REPORT NO. FRA-OR&D-76-143

PROGRAM MULTI

A MULTI-PURPOSE PROGRAM FOR COMPUTING AND GRAPHING ROOTS AND VALUES FOR ANY REAL FUNCTION

Users/Programmers Manual

Russel Brantman

U.S. DEPARTMENT OF TRANSPORTATION
Transportation Systems Center
Kendall Square
Cambridge MA 02142

MAY 1976

FINAL REPORT

Prepared for

U.S. DEPARTMENT OF TRANSPORTATION
FEDERAL RAILROAD ADMINISTRATION
Research and Development
Washington DC 20590

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FRA-OR&D-76-143		

4. Title and Subtitle
PROGRAM MULTI -- A Multi-Purpose Program for Computing and Graphing Roots and Values for Any Real Function -- Users/Programmers Manual

5. Report Date
May 1976

6. Performing Organization Code

7. Author(s)
Russel Brantman

8. Performing Organization Report No.
DOT-TSC-FRA-76-1

9. Performing Organization Name and Address
U.S. Department of Transportation
Transportation Systems Center
Kendall Square
Cambridge MA 02142

10. Work Unit No.
RR628/R6320

11. Contract or Grant No.
N.A.

12. Sponsoring Agency Name and Address
U.S. Department of Transportation
Federal Railroad Administration
Research and Development
Washington DC 20590

13. Type of Report and Period Covered
Final Report
July - December 1975

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

A generalized multi-purpose program has been developed that can be used to compute and graph cross sections of any surface in space, or to compute and graph the roots of any equation and any functions of these roots. It can therefore be used for a variety of applications, including the graphing of multi-valued functions whose branches are not known beforehand. This capability is unique among graphing programs, and it greatly facilitates the analysis of any system with multiple equilibrium branches.

The program is especially suited for computing the equilibrium branches and investigating the stability of nonlinear finite-degree of freedom systems subjected to static loads.

The program is oriented towards systems with one or two degrees of freedom, but it can also handle additional degrees of freedom and any number of parametric variables.

17. Key Words
Spatial Surfaces, Multi-Valued Functions, Roots, Computer Graphing, Nonlinear Systems, Equilibrium Branches, Elastic Stability

18. Distribution Statement
DOCUMENT IS AVAILABLE TO THE PUBLIC THROUGH THE NATIONAL TECHNICAL INFORMATION SERVICE, SPRINGFIELD, VIRGINIA 22161

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified		

Form DOT F 1700.7 (8-69)

PREFACE

The Transportation Systems Center (TSC), in support of the Federal Railroad Administration's Office of Rail Safety Research, is developing design guidelines for improved safety in rail equipment. As part of this responsibility (under the RR628 Rail Equipment Safety Project), TSC is acquiring, modifying, and developing computer programs for track/train dynamics analysis. These programs are being used to assess the behavior of alternate design concepts and operational procedures for improved safety, reliability, and performance of rail systems.

As part of this effort, TSC has been investigating the derailment behavior of trains when negotiating curves under buff or draft. To enable an analysis of how the critical car and train parameters affect the actual stability of the various train buckling modes, TSC has developed a generalized multi-purpose computer program.

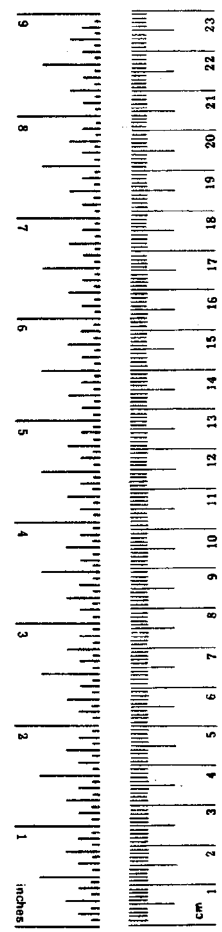
The author wishes to acknowledge the invaluable support of Mr. James Mulcahey and Mrs. Yvonne Silvia in providing assistance in the programming effort and in the development of the multi-valued graphing logic.

Preceding page blank

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	square miles	2.6	square kilometers	km^2
	acres	0.4	hectares	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	tablespoons	15	milliliters	ml
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C



Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (weight)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F

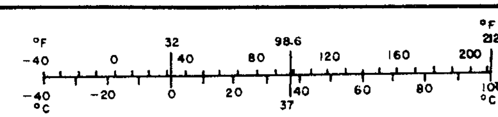


TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

1. INTRODUCTION

1.1 DESCRIPTION OF PROGRAM

The program is a generalized multi-purpose program that can be used to compute and graph cross sections of any surface in space, or to compute and graph the roots of any equation and any functions of these roots. It can therefore be used for a variety of applications, including the graphing of multi-valued functions whose branches are not known beforehand. This capability is unique among graphing programs, and it greatly facilitates the analysis of any system with multiple equilibrium branches.

In this respect, the program is especially suited for computing the equilibrium branches and investigating the stability of nonlinear finite-degree of freedom systems subjected to static loads.

Although the program is principally designed for functions of one or two variables, it can effectively handle any number of variables by looping the program and treating the additional variables as input parameters to be varied with each run.

1.1.1 Main Program

The main program provides for:

- Scanning of two independent variables (A=Alpha, B=Beta). The scan increments and ranges are independently specifiable. (If there is only one independent variable, the other may be treated as a parameter or need not appear at all.)
- Computation of the roots of any equation H(A,B)=0 either by linear interpolation or by Newton-Raphson iteration.
- Computation of up to any three functions P(A,B), W(A,B), and Z(A,B); where (A,B) are either the directly scanned values or the previously computed roots.
- Graphing of roots and functions, as selected.

1.1.2 Your Specifications

- Specifying your data and your equations is done in sequences C2 and C3 in Subroutine FRONT. Although your input of these items can be as complicated as you require (possibly involving additional reads, writes, and conditional branches), the only information actually required by the program is:

  C2) A data card specifying the data input variables. These variables select the scan ranges, the computational mode, and the functions you wish to compute and graph. A discussion on the use and specification of these variables is presented in Sections 2 and 3.

  C3) A list of the functions and/or characteristic root equation you wish to have computed. Specification of each equation is done according to a fixed format described in Section 4.

- Sequences C2 (Data Input) and C3 (Equation Specification) are divided into self-described subsequences Ci-j. Therefore, the following documentation should be read in conjunction with the accompanying program listings. These listings contain two examples of sample programs which have been inserted into sequences C2 and C3. (For a discussion of these sample programs, see Section 6.)

1.2 DESCRIPTION OF SUBROUTINES

The main program uses four subroutines:

- FRONT is used to input your data and equations.
- BRKUP and CONCAT contain the logic for organizing data for plotting functions that can be multi-valued, and the calls to the plotting routines stored in the system library (Scale, Plot, etc.).
- PBETAS is used to graph single-valued functions.

