi. Using a 3500 psi specified compressive strength of concrete, f'_c , the required average compressive strength of concrete is the larger of the following (51):

$$f'_{cr} = f'_c + 1.34 \cdot i = 3500 + 1.34 \cdot 550 = 4237 \text{ psi}$$

or

$$f'_{cr} = f'_c + 2.33 \cdot i - 500 = 3500 + 2.33 \cdot 550 - 500 = 4282 \text{ psi}$$

The coefficient of variation of f'_c for this range of characteristic compressive strength is then given by

$$V = i/f'_c = 550/4282 = 0.128$$

and consequently the coefficient of variation of the concrete compressive flexural strength was adopted to be 0.15.

External loads have different coefficients of variation for the different types of loads (52-53). For most design and construction in the United States a good estimate for the coefficient of variation of dead loads is 0.10. For the live loads the coefficient of variations are very high and range from 0.39 to 1.04. For that reason and since the building codes prescribe large values for live loads that exceed the mean value a single coefficient of variation of 0.15 was adopted for the combination of dead loads plus live loads.

Basic variables considered, f_c and M_e , are assumed to have a probability function with normal distribution. This assumption is correct for the characteristic compressive strength of concrete but it does not hold for all external loads that create M_e . In the case where a statistical refinement of the basic variable M_e is required, there are techniques available to address the problem (47).

Since flexural failure function, z , is linear the reliability index β of each element can be calculated for any given external moment, section and material properties. Denoting the basic variables f_c as x_1 , and M_e as x_2 , and eliminating the other parameters involved in the equation, the flexural failure function takes the form

$$z = a_1 x_1 + a_2 x_2 + b$$

where

$$a_1 = \alpha \Omega b (kd)^2;$$

$$a_2 = -1;$$

$$b = A_s f_{cs} d' - A_s f_y d.$$

Standardizing the variables x_1 and x_2 leads to a replacement of the basic variables

$$u_1 = (x_1 - \mu_1)/\sigma_1$$

$$u_2 = (x_2 - \mu_2)/\sigma_2$$

where

μ_1 - mean value of f_c ;

μ_2 - mean value of M_e ;

σ_1 - standard deviation of f_c ;

σ_2 - standard deviation of M_e .

Replacing the standard normal variables in the flexural failure function the expression assumes the following form:

$$z = a_1\sigma_1u_1 + a_2\sigma_2u_2 + a_1\mu_1 + a_2\mu_2 + b$$

Then the reliability index for each element is given by the distance from the standardized failure function to the origin of the standardized basic variables as follows:

$$\beta = (a_1\mu_1 + a_2\mu_2 + b) / (a_1\sigma_1 + a_2\sigma_2)^{\frac{1}{2}}$$

CHAPTER 5

SYSTEM RELIABILITY

Introduction

Optimum structural design techniques are mainly based on deterministic assumptions. There is no doubt that some of the design variables should be considered including their random nature (54-55). Of course system reliability problems are more complicated than element reliability problems. This is evident since it must consider all multiple element failure functions, the several failure modes and, in some cases, the correspondent statistical correlation.

Another reason for including reliability considerations in structural optimization procedures is that, in some instances, the optimal solutions found have less redundancy and smaller ultimate load reserve than those solutions obtained with traditional design techniques (56-57).

There is no doubt that the combination of optimum design techniques and reliability-based design procedures creates a powerful tool to obtain a practical optimized solution. The objective is to find a balanced design

between all those that satisfy the optimization constraints and at the same time will have the lowest allowable probability of failure (58).

The strategy employed to evaluate the system reliability is described in the rest of the chapter. The elementary failure mechanisms of the structure are determined using Watwood's method. Then the system reliability is approximated using the Beta unzipping method, which consists of determining the relevant collapse mechanisms through linear combinations with fundamental mechanisms. The theory related with these techniques is tentatively described.

System Reliability and Optimization

A possible inclusion of the system probability of failure is to attribute a cost to system failure. This option originated a formulation based on the minimization of the total cost with the traditional optimization constraints (59). The objective function is as follows:

$$\text{Minimize } C_t = C_o + C_f P_f$$

where

C_t - cost of the structure;

C_o - initial cost of the structure;

C_f - cost of failure;

P_f - probability of structure failure.

This option is not commonly used for inhabited structures since it is difficult to evaluate the economic value of a structural failure where human life losses are expected. A more popular alternative is to include an additional constraint representing the maximum probability of failure allowed for the structure (60). The constraint for the system reliability will be of the type

$$P_f(\underline{X}) \leq P_m$$

where

P_f - probability of system failure;

\underline{X} - vector of design variables;

P_m - allowable probability of system failure.

When performing structural optimization one may consider serviceability and ultimate limit states. This possibility leads to another type of formulation where the objective function and constraints for these limit states are considered simultaneously (61). This type of problems are called reliability-based optimization and can be summarized as follows:

Minimize C_0

subject to

$$G_i(\underline{X}) \leq 0, \quad i=1, m$$

$$P_u \leq P_{uo}$$

$$P_s \leq P_{so}$$

where

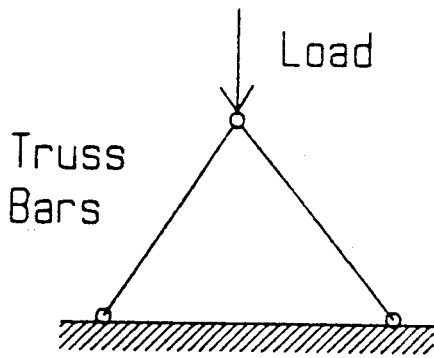
- G_i - optimization constraints;
- m - number of behavior constraints;
- P_u - probability of ultimate system failure;
- P_{uo} - maximum probability of ultimate system failure;
- P_s - probability of serviceability failure;
- P_{so} - maximum probability of serviceability failure.

The option adopted consisted of adding a constraint on the system failure. The value of the system failure at the end of the optimization cycle is compared with the target value. If it is not satisfactory the element requirements are modified and the optimization is restarted.

Methods

In determinate structures the collapse of any member will lead to system failure. The probability of system failure can be obtained as the probability of the union of member probability failures (16). These types of structural systems are called series systems or weakest-link systems. Redundant structures will fail only if all redundant members collapse. If this condition does not arise, whenever a member fails there will be a redistribution of loads in the system. These types of structures are called parallel systems. Graphic examples are presented in Figure 5.1.

SERIES SYSTEM



PARALLEL SYSTEM

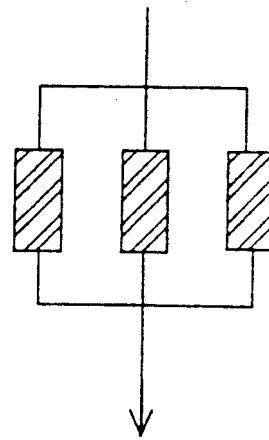
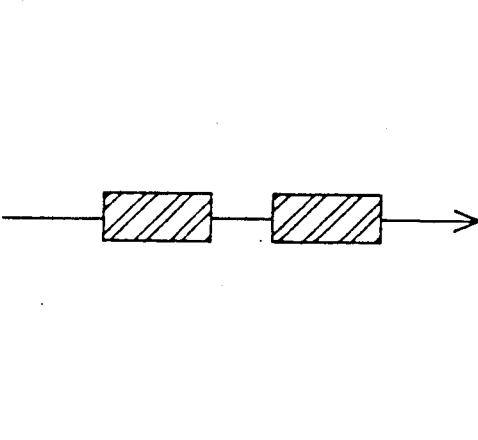
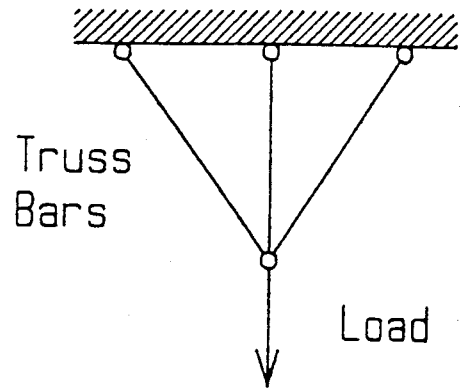


Figure 5.1. System models.

Series systems with n elements have n failure modes. Parallel systems with n elements have more than n failure modes. These failure modes in parallel systems are dependent on whether the failure type of the elements is brittle or ductile (62-63). For redundant brittle systems the failure of an element and consequent redistribution of the loads will provoke the system failure. In these cases the system behavior may be considered to be generally identical to that of as a series system.

Probability of failure of a series system can be considered as the union of the elements probability of failure

$$P_{fs} = P(U_i(Z_i \leq 0) | i=1, n)$$

where

U - union of events;

P_{fs} - probability of system failure;

Z_i - failure function of element i .

If the element failure functions are not correlated then the evaluation of P_{fs} is relatively easy and may be assumed as

$$P_{fs} = 1 - \pi_{i=1}^n (1 - P(e_i=0))$$

where

π - product;

$e_i = 0$ if element is in a failure state,

$e_i = 1$ if element i is in a non-failure state;

$P(e_i=0)$ - probability of failure of element i .

When there is correlation between element failure functions then the calculations become more complicated and time consuming. To avoid the exact evaluation, approximation and bound techniques have been developed (64-65). The best known is the simple bounds. In this approach the upper bound for the probability of system failure assumes that all element failure functions are uncorrelated and the lower bound is obtained assuming full dependence between the element failure functions. If a more sophisticated bounding technique is necessary the Ditlevesen bounds may be used (17). The drawback is that this sophistication implies the calculation of event joint probabilities. A similar simplified approach to that used in series systems may be adopted to find the simple bounds for the failure of a parallel system.

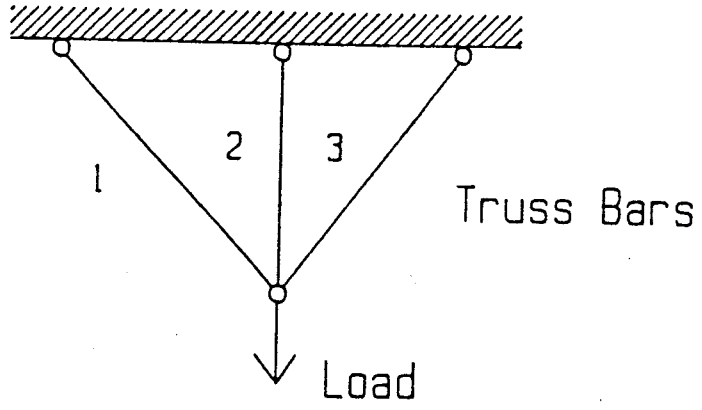
In the case of parallel systems the lower bound corresponds to the case where there is no dependence between the elements failure and the upper bound corresponds to full dependence between all elements failure (66). Exact evaluation of the probability of system failure is very difficult to obtain if the system has more than three elements. To solve a general problem, approximation or bounding techniques are used. For instance, for redundant

To exemplify the determination of all possible failure modes the initial step is to build a directed network, or directed graph, with all possible events involving element failures that will lead to a collapse. Each node represents a stable configuration and each branch corresponds to a element failure. Each path is a set of branches connecting the initial and final nodes. A cut of the graph is a set of branches containing only one branch from every path. A simple example is presented in Figure 5.2.

Methods based on the determination of fundamental failure mechanisms using practical simplifications from graph theory have been implemented (69). The Beta unzipping method and the branch and bound method are two examples. The principal advantages are that they are precise and easy to use. The Beta unzipping method finds the significant collapse mechanisms using combinations of fundamental mechanisms and rejecting those combinations that are outside a prescribed interval. The branch and bound method selects all failure paths that have high probabilities of occurrence. Although less exact, the Beta unzipping method was chosen due to its simplicity and performance.

Generation of Failure Modes

To define all failure mechanisms, the first step consists of determining the set through manipulation of elementary failure mechanisms. To obtain these, the method



FAILURE GRAPH

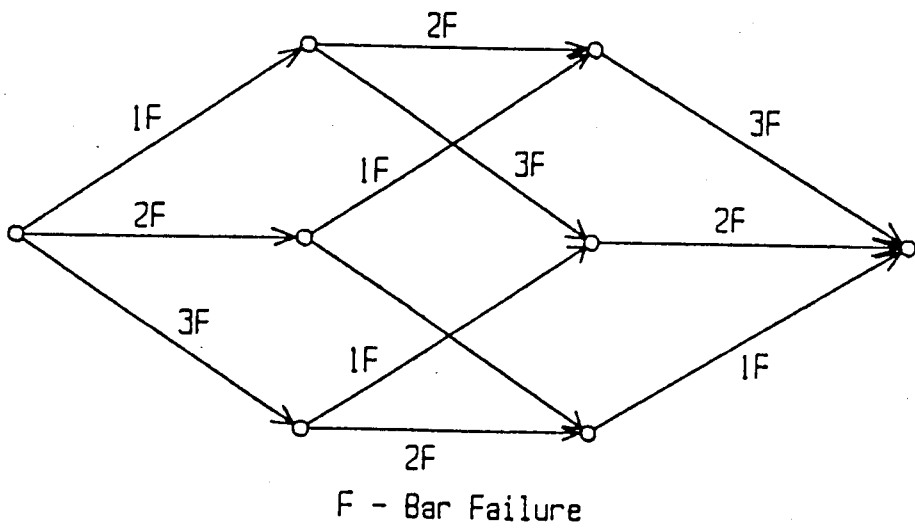


Figure 5.2. Failure graph.

adopted was conceived by Watwood (15). It is an automatic tool to generate all failure mechanisms with one degree of freedom, or elementary failure mechanisms, of a given frame. The set of these mechanisms and all their linear combinations constitute all possible collapse configurations (70). The technique is relatively simple to use since the input data for this method is the same for traditional elastic analysis like joint and element information.

Elementary failure mechanisms are dictated by the geometry of the structure and potential hinge locations created by the external load configuration. Hinge locations are considered at the end of each member. In the case where there are loads in the middle of the element, they are also considered at the points of concentrated or discretized loads. The element axial collapse is not considered in this formulation although it was included in the original version.

Element global displacements of a planar frame form a vector with six variables, $\{\underline{S}\}$. Using a cartesian referential set of axes x and y the displacements, S_1 to S_6 , may be represented as in Figure 5.3. Element deformation parameters may be defined by three independent quantities

- S'_1 - displacement about node i ;
- S'_2 - rotation of node i ;
- S'_3 - rotation of node j .

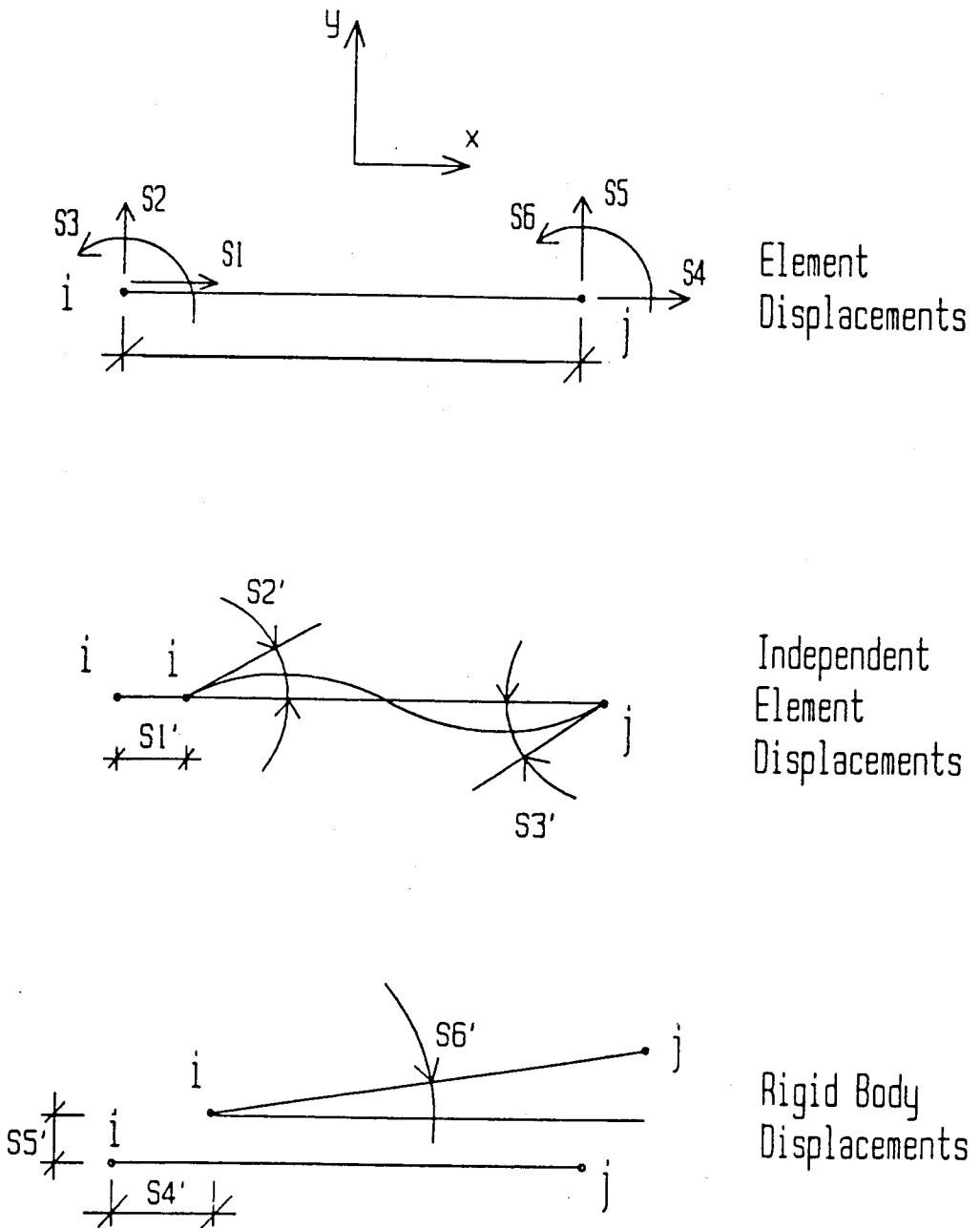


Figure 5.3. Element displacements definition.

When a mechanism is formed each element moves as a rigid body. The rigid body motion of an element of a planar frame can be defined by three parameters. They can be expressed in terms of the global coordinates x, y as

S'_4 - translation in the x direction;

S'_5 - translation in the y direction;

S'_6 - rotation about node i .

Two sets of three independent displacements, rigid body parameters and element deformations, create the transformed coordinate vector, $\{\underline{S}'\}$. A relation can be established between local global coordinates and transformed coordinate vector represented by a linear transformation $[T]$.

$$\{\underline{S}\} = [T] \{\underline{S}'\}$$

where

$$[T] = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -L \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

L - element length.

For any elementary failure mechanism the element deformations, S'_1, S'_2, S'_3 , must be zero. This is only for elements that do not have plastic hinges. To materialize this condition, a matrix C_k is introduced for each element

k. This matrix is created with the first three rows of matrix T^{-1} for the k^{th} element. The global condition matrix, C , is a block diagonal matrix consisting of the C_k matrices as follows:

$$C_k = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1/L & 1 & 0 & 1/L & 0 \\ 0 & -1/L & 0 & 0 & 1/L & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} C_1 & 0 & \dots\dots\dots & 0 \\ 0 & C_2 & \dots\dots\dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots\dots\dots & C_n \end{bmatrix}$$

Using the previous matrices and vectors the following relation now holds

$$C \{S\} = \{S'_d\}$$

where

$$\{S\} = \begin{bmatrix} S \\ - \\ S \\ - \\ \vdots \\ - \\ S \end{bmatrix} \begin{array}{l} \text{- first element} \\ \\ \text{- second element} \\ \\ \\ \text{- } n^{\text{th}} \text{ element} \end{array}$$

and

$$\{S'_d\} = \begin{bmatrix} S'_1 \\ S'_2 \\ S'_3 \\ --- \\ \vdots \\ --- \\ S'_1 \\ S'_2 \\ S'_3 \end{bmatrix} \begin{array}{l} \text{- first element} \\ \\ \\ \\ \\ \text{- } n^{\text{th}} \text{ element} \end{array}$$

Compatibility between the element displacements, $\{\underline{S}\}$ and the structure global degrees of freedom $\{\underline{r}\}$ can be established

$$\{\underline{S}\} = [\underline{Q}] [\underline{A}] \{\underline{r}\}$$

where

$[\underline{Q}]$ - rotation matrix;

$[\underline{A}]$ - compatibility matrix.

From previous equations the following expression holds

$$[\underline{C}] [\underline{Q}] [\underline{A}] \{\underline{r}\} = \{\underline{S}'_d\}$$

or

$$[\underline{B}] \{\underline{r}\} = \{\underline{S}'_d\}$$

An elementary mechanism of the structure is a solution of the homogeneous system

$$[\underline{B}] \{\underline{r}\} = \underline{0}$$

If the structure configuration is not a mechanism there is no solution for the system except the trivial solution. To obtain a mechanism, releases of the global degrees of freedom must be introduced. Two releases per element are added corresponding to the hinges at the ends or points of application of concentrated or discretized loads. Each

release corresponds to an addition of an external global degree of freedom.

Addition of external degrees of freedom is done by replacing a row in matrix $[Q] [A]$ with zeros. The changed rows correspond to the element degrees of freedom S_3 and S_6 , the node rotations. For each row that is replaced, a unit column vector is added to the matrix $[Q] [A]$ with a 1 in the row that has been replaced. The dimensionality of $\{\underline{r}\}$ is increased by the number of rows replaced in $[Q] [A]$. The total is a set of extra columns with a dimension that is twice the number of elements. The homogeneous system becomes

$$[C] ([Q] [A])^* \{\underline{r}_a\} = [B'] \{\underline{r}_a\} = 0$$

where

$([Q] [A])^*$ - matrix with extra $2n$ columns;

$\{\underline{r}_a\}$ - vector of increased global degrees of freedom.

Matrix $[B']$ is not square and has a greater number of columns than the number of rows. The solution of the system of homogeneous equations above may be obtained using a technique similar to that when solving for redundant unknowns in the force method. Difference between number of rows and number of columns is the number of independent solutions, that coincides with the number of elementary mechanisms. Suppose the rank of $[B']$ is the number of columns, m , and that the number of columns is p . In this

case one can find a matrix $[D]$, nonsingular with dimensions p by p such that

$$[B'] [D] = [I \mid 0]$$

where

$[I]$ - identity matrix, m by m ;

$[0]$ - null matrix, m by $(p-m)$.

Last columns of $[D]$ are independent solutions of the homogeneous system of equations since they are orthogonal to the rows of $[B']$. To obtain $[D]$, a reduction is performed on the columns of $[B']$ that is conceptually identical to a Gauss-Jordan reduction (15). The solution of such a system of equations is illustrated in Figure 5.4, where all elementary failure mechanisms for a two story frame are presented.

Beta Unzipping Method

Advantages of the Beta unzipping method, as stated before, are important. It can be used for reliability estimation of planar and spatial trusses and frames made with ductile or brittle elements. The probability of failure can be evaluated with different levels of accuracy. It is also a method that can be easily implemented for automated calculations.

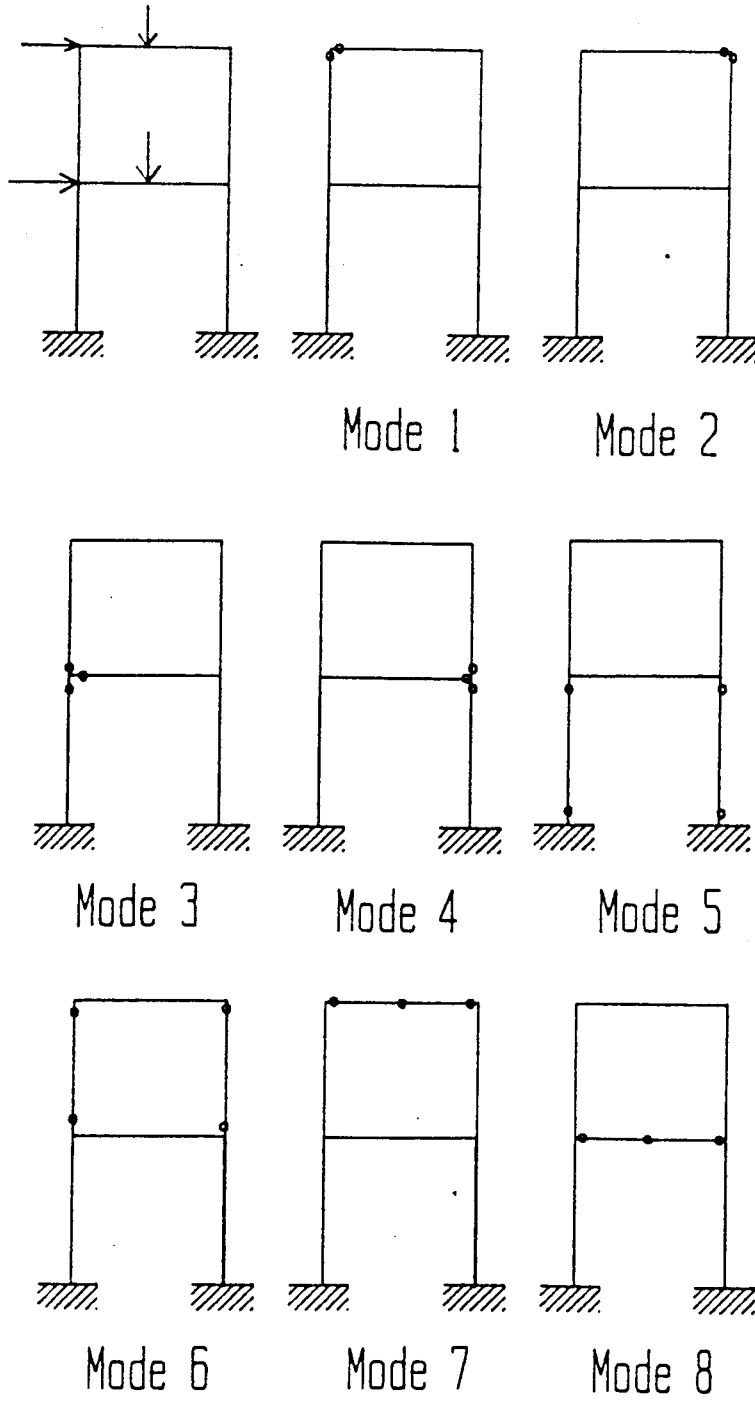


Figure 5.4. System failure modes.

At Level 0 the estimation of the system reliability is based on the failure of a single structural element. In this case the system reliability is equal to the reliability of the element with the higher probability of failure.

Level 1 gives more acceptable results. The concept is that the structural system is modelled as a series system. The system probability of failure is estimated as a function of element probabilities of failure. The calculation of this system probability can be made with acceptable accuracy using only those elements with a low reliability index. The interval where these significant or critical elements are located is defined by $[\bar{A}_{\min}, \bar{A}_{\min} + \bar{\Delta A}]$, where $\bar{\Delta A}$ is chosen adequately.

At Level 2, failure elements are grouped in pairs as parallel systems. These significant pairs of failure elements are obtained assuming failure of the significant elements as defined in Level 1. For element i , the load-carrying capacity is added as fictitious loads if the element is ductile. If the element is brittle, no fictitious loads are added. Then new element reliability indices of the modified structure are calculated and the critical pairs are formed with element i and the new significant elements.

The process can be analogously repeated for levels greater than 2 creating critical groups of 3, 4, or more elements. It is considered that above Level 3 there is no practical benefit from the extra calculations. The method

could be used for ductile structures with the formation of all collapse mechanisms but the computational effort of the reanalysis is too great. It is preferable instead to use the fundamental mechanisms and their linear combinations.

A structure with an elasto-plastic behavior and a given static load configuration has a certain number of fundamental failure mechanisms that can be determined using the Watwood's method. Since they are one degree of freedom mechanisms the failure function z_i for mechanism i can be evaluated as follows:

$$z_i = \sum |a_{ij}| R_j - \sum b_{ik} P_k$$

where

- a_{ij} - rotation at hinge j in mechanism i ;
- b_{ik} - displacement of load k in mechanism i ;
- R_j - strength of element j ;
- P_k - load number k .

