

COMPUTER-AIDED DESIGN OF A  
TWO-SPAN CANTILEVER HIGHWAY BRIDGE

Thesis for the Degree of Ph. D.  
MICHIGAN STATE UNIVERSITY  
KULKARNI SUDHAKAR R.  
1973

LIBRARY  
Michigan State  
University

This is to certify that the  
thesis entitled  
COMPUTER-AIDED DESIGN OF A  
TWO-SPAN CANTILEVER HIGHWAY BRIDGE

presented by  
KULKARNI SUDHAKAR R.

has been accepted towards fulfillment  
of the requirements for

Ph.D. degree in Civil Engineering

*Robert K. Wen*

Major professor

Date October 17, 1973

SEP 18 2006  
03 22 00

APR 05 2006  
03 22 06

SEP 18 2006  
04 21 06

## ABSTRACT

### COMPUTER-AIDED DESIGN OF A TWO-SPAN CANTILEVER HIGHWAY BRIDGE

By

Kulkarni Sudhakar R.

Computer-aided designs of two-span cantilever highway bridges are investigated. The designs are governed by the current AASHO Specifications. Three methods: linear programming, grid search, and the method of bounds are studied to obtain designs that are nearly optimal (minimum cost). The linear programming approach was formulated by a linearization of the nonlinear cost or objective function and of the nonlinear constraints. As some of the design variables are discrete, the algorithm of Land and Doig for mixed linear programming problems was used. The grid search method involved a search in the design space of the four independent design variables: the number of girders, the length of cantilever, the depth of web plate, and the width of flange plate. The method of bounds differs from the grid search method in regard to the termination criteria. While the former search terminates upon locating a (local) minimum, the latter ends when the latest feasible design reaches within a prescribed neighborhood on an "effective lower bound."



Three bridges that had been designed by the traditional method and built in Michigan were studied. The costs of the computer-aided designs were found to be 4% less than the cost of the designs obtained by the traditional method.

Of the three optimization methods considered the linear programming approach seems least efficient. The method of grid search offers a straight forward procedure for an automated design. The method of bounds would further reduce the computational work without appreciable loss of accuracy. A complete design by this method can be obtained by a single run of the computer program prepared.

COMPUTER-AIDED DESIGN OF  
A TWO-SPAN CANTILEVER HIGHWAY BRIDGE

By

Kulkarni, Sudhakar R.

A THESIS

Submitted to  
Michigan State University  
in partial fulfillment of the requirements  
for the degree of

DOCTOR OF PHILOSOPHY

Department of Civil Engineering

1973

686357

## ACKNOWLEDGMENTS

The author is deeply indebted to his major academic advisor, Dr. Robert K. L. Wen, Professor of Civil Engineering, for his continuous guidance and assistance during the course of this study. The author also wishes to thank the other members of his guidance committee, Dr. Charles E. Cutts, Dr. William A. Bradley, and Dr. J. S. Frame.

Sincere appreciation is extended to the Michigan Department of the State Highways for granting the permission to the author to pursue the graduate study while employed by the department, and to use the computer facility of the department for this investigation. The author wishes to thank his supervisors Mr. O. E. Mace, and Mr. R. G. Montgomery of the Design Division, and Mr. VanAuken, and Mr. Kellerman of the Computer Co-ordinating Section of the Design Division for their assistance.

Thanks are due to the relatives and the friends who helped the author to pursue the graduate study abroad.

## LIST OF CONTENTS

|                                                                                       | Page |
|---------------------------------------------------------------------------------------|------|
| ACKNOWLEDGMENTS . . . . .                                                             | ii   |
| LIST OF FIGURES . . . . .                                                             | v    |
| LIST OF TABLES . . . . .                                                              | vi   |
| CHAPTER I: INTRODUCTION . . . . .                                                     | 1    |
| 1.1 Objective . . . . .                                                               | 1    |
| 1.2 Scope . . . . .                                                                   | 2    |
| 1.3 Statement of the Design Problem . . . . .                                         | 6    |
| 1.4 Notation . . . . .                                                                | 14   |
| CHAPTER II: DESIGN BY LINEAR PROGRAMMING . . . . .                                    | 16   |
| 2.1 Application of Linear Programming<br>to A Nonlinear Programming Problem . . . . . | 16   |
| 2.1.1 Linearization . . . . .                                                         | 17   |
| 2.1.2 A Linear Programming Problem . . . . .                                          | 18   |
| 2.2 Application to Design . . . . .                                                   | 20   |
| 2.2.1 Statement of the Linear Programming<br>Problem . . . . .                        | 20   |
| 2.3 Iterative Procedure . . . . .                                                     | 24   |
| 2.4 Implementation and Examples . . . . .                                             | 27   |
| 2.4.1 Example 1 . . . . .                                                             | 27   |
| 2.4.2 Example 2 . . . . .                                                             | 28   |
| 2.4.3 Example 3 . . . . .                                                             | 28   |
| 2.4.4 Discussion . . . . .                                                            | 28   |
| CHAPTER III: DESIGN BY THE GRID SEARCH METHOD . . . . .                               | 29   |
| 3.1 Introduction . . . . .                                                            | 29   |
| 3.2 Problem Solution . . . . .                                                        | 30   |
| 3.2.1 Design Space and The Size of Grid . . . . .                                     | 31   |
| 3.2.2 Details of the Procedure . . . . .                                              | 32   |
| 3.3 Practical Applications . . . . .                                                  | 33   |
| 3.3.1 Example 1 . . . . .                                                             | 33   |
| 3.3.2 Example 2 . . . . .                                                             | 34   |
| 3.3.3 Example 3 . . . . .                                                             | 34   |
| CHAPTER IV: DESIGN BY THE METHOD OF BOUNDS . . . . .                                  | 36   |
| 4.1 Introduction . . . . .                                                            | 36   |
| 4.2 The Method of Bounds . . . . .                                                    | 37   |

|                                                                             | Page |
|-----------------------------------------------------------------------------|------|
| 4.3 Practical Applications . . . . .                                        | 38   |
| 4.3.1 Example 1 . . . . .                                                   | 39   |
| 4.3.2 Example 2 . . . . .                                                   | 39   |
| 4.3.3 Example 3 . . . . .                                                   | 39   |
| CHAPTER V: COMPARISON OF THE METHODS OF DESIGN<br>AND CONCLUSIONS . . . . . | 41   |
| 5.1 General . . . . .                                                       | 41   |
| 5.2 Comparison of the Results . . . . .                                     | 42   |
| 5.3 Conclusions . . . . .                                                   | 43   |
| LIST OF REFERENCES . . . . .                                                | 70   |
| APPENDIX I: USER'S MANUAL . . . . .                                         | 71   |
| APPENDIX II: COMPUTER PROGRAM . . . . .                                     | 84   |

## LIST OF FIGURES

| Figure                                                            | Page |
|-------------------------------------------------------------------|------|
| 1-1 Photographs of A Two-Span Cantilever Highway Bridge . . . . . | 45   |
| 1-2 Two-Span Cantilever Bridge Details . . . . .                  | 47   |
| 1-3 Two-Span Cantilever Girder Details . . . . .                  | 48   |
| 1-4 Standard HS 20 Truck Loading . . . . .                        | 50   |
| 2-1 Flow Chart of the Iterative Method . . . . .                  | 51   |
| 3-1 The Grid Search Method . . . . .                              | 52   |
| 3-2 Flow Chart of the Grid Search Method . . . . .                | 53   |
| 4-1 The Method of Bounds . . . . .                                | 55   |

## LIST OF TABLES

| Table                                                                                 | Page |
|---------------------------------------------------------------------------------------|------|
| 2-1 Data for Design of the Bridges . . . . .                                          | 56   |
| 2-2 Results of Cost-Optimization of Example 1 . . . . .<br>(using Linear Programming) | 57   |
| 2-3 Results of Cost-Optimization of Example 2 . . . . .<br>(using Linear Programming) | 59   |
| 2-4 Results of Cost-Optimization of Example 3 . . . . .<br>(using Linear Programming) | 61   |
| 3-1 Summary of the Results of Example 1 . . . . .<br>(using the Grid Search Method)   | 62   |
| 3-2 Summary of the Results of Example 2 . . . . .<br>(using the Grid Search Method)   | 63   |
| 3-3 Summary of the Results of Example 3 . . . . .<br>(using the Grid Search Method)   | 64   |
| 4-1 Summary of the Results of Example 1 . . . . .<br>(using the Method of Bounds)     | 65   |
| 4-2 Summary of the Results of Example 2 . . . . .<br>(using the Method of Bounds)     | 66   |
| 4-3 Summary of the Results of Example 3 . . . . .<br>(using the Method of Bounds)     | 67   |
| 5-1 Comparison of the Results . . . . .                                               | 68   |
| 5-2 Comparison of Computer Time . . . . .                                             | 69   |

## CHAPTER I

### INTRODUCTION

#### 1.1. Objective

Minimum-cost design of structures is an important problem in the field of structural engineering. It is particularly significant for structures that are constructed frequently. For these structures a modest reduction in the cost of one structure would mean a substantial saving in expenditure when a number of such structures are considered together. Two-span cantilever bridges (see photographs in Figure 1.1) fall into this category as they are frequently used for grade separation in highway systems.

The objective of the present study is to develop a computer program that can be used to produce complete designs of two-span cantilever bridges and such designs should be optimal or, at least nearly optimal, and, of course, satisfy the requirements of the current Standard Specifications for Highway Bridges adopted by the American Association of State Highway Officials (abbreviated as AASHO).

In recent years, since circa 1960, many researchers have attempted to develop various structural optimization methods. While a great many fundamental works have been published, most have been concerned with methodology and principles.



Applications to day-to-day structural designs as practiced by the engineering profession have been rather limited.

The author has been working as a highway bridge engineer with the Michigan Department of State Highways for the last six years. The selection of this particular problem is made with a view to increasing the effectiveness of highway bridge engineering as well as to providing an example of bridging the gap between engineering theory and application.

### 1.2. Scope

The "traditional method" to obtain a design of a highway bridge is largely based upon the engineer's intuition and certain empirical rules developed in the design office. Usually, after the values of certain key design variables are assumed the design of the structural components of the bridge such as the reinforced concrete slab and the welded steel plate girders proceeds by using design tables, slide rule methods, or by using computer programs.

The necessary input information for the design of a two-span cantilever bridge usually consists of the geometrical data, design loads, properties of construction materials, and cost (material and labor) data. A design of the bridge may be carried out after choices are made for the number of girders, length of the cantilever, depth of web plate, and width of the flange. These variables are defined as the "independent design variables." With the independent design variables known all other variables such as the slab thickness, thickness of the

flange plates, defined as "dependent design variables," may be obtained by the conventional elastic design method.

From a mathematical point of view, the problem of optimal design may be defined as one of minimizing the total material and labor cost of the structure--a function of the dependent and independent design variables--subject to certain constraints composed mainly of the specifications. (Specific design requirements are described in Section 1.3.) The cost function and most of the constraints are nonlinear. Thus, the minimum-cost design of the bridge is a nonlinear programming problem. An additional complication lies in the often required condition that the variables can take on only certain discrete values. An exact solution of the nonlinear programming problem is very difficult, and only approximate solutions of the problem can be attempted. In this thesis, three methods for optimization will be considered. They are the "linear programming method," the "grid search method," and the "method of bounds."

The use of linear programming is one approach to obtain approximate solutions of nonlinear programming problems. This approach has been used to solve structural optimization problems in References (2)\*, and (8).

The nonlinear programming problem may be solved approximately as a sequence of linear programming problems by linearization of the nonlinear problem at certain expansion

---

\* number in parentheses refer to entries in the list of references.

points. Linearization can be obtained as the linear terms of the Taylor series expansion of the nonlinear objective function and/or nonlinear constraints of the problem. Some of the variables of the problems such as the number of girders and the depth of the web plate are required to take on discrete values. Solutions of such "mixed" linear programming problems in which some variables are required to take on discrete values, may be obtained by the Land and Doig algorithm described in Reference (7). An iterative procedure solving a sequence of mixed linear programming problems can be constructed to obtain the values of the design variables which would minimize the nonlinear objective function.

Various direct search methods are well suited to the solution of a structural optimization problem in which the discrete design variables are few in number and the ranges of the values of such design variables are small. A direct search method uses simple strategy easily adaptable for computer solutions. A structural optimization problem using a direct search method--grid search--has been considered in Reference (4).

The minimum-cost design of the bridge can be formulated as a four dimensional minimization problem, by considering the cost of the bridge as a function of the independent design variables. A solution of the problem can be obtained by successive applications of a two-dimensional grid search which examines the discrete design space to determine the values of the independent design variables which would minimize the cost of the design.

The "method of bounds" is suggested in Reference (5). The basic idea of the method of bounds as adopted herein is that instead of finding the optimal design, a near optimal design is determined by terminating the search when the cost of a current design is within a specified tolerance of an "effective lower bound" (to be defined later). The procedure generally reduces the computational work provided that the values of tolerance and the effective lower bound are properly specified.

As it will be shown in later chapters, near optimal designs of the three bridges have been obtained by using the above described methods. These bridges designed according to the traditional method had actually been built in the State of Michigan. Comparisons of the designs by the various approaches indicate that (a) computer-aided designs are more economical than the traditional designs, and (b) the method of bounds using a direct search method together with certain judiciously chosen criteria to terminate the search provides an efficient method of automated design.

The designs by using linear programming, the grid search method and the method of bounds are presented in Chapters II, III, and IV, respectively. Comparisons of these approaches and conclusions based upon numerical results and computer times are presented in Chapter V. User's Manual for a computer program named "BRIDGE" which has been prepared to implement the method of bounds is given in Appendix I, and the program is presented in Appendix II.

### 1.3 Statement of the Design Problem

The type of bridge under consideration here consists of steel plate girders with a reinforced concrete slab. Two photographs of a typical bridge are shown in Figure 1-1, and the usual engineering details of the bridge are shown in Figure 1-2, and those of a single girder in Figure 1-3.

The information necessary for design such as the geometrical data, loading, material properties, and cost data is shown in Table 2-1. The details of live load are shown in Figure 1-4.

The expression of the total cost of the bridge which is to be minimized, is a function of the design variables, and is written as follows.

$$\begin{aligned}
 Z = & C_1 B_L B_W T_S + C_2 B_L B_W A_S \\
 & + 0.5 C_3 N (1 - K_1) L_1 (t_{f1} + t_{f3} + t_{f6} + t_{f8}) b_f \\
 & + C_3 N K_1 L_1 (t_{f2} + t_{f7}) b_f + C_3 N L_c (t_{f3} + t_{f8}) b_f \\
 & + C_3 N (1 - K_2) (L_B - L_c) (t_{f4} + t_{f9}) b_f \\
 & + C_3 N K_2 (L_B - L_c) (t_{f5} + t_{f10}) b_f \\
 & + C_3 N (L_1 + L_c) h t_{w1} + C_3 N (L_B - L_c) h t_{w2} \quad (1-1)
 \end{aligned}$$

where  $Z$  = total cost of the bridge (\$),

$A_S$  = cross-sectional area of steel reinforcement (in.<sup>2</sup>/ft.),

$B_L$  = length of bridge (ft.),

$B_W$  = width of bridge (ft.)

$b_f$  = width of flange plate (in.),

- $C_1$  = cost co-efficient derived from unit cost of concrete in \$/cu. yd.,  
 $C_2$  = cost co-efficient derived from unit cost of steel reinforcement in \$/lb.,  
 $C_3$  = cost co-efficient derived from unit cost of structural steel in \$/lb.,  
 $h$  = depth of web plate (in.),  
 $K_1$  = cut-off ratio in anchor span,  
 $K_2$  = cut-off ratio in suspended span,  
 $L_1$  = length of anchor span (ft.),  
 $L_B$  = length of span (ft.),  
 $L_C$  = length of cantilever (ft.),  
 $N$  = number of girders,  
 $T_S$  = thickness of slab (in.),  
 $t_{fi}$  = thickness of flange plates (in.),  
 $i = 1, 2, \dots, 10$ ,  
 $t_{w1}$  = thickness of web plate in anchor span (in.),  
 $t_{w2}$  = thickness of web plate in suspended span (in.).

The design of the bridge is controlled by the current Standard Specifications for Highway Bridges adopted by the American Association of State Highway Officials. The requirements of the specifications which are applicable to the design are as follows:

The reinforced concrete slab is designed according to the conventional elastic design procedure which is described in most text books on reinforced concrete design. The design



of slab is such that the concrete and reinforcing steel are stressed to their maximum allowable limits at the same time. The main reinforcement is perpendicular to the direction of traffic. Practical considerations would set the minimum thickness of the slab and the minimum reinforcement to be 7.5 inches, and  $0.66 \text{ in.}^2/\text{ft.}$ , respectively. The mathematical expression for the thickness of slab is given by the following equation

$$T_S = \sqrt{M/K} + c \quad (1-2)$$

where  $T_S$  = thickness of slab (in.),

$c$  = cover for main reinforcement (in.),

$M$  = maximum total external moment (ft.-lb. per ft. width of slab) which is the sum of the moment due to live load with impact and moment due to dead load of slab and future wearing surface (25 lbs. per sq. ft. of slab). The moment due to live load is determined according to Section 1.3.2 of the Specifications which states the following. The live load moment for simple spans shall be determined by the following formula (impact not included):

$$M_{L.L.} = \left( \frac{S + 2}{32} \right) P_{20} \quad (1-3)$$

in which  $M_{L.L.}$  = live load moment (ft.-lb.),

$S$  = girder spacing minus  $1/2$  the width of flange plate (ft.),

$P_{20}$  = load due to HS 20 truck loading (lbs.).



In slab continuous over three or more supports, a continuity factor of 0.8 shall be applied to the above formula (1-3) for both positive and negative moment. The value of impact is to be 0.30 (max.).

The positive and negative moments due to dead load are determined by the following formulas:

$$M_{D.L.}^{+} = (1/14) w S_g^2 \quad (1-4)$$

$$M_{D.L.}^{-} = (1/10) w S_g^2 \quad (1-5)$$

in which  $M_{D.L.}^{+}$  = positive moment (ft.-lbs.),

$M_{D.L.}^{-}$  = negative moment (ft.-lbs), and

$w$  = uniform dead load (lbs. per sq. ft. of slab).

Refer to Section 904 of the ACI Building code (ACI 318-63) for explanation of formulas (1-4) and (1-5).

$K$  = design constant determined from the elastic design as follows.

$$K = (1/2) f_c j k \quad (1-6)$$

in which  $f_c$  = allowable compressive stress of concrete (psi.),

$$j = 1 - 1/3 k,$$

$$k = 1 / [1 + f_s / (n f_c)]$$

$f_s$  = allowable tensile stress of steel (psi.),

$n$  = ratio of modulus of elasticity of steel to that of concrete.

The cross-sectional area of steel reinforcement is determined from the following equation.

$$A_s = M \text{ (in. - lbs.)} / (f_s j d) \quad (1-7)$$

where  $d = T_s - c$

The actual bending stresses in the top and bottom flanges must be less than or equal to the allowable bending stress. This requirement is usually checked at the points of the maximum bending moments and at the cut-off points (i.e., points at which the thickness of the flange plates is changed.) of the flange plates (Section 1.7.1 of the Specifications) by the following:

$$f_b = \frac{M_g C}{I} \leq F_b \quad (1-8)$$

where  $f_b$  = actual bending stress (psi.),

$F_b$  = allowable bending stress (psi.),

$C$  = distance from neutral axis (in.),

$I$  = moment of inertia (in.<sup>4</sup>),

$M_g$  = maximum total external moment (in.-lbs.).

At the cut-off points the flange plates of the girder are connected by butt welds. The actual bending stress in the weld metal and the base metal adjacent to the butt welds must be less than the allowable fatigue stress. Refer to Section 1.7.3 of the Specifications.

$$f_b \leq f_r \quad (1-9)$$

where  $f_r$  = allowable fatigue stress (psi.).

The current design practice is to use steel plate girders without transverse intermediate stiffeners. The thickness of web plate, when transverse intermediate stiffeners are omitted, shall not be less than the thickness determined by the following formula: (Section 1.7.72 of the Specifications)

$$t_w = \frac{h\sqrt{f_v}}{7500} \quad (1-10)$$

where  $t_w$  = the thickness of web plate (in.),

$h$  = the depth of web plate (in.),

$f_v$  = the average calculated shear stress (psi.) in the gross section of the web plate at the point considered. Thus,

$$f_v = V/(ht_w)$$

$$V = V_{D.L.} + (V_{L.L.}) (S_g/W_L) \quad (1-11)$$

$V_{D.L.}$  = shear due to dead load (lbs./girder),

$V_{L.L.}$  = maximum shear due to HS 20 truck, or HS 20 lane loading (lbs./lane),

$W_L$  = width of lane loading (ft.).

In no case, the web plate thickness shall be less than  $h/150$ . Refer to Section 1.7.72 of the Specifications.

According to prevalent practice  $t_w$  should not be less than 0.375 inches. Thus, the following inequalities hold:

$$t_w \geq h/150 \quad (1-12)$$

$$t_w \geq 0.375 \quad (1-13)$$

For composite girders, the ratio of the depth of the steel girder alone to the length of the span shall not be less than 1/30. (Section 1.7.11 of the Specifications). According to prevalent practice the depth of girder shall not be less than 42 inches. Thus, the following inequalities hold:

$$h/L \geq 1/30 \quad (1-14)$$

$$h \geq 42 \quad (1-15)$$

where  $L = L_1$  or  $L_2$

The ratio of the compression flange width of the thickness shall not exceed the value determined by the following formula. (Section 1.7.70 of the Specifications).

$$\frac{b_f}{t_f} = \frac{3250}{\sqrt{f_b}} \quad (1-16)$$

but in no case, shall  $b_f/t_f$  exceed 24. (Section 1.7.70 of the Specifications), i.e.,

$$\frac{b_f}{t_f} \leq 24 \quad (1-17)$$

The preceding constraint (1-17) also applies to the tension flange. (1-18)

The width of the top and bottom flanges should be the same, and should be equal to or greater than 14 inches.

$$b_f \geq 14 \quad (1-19)$$

The ratio of the live-load-plus-impact deflection at mid span to the length of the span shall be less than 1/1,000, and the ratio of the live-load-plus-impact deflection at the end of cantilever to its length shall be less than 1/350. Thus, the following constraints apply:

$$\frac{\Delta}{L} \leq \frac{1}{1000} \quad (1-20)$$

$$\frac{\Delta_c}{L_c} \leq \frac{1}{350} \quad (1-21)$$

where  $\Delta$  = deflection at mid span, and

$\Delta_c$  = deflection at the end of cantilever.

The factor of safety against lateral buckling of the steel girders, during transportation and erection must be equal to, or greater than 1.25, i.e.,

$$F_S = f_{cr} / f_b \geq 1.25 \quad (1-22)$$

where  $F_S$  = factor of safety against lateral buckling, and  
 $f_{cr}$  = critical bending stress. (See Reference (1) for the calculation of  $f_{cr}$ ).

According to prevalent practice, the minimum length of the cantilever is to be 5 feet. Thus,

$$L_c \geq 5 \quad (1-23)$$

The girder spacing should be within the range of 6.5 feet to 12.0 feet. Thus,

$$6.5 \leq S_g \leq 12.0 \quad (1-24)$$

The above described constraints are based upon the current AASHO specifications and prevalent design practice. It is recognized that from time to time, the requirements would be changed to reflect new research and/or economic conditions. These changes, however, can be incorporated in the computer programs prepared for this study without major difficulty.

#### 1.4 Notation

A list of important symbols used in this thesis is given as follows:

- $A = [a_{ij}]$ , matrix derived from constraints;
- $A_S$  = cross-sectional area of steel reinforcement (in.<sup>2</sup>/ft.);
- $B = [b_i]$ , matrix derived from constraints;
- $B_L$  = length of bridge (ft.);
- $B_W$  = width of bridge (ft.);
- $b_f$  = width of flange plate (in.);
- $C = \{c_j\}$  matrix of cost co-efficients;
- $C_1$  = cost co-efficient derived from unit cost of concrete in \$/cu. yd.;
- $C_2$  = cost co-efficient derived from unit cost of steel reinforcement in \$/lb.;
- $C_3$  = cost co-efficient derived from unit cost of structural steel in \$/lb.;
- $h$  = depth of web plate (in.);
- $K_1$  = cut-off ratio in anchor span;

$K_2$  = cut-off ratio in suspended span;  
 $L_1$  = length of anchor span (ft.);  
 $L_2$  = length of suspended span (ft.);  
 $L_B$  = length of span (ft.);  
 $L_C$  = length of cantilever (ft.);  
 $N$  = number of girders;  
 $S_g$  = girder spacing (ft.);  
 $T_S$  = thickness of slab (in.);  
 $t_{fi}$  = thickness of flange plate (in.);  
 $i$  = 1, 2, . . . , 10;  
 $t_{w1}$  = thickness of web plate in anchor span (in.);  
 $t_{w2}$  = thickness of web plate in suspended span (in.);  
 $X$  =  $\{x_j\}$  vector of design variables;  
 $Z$  = nonlinear objective function;  
 $Z_f$  = effective lower bound of  $Z$ ;  
 $Z_L$  = linearized objective function;  
 $[ ]$  = square or rectangular matrix;  
 $\{ \}$  = column vector.

## CHAPTER II

### DESIGN BY LINEAR PROGRAMMING

#### 2.1 Application of Linear Programming to A Non-Linear Programming Problem

A cost-optimization problem in structural design can usually be cast in the form of a problem of nonlinear programming. The general mathematical statement of a nonlinear programming problem is as follows.

Find  $x_1, x_2, \dots, x_n$  so as to minimize

$$Z = f(x_1, x_2, \dots, x_n) \quad (2-1)$$

subject to the constraints:

$$\begin{aligned} g_1(x_1, x_2, \dots, x_n) &\leq b_1 \\ g_2(x_1, x_2, \dots, x_n) &\leq b_2 \\ &\vdots \\ &\vdots \\ &\vdots \\ g_m(x_1, x_2, \dots, x_n) &\leq b_m \end{aligned} \quad (2-2)$$

and

$$x_j \geq 0 \text{ for } j = 1, 2, \dots, n,$$

where  $f(x_1, x_2, \dots, x_n)$  and the  $g_i(x_1, x_2, \dots, x_n)$  are given functions of the  $n$  variables.



The above problem can be re-written as follows. Find a vector  $X = \{x_1, x_2, \dots, x_n\}$  so as to minimize:

$$Z = f(X)$$

subject to  $g_i(X) \leq b_i$  for  $i = 1, 2, \dots, m$ ,

and  $x_j \geq 0$  for  $j = 1, 2, \dots, n$ .

For the bridge design problem the  $x$ 's are the design variables such as the number of girders and the web depth,  $Z$  is the cost of the structure, and  $g_i(X)$  represents a constraint such as a requirement of the AASHO specifications. When  $f(X)$  and/or any of the  $g_i(X)$  are nonlinear, the problem is one of nonlinear programming. Generally, an exact solution of a nonlinear programming problem is very difficult, and only approximate solutions can be obtained. The use of the linear programming method is one of the methods that may be used to obtain an approximate solution of the nonlinear programming problem. This approach is described in the following sections.

### 2.1.1 Linearization

The first step of the approach involves a linearization of the nonlinear problem. The step consists of the approximation of the nonlinear function by the linear terms of the Taylor series expansions. The linear terms of the Taylor series of a nonlinear function in  $n$  variables expanded about the point  $X^0 = (x_1^0, x_2^0, \dots, x_n^0)$ , are as follows:

$$f(X) = f(X^0) + \frac{\partial f(X^0)}{\partial x_1} (x_1 - x_1^0) + \dots + \frac{\partial f(X^0)}{\partial x_n} (x_n - x_n^0) \quad (2-3)$$

where 
$$\frac{\partial f(X^0)}{\partial x_j} = \frac{\partial f(X)}{\partial x_j} \Big|_{X=X^0}$$

By a linearization of the objective function and the constraints the problem becomes one of linear programming.

### 2.1.2 A Linear Programming Problem

The mathematical statement of the general form of the linear programming problem is as follows.

Find  $x_1, x_2, \dots, x_n$  which minimize the linear objective function,

$$Z_L = c_1x_1 + c_2x_2 + \dots + c_nx_n \quad (2-4)$$

subject to the constraints:

$$a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \leq b_1$$

$$a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n \leq b_2$$

.

.

.

$$a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \leq b_m$$

with  $x_j > 0$ ,

in which  $a_{ij}$ ,  $b_i$ , and  $c_j$  are all known constants.

The above can be written in matrix notation as follows.

Find  $X = \{x_1, x_2, \dots, x_n\}$  so as to minimize

$$Z = CX \quad (2-5)$$

subject to the constraints

$$AX \leq B$$

$$X \geq 0$$

where

$$A = [a_{ij}], B = \{b_i\}$$

$$C = \{c_j\}, X = \{x_j\}$$

for  $i = 1, 2, \dots, m$ , and  $j = 1, 2, \dots, n$ .

In contrast to the case of non-linear programming problem, a solution of a linear programming problem can usually be obtained by a specific algorithm.

A linear programming problem, in which some variables are required to take on only discrete values, is defined as a "mixed" problem. A solution of a mixed problem may be obtained by using the Land and Doig algorithm described in Reference (7), if an approximate solution obtained by rounding off the continuous solution to satisfy the discreteness requirements is deemed too crude.

The Land and Doig algorithm can be briefly described as follows. First, a continuous solution of a mixed problem is obtained. Then, two additional solutions of the problem are obtained by forcing a discrete variable to take on successively, the discrete values on either side of its value from the continuous solution, while all the other constraints remain unchanged. The solution with the smaller value of the linear objective function is selected, and the corresponding discrete variable is held constant. In this manner, other discrete variables are forced to take on discrete values, and each time

a solution with the smaller value of the linear objective function is selected. Proceeding this way, the vector of design variables in which all the variables required to take on discrete values have done so is considered to be a solution of the mixed linear programming problem.

An iterative procedure involving the solution of a mixed linear programming problems can be constructed to obtain an approximate solution of the non-linear programming problem. This is illustrated in the following.

## 2.2 Application to Design.

The application of the above mentioned method is explained in the following.

2.2.1 Statement of the Linear Programming Problem. The linearized objective function  $Z_L$  to be minimized is obtained by a linearization of the expression of  $Z$  as given by Equation (1-1) by a direct application of equation (2-3), and is presented as follows.

$$\begin{aligned}
 Z_L = & C_1 B_L B_W T_S + C_2 B_L B_W A_S + 0.5 C_3 N^\circ (1 - K_1^\circ) L_1 b_{ftf1}^\circ \\
 & + C_3 N^\circ K_1 L_1 b_{ftf2}^\circ + C_3 N^\circ [0.5 (1 - K_1^\circ) L_1 + L_c^\circ] b_{ftf3}^\circ \\
 & + C_3 N^\circ (1 - K_2^\circ) (L_B - L_c^\circ) b_{ftf4}^\circ + C_3 N^\circ K_2^\circ (L_B - L_c^\circ) b_{ftf5}^\circ \\
 & + 0.5 C_3 N^\circ (1 - K_1^\circ) L_1 b_{ftf6}^\circ + C_3 N^\circ K_1^\circ L_1 b_{ftf7}^\circ \\
 & + C_3 N^\circ [0.5 (1 - K_1^\circ) L_1 + L_c^\circ] b_{ftf8}^\circ + C_3 N^\circ (1 - K_2^\circ) (L_B - L_c^\circ) b_{ftf9}^\circ \\
 & + C_3 N^\circ K_2^\circ (L_B - L_c^\circ) b_{ftf10}^\circ
 \end{aligned}$$

$$\begin{aligned}
& + C_3 N^{\circ} [-0.5 (t_{f1}^{\circ} + t_{f6}^{\circ} + t_{f3}^{\circ} + t_{f8}^{\circ}) + t_{f2}^{\circ} + t_{f7}^{\circ}] L_1 b_f^{\circ} K_1 \\
& + C_3 N^{\circ} [-L_B (t_{f4}^{\circ} + t_{f9}^{\circ}) + L_C^{\circ} (t_{f4}^{\circ} + t_{f9}^{\circ}) - L_C^{\circ} (t_{f5}^{\circ} + t_{f10}^{\circ}) \\
& \quad + L_B (t_{f5}^{\circ} + t_{f10}^{\circ})] b_f^{\circ} K_2 \\
& + C_3 N^{\circ} (L_1 + L_C^{\circ}) h^{\circ} t_{w1} + C_3 N^{\circ} (L_B - L_C^{\circ}) h^{\circ} t_{w2} \\
& + C_3 N^{\circ} [(L_1 + L_C^{\circ}) t_{w1}^{\circ} + (L_B - L_C^{\circ}) t_{w2}^{\circ}] h \\
& + C_3 N^{\circ} [t_{f3}^{\circ} + t_{f8}^{\circ} - (1 - K_2^{\circ}) (t_{f4}^{\circ} + t_{f9}^{\circ}) - K_2^{\circ} (t_{f5}^{\circ} + t_{f10}^{\circ}) \\
& \quad + h^{\circ} (t_{w1}^{\circ} - t_{w2}^{\circ})] L_C \\
& + C_3 N^{\circ} [0.5 (1 - K_1^{\circ}) L_1 (t_{f1}^{\circ} + t_{f3}^{\circ} + t_{f6}^{\circ} + t_{f8}^{\circ}) \\
& \quad + K_1^{\circ} L_1 (t_{f2}^{\circ} + t_{f7}^{\circ}) + L_C^{\circ} (t_{f3}^{\circ} + t_{f8}^{\circ}) \\
& \quad + (1 - K_2^{\circ}) (L_B - L_C^{\circ}) (t_{f4}^{\circ} + t_{f9}^{\circ}) \\
& \quad + K_2^{\circ} (L_B - L_C^{\circ}) (t_{f5}^{\circ} + t_{f10}^{\circ})] b_f^{\circ} \\
& + C_3 \{ [0.5 (1 - K_1^{\circ}) L_1 (t_{f1}^{\circ} + t_{f3}^{\circ} + t_{f6}^{\circ} + t_{f8}^{\circ}) \\
& \quad + K_1^{\circ} L_1 (t_{f2}^{\circ} + t_{f7}^{\circ}) + L_C^{\circ} (t_{f3}^{\circ} + t_{f8}^{\circ}) \\
& \quad + (1 - K_2^{\circ}) (L_B - L_C^{\circ}) (t_{f4}^{\circ} + t_{f9}^{\circ}) \\
& \quad + K_2^{\circ} (L_B - L_C^{\circ}) (t_{f5}^{\circ} + t_{f10}^{\circ})] b_f^{\circ} \\
& \quad + (L_1 + L_C^{\circ}) h^{\circ} t_{w1}^{\circ} + (L_B - L_C^{\circ}) h^{\circ} t_{w2}^{\circ} \} N \quad (2-6)
\end{aligned}$$

The superscript "o" denotes the known values of the design variables of the initial design for the first cycle, or, for subsequent cycles, the known values of the design variables of the preceding cycle.

Some of the constraints imposed by the specifications are nonlinear. A nonlinear constraint can be linearized as

follows. All the variables of a nonlinear constraint are transferred to the left side of the equality, or inequality representing the constraint. The linear terms of Taylor series expansion of that side, about the given expansion point, yields a linearized constraint.

Consider a non-linear constraint  $\phi_i(X) \leq b_i$ . The linear terms of the Taylor series expression of  $\phi(X)$  at a expansion point  $X^o$  is as follows.

$$\begin{aligned} \phi_i(X) = \phi_i(X^o) + \frac{\partial \phi_i}{\partial x_1}(X^o) (x_1 - x_1^o) + \dots \\ + \frac{\partial \phi_i}{\partial x_n}(X^o) (x_n - x_n^o) \leq b_i \end{aligned} \quad (2-7)$$

A rearrangement of the equation (2-7) yields

$$\begin{aligned} \frac{\partial \phi_i}{\partial x_1}(X^o) \cdot x_1 + \dots + \frac{\partial \phi_i}{\partial x_n}(X^o) \cdot x_n \leq b_i \\ + \frac{\partial \phi_i}{\partial x_1}(X^o) \cdot x_1^o + \dots + \frac{\partial \phi_i}{\partial x_n}(X^o) \cdot x_n^o \end{aligned} \quad (2-8)$$

For purposes of calculation the following approximation was used.

$$\frac{\partial \phi_i}{\partial x_j}(X^o) = \frac{\phi_i(X^o + \Delta x_j) - \phi_i(X^o)}{\Delta x_j} \quad (2-9)$$

The treatment of the nonlinear constraints will be illustrated by a consideration of the nonlinear constraint Equation (1-10):

$$t_w = \frac{h\sqrt{f_v}}{7500} \quad (2-10)$$

Substituting  $f_v = V/(h t_w)$ , in which  $V$  is the total shear, the above is written as follows.

$$\frac{t_w^3}{hV} = \frac{1}{(7500)^2} \quad (2-11)$$

An expression of  $V$  at any point along the girder can be written in terms of the design variables. For example, the expression of  $V$  at the simple support of the suspended span  $L_2$ , is written as follows.

$$V = V_{D.L.} + (V_{L.L.}) (S_g/W_L) \quad (2-12)$$

The expressions to determine  $V_{D.L.}$  and  $V_{L.L.}$  can be written as follows.

$$V_{D.L.} = \frac{w L_2}{2} \quad (2-13)$$

$$V_{L.L.} = I \left[ 32 + \frac{32(L_2 - 14)}{L_2} + \frac{8(L_2 - 28)}{L_2} \right]$$

$$\text{or } I \left[ 26 + 0.640 L_2/2 \right] \text{ in kips} \quad (2-14)$$

which ever is larger.

where  $I$  = impact factor  $\left[ 50/(L_2 + 125) \right]$ , or maximum 0.30 ,  
and  $w$  = the total dead load including the bridge railings  
plus concrete slab and girder.

The quantity  $w$  is a function of all the design variables.

Thus, the constraint (2-11) can be written as

$$\phi(X) = \frac{t_w^3}{hV} = b = \frac{1}{(7500)^2} \quad (2-15)$$

Since, as explained above, the quantity  $V$  can be calculated for a given set of design variables, so can the function  $\phi$ .

Therefore, by use of equation (2-9) the derivatives can be computed. The nonlinear constraint can be thus linearized by a direct application of equation (2-8).

The co-efficients of the design variables in the linearized as well as linear constraints are the elements  $a_{ij}$  of the matrix A in Equation 2-5. The constant terms on the right hand side of the equality or inequality are the elements  $b_i$  of the matrix B in Equation 2-5.

The nonlinear constraints which check fatigue, local buckling of compression flanges, live load deflections, and factor of safety against lateral buckling of the girder, are not considered in the formulation of the problem using linear programming. Usually, these constraints are inactive when the other constraints are satisfied, and the inclusion of these constraints greatly increases the complexity of the problem. It will be shown later that even without these constraints the linear programming approach is rather time consuming as compared to the other methods.

### 2.3 Iterative Procedure

In this section will be described a procedure for an approximate solution of the nonlinear programming problem. The procedure involves a number of cycles of iterations. Each cycle consists the solution of a series of linear programming problems. A cycle of iterative procedure consists of: (a) the selection of the expansion point in the space of design variables, (b) the solution of a mixed linear programming problem obtained by linearizing the nonlinear programming problem at the expansion point, and (c) a check of the convergence of the iterative procedure. The details of the procedure are described in the following steps,

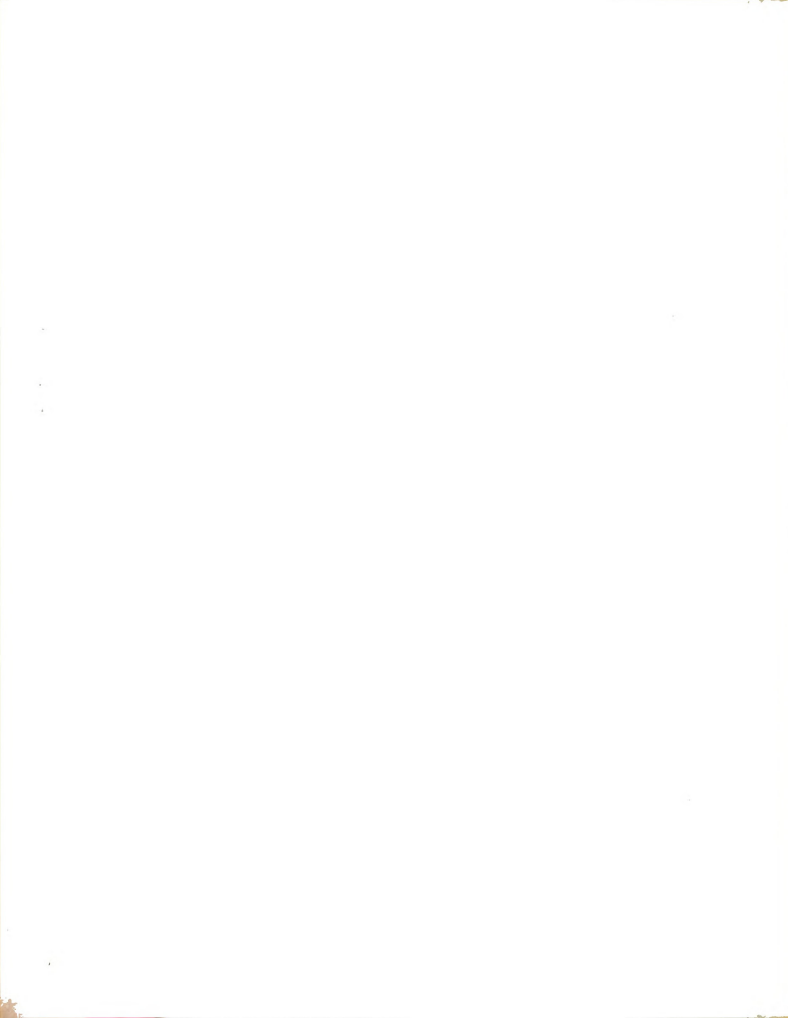


and a flow chart of the iterative procedure is shown in Figure 2-1.