2. SELECTION OF PROGRAM OPTIONS

In this section, we describe how the data input variables are used to select the various program options listed in Section 1. (A complete discussion of each individual input variable is presented in Section 3.) The basic computational modes are selected by the input variables IH and ISG. The following descriptions are for the case when the absolute value of the input variable ISG is equal to 1. (If the absolute value of ISG is equal to 2, then the following descriptions apply with A and B interchanged.)

2.1 ROOT COMPUTATION MODE (IH≠0)

For each scanned A, the B roots of any equation H(A,B)=0 are computed either by linear interpolation (NX=0) or by Newton-Raphson iteration (NX>0). In the latter case, DHB (the partial derivative of H with respect to B) must be specified. For each IY≠0 (Y=P,W,Z), the corresponding user specified function Y(A,B) is evaluated for the previously computed roots. If ISG>0, then for each IY>0, the main program will use a special algorithm to graph the corresponding function Y against the independent variable A. If also IH>0, then a graph of B versus A is also produced.*

2.2 FUNCTION SCANNING MODE (IH=0)

For each incremented B value, the entire range of A is scanned; and for each IY≠0 (Y=P,W,Z), the corresponding user specified function Y(A,B) is evaluated at each of these scanned values. If

*Since for each A there are generally multiple B roots of undefined connectivity, a special algorithm has been developed which attempts to define the probable loci of these roots and to thereby define the multiple branches of each function. This ability to provide plots of multi-valued functions whose general loci are undefined is unique among graphing programs, and it greatly facilitates the analysis of any system with multiple equilibrium branches.

ISG>0, then for each IY>0, the main program will graph the corresponding function Y against the independent variable A. A separate graph will be produced for each incremented B value, thereby providing a cross section of the surface Y(A,B) at each of these B values.

2.3 DEGREE/RADIAN SELECTION

If A and B are in degrees, then the program recognizes the symbols RA and RB as the corresponding variables in radian measure; therefore, where applicable, the appropriate equations should be specified in terms of (RA,RB) instead of the indicated values (A,B). However, data input and output are always directly in terms of A and B.

3. DATA SPECIFICATION: INPUT SEQUENCE C2

- Subsequence C2-1 is to be used to insert any user required specification statements (such as Type, Data, and Dimension statements).
- Subsequence C2-2 is to be used to insert any user required read statements. The program will execute these read statements after reading in the data input variables specified below.
- Subsequence C2-3 is to be used to insert any user desired write statements for printout of any initial comments and any input data. (Note, however, that the data input variables specified below are automatically printed out.)

3.1 THE DATA INPUT VARIABLES

The data input variables which must be specified for execution of the program are:*

DA, AI, AF, DB, BI, BF, DC, E, NX, IH, IP, IW, IZ, ISG

These input variables are individually described below.

3.1.1 DY, YI, YF (where Y = A,B)

- DY = Y Scan Increment Value.
- YI = Y Initial Value.
- YF = Y Final Value.

(The only limits on the magnitudes of the above values are those set by the output format statements. In these statements YI and YF have been set at F9.3 and DY has been set at F7.3. However,

*The values for these data input variables are to be read in on a data card (or file) in free format. The read statement is contained in the main program, and the read logical unit is "1".

when graphing, or computing roots, there are certain additional dimension restrictions. These are described in Section 5.3).

(If either A or B is to be used as a parameter, rather than as an incremented variable, then set the appropriate DY = 0.0, and set the corresponding YI = YF = the value of the desired parameter. On the other hand, if this variable does not appear at all in the equations, then just set the corresponding DY = YI = YF = 0.0).

3.1.2 DC = arbitrary divide check on the denominators of the functions P, W, Z

(See 4.4 for explanation of how DC is used by the program. However, if the divide check does not concern you, or if you only wish to avoid dividing by zero, set DC=0.0).

3.1.3 E = Newton-Raphson convergence test parameter
NX = maximum number of interations

(Units of E = that of the root variable, A or B, as defined by the units of its input scan range).

(Limits of NX = 0 to 99).

(If Newton-Raphson scheme is to be bypassed, set NX=0, in which case E can be used as any desired parameter).

3.1.4 IH = integer check for basic computational mode

- IH=0 causes bypass of root computations and selects Function Scanning Mode.
- IH≠0 causes selection of Root Computation Mode.
    - IH<0 indicates that a printout of the computed roots is desired, but that a graph of these roots is not desired.
    - IH>0 indicates that the printout and the graph of the computed roots are both desired.

3.1.5 IY (where Y = P,W,Z)

IY = integer check for computation of the function Y and its graph.

- IY<0 indicates that a printout of the computed values for the function Y is desired, but that its graph is not desired.
- IY=0 indicates that computation of Y is to be bypassed.
- IY>0 indicates that both the printout of Y and its graph are desired.

3.1.6 ISG = integer check for scan and graph modes

ISG selectes the independent scan variable for data output. The remaining, or dependent, scan variable is then used to either increment the independent scan cross section (IH=0), or to solve for the roots corresponding to each independent scan value (IH≠0). If any graphs are desired, they are plotted against the independent scan variable.

- ISG = ± 1 selects A as the independent scan variable.
- ISG = ± 2 selects B as the independent scan variable.
- ISG<0 indicates that no graphs are desired.
- ISG>0 indicates that at least one graph is desired.

(See Section 2 for a more detailed description of the computational sequences generated by ISG. Also note that if ISG<0, then no graphs are produced even if some IY>0).

3.2 GENERAL COMMENTS ON DATA SPECIFICATION

- None of the data input variables can be omitted. They are read in on a data card (or file) in free format, and need only be separated by commas. Real values must contain a decimal point and may be in either E or F form. (The program expects to read this data on logical unit "1".)
- For each subsequence C2-j, insert your own program statements between the title card and spacer card. Even if you have no statements to insert, do not remove the title and spacer cards.

- Your inserted statements may be in the form of program segments; in which case any number from 50 on may be used for your statement labels. (For example, see Sample Program 1.)

4. EQUATION SPECIFICATION: INPUT SEQUENCE C3

Sequence C3 is to be used to insert your equations according to the following format. However, you need specify only those equations you actually require. Also, if your equations involve only one independent variable, the other may be treated as a parameter or need not appear at all.

4.1 SUBSEQUENCE C3-1 (CONSTANT SPECIFICATION)

This subsequence is to be used to insert any user defined constants. The expressions for these constants can involve any user supplied input data read-in in sequence C2, but they cannot be functions of the scan variables alpha and beta.

4.2 SUBSEQUENCE C3-2 (ROOT-SCAN PHASE)

- H = any equation in (A,B) [or in the corresponding radian variables (RA,RB)] whose roots are desired.
- Particular root or scan regions may be skipped by simply setting ISKIP=1 for those values and then transferring control to statement 22 (for example, see Sample Program 1).
- Whether or not H is to be specified, you may specify any computations desired during the scanning phase.