Total number of collapse mechanisms is generally too high and a significant portion of these have a low probability of failure. For this reason, the Beta unzipping method is used as it only considers the most critical failure modes. The value found is a lower bound for the exact probability of failure, since some mechanisms are discarded. Once the identification of the fundamental mechanisms and respective reliability indices are obtained, the next step is to choose the elementary mechanisms that

will be the starting points. Ordering the reliability indices as follows:

$$\beta_1 \leq \beta_2 \leq \dots \leq \beta_q$$

where

β_i - reliability obtained using z_i .

A control value ϵ_1 is selected and added to β_1 . This value and β_1 define an interval $[\beta_1, \beta_1 + \epsilon_1]$. All mechanisms outside this interval are discarded for future combinations. The linear combinations to generate new failure mechanisms are obtained through combinatorial matching. First, elementary mechanism 1 is combined with the mechanisms in the interval and their reliability indices are evaluated. The same process is repeated for the rest of elementary mechanisms with those in the interval. The mechanisms are ordered in accordance with their reliability indices, a new interval is defined and a new generation of failure mechanisms is originated. The procedure is repeated until a sufficient number of generations is accomplished. Figure 5.5 exemplifies the failure tree creation.

To define the failure function z_{ij} for the combinations of the pair of mechanisms i and j can determined as

$$z_{ij} = \sum |a_{ir} \pm a_{jr}| R_r - \sum (b_{is} \pm b_{js}) P_s,$$

where the sign + or - is chosen to give the smallest reliability index. A graphical illustration of these combinations is shown in Figure 5.5.

Tree of Mechanism Combinations

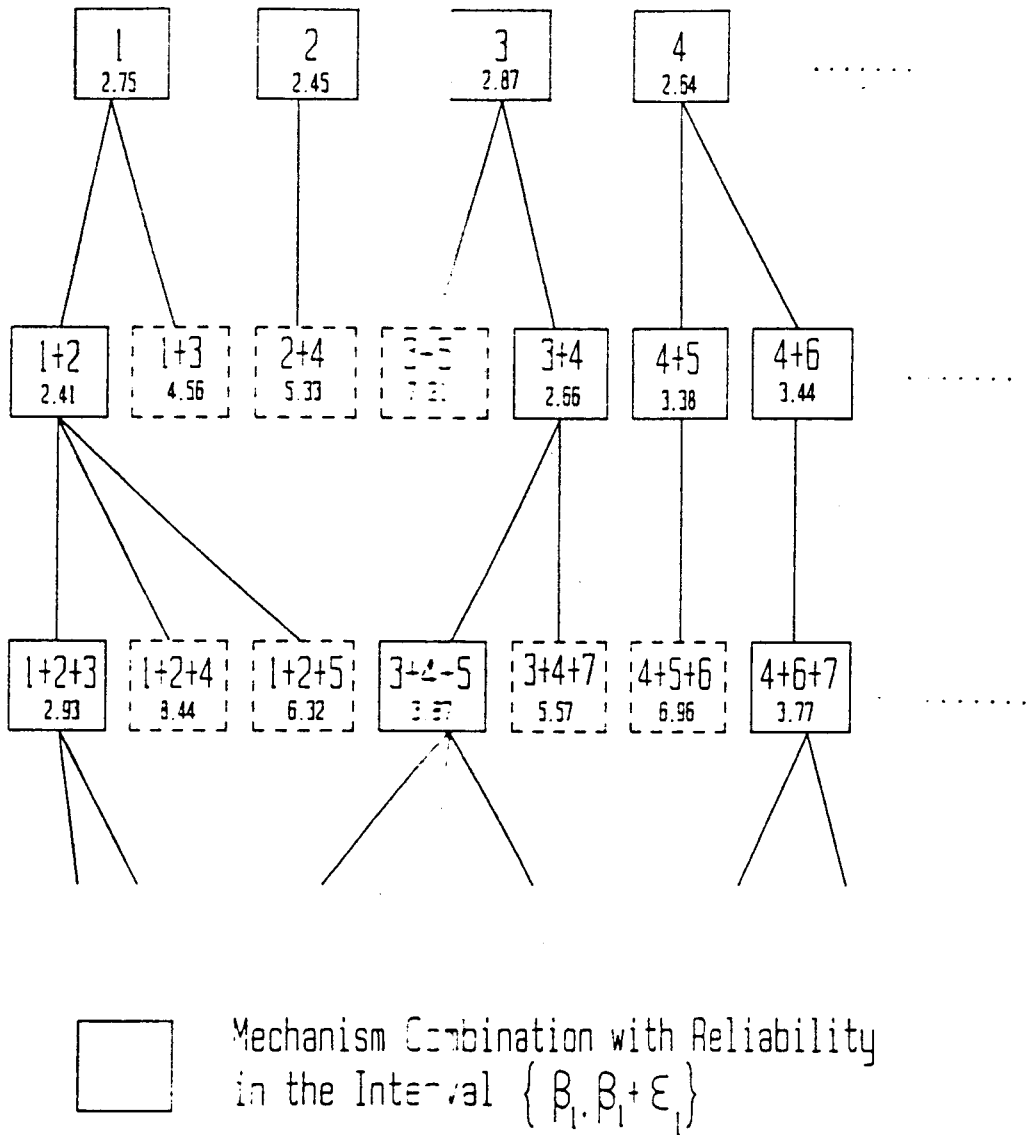


Figure 5.5. Combinatorial tree.

CHAPTER 6

PROCEDURE IMPLEMENTATION

Introduction

The optimization problem was tentatively solved using two strategies. These were an Augmented Lagrangian method, abridged by the class of penalty functions, and the Generalized Reduced Gradient method, classified in the group of gradient type techniques. The unconstrained minimization techniques experimented with the Augmented Lagrangian formulation are reported and performance is analyzed. Two final versions for these two options are discussed, with emphasis on the problems and decisions taken. Subroutines are described and their essential characteristics underlined.

Procedures for element and system reliability evaluations are detailed. Subroutines involved in the element reliability calculation are listed and their specific task described. The system reliability determination at the mechanism level is outlined with a summarized description of the Beta unzipping method. Particular problems, and respective solutions that arose

during the implementation and testing are presented and discussed.

Augmented Lagrangian Formulation

The set of design variables is divided in two main groups. These are the dimensions and steel area, defining each element cross section, and the global displacements. The objective function is the cost of the structure as a function of the the volume of concrete and steel. Equality constraints are defined by the global equilibrium equations. Inequality constraints include the bounds on global displacements and the minimum element reliability.

In a reinforced concrete portal frame the set of design variables x having n elements and m global degrees of freedom is partitioned as follows:

x_i , $i=1,4,\dots,3n-2$ - base of rectangular element section;
 x_j , $j=2,5,\dots,3n-1$ - height of rectangular element section;
 x_k , $k=3,6,\dots, 3n$ - area of steel on one side of section;
 x_l , $l=3n+1, \dots, 3n+m$ - global displacements.

The objective function used in this formulation was defined using the average costs of cast in place concrete for reinforced concrete frames and main reinforcing steel (71). Combined relative cost function was obtained dividing

both unitary costs by the cost of concrete and the result is as follows:

$$f(\underline{x}) = (x_i * x_j + x_k * 10) * L_p$$

where

L_p - length of element p , $p = 1, m$.

Equality constraints, one for each global degree of freedom, are defined as

$$h_q = \sum_r (k_{qr} * x_{3n+r}) - R_q, \quad q=1, \dots, m, \quad r=1, \dots, m$$

where

k_{qr} - global stiffness coefficient;

x_{3n+r} - global displacement r ;

R_q - external force q .

Inequality constraints that control the maximum global displacements and impose a minimum element reliability are as follows

$$g_i = x_{3n+r} - d_r \leq 0, \quad i=1, \dots, m$$

$$g_j = rel_j - beta_j, \leq 0 \quad j=1, \dots, n$$

where

d_r - maximum absolute value of global displacement r ;

rel_j - reliability index of element j ;

$beta_j$ - minimum reliability index prescribed for element j .

The minimization problem stated previously is highly nonlinear. Consequently the adoption of the optimization strategy was crucial and its characteristics played a big role in the decision process. The Augmented Lagrangian Multiplier method, or Augmented Lagrangian formulation, was the first choice. It allows for an adaptation of the search technique to the shape of the design surface since the dual variables, or lagrangian multipliers, are updated at the end of each minimization cycle in function of the constraints violations.

The constrained problem is thus transformed into an unconstrained function using the Augmented Lagrangian formulation with the addition of dual variables \underline{u} and \underline{v} and becomes

$$L(\underline{x}, \underline{u}, \underline{v}) = f(\underline{x}) + \underline{u}^t \cdot \underline{h}(\underline{x}) + P \cdot \underline{h}^t(\underline{x}) \cdot \underline{h}(\underline{x}) + \\ \underline{v}^t \cdot \underline{g}^*(\underline{x}) + P \cdot \underline{g}^{t*}(\underline{x}) \cdot \underline{g}^*(\underline{x})$$

where

$$\underline{g}^*(\underline{x}) = \max \{ \underline{g}(\underline{x}), -\underline{v}/(2P) \};$$

P - penalty factor.

Pseudo-objective function L is minimized with fixed values of \underline{u} and \underline{v} and these are updated using the following rules (72)


$$\underline{u}^{k+1} = \underline{u}^k + 2P\underline{h}(\underline{x})$$

$$\underline{v}^{k+1} = \underline{v}^k + 2P\underline{g}^*(\underline{x})$$

The minimization cycle is repeated until there is no significant improvement of the objective function $f(\underline{x})$. Penalty parameters, P , contribute significantly to the efficiency of the minimization procedure. Initially, there was only one penalty parameter for equality and inequality parameters alike. Since these two types of constraints have different sensitivities to changes of the design variables, different penalty parameters were introduced for the group of equality constraints and the group of inequality constraints. These starting values, and consequent updates, were tuned to the optimization performance to improve the procedure efficiency. Penalty parameters were obtained from a set of experimental trials and assessment of the results.

Scaling of the constraints and variables also played an important role in the optimization. The equality constraints and objective function were scaled to the same order of magnitude as the inequality constraints. This was an attempt to regularize the magnitude of the different functions composing the augmented lagrangian function with the intent of smoothing the design surface.

Two methods were tested for the unconstrained minimization of the augmented lagrangian function. These were the Conjugate Gradient method, or Fletcher-Reeves, and the Hooke and Jeeves method. The former one is a variation of the Steepest Descent method, or Gradient method, and is classified as a first order method since it is based on the gradient of the function. The latter is defined as a zero



order method because it does not rely on information about the shape of the function obtained from derivatives. Methods based on the second derivatives were discarded since the problem was highly nonlinear and inequality constraints had discontinuous second derivatives.

Conjugate Gradient is based on obtaining consecutive directions that are linearly independent, thus accelerating the search. The algorithm for the method is summarized as follows

Step 1: Calculate $\text{grad } f(\underline{x}^k)$;

Step 2: $d^k = -\text{grad } f(\underline{x})$;

Step 3: Find α^k so that $f(\underline{x}^k + \alpha^k.d^k) = \min$;

Step 4: $\underline{x}^{k+1} = \underline{x}^k + \alpha^k.d^k$

Step 5: Check convergence. If converged, stop.

Step 6: $d^{k+1} = d^k + [\text{grad } f(\underline{x}^{k+1})^2 / \text{grad } f(\underline{x}^k)^2].d^k$

Go to 3.

Conjugate Gradient method proved to be unsuitable for the type of function presented. Progress in the minimization was minimal due to the ridge-type shape of the function. Whenever the process started at any point where

the equality constraints were satisfied, the gradient method was unable to progress to a better point. This happened because any violation of the constraints created a high increase of the augmented lagrangian function and the point behaved as a local minimum. Otherwise, if the point did not satisfy the equality constraints then the accuracy of the derivatives obtained through forward difference, was not good enough to converge to a better design point. As an example of this abruptness the augmented lagrangian function is represented as a function of displacements x_2 and x_3 of the cantilever shown in Figure 6.1.

Several techniques were implemented to smooth the shape of the augmented lagrangian to no avail. Scaling of the variables, objective function and of the constraints were performed. The displacements were scaled by the multiplication of a constant regularizing the magnitude of the set of variables. The scaling of the constraints and objective function were already referred to as well as another technique based on the evenness of the rate of change of the constraints and objective function in terms of the design variables (12).

These reasons justified the final adoption for unconstrained minimization of the Hooke and Jeeves method. This technique had provided acceptable results and performances in the linear elastic formulation. The main algorithm may be summarized as follows

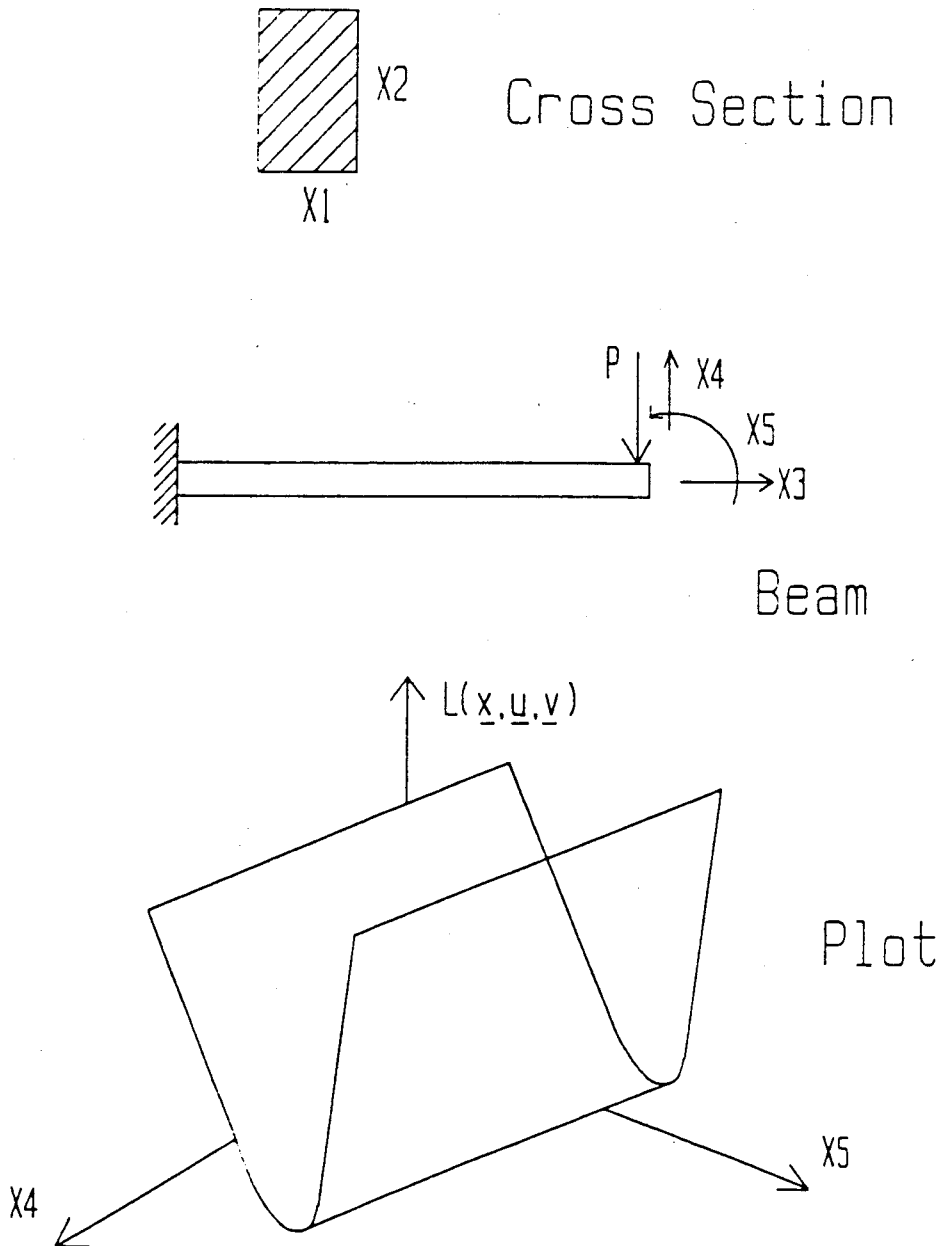


Figure 6.1. Augmented lagrangian function plot.

Step 1: $x_i^k = x_i^{k-1} * (1 - \alpha)$

Step 2: If $L(\underline{x}^k) < L(\underline{x}^{k-1})$, $\alpha = \alpha * \text{inc}$;
 Otherwise, $\alpha = \alpha * \text{dec}$;

Step 3: $i = i + 1$; go to 1 if $i \leq n$;

Step 4: Try pattern move $\underline{x}^* = \underline{x}^k + \beta(\underline{x}^k - \underline{x}^{k-1})$;

Step 5: Verify termination criteria. If not met, go
 to 1. Otherwise, stop.

where

inc - increase factor;

dec - decrease factor;

β - stepsize parameter.

Flowchart of the final group of subroutines is presented in Figure 6.2. The main program PRINCI, reads the main input data, initializes the correct displacement values, when solving the equilibrium equations, for the starting dimensions, calls the optimizer subroutine and writes the final results. Subroutine OPTIMI controls the optimization process by verifying if the convergence criteria is met, updating the lagrangian multipliers and verifies the system reliability at the end of the optimization cycle. Subroutine HOOJEE conducts the Hooke

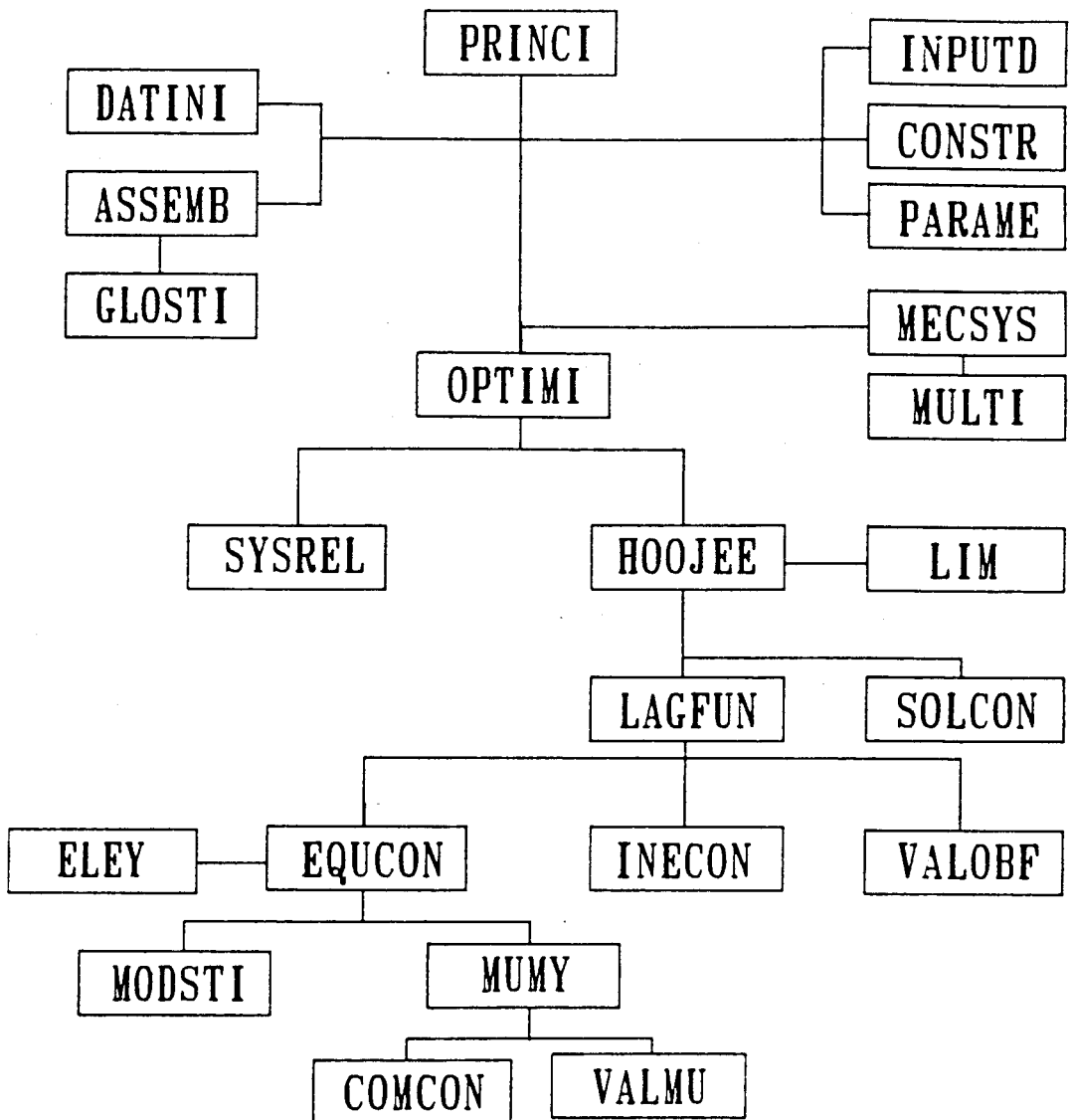


Figure 6.2. Augmented Lagrangian version flowchart.

and Jeeves minimization operation. Subroutine LAGFUN calculates the value of the augmented lagrangian function. Subroutine DATINI initializes the values of the scaling factors and lagrangian multipliers. ASSEMB is the subroutine that creates the initial global stiffness matrix with the starting values of the elements. Subroutine GLOSTI solves the initial equilibrium equation system to obtain good initial displacement values. INPUTD is the subroutine that reads all the data concerning the definition of the structure. CONSTR is the subroutine that reads the values of the constraints. PARAME is the subroutine that inputs all optimization parameters. Subroutine MECSYS defines all elementary failure mechanisms of the structure and MULTI is a related subroutine that multiplies matrices. SYSREL is the subroutine that calculates the element reliability. Subroutine LIM controls the maximum and minimum values of the design variables excluding displacements. SOLCON is the subroutine that obtains the global displacements with the nonlinear global stiffness formulation. Subroutine EQUCON evaluates equality constraints values, INECON calculates inequality constraint values and VALOBF obtains the objective function value. ELEY is the subroutine that recovers the element forces with the current displacement and element stiffness values. Subroutine MODSTI assembles the nonlinear stiffness values using the actual spring secant stiffness values. MUMY is the subroutine that evaluates the ultimate and yielding values calling,

respectively, subroutines VALMU and COMCON. The coding of the main subroutines is presented in Appendix A.

This method didn't provide acceptable performance for the nonlinear material behavior. The convergence of the method concerning the equality constraints was impossible to obtain, probably because the variations of the equality constraint values were severe whenever there was any change of the design variables. For this reason, a mixed method of integrated and cycling formulations was implemented. The approximate displacements were obtained using only once a Gauss type solution method of the equilibrium equations at the end of each optimization cycle. The Hooke and Jeeves search did not include the set of displacements although the group of equality constraints remained in the augmented lagrangian function definition. The main goal of this modification was to improve the convergence for the equality constraints while performing an optimization that would remain in the vicinity of the previous design point.

Generalized Reduced Gradient

The optimization strategy is based on the iterative solution of a system of nonlinear equalities. The method was initially implemented as an extension of the decomposition for linear programming problems (73). Several variations and enhancements of this initial formulation

followed this work (74-76) but the general formulation of the problem is

$$\begin{aligned} & \text{Minimize } f(\underline{x}) \\ & \text{subject to } h_i(\underline{x}) = 0 \quad i=1, \dots, m \\ & \quad \underline{x}^l \leq \underline{x} \leq \underline{x}^u, \underline{x} \text{ of } R^n, m \leq n \end{aligned}$$

Inequality constraints are handled as pseudo equality constraints with the addition of slack variables. This increase of the size of the variable set is balanced by the implicit variable elimination generated by the following relation between changes of design variables

$$d\underline{x}^b = -J^{-1} C d\underline{x}^{nb}$$

where

\underline{x}^b - vector of basic variables;

\underline{x}^{nb} - vector of nonbasic variables;

J^{-1} - columns of jacobian matrix of equality

constraints corresponding to basic variables

$[d\underline{h}/d\underline{x}^b]$;

C - other columns of jacobian matrix corresponding to the nonbasic variables $[d\underline{h}/d\underline{x}^{nb}]$.

Nonbasic variables are thus calculated as a function of the basic variables and eliminated from the gradient calculation. The gradient is calculated whenever a feasible point is obtained and a line search along that direction

tends to provide a better design point while maintaining feasibility. The values of the nonbasic variables are consequently evaluated and the process is restarted. If any of the basic variables is at any bound, a search is performed on the set of nonbasic variables to find a suitable replacement. The method creates basically a succession of feasible solutions $\underline{x}^0, \underline{x}^1, \dots, \underline{x}^p$, each one corresponding to an improvement of the objective function from the previous design point. The iteration is terminated whenever the convergence criteria is satisfied or the maximum prescribed number of iterations is exceeded.

The essence of this optimization technique seemed adequate for the integrated optimization with nonlinear constraints and nonlinear material behavior with a considerable number of equality constraints. The number of slack variables is not large, and as long as the initial point is feasible, convergence should be quicker than in the previous version.

To illustrate and assess the performance of the generalized reduced gradient method, an example of a cantilever beam with linear material behavior, submitted to displacement and stress constraints, was solved. This example is presented in Appendix B together with a flowchart of the general algorithm. These preliminary results were very promising and the following step was to extend the method to the formulation previously tested with the augmented lagrangian function.

The procedure was developed in two phases. First, a linear material behavior was assumed followed by the inclusion of material nonlinear behavior was included. The computer program versions were created combining a public domain software package and some subroutines already used in the prior formulation (76). The coded version of the Generalized Reduced Gradient method is a general purpose program for constrained optimization and was changed slightly when adapting to the present case.

Main modifications actually introduced in the software were to increase the maximum number of variables and constraints, the extension of the maximum number of Newton iterations, and the modification of the number of times the stepsize could be reduced when performing the line search. The size of the problems tested caused the first alteration. Although the authors had not tested the program with examples as large as those described in the next chapter, the computer code performed with no problems. The variation of the maximum number of iterations was required due to the material nonlinear behavior, which imposed a slower computation of the basic variables when iteratively solving the set of nonlinear equalities. The need for smaller step sizes was due to the fact that the order of magnitude of the change of variables in the vicinity of the design point is very small compared to the corresponding changes of the equilibrium equality constraints.

As specified before, some subroutines were directly used from the Augmented Lagrangian formulation while others had to be adapted or created. Since the data transfer between subroutines in the Generalized Reduced Gradient program was made through common data blocks, the same methodology was used for most of the added subroutines. The flowchart of this package is presented in Figure 6.3. An example of the input data files and the listing of the new or modified subroutines is presented in Appendix C. The unmodified subroutines perform the same tasks as described before.