1. For the first cycle, assume a set of feasible values for the independent design variables, and the dependent design variables are determined by using the conventional elastic design procedure. The vector of the design variables--dependent and independent--is taken as the initial expansion point, and the cost of the bridge (based upon the nonlinear function) is evaluated.

2. Obtain a linear programming problem by linearizing the nonlinear programming problem. This involves the determination of the elements of the matrices A, B, and C in Equation 2-5, as explained in Section 2.2.1.

3. Solve the mixed linear programming problem using the Land and Doig algorithm. Obtain a continuous solution of the linear programming problem. Denote the solution vector by  $X_p^q = X_1^1$ , in which q stands for the index of the cycle of iteration, and p the index of the linear programming problems for a given q (for a fixed expansion point  $X_0^q$ ). If value of N in  $X_1^1$  is not an integer, obtain two solutions of the linear programming problem by subjecting N to take on integer values of either side of its value in the continuous solution, while all other constraints remain unchanged. For example, if  $N = 5.36$ , obtain first solution with  $N = 5$ , and second  $N = 6$ , while all other constraints remain unchanged. These two vectors of design variables can be indicated by  $X_2^1$ , and  $X_3^1$ . Select the solution vector ( $X_2^1$ , or  $X_3^1$ ) with the smaller value of the linearized objective function, and hold



the corresponding value of  $N$  constant for the linear programming problems that are to follow. Similarly, discreteness requirements of the variables  $h$  and  $b_f$  are to be satisfied, successively, by using a solution vector with the smaller value of the linearized objective function  $(Z_L)_p^1$  where  $p = 4, 5, \dots$

The vector of the design variables with discrete values of  $N$ ,  $h$ , and  $b_f$ , and the smallest value of  $(Z_L)_p^q$ , is considered as a solution of the mixed linear programming program for  $q^{\text{th}}$  cycle of iteration, and it is denoted by  $\bar{X}_p^q$ . Compute the value of the nonlinear objective function  $Z_p^q$ , corresponding to the solution vector  $\bar{X}_p^q$ . Note that the values of  $(Z_L)_p^q$ , and  $Z_p^q$  corresponding to  $\bar{X}_p^q$  are different. (Numerical examples indicated that  $(Z_L)_p^q$  were greater than  $Z_p^q$ .)

4. Obtain the vector  $X_o^{q+1}$  by using the independent design variables from  $\bar{X}_p^q$ , and then determining the dependent design variables by using the conventional elastic design procedure. The vector  $X_o^{q+1}$  is taken as the expansion point for the  $(q+1)^{\text{th}}$  cycle of iteration, if it is necessary.

5. The iterative procedure is considered as "converged" if the following requirement is satisfied.

$$Z_o^q - Z_p^q \leq \epsilon \quad (2-16)$$

where  $Z_o^q$  is the value of the nonlinear objective function corresponding to  $X_o^q$ , and  $\epsilon$  is of the order of 1% of  $Z_o^q$ .

If the iterative procedure converges in the  $q^{\text{th}}$  cycle of iteration, the vector  $X_o^{q+1}$  is accepted as the design.



## 2.4 Implementation and Examples

A computer-aided design of the bridge using linear programming consists of the following steps: (a) Assume initial feasible values for the independent design variables, and determine the values of dependent design variables. A computer program named GIRDER was written for this purpose. (b) The initial vector of design variables is used to determine the elements of matrices A, B, and C. A computer program named MATRIX was written for this step. (c) The linear programming problem is solved by using the OPTIMA program which is available on the CDC 6500 Computer System at Michigan State University. Each step was completed separately. The entire procedure was semi-automatic, and did not prove to be quite efficient.

Three examples were considered. The data are given in Table 2-1. The initial designs of these bridges had been obtained by the conventional method, and they were actually built for use in the State of Michigan.

2.4.1 Example 1 The results of cost-optimization of this example are given in Table 2-2. The cost of initial design of the bridge, at the first cycle of iteration, is \$194,800. Two cycles of iteration requiring five solutions of linear programming problem are needed to determine a near minimum-cost design. The cost of the final design, \$184,900, is 5.1% less than the cost of the initial design.



2.4.2 Example 2 The results of cost-optimization of this example are given in Table 2-3. The cost of initial design of the bridge at the first cycle of iteration is \$173,500. Two cycles of iteration, requiring six solutions of linear programming problem are needed to determine a near minimum-cost design. The cost of the final design, \$161,200, is 7.1% less than the cost of the initial design.

2.4.3 Example 3 The results of cost-optimization of this example are given in Table 2-4. The cost of initial design of the bridge, at the first cycle of iteration, is \$46,440. One cycle of iteration, requiring three solutions of linear programming problem is needed to determine a near minimum-cost design. The cost of the final design, \$47,860, turns out to be greater than that of the initial design.

For this example, the iterative procedure converged to a design which costs more than the initial design. This is not entirely unexpected since linearization of a non-linear problem does introduce errors.

#### 2.4.4 Discussion

The linear programming method, as outlined in this chapter, seems to work for Example 1, and 2, but not for Example 3. Improvements in the solution of Example 3 could be attempted by selecting a different initial design point. However, such an approach is deemed time consuming and expensive. Therefore, alternative methods of optimization which would consider the nonlinear problem without linearization are investigated. These methods are described in the following chapters.

## CHAPTER III

### DESIGN BY THE GRID SEARCH METHOD

#### 3.1 Introduction

A successful application of linear programming to a minimum cost structural design problem must overcome the difficulties due to the nonlinearities of the problem and the discreteness of the design variables as illustrated in the preceding chapter. The procedure using linear programming as described earlier did appear somewhat unwieldy. As an alternative, a more direct search method suggests itself because the independent design variables are usually discrete. A straight forward examination of all the possibilities to determine the optimal design would involve  $\prod_{i=1}^n m_i$  designs, where  $m_i$  is equal to number of possible values that the  $i^{\text{th}}$  independent design variable can take on, and  $n$  is equal to number of independent design variables. Of course, it is not efficient to investigate all the possibilities in order to determine the optimal design. However, a suitable search method such as the grid search method could enable the engineer to reduce the number of possibilities to be investigated.

For the problem at hand, a two-dimensional grid search method is proposed to determine the values of the independent



design variables--  $N$ ,  $L_c$ ,  $h$  and  $b_f$  --which would minimize the total cost of the bridge. The feasible discrete values of  $N$  and  $L_c$  formulate the two-dimensional design space, and those of  $h$  and  $b_f$  formulate the other two-dimensional design space.

The first step in setting up the procedure is to establish the space of the search, i.e., the limits and increments of each of the independent variable. A typical cycle of the grid search consists of (a) the determination of the costs of designs corresponding to the, say,  $n$  points of the grid which is centered on a given point, (b) comparison of these costs to locate the best (the lowest cost) design, and (c) the use of the best design from the previous cycle as the grid center for the next cycle, unless it coincides with the center of the grid, in which case, the search method is terminated. The design corresponding to the lowest cost thus determined is taken to be the design desired. This approach implies the assumption that the design space is convex, and a local minimum is also the global minimum.

### 3.2. Problem Solution

For the application of the grid search method to obtain a near minimum-cost design of the bridge it is necessary to determine the limits of the design space, i.e., the lower and upper bounds of the feasible values of the design variables. The size of the grid should be such that the design space covers the practical possibilities of the design.

3.2.1 Design Space and the Size of Grid. The criteria used to determine the limits of the design space and grid size are as follows. The lower and upper bounds of the number of girders are such that resulting spacing between successive girders should be within the range of 6.5' to 12.0'. In any case, the number of girders shall not be less than three. The increment of  $N$  is equal to one. The lower bound and upper bound of length of cantilever  $L_c$  are  $(L_1/10-2)$  and  $(L_1/10+2)$ , respectively. The increment of  $L_c$  is set equal to one foot.

The lower bound of the depth of the web plate  $h$  is set by Section 1.7.11 of the Specifications which states that the ratio of depth of steel girder alone to length of span shall not be less than  $1/30$ . Prevailing practice usually considers 42" to be its minimum value. The upper bound of  $h$  is equal to its lower bound plus 24" unless it is determined by the grade of the roadway. The increment of  $h$  is taken to be 6". The lower and upper bound of the width of the flange plate  $b_f$  are 14" and 22", respectively. The increment of  $b_f$  is taken to be 2".

3.2.2 Details of the Procedure. The procedure to obtain a minimum-cost design of the bridge using the grid search method is depicted in Figure 3-2, and described in the following.

1. Refer to Figure 3-1. A starting point in the design space of  $N$  and  $L_c$  is selected with  $N$  corresponding to minimum girder spacing and  $L_c$  about one tenth of the length of anchor span  $L_1$ . A 3x3 nine point grid (referred to as Grid I) is

centered on this starting point, and the values of  $N$  and  $L_c$  corresponding to the other points of Grid I are determined.

2. For a given set of  $N$  and  $L_c$  at each of the nine points of Grid I, determine the values of  $h$  and  $b_f$  such that the total cost of the bridge corresponding to the point of Grid I is minimized. This is achieved by searching the design space of  $h$  and  $b_f$ . The following steps: 3, 4, and 5 describe the grid search in that design space.

3. A starting point in the design space of  $h$  and  $b_f$  is selected such that the values of both variables are minimum allowable. A 3x3 nine point grid (referred as Grid II; see Figure 3-1) is centered on this starting point. (It may be noted that if the center of the grid is on the boundary, the number of grid points will be less than nine.) The values of  $h$  and  $b_f$  corresponding to the points of Grid II are determined.

4. For each of the nine points of Grid II, the design of the bridge is completed by using the conventional elastic design procedure, and the cost of the bridge is determined. The costs of the nine designs are compared, and the design with the least cost is found.

5. If the least-cost point of Grid II is the center of the grid, the search in the design space of  $h$ , and  $b_f$  is terminated. Otherwise, the point corresponding to the least cost design is used as the center of Grid II for the next cycle, and the steps 3, 4, and 5 are repeated.

6. The cost of design corresponding to each of nine points of Grid I is determined. These costs are compared and the least cost design is located.

7. If the least cost point of Grid I is the center of the grid. The search in the design space of  $N$ , and  $L_c$  is terminated. Otherwise, the point corresponding to the least cost design is used as the center of Grid I for the next cycle, and the steps 1 through 7 are repeated.

To reduce computational work, the information about the designs corresponding to all grid points encountered in the search process is stored in computer memory.

### 3.3 Practical Applications

The designs of three bridges are considered to illustrate the application of the grid search method. The designs of these bridges have been also obtained by linear programming in Chapter II.

3.3.1 Example 1 The summary of the results of this example is given in Table 3-1. The lower bounds of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  are 5, 9', 48", and 14", respectively. The upper bounds of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  are 7, 13', 54", and 22", respectively. The number of designs required to be investigated to determine the optimal design by exhaustive search is 150.

For the first cycle,  $N = 7$ ,  $L_c = 11$ , and  $h = 48$ ,  $b_f = 14$  are the co-ordinates of the center of Grid I and Grid II, respectively. The number of designs that have been considered when the minimum-cost design by the grid search method is obtained is 36. The values of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  which minimize

Z are 7,10', 54", and 14", respectively, and the corresponding cost of the bridge is \$184,840 or \$15.66/sq. ft. of the bridge.

3.3.2 Example 2. The summary of the results of this example is given in Table 3-2. The lower bound of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  are 5,9', 48", and 14", respectively. The upper bound of  $N$ ,  $L_c$ ,  $h$  and  $b_f$  are 7,13', 54", and 22", respectively. The number of designs for an exhaustive search is 150.

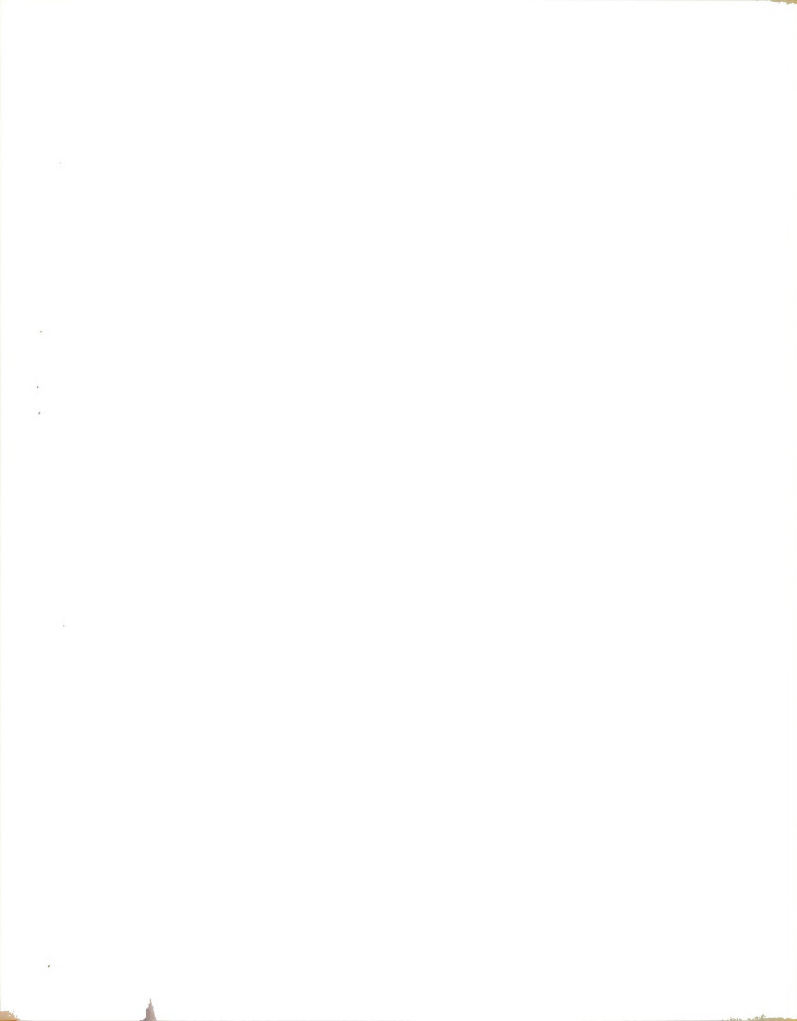
For the first cycle,  $N = 7$ ,  $L_c = 11$ , and  $h = 48$ ,  $b_f = 14$  are the co-ordinates of the centers of Grid I, and Grid II respectively. The number of designs covered when a minimum-cost design by the grid search method is obtained is 48. The values of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  which minimize  $Z$  are 6,13', 54", and 14", respectively. The corresponding cost of the bridge is \$160,130, or \$15.33/sq. ft. of the bridge.

3.3.3 Example 3. The summary of the results of this example is given in Table 3-3. The lower bounds of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  are 3,5', 42", and 14", respectively. The upper bounds of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  are 4,9', 66", and 22", respectively. The number of designs required to be investigated for an exhaustive search is 250.

For the first cycle,  $N = 4$ ,  $L_c = 7$ , and  $h = 42$ ,  $b_f = 14$  are the co-ordinates of the centers of Grid I, and Grid II, respectively. The number of designs investigated when the minimum-cost design by the grid search method is obtained is 32. The values of  $N$ ,  $L_c$ ,  $h$ , and  $b_f$  which

minimize are 4,8', 42", and 14", respectively. The corresponding cost of the bridge is \$45,960, or \$11.86/sq. ft. of the bridge.

It may be noted that each of the above examples the grid search method involved a substantially smaller number of designs than required by an exhaustive search. Comparison of procedures to obtain nearly optimal designs of the bridge by the grid search and linear programming method will be postponed after the next chapter. With a view to improving the efficiency of the search method, the method of bounds is considered and is described in the next chapter.



## CHAPTER IV

### DESIGN BY THE METHOD OF BOUNDS

#### 4.1 Introduction

The computer-aided designs of two-span cantilever highway bridges by using linear programming and grid search method have been explained in Chapters II, and III, respectively. The grid search method seems more efficient than the linear programming method. The method of bounds, as described in this chapter, is investigated as an improvement upon the grid search method by reducing the amount of computational work. The key idea that makes the reduction possible is that the search to determine a nearly optimal design may be terminated if the cost of the current feasible design is within a pre-specified "tolerance" of an "effective lower bound".

Thus the first step of the method is the establishment of an effective lower bound,  $Z_f$ . It should be as close to the largest lower bound as possible. The latter, of course, is the optimal solution and is unknown. A certain amount of insight of the engineering problem is necessary in order to establish an efficient  $Z_f$ . Aside from its dependence on an evaluation of  $Z_f$ , the accuracy of the method also depends on the tolerance,  $\epsilon$ . In fact, the method may not lead to a solution if values of  $Z_f$  and  $\epsilon$  are such that  $(Z_f + \epsilon)$  is less



than the true optimal solution. This can be seen with the help of Figure 4-1. The method is described in detail in the following section.

#### 4.2 The Method of Bounds

A nearly optimal design of the bridge by using the method of bounds may be determined as follows. Refer to Figure 4-1. (a) Determine an effective lower bound of the cost of the bridge,  $Z_f$ . (b) use a suitable search method to obtain a feasible design with cost  $Z$ , which is within a pre-specified tolerance  $\epsilon$  of the effective lower bound, and (c) terminate the search when first such design is found.

A feasible design is said to be a nearly optimal if both of the following requirements are satisfied.

$$Z \leq Z^u \quad (4-1)$$

$$Z \leq Z_f + \epsilon \quad (4-2)$$

where  $Z^u$  is an upper bound of  $Z$  (monotonically decreasing), and  $\epsilon$  is an assumed tolerance.

An upper bound of  $Z$  is the cost of a feasible design. During the search process, when a new feasible design is found and if its cost satisfies the condition (4-1) that cost will be taken as  $Z^u$  for the next feasible design. In this manner  $Z^u$  would be monotonically decreased.

An effective lower bound  $Z_f$ , is essentially a fully-stressed design of the bridge. It is obtained by the following procedure. The reinforced concrete slab is designed using



the conventional elastic design procedure. The welded steel plate girder is designed such that the flange plates satisfy the maximum bending stress criterion only, and the web plate satisfies the maximum shear stress criterion, and local buckling requirements. Other requirements such as lateral buckling of the girder are not satisfied. Therefore, the design is an infeasible one.

The depth of the web plate is set equal to the maximum allowable,  $1/20$  of the span length  $L_1$ . This results in less material in the flange plates. The width of the flange plates is set equal to the minimum allowable, i.e., 14". The values of  $N$  and  $L_c$  are such that  $Z_f$  is minimized by using an exhaustive search. As mentioned early, the main objective of this technique is to determine a lower bound sufficiently close to the least upper bound  $Z^*$  without exceeding it, and the procedure should not be overly time-consuming.

The value of the tolerance  $\epsilon$  may be assumed to be of the order of 10% of the effective lower bound. If the value of  $Z_f$  and  $\epsilon$  are such that a nearly optimal design cannot be located, then the values of  $Z_f$  and/or  $\epsilon$  should be revised.

#### 4.3 Practical Applications

The designs of the three bridges considered earlier in Chapters II, and III are again undertaken to test the method of bounds.

A design of the bridge is obtained using the computer program "BRIDGE", which is modified for the method of bounds by adding subroutines to determine  $Z_f$ , and adding the

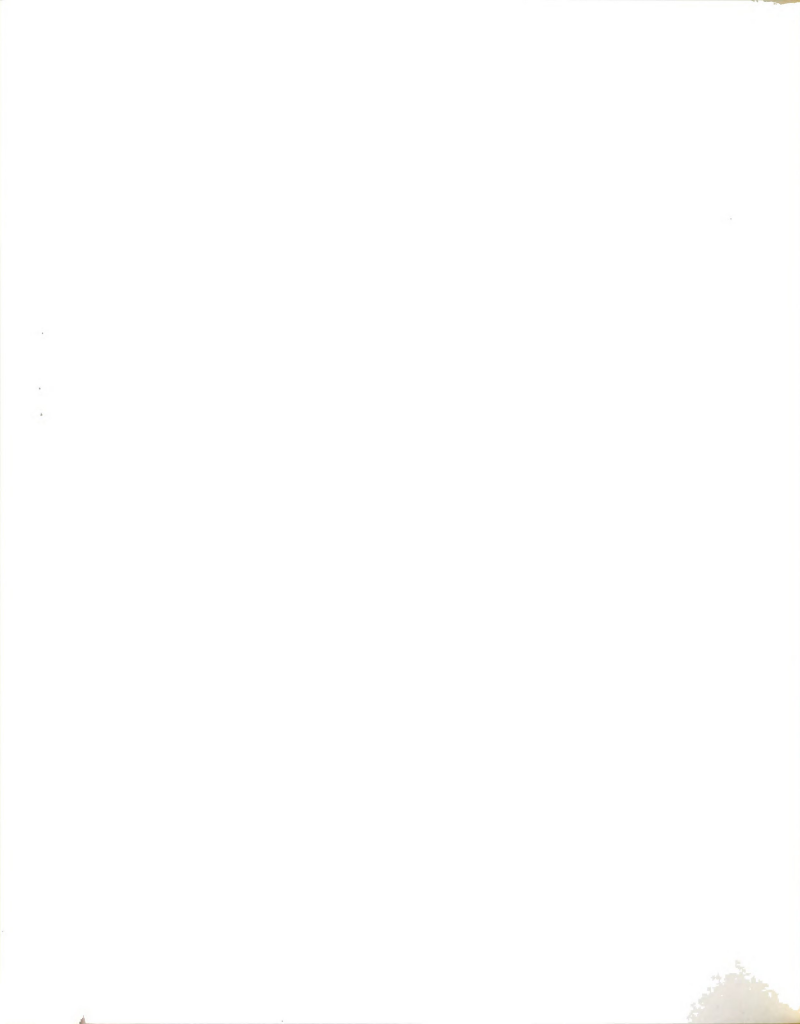
conditions (4-1), and (4-2) to check for termination of the procedure. The program is available for use on the B 5500 computer system of the MDSH. Input data for the program is the design data of bridge and output consists of the details of a complete design. The User's Manual of the program is given in Appendix I, the listing of the program is given in Appendix II.

4.3.1 Example 1: A summary of the results of Example 1 is given in Table 4-1. The values of  $Z^*$ ,  $Z_f + \epsilon$ , and  $Z_f$  are \$181,900, \$187,895, and \$169,275, respectively. Three designs were investigated to locate a nearly optimal design. The cost of the nearly optimal design is 2.5% more than that of the true optimal design.

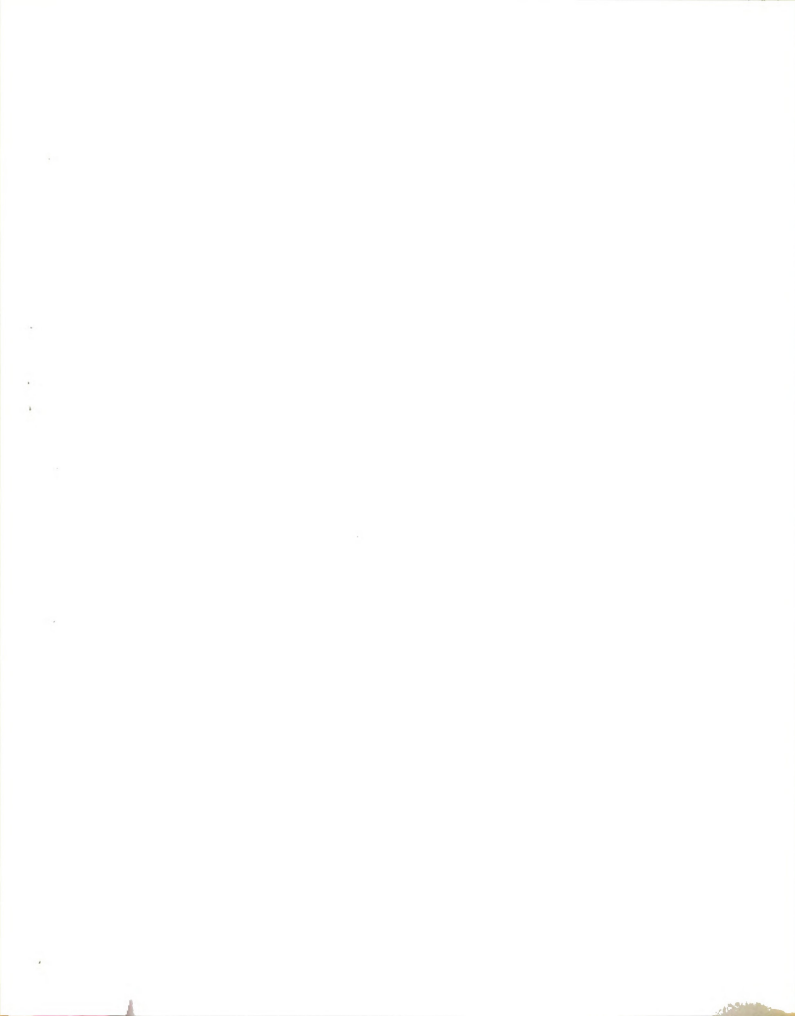
4.3.2 Example 2: A summary of the results of Example 2 is given in Table 4-2. The values of  $Z^*$ ,  $Z_f + \epsilon$ , and  $Z_f$  are \$160,130, \$162,616, and \$146,501, respectively. Twenty three designs were investigated to locate a nearly optimal design, and its cost is 0.24% more than that of the optimal design.

4.3.3 Example 3: A summary of the results of Example 3 is given in Table 4-3. The values of  $Z^*$ ,  $Z_f + \epsilon$ , and  $Z_f$  are \$45,956, \$46,344, and \$42,751, respectively. Nine designs were investigated to locate a nearly optimal design. The nearly optimal design of this example is the same as the optimal design determined by the exhaustive search.

The foregoing examples indicate that the method of bounds requires less number of designs than that required by the



grid search method. Thus, the computer-aided near optimal designs of three bridges have been completed by three different methods of optimization. A detailed comparison of the results of these examples is presented in the next chapter.



## CHAPTER V

### COMPARISON OF THE METHODS OF DESIGN AND CONCLUSIONS

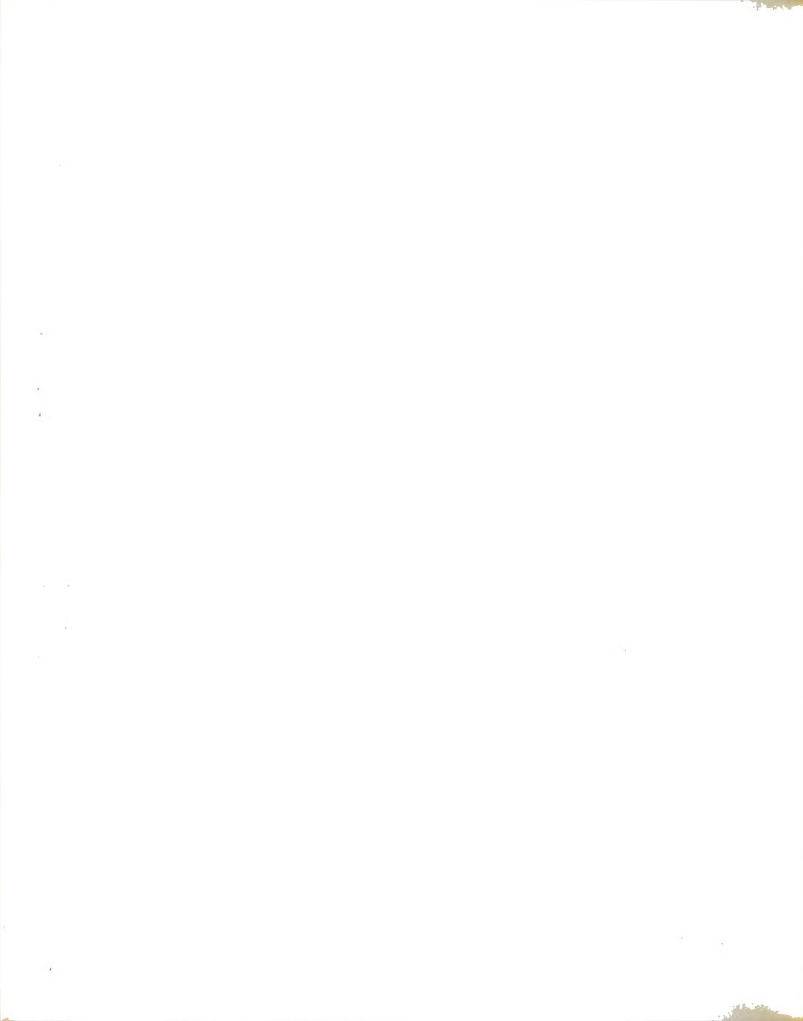
Computer-aided near optimal designs of two-span cantilever highway bridges have been presented in the previous chapters. Linear programming, the grid search, and the method of bounds have been used to obtain designs of three bridges. Computer programs have been prepared to implement the above methods of optimization. Comparison of these methods of optimization, and conclusions based upon the results of the research are given in the following sections.

#### 5.1 General

Viewed as a mathematical programming problem, the cost-optimization of the bridge design problem is a nonlinear one. Only approximate solutions of the problem are practicable. The approach considered consisted of linearization and the solutions of a series of linear programming problems. However, such a procedure proved to be rather cumbersome and with no guarantee of good accuracy.

The search methods--the grid search method, and the method of bounds--involve simple mathematical strategies with results that are at least no less accurate than the





linear programming approach. Another factor worthy of mentioning is the consideration of future changes in design specifications. It would appear that to incorporate in the computer program any change in the design requirements would involve considerably less difficulty for the search method than that for the linear programming method.

The major difference between the grid search method and the method of bounds lies in the criteria used to terminate the search. By terminating the search in the method of bounds when the cost of a current design is within a pre-specified percentage of its effective lower bound, the number of designs which are investigated to locate a nearly optimal design can be reduced considerably from that required by the grid search method. However, the formulation of the method of bounds requires considerable insight of the bridge design problem in order to determine an effective lower bound  $Z_f$  and a tolerance  $\epsilon$ . It seems that only a trained structural engineer would possess such insight. Furthermore, the method without modification, would fail if improper values of  $Z_f$  and  $\epsilon$  are used.

## 5.2 Comparison of the Results

The results of cost-optimization of Examples 1, 2, and 3 obtained by the various optimization methods, are given in Table 5-1. Computer times used to obtain the solutions of these examples are given in Table 5-2.

A comparison of the cost of the bridge design obtained by using the traditional method with the costs obtained by

using the optimization methods indicates that the latter produce more economical designs. Similarly, a comparison of the costs of the bridge obtained by the methods of optimization shows that the difference between the optimal cost determined by the exhaustive search and a nearly optimal cost determined by the grid search, or by the method of bounds, is less than 2.5% of the optimal cost.

The computer cost to obtain a nearly optimal solution is about 0.01% to 0.02% of the total cost of the bridge, whereas the average saving in the total cost of the bridge is about 4.0% of the total cost of the bridge designed by using the traditional method. Thus, the computer-aided near optimal design would result in a net saving of about 4.0%.

### 5.3 Conclusions

The following conclusions based upon the results of the present study.

(1) It is feasible to use computer-aided designs for the bridges considered herein and that such practice would result in engineering economy.

(2) The search method is more suitable than the linear programming method for the type of structural design problems considered in which the objective function and the constraints are nonlinear, and the variables are largely discrete.

(3) The method of bounds seems to be a more efficient approach than the grid search method. However, this method

does require an insight of the bridge design problem which only trained structural engineers are likely to possess.

An extension of the present work may include the application of a similar approach to other types of bridges and also to the use of the Load Factor design method allowed by the AASHO specifications. However, a more basic problem that needs study appears to be an automatic procedure for setting the new values for  $Z_f$  and  $\epsilon$  in the event that the search procedure could not produce a design that satisfies the bounds first set.

1850

1851

1852

1853

1854

1855

1856

1857

1858

1859



(a) General View

Figure 1-1 Photographs of A Two-Span Cantilever  
Highway Bridge.

(To be continued)



(b) Cantilever and Hinge

Figure 1-1 (Continued)





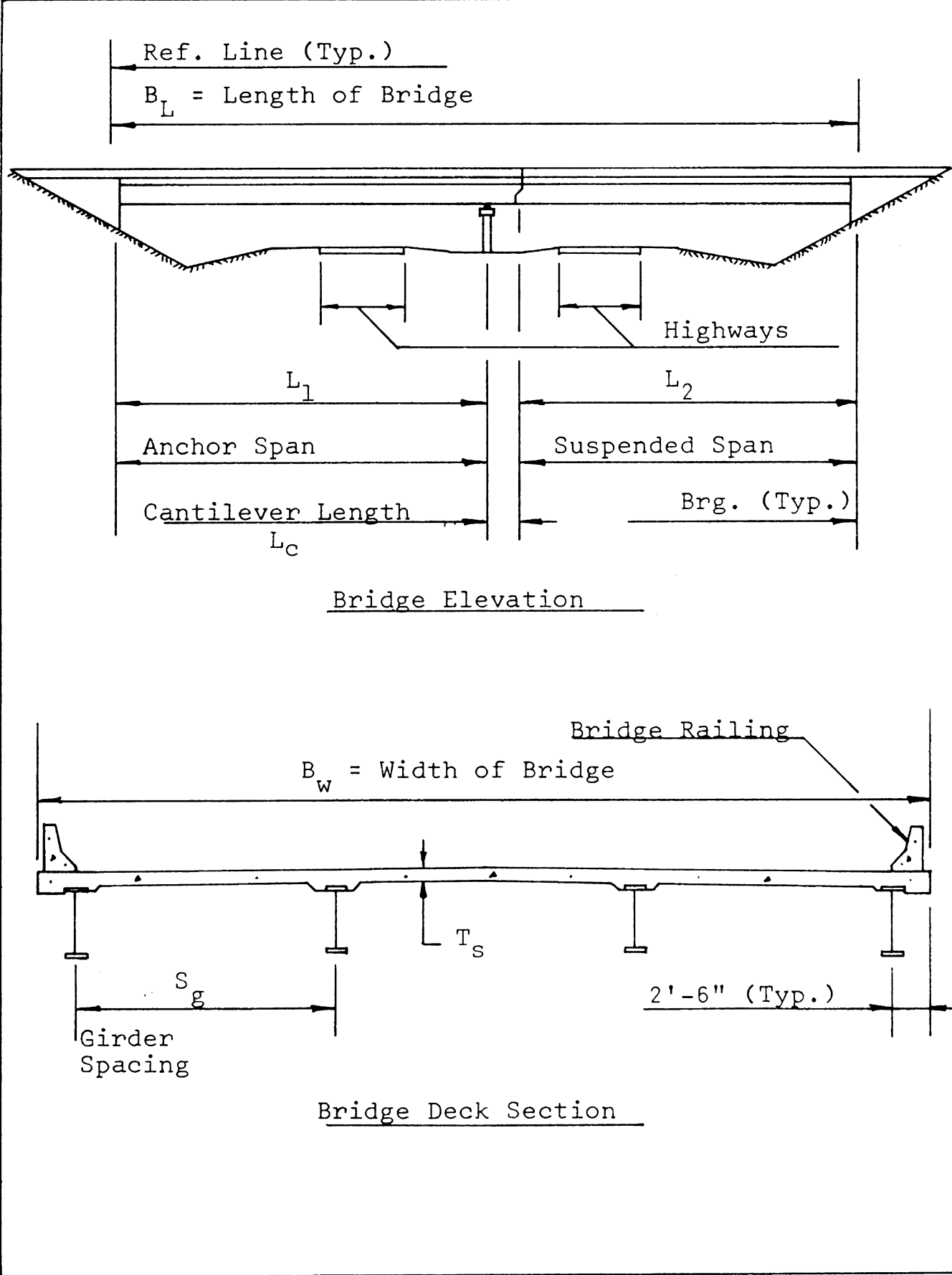


Figure 1-2: Two-Span Cantilever Bridge Details

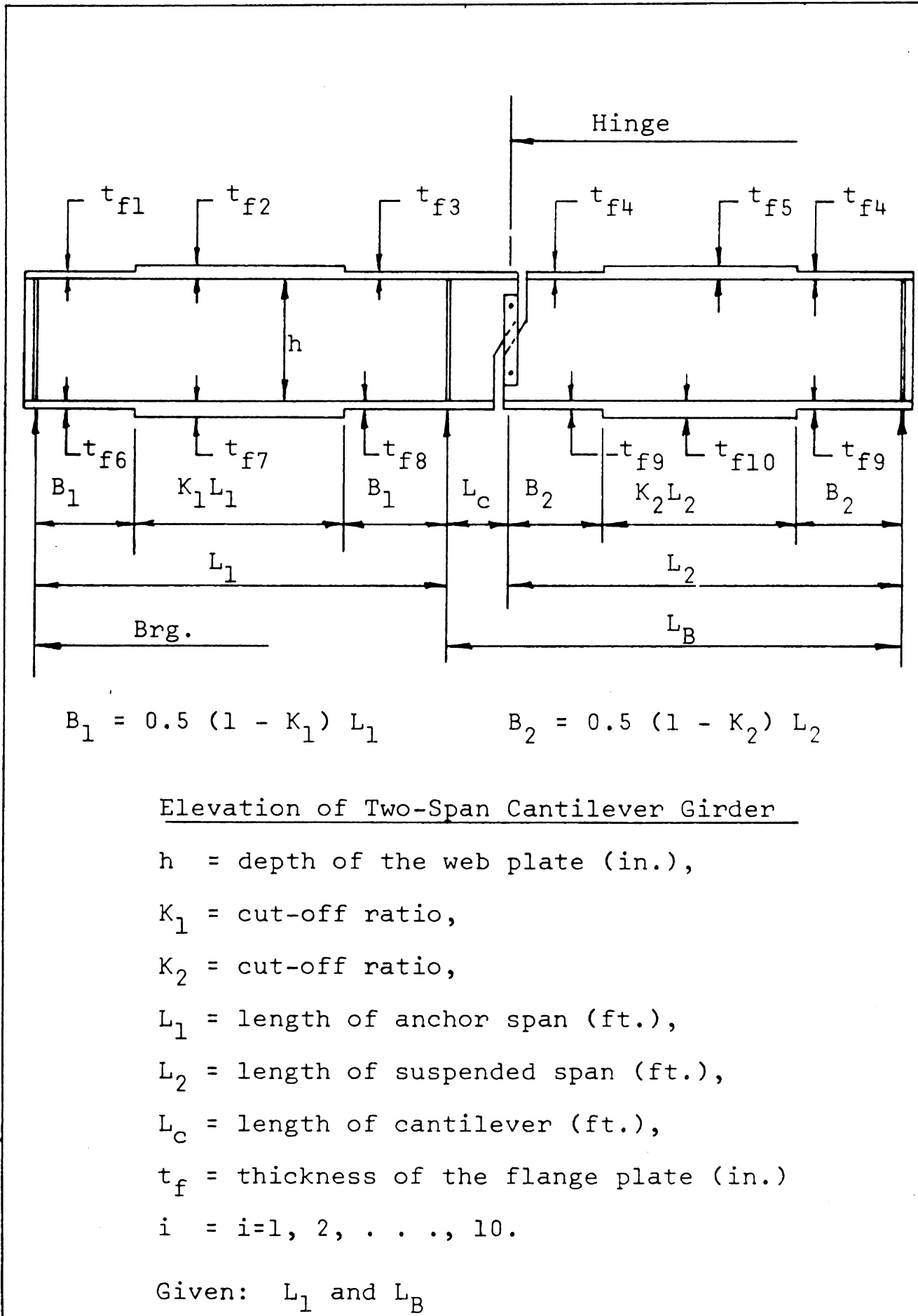
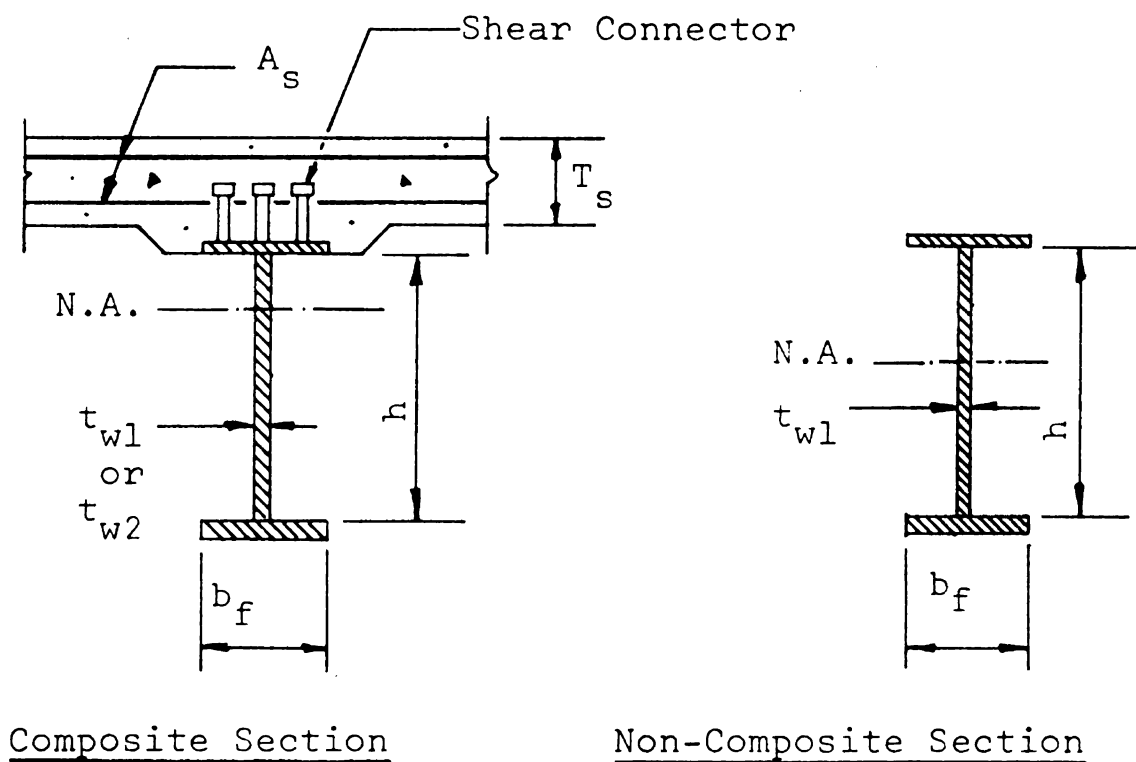


Figure 1-3: Two-Span Cantilever Girder Details

(To be continued)



$A_s$  = area of steel reinforcement in slab ( $\text{in.}^2/\text{ft.}$ ),

$b_f$  = width of flange plate (in.),

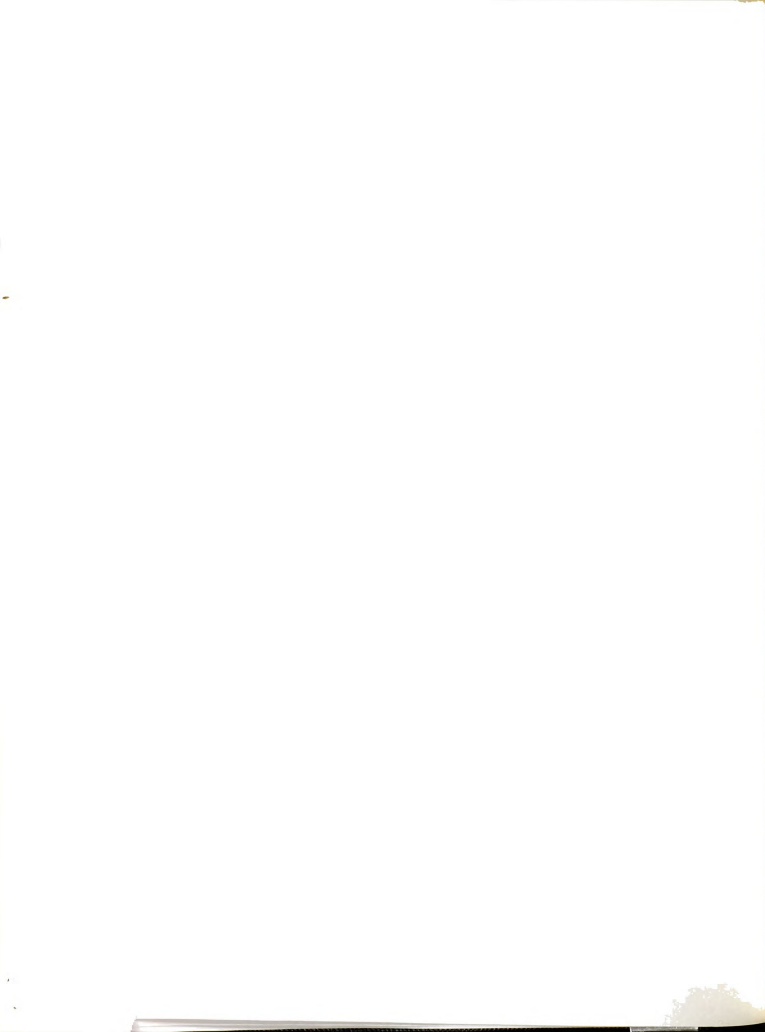
$T_s$  = thickness of slab (in.),

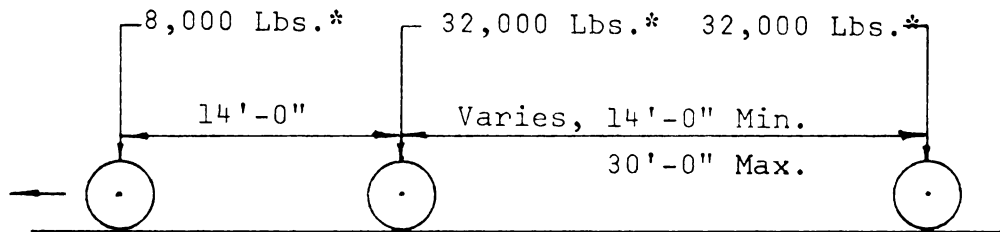
$t_{w1}$  = thickness of the web plate in the anchor span (in.),

$t_{w2}$  = thickness of the web plate in the suspended span (in.),

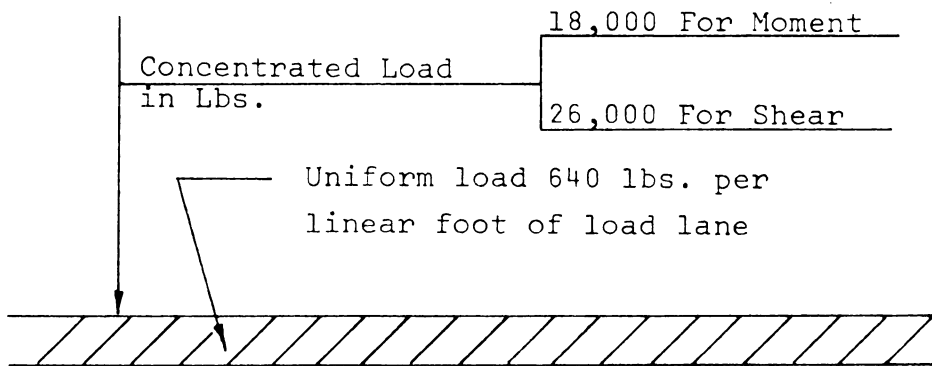
N.A.= neutral axis.