4.3 SUBSEQUENCE C3-3 (NEWTON-RAPHSON DERIVATIVE)

Let Y = A or B:

- DHY = partial derivative of H with respect to Y (or RY); where Y=A if ISG = $\pm$ 2, and Y=B if ISG = $\pm$ 1.
- If Y is in degrees and DHY is computed from the corresponding radian variable, then also set DHY=RAD*DHY.
- Specify DHA or DHB as required; or specify both derivatives (in which case the program will select the appropriate derivative).

4.4 SUBSEQUENCE C3-4 (FUNCTIONS P, W, Z)

Let Y = any function P,W,Z of (A,B) [or (RA,RB)] whose value is desired for each of the computed roots (or scan values if IH=0):

- YNUME = numerator of the function Y (if YNUME is not specified, then by default it is set = 0).
- YDNOM = denominator of the function Y (if YDNOM is not specified, then by default it is set = 1).

[Using this function computation approach, for each (A,B) YDNOM is compared against the value of the input parameter DC. If the magnitude of YDNOM is less than or equal to DC, then computation of the function Y is bypassed for that (A,B); and a notation is made on the printout indicating that Y(A,B) did not satisfy the divide check. (For example, see the output of Data Set 2 of Sample Program 1).]

4.5 SUBSEQUENCE C3-5 (FIXED GRAPH DATA)

The following fixed data items are used by the graphing subroutines. These data items are set by data statements to the indicated values. Ordinarily, the user will not need to change these values. (For a discussion of how these data items are used, and how they can provide greater graphing flexibility, see 5.5).

Let Y = H,P,W,Z (where P,W,Z is used to indicate the corresponding functions and where H is used to indicate the dependent variable of the alpha-beta pair):

- YVER = 0.0 and XHOR = 0.0
- YJOIN = 2.5

(YVER and XHOR are used in scaling the vertical and horizontal axes of the corresponding Y graph. YJOIN is used in assessing the probability of consecutive Y values belonging to the same locus).

4.6 GENERAL COMMENTS ON EQUATION SPECIFICATION

- Corresponding to each subsequence C3-j, insert your program statements, if any, between those statements labeled j0 and j1 of the appropriate subsequence. (For example, for subsequence C3-2, insert your program statements between those statements in subsequence C3-2 labeled 20 and 21).
- All subsequences C3-j, with their correspondingly labeled initial and final statements j0 and j1 (see above), should at all times be present – whether or not you have any statements to insert. Even if you do not require the radian variables (RA,RB), do not remove or alter these labeled subsequence statements.
- Your inserted statements may be in the form of program segments; in which case any number from 50 on may be used for your statement labels.
- Any quantity (including H, DHY, and the functions Y) may be redefined at any time on any condition.
- The Newton-Raphson Derivative and Function Computation subsequences (C3-3 and C3-4) can make use of any quantity defined in the Constant Specification and Root-Scan subsequences (C3-1 and C3-2) without that quantity having to be redefined.

5. GENERAL COMMENTS

5.1 DATA SETS

- A data set consists of the initial data for the data input variables and any data you specify, in accordance with subsequence C2-2, for one complete execution of the main progam.

- Before being read in, each data set is automatically sequentially numbered by the main program (L = Data Set Counter). When no data sets remain, the program exits normally (closing the plot output file if necessary).

- There can be up to 99 data sets (restricted only by main program formats); and there are no restrictions on the manner in which one may differ from another. In fact, with appropriate transfers (based on the value of L) in your inserted segments, you can effectively run completely different programs on any selected data sets. (For example, see the three data sets of Sample Program 1).

5.2 PROGRAM FLOW RESULTING FROM ZERO DATA

- NX=0 causes automatic bypass of all Newton-Raphson computations, even if DHA or DHB has been specified.

- IH=0 causes automatic bypass of the main program determination of roots and their corresponding graph, even if H and DHA or DHB have been specified. However, all input computations specified in the Root-Scan subsequence C3-2 are still performed for each scanned A and B.

- IY=0 (Y = P,W,Z) causes automatic bypass of the main program computation of the function Y and its graph, even if YNUME and YDNOM have been specified. However, all input computations specified in the Function Computation subsequence C3-4 are still performed.

5.3 DIMENSION RESTRICTIONS

- When computing functions directly from scanned values, no dimension restrictions are imposed.

- When computing roots, the program will allow 10 Beta (or Alpha) roots for each scanned Alpha (or Beta) value. There are no dimension restrictions, however, on the number of values to be included in the Alpha and Beta scans through which these roots are computed.

- When graphing either roots or functions, the maximum number of scanned values for the independent (X-axis) scan variable is 99.

- The program will allow a maximum of 20 rise pieces per graph, 20 fall pieces per graph, 50 stray points, and 10 equal pieces. (See subroutine BRKUP for the definition of these quantities).

- If either a dimension violation is encountered, or the wrong Newton-Raphson derivative has been specified, execution of that data set is thereupon terminated, an error message is printed out, and the next data set is then read in.

5.4 ROOTS

- If particular root or scan regions have been skipped, note that the actual excluded region will be the specified skipped region plus the subregion (if any) up to the first scan value outside the specified skipped region; i.e., if a root exists between the boundary of the specified skipped region and the first scan value outside the region, it will not be recorded.

- Also note that if more than one root exists bewteen successive scan values (as might occur near a bifurcation point), and if the number of such roots is even, they will go completely unrecorded; while if the number of such roots is odd, there will either be no convergence or only one root will be selected.

5.5 GRAPHING

In each of the following subsections let Y = H,P,W,Z (where P,W,Z is used to indicate the corresponding functions, and where H is used to indicate the dependent variable of the alpha-beta pair).

5.5.1 Scaling of Graph Axes (YVER and XHOR)

For each function Y to be graphed, the scaling of the vertical and horizontal axes is set by the minimum and maximum values in the corresponding arrays. (Note that since the horizontal axis is determined by the range of the independent scan variable, it is therefore common to all graphs of the same data set.) To enable adjustments to these scales, the data variables YVER and XHOR have been introduced. When computing the vertical scale for each Y graph and the common horizontal scale, the values of YVER and XHOR are temporarily included in the corresponding vertical and horizontal arrays.

Since YVER and XHOR are all presently set at 0.0, each graph will have a set of axes which include the point (0,0) in their scale. (For how YVER and XHOR are set, see 4.5).

5.5.2 Multi-Valued Graphing (YJOIN)

If the program is being run in the root computation mode ($IH \neq 0$), and multiple roots are computed for each independent scan value, then the program attempts to sort these multiple roots and their corresponding function values into multiple rising, falling, and equal arrays; and to then concatenate arrays of the same type if they occur sequentially. These arrays are used to define the probable branches of the multi-valued function; the points within these arrays are then connected when the graph is plotted. However, due to the problems which occur near bifurcation points (where branches cross), rising arrays are never connected to falling arrays, so that if a branch consists of both types, there will always be an unconnected gap when going from one type of array to the other.