Essential structure of this program is the same as presented by the authors of the optimization package. There is a program, OPTIMI, that calls the main subroutines PRINCI, DATAIN, GRG and OUTRES. PRINCI reads the initial data from file DATA1 that is not abridged by the typical input data of the optimization package, which is read in subroutine DATAIN. The subroutine GRG performs the problem optimization calling other subroutines. The only subroutine written for this implementation was GCOMP that computes the values of the equality constraints, the inequality constraints, and the objective function. The system reliability was evaluated and the process was restarted if the results were unsatisfactory. Subroutine OUTRES writes the final results of each optimization run to a file RESULT. This subroutine was also modified to include the relevant

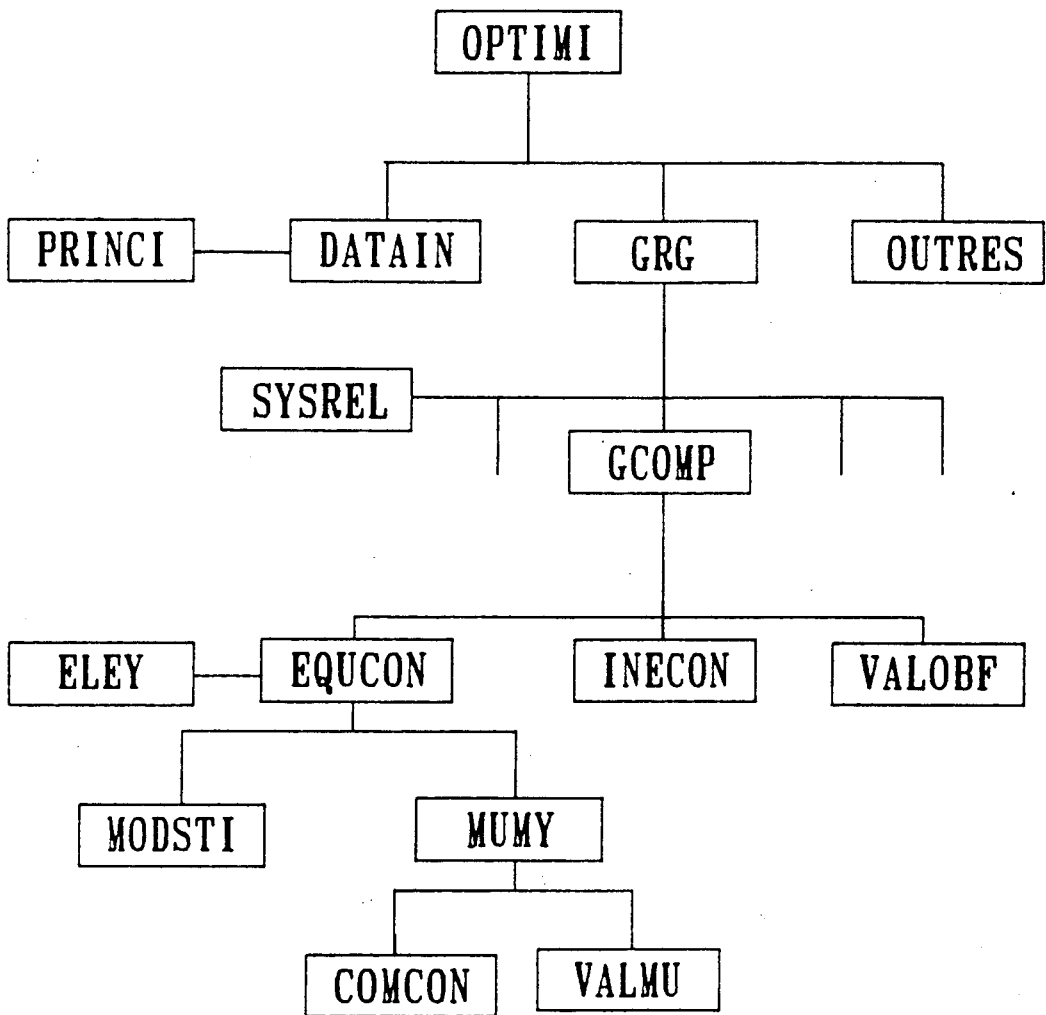


Figure 6.3. Generalized Reduced Gradient version flowchart.

results of this type of problems, and to modify the output format of the optimization information.

The particular structure of the optimization package created some conceptual alterations in the group of subroutines that evaluate the constraints and control the variables bounds. For instance, the changes in the design points are made simultaneously for all design variables. Therefore the limits of the areas of reinforcement had to be imposed as variable bounds instead of being controlled by a specific subroutine.

On the other hand the limits imposed on the displacements could be considered as upper and lower bounds of the correspondent design variables, and consequently, the total number of inequality constraints was substantially reduced. The only scaling introduced in the problem was the division of the equality constraints by the order of the magnitude of the maximum external force.

Reliability

The only statistical parameters considered for the probability of failure evaluation were the strength of the concrete and the external loads. For that reason the failure function is a linear function of these two basic variables and the reliability index is calculated using the formula referred in Chapter 4. Whenever there was a change of the design variables the value of the ultimate moment for

each element was calculated in the subroutine VALMU, the maximum element moment was evaluated in subroutine ELEY and the element reliability index is determined.

Subroutine VALMU calculates the ultimate moment and ultimate curvature for each configuration of the element cross section using the assumptions and correspondent formulas presented in Chapter 3. The ultimate curvature is limited to a maximum of four times the yielding curvature due to serviceability reasons. The curvatures of sections beyond this point are so high that the corresponding deformations will transform any regular frame to an unserviceable structure. Furthermore for curvatures above these values the strain hardening of high strength steel would have to be considered. This upper bound for the ultimate curvature is also a common value used in design of structures with dynamic loads.

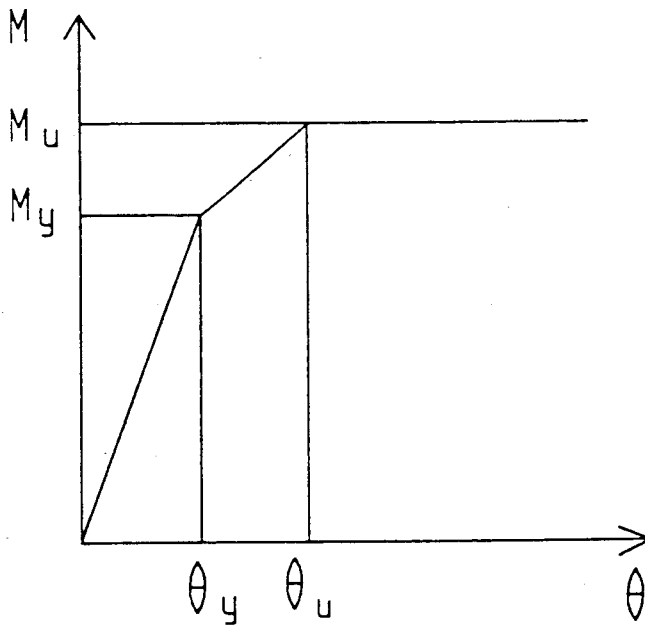
Recovery of the element moments at both ends is performed in the subroutine ELEY as a function of the global displacements and the current spring characteristics of the element. The element stiffness matrix considered in this evaluation results from the condensation of the element elastic stiffness matrix and the spring stiffness. During the optimization process the stiffness characteristics, including the secant spring stiffness, are those defined in the previous iteration.

All fundamental mechanisms of the initial structure are determined in subroutine MECSYS at the beginning of the

optimization process using the methodology described in Chapter 5. The sets of relative displacements corresponding to the fundamental elasto-plastic mechanisms are divided into two parts. The first one corresponds to the virtual displacements of the external loads and the second one to the added global degrees of freedom. Since the first set corresponds to global displacements, the joint mechanisms have to be transformed into element rotations. Since usually there are no concentrated moments applied at the nodes these mechanisms do not occur by themselves, they are active in the linear combination with the fundamental failure mechanisms that lead to the mechanisms with lower reliability indices.

After the optimization process is finished, subroutine SYSREL performs the evaluation of the system reliability at the mechanism level. For that purpose the material behavior is assumed plastic after the element rotation exceeds the ultimate value as illustrated in Figure 6.4. The combination with the elementary mechanisms is made in a combinatorial type process. The first mechanism is linearly combined with the remaining ones, the second mechanism with the following mechanisms and so forth until the penultimate is combined with the last one. The new mechanisms are ordered in terms of the reliability index and those that fall outside an acceptable interval are skipped from future combinations. The process is repeated until all possible combinations with fundamental mechanisms is performed. To

ASSUMED MOMENT ROTATION DIAGRAM



θ_y - Yielding Rotation
 θ_u - Ultimate Rotation
 M_u - Ultimate Moment
 M_y - Yielding Moment

Figure 6.4. Bilinear elastic-plastic diagram.

avoid possible precocious eliminations the fundamental mechanisms are ordered such that those that involve external work are placed before joint mechanisms. To facilitate the combination of the mechanisms all virtual displacements are scaled so that all hinge rotations are unitary. At the end, if the mechanism with lower reliability index is not acceptable, the elements with hinges that belong to this failure mechanism have their required element reliability indices increased. The optimization process is restarted with these indices modified by the same percentage of the system reliability violation.

CHAPTER 7

EXAMPLES

Introduction

Examples used to test the program versions are described and the conditions for the tests are presented. A one bay frame was used to debug the program during its development and enhancement. For result comparison, an available study in literature of a frame optimized using limit equilibrium theory was used to compare results obtained from the versions of the present optimization program. The program was finally tested with a realistic frame and loading configuration corresponding to an average building frame.

Three versions of the element stiffness were implemented and tested. The first one considered the material behavior as elastic and that provides a high value for the stiffness of the rotation springs. The second formulation provided a spring stiffness equal to the ratio between the element yielding moment and the yielding rotation. The last version used the secant spring stiffness. The final version was implemented both with the

Hooke and Jeeves and the Generalized Reduced Gradient methods. The relevant results of these examples are presented in Tables 7.1 through 7.8.

Result Verification

Validation of results from the three types of frames described above was accomplished with a common strategy implemented at three levels. These were element reliability, compatibility with element moment rotation diagram, and global structure equilibrium and compatibility. Control of results was extensively performed for the debug frame and carefully administered in the other two cases.

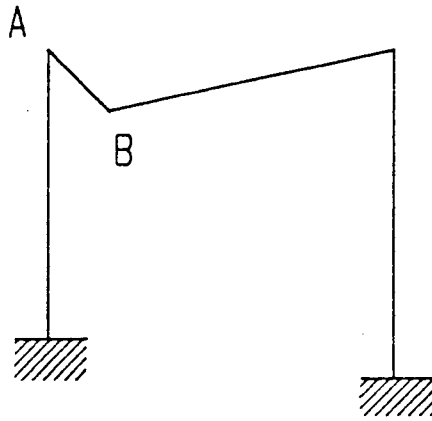
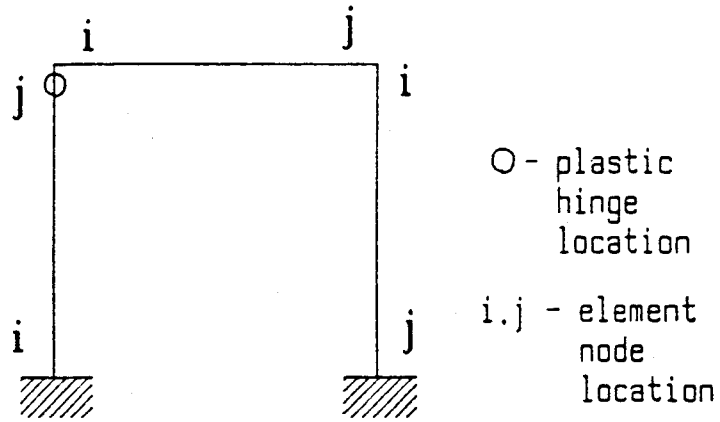
To evaluate the element reliability and the compatibility of the moment rotation diagram at the end or during the optimization process, a group of two programs was used. These computer programs called YIEL and ELTES are listed in Appendix C. The input data is composed of the dimensions of the cross section, the steel area reinforcement, the values of the secant stiffness of the springs, the length of the element and the global displacements of the element nodes. The output includes the element moments at the ends, the yielding and ultimate moments, the yielding and ultimate rotations, and the element reliability. These values are compared with those reported by the program results.

Global structural equilibrium and compatibility was verified using the program SSTAN (77). The program is prepared to handle linear elastic analysis. To verify the nonlinear results some extra elements were added to the initial structure simulating the nonlinear behavior. These additional elements placed at the hinge locations normal to the plane of the frame had a torsional rigidity equal to the spring secant stiffness. An example of a transformed structure used to test the accuracy of the displacements of a debug frame output is presented in Figure 7.1.

Debug Frame

The structure used to verify and evaluate the performance of the different versions of the program was a one bay rectangular frame subjected to a horizontal and a vertical load at the middle of the span. The material properties, geometric layout, initial dimensions, loads, reliability indices and other characteristics were arbitrarily selected, with no intent of creating a practical design. The global features of this frame are presented in Figure 7.2.

Results of the optimization performed using the Generalized Reduced Gradient and assuming linear behavior are presented in Table 7.1. Performance and final results were acceptable and satisfied the Kuhn-Tucker conditions.



Element AB

$$K_S = GJ/L$$

where

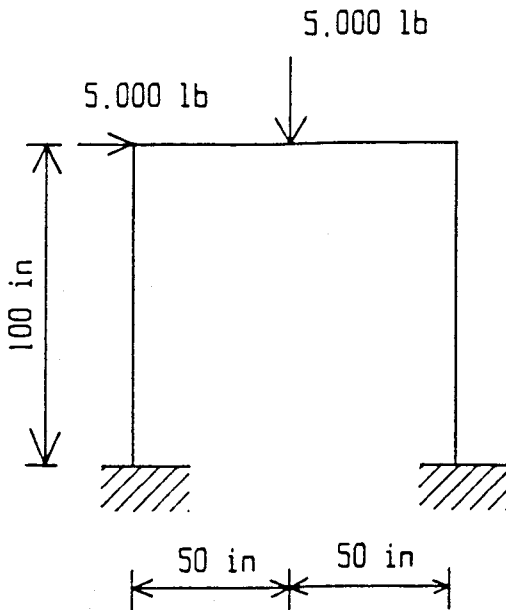
K_S - Secant spring stiffness;

G - Shear modulus;

J - Torsional moment of inertia;

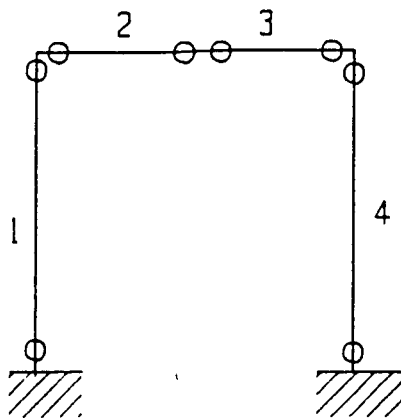
L - Element length.

Figure 7.1. Displacement verification.



Coefficients
of
variation
 $f_c = 0.15$
loads = 0.15

Materials
 $f_c = 3.000$ psi
 $f_y = 40.000$ psi



Hinge location

Figure 7.2. Debug frame.

Table 7.1. Debug frame (GRG): linear version results.

Element Section	Initial			Final			Reliability Index
	Base (in)	Height (in)	Area (in ²)	Base (in)	Height (in)	Area (in ²)	
1	5.0	10.0	1.0	2.0*	6.0*	0.20	2.0*
2	5.0	10.0	1.0	2.0*	6.0*	0.23	2.0*
3	5.0	10.0	1.0	2.0*	11.1	0.73	2.0*
4	5.0	10.0	1.0	2.0*	12.4	0.81	2.0*

* - lower bounds.

Total Initial Cost..... 18,000

Total Final Cost..... 6,890

Global Displacements

	1 (in)	2 (in)	3 (rad)	4 (in)	5 (in)	6 (rad)	7 (in)	8 (in)	9 (rad)
Initial	.50	.06	0.0	.90	.81	.07	.94	0.0	-.09
Final	.54	0.0	-.005	.53	-.087	.003	.53	0.0	-.003

The displacements satisfied the global equilibrium equations and all the constraints were satisfied.

The optimization problem with material nonlinear behavior was solved using two minimization techniques described in Chapter 6. Results of the version using Hooke and Jeeves method are listed in Table 7.2. However, evaluation of element moments did not correspond to the location of the hinges, i.e., the element moment values in some hinges were below the yielding moment value. There was no force equilibrium in some of the nodes. An extensive set of initial design points and optimization parameters were tested with negative results. For that reason the optimization technique was tentatively replaced by the Generalized Reduced Gradient.

First attempt to optimize with the Generalized Reduced Gradient assuming nonlinear material behavior, was with a secant spring stiffness equal to the ratio between the yielding moment and the yielding rotation. The option of using this spring stiffness had the advantage of avoiding the oscillation of the spring stiffness between the rigid and lower values. The implementation resulting from this choice was named yielding stiffness. Although providing incorrect displacements as the spring stiffness values did not represent the true material behavior, the yielding stiffness would model a situation somewhere between the linear and the nonlinear material behavior. The results are presented in Table 7.3. After adequate analysis it was

Table 7.2. Debug frame: Augmented Lagrangian version.

Element Section	Initial			Final			Reliability Index
	Base (in)	Height (in)	Area (in ²)	Base (in)	Height (in)	Area (in ²)	
1	5.0	10.0	1.0	2.0*	6.0*	0.06*	4.4
2	5.0	10.0	1.0	2.0*	6.8	0.07*	4.3
3	5.0	10.0	1.0	2.1*	8.4	0.09*	5.2
4	5.0	10.0	1.0	2.0*	9.4	0.09*	5.7

* - lower bounds.

Total Initial Cost..... 18,000

Total Final Cost..... 4,872

Global Displacements

	1 (in)	2 (in)	3 (rad)	4 (in)	5 (in)	6 (rad)	7 (in)	8 (in)	9 (rad)
Initial	.23	0.0	-.002	.23	-.041	0.0	.23	-.003	0.0
Final	1.1*	0.0	-.009	1.1*	-.126	.004	1.1*	-.008	-.006

Secant Spring Stiffness
(lb.in/rad)

Hinge Number	1, 3, 4, 5, 6, 7, 8	2
Spring Stiffness	10x10 ³⁰	6.7x10 ⁸

Table 7.3.Debug frame (GRG): yielding stiffness results.

Element Section	Initial			Final			Reliability Index
	Base (in)	Height (in)	Area (in ²)	Base (in)	Height (in)	Area (in ²)	
1	2.0	6.0	0.41	2.0*	6.0*	0.21	0.1*
2	2.0	6.0	0.41	2.0*	6.0*	0.21	0.1*
3	2.0	9.74	0.64	2.0*	9.75	0.64	0.1*
4	2.0	10.9	0.71	2.0*	10.9	0.71	0.1*

* - lower bounds.

Total Initial Cost..... 6,599

Total Final Cost..... 6,296

Global Displacements

	1 (in)	2 (in)	3 (rad)	4 (in)	5 (in)	6 (rad)	7 (in)	8 (in)	9 (rad)
Initial	.77	0.0	-.008	.77	-.115	.004	.76	-.007	-.004
Final	.82	0.0	-.008	.81	-.118	.004	.81	-.006	-.004

Yielding Spring Stiffness
(lb.in/rad)

Hinge Number	1	2	3	4	5	6	7	8
Spring Stiffness (x10 ⁷)	6.9	6.9	49	49	83	83	43	43

verified that the equilibrium of the moments at the nodes and the compliance with the moment rotation diagrams were satisfied. While this solution converged in some cases, it did not in others. There seemed to be no general pattern to the problem.

The next step was to test the formulation using the secant stiffness spring values obtained from the element moment rotation diagram. The results of one of these problems are presented in Table 7.4. In this case, a situation similar to the Hooke and Jeeves minimization was encountered. The node equilibrium and the element moment rotation diagram were not in accordance with the final values. To illustrate these discrepancies of the final results, the values of the yielding moments, ultimate moments, and moments at the nodes recovered using the condensed stiffness matrix are shown in Table 7.5.

To improve the performance of the optimization with a nonlinear material behavior, the use of better estimates of starting design points was tried. For that purpose the frame was optimized in the linear version having the ultimate moment set as the yielding moment, i.e, no element was allowed to yield. These solutions of the linear behavior would theoretically provide the best starting points. The optimization problem was thus transformed to a two stage process: a linear solution with the elements close to the yielding situation followed by a nonlinear optimization having as starting values the results of the

Table 7.4. Debug frame (GRG): secant stiffness results.

Element Section	Initial			Final			Reliability Index
	Base (in)	Height (in)	Area (in ²)	Base (in)	Height (in)	Area (in ²)	
1	2.0	6.0	0.41	2.0*	6.0*	0.38	0.1*
2	2.0	6.0	0.41	2.0*	6.0*	0.25	0.7
3	2.0	9.74	0.64	2.0*	9.81	0.64	0.1*
4	2.0	10.9	0.71	2.0*	10.9	0.72	0.1*

* - lower bounds.

Total Initial Cost..... 6,599

Total Final Cost..... 6,504

Global Displacements

	1 (in)	2 (in)	3 (rad)	4 (in)	5 (in)	6 (rad)	7 (in)	8 (in)	9 (rad)
Initial	.77	0.0	-.008	.77	-.115	.004	.76	-.007	-.004
Final	.88	0.0	-.007	.87	-.152	.005	.87	-.007	-.004

Secant Spring Stiffness
(lb.in/rad)

Hinge number	1	2	3	4	5	6	7	8
Spring Stiffness (x10 ⁷)	0.9	0.9	29	29	84	84	58	58

Table 7.5. Debug frame: element moments.

Element	Yielding (lb.in)	Ultimate (lb.in)
1	3.43e4	6.27e4
2	2.00e4	4.28e4
3	14.9e4	20.3e4
4	22.5e4	28.5e4

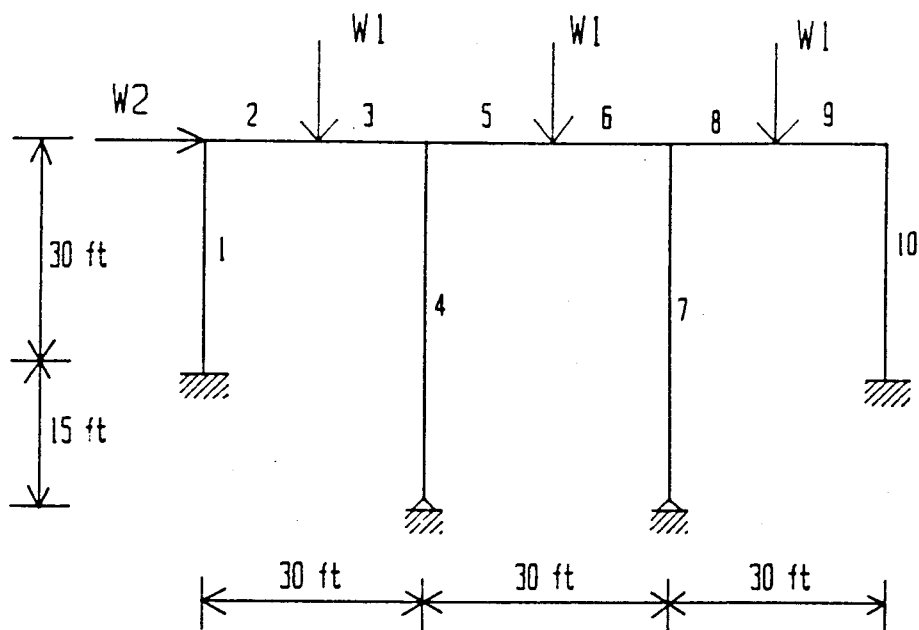
Element	Node i (lb.in)	Node j (lb.in)
1	-4.15e4	-5.46e4
2	1.21e4	3.27e4
3	-3.05e4	20.3e4
4	21.5e4	26.2e4

first stage. The results were, however, similar to those attained before.

Compared Frame

To complement the program testing with the one bay frame and evaluate the capacity of the program to obtain accurate and exact optimal solutions, a frame that was optimized using the theory of the Optimal Limit Design was also tested (78). This published example had the advantage of considering the nonlinear behavior of reinforced concrete elements at ultimate capacity. The resulting moment redistribution at the nodes was limited to values assuring a certain serviceability. The definition of the frame and respective loads are presented in Figure 7.3.

This choice presented some disadvantages. Optimization was carried out with the design variables as the steel reinforcement areas, the elements in the reference were singly reinforced, there were no reliability limits imposed, and the moment redistribution was limited to a maximum of 30%. It follows a similar approach to the system reliability in obtaining the optimal redistributed moment diagram when evaluating the performance of the several ultimate failure mechanisms. The failure mechanism with lower external work is defined and in Table 7.6 the moment redistribution coefficients, factored external moments and redistributed moments are presented. The steel reinforcement areas for the redistributed moments, or



Original Sections

ELEMENT	DIMENSIONS
1,4,7,10	18inx18in
others	11inx11in

LOADS	DEAD	LIVE
W1	7.5K	15K
W2	-	5K

EQUIVALENT LOADS
W1 - 40K W2 - 10K

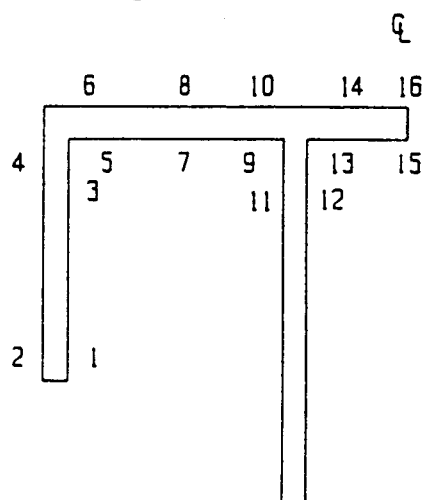


Figure 7.3. Compared frame.

Table 7.6. Compared frame: initial steel area reinforcement.

Section	c_j	M_u (k.ft) $1.8M_l + 1.5M_d$	$c_j M_u$ (k.ft)	Steel Area (in ²)	Comparison with elements
1	1.0	114	114	-----	
2	1.0	61	61		1
3	0.7	55	39		or 10
4	0.9	136	122	2.6	
5	0.7	55	39	-----	2
6	0.9	136	122	2.6	or 9
7	1.0	196	196	3.5	2 or 3
8	0.7	25	18		or 8 or 9
9	0.7	33	23	-----	3
10	1.0	203	203	3.7	or 8
11	0.7	44	31	-----	4
12	0.7	51	36	0.7	or 7
13	0.7	11	8	-----	
14	1.0	173	173	3.1	5
15	1.0	169	169		or 6
16	0.7	8	6	-----	

Legend:

----- - indicates separation of groups of element sections in the original study comparable with element sections in present research frame;

c_j - redistribution coefficient;
 M_u - ultimate section moment;
 M_l - live load moment;
 M_d - dead load moment.

optimal final moment diagram, are also presented in Table 7.6 and were calculated in accordance with ACI 318-63. This code was used in the original published work to define factored loads and ultimate section capacities.

The frame was initially tested assuming the linear material behavior and the final results are presented in Table 7.7. To simulate the same requirements the frame was optimized using an equivalent set of loads. This set resulted from the multiplication of the service loads by the correspondent load factors and by the inverse of the strength reducing factors prescribed in ACI 318-63. Final values are very close to those obtained with the Optimal Limit Design results and with a total lower cost.

Solution was then attempted with the yielding stiffness and the secant stiffness versions. Results fell in two unacceptable categories. The first category included the results with some optimization but no convergence of the equilibrium constraints. The other had very slight decrease of objective function and verification of equality and inequality constraints. Several starting points were tried, including the design points obtained from the linear solution, but no practical results were obtained. Convergence and oscillation were again the key problems.

Table 7.7. Compared frame results.

Element	1	2	3	4	5
Steel Area (in ²)					
Initial	3.0	3.0	3.0	3.0	3.0
Final	0.8	2.9	3.0	0.6	2.5
Reliability Index	0.0*	0.0*	0.0*	0.0*	0.0*
* - lower bounds.					

Element	6	7	8	9	10
Steel Area (in ²)					
Initial	3.0	3.0	3.0	3.0	3.0
Final	2.4	0.3	2.5	2.5	2.5
Reliability Index	0.0*	0.0*	0.0*	0.0*	0.0*
* - lower bounds.					

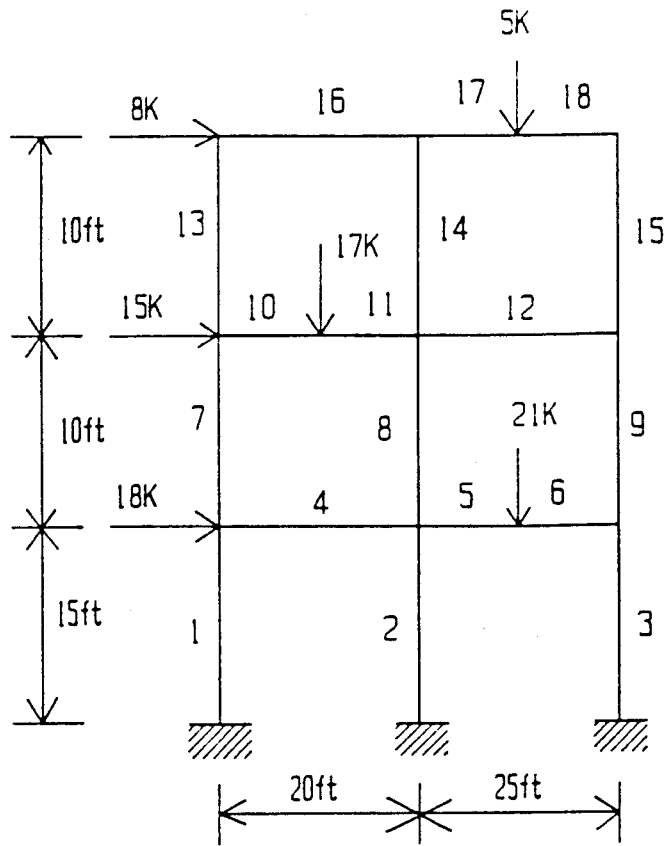
Initial Steel Cost..... 86,400
 Final Steel Cost..... 54,720
 OLD Steel Cost..... 63,360

Building Frame

To evaluate the performance of the program for a common practical design a typical rectangular building frame with two spans and three stories high was defined. Lateral and vertical loads were calculated using the Standard Building Code. Definition of the frame geometry, horizontal loads, vertical loads, material properties and floor plan are presented in Figure 7.4.