Figure 1-3 (continued)





Standard HS 20 Truck Loading



HS 20 Lane Loading

\* Weight on the axle.

Figure 1-4: Standard HS 20 Truck Loading



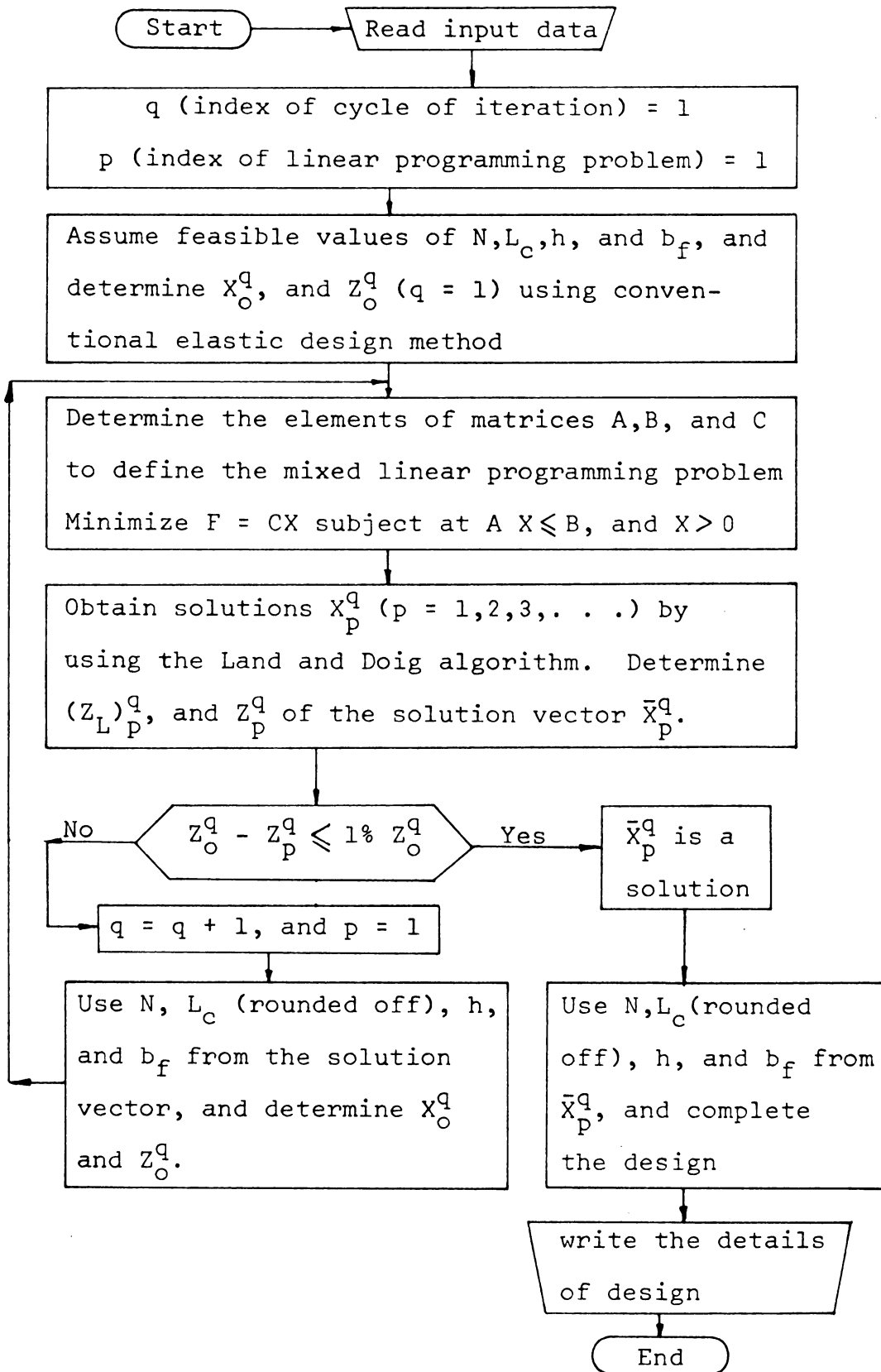


Figure 2-1: Flow Chart of the Iterative Method

To minimize  $Z = Z(N, L_c, h, b_f)$

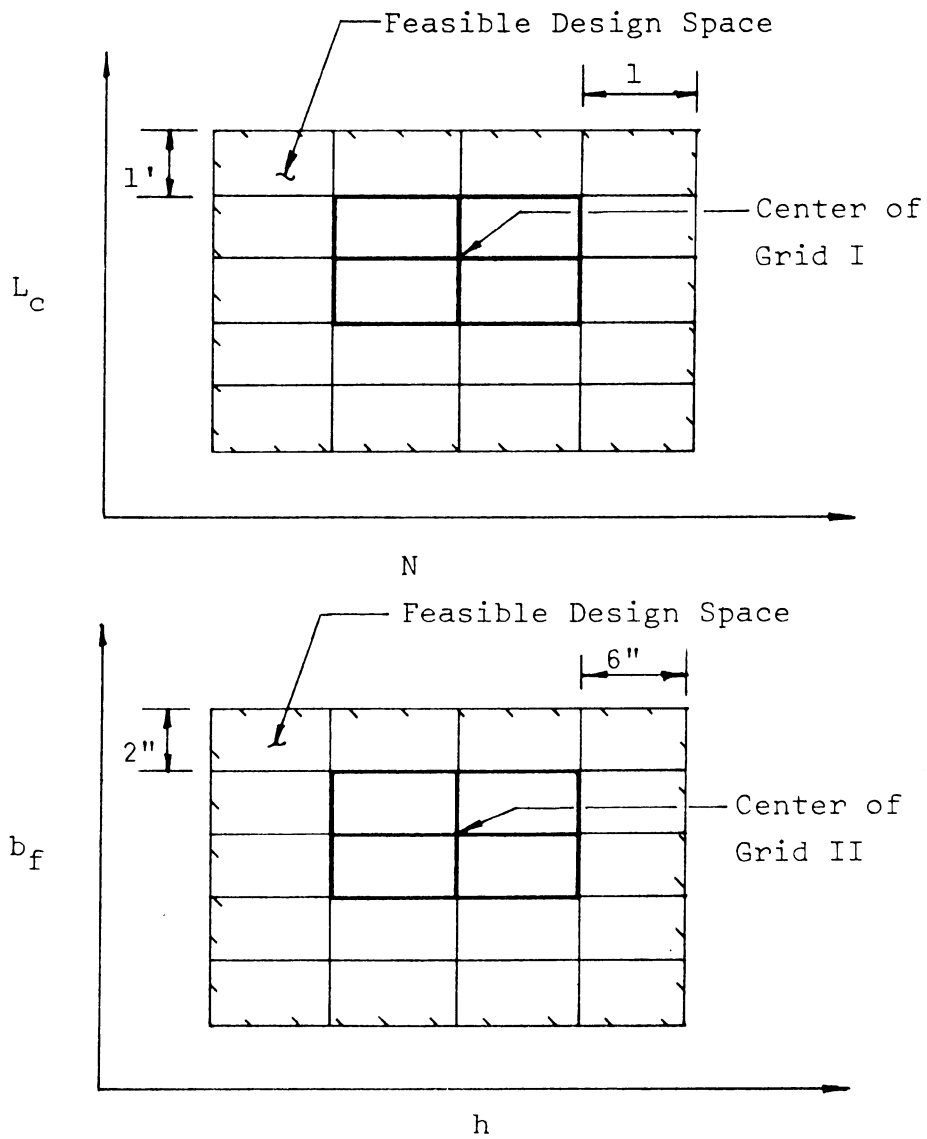


Figure 3-1: The Grid Search Method





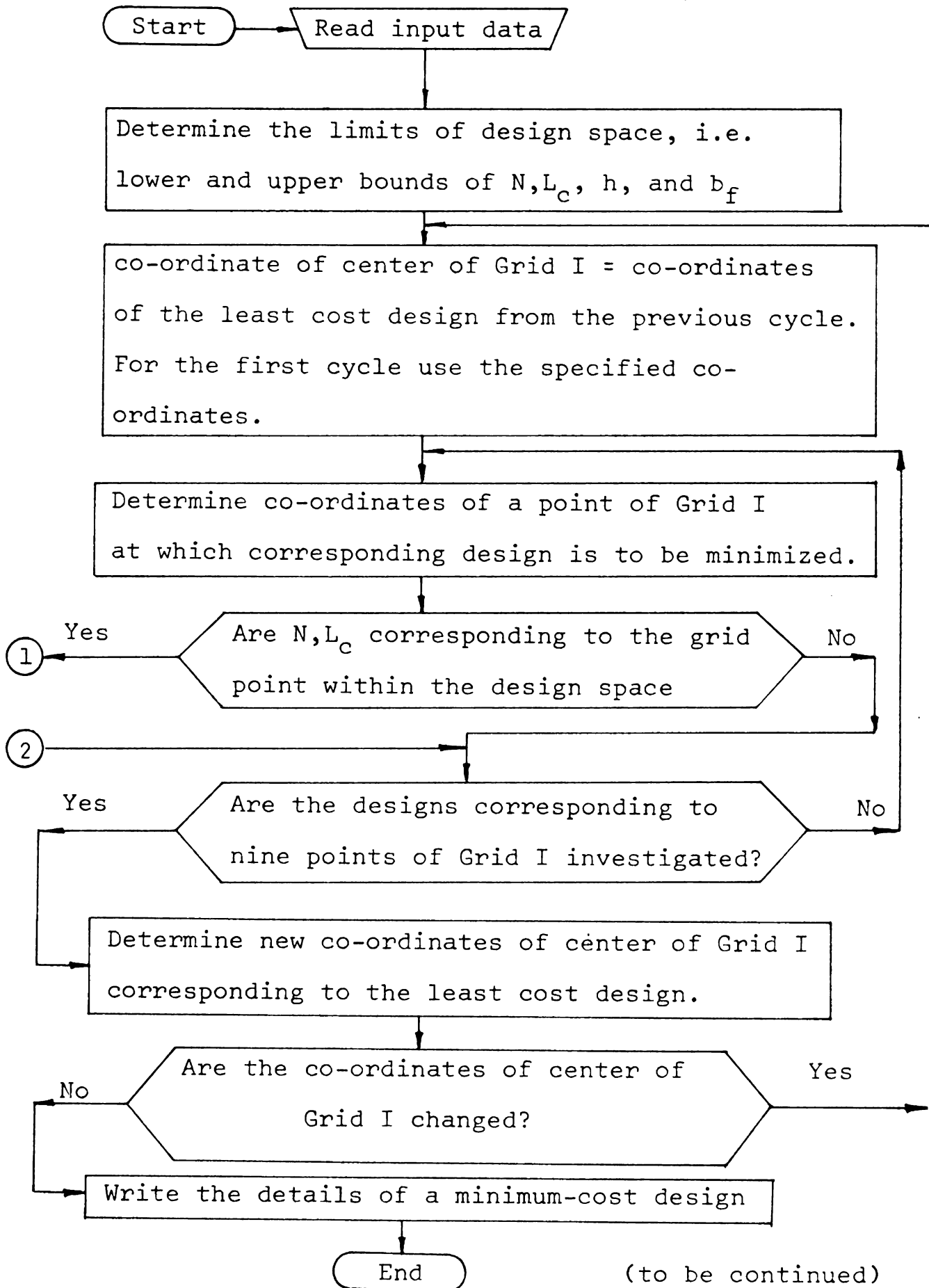


Figure 3-2: Flow Chart of the Grid Search Method



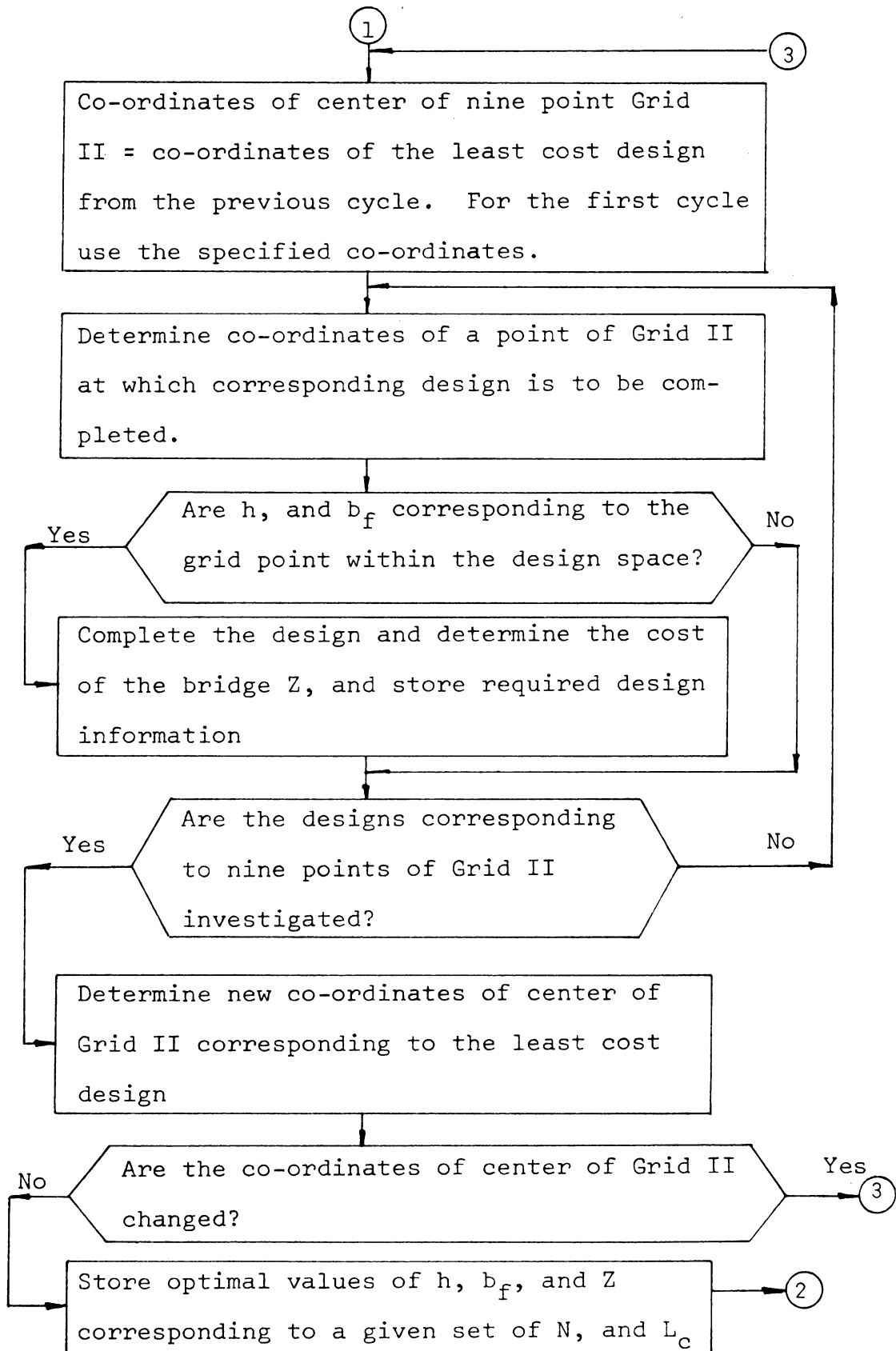
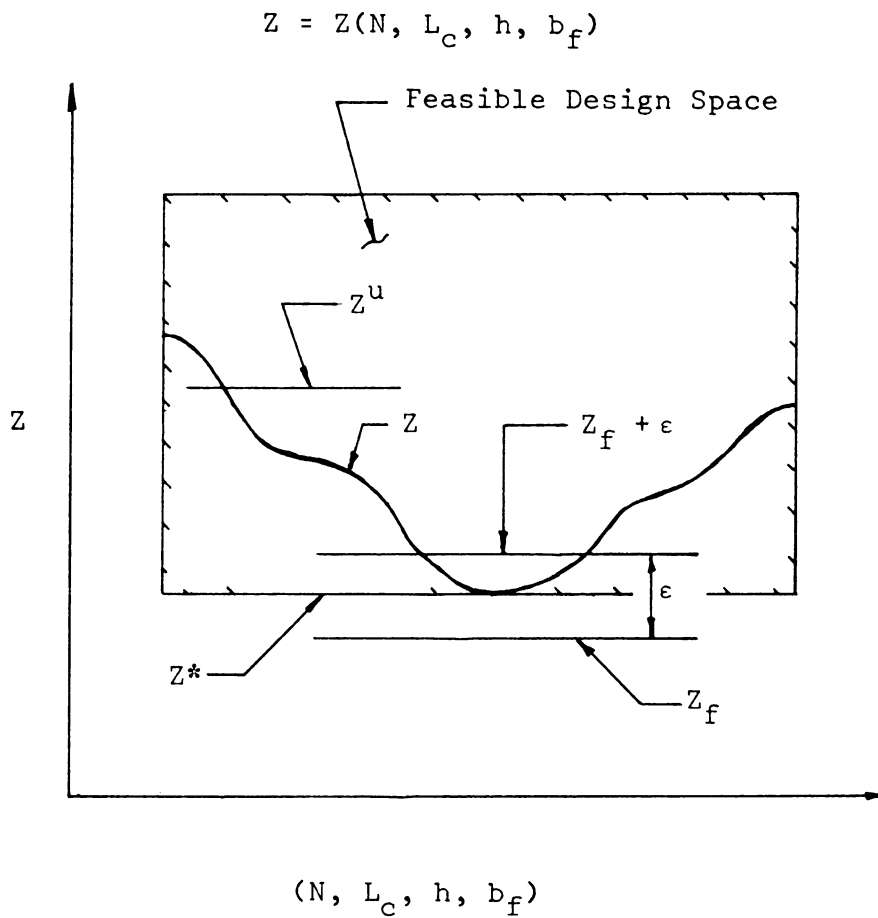


Figure 3-2 (continued)





$Z$  = cost of a current feasible design,

$Z^*$  = the least upper bound of  $Z$ ,

$Z^u$  = upper bound of  $Z$ , decreasing as the search goes on,

$Z_f$  = an "effective lower bound,"

$\epsilon$  = tolerance which is some assumed percentage of the value of  $Z_f$ .

Figure 4-1: The Method of Bounds



Table 2-1: Data for Design of the Bridges

| Dimensions and Loading | Example 1 | Example 2 | Example 3 |
|------------------------|-----------|-----------|-----------|
| Bridge Length, $B_L$   | 236.00    | 232.00    | 152.00    |
| Bridge Width, $B_W$    | 50.00     | 45.00     | 25.50     |
| Span Length, $L_1$     | 116.75    | 114.75    | 74.75     |
| Span Length, $L_B$     | 116.75    | 114.75    | 74.75     |
| Truck Loading          | HS 20     | HS 20     | HS 20     |

Properties ofMaterials used for Construction of the Bridges

Structural Concrete: minimum compressive strength  
at 28 days is 3,000 psi.

Steel Reinforcement: Intermediate Grade Steel  
deformed bars. Yield strength  
of steel is 40,000 psi., and  
allowable tension is 20,000 psi.

Structural Steel: ASTM A-36

Cost of Material

The cost of material includes costs of material and  
labor.

Structural Concrete: \$120.00/cu. yd.

Steel Reinforcement: \$ 0.22/lb.

Structural Steel: \$ 0.30/lb.





Table 2-2: Results of Cost-Optimization of Example 1  
(Using Linear Programming)

| Design Variables | $x_0^1$ | $x_1^1$ | $x_2^1$ | $x_3^1$ | $x_4^1$ |
|------------------|---------|---------|---------|---------|---------|
| $T_S$            | 7.50    | 7.50    | 8.00    | 8.00    | 8.00    |
| $A_S$            | 0.66    | 0.66    | 0.75    | 0.75    | 0.75    |
| $t_{f1}$         | 0.75    | 0.75    | 0.75    | 0.75    | 0.75    |
| $t_{f2}$         | 1.13    | 1.00    | 1.00    | 1.13    | 1.00    |
| $t_{f3}$         | 0.75    | 0.75    | 0.75    | 0.98    | 0.75    |
| $t_{f4}$         | 0.75    | 0.75    | 0.75    | 0.75    | 0.75    |
| $t_{f5}$         | 1.13    | 1.00    | 1.00    | 1.11    | 1.00    |
| $t_{f6}$         | 1.13    | 1.13    | 1.13    | 1.31    | 1.13    |
| $t_{f7}$         | 2.13    | 2.08    | 2.08    | 2.34    | 2.08    |
| $t_{f8}$         | 1.13    | 1.13    | 1.13    | 1.13    | 1.13    |
| $t_{f9}$         | 1.00    | 1.00    | 1.00    | 1.00    | 1.00    |
| $t_{f10}$        | 2.00    | 1.97    | 1.97    | 2.23    | 1.97    |
| $K_1$            | 0.64    | 0.64    | 0.64    | 0.64    | 0.64    |
| $K_2$            | 0.65    | 0.65    | 0.65    | 0.65    | 0.65    |
| $t_{w1}$         | 0.50    | 0.50    | 0.50    | 0.50    | 0.50    |
| $t_{w2}$         | 0.50    | 0.50    | 0.50    | 0.50    | 0.50    |
| $h$              | 48.     | 50.16   | 53.10   | 48.     | 54.     |
| $b_f$            | 16.     | 16.     | 16.     | 16.     | 16.     |
| $L_c$            | 7.00    | 11.06   | 11.10   | 11.50   | 11.10   |
| $N$              | 8.      | 7.36    | 7.      | 7.      | 7.      |
| $Z$              | 194,800 |         |         | 188,930 | 188,620 |
| $Z_L$            |         | 487,128 | 487,835 | 492,291 | 490,884 |

(To be continued)

Table 2-2 (Continued)

| Design Variables | $X_0^2$ | $X_1^2$ | $X_0^3$ |
|------------------|---------|---------|---------|
| $T_S$            | 8.00    | 8.50    | 8.50    |
| $A_S$            | 0.75    | 0.75    | 0.75    |
| $t_{f1}$         | 0.75    | 0.75    | 0.75    |
| $t_{f2}$         | 1.00    | 1.18    | 1.25    |
| $t_{f3}$         | 0.75    | 1.02    | 1.00    |
| $t_{f4}$         | 0.75    | 0.75    | 0.75    |
| $t_{f5}$         | 1.00    | 1.21    | 1.13    |
| $t_{f6}$         | 1.13    | 1.44    | 1.25    |
| $t_{f7}$         | 2.13    | 2.50    | 2.50    |
| $t_{f8}$         | 1.13    | 1.13    | 1.25    |
| $t_{f9}$         | 1.00    | 1.00    | 1.25    |
| $t_{f10}$        | 2.00    | 2.40    | 2.38    |
| $K_1$            | 0.62    | 0.65    | 0.65    |
| $K_2$            | 0.63    | 0.63    | 0.63    |
| $t_{w1}$         | 0.56    | 0.56    | 0.56    |
| $t_{w2}$         | 0.50    | 0.50    | 0.56    |
| $h$              | 54.     | 54.     | 54.     |
| $b_f$            | 16.     | 16.     | 16.     |
| $L_c$            | 11.50   | 11.50   | 11.50   |
| $N$              | 7       | 6       | 6       |
| $Z$              | 186,800 | 185,050 | 184,900 |
| $Z_L$            |         | 459,478 |         |



Table 2-3: Results of Cost-Optimization of Example 2  
(Using Linear Programming)

| Design Variables | $x_0^1$ | $x_1^1$ | $x_2^1$ | $x_3^1$ | $x_4^1$ |
|------------------|---------|---------|---------|---------|---------|
| $T_S$            | 8.00    | 8.00    | 8.00    | 8.00    | 8.00    |
| $A_S$            | 0.66    | 0.66    | 0.66    | 0.66    | 0.75    |
| $t_{f1}$         | 0.75    | 0.75    | 0.75    | 0.75    | 0.75    |
| $t_{f2}$         | 1.13    | 1.13    | 1.19    | 1.25    | 1.00    |
| $t_{f3}$         | 0.75    | 0.75    | 0.75    | 0.78    | 0.75    |
| $t_{f4}$         | 0.75    | 0.75    | 0.75    | 0.75    | 0.75    |
| $t_{f5}$         | 1.13    | 1.13    | 1.18    | 1.22    | 0.97    |
| $t_{f6}$         | 1.13    | 1.19    | 1.31    | 1.35    | 1.13    |
| $t_{f7}$         | 2.25    | 2.25    | 2.47    | 2.53    | 2.19    |
| $t_{f8}$         | 1.13    | 1.13    | 1.13    | 1.13    | 1.13    |
| $t_{f9}$         | 1.13    | 1.13    | 1.13    | 1.13    | 0.88    |
| $t_{f10}$        | 2.13    | 2.15    | 2.36    | 2.41    | 2.10    |
| $K_1$            | 0.65    | 0.65    | 0.65    | 0.65    | 0.65    |
| $K_2$            | 0.63    | 0.63    | 0.63    | 0.63    | 0.63    |
| $t_{w1}$         | 0.50    | 0.50    | 0.50    | 0.50    | 0.55    |
| $t_{w2}$         | 0.50    | 0.50    | 0.50    | 0.50    | 0.56    |
| $h$              | 48.     | 51.56   | 49.08   | 48.     | 54.     |
| $b_f$            | 16.     | 16.     | 16.     | 16.     | 16.     |
| $L_c$            | 6.      | 10.48   | 10.03   | 10.03   | 10.03   |
| $N$              | 7.      | 6.1     | 6.      | 6.      | 6.      |
| $Z$              | 173,500 |         |         | 166,500 | 161,300 |
| $Z_L$            |         | 436,471 | 437,756 | 438,822 | 438,362 |

(To be continued)

Table 2-3 (Continued)

| Design Variables | $x_0^2$ | $x_1^2$ | $x_2^2$ | $x_0^3$ |
|------------------|---------|---------|---------|---------|
| $T_S$            | 8.00    | 8.00    | 9.00    | 9.00    |
| $A_S$            | 0.75    | 0.75    | 0.81    | 0.81    |
| $t_{f1}$         | 0.75    | 0.75    | 0.75    | 0.75    |
| $t_{f2}$         | 1.00    | 1.03    | 1.34    | 1.25    |
| $t_{f3}$         | 0.75    | 0.78    | 1.03    | 0.88    |
| $t_{f4}$         | 0.75    | 0.75    | 0.75    | 0.75    |
| $t_{f5}$         | 1.00    | 1.00    | 1.32    | 1.35    |
| $t_{f6}$         | 1.13    | 1.24    | 1.56    | 1.50    |
| $t_{f7}$         | 2.13    | 2.29    | 2.77    | 2.88    |
| $t_{f8}$         | 1.13    | 1.13    | 1.14    | 1.50    |
| $t_{f9}$         | 1.00    | 1.00    | 1.00    | 1.38    |
| $t_{f10}$        | 2.00    | 2.17    | 2.64    | 2.63    |
| $K_1$            | 0.63    | 0.63    | 0.63    | 0.63    |
| $K_2$            | 0.62    | 0.63    | 0.62    | 0.62    |
| $t_{w1}$         | 0.56    | 0.56    | 0.56    | 0.63    |
| $t_{w2}$         | 0.56    | 0.56    | 0.56    | 0.56    |
| $h$              | 54.     | 54.     | 54.     | 54.     |
| $b_f$            | 16.     | 16.     | 16.     | 16.     |
| $L_c$            | 10.00   | 10.50   | 10.50   | 10.50   |
| $N$              | 6.      | 5.65    | 5.      | 5.      |
| $Z$              | 161,200 |         | 167,000 | 167,800 |
| $Z_L$            |         | 398,674 | 400,284 |         |

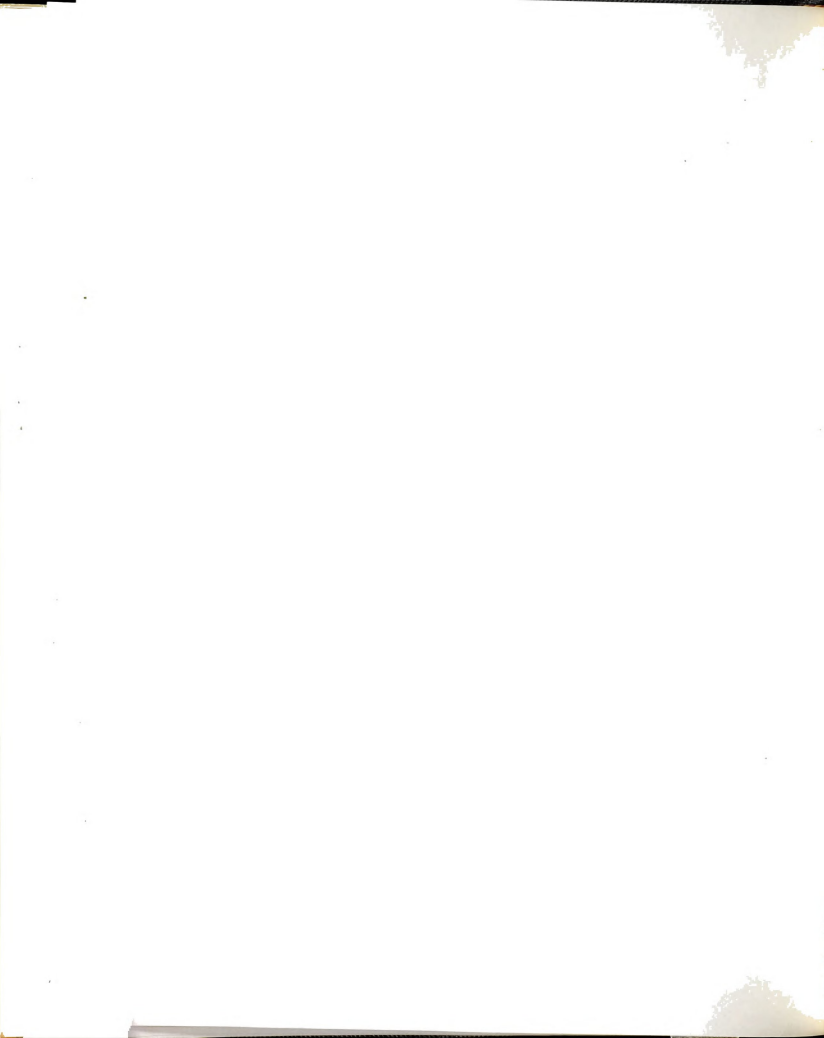


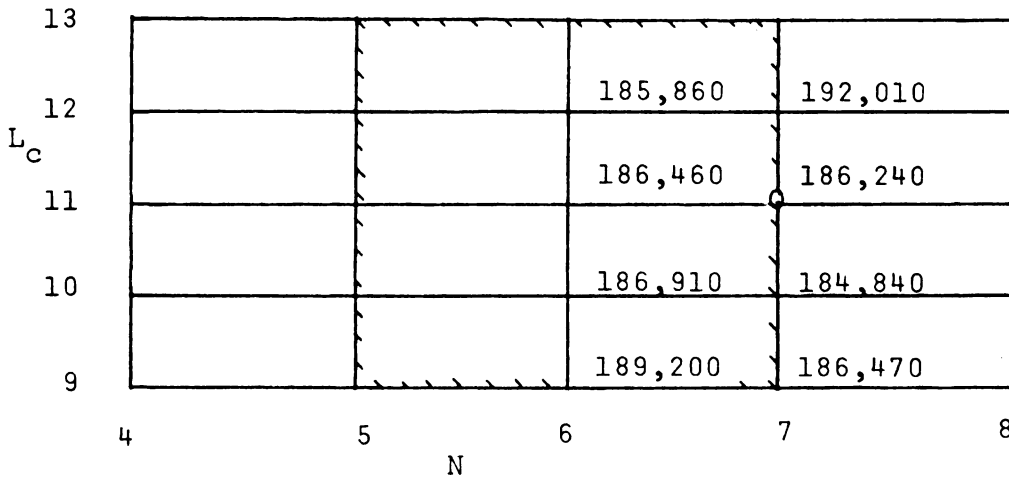
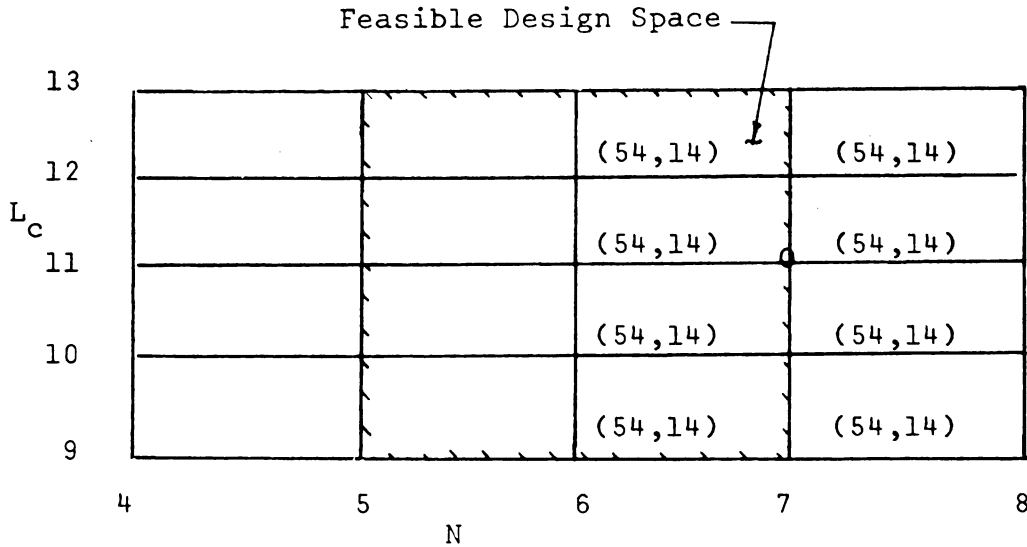
Table 2-4: Results of Cost-Optimization of Example 3  
(Using Linear Programming)

| Design Variables | $x_0^1$ | $x_1^1$ | $x_2^1$ | $x_3^1$ | $x_0^2$ |
|------------------|---------|---------|---------|---------|---------|
| $S_T$            | 8.00    | 8.00    | 9.00    | 9.00    | 9.00    |
| $A_S$            | 0.66    | 0.66    | 0.81    | 0.81    | 0.81    |
| $t_{f1}$         | 0.63    | 0.63    | 0.63    | 0.63    | 0.75    |
| $t_{f2}$         | 0.63    | 0.63    | 0.63    | 0.63    | 0.75    |
| $t_{f3}$         | 0.63    | 0.63    | 1.01    | 0.85    | 0.75    |
| $t_{f4}$         | 0.63    | 0.63    | 0.63    | 0.63    | 0.63    |
| $t_{f5}$         | 0.63    | 0.63    | 0.63    | 0.63    | 0.63    |
| $t_{f6}$         | 0.75    | 0.86    | 1.32    | 1.13    | 0.88    |
| $t_{f7}$         | 1.34    | 1.45    | 2.15    | 1.88    | 1.63    |
| $t_{f8}$         | 0.75    | 0.75    | 1.02    | 1.00    | 0.88    |
| $t_{f9}$         | 0.63    | 0.63    | 0.63    | 0.63    | 0.88    |
| $t_{f10}$        | 1.25    | 1.35    | 2.00    | 1.75    | 1.63    |
| $K_1$            | 0.59    | 0.58    | 0.60    | 0.60    | 0.59    |
| $K_2$            | 0.60    | 0.60    | 0.60    | 0.60    | 0.60    |
| $t_{w1}$         | 0.44    | 0.44    | 0.44    | 0.44    | 0.56    |
| $t_{w2}$         | 0.44    | 0.44    | 0.44    | 0.44    | 0.50    |
| $h$              | 42.     | 44.69   | 42.     | 48.     | 48.     |
| $b_f$            | 14.     | 14.     | 14.     | 14.     | 14.     |
| $L_C$            | 7.50    | 7.50    | 7.50    | 7.50    | 7.50    |
| $N$              | 4.      | 3.63    | 3.      | 3.      | 3.      |
| $Z$              | 46,440  |         | 46,320  | 46,260  | 47,860  |
| $Z_L$            |         | 105,671 | 108,075 | 107,962 |         |





Table 3-1: Summary of the Results of Example 1  
(Using the Grid Search Method)



Note: The optimal values of  $h$ , and  $b_f$  are shown in the first table, and corresponding values of  $Z$  are shown in the second table, for a given set of  $N$ , and  $L_c$ . 0 denotes the co-ordinates of the center of grid for the first cycle.



