UNIVERSITY OF ADELAIDE

DEPARTMENT OF CIVIL ENGINEERING

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

by

A.E. NOBBS B.E.

Thesis submitted for the degree of Master of Engineering Science, University of Adelaide, 1983.

Awarded 26 September 1985

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

CONTENTS

ABSTRACT ...x

STATEMENT ...xii

ACKNOWLEDGEMENTS ...xiii

GLOSSARY OF SYMBOLS ...xiv

INTRODUCTION ...1

SECTION 1 - SLENDERNESS EFFECTS ...4

1.1 Introduction ...4

1.2 Theory ...4

1.2.1 Linear Elastic/Plastic Behaviour ...4

1.2.2 Non Linear Behaviour ...5

1.2.3 Creep Buckling ...10

1.3 Slenderness Effects Explained ...13

1.4 The Simplified Method ...13

1.5 Comparison of the requirements of various codes ...15

1.5.1 Range of Application ...15

1.5.2 Load Augmentation ...18

1.6 Worked Examples ...22

1.6.1 End Conditions ...23

1.6.2 Failure Curves for Pin Ended Columns ...27

1.6.3 Load Carrying Capabilities ...28

1.7 Alternative Method for Slender Columns ...29

1.7.1 Assumptions ...29

1.7.2 Method ...29

1.7.3 Consideration of Creep ...31

1.8 Comparing the rationality of ACI and DIN/CP ...32

1.9 Shortcomings of the AS1480 method ...32

1.10 Conclusions and Recommendations ...34

SECTION 2 - INTEGRATION WITH GENESYS ...37

2.1 Introduction ...37

2.2 Explanation of Column Subsystem of GENESYS ...37

2.2.1 Introduction ...37

2.2.2 PUBLIC Variable System ...38

2.2.3 Operation of Column Subsystem ...39

2.3 Explanation of Original Module for Rectangular Column Design ...42

2.3.1 Introduction ...42

2.3.2 Design Philosophy ...42

2.3.3 Selection of load cases for Design ...46

2.3.4 Avoidance of bar clashing ...46

2.3.5 Calculation of effective lengths ...47

2.3.6 Basic Flow Chart ...47

2.4 Details of (Original Module/Public Variable) Interaction ...49

2.5 Details of Information supplied to System by Original Modules ...54

2.5.1 Introduction ...54

2.5.2 Subsystems Involved ...54

2.5.3 Variables Involved ...54

2.6 Details of Public Block System adopted for Replacement Module ...56

2.6.1 Introduction ...56

2.6.2 Block/COLUMNS/ ...56

2.6.3 Block/RCSYS/ ...56

2.6.4 Block/CR/ ...56

2.7 Conversion of Replacement Module Output for use by System ...64

2.7.1 Introduction ...64

2.7.2 Subroutines Involved ...64

2.7.3 Variables Involved ...64

2.7.4 Method of setting the variables ...65

2.7.5 Variable CLD ...65

2.7.6 Variable FK ...65

2.7.7 Variable IBARS ...66

2.7.8 Variable ICNT ...66

2.7.9 Variable TBLE ...67

2.7.10 Variable ALP ...68

2.8 Adaption of Original Module Subroutines to suit Module ...69

2.8.1 Introduction ...69

2.8.2 Structural Changes ...69

2.8.3 Subroutines Involved ...70

2.8.4 Subroutine CENTRY ...70

2.8.5 Subroutine DRIVER ...71

2.8.6 Overlay 52 "Rectangular Column" RC22 ...72

2.8.7 Overlay 54 "RC24" ...72

2.9 Error Message Summary ...73

2.9.1 Introduction ...73

2.9.2 Errors detectable by Original Module ...73

2.9.3 Errors detectable by the Replacement Module ...75

SECTION 3 - COLUMN DESIGN MODULE ...77

3.1 Introduction ...78

3.2 The Engineering Logic ...79

3.2.1 Introduction ...79

3.2.2 General Notes ...79

3.2.3 Column Design Procedure ...80

3.2.4 Selection of Load Cases for design ...83

3.2.5 Logic for avoidance of Bar Clashing ...84

3.2.6 Failure Curve for Unreinforced Concrete ...86

3.3 Organisation of the module ...91

3.3.1 Introduction ...91

3.3.2 Information supplied to module ...93

3.3.3 Public Variable System ...93

3.3.4 Sign Conventions ...94

3.4 Subroutines Involved ...94

3.4.1 Introduction ...94

3.4.2 Subroutine CENTRY ...95

3.4.3 Subroutine EFLTH ...95

3.4.4 Subroutine BCLASH ...95

3.4.5 Subroutine RTVP ...95

3.4.6 Subroutine DRIVER ...95

3.4.7 Subroutine SORT ...96

3.4.8 Subroutine COLLOAD ...96

3.4.9 Subroutine SORTL ...96

3.4.10 Subroutine RED ...96

3.4.11 Subroutine FINDP ...97

3.4.12 Subroutine CALCP ...97

3.4.13 Function FUNCP ...97

3.4.14 Subroutine ARRANG ...97

SECTION 4 - IMPLEMENTATION AND TESTING OF THE NEW MODULE ...98

4.1 Introduction ...98

4.2 Implementation of the GENESYS System ...98

4.3 Implementation of the RC-STRUCTURE Subsystem ...100

4.4 Implementation of the RC-COLUMN Subsystem ...101

4.5 Implementation of the AS-COLUMN Subsystem ...101

4.6 Testing of AS-COLUMN ...102

4.7 Comparison between the results of AS-COLUMN and RC-COLUMN ...112

SECTION 5 - CONCLUSIONS ...114

5.1. Comparison of Slenderness Provisions ...115

5.2 Column Design Program ...116

5.3 Modifications to RC-BUILDING ...117

5.4 Improving Subsystem Efficiency ...117

REFERENCES ...119

APPENDIX 1 - PROGRAM FITCURV ...121

APPENDIX 2A - PROGRAM LISTING FOR FITCURV ...129

APPENDIX 2B - PROGRAM OUTPUT FOR FITCURV ...133

APPENDIX 3 - OVERLAY 52 "RC22" ...138

APPENDIX 4 - OVERLAY 54 "RC24" ...146

APPENDIX 5 - PROGRAMMER'S REFERENCE MANUAL ...166

APPENDIX 6 - METHOD FOR AVOIDANCE OF COLUMN/BEAM BAR CLASHING ...251

LIST OF FIGURES

1.1	Failure Curve for Linear Elastic/Plastic material	...4
1.2	Bar -Spring Assemblage	...6
1.3	Deflection Analysis	...7
1.4	Stress/Strain response of Concrete	...8
1.5	Superimposition of Figures 1.3 and 1.4(b)	...9
1.6	P-Δ plot derived from Figure 1.5	...10
1.7	Effect of Creep on lateral deflection	...11
1.8	P-Δ curve after creep	...12
1.9	Worked Example Column Section	...23
1.10	Effective length factor, sidesway allowed	...25
1.11	Effective length factor,sidesway prevented	...26
1.12	Safe loads e/d = 0.2	...28
1.13	Variation of K with slenderness ℓ/D	...32
1.14	Variation of α with slenderness ℓ/D	...33
2.1	Subroutine calling sequence of Original Module	...48
3.1	Situation at failure - unreinforced section	...86
3.2	Subroutine calling sequence of Replacement Module	...92
3.3	Sign Conventions	...94

COMPUTER AIDED DESIGN OF CONCRETE COLUMN

LIST OF TABLES

1.1	Table of code specifications	...17
2.1	Summary of subprograms in AS-COLUMN/1	...40
2.2	PUBLIC Blocks accessed in AS-COLUMN/1	...41
2.3	Subroutines contained in original OVERLAY 52	...44
2.4	Subroutines contained in original OVERLAY 54	...45
2.5	PUBLIC variable usage summary of Original Module	...50
2.6	Variables in BLOCKS /COLUMN/,/CR/,/RCSYS/ used outside of original module	...55
2.7	Public variable usage summary of replacement module	...60
3.1	Equivalent Bar and Zone sizes for nominal bar sizes	...86
4.1	Testing interpolation between charts	...104
4.2	Benchmark Test 2	...105
4.3	Benchmark Test 3	...106
4.4	Benchmark Test 4	...107
4.5	Benchmark Test 5	...108
4.6	Benchmark Test 6	...109
4.7	Benchmark Test 7	...110
4.8	RC-COLUMN Benchmark Test 2	...111

LIST OF DIAGRAMS

4.9	Relative performance of AS-COLUMN and RC-COLUMN	...113

LIST OF GRAPHS

3.1	Graph of failure curve for unreinforced concrete	...86

ABSTRACT

A method is presented for the Computer Aided Design of Reinforced Concrete Columns complying with the provisions of Australian Standard AS1480/1974, "The Use of Reinforced Concrete in Structures".

Standard variables and interface parameters have been adopted to enable the various sub-programs described herein to be fully integrated with a complete Reinforced Concrete building design system.

A particular feature is that a comprehensive set of design charts has been converted into a series of equations thus obviating the need to determine the position of the neutral axis by iteration. A direct solution for the reinforcement required is therefore possible.

The Thesis has been divided into 5 sections followed by a group of appendices. The first section examines the slenderness provisions of the current version of AS1480/74. The various approaches used in other codes are compared and consideration is given to abandoning the simplified method in favour of a more general approach.

The Second Section considers the standardisation required to allow any Rectangular Reinforced Concrete Column design module to be compatible with the building design system used by GENESYS.

The Third Section contains the details of a computer program which uses the equations generated from the design charts (see Appendix 1) to provide a design solution. It indicates where the relevant sections of AS1480/74 have been involved and also how the required steel and its disposition within the column is determined.

The Fourth Section describes the implementation and testing of the program. The Fifth Section contains the conclusions and general comments.

STATEMENT

This Thesis contains no material previously submitted for a degree in any University, and, to the best of my knowledge and belief, contains no material previously published or written by another person except where due reference is made in the text.

A.E. NOBBS

ACKNOWLEDGEMENTS

I wish to express my sincere thanks to Mr D B Crawley for his advice and encouragement during the supervision of this work.

I am also sincerely thankful to the South Australian Public Service Board who granted me time-off during working hours to perform this work.

I am indebted to my typiste Mrs Adrienne Twisk and to my family for their patience during the course of my studies.

A.E. NOBBS

GLOSSARY OF SYMBOLS

α	–	A factor used to derive the biaxial moment from the mono-axial moments
δ	–	Deflection or magnification factor used in Moment Magnifier method
π	–	Pi, the ratio of the circumferance of a circle to its diameter
$\emptyset$	–	Capacity reduction factor
$\emptyset$ (t)	–	Time dependent function representing the visco-elastic behaviour of a concrete bending element
θ	–	Angular rotation
σ	–	Stress
σ_{cr}	–	Buckling Stress
$\sigma_{elastic}$	–	The elastic limit
∞	–	An infinite number
ν	–	Factor used in calculating the creep deflection of a column
$a_1, a_2.$	–	Series of coefficients

A_s	-	Area of Steel Reinforcement
A_{sc}	-	Area of Steel under compression
B,b	-	Breadth of a concrete column section
C_m	-	Equivalent eccentricity term
d	-	Effective depth of a concrete column section
D	-	Actual depth of a concrete column section
DL	-	Dead Load
C	-	Eccentricity
e_o	-	Primary design eccentricity
e_{add}	-	Additional eccentricity
e_{eq}	-	Equivalent eccentricity
e_l	-	Lateral deflection
E	-	Modulus of Elasticity
ECX	-	Eccentricity for X-direction bending
ECY	-	Eccentricity for Y-direction bending

ECXY	–	Eccentricity for biaxial bending
F_c'	–	Concrete cylinder strength
f_{sy}	–	Steel yield strength
g	–	Ratio of distance between opposite face bars to overall depth (D)
G_A	–	Stiffness ratio of column head
G_B	–	Stiffness ratio of column foot
G_{AU}	–	Average of G_A and G_B
I	–	Second moment of area of column section
K	–	Factor relating l_e and l_c
K_u	–	Curvature
l, l_e, ℓ	–	Effective length of column
l_c, L	–	Actual length of column
LL	–	Live Load
M^l	–	Ultimate applied moment
M_x	–	Ultimate applied moment about x-axis

M_y - Ultimate applied moment about y-axis

M_o - Primary design moment

M_{add} - Additional moment

Mi - Internal moment

M_{max} - Maximum moment

Pø - Load at which creep buckling will occur

P' - Utimate applied axial load

P'_b - Balanced failure axial load

P - Applied axial load

P_{cr} - Axial load at which buckling will occur

q - Rotational spring stiffness

r - Radius of gyration

R - Reduction Factor (R' if tension failure governs)

T - Factor used to calculate the balanced failure load

x	-	Increment along length
y	-	Deflection at x point

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

INTRODUCTION

Computer Aided Design (C.A.D.) is a technique where man and machine combine to form a problem solving team, the computer performing the routine aspects of design and presenting the consequences of the designer's decisions quickly and effectively. It is not the same as Design Automation in which the computer can handle all demands and constraints without recourse to the designer. C.A.D. may be applied to a wide range of activities for example, analysis, optimisation, performance, calculation etc., each requiring a different level of designer involvement. In analysis the computer relieves the engineer of tedious calculation and there is little or no input required from the engineer as the program is progressing. The finding of an optimum solution particularly lends itself to computerisation with opportunities for iteration, linear programming and hill climbing (finding the peaks in a multi-dimensional space). Designer involvement would vary according to the complexity of the problem. Performance calculation is also well suited to C.A.D. techniques. Results such as speed, efficiency, power, cost etc may be calculated for a range of different condition and the effect of changing parameters may be quickly seen. This involves a high level of designer involvement.

There are certain qualities that a C.A.D. system should possess. The system should accept information in a fairly flexible form preferably also checking that the input is sensible. It should present the consequences and inferences of the designer's decisions quickly and clearly, particularly if these are to form the basis for further decision. For these reasons it is desirable that the designer have access to peripherals such as graph plotters, graphics terminals, line printers, visual display units etc. Graphical displays are particularly valuable because of the variety of ways in which information may be displayed (graphs, maps, diagrams etc), and because they can be used for both input and output.

Ideally the programs used in a C.A.D. system should be modular, that is split into groups each of which is responsible for a particular part of the design. This results in clear logic and easy debugging and ensures that as new calculation methods are evolved, the old module may simply be replaced by a new one provided that the correct interfacing is used. If no coupling between the various modules is provided, the designer may run the modules in any order he sees fit. The advantage of this is that component routines may be used in isolation making the system much more flexible. Each module may be further subdivided containing some or all of the following types of program:

- Service routines which contain the sequences for performing the problem (such as input and output)

- Calculation routines for solving equations and performing arithmetic

- Logic routines for controlling the path of calculation

- Data bases for storing information relevant to the design

GENESYS is a computer system which has been developed specifically for C.A.D. It provides a means whereby component programs, written in a language known as GENTRAN may compile and execute on a wide range of different computers. It is modular with free format data (the input being via tables and commands) and the user does not need to know how to program, just the sequence of commands required.

GENESYS contains a library of engineering subsystems, one of which is RC - BUILDING/1 which itself contains 36 individual programs for Reinforced Concrete structures. It can produce the design and detail including bar fixing, bending and weight schedules for beams, columns and flat slabs using the British Code of Practice CP110-71. The disadvantage to potential Australian users is that the British Code of Practices differs in many respects from the corresponding Australian Code of Practice AS1480/1974. Therefore the RC/BUILDING/1 subsystem is, at present, not acceptable in Australia.

While researching the alterations required to convert from CP110, to AS1480, significant differences were found in the manner of handling slender columns. It was then decided that a review of the slenderness provisions of a number of Concrete Codes of Practice would be worth while.

This project therefore had two objectives:

- to examine the slenderness provisions of the Australian Concrete Code, suggesting possible improvements

- to produce a column design module compatible with the GENESYS RC-BUILDING subsystem but complying with the current Australian Concrete Code

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

SECTION 1 - SLENDERNESS EFFECTS

1.1 Introduction

The slenderness of a column is important because it is a measure of the tendency of the column to buckle.

In this section, the slenderness provisions of a number of Concrete Codes of practice including AS1480 (Ref 11) are compared. The merits of each are discussed and suggestions are made for possible changes to AS1480.

1.2 Theory

1.2.1 For a pin-ended member composed of an ideal linear elastic material, the stress at which buckling occurs is given by equation 1.2.1. This equation is commonly called the Euler's Curve.

$$\sigma_{cr} = \pi^2 E / \left(\frac{\ell}{r}\right)^2 \qquad \ldots\ldots \; 1.2.1$$

However equation 1.2.1 is only valid for values of ℓ/r less than the elastic limit. For design purposes the failure curve is as shown diagrammatically in Figure 1.1.

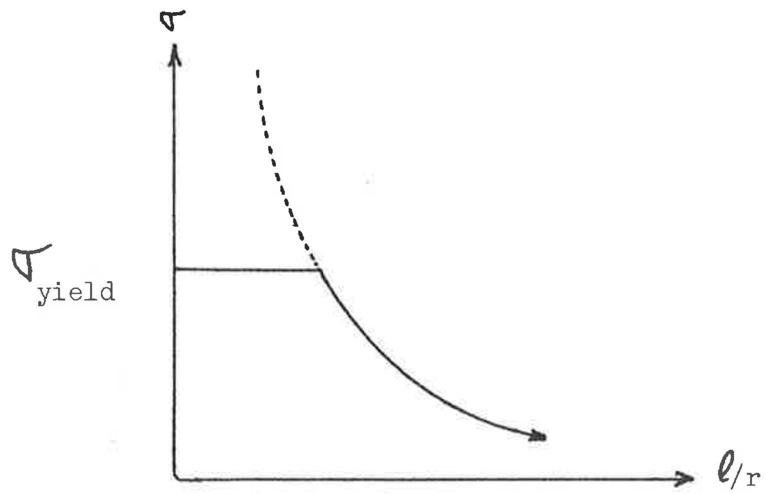


Figure 1.1 - Failure Curve for linear elastic-plastic material

The failure curve shown in Figure 1.1 cannot be used as the basis for designing concrete columns for the following reasons:

(1) Concrete is not an ideal linear-elastic/plastic material.

(2) Allowance must be made for non-homogeneity, creep, shrinkage and initial out-of-straightness.

(3) Columns are usually subjected to a combination of axial load and bending moment.

1.2.2 Non Linear Behaviour

The effect of the non-linear stress/strain response of concrete on buckling may be illustrated using the bar-spring assemblage shown in Figure 1.2

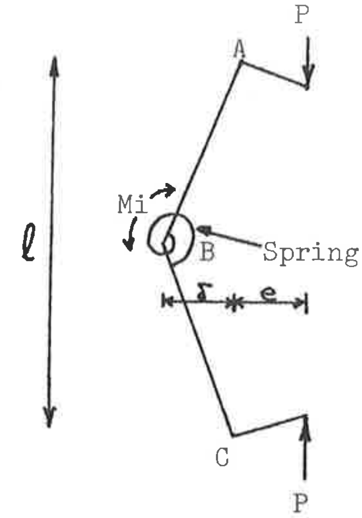


Figure 1.2 Bar-Spring Assemblage

From Reference (17), consideration of Figure 1.2 gives the following relationships:

$$\theta = 4\frac{\delta}{\ell} \qquad \ldots\ 1.2.2$$

$$M_i = q\theta = 4q\frac{\delta}{\ell} \qquad \ldots\ 1.2.3$$

$$P = 4\frac{q}{\ell}\frac{\delta}{e+\delta} \qquad \ldots\ 1.2.4$$

$$P_{cr} = 4\frac{q}{\ell}\ (\text{for } e=o) \qquad \ldots\ 1.2.5$$

$$M_i = P_{cr}\,\delta \qquad \ldots\ 1.2.6$$

Where q = Rotational spring stiffness (see Fig 1.2)

Equations 1.2.4 and 1.2.5 may be represented on an M-δ graph as shown in Figure 1.3.

The features of this representation are that:

(a) the slope of AC=P, the applied axial load

(b) the slope of OD=Pcr, the critical buckling load

(c) the initial eccentricity is represented by a shift in origin by an amount – e along the δ axis

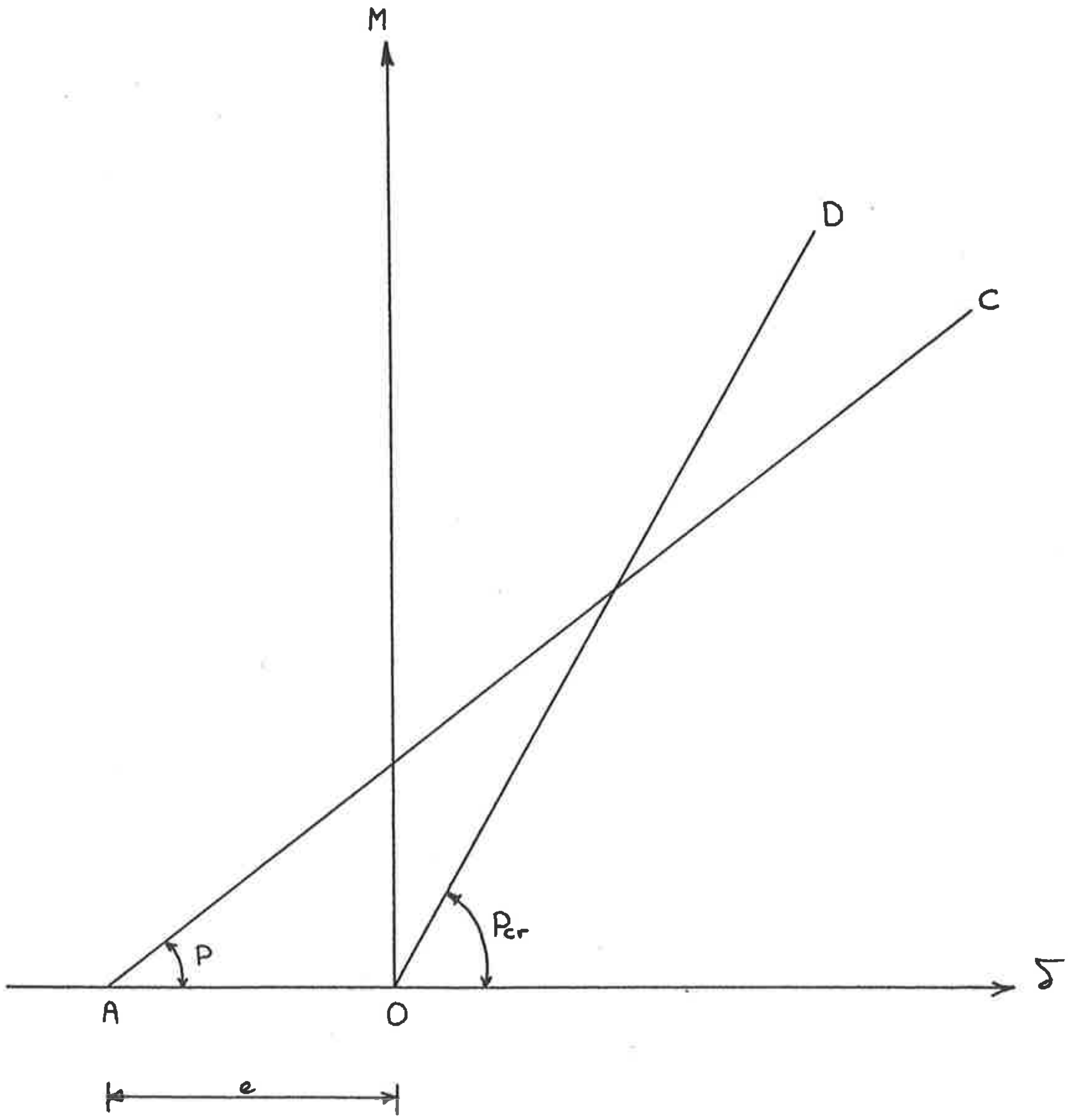
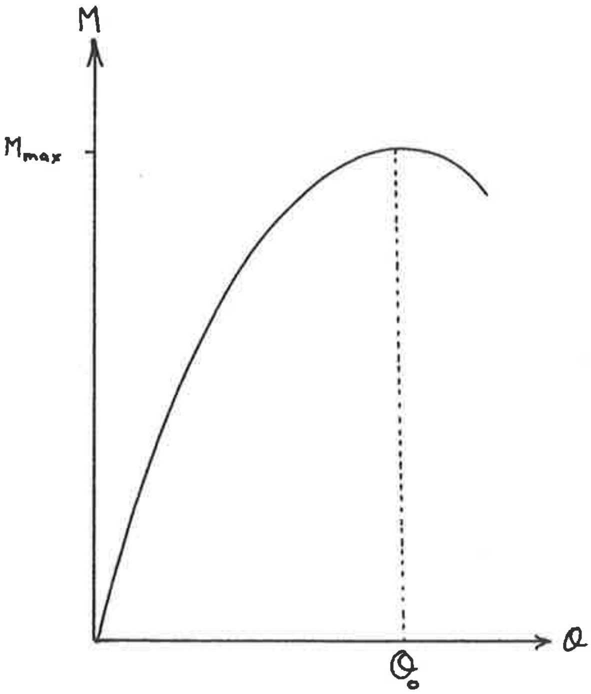


Figure 1.3 Deflection Analysis

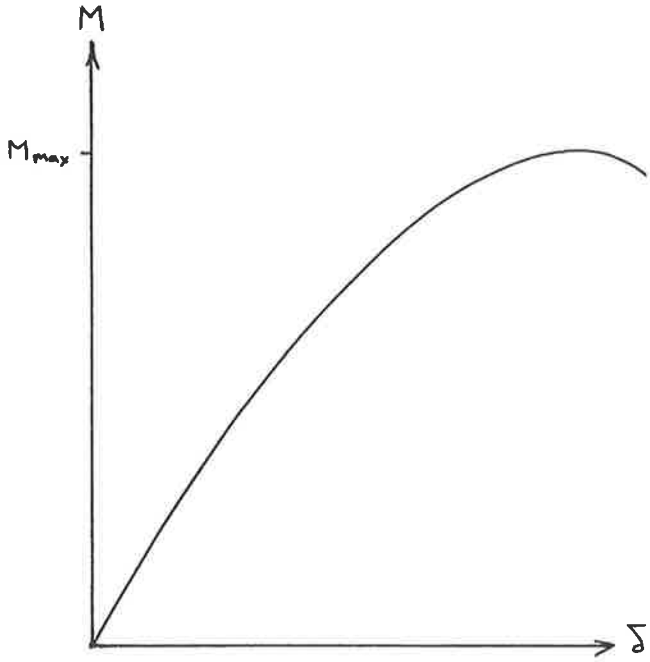
The loading response of a concrete column may be represented as the moment/curvature diagram shown in Figure 1.4 (a) or as the moment/deflection diagram shown in Figure 1.4(b) by the application of equation 1.2.3. (Note that q is non-linear).

θ_o = the rotation at which the maximum moment is achieved

M_{max} = the maximum achievable moment



a) Moment/Curvature



b) Moment/Deflection

Figure 1.4 Loading response of a concrete column

Figures 1.3 and 1.4(b) may be superimposed as shown in Figure 1.5. The slope, P of the line AX1X2 represents the applied axial force and is shown plotted against δ in Figure 1.6.

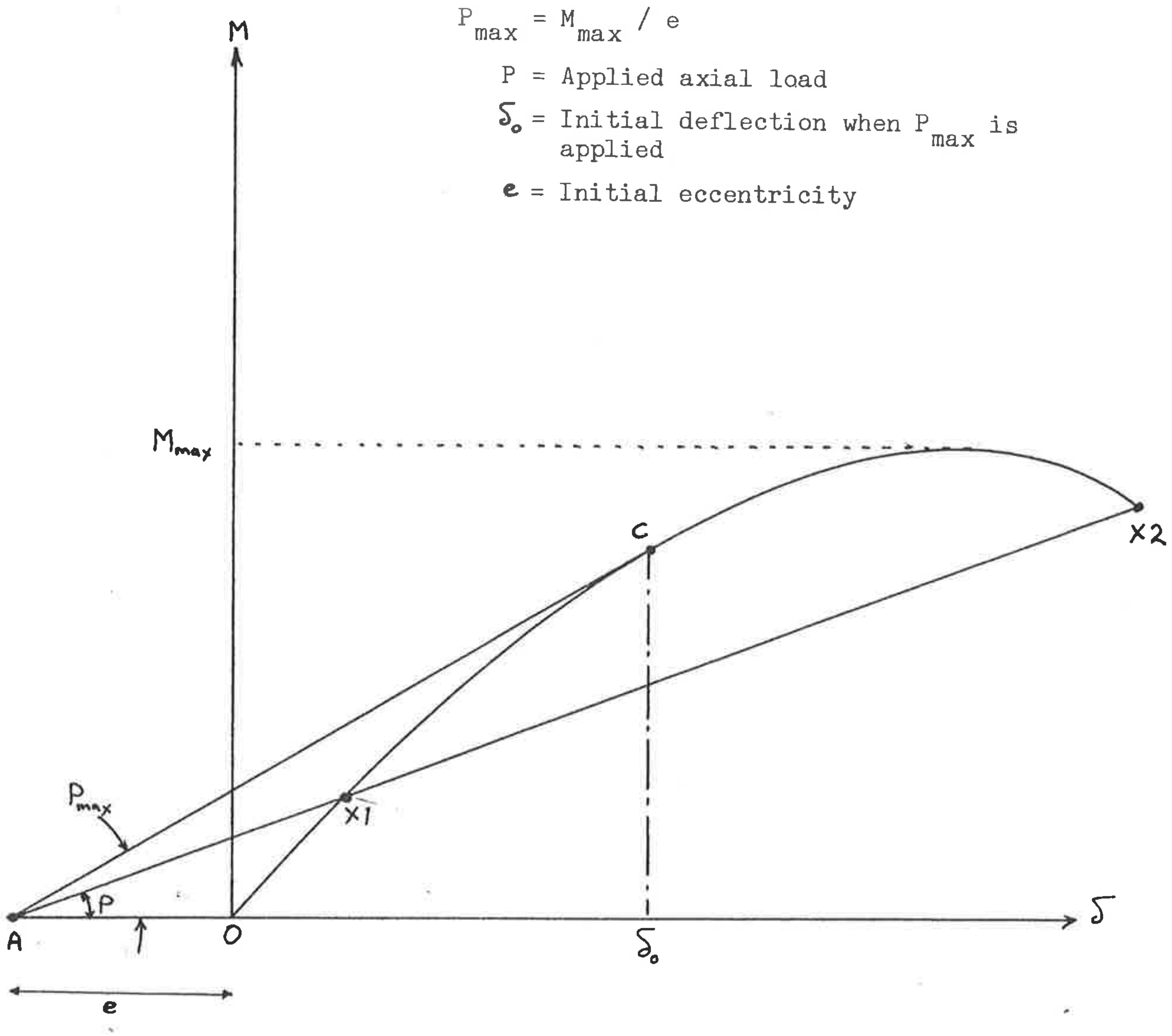


Figure 1.5 Superimposition of Figures 1.3 and 1.4(b)

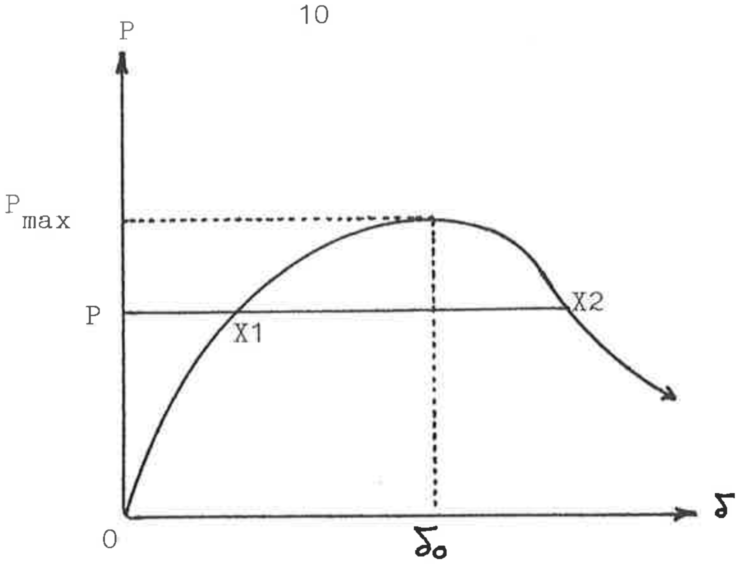


Figure 1.6 P-δ Plot derived from Figure 1.5

Reference (17) contains a detailed description of Figure 1.6 which may be summarized as follows.

The implication of Figure 1.6 is that the maximum axial load carrying capacity occurs before the full moment capacity of the section has been developed. As δ increases past δ_0, the moment carried is increasing but the axial load is decreasing.

This effect is most pronounced when e is very small because as the value of e tends towards infinity, the curve OX_1X_2 tends towards a straight line (e is represented by the segment AO on the δ axis in Figure 1.5) and the effect decribed above becomes less and less.

1.2.3 Creep Buckling

Another shortcoming of the linear-elastic theory in its application to concrete is that it fails to account for the long term reduction of load carrying capacity due to creep.

Under certain conditions, for example sustained overload, it is possible for this capacity reduction to induce a stability failure (Creep Buckling) after a finite period of time.

This phenomenon may be illustrated using the bar-spring assemblage developed in Section 1.2.2.

The effect of creep is to produce an additional lateral deflection which is a function of time as shown in Figure 1.7.

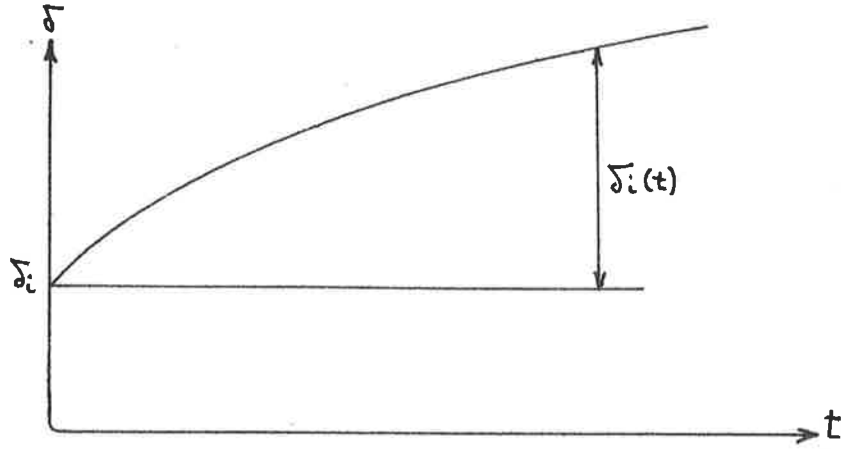


Figure 1.7 Effect of Creep on lateral deflection

The value of $\delta i(t)$ may be calculated as follows (Reference 13)

where $\upsilon = \frac{4q}{P_\phi \ell}$

$\phi(t)$ = the ratio of the rotation after time (t) to the initial rotation

P_ϕ = load at which creep buckling will occur (see Reference 13)

$$\delta i(t) = e\left[\exp\left(\frac{\phi(t)}{\upsilon - 1}\right) - 1\right] \quad \ldots\ldots 1.2.7$$

With reference to Figure 1.5, this additional lateral deflection reduces the slope of line AC and hence the value of P_{max}.

The application of load P_{ϕ} over a period of time has thus resulted in the new P-δ curve shown in Figure 1.8. If now the column were to be loaded by a load P then buckling will eventually occur due to the action of creep. This is because the curve $\delta_i(t)$ gradually changes with time so that eventually P_{ϕ} lies below P.

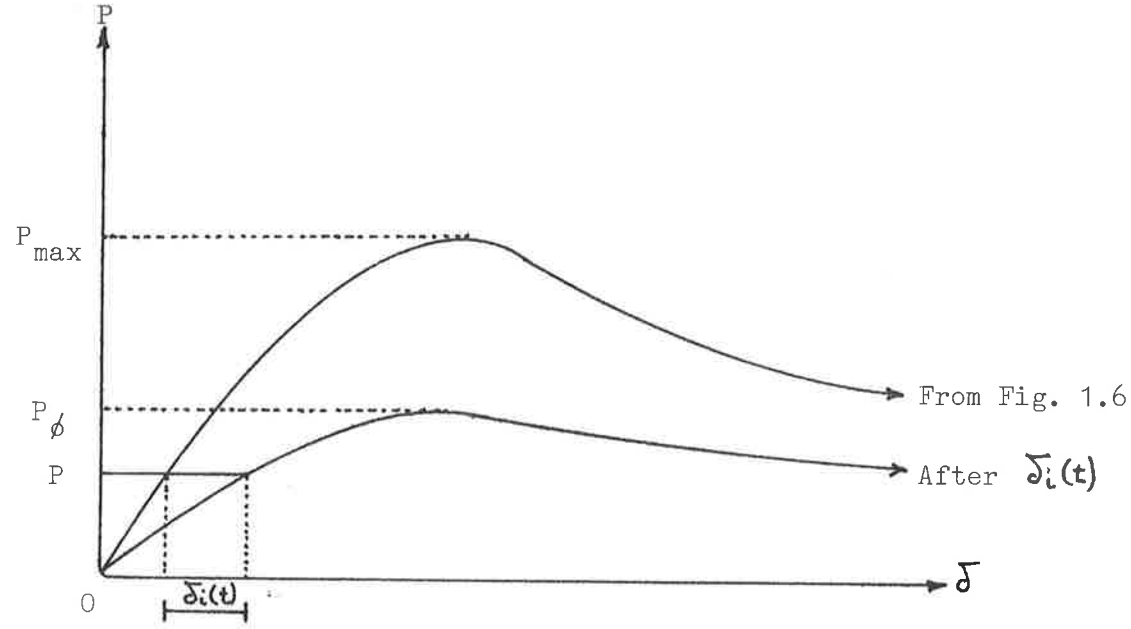


Figure 1.8 P-δ curve after creep

P = Initial axial load

The reduction factor R in AS1480-1974 includes some provision for the adverse effects of creep on long columns. However, as explained in Reference (17), it would be more rational if the creep were to be considered as an additional deflection. This could be included in future revisions of AS1480.

1.3 Slenderness Effects Explained

In addition to the applied moments (if any), all compression members carry moments due to the action of the axial load on the following:

(a) Unavoidable eccentricities

(b) Designed eccentricities

(c) Out-of-straightness of the member

(d) The lateral deflection

As the slenderness of the member increases, the lateral deflection effect increases to the point where an appreciable proportion of the internal moment is due to this P-Delta effect and the load carrying capacity begins to decrease. At this point a second order analysis should be used to describe the structural properties of the member. This second order analysis should account for the effect of the deformations on the equilibrium of the structure and the non- linearity of the materials.

In practice the application of these second order analyses to real-life problems is too complex for general use and Design Codes usually allow for a simplified method for members of medium slenderness. If the slenderness exceeds a specified amount then a full second order analysis must be used.

1.4 The Simplified Method

The Simplified Method is an approximation based upon a number of simplifying assumptions about both the external and internal influences upon the column. The method proceeds in three stages:

Stage 1: Isolate the member from the rest of the structure and model the second order structural effects as local behaviour in the isolated column.

Stage 2: From the isolated column produce an idealized pin-ended column whose length and eccentricity are adjusted to match the end constraints of the idealized column. This introduces the concept of effective length and equivalent eccentricity.

Columns in braced frames are treated differently from those in unbraced frames because for unbraced frames the lateral rigidity depends upon the column stiffness. The secondary moments are thus likely to be greater in unbraced frames than in braced ones.

Stage 3: Adjust the forces in the critical cross-section of the column to account for the slenderness of the standard pin ended column.

The design of the structural member has thus been reduced to the design of a single cross section.

There are three methods which may be used to implement stage 3 above. They are:

(1) The Moment Magnifier method in which the first order column moment is multiplied by a magnification factor, δ, which depends on the load, P, the effective length, ℓ, and the bending stiffness of the column cross section. This is the method used in the current American Code - "Building Code Requirements for Reinforced Concrete", ACI 318-71 (Ref. 1).

(2) The Additional Moment Method which is similar to the moment magnifier method in that the load P remains unaltered but differs in that an additional moment is added to the column moment instead of multiplying by a factor. This method is used in European Codes such as the British Code - "The Structural Use of Concrete", CP110(Ref. 3), the German Code - "Beton und Stahlbetonbau, Bemessung und Ausfuhrung", DIN 1045-1971 (Ref. 8) and the CEB Code - "International Recommendations for the Design and Construction of Concrete Structures", CEB - 1970 (Ref. 5).

(3) The Reduction Factor Method which magnifies both the moment and the axial force. In effect this is a load magnification method and is used in the current AS1480/1974 (Ref. 11), the earlier American Code ACI 318-66 (Ref. 2) and in the current Canadian Code - "Code for the Design and Construction of Plain or Reinforced Concrete Structures", CSA A23.3-1970 (Ref. 4).

1.5 Comparison of the Requirements of Various Codes

Five current codes and the CEB Recommendation were chosen for comparison. It should be noted that the Canadian Code CSA A23.3 is very similar to the earlier American Code ACI 318-63. A summary is given in Table 1.1 showing the method adopted, the slenderness limits, the effective load factors and the design eccentricities.

1.5.1 Range of Application

All codes concede that the simplified method is limited in its range of application. Usually if the slenderness is below a certain limit then slenderness effects may be ignored (short column) and if above a certain limit then the design must be based on first principles.

The lower limits may depend on a number of additional factors such as:

(1) In AS 1480 for unbraced columns, whether the design is controlled by tension or compression failure.

Also the method is only allowed if the average stiffness ratio of the head and foot joints is less than 10.

i.e. $G_{AV} = .5\ (G_A + G_B)$ ..... 1.5.1)

where G_A = stiffness ratio at head

G_B = stiffness ratio at foot

TABLE 1.1 - COMPARISON OF CODE SPECIFICATION

COUNTRY		AUSTRALIA	U.S.A.	CANADA	U.K	WEST GERMANY	EUROPE
DESIGNATION		AS1480	ACI318	CSA A23.3	CP110	DIN1045	CEB Recs
REFERENCE NO.		11	1	4	3	8	5
YEAR		1974	1971	1970	1972	1971	1970
Procedure Used		Reduction Factor	Moment Magnifier	Reduction Factor	Additional Moment	Additional Moment	Additional Moment
ℓ/r limit	Upper	100	100	70	150	70	140
	Lower	20	*	**	40	Equation	35
Load Factor	LL/DL=0	1.5	1.4	1.5	1.4	1.75 2.1	1.5
	LL/DL=1	1.65	1.5	1.65	1.5	Depending on	1.5
	LL/DL=∞	1.80	1.6	1.8	1.6	Steel Strain	1.5
Eccentricity	Min	None	25mm or .10*D	25mm or .10*D	0.05D	None	20mm or D/30****
	Addit.	25mm + D/100	***	None	None	L/300	None

* (22 for Unbraced)
(35 - $\left(\frac{M1}{M2}\right)$x12 for braced)

** (20 for Unbraced)
(10 for Braced, Single Curvature)
(27 for Braced, Double Curvature)

*** A proposed change would give .6 + 0.3D

**** OR reduce F'c by 27%

$$G = \frac{\Sigma(EI/\ell c) \text{ columns}}{\Sigma(EI/\ell b) \text{ beams}} \qquad \text{..... 1.5.2)}$$

(2) In ACI 318-71, the ratio of the end moments M_1/M_2.

(3) In DIN 1045, the load eccentricity.

(4) In CSA A23.3, whether the column is in single or double curvature.

The choice of the upper limit is somewhat more arbitrary and depends on the code writer's willingness to apply the simplified method to high slenderness ratios.

1.5.2 Load Augmentation

The simplified method is based on increasing the applied loads to account for the slenderness and each version has its own way of doing this.

(a) Reduction Factor Method: (AS 1480, CSA 23.3)

Both the current Australian and Canadian Codes are based on the earlier version of the American Code ACI 318-63. In the Canadian Code there are 3 separate equations for determining the reduction factor R which depends on whether the column is braced or not and whether the primary moments produce single or double curvature. The Australian Code is considerably simpler using only 1 equation viz:

$$R = 1.2 - 0.01 \left(\frac{l}{r}\right) \qquad \ldots\ldots 1.5.3)a$$

Where R = Reduction Factor

l = Effective length

r = Radius of gyration of section

In both the Australian and Canadian Codes, if tension failure governs then R is increased, which has the effect of reducing the required design load increase due to slenderness. This has been allowed because a buckling failure is less likely to occur when tension governs than when compression governs.

$$R' = 1 - (1-R)\frac{P}{P'_b} \qquad \ldots\ldots 1.5.3)b$$

Where R' = Reduction Factor when tension governs

P = Applied axial load

P'_b = Load at which balanced failure occurs

(b) Moment Magnifier Method (ACI 318-71)

In the American Code, the design moment is derived by multiplying the larger of the head and foot moments (M_2) by the factor F defined as follows:

$$F = \frac{C_m}{1 - \frac{P_u}{\phi P_c}} \quad \text{..... 1.5.4)}$$

where P_c = the Euler Buckling Load

$$= \frac{EI\pi^2}{\ell^2} \quad \text{..... 1.5.5)}$$

E = Code value for modulus of elasticity

I = Code value for moment of Inertia

P_u = Applied axial load

ϕ = Capacity reduction factor

and C_m = a term to provide an equivalent eccentricity

= 1 for unbraced frames

$= .6 + .4 \times \frac{M1}{M2}$ but not less than 0.4 for unbraced frames ..... 1.5.6)

(c) Additional Moment Method (DIN 1045)

The West German Approach is to calculate an additional eccentricity caused by the slenderness effect. The final design eccentricity thus has 3 components:

$$M = P\,(e_o + e_\ell + e_{add}) \quad \text{..... 1.5.7)}$$

e_o = the primary design eccentricity calculated from the applied moment

e_{add} = the initial eccentricity caused by "out-of-straightness" of the member, and is given by:

$$e_{add} = \frac{\ell}{300} \qquad \text{..... 1.5.8)}$$

e_ℓ = The lateral deflection at the design cross section and depends on both e_o and $(\frac{\ell}{r})$ as follows:

For $0 \leqslant \frac{e_o}{D} < 0.3$

$$e_\ell = D \times \frac{(\frac{\ell}{r} - 20)}{100} \sqrt{0.1 + \frac{e_o}{D}} \qquad \text{..... 1.5.9)}$$

For $0.3 \leqslant \frac{e_o}{D} < 2.5$

$$e_\ell = D \times \frac{(\frac{\ell}{r} - 20)}{100} \qquad \text{..... 1.5.9b)}$$

For $2.5 \leqslant \frac{e_o}{D} < 3.5$

$$e_\ell = D \times \frac{(\frac{\ell}{r} - 20)}{100} (3.5 - \frac{e_o}{D}) \qquad \text{..... 1.5.9c)}$$

Thus although the nominal upper limit for $\frac{\ell}{r}$ is 70, higher values may be used depending on the value of e_o.

Note also that e_ℓ and e_{add} should have the same sign as e_o so as to increase its numerical value.

(d) Additional Moment Method (CP110)

In the U.K. Code, an additional moment M_{add} is added to the primary design moment M_o (= Pe_{eq}). M_o is derived in a similar fashion as in the ACI 318-71 (Eq 1.5.6) from the head and foot moments as follows:

$$M_o = .4\,M_1 + .6\,M_2$$

but $$M_o \not> .4\,M_2 \text{ and } M_o \not> M_1 . \qquad \ldots\ldots 1.5.10$$

The additional moment (for monoaxial bending about the major axis) is given by:

$$M_{add} = \frac{PD}{1750}\left(\frac{\ell}{D}\right)^2 \left(1 - .0035\frac{\ell}{D}\right) \qquad \ldots\ldots 1.5.11$$

(e) Additional Moment Method (CEB)

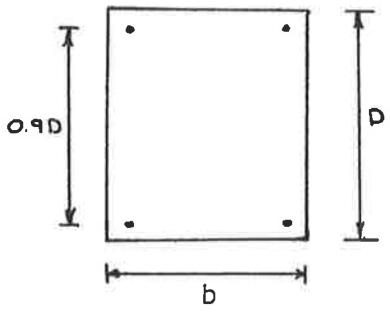
In the European code the same provisions as in CP110 apply except that M_{add} is calculated depending on the curvature K_u in the column section at failure.

$$M_{add} = PK_u\frac{\ell^2}{10} \qquad \ldots\ldots 1.5.12$$

1.6 Worked Examples

The performance of the codes is best compared using the results of a number of identical worked examples. The results given here are taken from references 7 and 9 and only the most relevant are presented.