Vertical loads were applied at the midspan of each beam. Values were equivalent to the distributed loads along the adjacent slabs since this formulation does not handle loading along the element. The pattern chosen for the distribution of the vertical loads aims to create maximum moments in the elements. For this reason the loading combination includes the wind loads.

The major frame was analyzed using the linear version of the Generalized Reduced Gradient method and the results are summarized in Table 7.8. Kuhn-Tucker conditions were verified and the final dimensions of the cross sections corresponded to the lower bounds. The exception to this last conclusion happened whenever the steel reinforcement attained the upper limit. Testing of the nonlinear versions, both with the yielding and the secant spring stiffness formulations, provided no acceptable results in a similar manner to that observed when testing the compared frame.



FLOOR PLAN

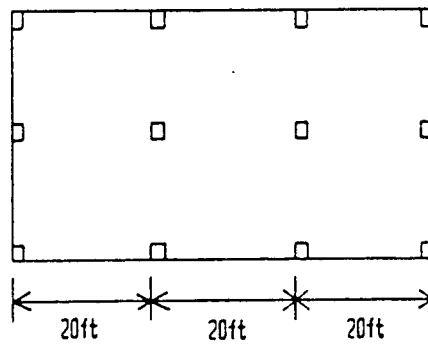


Figure 7.4. Building frame.

Table 7.8. Building frame results.

Element	1	2	3	4	5	6	7	8	9
Base (in)									
Initial	10	10	10	10	10	10	10	10	10
Final	8*	8*	8*	8*	8*	8*	8*	8*	8*
Height (in)									
Initial	25	25	25	30	30	30	25	25	25
Final	12*	12*	25	16*	16*	20	12*	12*	12*
Area (in ²)									
Initial	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Final	1.7	1.6	5.5	1.8	1.6	4.3	1.2	2.7	1.2
Reliability Index	3.0*	3.0*	3.0*	3.0*	3.0*	3.0*	3.0*	3.0*	3.0*
Element	10	11	12	13	14	15	16	17	18
Base (in)									
Initial	10	10	10	10	10	10	10	10	10
Final	8*	8*	8*	8*	8*	8*	8*	8*	8*
Height (in)									
Initial	25	25	25	30	30	30	25	25	25
Final	16*	16*	16*	12*	12*	12*	16*	16*	16*
Area (in ²)									
Initial	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Final	1.7	2.2	1.1	.57	1.2	1.1	.69	.71	.77
Reliability Index	3.0*	3.0*	3.0*	3.0*	3.0*	3.0	3.0*	3.0*	3.0*

* - lower bounds.

Total Initial Cost..... 515,400

Total Final Cost..... 401,274

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

Linear Material Behavior

This optimization approach with reliability constraints proved to be a valuable formulation for reinforced concrete frames with linear material behavior and static loading. The formulation addresses a universal procedure for obtaining optimal solutions independently of the local code restrictions. The choices for the element and system reliability indices are made by the user and may be chosen as a function of the particular problem conditions.

The approach depends on initial choices and these have a significant effect on final results. These effects can be overcome by careful evaluation and planning by the designer. Most relevant aspects are the choice of adequate element and system reliability indices, the definition of the material and of the load statistical values and the displacement limits. Solutions provided by the current approach are not definitive designs, since important aspects like axial forces and shear forces are not included.

The results presented showed a perfect convergence, even when the initial displacements were not those corresponding to the starting physical properties. The Kuhn-Tucker conditions were always verified unless the lower bounds were active, as in the case of the building frame. This implied that at least a local optimum was obtained. For instance, a good indication of the quality of the program performance was that in each case, the variables representing the element bases always converged to the lower bound. Another particular aspect of the program capabilities was that at the end of the optimization the displacement variables were always in the set of basic variables of the Generalized Reduced Gradient method. This meant that no improvement could be extracted from the objective function, except iterating on the equilibrium equations.

Integration of displacements in the set of design variables was a valid option for optimization with reliability considerations. Element reliability constraints were always active unless there were conflicting lower bounds. A good compromise was established between the optimization and safety requirements. System reliability was also satisfied every time required probabilities of failure for the elements and the system were of the same order of magnitude. The method proved to be adequate for optimal predesign without code limitations.

Nonlinear Material Behavior

Tests performed with material nonlinear behavior were not completely successful. The results of the debug frame showed optimized solutions, mainly if the two stage procedure was followed. However, as shown in Table 7.6 there was no complete node equilibrium. For the other types of frames, independent of the technique and initial values chosen, the results showed that simultaneous equilibrium convergence and optimization did not occur. In certain cases with these types of frames, there was satisfaction of the constraints and little improvement of the objective function. In other cases, the opposite results were obtained.

The most probable reason for these failures is attributed to the errors in the evaluation of the secant spring stiffness. The element forces and the global displacements are related to these values. On the other hand, the changes in the element properties during the optimization process create severe oscillations of the secant spring stiffness values. The values of the spring stiffness parameters oscillate abruptly between 10^{30} to 10^{10} , approximately, when the moment exceeds the yielding threshold. Also, after yielding, the spring stiffness values oscillate between values of different order of magnitude: the yielding stiffness and the ultimate stiffness. The nonlinear analysis is a path dependent event

and using a secant approach relies upon the fact the exact secant spring stiffness value is obtained. There are certain approximations in the determination of the yielding and ultimate rotations, that define the moment rotation diagram from which the secant stiffness is evaluated. All these instabilities and approximations may create the lack of convergence that the results have shown.

Future Work

A good approach to improve the adequacy of the formulation assuming linear material behavior would be the determination of the proper values for the mean and standard deviation values of the external loads and concrete strength. Presently, there is a lack of information to allow a practical choice of these parameters for each particular design situation. More research should be done to examine the influence of including other statistical parameters such as the cross section dimensions, position of reinforcing steel, steel strength and load characteristics.

Addition of other element effects will transform this formulation into a more complete optimization package. The main element force to be considered is the axial force that is decisive for column design. This will transform the system reliability evaluation and the element reliability constraints. Fundamental failure mechanisms will include axial failures coupled with flexural failures and there will

be additional element degrees of freedom in failure mechanism sets. At the element level, the element failure constraint would be replaced by a set of constraints concerning also the axial failure and the interaction of flexural and axial forces. Shear failure, important in reinforced concrete elements, could also be added in a similar fashion.

In the problem involving nonlinear material behavior some alterations could provide a better performance in the nonlinear optimization. These include the use of a mixed approach of the integrated and the cycling formulation, similar to that used in the Hooke and Jeeves version. A possible improvement is the inclusion of an intermediate stage where the solution for the exact displacements would be calculated whenever the absolute violation of the equality constraints exceeds an upper limit. This mixed approach could improve the efficiency of this approach since good displacements are essential for the definition of the correct global and element nonlinear behavior.

Another possible improvement is the use of a different model for the nonlinear reinforced concrete element. The substitution of the one-component model by a model of an element partitioned in several discrete elements. These discrete elements, each with linear stiffness characteristics, defined by the global element nonlinear properties, would provide better accuracy for the element behavior. This solution has the disadvantage of increasing

substantially the size of the problem. However, the benefits of this change could be significant.

Changing the nonlinear analysis method from secant stiffness approach to a tangent stiffness approach could be another solution to the lack of convergence. In this case a two stage process would be adopted. The first would consist of a linear optimization up to the formation of a hinge followed by a phase with a sequence of incremental loading and optimization procedures until convergence was obtained.

APPENDIX A
AUGMENTED LAGRANGIAN SUBROUTINES

```

program princi
implicit double precision (a-h,o-z)
character*40 title
dimension x(100),r(60),cl(60),cos1(60),cos2(60),xol(100),
* clah(80),clag(80),lm(6,50),vag(100),vah(80),
* ch(80),cg(80),xo(100),nol(80),no2(80),jm(6,80),alp(80),
* xc(80),yc(80),jdir(3),glk(80,80),d(1000),vinv(100),
* vahk(80),grad(100),xu(100),xl(100),xl(100),x2(100),vaho(80),
* vago(80),a(80,80),b(80,80),vahold(100),vjac(100,100),
* c(80),cm(80,80),qa(80,80),q(80,80),am(80,80),bl(80,80),
* theta(100,80),rv(80,100),beta(80),vmu(80),cvmu(80),
* cvload(80),become(100),lc(100),thesum(80),thesuml(80),
* dispsum(60),ni(60),nucomb(60),lct(100),bfal(100,100)
common /parr/ decfc,fcinc,cv,alpl,ec,rp,fc,es,ecm,relind
common /pari/ iter,numcy,niter,ga,igh,iqg,n,ntot,iqgn
common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
open ( 8,file='finres',form='formatted' )
rewind 8
open ( 9,file='data',form='formatted' )
rewind 9
c*****
c                               name of problem
c*****
  read ( 9,191 ) title
c*****
c                               # elements and # joints
c*****
  read (9,*) n, nj
c*****
c                               nodes per element
c*****
  do 100 i = 1,n
    read (9,*) nol(i), no2(i)
100  continue
c*****
c                               initialize jm matrix
c*****
  do 200 kk = 1,nj
    jm ( 1,kk ) = 1
    jm ( 2,kk ) = 2
    jm ( 3,kk ) = 3
200  continue
c*****
c                               support conditions and coordinates
c*****
  do 300 j = 1,nj
    read ( 9,* ) jdir(1),jdir(2),jdir(3),xc(j),yc(j)
    do 350 i =1,3
      if (jdir(i).gt.0) then
        jm (i,j) = 0
      endif
350  continue
300  continue
c*****

```

```

c          global degrees of freedom
c*****
      iqh=0
      do 500 j=1,nj
        do 500 l = 1,3
          if (jm(l,j).ne.0) then
            iqh=iqh+1
            jm (l,j)=iqh
          endif
        500 continue
        iqgn = iqh+n
        iqq = iqh
        ntot = iqgn+2*n
c*****
c          input data
c*****
      call inputd (cl,cos1,cos2,iqh,jdir,jm,lm,n,nj,nol,
*               no2,r,xc,yc)
c*****
c          reinforced concrete
c*****
      read(9,*)fc,fy,co
      ec=57000*sqrt(fc)
      vn=29e6/ec
      epsy=fy/29e6
      do 987 ijh=1,n
        vksi(ijh)=10e30
        vksj(ijh)=10e30
987      continue
      read(9,*)es,ecm
c*****
c          reinforcing steel guess
c*****
      n3=n*3
      read(9,*)(x(i),i=3,n3,3)
c*****
c          constraint values
c*****
      call constr (iqg,d)
c*****
c          determine bandwidth
c*****
      mband = 0
      do 450 k = 1,n
        do 450 i = 1,6
          if ( lm(i,k).eq.0) go to 450
          do 440 j = i,6
            if (lm(j,k).eq.0) go to 440
            max = abs(lm(i,k) - lm(j,k)) + 1
            if (max.gt.mband) mband=max
          440 continue
        450 continue
c*****
c          optimization parameters and initial guesses
c*****
      call parame(toll,x,n21,glk,mband,cl,cos1,cos2,lm,

```

```

*               r,delta,alpha,numec,rph)
c*****
c               elementary mechanisms
c*****
      call mecsys(n,igh,cl,cos1,cos2,lm,numec,rv,theta,rv)
c*****
c               coefficients of variation
c*****
      do 156 i=1,n
      read(9,*)cvmu(i)
156  continue
      do 157 i=1,igh
      read(9,*)cvload(i)
157  continue
c*****
c               lower bounds
c*****
      romin=200./fy
      read(9,*)x11,x12
c*****
c               interval for generation of mechanisms
c*****
      read(9,*)epsilon
c*****
c               write input data
c*****
      write (8,190) title
      write (8,110)
      write (8,130) n
      write (8,140) igh
      write (8,150) igg
      write (8,170) fc,fy
      write (8,240)
      do 501 k = 1,igh
      write (8,250) k, r(k)
501  continue
      write (8,260)
      do 601 k = 1,igg
      write (8,270) k, d(k)
601  continue
      write (8,351)
      do 701 k = 1,n
      na = 3 * k
      ne = na - 1
      no = ne - 1
      write (8,360) k, cl(k), x(na), x(no), x(ne)
701  continue
      write (8,650) rp
      write (8,760) ga
      write (8,860)
      do 900 i=n21,ntot
      k=i-3*n
      write (8,870)k,x(i)
900  continue
c*****
c               data initialization

```

```

C*****
      call datini (clah,clag,ch,cg)
C*****
C      subroutine optimization
C*****
      call optimi (vlag,r,x,cl,cos1,cos2,lm,d,clah,xol,
*   vag,toll,clag,vah,ch,cg,xo,vahk,grad,xu,x1,x1,x2,
*   delta,alpha,alp,vaho,vago,vn,co,epsy,fy,ast,beta,theta,
*   numec,vmu,cvmu,rv,cvload,become,lc,thesum,thesum1,
*   dispsum,ni,nucomb,lct,xl1,xl2,romin,rph,grad,vahold,
*   vjac,vinv,bfal,epsilo)
C*****
C      write final data
C*****
      write (8,880)
      write (8,890) iter
      write (8,910) vlag
      write (8,840)
      do 1000 k = 1,n
      na = 3 * k
      ne = na - 1
      no = ne - 1
      write (8,851) k, x(no), x(ne), x(na)
1000  continue
      write (8,960)
      do 1151 k = n21, ntot
      i = k - 3*n
      write (8,970) i, x(k)
1151  continue
      write (8,920)
      do 1101 k = 1,iqh
      write (8,930) k, vah(k)
1101  continue
      write (8,940)
      do 1200 k = 1,iqg
      write (8,950) k, vag(k)
1200  continue
      write (8,944)
      do 1211 k=iqg+1,iqgn
      write(8,946)k-iqh,beta(k-iqh)
1211  continue
      write(8,945)
      do 1241 k=1,n
      write(8,947)k,vksi(k),vksj(k)
1241  continue
      write(8,*)
      write(8,*)'          LAGRANGIAN MULTIPLIERS EQUALITIES'
      write(8,*)
      do 1277 i=1,iqh
      write(8,966)clah(i)
1277  continue
      write(8,*)
      write(8,*)'          LAGRANGIAN MULTIPLIERS INEQUALITIES'
      write(8,*)
      do 1278 i=1,iqh+n
      write(8,966)clag(i)

```

1278 continue

```

*****
c                                     output format
c*****
110  format ( //,10x,' ***** initial values *****',/ )
120  format ( a25 )
130  format ( /,10x,'number of elements = ',i3 )
140  format ( /,10x,'number of equality constraints = ',i3 )
150  format ( /,10x,'number of inequality constraints = ',i3 )
160  format ( /,10x,'number of iterations per cycle = ',i5 )
170  format ( /,10x,'fconcrete = ',e14.8,4x,'fsteel = ',e14.8)
180  format ( /,10x,'number of global degrees of freedom
    *      = ',i2 )
190  format ( ///,10x,a25,/// )
191  format (a)
240  format(///,10x,'global degree of freedom',10x,
    *      'external force')
250  format ( /,20x,i2,22x,e14.8 )
260  format ( ///,10x,'global degree of freedom',5x,
    *      'displacement constraint' )
270  format ( /,20x,i3,19x,e14.8 )
351  format ( ///,'element',8x,'length',10x,'steel',12x,
    *      'base',12x,'height')
353  format ( 18x,i3,6x,e14.8,/)
360  format ( /,i3,6x,e14.8,3x,e14.8,4x,e14.8,4x,e14.8 )
370  format ( /,10x,i3,14x,3i4,3x,3i4 )
380  format(///,10x,'location matrix for global degrees
    *      of freedom' )
390  format ( /,10x,' element ',10x,' node i',10x,'node j' )
640  format ( /,10x,'maximum number of cycles = ',i3 )
650  format ( /,10x,'penalty factor = ',e14.8 )
730  format ( /,10x,'factor of increase = ',e14.8 )
740  format ( /,10x,'factor of decrease = ',e14.8 )
760  format ( /,10x,'penalty factor multiplier = ',e14.8 )
840  format ( /,12x,'element',11x,'base',17x,'height',10x,
    *      'steel')
851  format ( /,15x,i2,8x,e14.8,8x,e14.8,5x,e14.8)
860  format ( /,10x,'global degree of freedom',5x,
    *      'initial guess' )
870  format ( /,20x,i2,18x,e14.8 )
880  format ( ///,10x,'***** final values *****',/// )
890  format ( /,10x,'number of iterations = ',i3 )
910  format ( /,10x,'value of lagrangian function = ',e14.8 )
920  format ( /,10x,'equality',17x,'final value' )
930  format ( /,12x,i3,17x,e14.8 )
940  format ( /,10x,'inequality',13x,'final value')
944  format ( /,5x,'element reliability',9x,'final value')
945  format (/,5x,'element',5x,'spring i',5x,'spring j')
946  format ( /,11x,i4,17x,e14.8)
947  format (/,8x,i5,8x,e14.8,3x,e14.8)
950  format ( /,12x,i3,17x,e14.8 )
960  format ( /,10x,'displacement',14x,'final value' )
966  format (/e14.8)
970  format ( /,12x,i3,17x,e14.8 )
stop
end

```

```

=====
=====

      subroutine comcon(aste,fy,es,d,b,co,epsy,ecm,fc,vmy,phiy)
      implicit double precision (a-h,o-z)
      kll=0
      node=0
      epso=0.002
C*****
C
C          EXC - CONCRETE STRAIN
C          EPCS - COMPRESSIVE STEEL STRAIN
C          EPSY - YIELD STRAIN
C
C          FIRST VALUE FOR A
C*****
      al=d/2.
      exc=al*epsy/(d-al)
      epcs=exc*(al-co)/al
      t=fy*aste
      cs=epcs*es
      eces=exc/epso
      alpha=ecm-ecm*ecm/3.
      cc=alpha*fc*b*al
      res1=cc+cs-t
C*****
C          SECOND VALUE FOR A
C*****
      a2=0.25*d
      exc=a2*epsy/(d-a2)
      epcs=exc*(a2-co)/a2
      cc=fc*a2*alpha*b
      eces=exc/epso
      cs=epcs*es
      res2=cc+cs-t
C*****
C          NEWTON ITERATION
C*****
      100  a=a2-res2*(a2-al)/(res2-res1)
          exc=a*epsy/(d-a)
          if(exc.gt.epso) go to 200
C*****
C          PARABOLIC SHAPE
C*****
      epcs=exc*(a-co)/a
      eces=exc/epso
      alpha=ecm-ecm*ecm/3.
      cc=fc*alpha*b*a
      cs=epcs*es
      res=cc+cs-t
      control=0.0001*b*d*fc
      if (abs(res).gt.control) then
          al=a2
          a2=a

```



```

        res1=res2
        res2=res
        kll=kll+1
        if(kll.gt.100)then
            stop
        endif
        go to 100
    endif
    gama=1.-(8.*epso-3.*exc)/(12.*epso-4.*exc)
    arm=d-gama*a
    vmy=cc*arm+epcs*es*aste*(d-co)
    phiy=epsy/(d-a)
    return
C*****
C                                CONCRETE STRAIN > EPSO
C*****
200  if(exc.gt.0.004) exc=0.004
      x1=epso*a/exc
      ccl=fc*x1*2./3.*b
      gama=3.6*exc*exc-200.*exc*exc*exc-0.0000128
      gama=gama/((exc-epso)*(7.2*exc-300*exc*exc-0.0132))-1.
      alpha=exc-50.*exc*exc+100.*exc*epso-0.0022
      alpha=alpha/(exc-epso)
      cc2=alpha*fc*(a-x1)*b
      epcs=exc*(d-co)/a
      t=fy*aste
      cs=epcs*es
      cc=ccl+cc2
      res=cc+cs-t
      control=0.0001*b*d*fc
      if (abs(res).gt.control) then
          a1=a2
          a2=a
          res1=res2
          res2=res
      kll=kll+1
      if(kll.gt.100)then
          stop
      endif
      go to 100
    endif
    arm1=d-a+2./3.*x1
    arm2=d-gama*(a-x1)
    vmy=ccl*arm1+epcs*es*(d-co)*aste+cc2*arm2
    phiy=epsy/(d-a)
    return
end

```

=====

```

subroutine eley(ec,tinert,cl,vki,vkj,u2,u3,u5,u6,fo3,fo6)
implicit double precision(a-h,o-z)
ei=ec*tinert
w=cl/(3.*ei)+1./vki

```

```

y=c1/(3.*ei)+1./vkj
z=-c1/(6.*ei)
det=w*y-z*z
a=y/det
b=-z/det
c=b
d=w/det
fo3=(a+b)/c1*u2+a*u3-(a+b)/c1*u5+b*u6
fo6=(c+d)/c1*u2+c*u3-(c+d)/c1*u5+d*u6
return
end

```

```

=====
=====
=====

```

```

      subroutine equcon (x,n,c1,lm,cos1,cos2,fc,ec,vn,co,epsy,
*      fy,ntot,iqh,vah,r,vahk,es,ecm,beta,cvmu,cvload,kl,
*      vmu)
      implicit double precision (a-h,o-z)
      dimension lm(6,n),cos1(n),cos2(n),vmu(n)
      dimension c1(n),x(ntot),vah(iqh),r(iqh)
      dimension vahk(iqh),beta(n),cvmu(n),cvload(iqh)
      common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
      do 150 k = 1,iqh
         vahk(k)=0.0
150    continue
C*****
C      GLOBAL DISPLACEMENTS PER ELEMENT
C*****
      n3=n+n+n
      do 100 kel = 1,n
         do 123 ipo=1,6
            u(ipo)=0.0
123        continue
            sigma2=0.0
            do 200 i = 1,6
               m=lm(i,kel)
               if (m.eq.0) go to 200
               if(cvload(m).gt.sigma2)sigma2=cvload(m)
               l = n3+m
               u(i) = x(l)
200        continue
            sigmal=cvmu(kel)
            c1=cos1(kel)
            c2=cos2(kel)
            d2=-c2*u(1)+c1*u(2)
            d3=u(3)
            d5=-c2*u(4)+c1*u(5)
            d6=u(6)
C*****
C      ELEMENT FORCES
C*****

```

```

45      c = cl(kel)
      continue
      base = x(3*kel-2)
      height = x(3*kel-1)
      aste = x(3*kel)
      area = base * height
      tinert = area*height*height/12.0
      al = ec*tinert/(c*c*c)
      fo3 = al*(6.0*c*(d2-d5) + 2.0*c*c*(2.0*d3+d6))
      fo6 = al*(6.0*c*(d2-d5) + 2.0*c*c*(d3+2.0*d6))
C*****
C      ULTIMATE AND YIELD MOMENTS
C*****
      call mummy(base,height,aste,vn,co,epsy,ec,fc,fy,
      *      vksi(kel),vksj(kel),c,fo3,fo6,es,ecm,betak,
      *      sigmal,sigma2,tinert,kl,vmuk,vmy,ijflag)
      if(vksi(kel).lt.10e20.or.vksj(kel).lt.10e20)then
      *      call eley(ec,tinert,c,vksi(kel),vksj(kel),
      *      d2,d3,d5,d6,fo3,fo6)
      *      call mummy(base,height,aste,vn,co,epsy,ec,fc,fy,
      *      vksi(kel),vksj(kel),c,fo3,fo6,es,ecm,betak,
      *      sigmal,sigma2,tinert,kl,vmuk,vmy,ijflag)
      endif
      beta(kel)=betak
      vmu(kel)=vmuk
C*****
C      GLOBAL MODIFIED STIFFNESS
C*****
      call modsti(area,ec,vksi(kel),vksj(kel),c,tinert,c1,c2)
      do 300 l = 1,6
      *      j = lm(l,kel)
      *      if (j.eq.0) go to 300
      *      do 400 ll = 1,6
      *      *      m = lm(ll,kel)
      *      *      if (m.eq.0) go to 400
      *      *      jj = n3+m
      *      *      vahk(j)=vahk(j)+ck(1,ll)*x(jj)
      *      400      continue
      *      300      continue
      *      100      continue
C*****
C      SUBTRACTION OF EXTERNAL GLOBAL FORCES
C*****
      rmax=0.01
      do 510 ilj=1,iqh
      *      if(abs(r(ilj)).gt.rmax)rmax=abs(r(ilj))
      *      510      continue
      *      do 500 kpj = 1,iqh
      *      *      if(abs(r(kpj)).lt.0.0001)then
      *      *      vah(kpj)=vahk(kpj)/rmax
      *      *      go to 500
      *      *      endif
      *      vah(kpj) = (vahk(kpj) - r(kpj))/rmax
      *      500      continue
      *      return
      *      end

```

```

=====
=====

      subroutine hoojee(tvah,vlag,r,vah,vag,x,cl,cos1,cos2,
*         lm,d,clah,clag,ch,cg,alp,xol,vahk,toll,vn,co,
*         epsy,fy,ast,beta,cvmu,cvload,xl1,xl2,romin,vmu,rph,
*         grad,vahold,vjac,vinv,bfal)
      implicit double precision (a-h,o-z)
      dimension x(ntot),cl(n),cos1(n),cos2(n),lm(6,n),vahk(igh),
*         d(igh),clah(igh),clag(igh),r(igh),vah(igh),vag(igh),
*         ch(igh),cg(igh),alp(ntot),xol(ntot),beta(n),
*         cvmu(n),cvload(igh),vmu(n),grad(ntot),vahold(igh),
*         vjac(igh,igh),vinv(igh),bfal(igh,igh)
      common /parr/ decfc,fcinc,cv,alp1,ec,rp,fc,es,ecm,relind
      common /pari/ iter,numcy,niter,ga,igh,igh,n,ntot,ighn
      common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
C*****
C      INITIALIZE LAGRANGIAN FUNCTION
C*****
      k=0
      kkl=0
      n3 = n+n+n
      romax=0.85*0.85*fc*87000./((fy*(87000.+fy))
      call lagfun (vlag,tvah,r,x,cl,cos1,cos2,lm,
*         d,clah,vag,clag,vah,ch,cg,vof,vahk,vn,co,epsy,fy,beta,
*         cvmu,cvload,k,vmu,rph)
      vlago = vlag
C*****
C      LOOP ON VARIABLE # K FOR THE SPECIFIED # OF ITERATIONS
C*****
      do 200 klj = 1,niter
      do 450 k = 1,ntot
450         alp(k) = alp1
      do 150 k = 1,ntot
150         xol(k) = x(k)
      do 100 k = 1,n3
          xopt = x(k)
600         continue
          kkl=kkl+1
C*****
C      VALUE OF LAGRANGIAN FUNCTION FOR INITIAL VALUES
C*****
      500         continue
          x(k)= xopt+alp(k)*xopt
          if (k.le.n3) call lim(x,n,ntot,xl1,
*             xl2,k,romin,romax)
C*****
C      NEW VALUE OF LAG. FUNCTION
C*****
      call lagfun (vlag,tvah,r,x,cl,cos1,cos2,lm,
*         d,clah,vag,clag,vah,ch,cg,vof,vahk,vn,co,epsy,fy,beta,
*         cvmu,cvload,k,vmu,rph)
          if (vlago.gt.vlag) then
C*****