Table 3-2: Summary of the Results of Example 2  
(Using the Grid Search Method)

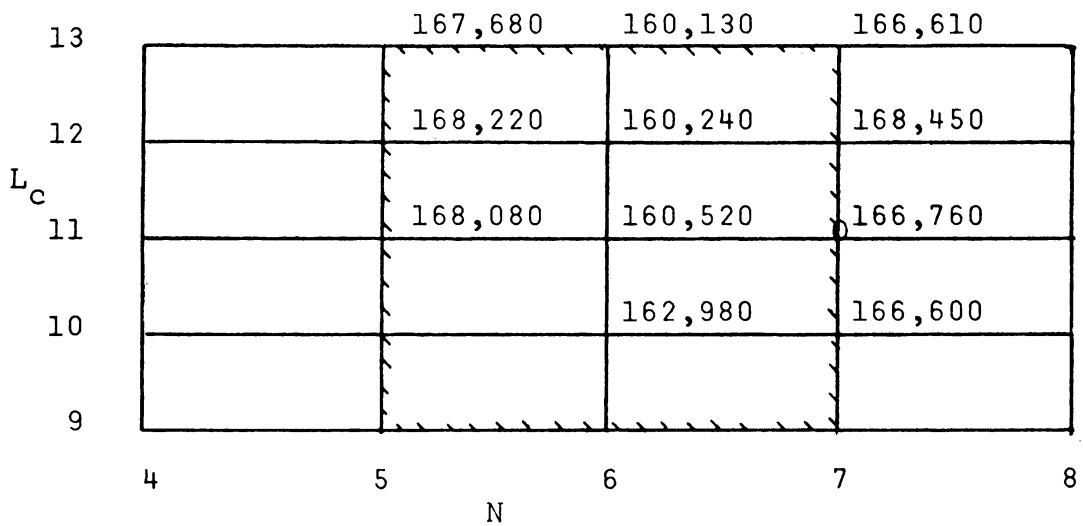
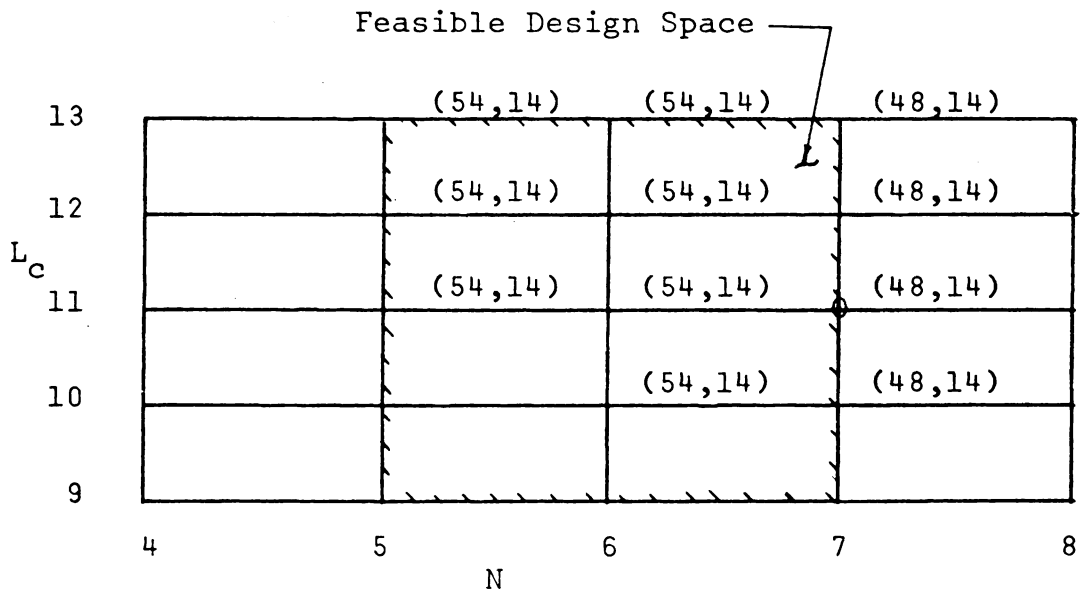




Table 3-3: Summary of the Results of Example 3  
(using the Grid Search Method)

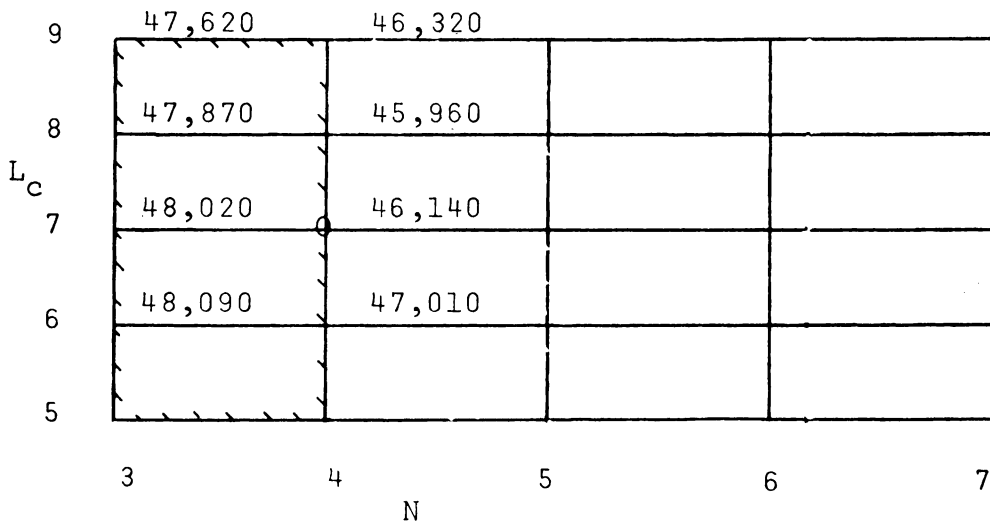
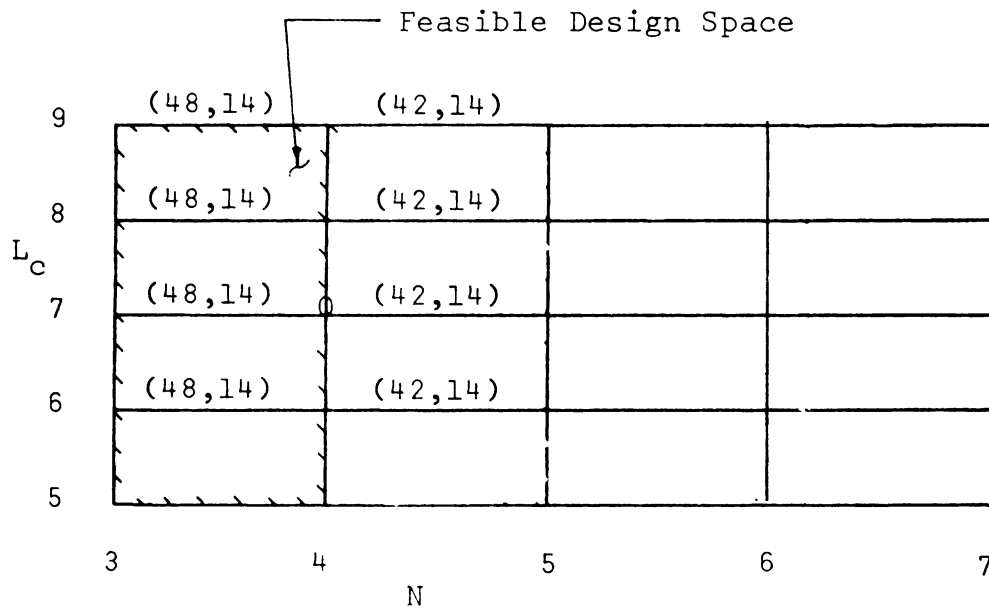
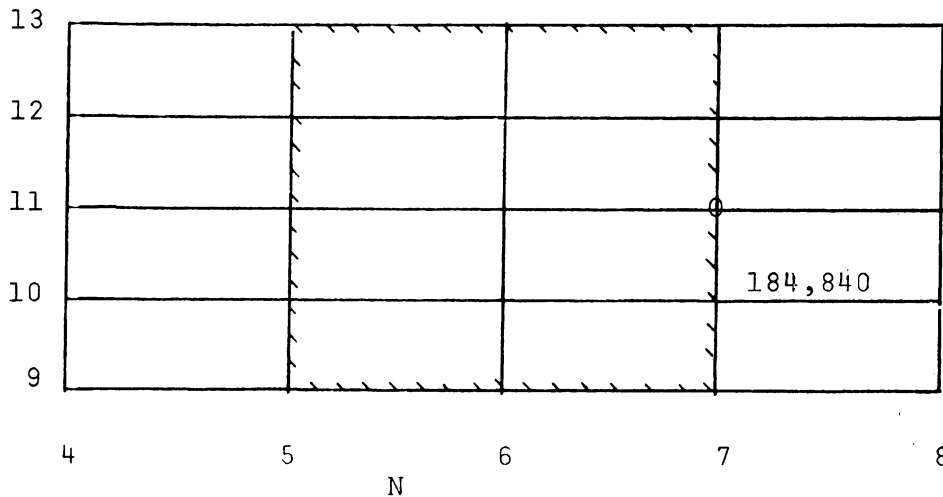
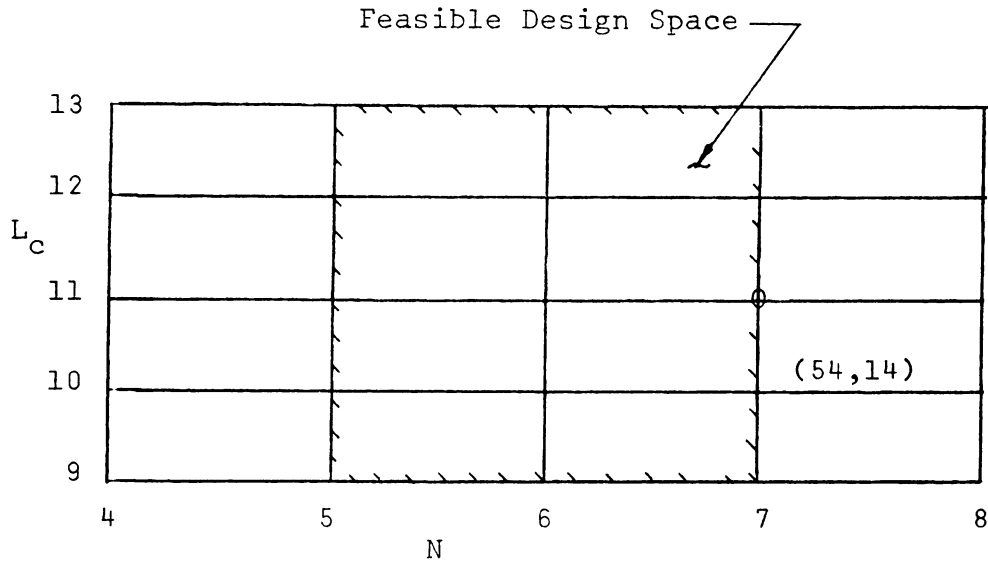




Table 4-1: Summary of the Results of Example 1  
(using The Method of Bounds)

$$Z^* = \$181,900 \quad Z_f = \$169,275 \quad Z_f + \epsilon = \$187,895$$



Note: The optimal values of  $h$ , and  $b_f$  are shown in the first table, and corresponding values of  $Z$  are shown in the second table, for a given set of  $N$ , and  $L_c$ . 0 denotes the co-ordinates of the center of grid for the first cycle.





Table 4-2: Summary of the Results of Example 2  
(using The Method of Bounds)

$$Z^* = \$160,130$$

$$Z_f = \$146,501$$

$$Z_f + \epsilon = \$162,616$$

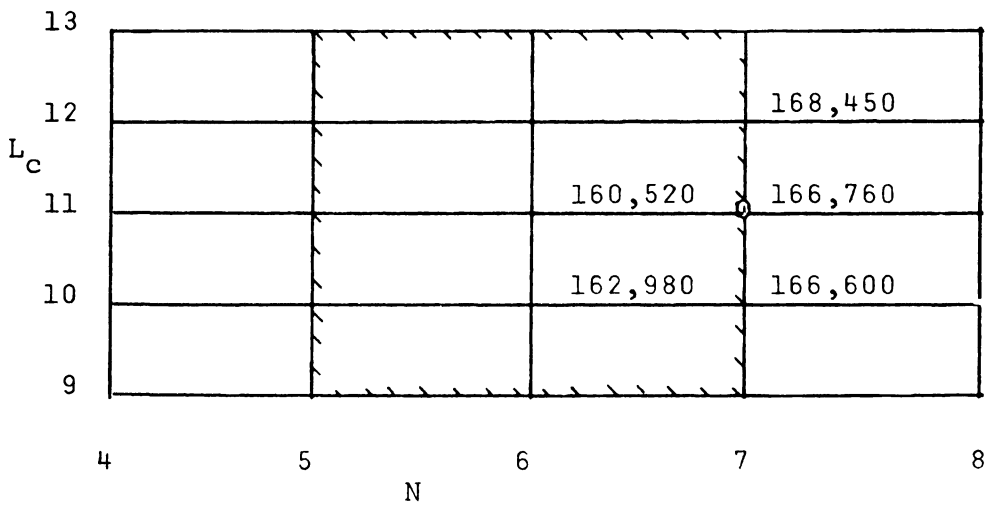
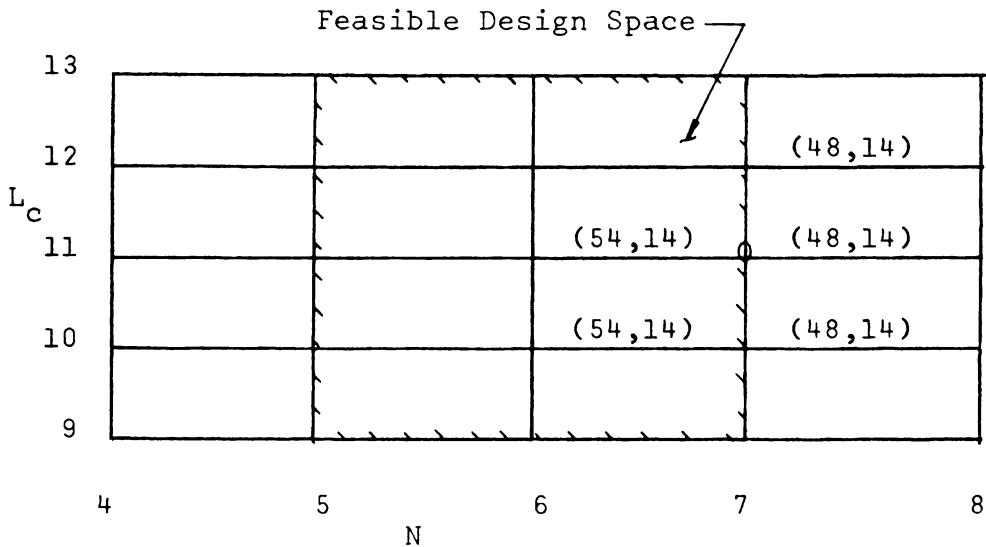




Table 4-3: Summary of the Result of Example 3  
(using The Method of Bounds)

$$Z^* = \$45,956$$

$$Z_f = \$41,751$$

$$Z_f + \epsilon = \$46,344$$

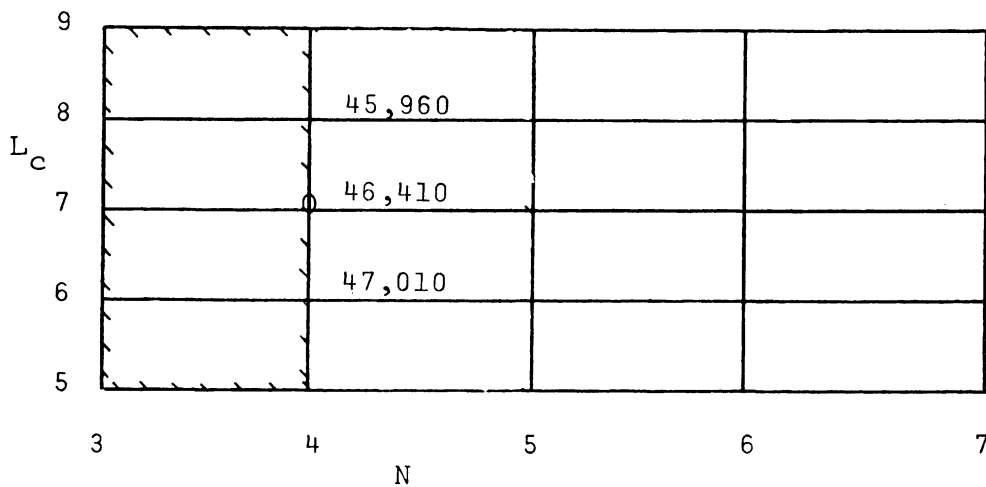
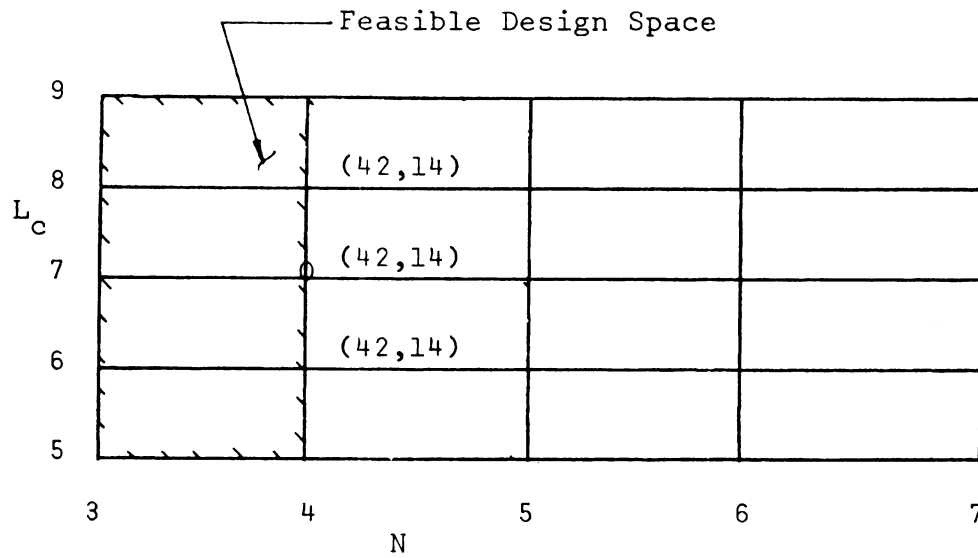




Table 5-1: Comparison of the Results

| Method<br>of<br>Optimization | Independent<br>Design<br>Variables |       |    |       | Cost<br>\$ per<br>sq. ft. | Number<br>of<br>Designs |
|------------------------------|------------------------------------|-------|----|-------|---------------------------|-------------------------|
|                              | N                                  | $L_c$ | h  | $b_f$ |                           |                         |
| <u>Example 1</u>             |                                    |       |    |       |                           |                         |
| Traditional Method           | 6                                  | 7.0   | 48 | 16    | 17.05                     |                         |
| Exhaustive Search            | 6                                  | 13.0  | 66 | 18    | 15.42                     | 150                     |
| Linear Programming           | 6                                  | 11.5  | 54 | 16    | 15.68                     |                         |
| Grid Search Method           | 7                                  | 10.0  | 54 | 14    | 15.66                     | 36                      |
| Method of Bounds             | 7                                  | 10.0  | 54 | 14    | 15.66                     | 3                       |
| <u>Example 2</u>             |                                    |       |    |       |                           |                         |
| Traditional Method           | 6                                  | 9.0   | 48 | 16    | 16.14                     |                         |
| Exhaustive Search            | 6                                  | 13.0  | 54 | 14    | 15.33                     | 150                     |
| Linear Programming           | 6                                  | 10.0  | 54 | 16    | 15.48                     |                         |
| Grid Search Method           | 6                                  | 13.0  | 54 | 14    | 15.33                     | 48                      |
| Method of Bounds             | 6                                  | 11.0  | 54 | 14    | 15.38                     | 23                      |
| <u>Example 3</u>             |                                    |       |    |       |                           |                         |
| Traditional Method           | 4                                  | 7.5   | 42 | 14    | 11.87                     |                         |
| Exhaustive Search            | 4                                  | 8.0   | 42 | 14    | 11.86                     | 250                     |
| Linear Programming           | 3                                  | 7.5   | 48 | 14    | 11.93                     |                         |
| Grid Search Method           | 4                                  | 8.0   | 42 | 14    | 11.86                     | 32                      |
| Method of Bounds             | 4                                  | 8.0   | 42 | 14    | 11.86                     | 9                       |



Table 5-2: Comparison of Computer Time

| Example | Method of Optimization | Total Computer Time in Sec. | Computer System |
|---------|------------------------|-----------------------------|-----------------|
| 1       | Linear Programming     | 501                         | CDC 6500        |
|         | Grid Search Method     | 321                         | B 5500          |
|         | Method of Bounds       | 199                         | B 5500          |
| 2       | Linear Programming     | 502                         | CDC 6500        |
|         | Grid Search Method     | 369                         | B 5500          |
|         | Method of Bounds       | 269                         | B 5500          |
| 3       | Linear Programming     | 321                         | CDC 6500        |
|         | Grid Search Method     | 305                         | B 5500          |
|         | Method of Bounds       | 213                         | B 5500          |





## LIST OF REFERENCES



## LIST OF REFERENCES

1. Clark, J. W., and Hill, H. N., "Lateral Buckling of Beams," Journal of the Structural Division, ASCE, Vol. 86, No. ST 7, July 1960.
2. Douty, R. T., "Optimization of a Two-Span Cover-Plated Steel Beam," Computers in Engineering Design Education-Civil Engineering, Vol. III, University of Michigan, College of Engineering, April, 1966.
3. Gass, S. I., Linear Programming 3rd edition, McGraw-Hill Book Company, Inc., New York, 1969, pp. 49-88.
4. Goble, G. G., and DeSantis, P.V., "Optimum Design of Mixed Steel Composite Girders," Journal of the Structural Division, ASCE, Vol. 92, NO. ST 6, December, 1966.
5. Hiller, F. S., and Lieberman, G. J., Introduction to Operations Research, Holden-Day, Inc., San Francisco, 1968, pp. 565-571.
6. Kelley, J. E., "The Cutting-Plane Method for Solving Convex Problems," Society of Industrial and Applied Mechanics Journal, Vol. 8, No. 4, December, 1960, pp. 703-711.
7. Land, A. H., and Doig, A. G., "An Automatic Method of Solving Discrete Programming Problems," Econometrica, Vol. 28, 1960, pp. 497-520.
8. Moses, F., "Optimum Structural Design Using Linear Programming," Journal of the Structural Division, ASCE, Vol. 90, No. ST 6, Proc. Paper 4163, December, 1964, pp. 89-104.
9. United State Steel (USS), "Composite: Welded Plate Girder," Highway Structures Design Handbook, United States Steel Corporation, Pittsburgh, Pa., 1965, pp. 4.1 - 4.68.



APPENDIX I

A COMPUTER PROGRAM - BRIDGE

-USER'S MANUAL-

By

Kulkarni Sudhakar R.

Department of Civil Engineering

Michigan State University

July, 1973



## TABLE OF CONTENTS

|                                    | <u>Page</u> |
|------------------------------------|-------------|
| 1. Introduction . . . . .          | 72          |
| 2. Scope and Limitations . . . . . | 72          |
| 3. User's Card Deck . . . . .      | 72          |
| 4. Data Card Deck . . . . .        | 73          |
| 5. Output . . . . .                | 75          |
| 6. Example . . . . .               | 76          |

## FIGURE

|                                 |    |
|---------------------------------|----|
| 1-A Example Structure . . . . . | 84 |
|---------------------------------|----|





## 1. Introduction

"BRIDGE" is a computer program for a nearly optimal design of two-span cantilever bridges. The program is written in FOTRAN IV, and it is available on the B 5500 computer system of the Michigan Department of State Highways. To use the program one needs the usual peripheral cards for access to the computer and data cards for his program.

In the following paragraphs, the scope and limitation of the program, the makeup of the user's card, the data card deck, and the description of the output are given. Next, an example is given.

The program is written by Kulkarni, S. R. The subroutines which are required to design the welded plate girder are provided by the computer programming section of the design division of the Michigan Department of State Highways.

## 2. Scope and Limitation

The program is intended to determine a near minimum-cost design of the two-span cantilever bridge. The type of bridge is reinforced concrete slab on welded steel plate girders with shear connectors.

## 3. User's Card Deck

A set of these cards can be obtained from the computer programming section. Usually, the first card of the set shows the account number of the program. The second card shows the computer time requested, and the remaining cards are the system cards for the access to computer. For data deck, see next section.



4. Data Card Deck

| <u>Card</u><br><u>No.</u> | <u>Data</u> | <u>Col. No.</u> | <u>Specification</u> | <u>Remarks</u>    |
|---------------------------|-------------|-----------------|----------------------|-------------------|
| 1                         | SQUAD, SQD  | 11 - 18         | 2A4                  | Identification    |
|                           | MONTH       | 21 - 22         | I2                   | Date              |
|                           | IDAY        | 24 - 25         | I2                   |                   |
|                           | IYEAR       | 27 - 28         | I2                   |                   |
| 2                         | BRL         | 11 - 20         | F10.2                | Bridge            |
|                           | BRW         | 21 - 30         | F10.2                | dimensions        |
|                           | SPANL1      | 31 - 40         | F10.2                |                   |
|                           | SPANLB      | 41 - 50         | F10.2                |                   |
|                           | RDW         | 51 - 60         | F10.2                |                   |
| 3                         | TUTIL       | 11 - 20         | F10.2                | Loading           |
|                           | TSWLL       | 21 - 30         | F10.2                |                   |
|                           | RAILWT      | 31 - 40         | F10.2                |                   |
|                           | LOAD        | 41 - 43         | I3                   |                   |
| 4                         | STRESA      | 11 - 20         | F10.2                | Material          |
|                           | FC          | 21 - 30         | F10.2                | properties        |
|                           | RMOD        | 31 - 40         | F10.2                |                   |
|                           | FR          | 41 - 50         | F10.2                |                   |
| 5                         | C1          | 11 - 20         | F10.2                | Cost data         |
|                           | C2          | 21 - 30         | F10.2                |                   |
|                           | C3          | 31 - 40         | F10.2                |                   |
| 6                         | UBWEBD      | 11 - 20         | F10.2                | Bound of variable |



| <u>Program</u> | <u>Problem</u> | <u>Explanation</u>                                     |
|----------------|----------------|--------------------------------------------------------|
| <u>Symbol</u>  | <u>Symbol</u>  |                                                        |
| BRL            | $B_L$          | Length of bridge (ft.),                                |
| BRW            | $B_W$          | width of bridge (ft.),                                 |
| C1             |                | cost of concrete (\$/cu. yd.),                         |
| C2             |                | cost of reinforcement (\$/lb.),                        |
| C3             |                | cost of structural steel (\$/lb.),                     |
| FC             | $f_c$          | allowable compressive stress of<br>concrete (psi.),    |
| FR             | $f_s$          | allowable tensile stress of rein-<br>forcement (psi.), |
| LOAD           |                | truck loading, 1 = HS, and 2 =<br>H loading,           |
| RAILWT         |                | total weight of bridge railings (lb.),                 |
| RMOD           | $n$            | modular ratio of concrete, usually<br>10,              |
| SPANL1         | $L_1$          | length of anchor span (ft.),                           |
| SPANB          | $L_B$          | length of span (ft.),                                  |
| STRESA         | $F_b$          | allowable stress in bending for<br>steel (psi.),       |
| RDW            | $R_D$          | clear roadway width (ft.),                             |
| TUTIL          |                | total utility (oil or water pipes)<br>load (lbs./ft.), |
| TSWLL          |                | total sidewalk live load (lbs./ft.),                   |
| UBWEBD         |                | upper bound of the depth of web<br>plate (in.).        |

## 5. Output

On the output sheets the following information is printed.

1. The input verification for the data of the problem as shown on page 78.
2. The details of the minimum-cost design of the bridge as shown on page 79, and
3. The details of the design of the two-span cantilever girder as shown on pages 80 through 83.

The input information as described in Section 4--Data Card Deck--is printed on the first sheet of the output.

The details of a near minimum-cost design of the two-span cantilever bridge as shown on page 79 consists of the following. The bridge data, load data, cost data, the values of the independent design variables which minimize the cost of the bridge, slab design, girder design and cost analysis. The average weight of girder in both spans and corresponding factor of safety against lateral buckling are given under girder design heading.

The details of the design of the two-span cantilever girder consists of the details of the design of girder in suspended span  $L_2$ , which are shown on pages 80 and 81, and the details of the design of girder in anchor span  $L_1$ , which are shown on pages 82 and 83.

The explanation of the details of the girder design is as follows. First, the average weight of girder is given. Then, the section properties (moment of inertia, section





modulus) are given for non-composite, and composite sections of the girder. Note that  $N$  denotes the modular ratio of concrete. The length (ft.) and thickness (in.) of flange plates are listed, and the maximum bending stresses in top and bottom flanges are listed next. The moments (ft.-kips) due to dead load, live load and sidewalk plus future wearing surface loads are given. At the end of the page, the maximum reactions (kips) at supports are listed and the design of the bearing stiffeners is given.

The details of the girder design of the suspended span are continued as shown on page 81. The deflections of the girder due to dead loads and ratio of live load deflection to span length at mid span are given. Usually the girders are designed without using the transverse intermediate stiffeners, and this is indicated by the statement--"none required between brgs." on the output sheet. The actual factor of safety against lateral buckling is given. The information - Input Verification - lists the data used by the subroutine GIRDER to carry out the design of the girder. The thickness and depth of web plate and width of flange plate are given here.

The details of the design of girder in the anchor span are described similarly and are shown on pages 82 and 83.

## 6. Example

Figure 1-A shows the two-span cantilever bridge. The data for the design is given as follows.



Bridge Dimensions:  $B_L = 236.00$  ft.,  $B_W = 50.00$  ft.,  
 $L_1 = 116.75$  ft.,  $L_B = 116.75$  ft., and  $R_D = 46.50$  ft.

Loading: Total utility load = 0., total sidewalk live load = 0., weight of railings = 954 lbs./ft., live load = HS 20 truck loading.

Material Properties:  $F_b$  (A 36 structural steel) = 20,000 psi.,  $f_c = 3,000$  psi.,  $n = 10$ ,  $f_s$  (reinforcement) = 20,000 psi.

Cost Data: The unit costs of concrete, reinforcement and structural steel are \$120/cu. yd., \$0.22/lb., and \$0.30/lb., respectively. The upper bound of the depth of web plate is 54 inches.

## INPUT VERIFICATION

SQUAD        = KULKARNI      DATE        = 27- 3-73  
 BRIDGE L    =    236.00    BRIDGE W=        50.00  
 SPANL1      =    116.75    SPANB        =    116.75  
 CL ROAD W=    46.50    TOTAL UTIL=        0.00  
 TOTAL SWLL =        0.00    RAILING WT=    954.00  
  
 TRUCK LOAD= 1            STRESS        = 20000.00  
 FC IN CONC =    1200.00    MOD. RATIO=        10.00  
 FS IN REINF= 20000.00

## COST DATA

CONCRETE    = 120.00 DOLLARS PER CU.YD.  
 REINF,STEEL=    0.22 DOLLARS PER LB.  
 STR. STEEL =    0.30 DOLLARS PER LB.  
  
 UPPER BND. WEB DEPTH =        54.00

2000

1999

1998

1997

1996

1995

1994

1993

1992

1991

1990

1989

1988

1987

## TWO-SPAN CANTILEVER BRIDGE DESIGN

SQUAD KULKARNI

## BRIDGE DIMENSIONS

## MATERIAL PROPERTIES

|               |   |        |     |            |   |        |      |
|---------------|---|--------|-----|------------|---|--------|------|
| BRIDGE LENGTH | = | 236.00 | FT. | FS(REINF.) | = | 20000. | PSI. |
| BRIDGE WIDTH  | = | 50.00  | FT. | FC(CONC.)  | = | 1200.  | PSI. |
| SPAN 1 LENGTH | = | 116.75 | FT. | FS(STEEL)  | = | 20000. | PSI. |
| SPAN 2 LENGTH | = | 116.75 | FT. | MOD. RATIO | = | 10.    |      |
| ROADWAY WIDTH | = | 46.50  | FT. |            |   |        |      |

## LOAD DATA

|                |   |        |             |
|----------------|---|--------|-------------|
| LOAD(1=HS 2=H) | = | 1      |             |
| RAILING WT.    | = | 954.00 | LBS./FT.    |
| SDWALK L. LOAD | = | 0.00   | LBS./FT.    |
| FUT. WEARING   | = | 25     | LBS./SQ.FT. |
| TOTAL UTILITY  | = | 0.00   | LBS./FT.    |

## COST DATA

|                |   |        |           |
|----------------|---|--------|-----------|
| COST OF CONC.  | = | 120.00 | \$/CU.YD. |
| COST OF REINF. | = | 0.22   | \$/LB.    |
| COST OF STEEL  | = | 0.30   | \$/LB.    |

## DESIGN VARIABLES

|                 |   |       |     |
|-----------------|---|-------|-----|
| GIRDER SPACING  | = | 90.00 | IN. |
| NO. OF GIRDERS  | = | 7.    |     |
| DEPTH OF WEB    | = | 54.00 | IN. |
| UPPER BND. WEB  | = | 54.00 | IN. |
| CANT. LENGTH    | = | 10.00 | FT. |
| FLANGE PL WIDTH | = | 14.00 | IN. |

## SLAB DESIGN

|               |   |      |            |
|---------------|---|------|------------|
| SLAB THICK.   | = | 8.00 | IN.        |
| TRANS. REINF. | = | 0.75 | SQ.IN./FT. |

## GIRDER DESIGN

|               |   |      |          |
|---------------|---|------|----------|
| AVEWT. SPAN 1 | = | 231. | LBS./FT. |
| AVEWT. SPAN 2 | = | 222. | LBS./FT. |
| SAFTY SPAN 1  | = | 1.30 |          |
| SAFTY SPAN 2  | = | 1.61 |          |

|                  |   |           |           |
|------------------|---|-----------|-----------|
| COST OF SLAB     | = | 35982.72  | \$        |
| COST OF REINF.   | = | 17667.22  | \$        |
| COST OF STEEL    | = | 131189.58 | \$        |
| MIN. BRIDGE COST | = | 184839.52 | \$        |
| BRIDGE COST      | = | 15.66     | \$/SQ.FT. |



## P L A T E   G I R D E R

SQUAD

KULKARNI

PROB. NO.    1.

AVG. WT./FT. =    222.

| G I R D E R |        |        | COMP. SECT. N=8 |         |
|-------------|--------|--------|-----------------|---------|
| CENTER      | I      | S(TOP) | S(BOTT)         | I       |
|             | 24317. | 774.   | 999.            | 69707.  |
|             | 31796. | 854.   | 1620.           | 103273. |
|             | 40710. | 1190.  | 1757.           | 104170. |
|             | 31796. | 854.   | 1620.           | 103273. |
|             | 24317. | 774.   | 999.            | 69707.  |

| C O M P O S I T E |        |        | S E C T I O N |        |        |         |
|-------------------|--------|--------|---------------|--------|--------|---------|
| N=10              |        |        | N=30          |        |        |         |
| CENTER            | I      | S(TOP) | S(BOTT)       | I      | S(TOP) | S(BOTT) |
|                   | 65944. | 6840.  | 1430.         | 47164. | 2430.  | 1298.   |
|                   | 96455. | 6514.  | 2293.         | 64949. | 2526.  | 2084.   |
|                   | 97761. | 6651.  | 2291.         | 69140. | 2826.  | 2101.   |
|                   | 96455. | 6514.  | 2293.         | 64949. | 2526.  | 2084.   |
|                   | 65944. | 6840.  | 1430.         | 47164. | 2430.  | 1298.   |

|                          | LENGTH | THICK. |
|--------------------------|--------|--------|
| T O P    P L A T E       | 27.6   | 0.625  |
|                          | 51.0   | 1.125  |
|                          | 28.2   | 0.625  |
| B O T T O M    P L A T E | 18.6   | 1.125  |
|                          | 70.0   | 2.250  |
|                          | 18.2   | 1.125  |

| S T R E S S E S |      | LT BRG | 0.50L  | RT BRG |
|-----------------|------|--------|--------|--------|
|                 |      |        |        |        |
|                 | TOP  | 0.     | 19422. | 0.     |
|                 | BOTT | 0.     | 19841. | 0.     |

| M O M E N T S | DEAD LOAD | 0. | 1501. | 0. |
|---------------|-----------|----|-------|----|
|               | LIVE LOAD | 0. | 1361. | 0. |
|               | SW + FWS  | 0. | 431.  | 0. |

| R E A C T I O N S<br>(MAX) | DEAD LOAD    | 70.5         | 70.5         |
|----------------------------|--------------|--------------|--------------|
|                            | L. L. W/IMP  | 54.5         | 54.5         |
|                            | L. L. WO/IMP | 44.8         | 44.8         |
|                            | BRG STIFF    | 0.625 X 6.50 | 0.625 X 6.50 |



## P L A T E   G I R D E R

SQUAD

KULKARNI

PROR. NO.    1.

|                 |                 | LT PIN | CL     | RT PIN |
|-----------------|-----------------|--------|--------|--------|
|                 | BEAM            | 0.00   | 0.71   | 0.00   |
|                 | SLAB ADJ. SPANS | 0.00   | 0.00   | 0.00   |
| D E F L E C T . | SLAB THIS SPAN  | 0.00   | 1.92   | 0.00   |
|                 | SIDEWALK        | 0.00   | 0.23   | 0.00   |
|                 | DELTA LL/L      |        | 1/1504 |        |

S T I F E .    S P A .                    NONE REQD BETWEEN BRGS

FACTOR OF SAFETY AGAINST LATERAL BUCKLING = 1.6

## I N P U T            V E R I F I C A T I O N

|               |        |                |      |
|---------------|--------|----------------|------|
| WEB THICK.    | 0.5000 | S.W. LIVE LOAD | 0.   |
| WEB DEPTH     | 54.00  | S.W. + F.W.S.  | 302. |
| FLANGE WIDTH  | 14.00  | CANT. LENGTH A | 0.00 |
| SLAB THICK.   | 8.00   | P (BEAM)       | 0.0  |
| GIRDER SPA.   | 90.00  | P (SLAB)       | 0.0  |
| SPAN LENGTH   | 106.75 | P (LIVE LOAD)  | 0.0  |
| LIVE LOAD     | HS 20  | P (SW + FWS)   | 0.0  |
| DIST. FACTOR  | 1.10   | CANT. LENGTH B | 0.00 |
| HAUNCH DEPTH  | 1.00   | P (BEAM)       | 0.0  |
| HAUNCH WIDTH  | 20.00  | P (SLAB)       | 0.0  |
| DESIGN STRESS | 20000. | P (LIVE LOAD)  | 0.0  |
| UTILITIES     | 0.     | P (SW + FWS)   | 0.0  |

## P L A T E   G I R D E R

SQUAD

KULKARNI

PROB. NO.   1.

AVG. WT./FT. =   231.

| G I R D E R |        |        | COMP. SECT. |   | N=8     |
|-------------|--------|--------|-------------|---|---------|
| CENTER      | I      | S(TOP) | S(BOTT)     | I |         |
|             | 26212. | 821.   | 1095.       |   | 75762.  |
|             | 33607. | 902.   | 1704.       |   | 108229. |
|             | 42594. | 1236.  | 1849.       |   | 109241. |
|             | 33607. | 902.   | 1704.       |   | 108229. |
|             | 26212. | 821.   | 1095.       |   | 75762.  |

| C O M P O S I T E |         |        | S E C T I O N |        |        |         |
|-------------------|---------|--------|---------------|--------|--------|---------|
| N=10              |         |        | N=30          |        |        |         |
| CENTER            | I       | S(TOP) | S(BOTT)       | I      | S(TOP) | S(BOTT) |
|                   | 71435.  | 6666.  | 1582.         | 50428. | 2456.  | 1427.   |
|                   | 100843. | 6463.  | 2436.         | 67450. | 2561.  | 2200.   |
|                   | 102292. | 6607.  | 2434.         | 71861. | 2859.  | 2220.   |
|                   | 100843. | 6463.  | 2436.         | 67450. | 2561.  | 2200.   |
|                   | 71435.  | 6666.  | 1582.         | 50428. | 2456.  | 1427.   |

|                         | LENGTH | THICK. |
|-------------------------|--------|--------|
| T O P   P L A T E       | 28.9   | 0.625  |
|                         | 50.0   | 1.125  |
|                         | 47.8   | 0.625  |
| B O T T O M   P L A T E | 20.9   | 1.250  |
|                         | 69.0   | 2.375  |
|                         | 36.8   | 1.250  |

|                 |      | LT BRG | 0.48L  | RT BRG |
|-----------------|------|--------|--------|--------|
| S T R E S S E S | TOP  | 0.     | 19443. | 19878. |
|                 | BOTT | 0.     | 19692. | 14894. |

|               |           |    |       |       |
|---------------|-----------|----|-------|-------|
| M O M E N T S | DEAD LOAD | 0. | 1536. | -600. |
|               | LIVE LOAD | 0. | 1502. | -582. |
|               | SW + FWS  | 0. | 430.  | -177. |

|                            |              |              |              |
|----------------------------|--------------|--------------|--------------|
| R E A C T I O N S<br>(MAX) | DEAD LOAD    | 71.1         | 169.0        |
|                            | L. L. W/IMP  | 54.5         | 93.4         |
|                            | L. L. WO/IMP | 45.2         | 71.6         |
|                            | BRG STIFF    | 0.625 X 6.25 | 0.875 Y 6.25 |

## P L A T E   G I R D E R

SQUAD

KULKARNI

PROR. NO.    1.

|                 |                 | LT PIN | CL     | RT PIN |
|-----------------|-----------------|--------|--------|--------|
|                 | BEAM            | 0.00   | 0.83   | -0.18  |
|                 | SLAB ADJ. SPANS | 0.00   | -0.57  | 0.34   |
| D E F L E C T . | SLAB THIS SPAN  | 0.00   | 2.60   | -0.77  |
|                 | SIDEWALK        | 0.00   | 0.26   | -0.05  |
|                 | DELTA LL/L      |        | 1/1267 | 1/ 445 |

S T I F E .    S P A .