The worked examples are treated in 3 groups. The first considers the various methods for treating end effects using the following column section.



F'c = 30 MPa

fsy = 410 MPa

Asc = Ast = .015bD

Figure 1.9: Worked Example Column Section

The second group considers the treatment of an idealized slender pin ended column so that a comparison can be made between the various method for treating the slenderness effect.

The third group compares the load-carrying capacity of a given column calculated in accordance with each code.

1.6.1 End Conditions

All codes require an effective length to be calculated to allow for the end conditions of the column where:

$$\ell_e = k\ell_c \qquad \ell_e = \text{effective length}, \quad \ell_c = \text{column length} \qquad \ldots\ldots 1.6.1$$

The American and Canadian Codes determine k for the frame in which the column is situated assuming it to be elastic and subjected to axial load conditions. They make use of the Jackson and Moreland charts (Ref. 9) which were originally developed from studies of elastic frame stability of steel compression members.

The CEB and West German codes are similar in approach to the American code in that k is determined from an appropriate elastic analysis or from a consideration of the elastic buckled shape of the frame. The British and Australian codes have adopted similar, simplified methods for determining k.

For Braced frames, AS 1480 gives k the values $.75 \leqslant k < 1$ depending on end conditions and CP110 adopts a similar range.

For Unbraced frames k is greater than unity and is independent of the type of curvature (single or double) as follows:

AS 1480 $\quad k = (1 + .3G_{AV} - 0.01\ G^2{}_{AV})$ 1.6.2

CP110 $\quad k = (1 + .3\ G_{AV})$ 1.6.3

but $\leqslant (2 + .3\ G_{min})$

Although CP110 is slightly more conservative than AS1480, the upper limit ensures that realistic values are adopted.

Comparisons of the various requirements have been made in Ref. 9 and are shown in Figures 1.10 (Unbraced) and 1.11 (Braced). The factor k is plotted against the stiffness ratio G_A as defined in Eq 1.5.1. Values for G_B, at the other end of the column, are shown on the appropriate curve. Note that for typical floors when the stiffness is roughly the same at the head and foot, the use of G_{AV} would be correct. However if G_A and G_B were unequal the error in making this assumption may be as much as 40% if $G_A = 0$ and $G_B = 10$ and 22% when $G_A = 1$ and $G_B = 10$.

This disadvantage is offset by the simplicity of the AS 1480 method when compared with the ACI nomogram.

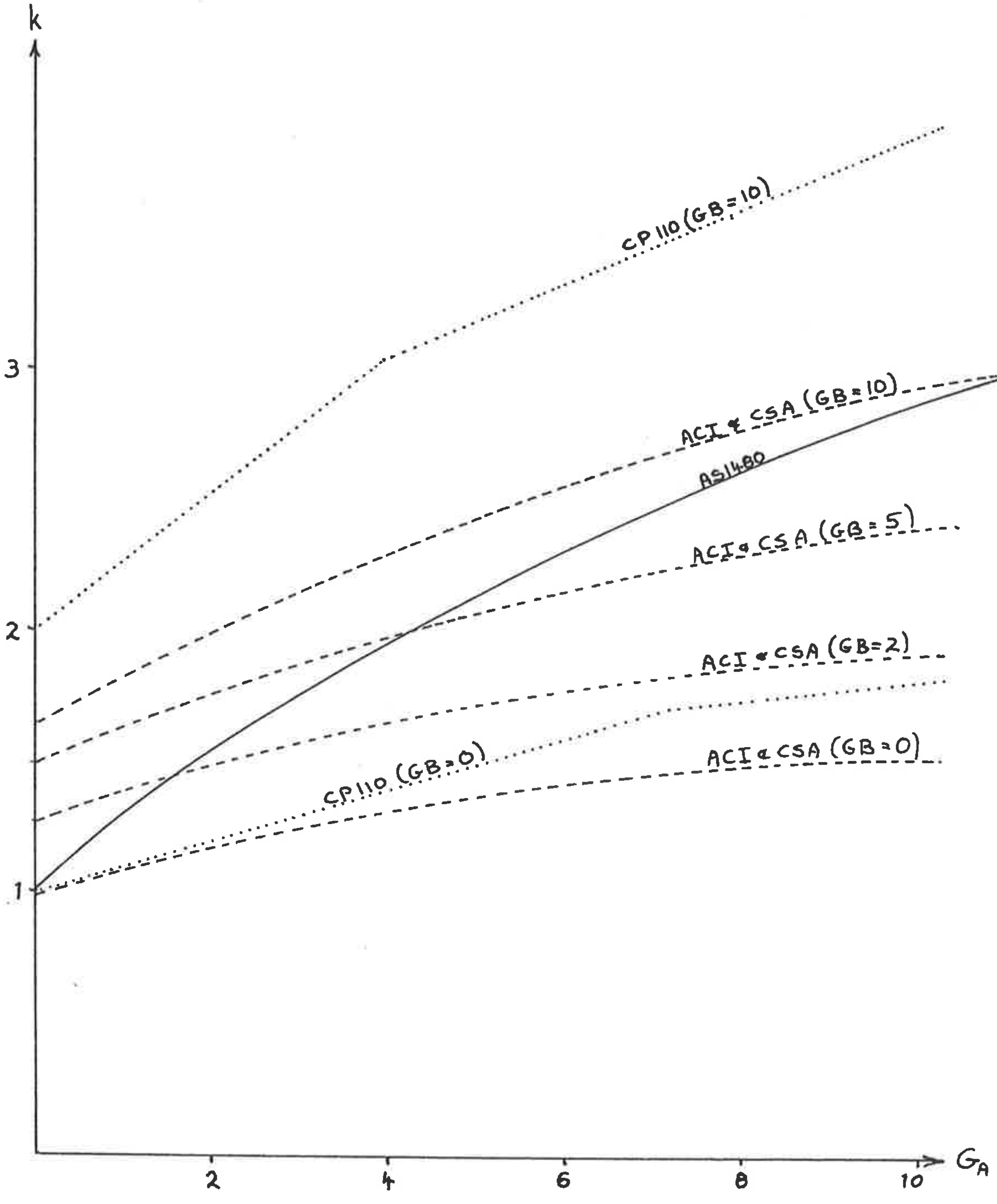


Figure 1.10 - Effective length factor, sidesway allowed

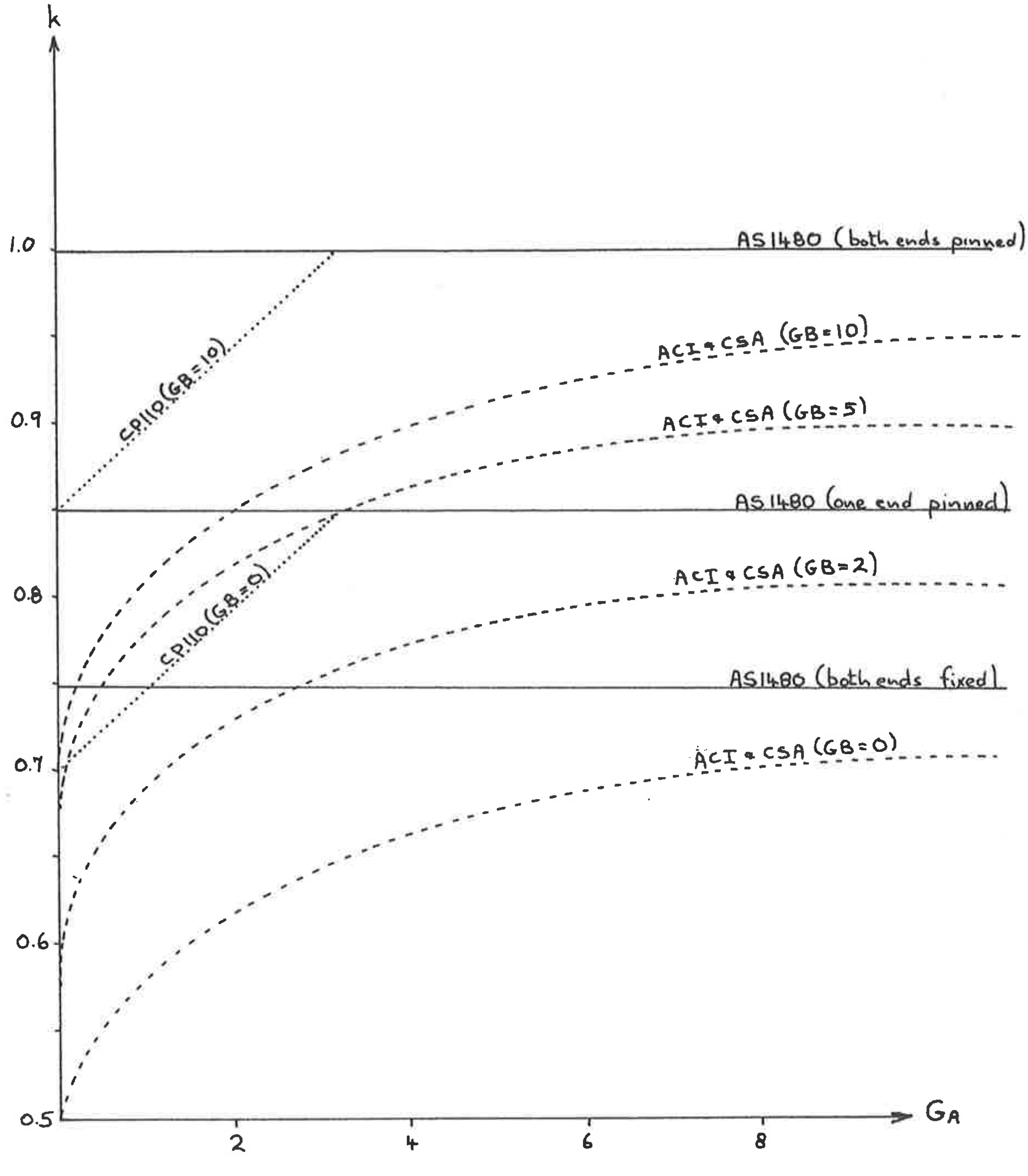


Figure 1.11 - Effective length factor, sidesway prevented

1.6.2 Failure Curves for Pin Ended Column

Ref. 7 gives a number of charts representing different $\frac{\ell}{r}$ ratios which show the failure curves of the same column derived from the provisions of the various codes.

From these charts the following conclusions may be reached.

(1) The AS 1480 column provisions are among the most conservative.

(2) The AS 1480 safety provisions (Table 1.1 load factors) are more conservative than in the other codes.

(3) The additional eccentricity requirement (e_{add}) of AS 1480 is seen in Table 1.1 to be more conservative than the other codes.

(4) In the compression failure zone, AS 1480 is one of the most conservative, although this is due partly to the conservative safety factors adopted.

(5) The codes which use the reduction factor method (AS 1480 and CSA) show large differences in the tension failure zone when the $\frac{\ell}{r}$ ratio is high. This is due to the way in which the reduction factor R and the capacity reduction factor simultaneously vary with increasing value of e.

1.6.3 Load Carrying Capabilities

Figure 1.13 shows a line chart comparing the load carrying capabilities of the same column as calculated using the provisions of the various codes. It should be noted that an l/r ratio of 100 is the upper limit allowed by AS1480.

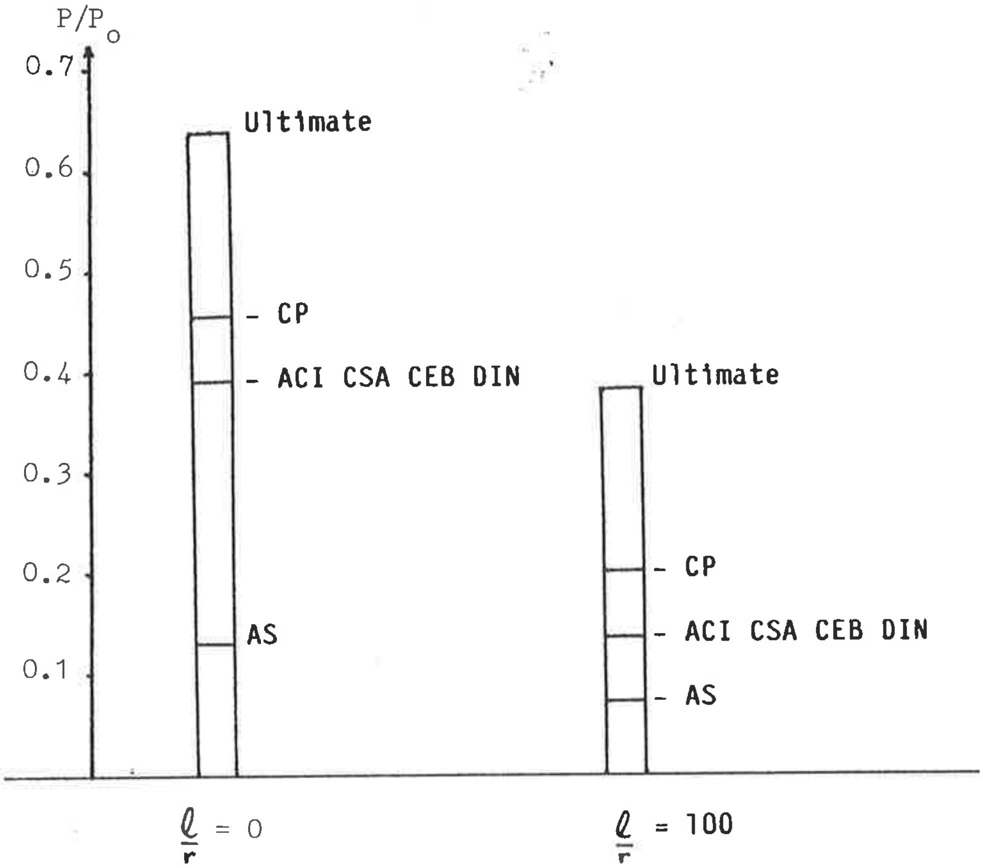


Figure 1.12 Safe Loads e/D = 0.7

1.7 Alternative Method for Buckling Load of a Very Slender Column

Ref. 13 gives an alternative method for calculating the buckling load of a very slender column. It is more accurate than the usual Euler approach in that the effect of the non-linear stress/strain relationship is taken into account.

1.7.1 Assumptions

The developed theory makes use of the following assumptions (Ref 13):

(a) Plane sections remain plane

(b) Perfect bond exists between steel and concrete

(c) Shortening under direct loads does not affect the geometry

(d) Reinforcement is elastoplastic

(e) The effect of strain hardening and relaxation of steel may be ignored

1.7.2 Method

The method is based on a fourth order polynomial describing the deflected shape of a slender reinforced concrete column pinned at both ends and loaded axially. The coefficients of this polynomial were determined by statistical curve fitting methods applied to experimentally measured deflection profiles.

$$y = a_1 x + a_4 (x^4 - 2Lx^3) \qquad \text{..... 1.7.1}$$

Where y = deflection

x = increment along length

a_1, a_4 = coefficients

L = Length

Using the polynomial, the buckling load is given as follows:

$$N \text{ (Axial load)} = \frac{\alpha K E c I}{L^2} \qquad \ldots\ldots \ 1.12.2)$$

where $\alpha = \dfrac{48a_4 L^3}{8a_1 - 3a_4 L^3}$ shown in Figure 1.14

K = A factor shown in non dimensional form in Figure 1.13 to account for the following:

(a) Initial crookedness and other imperfections

(b) Relationship between the cylinder strength and the in situ strength

(c) Casting positions of columns

Ec = Initial Modulus of Elasticity of the concrete cylinders

I = Equivalent Moment of Inertia of a very slender column accounting for effect of the steel

L = Equivalent length

α is a geometrical dimensionless factor similar to π^2 in the Euler formula except that it varies with the slenderness, ratio - see Figure 1.14.

1.7.3 Consideration of Creep

From observation of long term behaviour, the above buckling failure loads were found to be unsafe due to the effect of creep (Ref. 13). Further long term statistical analysis enabled the derivation of the following reduction factors which, when multiplied by the result from equation 1.12.2 gave the recommended safe axial load:

For * $30 < \ell/D \leqslant 40$) $R = 1.21 - .022 \dfrac{\ell}{D}$ 1.12.1
ie $100 < \ell/r \leqslant 133$)

$40 < \ell/D \leqslant 79$) $R = .48 - .004 \dfrac{\ell}{D}$ 1.12.2
$133 < \ell/r \leqslant 263$)

* Note that below $\ell/D = 30$, the normal code provisions apply.

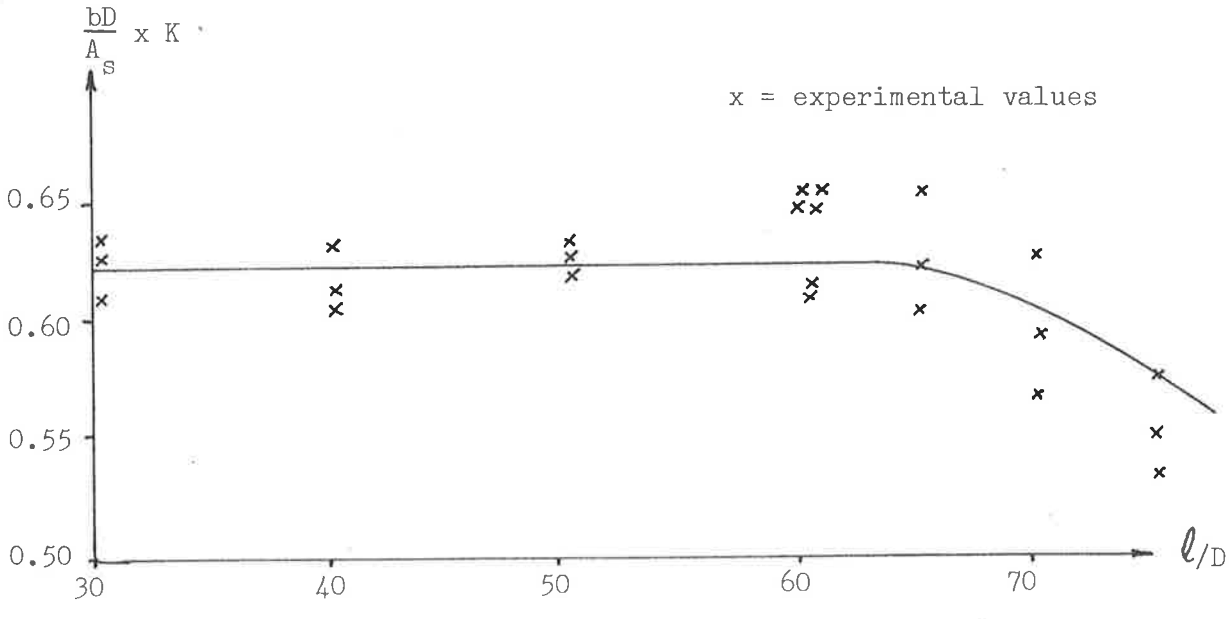


Figure 1.13 Variation of K vs Slenderness ℓ/D

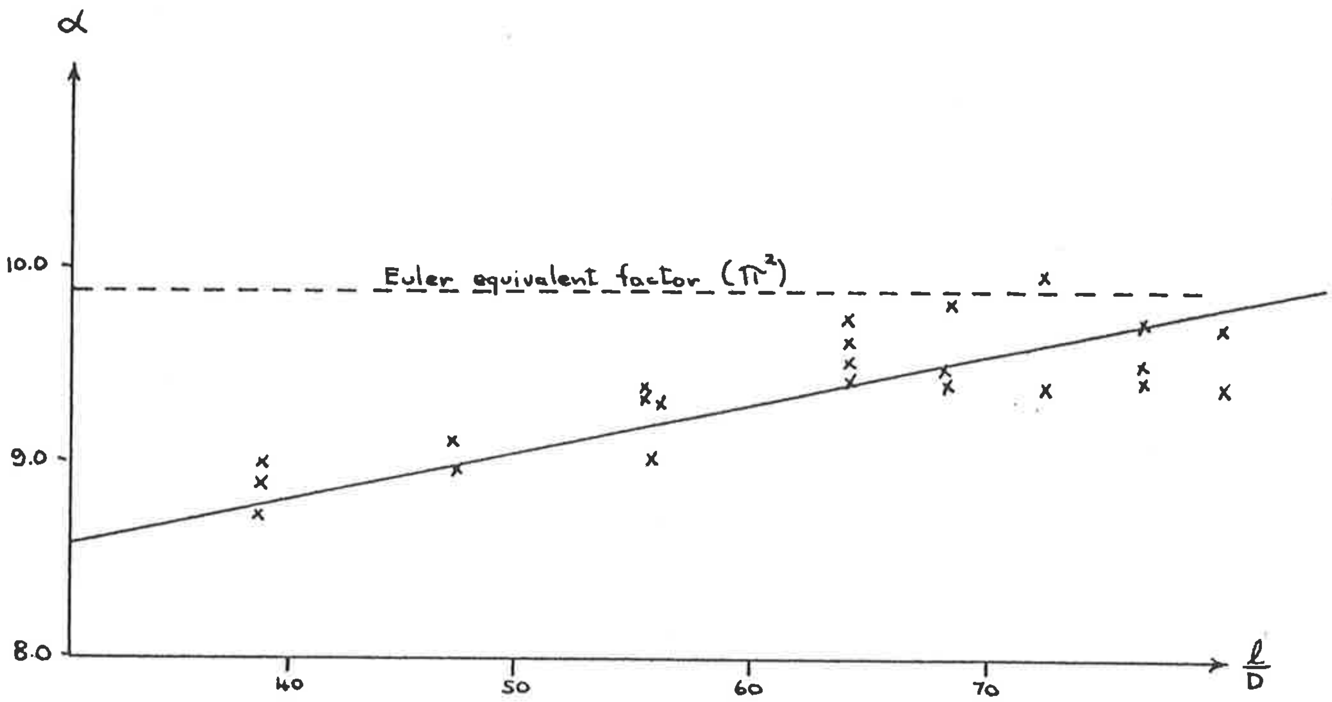


Figure 1.14 Variation of α Vs Slenderness ℓ/D

Note that r is approximately equal to .3D and so the above graphs are valid for an $\frac{\ell}{r}$ ratio from 100 to 263.

1.8 Comparing the Rationality of ACI and DIN/CP

The ACI magnifier method, which suggests that the loads on a column are amplified by the slenderness effect, appears to be more logical than the semi empirical moment addition method used in DIN and CP. However the ACI approach could be criticised on the following grounds:

(1) The evaluation of EI, and hence P_{cr}, uses empirical equations of limited accuracy.

(2) The ACI equations follow the linear elastic approach which leads to a very complicated sequence of design equations.

1.9 Shortcomings of the AS 1480 Method

The present AS 1480 slender column requirements deal inadequately with the following items:

(1) Secondary floor and beam moments induced by secondary column bending. These may be significant and ACI, CP and CEB require them to be considered.

(2) Biaxial bending. CP110 and ACI 318-71 contain various simplified procedures for biaxial bending.

(3) The adverse effects of creep. This is specifically accounted for in ACI, CP and DIN while AS 1480 has obviously allowed for it in the choice of R. DIN accounts for creep by the specification of an additional eccentricity. This approach is a good model of the physical effect of creep and is still simple.

(4) Overall stability. AS 1480 deals exclusively with single columns and no storey by storey check on stability is specified as is given in the ACI code.

(5) The slenderness of a column which is fully fixed or pin-ended. No provision is made in AS 1480 for such columns and they should be provided perhaps along the lines adopted in steel design.

1.10 Conclusions and Recommendations

The advantages of the AS1480 method are:

(1) It is simple to apply

(2) The concept of an additional eccentricity, which must be added to the design eccentricity, is to be preferred to the alternative concept of minimum eccentricity. In this way, the various effects explained in Section 1.3 are more logically taken into account.

The disadvantages of the AS 1480 method are:

(1) It is conservative over the whole range of its application

(2) It is less rational than either the additional moment or moment magnifier method

(3) It gives unrealistic results at the limit of the stated range of its application.

Although the AS 1480 method is much less complex than the ACI magnifier method it is not significantly simpler than the moment addition method of CP and DIN. The merit of CP and DIN lies in the choice of simple expressions and not in any basic difference between moment magnification and moment addition.

It is therefore recommended that in future revisions of AS 1480, consideration be given to the following:

(1) Removing the shortcomings listed in Section 1.9

(2) Reducing the load factors. This will reduce the conservatism of the current provisions which give a safe load typically 15 to 50 percent lower than calculated by ACI 318-71.

(3) Adopting the moment addition method as used in CP and DIN. This would allow greater rationality while maintaining a fair degree of simplicity. A consistent approach could be achieved by allowing for the following eccentricities:

(a) e_o – to give the applied moment (= $\frac{M}{P}$)

(b) e_{add} – to allow for out-of-straightness (= 25mm + .01 x D)

(c) e_d – to represent column deflection

(d) e_{creep} – to allow for creep

This section would then be designed for an internal force consisting of:

P = Axial Load

$$M = P\,(e_o + e_{add} + e_d + e_{creep}) \qquad .....1.10.1)$$

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

SECTION 2- INTEGRATION WITH GENESYS

2.1 Introduction

The CP110 design module currently used by GENESYS for Reinforced Concrete Columns is to be replaced by a more efficient version complying with AS 1480. This replacement module is shown in basic detail in Section 3 and in greater detail in the Programmer's Reference Manual, Appendix 5.

To enable this replacement module to be successfully integrated with GENESYS, it must conform to certain conventions relating to variable names and information to be passed between the various sub-programs.

This section shows how the module developed in Section 3 is integrated with GENESYS. It also shows how the original CP110 module operates and how efficiency has been improved.

2.2 Explanation of Column Subsystem of GENESYS

2.2.1 Introduction

All GENESYS subprograms which are relevant to column design and detailing are held within the subsystem called RC-COLUMN/1.

The original version of the subsystem (before alteration to conform to AS1480) contained 21 subprograms identified by both name and number.

The programs dealing with rectangular column design and analysis which were rewritten by myself to conform to AS 1480 and to achieve an increase in efficiency are:

. Program 52 OVERLAY "RC22"

. Program 54 OVERLAY "RC24"

In the original version, program 54 dealt exclusively with Biaxial loading design and although the programming for the replacement module could be placed in one module it was decided to retain both OVERLAYs to provide adequate storage for the 2000+ coefficients required for the replacement module. This ensures that the coefficients are not held in primary storage when they are no longer needed - the program enters the next OVERLAY once the correct subset of coefficients has been stored.

A summary of programs in the original version of RC-COLUMN/1 is shown in Table 2.1.

2.2.2 PUBLIC Variable System

To facilitate the passing of information between the various sub-programs GENESYS uses PUBLIC variables. These are similar to COMMON variables in FORTRAN but they have the advantage that they do not have to be DIMENSIONed and they are held in virtual storage. Thus unlike COMMON they are not part of the program and they may be expanded or reduced at will.

As with COMMON variables, PUBLIC variables have a blockname and a variable name. The block names used in the column subsystem and the programs which have access to them are shown in Table 2.2.

2.2.3 Operation of Column Subsystem

The sequence of events involved in the operation of the column sub- system is quite complex and a detailed account is not warranted here except to say that the subprograms are performed in basically the order of ascending program number.

This implies that any program which has a number higher than 52 may have to be checked to determine the effect of changes to program 52. In fact because of the way the PUBLIC system is organised it is only necessary to determine how the original module accessed the PUBLIC store and then to ensure that the replacement module acted likewise. This also included determining how subsequent programs use the information passed into the PUBLIC store accessed by program 52.

PROGRAM	TYPE	TITLE	FUNCTIONS
10	PROGRAM	CONTROL	List Subprograms
12	OVERLAY	RELEASE	Initiate message beginning subsystem
13	DEFINITIONS	RC-COLUMN/1	Set table headings and FORMAT statements
15	OVERLAY	MRADA	Assemble entry point, bases, walls and beams support check
16	"	MRADB	Assemble column, column sections, complete assemble command initiate distribution and summarise
20	"	MINIT	Set design options, material propeties and reinforcement costs
25	"	MSLECT	Select entry point, select slabs, flat-slabs and columns
27	"	MRAY1	Print command and other suppressible output
28	"	MRAY2	Assign titles of output tables in "MRAY1"
37	"	MSUM	Column load summation and entry point for design
38	"	MCGEOM	Column design - extract geometry from data bank
39	"	MCLOAD	Calculate loading combinations and initiate design
40	"	MSTORE	Store column geometry in interface
50	"	MAIN-COL-DESIGN	Primary design module
51	"	CIRCULAR COLUMN	Design of circular columns
52	"	RC22	Rectangular column design - monoaxial
53	"	COLUMN DETAIL	Detailing routine
54	"	RC24	Rectangular column design - biaxial
80	"	CRC DETAIL	Detailing of circular columns
81	"	REC COL DETAILING	Detailing of rectangular columns
82	"	REC COL SCHEDULE	Scheduling of rectangular columns

Table 2.1 - Summary of Subprograms in AS-Column/1

PROG	TITLE	PUBLIC BLOCK NAME											
		/BEAMS/	/CCD/	CIRC1/	/CIRC2/	/COLUMNS/	/COMMON/	/CR/	/FLAT/	/RCD/	/RCSYS/	/SBLK/	SLABS/
10	CONTROL												
12	RELEASE												
13	AS-COLUMNS/1												
15	MRADA						*						
16	MRADB												
20	MINIT	*					*		*				*
25	MSLECT												
27	MRAY1	*				*	*		*			*	*
28	MRAY2						*						
37	MSUM												
38	MCGEOM												
39	MCLOAD						*						
40	MSTORE						*						
50	MAIN-COL-DESIGN		*		*	*	*	*		*	*		
51	CIRCULAR COLUMN			*	*	*	*	*					
52	RC22					x		x			x		
53	COLUMN DETAIL					*							
54	RC24					x		x			x		
80	CRC DETAIL	*	*			*	*	*					
81	REC COL DETAILING	*				*	*	*		*			
82	REC COL SCHEDULE					*		*		*			

Table 2.2 Public Blocks accessed by programs in RC-COLUMN

* = accessed   x = accessed by altered program

2.3 Explanation of Original Module for Rectangular Column Design

2.3.1 Introduction

The original column design module is contained within OVERLAYs 52 and 54 and consists of 20 subroutines of which 5 are used by both OVERLAYs. These subroutines are summarised in Tables 2.3 and 2.4.

2.3.2 Design Philosophy

The module allows for a minimum moment value of 0.06Pd in the appropriate direction and if both moments exceed a value of 0.03Pd then the column is designed for biaxial bending. Note that P is the applied axial load and d is the effective depth of the column section.

Six different bar sizes in the range 40 to 12mm are used and it is possible to have two different bar sizes in the column - this will not be allowed in the replacement module because it is felt that the added complexity is not worth the steel saved.

The procedure then involves first sorting (and discarding redundant loads) the loads into descending order of applied axial load. Some improvements of this sorting procedure have been suggested in the current module's operation handbook and they have been included in the replacement version.

4 bars are selected so that they do not exceed the largest size permitted by the user and so that their area does not exceed 6% of the gross concrete area and that they do not violate the spacing limitations.

The first load case is then selected (the one with the highest axial load) and a series of design solutions is generated with different size corner bars (and internal bars if necessary) so that the given area is within 15% of the required area. This limit has been chosen to ensure an economical design. Note that this requires an iterative solution because the neutral axis position depends on the amount of steel present.

If the load is biaxial, the column is designed for an equivalent combined bending moment as specified in AS1480.

Serial	Subroutine Name	Size in Statements	Tasks
1	CENTRY	143	Main entry routine, control the calling sequence for performing the analysis and design on the column lift under consideration.
3	DRIVER	192	Initiates the analysis and design by calling the appropriate routine for the load case under consideration.
3	SORT	129	Sorts the load case in the descending order of the axial load applied. Also eliminates the redundant load case in the loading arrays.
4	FNL	3	General utility function.
5	BCLASH	33	Beam bar clashing avoidance routine.
6	EFLTH	58	Calculation of the effective length of the column under consideration.
7	MONOAXIAL	306	For the complete analysis and design of the column subjected to mono-axial load cases.
8	ADD	32	For calculating the additional moment if column under consideration is slender.
9	FNI	3	General utility function.
10	STCALC	57	Initialises the solution table.
11	ITERATE	179	To define the neutral axis for the column cross-section in order to calculate the nearest area of steel that can carry the load under consideration.
12	SEARCH	50	Adjustment of the neutral axis position.
13	ARRANG	57	Arranging the steel area computed according to the alternative bar sizes.
14	KRITICALD	50	Calculate the critical load value for the different bar combinations obtained.
15	SPCL	88	Calculate area of steel for lightly loaded columns.

Table 2.3 Subroutines contained in original OVERLAY 52

Note that in the latest version of RC-COLUMN/1, OVERLAY 52 has been broken up into a number of smaller overlays. However, the design philosophy remains the same.

Serial	Subroutine Name	Size in Statements	Tasks
1	BIAXIAL	256	Entry to the overlay, it completes the analysis and design of the column under consideration which is subject to biaxial load cases.
2	LAP	370	Initiates the calculation or checking of each load case.
3	BIADD	41	For calculating the additional moment if column under consideration is slender.
4	FNI	3	General utility function.
5	FNL	3	General utility function.
6	STCALC	57	Initialises the solution table.
7	ITERATE	179	To define the neutral axis for the column cross-section in order to calculate the nearest area of steel that can carry the load under consideration.
8	SEARCH	50	Adjustment of the neutral axis position.
9	ARRANG	57	Arranging the steel area computed according to the alternative bar sizes.
10	KRITICALD	50	Calculate the critical load value for the different bar combinations obtained.

<u>Table 2.4 Subroutines contained in original OVERLAY 54</u>
(Biaxial Load Design)

* Note that in the latest version of RC-COLUMN/1, OVERLAY 54 has been split into a number of smaller overlays.

Subsequent load cases are all checked against the generated solutions which are revised if necessary by the addition of extra face bars. A solution may be marked as "not possible" if it exceeds the required area by more than 15% or if it violates the spacing requirements.

The solution chosen for detailing is the cheapest (accounting for the variation of cost due to bar size, lap length and link size and spacing) unless the user has specified a preferred size in which case the cheapest solution with that size of corner bar will be chosen.

2.3.3 Selection of load cases for Design

The subroutine which performs the load sorting function is called SORT and is not to be confused with the SORTL routine used in the replacement module.

It is one of the three routines from the original module which have been retained (although in altered form) for use in the replacement module. It appears as the routine SORT in the replacement module and is discussed fully in Section 3.9.

2.3.4 Avoidance of bar clashing

This is the second of the three routines retained from the original module. Its operation is described in Section 3.6 under the routine BCLASH, and in greater detail in Appendix 6.

2.3.5 Calculation of effective lengths

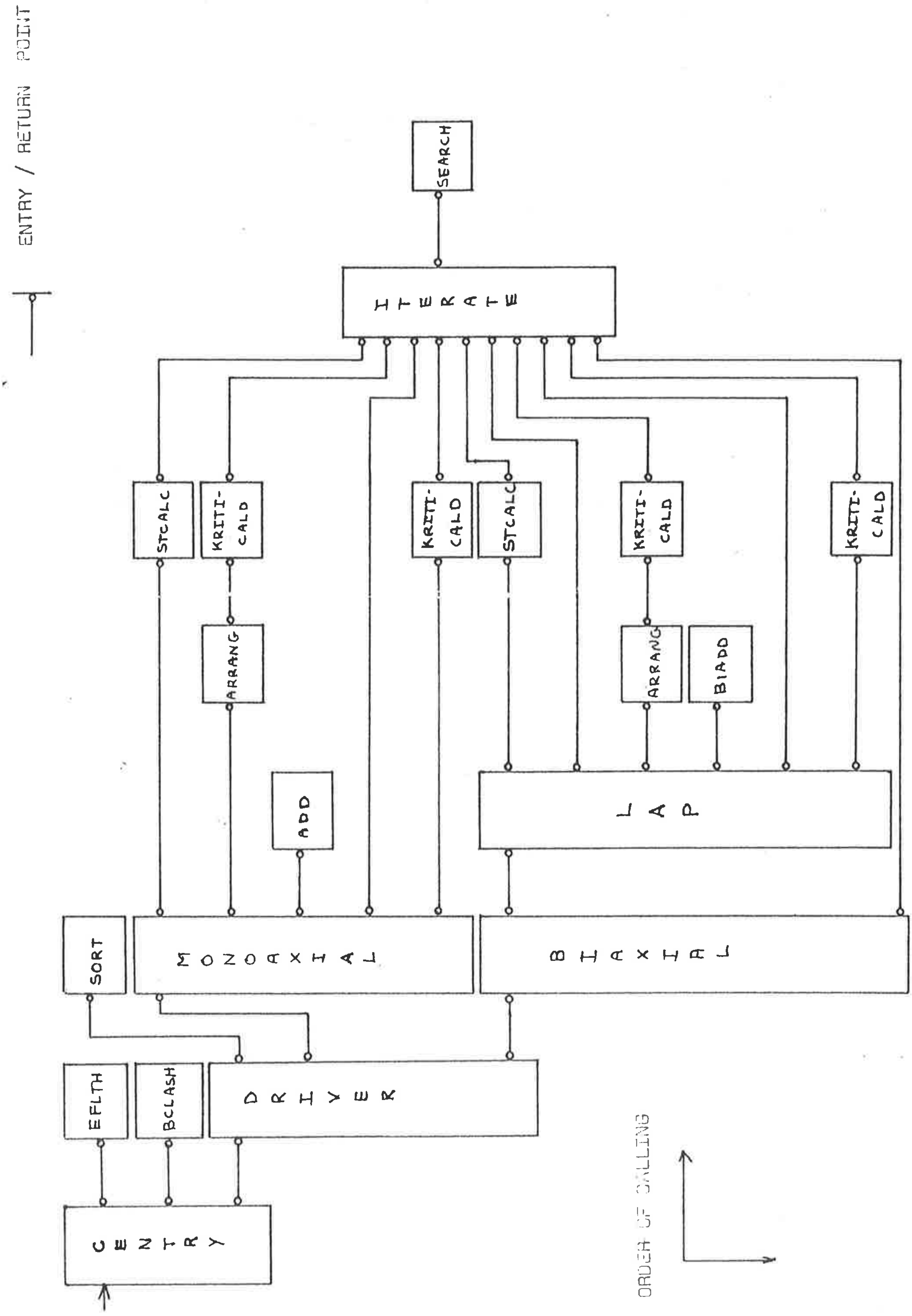
This is the third routine retained from the original module. Its operation is described in Section 3.5 under the routine EFLTH.

2.3.6 Basic Flow Chart

A diagram showing the subroutine calling sequence of the original module is shown in Figure 2.1 and may be compared with that of the replacement module shown in Figure 3.2.

FIGURE 2.1 Subroutine Calling Sequence of Original Module

Note: Some routines have been shown twice for clarity



2.4 Details of (Original Module/Public Variable) Interaction

In order to integrate the replacement module a proper understanding of how the original module relates to itself and other parts of GENESYS is required. Although a number of local variables are used they will not be considered because they are used in the internal workings of the programs and not in the passing of information.

Table 2.2 summarises the PUBLIC block names accessed by programs in RC-COLUMN/1. As shown, only variables in the following block names need be considered:

. /COLUMN/

. /CR/

. /RCSYS/

In addition, variables within these blocks which do not apply to rectangular columns have been omitted from consideration, and those parts of array elements which do not apply have been marked as not relevant. This includes parts referring to circular and L-shaped columns.

The variables used in /COLUMN/,/CR/,and /RCSYS/ are described in detail in appendix 5. Access to them is shown in Table 2.5.

Table 2.5 Public Variable Usage Summary of Original Module

(i.e. OVERLAY 52 and 54)

S - Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	DRIVER	BCLASH	EFLTH	MONOAXI	ADD	STCALC	ITERATE	SEARCH	ARRANG	KRITICA	BIAXIAL	LAP	BIADD
/COLUMNS/														
CSP(,)	U													
GL()		U		U										
IGL(,)	U	U												
IP(,)	U													
KOLUMN	U	U		U										
P(,)	U	U												
ST()	U	U	U		U									
MS()	U	U	U	U	U							U		
/RCSYS/														
IISYS		S			U							U		
ISPCL		S			U							U S		
ISYS	S	U										U		
LDBT()		S										U		
NSYS	S	U			U							U		
SLD(,)		U S			U									

Table 2.5 Public Variable Usage Summary of Original Module

(i.e. OVERLAY 52 and 54)

S = Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	DRIVER	BCLASH	EFLTH	MONOAXI	ADD	STCALC	ITERATE	SEARCH	ARRANG	KRITICA	BIAXIAL	LAP	BIADD
/CR/														
AA(,)		S			U					U	U	U	U	
ACRIT								S			U		U	
ADW()					S						S	S	U	
											S			
ASC					U		S	U		U		U	U	
						S								
B					U		U					U	U	
			S							S				
B1	S	U	U	U	U							U	U	
B2	S	U	U	U	U							U	U	
CLD(,)		S			S							U	U	
										S	S			
CONS1					S							S	U	
			U											
CONS3					S	U		U				S		
CONS4		S												
CONS5		S			U	U		U						
COV	S				U					U	U	U	U	
CRITLD(,)					U		S				U		U	
									S		S			
DIAM()		S			U						U	U	U	
EL				S										
EX		U		S									U	
EY		U		S									U	
FCU	S	U			U	U	U	U				U	U	U

Table 2.5 Public Variable Usage Summary of Original Module

(i.e. OVERLAY 52 and54)

S = Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	DRIVER	BCLASH	EFLTH	MONOAXI	ADD	STCALC	ITERATE	SEARCH	ARRANG	KRITICA	BIAXIAL	LAP	BIADD
/CR/ (Contd)														
FIVE		S			U			U				U	U	
FK(,)					S								S	
FY	S	U			U	U	U	U					U	
H					U	U	U	U				U	U	
IBARS(,)				S	U		S			S	S	U	U	
ICNT				S	S		S			U	S U	S U	U	
IERR	S				S				S			S	S	
INIT		S			S							S	U	
IWARN	U S				S						U	S S		
IZ														
KSXY	S		U		U					U			U	
MBI					S		U	U			S	S	U	
NB1			S							U U		U	U	
NB2			S							U		U	U	
NBI					S			U			S S	U	U	
NOER()					S				S	U		S S	U	
NOWR()					S							S		
NT1		S			S							U	U	
NUTRAL								S						

Table 2.5 Public Variable Usage Summary of Original Module

(i.e. OVERLAY 52 and 54)

S = Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	DRIVER	BCLASH	EFLTH	MONOAXI	ADD	STCALC	ITERATE	SEARCH	ARRANG	KRITICA	BIAXIAL	LAP	BIADD
/CR/ (Contd)														
OR								S					U	
OU								S					U	
PHI					U	U		U						
PCLD()		S			S S								S	
PP(,)	S	D												
SFS1		U S			U	U		U					U	U
SFS2		S			U	U		U	U				U	U
SFS3		S			U	U					U			
SXY(,)			S											
TBLE(,)					S		S			U		U S	U S	
XSL	U	U		U S										
YSL	U	U		S										
ZUN()					S								S	
CAM()												U U	S	
CA1M()												U	S	
CA2M()												U	S	
ALP()												U	S	

2.5 Details of Information Supplied to System by Original Module

2.5.1 Introduction

This section discusses the information which is supplied to the system by the original module and which must therefore be supplied by the replace module.

2.5.2 Subsystems Involved

From Table 2.2 the following subsystems were examined because they have access to the same PUBLIC blocks as does the original module:

(a) Program 27 "MRAYI"

(b) Program 50 "MAIN-COL-DESIGN"

(c) Program 53 "COLUMN DETAIL"

(d) Program 81 "REC-COL-DETAILING"

(e) Program 82 "REC-COL-SCHEDULE"

2.5.3 Variables Involved

Table 2.6 lists all of the variables in the BLOCKS /COLUMNS/, /CR/, /RCSYS/ which are used by the above overlays. It indicates whether they were set in the original module and, if so, by which particular subroutine.

Table 2.6 Variables in BLOCKS /COLUMNS/, /CR/, /RCSYS/ used outside of original module

Variable	Name of Subroutine(s) in which variable is set	Variable	Name of Subroutine(s) in set which variable is
/COLUMNS/		/CR/ (Contd)	
CSP(,)	Set in interface	CLD(,)	DRIVER; MONOAXIAL; BIAXIAL; LAP
GL()		COV	CENTRY
GL(,)	"	DIAM()	DRIVER
IP(,)	"	FCU	CENTRY
KOLUMN	"	FK(,)	MONOAXIAL
P(,)	"	FY	CENTRY
ST()	"	IBARS(,)	MONOAXIAL; STCALC; ARRANG; BIAXIAL;
MS()	"	ICNT	MONOAXIAL; STCALC;
		NOER()	MONOAXIAL; SEARCH; BIAXIAL
/RCSYS/			
		NOWR()	MONOAXIAL; BIAXIAL
ISPCL	DRIVER		
		NUTRAL	ITERATE
SLD(,)	DRIVER		
		PCLD()	DRIVER; MONOAXIAL;
		TBLE(,)	MONOAXIAL; STCALC; BIAXIAL; LAP
/CR/			
AA(,)	DRIVER	XSL	BCLASH
		YSL	BCLASH
B1	CENTRY	ALP()	LAP
B2	CENTRY		

2.6 Details of Public Block System adopted for Replacement Module

2.6.1 Introduction

The Public Block system remains essentially the same in the replacement module as for the original module. It was not considered feasible to reduce the number of Public Blocks because each is used elsewhere in the system. However the variables within the blocks have been altered to remove those which have become redundant and to add several which are needed by the replacement module. The following parts of this section examine each Block separately and indicate which variables have been retained, deleted or added. A complete summary is shown in Table 2.7.

2.6.2 Block/COLUMNS/

All have been retained.

2.6.3 Block/RCSYS/

Remove ISYS, LDBI(I) and retain all others. Note however that ISPCL has no significance because the term "light load" has no meaning under the design philosophy of the replacement module. However it is used elsewhere in the system for reference so it is retained but set to zero.

2.6.4 Block/CR/

The following variables have been retained, + indicates that the variable is used only by the system and is not really necessary for replacement module.

-AA(,) +
-ASC
-B1
-B2
-CLD (,)+
-COV
-DIAM ()
-EL
-EX
-EY
-FCU
-FIVE
-FK(,)+

-FY
-IBARS(,)
-ICNT
-IERR
-IWARN
-KSXY
-NB1
-NB2
-NOER ()
-NOWR()
-NUTRAL
-PCLD()
-PP(,)
-SXY(,)
-TBLE(,)
-XSL
-YSL
-ALP()+

The following variables have been omitted:

-ACRIT
-ADW()
-B
-CONS1
-CONS3
-CONS4
-CONS5
-CRITLD(,)

-NBI
-NT1
-OR
-OU
-PHI
-SFS1
-SFS2
-SFS3

-H	-ZUN()
-INIT	-CAM()
-IZ	-CA1M()
-MBI	-CA2M()
-N60	

The following variables have been added, a complete description of which appears in the Programmer's Reference Manual, Appendix 5.