```

[illegible]

```

      call jacequ(x,n,cl,lm,cos1,cos2,fc,ec,vn,co,epsy,
*      fy,ntot,igh,vah,r,vahk,es,ecm,beta,cvmu,cvload,k,
*      vmu,vjac)
      call sol(igh,vjac,r,vinv)
      do 5890 jgo=1,igh
          x(jgo+n3)=vinv(jgo)
5890      continue
200      continue
C*****
C      REINITIALIZE VALUES
C*****
      k=0
      call lagfun (vlag,tvah,r,x,cl,cos1,cos2,lm,
*      d,clah,vag,clag,vah,ch,cg,vof,vahk,vn,co,epsy,fy,beta,
*      cvmu,cvload,k,vmu,rph)
      return
      end

=====
=====

      subroutine lagfun (vlag,tvah,r,x,cl,cos1,cos2,lm,
*      d,clah,vag,clag,vah,ch,cg,vof,vahk,vn,co,epsy,fy,beta,
*      cvmu,cvload,kl,vmu,rph)
      implicit double precision (a-h,o-z)
      dimension x(ntot),cl(n),cos1(n),cos2(n),lm(6,n),d(iqg),
*      clah(igh), clag(iqgn), vag(iqgn), r(igh), vah(igh),
*      ch(igh), cg(iqgn), vahk(igh), beta(n),
*      cvmu(n), cvload(igh),vmu(n)
      common /parr/ decfc,fcinc,cv,alpl,ec,rp,fc,es,ecm,relind
      common /pari/ iter,numcy,niter,ga,igh,iqg,n,ntot,iqgn
      common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
      tvah = 0.0
      tvag = 0.0
      rp2 = 2.0*rp
C*****
C      EQUALITY CONSTRAINTS
C*****
      call equcon (x,n,cl,lm,cos1,cos2,fc,ec,vn,co,epsy,
*      fy,ntot,igh,vah,r,vahk,es,ecm,beta,cvmu,cvload,kl,vmu)
      do 100 k=1,igh
          vc=vah(k)*ch(k)
          tvah=tvah+clah(k)*vc+rp2*vc*vc
100      continue
C*****
C      DISPLACEMENT CONSTRAINTS
C*****
      call inecon (iqg,n,ntot,vag,x,d,relind,beta)
      do 200 k=1,iqgn
          v=vag(k)*cg(k)
          z=-clag(k)/rp2
          psi=max(v,z)

```

```

                tvag=tvag+clag(k)*psi+psi*psi*rp
200    continue
C*****
C                VALUE OF LAGRANGIAN FUNCTION
C*****
    call valobf (n,ntot,vof,x,cl)
    avof = cv*vof
    vlag = avof+tvah+tvag
    return
end

```

```

=====
=====

```

```

    subroutine modsti(area,ec,vki,vkj,cl,tinert,cos1,cos2)
    implicit double precision (a-h,o-z)
    common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
C*****
C                FLEXIBILITY MATRIX (2x2)
C*****
    do 10 i=1,6
    do 20 j=1,6
    ck(i,j)=0.0
20    continue
10    continue
    x=cl/(3*ec*tinert)+1./vki
    y=cl/(3*ec*tinert)+1./vkj
    z=-cl/(6*ec*tinert)
C*****
C                INVERSION OF MATRIX
C*****
    det=x*y-z*z
    a=y/det
    b=-z/det
    c=b
    d=x/det
C*****
C                EXPANDED MATRIX (6x6)
C*****
    ck11=ec*area/cl
    ck14=-ck11
    ck41=-ck11
    ck44=ck11
    ck22=(a+b+c+d)/(cl*cl)
    ck25=-ck22
    ck52=ck25
    ck55=ck22
    ck23=(a+c)/cl
    ck53=-ck23
    ck26=(b+d)/cl
    ck56=-ck26
    ck33=a

```

```

ck36=b
ck63=c
ck66=d
ck32=(a+b)/c1
ck35=-ck32
ck62=(c+d)/c1
ck65=-ck62

```

```

C*****

```

```

C          ROTATED MATRIX

```

```

C*****

```

```

c2=cos1*cos1
s2=cos2*cos2
cs=cos1*cos2
ck(1,1)=ck11*c2+ck22*s2
ck(1,2)=ck11*cs-ck22*cs
ck(1,3)=-ck23*cos2
ck(1,4)=-ck11*c2-ck22*s2
ck(1,5)=-ck11*cs+ck22*cs
ck(1,6)=-ck26*cos2
ck(2,1)=ck11*cs-ck22*cs
ck(2,2)=ck11*s2+ck22*c2
ck(2,3)=ck23*cos1
ck(2,4)=-ck11*cs+ck22*cs
ck(2,5)=-ck11*s2-ck22*c2
ck(2,6)=ck26*cos1
ck(3,1)=-ck32*cos2
ck(3,2)=ck32*cos1
ck(3,3)=ck33
ck(3,4)=-ck(3,1)
ck(3,5)=-ck(3,2)
ck(3,6)=ck36
ck(4,1)=-ck(1,1)
ck(4,2)=-ck11*cs+ck22*cs
ck(4,3)=ck23*cos2
ck(4,4)=ck11*c2+ck22*s2
ck(4,5)=ck(2,1)
ck(4,6)=ck26*cos2
ck(5,1)=ck(2,4)
ck(5,2)=ck(2,5)
ck(5,3)=-ck(2,3)
ck(5,4)=-ck(2,4)
ck(5,5)=-ck(2,5)
ck(5,6)=-ck26*cos1
ck(6,1)=-ck62*cos2
ck(6,2)=ck62*cos1
ck(6,3)=ck36
ck(6,4)=-ck(6,1)
ck(6,5)=-ck(6,2)
ck(6,6)=ck66
return
end

```

```

=====
=====

```



```

subroutine mummy(b,h,aste,vn,co,epsy,ec,fc,fy,
* vki,vkj,clk,fo3,fo6,es,ecm,betak,sigmal,sigma2,
* tinert,kl,vmuk,vmy,ijflag)
implicit double precision (a-h,o-z)
nodel=0
node2=0
ijflag=0
d=h-co
C*****
C          EVALUATION OF YIELDING MOMENT
C*****
call comcon(aste,fy,es,d,b,co,epsy,ecm,fc,vmy,phiy)
afo3=abs(fo3)
afo6=abs(fo6)
vki=10e30
vkj=10e30
vm=max(afo3,afo6)
C*****
C          IDENTIFICATION OF HINGE NODE
C*****
if (afo3.gt.vmy) nodel=1
if (afo6.gt.vmy) node2=1
C*****
C          ULTIMATE MOMENT AND RELIABILITY
C*****
call valmu(aste,b,betak,co,d,es,epsy,fc,fy,
* phiu,sigmal,sigma2,vm,vmuk,vmy,phiy)
C*****
C          INTEGRATION OF CURVATURE
C*****
if(nodel.gt.0.or.node2.gt.0)then
  if((fo3*fo6).gt.0) then
    if(afo3.gt.afo6)then
      zero=0.000001*afo3
      if(abs(afo3-afo6).lt.zero)then
        vlp=clk
        go to 145
      endif
      vlp=(afo3-vmy)/(afo3-afo6)*clk
    endif
    if(afo3.lt.afo6)then
      zero=0.000001*afo6
      if(abs(afo3-afo6).lt.zero)then
        vlp=clk
        go to 145
      endif
      vlp=(afo6-vmy)/(afo6-afo3)*clk
    endif
  endif
  continue
  tetay = phiy*clk*0.5
  tetau = (phiu-phiy)*vlp*0.5+phiy*clk
  endif
  if((fo3*fo6).lt.0) then
    vlp=(vmuk-vmy)*clk/vmuk
    tetay=clk*phiy*0.25
    tetau=phiy*clk*0.25+(phiu-phiy)*vlp
  endif

```

```

endif
vksp=(vmuk-vmy)/(tetau-tetay)
C*****
C          SPRING VALUES
C*****
      if(node1.eq.1) then
        vki=vksp
        vkj=10.0e20*ec
      endif
      if(node2.eq.1) then
        vki=10.0e20*ec
        vkj=vksp
      endif
      if(node1.eq.1.and.node2.eq.1) then
        vki=vksp
        vkj=vksp
      endif
endif
return
end

```

```

=====
=====

```

```

      subroutine optimi(vlag,r,x,cl,cos1,cos2,lm,d,clah,xol,
*   vag,toll,clag,vah,ch,cg,xo,vahk,grad,xu,xl,x1,x2,
*   delta,alpha,alp,vaho,vago,vn,co,epsy,fy,ast,beta,theta,
*   numec,vmu,cvmu,rv,cvload,become,lc,thesum,thesuml,
*   dispsum,ni,nucomb,lct,xl1,xl2,romin,rph,grad,vahold,
*   vjac,vinv,bfal,epsilo)
      implicit double precision (a-h,o-z)
      dimension r(igh),x(ntot),cl(n),cos1(n),cos2(n),d(iggn),
*   alp(ntot),clag(iggn),lm(6,n),vag(iggn),vah(igh),
*   ch(igh),xo(ntot),clah(igh),cg(iggn),beta(n),
*   bfal(igh,igh),vahk(igh),grad(ntot),xu(ntot),xl(ntot),
*   xl(ntot),x2(ntot),xol(ntot),vaho(igh),vago(iggn),
*   ast(n),theta(2*n,numec),vmu(n),cvmu(n),rv(igh,numec),
*   cvload(igh),become(100),lc(100),thesum(n),thesuml(n),
*   dispsum(igh),ni(20),nucomb(20),lct(100),vahold(igh),
*   vjac(igh,igh),vinv(igh)
      common /parr/ decfc,fcinc,cv,alpl,ec,rp,fc,es,ecm,relind
      common /pari/ iter,numcy,niter,ga,igh,iggn,n,ntot,iggn
      common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
      iter = 0
C*****
C          initializing for scaling
C*****
      call lagfun (vlag,tvah,r,x,cl,cos1,cos2,lm,d,clah,vag,
*   clag,vah,ch,cg,vof,vahk,vn,co,epsy,fy,beta,
*   cvmu,cvload,kl,vmu,rph)
C*****

```

```

c               scaling objective function
c*****
      cv = 5./vof
c*****
c               hooke & jeeves
c*****
444      call hoojee (tvah,vlag,r,vah,vag,x,cl,cos1,cos2,lm,
*      d,clah,clag,ch,cg,alp,xol,vahk,toll,vn,co,epsy,fy,ast,
*      beta,cvmu,cvload,xl1,xl2,romin,vmu,rph,grad,vahold,
*      vjac,vinv,bfal)
      iter=iter+1
      write (8,1500) iter
1500      format ('end of loop =',i3)
c*****
c               control of maximum number of iterations
c*****
      if (iter.gt.numcy) go to 99
      rp2 = rp+rp
c*****
c               updating lagrangian mult.  equal.cons.
c*****
      do 100 k = 1,iqh
          clah(k)=clah(k)+rp2*vah(k)*ch(k)
100      continue
c*****
c               updating lagrangian mult.  ineq. cons.
c*****
      do 200 k=1,iqgn
          v=vag(k)*cg(k)
          z=-clag(k)/rp2
          psi=max(v,z)
          clag(k)=clag(k)+rp2*psi
200      continue
c*****
c               updating penalty factor
c*****
      rp=ga*rp2
      rph=ga*ga*rph
c*****
c               system reliability evaluation
c*****
      jflag=0
      call sysrel(n,numec,iqh,theta,rv,vmu,cvmu,r,
*      cvload,jflag)
      if(jflag.gt.0)go to 444
      return
      end

```

```

=====
=====

```

subroutine sol(n,a,b,c)

```

implicit double precision(a-h,o-z)
dimension a(n,n),b(n),c(n)
do 50 ik=1,n
c(ik)=b(ik)
50 continue
do 100 k=1,n
vmax=abs(a(k,k))
krow=k
do 120 kj=k+1,n
if(abs(a(kj,k)).gt.vmax)then
vmax=abs(a(kj,k))
krow=kj
endif
120 continue
if (krow.gt.k)then
do 140 jj=k,n
temp=a(krow,jj)
a(krow,jj)=a(k,jj)
a(k,jj)=temp
140 continue
temp=c(krow)
c(krow)=c(k)
c(k)=temp
akk=a(k,k)
endif
do 200 i=k+1,n
w=a(i,k)/akk
do 300 j=k+1,n
a(i,j)=a(i,j)-a(k,j)*w
300 continue
c(i)=c(i)-c(k)*w
200 continue
100 continue
do 400 k=1,n
i=n-k+1
do 600 j=i+1,n
c(i)=c(i)-a(i,j)*c(j)
600 continue
c(i)=c(i)/a(i,i)
500 continue
400 continue
return
end

```

```

=====
=====

```

```

subroutine valmu(aste,b,betak,co,d,es,epsy,fc,fy,
*   phiu,sigma1,sigma2,vm,vmul,vmy,phiy)
implicit double precision (a-h,o-z)

```

```

C*****
C                               NEUTRAL AXIS

```

```

C*****
  x=47./60.*b*fc
  y=0.004*es*aste-aste*fy
  z=-0.004*es*co*aste
  if((y*y).lt.(4.*x*z)) then
    y=sqrt(4.*x*z)
  endif
  vkd=(-y+sqrt(y*y-4.*x*z))/(2.*x)
  epcs=0.004*(vkd-co)/vkd
  if(epcs.ge.epsy) then
    epcs=epsy
  endif
C*****
C      CONCRETE FORCE IN REGION AB
C*****
  alphas=2./3.
  ccab=alphas*b*0.5*vkd*fc
C*****
C      CONCRETE FORCE IN REGION BC
C*****
  alpha2=0.9
  ccbc=alpha2*b*0.5*vkd*fc
C*****
C      DISTANCE OF CENTROID TO TOP IN AB
C*****
  gamas=0.875*vkd
C*****
C      DISTANCE OF CENTROID TO TOP IN BC
C*****
  gama2=0.259255*vkd
C*****
C      COEFFICIENTS FOR FAILURE FUNCTION
C*****
  al=(ccab*(dd-gamas)+ccbc*(dd-gama2))/fc
  a2=-1.
C*****
C      COSINE DIRECTORS
C*****
  tetas=al*sigmas*fc
  tetas2=a2*sigma2*vm
C*****
C      INDEPENDENT TERM
C*****
  fps=0.004*es*(vkd-co)/vkd
  bi=aste*fps*(dd-co)
C*****
C      RELIABILITY INDEX
C*****
  beta(kel)=(al*fc+a2*vm+bi)/sqrt(tetas*tetas+tetas2*tetas2)
C*****
C      ULTIMATE MOMENT AND ROTATION
C*****
  vmu(kel)=al*fc+bi
  phiu=0.004/vkd
  if((4.*phiy).lt.phi) then
    vmu(kel)=(vmu(kel)-vmy)/(phi-phiy)*3.*phiy+vmy

```

```

phiu=4.*phiy
endif
return
end

```

```

=====
=====

```

```

      subroutine assemb (e,igh,n,ntot,x,cl,cos1,cos2,
*                lm,glk)
      implicit double precision (a-h,o-z)
      dimension x(ntot),cl(n),cos1(n),cos2(n)
      dimension lm(6,n),glk(igh,igh)
      common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
      do 100 k = 1,igh
      do 150 j = 1,igh
150         glk (j,k) = 0.0
100     continue
c*****
c                global stiffness evaluation
c*****
      do 700 j = 1,n
        call glosti (e,j,n,ntot,cl,x,cos1,cos2)
        do 300 l = 1,6
          k = lm (l,j)
          if (k.eq.0) go to 300
          do 200 ll = 1,6
            m = lm (ll,j) - k + 1
            if (m.le.0) go to 200
            glk(k,m)=ck(l,ll)+glk(k,m)
200             continue
300         continue
700     continue
      return
      end

```

```

=====
=====

```

```

      subroutine constr (iqg,d)
      implicit double precision (a-h,o-z)
      dimension d(iqg)
c*****
c                displacement constraints
c*****
      do 201 k = 1,iqg
        read ( 9,* ) d(k)
201     continue

```

```

201  continue
      return
      end

```

```

=====
=====

```

```

      subroutine datini (clah,clag,ch,cg)
      implicit double precision ( a-h,o-z )
      dimension clah(igh), clag(igq), ch(igh), cg(igq)
      common /parr/ decfc,fcinc,cv,alpl,ec,rp,fc,es,ecm,
*               relind
      common /pari/ iter,numcy,niter,ga,igh,igq,n,ntot,igqn
C*****
C      lag. mult. of equality c.
C*****
      do 100 k = 1,igh
100    clah(k) = 0.0
C*****
C      lag. mult. of inequality c.
C*****
      do 200 k = 1,igqn
        clag(k) = 0.0
200    continue
C*****
C      scaling factors of equ. c.
C*****
      do 300 k = 1,igh
300    ch(k) = 1.0
C*****
C      scaling factors of ineq. c.
C*****
      do 400 k = 1,igqn
400    cg(k) = 1.0
C*****
C      scaling factor of objective fun.
C*****
      cv = 1.0
      return
      end

```

```

=====
=====

```

```

      subroutine glosti (e, j, n, ntot, cl, x, cos1, cos2)
      implicit double precision ( a-h,o-z )
      dimension cl(n), x(ntot), cos1(n), cos2(n)

```

```

common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
c1 = cos1(j)
c2 = cos2(j)
c12 = c1 * c1
c22 = c2 * c2
area = x(j*3-1)*x(3*j-2)
tinert = area*x(3*j-1)*x(3*j-1)/12.
c11 = c1(j)
c12 = c11*c11
c13 = c12*c11
a = e*tinert/c13
b = area*c12/tinert
g1 = a*(b*c12+12.*c22)
g2 = a*c1*c2*(b-12.)
g3 = a*(b*c22+12.*c12)
g4 = -a*6.*c11*c2
g5 = a*6.*c11*c1
g7 = a*2.*c12
g6 = g7 + g7
ck(1,1) = g1
ck(2,1) = g2
ck(3,1) = g4
ck(4,1) = - g1
ck(5,1) = - g2
ck(6,1) = g4
ck(1,2) = g2
ck(2,2) = g3
ck(3,2) = g5
ck(4,2) = - g2
ck(5,2) = - g3
ck(6,2) = g5
ck(1,3) = g4
ck(2,3) = g5
ck(3,3) = g6
ck(4,3) = - g4
ck(5,3) = - g5
ck(6,3) = g7
ck(1,4) = - g1
ck(2,4) = - g2
ck(3,4) = - g4
ck(4,4) = g1
ck(5,4) = g2
ck(6,4) = - g4
ck(1,5) = - g2
ck(2,5) = - g3
ck(3,5) = - g5
ck(4,5) = g2
ck(5,5) = g3
ck(6,5) = - g5
ck(1,6) = g4
ck(2,6) = g5
ck(3,6) = g7
ck(4,6) = - g4
ck(5,6) = - g5
ck(6,6) = g6
return

```


end

```
=====
=====
```

```

      subroutine inecon (iqg,n,ntot,vag,x,d,relind,beta)
      implicit double precision (a-h,o-z)
      dimension vag(iqg+n),x(ntot),d(iqg),beta(n)
      nel3=3*n
c*****
c      displacements
c*****
      do 100 k = 1,iqg
          j = nel3 + k
          vag(k) = abs(x(j)) / d(k) - 1.
100    continue
c*****
c      reliability
c*****
      iqgpl=iqg+1
      iqgn=iqg+n
      do 200 k=iqgpl,iqgn
          vag(k) = relind/beta(k-iqg)-1.
200    continue
      return
      end
=====
=====
```

```

      subroutine inputd (cl,cosl,cos2,iqh,jdir,jm,lm,n,nj,
*      nol,no2,r,xc,yc)
      implicit double precision (a-h,o-z)
      dimension nol(n),no2(n),jdir(3),xc(nj),yc(nj),
*      jm(6,nj),lm(6,n),cl(n),cosl(n),cos2(n),r(iqh)
c*****
c      filling lm matrix
c*****
      do 600 i = 1,n
          j = nol(i)
          k = no2(i)
          do 600 l = 1,3
              lm(l,i) = jm(l,j)
              lm(l+3,i) = jm(l,k)
600    continue
c*****
c      geometric characteristics
c*****
      do 800 ii = 1,n
=====
=====
```

```

      j = nol(ii)
      k = no2(ii)
      ell = xc(k) - xc(j)
      el2 = yc(k) - yc(j)
      cl(ii) = sqrt(ell*ell+el2*el2)
      cos1(ii) = ell/cl(ii)
      cos2(ii) = el2/cl(ii)
800  continue
c*****
c      initialization of global forces
c*****
      do 850 k=1,igh
          r(k) = 0.0
850  continue
c*****
c      global forces
c*****
900  read(9,*)jnum,jdire,force
      if(jnum.ne.0)then
          k=jm(jdire,jnum)
          r(k)=force
          go to 900
      endif
      return
      end

```

```

=====
=====

```

```

subroutine lim(x,n,ntot,xl1,xl2,k,romin,romax)
implicit double precision (a-h,o-z)
dimension x(ntot)
n3=3*n
do 10 i=1,n3,3
  l1=i
  l2=i+1
  l3=i+2
  astma=romax*x(l1)*x(l2)
  astm=romin*x(l1)*x(l2)
  if(k.eq.l1)then
    if(x(k).lt.xl1)x(k)=xl1
    basemin=x(l3)/(romax*x(l2))
    if(x(k).lt.basemin)x(k)=basemin
    basemax=x(l3)/(romin*x(l2))
    if(x(k).gt.basemax)x(k)=basemax
  i=n3
go to 10
endif
if(k.eq.l2)then
  if(x(k).lt.xl2) x(k)=xl2
  heightmin=x(l3)/(romax*x(l1))
  if(x(k).lt.heightmin)x(k)=heightmin

```

```

heightmax=x(13)/(romin*x(11))
if(x(k).gt.heightmax)x(k)=heightmax
i=n3
go to 10
endif
if(k.eq.13)then
if(x(k).gt.astma) x(k)=astma
if(x(k).lt.astm) x(k)=astm
i=n3
endif
10 continue
return
end

```

```

=====
=====

      subroutine parame (toll,x,n21,glk,mband,cl,
*               cos1,cos2,lm,r,delta,alpha,numec,rph)
      implicit double precision (a-h,o-z)
      dimension x(ntot),glk(iqh,iqh),cl(n),cos1(n),
*               cos2(n), lm(6,n), r(iqh)
      common /parr/ decfc,fcinc,cv,alpl,ec,rp,fc,es,ecm,
*               relind
      common /pari/ iter,numcy,niter,ga,iqh,iqg,n,ntot,iqgn
      common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
c*****
c               penalty factor
c*****
      read (9,*) rp,rph,alpl
c*****
c               gamma,# of iterations, # of cycles
c*****
      read (9,*) ga,niter,numcy
c*****
c               decrease and increase factors
c*****
      read (9,*) decfc,fcinc
c*****
c               control tollerance
c*****
      read ( 9,* ) toll
c*****
c               initial guesses of dimensions
c*****
      n3=n+n+n
      n21=n3+1
      read (9,*) (x(i),x(i+1),i=1,n3,3)
c*****
c               increment for slope evaluation
c*****
      read(9,*)delta,alpha
c*****

```

```

c               element reliability
c*****
      read(9,*)relind
c*****
c               number of elementary mechanisms
c*****
      read(9,*)numec
c*****
c               generation of global stiffness
c*****
      call assemb(ec,iqh,n,ntot,x,cl,cos1,cos2,lm,glk)
c*****
c               initial displacements
c*****
      call sysmol(glk,r,x(n21),iqh,mband)
      return
      end

```

```

=====
=====

```

```

      subroutine valobf (n,ntot,vof,x,cl)
      implicit double precision (a-h,o-z)
      dimension x(ntot), cl(n)
      vof = 0.0
      do 100 k = 1,n
         base = x(3*k-2)
         height = x(3*k-1)
         steel = x(3*k)
         area = base * height
         vof=vof+(area+steel*10)*cl(k)
100    continue
      return
      end

```

```

=====
=====

```

```

      subroutine sysrel(nel,numec,ndof,theta,r,vmu,cvmu,p,
*               cvload,jflag)
      implicit double precision (a-h,o-z)
      dimension theta(200,100),p(100),rb(200,100),
*      become(500),lc(300),thesum(100),temp(100,100),
*      locmec(100),thesul(100),dispsu(100),ni(100),
*      nucomb(100),lct(300),elerel(100),cvmu(100),
*      r(100),vmu(100),cvload(100)
c*****
c               form fundamental mechanisms
c

```

```

C*****
      ndof=iqh
      do 20 j=1,ndof
        p(j)=r(j)
20      continue
C*****
C      ordering theta and r matrices
C*****
      do 710 k=1,numec-1
        jflag=0
        do 720 i=1,ndof
          if(abs(rb(i,k)).gt.0.0)jflag=i
720        continue
          if (jflag.eq.0)then
            do 730 l=k+1,numec
              do 740 li=1,ndof
                if(abs(rb(li,l)).gt.0.0)then
                  do 750 lj=1,ndof
                    temp(lj,l)=rb(lj,l)
                    rb(lj,l)=rb(lj,jflag)
                    rb(lj,jflag)=temp(lj,l)
750                  continue
                    do 760 lj=1,2*nel
                      temp(lj,l)=theta(lj,l)
                      theta(lj,l)=theta(lj,jflag)
                      theta(lj,jflag)=temp(lj,l)
760                    continue
                    go to 733
                  endif
                continue
              continue
            continue
          endif
710        continue
C*****
C      normalizing theta and r vectors
C*****
      do 810 i=1,numec
        do 820 j=1,2*nel
          if(abs(theta(j,i)).ne.1.0.and.
            *      theta(j,i).ne.0.0)then
            fact=abs(1./theta(j,i))
            do 830 jj=1,2*nel
              theta(jj,i)=theta(jj,i)*fact
830            continue
            do 840 jj=1,ndof
              rb(jj,i)=rb(jj,i)*fact
840            continue
            go to 734
          endif
        continue
      continue
820      continue
830      continue
810      continue
C*****
C      transpose theta and r matrices
C*****

```

```

do 25 j=1,numec
do 91 i=nel
temp(i,j)=theta(i,j)
91 continue
25 continue
do 56 i=1,numec
do 55 j=1,2*nel
theta(i,j)=temp(j,i)
55 continue
56 continue
do 28 j=1,numec
do 27 i=1,ndof
temp(i,j)=rb(i,j)
27 continue
28 continue
do 66 i=1,numec
do 65 j=1,ndof
rb(i,j)=temp(j,i)
65 continue
66 continue
C*****
C      reliability of fundamental mechanisms
C*****
do 102 i=1,numec
vmeanr=0.0
stdevr=0.0
do 202 k=1,nel
j=2*k-1
theji=abs(theta(i,j))
thejil=abs(theta(i,j+1))
if(theji.lt.0.0001.and.thejil.lt.
*      0.0001)goto 202
term2=(theji+thejil)*cvmu(k)*vmu(k)
stdevr=stdevr+term2*term2
vmeanr=term2/cvmu(k)+vmeanr
202 continue
vmeanl=0.0
stdevl=0.0
do 302 k=1,ndof
if(abs(p(k)).lt.0.001)go to 302
term=p(k)*rb(i,k)
vmeanl=vmeanl+term
stdevl=stdevl+term*term*(cvload(k)
*      *vload(k))
302 continue
vmean=vmeanr-vmeanl
stdev=stdevr+stdevl
become(i)=vmean/sqrt(stdev)
lnumbe=lnumbe+1
102 continue
C*****
C      reliability of combined mechanisms
C      (internal work)
C*****
do 50 l=1,numec
lc(l)=1

```

```

      lct(1)=1
50    continue
      lp=numec
      ltemp=numec
      ni(1)=1
      numax=6
      nucome=1
      nucomb(1)=numec
      lpt=numec
      lpti=numec+1
111   continue
C*****
C      loop over mechanisms in location vector
C*****
      do 699 kmo=1,nel
          thesul(kmo)=0.0
          thesum(kmo)=0.0
699   continue
          do 698 klp=1,ndof
              dispsu(klp)=0.0
698   continue
          kmu=1
          nic=ni(nucome)
          nif=nic+nucomb(nucome)*nucome-1
          nucomb(nucome+1)=0
C*****
C      define acceptable interval
C*****
          if(nucome.gt.1)then
              nifbet=lpti-1
              nicbet=lpti-nucomb(nucome)
              do 5544 ia=nicbet+1,nifbet
                  betaal=become(nicbet)+epsilo
                  if(become(ia).gt.betaal)then
                      do 5545 ib=ia,nifbet
                          become(ib)=1000.
5545                  continue
                      go to 5541
              endif
5544          continue
5541          continue
          niccon=nicbet
      endif
C*****
C      reliability of combined mechanisms
C      (internal work)
C*****
      do 200 j=nic,nif,nucome
          icontr=0
          if(nucome.gt.1)then
              if(become(niccon).gt.100.)icontr=1
              niccon=niccon+1
              if (icontr.eq.1)go to 5564
          endif
          do 300 l=1,nucome
              lj=lc(j+l-1)