NONE REQD BETWEEN BRGS

FACTOR OF SAFETY AGAINST LATERAL BUCKLING    =    1.3

## I N P U T        V E R I F I C A T I O N

|               |        |                |       |
|---------------|--------|----------------|-------|
| WEB THICK.    | 0.5625 | S.W. LIVE LOAD | 0.    |
| WEB DEPTH     | 54.00  | S.W. + F.W.S.  | 302.  |
| FLANGE WIDTH  | 14.00  | CANT. LENGTH A | 0.00  |
| SLAB THICK.   | 8.00   | P (BEAM)       | 0.0   |
| GIRDER SPA.   | 90.00  | P (SLAB)       | 0.0   |
| SPAN LENGTH   | 116.75 | P (LIVE LOAD)  | 0.0   |
| LIVE LOAD     | HS 20  | P (SW + FWS)   | 0.0   |
| DIST. FACTOR  | 1.10   | CANT. LENGTH B | 10.00 |
| HAUNCH DEPTH  | 1.00   | P (BEAM)       | 14.0  |
| HAUNCH WIDTH  | 20.00  | P (SLAB)       | 41.2  |
| DESIGN STRESS | 20000. | P (LIVE LOAD)  | 44.8  |
| UTILITIES     | 0.     | P (SW + FWS)   | 16.1  |



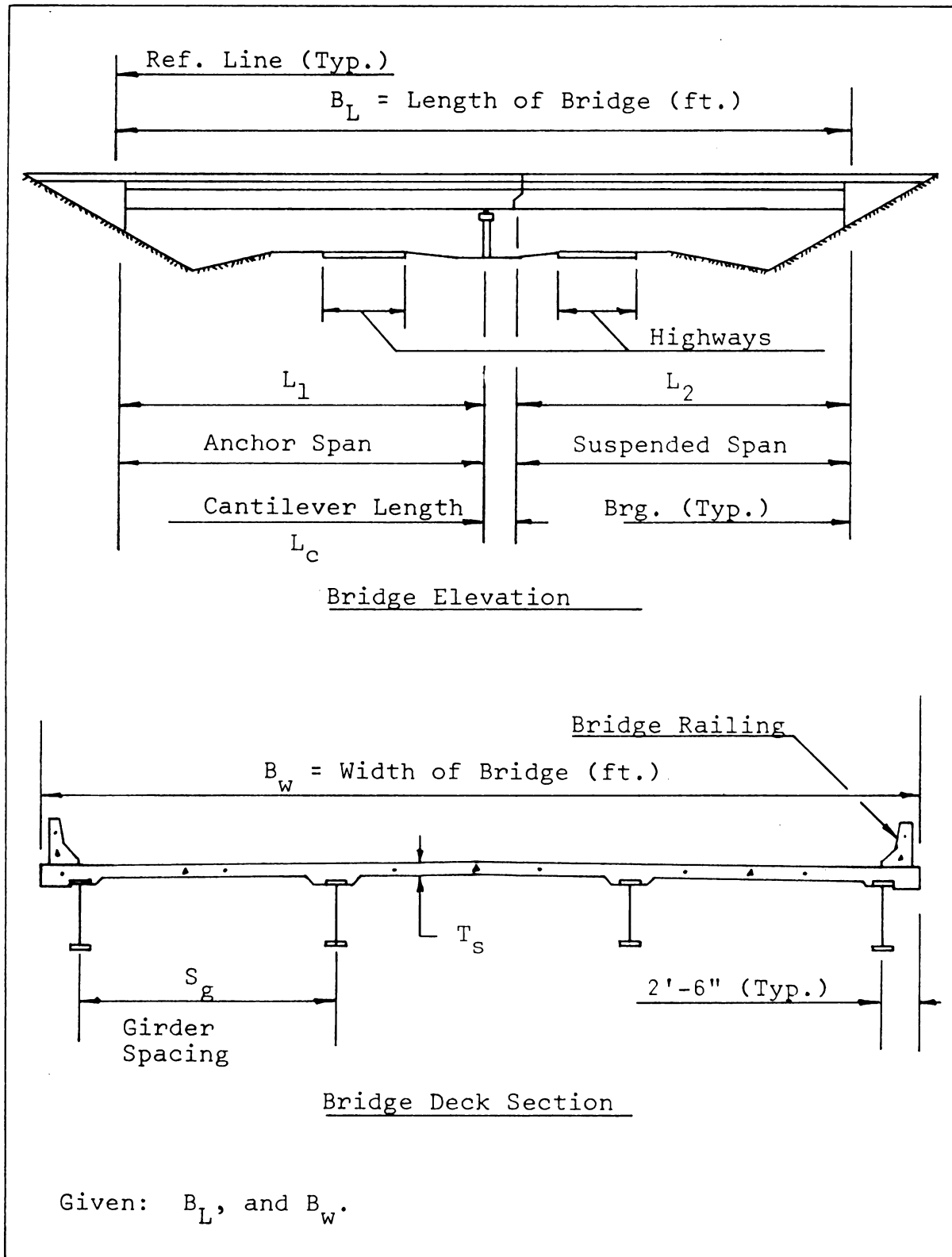
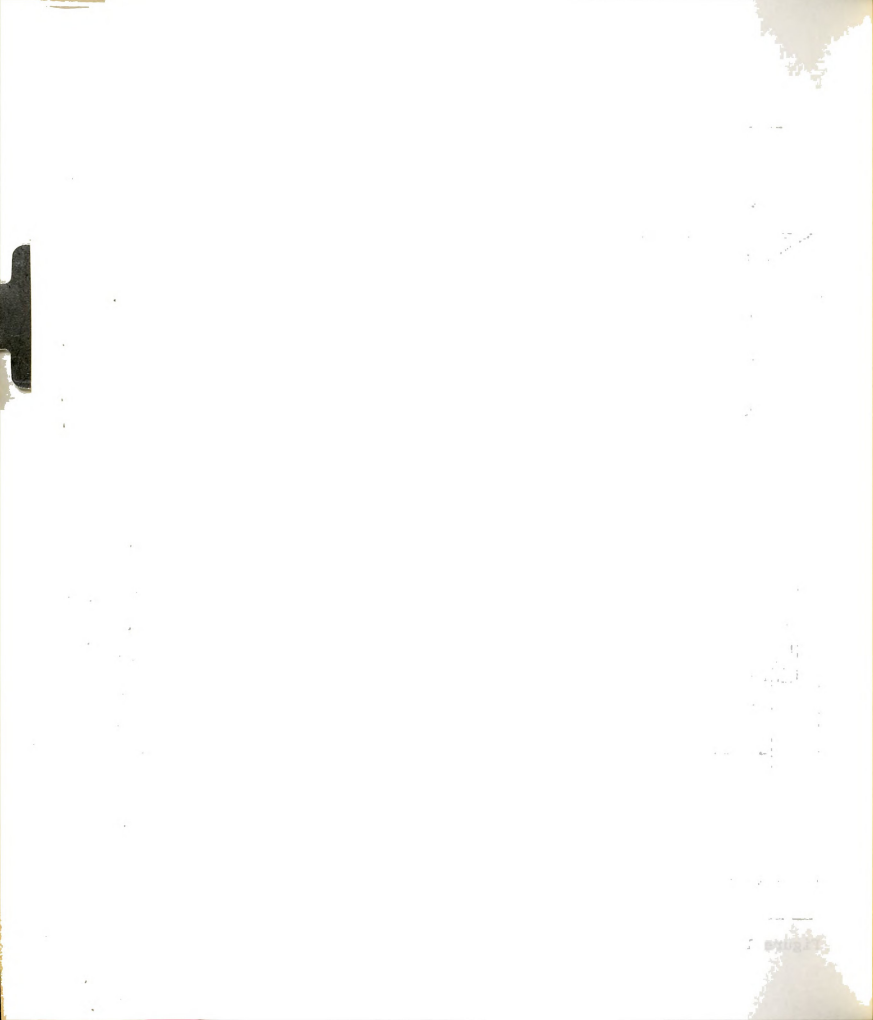


Figure 1-A: Example Structure



## APPENDIX II

### COMPUTER PROGRAM

#### A.2.1 Description of Routines

The computer program BRIDGE consists of the main routine BRIDGE, and the following subroutines: GRID1, GRID2, SLAB, PRELIM, DESGIR, GIRDER, PROPTY, MOMENT, STRESS, DELTA, SHEAR, FINDM, DETAIL, DETGIR, FSDES, MINBM, and MINC. The subroutines GRID1, and GRID2 use the grid search method to determine the values of number of girders, the length of cantilever, the depth of web plate, and the width of flange plates such that the corresponding cost of the bridge is minimized.

The subroutine SLAB determines slab thickness, and transverse reinforcement in slab. The subroutine PRELIM determines utility and sidewalk loads carried by a girder. The subroutine DESGIR determines the data for calling the subroutine GIRDER, and completes a design of the two-span cantilever girder. The subroutine GIRDER designs the welded steel plate girder. The subroutine PROPTY determines the section properties such as moment of inertia and section moduli of a composite and non-composite section. The subroutine MOMENT determines moments due to dead load and live load (HS 20 truck loading) at a section along the length of girder. The subroutine STRESS determines the bending





stresses due to the moments. The subroutine DELTA determines the deflections due to dead load and live load at the mid-span and at the end of cantilever. The SHEAR determines the shear due to dead load and live load at a section along the length of girder. The subroutine FINDM selects the section properties to be used to determine the spacing shear connectors along the length of girder. The subroutine DETAIL is used to write the details of a bridge design. The subroutine DETGIR is used to write the details of a girder design.

The subroutine FSDES, MINBM, and MINC are used to determine an "effective lower bound" of the cost of the bridge. The subroutine FSDES determines the value of number of girder such that an "effective lower bound" is minimized. The subroutine MINBM determines the length of cantilever which minimizes an absolute sum of area under a bending moment diagram of the two-span cantilever girder. The subroutine MINC determines the cost of a "fully-stressed" design of the bridge.

#### A.2.2 List of Identifiers

The important identifiers, used in the program BRIDGE, are identified below in alphabetical order.

ALLOWC = Allowable live load deflection ratio at mid-span

ALLOWE = Allowable live load deflection ratio at the end of  
cantilever

AVEWT = Average weight of girder (lb./ft.)

BCOST = Total cost of the bridge (\$)



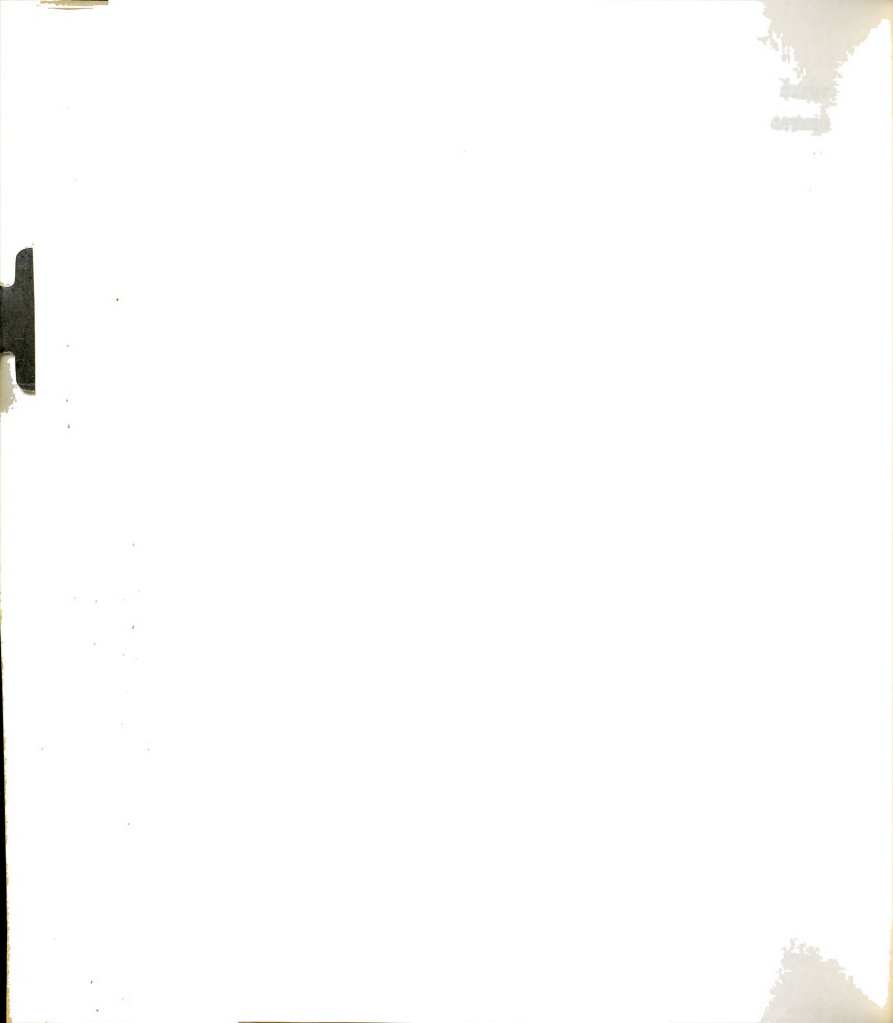
BMDL = Bending moment due to dead load at a section X  
 along the length of girder (ft.-kips)  
 BMSW = Bending moment due to sidewalk dead load  
 (ft.-kips)  
 BMSWLL = Bending moment due to sidewalk live load  
 (ft.-kips)  
 BOTPL (M)= Length of bottom flange plate (ft.), M=location  
 BOTTP(T)= Thickness of bottom flange plate (in.)  
 BRL = The length of the bridge (ft.)  
 BRW = The width of the bridge (ft.)  
 CANTB = The length of cantilever (ft.)  
 CG (M,K) = Center of gravity of section, M=location,  
 K=Modular ratio  
 C1 = Cost of concrete (\$/cu.yd.)  
 C2 = Cost of reinforcement (\$/lb.)  
 C3 = Cost of structural steel (\$/lb.)  
 DELT(J,K)= Dead load deflections (in.)  
 DISTLL = Live load distribution factor  
 FATIGC = Allowable fatigue stress for butt weld (psi.)  
 FATIGT = Allowable fatigue stress for butt weld (psi.)  
 FC = Allowable compressive stress in extreme fiber of  
 concrete (psi.)  
 FS(M) = Actual bending stress (psi.)  
 FSAFTY = Factor of safety against lateral buckling  
 FR = Allowable tensile stress in reinforcement (psi.)  
 GRNO = Number of girders  
 GSPA = Girder spacing (in.)  
 HDEPTH = Haunch depth (in.)



HWIDTH = Haunch width (in.)  
 I(M,K) = Moment of inertia (in.<sup>4</sup>)  
 LOAD = 1 for HS loading, and 2 for H loading  
 NEGLL = Negative live load moment at point X (ft.-kips)  
 PBMB = Cantilever load (kips)  
 PSLB = Cantilever load (kips)  
 PSWB = Cantilever load (kips)  
 PLLB = Cantilever load (kips)  
 PLATEW = Width of flange plate (in.)  
 POSLL = Positive live load moment at point X (ft.-kips)  
 RAILWT = Total railing weight (lbs./ft.)  
 RDW = The width of roadway (ft.)  
 REINF(2) = Transverse reinforcement (in.<sup>2</sup>/ft.)  
 RMOD = Modular ratio of concrete  
 SB(M,K) = Section modulus (in.<sup>3</sup>)  
 SLABT = Slab thickness (in.)  
 SLABW = Effective width of slab (in.)  
 SPANL1 = Length of anchor span (ft.)  
 SPANL2 = Length of span 2 - suspended span (ft.)  
 SPANB = Length of span B = SPANL2 + CANTB  
 ST(M,K) = Section modulus of top flange (in.<sup>3</sup>)  
 STRESA = Allowable stress in bending for steel (psi.)  
 STUD = Stud spacing (in.)  
 STUDS = Number of studs/row  
 SWFWS = Sidewalk + future wearing course (lb./ft.)  
 SWLL = Sidewalk live load (lb./ft.)  
 TOPPL(M) = Length of top flange plate (ft.)  
 TOPPT(M) = Thickness of top flange plate (in.)



TUTIL = Total utility load (lbs./ft.)  
UBWEBD = Upper bound of the depth of web plate (in.)  
UTIL = Utility load (lbs./ft. of girder)  
VDL = Shear due to dead load at point X (kips)  
VSW = Shear due to sidewalk dead load (kips)  
VLL = Shear due to live load at point X (kips)  
WEBD = The depth of web plate (in.)  
WEBT = The thickness of web plate (in.)  
WTFTSL = Slab weight (lbs./ft. of girder)  
X = Distance from left support (ft.)  
ZMINC = Value of an "effective lower bound" of the cost  
of the bridge (\$).





C A.2.3 LISTING OF PROGRAM

C PROGRAM BRIDGE

C MINIMUM-COST DESIGN OF A TWO-SPAN CANTILEVER BRIDGE

C  
C  
C  
C  
C

```

COMMON BMDC,BMLL,BMSW,BMSWL,BMTRK,BOTTP(6),CANTA,CANIB,CG(6,4),
A DELTA,DELTAH,DELTAH,DELTA(3,4),DISTRB,EMPACT,EXTRA,PATIG(2),
B FS(4),GSPA,HUEPTH,HWIDTH,I(6,4),LCAD,M,NEGLL,PBMA,FOMB,PLLA,
C PLLB,PLATET(2),PLATEW,PUSLL,PSLA,PSLB,SLABW,SB(6,4),SLABI,SPANL
D ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TUPPT(6),UTIL,VDL,
E VLL,VLLMIN,VSW,VTOTAL,WEDC,WERT,WTF TG(5),WTFISL,X,FSPA,PSWB
F ,XL,XR,LL,FSAFETY,AVGWT,CCELL
COMMON/DES1/ALLOWC(4),ALLOWE(4),MUT(12),PROBLL(4),TRATIO(2)
COMMON/DES2/ADU(4),DLRT(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),
A 1STIFF(2),TWER(9),LLRWUI(2)
COMMON/GRDATA/BRL,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMOD,FR,C1,C2,
B C3,GRND,GCCST,GCCSTI,WG(2),OPTWEH(10),OPTSPA(10),OPTPL(10),
C OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFETY,WEDU1,
D UBWEDD,TPT(1),BPT(1),IDEL
COMMON/DET/TOPPL1,TOPPL2,TOPPL3,BOTPL1,BOTPL2,BOTPL3,XCENT,
A FTOPL,FSTOP,FTUPR,FBOTTL,FSBOTI,FBOTIR,BMDLA,BMDLC,BMDLB,BMLLA,
B BMLLC,BMLLB,BMSWA,BMSWC,BMSWB,WSTIFF,STUD,STIFA,PI,STII,STIW,
C STIFR,STIFB,STIFL
COMMON/GRID/UBGRND,GRNULB,NG,CANTLB,CANTUB,WEDDUB,WEDULB,ND,
A PLWLB,PLWUB,IXG,IYC,IXGC,IYCC,IXH,IYB,IXHC,IYBC,ZGC(5,5),
B CSLAB,CREINF,CSTEEL,BCOST,ZHB(5,5),SAFETY(5,5),PLW(5,5),
C WH(5,5),DPTH(5,5),OPTB(5,5),OPTG(5,5),OPTC(5,5),SAF11(5,5)
COMMON/SRCH/OCCST,BGRND,BCANT,BWEDD,BPLATE,DATUM,UH,UP
COMMON/LBC/CANT,/MINC
1 FORMAT(10X,2A4,2X,2(12,1X),12)
2 FORMAT(10X,5F10.2)
3 FORMAT(10X,3F10.2,13)
4 FORMAT(10X,F10.2)
6 FORMAT(1H1,////////)
A 40X,19HINPUT VERIFICATION,///20X,12H$W$AD ==
B 2A4,4X,11HDATE ==,2(12,1H-),12,/)
7 FORMAT(20X,12HBRIDGE L ==,F10.2,2X,11HBRIDGE W==,F10.2,2,/)
A 20X,12HSPANL1 ==,F10.2,2X,11HSPANB ==,F10.2,2,/)
8 FORMAT(20X,12HCL ROAD W==,F10.2,2X,11HTOTAL UTIL==,F10.2,2,20X,
A 12HTOTAL SWLL ==,F10.2,2X,11HRAILING WT==,F10.2,2,/)
9 FORMAT(20X,12HTRUCK LOAD==,13,9X,11HSTRESS ==,F10.2,2,/)
A 20X,12HFC IN CONC ==,F10.2,2X,11HMOD. RATIO==,F10.2,2,/)
10 FORMAT(20X,12HFS IN REINF==,F10.2,2,40X,10HCCST DATA,/)
11 FORMAT(20X,12HCONCRETE ==,F8.2,2X,18HDOLLARS PER CU.YD,/)
A 20X,12HREINF. STEEL==,F8.2,2X,15HDOLLARS PER LB,/)
B 20X,12HSTR. STEEL ==,F8.2,2X,15HDOLLARS PER LB,/)
12 FORMAT(40X,15HINITIAL DESIGN,///20X,13HGIRDER SP ==,F10.2,8X,

```

```

A 14HNU. OF GIRDERS=,F4.0,8X,11HWEB DEPTH=,F10.2,
B 14HCANT. LENGTH =,F4.0,/)

```

```

14 FORMAT(20X,13HFLANGE WIDTH=,F10.2,2X,6HWEB DI=,F10.2,/)
15 FORMAT(20X,32HCOST OF THE INITIAL DESIGN =,F10.2,7HDOLLARS,)
16 FORMAT(20X,26HCPT BCOST AT INITIAL GSPA=,F10.2,7HDOLLAR,/)
20 FORMAT(20X,11HNU. OF ITER=,I4,2X,5HGRND=,F4.0,2X,5HGSFA=,F4.0,/)
21 FORMAT(20X,32HCOST OF THE CURRENT DESIGN =,F10.2,7HDOLLARS,/)
22 FORMAT(20X,15HOPTIMUM DESIGN,/,20X,5HGSPA=,F5.1,2X,5HWEBD=,F5.1,
A 2X,7HPLATEW=,F5.1,2X,6HCANTB=,F5.1,/)
23 FORMAT(20X,14HBRIDGE COST =,F6.2,2X,18HDOLLARS PER SQ.FT.)
24 FORMAT(20X,17HMIN. BRIDGE COST=,F10.2,2X,7HDOLLARS,/)

25 FORMAT(20X,22HUPPER BND. WEB DEPTH =,F10.2,/)
30 FORMAT(20X,4HMAIN,2X,5HGCSI,2X,2F10.2)
31 FORMAT(20X,14HCOST OF SLAB =,F11.2,2X,7HDOLLARS,/
A 20X,14HCOST OF REINF=,F11.2,2X,7HDOLLARS)

33 FORMAT(20X,14HCOST OF STEEL=,F11.2,2X,7HDOLLARS)

40 FORMAT(20X,9HBND GRND, 2F10.2,14)
41 FORMAT(20X,10HBND CANTB, 3F10.2)
42 FORMAT(20X,9HBND WEBD, 2F10.2, 14)
43 FORMAT(20X,11HBND PLATEW, 2F10.2)
45 FORMAT(20X,32HCOORDINATES OF MIN. COST DESIGN,/,20X,2(14,2X))
46 FORMAT(20X,36HDESIGN VARIABLES FOR MIN.COST DESIGN,/,20X,5F10.2)
58 FORMAT(33X,4HTIME,2X,2F7.2)

```

C  
C  
C  
C

```

      TO READ DESIGN DATA SUCH AS BRIDGE DIMENSIONS, MATERIAL
      PROPERTIES, HIGHWAY LOADING AND COST DATA

```

```

170 READ(1,1,END = 180) SQUAD,SQU,MONTH,ICAY,IYEAR
      READ(1,2) BRL,BRW,SPANL1,SPANB,RDW
      READ(1,3) TUTIL,TSWLL,RAILWT,LOAD
      READ(1,2) STRESA,FC,RMODU,FR
      READ(1,2) C1,C2,C3
      READ(1,4) UBWEBD

```

C  
C  
C

```

      INPUT VERIFICATION

```

```

      WRITE(3,6) SQUAD,SQU,MONTH,ICAY,IYEAR
      WRITE(3,7) BRL,BRW,SPANL1,SPANB
      WRITE(3,8) RDW,TUTIL,TSWLL,RAILWT
      WRITE(3,9) LOAD,STRESA,FC,RMODU
      WRITE(3,10) FR
      WRITE(3,11) C1,C2,C3
      WRITE(3,25) UBWEBD

```

C  
C

```

      TO DETERMINE THE LIMITS OF THE DESIGN SPACE
      NUMBER OF GIRDER CHOICES AND CANTILEVER LENGTHS

```

```

IDET= 1
GRNU = 20.
DO 100 K = 1,17
GRNU = GRNU - 1.0
GSPA = ( BRN - 2.*2.5)*12./(GRNU - 1.)
IF(GSPA .GE.78.) GO TO 110
100 CONTINUE
110 UGRNU = GRNU
DO 120 K = 1,4
GRNU = GRNU - 1.
GSPA = ( BRN - 2.*2.5)*12./(GRNU - 1.)
IF(GRNU .EQ.3.) GO TO 125
IF(GSPA .GT. 144.) GO TO 130
120 CONTINUE
125 GRNULB = GRNU
NG = IFIX(UGRNU-GRNULB)+1
GO TO 135
130 GRNULB = GRNU + 1.
NG = IFIX(UGRNU-GRNULB)+1
135 CANTB = FLOAT(IFIX(SPANL1/10.))
CANTUB= FLOAT(IFIX(SPANL1/10.)) + 2.0
CANTLB= FLOAT(IFIX(SPANL1/10.)) -2.0
C    TO DETERMINE UPPER AND LOWER LIMITS OF DESIGN SPACE
C    FOR WEB PLATE DEPTH AND FLANGE PLATE WIDTH
WEBD = 36.0
DO 140 K = 1,10
WEBD = WEBD + 6.0
IF((SPANL1*12./WEBD).LE. 30.) GO TO 145
140 CONTINUE
145 WEBDLB = WEBD
WEBDUB = WEBDLB + 24.0
ND = 4 + 1
IF(UBWEBD .EQ.150.) GO TO 150
ND = IFIX((UBWEBD - WEBDLB)/6.0) + 1
150 PLWLB = 14.0
PLWUB = 22.0

WRITE(3,40)GRNULB,UGRNU,NG
WRITE(3,41) CANTLB,CANTB,CANTUB
WRITE(3,42) WEBDLB,WEBDUB,ND
WRITE(3,43) PLWLB,PLWUB
C    NOW THE LIMITS OF THE DESIGN SPACE ARE DETERMINED
C    START THE SEARCH FOR THE MINIMUM-COST DESIGN BY
C    USING THE GRID SEARCH TECHNIQUE
CALL FSUES
CALL GRID1
C    CO-ORDINATES OF THE MINIMUM-COST DESIGN AND COST
WRITE(3,45) IXGC,IYCC

```



```

GRNU = JBGRNU - (FLOAT(IXGC - 1))*1.
WRITE(3,50) GRNU
50 FORMAT(20X,4HGRNU, F6.0)
GSPA = (BRW - 2.*2.5)*12./(GRNU - 1.)
CANTB = FLOAT(IFIX(SPANL1/10.)) - 2.0 + (FLOAT(IYCC-1))*1.
WEBD = UPTH(IXGC,IYCC)
PLATEW = OPTB(IXGC,IYCC)
WRITE(3,46) GRNU,GSPA,CANTB,WEBD,PLATEW
C TO OBTAIN THE FINAL DESIGN
WRITE(3,22) GSPA,WEBD,PLATEW,CANTB
CALL SLAB
CALL PRELIM
IDET = 1
CALL DESIGN
CSLAB = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRNU))
CREINF = C2*490.*BRL*BRW/54.*REINF(2)
CSTEEL = C3*(WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*GRNU*1.18
BCOST = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRNU)) + C2*490.*BRL*BRW/54.*REINF(2)
H + C3*(WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*GRNU*1.18
CALL DETAIL
WRITE(3,31) CSLAB,CREINF
WRITE(3,33) CSTEEL
WRITE(3,24) BCOST
UBCOST = BCOST/(BRL*BRW)
WRITE(3,23) UBCOST
PTIME = TIME(2)/60. - PTIME
OTIME = TIME(3)/60. - OTIME
WRITE(3,58) PTIME,OTIME
GO TO 170
180 STOP
END

SUBROUTINE GRID1
COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BCTTPT(6),CANTA,CANTB,CG(6,4),
A DELTLA,DELTLC,DELTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PMBB,PLLA,
C PLLB,PLATEI(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D ,ST(6,4),STATM(6),STRESSA,STUDS,SWFWS,SWLL,TUPPT(6),UTIL,VOL,
E VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WTFITG(5),WTFITSL,X,PSWA,PSWB
F ,XL,XR,LL,FSAFETY,AVGWT,CODELL
COMMON/BRDATA/BRL,BRW,SPANL1,SPANB,SPANL2,ROW,FU,RMOD,FR,C1,C2,
B C3,GRNU,GCOST,GCOSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
C OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFI,WEBDI,
D UHWEBD,TPT(1),BPT(1),IDET

```



```

COMMON/GRID/ UBGRND,GRNDLBN,IG,CANTLB,CANTUB,WEBDUB,WEBDLBN,ND,
A   PLWLB,PLWUB,IXG,IYC,IXGC,IYCC,IXH,IYB,IXHC,IYBC,ZGC(5,5),
B   CSLAB,CREINF,CSTEEL,BCOST,ZHB(5,5),SAFTY(5,5),PLW(5,5),
C   WH(5,5),DPTH(5,5),DPTB(5,5),DPTG(5,5),DPTC(5,5),SAFTI(5,5)
COMMON/SRCH/OCOST,BGRND,BCANT,BWEBD,BPLATE,DATUM,OH,OP
COMMON/LBC/CANT,ZMINC
10 FORMAT(20X,17HOUTPUT GRID1 GRND,//      5(F10.2,2X))
11 FORMAT(20X,17HOUTPUT GRID1 CANT,//      5(F10.2,2X))
12 FORMAT(20X,17HOUTPUT GRID1 WEBD,//      5(F10.2,2X))
14 FORMAT(20X,19HOUTPUT GRID1 PLATEW,//    5(F12.2,2X))
15 FORMAT(20X,17HOUTPUT GRID1 COST,//      5(F10.2,2X))

16 FORMAT(20X,21HCHECK FOR CONVERGENCE,// 20X,2F10.2,4(14,2X))
17 FORMAT(1H1,//// 20X,15HOUTPUT SUMMARY,2X,5HGRID1)
   IR = 1
   IS = 3
C   IF THE STARTING POINT IS THE OPTIMUM POINT
   IXGT = 1
   IYCT = 3
   DO 80 K = 1,5
   DO 75 J = 1,5
   ZGC(K,J) = 0.0
75 CONTINUE
80 CONTINUE
   ISPEC1 = 1
   IF(ISPEC1 .EQ. 1) GO TO 90
   IR = 3
   IS = 3
C   CO-ORDINATES OF CENTER OF GRID,SPECIFY ONE OF THE FOLLOWING
C   CENTER OF DESIGN SPACE OR
C   SPECIFIED BY THE DESIGNER
90 IXGC = IR
   IYCC = IS
C   TO DETERMINE GRID CO-ORDINATES AND CORRESPONDING DESIGN COSTS
   DO 120 K = 1,3
   DO 110 J = 1,3
   IXG = IXGC + K*1 - 2
   IYC = IYCC + J*1 - 2
C   TO CHECK IF GRID POINT IS IN DESIGN SPACE
   IF(IXG .LT. 1 .OR. IXG .GT. NG) GO TO 110
   IF(IYC .LT. 1 .OR. IYC .GT. 5) GO TO 110
C   TO CHECK IF GRID POINT WAS A PART OF THE PREVIOUS GRID SEARCH
C   TO AVOID REPEATING THE SAME DESIGN
   IF(ZGC(IXG,IYC) .NE. 0.) GO TO 110
C   TO DETERMINE DESIGN COST AT THE POINT (IXG,IYC)
   GRND = UBGRND - (FLOAT(IXG - 1))*1.
   GSPA = (BRW - 2.*2.5)*12./(GRND - 1.)
   WRITE(3,20) GRND,GSPA
20 FORMAT(20X,5HGRID1, 2F10.2)

```





```

CANIB = FLOAT(IFIX(SPANL1/10.)) - 2.0 + (FLOAT(IYC - 1)) * 1.
CALL GRID2
WEBU = WEBDLB + (FLOAT(IXHC - 1)) * 6.0

```

```

C
PLATEW = 14.0 + (FLOAT(IYBC - 1)) * 2.0
TO STORE OPTIMUM PLATEW AND WEBU AT POINT (IXG,IYC)
OPTG(IXG,IYC) = GRNU
OPTC(IXG,IYC) = CANIB
OPTH(IXG,IYC) = WEBU
OPTB(IXG,IYC) = PLATEW
CALL SLAB
CALL PRELIM
CALL DESGIR
CSLAB = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRNU))
CREINF = C2*490.*BRL*BRW/54.*REINF(2)
CSTEEL = C3*(WG(1)*(SPANL1 + CANIB) + WG(2)*SPANL2)*GRNU*1.18
BCOST = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRNU)) + C2*490.*BRL*BRW/54.*REINF(2)
B + C3*(WG(1)*(SPANL1 + CANIB) + WG(2)*SPANL2)*GRNU*1.18
ZGC(IXG,IYC) = BCOST/1000.
IF((ZGC(IXG,IYC)*1000.) .LT. ZMINC) GO TO 105
GO TO 106
105 IXGC = IXG
IYCC = IYC
GO TO 170
106 CONTINUE
110 CONTINUE
120 CONTINUE
WRITE(3,15) ((ZGC(J,K),K = 1,5),J = 1,5)
C
TO LOCATE THE CENTER OF THE GRID
SMALL1 = ZGC(IXGC,IYCC)
WRITE(3,16) ZGC(IXGC,IYCC),SMALL1,IXGC,IYCC,IR,IS
DO 135 K = 1,3
DO 130 J = 1,3
IXG = IXGC + K*1 - 2
IYC = IYCC + J*1 - 2
C
TO CHECK IF THE GRID POINT IS IN THE DESIGN SPACE
IF(IXG.LT.1 .OR. IXG.GT.5) GO TO 130
IF(IYC.LT.1 .OR. IYC.GT.5) GO TO 130
IF(ZGC(IXG,IYC) .LT. SMALL1) GO TO 121
GO TO 130
121 SMALL1 = ZGC(IXG,IYC)
IXGT = IXG
IYCT = IYC
130 CONTINUE
135 CONTINUE
C
SET CO ORDINATES OF MIN. COST DESIGN AS CO ORDINATES OF NEW CENTER
IXGC = IXGT
IYCC = IYCT

```

0-100 = 6840

12  
 13  
 14  
 15  
 16  
 17  
 18  
 19  
 20  
 21  
 22  
 23  
 24  
 25  
 26  
 27  
 28  
 29  
 30  
 31  
 32  
 33  
 34  
 35  
 36  
 37  
 38  
 39  
 40  
 41  
 42  
 43  
 44  
 45  
 46  
 47  
 48  
 49  
 50  
 51  
 52  
 53  
 54  
 55  
 56  
 57  
 58  
 59  
 60  
 61  
 62  
 63  
 64  
 65  
 66  
 67  
 68  
 69  
 70  
 71  
 72  
 73  
 74  
 75  
 76  
 77  
 78  
 79  
 80  
 81  
 82  
 83  
 84  
 85  
 86  
 87  
 88  
 89  
 90  
 91  
 92  
 93  
 94  
 95  
 96  
 97  
 98  
 99  
 100  
 101  
 102  
 103  
 104  
 105  
 106  
 107  
 108  
 109  
 110  
 111  
 112  
 113  
 114  
 115  
 116  
 117  
 118  
 119  
 120  
 121  
 122  
 123  
 124  
 125  
 126  
 127  
 128  
 129  
 130  
 131  
 132  
 133  
 134  
 135  
 136  
 137  
 138  
 139  
 140  
 141  
 142  
 143  
 144  
 145  
 146  
 147  
 148  
 149  
 150  
 151  
 152  
 153  
 154  
 155  
 156  
 157  
 158  
 159  
 160  
 161  
 162  
 163  
 164  
 165  
 166  
 167  
 168  
 169  
 170  
 171  
 172  
 173  
 174  
 175  
 176  
 177  
 178  
 179  
 180  
 181  
 182  
 183  
 184  
 185  
 186  
 187  
 188  
 189  
 190  
 191  
 192  
 193  
 194  
 195  
 196  
 197  
 198  
 199  
 200  
 201  
 202  
 203  
 204  
 205  
 206  
 207  
 208  
 209  
 210  
 211  
 212  
 213  
 214  
 215  
 216  
 217  
 218  
 219  
 220  
 221  
 222  
 223  
 224  
 225  
 226  
 227  
 228  
 229  
 230  
 231  
 232  
 233  
 234  
 235  
 236  
 237  
 238  
 239  
 240  
 241  
 242  
 243  
 244  
 245  
 246  
 247  
 248  
 249  
 250  
 251  
 252  
 253  
 254  
 255  
 256  
 257  
 258  
 259  
 260  
 261  
 262  
 263  
 264  
 265  
 266  
 267  
 268  
 269  
 270  
 271  
 272  
 273  
 274  
 275  
 276  
 277  
 278  
 279  
 280  
 281  
 282  
 283  
 284  
 285  
 286  
 287  
 288  
 289  
 290  
 291  
 292  
 293  
 294  
 295  
 296  
 297  
 298  
 299  
 300  
 301  
 302  
 303  
 304  
 305  
 306  
 307  
 308  
 309  
 310  
 311  
 312  
 313  
 314  
 315  
 316  
 317  
 318  
 319  
 320  
 321  
 322  
 323  
 324  
 325  
 326  
 327  
 328  
 329  
 330  
 331  
 332  
 333  
 334  
 335  
 336  
 337  
 338  
 339  
 340  
 341  
 342  
 343  
 344  
 345  
 346  
 347  
 348  
 349  
 350  
 351  
 352  
 353  
 354  
 355  
 356  
 357  
 358  
 359  
 360  
 361  
 362  
 363  
 364  
 365  
 366  
 367  
 368  
 369  
 370  
 371  
 372  
 373  
 374  
 375  
 376  
 377  
 378  
 379  
 380  
 381  
 382  
 383  
 384  
 385  
 386  
 387  
 388  
 389  
 390  
 391  
 392  
 393  
 394  
 395  
 396  
 397  
 398  
 399  
 400  
 401  
 402  
 403  
 404  
 405  
 406  
 407  
 408  
 409  
 410  
 411  
 412  
 413  
 414  
 415  
 416  
 417  
 418  
 419  
 420  
 421  
 422  
 423  
 424  
 425  
 426  
 427  
 428  
 429  
 430  
 431  
 432  
 433  
 434  
 435  
 436  
 437  
 438  
 439  
 440  
 441  
 442  
 443  
 444  
 445  
 446  
 447  
 448  
 449  
 450  
 451  
 452  
 453  
 454  
 455  
 456  
 457  
 458  
 459  
 460  
 461  
 462  
 463  
 464  
 465  
 466  
 467  
 468  
 469  
 470  
 471  
 472  
 473  
 474  
 475  
 476  
 477  
 478  
 479  
 480  
 481  
 482  
 483  
 484  
 485  
 486  
 487  
 488  
 489  
 490  
 491  
 492  
 493  
 494  
 495  
 496  
 497  
 498  
 499  
 500  
 501  
 502  
 503  
 504  
 505  
 506  
 507  
 508  
 509  
 510  
 511  
 512  
 513  
 514  
 515  
 516  
 517  
 518  
 519  
 520  
 521  
 522  
 523  
 524  
 525  
 526  
 527  
 528  
 529  
 530  
 531  
 532  
 533  
 534

1994

1. The first step is to identify the problem or question being asked.  
 2. Next, we need to gather relevant information and data.  
 3. Then, we analyze the information to determine the cause of the problem.  
 4. After analysis, we develop a plan to address the issue.  
 5. Finally, we implement the plan and monitor the results.

```

C      TO CHECK THE CONVERGENCE IF THE MINIMUM-COST DESIGN
C      IS THE CENTER OF GRID CONVERGENCE IS OBTAINED
      WRITE(3,16) ZGC(IXGC,IYCC),SMALL1,IXGC,IYCC,IR,IS
      IF(IXGC.EQ.IR .AND. IYCC.EQ.IS) GO TO 170
      IR = IXGC
      IS = IYCC
      SMALL1=ZGC(IXGC,IYCC)
      GO TO 90
170 CONTINUE
      WRITE(3,17)
      WRITE(3,10) ((OPTG(J,K),K=1,5),J = 1,5)
      WRITE(3,11) ((OPTC(J,K),K=1,5),J = 1,5)
      WRITE(3,12) ((OPTH(J,K),K=1,5),J=1,5)
      WRITE(3,14) ((OPTB(J,K),K= 1,5),J = 1,5)
      WRITE(3,15) ((ZGC(J,K),K = 1,5),J= 1,5)
      RETURN
      END