BIAX(,)	: Contains the considered Biaxial Load cases
COEFB(,)	: Contains the coefficients of the balanced failure line shown on the design charts
COEFP(,,)	: Contains the coefficients of the equations representing the failure curves on the design charts
ECX	: Initial eccentricity in X-direction
ECY	: Initial eccentricity in Y-direction
IBIAX	: Initial number of Biaxial cases found
IFLAG	: Pointer to the Loading type being considered
IMON	: Initial number of Monoaxial cases found
MONX(,)	: Contains the considered Monoaxial-X load cases
MONY(,)	: Contains the considered Monoaxial-Y load cases
NBUNDX()	: Contains the number of bundled X-Face bars (if bundling is permitted)
NBUNDY()	: Contains the number of bundled Y-Face bars (if bundling is permitted)
NSYSB	: Total number of Biaxial Load cases in BIAX
NSYSX	: Total number of Monoaxial-X load cases in MONX
NSYSY	: Total number of Monoaxial-Y load cases in MONY
PMAX	: Intercept of P=8% curve on the P/BD axis on Design charts

PMIN : Intercept of P=0% curve on P/BD axis on Design Charts (= 0.595 * FCU)

STEELP(,) : Steel % to cover all loading cases

SLOPE() : Slope of failure curves at the instant they cross the M=0 axis

Table 2.7 Public Variable Usage Summary of Replacement Module

S = Set by, U = Used by

VARIABLE	SUBROUTINE NAME													
	CENTRY	EFLTH	BCLASH	RTVP	DRIVER	SORT	COLLOAD	FINDP	ARRANG	SORTL	RED	CALCP	FUNCP	FUNCD
/COLUMNS/														
CSP(,)	U						U							
GL()		U			U									
IGL(,)	U				U									
IP(,)	U													
KOLUMN	U	U			U									
P(,)	U				U									
ST()	U		U		U									
MS()	U	U	U		U									
/RCSYS/														
ISYS					S		U							
ISPCL					S									
ISYS	S				U		U							
NSYS	S				U		U							
SLD(,)					S U	U	U					U		

Table 2.7 Public Variable Usage Summary of Replacement Module

S = Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	EFLTH	BCLASH	RTVP	DRIVER	SORT	COLLOAD	FINDP	ARRANG	SORTL	RED	CALCP	FUNCP	FUNCD
/CR/														
AA(,)					S									
ASC									S					
B1	S	U	U	U	U		U		U		U			
B2	S	U	U	U	U		U		U		U			
BIAX(,)							S	U						
CLD(,)					S				S					
COEFB(,)				S							U			
COEFP(,,)				S									U	U
COV	S			U					U					
DIAM()					S									
ECX,ECY							S							
EL		S					U		U		U			
EX		S			U						U			
EY		S			U						U			
FCU	S			U	U					U	U			
FIVE					S									
FK(,)									S					
FY				U	U						U			
IBARS(,)									S					
IBIAX							S							

Table 2.7 Public Variable Usage Summaryof Replacement Module

S = Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	EFLTH	BCLASH	RTVP	DRIVER	SORT	COLLOAD	FINDP	ARRANG	SORTL	RED	CALCP	FUNCP	FUNCD
/CR/ (Contd)														
ICNT									S					
IERR	S			S	S			S	S					
IFLAG							S	S			U	U	U	U
IWARN	S			S	S									
IMON							S							
KSXY	S		U						U					
MONX(,)							S	U						
MONY(,)							S	U						
NB1			S						U					
NB2			S						U					
NT1		U			S									
NBUNDX()									S	(if bundling is permitted)				
NBUNDY()									S	(if bundling is permitted)				
NOER()	S			S	S			S	S					
NOWR()	S			S	S									
NSYSB							S	U						
NSYSX							S	U						
NSYSY							S	U						
NUTRAL						U			S		S			

Table 2.7 Public Variable Usage Summary of Replacement Module

S = Set by U = Used by D = Destroyed by

VARIABLE	SUBROUTINE NAME													
	CENTRY	EFLTH	BCLASH	RTVP	DRIVER	SORT	COLLOAD	FINDP	ARRANG	SORTL	RED	CALCP	FUNCP	FUNCD
/CR/ (Contd)														
PCLD()					S									
PMAX				S							U			
PMIN				S							U			
PP(,)	S	D			U									
STEELP(,)								S	U					
SXY(,)			S						U					
TBLE(,)									S					
XSL	U	S			U									
YSL	U	S			U									
ALP()									S					

2.7 Conversion of Replacement Module Output for use by System

2.7.1 Introduction

Because the replacement module tackles the design in a different way the initial output is different from that of the original module.The replacement module calculates the bars required directly whereas the original module works by adding one bar at a time until the column possesses adequate strength. For this reason the output of the program detailed in Section 3 is not immediately compatible with the rest of the system. This section explains how the replacement module output is tailored to meet the requirements of the rest of the system.

2.7.2 Subroutines Involved

All of this output modification is done in the Subroutine ARRANG because it is here that the design solution is determined.

2.7.3 Variables Involved

The variables involved are listed below and are those variables appearing in Table 2.6 which have not already been set by other parts of the module.

IBARS(,)	ALP()
CLD(,)	ICNT
FK(,)	TBLE(,)

2.7.4 Method of setting the variables

All variables are set just before leaving ARRANG.

2.7.5 Variable CLD(I,J)

This contains the computational results of load case J in array SLD(,). The values for I = 2,8 are set in DRIVER as in the original module. I = 1 contains a "0" if the load case is not critical and "1" if the load case is one used for determining the steel. The load case numbers used for determining the steel are held in the array STEELP(I,2),I=1,3 therefore set:

```
DO I = 1,3
CLD(1,STEELP(I,2))=1     (Note that a dummy integer variable
                          is used for STEELP)
```

CLD(I,I),I = 9 and 10 contains the design moments. In the original module these were set in DRIVER but now they are set in ARRANG as follows:

DO I = 1,NSYS

CLD(9,J)=CLD(7,J)+CLD(6,J)*ECX * 10^{-6}

CLD(10,J)=CLD(8,J)+CLD(6,J)*ECY * 10^{-6}

Design Moment = Applied moment + Force * eccentricity

2.7.6 Variable FK(,)

This is the adjustment factor finally used in the original module in conjunction with the additional moment induced in the column by its deflection.

This variable is thus meaningless in the terms of the replacement module because here the initial eccentricity is independent of the steel. However the variable must be retained because it is used elsewhere in the system. It is set = -1.0 for all members.

2.7.7 Variable IBARS(I,J)

This contains the number of bars required for arrangements obtained in the run. It is initialised to contain the symbol "BL" for all bar sizes. The replacement module version is simpler than the original module version because all arrangements contain only bars of the same size.

J = 1,7 : Corner bar size (36mm to 12mm)

I = 1,7 : Intermediate bar size in X-direction (36mm to 12mm)

I = 8,14 : Intermediate bar size in Y-direction (36mm to 12mm)

In the replacement module the number of Y-face bars is = IFACEY(I) and the number of X-face bars = IFACEX(I). Note that two of these bars are corner bars.

The replacement module indicates an invalid solution by setting COST(I) = 10 000. Thus before setting IBARS(,) first a check must be made that COST = 10 000.

2.7.8 Variable ICNT

This is the number of design solutions obtained for the column. It is set while setting IBARS(,) as above.

2.7.9 Variable TBLE(I,ICNT)

This contains the analysis results of the column. ICNT is the solution number and may range from 1 to NT1, the maximum user permitted bar size. If there are no valid solutions then ICNT=0 and TBLE(,) is undefined. I ranges from 1 to 9 and takes the following values (provide that COST(J) = 10 000):

I = 1: This is the area of steel required when using the bar diameter specified as TBLE(7,ICNT)

In the replacement module, this area is independent of ICNT and so set TBLE(1,ICNT) = maximum steel required.

I = 2: This is the Applied moment direction
= 1 for monoaxial-x; =2 for monoaxial-y; =3 for biaxial. It is set according to whichever requires the larger steel area. However it must be noted that although one direction is indicated, significant moments in the other direction may also occur.

Again this value is independent of ICNT.

I = 3: This is the value of the design moment in the X-direction and is independent of bar size.

$$TBLE(3,ICNT) = CLD(9,STEELP(1,2)) * B1 * B2^2 * 10^{-6} \text{ kNm}$$

I = 4: As for I = 3 but in the Y-direction.

$$TBLE(4,ICNT) = CLD(10,STEELP(2,2)) * B1^2 * B2 * 10^{-6} \text{ kNm}$$

I = 5: This is the value of the design axial load and is independent of bar size. Because the axial load may not be the same for the Mono-X and Mono-Y cases, it has been decided to set it to the maximum.

TBLE(5,ICNT) = MAX(CLD(2,STEELP(1,2)),CLD(2,STEELP(2,2))) kN

I = 6: This is the maximum allowable moment in the X-direction and depends on the bars present. In calculating this value, the axial load applied is that used in determining the steel required.

The algorithm used in this calculation relies on the assumption that, at a particular axial stress value, the distance between adjacent steel % lines is independent of the steel %. The algorithm is shown in detail at line 588 of ARRANG in Appendix 4

I = 7: This is the bar size marker and is set to the bar diameter.

TBLE(7,ICNT)=(8+(4x(NT1-J+1)))

whereJ = bar size indicator

I = 8: This is the maximum allowable moment in the y-direction. The same procedure as in I = 6 is used.

I = 9: This is a table marker, set = ICNT.

2.7.10 Variable ALP(,)

This is the value of ALPHA used when combining the moments to create biaxial load cases. In the replacement module it is independent of bar size.

In the CP110 version for biaxial design the moments are combined as follows:

$$M_e = \left[(M_x)^{\alpha} + (M_y)^{\alpha} \right]^{1/\alpha} \qquad \ldots\ 2.1.1$$

In effect, AS1480-1974 has adopted $\alpha = 1$ for all cases thus:

ALP(ICNT, ICNT) = 1

2.8 Adaption of Original Module Subroutines to Suit Replacement Module

2.8.1 Introduction

Of the 15 subroutines making up the Original Module, 5 have been retained either whole or in part for the Replacement Module. This section discusses what changes, if any, are to be made.

2.8.2 Structural Changes

As discussed previously a major structural change has been made to cater for the storage of the chart coefficients. Briefly this is to put CENTRY, EFLTH, BCLASH and RTVP in OVERLAY(52) and the remaining ones in OVERLAY (54). The previous module had most of the program on OVERLAY (52) and just the biaxial programming on OVERLAY (54). This new arrangement does not have the considerable duplication of program that was previously required. The SUBROUTINE CENTRY now contains a PERFORM statement to effect the jump to the second overlay.

2.8.3 Subroutines Involved

The following subroutines were retained unaltered:

EFLTH, BCLASH, SORT.

The following required modification:

CENTRY, DRIVER.

2.8.4 Subroutine CENTRY

The following summarizes what is retained or omitted:

(a) Adjust the Public Variable List to suit.

(b) Retain all programming, including initialisation of certain variables, calling EFLTH and BCLASH,creation of the design loading array by the combination of the head and foot moment, and retrieval of the material properties for the current lift.

(c) Add a call to the subroutine RTVP to retrieve chart coefficients.

(d) Add a PERFORM command to transfer control to OVERLAY 54 "DRIVER" when OVERLAY 52 is complete.

2.8.5 Subroutine DRIVER

The following summarizes what is retained or omitted:

(a) Adjust the Public Variable List to suit.

(b) Remove the initialisation of: RFCU, SFS1, SFS2, SFS3.

(c) Change the units to Newtons and millimetres.

(d) Create sorted load array as before - CALL SORT.

(e) Transfer all loads from SLD(,) to SLDTEMP(,) and then back to SLD(,) in their correct order (of decreasing axial load). At the same time loads which were indicated by SORT to be redundant are omitted and NSYS is set to the revised number of loads in SLD(,). Note that redundant loads have been given an axial load value less than 0 by SORT. (See Section 2.1 Engineering Logic). This is necessary because in the original version, DRIVER took the loads from SLD(,) in their correct order and initiated the design sequence. The actual order of loads in SLD(,) was unimportant. In the replacement version, DRIVER has no control over the order of consideration hence the loads are put in their correct order in SLD(,) before the design sequence is called.

(f) Remove the programming which includes loadings from the column lift above as given in Section 2.1. This is indicated by a test for the status of KOLUMN.

(g) Add the following (with appropriate error detection):

CALL	COLLOAD
CALL	FINDP
CALL	ARRANG
PERFORM	''MAIN-COL-DESIGN'' CROUT

2.8.6 Overlay 52 "RECTANGULAR COLUMN" RC 22

(a) Delete the following subprogram list:

CENTRY, DRIVER, SORT, EFLTH, MONOAXIAL, ADD, KRITICALD, ARRANG, SEARCH, ITERATE, STCALC, FNI, FNL, BCLASH.

(b) Add the following subprogram list:

CENTRY, EFLTH, BCLASH, RTVP.

(c) Delete DRIVER from the ENTRIES LIST.

2.8.7 Overlay 54 "RC24"

(a) Delete the following subprogram list:

BIAXIAL, LAP, BIADD, KRITICALD, ARRANG, SEARCH, ITERATE, STCALC, FNI, FNL.

(b) Add the following subprogram list:

DRIVER, SORT, COLLOAD, FINDP, ARRANG, SORTL, RED, CALCP, FUNCP, FUNCD.

(c) Delete BIAXIAL from the ENTRIES list and insert DRIVER instead.

2.9 Error Message Summary

2.9.1 Introduction

A good method for reporting and locating errors is essential in a large program such as the one which has been developed for column design. The original module made use of a separate array which summarised the errors but for reasons explained below, the replacement module has included a greater variety of error messages. This section gives details of the error messages produced and their implications.

2.9.2 Errors detectable by the Original Module

The error message system of the original module is based around the arrays NOER() for errors and NOWR() for warnings. The whole system has access to these arrays and no doubt other modules make use of them but as far as the original module is concerned NOER() is only used to store any failed loadings and NOWR() is used only if the bar size chosen conflicts with the user given preferred size.

Any errors or messages which are detectable and relate to the original module are summarised by the SUBROUTINE WREMSG(S) occuring in OVERLAY 50 "MAIN-COL-DESIGN".

In addition to the error and warning arrays, there are also two cases of a directly printed error message. The following summarizes the possible error/warning messages produced by the original module subroutines:

Subroutine	Error/Warning
CENTRY	None
DRIVER	None
SORT	None
BCLASH	None
MONOAXIAL	Set NOWR( ) = 3 if user preferred size is not possible = 6 if user preferred size is allowable Set NOER( ) = I*100 if the loading fails (where I = loading number). Abort run with a printed message if there is in-sufficient space in the section for 4 bars of the minimum size permitted by the designer
ADD	None
EFLTH	None
STCLAC	None
ITERATE	None

SEARCH	Set NOER( ) = -200 if there is a failure in in the neutral axis search
ARRANG	Abort run with a printed message if there is an error in the steel arrangement.
KRITICALD	None
BIAXIAL	As for MONOAXIAL except set NOER( ) =-100 if even the smallest possible sizes will not fit.
LAP	Set NOER( ) = I x 100 if loading fails NOER( ) = -100 if the bar size fails
BIADD	None

2.9.3 Errors detectable by the replacement module

As in the original module the error system for the replacement system consists of:

(a) Arrays to store the information for future printing.

(b) A direct message to the printer or user.

In the replacement module (a) has been retained to store load numbers of failed loadings and (b) has been used much more extensively to facilitate quick location of errors. Failed loadings are exclusively those which require a steel % greater than 8. Arrangement errors are now reported directly and are not associated with a particular loading number. Also the message associated with a neutral axis search failure is obviously no longer needed.

The errors detectable by the various subroutines are shown as follows:

Subroutine	Error Message
CENTRY	None
EFLTH	None
BCLASH	None
RTVP	Invalid FCU or FY Invalid G value
DRIVER	None
SORT	None
COLLOAD	None
FINDP	Set NOER( ) = I* 100 where I is the loading number which required the largest steel % above .08 for each loading type.
ARRANG	If detailing should fail for any reason then a message and a print statement are sent indicating that the column is too small to suit detailing. Set NOWR( ) = 6 if the user specified size has resulted in a solution. Set NOWR( ) = 3 if the user specified size has been rejected
SORTL	None
RED	None
CALCP	None
FUNCP	None
FUNCD	None

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

SECTION 3 - COLUMN DESIGN MODULE

SECTION 3 - COLUMN DESIGN MODULE

3.1 Introduction

This Section explains in detail the operation of a computer program module which designs rectangular concrete columns according to the provisions of AS1480-1974.

The design method is based on that adopted in the A.R.C. Design Handbook (Ref 15) and uses a digital representation of the charts therein to avoid the necessity for lengthy iterations to calculate the neutral axis depth.

Although this module is intended to replace one of similar function in the GENESYS column sub-system, several of the original subroutines have been retained mainly to facilitate integration of the replacement module. These subroutines were altered slightly to suit the new design method and they are explained along with the new subroutines.

3.2 The Engineering Logic

3.2.1 Introduction

The analysis method used in this rectangular column design module is based on that given in the A.R.C. Design Handbook (Ref 15). The design charts therein have been utilised by determining the failure curve equations as shown in Appendix 1. The detailed explanation of how the charts are used is shown in the Programmer's Reference Manual (Appendix 5) in the discussion of the subroutines which use them.

3.2.2 General Notes

(a) AS1480 requires that all members subjected to combined bending and compression shall be investigated for slenderness effects. The module therefore calculates the l/r ratio from the physical dimensions of the column and uses the code equations to calculate the reduction factors R and R'. Note that the beam/column stiffness ratio has been calculated elsewhere in the GENESYS RC-BUILDING subsystem and is used to calculate the effective length of the column as required.

(b) The ultimate failure curves shown in the A.R.C. Design Chart C5 are used for the analysis. The 20 charts with steel on two faces are used for the monoaxial bending cases and the 20 charts with steel on four faces are used for the bi-axial cases. It may be noted that these curves already include the capacity reduction factor ϕ.

(c) A bi-axial case is formed when both of the moments exceed a value of .03P'D. This is the CP110 stipulation which has been adopted because AS1480 does not define when a loading is to be classed as biaxial. If either M'_x or M'_y is less than .03P'D then a monoaxial loading pair is formed, each of which is assumed to act independently of the other.

(d) For columns in biaxial bending, the moments about both axes are added and the design is based on the charts for equal steel on four faces.

(e) The module makes allowance for the requirements of minimum eccentricities stated in AS1480, Rules 9.13.3(b) and (c). In these rules the column is required to be designed for an additional eccentricity of 25 + .01D (mm) in the direction of the major axes. In addition it is required to carry, separately a moment arising from the action of the loading condition on a similar initial eccentricity about the minor axis. These requirements are met during the creation of the Monoaxial and Biaxial load cases to be tested.

3.2.3 Column Design Procedure

The design sequence is as follows:

(a) Retrieve the column dimensions, loadings and material properties from the RC-BUILDING interface.

(b) Compute the effective length of the column group and the l/r ratios.

(c) If required then set up the relevant information required for beam bar clashing avoidance. See Section 3.2.5.

(d) Combine the head and foot moments as specified in Section 3.2.4 depending on column slenderness.

(e) Retrieve the coefficients for the charts which will be required depending on the concrete strength (FCU) and the steel strength (FY). Then form a coefficient table of three charts (Mono-X, Mono-Y and Biaxial) depending on the values of GX, GY and GXY.

(f) Sort the loads in descending order of axial load as specified in Section 3.2.4. This is not necessary for the analysis but simplifies the determination of unnecessary loadings.

(g) Create the Monoaxial and Biaxial Loading cases as follows:

Mono-X - P'; M'_x + P' * ECX ; ILX

Mono-Y - P'; M'_y + P' * ECY ; ILY

Biaxial - P'; (M'_x + M'_y + P' * ECXY); ILB

where:

(i) ECX, ECY and ECXY are the initial additional eccentricities.

(ii) The Biaxial case is only formed if:

$M'_x > 0.03 * P' * B2$

and

$M'_y > 0.03 * P' * B1$

If a Biaxial case is formed then the mono-axial cases are not.

These limits were chosen because they appear in CP110 and are not defined in AS1480.

(iii) ILX, ILY and ILXY are the load numbers in SLD(,) from which the load was derived.

(h) From each of these loading cases discard those loadings for which:

(i) The required steel % is zero (see Section 3.2.6) or

(ii) There is another load case with the same axial force and a higher moment.

(i) Apply the reduction factors R and R' to the remaining loads.

(j) Calculate the steel percentage for each of the loading types that will satisfy the worst loading condition in that group. Also store the loading number which caused the worst condition.

(k) For each of the bar sizes permitted (12, 16, 20, 24, 28, 32, 36 mm) determine a steel arrangement that will satisfy the required steel percentages. Choose the one with the lowest relative cost which does not violate spacing requirements. (See Programmers' Reference Manual Appendix 5).

3.2.4 Selection of Load Cases for design

This has been adapted from the original module. The load cases to be tested by the module are determined by the following considerations:

(a) The number of load cases tested in the design of the column lift being considered are relevant to this column and not, as previously, including those applicable to the column above.

(b) The loading cases as presented by GENESYS consist of a permutation of possible load cases occurring at the head and foot of one or more columns in a batch.

(c) For one column a typical set of load cases is made up of 18 load cases at the head combined with 8 load cases at the foot giving a total of 18 x 8 = 144 load cases per column. This number is larger for a group of columns having variation of load between members.

3.2.4.1 Short Column ($l/r < 20$)

It is only for slender columns that it is necessary to consider combinations of head and foot moments acting simultaneously. Therefore the number of load cases for short columns may be reduced.

Details of how this is done may be found in the discussion of subroutine CENTRY in the Programmer's Reference Manual (Appendix 5).

3.2.4.2 Long Columns ($l/r > 20$)

It is not possible to effect much reduction in the number of load cases to be tested for slender columns.Among the 144 load cases sent by GENESYS, some will consist of mono-x at the top combined with mono-y at the foot and were it possible to delete those cases (which some authorities believe unnecessary) then the number of load cases tested could be cut to about 48.

3.2.5 Logic for avoidance of Bar Clashing at Column/Beam Intersections

(from original module)

Beams and columns may each be designed independently of each other. Column bars will be detailed to pass unhindered through intersections and any clashing avoidance necessary will be taken by the beam steel. Beam steel may be single or paired and the column steel may be bundled.

If one or more of the beam outer bars falls in line with, or inside the corner* column bars then these outer bars will be curtailed at the column face and splice bars will be inserted. Internal beam bars will pass through the spaces between the column bars and the number of spaces that are available will depend on the maximum number of internal column bars that can be accommodated within the column. This maximum number will be calculated in such a way as to ensure that:

* The position of the corner bars is set by the User - specified cover.

(a) The space "B" between the column bars will not be less than the nominal size of the largest permitted column bar (C_{max}) or the column aggregate size plus 5mm if this be the larger. In addition the space must be large enough to equal the zone size BZ_{max} corresponding to the largest permitted beam bar or beam bar pair. B = MAX(C_{max}, BZ_{max}, (Col Agg + 5))

(b) The space "M" between beam bars will not be less than the nominal size "BE" of the largest permitted beam bar "BM_{max}" or, if paired bars are used, to the size of the bar of equivalent area to the largest permitted bar pair (2xBM_{max}). In addition it will not be less than the beam aggregate plus 5mm or the zone size CZ_{max} corresponding to the largest permitted column bar.

M = MAX(BE_{max}, CZ_{max}, (Beam Agg + 5))

(c) Zone sizes BZ or CZ will be taken as 1.1x Nominal size for single bars and 2.2x Nominal size for bar pairs. The zone sizes are then rounded to the nearest whole number.

Equivalent Bar and Zone sizes for single and paired bars of Nominal sizes BM or CM are shown in Table 3.1.

Nom Bar Size	Eq Size "BE"		Zone size BZ or CZ	
	Single Bar	Bar Pair	Single Bar	Bar Pair
BM or CM	BM or CM	2xBM	1.1xBM or 1.1xCM	2.2xBM
12	12	17	13	26
16	16	23	18	35
20	20	28	22	44
24	24	34	26	53
28	28	40	31	62
32	32	45	35	70
36	36	51	40	79

Table 3.1 Equivalent Bar and Zone Sizes

The method adopted to avoid beam/column bar clashing is discussed in detail in Appendix 6.

3.2.6 Derivation of failure curve for unreinforced concrete

The equation governing the failure curve for an unreinforced concrete column may be derived as follows:

Assume a rectangular stress block shown in Figure 3.1

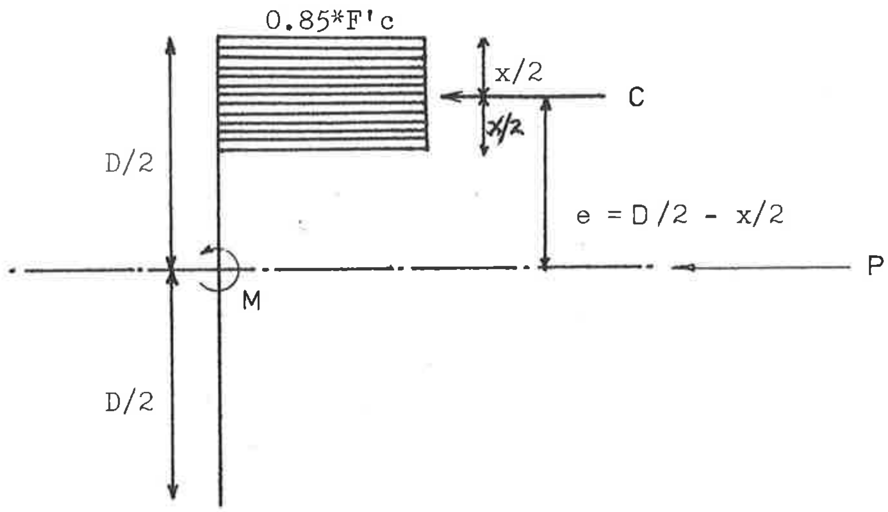


Figure 3.1 Situation at Failure

At equilibrium, P = C (Newtons) and M = C*e (Newton - millimetres)

$$P = C = 0.85 * F'c * b * X * \phi$$

which gives $X = \dfrac{P}{\phi * 0.85 * F'_c * b}$

also $M = C*e = \phi * 0.85 * F'_c * b * X * (\frac{D}{2} - \frac{X}{2})$

Substituting for X gives:

$$M = \frac{\phi * 0.85 * F'_c * b * P}{\phi * 0.85 * F'_c * b} * (\frac{D}{2} - \frac{P}{2 * \phi * 0.85 * F'_c * b})$$

$$= \frac{P*D}{2} - \frac{P^2}{1.7 * \phi * F'_c * b}$$

Dividing by bD^2 and substituting $M' = M/bD^2$ and $P' = P/bD$ and also setting $\phi = 0.7$ gives:

$$\underline{M' = P'/2 - P'^2/1.19\ F'c} \qquad .....3.1$$

PMIN is defined as the value of P' where the curve crosses the M'=0 axis and may be found by setting M'=0 in equation 3.1.

$$\underline{PMIN = 0.595\ F'_c\ (MPa)} \qquad .....3.2$$

The slope of the curve as it crosses the M'= 0 axis is also required for the analysis. This may be determined by calculating the differential of 3.1 at PMIN.

$$\frac{dM'}{dP'} = \frac{1}{2} - \frac{2P'}{1.19\ F'_c}$$

At $PMIN = 0.595\ F'_c$

$$\frac{dM'}{dP'} = \frac{1}{2} - 1 = \underline{-0.5} \qquad .....3.3$$

These relations plot correctly on the charts and are therefore satisfactory. See Design Charts on following pages.

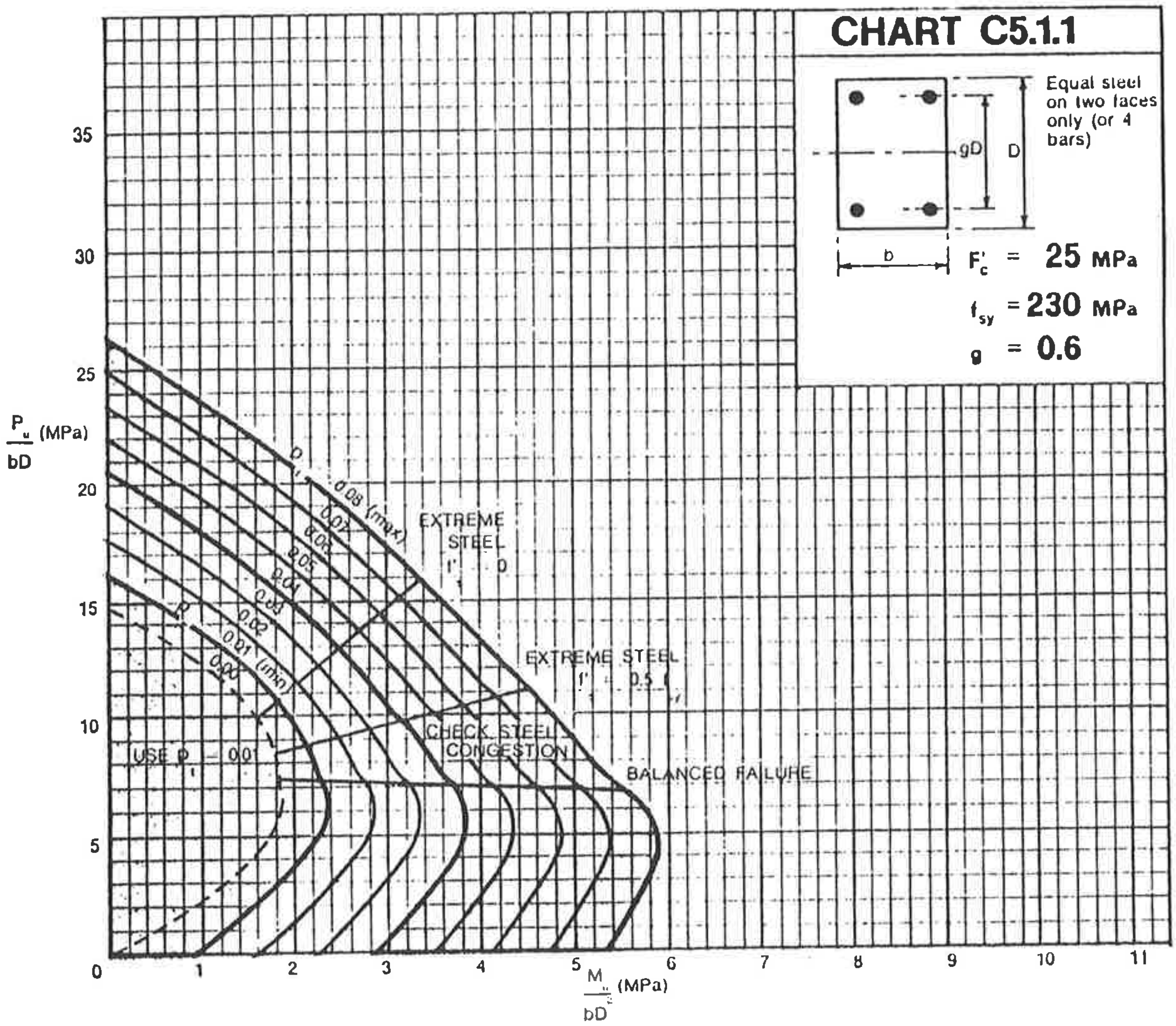


Chart 3.1 - Design Chart C5.1.1 (Ref. 15)

$M_u = M'$ $P_u = P'$

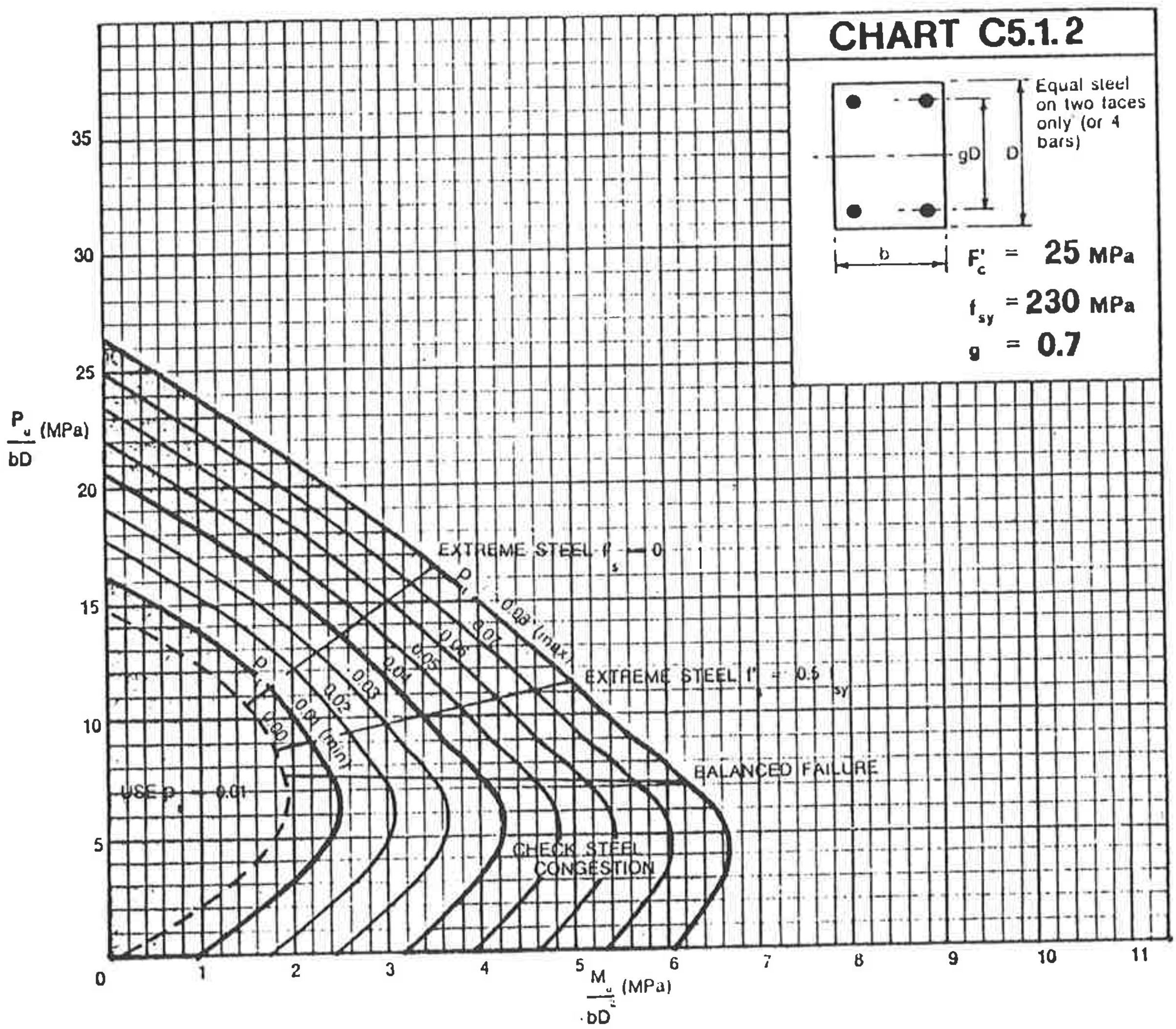


Chart 3.2 - Design Chart C5.1.2 (Ref. 15)

$M_u = M' \quad P_u = P'$

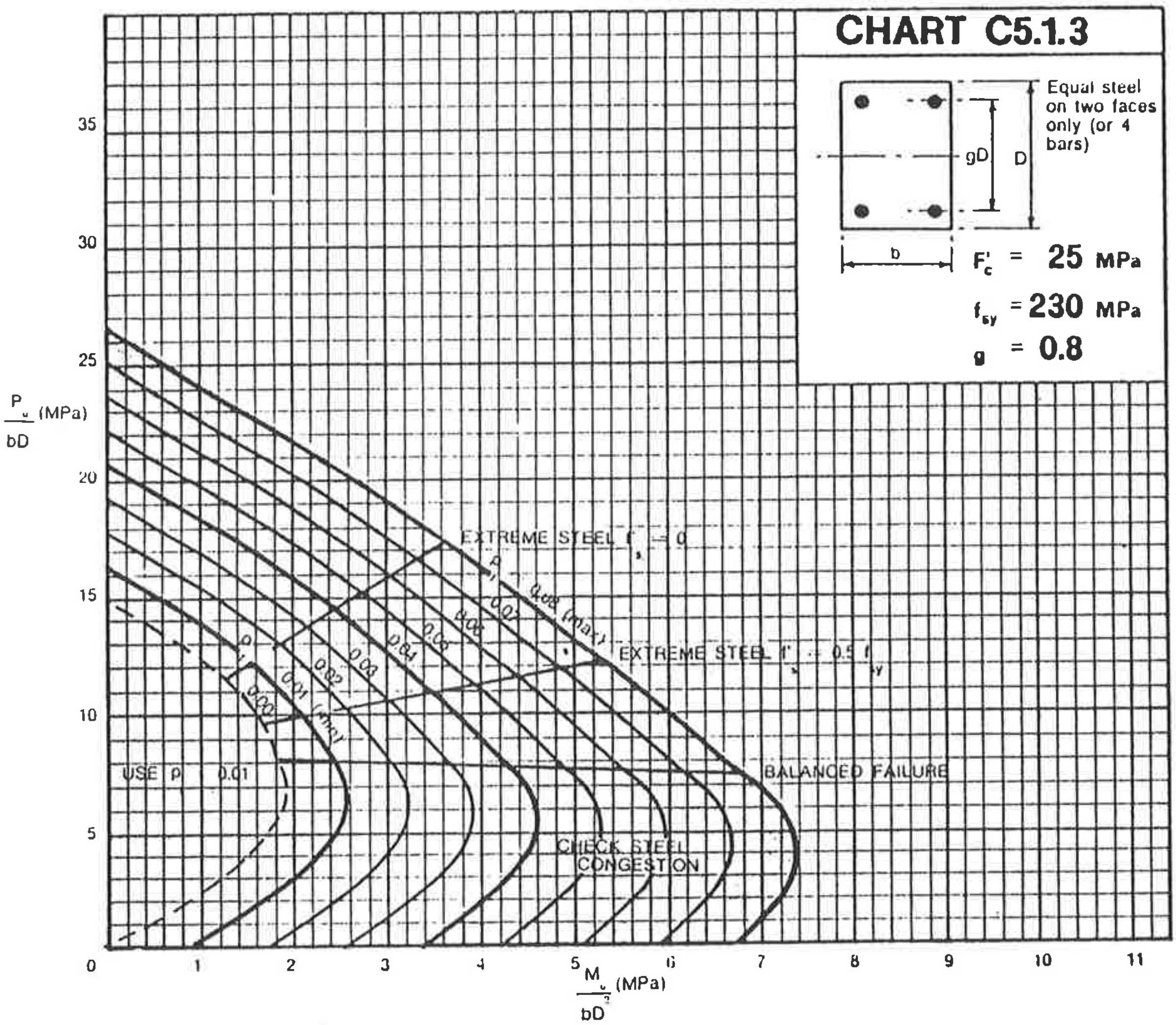


Chart 3.3 - Design Chart C5.1.3 (Ref. 15)

$M_u = M'$ $P_u = P'$

3.3 Organisation of the Module

3.3.1 Introduction

The module has been organised to take advantage of the facility offered by the GENESYS system which allows a program to be split into OVERLAYs, only one of which resides in central memory at any one time. The program efficiency may be increased by the effective use of this facility.

The module contains an extensive table of coefficient values and it was considered expedient to break the module into two OVERLAYs. The coefficients and related subroutines were therefore placed in one OVERLAY and the main design programme in the other.

The OVERLAY names and numbers adopted are the same as these used in the original module and the contents of each are as follows:

```
OVERLAY 52 "RC22":    Routine CENTRY
                         "    EFLTH
                         "    BCLASH
                         "    RTVP

OVERLAY 54 "RC24":    Routine DRIVER
                         "    SORT
                         "    COLLOAD
                         "    RED
                         "    SORTL
                         "    FINDP
                         "    CALCP
                         "    FUNCP
                         "    ARRANG
```

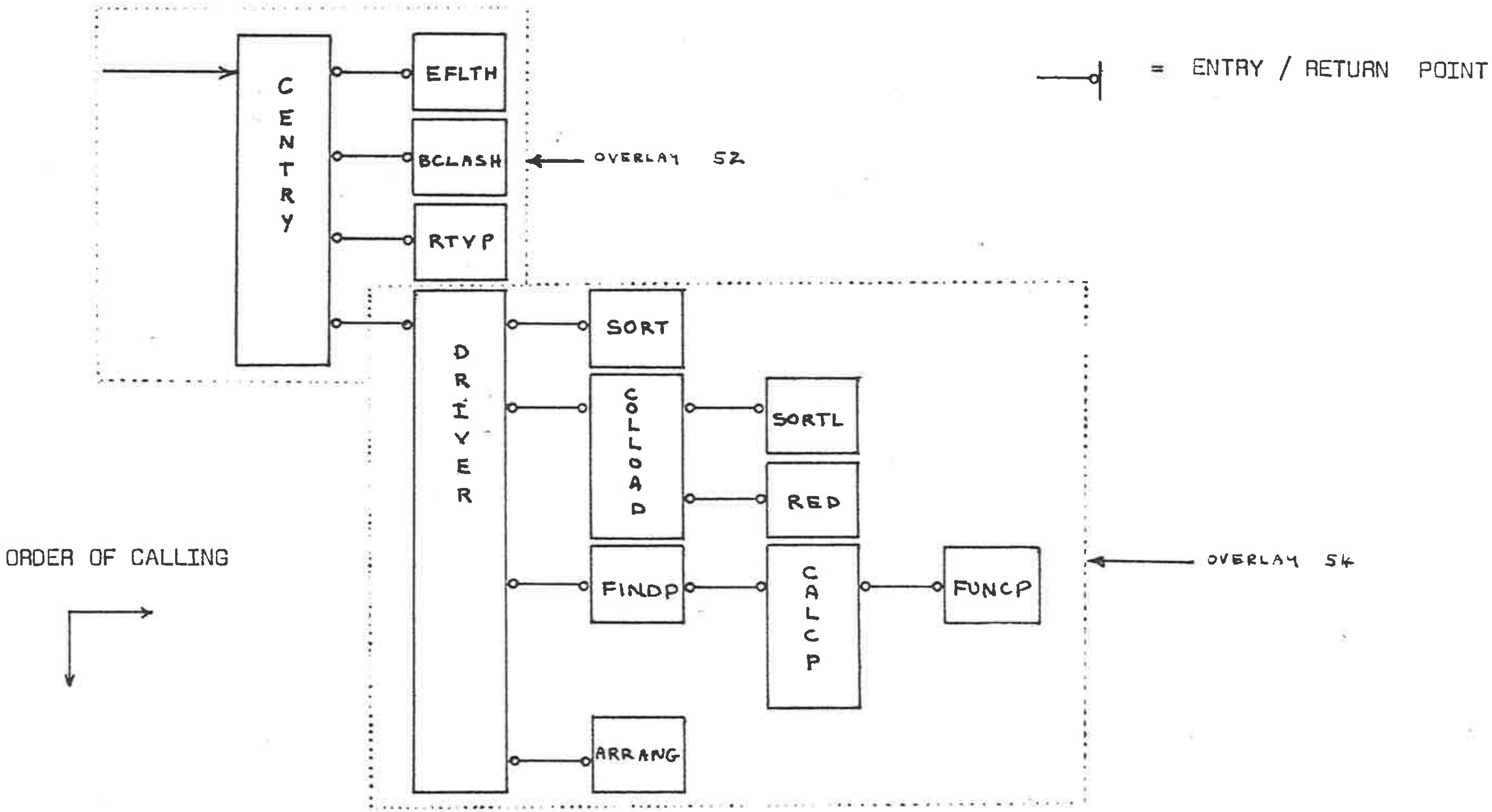


FIGURE 3.2 - Subroutine Calling Sequence of Replacement Module

The basic module organisation is shown in the flow diagram for the subroutine CENTRY which is the entry point for the module. Details of CENTRY may be found in the Programmer's Reference Manual in Appendix 5. The subroutine calling sequence is shown in Figure 3.2 and may be compared with that of the original module shown in Figure 2.1. The programming for OVERLAYs 52 and 54 is shown in Appendices 3 and 4.

3.3.2 Information supplied to module

The module is supplied with the following information:

- Column dimensions and lengths
- Properties of beams/slabs which are connected
- Material properties
- A loading array containing all of the loading combinations that can be applied to the column. Each loading consists of an axial load and head and foot moments if they exist.

From this information the module determines the cheapest steel arrangement (if any exist) capable of withstanding the worst loading combination.

3.3.3 Public Variable System

For a detailed explanation of the PUBLIC VARIABLE SYSTEM and the function of each variable, see Section 2.

3.3.4 Sign Conventions

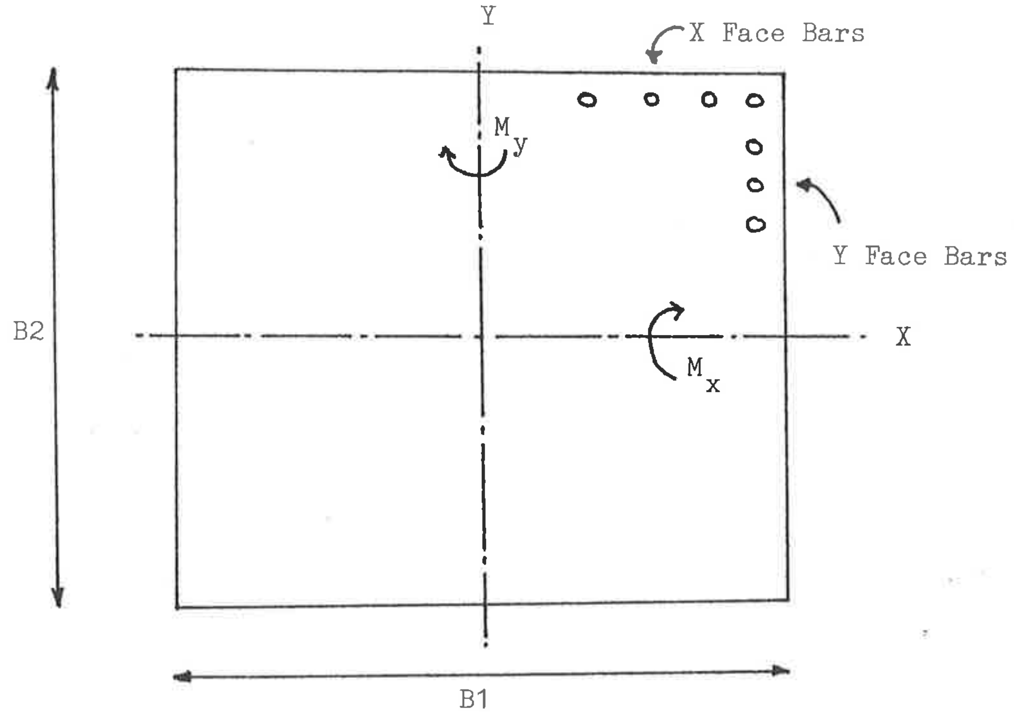


Figure 3.3 Sign Conventions

3.4 Subroutines Involved

3.4.1 Introduction

As stated in section 3.3.1, the new module consists of 13 subroutines split into 2 groups. This section explains the function of each subroutine. A more detailed explanation may be found in the Programmer's Reference Manual, Appendix 5.

3.4.2 Subroutine CENTRY

This routine is the entry and exit point for the design of rectangular columns. It controls the design sequence by calling the various utility subroutines in their correct order and contains the PUBLIC variables required to link with the other subroutines in the main program. The flow chart for this subroutine is shown in Appendix 5, Section 2.4.

3.4.3 Subroutine EFLTH

The effective length of the column group is calculated for both axes. The flow chart is shown in Appendix 5, Section 2.5.

3.4.4 Subroutine BCLASH

This routine establishes the permitted number of bars in each face of the column if beam bar clashing is to be avoided. The flow chart is shown in Appendix 5, Section 2.6.

3.4.5 Subroutine RTVP

This routine sets the value of the coefficients to be used to represent the charts. The flow chart is shown in Appendix 5, Section 2.7.

3.4.6 Subroutine DRIVER

This is the main driving routine controlling the design sequence. It is entered initially from CENTRY and, when complete, it initiates the detailing routine if requested. The flow chart is shown in Appendix 5, Section 2.8.

3.4.7 Subroutine SORT

This routine has two functions. The first is to eliminate the unnecessary load cases and the second is to sort the loadings in descending order of the applied axial load. The flow chart is shown in Appendix 5, Section 2.9.