In addition to defining and concatenating arrays, the program must also attempt to assess the probability of consecutive Y values belonging to the same locus. Consider three consecutive Y values (Y_0, Y_1, Y_2) and their differences $\Delta Y_1 = Y_1 - Y_0$ and $\Delta Y_2 = Y_2 - Y_1$. These differences are used to sort the Y values into rising, falling, and equal arrays based upon whether ΔY_n is positive, negative, or zero. However, if in any instance

$$|\Delta Y_n| \begin{cases} > \text{YJOIN}*|\Delta Y_{n-1}| & \\ < |\Delta Y_{n-1}|/\text{YJOIN} & (\text{if YJOIN} \geq 1) \\ < |\Delta Y_{n-1}|/5 & (\text{if YJOIN} < 1) \end{cases}$$

then Y_n is placed into what is called a stray array.

Stray array points are plotted, but no lines are drawn to connect them. This is based upon the hypothesis that if $|\Delta Y_n|$ is either significantly greater or smaller than $|\Delta Y_{n-1}|$, then the point Y_n probably should not belong to that branch. Accordingly, it is then put into a stray array. Based upon an examination of the various branches that have been drawn in, it is usually easy to then identify which branch a stray point actually belongs to, and to then manually draw in the connecting line.

In practice, if one sees a point on his graph that appears as if it should belong to a particular branch, but which is connected instead to some point way above or below that branch, then YJOIN is too large and should be reduced.

On the other hand, if there are too many stray (unconnected) points on the graph, then YJOIN is too small and should be increased. (Remember, though, that there are always gaps between rising and falling arrays.)

Presently, YJOIN is set at 2.5 for all graphs (see 4.5 for how YJOIN is set).

5.5.3 Bifurcation Points

It should be noted that in the neighborhood of bifurcation points (where curves cross), the graphing logic has been designed to indicate the probable loci; but, because of the uncertainties involved, the user should rely on his own knowledge of the problem to specify exactly what happens there.

5.5.4 Excluded Points

If any function P,W,Z does not satisfy the divide check for a particular value of (A,B), then that (A,B) value will be uniformly excluded from all the graphs (except the graph of the roots themselves). If the Newton-Raphson iteration scheme is used and convergence is not achieved, then the numerical printout will display the linearly interpolated value (with appropriate indication). However, this value will be excluded from all graphical output (including the graph of the roots themselves).

5.6 VARIABLE NAMES USED BY SUBROUTINE FRONT

A, AF, AI, B, BF, BI, DA, DB, DC, DHA, DHB, E, FLAGS (common block), FRONT (program name), H, HJOIN, HVER, IH, IP, ISCAN, ISG, ISKIP, IW, IZ, L, N, NX, PDNOM, PJOIN, PNUME, PVER, RA, RAD, RB, SUB (common block), WDNOM, WJOIN, WNUME, WVER, XHOR, ZDNOM, ZJOIN, ZNUME, ZVER.

6. SAMPLE PROGRAMS

The sample programs are used to demonstrate general program use, special program features, interpretation of program output, and to illustrate the general flexibility of the program. Input and output listings are provided along with the associated graphs that were produced.

An example of what the input program looks like before insertion of your data and equations can be obtained by first turning to the input listing for Sample Program 2 – since, in that program, the user inserted segments consist of only one equation and two write statements. The actual input program starts with the title sequence C1. All dotted or unmarked comments which follow are fixed parts of the input program (Subroutine FRONT).

6.1 SAMPLE PROGRAM 1

Sample Program 1 consists of three data sets. The input listing shows how the data and equations for the three data sets were entered and differentiated one from the other. Starred comments have also been inserted to draw attention to the use of program features of special interest. These features include insertion of program segments with transfers of control, skipping of particular root regions, and conditional redefinition of a specified function.

Following is a description of the three data sets and of particular features to be noted on their numerical and graphical printouts.

6.1.1 Data Set 1 (Sample Program 1)

On Data Set 1, the equilibrium states of a two-degree of freedom rhomboid frame are computed. Alpha and Beta represent the two degrees of freedom, P is the applied load, and Z is the included angle of the frame's base. An understanding of the problem, however, is not necessary to see how the program is used for its solution.

The basic frame parameters are read in after the input data required by the program by an inserted read statement in subsequence C2-2. The root computation mode of the program is used to compute the equilibrium states of the frame from its characteristic equation H(RA,RB)=0. These states are non-unique as shown by the multiple equilibrium paths depicted on the graph of Beta vs. Alpha. In this graph, the program was able to define the equilibrium branches without any uncertainty and to precisely plot the intersections. On the next two pages, graphs of P and Z corresponding to these roots are shown. Herein, the program was uncertain as to what exactly occurred in the neighborhood where the branches met. However, the user is alerted to this uncertainty by the fact that gaps were left, and so he is made aware that he must rely on his own knowledge of the problem to define what exactly occurs in these areas.

The numerical printout depicts the general format to be expected when the root computation mode is selected. For the example under consideration, ISG was set equal 1; so that alpha was selected as the independent scan variable, thereby causing the roots to be calculated in terms of beta. These roots and the corresponding function values were then plotted against the independent scan variable.

Further examination of the printout reveals that the first four lines of comments (from "Sample Program 1" through "Z =") were user supplied; the rest are supplied automatically by the program based upon the initial input data. Checking the column labeled ITERATIONS, which gives the number of iterations that were required for Newton-Raphson convergence, we see that the printout also includes a case in which the function divide check was not satisfied (ITERATIONS=-100), and a case in which Newton-Raphson convergence was not achieved (ITERATIONS=-NMAX=-4). (In this latter case the root value supplied is the linearly interpolated value.) When ITERATIONS=0 it means that a scanned value exactly coincided with a root.

6.1.2 Data Set 2 (Sample Program 1)

On Data Set 2, the function scanning mode of the program is used to simply compute the single variable function P = CTN(RA). By breaking up the function into its separate numerator and denominator, the divide check on the denominator of P is then used to limit the computed values to less than 20. On the numerical printout we see that the first computed value did not satisfy the denominator divide check; therefore, this point is excluded from the corresponding graph.

Also note that on the printout, when using only the function scanning mode, the word ROOTS actually refers to the scanned values; thus, 15 ROOTS means there were 15 scanned values of the independent variable alpha.

6.1.3 Data Set 3 (Sample Program 1)

On Data Set 3, the function scanning mode is again selected; but this time ISG is set equal to 2, so that the independent scan variable is beta. Also, in this case, we will be using a two variable function; however, the beta scan will be run through only once, for alpha = 45°. The function to be computed is P = H(RA,RB), where H is the characteristic root equation of Data Set 1. Accordingly, on the graphical output, we see that the values of beta at which the curve crosses the P=0 axis correspond to the beta roots that were calculated in Data Set 1 at this particular value of alpha.