```

```

      SUBROUTINE GRID2
      COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BCTTPT(6),CANTA,CANIB,CG(6,4),
A      DELTLA,DELILB,DELTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B      FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,POMB,PLLA,
C      PLLB,PLATEI(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D      ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TUPPT(6),UTIL,VDL,
E      VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WTF TG(5),WTF TSL,X,PSWA,PSWB
F      ,XL,XR,LL,FSAFETY,AVGWT,CODELL
      COMMON/ORDATA/BRL,BRW,SPANL1,SPANB,SPANL2,RDW,FLC,RMODU,FR,C1,C2,
B      C3,GRND,GCCOST,GCCOSTI,WG(2),OPTWEB(10),OPTSPA(10),CPIPL(10),
C      OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFITI,WEBDI,
D      UBWEBD,TPT(1),BPT(1),IDET
      COMMON/GRID/ UBGRND,GRNDLB,NG,CANTLB,CANTUB,WEBDUB,WEBULB,ND,
A      PLWLB,PLWUB,IXG,IYC,IXGC,IYCC,IXH,IYB,IXHC,IYBC,ZGC(5,5),
B      CSLAB,CREINF,CSTEEL,HCCOST,ZHB(5,5),SAFTY(5,5),PLW(5,5),
C      WH(5,5),OPTH(5,5),OPTB(5,5),OPTG(5,5),OPTC(5,5),SAFTI(5,5)
      COMMON/SRCH/OCCOST,BGRND,BCANT,BWEBD,BPLATE,DATUM,OH,LF
      COMMON/LBC/CANT,ZMINC
10  FORMAT(20X,12HOUTPUT GRID2,4FGRND,2X,F5.0,5HCANIB,2X,F5.0)
11  FORMAT(20X,6HPLATEW,/,/, 5(F6.1,2X))
12  FORMAT(20X,4HWEBD,/,/, 5(F6.1,2X))
14  FORMAT(20X,5HSAFTY,/,/, 5(F6.2,2X))
15  FORMAT(20X,4HCCOST,/,/, 5(F10.2,2X))
16  FORMAT(20X,17HCONVERGENCE CHECK,/,20X,2(F10.2,2X),4(I4,2X))
17  FORMAT(1H1,/,/,/, 20X,15HOUTPUT SUMMARY,2X,5HGRID2)
18  FORMAT(20X,5HSAFTI,/,/, 5(F6.2,2X))
      WRITE(3,17)
      WRITE(3,10) GRND,CANTB

```

[illegible]

```

DO 80 K = 1,5
DO 75 J = 1,5
ZHB(K,J) = 0.
WH(K,J) = 0.0
PLW(K,J) = J.0
SAFTY(K,J) = 0.0
75 CONTINUE
80 CONTINUE
IP = 1
IQ = 1
C IF THE STARTING POINT IS THE OPTIMUM POINT
IXHT = 1
IYBT = 1
ISPLC2 = 1
IF(ISPLC2 .EQ.1) GO TO 90
C CO-ORDINATES OF CENTER OF NINE POINT GRID IS ASSUMED TO
C CENTER OF DESIGN SPACE
IP = 3
IQ = 3
90 IXHC = IP
IYBC = IQ
C GIRDER DESIGN AT GRID CENTER MUST SATISFY THE REQUIREMENTS
C OF LATERAL BUCKLING
WEBD = WEBDLB + (FLOAT(IXHC - 1))*6.0
95 PLATEW = 14.0 + (FLOAT(IYBC - 1))*2.0
CALL SLAB
CALL PRELIM
CALL DESGIR
CSLAB = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRND))
CREINF = C2*490.*BRL*BRW/54.*REINF(2)
CSTEEL = C3*(WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*GRND*1.18
BCOST = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRND)) + C2*490.*BRL*BRW/54.*REINF(2)
B C3*(WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*GRND*1.18
ZHB(IXHC,IYBC) = BCOST/1000.
C F.S. IN SAPN 1
SAFTY(IXHC,IYBC) = FSAFTY
C F.S. IN SPAN 2
SAFTI(IXHC,IYBC) = FSAFTI
PLW(IXHC,IYBC) = PLATEW
WH(IXHC,IYBC) = WEBD
IF(FSAFTI.GE.1.25.AND. FSAFTY.GE.1.25) GO TO 100
IYBC = IYBC + 1
GO TO 95
C TO DETERMINE GRID CO-ORDINATES AND CORRESPONDING DESIGN COSTS
100 DO 120 K = 1,3
DO 110 J = 1,3
IF((ZHB(IXHC,IYBC)*1000.) .LT. ZMINC) GO TO 170

```

05 08 A  
06 25 C  
07 26 B  
08 27 A  
09 28 B  
10 29 A  
11 30 B  
12 31 A  
13 32 B  
14 33 A  
15 34 B  
16 35 A  
17 36 B  
18 37 A  
19 38 B  
20 39 A  
21 40 B  
22 41 A  
23 42 B  
24 43 A  
25 44 B  
26 45 A  
27 46 B  
28 47 A  
29 48 B  
30 49 A  
31 50 B  
32 51 A  
33 52 B  
34 53 A  
35 54 B  
36 55 A  
37 56 B  
38 57 A  
39 58 B  
40 59 A  
41 60 B  
42 61 A  
43 62 B  
44 63 A  
45 64 B  
46 65 A  
47 66 B  
48 67 A  
49 68 B  
50 69 A  
51 70 B  
52 71 A  
53 72 B  
54 73 A  
55 74 B  
56 75 A  
57 76 B  
58 77 A  
59 78 B  
60 79 A  
61 80 B  
62 81 A  
63 82 B  
64 83 A  
65 84 B  
66 85 A  
67 86 B  
68 87 A  
69 88 B  
70 89 A  
71 90 B  
72 91 A  
73 92 B  
74 93 A  
75 94 B  
76 95 A  
77 96 B  
78 97 A  
79 98 B  
80 99 A  
81 100 B

```

IXH = IXHC + K*1 - 2
IYB = IYBC + J*1 - 2
C TO CHECK IF GRID POINT IS IN DESIGN SPACE
IF(IXH.LT. 1 .OR. IXH.GT. ND) GO TO 110
IF(IYB.LT. 1 .OR. IYB.GT. 5) GO TO 110
C TO CHECK IF GRID POINT WAS A PART OF THE PREVIOUS GRID SEARCH
C TO AVOID REPEATING THE SAME DESIGN
IF(ZHB(IXH,IYB).NE. 0.) GO TO 110
C TO DETERMINE DESIGN COST AT THE POINT(IXH,IYB)
WEBU = WEBDLB + (FLOAT(IXH - 1))*6.0
PLATEW = 14.0 + (FLOAT(IYB - 1))*2.0
CALL SLAB
CALL PRELIM
CALL DESGIR
C F. S. IN SPAN 1
SAFTY(IXH,IYB) = FSAFTY
C F.S. IN SPAN 2
SAFTI(IXH,IYB) = FSAFTI
PLW(IXH,IYB) = PLATEW
WH(IXH,IYB) = WEBU
CSLAB = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRND))
CREINF = C2*490.*BRL*BRW/54.*REINF(2)
CSTEEL = C3*(WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*GRND*1.18
BCOST = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRND)) + C2*490.*BRL*BRW/54.*REINF(2)
B +C3*(WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*GRND*1.18
ZHB(IXH,IYB) = BCOST/1000.
C METHOD OF BOUNDS GRID2
IF(SAFTY(IXH,IYB).LT.1.25 .OR. SAFTI(IXH,IYB).LT.1.25) GO TO 110
C TO CHECK IF GRID POINT SATISFIES BCOST LT ZMINC
IF((ZHB(IXH,IYB)*1000.) .LT. ZMINC) GO TO 107
GO TO 108
107 IXHC = IXH
IYBC = IYB
GO TO 170
108 CONTINUE
110 CONTINUE
120 CONTINUE
C TO LOCATE THE CENTER OF THE GRID
SMALL2 = ZHB(IXHC,IYBC)
DO 135 K = 1,3
DO 130 J = 1,3
IXH = IXHC + K*1 - 2
IYB = IYBC + J*1 - 2
C TO CHECK IF GRID POINT IS IN DESIGN SPACE
IF(IXH.LT. 1 .OR. IXH.GT. ND) GO TO 130
IF(IYB.LT.1 .OR. IYB.GT.5) GO TO 130
C TO AVOID THE UNSTABLE GIRDER DESIGN FROM COMPARISON
IF(SAFTY(IXH,IYB) .LT. 1.25) GO TO 130

```





```
IF(SAFTI(IXH,IYB).LT.1.25) GO TO 130
```

```
IF(ZHB(IXH,IYB).LT.SMALL2) GO TO 121
```

```
GO TO 130
```

```
121 SMALL2 = ZHB(IXH,IYB)
```

```
IXHT = IXH
```

```
IYBT = IYB
```

```
130 CONTINUE
```

```
135 CONTINUE
```

```
SET CO ORDINATES OF MIN. COST DESIGN AS CO ORDINATES OF NEW CENTER
```

```
IXHC = IXHT
```

```
IYBC = IYBT
```

```
TO CHECK THE CONVERGENCE IF THE MINIMUM COST DESIGN
```

```
IS THE CENTER OF GRID CONVERGENCE IS OBTAINED
```

```
WRITE(3,16) ZHB(IXHC,IYBC),SMALL2,IXHC,IYBC,IP,IQ
```

```
IF(IXHC.EQ.IP.AND.IYBC.EQ.IQ) GO TO 170
```

```
IP = IXHC
```

```
IQ = IYBC
```

```
SMALL2=ZHB(IXHC,IYBC)
```

```
GO TO 90
```

```
170 CONTINUE
```

```
WRITE(3,11) ((PLW(J,K),K=1,5),J=1,5)
```

```
WRITE(3,12) ((WH(J,K),K=1,5),J=1,5)
```

```
WRITE(3,18) ((SAFTI(J,K),K=1,5),J=1,5)
```

```
WRITE(3,14) ((SAFTY(J,K),K=1,5),J=1,5)
```

```
WRITE(3,15) ((ZHB(J,K),K=1,5),J=1,5)
```

```
RETURN
```

```
END
```

# SUBROUTINE SLAB

```
COMMON BMDL,BMLL,BMSW,BMSWL,BMTRK,BOTPT(6),CANTA,CANIB,CG(6,4),
```

```
A DELTLA,DELTLE,DELTLC,DELT(3,4),DISTRB,IMPACT,EXTRA,FATIG(2),
```

```
B FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
```

```
C PLLB,PLATEI(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
```

```
D ,ST(6,4),STATM(6),STRESSA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VOL,
```

```
E VLL,VLLMIN,VSW,VTOTAL,WEBG,WERT,WTFITG(5),WTFISL,X,PSWA,PSWB
```

```
F ,XL,XR,LL,FSAFETY,AVGWT,COEFL
```

```
COMMON/ORDATA/BRLL,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMOD,RK,C1,C2,
```

```
B C3,GRND,GCUST,GCUSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
```

```
C OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFETY,WEBDI,
```

```
D UBWEBD,TPT(1),BPT(1),IDEI
```

```
DATA COVERP/2.75/,COVERN/2.75/
```

```
ASSUMPTIONS
```



C 1 MAIN REINF PERPENDICULAR TO THE TRAFFIC  
 C 2 TOP ONE INCH OF THE CONCRETE IS BE CONSIDERED AS A WEARING  
 C COURSE, AND WILL NOT BE USED FOR D DISTANCE IN DESIGN.  
 C 3 WHEN CHECKING SLAB THICKNESS FOR NEGATIVE MOMENT, THE HAUNCH  
 C WILL NOT BE USED TO CALCULATE THE EFFECTIVE D.  
 C 4 SLAB CONTINUOUS OVER THREE OR MORE SUPPORTS.

IF(LOAD.EQ.1) SLMIN = 7.5  
 IF(LOAD.EQ.2) SLMIN = 7.0  
 SK=1./(1.+(FR/(RMUD\*FC)))  
 SJ=1.-SK/3.  
 RF=0.5\*FC\*SK\*SJ

C  
 C TO DETERMINE THE MOMENTS IN SLAB DUE TO DEAD LOAD AND LIVE LOAD  
 C

ESPAN=GSPA=PLATEW/2.  
 SLBT= GSPA/12.  
 K=0  
 1 SBMDLP=(12.5\*SLBT +25.)\*ESPAN\*ESPAN/2016.  
 SBMDLN=(12.5\*SLBT +25.)\*ESPAN\*ESPAN/1440.  
 IF(LOAD.EQ.1) GO TO 4  
 SBMLL=(ESPAN/12. +2.)/32.\*12000.\*0.8\*1.3  
 GO TO 7  
 4 SBMLL=(ESPAN/12. +2.)/32.\*16000.\*0.8\*1.3  
 7 SBMTLP=SBMDLP +SBMLL  
 SBMTLN=SBMDLN +SBMLL  
 SLABDP=SQRT(SBMTLP/RF)  
 SLABDN=SQRT(SBMTLN/RF)  
 SLABTP=SLABDP+COVERP  
 SLABTN=SLABDN+COVERN  
 SLABTH=AMAX1(SLABTP,SLABTN,SLMIN)  
 SLBT = SLABTH  
 K=K+1  
 IF(K-2)1,2,2  
 2 S = SLBT  
 SLBT=AMAX1((FLOAT(IFIX(S\*2.+.99)))/2.,SLMIN)  
 REINFP=SBMTLP\*12./(20000.\*0.9\*(SLBT-COVERP))  
 REINFN=SBMTLN\*12./(20000.\*0.9\*(SLBT-COVERN))  
 REINF(2)=AMAX1(REINFP,REINFN)  
 SLBT= 10.  
 IF(GSPA.LE.144.) SLBT= 9.5  
 IF(GSPA.LE.129.) SLBT= 9.0  
 IF(GSPA.LE.114.) SLBT= 8.5  
 IF(GSPA.LE. 99.) SLBT= 8.0  
 IF(GSPA.LE. 78.) SLBT= 7.5  
 REINF(2)= 1.0  
 IF(GSPA.LE.144.) REINF(2)= 0.88  
 IF(GSPA.LE.123.) REINF(2)= 0.81  
 IF(GSPA.LE.108.) REINF(2)= 0.75



```

IF(GSPA.LE. 87.) REINF(2)= 0.66
RETURN
END

```

## SUBROUTINE PRELIM

```

COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BCTTPT(6),CANTA,CANIB,CG(6,4),
A  DELTLA,DELTLB,DELTLC,DELI(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FSC(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VOL,
E  VLL,VLLMIN,VSW,VTOTAL,WBBD,WBET,WTFTG(5),WTFTSL,X,PSWA,PSWB
F  ,XL,XR,LL,FSAFETY,AVGWT,COEELL
COMMON/BRDATA/BRL,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMODU,FR,C1,C2,
B  C3,GRND,GCOST,GCOSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
C  OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFI,WBBDI,
D  UBWBBD,TPT(1),BPT(1),IDEI
C  TO DETERMINE UTIL,SWLL, AND SWFWS IN LBS. PER FOOT OF BEAM
UTIL = TUTIL/GRND
SWLL = TSWLL/GRND
SWFWS = (RAILWT + 25.0*RDW)/GRND
HDEPTH = 1.0
HWIDTH = PLATEW + 6.0
C  TO DETERMINE CANTILEVER REACTIONS
CANTA = 0.
PBMA = 0.
PSLA = 0.
PSWA = 0.
PLLA = 0.
RETURN
END

```

## SUBROUTINE DESGIR

```

COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BCTTPT(6),CANTA,CANIB,CG(6,4),
A  DELTLA,DELTLB,DELTLC,DELI(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FSC(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VOL,
E  VLL,VLLMIN,VSW,VTOTAL,WBBD,WBET,WTFTG(5),WTFTSL,X,PSWA,PSWB
F  ,XL,XR,LL,FSAFETY,AVGWT,COEELL
COMMON/BRDATA/BRL,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMODU,FR,C1,C2,
B  C3,GRND,GCOST,GCOSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
C  OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFI,WBBDI,
D  UBWBBD,TPT(1),BPT(1),IDEI

```

```

C
C
C   TO DESIGN GIRDER IN SPAN 2

CLB = CANTB
SPANL = SPANB - CANTB
TOPPT(1) = 0.
BOTTP(1) = 0.
WEBT = 0.
IF(LOAD.EQ.1) CUDELL = 72.
IF(LOAD.EQ.2) CUDELL = 40.
DISTLL = 1.1
HDEPTH = 1.0
HWIDTH = PLATEW + 6.0
STUUS = 4.0
CANTB = 0.
PBMB = 0.
PSLB = 0.
PSWB = 0.
PLLB = 0.
CALL GIRDER
FSAF11 = FSAFTY
IF(IDET.EQ.1) GO TO 90
CALL DETGIR
90 WG(2) = AVGWT

C
C   TO ORIGINAL VALUES OF CANTILEVER
CANTB = CLB

C
C   TO DESIGN GIRDER IN SPAN 1

C
C   TO DETERMINE CANTILEVER REACTIONS FOR B
SPANL2 = SPANB - CANTB
WTFTSL = 1.042*(GSPA*SLABT + HWIDTH*HDEPTH)
IF(WG(2).EQ.0.) PBMB = 0.25*SPANL2/2.
IF(WG(2).NE.0.) PBMB = WG(2)*SPANL2/2000.*1.18
PSLB = WTFTSL*SPANL2/2000.
PSWB = SWFWS*SPANL2/2000.
IF(LOAD.EQ.1) GO TO 110
PLLB = AMAX1((40.-112./SPANL2)*(GSPA/(120.*1.1)),
A          (26. + 0.32*SPANL2)*(GSPA/(120.*1.1)))
GO TO 111
110 PLLB = AMAX1((72. - 672./SPANL2)*(GSPA/(120.*1.1)),
A          (26. + 0.32*SPANL2)*(GSPA/(120.*1.1)))
111 SPANL = SPANL1
WEBT = 0.0
TOPPT(1) = 0.
BOTTP(1) = 0.
CALL GIRDER
TPT(1) = TOPPT(1)
BPT(1) = BOTTP(1)

```

1000

1000  
1000  
1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

1000

```

      IF(CIDET.EQ.1) GO TO 115
      CALL DELGIR
115  WG(1) = AVGW
      GCOST = (WG(1)*(SPANL1 + CANTB) + WG(2)*SPANL2)*1.18*G3
      RETURN
      END

```

## SUBROUTINE GIRDER

P L A T E G I R D E R D E S I G N

MARCH 29, 1972.

```

REAL I,ITOP,IWEB,IBUTT,N,LENGTH,IDEF,NEGLL,IY,LLR,LLMAXR,LLRMAX,
A   LLRWDI,IX
COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BOTTP(6),CANTA,CANTB,CG(6,4),
A   DELTLA,DELTLC,DELTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B   FSC(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PMBB,PLIA,
C   PLLB,PLATET(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D   ,ST(6,4),STATM(6),STRESA,STUUS,SWFWS,SWLL,TUPPT(6),UTIL,VOL,
E   VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WTFITG(5),WTFITSL,X,PSWA,PSWB
F   ,XL,XR,LL,FSAFTY,AVGW,CODELL
COMMON/DES1/ALLOWC(4),ALLOWE(4),MUT(12),PROBLL(4),TRATIO(2)
COMMON/DES2/ADJ(4),DLRT(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),
A   ISTIFF(2),TWEB(9),LLRWDI(2)
COMMON/DET/TOPPL1,TOPPL2,TOPPL3,BOTPL1,BOTPL2,BOTPL3,XCENT,
A   FTUPL,FSTOP,FTUPR,FBUPL,FBUPL,FBUPL,FBUPL,BMDLA,BMDLC,BMDLB,BMLLA,
B   BMLLC,BMLLB,BMSWA,BMSWC,BMSWB,WSTIFF,STUD,STIFA,PT,STIT,STIW,
C   STIFR,STIFB,STIFL
DATA E/29000000./,EMPMAX/1.3/,PLMIN/0.3/5/,PSTUDM/24./
XL = 0.0
XR = 1.0
LOAD = 1
IF(CODELL.EQ.40.) LOAD = 2
IF(CODELL.EQ.85.) LOAD = 3
IF(CODELL.EQ.80.) LOAD = 4
DISTLL=1.1
IF(LOAD.EQ.1)LL=20
IF(LOAD.EQ.2)LL=20
IF(LOAD.EQ.3)LL=85
IF(LOAD.EQ.3)LL=80
TOTALL = CANTA+SPANL+CANTB
EMPACT = AMIN1(EMPMAX*1.0+50./(SPANL+125.))
IF(LOAD.EQ.3) EMPACT = 1.0
RRIMP = 1.0+(60.-3.*SPANL**2/1600.)/100.
IF(SPANL.GT.80.0) RRIMP = 1.0+(36.+600./(SPANL-30))/100.
DISTRB = EMPACT*GSPA*LL/(120.*PROBLL(LOAD)*DISTLL)
IF(LOAD.EQ.3)DISTRB = LL*GRSP/12./(DISTLL*PROBLL(LOAD))

```





```

IF (LOAD.EQ.4) DISTRB = RRIMP*GSPA/(120.*DISTLL)
EXTRA = 1.18
IF (WEBT.EQ.0.) EXTRA = 1.12
ADJUST = -1.0
PLG = 0.0
IF (TOPPT(1).NE.0.) GO TO 126
IF (PLATEW.NE.0.0) GO TO 123
PLG = 1.0
PLATEW = AMAX1(3.0+FIX(SPANL/20.0)*2.0, FIX(WEBD/15.0+4.80))
PLATEW = PLATEW + 2.0
HWIDTH = PLATEW+6.0
123 ADJUST = 1.
TOPPT(1) = 0.375
BOTPT(1) = 0.375
126 WTFISL = 1.042*(GSPA*SLANT+HWIDTH*HDEPTH)
IF (WEBT.GT.0.0) GO TO 130
WEBT = AMAX1((FLOAT(FIX(WEBD/150.*16.+0.99)))/16.0, PLMIN)
WTFISL(1) = 300.
CALL SHEAR (0.0)
VL = VTOTAL
CALL SHEAR (8.0)
T = AMAX1(WEBD/150., (WEBD*AMAX1(VL, VTOTAL)*1000./7500.**2)**0.333)
TREWD = AMAX1(FIX(1*16.+0.99)/16.0, PLMIN)
WEBT = TREWD
IF (LOAD.EQ.4) WEBT = AMAX1(FIX(WEBD/145.*16.+0.99)/16.0, PLMIN)
IF (LOAD.EQ.4.AND. STRESA.LE.23500.) WEBT = AMAX1(FIX(WEBD/170.*16.
A +0.99)/16.0, PLMIN)
C
C          P L A T E   T H I C K N E S S E S   +   L E N G T H S
130 X = SPANL/2. + 1.0
IF ((PBMA+PSLA)*CANTA.LE.(PBMb+PSLB)*CANTB) X = SPANL/3.
134 X = X-4.0
FSBUTT = -99999.
CALL PROPTY (1)
152 CALL MOMENT
CALL STRESS(1)
IF (FS(2).LE.FSBUTT) GO TO 154
X = X + 1.0
FSBUTT = FS(2)
GO TO 152
154 IF (ADJUST.LE.0.0) GO TO 162
CODE = 0.0
XTOP = AMAX1(PLMIN, TOPPT(1)*FS(1)/(STRESA+100.), PLATEW*
A SQRT(FS(1))/3250., PLATEW/24.)
XBUTT = AMAX1(PLMIN, BOTPT(1)*FS(2)/(STRESA+100.), PLATEW/24.)
IF (ABS(TOPPT(1)-XTOP).GE.0.007) CODE = 1.0
IF (ABS(TOPPT(1)-XTOP).GE.0.007) TOPPT(1) = XTOP
IF (ABS(BOTPT(1)-XBUTT).GE.0.007) CODE = 1.0
IF (ABS(BOTPT(1)-XBUTT).GE.0.007) BOTPT(1) = XBUTT
IF (CODE.EQ. 1.0) GO TO 134

```



```

      TOPPT(1) = IFIX(TOPPT(1)*8.0+0.99)/8.0
      BOTPT(1) = IFIX(BOTPT(1)*8.0+0.99)/8.0
162  XCEN = X
      DO 164 M=1,5,1
      TOPPT(M) = TOPPT(1)
      BOTPT(M) = BOTPT(1)
164  CALL PRUPTY (M)
      WFTIDL = WFTIG(1)+WFTISL+UTIL
      CALL MUMENT
      CALL STRESS(1)
      FSBUTT = FS(2)
      X = -0.299
172  X = X+1.0
      CALL MUMENT
      CALL STRESS(1)
      IF (BMUL+POSLL+BMSW+BMSWLL.LT.0.0) GO TO 172
      XL = X/SPANL
      X = SPANL+0.299
176  X = X-1.0
      CALL MUMENT
      CALL STRESS(1)
      IF (BMUL+POSLL+BMSW+BMSWLL.LT.0.0) GO TO 176
      XR = X/SPANL

      FLANGE PLATE CUTOFFS
      TOPPT(2) = IFIX(AMAX1(TOPPT(1)*4.+0.5,8.0*(PLATEW/(3250./
A      SQRT(STRESSA))+0.124)))/8.0
      IF(TOPPT(2)+0.125.GE.TOPPT(1)) GO TO 266
      IF(TOPPT(2)+0.250.GE.TOPPT(1) .AND. TOTAL.LE. 60.0) GO TO 266
      DO 224 K=3,5,1
224  TOPPT(K) = TOPPT(2)
      X = XCEN
      CALL PRUPTY (2)
228  CALL MUMENT
      CALL STRESS(2)
      IF(FS(1).LE.STRESSA .AND. FATIG(1).EQ.0.0) GO TO 232
      X = X-1.0
      GO TO 228
232  TOPPL1 = X+CANTB
      X = XCEN
      CALL PRUPTY (3)

234  CALL MUMENT
      CALL STRESS(3)
      IF(FS(1).LE.STRESSA .AND. FATIG(1).EQ.0.0) GO TO 238
      X = X + 1.0
      GO TO 234
238  TOPPL3 = SPANL-X+CANTB
      TOPPL2 = TOTAL-TOPPL1-TOPPL3

```

[illegible]

1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214, 2215, 2216, 2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227, 2228, 2229, 2230, 2231, 2232, 2233, 2234, 2235, 2236, 2237, 2238, 2239, 2240, 2241, 2242, 2243, 2244, 2245, 2246, 2247, 2248, 2249, 2250, 2251, 2252, 2253, 2254, 2255, 2256, 2257, 2258, 2259, 2260, 2261, 2262, 2263, 2264, 2265, 2266, 2267, 2268, 2269, 2270, 2271, 2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2280, 2281, 2282, 2283, 2284, 2285, 2286, 2287, 2288, 2289, 2290, 2291, 2292, 2293, 2294, 2295, 2296, 2297, 2298, 2299, 2300, 2301, 2302, 2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312, 2313, 2314, 2315, 2316, 2317, 2318, 2319, 2320, 2321, 2322, 2323, 2324, 2325, 2326, 2327, 2328, 2329, 2330, 2331, 2332, 2333, 2334, 2335, 2336, 2337, 2338, 2339, 2340, 2341, 2342, 2343, 2344, 2345, 2346, 2347, 2348, 2349, 2350, 2351, 2352, 2353, 2354, 2355, 2356, 2357, 2358, 2359, 2360, 2361, 2362, 2363, 2364, 2365, 2366, 2367, 2368, 2369, 2370, 2371, 2372, 2373, 2374, 2375, 2376, 2377, 2378, 2379, 2380, 2381, 2382, 2383, 2384, 2385, 2386, 2387, 2388, 2389, 2390, 2391, 2392, 2393, 2394, 2395, 2396, 2397, 2398, 2399, 2400, 2401, 2402, 2403, 2404, 2405, 2406, 2407, 2408, 2409, 2410, 2411, 2412, 2413, 2414, 2415, 2416, 2417, 2418, 2419, 2420, 2421, 2422, 2423, 2424, 2425, 2426, 2427, 2428, 2429, 2430, 2431, 2432, 2433, 2434, 2435, 2436, 2437, 2438, 2439, 2440, 2441, 2442, 2443, 2444, 2445, 2446, 2447, 2448, 2449, 2450, 2451, 2452, 2453, 2454, 2455, 2456, 2457, 2458, 2459, 2460, 2461, 2462, 2463, 2464, 2465, 2466, 2467, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2487, 2488, 2489, 2490, 2491, 2492, 2493, 2494, 2495, 2496, 2497, 2498, 2499, 2500, 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 2509, 2510, 2511, 2512, 2513, 2514, 2515, 2516, 2517, 2518, 2519, 2520, 2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544, 2545, 2546, 2547, 2548, 2549, 2550, 2551, 2552, 2553, 2554, 2555, 2556, 2557, 2558, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2567, 2568, 2569, 2570, 2571, 2572, 2573, 2574, 2575, 2576, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2592, 2593, 2594, 2595, 2596, 2597, 2598, 2599, 2600, 2601, 2602, 2603, 2604, 2605, 2606, 2607, 2608, 2609, 2610, 2611, 2612, 2613, 2614, 2615, 2616, 2617, 2618, 2619, 2620, 2621, 2622, 2623, 2624, 2625, 2626, 2627, 2628, 2629, 2630, 2631, 2632, 2633, 2634, 2635, 2636, 2637, 2638, 2639, 2640, 2641, 2642, 2643, 2644, 2645, 2646, 2647, 2648, 2649, 2650, 2651, 2652, 2653, 2654, 2655, 2656, 2657, 2658, 2659, 2660, 2661, 2662, 2663, 2664, 2665, 2666, 2667, 2668, 2669, 2670, 2671, 2672, 2673, 2674, 2675, 2676, 2677, 2678, 2679, 2680, 26

```

240 BOTIPT(4) = IF(X*AMAX1(BOTIPT(1)*4.+.5,6.*(PLATEW/24.+.120)))/8.0
    BOTIPT(5) = BOTIPT(4)
    IF(BOTIPT(4)+0.125.GE.BOTIPT(1)) GO TO 262
    IF(BOTIPT(4)+0.250.GE.BOTIPT(1).AND. TOTALL.LE.60.0) GO TO 262
    X = XCENT
    CALL PRUPTY (4)
250 CALL MOMENT
    CALL STRESS(4)
    IF(FS(2).LE.STRESSA .AND. FATIG(2).EQ. 0.0) GO TO 254
    X = X-1.0
    GO TO 250
254 BOTPL1 = X+CANTA
    X = XCENT
    CALL PRUPTY (5)
258 CALL MOMENT
    CALL STRESS(5)
    IF(FS(2).LE.STRESSA .AND. FATIG(2).EQ. 0.0) GO TO 260
    X = X + 1.0
    GO TO 258
260 BOTPL3 = SPANL-X+CANTB
    BOTPL2 = TOTALL-BOTPL1-BOTPL3
    GO TO 270
262 BOTPL1 = 0.0
    BOTPL2 = TOTALL
    BOTPL3 = 0.0
    BOTIPT(4) = BOTIPT(1)
    BOTIPT(5) = BOTIPT(1)
    GO TO 270
266 TOPPL1 = 0.0
    TOPPT(2) = TOPPT(1)
    TOPPL2 = TOTALL
    TOPPL3 = 0.0
    GO TO 240

```

C  
C

# C A N T I L E V E R      S T R E S S E S

```

270 DO 272 K=1,4,1
272 ADJ(K) = 0.0
274 BMDLA = -(WTFITG(4)+WTFISL+UTIL)*CANTA**2/2000.-(PBMA+PSLA)*CANTA
    BMDLB = -(WTFITG(5)+WTFISL+UTIL)*CANTB**2/2000.-(PBMB+PSLB)*CANTB
    BMLLA = -CANTA*PLLA*1.3
    BMLLB = -CANTB*PLLB*1.3
    IF(LOAD.EQ.3) BMLLA = BMLLA/1.3
    IF(LOAD.EQ.3) BMLLB = BMLLB/1.3
    BMSWA = -SWFWS*CANTA**2/2000.-CANTA*PSWA
    BMSWB = -SWFWS*CANTB**2/2000.-CANTB*PSWB
    FTOPL = -(BMDLA+BMLLA+BMSWA)*12000./ST(4,1)
    FBOTTL = -(BMDLA+BMLLA+BMSWA)*12000./SB(4,1)
    FTOPR = -(BMDLB+BMLLB+BMSWB)*12000./ST(5,1)
    FBOTTR = -(BMDLB+BMLLB+BMSWB)*12000./SB(5,1)
    IF(ADJUST.LE. 0.0) GO TO 376

```

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

1001111

```

ALCOMP = AMIN1(STRESA+100.,(3250.*BOTIPT(4)/PLATEW)**2)
IF(FTOPL.LE.STRESA+100. .AND. FBOTTL.LE.ALCOMP) GO TO 277
ADJ(1) = 1.0
IF(FTOPL.GE.STRESA+100. .AND. TOPPT(4).LE.BOTIPT(4))
ATOPPT(4) = TOPPT(4) + 0.125

IF(FBOTTL.GE.ALCOMP) BOTIPT(4) = BOTIPT(4) + 0.125
IF(BOTIPT(4).LE.TOPPT(4)) BOTIPT(4) = TOPPT(4)

CALL PROPTY (4)
GO TO 274
277 IF(ADJ(1).EQ. 0.0) GO TO 286
ADJ(1) = 0.0
IF(TOPPT(4).EQ.TOPPT(1)) GO TO 279
X = XCEN
IF(TOPPT(4).LE.TOPPT(1)) GO TO 278
BOTIPT(2) = BOTIPT(4)
CALL PROPTY (2)
X = 1.0
278 IF(TOPPT(4).GE.TOPPT(1)) CALL MOMENT
IF(TOPPT(4).GE.TOPPT(1)) CALL STRESS(2)
IF(TOPPT(4).LE.TOPPT(1)) CALL MOMENT
IF(TOPPT(4).LE.TOPPT(1)) CALL STRESS(4)
IF(FS(1).LE.STRESA .AND. FATIG(1) .EQ. 0.0) GO TO 280
IF (TOPPT(4).LT.TOPPT(1)) X = X-1.0
IF (TOPPT(4).GT.TOPPT(1)) X = X+1.0
GO TO 278
279 TOPPL1 = 0.0
GO TO 281
280 TOPPL1 = X+CANTA
281 TOPPL2 = TOTALL-TOPPL1-TOPPL3
IF(BOTIPT(4).EQ.BOTIPT(1)) GO TO 285
X = XCEN
IF (BOTIPT(4).GT.BOTIPT(1)) X = 1.0
284 IF(BOTIPT(4).GE.BOTIPT(1)) CALL MOMENT
IF(BOTIPT(4).GE.BOTIPT(1)) CALL STRESS(2)
IF(BOTIPT(4).LE.BOTIPT(1)) CALL MOMENT
IF(BOTIPT(4).LE.BOTIPT(1)) CALL STRESS(4)
IF(FS(2).LE.STRESA .AND. FATIG(2) .EQ. 0.0) GO TO 286
IF (BOTIPT(4).LT.BOTIPT(1)) X = X-1.0
IF (BOTIPT(4).GT.BOTIPT(1)) X = X+1.0
GO TO 284
285 BOTPL1 = 0.0
GO TO 287
286 BOTPL1 = X+CANTA
287 BOTPL2 = TOTALL-BOTPL1-BOTPL3
288 ALCOMP = AMIN1(STRESA+100.,(3250.*BOTIPT(5)/PLATEW)**2)
IF(FTOPR.LE.STRESA+100. .AND. FBOTTR.LE.ALCOMP) GO TO 289
ADJ(3) = 1.0
IF(FTOPR.GE.STRESA+100. .AND. TOPPT(5).LE.BOTIPT(5))

```



8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

8024A

```

      ATOPPT(5) = TOPPT(5) + 0.125
      IF(FBOTTR.GE.ALCOMP) BOTIPT(5) = BOTIPT(5) + 0.125
      IF(BOTIPT(5).LE.TOPPT(5)) BOTIPT(5) = TOPPT(5)
      CALL PRUPTY (5)
      GO TO 274
289  IF(ADJ(3).EQ. 0.0) GO TO 302
      ADJ(3) = 0.0
      IF(TOPPT(5).EQ. TOPPT(1)) GO TO 291
      X = XCEN
      IF(TOPPT(5).LE.TOPPT(1)) GO TO 290
      BOTIPT(3) = BOTIPT(5)
      CALL PRUPTY (3)
      X = SPANL-1.0
290  IF(TOPPT(5).GE.TOPPT(1)) CALL MOMENT
      IF(TOPPT(5).GE.TOPPT(1)) CALL STRESS(3)
      IF(TOPPT(5).LE.TOPPT(1)) CALL MOMENT
      IF(TOPPT(5).LE.TOPPT(1)) CALL STRESS(5)
      IF(FS(1).LE.STRESSA .AND. FATIG(1).EQ. 0.0) GO TO 292
      IF (TOPPT(5).LT.TOPPT(1)) X = X+1.0
      IF (TOPPT(5).GT.TOPPT(1)) X = X-1.0
      GO TO 290

291  TOPPL3 = 0.0
      GO TO 293
292  TOPPL3 = SPANL-X+CANTB

293  TOPPL2 = TOTALL-TOPPL1-TOPPL3
      IF(BOTIPT(5).EQ.BOTIPT(1)) GO TO 297
      IF(ADJ(4).EQ. 0.0) GO TO 302
      ADJ(4) = 0.0
      X = XCEN
      IF (BOTIPT(5).GT.BOTIPT(1)) X = SPANL-1.0
296  IF(BOTIPT(5).GE.BOTIPT(1)) CALL MOMENT
      IF(BOTIPT(5).GE.BOTIPT(1)) CALL STRESS(3)
      IF(BOTIPT(5).LE.BOTIPT(1)) CALL MOMENT
      IF(BOTIPT(5).LE.BOTIPT(1)) CALL STRESS(5)
      IF(FS(2).LE.STRESSA .AND. FATIG(2).EQ. 0.0) GO TO 298
      IF (BOTIPT(5).LT.BOTIPT(1)) X = X+1.0
      IF (BOTIPT(5).GT.BOTIPT(1)) X = X-1.0
      GO TO 296
297  BOTPL3 = 0.0
      GO TO 300
298  BOTPL3 = SPANL-X+CANTB
300  BOTPL2 = TOTALL-BOTPL1-BOTPL3
302  TOPPT(2) = TOPPT(4)
      IF (TOPPL1.LT.BOTPL1) TOPPT(2) = TOPPT(1)
      BOTIPT(2) = BOTIPT(1)
      IF (TOPPL1.LT.BOTPL1) BOTIPT(2) = BOTIPT(4)
      CALL PRUPTY (2)
      TOPPT(3) = TOPPT(5)

```



```

IF (TOPPL3.LT.BOTPL3) TOPPT(3) = TOPPT(1)
BOTIPT(3) = BOTIPT(1)
IF (TOPPL3.LT.BOTPL3) BOTIPT(3) = BOTIPT(5)
CALL PRUPTY (3)

```

C  
C

# C A N T I L E V E R      D E F L E C T I O N S

```

CALL DELTA (BOTPL1,BOTPL2,BOTPL3,TOPPL1,TOPPL2,TOPPL3)
IF(DELTA.GE.ALLOWE(LOAD)-4.0.OR. CANTA.EQ. 0.0) GO TO 322
TOPPT(4) = TOPPT(4)+0.125
ADJ(1) = 1.0
IF (TOPPT(4).LE.BOTIPT(4)) GO TO 320
BOTIPT(4) = BOTIPT(4)+0.125
ADJ(2) = 1.0
320 CALL PRUPTY (4)
GO TO 274
322 IF(DELTA.GE.ALLOWE(LOAD)-4.0.OR. CANTB.EQ. 0.0) GO TO 348
TOPPT(5) = TOPPT(5)+0.125
ADJ(3) = 1.0
IF (TOPPT(5).LE.BOTIPT(5)) GO TO 324
BOTIPT(5) = BOTIPT(5)+0.125
ADJ(4) = 1.0
324 CALL PRUPTY (5)
GO TO 274

```

C  
C

# C E N T E R L I N E      D E F L E C T I O N S

```

348 IF (DELTA.GT.ALLOWC(LOAD)) GO TO 376
IF(STUDS.EQ.0.0.AND.TOPPT(1).LE.BOTIPT(1)) TOPPT(1)=TOPPT(1)+0.125
BOTIPT(1) = BOTIPT(1)+0.125
CALL PRUPTY (1)
IF (TOPPL1.GT.BOTPL1) GO TO 354
IF (BOTIPT(1).LE.BOTIPT(2)*2.0) GO TO 356
BOTIPT(2) = BOTIPT(2)+0.125
CALL PRUPTY (2)
352 BOTIPT(4) = BOTIPT(4)+0.125
CALL PRUPTY (4)
GO TO 356
354 BOTIPT(2) = BOTIPT(1)
CALL PRUPTY (2)
IF (BOTIPT(1).GT.BOTIPT(4)*2.0) GO TO 352
356 IF (TOPPL3.GT.BOTPL3) GO TO 360
IF (BOTIPT(1).LE.BOTIPT(3)*2.0) GO TO 372

BOTIPT(3) = BOTIPT(3)+0.125

CALL PRUPTY (3)
358 BOTIPT(5) = BOTIPT(5)+0.125
CALL PRUPTY (5)
GO TO 372
360 BOTIPT(3) = BOTIPT(1)