3.4.8 Subroutine COLLOAD

This routine is used to compile the final loading set subject to:

(a) Initial eccentricities
(b) Reduction factors
(c) Removal if required steel % = 0
(d) Removal if same axial load but smaller moment
(d) Creation of biaxial loading cases

The flow chart is shown in Appendix 5, Section 2.10

3.4.9 Subroutine SORTL

This routine is called from COLLOAD to perform the functions (c) and (d) as detailed in section 3.4.8. The flow chart is shown in Appendix 5, Section 2.11.

3.4.10 Subroutine RED

This routine is called from COLLOAD to apply the reduction factors. The flow chart is shown in Appendix 5, Section 2.12.

3.4.11 Subroutine FINDP

This routine searches through the loading cases for the maximum steel % required. The flow chart is shown in Appendix 5, Section 2.13.

3.4.12 Subroutine CALCP

This routine is called from FINDP to establish the required steel % for a given axial force/moment combination. The flow chart is shown in Appendix 5, Section 2.14.

3.4.13 Function FUNCP

This function uses the chart equations to determine the failure moment given the axial force and the steel %. The flow chart is shown in Appendix 5, Section 2.15.

3.4.14 Subroutine ARRANG

This routine determines the most economical way of ensuring that the required steel % is supplied. The flow chart has been broken into 4 segments as shown in Appendix 5, Section 2.16.

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

SECTION 4 - IMPLEMENTATION AND TESTING OF THE NEW MODULE

4.1 Introduction

The implementation and testing of the new module for rectangular column design required six distinct stages:

(a) The implementation of the GENESYS system itself

(b) The implementation of the RC-STRUCTURE subsystem

(c) The implementation of the RC-COLUMN subsystem

(d) The implementation of the AS-COLUMN subsystem

(e) Testing

(f) Comparison between the results of AS-COLUMN and RC-COLUMN

All implementation,testing and debugging was done using a CDC CYBER 170/720 model computer running the NOS/BE operating system.

4.2 Implementation of the GENSYS System

Initially the CDC CYBER version of GENESYS running under the NOS operating system was implemented. The different operating system in use (NOS/BE) required some changes but eventually a version existed which could compile and load GENTRAN overlays. However, errors occurred during the execution of the subsystem RC-STRUCTURE which were traced to the inability of GENESYS to re-load an overlay properly.

The CYBER version of GENESYS controls the overlay execution by reading the required overlay into BLANK COMMON and then continuing execution from a specified address within the BLANK COMMON area. This address is fixed within the COMPASS code effecting the overlay loading and there is no allowance made for the differences in operating systems and compiler software that exist between any two computer installations. Thus there is no guarantee that a version working at one computer site will work at another without major modification. This is especially so if there are major operating system differences (NOS vs NOS/BE) because the assumptions made about the position within memory of certain control information will not be correct. It was then decided to try to implement the Perkin-Elmer version of GENESYS on the CYBER.

This decision was made because, unlike the CYBER version, the P.E. version is written entirely in FORTRAN and therefore easier to debug. Note that a NOS/BE version of GENESYS was not available at the time this work was being undertaken.

From the user's point of view, the major change caused by using the P.E. version is that overlay loading and unloading is no longer performed by GENESYS itself but by SEGLOADER, the CDC product for handling segmented programs. A number of other minor changes were required which were caused by the difference in computer word size (60 bits on the CYBER and 32 bits on the P.E.) and also by differences in file handling procedures. Once these differences had been resolved, a working version of GENESYS was available which could compile, load and execute any subsystem.

The advantages of the new GENESYS version are:

(a) file space during program development is reduced

(b) Each subsystem is a program in its own right

(c) Changes may be made to GENESYS itself without the necessity to recompile the overlays.

The following disadvantages exist, but they are operating system dependent and would vary in significance depending on the machine being used.

(a) Execution time is slightly longer because of the overhead associated with SEGLOADER

(b) It is necessary to reload the entire subsystem if a change is made to an overlay

(c) The SEGLOADER directives must be tailored to ensure efficient use of computer memory

(d) Automatic ATTACHing and CATALOGing of FATHER and SON files is not possible

4.3 Implementation of the RC-STRUCTURE Subsystem

Implementation of the RC-STRUCTURE subsystem was required so that a valid input interface could be created for subsequent testing of the column design subsystems. This interface contains information required to define the column properties and applied loadings.

The implementation of RC-STRUCTURE involved the compiling of its component overlays and the specification of a suitable set of SEGLOADER directives. The only changes made to the subsystem were in the LIMIT statements in overlay 10. These changes were necessary to account for the particular dynamic memory parameters adopted during the implementation of GENESYS.

4.4 Implementation of the RC-COLUMN Subsystem

Implementation of the RC-COLUMN subsystem was undertaken so that a base for comparing the performance of the new modules could be established. The implementation involved the same steps as for RC-STRUCTURE and required similar changes to the LIMIT statements in overlay 10.

4.5 Implementation of the AS-COLUMN Subsystem

The AS-COLUMN subsystem was generated by replacing the rectangular column design modules in RC-COLUMN with those modules described in Section 3 of this thesis. Several minor changes have been made to RC-COLUMN since the new modules were originally designed. This has meant that similar minor changes have had to be made to the new modules although the basic philosophy has remained the same. These changes may be summarised as follows:

(a) Overlay "RC22" has been broken into 5 separate overlays, one for each chart group.

(b) The temporary chart coefficients are now derived using FUNCTION calls and are not stored in dynamic memory.

There are now 6 overlays for rectangular column design in AS-COLUMN. They take the same overlay numbers as the 6 which they replace in RC-COLUMN but only 2 are ever used in a particular job and each is never used more than once per job. This partly explains the performance improvement detailed in Section 4.5.

4.6 Testing of AS COLUMN

Testing and evaluating the capabilities of AS-COLUMN required consideration of (a) the ability of the design algorithms to faithfully reproduce the values derived from the charts and (b) the ability of the arrangement algorithm to produce an acceptable steel arrangement which could be used in the detailing modules of the subsystem.

The equations representing the charts were tested for each 1 MPa increment along the Mu/bD^2 axis and for each 5 MPa increment along the Pu/bD axis. A total of 3534 Moment/Axial Load combinations were tested (an average of 88 combinations per chart) to ensure that the entire range of each chart was covered.

In all cases the derived steel percentage is a better value than could be interpolated from the lines on the charts. The equations were originally chosen on the basis of giving a maximum of 3% difference from the charts and it is considered that the derived equations used in the Subsystem produce a more reliable and repeatable result than possible by visually scanning the charts.

The next phase was to test the effect of interpolating between the charts - i.e. using non-standard g values. This was done through the interface to RC-STRUCTURE and involved the running of Benchmark Test 1. The loadings applied to the column comprised the loads passed from RC-STRUCTURE together with loads passed through the 'XTRAL' table in the job stream. These extra loads were necessary because all loadings passed from RC-STRUCTURE required less than 1% steel.

Table 4.1 shows the results of computations made with a d/D value of 0.77 and interpolating between charts C5.1.10 and C5.1.11. The results demonstrate the correctness of the interpolating algorithm.

The last phase of testing involved evaluation of the steel arrangement algorithm. This was achieved by running Benchmark Tests 2 to 7 which are illustrated in Tables 4.2 to 4.7. They show that:

(a) The algorithm is able to generate an acceptable solution. In all cases the derived steel percent is within + 5% of the target.

(b) The costing algorithm is efficient in determining the cheapest solution.

(c) The algorithm is able to reject solutions which do not meet spacing and bar clashing requirements. In the examples given, solutions are rejected as violating bar clashing requirements if there are more than 5 X-Face bars or more than 2 Y-Face bars.

Load	Pu/bD	Mu/bD^2	C5.1.10 Values	C5.1.11 Values	SUM *	AS-COLUMN Values
1	25	1.3	3.9	3.9	3.90	3.87
2	25	2.3	4.9	4.7	4.76	4.77
3	25	3.3	5.9	5.6	5.69	5.72
4	25	4.3	7.0	6.6	6.72	6.72
5	25	5.3	8.1	7.6	7.75	7.74
6	25	6.3	9	8.5	9	9
7	10	3.5	1.7	1.3	1.42	1.48
8	10	4.5	2.9	2.4	2.55	2.51
9	10	5.5	4.0	3.4	3.58	3.51
10	10	6.5	5.1	4.3	4.54	4.53
11	10	7.5	6.2	5.3	5.57	5.53
12	10	8.5	7.2	6.2	6.50	6.52
13	10	9.1	8.2	7.2	7.50	7.50
14	0	11	8.4	7.3	7.63	7.70
15	0	10	7.6	6.7	6.97	7.00
16	0	9	6.8	6.0	6.24	6.27
17	0	8	6.0	5.2	5.44	5.54
18	0	7	5.2	4.6	4.78	4.81
19	0	6	4.3	4.0	4.09	4.13
20	0	5	3.7	3.2	3.35	3.39
21	0	4	2.9	2.6	2.69	2.67
22	0	3	2.1	1.9	1.96	2.00
23	0	2	1.3	1.2	1.23	1.28
24	0	1	0.6	0.7	0.67	0.62

* SUM = .3 x Chart C5.1.10 + .7 x Chart C5.1.11

See Design Charts in Ref 15, Chart 5.1.10 for g = 0.7
5.1.11 for g = 0.8

TABLE 4.1 - Testing Interpolation between charts

AS-COLUMN BENCHMARK TEST 2   1442   01/03/83   PAGE0014

****
DESIGN SOLUTIONS
****

** TARGET STEEL PERCENTAGES

MONO-X = 1.00 - DERIVED FROM LOADING NO. 0
MONO-Y = 0.00 - DERIVED FROM LOADING NO. 0
BIAXIAL = 0.00 - DERIVED FROM LOADING NO. 0

SOLN NO.	BAR SIZE	NO. OF BARS		STEEL PERCENTAGE			COST
		X	Y	X	Y	TOTAL	
1	32	2	2	1.34	1.34	1.34	3.51
2	28	2	2	1.03	1.03	1.03	2.79
3	24	3	2	1.13	0.75	1.13	3.24
4	20	4	2	1.05	0.52	1.05	3.24
5	16	6	2	1.01	0.34	1.01	R
6	12	11	2	1.04	0.19	1.04	R

** REJECTED SOLUTIONS ARE MARKED -R- IN THE COST COLUMN

SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACING
OR BAR CLASHING REQUIREMENTS

SOLUTION NUMBER 2 HAS BEEN CHOSEN FOR DETAILING

TABLE 4.2 - Benchmark Test 2

AS-COLUMN BENCHMARK TEST 3 1500 01/03/83 PAGE0015

****
DESIGN SOLUTIONS
****

** TARGET STEEL PERCENTAGES

MONO-X = 1.84 - DERIVED FROM LOADING NO. 1
MONO-Y = 0.00 - DERIVED FROM LOADING NO. 0
BIAXIAL = 0.00 - DERIVED FROM LOADING NO. 0

SOLN NO.	BAR SIZE	NO. OF BARS X	NO. OF BARS Y	STEEL PERCENTAGE X	STEEL PERCENTAGE Y	STEEL PERCENTAGE TOTAL	COST
1	32	3	2	2.01	1.34	2.01	5.27
2	28	4	2	2.05	1.03	2.05	5.58
3	24	5	2	1.88	0.75	1.88	5.40
4	20	7	2	1.83	0.52	1.83	R
5	16	11	2	1.84	0.34	1.84	R
6	12	19	2	1.79	0.19	1.79	R

** REJECTED SOLUTIONS ARE MARKED -R- IN THE COST COLUMN

SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACING
OR BAR CLASHING REQUIREMENTS

SOLUTION NUMBER 1 HAS BEEN CHOSEN FOR DETAILING

TABLE 4.3 - Benchmark Test 3

AS-COLUMN BENCHMARK TEST 4   1503   01/03/83   PAGE0015

****
DESIGN SOLUTIONS
****

** TARGET STEEL PERCENTAGES

MONO-X = 2.25 - DERIVED FROM LOADING NO. 1
MONO-Y = 0.00 - DERIVED FROM LOADING NO. 0
BIAXIAL = 0.00 - DERIVED FROM LOADING NO. 0

SOLN NO.	BAR SIZE	NO. OF BARS		STEEL PERCENTAGE			COST
		X	Y	X	Y	TOTAL	
1	32	4	2	2.68	1.34	2.68	7.02
2	28	5	2	2.57	1.03	2.57	R
3	24	6	2	2.26	0.75	2.26	R
4	20	9	2	2.36	0.52	2.36	R
5	16	13	2	2.18	0.34	2.18	R
6	12	23	2	2.17	0.19	2.17	R

** REJECTED SOLUTIONS ARE MARKED -R- IN THE COST COLUMN

SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACING
OR BAR CLASHING REQUIREMENTS

SOLUTION NUMBER 1 HAS BEEN CHOSEN FOR DETAILING

TABLE 4.4 - Benchmark Test 4

AS-COLUMN BENCHMARK TEST 5 1510 01/03/83 PAGE0015

****
DESIGN SOLUTIONS
****

** TARGET STEEL PERCENTAGES

MONO-X = 2.67 - DERIVED FROM LOADING NO. 1
MONO-Y = 0.00 - DERIVED FROM LOADING NO. 0
BIAXIAL = 0.00 - DERIVED FROM LOADING NO. 0

SOLN NO.	BAR SIZE	NO. OF BARS X	NO. OF BARS Y	STEEL PERCENTAGE X	STEEL PERCENTAGE Y	STEEL PERCENTAGE TOTAL	COST
1	32	4	2	2.68	1.34	2.68	7.02
2	28	5	2	2.57	1.03	2.57	R
3	24	7	2	2.64	0.75	2.64	R
4	20	10	2	2.62	0.52	2.62	R
5	16	16	2	2.68	0.34	2.68	R
6	12	27	2	2.54	0.19	2.54	R

** REJECTED SOLUTIONS ARE MARKED -R- IN THE COST COLUMN

SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACING
OR BAR CLASHING REQUIREMENTS

SOLUTION NUMBER 1 HAS BEEN CHOSEN FOR DETAILING

TABLE 4.5 - Benchmark Test 5

AS-COLUMN BENCHMARK TEST 6   1506 01/03/83 PAGE0015

****
DESIGN SOLUTIONS
****

** TARGET STEEL PERCENTAGES

MONO-X = 3.09 - DERIVED FROM LOADING NO. 1
MONO-Y = 0.00 - DERIVED FROM LOADING NO. 0
BIAXIAL = 0.00 - DERIVED FROM LOADING NO. 0

SOLN NO.	BAR SIZE	NO. OF BARS X	NO. OF BARS Y	STEEL PERCENTAGE X	STEEL PERCENTAGE Y	STEEL PERCENTAGE TOTAL	COST
1	32	5	2	3.35	1.34	3.35	R
2	28	6	2	3.08	1.03	3.08	R
3	24	8	2	3.02	0.75	3.02	R
4	20	12	2	3.14	0.52	3.14	R
5	16	18	2	3.02	0.34	3.02	R
6	12	32	2	3.02	0.19	3.02	R

** REJECTED SOLUTIONS ARE MARKED -R- IN THE COST COLUMN

SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACING
OR BAR CLASHING REQUIREMENTS

** NO SATISFACTORY SOLUTIONS

TABLE 4.6 - Benchmark Test 6

AS-COLUMN BENCHMARK TEST 7 1511 01/03/83 PAGE0015

****
DESIGN SOLUTIONS
****

** TARGET STEEL PERCENTAGES

MONO-X = 3.29 - DERIVED FROM LOADING NO. 1
MONO-Y = 0.00 - DERIVED FROM LOADING NO. 0
BIAXIAL = 0.00 - DERIVED FROM LOADING NO. 0

SOLN NO.	BAR SIZE	NO. OF BARS X	NO. OF BARS Y	STEEL PERCENTAGE X	STEEL PERCENTAGE Y	STEEL PERCENTAGE TOTAL	COST
1	32	5	2	3.35	1.34	3.35	R
2	28	7	2	3.59	1.03	3.59	R
3	24	9	2	3.39	0.75	3.39	R
4	20	12	2	3.14	0.52	3.14	R
5	16	19	2	3.18	0.34	3.18	R
6	12	34	2	3.20	0.19	3.20	R

** REJECTED SOLUTIONS ARE MARKED -R- IN THE COST COLUMN

SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACING
OR BAR CLASHING REQUIREMENTS

** NO SATISFACTORY SOLUTIONS

TABLE 4.7 - Benchmark Test 7

RC-COLUMN BENCHMARK TEST 2 1023 09/03/83 PAGE0012

SECTION PROPERTIES

SHAPE	AC	IXX	IYY	LEX	LEY
	M**2	MM**4	MM**4	MM	MM
R	0.2400	3200	7200	3355	4266

VALUES OF INERTIAS ARE DIVIDED BY 10**6

MAIN STEEL TYPE *HY* LINK STEEL TYPE *R*

****
DESIGN SOLUTIONS
****

SOLN NO.	CRNR BAR SIZE MM	NO. OF BARS PER FNSIDE	INTRNL FACE RLSIDE	INTRNL BAR SIZE MM	REQD. STEEL PRCNT	PROVD. STEEL PRCNT	LINK SIZE MM	LINK PITCH MM	COST
1-1	32	0	0	0	1.00	1.34	8	384	11.07
2-1	25	0	0	0	0.80	0.82	8	300	7.16
3-1	20	2	0	16	0.80	0.86	6	240	9.53
4-1	16	3	0	16	0.80	0.84	6	192	7.91

SOLUTION NUMBER 2-1 HAS BEEN CHOSEN FOR DETAILING
****

*WARNING C3542 MAXIMUM BAR ARRANGEMENT CONSIDERED IN DESIGN

*WARNING C3548 PREFERRED BAR ARRANGEMENT REQUESTED
IS NOT A POSSIBLE SOLUTION

CALCULATION DETAILS FOR DESIGN SOLUTION

SOLNO	NUZ	N	MI	MUX	MUY
	KN	KN	KN*M	KN*M	KN*M
2	3844	2400	200	195	305

****
E N D O F C O L U M N A N A L Y S I S
FOR GROUP *TYPED* BATCH NO. 1
****

TABLE 4.8 - RC-COLUMN Benchmark Test 2

4.7 Comparison between the results of AS-COLUMN and RC-COLUMN

Benchmark Test number two was run using both AS-COLUMN (Table 4.2) and RC-COLUMN (Table 4.8). The results were used to compare both the output and relative performance of the two subsystems under identical design conditions.

4.7.1 Output

The output for both Subsystems is identical (see Appendix 7) with the exception of the page concerned with the Design Solutions. Differences on this page may be summarised as follows:

(a) The format was changed to suit the results generated by AS-COLUMN.

(b) Although the cost algorithm was altered to suit Australian conditions it still chose solution 2 as the best.

(c) AS-COLUMN chose a minimum steel percent of 1.0 whereas RC-COLUMN chose a minimum steel percent of 0.8.

(d) Both subsystems chose an arrangement of 4 bars with a diameter of either 25mm or 28mm.

The major difference between the arrangement algorithms of both subsystems is that RC-COLUMN allows mixed bar sizes whereas AS-COLUMN allows only a single bar size per solution. However, the results indicate that both produce similar solutions.

4.7.2 Performance

The performance of both subsystems is summarised in Table 4.9. The interface time is the time taken if no design work is done. The results are for Benchmark Test 2 where 1 column is subjected to 17 loading combinations.

SUBSYSTEM	TOTAL TIME	FIELD LENGTH	DESIGN TIME
RC-COLUMN	23.028 Sec	123600 Octal	3.659
AS-COLUMN	21.101	123500 Octal	1.732

Table 4.9 - Relative Performance of AS-COLUMN and RC-COLUMN

These results indicate that the new module is 50% faster and uses slightly less memory. This difference would be significant in large jobs.

SECTION 5 - CONCLUSIONS

This project had the following primary objectives:

(a) To examine and compare the slenderness provisions of various Codes of Practice,

(b) To develop a computer program for the design of Concrete Columns according to AS1480-1974,

(c) To modify the rectangular column design module in the GENESYS RC-BUILDING suite so that it would conform with the provisions of AS1480-1974, and

(d) To improve the efficiency of the GENESYS RC-COLUMN subsystem

5.1 Comparison of Slenderness Provisions

A comparison between the slenderness provisions of a number of Codes of Practice has been presented in Section 1. The results of this comparison may be summarised as follows:

(a) The advantages of the AS1480 method are firstly that it is simple to apply and secondly that the concept of additional eccentricity is more rational than the alternative concept of minimum eccentricity.

The disadvantages of the AS1480 method are:

(i) it is more conservative than other Codes for all values of slenderness Ratio >20

(ii) it is less rational than either the additional moment, or moment magnifier method,

(iii) it gives unrealistic results at the limits of the stated range of its application, and

(iv) it has a number of minor shortcomings which have been itemised in Section 1.9.

It was therefore proposed that in future revisions of AS1480, consideration be given to the following:

(i) Reducing the load factors,

(ii) Adopting the additional moment method as used in CP and DIN, or the ACI moment magnifier method, and

(iii) Addressing the shortcomings listed in Section 1.9.

5.2 Column Design Program

Algorithms for the Computer Aided Design of R.C. Columns have been presented in Section 3. These algorithms follow the steps given in the A.R.C. Design Handbook (Ref. 15) and will produce the steel percentage required for each loading case. A steel arrangement algorithm is also presented which produces a number of alternative bar arrangements which will satisfy the worst loading combinations. This arrangement algorithm will select the cheapest solution after having rejected those bar arrangements which violate spacing or beam bar clashing requirements.

As a result of this approach,the program has the following attributes:

(a) It executes very quickly,
(b) The amount of coding is minimal and well structured, and
(c) The designer is able to follow through each step of the design process and be satisfied that the results are correct.

The program has been thoroughly tested over a wide range of input conditions and is currently available to any of the S.A. Government Departments which have access to the facilities at the Government Computing Centre. I have personally used the program in a real design environment and am satisfied that it produces reliable results.

5.3 Modifications to RC-BUILDING

The RC-COLUMN subsystem of the GENESYS RC-BUILDING suite was modified so that it conformed with the provisions of AS1480-1974. These modifications involved converting the program described in 5.2 into the language GENTRAN and ensuring that the correct information was passed into and out of the module. The only major change required to the original program was to implement the virtual array processing technique available through GENTRAN to maximise program efficiency.

The operation of the new subsystem is demonstrated in the Benchmark tests described in Section 4. These show that the program operates successfully over a range of input values and, not unexpectedly, gives results which are similar to those of the original module.

5.4 Improving Subsystem Efficiency

During initial evaluation, the RC-COLUMN subsystem appeared to be using an unusually large amount of computing time. An examination of the program code traced the cause to a large number of iterative-type calculations in the design phase. For each loading case the column steel was incremented until the capacity exceeded the loading. However each steel augmentation required the recalculation of both the Neutral Axis depth and a new steel arrangement. The basic problem was that the module could not discern the critical loading and therefore had to test them all.

The new module has been designed to avoid such iterative calculations and is able to produce a small range of steel solutions which will cater for the critical load.

In quantifiable terms this has led to an increase in design speed of the order of 50% and to a reduction in the number of active lines of code from 4700 to 1200.

REFERENCES

1. ACI Committee 318, Building Code Requirements for Reinforced Concrete, ACI 318-71, ACI Detroit, 1971.

2. ACI Committee 318, Building Code Requirements for Reinforced Concrete, ACI 318-63, ACI Detroit, 1963.

3. British Standards Institution, The Structural Use of Concrete CP110, Part 1, 1972.

4. Canadian Standards Association, Code for the Design of Plain or Reinforced Concrete Structures, CSA Standard A 23.3, 1970.

5. Comite European du Beton, International Recommendations for the Design and Construction of Concrete Structures, FIP Sixth Congress, Prague, June 1970.

6. Constantinescu, D.R., Illston, J.M., Computer Experiments on slender beam - columns of reinforced concrete, Magazine of Concrete Research Vol. 30, No. 105: Dec. 1978.

7. Corderoy, H.J.H., Warner, R.F., Slender Column Design in AS 1480/1974, The Institution of Engineers Australia, Civil Engineering Transactions, 1977.

8. Deutscher Industrie Normen, DIN 1045, Beton und Stahlbetonbau, Bemessungund Ausfuhrung, Betonkalender, 1971.

9. MacGregor, J.G., Breen, J.E., Pfrany, E.O., Design of Slender Concrete Columns, Journal ACI, Proc. Vol. 67, Jan 1970.

10. Menn, C., Symposium on Design and Safety of Reinforced Concrete Compression Members. International Association for Bridge and Structural Engineering - Final Report, Quebec 1974.

11. Standards Association of Australia, The Use of Reinforced Concrete in Structures, AS 1480/1974.

12. Warner, R.F., Physical - Mathematical Models and Theoretical Considerations, Theme I Introductory Report: 1974 Quebec Symposium on Design and Safety of Reinforced Concrete Compression Members, IABSE Zurich.

13. Wilby, C.B., Pacholi, V.R., Design of Very Slender Reinforced Concrete columns, Civil Engineering, Nov. 1978.

14. Timoshenko, S., Young, D.H., Elements of Strength of materials, 4th Education, Van Nostrands Company Ltd., 1964.

15. Cement and Concrete Association, Australian Reinforced Concrete Design Handbook, Harbour Press, 1976.

16. Furlong, R.W., Concrete Columns under Biaxially Eccentric Thrust, ACI Journal, October, 1979.

17. Warner, R.F., Rangan, B.V., Hall, A.S., Reinforced Concrete, Pitman Australia, 1976.

18. Behan, J.E., O'Connor, C., Creep Buckling of Reinforced Concrete Columns, Research Report No. CE3, Department of Civil Engineering, University of Queensland, April, 1979.

19. Brettle, H.J., Warner, R.F., Ultimate Strength Design of Rectangular Reinforced Concrete Sections in Compression and Biaxial Bending, Civil Engineering Transactions, April 1968.

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 1

PROGRAM FITCURV

I N D E X

A 1.1	Introduction	...122
A 1.2	Choice of Equation	...122
A 1.3	Program FITCURV	...123
A1.4	Application of FITCURV to DESIGN charts	...124
A 1.5	Single Equation for each chart	...126
A 1.6	Conclusions	...128

A 1.1 INTRODUCTION

Inspection of the various column design procedures available reveals that in all cases the design procedes in an iterative manner - the steel required depends on the neutral axis position which depends on the steel present etc... It was thought that Design charts usually provide an effective way around this problem and so a method was sought whereby a set of Design charts could be incorporated into a computer program. The set of charts chosen appears in the Australian Reinforced Concrete Design Handbook (Ref. 15).

The most convenient way to use these charts is to find an equation relating all of the points on the chart. It was hoped that a general equation could be found for each chart but, as will be shown later, this was not practical. The method adopted was to find the equation of each line of each chart and store it for use by the column design program which would decide which equation to use.

A 1.2 CHOICE OF EQUATION

While being basically similar there is a wide variation within and between the charts. An equation was therefore sought which would contain sufficient degrees of freedom to cater for this variation yet be reasonably simple to use. It was also required that the same equation be used for all of the charts so that the column design program would be straight forward. Given these constraints it was decided to use an (n-1) order polynomial of the form:

$$M = C_1 + C_2P + C_3P^2 + C_4P^3 + \ldots + C_nP^{n-1}$$

where n would be determined considering the accuracy required and the degree of variation between the charts. Thus each line on each chart is represented by the same polynomial but with different coefficients.

It was suspected that an exact equation could not be found for each curve and that some sort of curve fitting procedure would be needed. This was not expected to be significant because the charts cannot be read to an accuracy greater than .1MPa. In this respect the curve fitting program may be considered superior to a human user because it "smoothes out" irregularities caused by misreading and in many cases when comparing the resubstituted values produced by the equations, they are closer to the curve than the points originally read in. Given that no exact equation exists, the procedure is then to convert the curves to a series of coordinate pairs and find the equation which produces the best fit.

A 1.3 PROGRAM FITCURV

This program is basically a regression analysis using the method of least squares. Its purpose is to accept a series of coordinate pairs, perform a regression analysis and produce the group of equations of order 2 to 10 which gives the best approximation. The coefficients in the output are given to 14 decimal places because in the higher orders the X terms become very large.

In addition the program also produces tables showing the following:

(a) The y values when x is resubstituted, thus indicating what the equations will produce.

(b) The percentage error between the input and the produced value.

(c) The difference between the input and the produced value.

These tables were used initially to test the validity of the program (see Appendix 2-B) and later to detect errors in the input data and to resolve which order equation to adopt for the column design programs. However when considering these tables, it must be remembered that the accuracy of the input data is to the nearest .1MPa. The computer works to much greater precision than this, smoothing out the data. Thus the derived equations may be a better approximation to the curves than inferred by the tables.

The program listing for FITCURV is shown in Appendix 2-A.

A 1.4 APPLICATION OF FITCURV TO DESIGN CHARTS

Having determined that FITCURV gives reliable results, each chart line was converted to a set of coordinates and processed through the program. The output for chart C5.1.1 p=.01 is shown in Appendix 2-B.

At this stage the outputs were examined to determine the correctness of the input data. Any typing/reading error would be corrected and the data re-processed before accepting the output.

Unusual results could also occur which were not caused by input errors. They were found to be caused by reading the input values to .1MPa and were prevalent when dealing with small P'/bD values - which is to be expected when for P'/bD = 1 MPa, the percentage error is 10%. Errors such as these were discounted if the resubstituted value was close to the chart value. In all cases where this was not so the reason was found to be an input error.

Once every chart had been processed, the outputs were examined to determine which order equation to adopt for the column design program. The criteria used to do this were:

(a) Limit the percentage error to a maximum of + 3% (or + 5% in cases where the error was due to reading to .1 MPa and the resubstituted value was a better approximation than the input value). Also high values caused by the smallness of Y were ignored.

(b) Limit the waviness of the output curve – the higher orders may be expected to be more wavy between points.

(c) Ensure the repeatability of the resubstituted value using a calculator. This was especially important in the higher orders where the accuracy and the number of significant figures of the coefficient were expected to become more important.

The general conclusions from this examination are:

(a) The 6th order is just as good as the 10th in reproducing the input data, but is significantly better than the 5th and lower orders.

(b) The 6th order equation is generally less wavy than the higher orders.

(c) The 10th order equation as printed in the output does not give the correct values when checked using a calculator.

This is because 14 digit accuracy is required for the 10th order equation but most hand-held calculators provide, at most, 12 digit accuracy.

(d) The percentage error for the 6th order equation was never more than 3%.

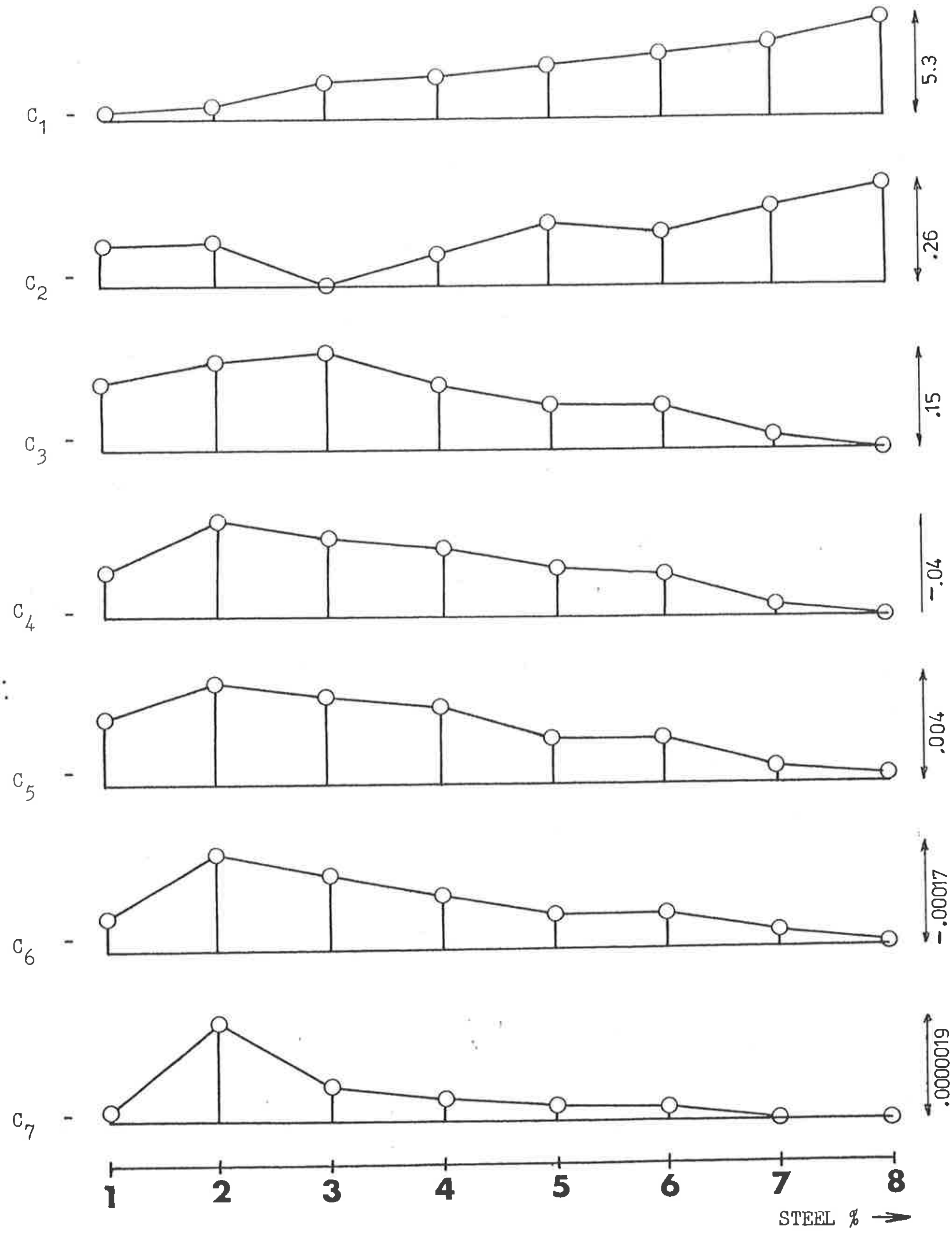
For these reasons it was decided that the 6th order equation would be adopted for the column design programs. The mean of all of the 6th order difference values for all of the charts was calculated to be = .03 MPa which indicates that reading to the nearest .1 MPa is justified.

A 1.5 SINGLE EQUATION FOR EACH CHART

Initially it was hoped that a single equation for each chart could be generated by finding a relationship between the coefficients and the steel percentage. This idea was abandoned for the following reasons:

(a) The 6th order coefficients were plotted as a function of p for chart C5.1.1. This is shown on Graph A1.1 and although there appears to be a general trend, it is not specific enough to allow accurate curve fitting.

(b) If it is assumed that a 10th order equation would be required to reproduce accurately the coefficients, then more coefficients would need to be stored than for the 8 equations separately.



$$Y = C_1 + C_2X + C_3X^2 + C_4X^3 + C_5X^4 + C_6X^5 + C_7X^6$$

GRAPH A1.1 Relative effect of Steel % on Coefficient Values (C5.1.1)

A 1.6 CONCLUSIONS

The program FITCURV is a reliable program with general application although the 10th order equation produced may be unrepeatable. The single factor which determines the unrepeatability is the accuracy of the computing machine. For example, if the coefficient C_{10} were found to be .00000000000135 then unless the computer/calculator could accept 14 significant digits, the equation would become inaccurate when the term $C_{10} * X^{10}$ became significant. If the tolerance was specified to be .01, then the equation would become inaccurate when:

$$.00000000000135 * x^{10} = 0.01$$

$$\text{or } x = 9.7$$

Thus when X exceeds 9.7, a handheld calculator would not give the same results as the program indicates. This could be overcome using an E-formatted output* for the coefficients but this was not considered necessary when, for the purposes of this work, the 6th order is adequate.

The coefficients adopted are shown in Appendix 3 "OVERLAY 52" in subroutine RTVP.

* F - Format is easier to check.

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 2 - A

PROGRAM LISTING FOR FITCURV

```
      PROGRAM FITCURV(TAPE8,INPUT=101B/80,OUTPUT=101B/136,TAPE1)
C     THIS PROGRAM ACCEPTS A SERIES OF COORDINATES AND WILL
C     FIT 9 CURVES OF ORDER 2 - 10
      DIMENSION X1(11,100),X2(11,100),A(21),Q(11,12),E(12),COEFF(9,1
C     INPUT TITLE
1     PRINT*,"TITLE"
      READ2,ITITL1
2     FORMAT(A8)
C     INPUT NUMBER OF POINTS
      PRINT*,"NUMBER OF POINTS"
      READ*,N
C     INPUT COORDINATES
      DO 10 I=1,N
      READ(8,*)X1(1,I),X1(2,I)
      X2(1,I)=X1(1,I)
      X2(2,I)=X1(2,I)
10    CONTINUE
      IF(X1(1,1).EQ.0.)X1(1,1)=0.000001
      WRITE(1,5)ITITL1
5     FORMAT(1H1,///1X,///20X,A20,///)
      IERR=0
      DO 170 M=2,10
      M1=2*M+1
      DO 15 I=2,M1
      A(I)=0.
15    CONTINUE
      M2=M+2
      DO 16 I=1,M2
      E(I)=0.
16    CONTINUE
      A(1)=N
      M3=M+1
      DO 40 I=1,N
      DO 20 J=2,M1
      A(J)=A(J)+X1(1,I)**(J-1)
20    CONTINUE
      DO 30 J=1,M3
      E(J)=E(J)+X1(2,I)*X1(1,I)**(J-1)
      Q(J,M+2)=E(J)
30    CONTINUE
      E(M+2)=E(M+2)+X1(2,I)**2
40    CONTINUE
      DO 60 I=1,M3
      DO 50 J=1,M3
      Q(I,J)=A(I+J-1)
50    CONTINUE
60    CONTINUE
      DO 135 K=1,M3
      DO 70 L=K,M3
      IF(Q(L,K).NE.0.)GOTO90
70    CONTINUE
      WRITE(1,80)M
80    FORMAT(1X,/28HNO UNIQUE SOLUTION FOR ORDER,I2/)
      IERR=IERR+1
      GO TO 170
90    DO 100 J1=1,M2
      B=Q(K,J1)
```

```
      Q(K,J1)=Q(L,J1)
      Q(L,J1)=B
100   CONTINUE
      C=1/Q(K,K)
      DO 110 J1=1,M2
      Q(K,J1)=C*Q(K,J1)
110   CONTINUE
      DO 130 L=1,M3
      IF(L.EQ.K)GO TO 130
      C=-Q(L,K)
      DO 120 J1=1,M2
      Q(L,J1)=Q(L,J1)+C*Q(K,J1)
120   CONTINUE
130   CONTINUE
135   CONTINUE
      DO 140 I=1,M3
      COEFF(M-1,I)=Q(I,M+2)
140   CONTINUE
      DO 160 I=1,N
      S=Q(1,M+2)
      DO 150 J=1,M
      S=S+Q(J+1,M+2)*X1(1,I)**J
150   CONTINUE
      X1(M+1,I)=S
160   CONTINUE
170   CONTINUE
      IF(IERR.EQ.5)GO TO 270
      IF(X1(1,1).EQ.0.000001) X1(1,1)=0.
      WRITE(1,180)
180   FORMAT(1X,49HEQUATIONS GIVING CLOSEST APPROXIMATION BY METHOD
     C 17H OF LEAST SQUARES//)
      WRITE(1,190)(COEFF(1,I),I=1,3)
190   FORMAT(1X,9H2ND ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2)/)
      WRITE(1,200)(COEFF(2,I),I=1,4)
200   FORMAT(1X,9H3RD ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2),3H + ,F18.14,5H*X(3)/)
      WRITE(1,210)(COEFF(3,I),I=1,5)
210   FORMAT(1X,9H4TH ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2),3H + ,F18.14,5H*X(3),3H + ,F18.14,5H*X(4)/)
      WRITE(1,220)(COEFF(4,I),I=1,6)
220   FORMAT(1X,9H5TH ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2),3H + ,F18.14,5H*X(3),3H + ,F18.14,5H*X(4),3H +
     C /58X,F18.14,5H*X(5)/)
      WRITE(1,230)(COEFF(5,I),I=1,7)
230   FORMAT(1X,9H6TH ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2),3H + ,F18.14,5H*X(3),3H + ,F18.14,5H*X(4),3H +
     C /58X,F18.14,5H*X(5),3H + ,F18.14,5H*X(6)/)
      WRITE(1,280)(COEFF(6,I),I=1,8)
280   FORMAT(1X,9H7TH ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2),3H + ,F18.14,5H*X(3),3H + ,F18.14,5H*X(4),3H +
     C /58X,F18.14,5H*X(5),3H + ,F18.14,5H*X(6),3H +
     C F18.14,5H*X(7)/)
      WRITE(1,290)(COEFF(7,I),I=1,9)
290   FORMAT(1X,9H8TH ORDER,2X,2HY=,F18.14,3H + ,F18.14,2H*X,3H +
     C F18.14,5H*X(2),3H + ,F18.14,5H*X(3),3H + ,F18.14,5H*X(4),3H +
     C /58X,F18.14,5H*X(5),3H + ,F18.14,5H*X(6),3H +
```

```
      C F18.14.5H*X(7)/58X.F18.14.5H*X(8)/)
        WRITE(1.300)(COEFF(8,I).I=1,10)
300     FORMAT(1X,9H9TH ORDER,2X,2HY=.F18.14,3H + .F18.14,2H*X.3H +
      C F18.14.5H*X(2).3H + .F18.14,5H*X(3).3H + ,F18.14.5H*X(4),3H +
      C /59X,F18.14,5H*X(5).3H + .F18.14.5H*X(6).3H +
      C F18.14.5H*X(7)/58X.F18.14.5H*X(8),3H + ,F18.14.5H*X(9)/)
        WRITE(1.310)(COEFF(9.I).I=1,11)
310     FORMAT(1X,10H10TH ORDER.1X,2HY=.F18.14.3H + .F18.14.2H*X.3H +
      C F18.14,5H*X(2).3H + .F18.14,5H*X(3).3H + .F18.14,5H*X(4).3H +
      C /58X.F18.14.5H*X(5).3H + .F18.14.5H*X(6).3H + .F18.14,5H*X(7)
      C /58X.F18.14,5H*X(8).3H + .F18.14,5H*X(9),3H + ,F18.14,6H*X(10
        WRITE(1.239)
239     FORMAT(1X.47HY VALUES PRODUCED WHEN INPUT X IS RESUBSTITUTED/
        WRITE(1.240)
240     FORMAT(1X.7HINPUT X.2X.7HINPUT Y.2X,9H2ND ORDER,2X,9H3RD ORDE
      C 2X.9H4TH ORDER.2X,9H5TH ORDER.2X.9H6TH ORDER.2X,9H7TH ORDER
      C 2X,9H8TH ORDER.2X,9H9TH ORDER.2X,10H10TH ORDER/)
        WRITE(1.250)((X1(I.J).I=1.11).J=1.N)
250     FORMAT(2X.F6.2,1X.F8.4.1X,F8.4.3X,F8.4.3X.F8.4.3X,F8.4
      C .3X.F8.4,3X.F8.4.3X.F8.4.3X.F8.4.3X.F8.4)
        WRITE(1.260)
260     FORMAT(///1X.39HTABLE OF PERCENTAGE ERROR W.R.T INPUT Y//)
        DO 261 I=1.N
        IF(X1(2.I).EQ.0.)X1(2.I)=0.0001
        DO 262 J=3,11
        X2(J.I)=X1(J.I)-X1(2.I)
        X1(J.I)=(X1(J.I)/X1(2.I)-1.)*100.
262     CONTINUE
261     CONTINUE
        WRITE(1,240)
        WRITE(1.250)((X1(I.J).I=1.11).J=1.N)
        WRITE(1.263)
263     FORMAT(///1X.38H DIFFERENCE BETWEEN Y AND CALCULATED Y//)
        WRITE(1.240)
        WRITE(1.250)((X2(I.J).I=1.11).J=1.N)
270     STOP
        END
```

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 2 - B

PROGRAM OUTPUT FOR FITCURV

APPLIED TO CHART C5.1.1 p = .01

511 P01

EQUATIONS GIVING CLOSEST APPROXIMATION BY METHOD OF LEAST SQUARES

```
2ND ORDER  Y=   .91537650529947 +    .38630035206670*X +    -.02760578132172*X(2)

3RD ORDER  Y=   .90381809478312 +    .39655155795734*X +    -.02925698211256*X(2) +     .00006880002975*X(3)

4TH ORDER  Y=   .93477781571045 +    .34336439610698*X +    -.01323137637039*X(2) +    -.00151887591364*X(3) +     .00004961487020*X(4)

5TH ORDER  Y=   .99401430418819 +    .15490925529003*X +     .07904059627068*X(2) +    -.01751458020189*X(3) +     .00118877470309*X(4) +
                                                            -.00002847899395*X(5)

6TH ORDER  Y=  1.00054729460084 +    .11601303076217*X +     .10746941303568*X(2) +    -.02505260139779*X(3) +     .00209438406590*X(4) +
                                                            -.00007873245700*X(5) +     .00000104694708*X(6)

7TH ORDER  Y=  1.00395740241467 +    .07580511993745*X +     .14863241807611*X(2) +    -.04044063209310*X(3) +     .00483117983366*X(4) +
                                                            -.00032902245373*X(5) +     .00001241382802*X(6) +    -.00000020298000*X(7)

8TH ORDER  Y=   .99315557767654 +    .35477917065552*X +    -.22937282669928*X(2) +     .14727747746571*X(3) +    -.04098010476414*X(4) +
                                                             .00573174759757*X(5) +    -.00043389252927*X(6) +     .00001694203104*X(7)
                                                            -.00000026789065*X(8)

9TH ORDER  Y=   .99961814855004 +   -.06689971049393*X +     .48655466106343*X(2) +    -.30247747616381*X(3) +     .10095916720931*X(4) +
                                                            -.01954416984425*X(5) +     .00222518879583*X(6) +    -.00014669754045*X(7)
                                                             .00000517686055*X(8) +    -.00000007562016*X(9)

10TH ORDER Y=  1.00119359892417 +   -.36822578165055*X +    1.09471915069462*X(2) +    -.76458593735606*X(3) +     .28129220517922*X(4) +
                                                            -.06045354379233*X(5) +     .00792783123499*X(6) +    -.00064206372465*X(7)
                                                             .00003130978783*X(8) +    -.00000084163900*X(9) +     .00000000956785*X(10)
```

S11 P01

Y VALUES PRODUCED WHEN INPUT X IS RESUBSTITUTED

INPUT X	INPUT Y	2ND ORDER	3RD ORDER	4TH ORDER	5TH ORDER	6TH ORDER	7TH ORDER	8TH ORDER	9TH ORDER	10TH ORDER
0.00	1.0000	.9154	.9038	.9348	.9940	1.0005	1.0040	.9932	.9996	1.0012
1.00	1.2000	1.2741	1.2712	1.2634	1.2115	1.2010	1.1925	1.2303	1.2004	1.1913
2.00	1.5000	1.5776	1.5804	1.5572	1.4980	1.4931	1.4941	1.4671	1.5075	1.5259
3.00	1.8000	1.8258	1.8320	1.8088	1.7866	1.7907	1.7971	1.7701	1.7742	1.7642
4.00	2.0000	2.0189	2.0263	2.0120	2.0325	2.0406	2.0445	2.0549	2.0226	2.0108
5.00	2.2000	2.1567	2.1638	2.1620	2.2092	2.2151	2.2131	2.2434	2.2291	2.2362
6.00	2.4000	2.2394	2.2447	2.2549	2.3050	2.3051	2.2995	2.3148	2.3358	2.3484
7.00	2.3000	2.2668	2.2697	2.2881	2.3195	2.3141	2.3095	2.2950	2.3233	2.3203
8.00	2.2000	2.2390	2.2390	2.2604	2.2604	2.2529	2.2529	2.2238	2.2238	2.2103
9.00	2.1000	2.1560	2.1531	2.1716	2.1403	2.1349	2.1395	2.1250	2.0967	2.0933
10.00	2.0000	2.0178	2.0124	2.0226	1.9724	1.9726	1.9782	1.9936	1.9726	1.9949
11.00	1.8000	1.8244	1.8174	1.8156	1.7683	1.7741	1.7761	1.8064	1.8218	1.8301
12.00	1.6000	1.5757	1.5683	1.5540	1.5335	1.5416	1.5377	1.5480	1.5803	1.5691
13.00	1.2000	1.2719	1.2657	1.2425	1.2647	1.2688	1.2623	1.2353	1.2313	1.2213
14.00	.9000	.9128	.9109	.8867	.9460	.9411	.9401	.9130	.8727	.8905
15.00	.6000	.4986	.5015	.4937	.5456	.5349	.5435	.5813	.6111	.6025
16.00	0.0000	.0291	.0407	.0716	.0124	.0189	.0155	.0047	-.0017	-.0003