6.2 SAMPLE PROGRAM 2

The purpose of Sample Program 2 is to demonstrate the ability of the program to generate cross sections of a 3-dimensional surface in space. (Such a surface might represent the total potential energy of a system whose stability is to be investigated.) The equation of the surface to be computed is $P = A + B^2$.

6.2.1 Data Set 1 (Sample Program 2)

On Data Set 1, ISG is set equal 1; thereby selecting A as the independent scan variable. Accordingly, a separate graph of P vs A is then produced for each incremented B value in its scan range. These graphs are straight lines at different intercepts (each intercept being the corresponding value of B^2). (Only the first and last graphs are shown.)

The numerical printout depicts the general format to be expected when the function scanning mode is selected and a two variable scan is employed.

Also note that on the printout, when using only the function scanning mode, the word ROOTS actually refers to the scanned values; so that 11 ROOTS means there were 11 scanned values of the independent variable alpha.

6.2.2 Data Set 2 (Sample Program 2)

On Data Set 2 we change the scan direction by setting ISG equal to 2, so that B is now the independent scan variable. Accordingly, a separate graph of P vs B is now produced for each incremented A value in its scan range. These graphs are parabolas having different vertices (each vertex being the corresponding value of A). (Only the first and last graphs are shown.)

APPENDIX A

I/O FOR SAMPLE PROGRAM 1

```
      SUBROUTINE FRONT
C
C
C***SAMPLE PROGRAM 1 INSERTED HERE DEMONSTRATES HOW PROGRAM MULTI
C***CAN BE USED TO CALCULATE AND GRAPH ROOTS AND ALSO HOW IT WILL
C***SCAN AND GRAPH VALUES OF FUNCTIONS.
C
C    --ON DATA SET 1 COMPUTE THE EQUILIBRIUM STATES OF A 2-DEGREE OF
C      FREEDOM RHOMBOID FRAME.
C    --ON DATA SET 2 SIMPLY SCAN ALPHA AND COMPUTE P = CTN(ALPHA),
C      FOR CTN(ALPHA) .LT. 20
C    --ON DATA SET 3 SIMPLY SCAN BETA (FOR ALPHA=45) AND COMPUTE P = H,
C      WHERE H IS THE ROOT EQUATION OF DATA SET 1.
C
C
C***THE SAMPLE PROGRAM WAS RUN WITH THE FOLLOWING DATA INPUT FILE:
C
C    5.0,-15.0,115.0,10.0,-15.0,115.0,.01,.001,4,2,3,0,2,1
C    10.0,1.0,.5
C    5.0,0.0,70.0,0.0,0.0,0.0,.05,0.0,0,0,1,0,0,1
C    0.0,45.0,45.0,5.0,-20.0,120.0,0.0,0.0,0,0,1,0,0,2
C
C***THE FIRST 2 INPUT LINES COMPRISE DATA SET 1 (THE 2ND LINE CORRESPONDS
C***TO THE INSERTED READ STATEMENT 521), THE NEXT LINE SPECIFIES DATA
C***SET 2, AND THE LAST LINE SPECIFIES DATA SET 3.
C
C
C
C
C1...MULTI-PURPOSE PROGRAM FOR COMPUTING AND GRAPHING
C     ROOTS AND/OR VALUES FOR ANY REAL FUNCTION
C     --BY RUSSEL BRANTMAN
C
C
      COMMON/FLAGS/ISCAN,IH,IP,IW,IZ,ISG,L,HJOIN,PJOIN,WJOIN,ZJOIN,
     &              HVER,PVER,WVER,ZVER,XHOR

      COMMON/SUB/DA,AI,AF,DB,BI,BF,DC,E,NX,N,ISKIP,H,A,B,DHA,DHB,
     &             PNUME,PDNOM,WNUME,WDNOM,ZNUME,ZDNOM
C
C          SEQUENCE CONTROL
      IF(N.EQ.-2) GO TO 40
      IF(N.GT.-2) GO TO 20
C
C
C
C2...DATA SPECIFICATION  (SEE DOCUMENTATION)
C
C  L IS THE DATA SET COUNTER WHICH IS INITIALIZED TO 1 AND AUTOMATICALLY
C  INCREMENTED BY THE PROGRAM.   L MAY BE TESTED FOR TRANSFERRING TO THE
C  INFORMATION APPROPRIATE TO THE DIFFERENT DATA SETS.
C  (ON OUTPUT, EACH DATA SET WILL START ON A NEW PAGE)
C
C
C2-1...SPECIFY ANY SPECIFICATION STATEMENTS NEEDED FOR YOUR MATERIAL
C
      REAL K1,K2
C
C
C2-2...SPECIFY ANY DESIRED READ STATEMENTS
C
```

```
C***NOTE FOLLOWING TRANSFER OF CONTROL BASED UPON THE DATA SET
C***COUNTER L.  THIS IS ALSO USED IN SUBSEQUENCES C3-2 AND C3-4 TO
C***TRANSFER TO THE EQUATIONS APPROPRIATE TO EACH DATA SET.  ALSO
C***NOTE USE OF STATEMENT NUMBERS .GE. 50 FOR YOUR INSERTED
C***PROGRAM SEGMENTS.
C
      GO TO (521,522,523),L
521   READ(1,65) T,K1,K2
65    FORMAT(3F)
      GO TO 531
522   GO TO 532
523   GO TO 533
C
C
C2-3...SPECIFY ANY DESIRED WRITE STATEMENTS
C
531   PRINT 546
546   FORMAT(//'     SAMPLE PROGPAM 1')
      PRINT 500,T,K1,K2
500   FORMAT(///' THETA ZERO = ',F4.1,7X,'K1 = ',F3.1,7X,
     1'K2 =',F3.1//' P = APPLIED LOAD ON RHOMBOID FRAME'/
     2' Z = INCLUDED ANGLE OF FRAME''S BASE JOINT',
     3' (TO NEAREST DEGREE)'//)
      GO TO 545
532   PRINT 510
510   FORMAT(//' P = CTN(ALPHA), FOR CTN(ALPHA) .LT. 20'//)
      GO TO 545
533   PRINT 543
543   FORMAT(//' P = H, WHERE H IS THE ROOT EQUATION OF DATA SET 1'
     1//)
545   CONTINUE
C
C
C
C3...EQUATION SPECIFICATION  (SEE DOCUMENTATION)
C
C
C3-1...SPECIFY YOUR CONSTANTS
C
C   DO NOT REMOVE STATEMENT 10 (RAD=)
10    RAD = 3.1415927/180.0
C
      RT = RAD*T
C
11    RETURN
C
C
C3-2...SPECIFY H AND ANY COMPUTATIONS DESIRED DURING SCANNING PHASE
C
C   DO NOT REMOVE STATEMENT 20 OR THE NEXT STATEMENT (RA=,RB=)
20    RA = RAD*A
      RB = RAD*B
C   TO SKIP OVER ANY SCANNED VALUES (OR ANY ROOT REGIONS),
C   SET ISKIP = 1 FOR THOSE VALUES AND THEN GO TO STATEMENT 22
C
C***NOTE FOLLOWING TRANSFER OF CONTROL BASED UPON THE DATA
C***SET COUNTER L.
C
      GO TO (551,552,551),L
551   RTA = RT + RA
      SA = SIN(RTA)
```

```
      CA = COS(RTA)
      RTB = RT + RB
C
C***SKIP REGION IN WHICH CROSSOVER OF FRAME'S ARMS OCCURS
      IF((RTA+RTB) .GE. 0) GO TO 80
      ISKIP = 1
      GO TO 22
C
C***THE ABOVE STATEMENTS ELIMINATE FROM THE (A,B) SCAN REGION
C***THOSE VALUES OF A AND B FOR WHICH (RTA+RTB).LT.0
C
80    RAB = RA + RB
      CB = COS(RTB)
      SB = SIN(RTB)
      SBA = SB - SA
      SK = (K1 + K2*SBA)*SBA
      H1 = RAB - SK*CA
      H2 = RAB + SK*CB
      H = H1*SB - H2*SA
552   CONTINUE
C
21    IF(N)22,40,30
22    RETURN
C
C
C3-3...SPECIFY NEWTON-RAPHSON VARIABLE DHY,  WHERE Y = A OR B
C       (IF DHY COMPUTED FROM RY,  SET DHY = RAD*DHY)
C
30    CONTINUE
C
      CK = (K1 + 2*K2*SBA)*CB
      DH1 = 1 - CK*CA
      DH2 = 1 - SK*SB + CK*CB
      DHB = H1*CB + DH1*SB - DH2*SA
      DHB = RAD*DHB
C
31    RETURN
C
C
C3-4...SPECIFY YNUME AND YDNOM OF FUNCTION Y,  Y = P,W,Z
C       (DEFAULTS ARE YNUME=0 AND YDNOM=1)
C
40    CONTINUE
C
C***NOTE FOLLOWING TRANSFER OF CONTROL BASED UPON THE DATA SET
C***COUNTER L.  NOTE SPECIFICATION OF FUNCTIONS IN TERMS OF
C***VARIABLES PREVIOUSLY DEFINED IN C3-2.  ALSO NOTE CONDITIONAL
C***DEFINITION OF ZNUME WITHIN A DATA SET ITSELF.
C
      GO TO (561,562,563),L
561   PNUME = H1
      PDNOM = SA
      ZNUME = A + B + 2*T
C***ROUND OFF Z TO THE NEAREST INTEGER IF MAGNITUDE OF Z .GT. 1
      IF (ABS(ZNUME).LE.1) GO TO 565
      IZNUME=ZNUME+.5
      ZNUME=IZNUME
      GO TO 565
562   PNUME = COS(RA)
      PDNOM = SIN(RA)
C
```

```
C***IN THIS DATA SET, IT WAS POSSIBLE TO SPECIFY THE DIVIDE CHECK
C***INPUT VARIABLE, DC, SUCH THAT THE MAIN PROGRAM DIVIDE CHECK ON
C***PDNOM WOULD IN EFFECT RETAIN ONLY THOSE VALUES OF THE FUNCTION
C***P .LT. 20.  HOWEVER, IF P WERE A MORE COMPLICATED FUNCTION, THIS
C***COULD HAVE BEEN ACCOMPLISHED BY CONDITIONALLY REDEFINING PDNOM
C***BASED UPON THE FOLLOWING CHECK:
C                    PNUME = YOUR SPECIFIED EXPRESSION
C                    PDNOM = YOUR SPECIFIED EXPRESSION, IF ANY
C                    P = PNUME/PDNOM
C                    IF (P .GE. 20) PDNOM = DC
C
      GO TO 565
563   PNUME=H
565   CONTINUE
C
41    RETURN
C
C
C3-5...FIXED GRAPH DATA
C
      DATA XHOR/0.0/
      DATA HVER,PVER,WVER,ZVER/4*0.0/
      DATA HJOIN/2.5/
      DATA PJOIN/2.5/
      DATA WJOIN/2.5/
      DATA ZJOIN/2.5/
C
C
C...END OF INPUT TO MAIN PROGRAM
C    (DON'T FORGET YOUR DATA CARD FOR THE DATA INPUT VARIABLES --
     DA,AI,AF,DB,BI,BF,DC,E,NX,IH,IP,IW,IZ,ISG)
C
      END
```