```



```

CALL PROPTY (3)
IF (BOTIPT(1).GT.BOTIPT(5)*2.0) GO TO 358
372 CALL DELTA (BOTPL1,BOTPL2,BOTPL3,TOPPL1,TOPPL2,TOPPL3)
GO TO 274
376 X = XCENT
IF(TOPPT(1) + 0.125.NE.TOPPT(4) .AND.
A TOPPT(1) + 0.125.NE. TOPPT(5)) GO TO 378
TOPPT(1) = TOPPT(1)+0.125
GO TO 162
378 IF(BOTIPT(1) + 0.125.NE. BOTIPT(4) .AND.
ABOTIPT(1) + .125 .NE. BOTIPT(5)) GO TO 380
BOTIPT(1) = BOTIPT(1)+0.125
GO TO 162
380 CALL MOMENT
CALL STRESS(1)
400 FSTOP = FS(1)
FSBOTI = FS(2)
BMDLC = BMOL
BMLLC = POSLL
BMSWC = RMSW
CALL DELTA (BOTPL1,BOTPL2,BOTPL3,TOPPL1,TOPPL2,TOPPL3)
412 AVGW = ((TOPPL1*TOPPT(4)+TOPPL3*TOPPT(5)+BOTPL1*BOTIPT(4)+BOTPL3*
A BOTIPT(5)+TOPPL2*TOPPT(1)+BOTPL2*BOTIPT(1))*PLATEW/TOTALL+
B WEBO*WEBT)*3.4

```

C  
C

```

S H E A R C O N N E C T O R S + S T I F F E N E R S
SSPAMN = 99999.
IF(EXTRA.EQ.1.18) GO TO 416
CALL SHEAR (0.0)
VL = VTOTAL
CALL SHEAR (8.0)
T = AMAX1(WEBD/150.,(WEBD*AMAX1(VL,VTOTAL)*1000./7500.**2)**0.333)
IF(LOAD.EQ.4) T = WEBD/51.
IF(LOAD.EQ.4 .AND. STRESA.LE.23500.) T = WEBD/60.
TREQD = AMAX1(IFIX(T*16.+0.99)/16.0,PLMIN)
IF(TREQD.LE.WEBT) GO TO 416
WEBT = TREQD
GO TO 123
416 DO 420 J=1,9,1
VN = J-1
CALL SHEAR (VN)
X = VN*SPANL/8.
CALL FINDM (TOPPL1,TOPPL3,BOTPL1,BOTPL3)
PSTUD(J) = 5.96*STUDS*I(M,3)/(CVLL+VLLMIN)*STATM(M)+0.001)
IF(LOAD.EQ.3) PSTUD(J) = PSTUD(J)*7.32/5.96
AWEB = WEBT*(WEBD+TOPPT(M)+BOTIPT(M))
TWEB(J) = WEBD*SQRT(1000.*VTOTAL/AWEB)/7500.
IF(LOAD.EQ.4) TWEB(J) = WEBD/51.
IF(LOAD.EQ.4 .AND. STRESA.LE.23500.) TWEB(J) = WEBD/60.
STIFF(J) = AMIN1(11000.*WEBT/SQRT(1000.*VTOTAL/AWEB),WEBD)

```

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

100-1000

```

      IF (LOAD.EQ.4) STIFF(J) = AMIN1(10500.*WEBT/SQRT(1000.*VJOTAL/
A      AWEB),72.0)

420 SSPAMN = AMIN1(SSPAMN,STIFF(J))
      IF (LOAD.EQ.1) PCTV = 40./72.
      IF (LOAD.EQ.2) PCTV = 8./40.
      PLLL = PLLA*(1.0+PCTV*CANTA/(PSLA*1000./WFTSL+0.001))
      VL = 1000.*(PBMA+PSLA+PLLL*1.3+PSWA)+(WFTG(4)+WFTSL)*CANTA+0.01
      STIFA = AMIN1(11000.*WEBT/SQRT(VL/(WEBT*(WEBD+TOPPT(4)+BUTTP(4))))
A      )) ,WEBD)
      TWEBL = WEBD*SQRT(VL/(WEBT*(WEBD+TOPPT(4)+BUTTP(4))))/7500.
      PLLR = PLLB*(1.0+PCTV*CANTB/(PSLB*1000./WFTSL+0.001))
      VR = 1000.*(PBMB+PSLB+PLLR*1.3+PSWB)+(WFTG(5)+WFTSL)*CANTB+0.01
      STIFB = AMIN1(11000.*WEBT/SQRT(VR/(WEBT*(WEBD+TOPPT(5)+BUTTP(5))))
A      )) ,WEBD)
      TWEBR = WEBD*SQRT(VR/(WEBT*(WEBD+TOPPT(5)+BUTTP(5))))/7500.
      IF (WEBT.GT.WEBD/150.) GO TO 426
422 STIFL = 0.50
      STIFR = 0.50
      GO TO 446
426 DO 428 K=1,5,1
      IF (TWEB(K).LE.WEBT) GO TO 430
428 CONTINUE
      GO TO 422
430 IF (K.EQ.1) STIFL = 0.0
      IF (K.GE.2) STIFL = (K-1)/8.-(WEBT-TWEB(K))/8./(TWEB(K-1)-TWEB(K))
      DO 440 K=5,9,1
      IF (TWEB(14-K).LE.WEBT) GO TO 442
440 CONTINUE
      GO TO 422
442 IF (K.EQ.5) STIFR = 1.0
      IF (K.GE.6) STIFR = (13-K)/8.+(WEBT-TWEB(14-K))/8./(TWEB(15-K)-
A      TWEB(14-K))
446 IF (STUDS.EQ.0.0) GO TO 461
      PSTUD1 = AMIN1(PSTUD(1),PSTUD(2),PSTUD(5),PSTUD(8),PSTUD(9))
      H1 = ((TOPPT(1)+BUTTP(1))*PLATEW+WEBD*WEBT)*STRESA/0.55
      H2 = 0.85*3000.*SLABT*SLABW
      PSTUD2 = SPANL*(XR-XL)*12.*STUDS/(2.*AMIN1(H1,H2)/(0.85*28700.))
      STUD = IFIX(AMIN1(PSTUD1,PSTUD2,PSTUDM))
      IF (STIFL.EQ.0.0 .AND. STIFR.EQ.1.0 .AND. (CANTA*12.0.LE.STIFA .AND.
A      WEBT.GE.TWEBL).AND. (CANTB*12.0.LE.STIFB .AND. WEBT.GE.TWEBR))
B      GO TO 461
      STII = 0.375
      STIW = AMAX1(2.0+WEBD/30.0,PLATEW/4.0)
      IF (AMOD(STIW,0.5).EQ.0.0) STIW = STIW-0.5
      STIW = STIW-AMOD(STIW,0.5)
450 STIW = STIW+0.5
      STIU = 25.0*WEBD**2/AMIN1(SSPAMN,STIFA,STIFB)**2-20.0

```



00000001

00000002

00000003

00000004

00000005

00000006

00000007

00000008

00000009

00000010

00000011

00000012

00000013

00000014

00000015

00000016

00000017

00000018

00000019

00000020

00000021

00000022

00000023

00000024

00000025

00000026

```

      STIIR = AMIN1(SSPAMN,STIFA,STIFB)*WEBT**3*STIJ/10.92
452 STIIA = STIT*STIW**3/3.0
      IF(STIIA.GE.STIIR) GO TO 461
      IF(STIW.GT.(PLATEW-WEBT)/2.0-0.25) GO TO 454
      IF(STIT*16.0.GE.STIW) GO TO 450
454 STIT = STIT+0.125
      IF(STIW.GT.(PLATEW-WEBT)/2.0-0.25) GO TO 452
      GO TO 450

```

C  
C

# L A T E R A L     B U C K L I N G

```

461 AMINT = (TOPPT(4)+TOPPT(5))/2.0
      PLTTOP = TOPPT(1)*AMIN1(AMINT/TOPPT(1)+0.2,1.0)
      AMINB = (BOTTP(4)+BOTTP(5))/2.0
      PLTBOT = BOTTP(1)*AMIN1(AMINB/BOTTP(1)+0.2,1.0)
      HB = (PLATEW*PLTTOP*(PLTTOP/2.+WEBD+PLTBOT)+WEBD*WEBT*(WEBD/2.+
A      PLTBOT)+PLATEW*PLTBOT**2/2.)/(PLATEW*(PLTTOP+PLTBOT)+WEBD*
B      WEBT)-PLTBOT/2.0
      HT = PLTTOP/2.0+WEBD+PLTBOT/2.0-HB

      IX = (PLATEW*(PLTTOP**3+PLTBOT**3)+WEBT*WEBD**3)/12.+PLATEW*
A      PLTTOP*HT**2+WEBT*WEBD*(HT-PLTTOP/2.-WEBD/2.)*2+PLATEW*
B      PLTBOT*HB**2
      IY = (PLATEW**3*(PLTTOP+PLTBOT)+WEBD*WEBT**3)/12.
      TURC = (PLATEW*(PLTTOP**3+PLTBOT**3)+WEBD*WEBT**3)/3.-0.21*(
A      PLTTOP**4+PLTBOT**4)
      WARPC = (PLTTOP+WEBD+PLTBOT)**2*PLTTOP*PLTBOT*PLATEW**6/(144.*IY)
      ECCEN = (HT*PLTTOP-HB*PLTBOT)*PLATEW**3/(12.*IY)

      CONSTJ = ECCEN+(HB*(PLTBOT*PLATEW**3/12.+PLTBOT*PLATEW*HB**2+
A      HB**3*WEBT/4.))-HT*(PLTTOP*PLATEW**3/12.+PLTTOP*PLATEW*
B      HT**2+HT**3*WEBT/4.))/(2.*IX)
      FCR = 1.13*3.14**2*E*IY/(IX/HT*SPANL**2*144.)*((0.45*ECCEN+0.52*
A      CONSTJ)+SQRT((0.45*ECCEN+0.52*CONSTJ)**2+WARPC/IY*(1.+E/2.6*
B      TURC*SPANL**2*144./(3.14**2*E*WARPC))))
      FSBEAM = 3.*WTFITG(1)/EXTRA*(SPANL**2/2.-CANTA**2-CANTB**2)/ST(1,1)
      FSAFTY = FCR/FSBEAM

```

C  
C

# R E A C T I O N S     +     B E A R I N G     S T I F F .

```

DO 477 K=1,2,1
      CANTX = CANTA
      SPANLX = 2.*PSLA/WTFITSL*1000.
      IF(K.EQ.2) CANTX = CANTB
      IF(K.EQ.2) SPANLX = 2.*PSLB/WTFITSL*1000.
      EMPA = (EMPACT+1.3)/2.
      LLR(1) = (.64*(SPANL+CANTX)**2/2.+(.64*SPANLX/2.+26.)*(SPANL+
A      CANTX))*EMPA/SPANL
      IF(CANTX.EQ.0.0)LLR(1)=(0.64*SPANL/2.+ 26.)*EMPACT
      IF(LOAD.EQ.2) GO TO 465
      IF(CANTX.EQ.0.0) GO TO 463
      LLR(2) = (72.*SPANL+72.*CANTX-672.)*EMPA/SPANL

```



```

      LLR(3) = (40.*SPANL-112.+32.*(SPANLX+CANTX-14.)*(SPANL+CANTX)/
A      SPANLX)*EMPA/SPANL
      LLR(4) = (32.*SPANL+((SPANLX+CANTX-14.)*32.+(SPANLX+CANTX-26.)*6.)*
A      *(SPANL+CANTX)/SPANLX)*EMPA/SPANL
      LLR(5) = (32.*(SPANL+CANTX-14.)+32.*(SPANL+CANTX)+8.*(SPANLX-14.)*
A      (SPANL+CANTX)/SPANLX)*EMPA/SPANL
      LLR(6) = ((72.*SPANLX-672.)*(SPANL+CANTX)/SPANLX)*1.3/SPANL
463 LLR(7) = (72.*SPANL-672.)*EMPACT/SPANL
      GO TO 470
465 IF(CANTX.EQ.0.0) GO TO 467
      LLR(2) = (8.*(SPANLX-14.)/SPANLX+32.)*(SPANL+CANTX)*1.3/SPANL
      LLR(3) = (5.*SPANL+5.*CANTX-14.)*8.*EMPA/SPANL
      LLR(4) = (32.*(SPANLX+CANTX-14.)/SPANLX*(SPANL+CANTX)+6.*SPANL)*
A      EMPA/SPANL
      LLR(5) = (8.*(SPANLX+CANTX-14.)/SPANLX*(SPANL+CANTX)+32.*SPANL)*
A      EMPA/SPANL
467 LLR(6) = (5.*SPANL-14.)*8.*EMPACT/SPANL
470 LLMAXR = AMAX1(LLR(1),LLR(2),LLR(3),LLR(4),LLR(5),LLR(6),LLR(7))
A      *DISTRB/EMPACT
      RSWLL = SWLL/1000.*TOTALL*(TOTALL/2.-CANTX)/SPANL
      LLRMAX(K) = AMAX1(LLMAXR,(LLMAXR+RSWLL)*0.8)
      LLRWUI(K) = AMAX1(LLR(1)/EMPA,LLR(2)/EMPA,LLR(3)/EMPA,LLR(4)/EMPA,
A      LLR(5)/EMPA,LLR(6)/1.3,LLR(7)/EMPACT)
      IF(CANTX.EQ.0.0) LLRWUI(K) = AMAX1(LLR(1)/EMPACT,LLRWUI(K))
      IF(LOAD.EQ.2) LLRWUI(K) = AMAX1(LLR(1)/EMPA,LLR(2)/1.3,LLR(3)/EMPA,
A      LLR(4)/EMPA,LLR(5)/EMPA,LLR(6)/EMPACT)
      IF(LOAD.EQ.2.AND.CANTX.EQ.0.) LLRWUI(K) = AMAX1(LLR(1)/EMPACT,
ALLRWUI(K))
      LLRWUI(K) = LLRWUI(K)*DISTRB/EMPACT
      LLRWUI(K) = AMAX1(LLRWUI(K),(LLRWUI(K)+RSWLL)*0.8)
      IF(LOAD.EQ.3) LLRMAX(K) = DISTRB*(0.085*(SPANL+CANTX)**2/2.+(
A      0.085*SPANLX/2.)*(SPANL+CANTX))/SPANL
      IF(LOAD.EQ.3) LLRWUI(K) = LLRMAX(K)
      IF(LOAD.EQ.4) LLRMAX(K) = 6.15384*LL*DISTRB*(SPANL+21.125)/100.

      IF(LOAD.EQ.4) LLRWUI(K) = LLRMAX(K)/RRIMP
      DO 472 L=1,7,1
472 LLR(L) = 0.0
      DLAVG = (WIFTSI+AVGWT*EXTRA+LTIIL+SWFWS)/1000.
      DLRT(2) = (DLAVG *TOTALL*(TOTALL/2.-CANTX)+(PBMB+PSLB+PSWB)*
A      (SPANL+CANTX)-(PBMA+PSLA+PSWA)*CANTX)/SPANL
      DLRT(1) = DLAVG*TOTALL+PBMA+PSLA+PSWA+PBMB+PSLB+PSWB-DLRT(2)
      WSTIFF = (PLATEW-WEBT)/2.-AMCUC((PLATEW-WEBT)/2.,0.25)-0.25
      TSTIFF(K) = WSTIFF/12.*SQRT(50./33.)

      IF(STRESSA.LE.23500.) TSTIFF(K) = WSTIFF/12.*SQRT(36./33.)
      TSTIFF(K) = AMAX1(IFIX(TSTIFF(K)*8.+0.999)/8.,PLMIN)-0.125
474 TSTIFF(K) = TSTIFF(K)+0.125
      BRGSTK = (LLRMAX(K)+DLRT(K))/(2.*TSTIFF(K)*(WSTIFF-0.50))

```



```

      IF (BRGSTR.GT.40.) GO TO 474
      IF (BRGSTR.GT.29. .AND. STRESA.LE. 23500.) GO TO 474
      RGYR = (2.*WSTIFF*TSTIFF(K)*((WEBT+WSTIFF)/2.)*2+2.*TSTIFF(K)
A      *WSTIFF**3/12.)/(2.*WSTIFF*TSTIFF(K)+13.*WEBT**2)
      FALLOW = 16000.-0.3*WEBD**2/RGYR
      IF (STRESA.GT.23500.) FALLOW = 22000.-0.56*WEBD**2/RGYR
      IF ((LLRMAX(K)+DLRT(K))/(2.*WSTIFF*TSTIFF(K)+13.*WEBT**2).GT.
A      FALLOW/1000.) GO TO 474
477 CONTINUE
478 CONTINUE
      RETURN
      END

```

C

```

BLOCK DATA
DESIGN  CONSTANTS
COMMON/UES1/ALLOWC(4),ALLOWE(4),MUT(12),PROBLL(4),TRATIO(2)
DATA ALLOWC/1000.,800.,0.,640./
DATA ALLOWE/350.,300.,0.,0./
DATA MUT/0.,31,60,91,121,152,182,213,244,274,305,335/
DATA PROBLL/20.,20.,85.,80./
DATA TRATIO/0.,24./
END

```

C

```

SUBROUTINE PROPIY (M1)
P R O P E R T I E S

REAL I,M
DIMENSION N(8)
COMMON BMDEL,BMELL,BMSW,BMSWLL,BMTRK,BCTIPT(6),CANTA,CANIB,CG(6,4),
A  DELTIL,DELTILB,DELTLC,DELI(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LCAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,PUSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STAIM(6),STRESA,STUDS,SWFWS,SWLL,TUPPT(6),UTIL,VDL,
E  VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WTFITG(5),WTFITSL,X,PSWA,PSWB
F  ,XL,XR,LL,FSAFETY,AVGWT,CQUELL
DATA N/999999.,999999.,999999.,999999.,999999.,8.,10.,30./
M = M1
WTFITG(M) = ((TOPPT(M)+BOTPT(M))*PLATEW+WEBD*WEBT)*3.4*EXTRA
SLABH = SLABT+HDEPTH+TUPPT(1)-TUPPT(M)
SLABW = AMIN1(GSPA,SLABT*12.,12.*SPANL/4.)
DO 8 K=1,4,1
J = K+4*IFIX(STUDS/50.+0.99)

```



```

CG(M,K) = ((SLABW-HWIDTH)*SLABT**2/(2.*N(J))+HWIDTH*SLABH**2/
A      (2.*N(J))+(TUPPT(M)*PLATEW)*(SLABH+TUPPT(M)/2.)+(WEBT*WEBD)*
B      (SLABH+TUPPT(M)+WEBD/2.)+(BOTTP(T(M)*PLATEW)*(SLABH+TUPPT(M)+
C      WEBD+BOTTP(T(M)/2.))/((SLABW-HWIDTH)*SLABT/N(J)+SLABH*HWIDTH
D      /N(J)+TUPPT(M)*PLATEW+WEBT*WEBD+BOTTP(T(M)*PLATEW)
I(M,K) = (SLABW-HWIDTH)*SLABT/N(J)*((CG(M,K)-SLABT/2.))**2+SLABT**2
A      /12.)+SLABH*HWIDTH/N(J)*((ABS(CG(M,K)-SLABH/2.))**2+SLABH**2
B      /12.)+TUPPT(M)*PLATEW*((ABS(CG(M,K)-SLABH-TUPPT(M)/2.))**2+
C      TUPPT(M)**2/12.)+(WEBT*WEBD*((ABS(CG(M,K)-SLABH-TUPPT(M)-WEBD/
D      2.))**2+WEBD**2/12.)+(BOTTP(T(M)*PLATEW*((SLABH+TUPPT(M)+WEBD+
E      BOTTP(T(M)/2.-CG(M,K))**2+BOTTP(T(M)**2/12.))
ST(M,K) = I(M,K)/(CG(M,K)-SLABH)
8 SB(M,K) = I(M,K)/(SLABH+TUPPT(M)+WEBD+BOTTP(T(M)-CG(M,K))
STATM(M) = (CG(M,1)-CG(M,3))*WFTTG(M)/(3.4*EXTRA)
RETURN
END

```

## SUBROUTINE MOMENT

```

C
C
M O M E N T S
DIMENSION FATIGC(2),FATIGT(2),R(6)
COMMON BMDL,BMLL,BMSW,BMSWL,BMTRK,BOTTP(T(6),CANTA,CANTB,CG(6,4),
A      DELTAL,DELTALB,DELTALC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B      FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGGL,PBMA,PMB,PLLA,
C      PLLB,PLATEI(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D      ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TUPPT(6),UTIL,VDL,
E      VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WFTTG(5),WFTTSL,X,PSWA,PSWB
F      ,XL,XR,LL,FSAFETY,AVGWT,CODELL
X1 = (CANTA+SPANL+CANTB)*(CANTA+SPANL-CANTB)
X2 = CANTA+X
WFTTDL = WFTTSL+WFTTG(1)+UTIL
BMDL = (WFTTDL*X1/(2000.*SPANL)+(PBMA+PSLA)*(CANTA+SPANL)/SPANL-
A      (PMB+PSLB)*CANTB/SPANL)*X-WFTTDL*X2*X2/2000.-(PBMA+PSLA)*X2
BMSW = (SWFWS*X1/(2000.*SPANL)+PSWA*(CANTA+SPANL)/SPANL
A      -PSWB*CANTB/SPANL)*X-SWFWS*X2*X2/2000.-PSWA*X2
BMSWL = SWLL*X*(SPANL-X)/2000.
NEGGL = -1.3*(PLLA*CANTA*(SPANL-X)+PLLB*CANTB*X)/SPANL
IF(LOAD.EQ.3) GO TO 6
BML1 = X*(SPANL-X)*(0.32+18./SPANL)
IF(LOAD.EQ.1) GO TO 2
BMTRK = AMAX1((SPANL-X)*(40.*X-112.),X*(40.*(SPANL-X)-112.))/SPANL
GO TO 4
2 XMAX = AMIN1(X,SPANL-X)
BMTRK = AMAX1(XMAX*(72.*(SPANL-XMAX)-672.)/SPANL,XMAX*(72.*(SPANL-
A      XMAX)-336.)/SPANL-112.,XMAX*(64.*(SPANL-XMAX)-448.)/SPANL)
4 POSLL = AMAX1(BMTRK,BML1)*DISTRB

```





```

      BMTRK = BMTRK*DISTRB
      GO TO 7
6  BMTRK = 0.0
      Z = AMIN1(X/SPANL,1.-X/SPANL)
      POSLL = 10.*LL*DISTRB/65.*(3.099071*SPANL+0.0730426*SPANL**2+
      3.8/10**9*SPANL**5-28.3621)*Z*(4.16-4.32*Z)
      IF(LOAD.EQ.3) POSLL = 0.085*DISTRB*X/2.*(SPANL-X)
      IF(LOAD.EQ.3) NEGLL = NEGLL/1.3
7  CONTINUE
      RETURN
      END

```

```

SUBROUTINE STRESS(M1)
  DIMENSION FATIGC(2),FATIGT(2),R(6)
  COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,HOTTP(6),CANTA,CANID,CG(6,4),
  A  DELTLA,DELTLC,DELTLC,DELTLC(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
  B  FS(4),GSPA,HDEPTH,HWIDTH,1(6,4),LOAD,M,NEGLL,PBMA,POMB,PLLA,
  C  PLLB,PLALET(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
  D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VDL,
  E  VLL,VLLMIN,VSW,VITAL,WEBD,WEBT,WTFTG(5),WTFISL,X,PSWA,PSWB
  F  ,XL,XR,LL,FSAFETY,AVGWT,CQUELL
  M = M1
  N1 = 3
  N2 = 4
  IF(X/SPANL.LE.XL .OR. X/SPANL.GE.XR) N1 = 1
  IF(X/SPANL.LE.XL .OR. X/SPANL.GE.XR) N2 = 1
  TOPPOS = (BMDL/ST(M,1)+BMTRK/ST(M,N1)+BMSW/ST(M,N2))*12000.
  TOPNEG = (-BMDL/ST(M,1)-NEGLL/ST(M,N1)-BMSW/ST(M,N2))*12000.
  R(1) = -1.0*TOPPOS/TOPNEG
  BOTPOS = (BMDL/SB(M,1)+BMTRK/SB(M,N1)+BMSW/SB(M,N2))*12000.
  BOTNEG = (-BMDL/SB(M,1)-NEGLL/SB(M,N1)-BMSW/SB(M,N2))*12000.
  R(2) = -1.0*BOTPOS/BOTNEG
  DO 8 J=1,2,1
  FATIG(J) = 0.0
  IF (ABS(R(J)).GT.1.0) R(J) = 1.0/R(J)
  IF (STRESA.GT.23500.) FATIGC(J) = 27000./ABS(1.-1.43*R(J))
  IF (STRESA.LE.23500.) FATIGC(J) = 20000./(1.-0.89*R(J))
  IF (STRESA.GT.23500.) FATIGT(J) = 18000./(1.-0.62*R(J))
  IF (STRESA.LE.23500.) FATIGT(J) = 17200./(1.-0.62*R(J))
8  CONTINUE
  IF (TOPPOS.GT.FATIGC(1) .OR. TOPNEG.GT.FATIGT(1)) FATIG(1) = 1.0
  IF (BOTPOS.GT.FATIGT(2) .OR. BOTNEG.GT.FATIGC(2)) FATIG(2) = 1.0
  FS(1) = AMAX1(TOPNEG,TOPPOS+(POSLL-BMTRK)/ST(M,N1)*12000.,
  A  (TOPPOS+(POSLL-BMTRK+BMSWLL)/ST(M,N1)*12000.)*0.8)
  FS(2) = AMAX1(BOTNEG,BOTPOS+(POSLL-BMTRK)/SB(M,N1)*12000.,
  A  (BOTPOS+(POSLL-BMTRK+BMSWLL)/SB(M,N1)*12000.)*0.8)
  RETURN
  END

```

SUBROUTINE DELTA (BOTPL1,BOTPL2,BOTPL3,TOPPL1,TOPPL2,TOPPL3)

## D E F L E C T I O N S

```

REAL IDEF,I1,I2,I
DIMENSION BMD(21),IDEF(42),P(21),PL(4),PR(4),V(21),WT(4)
COMMON BMDL,BMLL,BMSW,BMSWLL,BMIRK,BOTTP(6),CANTA,CANTB,CG(6,4),
A  DELTLA,DELTBL,DELTCL,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,PSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VDL,
E  VLL,VLLMIN,VSW,VTOTAL,WEBU,WEBT,WFTIG(5),WFTISL,X,PSWA,PSWB
F  ,XL,XR,LL,FSAFTY,AVGWT,CDELL
SWFACT = (SWFWS-20.*GSPA/12.)/(SWFWS+0.01)
IF (SWFACT.LT.0.0) SWFACT = 0.0
DO 40 K1=1,4,1
DO 1 K=1,21,1
1 P(K) = 0.0
GO TO (2,6,10,14),K1
2 DO 4 K=1,21,1
X = (K-1)*SPANL/20.
CALL FINDM (TOPPL1,TOPPL3,BOTPL1,BOTPL3)
IDEF(K) = I(M,1)
IDEF(K+21) = I(M,4)
4 P(K) = (WFTIG(M)+UTIL)*SPANL/20.
BMD(1) = -PBMA*CANTA*1000.-WFTIG(4)*CANTA**2/2.
BMD(21) = -PBMB*CANTB*1000.-WFTIG(5)*CANTB**2/2.
PL(1) = PBMA
PR(1) = PBMB
WT(1) = WFTIG(1)
GO TO 10
6 BMD(1) = -PSLA*CANTA*1000.
BMD(21) = -PSLB*CANTB*1000.
PL(2) = PSLA
PR(2) = PSLB
GO TO 10
10 BMD(1) = -WFTISL*CANTA**2/2.
BMD(21) = -WFTISL*CANTB**2/2.
DO 12 K=1,21,1
12 P(K) = WFTISL*140./150.*SPANL/20.
WT(3) = WFTISL*140./150.
GO TO 10
14 BMD(1) = -PSWA*SWFACT*CANTA*1000.
BMD(21) = -PSWB*SWFACT*CANTB*1000.
DO 16 K=1,21,1
P(K) = SWFWS*SWFACT*SPANL/20.
16 IDEF(K) = IDEF(K+21)
IDEF(1) = I(4,1)
IDEF(21) = I(5,1)
PL(4) = PSWA*SWFACT

```

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

11/11/11

```

      PR(4) = PSWB*SWFACT
18  V(1) = (-BMD(1)+BMD(21))/SPANL
      DO 20 K=1,19,1
20  V(1) = V(1)+P(K+1)*(20-K)/20.
      DO 24 K=1,19,1
24  V(K+1) = V(K)-P(K+1)
      DO 26 K=1,19,1
26  BMD(K+1) = BMD(K)+V(K)*SPANL/20.
      RL = 0.0
      RR = 0.0
      DO 30 K=1,20,1
      RL = RL+(41-2*K)*(BMD(K)/IDEF(K)+BMD(K+1)/IDEF(K+1))*0.09*SPANL
30  RR = RR+(K+2-1)*(BMD(K)/IDEF(K)+BMD(K+1)/IDEF(K+1))*0.09*SPANL
      XM = 0.0
      DO 34 K=1,10,1
34  XM = XM+(21-2*K)*(BMD(K)/IDEF(K)+BMD(K+1)/IDEF(K+1))*0.09*SPANL
      DELT(1,K1) = -CANTA*12./29000000.*(RL-WI(K1)*18.*CANTA**3/IDEF(1)
      A      -PL(K1)*CANTA**2*48000./IDEF(1))
      DELT(2,K1) = (RL/2.-XM)*SPANL*12./29000000.
40  DELT(3,K1) = -CANTB*12./29000000.*(RR-WT(K1)*18.*CANTB**3/IDEF(21)
      A      -PR(K1)*CANTB**2*48000./IDEF(21))
      TOPPL = (TOPPL1+TOPPL3)/2.
      BOTPL = (BOTPL1+BOTPL3)/2.
      I1 = (I(2,2)+I(3,2))/2.
      I2 = (I(4,2)+I(5,2))/2.
      EQVI8 = 1./(1./I(1,2)+14.0/SPANL**4*(TOPPL**3*(SPANL-TOPPL)/I(1,2)
      A      *(I(1,2)/I1-1.)+BOTPL**3*(SPANL-BOTPL)/I1*(I1/I2-1.)))
      DELTL1 = .000496*(SPANL+45.)*SPANL**3
      IF (LOAD.EQ.1) DELTL2 = .0894*(SPANL**3-555.*SPANL+4780.)
      IF (LOAD.EQ.2) DELTL2 = .0496*(SPANL**3-200.*SPANL+1610.)
      DELTLC = AMIN1(9999.,EQVI8*SPANL*12./(AMAX1(DELT1,DELT2)*DISTRB))
      W1 = (WFTG(4)+WFTSL+UIIL+SWFWS)/1000.
      DL = SPANL+CANTA-((W1*(SPANL**2-CANTA**2)-2.*(PBMA+PSLA+PSWA)*
      A      CANTA)/(W1*SPANL))*3/SPANL**2*(1.-I(4,1)/EQVI8)
      DELTLA = AMIN1(9999.,CANTA*604.*I(4,1)/(PLLA*1.3*CANTA**2*DL+.001))
      W1 = (WFTG(5)+WFTSL+UIIL+SWFWS)/1000.
      DR = SPANL+CANTB-((W1*(SPANL**2-CANTB**2)-2.*(PBMH+PSLB+PSWB)*
      A      CANTB)/(W1*SPANL))*3/SPANL**2*(1.-I(5,1)/EQVI8)
      DELTLB = AMIN1(9999.,CANTB*604.*I(5,1)/(PLLB*1.3*CANTB**2*DR+.001))
      IF(LOAD.EQ.3) DELTLC = AMIN1(12.*EQVI8*384.*29000000./(5.*1728.*
      A      SPANL**3*LL*DISTRB),9999)
      IF(LOAD.EQ.3) DELILA = DELTLA*1.3
      IF(LOAD.EQ.3) DELTLB = DELTLB*1.3
      IF(LOAD.EQ.4) DELTLC = SPANL*12.*EQVI8/((107.6-9.09*SPANL+0.234*
      A      SPANL**2+6.43E-5*SPANL**4+1.67E-7*SPANL**5)*LL*DISTRB)
      RETURN
      END

```



## SUBROUTINE SHEAR (VN)

## S H E A R S

```

COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,HOTTP(6),CANTA,CANTB,CG(6,4),
A  DELTLA,DELTBL,DELTCL,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,PUSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TUPP(6),UTIL,VDL,
E  VLL,VLLMIN,VSW,VTOTAL,WEBD,WEFT,WTFTG(5),WTFISL,X,PSWA,PSWB
F  ,XL,XR,LL,FSAFETY,AVGWT,CUDELL

VX = AMIN1(VN,8-VN)
IF (LOAD.EQ.2) VLL = AMAX1(5.0*(8.-VX-22.4/SPANL),(8.-VX)**2*
A  0.005*SPANL+26.0-3.25*VX)*DISTRB
IF (LOAD.EQ.1) VLL = AMAX1(9.0*(8.-VX-74.7/SPANL),(8.-VX)**2*
A  0.005*SPANL+26.0-3.25*VX)*DISTRB
IF (LOAD.EQ.3) VLL = 0.085*DISTRB*(4.-VX)*SPANL/8.
IF (LOAD.EQ.4) VLL = 6.15384*LL*DISTRB*((8.-VX)*SPANL/8.+
A  21.125)/100.
X1 = (CANTA+SPANL+CANTB)*(CANTA+SPANL-CANTB)/2000.
VLDL = ((WTFISL+WTFTG(1)+UTIL)*X1+(PBMA+PSLA)*(CANTA+SPANL)-
A  (PBMB+PSLB)*CANTB)/SPANL
VLSW = (SWFWS*X1+PSWA*(CANTA+SPANL)-PSWB*CANTB)/SPANL
VLSWLL = SWLL*SPANL/2000.
VDL = ABS(VLDL-PBMA-PSLA-(WTFISL+WTFTG(1)+UTIL)*(CANTA+VN/8.*
A  SPANL)/1000.)
VSW = ABS(VLSW-PSWA-SWFWS*(CANTA+VN/8.*SPANL)/1000.)
VSWLL = ABS(VLSWLL-VN/8.*SWLL*SPANL/1000.)
VTOTAL = AMAX1(VDL+VLL+VSW,(VDL+VLL+VSWLL+VSW)*0.8)
VLL = AMAX1(VLL,(VLL+VSWLL)*0.8)
VLLMIN = AMAX1(4.*VX,5.*VX-112./SPANL,3.25*VX+.005*VX**2*SPANL)
IF (LOAD.EQ.1) VLLMIN = AMAX1(VLLMIN,8.*VX-448./SPANL,9.*VX-672./
A  SPANL)
IF (LOAD.LE.4) VLLMIN = AMAX1(VLLMIN*DISTRB,PLLB*1.3*CANTB/SPANL)
IF (LOAD.GE.5) VLLMIN = AMAX1(VLLMIN*DISTRB,PLLA*1.3*CANTA/SPANL)
RETURN
END

```

## SUBROUTINE FINDM (TOPPL1,TOPPL3,BOTPL1,BOTPL3)

## FIND M (ORDER LEFT TO RIGHT 4,2,1,3,5)

```

COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,HOTTP(6),CANTA,CANTB,CG(6,4),
A  DELTLA,DELTBL,DELTCL,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,PUSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TUPP(6),UTIL,VDL,
E  VLL,VLLMIN,VSW,VTOTAL,WEBD,WEFT,WTFTG(5),WTFISL,X,PSWA,PSWB
F  ,XL,XR,LL

M = 1

```





```

IF(X+CANTA,LE.TOPPL1 .AND. X+CANTA,LE.BOTPL1) M = 4
IF(X+CANTA,LE.TOPPL1 .AND. X+CANTA,GE.BOTPL1) M = 2
IF(X+CANTA,GE.TOPPL1 .AND. X+CANTA,LE.BOTPL1) M = 2
IF(SPANL+CANTB-X,LE.TOPPL3 .AND. SPANL+CANTB-X,LE.BOTPL3) M = 5
IF(SPANL+CANTB-X,LE.TOPPL3 .AND. SPANL+CANTB-X,GE.BOTPL3) M = 3
IF(SPANL+CANTB-X,GE.TOPPL3 .AND. SPANL+CANTB-X,LE.BOTPL3) M = 3
RETURN
END

```

## SUBROUTINE DETAIL

```

COMMON BMDL,BMLL,BMSW,BMSWL,BMTRK,BOTPT(6),CANTA,CANTB,CG(6,4),
A DELTLA,DELTLC,DELTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C PLLB,PLATET(2),PLATEW,POSLL,PSLA,PSLB,SLABX,SR(6,4),SLABT,SPANL
D ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VDL,
E VLL,VLLMIN,VSW,VTOTAL,WEBD,WEHT,WFTG(5),WFTSL,X,PSWA,PSWB
F ,XL,XR,LL,FSAFTY,AVGWT,CODELL
COMMON/DES1/ALLOWC(4),ALLOWE(4),MOT(12),PROBLL(4),TRATIO(2)
COMMON/DES2/ADJ(4),DLRT(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),
A TSTIFF(2),TWEB(9),LLRWDI(2)
COMMON/BRDATA/BRL,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMOD,FR,C1,C2,
B C3,GRND,GCOST,GCOSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
C OPTCAN(10),P(4),TUTIL,TSWL,RAILWT,REINF(2),ITER,FSAFTI,WEBDI,
D UBWEBD,TPT(1),BPT(1),IDET
COMMON/DET/TOPPL1,TOPPL2,TOPPL3,BOTPL1,BOTPL2,BOTPL3,XCENT,
A FTOPL,FSTOP,FTOPR,FROTTL,FSROTT,FROTTT,BMDLA,BMDLC,BMDLB,BMLLA,
B BMLLC,BMLLB,BMSWA,BMSWC,BMSWB,WSTIFF,STUD,STIFA,PT,STIT,STIW,
C STIFR,STIFB,STIFL

```

## OUTPUT

```

125 FORMAT(1H1,////////)
A 20X,33HTWO-SPAN CANTILEVER BRIDGE DESIGN,
B 15X,5HSQUAD,3X,8HKULKARNI,/)
101 FORMAT(20X,16HBRIDGE DIMENSIONS,16X,20HMATERIAL PROPERTIES,/)
102 FORMAT(20X,16HBRIDGE LENGTH =,F10.2,2X,3HFT.,3X,12HFS(REINF.) =,
A F8.0,2X,4HPSI.,/20X,16HBRIDGE WIDTH =,F10.2,2X,3HFT.,3X,
B 12HFC(CONC.) =,F8.0,2X,4HPSI.)
103 FORMAT(20X,16HSPAN 1 LENGTH =,F10.2,2X,3HFT.,3X,12HFS(STEEL) =,
A F8.0,2X,4HPSI.,/20X,16HSPAN 2 LENGTH =,F10.2,2X,3HFT.,3X,
B 12HMOD. RATIO =,F8.0)
104 FORMAT(20X,16HROADWAY WIDTH =,F10.2,2X,3HFT.,//20X,
A 10HLOAD DATA,/)
105 FORMAT(20X,16HLOAD(1=HS 2=H) =,I7,/,20X,16HRAILING WT. =,F10.2,
A 2X,8HLBS./FT.,/20X,16HSDWALK L. LOAD =,F10.2,2X,8HLBS./FT.)
106 FORMAT(20X,16HFUT. WEARING =,5X,18H25 LBS./SQ.FT.,/
A 20X,16HTOTAL UTILITY =,F10.2,2X,8HLBS./FT.,//
B 20X,10HCOST DATA,/)
107 FORMAT(20X,16HCOST OF CONC. =,F10.2,2X,8H$/CU.YD.,/
A 20X,16HCOST OF REINF. =,F10.2,2X,5H$/LR.,/
B 20X,16HCOST OF STEEL =,F10.2,2X,5H$/LR.,//20X,
C 17HDESIGN VARIABLES,/)

```



```

108 FORMAT(20X,16HGIRDER SPACING =,F10.2,2X,3HIN.,/
A 20X,16HNO. OF GIRDERS =,F8.0)

109 FORMAT(20X,16HDEPTH OF WEB =,F10.2,2X,3HIN.,/
A 20X,16HUPPER BND. WEB =,F10.2,2X,3HIN.,/
B 20X,16HCANT. LENGTH =,F10.2,2X,3HFT.)

110 FORMAT(20X,16HFLANGE PL WIDTH=,F10.2,2X,3HIN.,//
A 20X,12HSLAB DESIGN,/)

115 FORMAT(20X,16HSLAB THICK. =,F10.2,2X,3HIN.,/
A 20X,16HTRANS. REINF. =,F10.2,2X,10HSQ.IN./FT.,/)

120 FORMAT(20X,14HGIRDER DESIGN,/)
121 FORMAT(20X,16HAVEWT. SPAN 1 =,F8.0,4X,8HLLBS./FT.)
122 FORMAT(20X,16HAVEWT. SPAN 2 =,F8.0,4X,8HLLBS./FT.)
123 FORMAT(20X,16HSAFTY SPAN 1 =,F10.2,/
A 20X,16HSAFTY SPAN 2 =,F10.2,/)

WRITE(3,125)
WRITE(3,101)
WRITE(3,102) BRL,FR,BRW,FC
WRITE(3,103) SPANL1,STRESA,SPANB,RMOD
WRITE(3,104) ROW
WRITE(3,105) LOAD,RAILWT,TSWLL

WRITE(3,106) UTIL
WRITE(3,107) C1,C2,C3
WRITE(3,108) GSPA,GRNO
WRITE(3,109) WEBD,UBWEBD,CANTB
WRITE(3,110) PLATEW
WRITE(3,115) SLART,REINF(2)
WRITE(3,120)
WRITE(3,121) WG(1)
WRITE(3,122) WG(2)
WRITE(3,123) FSAFTY,FSAFTI
RETURN
END