511 P01

TABLE OF PERCENTAGE ERROR W.R.T INPUT Y

INPUT X	INPUT Y	2ND ORDER	3RD ORDER	4TH ORDER	5TH ORDER	6TH ORDER	7TH ORDER	8TH ORDER	9TH ORDER	10TH ORDER
0.00	1.0000	-8.4623	-9.6182	-6.5222	-.5986	.0547	.3957	-.6844	-.0392	.1193
1.00	1.2000	6.1726	5.9318	5.2868	.9675	.0828	-.6276	2.5228	.0329	-.7287
2.00	1.5000	5.1703	5.3629	3.8149	-.1342	-.4608	-.3926	-2.1928	.4981	1.7256
3.00	1.8000	1.4348	1.7788	.4888	-.7453	-.5185	-.1585	-1.6587	-1.4339	-1.9865
4.00	2.0000	.9443	1.3158	.6013	1.6266	2.0286	2.2240	2.7433	1.1290	.5409
5.00	2.2000	-1.9666	-1.6477	-1.7284	.4200	.6856	.5961	1.9746	1.2771	1.6471
6.00	2.4000	-6.6929	-6.4693	-6.0476	-3.9591	-3.9539	-4.1889	-3.5484	-2.6744	-2.1491
7.00	2.3000	-1.4437	-1.3180	-.5156	.8465	.6143	.4147	-.2174	1.0109	.8821
8.00	2.2000	1.7732	1.7732	2.7474	2.7474	2.4048	2.4048	1.0829	1.0834	.4696
9.00	2.1000	2.6672	2.5296	3.4085	1.9167	1.6625	1.8811	1.1887	-.1562	-.3179
10.00	2.0000	.8901	.6218	1.1278	-1.3783	-1.3720	-1.0900	-.3215	-1.3709	-.7564
11.00	1.8000	1.3545	.9646	.9654	-1.7610	-1.4364	-1.3272	.3577	1.2090	1.6748
12.00	1.6000	-1.5157	-1.9801	-2.8732	-4.1548	-3.6522	-3.8965	-3.2474	-1.2300	-1.9342
13.00	1.2000	5.9920	5.4760	3.5410	5.3922	5.7324	5.1925	2.9422	2.6066	1.7712
14.00	.9000	1.4276	1.1065	-1.4735	5.1084	4.5639	4.4503	1.4499	-3.0329	-1.0574
15.00	.6000	-16.9032	-16.4216	-17.7116	-9.0729	-10.8423	-9.4214	-3.1207	1.8545	.4124
16.00	.0001	********	********	********	********	********	********	********	********	********

511 P01

DIFFERENCE BETWEEN Y AND CALCULATED Y

INPUT X	INPUT Y	2ND ORDER	3RD ORDER	4TH ORDER	5TH ORDER	6TH ORDER	7TH ORDER	8TH ORDER	9TH ORDER	10TH ORDER
0.00	1.0000	-.0846	-.0962	-.0652	-.0060	.0005	.0040	-.0068	-.0004	.0012
1.00	1.2000	.0741	.0712	.0634	.0116	.0010	-.0075	.0303	.0004	-.0097
2.00	1.5000	.0776	.0804	.0572	-.0020	-.0069	-.0059	-.0329	.0075	.0259
3.00	1.8000	.0258	.0320	.0088	-.0134	-.0093	-.0029	-.0299	-.0258	-.0358
4.00	2.0000	.0189	.0263	.0120	.0325	.0406	.0445	.0549	.0225	.0104
5.00	2.2000	-.0433	-.0362	-.0380	.0092	.0151	.0131	.0434	.0281	.0362
6.00	2.4000	-.1606	-.1553	-.1451	-.0950	-.0949	-.1005	-.0852	-.0642	-.0516
7.00	2.3000	-.0332	-.0303	-.0119	.0195	.0141	.0095	-.0050	.0233	.0203
8.00	2.2000	.0390	.0390	.0604	.0604	.0529	.0529	.0238	.0238	.0103
9.00	2.1000	.0560	.0531	.0716	.0403	.0349	.0395	.0250	-.0033	-.0067
10.00	2.0000	.0178	.0124	.0226	-.0276	-.0274	-.0218	-.0064	-.0274	-.0151
11.00	1.8000	.0244	.0174	.0156	-.0317	-.0259	-.0239	.0064	.0218	.0301
12.00	1.6000	-.0243	-.0317	-.0460	-.0655	-.0584	-.0623	-.0520	-.0197	-.0309
13.00	1.2000	.0719	.0657	.0425	.0647	.0688	.0623	.0353	.0313	.0213
14.00	.9000	.0128	.0100	-.0133	.0460	.0411	.0401	.0130	-.0273	-.0095
15.00	.6000	-.1014	-.0985	-.1063	-.0544	-.0651	-.0565	-.0187	.0111	.0025
16.00	0.0000	.0290	.0406	.0715	.0123	.0188	.0154	.0046	-.0018	-.0004

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 3

OVERLAY 52 "RC22"

```
COMPILE AS-COLUMN/1                                  1226  31/08/83  PAGE0002
*OVERLAY

          ↑RC22↑
          ↑↑
          SUBPROGRAMS...
          ↑↑**********
          ↑↑
          CENTRY,EFLTH,BCLASH,RTYP
          ↑↑
          ↑↑
          ↑↑
                ENTRIES...
          ↑↑******
                CENTRY
          ↑↑
          ↑↑***************************************************************
                SUBROUTINE CENTRY
          ↑↑
          ↑↑ENTRY TO THE WHOLE RECTANGULAR COLUMN MODULE
          ↑↑
                PUBLIC/COLUMNS/CSP(,),P(,),ST(),MS(),IGL(,),IP(,),KOLUMN
          ↑↑
                PUBLIC/RCSYS/NSYS,ISYS
          ↑↑
                PUBLIC/CR/B1,B2,FCU,FY,XSL,YSL,PP(,),IWARN,...
                          IERR,KSXY,COV,NOWR(),NOER()
                COV=ST(9)
                B1=CSP(1,2-KOLUMN)
                B2=CSP(2,2-KOLUMN)
          ↑↑COMPUTE THE EFFECTIVE LENGTH OF COLUMN TO BE CONSIDERED
          ↑↑
                CALL EFLTH
          ↑↑
                KSXY=0
          ↑↑TEST IF BEAM BAR CLASHING IS TO BE CONSIDERED
                IF(MS(20) .GT. 0) CALL BCLASH
                IERR=0
                IWARN=0
                LENGTH(P(,),N)
                IF(KOLUMN .EQ. 0) N=MS(3)
                MM=KOLUMN*MS(3)+1
                IJ=1
                IK=0
                IF(XSL .GT. 0 .OR. YSL .GT. 0) IJ=2
          ↑↑ PUT ALL LOAD CASES AT HEAD AND FOOT OF COLUMN IN ARRAY PP
                DO 10 I=MM,N
                GOTO(5,6),IJ
          ↑↑ PART ONE FOR SHORT COLUMNS
          5     INC=2
                I8=1
                X1=P(2,I)
                X2=P(4,I)
                Y1=P(3,I)
                Y2=P(5,I)
                X1=ABS(X1)
```

```
              Y1=ABS(Y1)
              Y2=ABS(Y2)
              IF(X1.GT.X2.AND.Y1.GT.Y2)INC=1
              IF(X1.LT.X2.AND.Y1.LT.Y2)INC=1
              IF(X1.LT.X2.AND.Y1.LT.Y2)I8=2
     17       I2=2*I8
              I4=6-2*I8
              I3=1+2*I8
              I5=7-2*I8
              LENGTH(IP(,I),N10)
              N11=N10+3
              DO 20 I1=1,INC
              IK=IK+1
              REDEFINE(PP(,),IK),(PP(,IK),N11)
              I6=2-I1
              I7=I1-1
              X=P(I2,I)*I6+P(I4,I)*I7
              Y=P(I3,I)*I6+P(I5,I)*I7
              X=ABS(X)
              Y=ABS(Y)
              PP(1,IK)=P(1,I),X,Y
              DO 19 J19=4,N11
              PP(J19,IK)=IP(J19-3,I)
     19       CONTINUE
     20       CONTINUE
     ↑↑
              GOTO 10
     6        X=P(2,I)
              Y=P(3,I)
              X=ABS(X)
              Y=ABS(Y)
              X1=P(4,I)
              Y1=P(5,I)
              IK=IK+1
     ↑↑
              X1X=ABS(AMAX1(X1,X))*.4
              Y1Y=ABS(AMAX1(Y1,Y))*.4
              XX=.4*AMIN1(X,X1)
              XX=XX+.6*AMAX1(X1,X)
              YY=.4*AMIN1(Y,Y1)
              YY=YY+.6*AMAX1(Y1,Y)
              X=ABS(XX)
              IF(X .LT. X1X) X=X1X
              Y=ABS(YY)
              IF(Y .LT. Y1Y) Y=Y1Y
     ↑↑
     ↑↑ PART TWO FOR SLENDER COLUMNS
              REDEFINE (PP(,),IK)
              LENGTH(IP(,I),N10)
              N11=N10+3
              REDEFINE(PP(,IK),N11)
     ↑↑
              PP(1,IK)=P(1,I),X,Y
     ↑↑
              DO 119 J19=4,N11
              PP(J19,IK)=IP(J19-3,I)
     119      CONTINUE
```

```
COMPILE AS-COLUMN/1                                  1226   31/08/83   PAGE0004
          ↑↑     GET CONCRETE AND STEEL STRENGTHS FROM INTERFACE
                 FCU=ST(1+KOLUMN*3)
                 FY=ST(6)
          ↑↑     INITIALIZE SYSTEM FLAGS NSYS=NUMBER OF LOADS, ISYS=CURRE
                 ISYS=0
                 NSYS=IK
                 CALL RTVP
                 IF(IERR.NE.0) GO TO 130
                 PERFORM ↑RC24↑ DRIVER
                 RETURN
          130    STOP
                 END
```

```
↑↑*****************************************************************************
      SUBROUTINE BCLASH
↑↑
      PUBLIC/CR/B1,B2,SXY(,),NB1,NB2,KSXY
      PUBLIC/COLUMNS/MS().ST()
↑↑ TO ESTABLISH THE PERMITTED NO OF BARS IN EACH FACE OF THE C
↑↑ IF BEAM BAR CLASHING IS TO BE CONSIDERED
↑↑ KEY TO VARIABLES NAMES IS .....CZ COLUMN ZONE..
↑↑ CB COLUMN BAR SIZE.. BZ BEAM BAR ZONE.. BB BEAM BAR SIZE
↑↑ NB1 NO OF BARS IN Y-DIRECTION AND NB2 IN Y-DIRECTION
      KSXY=1
      CB=MS(12)
      BB=MS(18)
      CZ=1.1*CB
      BZ=1.1*BB          ↑↑BEAM BARS IN PAIRS ONE ABOVE THE OTHER
      EB=(1+MS(19))*BB↑↑BAR SPACING SEE AS1480 TABLE 6.7.1
      U15=AMAX1(CZ,EB)
      X=1.5*ST(12)       ↑↑DITTO FOR AGGREGATE SIZE
      D15=AMAX1(BZ,CB,X)
      L=6
      IF(CB.GT.24.0)L=10
      NB1=0
      NB2=0
      X1=X/1.2           ↑↑MINIMUM COVER 1.25*AGGREGATE SIZE
      C=AMAX1(40.,ST(9),X1)
      IF(MS(20) .NE. 2)NB1=(B1-2*(C+L+U15)-D15)/(U15+D15)
      IF(MS(20) .NE. 1)NB2=(B2-2*(C+L+U15)-D15)/(U15+D15)
      REDEFINE(SXY(,),2),(SXY(,1),2),(SXY(,1),2)
↑↑
      SXY(2,1)=(B1-2*C-U15)/(NB1+1)
      SXY(2,2)=(B2-2*C-U15)/(NB2+1)
      SXY(1,1)=SXY(2,1)+U15/2
      SXY(1,2)=SXY(2,2)+U15/2
↑↑
      RETURN
      END
```

```
↑↑************************************************************************
      SUBROUTINE EFLTH
↑↑    ****************
↑↑
      PUBLIC/COLUMNS/KOLUMN,GL(),MS()
↑↑
      PUBLIC/CP/EL,EX,EY,XSL,YSL,B1,B2
↑↑
↑↑ CALCULATION OF EFFECTIVE LENGTH IS ACCORDING TO CP110
↑↑ EQUATIONS - LENGTH IS CONSIDERED IN BOTH X AND Y FACES
↑↑
      I=2*KOLUMN
      J=I*3
      X1=GL(10)
      X2=GL(12)
      Y1=GL(11+I)
      Y2=GL(13+I)
      D1=GL(2+J)
      D2=GL(3+J)
      K=MS(5)
↑↑
      X=AMIN1(X1,X2)
      Y=AMIN1(Y1,Y2)
      I1=K
      IF(I1.EQ.4) I1=0
      I2=I1
      IF(I1.LT.2) GOTO 3
      I1=0
      I2=0
      IF(K.EQ.2) I2=1
      IF(K.EQ.3) I1=1
↑↑
3     E1=D1*(.7+.3*I1+(.05+.1*I1)*(X1+X2))
      E2=D1*(.85+1.15*I1+(.05+.25*I1)*X)
      IF(I1.EQ.0) GOTO 6
      E1=E1-D1*(X1+X2)**2*0.0025
      GOTO 8
6     E1=AMIN1(E1,D1)
8     EX=AMIN1(E1,E2)
↑↑
      E1=D2*(.7+.3*I2+(.05+.1*I2)*(Y1+Y2))
      E2=D2*(.85+1.15*I2+(.05+.25*I2)*Y)
      IF(I2.EQ.0) GOTO 10
      E1=E1-D2*(Y1+Y2)**2*0.0025
      GOTO 12
10    E1=AMIN1(E1,D2)
12    EY=AMIN1(E1,E2)
↑↑ SLENDERNESS RATIO IS CALCULATED AS FOLLOWS
      EL=AMIN1(EX,EY)
      XSL=0
      YSL=0
      E1=EX/(0.3*B1)
      E2=EY/(0.3*B2)
      IF(E1.GT.20.0) XSL=0.3*E1
      IF(E2.GT.20.0) YSL=0.3*E2
↑↑
↑↑
      RETURN
```



```
↑↑
      END
```

```
      ******************************************************************
            SUBROUTINE RTVP
            HIGH COEFP,CP
            PUBLIC/CP/B1,B2,COEFB(,),COEFP(,,),COV,FCU,FY,SLOPE(),...
            IERR,NOER(),PMAX,PMIN
            DIMENSION CP(,,),CB(,),SL()
      ** SET PMIN
            PMIN = 0.595 * FCU
      ** CALCULATE GX,GY,GXY
      ** ASSUME 36MM BARS AND 10MM STIRRUPS
            GX=(B2-2*(COV+18+10))/B2
            GY=(B1-2*(COV+18+10))/B1
            GXY=AMIN(GX,GY)
      ** JUMP TO COEFFICIENT SET DEPENDING ON MATERIAL PROPS
            IF(FCU.EQ.25..AND.FY.EQ.230.) GOTO 100
            IF(FY.NE.410.) GOTO 85
            IF(FCU.EQ.25.) GOTO 200
            IF(FCU.EQ.30.) GOTO 300
            IF(FCU.EQ.40.) GOTO 400
            IF(FCU.EQ.45.) GOTO 500
            GOTO 85
      10    IFL=0
      15    IFL=IFL+1
      **FIND RATIOS TO BE USED TO MODIFY COEFFICIENT SET
            IF(IFL.EQ.1) G=GX
            IF(IFL.EQ.2) G=GY
            IF(IFL.EQ.3) G=GXY
            IF(IFL.GT.3) RETURN
            IF(G.LE.0.6.OR.G.GT.1.) GOTO 95
            IF(G.GT.0.9) G=0.9
            IF(G.GE.0.7) GOTO 20
            GH=(G-0.6)/0.1
            GL=1-GH
            IL=1
            IH=2
            GOTO 50
      20    IF(G.GE.0.8) GOTO 30
            GH=(G-0.7)/0.1
            GL=1-GH
            IL=2
            IH=3
            GOTO 50
      30    GH=(G-0.8)/0.1
            GL=1-GH
            IH=4
            IL=3
      50    IF(IFL.NE.3) GOTO 55
            IL=IL+4
            IH=IH+4
      ** PRODUCE THE MODIFIED COEFFICIENT SET FOR THE TYPE  CONSIDER
      55    DO 65 J=1,8
            DO 60 K=1,7
            COEFP(IFL,J,K)=GL*CP(IL,J,K)+GH*CP(IH,J,K)
      60    CONTINUE
      65    CONTINUE
            DO 70 J=1,2
            COEFB(IFL,J)=GL*CB(IL,J)+GH*CB(IH,J)
      70    CONTINUE
```

```
             SLOPE(IEL) = GL * SL(IL) + GH * SL(IH)
             GOTO 15
       85    IERR=IERR+1
             MESSAGE↑RUN ABORTED - INVALID ECU OR EY↑
             SKIP(2)LINES
             PRINT 86
       86    FORMAT(↑RUN ABORTED - INVALID ECU OR EY↑)
             RETURN
       95    IERR=IERR+1
             MESSAGE↑RUN ABORTED - INVALID G VALUE↑
             SKIP(2)LINES
             PRINT 96
       96    FORMAT(↑RUN ABORTED - INVALID G VALUE↑)
             RETURN
      100    CP(1,1,1)=...
              1.00054729460100,  .11601303076220,  .10746941303570,...
              -.02505260139779,  .00209438406590, -.00007873245700,...
               .00000104694708
             CP(1,2,1)=...
              1.59914280002100,  .12537012891300,  .13100391486410,...
              -.03635927018344,  .00368446210220, -.00017119281303,...
               .00000301184489
             CP(1,3,1)=...
              2.32543085512400,  .00303645736643,  .14727071685310,...
              -.03456445697717,  .00308831254224, -.00012612512147,...
               .00000194208925
             CP(1,4,1)=...
              2.90706714047499,  .08495338411169,  .10458171330110,...
              -.02755912348774,  .00251398203350, -.00010215696184,...
               .00000154596062
             CP(1,5,1)=...
              3.49049997942900,  .16118650873150,  .06283254237995,...
              -.02010381471604,  .00185412597821, -.00007365477184,...
               .00000107501256
             CP(1,6,1)=...
              4.09004728091600,  .12716397064510,  .06688385672487,...
              -.02023883320900,  .00181178453041, -.00006951617225,...
               .00000097572470
             CP(1,7,1)=...
              4.67643669982701,  .21137951957770,  .01308142152320,...
              -.01030046214275,  .00098698950774, -.00003784470103,...
               .00000051973321
             CP(1,8,1)=...
              5.26159290529000,  .26390074331600, -.01787664678264,...
              -.00523072290883,  .00059867080847, -.00002354256377,...
               .00000031792729
             CB(1,1)=...
              -.1091, 7.6000
       ↑↑
             CP(2,1,1)=...
              1.01365741545100,  .16608714861570,  .11011674403960,...
              -.02726196668510,  .00234822525858, -.00009166986735,...
               .00000131967977
             CP(2,2,1)=...
              1.70969482069800,  .23469203840430,  .07688329348249,...
              -.02230001400705,  .00211056204897, -.00008799631120,...
               .00000139430969
```

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 4

OVERLAY 54 "RC24"

```
COMPILE AS-COLUMN/1                                              I230   31/08/83   PAGE0002
*OVERLAY

          ^RC24^
LINE
          ^^
                  SUBPROGRAMS...
          ^^**********
          ^^
                  DRIVER,SORT,ZCOLLOAD,FINDP,ARRANG,SORTL,RED,CALCP,FUNCP
          ^^
          ^^
          ^^
                  ENTRIES ...
          ^^******
          ^^
                  DRIVER
          ^^
          ^^*******************************************************************
                  SUBROUTINE DRIVER
          ^^
          ^^
                  PUBLIC/COLUMNS/P(,),GL(),ST(),MS(),IGL(,),KOLUMN,
          ^^
                  PUBLIC/PCSYS/NSYS,ISYS,IISYS,SLD(,),LDBI()
          ^^
                  PUBLIC/CR/B1,B2,FCU,FY,NT1,EX,EY,FIVE,ICNT,XSL,YSL,AA(,)
                             DIAM(),PCLD(),CLD(,),NOER(,),NOWR(),PP(,),IWAR
                             IERR,COV,TBLE(,),FK(,),IERCOL(,),MONX(,),MONY(
                             BIAX(,),STEELP(,),COEFP(,,),COEFB(,),SLOPE(),.
                             COST()
                  MEDIUM COEFP
                  REAL MONX,MONY
          ^^
                  DIMENSION LST(,),LGRP(,),SLDTEMP(,)
                  DESTROY TBLE(,),LDBI(),FK(,),IERCOL(,)
                  N=NSYS
                  INIT=-2
                  NT1=0
                  ISH=1
          ^^ ROUTINE FOR DRIVING THE CALCULATION OF COLUMN ANALYSIS
          ^^INITIALIZE ALL THE CONSTANTS
          ^^
                  SECA=B1*B2
                  FIVE=0.08*SECA
                  REDEFINE(DIAM(),7)
                  DIAM(1)=12,16,20,24,28,32,36
          ^^PUT ALL BAR ARES (OF PAIRS OF BARS) IN ARRAY AA
          ^^1 TO 7 CORRESPONDS TO 12 TO 36 MM. BAR SIZE
                  DO 60 I=1,7
                  REDEFINE(AA(,),I),(AA(,I),30)
          ^^
                  AJ=1.571428*DIAM(I)**2
          ^^
                  DO 3 J=1,30
                  AA(J,I)=AJ*J
          ^^
          3       CONTINUE
```

```
        **NT1 CORRESPONDS TO THE MAX BAR SIZE THAT CAN BE USED AT CORN
        **WITHOUT VIOLATING THE 8 PERCENT LIMIT OF AREA
              IF(MS(12) .EQ. 0) GOTO 59
              IF(DIAM(I) .GT. MS(12)) GOTO 60
        59    IF(AA(2,I) .LT. FIVE) NT1=I
        60    CONTINUE
        **
              REDEFINE(PCLD(),15)
              KPC=KOLUMN*6+2
              PCLD(1)=GL(KPC),GL(KPC+1)   **CLEAR HEIGHT IN X-DIR AND
                                          **CLEAR HEIGHT IN Y-DIR
              PCLD(3)=ST(9),ST(8)   **COVER , KICKER SIZE
        **
              PCLD(5)=YSL,YSL,0.0 **SLENDERNESS RATIO IN X-DIR , Y-DIR
        **
              PCLD(9)=SECA **SECTION AREA
        **
              PCLD(10)=SECA*B2*B2/12 **INERTIA IN X-DIR
        **
              PCLD(11)=SECA*B1*B1/12** INERTIA IN Y-DIR
        **
              PCLD(12)=EX,EY   **EFFECTIVE LENGTH IN X-DIR , Y-DIR
        **
        **INITIALIZE SHORT COLUMN FLAG 1 IF SHORT, 0 IF SLENDER
        **
              IF(XSL .GT. 0 .OR. YSL .GT. 0) ISH=0 **COLUMN IS SLENDER
              ITSYS=ISH
        **
              IX1=10
        **  THE FOLLOWING IF() WAS COMMENTED OUT BECAUSE ASCOLUMN
        **  DOES NOT DIFFERENTIATE BETWEEN SHORT OR LONG
        **      IF(ISH .EQ. 1) IX1=8
        **
              LENGTH(IGL(,),NGR)
              REDEFINE(LGPP(,),NGR)
              DO 88 J88=1,NGR
              REDEFINE(LGPP(,J88),2)
        88    CONTINUE
        **
        **CREATE AN INTERFACE ARRAY FOR SORTED LOAD COMBINATIONS
              DO 4 I=1,N
        **
              REDEFINE(SLD(,),I),(CLD(,),I)
        **
              LENGTH(PP(,I),N11)
              REDEFINE(SLD(,I),N11),(CLD(,I),IX1)
        **
              EQUATE 4 (SLD(,I),S),(PP(,I),P1)
        **
              X=1000
        **
              DO 4 J=1,N11
              IF(J .NE. 1)X=1E6    ** CHANGE UNITS TO N AND MM
        **
              IF(J .GT. 3)X=1
              S(J)=P1(J)*X
        **
        4     CONTINUE
```

```
      ↑↑
           DESTROY PP
      ↑↑
           LLK=1
           KS5=1
           KF5=1
           CALL SORT(SLD,N,LST,LLK,LGRP,NGP)  ↑↑ SORT LOAD COMBINAT
      ↑↑
           SWAP(SLD(,),SLDTEMP(,))    ↑↑COPY SLD INTO SLDTEMP
           J=0
      ↑↑COPY SLDTEMP BACK INTO SLD IN CORRECT ORDER OF AXIAL LOAD
           NEW PAGE
           SKIP (5) LINES
           MESSAGE ↑ *******  LOADINGS CONSIDERED IN DESIGN↑
           SKIP (3) LINES
           PRINT 100
      100  FORMAT(1X,↑LOAD          AXIAL              MOMENT-X↑,...
             ↑               MOMENT-Y↑)
           PRINT 200
      200  FORMAT(1X,↑         FORCE    STRESS      FORCE    STRESS↑,...
                     ↑       FORCE    STRESS↑)
           SKIP (1) LINE
           DO 6 ILK=1,LLK
           IF(N .GT. 1) KS5=LST(1,ILK)
           IF(N .GT. 1) KF5=LST(2,ILK)
           DO 5 ISYS=KS5,KF5
           I=ISYS
           EQUATE 5 (SLDTEMP(,I),W)
           IF(W(1) .LT. 0) GOTO 5
           J=J+1
           REDEFINE (SLD(,),J),(SLD(,J),3)
           EQUATE 5 (CLD(,J),C),(SLD(,J),SL)
           SL(1)=W(1)          ↑↑AXIAL LOAD IN N
           SL(2)=W(2)          ↑↑ MOMENT-X IN NMM
           SL(3)=W(3)          ↑↑ MOMENT-Y IN NMM
           C(1)  = 0
           C(2)  = W(1)/1000           ↑↑ AXIAL LOAD IN KN
           C(3)  = W(2)/1E6            ↑↑ MOMENT-X IN KNM (INIT)
           C(4)  = W(3)/1E6            ↑↑ MOMENT-Y IN KNM (INIT)
           C(5)  = 0                   ↑↑ WIND EFFECT NOT CONSIDERED
           C(6)  = SL(1)/(SECA)         ↑↑AXIAL LOAD / BH -- IN N/MM2
           C(7)  = SL(2)/(SECA*B2)      ↑↑INITIAL MOMENT-X/BHH -- IN N/
           C(8)  = SL(3)/(SECA*B1)      ↑↑INITIAL MOMENT-Y/BBH -- IN N/
           PRINT 300,J,C(2),C(6),C(3),C(7),C(4),C(8)
      300  FORMAT(1X,I2,2X,F10.3,F7.3,2X,F10.3,F7.3,2X,F10.3,F7.3)
      5    CONTINUE
      6    CONTINUE
           NSYS=J
           SKIP (5) LINES
           MESSAGE ↑******** SECTION PROPERTIES↑
           SKIP (3) LINES
           GX = (B2 - 2 * (COV + 28))/B2
           GY = (B1 - 2 * (COV + 28))/B1
           PRINT 400,B1,B2,EX,EY
      400  FORMAT(1X,↑B1 =↑,F8.2,↑  B2 =↑,F8.2,↑  ELX =↑,F8.2,↑  ELY
           SKIP (1) LINES
           PRINT 500,GX,GY
```

```
      ↑↑
                CALL ZCOLLOAD   ↑↑CREATE LOADING CASE TYPES
                CALL FINDP     ↑↑DETERMINE DESIGN STEEL PERCENTAGE
      ↑↑
                IF(IERR .NE. 0) RETURN    ↑↑RETURN IF ANY ERRORS WERE ENC
      ↑↑
                CALL ARRANG    ↑↑ARRANGE STEEL PERCENTAGES IN EACH FACE
      ↑↑
      ↑↑          IF(IERR .NE. 0) RETURN   ↑↑DO NOT PROCEDE IF ERRORS WER
      ↑↑
                DESTROY MONX(,),MONY(,),BIAX(,),STEELP(,),COEFB(,),COEFP
                DESTROY SLOPE(),COST()
                MESSAGE ↑ BEFORE CALL TO PERFORM RECOUT ↑
                PERFORM ↑RECOUT↑ CROUT
      ↑↑
                STOP
                END
```

```
      ↑↑*************************************************************
             SUBROUTINE ZCOLLOAD
             PUBLIC/COLUMNS/CSP(,)
             PUBLIC/RCSYS/NSYS,SLD(,),LDBI()
             PUBLIC/CR/B1,B2,BIAX(,),IBIAX,IFLAG,IMON,MONX(,),...
             MONY(,),NSYSX,NSYSY,NSYSB,ECX,ECY
             REAL MONX,MONY
             NEW PAGE
             SKIP (5) LINES
             MESSAGE ↑***** DESIGN LOAD CASES AFTER -----↑
             MESSAGE ↑                   A) REMOVING REDUNDANT LOADS↑
             MESSAGE ↑                   B) APPLYING ADDITIONAL ECCENTRICI
             MESSAGE ↑                   C) APPLYING REDUCTION FACTORS↑
             SKIP (5) LINES
             IMON=0
             IBIAX=0
             NSYSX=0
             NSYSY=0
             NSYSB=0
             ECX=25 + .01*B2 ↑↑      + MAX(CSP(1,4),CSP(2,4))
             ECY=25 + .01*B1 ↑↑      + MAX(CSP(1,3),CSP(2,3))
             ECXY=MAX(ECX,ECY)
      ↑↑ NOTE THAT UNITS SHOULD BE NEWTONS AND MM
             DO 10 I=1,NSYS
             IF(SLD(1,I).LT.0.) GOTO 10
             SL1=SLD(1,I)/(B1*B2)
             SL2=SLD(2,I)/(B1*B2*B2)
             SL3=SLD(3,I)/(B1*B1*B2)
             BX=SL1*.03
             BY=SL1*.03
             IF(SL2.GT.BX.AND.SL3.GT.BY) GOTO 20
             IMON=IMON+1
             REDEFINE (MONX(,),IMON)
             REDEFINE (MONY(,),IMON)
             REDEFINE (MONX(,IMON),3)
             REDEFINE (MONY(,IMON),3)
             MONX(1,IMON)=SL1
             MONY(1,IMON)=SL1
             MONX(2,IMON)=SL2+SL1*ECX/B2
             MONX(3,IMON)=I
             MONY(2,IMON)=SL3+SL1*ECY/B1
             MONY(3,IMON)=I
             GOTO 10
      ↑↑  BIAXIAL LOADS MAY NEED TO BE COMBINED USING FACTOR
      20     IBIAX=IBIAX+1
             REDEFINE (BIAX(,),IBIAX)
             REDEFINE (BIAX(,IBIAX),3)
             REDEFINE (LDBI(),IBIAX)
             BIAX(1,IBIAX)=SL1
             BIAX(2,IBIAX)=SL2 + SL3 + SL1*ECXY/AMAX1(B1,B2)
             BIAX(3,IBIAX)=I
             LDBI(IBIAX) = I
      10     CONTINUE
             IF(IMON.EQ.0) GOTO 30
             CALL SORTL(MONX,IMON,NSYSX)
             CALL SORTL(MONY,IMON,NSYSY)
      30     IF(IBIAX.EQ.0) GOTO 40
```

```
40   IF(NSYSX.EQ.0) GOTO 50
     REDEFINE(MONX(.),NSYSX)
     REDEFINE(MONY(,NSYSX),3)
     IFLAG=1
     MESSAGE↑**    MONOAXIAL-X↑
     CALL RED(MONX,NSYSX)
     SKIP (5) LINES
50   IF(NSYSY.EQ.0) GOTO 60
     REDEFINE(MONY(,),NSYSY)
     REDEFINE(MONY(,NSYSY),3)
     IFLAG=2
     MESSAGE ↑**    MONOAXIAL-Y↑
     CALL RED(MONY,NSYSY)
     SKIP (5) LINES
60   IF(NSYSB.EQ.0) GO TO 70
     REDEFINE (BIAX(.),NSYSB)
     REDEFINE (BIAX(,NSYSB),3)
     IFLAG=3
     MESSAGE↑**       BIAXIAL LOADS↑
     CALL RED(BIAX,NSYSB)
70   IF(NSYSX .NE. 0) RETURN
     IF(NSYSY .NE. 0) RETURN
     IF(NSYSB .NE. 0) RETURN
     MESSAGE ↑**   ALL LOADS WERE FOUND TO BE REDUNDANT↑
     RETURN
     END
```

```
      ******************************************************************
            SUBROUTINE FINDP
            PUBLIC/CP/MONX(,),MONY(,),BIAX(,),NSYSX,NSYSY,NSYSB,...
            STEELP(,),IFLAG,NOERR(,),IERR
            REAL MONX,MONY
            REDEFINE(STEELP(,),2),(STEELP(,1),3),(STEELP(,2),3)
      **      MESSAGE * IN FINDP *
            IF(NSYSX .NE. 0) GO TO 5
            IF(NSYSY .NE. 0) GO TO 5
            IF(NSYSB .NE. 0) GO TO 5
            STEELP(1,1) = 0
            STEELP(1,2) = 0
            STEELP(2,1) = 0
            STEELP(2,2) = 0
            STEELP(3,1) = 0
            STEELP(3,2) = 0
            RETURN
      5     NEW PAGE
            SKIP (5) LINES
            MESSAGE ****** RESULTS OF STEEL PERCENTAGE CALCULATIONS*
            STEELP(1,1)=0
            STEELP(1,2) = 0
            IFLAG=1
            IF(NSYSX.EQ.0) GOTO 15
            SKIP (3) LINES
            MESSAGE***    MONOAXIAL-X*
            SKIP (2) LINES
            MESSAGE* LOAD          FORCE          MOMENT     PERC*
            SKIP (1) LINES
            DO 10 I=1,NSYSX
            CALL CALCP(MONX(,),I,S)
            III = INT(MONX(3,I))
            PRINT 100,III,MONX(1,I),MONX(2,I),S
      100   FORMAT(1X,I3,4X,F10.2,3X,F10.2,2X,F7.3)
            IF(S.LT.STEELP(1,1)) GOTO 10
            STEELP(1,1)=S
            STEELP(1,2)=MONX(3,I)
      10    CONTINUE
      15    STEELP(2,1)=0
            STEELP(2,2)=0
            IFLAG=2
            IF(NSYSY.EQ.0) GOTO 25
            SKIP(3) LINES
            MESSAGE***    MONOAXIAL-Y*
            SKIP (2) LINES
            MESSAGE* LOAD          FORCE          MOMENT     PERC*
            SKIP (1) LINES
            DO 20 I=1,NSYSY
            CALL CALCP(MONY(,),I,S)
            III = INT(MONY(3,I))
            PRINT 100,III,MONY(1,I),MONY(2,I),S
            IF(S.LT.STEELP(2,1)) GOTO 20
            STEELP(2,1)=S
            STEELP(2,2)=MONY(3,I)
      20    CONTINUE
      25    STEELP(3,1)=0
            STEELP(3,2)=0
```

```
            IF(NSYSB.EQ.0) GOTO 35
            SKIP(3) LINES
            MESSAGE↑**   BIAXIAL↑
            SKIP (2) LINES
            MESSAGE↑ LOAD          FORCE          MOMENT     PERC↑
            SKIP (1) LINES
            DO 30 I=1,NSYSB
            CALL CALCP(BIAX(,),I,S)
            III = INT(BIAX(3,I))
            PRINT 100,III,BIAX(1,I),BIAX(2,I),S
            IF(S.LT.STEELP(3,1)) GOTO 30
            STEELP(3,1)=S
            STEELP(3,2)=BIAX(3,I)
   30       CONTINUE
   35       IF(STEELP(1,1) .GT. 8.0) GOTO 40
   36       IF(STEELP(2,1) .GT. 8.0) GOTO 50
   37       IF(STEELP(3,1) .GT. 8.0) GOTO 60
            RETURN
   40       IERR=IERR+1
   ↑↑         LENGTH(NOER(,1),N)
   ↑↑         N=N+1
   ↑↑         REDEFINE(NOER(,1),N)
   ↑↑         NOER(N,1)=STEELP(1,2)*100
            SKIP (2) LINES
            MESSAGE↑ ERROR - STEEL PERCENTAGE FOR X DIR EXCEEDS 8.0↑
            GOTO 36
   50       IERR=IERR+1
   ↑↑         LENGTH(NOER(,1),N)
   ↑↑         N=N+1
   ↑↑         REDEFINE(NOER(,1),N)
   ↑↑         NOER(N,1)=STEELP(2,2)*100
            SKIP (2) LINES
            MESSAGE↑ ERROR - STEEL PERCENTAGE FOR Y DIR EXCEEDS 8.0↑
            GOTO 37
   60       IERR=IERR+1
   ↑↑         LENGTH(NOER(,1),N)
   ↑↑         N=N+1
   ↑↑         REDEFINE(NOER(,1),N)
   ↑↑         NOER(N,1)=STEELP(3,2)*100
            SKIP (2) LINES
            MESSAGE↑ ERROR - STEEL PERCENTAGE FOR BIAX EXCEEDS 8.0↑
            RETURN
            END
```

```
↑↑**********************************************************************
          SUBROUTINE ARRANG
          PUBLIC/COLUMNS/ MS()
          PUBLIC/PCSYS/ NSYS
          PUBLIC/CR/B1,B2,CLD(,),COV,IBARS(,),ICNT,...
                IERR,KSXY,NB1,NB2,NOEF(,),IWARN,NOWR(),NUTRAL,...
                STEELP(,),TBLE(,),SXY(,),COST(),NT1,ZUN(),...
                ASC,EK(,),ALP(),ECX,ECY,IFLAG,IWOR,NSYSB,PMAX,PMIN
          LOCAL NBIBAR(7),IFACEX(7),IFACEY(7),EFFBAR(7),IDIFFX(7),.
                    IDIFFY(7),NSPACX(7),NSPACY(7),AST(3),NFACEX(7),
                    NFACEY(7),SPACEX(7),SPACEY(7),SP1(7),SP2(7),SP
          INTEGER T1,T2
↑↑          MESSAGE ↑ IN ARRANG ↑
          IF(STEELP(1,1) .GT. 0.0) GO TO 5
          IF(STEELP(2,1) .GT. 0.0) GO TO 5
          IF(STEELP(3,1) .GT. 0.0) GO TO 5
          IF(B1.GT.B2) STEELP(1,1)=1.0
          IF(B1.LE.B2) STEELP(2,1)=1.0
          GO TO 6
5         IF(STEELP(1,1).LT.1.0.AND.STEELP(1,1).GT.0.) STEELP(1,1)
          IF(STEELP(2,1).LT.1.0.AND.STEELP(2,1).GT.0.) STEELP(2,1)
          IF(STEELP(3,1).LT.1.0.AND.STEELP(3,1).GT.0.) STEELP(3,1)
6         ICOST=0
          NEW PAGE
          SKIP (10) LINES
          MESSAGE ↑ *****************↑
          MESSAGE ↑ DESIGN SOLUTIONS↑
          MESSAGE ↑ *****************↑
          SKIP(3) LINES
          MESSAGE ↑** TARGET STEEL PERCENTAGES ↑
          SKIP (1) LINES
          PRINT 700,STEELP(1,1),STEELP(1,2)
700       FORMAT(1X,↑MONO-X    =↑,F5.2,↑  - DERIVED FROM LOADING NO.
          PRINT 800,STEELP(2,1),STEELP(2,2)
800       FORMAT(1X,↑MONO-Y    =↑,F5.2,↑  - DERIVED FROM LOADING NO.
          PRINT 900,STEELP(3,1),STEELP(3,2)
900       FORMAT(1X,↑BIAXIAL   =↑,F5.2,↑  - DERIVED FROM LOADING NO.
          SKIP (2) LINES
          PRINT 105
105       FORMAT(1X,↑SOLN   BAR    NO. OF       STEEL PERCENTAGE
          PRINT 200
200       FORMAT(1X,↑ NO.   SIZE      BARS↑)
          PRINT 300
300       FORMAT(1X,↑                 X   Y        X         Y      TOTAL↑)
          SKIP (2) LINES
↑↑          MESSAGE ↑ AFTER LABEL 5 ↑
↑↑          MESSAGE ↑ B1 = ↑,B1,↑ B2 = ↑,B2,↑ COVER = ↑,COV
          TCOST=10000.
          DX1=B1
          DY1=B2
          PI = 3.14159
↑↑
↑↑    DEFINE COST ARRAY
↑↑
          DEFINE(COST(),NT1)
          DO 10 I=1,3
          AST(I)=STEELP(I,1)*B1*B2/100.
```

```
10    CONTINUE
      DO 50 I=1,NT1
      DB=4 * LOW (I+2)
      BAR=PI * DB *DB / 4.0
      NBIBAR(I)=4*INT(((AST(3)/BAR)*.95/4)+1)
      IF(NBIBAR(I).EQ.8) NBIBAR(I)=12
      IFACEY(I)=(NBIBAR(I)/4)+1
      IFACEX(I) = IFACEY(I)
      T1=IFACEX(I)/2
      T2 = INT(T1)
      IF(T1.EQ.T2) EFFBAR(I)=.97*IFACEX(I)-2.55
      IF(T1.NE.T2) EFFBAR(I)=IFACEX(I)-3
      IF(IFACEX(I).EQ.2) EFFBAR(I)=0
      IDIFFX(I)=INT(.95*.5*((AST(1)-EFFBAR(I)*BAR)/BAR))+1.0 ..
                - IFACEX(I)
      IDIFFY(I)=INT(.95*.5*((AST(2)-EFFBAR(I)*BAR)/BAR))+1.0 ..
                - IFACEY(I)
      IF(IDIFFX(I).GT.0) IFACEX(I)=IFACEY(I)+IDIFFX(I)
      IF(IDIFFY(I).GT.0) IFACEY(I)=IFACEY(I)+IDIFFY(I)
      CFACT =  0.0111*(I-1)*(I-1) + 0.08*I + .12
      COST(I)=2.0 * LOW(IFACEX(I)+IFACEY(I)-2) * CFACT
      NSPACX(I)=IFACEX(I)-1
      NSPACY(I)=IFACEY(I)-1
      NFACEX(I)=IFACEX(I)
      NFACEY(I)=IFACEY(I)
      SPACEX(I)=(DX1-2*COV-NFACEX(I)*DB)/NSPACX(I)
      SPACEY(I)=(DY1-2*COV-NFACEY(I)*DB)/NSPACY(I)
      D=1.5*DB
      SP1(I) = 100.0 * BAR * 2 * IFACEX(I) / (B1*B2)
      SP2(I) = 100.0 * BAR * 2 * IFACEY(I) / (B1*B2)
      SP3(I) = 100.0 * BAR *(2*(IFACEX(I)+IFACEY(I))-4)/(B1*B2)
      IF(SPACEX(I).LT.D.OR.SPACEX(I).LT.40) GOTO 40
      IF(SPACEY(I).LT.D.OR.SPACEY(I).LT.40) GOTO 40
      IF(SP1(I).GT.8.0) GOTO 40
      IF(SP2(I).GT.8.0) GOTO 40
      IF(SP3(I).GT.8.0) GOTO 40
      IF(KSXY.EQ.0) GOTO 35
      IF(NFACEX(I).GT.NB1) GOTO 40
      IF(NFACEY(I).GT.NB2) GOTO 40
      IF(SPACEX(I).LT.SXY(1,1)) GOTO 40
      IF(SPACEY(I).LT.SXY(1,2)) GOTO 40
35    IF(COST(I).GT.TCOST) GOTO 50
      TCOST=COST(I)
      ICOST=I
      GOTO 50
40    COST(I)=10000
50    CONTINUE
      DO 51 J = 1,NT1
      I = NT1 + 1 - J
      IDB = 4 * (I + 2)
      PRINT 400,J,IDB,NFACEX(I),NFACEY(I),SP1(I),SP2(I),SP3(I),
400   FORMAT(2X,I2,4X,I2,4X,I2,2X,I2,3X,F6.2,2X,F6.2,2X,F6.2,3)
51    CONTINUE
      SKIP (2) LINES
      MESSAGE ↑ ** REJECTED SOLUTIONS ARE MARKED -R- IN THE COS
      SKIP(1) LINES
      MESSAGE ↑    SOLUTIONS ARE REJECTED IF THEY VIOLATE SPACI
```

```
                SKIP (5) LINES
                ICOS = NT1 + 1 - ICOST
                IF(ICOS .GT. NT1) GO TO 600
                PRINT 500,ICOS
        500     FORMAT(1X,*SOLUTION NUMBER *,I2,* HAS BEEN CHOSEN FOR DE1
                GO TO 601
        600     SKIP (2) LINES
                MESSAGE *** NO SATISFACTORY SOLUTIONS*
                RETURN
        601     IF(ICOST.NE.10000) GOTO 60
                IERR=IERR+1
                MESSAGE* COLUMN TOO SMALL TO SUIT DETAILING*
                PRINT 56
        56      FORMAT(*RUN ABORTED - COLUMN TOO SMALL TO SUIT DETAILING*
                RETURN
        **
        **       DETERMINE NUMBER OF WORST LOADING
        **       AND MAXIMUM STEEL PERCENT
        **
        60      SMAX = 0
                DO 70 I = 1,3
                IF(STEELP(I,1) .LT. SMAX) GO TO 70
                SMAX = STEELP(I,1)
                IND = I
                IWOR = INT(STEELP(I,2))
        70      CONTINUE
                IF(IWOR .EQ. 0) IWOR = 1
        **
        **   SET VALUES OF CLD()
        **
                EQUATE 80 (CLD(,IWOR),CLS)
                CLS(1) = 1
                CLS(9) = CLS(3) + CLS(2) * ECX / 1000
                CLS(10)= CLS(4) + CLS(2) * ECY / 1000
        80      CONTINUE
        **
        **       SET VALUES OF IBARS,TBLE,EK
        **                      ICNT,ZUN,ALP
        **
                ICNT = 1
                DO 120 I000 = 1,NT1
                NT = NT1 - I000 + 1
        **
        **       TEST IF A SOLUTION EXISTS
        **
                IF(COST(NT) .EQ. 10000) GO TO 120
        **
        **    A SOLUTION EXISTS THEREFORE INITIALIZE ALP
        **
                IF(NSYSB .EQ. 0) GO TO 90
                REDEFINE (ALP(),ICNT)
                ALP(ICNT) = 1.0
        **
        **    INITIALISE VALUES OF ZUN
        **
        90      REDEFINE(ZUN(),ICNT)
                PDES = PMIN + ( PMAX - PMIN ) * (SP3(NT) - 1) / 7.0
```

```
      ↑↑
      ↑↑       INITIALISE FK
      ↑↑
            REDEFINE(FK(,),ICNT),(FK(,ICNT),NSYS)
            EQUATE 100 (FK(,ICNT),FKS)
            DO 100 KK = 1,NSYS
            FKS(KK) = -1.0
      100   CONTINUE
      ↑↑
      ↑↑      DEFINE ARRAYS FOR RESULTS
      ↑↑
            REDEFINE (TBLE(,),ICNT),(IBARS(,),ICNT)
            REDEFINE (TBLE(,ICNT),9),(IBARS(,ICNT),14)
            EQUATE 120 (TBLE(,ICNT),TBL),(IBARS(,ICNT),IBL)
      ↑↑
      ↑↑    INITIALISE IBARS
      ↑↑
            DO 110 KK = 1,14
            IBL(KK) = ↑BL↑
      110   CONTINUE
            IBL( 8 - NT ) = IFACEX(NT) - 2
            IBL(15 - NT ) = IFACEY(NT) - 2
      ↑↑
      ↑↑    NOW SET TBLE
      ↑↑
            TBL(1) = SMAX
            TBL(2) = IND + .0001
            TBL(3) = CLD(3,IWOR)
            TBL(4) = CLD(4,IWOR)
            TBL(5) = CLD(2,IWOR)
            TBL(7) = NT
            TBL(9) = ICNT
      ↑↑
      ↑↑     CALCULATE SECTION CAPACITY FOR X DIR
      ↑↑
            IFLAG = 1
            PPP = CLD(6,IWOR)
            AM1 = FUNCP(PPP,7)
            AM2 = FUNCP(PPP,8)
            DIST = AM2 - AM1
            IPER = INT(SP1(NT)) + 1
            AMS = FUNCP(PPP,IPER)
            VAL = AMS - ( LOW(IPER) - SP1(NT) ) * DIST
      ↑↑
      ↑↑    VAL = M/BD**2 OF SECTION
      ↑↑
            TBL(6) = VAL * B1 * B2 * B2 * 1E-6
      ↑↑
      ↑↑    CALCULATE SECTION CAPACITY FOR Y-DIR
      ↑↑
            IFLAG = 2
            AM1 = FUNCP(PPP,7)
            AM2 = FUNCP(PPP,8)
            DIST = AM2 - AM1
            IPER = INT(SP2(NT)) + 1
            AMS = FUNCP(PPP,IPER)
            VAL = AMS - ( LOW(IPER) - SP2(NT)) * DIST
            TBL(8) = VAL * B1 * B1 * B2 * 1E-6
```

```
          ICNT = ICNT + 1
  120     CONTINUE
          RETURN
          END
```

```
*********************************************************************
      SUBROUTINE SORTL(LD3,ILD3,I1LD3)
      PUBLIC/CR/FCU
      DIMENSION LD3(.)
      REAL LD3,M,MT
      I1LD3=0
      ICOUNT=0
   10 ICOUNT=ICOUNT+1
      IF(ICOUNT.GT.ILD3) RETURN
      P=LD3(1,ICOUNT)
      M=LD3(2,ICOUNT)
      K=LD3(3,ICOUNT)
   20 IT=ILD3-1
      IF(ICOUNT.GT.IT) GOTO 40
      ICOUN1=ICOUNT+1
      IF(P.NE.LD3(1,ICOUN1)) GOTO 40
      IF(M.GT.LD3(2,ICOUN1)) GOTO 30
      M=LD3(2,ICOUN1)
      K=LD3(3,ICOUN1)
   30 ICOUNT=ICOUN1
      GOTO 20
   40 MT=P/2 - (P*P)/(1.19*FCU)
      IF(M.LE.MT) GOTO 10
      I1LD3=I1LD3+1
      LD3(1,I1LD3)=P
      LD3(2,I1LD3)=M
      LD3(3,I1LD3)=K
      GOTO 10
      END
```

```
      ****************************************************************
      SUBROUTINE RED(LD2,ILD2)
      PUBLIC/CP/B1,B2,COEFB(,),FX,FY,EL,FCU,FY,IFLAG,NUTRAL
      DIMENSION LD2(,)
      REAL LD2,L
**      MESSAGE * IN RED *
      SKIP(2) LINES
      MESSAGE* LOAD          FORCE          MOMENT    FACTOR*
      SKIP(1) LINE
      R=1.2 - 0.01*(EL/(0.3*AMIN1(B1,B2)))
      IF(IFLAG.EQ.1) R=1.2 - 0.01*(EY/(0.3*B2))
      IF(IFLAG.EQ.2) R=1.2 - 0.01*(EY/(0.3*B1))
      IF(R .GT. 1.) R = 1.
      DO 10 I=1,ILD2
      L=COEFB(1,IFLAG)*LD2(2,I)+COEFB(2,IFLAG)
      IF(LD2(1,I).LT.L) GOTO 20
      LD2(1,I)=LD2(1,I)/R
      LD2(2,I)=LD2(2,I)/R
      GOTO 30
20    NUTRAL=1
      T=36*LD2(1,I)
      IF(FCU.EQ.20) T=T/6.7
      IF(FCU.EQ.25) T=T/8.4
      IF(FCU.EQ.30) T=T/10.2
      IF(FCU.EQ.40) T=T/12.6
      IF(FCU.EQ.45) T=T/13.4
      PPB=T/29.5
      IF(FY.EQ.230) PPB=T/36
      IF(PPB.GT.1.) PPB=1
      R=1-(1-R)*PPB
      LD2(1,I)=LD2(1,I)/R
      LD2(2,I)=LD2(2,I)/R
30    III = INT(LD2(3,I))
      PRINT 100,III,LD2(1,I),LD2(2,I),R
100   FORMAT(1X,I3,4X,F10.2,3X,F10.2,F8.2)
10    CONTINUE
      RETURN
      END
```

```
COMPILE AS-COLUMN/1                                   1230   31/08/83   PAGE0017

      ********************************************************************
            SUBROUTINE CALCP(LD4,I,SP)
            PUBLIC/CR/IFLAG,PMAX,PMIN,FCU,SLOPE()
            DIMENSION LD4(,)
            REAL MHIGH,MLOW,LD4,M
            IPLOW=0
            P=LD4(1,I)
            M=LD4(2,I)
            IF(P.GT.PMAX) GO TO 50
            IF(P.EQ.PMAX.AND.M.GT.0) GO TO 50
            IF(P.LE.PMIN) GOTO 20
            IPLOW=INT(8*(P-PMIN)/(PMAX-PMIN))+1
            MHIGH=FUNCP(P,IPLOW)
            IF(MHIGH.LT.M) GOTO 40
            PLOW=PMIN +((IPLOW - 1)/8)*(PMAX-PMIN)
            MLOW=(P-PLOW)* SLOPE(IFLAG)
            IF(IPLOW.EQ.1) MLOW=(P-PMIN)*(0.5-PMIN/(.585*FCU))
 10         SP=IPLOW-(MHIGH-M)/(MHIGH-MLOW)
            RETURN
 20         MLOW=P/2-(P*P)/(1.17*FCU)
 30         IPLOW=IPLOW+1
            IF(IPLOW.GT.8) GOTO 50
            MHIGH=FUNCP(P,IPLOW)
            IF(M.LE.MHIGH) GOTO 10
 40         MLOW=MHIGH
            GOTO 30
 50         SP=9
            RETURN
            END
```

```
      ******************************************************************
            FUNCTION FUNCP(X,I)
            PUBLIC/CP/COEFP(,,),IFLAG
            MEDIUM COEFP
            S=COEFP(1,I,IFLAG)
            DO 10 K=2,7
            S=S + COEFP(K,I,IFLAG)*X**(K-1)
   10       CONTINUE
            FUNCP=S
            RETURN
            END
```

```
↑↑*****************************************************************
      SUBROUTINE SORT(S,N,LST,JL,LG,NG)
      DIMENSION S(,),LG(,),LST(,)
↑↑ THE SORT ROUTINE IS BUBBLE SORT ALGORITHM
↑↑
      IF(N .EQ. 1)RETURN
      FIX 100 S
      LL=N
↑↑ ELIMINATION OF DUPLICATE LOAD CASES
      JL=0
      KS=1
   30 KS1=KS+1
      KE=KS
      IF(KE .EQ. N)GOTO 61
      EQUATE 35 (S(,KS),S01)
      S1=S01(1)
      S2=S01(2)
      S3=S01(3)
 35   CONTINUE
      E=.01*S1
      DO 50 K=KS1,N
      IF(ABS(S1-S(1,K)) .GT. E)GOTO 60
   50 CONTINUE
      K=N+1
   60 KE=K-1
   61 JL=JL+1
      REDEFINE(LST(,),JL),(LST(,JL),2)
      LST(1,JL)=KS,KE
      IF(KS .GE. KE)GOTO 71
      DO 70 K=KS,KE
      EQUATE 65 (S(,K),S02)
      S1=S02(1)
      IF(S1 .LT. 0)GOTO 70
      S2=S02(2)
 65   S3=S02(3)
      K1=K+1
      IF(K1.GT.KE)GOTO 70
      DO 80 K2=K1,KE
      EQUATE 67 (S(,K2),S03)
      A1=S03(1)
      IF(A1 .LT. 0)GOTO 80
      A2=S03(2)
 67   A3=S03(3)
      Z1=A2-S2
      Z2=A3-S3
      IF(Z1*Z2)80,90,90
   90 IF(Z1.LE.0.AND. Z2.LE.0)GOTO 99
      IF(Z1.GT.0.0.AND.Z2.GT.0.0)GOTO 91
      GOTO 80
 91   S2=A2
      S3=A3
      GOTO 80
   99 S(1,K2)=-10
   80 CONTINUE
   70 CONTINUE
   71 KS=KE+1
      IF(KS .LE. N)GOTO 30
```

```
              N=JL
              DO 2 I=1,N
              K=2**I
              IF(K .GT. N)GOTO 3
          2   CONTINUE
          3   K=2**(I-1)-1
      ↑↑
      ↑↑SORTING LOADS INTO ASCENDING ORDER
          4   N1=N-K
              FIX 100 LST,LG
              DO 6 I=1,K
              IF(I .GT. N1)GOTO 7
              DO 6 J=I,N1,K
              I1=LST(1,J+K)
              I3=LST(2,J+K)
              EQUATE 12 (S(,I1),SQ4)
              S1=SQ4(1)
              S2=SQ4(2)
         12   S3=SQ4(3)
              DO 5 M=I,J,K
              L=J-M+I
              I2=LST(1,L)
              A1=S(1,I2)
              IF(A1 .GT. S1)GOTO 5
              SWAP (LST(,L+K),LST(,L))
              LST(1,L)=I1,I3
           5  CONTINUE
           6  CONTINUE
           7  K=K/2
              IF(K .GT. 0)GOTO 4
      ↑↑
      ↑↑FOR EACH COLUMN IN BATCH INDICATE THE HIGHEST AND LOWEST
      ↑↑    AXIAL LOAD CASES
              N=LL
              DO 9 I1=1,NG
              LG(1,I1)=0.0
            9 CONTINUE
              DO 8 LL=1,JL
              I1=LST(1,LL)
              I2=LST(2,LL)
              LENGTH(S(,I1),N1)
              DO 10 I4=4,N1
              I3=S(I4,I1)
              EQUATE 10 (LG(,I3),LG1)
              IF(LG1(1) .LT. 1)LG1(1)=LL
              LG1(2)=LL
           10 CONTINUE
          8   CONTINUE
      ↑↑
        100   CONTINUE
              RETURN
              END
```