SAMPLE PROGRAM 1

THETA ZERO = 10.0   K1 = 1.0   K2 = .5

P = APPLIED LOAD ON RHOMBOID FRAME
Z = INCLUDED ANGLE OF FRAME'S BASE JOINT (TO NEAREST DEGREE)

ALPHA RANGE = -15.000 TO 115.000   ALPHA INCREMENT SCAN = 5.000
BETA RANGE = -15.000 TO 115.000   BETA INCREMENT SCAN = 10.000

DATA SET 1 --- ALPHA AND BETA SCANS, WITH LINEAR INTERPOLATION ON BETA AS INITIAL VALUE IN NEWTON-RAPHSON SCHEME
--- ACCURACY OF BETA ROOTS = 1.E-03
--- IF NO CONVERGENCE IN NMAX = 4 ITERATIONS, THEN BETA = INTERPOLATED VALUE, AND PRINTED ITERATIONS = -4
--- REMEMBER - THERE MAY BE ADDITIONAL ROOTS OUTSIDE THE SPECIFIED SCAN RANGE
--- P,W,Z = FUNCTIONS OF ROOTS
--- IF (ABS(DENOM OF P,W, OR Z) .LE. 1.E-02), THEN PRINTED FUNCTION VALUE = ***, AND PRINTED ITERATIONS = -100
--- ALL ANGLES = DEGREES

ALPHA = -15.000	BETA	P	W	Z	ITERATIONS	1 ROOTS (DATA SET 1)
	99.196	1.012E+00	0.000E-01	1.040E+02	2	

ALPHA = -10.000	BETA	P	W	Z	ITERATIONS	1 ROOTS (DATA SET 1)
	93.018	***	0.000E-01	1.030E+02	-100	

ALPHA = -5.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	-5.000	-2.003E+00	0.000E-01	1.000E+01	0	
	86.568	1.281E+00	0.000E-01	1.020E+02	2	

ALPHA = 0.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	0.000	4.747E-09	0.000E-01	2.000E+01	2	
	79.790	1.397E+00	0.000E-01	1.000E+02	3	

ALPHA = 5.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	5.000	6.743E-01	0.000E-01	3.000E+01	0	
	72.602	1.496E+00	0.000E-01	9.800E+01	3	

ALPHA = 10.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	10.000	1.621E+00	0.000E-01	4.000E+01	2	
	64.881	1.575E+00	0.000E-01	9.500E+01	2	

ALPHA = 15.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	15.000	1.239E+00	0.000E-01	5.000E+01	0	
	56.423	1.629E+00	0.000E-01	9.100E+01	3	