```

# SUBROUTINE DETGIR

```

COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BOTTP(6),CANTA,CANTB,CG(6,4),
A DELTLA,DETLR,DETLTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PRMR,PLLA,
C PLLB,PLATFT(2),PLATEW,POSIL,PSLA,PSLB,SLABW,SB(6,4),SLART,SPANL
D ST(6,4),STATH(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VOL,
E VLL,VLLMIN,VSW,VTOTAL,WERO,WERT,WTFTG(5),WTFTSL,X,PSWA,PSWB
F ,XL,XR,LL,FSAFTY,AVGWT,CODELL

COMMON/DES1/ALLOWC(4),ALLOWE(4),MOT(12),PROBLL(4),TRATIO(2)
COMMON/DES2/ADJ(4),DLRT(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),

```



```

A   TSTIFF(2),TWEB(9),LLRWDL(2)
COMMON/BRDATA/BRH,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMOD,FR,C1,C2,
B   C3,GRNG,GCCST,GCCSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
C   OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFTI,WBOD1,
D   WBWBD,TPT(1),BPT(1),IDEL
COMMON/DET/TOPPL1,TOPPL2,TOPPL3,BOTPL1,BOTPL2,BOTPL3,XCENT,
A   FTOPL,FSTOP,FUPR,FBUPL,FSBUT,FBOTIR,BMDLA,BMDLC,BMDLB,BMLLA,
B   BMLLC,BMLLB,BMSWA,BMSWC,BMSWB,WSTIFF,S100,S1IFA,PI,STIT,STIW,
C   STIFR,STIFB,STIFL
1  FORMAT(6X,1A8,6X,F4.0)
2  FORMAT(6X,7F10.4)
3  FORMAT(6X,F10.4,F7.1,I3,5F10.4)
4  FORMAT(1H4,19X,23HP L A T E   G I R D E R,16X,5HSGUAD,6X,
A8HKULKARNI,7724X,9HPRG. NU.,FS.0)
16 FORMAT (///17X,44HFACTOR OF SAFETY AGAINST LATERAL BUCKLING =,
A   F5.1,7722X,37H I N F U I   V E R I F I C A T I O N,7712X,
B   10HWEB THICK.,F13.4,12X,14HS.W. LIVE LOAD,F9.0)
17 FORMAT (12X,9HWEB DEPTH,F14.2,12X,13HS.W. + F.W.S.,F10.0,
A   /12X,12HFLANGE WIDTH,F11.2,12X,14HCANT. LENGTH A,F9.2)

19 FORMAT (12X,11HSLAB THICK.,F12.2,12X,8HP (BEAM),F15.1/12X,
A   11HGIRDER SPA.,F12.2,12X,8HP (SLAB),F15.1/12X,11HSPAN LENGTH,
B   F12.2,12X,13HP (LIVE LOAD),F10.1)
23 FORMAT (12X,9HLIVE LOAD,8X,F5.0,I3,12X,12HP (SW + FWS),F11.1,
A   /12X,12HDIST. FACTOR,F11.2,12X,14HCANT. LENGTH B,F9.2)
25 FORMAT (12X,12HHAUNCH DEPTH,F11.2,12X,8HP (BEAM),F15.1,
A   /12X,12HHAUNCH WIDTH,F11.2,12X,8HP (SLAB),F15.1)
27 FORMAT (12X,13HDFSIGN STRESS,F10.0,12X,13HP (LIVE LOAD),F10.1,
A   /12X,9HUTILITIES,F14.0,12X,12HP (SW + FWS),F11.1)
29 FORMAT (36X,F5.2,1HL,F10.0)
31 FORMAT (12X,21HS T I F F .   S P A .,4X,5H0.50L,F10.0)
32 FORMAT (15X,1H(,F5.3,3H" X,F4.1,2H"),7X,5H0.62L,F10.0)
34 FORMAT ( //47X,18HLENGTH   THICK.,19X,1H1,/)
36 FORMAT (15X,18HT U P   P L A T E,F19.1,F12.3)

38 FORMAT (42X,F10.1,F12.3)
40 FORMAT (15X,23H B U T T O M   P L A T E,F14.1,F12.3)
42 FORMAT (//52X,6HLT BRG,4X,F4.2,1HL,5X,6HRT BRG,715X,9HS T R E S,
A   6H S E S,5X,3HTUP,F20.0,2F10.0,735X,4HBUIT,F19.0,2F10.0)
45 FORMAT (//35X,9HDEAD LOAD,F14.0,2F10.0,715X,13HM O M E N T S,7X,
A   9HLIVE LOAD,F14.0,2F10.0,735X,8HSW + FWS,F15.0,2F10.0)
48 FORMAT (//35X,9HDEAD LOAD,F14.1,F20.1/15X,17HR E A C T I O N S,3X,
A   12HL. L. W/IMP,F11.1,F20.1/21X,5H(MAX),9X,12HL. L. WU/IMP,
B   F11.1,F20.1/35X,9HBRG STIFF,F10.3,2H X,F5.2,F13.3,2H X,F5.2)
52 FORMAT (///50X,6HLT PIN,6X,2HCL,6X,6HRT PIN,732X,4HBEAM,F19.2,2F
A   10.2/32X,15HSLAB ADJ. SPANS,F8.2,2F10.2/12X,15HD E F L E C T .,
B   5X,14HSLAB THIS SPAN,F9.2,2F10.2/32X,8HSIDEWALK,F15.2,2F10.2)

57 FORMAT (32X,10HDELTA LL/L,7X,2(2H1/,I4,4X),2H1/14///)
58 FORMAT(32X,10HDELTA LL/L,17X,2H1/,I4,4X,2H1/,I4///)

```



```

59 FORMAT(32X,10HDELTA LL/L,7X,2H1/,14,4X,2H1/,14///)
60 FURMAT (32X,10HDELTA LL/L,17X,2H1/,14///)
61 FORMAT (/34X,17HC U M P D S I T E,8X,13HS E C I I U N/,36X,
A      4HN=10,26X,4HN=30/,21X,2(7X,1H,6X,16HS(TUP) S(BUIT)))

64 FORMAT ( 16X,6HCENTER,F10.0,2F9.0,F12.0,2F9.0)
65 FURMAT (22X,F10.0,2F9.0,F12.0,2F9.0)
66 FORMAT (12X,7HS T U D,3X,13HS P A C I N G,18,4H AT,14,1H",
A      5X,9HSIARTS AT,F6.2,1HL/,57X,9HENUS AT,F6.2,1HL///)

70 FORMAT (/21X,14HAVG. WT./FT. =,F7.0,////33X,11H I R D L R,18X,
A      16HCUMP. SECT. N=8/,28X,1H,6X,6HS(TUP),3X,7HS(BUIT),19X,1H)
72 FORMAT (37X,5HCANT.,F10.0)
74 FORMAT ( 16X,6HCENTER,F10.0,2F9.0,14X,F10.0)
75 FORMAT (22X,F10.0,2F9.0,14X,F10.0)
80 FORMAT (/12X,21HS T I T L E . S.P.A.,,10X,
A      22HNUNE REQD BETWEEN BRGS)
81 FORMAT (43X,14HLEFT CANT. ARM,F9.0,2HIN)
82 FORMAT (43X,15HRIGHT CANT. ARM,F8.0,2HIN)
83 FORMAT (47X,2H(F5.3,3HINX,F4.1,2HIN)
C      1 S T O U T P U T S H E E T

```

```

PROB=1.
WRITE(3,4) PRGR
WRITE(3,70) AVGW
IF(STUDS.EQ.0.) GO TO 480
IF(BCTPL1.NE.0.)WRITE(3,75)I(4,1),ST(4,1),SB(4,1),I(4,2)
IF(TOPPL1.NE.0.)WRITE(3,75)I(2,1),ST(2,1),SB(2,1),I(2,2)
WRITE (3,74) I(1,1),ST(1,1),SB(1,1),I(1,2)
IF(TOPPL3.NE.0.)WRITE(3,75)I(3,1),ST(3,1),SB(3,1),I(3,2)
IF(BCTPL3.NE.0.)WRITE(3,75)I(5,1),ST(5,1),SB(5,1),I(5,2)
GO TO 482
480 IF(BCTPL1.NE.0.)WRITE(3,75)I(4,1),ST(4,1),SB(4,1)
IF(TOPPL1.NE.0.)WRITE(3,75)I(2,1),ST(2,1),SB(2,1)
WRITE(3,74) I(1,1),ST(1,1),SB(1,1)
IF(TOPPL3.NE.0.)WRITE(3,75)I(3,1),ST(3,1),SB(3,1)
IF(BCTPL3.NE.0.)WRITE(3,75)I(5,1),ST(5,1),SB(5,1)
GO TO 490
482 WRITE(3,61)
IF(BCTPL1.NE.0.)WRITE(3,65)(I(4,K),ST(4,K),SB(4,K),K=3,4)
IF(TOPPL1.NE.0.)WRITE(3,65)(I(2,K),ST(2,K),SB(2,K),K=3,4)
WRITE(3,64) I(1,3),ST(1,3),SB(1,3),I(1,4),ST(1,4),SB(1,4)
IF(TOPPL3.NE.0.)WRITE(3,65)(I(3,K),ST(3,K),SB(3,K),K=3,4)
IF(BCTPL3.NE.0.)WRITE(3,65)(I(5,K),ST(5,K),SB(5,K),K=3,4)
490 WRITE(3,34)
IF(TOPPL1.NE.0.)WRITE(3,38) TUPPL1,ICPPI(4)
WRITE(3,36) TUPPL2,TUPPI(1)
IF(TOPPL3.NE.0.)WRITE(3,38) TUPPL3,ICPPI(5)
WRITE(3,38)
IF(BCTPL1.NE.0.)WRITE(3,38)BCTPL1,BCTTPT(4)

```





```

WRITE(3,40) BOTPL2,BOTIPT(1)
IF(BOTPL3.NE.0.)WRITE(3,38)BOTPL3,BOTIPT(5)
XCENT = XCENT/SPANL
WRITE(3,42) XCENT,FTOPL,FSICP,FTOPR,FBOTIL,FSBOTI,FBOTIR
WRITE(3,45) BMOLA,BMOLC,BMOLB,BMLLA,BMLLC,BMLLB,BMSWA,BMSWC,BMSWB
WRITE(3,48) DLRT(1),DLRT(2),LLRMAX(1),LLRMAX(2),LLRWUI(1),
A LLRWUI(2),ISTIFF(1),WSTIFF,TSTIFF(2),WSTIFF

```

## 2 N D O U T P U T S H E E T

```

WRITE(3,4) PROB
WRITE(3,52) ((DELT(K,L),K=1,3),L=1,4)
IF(CANTA.NE.0. .AND. CANTB.NE.0.)WRITE(3,57)DELTLC,DELTLC,DELTLC
IF(CANTA.NE.0. .AND. CANTB.NE.0.)WRITE(3,59)DELTLC,DELTLC
IF(CANTA.EQ.0. .AND. CANTB.NE.0.)WRITE(3,58)DELTLC,DELTLC
IF(CANTA.EQ.0. .AND. CANTB.EQ.0.)WRITE(3,60)DELTLC
IF(STUDS.EQ.0.)WRITE(3,66)STUDS,STUD,XL,XK
IF(STIFL.EQ.0.0 .AND. STIFR.EQ.1.0) GO TO 498
IF (CANTA*12.0.GT.STIFA .AND. WEBT.LT.TWEBL)WRITE(3,72) STIFA
DO 492 J=1,4,1
PT = (J-1)/8.+.001
IF (PT.GT.STIFL)WRITE(3,29) PI
IF (PT.LE.STIFL .AND. STIFL.GT.0.005)WRITE(3,29) PT,STIFF(J)
492 IF (STIFL.GT.PT .AND. STIFL.LT.PT+0.125 .AND. STIFL.NE.0.5)
AWRITE(3,29) STIFL,STIFF(5)
IF (STIFL.LT.0.5)WRITE(3,31)
IF (STIFL.EQ.0.5) WRITE(3,31) STIFF(5)
IF (STIFR.GT.0.5)WRITE(3,32) STIT,STIW
IF (STIFR.EQ.0.5)WRITE(3,32)STIT,STIW,STIFF(6)
DO 496 J=6,9,1
PT = (J-1)/8.
IF (STIFR.GE.PT .AND. J.NE.6)WRITE(3,29) PI
IF (STIFR.GT.PT .AND. STIFR.LT.PT+.125 .AND. STIFR.LT.0.995)
AWRITE(3,29) STIFR,STIFF(6)
496 IF (STIFR.LT.PT .AND. J.NE.6)WRITE(3,29) PT,STIFF(J)
IF (CANTB*12.0.GT.STIFB .AND. WEBT.LT.TWEBR)WRITE(3,72) STIFB
GO TO 499
498 WRITE(3,80)
IF (CANTA*12.0.GT.STIFA .AND. WEBT.LT.TWEBL) WRITE(3,81) STIFA
IF (CANTB*12.0.GT.STIFB .AND. WEBT.LT.TWEBR) WRITE(3,82) STIFB
IF ((CANTA*12.0.GT.STIFA .AND. WEBT.LT.TWEBL) .OR. (CANTB*12.0
A .GT.STIFB .AND. WEBT.LT.TWEBR)) WRITE(3,83) STIT,STIW
499 WRITE(3,16) FSAFTY,WEBT,SWLL
WRITE(3,17) WEBD,SWFWS,PLATEW,CANTA
WRITE(3,19) SLABT,PBMA,GSPA,PSLA,SPANL,PLLA
WRITE(3,23) CODELL,LL,PSWA,CISTLL,CANTB
WRITE(3,25) HDEPTH,PBMB,HWIDTH,PSLB
WRITE(3,27) STRESA,PLLB,UTIL,PSWB
RETURN
END

```



```

SUBROUTINE FSUES
COMMON BMDL,BMLL,BMSW,BMSWLL,BMIRK,BCTTFT(6),CANTA,CANIB,CU(6,4),
A  DELTLA,DELTLB,DELTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,PAIIG(2),
B  FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PBMB,PLLA,
C  PLLB,PLATET(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D  ,ST(6,4),STATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VDL,
E  VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WTF TG(5),WTFISL,X,PSMA,PSWB
F  ,XL,XR,LL,FSAFETY,AVGWT,CODELL
COMMON/DES1/ALLOWC(4),ALLOWE(4),MDT(12),PROBLL(4),TRAIIG(2)
COMMON/DES2/ADJ(4),ULRT(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),
A  TSTIFF(2),TWEH(9),LLRWI(2)
COMMON/BRDATA/BRK,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMODU,FR,C1,C2,
B  C3,GRND,GCCST,GCCST1,WG(2),OPTWEH(10),OPTSPA(10),OPTPL(10),
C  OPTCAN(10),F(4),TUTIL,ISWLL,RAILWT,REINF(2),ITER,FSAFETY,WEBDI,
D  UBWEBD,TPT(1),BPT(1),IDEL
COMMON/DET/TOPPL1,TOPPL2,TOPPL3,BOTPL1,BOTPL2,BOTPL3,XCENT,
A  FTOPL,FSTOP,FTOPR,FBOTL,FSBOT,FBOTR,BMDLA,BMDLC,BMDLB,BMLLA,
B  BMLLC,BMLLB,BMSWA,BMSWC,BMSWB,WSTIFF,STUD,STIFA,PI,STIT,STIW,
C  STIFR,STIFB,STIFL
COMMON/GRID/UBGRND,GRNULB,NG,CANTLB,CANTUB,WEBDUB,WEBDLB,ND,
A  PLWLB,PLWUB,IXG,IYC,IXGC,IYCC,IXH,IYB,IXHC,IYBC,ZGC(5,5),
B  CSLAB,CREINF,CSTEEL,BCOST,ZHB(5,5),SAFTY(5,5),PLW(5,5),
C  WH(5,5),DPTH(5,5),OPTB(5,5),OPTG(5,5),OPTC(5,5),SAFTI(5,5)
COMMON/SRCH/BCGST,BGRND,BCANT,BWEBD,BPLATE,DATUM,GH,CP
COMMON/LBC/CANT,ZMINC
60 FORMAT(20X,4HMAIN,4F10.2,2X,2F10.2)
61 FORMAT(20X,13H A LOWER BOUND,2X,2F10.0)
C  TO DETERMINE A LOWER BOUND FOR SUPERSTRUCTURE COST
C  SLAB DESIGN AS PER AASHO 1.3.2 AND
C  GIRDER DESIGN - WEB PLATE SATISFIES SHEAR AND LOCAL BUCKLING
C  FLANGE PLATE SATISFIES BENDING STRESS ONLY
C  EXHAUSTIVE SEARCH TO DETERMINE A LOWER BOUND
GRND = UBGRND
DATUM = 1000000000.
STUDS = 4.0
EXTRA = 1.18
PLATEW = 14.
GSPA = (BRW - 2.*2.5)*12./(GRND - 1.)
CALL SLAB
CALL PRELIM
DO 120 IG = 1,5
CALL MINC
IF(BCOST.LI. DATUM) GO TO 90
GO TO 95
90 DATUM = BCOST
BGRND = GRND
BCANT = CANIB
BWEBD = WEBD
BPLATE = PLATEW

```

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10

10-10-10-10-10



```

95 GRND = GRND - 1.
   GSPA = (BRN - 2.*2.5)*12./(GRND - 1.)
   IF (GRND.LT.GRNDLB) GO TO 125
   CALL SLAB
   CALL PRELIM
120 CONTINUE
125 WRITE(3,60) BGRND,BCANT,BWEBD,BPLATE,DATUM,BCUST
   THE VALUE OF EPSILON IS 11 PERCENT OF A LOWER BOUND
   ZMINC = DATUM*1.11
   WRITE(3,61) ZMINC,DATUM
   RETURN
END

```

## SUBROUTINE MINBM

```

COMMON BMDL,BMLL,BMSW,BMSWLL,BMTRK,BCTTPT(6),CANTA,CANTB,CG(6,4),
A DELTLA,DELTLB,DELTLC,DELT(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B FS(4),GSPA,HUEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,POMB,PLLA,
C PLLB,PLATET(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLABT,SPANL
D ,ST(6,4),SIATM(6),STRESA,STUDS,SWFWS,SWLL,TOPPT(6),UTIL,VDL,
E VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBT,WFTTG(5),WFTISL,X,PSWA,PSWB
F ,XL,XR,LL,FSAFETY,AVGWT,CCOELL
COMMON/DEF1/ALLOWC(4),ALLOWE(4),MOT(12),PROBLL(4),TRATIO(2)
COMMON/DEF2/ADJ(4),DLRT(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),
A STIFF(2),TWEB(9),LLRW(2)
COMMON/BRDATA/BRLL,BRW,SPANL1,SPANB,SPANL2,RDW,FC,RMOD,FR,C1,C2,
B C3,GRND,GCUST,GCUSTI,WG(2),OPTWEB(10),OPTSPA(10),OPTPL(10),
C OPTCAN(10),P(4),UTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFETY,WEBDI,
D UWBEBD,TPT(1),OPT(1),IDEI
COMMON/GRID/ UGRND,GRNDLB,NG,CANTLB,CANTUB,WEBDUB,WEBULB,NU,
A PLWLB,PLWUB,IXG,IYC,IXGC,IYCC,IXH,IYB,IXHC,IYBC,ZGC(5,5),
B CSLAB,CREINF,CSTEEL,GCUSTI,ZHB(5,5),SAFTY(5,5),PLW(5,5),
C WH(5,5),OPTH(5,5),OPTB(5,5),OPTG(5,5),OPTC(5,5),SAFTI(5,5)
COMMON/BRCH/UCOST,BGRND,BCANT,BWEBD,BPLATE,DATUM,OH,OF
COMMON/LBC/CANT,ZMINC
10 FORMAT(20X,5HMINBM,2X,6HGRND =,2X,F4.0,6HGSPA =,
A 2X,F6.2,6HWEBD =,F6.2,2X,8HPLATEW =,F6.2)

11 FORMAT(20X,5HMINBM,2X,7HCANTB =,F6.2,2X,7HTSUMA =,
A F10.2,7HFT.KIPS)
C TO DETERMINE CANTB SO THAT SUM OF AREA UNDER B.M. DIAGRAM IS MIN.
C TO DETERMINE SUM OF B.M. IN SPAN TWO
ISUMA = 1000000000.
CANTB = CANTLB
40 CLB = CANTB
SPANL2 = SPANB - CANTB
SPANL = SPANL2

```

[illegible]

CANTB = 0.

PBMB = 0.

PSLB = 0.

PLLB = 0.

PSWB = 0.

SUMA = 0.

X = 0.

WTFITG(1) = 200.

DISILL = 1.1

EMPMAX = 1.3

EMPACT = AMIN1(EMPMAX, 1.0+50./(SPANL+125.))

DISTRB = EMPACT\*GSPA/(120.\*DISILL)

WTFITSL = 1.042\*(GSPA\*SLABT + HWIDTH\*HDEPTH)

50 CALL MOMENT

BMT1 = AMAX1((BMDL + BMSW + BMSWLL + POSLL)\*0.8, (BMDL+BMSW+POSLL))

BMT2 = AMAX1((BMDL+BMSW+BMSWLL+NEGLL)\*0.8, (BMDL+BMSW+NEGLL))

SUMA = SUMA + ABS(AMAX1(BMT1, BMT2))

X = X + 1.

IF(X.LE.SPANL2) GO TO 50

SUMA2 = SUMA

CANTILEVER REACTIONS

PBMB = WTFITG(1)\*SPANL/2000.

PSLB = WTFITSL\*SPANL/2000.

PSWB = SWFWS\*SPANL/2000.

PLLB = AMAX1((72. - 672./SPANL)\*(GSPA/(120.\*1.1)),

A (26. + 0.32\*SPANL)\*(GSPA/(120.\*1.1)))

C TO DETERMINE SUM OF B.M. IN SPAN ONE

SPANL = SPANL1

EMPMAX = 1.3

EMPACT = AMIN1(EMPMAX, 1.0+50./(SPANL+125.))

DISTRB = EMPACT\*GSPA/(120.\*1.1)

CANTB = CLB

SUMA = 0.

X = 0.

60 CALL MOMENT

BMT1 = AMAX1((BMDL + BMSW + BMSWLL + POSLL)\*0.8, (BMDL+BMSW+POSLL))

BMT2 = AMAX1((BMDL+BMSW+BMSWLL+NEGLL)\*0.8, (BMDL+BMSW+NEGLL))

SUMA = SUMA + ABS(AMAX1(BMT1, BMT2))

X = X + 1.

IF(X.LE.SPANL1) GO TO 60

C TO DETERMINE SUM OF B M OVER CANTILEVER

X = CANTB

65 BMT2 = X\*(PBMB + PSLB + PSWB + PLLB\*1.3) +

A WTFITG(1)\*X\*X/2000. + WTFITSL\*X\*X/2000.

SUMA = SUMA + ABS(BMT2)

X = X - 1.0

IF (X.GT. 0.) GO TO 65

SUMA1 = SUMA

TSUMA = SUMA1 + SUMA2

IF(TSUMA.LT.ISUMA) GO TO 70

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100

100-100



```

      GO TO 90
70  ISUMA = TSUMA
      CANTB = CANTB + 1.
      IF(CANTB.GT.CANTUB) GO TO 100
      GO TO 40
90  CANT = CANTB - 1.
      TSUMA = ISUMA
      GO TO 105
100 CANT = CANTB - 1.0
      TSUMA = ISUMA
105  WRITE(3,11) CANT ,TSUMA
      RETURN
      END

```

```

SUBROUTINE MINC
COMMON BMUL,BMLL,BMSW,BMSWLL,BMTRK,BOTTP(6),CANTA,CANTB,CG(6,4),
A  DELTA,DELTLB,DELTLG,DELI(3,4),DISTRB,EMPACT,EXTRA,FATIG(2),
B  FS(4),GSPA,HDEPTH,HWIDTH,I(6,4),LOAD,M,NEGLL,PBMA,PMBB,PLLA,
C  PLLB,PLATEI(2),PLATEW,POSLL,PSLA,PSLB,SLABW,SB(6,4),SLAB1,SPANL
D  ,ST(6,4),STATM(6),STRESA,STOOS,SWFWS,SWLL,TOPPT(6),UTIL,VOL,
E  VLL,VLLMIN,VSW,VTOTAL,WEBD,WEBI,WTF(5),WTFISL,X,PSWA,PSWB
F  ,XL,XR,LL,FSAFY,AVGWT,CODELL
COMMON/DES1/ALLOWC(4),ALLOWL(4),MUT(12),PRUMLL(4),TRAJ1U(2)
COMMON/DES2/ABJ(4),DLRI(2),LLR(7),LLRMAX(2),PSTUD(9),STIFF(9),
A  TSTIFF(2),TWB(9),LLRW01(2)
COMMON/BRDATA/BRLL,BRW,SPANL1,SPANB,SPANL2,RDWR,FC,RMOD,FR,C1,C2,
B  C3,GRNU,GCOST,GCUST1,WG(2),OPTWEH(10),OPTSPA(10),UPIPL(10),
C  OPTCAN(10),P(4),TUTIL,TSWLL,RAILWT,REINF(2),ITER,FSAFI,WEBD1,
D  UBWEBD,TPT(1),BPT(1),IDEI
COMMON/GRID/UBGRNU,GRNULB,NG,CANTLB,CANTUB,WEBDUB,WEBDLB,ND,
A  PLWLB,PLWUB,IXG,IYC,IXGC,IYCC,IXH,IYB,IXHC,IYBC,ZGC(5,5),
B  CSLAB,CREINF,CSTEEL,BCOST,ZHB(5,5),SAFY(5,5),PLW(5,5),
C  WH(5,5),OPTH(5,5),OPTB(5,5),CPTG(5,5),OPTC(5,5),SAFI(5,5)
COMMON/SRCH/UCOST,BGRNU,BCANT,BWEBD,BPLATE,DATUM,UH,UP
COMMON/LBC/CANT,ZMINC
10  FORMAT(20X,4HMINC,2X,4HGCOST,2X,3F10.2)
11  FORMAT(20X,4HMINC,23HCANT, 4R MIN WEB STEEL,2X,
A  F10.1,2F10.3,F10.0)
12  FORMAT(20X,13HFLANGE COST =,F10.0,
A  10HWEB COST =,F10.0)
14  FORMAT(20X,4HMINC,2X,3HSTF,2X,5HTPLTH,2F10.2)
15  FORMAT(20X,4HMINC,2X,7HFLANGES,2X,2F10.4)
16  FORMAT(20X,4HMINC,2HXL,2X,2HXR,2X,2F10.2)
17  FORMAT(20X,5HTPLTH,2X,F10.2)

```

C

C

TO DETERMINE CANTILEVER LENGTH SUCH THAT STRUCTURAL



```

C      STEEL IN WEB IS MINIMUM
C      ALLOWABLE SHEAR IN GIRDER WEB = GROSS SECTION IS 12 ksi.
      ICANT = 3
C      TO DETERMINE WEB THICKNESS IN SPAN 2
      SWEBI = 1000000000.
      CANTB = CANTLB
      WTFIG(1) = 200.
      WTFTSL = 1.042*(GSPA*SLANT + HWIDTH*HDEPTH)
      DISTLL = 1.1
40    CLB = CANTB
      SPANL2 = SPANB - CANTB
      CANTB = 0.
      PBMB = 0.
      PSLB = 0.
      PLLB = 0.
      PSWB = 0.
      SPANL = SPANL2
      EMPMAX = 1.3
      EMPACT = AMIN1(EMPMAX,1.0+50./(SPANL+125.))
      DISTRB = EMPACT*GSPA/(120.*DISTLL)
      WEBU = SPANL1*12./20.
      CALL SHEAR(0.)
      VL = VTOTAL
      CALL SHEAR(0.0)
      PLMIN = 0.375
      T = AMAX1(WEBU/150.,(WEBU*AMAX1(VL,VTOTAL)*1000./7500.**2)**0.333)
      TREWD = AMAX1(FIX(T*16.+0.99)/16.0,PLMIN)
      WEBT = TREWD
      T2 = WEBT

C
C      TO DETERMINE WEB THICKNESS IN SPAN 1
      CANTB = CLB
      SPANL2 = SPANB - CANTB
      PBMB = WTFIG(1)*SPANL/2000.
      PSLB = WTFTSL*SPANL/2000.
      PSWB = WTFTS*SPANL/2000.
      PLLB = AMAX1((72. - 672./SPANL)*(GSPA/(120.*1.1)),
A      (26. + 0.32*SPANL)*(GSPA/(120.*1.1)))
      SPANL = SPANL1
      EMPMAX = 1.3
      EMPACT = AMIN1(EMPMAX,1.0+50./(SPANL+125.))
      DISTRB = EMPACT*GSPA/(120.*1.1)
      WEBU = SPANL1*12./20.
      CALL SHEAR(0.)
      VL = VTOTAL
      CALL SHEAR(0.0)
      PLMIN = 0.375
      T = AMAX1(WEBU/150.,(WEBU*AMAX1(VL,VTOTAL)*1000./7500.**2)**0.333)
      TREWD = AMAX1(FIX(T*16.+0.99)/16.0,PLMIN)

```

1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216  
2217  
2218  
2219  
2220  
2221  
2222  
2223  
2224  
2225  
2226  
2227  
2228  
2229  
2230  
2231  
2232  
2233  
2234  
2235  
2236  
2237  
2238  
2239  
2240  
2241  
2242  
2243  
2244  
2245  
2246  
2247  
2248  
2249  
2250  
2251  
2252  
2253  
2254  
2255  
2256  
2257  
2258  
2259  
2260  
2261  
2262  
2263  
2264  
2265  
2266  
2267  
2268  
2269  
2270  
2271  
2272  
2273  
2274  
2275  
2276  
2277  
2278  
2279  
2280  
2281  
2282  
2283  
2284  
2285  
2286  
2287  
2288  
2289  
2290  
2291  
2292  
2293  
2294  
2295  
2296  
2297  
2298  
2299  
2300  
2301  
2302  
2303  
2304  
2305  
2306  
2307  
2308  
2309  
2310  
2311  
2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325  
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350  
2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363  
2364  
2365  
2366  
2367  
2368  
2369  
2370  
2371  
2372  
2373  
2374  
2375  
2376  
2377  
2378  
2379  
2380  
2381  
2382  
2383  
2384  
2385  
2386  
2387  
2388  
2389  
2390  
2391  
2392  
2393  
2394  
2395  
2396  
2397  
2398  
2399  
2400  
2401  
2402  
2403  
2404  
2405  
2406  
2407  
2408  
2409  
2410  
2411  
2412  
2413  
2414  
2415  
2416  
2417  
2418  
2419  
2420  
2421  
2422  
2423  
2424  
2425  
2426  
2427  
2428  
2429  
2430  
2431  
2432  
2433  
2434  
2435  
2436  
2437  
2438  
2439  
2440  
2441  
2442  
2443  
2444  
2445  
2446  
2447  
2448  
2449  
2450  
2451  
2452  
2453  
2454  
2455  
2456  
2457  
2458  
2459  
2460  
2461  
2462  
2463  
2464  
2465  
2466  
2467  
2468  
2469  
2470  
2471  
2472  
2473  
2474  
2475  
2476  
2477  
2478  
2479  
2480  
2481  
2482  
2483  
2484  
2485  
2486  
2487  
2488  
2489  
2490  
2491  
2492  
2493  
2494  
2495  
2496  
2497  
2498  
2499  
2500  
2501  
2502  
2503  
2504  
2505  
2506  
2507  
2508  
2509  
2510  
2511  
2512  
2513  
2514  
2515  
2516  
2517  
2518  
2519  
2520  
2521  
2522  
2523  
2524  
2525  
2526  
2527  
2528  
2529  
2530  
2531  
2532  
2533  
2534  
2535  
2536  
2537  
2538  
2539  
2540  
2541  
2542  
2543  
2544  
2545  
2546  
2547  
2548  
2549  
2550  
2551  
2552  
2553  
2554  
2555  
2556  
2557  
2558  
2559  
2560  
2561  
2562  
2563  
2564  
2565  
2566  
2567  
2568  
2569  
2570  
2571  
2572  
2573  
2574  
2575  
2576  
2577  
2578  
2579  
2580  
2581  
2582  
2583  
2584  
2585  
2586  
2587  
2588  
2589  
2590  
2591  
2592  
2593  
2594  
2595  
2596  
2597  
2598  
2599  
2600  
2601  
2602  
2603  
2604  
2605  
2606  
2607  
2608  
2609  
2610  
2611  
2612  
2613  
2614  
2615  
2616  
2617  
2618  
2619  
2620  
2621  
2622  
2623  
2624  
2625  
2626  
2627  
26

```

WEBT = TREND
T1 = WEBT
STEELW = WEBD*((SPANL1 + CANTB)*T1 + SPANL2*12)*3.4*GRNU
IF(ICANT.EQ. 1) GO TO 60
IF(STEELW.LI.SWEB1) GO TO 45
GO TO 50
45 CANTB = CANTB
SWEB1 = STEELW
50 CANTB = CANTB + 1.
IF(CANTB.GT. CANTUB) GO TO 55
GO TO 40
55 CANTB = CANTB
ICANT = 1
GO TO 40
60 WRITE(3,11) CANTB,T1,12,STEELW
C USE CANTILEVER WHICH MINIMIZES AREA UNDER B M DIAGRAM
C CALL MINBM
TO DETERMINE STRUCTURAL STEEL IN FLANGES SPAN 2
CANTB = CANT
CLB = CANTB
SPANL2 = SPANB - CANTB
CANTB = 0.
PBMB = 0.
PSLB = 0.
PLLB = 0.
PSWB = 0.
SPANL = SPANL2
EMPMAX = 1.3
EMPACT = AMIN1(EMPMAX,1.0+50./(SPANL+125.))
DISTRB = EMPACT*GSPA/(120.*1.1)
X = 1.
C TO DEFINE XL AND XR
XL = 0.
XR = 1.0
TPLTH = 0.
TOPPT(1) = .0625
BOTPT(1) = 0.0625
WEBT = 12
100 CALL PROPT(1)
CALL MUMENT
CALL STRESS(1)
IF(FS(2).GT.20000.) BOTPT(1) = BOTPT(1) + 0.0625
IF(FS(1).GT. 20000.) TOPPT(1) = TOPPT(1) + 0.0625
IF(FS(2).GT.20000. .OR. FS(1).GT.20000.) GO TO 100
TPLTH = TPLTH + TOPPT(1) + BOTPT(1)
IF(X.EQ. 35.) GO TO 102
GO TO 103
102 WRITE(3,15) TOPPT(1),BOTPT(1)

```



```

103 X = X + 1.
    TOPPT(1) = 0.0625
    BOTIPT(1) = 0.0625
    IF(X.GE.SPANL) GO TO 105
    GO TO 100
C   TO DETERMINE STRUCTURAL STEEL IN FLANGES SPAN 1
105 STF2 = IPLTH*14.*1.*3.4*GRND*1.31
    WRITE(3,14) STF2,IPLTH
    CANTB = CANT
    SPANL2 = SPANB - CANTB
    PBMB = WTFTG(1)*SPANL/2000.
    PSLB = WTFTSL*SPANL/2000.
    PSWB = SWFWS*SPANL/2000.
    PLLB = AMAX1((72. - 672./SPANL)*(GSPA/(120.*1.1)),
A      (26. + 0.32*SPANL)*(GSPA/(120.*1.1)))
    SPANL = SPANL1
    EMPMAX = 1.3
    EMPACT = AMIN1(EMPMAX,1.0+50./(SPANL+125.))
    DISTRB = EMPACT*GSPA/(120.*1.1)
    X = SPANL1
106 X = X - 1.
    CALL MDMENT
    IF((BMDL + POSLL + BMSW) .LE. 0.) GO TO 100
    XR = X/SPANL1
    X = 1.
    TPLH = 0.
    TOPPT(1) = 0.0625
    BOTIPT(1) = 0.0625
    WEBT = 11
110 CALL PROPTY(1)
    CALL MDMENT
    CALL STRESS(1)
    IF(FS(1) .GT. 20000.) TOPPT(1) = TOPPT(1) + 0.0625
    IF(FS(2) .GT. 20000.) BOTIPT(1) = BOTIPT(1) + 0.0625
    IF(FS(2) .GT. 20000. .OR. FS(1) .GT. 20000.) GO TO 110
    TPLH = TPLH + TOPPT(1) + BOTIPT(1)
    IF(X.EQ.35.) GO TO 112
    GO TO 113
112 WRITE(3,15) TOPPT(1),BOTIPT(1)
113 X = X + 1.
    IF(X.GE.SPANL) GO TO 120
    TOPPT(1) = 0.0625
    BOTIPT(1) = 0.0625
    GO TO 110
C   FLANGE PLATE THICK. OVER CANTILEVER
120 X = CANTB
    WRITE(3,17) TPLH
125 TOPPT(1) = 0.0625
    BOTIPT(1) = 0.0625

```

[illegible]



```

126 CALL PROPTT(1)
BMDLB = -(WTF1G(1) + WTF1SL + U1IL)*X**2/2000. - (PBMB + PSLB)*X
BMLLB = -X*PLLH*1.3
BMSWB = -SWFWS*X**2/2000. - X*PSWB
FTOPR = -(BMDLB + BMSWB + BMLLB)*12000./ST(1,1)
FBGTR = -(BMDLB + BMSWB + BMLLB)*12000./SB(1,1)
FS(1) = FTOPR
FS(2) = FBGTR
IF(FS(1).GT.20000.) TOPPT(1) = TOPPT(1) + 0.0625
IF(FS(2).GT.20000.) BOTPT(1) = BOTPT(1) + 0.0625
IF(FS(2).GT.20000. .OR. FS(1).GT.20000.) GO TO 126
TPLTH = TPLTH + TOPPT(1) + BOTPT(1)
X = X-1.
IF(X.GT.0.) GO TO 125

STF1 = TPLTH*14.*1.*3.4*GRNU*1.31
WRITE(3,14) STF1,TPLTH
TO DETERMINE BRIDGE COST

```

```

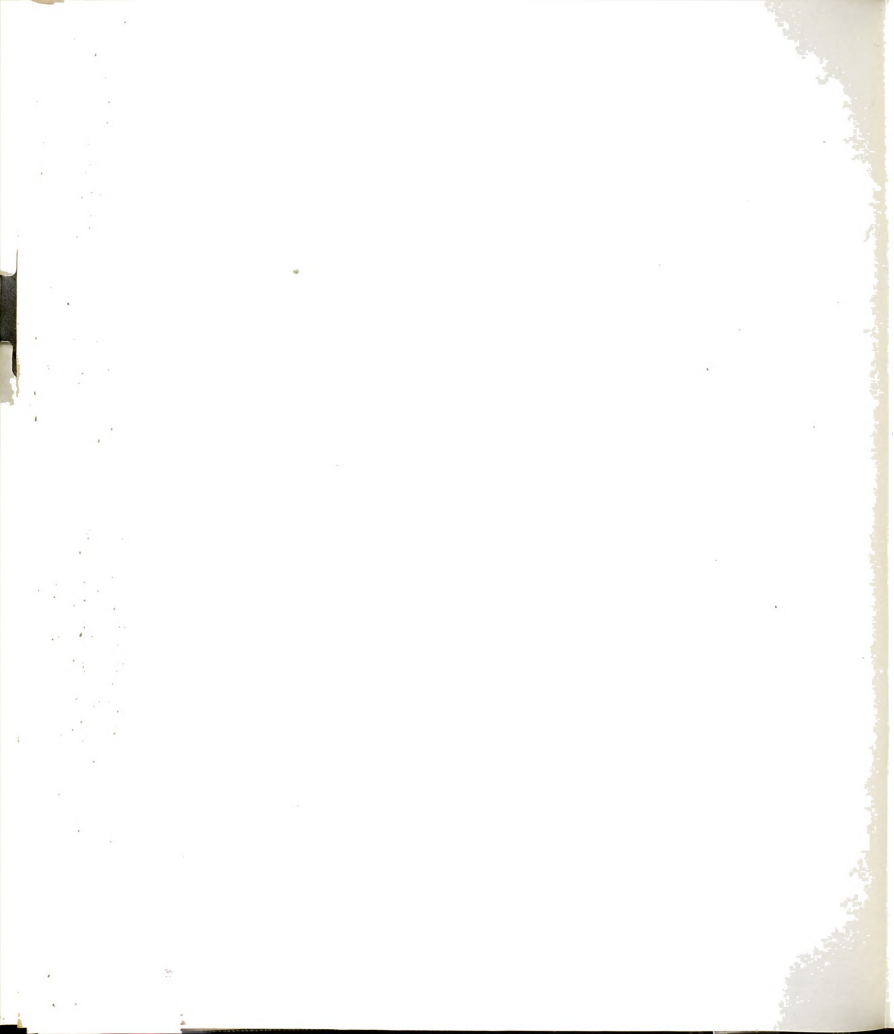
CSLAB = C1*(BRL*BRW*SLABT/(12.*27.) + (BRL*HWIDTH*HDEPTH/(144.*
A 27.)*GRNU))
CREINF = C2*490.*BRL*BRW/54.*REINF(2)
CSTEEL = C3*(STF1 + STF2 + STEELW)*1.18
BCOST = CSLAB + CREINF + CSTEEL
WRITE(3,10) CSLAB,CREINF,CSTEEL
CSTF = C3*(STF1 + STF2)*1.18
CWEB = C3*STEELW*1.18
WRITE(3,12) CSTF,CWEB
RETURN
END

```











MICHIGAN STATE UNIVERSITY LIBRARIES



3 1293 01756 9199