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 5

PROGRAMMER'S REFERENCE MANUAL

FOR

REINFORCED RECTANGULAR CONCRETE COLUMN DESIGN MODULE

INDEX

1. INTRODUCTION ...168

2. ORGANIZATION OF THE MODULE ...168

2.1 Introduction ...168

2.2 Information supplied to the module ...169

2.3 Sign Convention ...171

2.4 Subroutine CENTRY ...171

2.5 " EFLTH ...175

2.6 " BCLASH ...179

2.7 " RTVP ...180

2.8 " DRIVER ...183

2.9 " SORT ...187

2.10 " COLLOAD ...190

2.11 " SORTL ...195

2.12 " RED ...197

2.13 " FINDP ...203

2.14 " CALCP ...206

2.15 Function FUNCP ...211

2.16 Subroutine ARRANG ...213

3. DESCRIPTION OF PUBLIC VARIABLES ...234

3.1 Introduction ...234

3.2 Variables in PUBLIC/COLUMN/ ...234

3.3 Variables in PUBLIC/RCSYS/ ...240

3.4 Variables in PUBLIC/CR/ ...246

3.5 Variables in PUBLIC/CR/ added for new module ...247

1. INTRODUCTION

This manual explains in detail the operation of a computer program module which designs rectangular concrete columns according to the requirements of the Australian Concrete Code, AS1480-1974.

The design method is based on that adopted in the A.R.C. Design Handbook and uses a digital representation of the charts therein to avoid the necessity for lengthy iterations to calculate the neutral axis depth.

Although this module is intended to replace one of similar function in the GENESYS column sub-system, it has been designed to operate in isolation if so desired and if minor modifications are made. Several of the original subroutines have been retained mainly to facilitate integration of the replacement module. These subroutines were altered slightly to suit the new design method and they are explained along with the new subroutines.

2. ORGANISATION OF THE MODULE

2.1 Introduction

The module has been organised to take advantage of the facility offered by the GENESYS system which allows a program to be split into OVERLAYs, only one of which resides in central memory at any one time. The program efficiency may be increased by the effective use of this facility.

The module contains an extensive table of coefficient values and it was considered practical to break the module into two OVERLAYs, the coefficients and related subroutines in one, and the main design programme in the other.

The OVERLAY names and numbers which have been adopted are the same as those used in the original module and the contents of each are as follows:

OVERLAY 52 "RC22"	Routine	CENTRY
	"	EFLTH
	"	BCLASH
	"	RTVP
OVERLAY 54 "RC24"	Routine	DRIVER
	"	SORT
	"	COLLOAD
	"	RED
	"	SORTL
	"	FINDP
	"	CALCP
	"	FUNCP
	"	ARRANG

The basic module organisation is shown in the flow diagram for the subroutine CENTRY which is the entry point for the module. The subroutine calling sequence is shown in Figure 2.1.

2.2 Information supplied to the module

The module is supplied with the following information passed through PUBLIC variables.

- Column dimensions and lengths

- Properties of beams/slabs which join the column

- Material properties

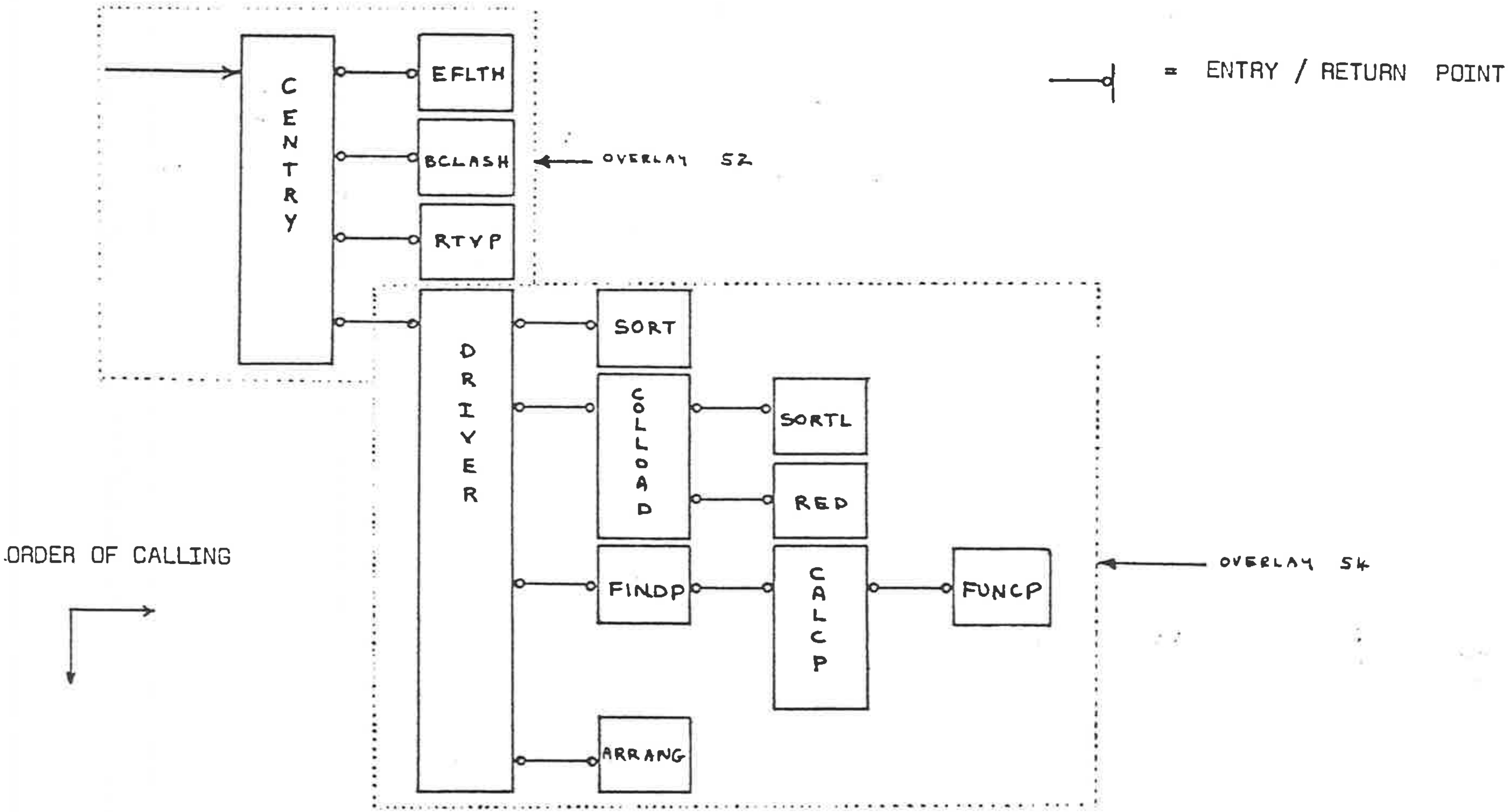


FIGURE 2.1 - Subroutine Calling Sequence of Replacement Module

- A loading array containing all of the loading combinations that can be applied to the column. Each loading consists of an axial load and head and foot moments if they exist.

From this information the module determines the cheapest steel arrangement (if any exist) which is capable of withstanding the worst loading combination.

2.3 Sign Convention: See Figure 3.3 in Thesis main body.

2.4 Subroutine Centry: Column Design entry

2.4.1 Duty Specification

This routine is the entry and exit point of the rectangular shaped column group under consideration as called by the control module. It controls the design sequence by calling the various utility subroutines in their correct order and contains the PUBLIC variables required to link with the rest of the main program.

2.4.2 Controlling Parameters

None

2.4.3 Procedure and Justification

		Subroutine
(a)	Enter Column Design Module	CENTRY
(b)	Retrieve Column Dimensions and cover from Interface	
(c)	Compute the Effective Length of the column group.	EFLTH
(d)	If required then consider beam bar clashing.	BCLASH
(e)	Combine head and foot moments as specified in Section 2.2 depending on column slenderness.	
(f)	Retrieve material strengths associated with the current column group.	
(g)	Retrieve Coefficient Tables determined by column dimensions and properties.	RTVP
(h)	PERFORM DRIVER (new OVERLAY).	DRIVER
(i)	EXIT	

2.4.4 PUBLIC Storage (for details see Section 3)

```
PUBLIC/COLUMNS/CSP(,),P(,),ST(),MS(),IGL(,),IP(,),KOLUMN
PUBLIC/RCSYS/NSYS,ISYS
PUBLIC/CR/B1,B2,FCU,FY,XSL,YSL,PP(,),IWARN,IERR,KSXY,COV,NOWR(),NOER(),
        NT1
```

2.4.5 Local Storage

N	Number of load cases in the run
MM	Starting point for load cases to be considered (upper and lower lifts)
IJ	Slenderness indicator
I	Control Variable
INC	" "
I8	" "
X1	Head moment in X-direction
X2	Foot " "
Y1	Head " Y-direction
Y2	Foot " "
I2	Control Variable
I3	" "
I4	" "
I5	" "
N10	Number of columns in group the load cases applies to
N11	Control Variable
I1	" "
IK	Number of load considered
I6	Control Variable
I7	" "
X	Moment in X-direction for slender column
Y	Moment in Y-direction for slender column
J19	Control Variable
X1X	Used in efffective moment calculation for slender columns
Y1Y	" " " " " " " "
XX	" " " " " " " "
YY	" " " " " " " "

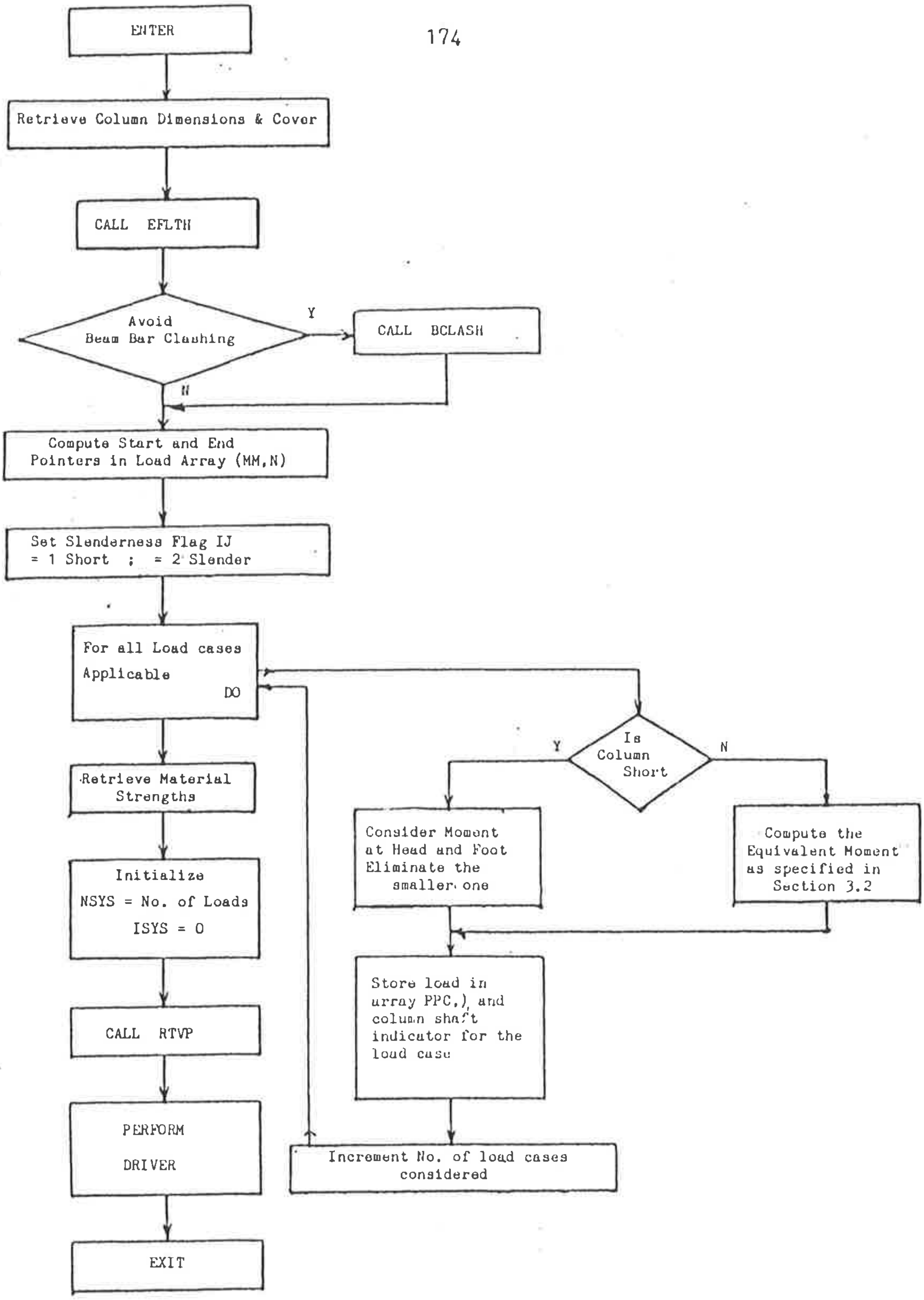


FIGURE A2.4.1 - Flow Chart - CENTRY

2.5 Subroutine EFLTH: Effective Length calculation

2.5.1 Duty Specification

This routine calculates the effective length of the column group under consideration. It uses the method given in CP110.

2.5.2 Controlling parameters

Column dimensions and length.

2.5.3 Procedure and Justification

The procedure for calculating the effective length has been retained as the one specified in CP110. The AS1480 method while being similar to that used by CP110, is not compatible with the rest of the GENESYS program in its present form. Information necessary for the AS1480 method is not available via the current interface. Future modifications to the interface should include provisions for the information necessary for the AS1480 method.

The procedure is as follows:

(a) Retrieve the various column/beam stiffness ratios

(b) Calculate effective lengths as shown page 39 CP110, part 1

(c) Calculate slenderness ratios

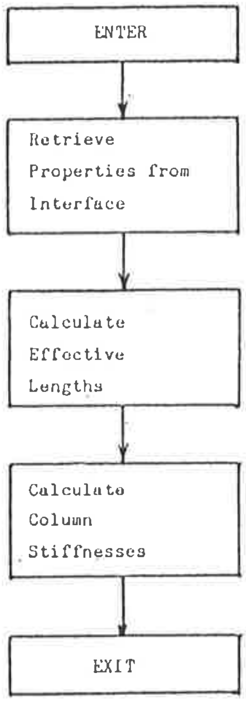


FIGURE A2.5.1 - Flow Chart - EFLTH

2.5.4 Public Storage (For details see Section 3)

```
PUBLIC/COLUMNS/KOLUMN,GL(),MS()

PUBLIC/CR/EL,EX,EY,XSL,YSL,B1,B2
```

2.5.5 Local Storage

I	Control Variable
J	" "
I1	" "
I2	" "
X1	Ratio of column/beam/flat slab stiffness about X-X axis
X2	" " " " " " " " Y-Y "
O1	Overall length in X-direction
O2	" " " Y-direction
K	Bracing condition of the column
X	MIN (X1 & X2)
Y	MIN (Y1 & Y2)
E1	Effective length
E2	Effective length

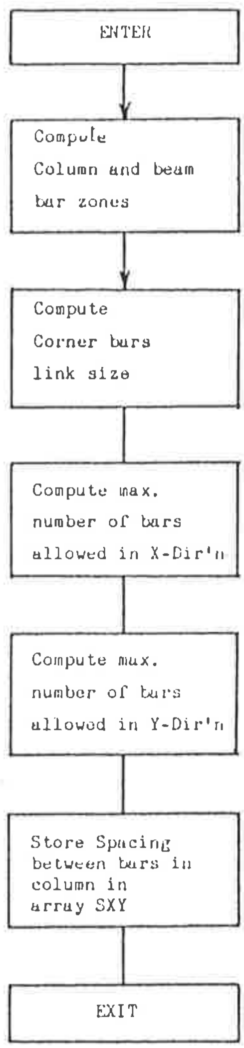


FIGURE A2.6.1 - Flow Chart - BCLASH

2.6 Subroutine BCLASH: Bar clashing avoidance

2.6.1 Duty Specification

This routine establishes the permitted number of bars in each face of the column if beam bar clashing is to be avoided. It returns with the number of bars permitted to avoid beam bar clashing.

2.6.2 Controlling Parameters

Various information regarding the beam steel present.

2.6.3 Public Storage (for details see Section 3)

PUBLIC/COLUMNS/MS(),ST()

PUBLIC/CR/B1,B2,SXY(,),NB1,NB2,KSXY

2.6.4 Local Storage

CB	Maximum corner bar size
BB	Maximum beam bar size
CZ	Column zone
EB	Effective beam zone
U15	Maximum of beam and column zones
X	Control Variable
D15	Control Variable
L	Link Size
C	Tolerance

2.7 Subroutine RTVP: Coefficient retrieval

2.7.1 Duty Specification

This routine sets the value of the coefficients to be used to represent the charts. Only one chart is to be stored for each of: Biaxial, Monoaxial-X, Monoaxial-Y. For purposes ofcalculating g, 24mmØ bars are assumed. For Biaxial cases g is based on the smaller of the lateral dimensions. Note that the ligature size does not affect g because cover specified is to main bars.

2.7.2 Controlling Parameters

(a) FY

(b) FCU

(c) g value interpolation

2.7.3 Procedure and Justification

G value interpolation is allowed in the A.R.C. Design Handbook (Ref 15). First check that the calculated g value is permissible ($.6 < g < 1$ and if $g > .9$ set $g = .9$) then determine between which two charts the g value lies. Assume a straight line relation between the two charts thus calculating their relative contributions. This is represented by two factors (one for each chart) lying between 0 and 1. As the equations in each chart are of the same order, a single coefficient set may be generated by summing the corresponding coefficients multiplied by the appropriate factor.

For example: Assume g=G and lying between .6 and .7

Then G(.7) = (g-.6)/.1 and G(.6) = 1 - G(.7)

MD(.6) = ((Coefficient(.6)*(PD))

MD(.7) =((Coefficient(.7)*(PD))

Thus adding the two

MD(g) = G(.6)*((Coefficient(.6))*(PD) +

G(.7)*((Coefficient(.7))*(PD)

This results in a single set of coefficients generated by summation with (PD) as the common factor.

2.7.4 Program Layout

The program contains the logical statements followed by 6 blocks of the 8 charts groups with the same physical properties. These blocks are accessed only once when the charts, with the same physical properties, are brought in for temporary storage. The first 4 charts in the group are the monoaxial cases and the second four are the biaxial cases. In addition PMAX and PMIN which are common to all charts with the same physical properties are also stored. Note that PMIN = 0.595 * FCU.

2.7.5 Public Storage (For details see Section 3)

```
PUBLIC/CR/B1,B2,COEFB(,),COEFP(,,),FCU,FY,IERR,NOER(),PMAX,PMIN,SLOPE()
```

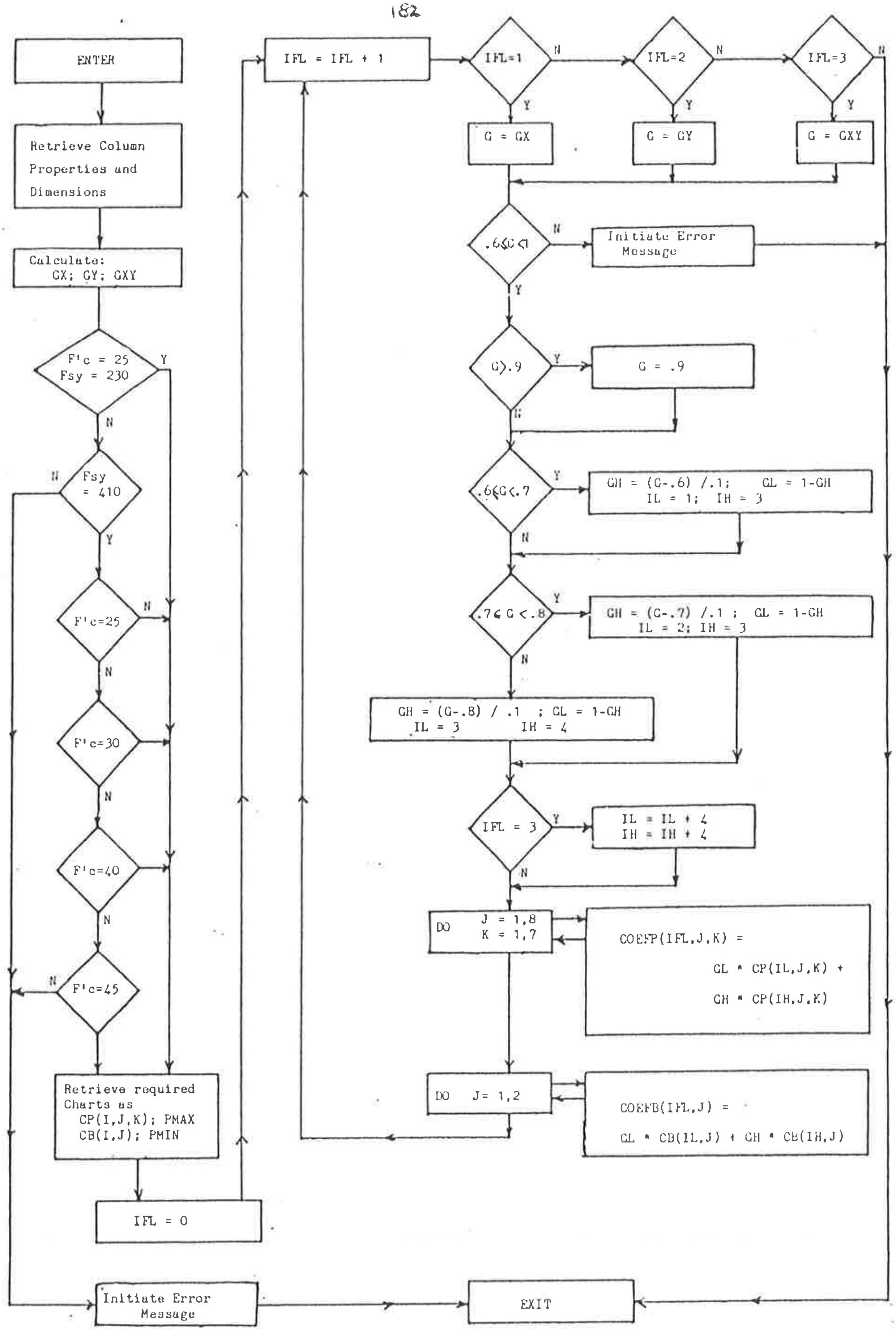


FIGURE A2.7.1 - Flow Chart - RTVP

2.7.6 Local Storage

CP(I,J,K)	Temporary storage for equation coefficients
	I = 1,8 Chart numbers
	J = 1,8 Steel %
	K = 1,7 Coefficient Numbers
CB(I,J)	Temporary storage for balanced failure equation
	I = 1,8 Chart numbers
	J = 1,2 Coefficients
G,GX,GY,GXY	g values
IL	Low chart number for g
IH	High chart number for g
GL	Factor for IL
GH	Factor for IH
IFL	Pointer to type case
	= 1 Biaxial
	= 2 Monoaxial-X
	= 3 Monoaxial-Y
CS(I)	Temporary storage for slope of curves as they cross M=0 axis
	I= 1,8 Chart numbers

2.8 Subroutine DRIVER: Driving the design sequence

2.8.1 Duty Specification

This is the driving routine that controls the entire design sequence. It is entered initially from CENTRY and, when complete, it initiates the detailing routine if requested.

2.8.2 Controlling Parameters

None

2.8.3 Procedure and Justification

(a) Initialise Public variables used in the analysis routine

(b) Store appropriate physical data in array PCLD for use by O/P routines

(c) Create the arrays for sorting the load cases and analysis results. This involves copying the contents of PP(,) to SLD(,) and creating the array CLD(,).

(d) Change the units of the loading to Newtons and Millimeters

(e) Sort loads in order of descending axial load (Call SORT)

(f) Create monoaxial and biaxial loading cases (Call COLLOAD)

(g) Calculate the required steel percentages in each face (CALL FINDP)

(h) Arrange the steel percentages found with various bar sizes (CALL ARRANG)

(i) Initiate detailing if required

(j) Exit

2.8.4 Public Storage (For details see Section 3)

PUBLIC/COLUMNS/P(,),GL(),ST(),MS(),IGL(,),KOLUMN

PUBLIC/RCSYS/NSYS,ISYS,IISYS,SLD(,)

PUBLIC/CR/B1,B2,FCU,FY,EX,EY,ICNT,INIT,XSL,YSL,AA(,),DIAM(),PCLD(),
CLD(,),NOER(),NOWR(),PP(,),IWARN,IERR,FIVE

2.8.5 Local Storage

ISH	Short or slender column
SECA	Area of column cross-section
I,J	Control Variable
AJ	Bar area
KPC	Control Variable
IX1	" "
NCZ	" "
LLK	" "
KS5	" "
KF5	" "
SLDTEMP(,)	Temporary storage for array SLD(,) while the loads are put in their correct order

2.8.6 Subroutine/Functions accessed

SORT(SLD,N,LST,LLK,GRP,NGR)

COLLOAD

FINDP

ARRANG

CROUT (in OVERLAY "Main Column Design")

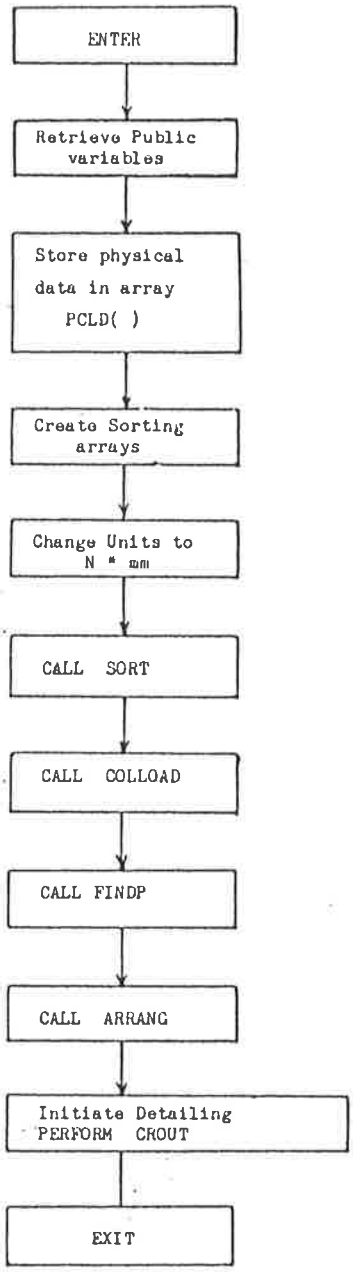


FIGURE A2.8.1 - Flow Chart - DRIVER

2.9 Subroutine SORT: Initial load sort

2.9.1 Duty Specification

This routine performs two tasks:

(a) Eliminate the unnecessary load cases in the set of loads applicable to the column group.

(b) Sort the load cases in descending order of the axial load applied.

The load cases which have been eliminated from the interface array SLD(,) have been given an axial load less than 0. The sorting algorithm is the usual Bubble Sort.

2.9.2 Controlling Parameters

None

2.9.3 Procedure and Justification

Bubble sort algorithm

2.9.4 Public Storage

None

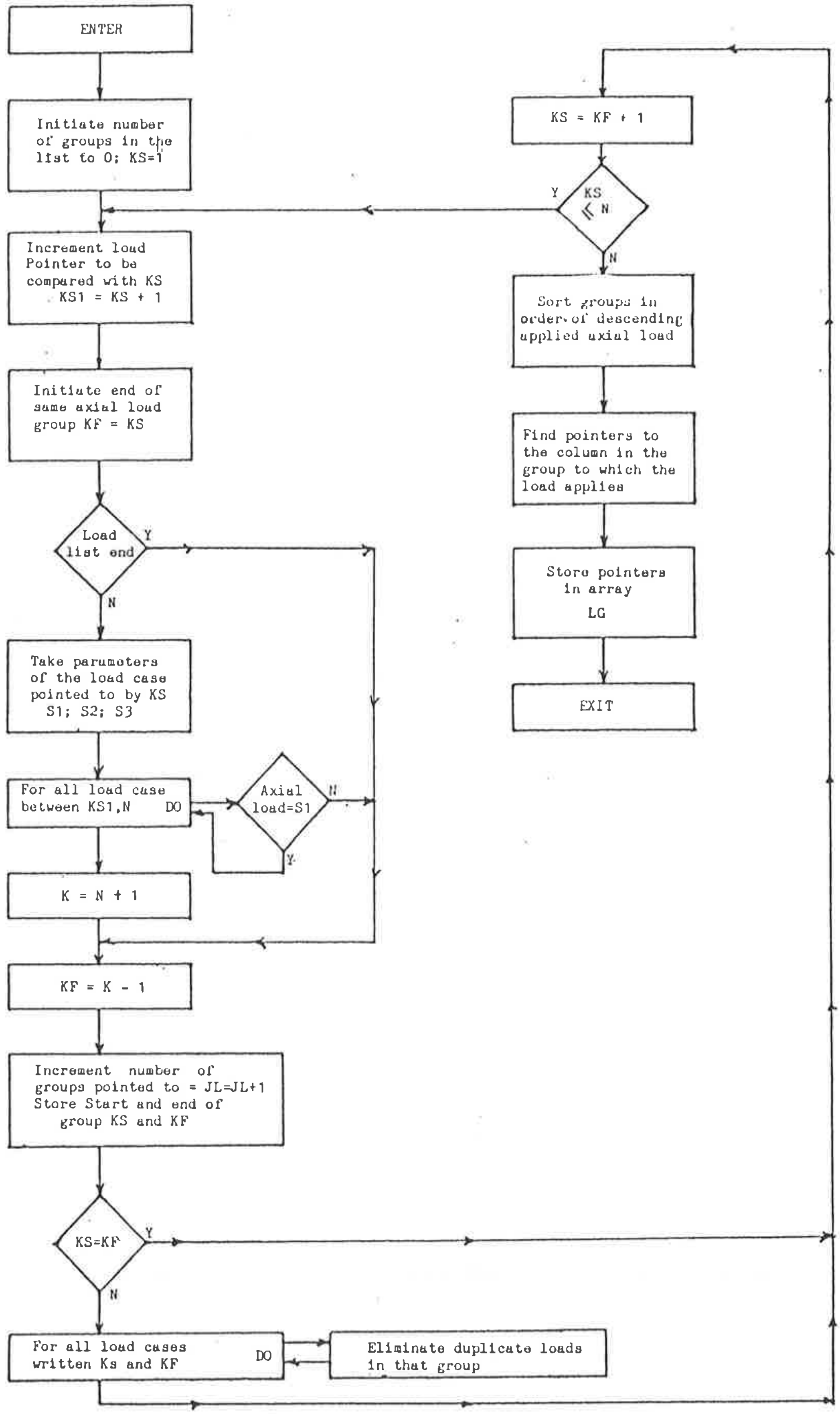


FIGURE A2.9.1 - Flow Chart - SORT

2.9.5 Local Storage (See Section 3)

LL	Holds intermediate results
KS	Elimination loop control variable
KS1	" " " "
S1	Axial load value
S2	Moment in X-direction
S3	Moment in Y-direction
K	Control Variable
K1	" "
K2	" "
A1	As S1 for compared load case
A2	" S2 " " " "
A3	" S3 " " " "
Z1	Comparison control variable
Z2	" " "
I	Sorting control variable
N1	" " "
J	" " "
I1	" " "
I3	Control Variable
M	" "
L	" "
I2	" "
LL	" "
I4	" "

2.9.6 Argument Variables

S	Array containing the unsorted load cases
N	Number of load cases
LST	Returns with pointers referring to entries in array S which indicates the sorted loads order
JL	Number of entries in the array LST
LG	Array containing pointers to the column in the group that the load case applies to
NG	Number of columns in the group

2.10 Subroutine COLLOAD: Load Conditioning

2.10.1 Duty Specification

This routine compiles the loading set to be used in the column design with account taken of the following:

(a) Initial eccentricities

(b) Reduction factors

(c) Removal if required steel % = 0

(d) Removal if same axial force but smaller moment

(e) Creation of biaxial loading cases

2.10.2 Controlling Parameters

None

2.10.3 Procedure and Justification

(a) Type MONX and MONY : REAL, because they may contain fractions.

(b) Calculate eccentricities ECX and ECY from CSP(I,J) by taking the value of the current lift or lift above depending which is the larger (in mm). (J = 3,4)

(c) Retrieve the set of loading cases which is held in SLD(I,J) SLD(I,J):

I = 1 Axial load

= 2 X-Moment

= 3 Y-Moment

J = Load case number

(d) Convert loading table into form $P/BD, M/DB^2$

$$P' = \frac{P * 10^3}{B1 * B2} \; ; \; M'_x = \frac{MX}{B1 * B2^2} \; ; \; M'_y = \frac{MY}{B1^2 * B2}$$

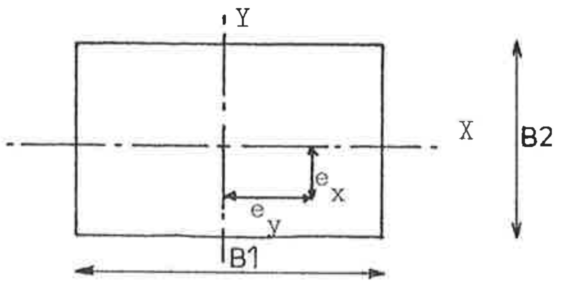


Figure 2.10.1 - Definition of Eccentricities

(e) Calculate design eccentricities as the sum of the initial eccentricities and the actual eccentricities. First determine the actual eccentricities which are = the maximum of the current lift and the lift above:

EX1 = AMAX(CSP(1,4),CSP(2,4))

EX2 = AMAX(CSP(1,3),CSP(2,3))

Then:

ECX = 25 + .01*B2 + EX1 (mm)

ECY = 25 + .01*B1 + EY1 (mm)

ECXY = AMAX(ECX,ECY) (mm)

(f) Split into three groups:

- MONX(I,J),I=1,2,3 $\quad$ $P', M'_x + P' * ECX$, IL
- MONY(I,J),I=1,2,3 $\quad$ $P', M'_y + P' * ECY$, IL
- BIAX(I,J),I=1,2,3 $\quad$ $P'(M'_x + M'_y + P' * ECXY)$,IL

The biaxial case is only formed if both M'_x .03 * P' * B2 (BX)
and M'_y .03 * P' * B1 (BY)

J = The load case number.

IL is the load number in SLD(,) from which the loading is derived.

(g) Apply SUBROUTINE SORTL to MONX,MONY,BIAX which will discard redundant loadings if same P but smaller M and if the loading will require a steel % equal to zero.

(h) Apply SUBROUTINE RED to MONX,MONY,BIAX which will apply the reduction factors R & R' and set NUTRAL = 1 if any tension governing cases are found.

(i) In all cases the load number in SLD(,) from which the load case was derived is stored with the loading case.

(j) EXIT.