ALPHA = 20.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	20.000	1.396E+00	0.000E-01	6.000E+01	3	
	46.859	1.651E+00	0.000E-01	8.700E+01	3	

ALPHA = 25.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	25.000	1.521E+00	0.000E-01	7.000E+01	0	
	35.409	1.627E+00	0.000E-01	8.000E+01	3	

ALPHA = 30.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	19.987	1.515E+00	0.000E-01	7.000E+01	4	
	27.174	1.597E+00	0.000E-01	7.700E+01	-4	

ALPHA = 35.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	-3.357	1.198E+00	0.000E-01	5.200E+01	3	
	35.000	1.728E+00	0.000E-01	9.000E+01	0	

ALPHA = 40.000	BETA	P	A	Z	ITERATIONS	1 ROOTS (DATA SET 1)
	40.000	1.823E+00	0.000E-01	1.000E+02	3	

ALPHA = 45.000	BETA	P	A	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	45.000	1.918E+00	0.000E-01	1.100E+02	0	

		BETA	P	W	Z	ITERATIONS		
		115.000	3.409E+00	0.000E-01	1.800E+02	0		
ALPHA =	50.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		50.000	2.015E+00	0.000E-01	1.200E+02	2		
		110.000	3.225E+00	0.000E-01	1.800E+02	3		
ALPHA =	55.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		55.000	2.118E+00	0.000E-01	1.300E+02	0		
		105.000	3.081E+00	0.000E-01	1.800E+02	1		
ALPHA =	60.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		60.000	2.229E+00	0.000E-01	1.400E+02	2		
		100.000	2.972E+00	0.000E-01	1.800E+02	3		
ALPHA =	65.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		65.000	2.349E+00	0.000E-01	1.500E+02	0		
		95.000	2.891E+00	0.000E-01	1.800E+02	0		
ALPHA =	70.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		70.000	2.481E+00	0.000E-01	1.600E+02	3		
		90.000	2.836E+00	0.000E-01	1.800E+02	4		
ALPHA =	75.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		75.000	2.628E+00	0.000E-01	1.700E+02	0		
		85.000	2.803E+00	0.000E-01	1.800E+02	0		
ALPHA =	80.000	BETA	P	W	Z	ITERATIONS	0 ROOTS	(DATA SET 1)
ALPHA =	85.000	BETA	P	W	Z	ITERATIONS	2 ROOTS	(DATA SET 1)
		75.000	2.803E+00	0.000E-01	1.800E+02	0		
		85.000	2.978E+00	0.000E-01	1.900E+02	0		

ALPHA = 90.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	70.000	2.836E+00	0.000E-01	1.800E+02	3	
	90.000	3.190E+00	0.000E-01	2.000E+02	4	

ALPHA = 95.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	65.000	2.891E+00	0.000E-01	1.800E+02	0	
	95.000	3.433E+00	0.000E-01	2.100E+02	0	

ALPHA = 100.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	60.000	2.972E+00	0.000E-01	1.800E+02	2	
	100.000	3.715E+00	0.000E-01	2.200E+02	3	

ALPHA = 105.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	55.000	3.081E+00	0.000E-01	1.800E+02	1	
	105.000	4.044E+00	0.000E-01	2.300E+02	0	

ALPHA = 110.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	50.000	3.225E+00	0.000E-01	1.800E+02	2	
	110.000	4.434E+00	0.000E-01	2.400E+02	3	

ALPHA = 115.000	BETA	P	W	Z	ITERATIONS	2 ROOTS (DATA SET 1)
	45.000	3.409E+00	0.000E-01	1.800E+02	0	
	115.000	4.901E+00	0.000E-01	2.500E+02	0	

P = CTN(ALPHA), FOR CTN(ALPHA) .LT. 20

ALPHA RANGE = 0.000 TO 70.000 ALPHA INCREMENT SCAN = 5.000

BETA RANGE = 0.000 TO 0.000 BETA INCREMENT SCAN = 0.000

DATA SET 2 --- ALPHA AND BETA SCANS, WITH ALPHA SCANNED FOR EACH BETA
--- FOR THIS DATA SET, THE ROOTS ARE THE SCANNED VALUES
--- P,W,Z = FUNCTIONS OF ROOTS
--- IF (ABS(DENOM OF P,W, OR Z) .LE. 5.E-02), THEN PRINTED FUNCTION VALUE = ***, AND PRINTED ITERATIONS = -100
--- ALL ANGLES = DEGREES

BETA = 0.000	ALPHA	P	W	Z	ITERATIONS	15 ROOTS (DATA SET 2)
	0.000	***	0.000E-01	0.000E-01	-100	
	5.000	1.143E+01	0.000E-01	0.000E-01	0	
	10.000	5.671E+00	0.000E-01	0.000E-01	0	
	15.000	3.732E+00	0.000E-01	0.000E-01	0	
	20.000	2.747E+00	0.000E-01	0.000E-01	0	
	25.000	2.145E+00	0.000E-01	0.000E-01	0	
	30.000	1.732E+00	0.000E-01	0.000E-01	0	
	35.000	1.428E+00	0.000E-01	0.000E-01	0	
	40.000	1.192E+00	0.000E-01	0.000E-01	0	
	45.000	1.000E+00	0.000E-01	0.000E-01	0	
	50.000	8.391E-01	0.000E-01	0.000E-01	0	
	55.000	7.002E-01	0.000E-01	0.000E-01	0	
	60.000	5.774E-01	0.000E-01	0.000E-01	0	
	65.000	4.663E-01	0.000E-01	0.000E-01	0	
	70.000	3.640E-01	0.000E-01	0.000E-01	0	