```

```

        if(lj.lt.-0.1)then
            kmu=-1
            lj=abs(lj)
        endif
        do 400 kk=1,nel
            kkj=2*kk-1
            thesum(kk)=thesum(kk)+theta(lj, kkj)*kmu
            thesul(kk)=thesul(kk)+theta(lj, kkj+1)*kmu
400            continue
            kmu=1
300        continue
c*****
c            reliability of combined mechanisms
c            (external work)
c*****
        do 372 l=1,nucome
            lcl=lc(j+1-1)
            if(lcl.lt.-0.1)then
                kmu=-1
                lcl=abs(lcl)
            endif
            do 472 kk=1,ndof
                if(abs(p(kk)).lt.0.001)go to 472
                dispsu(kk)=dispsu(kk)+rb(lcl, kk)*kmu
472            continue
            kmu=1
372        continue
5564    continue
c*****
c            combination with fundamental mechanisms
c            (internal work)
c*****
        do 100 k=1,numec
            vmeanr=0.0
            vmeanl=0.0
            stdevr=0.0
            stdevl=0.0
            vmanrm=0.0
            vmanlm=0.0
            stdvrm=0.0
            stdvlm=0.0
            do 499 lll=j,j+nucome-1
                if(abs(lc(lll)).ge.k)go to 100
499            continue
            if(nucome.gt.1)then
                if (icontr.eq.1)then
                    becomi=400.
                    becopl=500.
                    go to 5574
                endif
            endif
            thesu=0.0
            thesu2=0.0
            thesui=0.0
            theslm=0.0
            do 600 kk=1,nel

```



```

      kkj=2*kk-1
      thesu=thesum(kk)+theta(k,kkj)
      thesu2=thesu1(kk)+theta(k,kkj+1)
      term=(abs(thesu)+abs(thesu2))*cvmu(kk)
*
      *vmu(kk)
      vmeanr=term/cvmu(kk)+vmeanr
      stdevr=stdevr+term*term
      thesui=thesum(kk)-theta(k,kkj)
      theslm=thesu1(kk)-theta(k,kkj+1)
      termm=(abs(thesui)+abs(theslm))*cvmu(kk)
*
      *vmu(kk)
      vmanrm=termm/cvmu(kk)+vmanrm
      stdvrm=stdvrm+termm*termm
600      continue
C*****
C      combination with fundamental mechanisms
C      (external work)
C*****
      do 672 kk=1,ndof
        if(abs(p(kk)).lt.0.001)go to 672
        dispkk=(dispsu(kk)+rb(k,kk))*p(kk)
        vmeanl=vmeanl+dispkk
        stdevl=stdevl+dispkk*cvload(kk)*dispkk
*
        *cvload(kk)
        dispkm=(dispsu(kk)-rb(k,kk))*p(kk)
        vmanlm=vmanlm+dispkm
        stdvlm=stdvlm+dispkm*cvload(kk)*dispkm
*
        *cvload(kk)
672      continue
      becopl=(vmeanr-vmeanl)/sqrt(stdevr+stdevl)
      becomi=(vmanrm-vmanlm)/sqrt(stdvrm+stdvlm)
5574      continue
      if(becomi.lt.becopl) then
        do 138 lk=1,nucome
          ltemp=ltemp+1
          lc(ltemp)=lc(j+lk-1)
138      continue
          ltemp=ltemp+1
          lc(ltemp)=-k
          lctlp=-k
          becote=becomi
        else
          do 139 lk=1,nucome
            ltemp=ltemp+1
            lc(ltemp)=lc(j+lk-1)
139      continue
            lctlp=k
            ltemp=ltemp+1
            lc(ltemp)=k
            becote=becopl
          endif
          lpt=lpt+1
          iflag=0
C*****
C      ordering the combined beta values in the same row
C*****

```

```

      if (lpti.gt.lpt-1)then
        do 524 lk=1,nucome
          lp=lp+1
          lct(lp)=lc(j+lk-1)
524      continue
          lp=lp+1
          lct(lp)=lctlp
          become(lpt)=becote
      else
        do 510 jkj=lpti,lpt-1
          if(become(jkj).gt.becote)then
            iflag=-1
            do 511 kjk=jkj,lpt-1
              itemp=lpt-1+jkj-kjk
              become(itemp+1)=become(itemp)
511      continue
              become(jkj)=becote
              becote=become(lpt)
C*****
C      moving lc array
C*****
              movini=nic+nucomb(nucome)*nucome+
              *      (jkj-lpti)*(nucome+1)
              movfin=(lpt-lpti)*(nucome+1)+nic+
              *      nucomb(nucome)*nucome-1
              do 512 lmn=movini,movfin
                lcou=movini+movfin-lmn
                nucl=nucome+1
                lct(lcou+nucl)=lct(lcou)
512      continue
                do 513 n=movini,movini+nucome-1
                  lptaa=n-movini+1
                  lct(n)=lc(j+lptaa-1)
513      continue
                  lct(movini+nucome)=lctlp
              endif
510      continue
          if(iflag.eq.0)then
            do 124 lk=1,nucome
              lp=lp+1
              lct(lp)=lc(j+lk-1)
124      continue
              lp=lp+1
              lct(lp)=lctlp
              become(lpt)=becote
          else
            lp=lp+nucome+1
          endif
        endif
        nucomb(nucome+1)=nucomb(nucome+1)+1
        lnumbe=lnumbe+1
100      continue
        do 6991 kmo=1,nel
          thesul(kmo)=0.0
          thesum(kmo)=0.0
6991      continue

```

```

        do 6981 klp=1,ndof
            dispsu(klp)=0.0
6981      continue
200      continue
c*****
c          control of maximum number of tree rows
c*****
        lpti=lpti+nucomb(nucome+1)
        nucome=nucome+1
        ni(nucome)=ni(nucome-1)+(nucome-1)*nucomb(nucome-1)
        if(nucome.lt.(numax))go to 111
c*****
c          find minimum beta value
c*****
        betmin=100
        do 7890 i=1,lnumbe
            if(become(i).lt.betmin)then
                betmin=become(i)
                itab=i
            endif
7890      continue
c*****
c          find mechanisms involved
c*****
        mecoun=0
        mcomb=1
        do 7891 j=1,nucome-1
            do 7895 jk=1,nucomb(j)
                mecoun=1+mecoun
                if(itab.eq.mecoun)then
                    nistar=mcomb
                    do 7893 l=1,j
                        locmec(l)=lct(nistar+l-1)
7893          continue
                    nummec=j
                    go to 7894
                endif
            mcomb=j+mcomb
7895      continue
7891      continue
7894      continue
c*****
c          find elements involved
c*****
        nk=1
        numele=0
        do 8000 m=1,nel
            do 8001 k=1,numec
                do 8002 l=1,nummec
                    if(abs(locmec(l)).eq.k)then
                        m1=2*m-1
                        m2=2*m
                        if(abs(theta(m1,k)).gt.0.0.
*                        or.abs(theta(m2,k)).gt.0.0)then
                            if(nk.gt.1)then
                                do 8003 lmi=1,nk

```

```

                                if(invmeclmi).eq.m)
                                go to 8004
*
8003                                continue
                                endif
                                nk=nk+1
                                numele=numele+1
                                invmec(n)=m
                                go to 8000
8004                                continue
                                endif
                                endif

8002                                continue
8001                                continue
8000                                continue
C*****
C                                control of system reliability
C*****
    write(8,*)
    write(8,*) 'BETA MINIMAL FOR THE SYSTEM = ',betmin
    write(8,*)
    jflag=0
    if(betmin.lt.relind)then
    delta=(relind-betmin)/relind
        do 3891 i=1,nel
            do 3892 j=1,numele
                if(invmeclj).eq.i)then
                    elerel(i)=(1.+delta)*elerel(i)
                endif
3892                                continue
3891                                continue
                                jflag=1
    endif
    return
end

```

```

=====
=====

```

```

subroutine mecsys(n,iqh,cl,cost,sint,lm,numec,r,theta)
implicit double precision (a-h,o-z)
dimension a(100,100),b(100,100),c(100),cm(100,100),
*      cost(n),qa(100,100),sint(n),cl(n),q(100,100),
*      lm(6,n),am(100,100),bl(100,100),theta(200,100),
*      r(iqh,100)
C*****
C                                Constraint matrix for the structure
C*****
    do 300 i=1,3*n
        do 400 j=1,6*n
            cm(i,j)=0.0
            q(i,j)=0.0
400                                continue

```

```

300  continue
      do 60 k=1,n
          i=3*k
          j=6*k
          at=1.0/cl(k)
          im2=i-2
          im1=i-1
          jml=j-1
          jm2=j-2
          jm3=j-3
          jm4=j-4
          jm5=j-5
          cm(im2,jm5)=-1.0
          cm(im2,jm2)=1.0
          cm(im1,jm3)=1.0
          cm(i,j)=1.0
          cm(im1,jm4)=-at
          cm(im1,jml)=at
          cm(i,jm4)=-at
          cm(i,jml)=at
60    continue
C*****
C      Coordinate trnsformation matrix
C*****
      do 70 k=1,n
          co=cost(k)
          si=sint(k)
          j=6*k
          do 80 i=1,2
              jum=j-3*i+1
              jdois=j-3*i+2
              jtres=j-3*i+3
              q(jum,jum)=co
              q(jum,jdois)=si
              q(jdois,jum)=-si
              q(jdois,jdois)=co
              q(jtres,jtres)=1.0
80    continue
70    continue
C*****
C      Compatibility matrix from LM matrix
C*****
      n6=6*n
      do 500 i=1,n6
          do 510 k=1,igh
              am(i,k)=0.0
510    continue
500    continue
      do 520 i=1,6
          do 530 k=1,n
              iflag=lm(i,k)
              if (iflag.gt.0) am(6*(k-1)+i,iflag)=1.0
530    continue
520    continue
C*****
C      Rotation of basic compatibility matrix

```

```

C*****
      mm=80
      call multi(q,am,qa,n6,igh,n6,mm)
C*****
C      Expansion of QA matrix
C*****
      do 585 i=1,n
          i6=6*i
          i3=i6-3
          do 595 j=1,igh
              qa(i3,j)=0.0
              qa(i6,j)=0.0
595          continue
              i2=2*i
              qa(i3,igh+i2-1)=1.0
              qa(i6,igh+i2)=1.0
855      continue
C*****
C      Matrix A = C * QA (transformed)
C*****
      m=3*n
      nt=igh+2*n
      call multi(cm,qa,a,m,nt,n6,mm)
C*****
C      Solution for virtual displacements
C*****
      do 150 k=1,nt
          do 175 l=1,nt
              b(k,l)=0.0
175      continue
150      continue
          do 160 k=1,nt
              b(k,k)=1.0
160      continue
          do 200 i=1,m
              amax=0.0
              iflag=0
              do 250 j=i,nt
                  if (abs(a(i,j)).gt.amax) then
                      iflag=j
                      amax=abs(a(i,j))
                  endif
250      continue
              do 305 k=1,m
                  c(k)=a(k,iflag)
                  a(k,iflag)=a(k,i)
                  a(k,i)=c(k)
305      continue
              do 310 k=1,nt
                  c(k)=b(k,iflag)
                  b(k,iflag)=b(k,i)
                  b(k,i)=c(k)
310      continue
              do 280 j=i+1,nt
                  if (abs(a(i,j)).gt.0.00001) then
                      fact=-a(i,j)/a(i,i)

```

```

do 290 kk=1,m
  a(kk,j)=a(kk,j)+a(kk,i)*fact
290 continue
do 291 kk=1,nt
  b(kk,j)=b(kk,j)+b(kk,i)*fact
291 continue
endif
280 continue
200 continue
numec=nt-3*n
C*****
C Forming b1
C*****
lcount=nt-numec+1
ki=1
do 800 i=lcount,nt
  do 810 j=1,nt
    b1(j,ki)=b(j,i)
810 continue
    ki=ki+1
800 continue
C*****
C Creating Theta matrix
C*****
do 156 j=1,numec
  do 157 i=1,2*n
    k=igh+i
    theta(i,j)=b1(k,j)
157 continue
156 continue
C*****
C Creating virtual displacements
C*****
do 169 j=1,numec
  do 158 i=1,igh
    r(i,j)=b1(i,j)
158 continue
C*****
C Adding joint mechanisms
C*****
do 161 i=3,igh,3
  if (abs(r(i,j)).gt.0.000001)then
    r(i,j)=0.0
    do 162 k=1,n
      if(lm(3,k).eq.i)then
        lpo=2*k-1
        theta(lpo,j)=1
      endif
      if(lm(6,k).eq.i)then
        lpo=2*k
        theta(lpo,j)=1
      endif
    endif
162 continue
  endif
161 continue
169 continue

```

```

return
end

```

```

=====
=====

subroutine multi(aa,bb,cc,l,m,n,k)
implicit double precision (a-h,o-z)
dimension aa(l,n),bb(n,m),cc(l,m)
do 10 i=1,l
do 20 j=1,m
d=0.0
do 30 kk=1,n
d=d+aa(i,kk)*bb(kk,j)
30      continue
cc(i,j)=d
20      continue
10      continue
return
end

```

```

=====
=====

subroutine jacequ (x,n,cl,lm,cosl,cos2,fc,ec,vn,co,epsy,
*      fy,ntot,iqh,vah,r,vahk,es,ecm,beta,cvmu,cvload,kl,
*      vmu,vjac)
implicit double precision (a-h,o-z)
dimension lm(6,n),cosl(n),cos2(n),vmu(n),vjac(iqh,iqh)
dimension cl(n),x(ntot),vah(iqh),r(iqh)
dimension vahk(iqh),beta(n),cvmu(n),cvload(iqh)
common /esq/ u(6),ck(6,6),vksi(100),vksj(100)
do 150 kme = 1,iqh
do 160 kmo=1,iqh
vjac(kmo,kme)=0.0
160      continue
150      continue
do 100 k=1,n
cl=cosl(k)
c2=cos2(k)
c = cl(k)
base = x(3*k-2)
height = x(3*k-1)
aste = x(3*k)
area = base * height
tinert = area*height*height/12.0
C*****
C      GLOBAL MODIFIED STIFFNESS
C*****
call equcon (x,n,cl,lm,cosl,cos2,fc,ec,vn,co,epsy,fy,

```



```

*      ntot, iqh, vah, r, vahk, es, ecm, beta, cvmu, cvload, kl, vmu)
      call modsti(area, ec, vksi(k), vksj(k), c, tinert, c1, c2)
      do 300 l=1,6
        j=lm(l,k)
        if (j.eq.0) go to 300
        do 400 ll = 1,6
          m=lm(ll,k)
          if (m.eq.0) go to 400
          vjac(j,m)=vjac(j,m)+ck(l,ll)
400      continue
300      continue
100      continue
      return
      end

```

0.15
0.15
0.15
0.15
0.15
2,6
1.5

User's Manual
Augmented Lagrangian Formulation

Example: Debug Frame

Input File: DATA

- Line 1
 Problem title.
- Line 2
 Number of elements, number of nodes.
- Line 3 to line 6
 Node i, node j of element 1 through 4.
- Line 7 to line 11
 Boundary conditions of displacement in the horizontal direction, vertical direction, in-plane rotation, horizontal coordinate, vertical coordinate.
- Line 12 and line 13
 Node where force is applied, direction of load and magnitude of load.
- Line 14
 Termination of force information.
- Line 15
 Flexural strength of concrete, yielding stress of steel and reinforcement cover.
- Line 16
 Steel modulus of elasticity and concrete ultimate strain.
- Line 17 to line 20
 Initial steel reinforcement areas.

- Line 21 to 29

Displacement limits.

- Line 30

Penalty factor, equality penalty factor and stepsize.

- Line 31

Factor of penalty increase, maximum number of iterations and maximum number of cycles.

- Line 32

Decrease factor and increase factor.

- Line 33

Convergence tolerance.

- Line 34

Element and system reliability index.

- Line 35

Number of elementary mechanisms.

- Line 36 to line 39

Coefficient of variation of concrete strength.

- Line 40 to line 48

Coefficient of variation of external loads.

- Line 49

Lower bounds of cross section dimensions.

- Line 50

Value of interval gap in the Beta unzipping method.

APPENDIX B

GENERALIZED REDUCED GRADIENT EXAMPLE

The example and correspondent optimization conditions chosen to illustrate the performance of the Generalized Reduced Gradient method using the integrated formulation are presented in Figure B.1. The maximum flexural stress, compression or tension, is 1,000 psi. The problem is solved in separate steps presented below.

Step A - Problem Formulation

Objective Function

$$\text{Minimize } f(\mathbf{x}) = 10x_1x_2$$

Equality Constraints

$$h_1(\mathbf{x}) = 0.03x_1x_2^3x_3 - 0.015x_1x_2^3x_4 + 1 = 0$$

$$h_2(\mathbf{x}) = -0.15x_1x_2^3x_3 + x_1x_2^3 = 0$$

Inequality Constraints

$$h_3(\mathbf{x}) = 60/(x_1x_2^2) - 1 + x_5 = 0$$

where x_5 - slack variable;

Variable bounds

$$x_1 \geq 0.5 \text{ in}$$

$$x_2 \geq 0.5 \text{ in}$$

$$|x_3| \leq 0.5 \text{ in}$$

$$|x_4| \leq 0.5 \text{ rad}$$

Step B - Explicit Derivatives

$$df/dx_1 = 10x_2 \quad df/dx_2 = 10x_1 \quad df/dx_3 = \dots = 0;$$

$$dh_1/dx_1 = x_2^3(0.03x_3 - 0.15x_4);$$

$$dh_1/dx_2 = x_1x_2^2(0.09x_3 - 0.45x_4);$$

$$dh_1/dx_3 = 0.03x_1x_2^3;$$

$$dh_1/dx_4 = -0.15x_1x_2^3;$$

$$dh_1/dx_5 = 0;$$

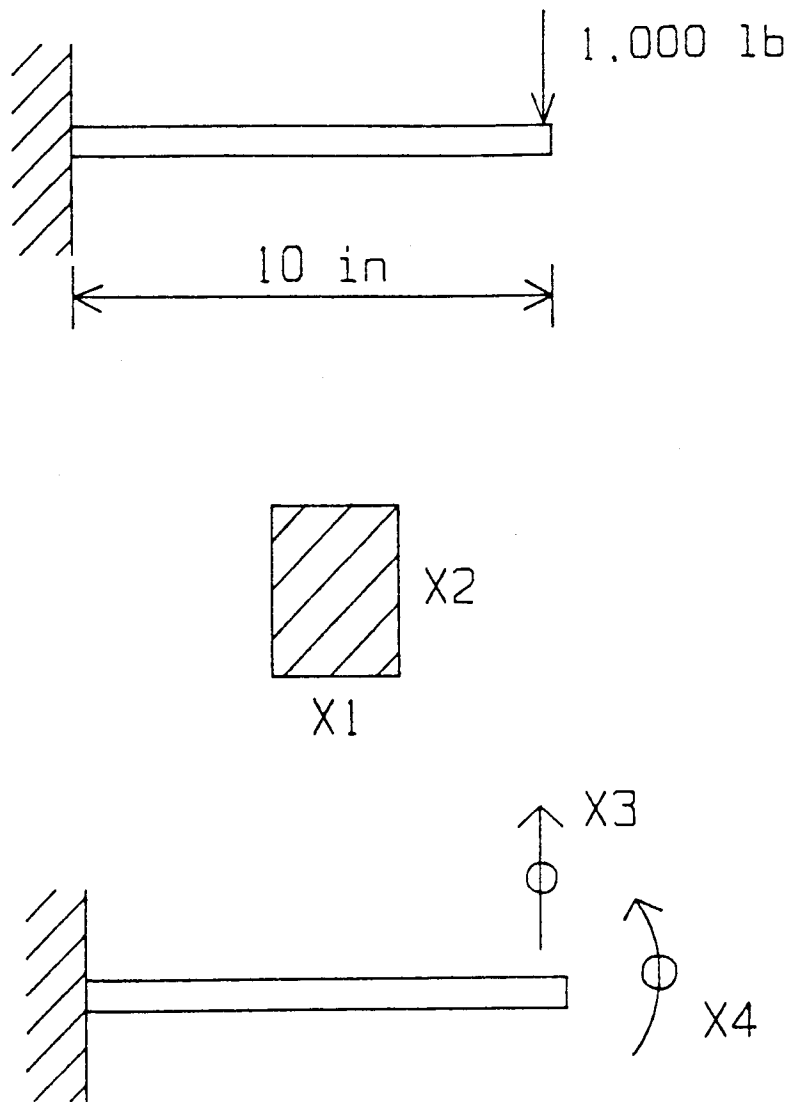


Figure B.1. Integrated optimization example.

$$dh_2/dx_1 = x_2^3(-0.15x_3 + x_4);$$

$$dh_2/dx_2 = x_1x_2^2(-0.45x_3 + 3x_4);$$

$$dh_2/dx_3 = -0.15x_1x_2^3;$$

$$dh_2/dx_4 = x_1x_2^3;$$

$$dh_2/dx_5 = 0;$$

$$dh_3/dx_1 = dh_3/dx_2 = dh_3/dx_4 = 0;$$

$$dh_3/dx_3 = -0.45x_3;$$

$$dh_3/dx_5 = 1;$$

Step C - Initial Design Point and Initial Values

Dependent variables - $d_d^t = (x_1, x_2, x_3);$

Independent variables - $d_i^t = (x_4, x_5);$

$$d_d^t = (1, 10, -0.1333) \quad d_i^t = (-0.02, 0.4)$$

$$\text{grad } f_d^t = (100, 10, 0) \quad \text{grad } f_i^t = (0, 0)$$

where grad f is gradient of f;

$$H = [J \mid C]$$

where H is Hessian matrix of the equalities;

$$J = \begin{bmatrix} -1 & -0.3 & 30 \\ 0 & 0 & -150 \\ -0.6 & -0.12 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} -150 & 0 \\ 1000 & 0 \\ 0 & 1 \end{bmatrix}$$

Step D - Recurrence Formulas

$$d_{kt} = (d_d^k \mid d_i^k);$$

$$d_i^{k+1} = - \text{grad } f_i^k + (J^{-1}C) \text{ grad } f_d^k;$$

$$d_d^{k+1} = - J^{-1}C d_i^{k+1};$$

Step E - Iterations

First iteration:

$$\underline{x}^{0t} = \{1 \ 10 \ -0.133 \ -0.02 \ 0.4\};$$

$$\underline{d}^{1t} = \{-501667 \ 2505556 \ 33333 \ 5000 \ -333\};$$

$$\alpha^1 = 10^{-6}, \text{ because } x_1 \geq 0.5;$$

$$\underline{x}^{1t} = \{0.5 \ 12.506 \ -0.1 \ -0.015 \ 0.3997\};$$

Second iteration:

Change of variables - x_4 replaces x_1 that is at a lower bound;

$$\underline{d}^{2t} = \{0 \ -332.09 \ -7.9664 \ -1.195 \ 40.75\};$$

$$\alpha^2 = 0.3997/40.75, \text{ because } \sigma \leq 1000;$$

$$\underline{x}^{2t} = \{0.5 \ 9.249 \ -0.178 \ -0.0267 \ 0\};$$

Independent variables, $\{x_1, x_5\}$ are at their lower bounds;

Iteration is performed on the set of dependent variables $\{x_2, x_3, x_4\}$;

Third iteration:

$$-J^{-1} h(\underline{x}^2)^t = \{1.3228 \ -0.082 \ -0.0124\};$$

$$\underline{x}_d^{3t} = \{10.58 \ -0.26 \ -0.0391\};$$

Fourth iteration:

$$-J^{-1} h(\underline{x}^3)^t = \{0.358 \ 0.061 \ 0.009\};$$

$$\underline{x}_d^{3t} = \{10.935 \ -0.199 \ -0.0301\};$$

Fifth iteration:

$$-J^{-1} h(\underline{x}^4)^t = \{0.019 \ -0.004 \ -0.003\};$$

$$\underline{x}_d^{3t} = \{10.954 \ -0.203 \ -0.0304\};$$

Stop.

Optimum design

$$\underline{x}^{*t} = \{0.5 \ 10.954 \ -0.203 \ -0.304 \ 0\}.$$

APPENDIX C

GENERALIZED REDUCED GRADIENT SUBROUTINES

APPENDIX C

GENERALIZED REDUCED GRADIENT SUBROUTINES

```

program optim
call datain
call grg
call outres
stop
end

```

```

=====
=====

```

```

subroutine gcomp(g,x)
implicit real*8 (a-h,o-z)
dimension g(1),x(1)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*             cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*             no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/ iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),ck(6,6)
common /equal/ vah(100),vmu(100)
call equcon(x)
call inecon(x)
call valobf(x,vof)
do 100 i=1,iqh
g(i)=vah(i)
100 continue
C*****
C             ELEMENT RELIABILITY LIMITS
C*****
do 200 i=iqh+1,iqh+nel
g(i)=-vag(i-iqh)
200 continue
C*****
C             REINFORCEMENT LIMITS
C*****
do 300 i=iqh+nel+1,iqh+2*nel
kj=i-iqh-nel
g(i)=1000.*x(3*kj)/(x(3*kj-2)*x(3*kj-1))
300 continue
g(iqh+2*nel+1)=vof
return
end

```

```

=====
=====

```

```

subroutine princi
implicit double precision (a-h,o-z)

```

```

      common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*          cvmu(100)
      common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
      common /xcord/ xc(100),yc(100),d(100),jdir(3)
      common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
      common /pari/ iqh,iqg,nel,ntot,iqgn
      open ( 9,file='data',form='formatted' )
      rewind 9
C*****
C          # ELEMENTS AND # JOINTS
C*****
      read (9,*)nel,nj
C*****
C          NODES PER ELEMENT
C*****
      do 100 i=1,nel
      read(9,*)nol(i),no2(i)
100    continue
C*****
C          JM MATRIX
C*****
      do 200 kk=1,nj
      jm(1,kk)=1
      jm(2,kk)=2
      jm(3,kk)=3
200    continue
C*****
C          SUPPORT CONDITIONS AND COORDINATES
C*****
      do 300 j=1,nj
      read(9,*)jdir(1),jdir(2),jdir(3),xc(j),yc(j)
      do 350 i =1,3
      if (jdir(i).gt.0) then
      jm (i,j) = 0
      endif
350    continue
300    continue
C*****
C          GLOBAL DEGREES OF FREEDOM
C*****
      iqh=0
      do 510 j=1,nj
      do 500 l=1,3
      if (jm(l,j).ne.0) then
      iqh=iqh+1
      jm (l,j)=iqh
      endif
500    continue
510    continue
      iqgn=iqh+nel
      iqg=iqh
      ntot=iqgn+2*nel
C*****
C          FILLING LM MATRIX
C*****

```

```

common /vgeom/ cl(100),cosl(100),cos2(100),lm(6,100),
*          cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/ iqh,iqg,nel,ntot,iqgn
open ( 9,file='data',form='formatted' )
rewind 9
C*****
C          # ELEMENTS AND # JOINTS
C*****
      read (9,*)nel,nj
C*****
C          NODES PER ELEMENT
C*****
      do 100 i=1,nel
      read(9,*)nol(i),no2(i)
100      continue
C*****
C          JM MATRIX
C*****
      do 200 kk=1,nj
      jm(1,kk)=1
      jm(2,kk)=2
      jm(3,kk)=3
200      continue
C*****
C          SUPPORT CONDITIONS AND COORDINATES
C*****
      do 300 j=1,nj
      read(9,*)jdir(1),jdir(2),jdir(3),xc(j),yc(j)
      do 350 i=1,3
      if (jdir(i).gt.0) then
      jm (i,j) = 0
      endif
350      continue
300      continue
C*****
C          GLOBAL DEGREES OF FREEDOM
C*****
      iqh=0
      do 510 j=1,nj
      do 500 l=1,3
      if (jm(l,j).ne.0) then
      iqh=iqh+1
      jm (l,j)=iqh
      endif
500      continue
510      continue
      iqgn=iqh+nel
      iqg=iqh
      ntot=iqgn+2*nel
C*****
C          FILLING LM MATRIX
C*****

```

end

```

=====
subroutine valobf(x,vof)
implicit double precision (a-h,o-z)
dimension x(1)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*          cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/ iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),ck(6,6)
common /equal/ vah(100),vmu(100)
vof = 0.0
do 100 k = 1,nel
base = x(3*k-2)
height = x(3*k-1)
steel = x(3*k)
area = base * height
vof=vof+(area+steel*10)*cl(k)
100 continue
return
end
=====

```

```

subroutine inecon(x)
implicit double precision (a-h,o-z)
dimension x(1)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*          cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/ iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),ck(6,6)
common /equal/ vah(100),vmu(100)
nel3=3*nel
C*****
C          RELIABILITY
C*****
do 200 k=1,iqg
vag(k) = relind-beta(k)
200 continue
return
end
=====