2.10.4 Public Storage (see Section 3)

PUBLIC/COLUMNS/ CSP(,)

PUBLIC/RCSYS/NSYS,SLD(,)

PUBLIC/CR/B1,B2,BIAX(,),IBIAX,IFLAG,IMON,MONX(,),MONY(,),NSYSX,

NSYSY,NSYSB,ECX,ECY

2.10.5 <u>Local Storage</u>

I	Counting variable
BX	)
	) Minimum moments for biaxial case
BY	)
ECXY	Maximum of ECX and ECY
SL1	= P/BD
SL2	$= M_x / B1 \times B2^2$
SL3	$= M_y / B1^2 \times B2$

2.10.6 <u>Subroutines/Functions accessed</u>

SORTL(LD3(,),ILD3,I1LD3)

RED(LD2(,),ILD2)

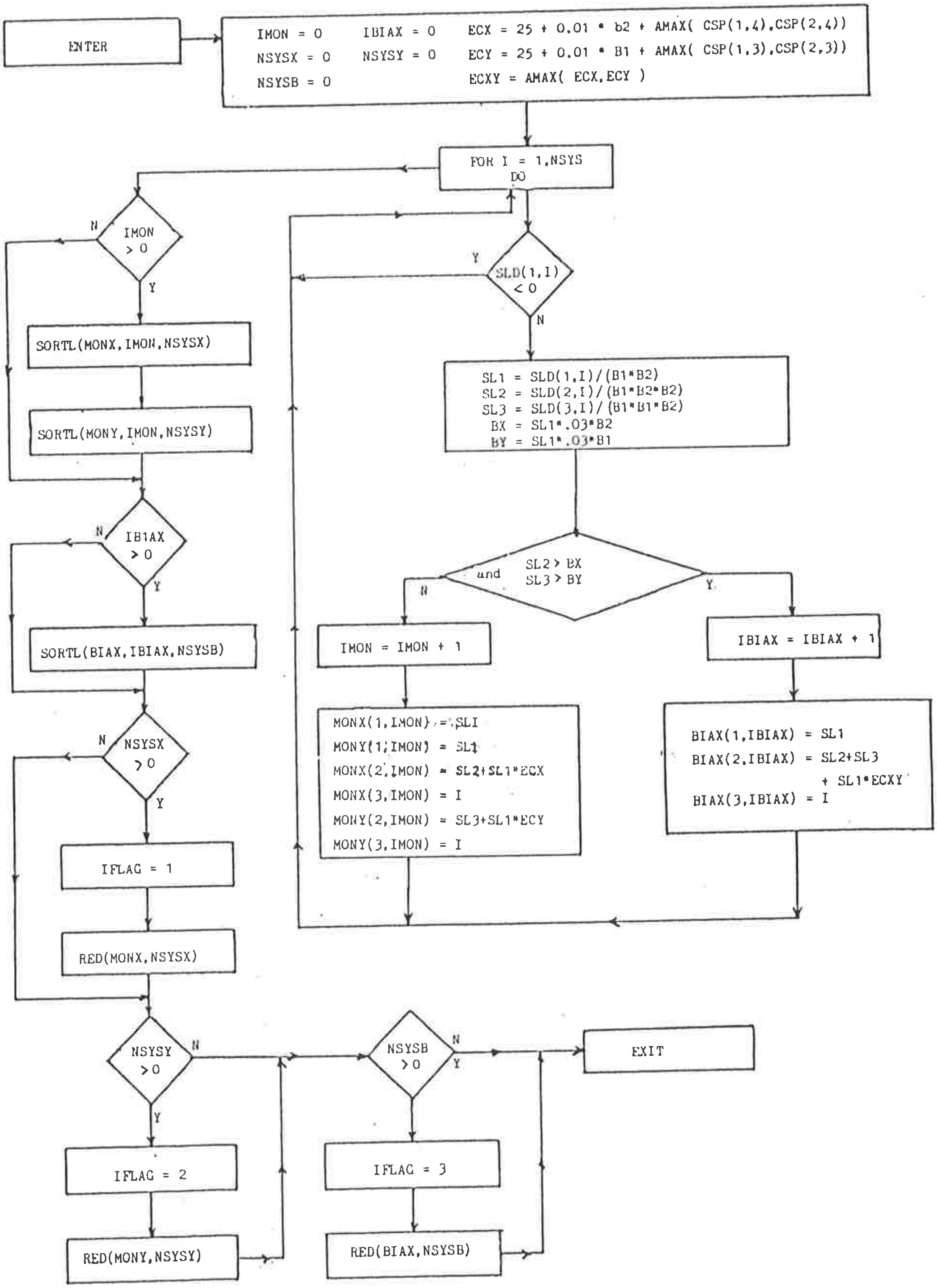


FIGURE A2.10.2 - Flow Chart - COLLOAD

2.11 Subroutine SORTL(LD3,ILD3,I1LD3): Loading Sort

2.11.1 Duty Specification

Sort through each group of loadings and remove those for which:

(a) Same Axial force but smaller Moment,

(b) The required steel % = 0.

2.11.2 Procedure and Justification

First specify TYPE statement for REAL variables. The loadings have already been sorted according to decreasing axial load. The program first sorts through loadings with equal P values and finds the highest moment. The program then stores the loading if the required steel percent is greater than zero according to the formula derived in Section 3.2.6. It is possible that no loads will be found which require a steel percentage greater than zero in which case NSYS will be zero.

2.11.3 Public Storage: (See Section 3)

PUBLIC/CR/FCU

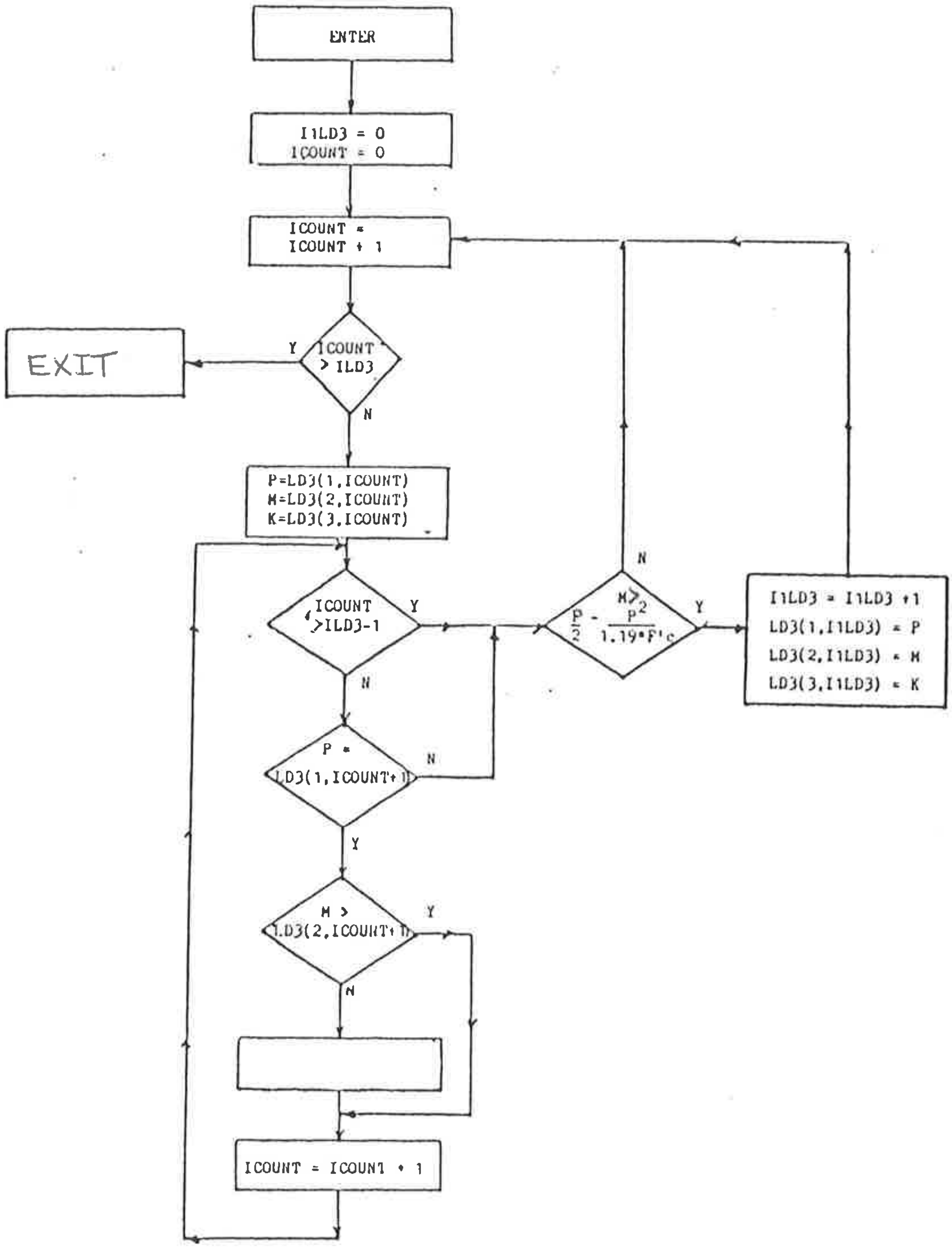


FIGURE A2.11.1 - Flow Chart - SORTL

2.11.4 Local Storage

MT - Moment capacity with no steel present

M - Current Moment Value

P - Current Axial Force Value

ICOUNT - Counting variable

K - Load case number in SLD(,) from which the current load was derived

IT - (=ILD3-1) Load number test variable

2.11.5 Argument Variables

LD3 Input/Output load array name, set by COLLOAD to either MONX,MONY, BIAX on entering

ILD3 Number of loads to consider, set by COLLOAD to either IMON or IBIAX on entering

I1LD3 Number of loads remaining after sort, set by COLLOAD to either NSYSX, NSYSY, or NSYSB on leaving routine.

2.12 Subroutine RED(LD2,ILD2): Reduction Factor Routine

2.12.1 Duty Specification

Determine and apply the load reduction factors to the loading specified.

2.12.2 Controlling Parameters

The operation of the subroutine is controlled by:

- The loading type (MONX,MONY or BIAX)
- The number of loading cases (NSYSX, NSYSY or NSYSB)
- The values of F'C and FSY

2.12.3 Procedure and Justification

The procedure is based on the graphs appearing in the ARC Design Handbook (Ref 15). As the balanced failure line depends on the loading type, the subroutine COLLOAD will set IFLAG = 1,2,3 depending on the loading being considered. The routine is skipped if the number of reduced loadings = zero.

(a) Dimension the dummy array LD2 using dummy array length ILD2. (they appear in the argument list)

(b) Specify the TYPE of the REAL variables.

(c) Retrieve the balanced load line coefficients according to the load type being examined.

(d) Check whether the loading lies below the balanced failure line, and if so, set NUTRAL = 1.

(e) Determine the value of T (an intermediate variable used to calculate the reduction ratios) as follows:

From the graphs in the ARC Handbook the following may be determined using similar triangles:

For F'c = 20, T = 36/6.7 * P/BD

For F'c = 25, T = 36/8.4 * P/BD

For F'c = 30, T = 36/10.9 * P/BD

For F'c = 40, T = 36/12.6 * P/BD

For F'c = 45, T = 36/13.4 * P/BD

(f) Determine the value of PPB (if tension failure occuring (PPB<1)):

For FSY = 230, PPB = T/36

= 410, PPB = T/29.5

(g) Apply the reduction factor:

For Compression Failure R = 1.2 - .01 * 1/r

Tension " R' = 1-(1-R) * PPB

allowing a maximum value of PPB = 1

for Monoaxial-X 1 = EX (by EFLTH)

r = .3 * B2

" Y 1 = EY (by EFLTH)

r = .3 * B1

Biaxial 1 = EL (by EFLTH)

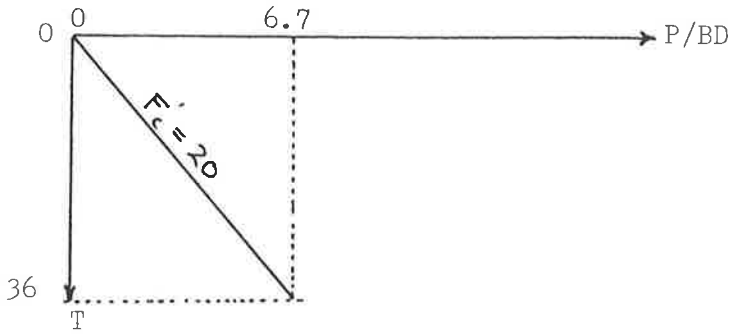
r = .3 * AMIN(B2,B1)

(h) EXIT

2.12.4 Derivation of Relationships

The derivation of these relationships is illustrated below (for F'c = 20 and Fsy = 230) and is valid for all lines on chart C4 (ref 15).

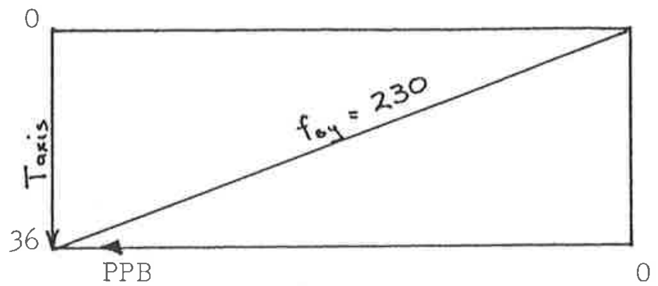
Stage 1 - determination of T



T = P/BD * 36/6.7

Figure 2.12.1 - Determination of T

Stage 2 - determination of PPB



PPB = T/36

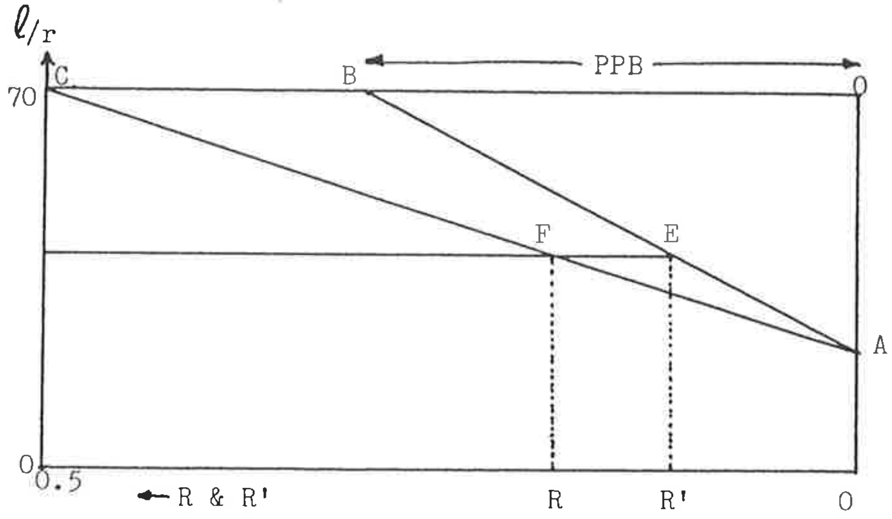
Figure 2.12.2 - Determination of PPB

Stage 3 - determination of R

If compression failure governs then from AS1480:

R = 1.2 - .01 * 1/r

Stage 4 - determination of R'



(Note that R' = R if PPB = 1)

Figure 2.12.3 - Determination of R'

By similar triangles $\frac{CB}{CA} = \frac{FE}{FA}$ i.e. $\frac{1 - PPB}{1} = \frac{R' - R}{1 - R}$

And so (1-R) * (1 - PPB) = R' - R

1 - R - PPB + PPB * R = R' - R

R' = 1 - PPB + PPB * R

R' = 1 - (1 - R) * PPB

2.12.5 Public Storage (See Section 3)

PUBLIC/CR/ B1,B2,COEFB(,),EX,EY,EL,FCU,FY,IFLAG,NUTRAL

2.12.6 Local Storage

T - Used to calculate PPB accounting for FCU and FY

PPB - is the symbol for Pu/Pb'

R - Compression failure reduction factor

RD - Tension failure reduction factor

I - Counting variable for DO loop

L - Balanced Failure axial load (N/mm^2) for applied moment

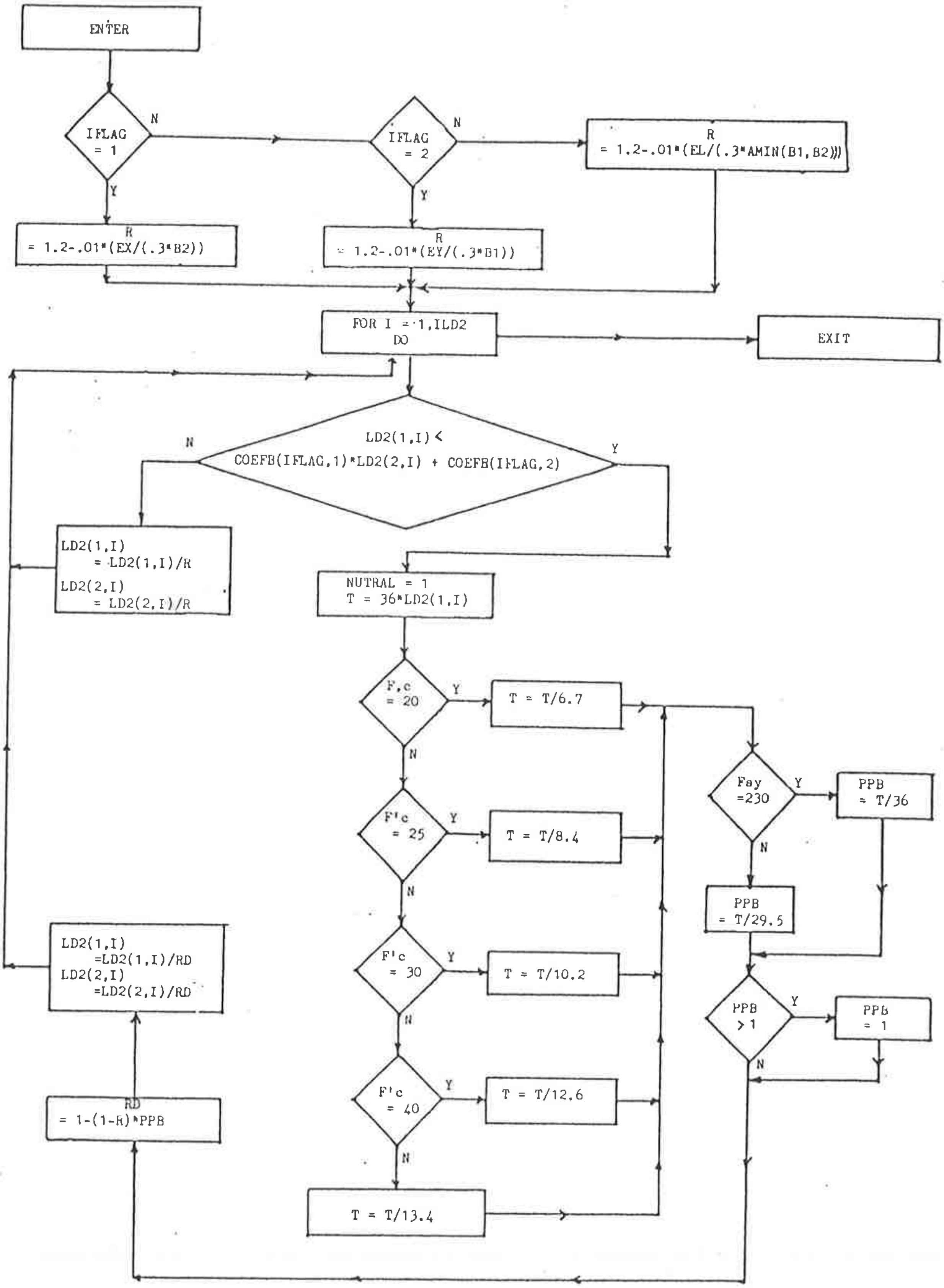


FIGURE 2.12.4 - Flow Chart - RED

2.12.7 Argument Variables

LD2(J,I), J = 1,2 = P,M: Input/Output load name which will be either MONX,MONY or BIAX.

ILD2: Number of loadings in LD2 - will be set to either NSYSX,NSYSY or NSYSB by COLLOAD.

2.13 Subroutine FINDP: Calculate maximum steel % required

2.13.1 Duty Specification

Search through the reduced loadings for the maximum steel % required. Store the load numbers in STEELP from which the maximum steel % was derived.

2.13.2 Procedure and Justification

(a) Check that the number of loadings is greater than zero.

(b) For each load type:
- . assume steel % = 0
- . search through all existing reduced loadings and find the maximum steel % required

(c) If the resulting steel % is greater than 8 then an error condition exists (column too small)

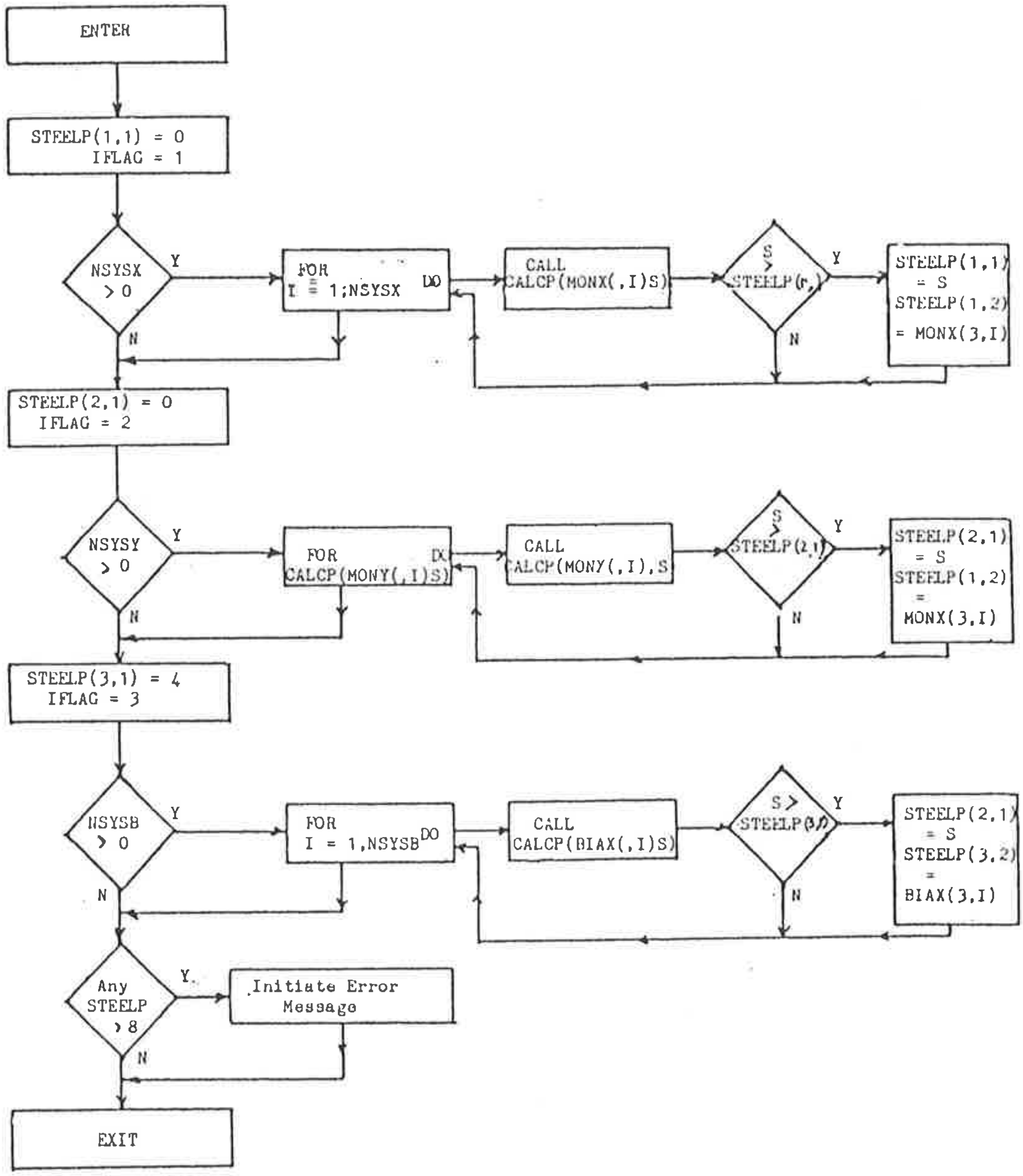


FIGURE 2.13.1 - Flow Chart - FINDP

(d) Store the steel % and corresponding load case numbers in array STEELP(,).

(e) Exit

2.13.3 Public Storage: (See Section 3)

PUBLIC/CR/ MONX(,),MONY(,),BIAX(,),NSYSX,NSYSY,NSYSB,STEELP(,),
IFLAG,NOER(),IERR

2.13.4 Local Storage

I - Counting Variable

S - Temporary storage for steel % returned from CALCP (Exists in CALCP as SP)

2.13.5 Argument Variables

None

2.13.6 Functions/Subroutines accessed

CALCP(LD4(,I),SP)

2.14 Subroutine CALCP(LD4(,I),SP): Steel % Calculation

2.14.1 Duty Specification

Given P & M and the design chart, find the steel percentage required.

2.14.2 Procedure and Justification

The procedure is explained with reference to Figure 2.14.1.

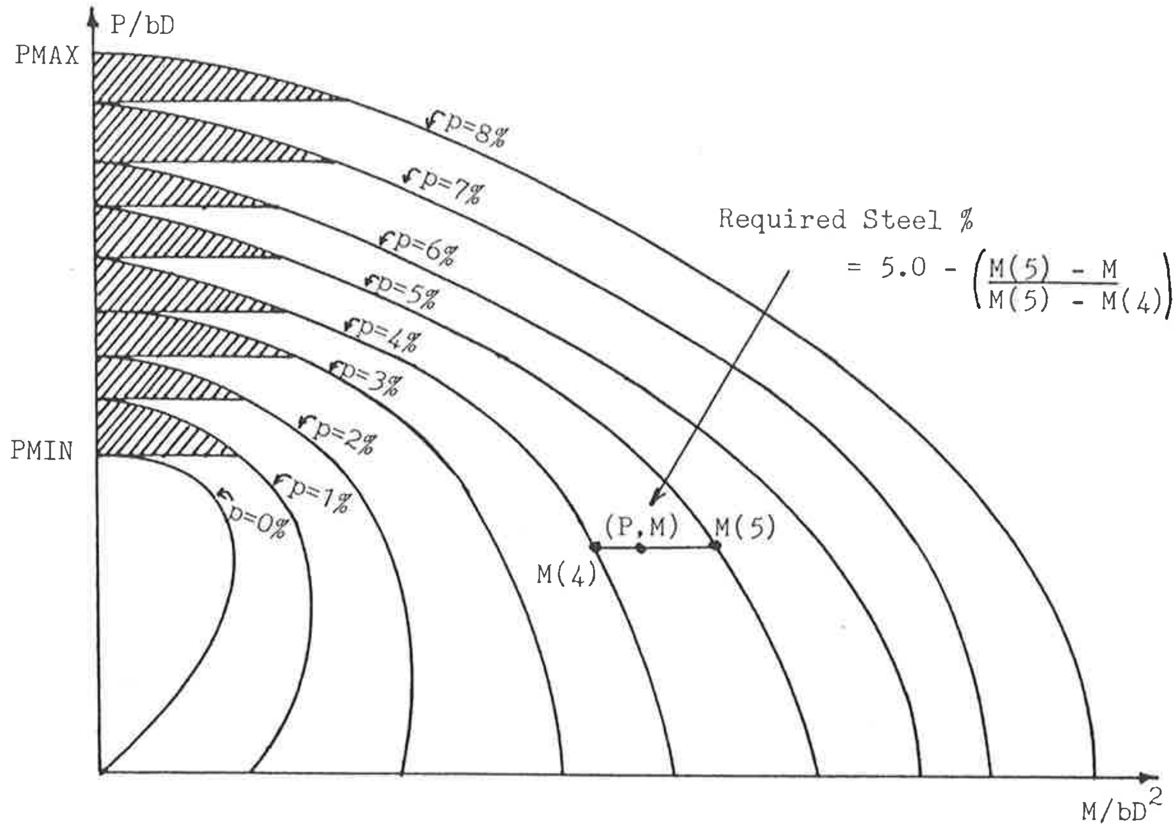


Figure 2.14.1 - Typical Set of Failure Curves

(a) Enter the subroutine and set IPLOW=0 and P & M to the current loading condition. Retrieve PMAX and PMIN from PUBLIC storage.

(b) Test whether P lies within the range PMAX to PMIN. If P is greater than PMAX then SP is set =9 and the subroutine is skipped. One of three possible actions is then taken depending on the value of P (see c), d) and e) following).

(c) If P lies below PMIN (see Fig 2.4.1) then a single iterative solution is done to determine the %. Starting with %=0, the % is increased until the moment capacity provided is greater than the applied moment. The required steel % is then calculated as shown in Figure 2.14.1.

(d) If P lies above PMIN and outside the shaded areas shown in Figure 2.14.1 then the simple iteration procedure is used, starting with the steel % just underneath the shaded area which contains the P value with M=0. By doing this, it is not necessary to test for the lower steel percentages if they are obviously too low.

(e) If P lies within a shaded area then the slope of the lower % line is used to calculate the value of MLOW. This is because the equations are unreliable after the curves cross the M=0 line. Two possibilities then occur:

(1) If the value of IPLOW is equal to 1 then the slope of the 0% line must be used.

Given $M=P/2 - P^2/1.19F'c$ (Section 3.11.3)

$$\left(\frac{dM}{dP}\right) = 1/2 - PMIN/.595F'c \text{ (at } P=PMIN)$$

At P=PMIN=0.595 F'c, $\frac{dM}{dP} = 1/2 - 1 = -0.5$

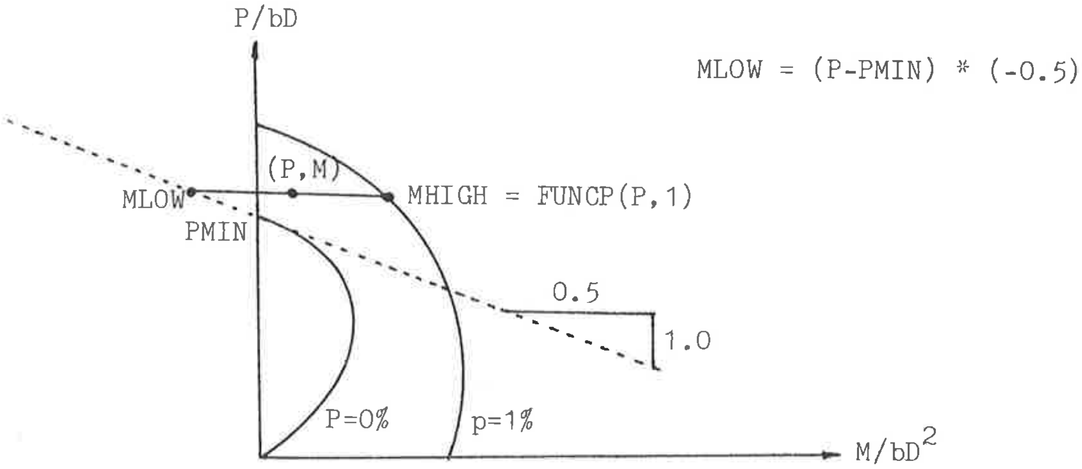


Figure 2.14.2 - Value of IPLOW equal to 1

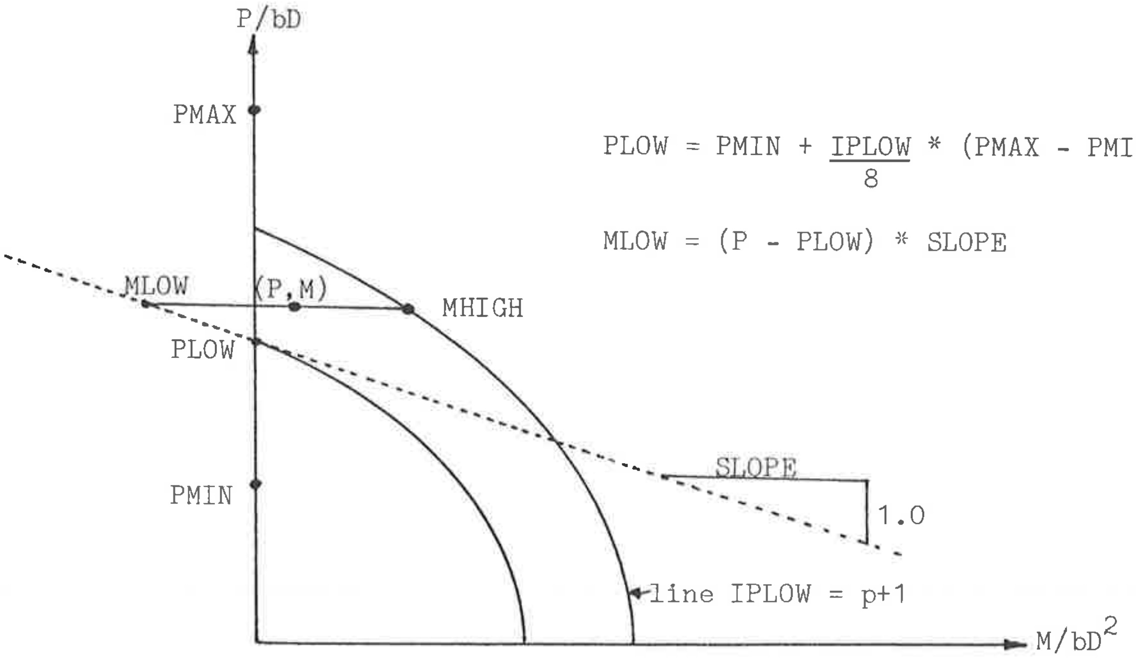


Figure 2.14.3 - Value of IPLOW greater than 1

(2) If the value of IPLOW is greater than 1 then the slope of the next lowest steel % must be used. Note that the slope is calculated from coefficients in subroutine RTVP. (See Figure 2.14.3)

$$PLOW = PMIN + \frac{IPLOW}{8} (PMAX - PMIN)$$ (= position where p=0 line crosses P/bD axis)

From this MLOW = (P-PLOW)* SLOPE

(f) If it is found that the required steel % is greater than 8 then SP is set =9. This is an error condition but it is detected later in the program.

2.14.3 <u>Public Storage</u>: (See Section 3)

PUBLIC/CR/FCU,IFLAG,PMIN,PMAX

2.14.4 <u>Local Storage</u>

IPLOW - Current integer value of the steel being considered

P)
)
M) Temporary storage for stress values
)
MHIGH)
)
MLOW)

PLOW P value where IPLOW line crosses the M=0 axis.

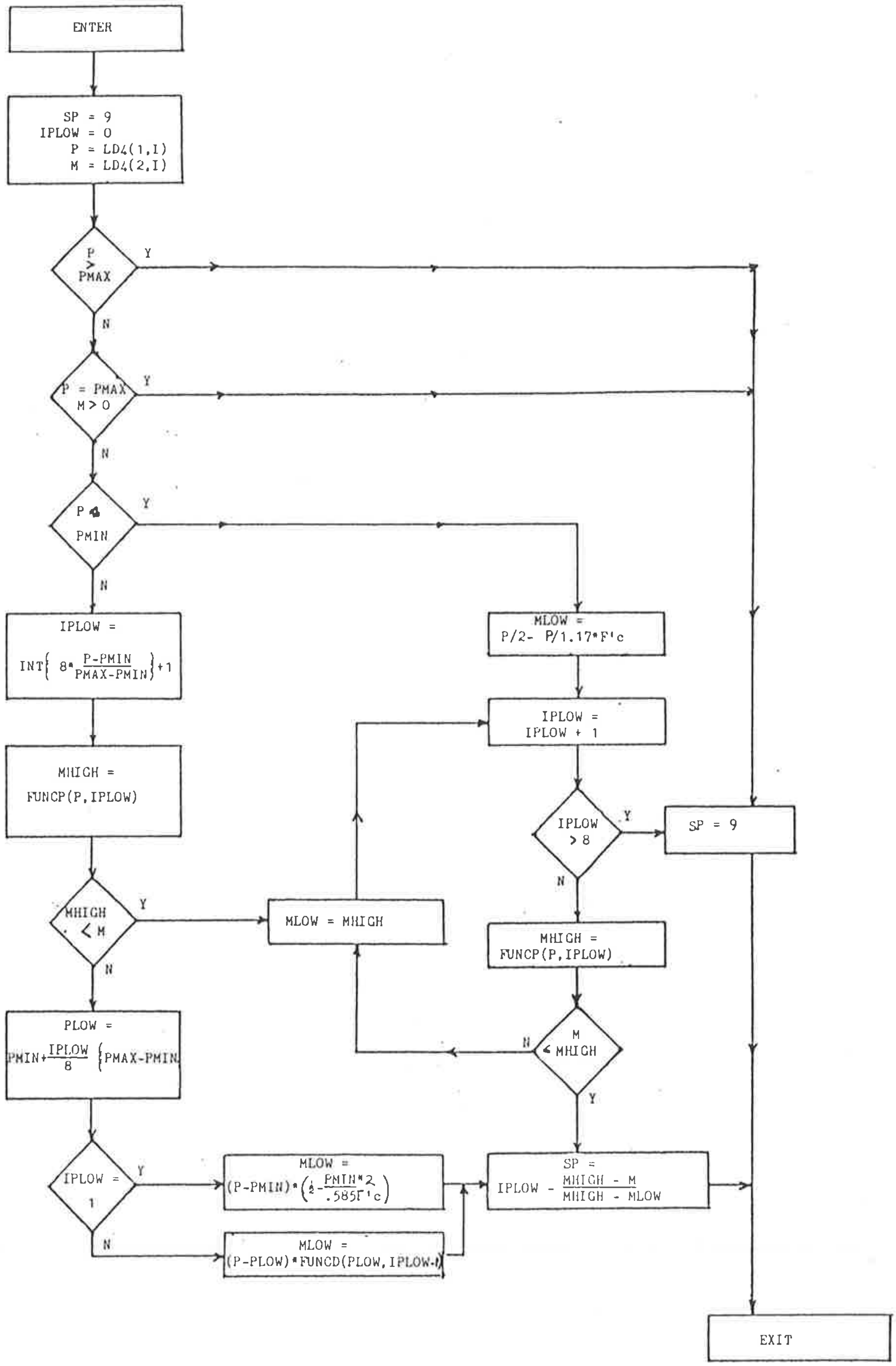


FIGURE 2.14.4 - Flow Chart - CALCP

2.14.5 Argument Variables

SP - Contains the value of steel % required. (set to =S by FINDP)

LD4(I,J) - Contains the coordinate load values being considered

I=1,2 -P,M

J - Load number being considered

2.14.6 Subroutines/Functions accessed

FUNCP(P,IPLOW)

2.15 Function FUNCP(X,I): moment capacity determination

2.15.1 Duty Specification

Find the appropriate moment capacity given the axial force existing.

2.15.2 Procedure and Justification

The function uses the coefficients stored by RTVP. Each steel % line has been stored in the form:

$$Y = A + Bx + Cx^2 + Dx^3 + Ex^4 + Fx^5 + Bx^6$$

Where: A = COEFP(IFLAG,%,1)
B = " 2)
C = " 3)
D = " 4)
E = " 5)
F = " 6)
G = " 7)

Y = Moment X = Axial Force

It is therefore only necessary to do a repetitive summation and multiplication.

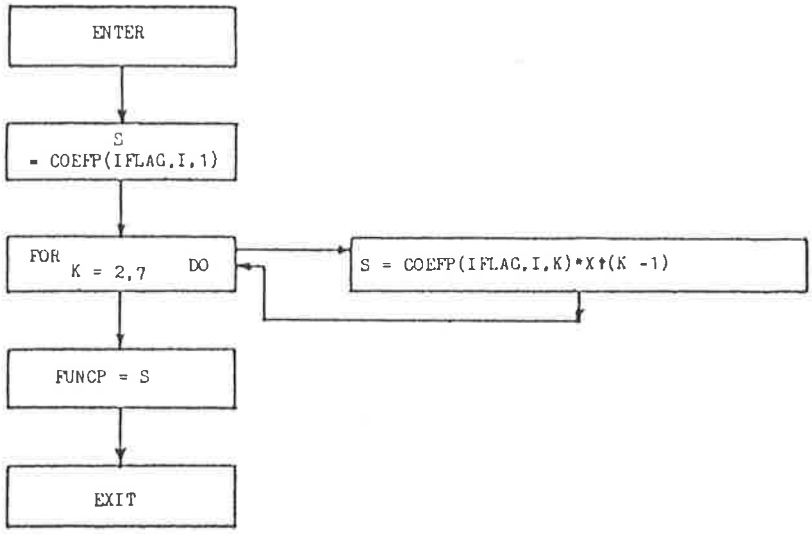


FIGURE 2.15.1 - Flow Chart - FUNCP

2.15.3 Public Storage: (See Section 3)

PUBLIC/CR/COEFP(,,),IFLAG

2.15.4 Local Storage

S - Summation Variable

K - Iteration Variable

2.15.5 Argument Variables

X - Contains value of axial load

I - Steel % being considered

2.16 Subroutine ARRANG: Steel arrangement

2.16.1 Duty Specification

Determine the most economical arrangement of steel bars which satisfies the steel percentage requirements of the analysis routines. A description of column bar bundling is included although the GENESYS system does not at present permit it.

2.16.2 Controlling Parameters

Steel % - PXY (STEELP(3,1))
- PX (STEELP(1,1))
- PY (STEELP(2,1))

Stipulate that only bars of the same diameter are to be used.

2.16.3 Procedure and Justification

(a) Retrieve the steel percentages to be achieved and perform the following tests:

(i) If any of (PX , PY , PXY) are greater than 8 then EXIT.

(ii) If all of (PX , PY , PXY) are less than 1 then set PX or PY =1 depending on which has the larger dimension. This will lead to the largest bar spacing.

(b) Calculate the Steel areas to be provided:

Mono-X AST(1) = STEELP(1,1) * B1 * B2/100

Mono-Y AST(2) = STEELP(2,1) * B1 * B2/100

Biaxial AST(3) = STEELP(3,1) * B1 * B2/100

(c) Allow the following bar sizes:

12,16,20,24,28,32,36 mm

(d) Calculate the number of bars required for biaxial bending with the following criteria for each bar size:

(i) The number of bars must be an integer and a multiple of 4

(ii) Do not allow 8 bars (because the charts do not apply) and instead use 12

(iii) At least 4 corner bars are put in.

This leads to the following statements:

```
NBIBAR(1)= 4 x INT((AST(3)/110) x 1/4 +1) ... 12mm
NBIBAR(2)= 4 x INT((AST(3)/200) x 1/4 +1) ... 16mm
NBIBAR(3)= 4 x INT((AST(3)/310) x 1/4 +1) ... 20mm
NBIBAR(4)= 4 x INT((AST(3)/450) x 1/4 +1) ... 24mm
.
.
.
etc...
```

Where 110,200,310,etc.. are the single bar areas and NBIBAR refers to the number of 12,16,20,etc... bars required.

(e) Determine the effect of the adjacent face bars by finding an equivalent number of bars which would be equal in capacity if placed at the same distance from the centreline as the face bars. This produces the number EFFBAR() which is the same for X & Y directions.

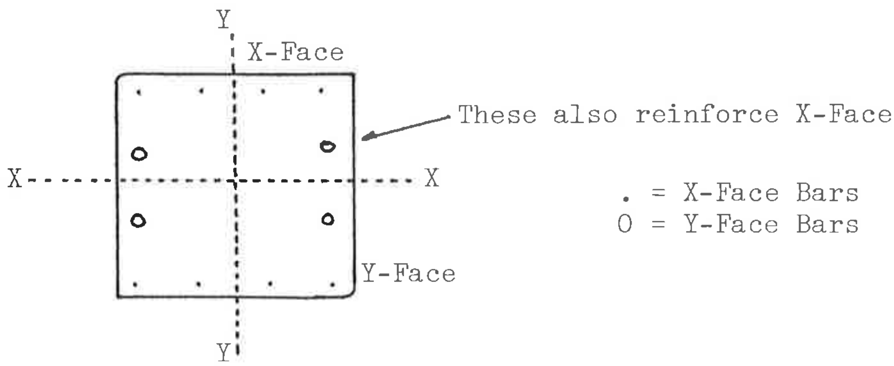


Figure 2.16.1 - Adjacent Face Bars

These bars are also reinforcing the X-face by a reduced amount depending on their positions relative to the X-face bars.

The steel area required in the X face is thus reduced by presence of the Y-face bars.

The effectiveness of the adjacent face bars is given by the following relationships.

(i) If NBIBAR() = 4 then no allowance can be made. (Note that a similar situation could have arisen in the case of NBIBAR() = 8 because the adjacent face bar would be on the centreline. However 8 bars are not permitted.

(ii) The number of face bars is (initially) given by:

$$\left.\begin{array}{l}\text{IFACEX()}\\ \text{IFACEY()}\end{array}\right\} = \frac{\text{NBIBAR()} + 1}{4}$$

For example:

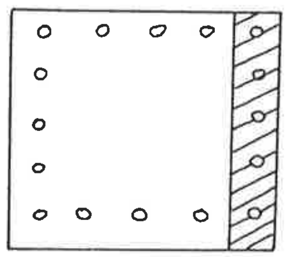


Figure 2.16.2 - Face Bars

NBIBAR() = 16

IFACEX() = 5

IFACEY() = 5

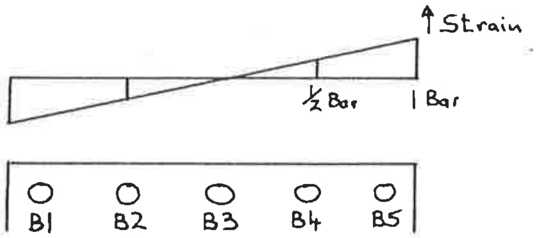
Note also that for the purposes of calculating PX & PY face bars on both faces must be considered.

X bars provided = 2 x IFACEX()

Y barsprovided = 2 x IFACEY()

(iii) If IFACEX() is odd then the following applies:

Assume a linear strain distribution.

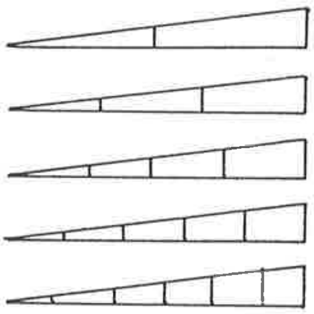


IFACEX() = 5

Figure 2.16.3 - IFACEX ( ) is odd

In the above case, B1 and B5 are already considered as X-face bars, B3 is ineffective and both B2 and B4 have the effect of 1/2 bar.

This may be followed through a number of bars:



IFACEX()=5 N = 1/2 = 0.5

IFACEX()=7 N = 1/3 + 2/3 = 1.0

IFACEX()=9 N = 1/4 + 2/4 + 3/4 = 1.5

IFACEX()=11 N = 1/5 + 2/5 + 3/5 + 4/5 = 2.0

IFACEX()=13 N = 1/6 + 2/6 + 3/6 + 4/6 + 5/6 = 2.5

The relationship is obviously linear but note that the total contribution is 4 x N because there are +ve and -ve bars on two faces (symmetrical about centreline).

Therefore the contribution of the side face bars is given as follows:

IFACEX() = 3 - Not permitted

IFACEX() = 5 - 4 x .5 = 2

IFACEX() = 7 - 4 x 1.0 = 4

IFACEX() = 9 - 4 x 1.5 = 6

IFACEX() = 11 - 4 x 2.0 = 8

IFACEX() = 13 - 4 x 2.5 = 10

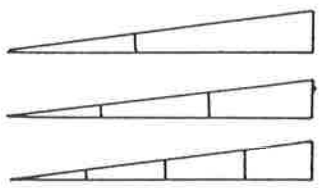
Therefore if IFACEX() is ODD EFFBAR() = IFACEX() - 3

Note that at this stage IFACEX() = IFACEY().

(iv) If IFACEX() is even then the following applies:

If IFACEX() = 2 then set EFFBAR() = 0 because the corner bars have already been considered.

Following the same method as in (iii) the following may be derived:



IFACEX()=4 N = 1/3 = 0.33

IFACEX()=6 N = 1/5 + 3/5 = 0.8

IFACEX()=8 N = 1/7 + 3/7 + 3/7 = 1.29

Similarly if IFACEX() = 10, N = 1/9 + 3/9 + 5/9 + 7/9 = 1.78

IFACEX() = 12, N = 1/11 + 3/11 + 5/11 + 7/11 + 9/11

= 2.27

IFACEX() = 4 - 4 x .33 = 1.33

IFACEX() = 6 - 4 x .80 = 3.20

IFACEX() = 8 - 4 x 1.29 = 5.14

IFACEX() = 10 - 4 x 1.78 = 7.11

IFACEX() = 12 - 4 x 2.27= 9.09

This plots as a straight line and using the usual method of finding the equation of a line then:

If IFACEX() is EVEN EFFBAR() = 0.97 x IFACEX() - 2.55

(f) Calculate the number of additional X & Y face bars required to satisfy PX & PY accounting for the bars already present for the biaxial case as follows:

Given:

IFACEX()

= number offace bars present

IFACEY()

EFFBAR() = effective number of side face bars

+1 = factor to ensure that if the number of bars required is between two integers then the higher of the two is chosen.

AS= 110,200,310 = single bar areas

AST(1),AST(2) = required monoaxial steel areas

Note that EFFBAR() is multiplied and then divided by the bar area because EFFBAR() could be fractional.

The additional bars required in each face are given as follows:

IDIFFX() = INT(1/2 * ((AST(1) - EFFBAR()*AS)/AS))+1 - IFACEX()

IDIFFY() = INT(1/2 * ((AST(2) - EFFBAR()*AS)/AS))+1 - IFACEY()

Then provided IDIFFX() and IDIFFY() are positive i.e. if the biaxial steel provided is insufficient to cater for the monoaxial loads then:

<u>IFACEX() = IFACEX() + IDIFFX()</u>

<u>IFACEY() = IFACEY() + IDIFFY()</u>

Note that (initially) IFACEX() =IFACEY() and that this is not true only if bar bundling is to be considered.

(G) Calculate the cost of each of the seven solutions. The method of costing the solutions is based on the procedure adopted by the Estimating Section, Design Services Branch, Engineering and Water Supply Department (E & WS).

For estimating purposes the E & WS use the following rates for the supply, bending and fixing of reinforcement in concrete columns (as at July, 1979):

\$650/tonne - large bars

\$700/tonne - small bars

It was decided to use these figures to find the relative cost per bar of the different sized bars on a linear scale with 12 mm representing the small bar and 36mm the large bar.

To further simplify the relation between the bars it was decided to assume 100 metre lengths of bar.

Therefore if N = number of bars

A = area of one bar (mm^2)

S = density of steel = 7.7 tonnes/m^3

Weight of N bars $W = 7.7 * 100 * A * 10^{-6} * N$ tonnes

$= 7.7 * 10^{-4} * A * N$ tonnes

therefore for W = 1 tonne

$$N = \frac{1}{7.7 \times 10^{-4} \times A}$$ = number of bars in one tonne @ 100m long.

This leads to the following table:

Bar mm	Area mm2	Bars/tonne	Cost/tonne \$	Cost/bar \$	Relative Cost/bar \$
36	1020	1.273	650	510	1.00
32	800	1.623	658	405	.79
28	620	2.095	667	318	.62
24	450	2.886	675	233	.46
20	310	4.189	683	163	.32
16	200	6.494	692	107	.21
12	110	11.806	700	59	.12

Table 2.16.1 Relative Cost Per Bar

If this is plotted on a graph it appears as follows:

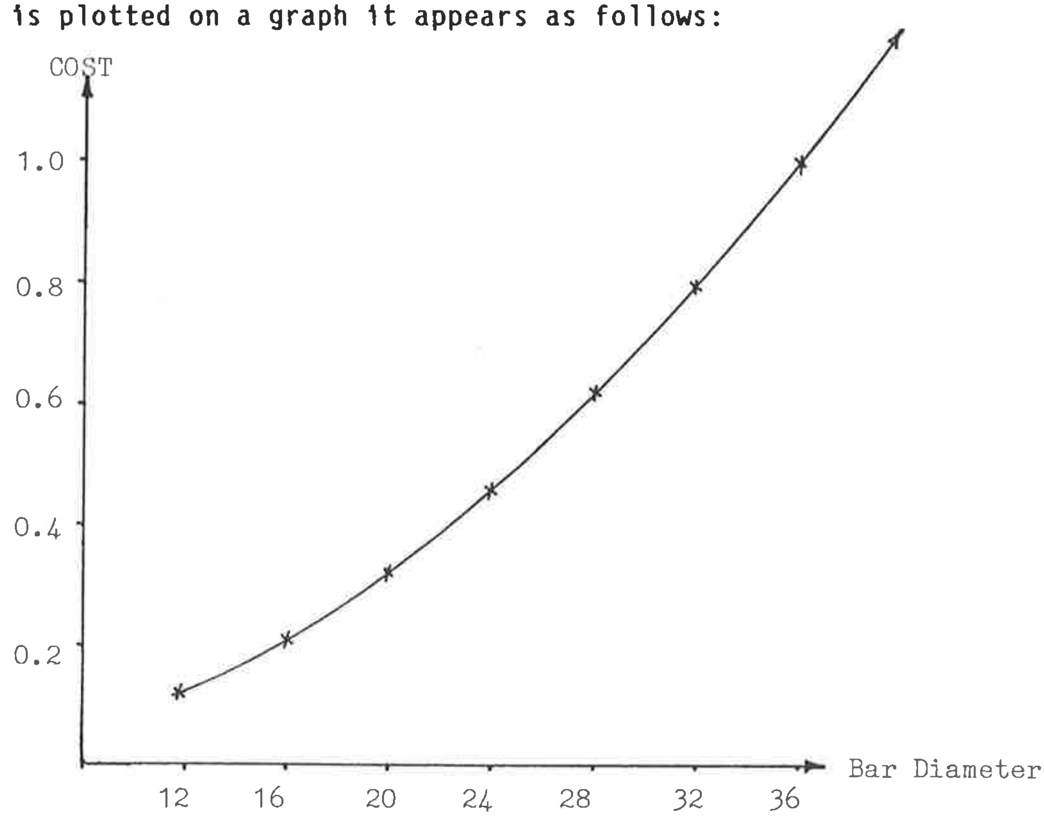


Figure 2.16.4 Cost Vs Bar Diameter

The graph looks parabolic so assume that:

$$COST = Ax^2 + Bx + C, \quad x = \frac{\text{bar diameter}}{4} - 3$$

Solving the quadratic equation gives:

$$Y = .0111x^2 + .08X + .12$$

If a resubstitution is made then the following results arises:

X	F(X)	Y
0	.12	.12
1	.21	.21
2	.32	.32
3	.46	.46
4	.62	.62
5	.80	.79
6	1.00	1.00

Thus it would appear that the equation is adequate.

Therefore given that 0 = 12mm bar= counting variable 1

1 = 16mm bar = counting variable 2

$$COST = .0111*(I-1)^2 + .08*(I-1) + .12$$

The total number of bars in a particular solution is

= 2*(IFACEX() + IFACEY())-4

Therefore COST(1) = (2*(IFACEX(1) + IFACEY(1))-4)*f(1)

COST(2) = (2*(IFACEX(2) + IFACEY(2))-4)*f(2)

Where $f(I) = .0111*(I-1)^2 + .08*(I-1) + .12$

(h) Calculate the spacings in each face and bundle the bars if the spacing is less than 40mm or 1.5(bar diameter). Note that the facility to bundle column bars has not been included in the replacement module because the rest of the system does not allow it. However a method of doing it has been included in case the rest of the system is altered to permit bundled column bars.

If bundling is permitted it should conform to the following:

(i) Bundling is not permitted for bar diameters less than 20mm (i.e. for bars 12mm and 16mm)

(ii) Only corner bars are to be bundled. This greatly simplifies the stirrup requirements and also the setting out of the solution. This is considered reasonable because the necessity to bundle bars usually indicates inadequate column dimensions.

(iii) The maximum number of bars per bundle is 4. Because both the X & Y faces may need to have their bars bundled, the total number of bars per bundle is the sum of those required independently for the X & Y faces. This figure must not be greater than 4.

(iv) An unsatisfactory solution will be indicated by setting the COST of the solution to a very high number such as 10,000. This represents 10,000 of 36mm bars and is unlikely to occur in practice.

(v) The method of solution is as follows:

I. Assume NBUNDX() and NBUNDY() = 0

II. Consider X face and calculate SPACEX(). If SPACEX() is not within limits then increment NBUNDX() (to a maximum of 3) and recalculate SPACEX(). If SPACEX() cannot be made within limits then set COST() = 10,000 and procede to the next bar size.

III. Consider Y face as in (II)

IV. Test if this is the first time through - i.e. if IFLAG = 0.

V. If IFLAG = 0 then:

if NBUNDX() is 1 set DY = DY - 2 * DB*
and IFLAG = 1 and goto step 1

if NBUNDY() is 1 set DX = DX - 2 * DB*
and IFLAG = 1 and goto step 1

(This is necessary because if more than 1 bar is bundled then the spacing in the other face is affected)

VI. Test if NBUNDX() + NBUNDY() is >3 and if so set COST() = 10,000. This ensures that the total number of bundled bars is not greater than 3 - i.e. 4 bars per corner

VII. Proceed to the next bar size.

* DB = bar diameter.

(vi) The calculation of SPACEX() and SPACEY() is as follows:

I. the number of bars adjacent to the face is given as:

<u>NFACEX() = IFACEX() - 2*(NBUNDX() -1)</u>

(see following table)

II. the number of clear spaces between bars is given as

<u>NSPACEX() = (IFACEX() - (2*NBUNDX()+1)</u>

III. the above relationships may be illustrated in the following table:

IFACEX()	NBUNDX() =1		NBUNDX() =2		NBUNDX() =3	
	NFACE	NSPACE	NFACE	NSPACE	NFACE	NSPACE
2	2	-2	0	-3	-2	-5
3	3	0	1	-2	-1	-4
4	4	1	2	-1	0	-3
5	5	2	3	0	1	-2
6	6	3	4	1	2	-1
7	7	4	5	2	3	0
8	8	5	6	3	4	1

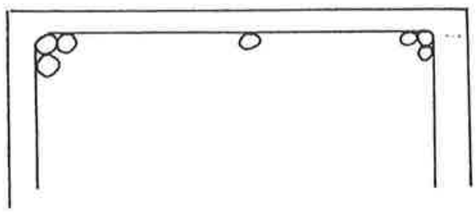
A -ve NFACE indicates that there are no bars near the face. If NSPACE is <1 then the bundled solution is not valid. This fact is used to check the validity of any bundling attempt.

<u>Table 2.16.2 Bar Relationships</u>

IV. The spacing is then calculated as follows:

$$SPACEX() = \frac{DX - 2*COV - NFACEX*DB}{NSPACEX()}$$ DB = bar diameter.

For example:



IFACE = 7

IBUND = 2

NFACE = 5

NSPACE = 2

Figure 2.16.5 Bundled Bars

(i) Check whether beam bar spacing is to be considered and if so check the spacing and number of face bars against the number allowed by BCLASH. If this is not satisfied then set COST = 10,000.

For bar sizes greater than that specified as the maximum allowed by the user, set COST = 10,000.

If the steel percentage finally provided is greater than .08 then set COST =10,000.

(j) Determine the cheapest solution found which is less than 10,000. This is done by sorting through COST and storing the value of I if a solution is found which is cheaper than the previous. An error flag is set if the COST turns out to be 10,000.

(k) Check splice congestion (divide SPACEX() and SPACEY() by 2 which gives the spacing at a splice because double the number of bars is present). If this is not greater than 40mm or 1.5DB then set ISPLICE = 1 which indicates that the bars must be offset.

(l) Design the stirrup reinforcement (Section 6.7 and 6.8 in AS1480) For each bar diameter:

i If NBUNDX() or NBUNDY() is greater than D
then stirrup dia = 10 mm and spacing = D/2 or BxDB

ii If both NBUNDX() and NBUNDY() = 0
then stirrup dia = 6 mm and spacing = D or 16xDB

Where D = the smaller of DX and DY. (Pg. 225)

Note that Section 11.10.3.1 c) v) in AS1480 allows the floor slab to provide lateral restraint at a splice. As there will be no splices within a column length it will not be necessary to provide extra lateral support (in the form of more stirrups) for the offset bars in a splice.

Note that steps J, K, L are not included in the Replacement Module because their function is carried out in the Detailing Module.

(m) Put the solution into a form usable by the detailing module of GENESYS.

If I is the solution number chosen then set DB = 4(I+2). This is discussed more fully in Section 4.

2.16.4 Public Storage: (See Section 3)

PUBLIC/CR/ALP,ASC,B1,B2,CLD(,),COST(),COV,ECX,ECY,FK(,),IBARS(,),IBIAX,ICNT,IERR ,KSXY,NB1,NB2,NOER(),NT1,NUTRAL,STEELP(,),SXY(),TBLE(,) Also NBUNDX() and NBUNDY() if bar bundling is to be permitted.

2.16.5 Local Storage

In the following, I = 1,7 is the bar size indicator

I = 1 represents a 12mmbar

I = 2 represents a 16mm bar etc

	A	Temporary storage of steel area
	AST(I),I=1,3	Steel area calculated from STEELP(I,1).
+	COST(I),I=1,7	Cost of each solution
	D	Spacing test variable
**	DX1,DY1	Temporary storage for column dimensions used when considering the effects of the bundled bars on side face
	EFFBAR(I),I=1,7	Effective number of side face bars
	IB	Load case number producing biaxial design case
**	IFL	=0 if going through SPACEing for the first time
	ICOST	I value of the cheapest solution
	IDIFFX(I),I=1,7	Number of additional bars in X-Face
	IDIFFY(I),I=1,7	Number of additional bars in Y-Face
	IFACEX(I),I=1,7	Number of X-Face bars
	IFACEY(I),I=1,7	Number of Y-Face bars
	ISP	Next integer value above SP (but not greater than 8)
	IX	Load case number producing Mono-X design solution
	IY	Load case number producing Mono-Y design solution
	K	Moment, direction indicator
	MHIGH	Moment capacity if ISP were present
	MU	Applied ultimate moment
	NBIBAR(I),I=1,7	Number of bars required for biaxial solution
	NFACEX(I),I=1,7	Number of bars closest to the X-Face
	NFACEY(I),I=1,7	Number of bars closest to the Y-Face
	NSPACEX	Number of spaces in the X-Face
	NSPACEY	Number of spaces in the Y-Face
	SP	Steel percent provided
	SPACEX(I),I=1,7	Spacing of bars in the X-Face
	SPACEY(I),I=1,7	Spacing of bars in the Y-Face
	TCOST	Current lowest cost
	Also	I Counting variable representing bar size

** Only used if bar bundling is permitted

\+ Not used by the replacement module because the system contains its own routine for determining the cheapest solution.

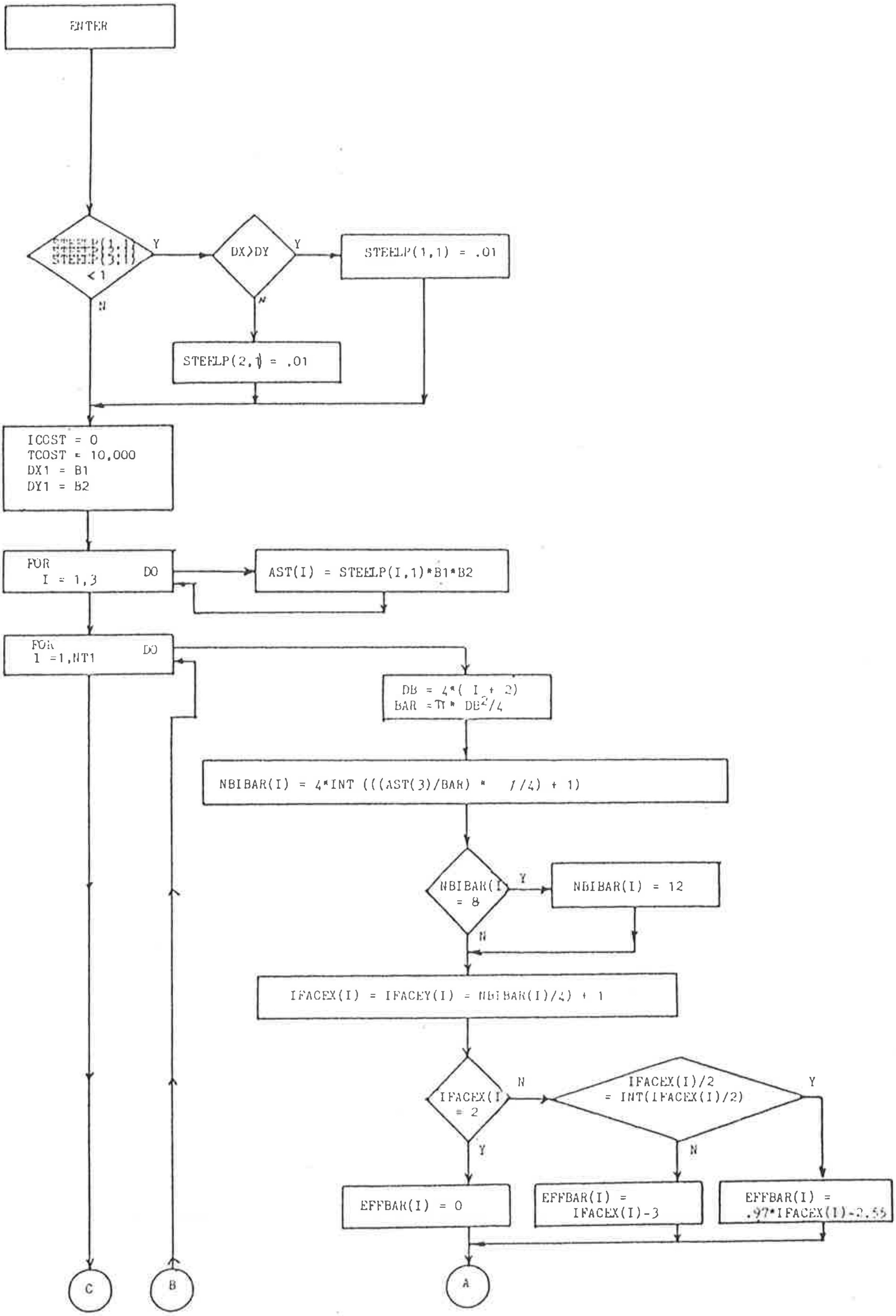


FIGURE 2.16.6 - Flow Chart - ARRANG Part 1

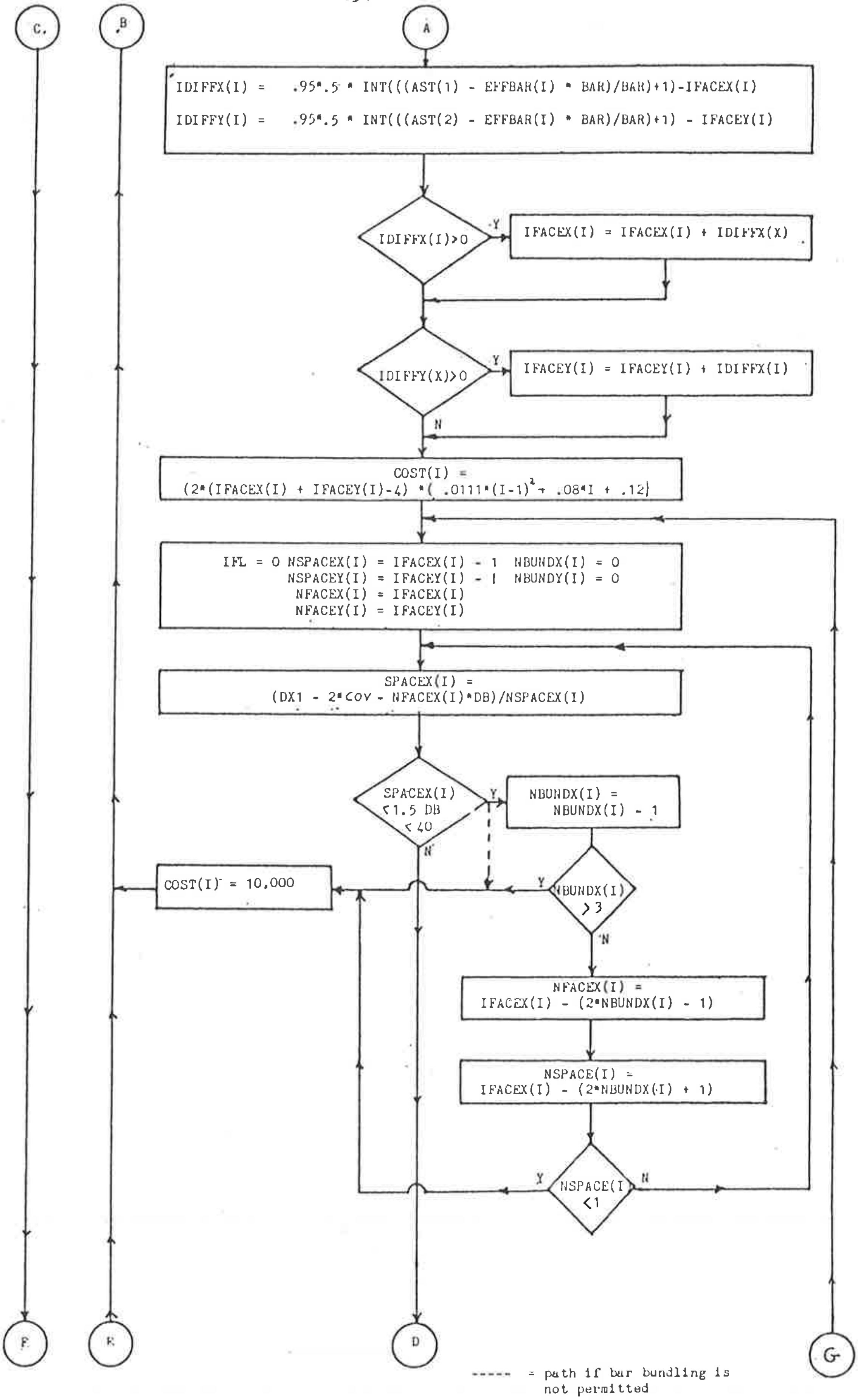


FIGURE 2.16.6 - Flow Chart - ARRANG Part 2

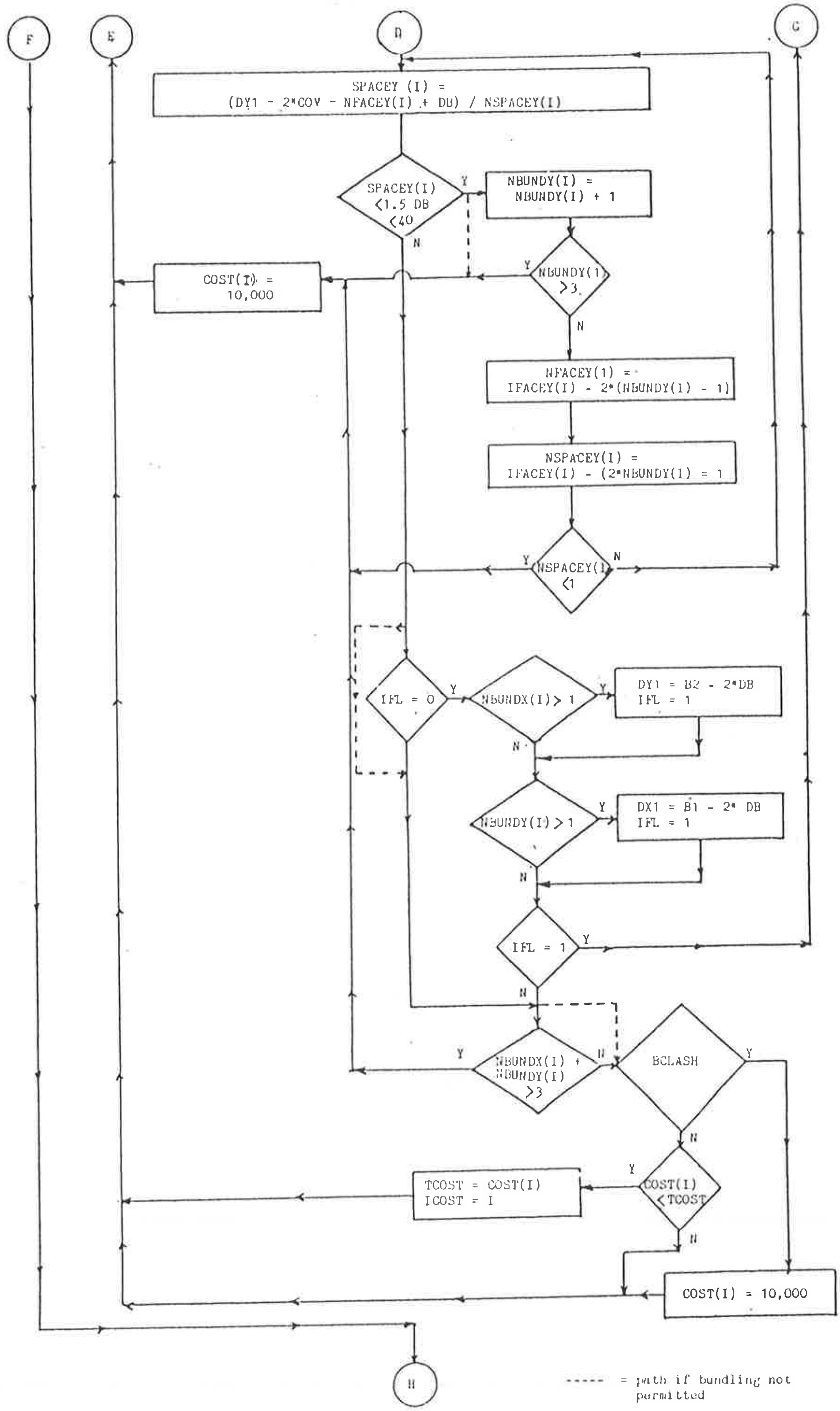
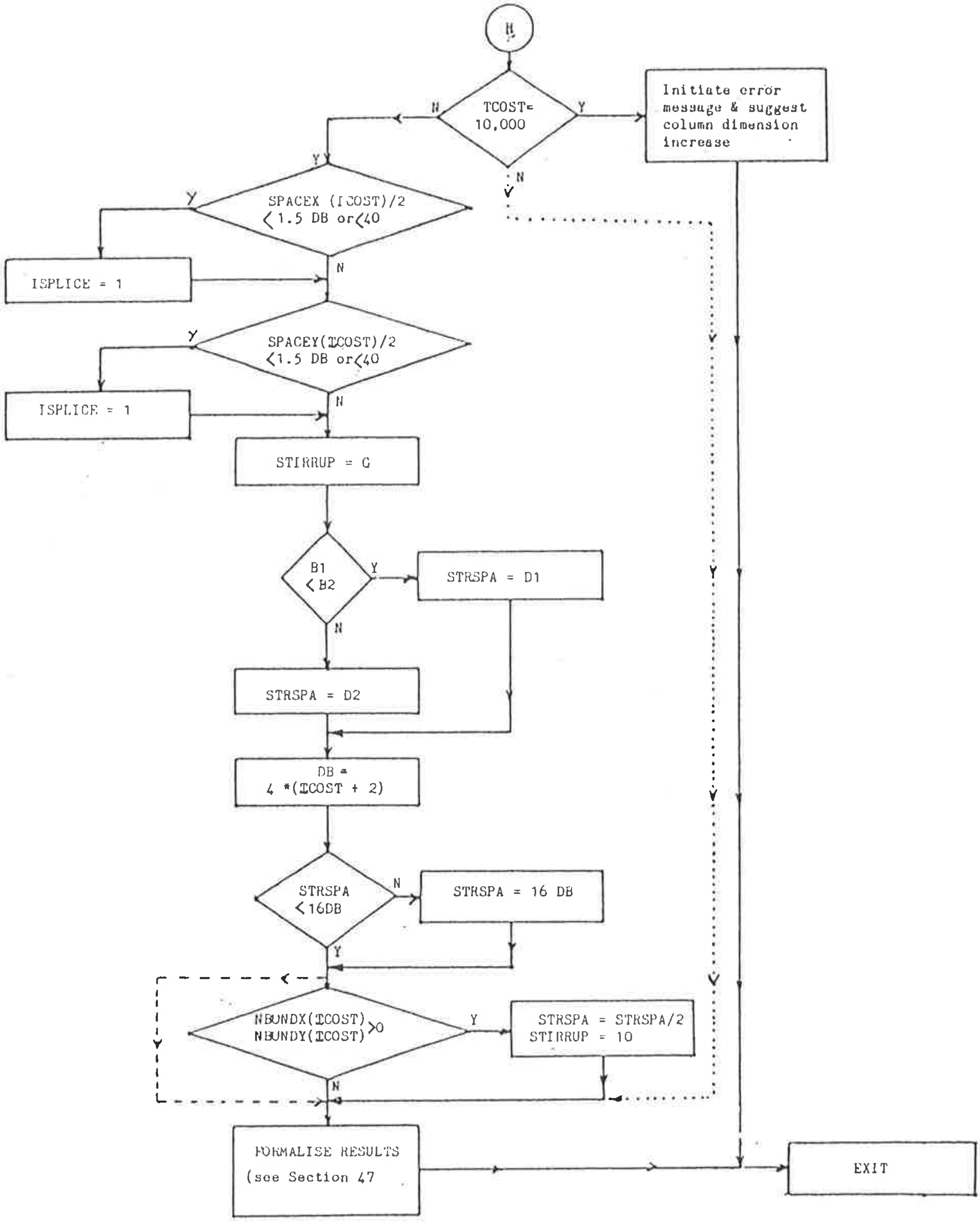


FIGURE 2.16.6 - Flow Chart - ARRANG - Part 3



----- = path if bar bundling not permitted

..... = path if stirrup design is done elsewhere

FIGURE 2.16.6 - Flow Chart - ARRANG - Part 4

3. DESCRIPTION OF PUBLIC VARIABLES

3.1 Introduction

This section describes in detail the function of each PUBLIC variable used in the module. Included as well are the PUBLIC variables that were used in the original GENESYS module but no longer required.

3.2 Variables in PUBLIC/COLUMNS/

CSP(IPROP,INO): Defines dimensions and eccentricities of all columns. If there is no column above the current column then relevant data vectors do not exist.

INO=1: Properties relate to column above.
=2: " " " " considered

IPROP =1 Width about Y-Y axis (mm)
=2 " " X-X axis (mm)
=3 Eccentricity about Y-Y axis (mm)
=4 " " X-X axis (mm)

In addition IPROP may indicate various properties of circular or L-shaped columns which are not relevant here.

GL(IPROP)		Contains various details of column properties - heights, shears, and stiffness.
IPROP	=1	Distance from the highest structural level at bottom of column to lowest structural level at the top of the column.
	=2	Clear height of current column for bending about the Y-Y axis (mm).
	=3	As for 2 but for X-X axis (mm)
	=4	Distance from highest structural level at bottom of column to highest structural level at top of column(mm)
	=5	Maximum structural thickness at top of current column (mm)
	=6	Maximum shear along X-X axis (kN)
	=7	Maximum shear along Y-Y axis (kN)
	=8	Clear height of column above current column (bending about Y-Y axis (mm))
	=9	As above for X-X axis
	=10	Ratio of sum of column stiffnesses to the sum of the beam/flat slab stiffnesses at the bottom of the current column for bending about the Y-Y axis
	=12	As above for the top of column about Y-Y axis
	=13	As above for the top of column about X-X axis
	=14	As above for top of column above for Y-Y axis
	=15	As above for top of column above for X-X axis

Note: (a) structural levels and structural thicknesses refer to flat-slabs where present, otherwise to beams

(b) clear height refers to the minimum clear height taking into account both beams and flat-slabs.

IGL(IPROP,ICOL)		Defines grids, floors and orientation of columns within batch
ICOL		number of column within batch
IPROP	=1	Engineer's entry under GRID1 of input table
	=2	Engineer's entry under GRID2 of input table
	=3	slashed form "f/f" defining start and end of column lift
	=4	"FAR", "RIGHT", "NEAR", or "LEFT" - the first face of the column into which a column strip frames
	=5	title of column strip framing into the face named under IPROP = 4(in form "gL/g2/g1") related to the grid references of the columns
IP(IC,INO)		Stores references to the columns to which each loading combination in the public array P applies
INO		Number of load combination
IC	=1	Contains the subscript ICOL in the public array IGL defining the ith. column to which the loading combination applies
KOLUMN	Indicator	to lower or upper column shaft being designed
	=0	lower column lift
	=1	upper column lift
P(IPROP,INO)		Stores all loading combinations applied to the column being designed and to the column above it
INO	Number of	load combinations (maximum is n + m as given in MS()

IPROP =1 Axial load in lift (inc selfW.)(kN)

=2 Moment X-X at head (kNm)

=3 Moment Y-Y at head (kNm)

=4 Moment X-X at foot (kNm)

=5 Moment Y-Y at foot (kNm)

=6 = -1.0 : loading contains wind load

= 1.0: loading does not contain wind load

Note: moments are stored clockwise +ve when the gridline defining the axis is viewed with its positive direction running from left to right.

ST(IPROP) for defining characteristic strengths, cover and kicker size

IPROP =1 characteristic strength of concrete in current column (MPa)

=2 ultimate anchorage bond stress in compression for current column (MPa) (not used in AS 1480 version)

=3 maximum shear stress for current column (Nmm^{2})

=4 characteristic strength of concrete for column above current column (MPa)

=5 ultimate anchorage bond stress in compression for column above current column (MPa)

=6 characteristic strength of main steel (bars > 16mm) (MPa)

=7 characteristic strength of main steel (bars < 16mm) (MPa)

=8 kicker sizer (mm)

=9 cover

=10 ultimate anchorage bond stress in tension for current column

=11 ultimate anchorage bond stress in tension for column above current lift

=12 maximum aggregate size (mm)

=13 tensile force in column tie (kN) in X - direction (=0.0 if none specified)

=14 as for 13 but in Y - direction

=15 characteristic strength of link steel(bars > 16mm) (MPa)

=16 characteristic strength of link steel (bars < 16mm)(MPa)

Note: The anchorage bond stress is calculated for the concrete grade and steel designation selected by the user for the respective column lift, unless defined specifically in the concrete properties table.

MS(IPROP) Defines miscellaneous data

IPROP =1 = 0 : design only

> 0 : detailing required

=2 number of columns within Type

=3 n, number of load cases on column lift to be designed

=4 m, number of load cases on column lift above lift to be designed

=5	= 1 : column is unbraced
	= 2 : column is braced E - W only
	= 3 : column is braced N - S only
	= 4 : column is braced in both directions
=6	title of group
=7	= 2 : column is rectangular
=8	= 0 : column above exists
	= 1 : no column above
=9	BATCH number
=10	Designation of main steel ("R", "HY", or "CW")
=11	Designation of link steel ("R" etc)
=12	Maximum bar size for corner bars (mm)
=13	Preferred bar size for corner bars (mm)
=14	Maximum bar size for intermediate bars (mm)
=15	Preferred bar size for intermediate bars (mm)
=16	Concrete Strength (MPa)
=17	Steel type (0, 1, or 2 specified in CP110)
=18	Maximum bar size for beam bars (mm)
=19	= 0 : beam bars single
	= 1 : beam bars paired
=20	= 0 : no beams in EW or NS direction
	= 1 : beam in E - W direction only
	= 2 : beam in N - S "
	= 3 :beam in both directions

3.3 Variables in PUBLIC/RCSYS

IISYS Column slenderness marker

=0 slender column

=1 short column

ISPCL Indicator to whether the load case is a light load or not

ISYS Load case number under consideration

LDBI(I) Contains pointers to the biaxial load cases in the array SLD (,)

I = 1,N Total number of biaxial cases in the run = N

NSYS Total number of load cases in the run

SLD(I,J) Contains the considered load cases in the analysis stored in order of decreasing values of the applied axial load

J Load case number

I =1 Axial load (N)

=2 Moment in X - direction (Nmm)

=3 Moment in Y - direction (Nmm)

3.4 Variables in PUBLIC/CR/

AA(I,J) Area of steel bars taken in pairs

J Bar size marker

I Number of pairs

ACRIT		The critical axial load value for a design solution
ADW(I)		Holds area of steel of bars in each row in both directions I and normal to I
ASC		Required area of steel for the load case
B		Width of column (mm)
B1		Dimension of column in X - direction (mm)
B2		Dimension of column in Y - direction (mm)
CLD(I,J)		Defines the computational results of load case J
J		Load case number, as in SLD(,)
I	=1	= 0 :load case is not critical
		= 1 : load case used for determining steel
	=2	Value of axial load (kN)
	=3	Initial moment in X- direction (kNm)
	=4	" " " Y- " "
	=5	= 0 (Dummy)
	=6	N / (B x H)
	=7	Initial moment in X- direction /(BH)
	=8	Initial moment in Y- direction /(BH)
	=9	Moment in X plus addition due to slenderness in X direction (kNm)
	=10	Moment in Y plus addition due to slenderness in Y direction (kNm)

CONS 1)
)
CONS 3)
)   Commonly used factors
CONS 4)
)
CONS 5)

COV   Cover to main bars (mm)

CRITLD(I,J)   Contains the maximum values of the critical load for a given bar arrangement inboth X and Y directions

J   =1   X - direction

     =2   Y - direction

I= 1,N   Where N = number of arrangements obtained

DIAM(I)   Alternative bar sizes for column reinforcement

I   =1   Size 12 mm

     =7   Size 40 mm

EL   Effective length of the column

EX   Effective length in X - direction

EY	Effective length in X - direction
FCU	Concrete strength (MPa)
FIVE	Maximum steel % allowed
FK(I,ICNT)	Adjustment factor used in conjunction with the additional moment induced in the column by its deflection
I = 1,NSYS	Loading number
ICNT	Solution number
FY	Steel strength (MPa)
H	Depth of column w.r.t. load case direction (mm)
IBARS(I,J)	Contains the number of bars for arrangements obtained in the run. Initialised to contain the symbols "BL" for all bar sizes
J = 1,7	Pointer to corner bar size
I = 1,7	Pointer to intermediate bar size in X - direction
= 8,14	Pointer to intermediate bar size in Y - direction
ICNT	Number of design solutions obtained for the column
IERR	Number of error messages
INIT	Marker for the control of subsequent load cases analysis

IWARN		Number of warning messages
IZ		Counter to the design solution being checked
KSXY		Pointer to whether beam bar clashing is to be considered or not
MBI		Flag to the load case type
	=0	First load case
	=1	Subsequent load case
	=2	Calculation of critical axial load
N60		Error flag if combination violates spacing
NB1		Maximum number of bars permitted in X - faces
NB2		Maximum number of bars permitted in Y - faces
NBI		Number of intermediate bars in face normal to the moment direction
NOER(I)		Contains the warning messages identification if any encountered during the course of design
NOWR(I)		Contains the error messages identification if any during the course of design

NT1		Marker to the maximum bar diameter to be used which will not exceed 8% steel area limitation or = maximum user specified size (Integer value 1—7)
NUTRAL		Indicator to ultimate failure type
	=0	Tension controlled
	=1	Compression controlled
OR		Control variable used by iterate routine
OU		Control variable used by iterate routine
PHI		Capacity reduction factor
PCLD(I)		Contains physical properties of column under consideration
I	=1	Clear height in X - direction
	=2	Clear height in Y - direction
	=3	Cover to main bars in X - direction
	=4	Cover to main bars in Y - direction
	=5	Slenderness ratio in X - direction
	=6	Slenderness ratio in Y - direction
	=7	Dummy
	=8	Dummy
	=9	Column section area
	=10	Inertia of column in X - direction
	=11	Inertia of column in Y - direction
	=12	Effective length in X - direction
	=13	Effective length in Y - direction

PP(J,J)		Array holding the value of applied moments in the column lift after considering the head and foot moments (if short, then the maximum of the respective head and foot moments will be taken. If slender then combined moment is taken)
J		Load case counter
I	=1	Value of applied axial load
	=2	Value of moment in X - direction
	=3	Value of moment in Y - direction
	=4,N11	pointer to the column reference that the load case applies to
SFS 1)		
SFS 2)		Constants used frequently in analysis
SFS 3		
SXY(I,J)		Contains the minimum permitted spacing between column bars in X - Y directions when bar clashing is to be considered in detailing
J	=1	X - direction
	=2	Y - direction
I	=1	Spacing between bars
	=2	Beam zone and spacing between bars in column section
TBLE(I,J)		Contains the analysis results of the column
J	=1	Bar size 12
	=2	Bar size 16

=NT1		Maximum bar size permitted (36mm)
I	=1	Area of steel required when using diameter corresponding to J'th entry
	=2	Applied moment direction, X, Y or both
	=3	Value of moment in X - direction
	=4	Value of moment in Y - direction
	=5	Axial load value
	=6	Maximum allowable moment in X - direction (kNm)
	=7	Bar size marker
	=8	Maximum allowable moment in Y - direction (kNm)
	=9	Table marker (ICNT)
XSL		Slenderness ratio in the X - direction
YSL		Slenderness ratio in the Y -direction
ZUN(I)		Contains the value of Nuz, the axial nominal load defined for all possible corner bar size used in the design of a column life
I	=1,NT1	Size indicator where NT1 is the maximum corner bar size usable

3.5Variables in PUBLIC/CR/: which were added for use by the new module

BIAX(I,J)		Contains the considered Biaxial Load casaes
I	=1	Axial load (P / bD)
	=2	Combined moment ($M_x/bD^2 + M_y/b^2D + P * Ec_{xy}$)
	=3	Corresponding load number in SLD()
J		Load number

COEFB(I,J)		Contains the coefficients of the balanced failure line shown on the design charts
I	=1	Line for Monoaxial - X chart
	=2	Line for Monoaxial - Y chart
	=3	Line for Biaxial chart
J	=1	Slope of line
	=2	Intercept on P/bD axis
COEFP(I,J,K)		Contains the coefficients of the equations representing the failure curves on the design charts
K	=1	Chart for Monoaxial - X
	=2	Chart for Monoaxial - Y
	=3	Chart for Biaxial
J	= 1,8	Steel percentage
I	= 1,7	Coefficient number
ECX		Initial eccentricity in X - direction
ECY		Initial eccentricity in Y -direction
IBIAX		Initial number of Biaxial cases found
IFLAG		Pointer to loading type being considered
	=1	Monoaxial - X
	=2	Monoaxial - Y
	=3	Biaxial
IMON		Initial number of monoaxial cases found

MONX(I,J)		Contains the considered Monoaxial - X Load caases
I	=1	Axial load (P /bD)
	=2	Moment (M_x/bD^2 + P *ECX)
	=3	Corresponding load number in SLD(,)
J		Load number
MONY(I,J)		Contains the considered Monoaxial - Y Load casaes
I	=1	Axial load (P /bD)
	=2	Moment (M_y/b^2D + P *ECY)
	=3	Corresponding load number in SLD(,)
J		Load number
NBUNDX(I)		Contains the number of bundled X -face bars (if bundling permitted)
I	=1,7	Bar diameter
NBUNDY(I)		Contains the number of bundled Y - face bars (if bundling permitted)
I	=1,7	Bar diameter
NSYSB		Total number of Biaxial Load cases in BIAX(,)
NSYSX		Total number of Monoaxial -x load cases in MONX(,)
NSYSY		Total number of Monoaxial -Y Load cases in MONY(,)

PMAX		Intercept of P = 8% curve on P/bD axis on Design Charts
PMIN		Intercept of P = 0% curve on P/bD axis on Design Charts
SLOPE(I)		Slope of P=0% curve as it crosses P/bD axis
I	=1	Monoaxial - X Chart
	=2	Monoaxial - Y Chart
	=3	Biaxial Chart
STEELP(I,J)		Steel % required to cover all loading cases
I	=1	For Monoaxial - X
	=2	For Monoaxial - Y
	=3	For Biaxial
J	=1	Steel percentage
	=2	Load number in SLD(,) from which the percentage was derived

COMPUTER AIDED DESIGN OF CONCRETE COLUMNS

APPENDIX 6

METHOD FOR AVOIDANCE OF COLUMN/BEAM BAR CLASHING

1. METHOD

The method adopted to avoid bar clashing at beam/column intersections is as follows:

(a) Select the largest column bar size the user has given (default 32mm) and place in each corner bar position.

(b) Set the link bar size (l_{max}) to the maximum possible (i.e. = 10 mm for bundled column bars)

(c) Check that the cover to the maximum main corner bar "Cv" given by ($Cl + l_{max}$) is equal to or greater than the maximum column bar nominal size. If not, increase cover (Cl) until cover to main bar ($Cl + l_{max}$) is equal to the nominal size of the max column bar CM_{max}.

$$Cv = MAX(\ CM_{max},\ (Cl + l_{max}))$$

(d) Calculate the theoretical maximum number of internal bars (N), of the maximum user given size, that could be placed in each column face.

Let X = length of the long face

Y = length of the short face

$$Nx = \frac{X - 2(Cv + M) - B}{M + B}$$

$$Ny = \frac{Y - 2(Cv + M) - B}{M + B}$$

$M = MAX(BE_{max}, CZ_{max}, (Beam\ Agg + 5))$

$B = MAX(CM_{max}, BZ_{max}, (Col\ Agg + 5))$

$Cv = MAX(CM_{max}, (Cl + l_{max}))$

Put the value of Nx and Ny equal to the integer part of Nx and Ny

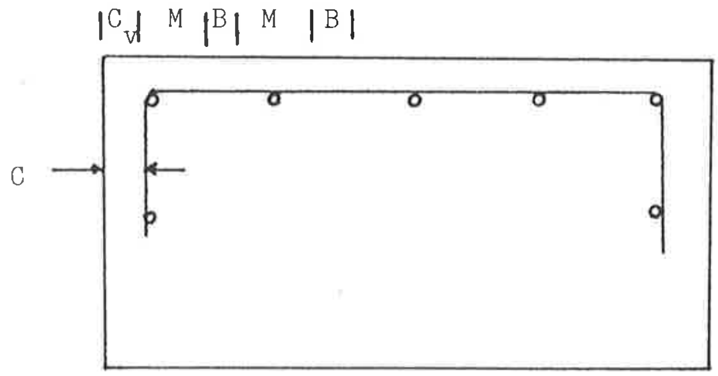


Figure 6.1 Typical Section

Nx = No of internal bars in X - direction

Ny = No of internal bars in Y - direction

$$X = 2(Cv) + (Nx + 2)*M + (Nx + 1)*B$$

$$= 2(Cv) + 2M + B + Nx*(M + B)$$

$$Nx = \frac{X - 2(Cv + M) - B}{M + B}$$

(e) Calculate the spacings Sx and Sy between internal bar positions on the long and short faces as follows:

On the long face $Sx = \frac{X - 2(Cv) - M}{(Nx + 1)}$

On the short face $Sy = \frac{Y - 2(Cv) - M}{(Ny + 1)}$

(f) Calculate the location of the first internal bars on the long face relative to the left hand and right hand short faces of the column from:

$$DIMF1_x = (Cv + \frac{M}{2}) + Sx \text{ (first internal on long face)}$$

Also the location of the first internal bars on the short face relative to the long faces of the column from:

$$DIMF1_y = (Cv + \frac{M}{2}) + Sy$$

If other internal bars are possible locate them (or it) at successive distances of Sx or Sy from the first internal bars on the face being considered.

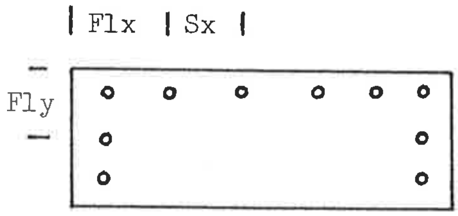


Figure 6.2 Typical Section

Note that if only one internal bar is possible its location will be the centreline of the column.

$$F1X = Cv + \frac{M}{2} + Sx \qquad F1Y = Cv + \frac{M}{2} + Sy$$

Where $Cv = Max\ (Cm_{max}, (C1 + 1_{max}))$

(g) The location of internal bar positions once determined will not change whatever the size of the bar finally selected for use.

The centreline positions of corner bars in the column will however vary with the bar size selected and will depend on the bar diameter although the cover to corner bars always remains a constant at $(Cl + l_{max})$ or CM_{max} whichever is larger. For this reason, for analysis internal bar locations are not measured from the centreline of corner bars but from the column's external concrete faces. For detailing the DIMENSION DIMF will be given from the face of the corner bar (inside of link) as in R.C. - BUILDING/1.

(h) If the number of internal bars required for a solution is less than the maximum possible number that could be accommodated then some locations will be left unfilled.

(i) If the maximum possible number of internal bars is an even number the first positions to be occupied will be the two nearest to the corner bars. Additional bars being added if needed in pairs at successive distances "S" moving towards the centreline.

(j) If the maximum possible number of internal bars is an odd number and the number required for a solution is even, procede as in I above filling the outside positions first and working in pairs towards the column centreline.

(k) If the maximum posssible number of bars is odd and the number required for a solution is also odd, then first locate one bar on the column centreline and then any additional bars required in pairs starting as in I from the outside locations and working inwards towards the column centreline.

(1) It should be noted that where the maximum possible number of internal bars is even, only an even number can be used. i.e. a maximum possible four internal bar column can have only two or four bars. It cannot contain three bars per face because of lack of symmetry. Where the maximum possible number is odd for example a possible five internal bar column, any number from 1 up to five may be used. Odd bar arrangements will contain a bar on the column centreline position and for even bar arrangements the column centreline will remain empty.

2. ALGORITHM

$$\text{Number of bars} = \mathrm{INT}\left(\frac{(B - 2(C+L+M)) - D}{M+D}\right)$$

Where L = 10 (link size for bundled bars)

M	= MAX(MCZ,EBBS)
D	= MAX((MBZ,MCBS),Col agg + 5))
MCZ	= 1.1 x MCBS
MBZ	= 1.1 x MBBS if beam bars single
	= 2.2 x EBBS if beam bars paired
EBBS	= MBBS if beam bars single
	= 2 xMBBS if beam bars paired
MCBS	= Permitted maximum column bar size
C	= cover to link
B	= dimension of the column face under consideration
MBBS	= permitted maximum beam bar size