```



```

=====

subroutine modsti(kel,tinert,area,vksi,vksj)
implicit double precision (a-h,o-z)
common /vgeom/ cl(100),cosl(100),cos2(100),lm(6,100),
*          cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/ iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),ck(6,6)
common /equal/ vah(100),vmu(100)
C*****
C          FLEXIBILITY MATRIX (2x2)
C*****
      n=nel
      do 10 i=1,6
      do 20 j=1,6
      ck(i,j)=0.0
20    continue
10    continue
      xd=cl(kel)/(3*ec*tinert)+1./vki
      y=cl(kel)/(3*ec*tinert)+1./vkj
      z=-cl(kel)/(6*ec*tinert)
C*****
C          INVERSION OF MATRIX
C*****
      det=xd*y-z*z
      a=y/det
      b=-z/det
      c=b
      dd=xd/det
C*****
C          EXPANDED MATRIX (6x6)
C*****
      ck11=ec*area/cl(kel)
      ck14=-ck11
      ck41=-ck11
      ck44=ck11
      ck22=(a+b+c+dd)/(cl(kel)*cl(kel))
      ck25=-ck22
      ck52=ck25
      ck55=ck22
      ck23=(a+c)/cl(kel)
      ck53=-ck23
      ck26=(b+dd)/cl(kel)
      ck56=-ck26
      ck33=a
      ck36=b
      ck63=c
      ck66=dd
      ck32=(a+b)/cl(kel)
      ck35=-ck32
      ck62=(c+dd)/cl(kel)

```

```

      ck65=-ck62
C*****
C      ROTATED MATRIX
C*****
      cs1=cos1(kel)
      cs2=cos2(kel)
      c2=cs1*cs1
      s2=cs2*cs2
      cs=cs1*cs2
      ck(1,1)=ck11*c2+ck22*s2
      ck(1,2)=ck11*cs-ck22*cs
      ck(1,3)=-ck23*cs2
      ck(1,4)=-ck11*c2-ck22*s2
      ck(1,5)=-ck11*cs+ck22*cs
      ck(1,6)=-ck26*cs2
      ck(2,1)=ck11*cs-ck22*cs
      ck(2,2)=ck11*s2+ck22*c2
      ck(2,3)=ck23*cs1
      ck(2,4)=-ck11*cs+ck22*cs
      ck(2,5)=-ck11*s2-ck22*c2
      ck(2,6)=ck26*cs1
      ck(3,1)=-ck32*cs2
      ck(3,2)=ck32*cs1
      ck(3,3)=ck33
      ck(3,4)=-ck(3,1)
      ck(3,5)=-ck(3,2)
      ck(3,6)=ck36
      ck(4,1)=-ck(1,1)
      ck(4,2)=-ck11*cs+ck22*cs
      ck(4,3)=ck23*cs2
      ck(4,4)=ck11*c2+ck22*s2
      ck(4,5)=ck(2,1)
      ck(4,6)=ck26*cs2
      ck(5,1)=ck(2,4)
      ck(5,2)=ck(2,5)
      ck(5,3)=-ck(2,3)
      ck(5,4)=-ck(2,4)
      ck(5,5)=-ck(2,5)
      ck(5,6)=-ck26*cs1
      ck(6,1)=-ck62*cs2
      ck(6,2)=ck62*cs1
      ck(6,3)=ck36
      ck(6,4)=-ck(6,1)
      ck(6,5)=-ck(6,2)
      ck(6,6)=ck66
      return
      end

```

```

=====
=====

subroutine sysrel(jflag)
implicit double precision (a-h,o-z)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),

```

```

*          cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),no1(100),
*          no2(100)
common /pari/ iqh,iqg,nel,ntot,iqgn
common /sysr/ betmin,epsilo,elerel(100),invmec(100)
common /equal/ vah(100),vmu(100)
dimension theta(200,100),p(100),rb(200,100),
*   become(500),lc(300),thesum(100),temp(100,100),
*   locmec(100),thesul(100),dispsu(100),ni(100),
*   nucomb(100),lct(300)
C*****
C          form fundamental mechanisms
C*****
call mecsys(nel,iqh,cl,cos1,cos2,lm,numec,rb,theta)
ndof=iqh
do 20 j=1,ndof
    p(j)=r(j)
20  continue
C*****
C          ordering theta and r matrices
C*****
do 710 k=1,numec-1
    jflag=0
    do 720 i=1,ndof
        if(abs(rb(i,k)).gt.0.0)jflag=i
720  continue
    if (jflag.eq.0)then
        do 730 l=k+1,numec
            do 740 li=1,ndof
                if(abs(rb(li,l)).gt.0.0)then
                    do 750 lj=1,ndof
                        temp(lj,l)=rb(lj,l)
                        rb(lj,l)=rb(lj,jflag)
                        rb(lj,jflag)=temp(lj,l)
750  continue
                    do 760 lj=1,2*nel
                        temp(lj,l)=theta(lj,l)
                        theta(lj,l)=theta(lj,jflag)
                        theta(lj,jflag)=temp(lj,l)
760  continue
                    go to 733
                endif
            continue
        continue
730  continue
    endif
710  continue
C*****
C          normalizing theta and r vectors
C*****
do 810 i=1,numec
    do 820 j=1,2*nel
        if(abs(theta(j,i)).ne.1.0.and.
*          theta(j,i).ne.0.0)then
            fact=abs(1./theta(j,i))
            do 830 jj=1,2*nel

```

```

830                                theta(jj,i)=theta(jj,i)*fact
                                continue
                                do 840 jj=1,ndof
                                    rb(jj,i)=rb(jj,i)*fact
840                                continue
                                go to 734
                                endif
820                continue
734        continue
810    continue
c*****
c            transpose theta and r matrices
c*****
      do 25 j=1,numec
      do 91 i=nel
      temp(i,j)=theta(i,j)
91      continue
25      continue
      do 56 i=1,numec
      do 55 j=1,2*nel
      theta(i,j)=temp(j,i)
55      continue
56      continue
      do 28 j=1,numec
      do 27 i=1,ndof
      temp(i,j)=rb(i,j)
27      continue
28      continue
      do 66 i=1,numec
      do 65 j=1,ndof
      rb(i,j)=temp(j,i)
65      continue
66      continue
c*****
c            reliability of fundamental mechanisms
c*****
      do 102 i=1,numec
      vmeanr=0.0
      stdevr=0.0
          do 202 k=1,nel
              j=2*k-1
              theji=abs(theta(i,j))
              thejil=abs(theta(i,j+1))
              if(theji.lt.0.0001.and.thejil.lt.
*                  0.0001)goto 202
              term2=(theji+thejil)*cvmu(k)*vmu(k)
              stdevr=stdevr+term2*term2
              vmeanr=term2/cvmu(k)+vmeanr
202          continue
          vmeanl=0.0
          stdevl=0.0
          do 302 k=1,ndof
              if(abs(p(k)).lt.0.001)go to 302
              term=p(k)*rb(i,k)
              vmeanl=vmeanl+term
              stdevl=stdevl+term*term*(cvload(k)

```

```

*                                *vload(k))
302      continue
      vmean=vmeanr-vmeanl
      stdev=stdevr+stdevl
      become(i)=vmean/sqrt(stdev)
      lnumbe=lnumbe+1

102      continue
C*****
C      reliability of combined mechanisms
C      (internal work)
C*****
      do 50 l=1,numec
        lc(l)=1
        lct(l)=1
50      continue
        lp=numec
        ltemp=numec
        ni(l)=1
        numax=6
        nucome=1
        nucomb(l)=numec
        lpt=numec
        lpti=numec+1
111      continue
C*****
C      loop over mechanisms in location vector
C*****
      do 699 kmo=1,nel
        thesul(kmo)=0.0
        thesum(kmo)=0.0
699      continue
        do 698 klp=1,ndof
          disp su(klp)=0.0
698      continue
          kmu=1
          nic=ni(nucome)
          nif=nic+nucomb(nucome)*nucome-1
          nucomb(nucome+1)=0
C*****
C      define acceptable interval
C*****
          if(nucome.gt.1)then
            nifbet=lpti-1
            nicbet=lpti-nucomb(nucome)
            do 5544 ia=nicbet+1,nifbet
              betaal=become(nicbet)+epsilo
              if(become(ia).gt.betaal)then
                do 5545 ib=ia,nifbet
                  become(ib)=1000.
5545      continue
                  go to 5541
                endif
            endif
5544      continue
5541      continue
            niccon=nicbet
          endif

```

```

C*****
C      reliability of combined mechanisms
C      (internal work)
C*****
      do 200 j=nic,nif,nucome
        icontr=0
        if(nucome.gt.1) then
          if(become(niccon).gt.100.) icontr=1
          niccon=niccon+1
          if (icontr.eq.1) go to 5564
        endif
        do 300 l=1, nucome
          lj=lc(j+l-1)
          if(lj.lt.-0.1) then
            kmu=-1
            lj=abs(lj)
          endif
          do 400 kk=1,nel
            kkj=2*kk-1
            thesum(kk)=thesum(kk)+theta(lj,kkj)*kmu
            thesul(kk)=thesul(kk)+theta(lj,kkj+1)*kmu
400          continue
            kmu=1
300          continue
C*****
C      reliability of combined mechanisms
C      (external work)
C*****
        do 372 l=1, nucome
          lcl=lc(j+l-1)
          if(lcl.lt.-0.1) then
            kmu=-1
            lcl=abs(lcl)
          endif
          do 472 kk=1,ndof
            if(abs(p(kk)).lt.0.001) go to 472
            dispsu(kk)=dispsu(kk)+rb(lcl,kk)*kmu
472          continue
            kmu=1
372          continue
5564          continue
C*****
C      combination with fundamental mechanisms
C      (internal work)
C*****
        do 100 k=1,numec
          vmeanr=0.0
          vmeanl=0.0
          stdevr=0.0
          stdevl=0.0
          vmanrm=0.0
          vmanlm=0.0
          stdvrm=0.0
          stdvlm=0.0
          do 499 lll=j,j+nucome-1
            if(abs(lc(lll)).ge.k) go to 100

```

499

```

continue
if(nucome.gt.1)then
    if (icontr.eq.1)then
        becomi=400.
        becopl=500.
        go to 5574
    endif
endif
thesu=0.0
thesu2=0.0
thesui=0.0
theslm=0.0
do 600 kk=1,nel
    kkj=2*kk-1
    thesu=thesum(kk)+theta(k,kkj)
    thesu2=thesul(kk)+theta(k,kkj+1)
    term=(abs(thesu)+abs(thesu2))*cvmu(kk)
    *
    vmeanr=term/cvmu(kk)+vmeanr
    stdevr=stdevr+term*term
    thesui=thesum(kk)-theta(k,kkj)
    theslm=thesul(kk)-theta(k,kkj+1)
    termm=(abs(thesui)+abs(theslm))*cvmu(kk)
    *
    vmanrm=termm/cvmu(kk)+vmanrm
    stdvrm=stdvrm+termm*termm
600 continue
C*****
C combination with fundamental mechanisms
C (external work)
C*****
do 672 kk=1,ndof
    if(abs(p(kk)).lt.0.001)go to 672
    dispkk=(dispsu(kk)+rb(k,kk))*p(kk)
    vmeanl=vmeanl+dispkk
    stdevl=stdevl+dispkk*cvload(kk)*dispkk
    *
    dispkm=(dispsu(kk)-rb(k,kk))*p(kk)
    vmanlm=vmanlm+dispkm
    stdvlm=stdvlm+dispkm*cvload(kk)*dispkm
    *
672 continue
becopl=(vmeanr-vmeanl)/sqrt(stdevr+stdevl)
becomi=(vmanrm-vmanlm)/sqrt(stdvrm+stdvlm)
5574 continue
if(becomi.lt.becopl) then
    do 138 lk=1,nucome
        ltemp=ltemp+1
        lc(ltemp)=lc(j+lk-1)
138 continue
        ltemp=ltemp+1
        lc(ltemp)=-k
        lctlp=-k
        becote=becomi
    else
        do 139 lk=1,nucome

```

```

                                ltemp=ltemp+1
                                lc(ltemp)=lc(j+lk-1)
139      continue
                                lctlp=k
                                ltemp=ltemp+1
                                lc(ltemp)=k
                                becote=becopl
                                endif
                                lpt=lpt+1
                                iflag=0
C*****
C      ordering the combined beta values in the same row
C*****
                                if (lpti.gt.lpt-1)then
                                    do 524 lk=1,nucome
                                        lp=lp+1
                                        lct(lp)=lc(j+lk-1)
524      continue
                                        lp=lp+1
                                        lct(lp)=lctlp
                                        become(lpt)=becote
                                else
                                    do 510 jkj=lpti,lpt-1
                                        if(become(jkj).gt.becote)then
                                            iflag=-1
                                            do 511 kjk=jkj,lpt-1
                                                itemp=lpt-1+jkj-kjk
                                                become(itemp+1)=become(itemp)
511      continue
                                                become(jkj)=becote
                                                becote=become(lpt)
C*****
C      moving lc array
C*****
                                movini=nic+nucomb(nucome)*nucome+
                                *                               (jkj-lpti)*(nucome+1)
                                movfin=(lpt-lpti)*(nucome+1)+nic+
                                *                               nucomb(nucome)*nucome-1
                                do 512 lmn=movini,movfin
                                    lcou=movini+movfin-lmn
                                    nucl=nucome+1
                                    lct(lcou+nucl)=lct(lcou)
512      continue
                                    do 513 n=movini,movini+nucome-1
                                        lptaa=n-movini+1
                                        lct(n)=lc(j+lptaa-1)
513      continue
                                    lct(movini+nucome)=lctlp
                                endif
510      continue
                                if(iflag.eq.0)then
                                    do 124 lk=1,nucome
                                        lp=lp+1
                                        lct(lp)=lc(j+lk-1)
124      continue
                                        lp=lp+1

```



```

                                lct(lp)=lctlp
                                become(lpt)=becote
                                else
                                    lp=lp+nucome+1
                                endif
                            endif
                            nucomb(nucome+1)=nucomb(nucome+1)+1
                            lnumbe=lnumbe+1
100      continue
        do 6991 kmo=1,nel
            thesul(kmo)=0.0
            thesum(kmo)=0.0
6991      continue
        do 6981 klp=1,ndof
            dispsu(klp)=0.0
6981      continue
200      continue
c*****
c      control of maximum number of tree rows
c*****
        lpti=lpti+nucomb(nucome+1)
        nucome=nucome+1
        ni(nucome)=ni(nucome-1)+(nucome-1)*nucomb(nucome-1)
        if(nucome.lt.(numax))go to 111
c*****
c      find minimum beta value
c*****
        betmin=100
        do 7890 i=1,lnumbe
            if(become(i).lt.betmin)then
                betmin=become(i)
                itab=i
            endif
7890      continue
c*****
c      find mechanisms involved
c*****
        mecoun=0
        mcomb=1
        do 7891 j=1,nucome-1
            do 7895 jk=1,nucomb(j)
                mecoun=1+mecoun
                if(itab.eq.mecoun)then
                    nistar=mcomb
                    do 7893 l=1,j
                        locmec(l)=lct(nistar+1-1)
7893      continue
                    nummec=j
                    go to 7894
                endif
            enddo
            mcomb=j+mcomb
7895      continue
7891      continue
7894      continue
c*****
c      find elements involved

```

```

C*****
      nk=1
      numele=0
      do 8000 m=1,nel
        do 8001 k=1,numec
          do 8002 l=1,nummec
            if(abs(locmec(l)).eq.k)then
              m1=2*m-1
              m2=2*m
              if(abs(theta(m1,k)).gt.0.0.
*              or.abs(theta(m2,k)).gt.0.0)then
                if(nk.gt.1)then
                  do 8003 lmi=1,nk
                    if(invmec(lmi).eq.m)
*                    go to 8004
8003                continue
                    endif
                    nk=nk+1
                    numele=numele+1
                    invmec(n)=m
                    go to 8000
8004                continue
                    endif
            endif
          endif
        continue
      continue
8001    continue
8000    continue
C*****
C      control of system reliability
C*****
      write(3,*)
      write(3,*)'BETA MINIMAL FOR THE SYSTEM = ',betmin
      write(3,*)
      jflag=0
      if(betmin.lt.relind)then
        delta=(relind-betmin)/relind
        do 3891 i=1,nel
          do 3892 j=1,numele
            if(invmec(j).eq.i)then
              elerel(i)=(1.+delta)*elerel(i)
            endif
3892          continue
3891        continue
          jflag=1
        endif
      return
      end

```

```

=====
=====

subroutine mecsys(n,igh,cl,cost,sint,lm,numec,r,theta)

```

```

implicit double precision (a-h,o-z)
dimension a(100,100),b(100,100),c(100),cm(100,100),
*   cost(n),qa(100,100),sint(n),cl(n),q(100,100),
*   lm(6,n),am(100,100),bl(100,100),theta(200,100),
*   r(iqh,100)
C*****
C   Constraint matrix for the structure
C*****
      do 300 i=1,3*n
        do 400 j=1,6*n
          cm(i,j)=0.0
          q(i,j)=0.0
400      continue
300    continue
      do 60 k=1,n
        i=3*k
        j=6*k
        at=1.0/cl(k)
        im2=i-2
        im1=i-1
        jm1=j-1
        jm2=j-2
        jm3=j-3
        jm4=j-4
        jm5=j-5
        cm(im2,jm5)=-1.0
        cm(im2,jm2)=1.0
        cm(im1,jm3)=1.0
        cm(i,j)=1.0
        cm(im1,jm4)=-at
        cm(im1,jm1)=at
        cm(i,jm4)=-at
        cm(i,jm1)=at
60    continue
C*****
C   Coordinate trnsformation matrix
C*****
      do 70 k=1,n
        co=cost(k)
        si=sint(k)
        j=6*k
        do 80 i=1,2
          jum=j-3*i+1
          jdois=j-3*i+2
          jtres=j-3*i+3
          q(jum,jum)=co
          q(jum,jdois)=si
          q(jdois,jum)=-si
          q(jdois,jdois)=co
          q(jtres,jtres)=1.0
80      continue
70    continue
C*****
C   Compatibilibity matrix from LM matrix
C*****
      n6=6*n

```

```

do 500 i=1,n6
    do 510 k=1,iqh
        am(i,k)=0.0
510    continue
500    continue
do 520 i=1,6
    do 530 k=1,n
        iflag=lm(i,k)
        if (iflag.gt.0) am(6*(k-1)+i,iflag)=1.0
530    continue
520    continue
C*****
C      Rotation of basic compatibility matrix
C*****
mm=80
call multi(q,am,qa,n6,iqh,n6,mm)
C*****
C      Expansion of QA matrix
C*****
do 585 i=1,n
    i6=6*i
    i3=i6-3
    do 595 j=1,iqh
        qa(i3,j)=0.0
        qa(i6,j)=0.0
595    continue
    i2=2*i
    qa(i3,iqh+i2-1)=1.0
    qa(i6,iqh+i2)=1.0
585    continue
C*****
C      Matrix A = C * QA (transformed)
C*****
m=3*n
nt=iqh+2*n
call multi(cm,qa,a,m,nt,n6,mm)
C*****
C      Solution for virtual displacements
C*****
do 150 k=1,nt
    do 175 l=1,nt
        b(k,l)=0.0
175    continue
150    continue
do 160 k=1,nt
    b(k,k)=1.0
160    continue
do 200 i=1,m
    amax=0.0
    iflag=0
    do 250 j=i,nt
        if (abs(a(i,j)).gt.amax) then
            iflag=j
            amax=abs(a(i,j))
        endif
250    continue

```

```

do 305 k=1,m
    c(k)=a(k,iflag)
    a(k,iflag)=a(k,i)
    a(k,i)=c(k)
305    continue
    do 310 k=1,nt
        c(k)=b(k,iflag)
        b(k,iflag)=b(k,i)
        b(k,i)=c(k)
310    continue
    do 280 j=i+1,nt
        if (abs(a(i,j)).gt.0.00001) then
            fact=-a(i,j)/a(i,i)
            do 290 kk=1,m
                a(kk,j)=a(kk,j)+a(kk,i)*fact
290            continue
            do 291 kk=1,nt
                b(kk,j)=b(kk,j)+b(kk,i)*fact
291            continue
            endif
280        continue
200    continue
    numec=nt-3*n
C*****
C    Forming b1
C*****
    lcount=nt-numec+1
    ki=1
    do 800 i=lcount,nt
        do 810 j=1,nt
            b1(j,ki)=b(j,i)
810        continue
        ki=ki+1
800    continue
C*****
C    Creating Theta matrix
C*****
    do 156 j=1,numec
        do 157 i=1,2*n
            k=iqh+i
            theta(i,j)=b1(k,j)
157        continue
156    continue
C*****
C    Creating virtual displacements
C*****
    do 169 j=1,numec
        do 158 i=1,iqh
            r(i,j)=b1(i,j)
158        continue
C*****
C    Adding joint mechanisms
C*****
    do 161 i=3,iqh,3
        if (abs(r(i,j)).gt.0.000001) then
            r(i,j)=0.0

```

```

do 162 k=1,n
  if(lm(3,k).eq.i)then
    lpo=2*k-1
    theta(lpo,j)=1
  endif
  if(lm(6,k).eq.i)then
    lpo=2*k
    theta(lpo,j)=1
  endif
162 continue
endif
161 continue
169 continue
return
end

```

```

=====

subroutine multi(aa,bb,cc,l,m,n,k)
implicit double precision (a-h,o-z)
dimension aa(1,n),bb(n,m),cc(1,m)
do 10 i=1,l
  do 20 j=1,m
    d=0.0
    do 30 kk=1,n
      d=d+aa(i,kk)*bb(kk,j)
30      continue
      cc(i,j)=d
20      continue
10      continue
return
end

```

```

=====

subroutine equcon(x)
implicit double precision (a-h,o-z)
dimension x(1)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*               cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*               no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/  cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/  iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),ck(6,6)
common /equal/ vah(100),vmu(100)
common /springs/ vksi(100),vksj(100)

```

```

      do 121 ikl=1,igh
        vahk(ikl)=0.0
121    continue
      n=nel
      n3=n+n+n
      do 100 k = 1,n
        sigma2=0.0
        do 111 ipo=1,6
          u(ipo)=0.0
111    continue
        do 200 i = 1,6
          m=lm(i,k)
          if (m.eq.0) go to 200
          if(cvload(m).gt.sigma2)sigma2=cvload(m)
          l = n3 + m
          u(i) = x(l)
200    continue
        c1=cos1(k)
        c2=cos2(k)
        d2=-c2*u(1)+c1*u(2)
        d3=u(3)
        d5=-c2*u(4)+c1*u(5)
        d6=u(6)
C*****
C          element forces
C*****
        c = c1(k)
        base = x(3*k-2)
        height = x(3*k-1)
        aste = x(3*k)
        area = base * height
        tinert = area*height*height/12.0
        al = ec*tinert/(c*c*c)
        call eley (ec,tinert,c,vksi(k),vksj(k),d2,d3,d5,d6,
          *          fo3,fo6)
        signal=cvmu(k)
C*****
C          ultimate and yield moments
C*****
        call newmum(k,x,signal,sigma2,fo3,fo6,vksi(k),vksj(k),
          *          d2,d5,d3,d6)
C*****
C          global modified stiffness
C*****
        call modsti(k,tinert,area,vksi(k),vksj(k))
        do 300 l=1,6
          j=lm(l,k)
          if (j.eq.0) go to 300
          do 400 ll=1,6
            m = lm(ll,k)
            if (m.eq.0) go to 400
            jj = n3+m
            vahk(j)=vahk(j)+ck(l,ll)*x(jj)
400        continue
300    continue
100    continue

```

```

C*****
C      subtraction of external global forces
C*****
      rmax=0.01
      do 510 i=1,igh
          if(abs(r(i)).gt.rmax)rmax=abs(r(i))
510      continue
      do 500 k=1,igh
          if(abs(r(k)).lt.0.0001)then
              vah(k)=vahk(k)/rmax
              go to 500
          endif
          vah(k)=(vahk(k) - r(k))/rmax
500      continue
      return
      end

=====
=====

      subroutine mummy(kel,x,sigma1,sigma2,fo3,fo6,vki,vkj,displ,
*          disp2,rot1,rot2)
      implicit double precision (a-h,o-z)
      dimension x(1)
      common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*          cvmu(100)
      common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
      common /xcord/ xc(100),yc(100),d(100),jdir(3)
      common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
      common /pari/ igh,igg,nel,ntot,iggn
      common /inequa/ vag(100),beta(100),u(6),vahk(100),ck(6,6)
      common /equal/ vah(100),vmu(100)
      common /springs/ vksi(100),vksj(100)
      n=nel
      node1=0
      node2=0
      b=x(3*kel-2)
      h=x(3*kel-1)
      dd=h-co
      aste=x(3*kel)
      clk=cl(kel)
      ei=57000.*sqrt(ec)*h*h*h*b/12.
C*****
C      evaluation of yielding moment
C*****
      call comcon(aste,dd,b,vmy,phiy)
      afo3=abs(fo3)
      afo6=abs(fo6)
      vm=max(afo3,afo6)
C*****
C      ultimate moment and reliability
C*****
hl2v0s0b0a4w1l5b1ha0a8te,b,kel,dd,vm,vmy,phiy,phiu,sigma1,sigma2)

```



```

      vmuk=vmu(kel)
C*****
C      integration of curvature
C*****
      if(vmy.lt.afo3) node1=1
      if(vmy.lt.afo6) node2=1
      if(node1.eq.1.or.node2.eq.1) then
        if((fo3*fo6).gt.0.) then
          if(afo3.ge.afo6) then
            tetay=(vmy/(3.*ei)+afo6/(6.*ei))*clk
            vlp=(afo3-vmy)/(afo3-afo6)*clk
            if(abs(vlp).gt.clk) vlp=clk
            tetal=(vmuk/(3.*ei)+afo6/(6.*ei))*clk
            tetau=tetal+(phiu-phiy)*vlp
          endif
          if(afo3.lt.afo6) then
            tetay=(vmy/(3.*ei)+afo3/(6.*ei))*clk
            vlp=(afo6-vmy)/(afo6-afo3)*clk
            if(abs(vlp).gt.clk) vlp=clk
            tetal=(vmuk/(3.*ei)+afo3/(6.*ei))*clk
            tetau=tetal+(phiu-phiy)*vlp
          endif
        endif
        if((fo3*fo6).lt.0.) then
          if(afo3.ge.afo6) then
            tetay=(vmy/(3.*ei)-afo6/(6.*ei))*clk
            vlp=(afo3-vmy)/(afo3+afo6)*clk
            tetal=(vmuk/(3.*ei)-afo6/(6.*ei))*clk
            tetau=tetal+(phiu-phiy)*vlp
          endif
          if(afo3.lt.afo6) then
            tetay=(vmy/(3.*ei)-afo3/(6.*ei))*clk
            vlp=(afo6-vmy)/(afo3+afo6)*clk
            tetal=(vmuk/(3.*ei)-afo3/(6.*ei))*clk
            tetau=tetal+(phiu-phiy)*vlp
          endif
        endif
      endif
      crot1=rot1-(-displ/clk+disp2/clk)
      crot2=rot2-(-displ/clk+disp2/clk)
      vksp=(vmuk-vmy)/(tetau-tetay)
C*****
C      spring rotation
C*****
      srot1=abs(crot1)-tetay
      srot2=abs(crot2)-tetay
C*****
C      new secant spring values
C*****
      if(node1.eq.1) then
        if(srot1.le.0.) then
          vki=vmy/tetay
          go to 123
        endif
        vki=(vksp*srot1+vmy)/abs(crot1)
        if(vm.gt.vmuk) vki=vmuk/tetau
      endif

```

```

123         if(abs(crot1).gt.tetau) vki=vmuk/tetau
            continue
        endif
        if(node2.eq.1) then
            if(srot2.le.0.) then
                vkj=vmy/tetay
                go to 124
            endif
            vkj=(vksp*srot2+vmy)/abs(crot2)
            if(vm.gt.vmuk) vkj=vmuk/tetau
            if(abs(crot2).gt.tetau) vkj=vmuk/tetau
124         continue
        endif
        if(node1.eq.1.and.node2.eq.1) then
            if(srot1.le.0.) then
                vki=vmy/tetay
                go to 125
            endif
            vki=(vksp*srot1+vmy)/abs(crot1)
            if(vm.gt.vmuk) vki=vmuk/tetau
            if(abs(crot1).gt.tetau) vki=vmuk/tetau
125         continue
            if(srot2.le.0.) then
                vkj=vmy/tetay
                go to 126
            endif
            vkj=(vksp*srot2+vmy)/abs(crot2)
            if(vm.gt.vmuk) vkj=vmuk/tetau
            if(abs(crot2).gt.tetau) vkj=vmuk/tetau
126         continue
        endif
        return
    end

```

```

=====
=====

```

```

subroutine valmu(aste,b,kel,dd,vm,vmy,phiy,phiu,
*          sigma1,sigma2)
implicit double precision (a-h,o-z)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*          cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*          no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/ cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/ iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),
*          ck(6,6)
common /equal/ vah(100),vmu(100)
n=nel
C*****

```

```

c          neutral axis
c*****
      x=47./60.*b*fc
      y=0.004*es*aste-aste*fy
      z=-0.004*es*co*aste
      vkd=(-y+sqrt(y*y-4.*x*z))/(2.*x)
      epcs=0.004*(vkd-co)/vkd
      if(epcs.gt.epsy) then
          epcs=epsy
      endif
c*****
c          concrete force in region ab
c*****
      alphas=2./3.
      ccab=alpha*b*0.5*vkd*fc
c*****
c          concrete force in region bc
c*****
      alpha2=0.9
      ccbc=alpha2*b*0.5*vkd*fc
c*****
c          distance of centroid to top in ab
c*****
      gamal=0.875*vkd
c*****
c          distance of centroid to top in bc
c*****
      gama2=0.259255*vkd
c*****
c          coefficients for failure function
c*****
      a1=(ccab*(dd-gamal)+ccbc*(dd-gama2))/fc
      a2=-1.
c*****
c          cosine directors
c*****
      tetal=a1*sigma1*fc
      teta2=a2*sigma2*vm
c*****
c          independent term
c*****
      fps=0.004*es*(vkd-co)/vkd
      bi=aste*fps*(dd-co)
c*****
c          reliability index
c*****
      beta(kel)=(a1*fc+a2*vm+bi)/sqrt(tetal*tetal+teta2*teta2)
c*****
c          ultimate moment and rotation
c*****
      vmu(kel)=a1*fc+bi
      phiu=0.004/vkd
      vmuk=vmu(kel)
      if((4.*phiy).lt.phi) then
          vmu(kel)=(vmuk-vmu)/(phiu-phiy)*3*phiy+vmu
          phiu=3*phiy

```

```

endif
return
end

```

```

=====
=====

```

```

subroutine comcon(aste,dd,b,vmy,phiy)
implicit double precision (a-h,o-z)
dimension x(1)
common /vgeom/ cl(100),cos1(100),cos2(100),lm(6,100),
*               cvmu(100)
common /vload/ r(100),cvload(100),jm(6,100),nol(100),
*               no2(100)
common /xcord/ xc(100),yc(100),d(100),jdir(3)
common /parr/  cv,ec,rp,fc,es,ecm,relind,co,fy,epsy
common /pari/  iqh,iqg,nel,ntot,iqgn
common /inequa/ vag(100),beta(100),u(6),vahk(100),
*               ck(6,6)
common /equal/ vah(100),vmu(100)
node=0
epso=0.002
C*****
C
C               exc - concrete strain
C               epcs - compressive steel strain
C               epsy - yield strain
C
C*****
C               first value for a
C*****
      al=dd/2.
      exc=al*epsy/(dd-al)
      epcs=exc*(al-co)/al
      t=fy*aste
      cs=epcs*es*aste
      eces=exc/epso
      alpha=eces-eces*eces/3.
      cc=alpha*fc*b*al
      res1=cc+cs-t
C*****
C               second value for a
C*****
      a2=0.25*dd
      exc=a2*epsy/(dd-a2)
      epcs=exc*(a2-co)/a2
      cc=fc*a2*alpha*b
      eces=exc/epso
      cs=epcs*es*aste
      res2=cc+cs-t
C*****
C               newton iteration
C*****
100   a=a2-res2*(a2-al)/(res2-res1)

```

```

      exc=a*epsy/(dd-a)
      if(exc.gt.epso) go to 200
C*****
C      parabolic shape
C*****
      epcs=exc*(a-co)/a
      eces=exc/epso
      alpha=ecses-ecses*ecses/3.
      cc=fc*alpha*b*a
      cs=epcs*es*aste
      res=cc+cs-t
      control=0.0001*b*dd*fc
      if (abs(res).gt.control) then
          a1=a2
          a2=a
          res1=res2
          res2=res
          go to 100
      endif
      gama=1.-(8.*epso-3.*exc)/(12.*epso-4.*exc)
      arm=dd-gama*a
      vmy=cc*arm+epcs*es*aste*(dd-co)
      phiy=epsy/(dd-a)
      return
C*****
C      concrete strain > epso
C*****
200  if(exc.gt.0.004) exc=0.004
      x1=epso*a/exc
      ccl=fc*x1*2./3.*b
      gama=3.6*exc*exc-200.*exc*exc*exc-0.0000128
      gama=gama/((exc-epso)*(7.2*exc-300*exc*exc-0.0132))-1.
      alpha=exc-50.*exc*exc+100.*exc*epso-0.0022
      alpha=alpha/(exc-epso)
      cc2=alpha*fc*(a-x1)*b
      epcs=exc*(dd-co)/a
      t=fy*aste
      cs=epcs*es*aste
      cc=ccl+cc2
      res=cc+cs-t
      control=0.0001*b*dd*fc
      if (abs(res).gt.control) then
          a1=a2
          a2=a
          res1=res2
          res2=res
          go to 100
      endif
      arml=dd-a+2./3.*x1
      arm2=dd-gama*(a-x1)
      vmy=ccl*arml+epcs*es*(dd-co)*aste+cc2*arm2
      phiy=epsy/(dd-a)
      return
end

```

```

=====
=====
program eley
implicit double precision(a-h,o-z)
open(1,file='ydata',form='formatted')
rewind 1
read(1,*)ec,tinert,cl,vki,vkj,u2,u3,u5,u6
ei=ec*tinert
w=cl/(3.*ei)+1./vki
y=cl/(3.*ei)+1./vkj
z=-cl/(6.*ei)
det=w*y-z*z
a=y/det
b=-z/dt
c=b
d=w/det
fo3=(a+b)/cl*u2+a*u3-(a+b)/cl*u5+b*u6
fo6=(c+d)/cl*u2+c*u3-(c+d)/cl*u5+d*u6
write(*,*)'fo3 = ',fo3,' fo6 = ',fo6
stop
end

```

```

=====
=====
program yiel
implicit double precision (a-h,o-z)
open(1,file='yielm',form='formatted')
rewind 1
read(1,*)b,d,aste,epsy,es,co,fy,fc,ecm,vm,sigma1,sigma2
ec=3122019
node=0
epso=0.002

```

```

C*****
C
C          EXC - CONCRETE STRAIN
C          EPCS - COMPRESSIVE STEEL STRAIN
C          EPSY - YIELD STRAIN
C
C          FIRST VALUE FOR A
C*****
dd=x(3*kel-1)-co
al=dd/2.
exc=al*epsy/(dd-al)
epcs=exc*(al-co)/al
t=fy*aste
cs=epcs*es*aste
eces=exc/epso
alpha=eces-eces*eces/3.
cc=alpha*fc*b*al

```

```

      res1=cc+cs-t
C*****
C      SECOND VALUE FOR A
C*****
      a2=0.25*dd
      exc=a2*epsy/(dd-a2)
      epcs=exc*(a2-co)/a2
      cc=fc*a2*alpha*b
      eces=exc/epso
      cs=epcs*es*aste
      res2=cc+cs-t
C*****
C      NEWTON ITERATION
C*****
100   a=a2-res2*(a2-a1)/(res2-res1)
      exc=a*epsy/(dd-a)
      if(exc.gt.epso) go to 200
C*****
C      PARABOLIC SHAPE
C*****
      epcs=exc*(a-co)/a
      eces=exc/epso
      alpha=ecses-ecses*ecses/3.
      cc=fc*alpha*b*a
      cs=epcs*es*aste
      res=cc+cs-t
      control=0.0001*b*dd*fc
      if (abs(res).gt.control) then
        a1=a2
        a2=a
        res1=res2
        res2=res
        go to 100
      endif
      gama=1.-(8.*epso-3.*exc)/(12.*epso-4.*exc)
      arm=dd-gama*a
      vmy=cc*arm+epcs*es*aste*(dd-co)
      phiy=epsy/(dd-a)
C*****
C      CONCRETE STRAIN > EPSO
C*****
200   if(exc.gt.0.004) exc=0.004
      x1=epso*a/exc
      ccl=fc*x1*2./3.*b
      gama=3.6*exc*exc-200.*exc*exc*exc-0.0000128
      gama=gama/((exc-epso)*(7.2*exc-300*exc*exc-0.0132))-1.
      alpha=exc-50.*exc*exc+100.*exc*epso-0.0022
      alpha=alpha/(exc-epso)
      cc2=alpha*fc*(a-x1)*b
      epcs=exc*(dd-co)/a
      t=fy*aste
      cs=epcs*es*aste
      cc=ccl+cc2
      res=cc+cs-t
      control=0.0001*b*dd*fc
      if (abs(res).gt.control) then

```

```

a1=a2
a2=a
res1=res2
res2=res
go to 100
endif
arm1=dd-a+2./3.*x1
arm2=dd-gama*(a-x1)
vmy=cc1*arm1+epcs*es*(dd-co)*aste+cc2*arm2
phiy=epsy/(dd-a)
C*****
C          LINEAR APPROXIMATION
C*****
ro=aste/(b*d)
vn=es/ec
vk=sqrt(4.*ro*ro*vn*vn+2.*(ro+ro*co/d)*vn)-2.*ro*vn
vkd=vk*d
exc=epsy/(d-vkd)*vkd
cc=0.5*vkd*b*ec*exc
excs=exc/vkd*(vkd-co)
fps=excs*es
cs=fps*aste
vmy=cc*(d-vkd/3.)+cs*(d-co)
phiy=exc/vkd
write(*,*) 'cc=',cc,'cs=',cs,'exc=',exc,'excs=',excs
write(*,*) 'linear approx','vmy=',vmy,'phiy=',phiy
stop
end

```


Example: Debug Frame

Input File: DATA

TESTE LINEAR BETAO ARMADO

21	17	9					
8							
1	2.0						
2	6.0						
4	2.0						
5	6.0						
7	2.0						
8	6.0						
10	2.0						
11	6.0						
0							
4							
14	27.8						
15	27.8						
16	27.8						
17	27.8						
5.00	10.00	1.00	5.00	10.00	1.00	5.0	10.0
1.00	5.0	10.0	1.0	.5	.059	.0	.900
.805	.07	.94	.0	-.092			
2							
EPNE	0.1						
EPST	0.000001						
1							
1500	500010000						
1							
0							
0							
4							
2	5	8	11				
0							

User's Manual
Generalized Reduced Gradient Method
Example: Debug Frame
Input File: DATA

- Line 1
 Problem title.
- Line 2
 Number of variables, number of constraints, number of equality constraints.
- Line 3
 Number of variables with lower bounds.
- Line 4 to line 11
 Variable number and respective lower bound.
- Line 12
 Number of constraints with upper bounds.
- Line 13 to line 16
 Constraint number and respective upper bound.
- Line 17 to line 19
 Initial values of design variables.
- Line 20
 Number of prescribed optimization parameters.
- Line 21
 Constraint tolerance.
- Line 22
 Convergence tolerance.
- Line 23

Parameter indicating alteration of the limit of number of iterations.

- Line 24

Maximum number of consecutive iterations without objective function improvement, maximum number of consecutive Newton iterations, maximum number of completed one dimensional searches.

- Line 25

Parameter that controls the quantity of information in the output file.

- Line 26

Number indicating minimum printed information.

- Line 27

Indication that tangent vector extrapolation should be used for estimating initial values of basic variables.

- Line 28

Number of design variables initially included in the basis.

- Line 29

Numbers of design variables of the initial basis.

- Line 30

Parameter that indicates if new data should be read.

Example: Debug Frame

Input File: DATA1

[illegible]

User's Manual

Generalized Reduced Gradient Method

Example: Debug Frame

Input File: DATA1

- Line 1

Number of elements, number of nodes.

- Line 2 to line 5

Node i, node j of element 1 through 4.

- Line 6 to line 10

Boundary conditions of displacement in the horizontal direction, vertical direction, in-plane rotation, horizontal coordinate, vertical coordinate.

- Line 11 and line 12

Node where force is applied, direction of load and magnitude of load.

- Line 13

Termination of force information.

- Line 14

Flexural strength of concrete, yielding stress of steel and reinforcement cover.

- Line 15

Steel modulus of elasticity and concrete ultimate strain.

- Line 16

Minimum element and system reliability index.

- Line 17 to line 20

Coefficient of variation of flexural concrete strength.

- Line 21 to line 29

Coefficient of variation of external global loads.

- Line 30 to line 33

Coefficient of variation of element ultimate moment.

- Line 34

Value of interval gap in the Beta unzipping method.

REFERENCES

- (1) - U. Kirsch, Optimum Structural Design, McGraw-Hill, New York, 1981.
- (2) - Brandt, A. M., Criteria and Methods of Structural Optimization, Warszawa/Martinus Nijhoff Publishers, The Hague, 1984.
- (3) - Spillers, W. R., Iterative Structural Design, North-Holland Publishing Company, Amsterdam, 1975, Appendix A.
- (4) - Magnel, G., Prestressed Concrete, McGraw-Hill, New York, 1954.
- (5) - Schmit, L. A., Structural Design by Systematic Approach, Proceedings of the Second National Conference on Electronic Computation, Structural Division of ASCE, Pittsburgh, 1960.
- (6) - Morris, A. J., Foundations of Structural Optimization: a Unified Approach, John Wiley and Sons, New York, 1982.
- (7) - Fleury, C., and Geradin, M., Optimality Criteria and Mathematical Programming in Structural Weight Design, Computers and Structures, Vol. 8, no. 1, 1978, pg. 7-18.
- (8) - Haftka, R. T., and Kamat, M. P., Elements of Structural Optimization, Martinus Nijhoff, Amsterdam, 1985.
- (9) - Farshi, B., and Schmit, L. A., Minimum Weight Design of Stress Limited Trusses, Journal of Structural Division, ASCE, Vol. 100, no. ST1, 1974, pg. 97-107.
- (10) - Gallagher, R. H., and Zienkiwicz, O. C., Optimum Structural Design, John Wiley and Sons, New York, 1977.
- (11) - Grierson, D. E., and Cameron, G. E., Computer-Automated Synthesis of Building Frameworks, Canadian Journal of Civil Engineering, Vol. 11, no. 4, 1984, pg. 863-874.
- (12) - Vanderplaats, G. N., Numerical Optimization Techniques for Engineering Design, McGraw-Hill, New York, 1984.
- (13) - Wellen, H., and Bartholomew, P., Structural Optimization in Aircraft Construction, Computer Aided Optimal Design: Structural and Mechanical Systems, Springer-Verlag, Berlin, 1986.
- (14) - Avriel, M., Nonlinear Programming, Prentice-Hall, Englewood Cliffs, New Jersey, 1976.

- (15) - Watwood, V. B., Mechanism Generation for Limit Analysis of Frames, Journal of Structural Division, ASCE, Vol. 109, ST1, 1979, pg. 1-15.
- (16) - Thoft-Christensen, P., and Murotsu, Y., Application of Structural Systems Reliability Theory, Springer-Verlag, Berlin, 1986.
- (17) - Ditlevsen, O., Narrow Reliability Bounds for Structural Systems, Journal of Structural Mechanics, Vol. 7, 1979, pg. 435-451.
- (18) - Schmit, L. A., and Fox, R. L. Fox, An Integrated Approach to Structural Synthesis and Analysis, AIAA Journal, Vol. 3, No. 6, 1960, pg. 1104-1112.
- (19) - Grierson, D. E., and Schmit, L. A., Synthesis under Service and Ultimate Performance Constraints, Computers and Structures, Vol. 15, No. 4, 1982, pg. 405-417.
- (20) - Haftka, R. T., Simultaneous Analysis and Design, AIAA Journal, Vol. 23, No. 7, 1985, pg. 1099-1103.
- (21) - Hughes, T. J. R., Winger, J., Levit, I., and Tezduar, T. E., New Alternating Direction Procedures in Finite Element Analysis Based on EBE Approximate Factorization, Computer Method for Nonlinear Solids and Structural Mechanics, AMD, Vol. 54, 1983, pg. 75-109.
- (22) - Burns, S. A., Simultaneous Design and Analysis Using Geometric Programming and the Integrated Formulation, Proceedings of the Swanson Analysis Systems, Newport Beach, California, 1987.
- (23) - Marc I. Hoit, Alfredo V. Soeiro and Fernando E. Fagundo, Integrated Structural Sizing Optimization, Engineering Optimization, Vol. 12, No.3, 1987, pg. 207-218..
- (24) - Soeiro, A., and Hoit, M., Sizing Optimization, Proceedings of the ASCE Structural Conference, Orlando, Florida, 1987.
- (25) - Hoit, M., and Soeiro, A., Integrated Structural Optimization, Proceedings of the International Conference on Computational Engineering Science, Atlanta, 1988.
- (26) - Soeiro, A., Integrated Analysis and Optimal Design, Thesis for the degree of Master of Engineering, University of Florida, Gainesville, Florida, 1986.
- (27) - Avriel, M., Nonlinear Programming: Analysis and Methods, Prentice-Hall, Englewood Cliffs, New Jersey, 1976.
- (28) - Park, R., and Paulay, T., Reinforced Concrete Structures, John Wiley and Sons, New York, 1975.

- (29) - Ali, M. M., and Grierson, D., Nonlinear Design of Reinforced Concrete Frameworks, Journal of Structural Engineering, ASCE, Vol. 112, No.10, 1986, pg. 2216-2233.
- (30) - Darvall, P. L., and Mendis, P. A., Elastic-Plastic Softening Analysis of Plane Frames, Journal of Structural Engineering, ASCE, Vol. 111, No. 4, 1985, pg. 871-887.
- (31) - Chajes, A. and, Churchill, J. E., Nonlinear Frame Analysis by Finite Element Methods, Journal of Structural Engineering, ASCE, Vol. 113, No. 6, 1987, pg. 1221-1235.
- (32) - Kayal, S., Finite Element Analysis of RC Frames, Journal of Structural Engineering, ASCE, Vol. 110, No. 12, 1984., pg. 2891-2907.
- (33) - Otani, S., Inelastic Analysis of R/C Frame Structures, Journal of Structural Division, ASCE, Vol. 100, N. ST7, 1974, pg. 1433-1457.
- (34) - Tsay, J. J., and Arora, J. S., Variational Methods for Design Sensitivity Analysis of Nonlinear Response with History Dependent Effects, Proceedings of the International Conference on Computational Engineering Science, Atlanta, 1988.
- (35) - Umerura, H., and Takizawa, H., Dynamic Response of Reinforced Concrete Buildings, IABSE Structural Engineering Documents, Number 2, Zurich, 1982.
- (36) - Kanaan, A. E., and Powell, G. H., DRAIN-2D, A General Purpose Computer Program for Dynamic Analysis of Inelastic Plane Structures, Report n^o EERC 73-6 and EERC 73-22, University of Berkeley, Berkeley, 1975.
- (37) - Breyse, D., and Mazars, J., Simplified Approach of Nonlinearity in R/C Beams, Journal of Structural Engineering, ASCE, Vol. 114, n. 2, 1988, pg. 251-268.
- (38) - Gedling, J. S., Mistry N. S., and Welch, A. K., Evaluation of Material Models for Reinforced Concrete Structures, Computers and Structures, Vol. 24, n. 2, 1986, pg. 225-232, 1986.
- (39) - Cauvin, A., Nonlinear Elastic Design and Optimization of Reinforced Concrete Frames, CSCE-ASCE-ACI-CEB International Symposium, Ontario, Canada, 1979.
- (40) - Charney, F. A., Correlation of the Analytical and Experimental Seismic Response of a 1/5th-Scale Seven-Story Reinforced Concrete Frame-Wall Structure, PhD Dissertation, University of California, Berkeley, 1986.
- (41) - Augusti, G., Baratta, A., and Casciati, F., Probabilistic Methods in Structural Engineering, Chapman and Hall, New York, 1984.

(42) - Madsen, H. O., Krenk, S., and Lind, N. C., Methods of Structural Safety, Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

(43) - Regan, P. E., and Yu, P. E., Limit State Design of Structural Concrete, John Wiley & Sons, New York, 1973.

(44) - Leporati, E., The Assessment of Structural Safety, Research Studies Press, Forest Grove, Oregon, 1977.

(45) - Pahoheimo, E. and Hannus, M., Structural Design Based on Weighted Fractiles, Journal of Structural Division, ASCE, Vol. 100, ST7, 1974, pg. 1367-1378.

(46) - Regulamento de Estruturas de Betão Armado e Pré-Esforçado, Imprensa Nacional Casa da Moeda, Lisboa, Portugal, 1983.

(47) - Thoft-Christensen, P., and Baker, M. J., Structural Reliability Theory and its Applications, Springer-Verlag, Heidelberg, West Germany, 1982.

(48) - Neville, A. M., Properties of Concrete, Pitman, Bath, United Kingdom, 1983.

(49) - Recommended Practice for Evaluation of Strength Test Results for Concrete, ACI 214-71, American Concrete Institute, Detroit, 1976.

(50) - Mindess, S. and Young, J. F., Concrete, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

(51) - Building Code Requirements for Reinforced Concrete, (ACI 318-83), American Concrete Institute, Detroit, 1976.

(52) - Hart, G., Uncertainty Analysis, Loads and Safety in Structural Engineering, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

(53) - Chou, Karen C., and Corotis, Ross B., Nonlinear Response to Sustained Load Processes, Journal of Structural Engineering, ASCE, Vol. 111, No. 1, 1985, pg. 142-155.

(54) - Probabilistic Mechanics and Structural Reliability, 4th ASCE Specialty Conference, ASCE, New York, 1984.

(55) - Blockley, D. I., The Nature of Structural Design and Safety, John Wiley and Sons, Chichester, United Kingdom, 1980.

(56) - Feng, Y. S., and Moses, F., A Method of Structural Optimization Based on Structural System Reliability, Journal of Structural Mechanics, Vol. 14, No. 4, 1986, pg. 437-453.

(57) - Frangopol, D. M., Sensitivity of Reliability-Based Optimum Design, Journal of Structural Engineering, ASCE, Vol. 111, No. 8, 1985, pg. 1703-1721.

(58) - Moses, F., Structural System Reliability and Optimization, Computers and Structures, Vol. 7, 1977, pg. 283-290.

(59) - Sexsmith, R., and Mau, S.-T., Reliability Design with Expected Cost Optimization, Safety and Reliability of Metal Structures, ASCE, New York, 1977, pg. 427-443.

(60) - Yao, J. T. P., Safety and Reliability of Existing Structures, Pitman Publishing, London, 1985.

(61) - Frangopol, D. M., Structural Optimization under Conditions of Uncertainty, with Reference to Serviceability and Ultimate Limit States, Structural Optimization Developments, ASCE, New York, 1986, pg. 54-71.

(62) - Rashedi, R., and Moses, F., Identification of Failure Modes in System Reliability, Journal of Structural Division, ASCE, Vol. 114, No. 2, 1988, pg. 292-313.

(63) - Bennett, R. M., and Ang, A. H-S., Investigation of Methods for Structural System Reliability, Structural Research Series No. 510, University of Illinois, Urbana, Illinois, 1983.

(64) - Spencer, B. F., and Elishakoff, I., Reliability of Uncertain Linear and Nonlinear Systems, Journal of Engineering Mechanics, ASCE, Vol. 114, No. 1, 1988, pg. 135-148.

(65) - Yuansheng, F., and Moses, F., Optimum Design, Redundancy and Reliability of Structural Systems, Computers and Structures, Vol. 24, No. 24, 1986, pg. 239-251.

(66) - Kam, T., Corotis, R. B., and Rossow, E. C., Reliability of Nonlinear Framed Structures, Journal of Structural Engineering, ASCE, Vol. 109, No. 7, 1983, pg. 1585-1601.

(67) - Austin, M. A., Pister, K. S., and Mahin, S. A., Probabilistic Limit States Design of Moment Resistant Frames under Seismic Loading, Structural Optimization Developments, ASCE, New York, 1986, pg. 1-16.

(68) - Ma, H-F., and Ang, A. H-S., Reliability Analysis of Redundant Ductile Structural Systems, Structural Research Series No. 494, University of Illinois, Urbana, Illinois, 1981.

(69) - Advances in Structural Reliability, Proceedings of the Advanced Seminar on Structural Reliability, D. Reidel Publishing Company, Dordrecht, West Germany, 1987.

(70) - Krajcinovic, D., Limit Analysis of Structures, Journal of Structural Division, ASCE, Vol. 95, ST9, 1969, pg. 1901-1909.

(71) - Building Construction Cost Data, Robert Snow Means Company, Duxbury, Massachusetts, 1985.

(72) - Schuldt, S. B., A Method of Multipliers for Mathematical Programming Problems with Equality and Inequality Constraints, Journal of Optimization Theory and Applications, V. 17, 1975, pg. 155-161.

(73) - Abadie, J. and Carpentier, J., Generalization of the Wolfe Reduced Gradient Method to the Case of Nonlinear Constraints, Optimization, Academic Press, London, 1969.

(74) - Smeers, Y., Generalized Reduced Gradient Method as an Extension of Feasible Direction Methods, Journal of Optimization Theory and Applications, Vol. 22, 1977, pg. 209-226.

(75) - Gabriele, G. A., and Ragsdell, K. M., The Generalized Reduced Gradient Method: A Reliable Tool for Optimal Design, ASME Journal of Engineering, Vol. 99, 1977, pg. 384-400.

(76) - Lasdon, L. S., Waren, A. D., Jain, A., and Ratner, M., Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming, ACM Transactions of Mathematical Software, Vol. 4, 1978, pg. 34-50.

(77) - Hoit, M. I., SSTAN-Simple Structural Analysis Program, University of Florida, Gainesville, Florida, 1987.

(78) - Cohn, M. Z., Analysis and Design of Inelastic Structures, Vol.2: Problems, University of Waterloo Press, Ontario, Canada, 1972.

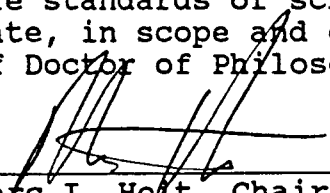
BIOGRAPHICAL SKETCH

The author was born in Porto, Portugal in 1954. He finished high school in D. Manuel II, Porto in 1971, graduated from the University of Porto in Civil Engineering, majoring in Structures and obtained the degree of Master in Engineering from the University of Florida, United States of America in 1986.

The writer taught introductory courses in the University of Porto, College of Engineering, between 1976 and 1985, and is on leave to pursue his Ph.D. degree. He was a structural consultant between 1977 and 1984.

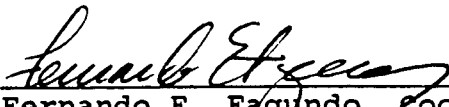
He is a recipient of a scholarship from the binational Fulbright Commission during his studies in the United States of America. The author received a grant from Fundação Oriente to present a paper in the IV World Conference on Continuing Engineering Education. He is a member of Phi Beta Delta, ASCE and ACI.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



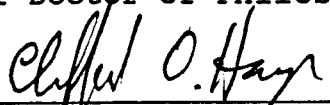
Marc I. Hoit, Chair
Assistant Professor of Civil
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



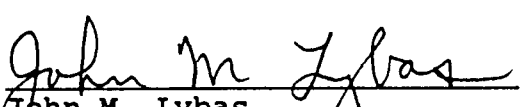
Fernando E. Fagundo, Cochair
Associate Professor of Civil
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



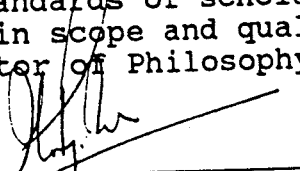
Clifford O. Hays
Professor of Civil Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



John M. Lybas
Associate Professor of Civil
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Prabhat Hajela
Associate Professor of
Aerospace Engineering,
Mechanics, and Engineering
Science

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August, 1989

Dean, College of Engineering

Dean, Graduate School



FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

BIBLIOTECA



000005912