P = H, WHERE H IS THE ROOT EQUATION OF DATA SET 1

ALPHA RANGE = 45.000 TO 45.000 ALPHA INCREMENT SCAN = 0.000

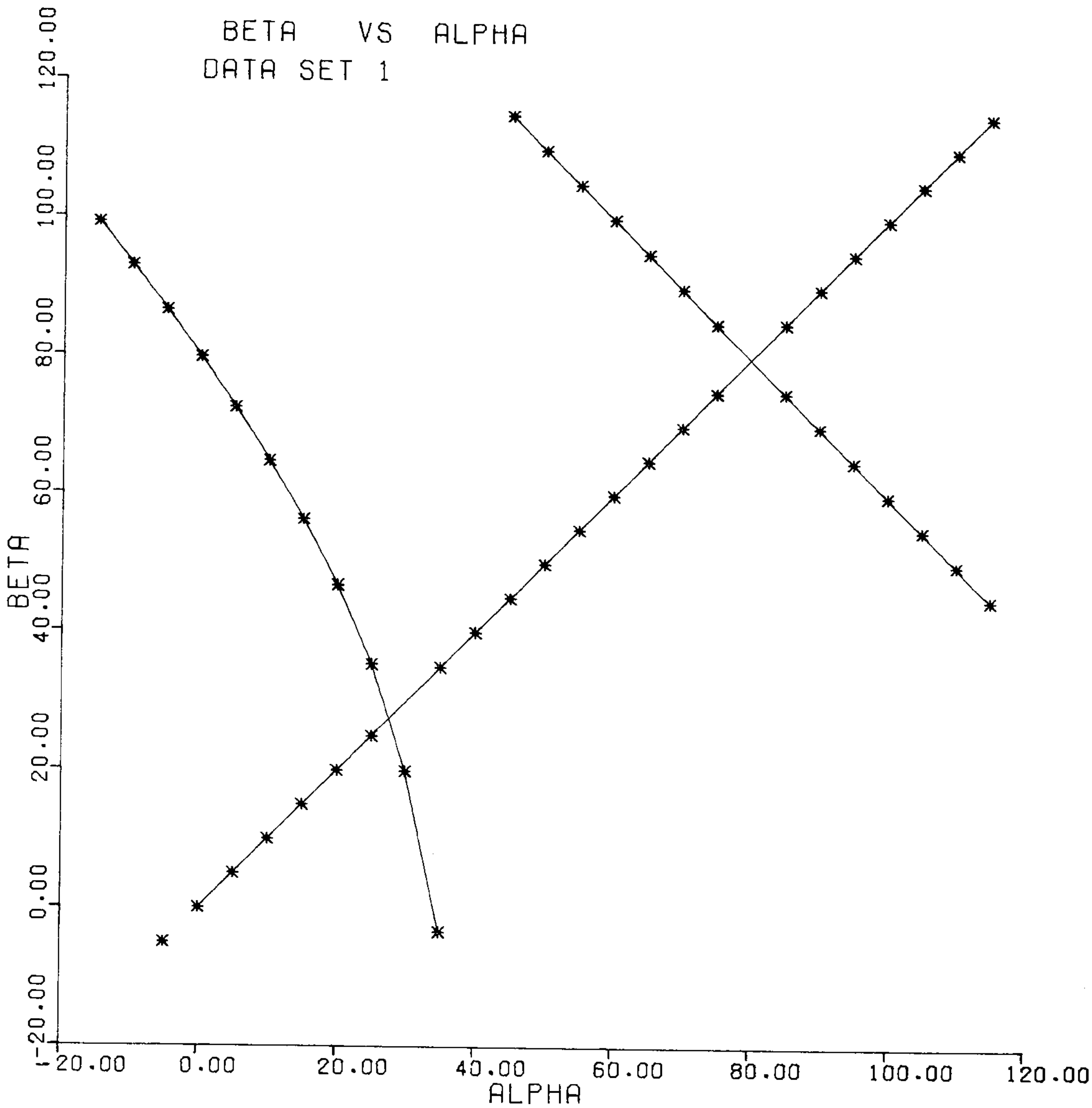
BETA RANGE = -20.000 TO 120.000 BETA INCREMENT SCAN = 5.000

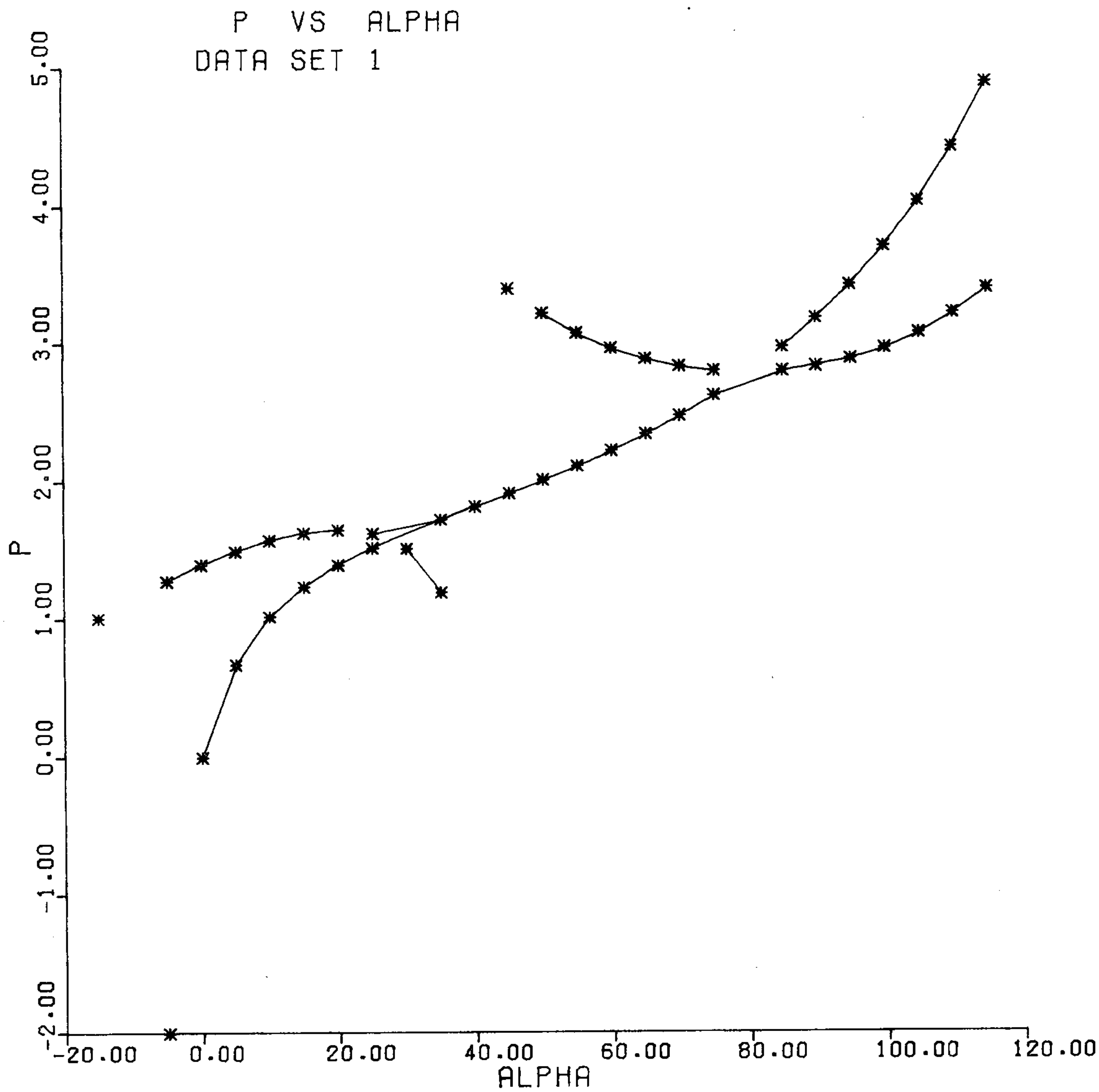
DATA SET 3 --- ALPHA AND BETA SCANS, WITH BETA SCANNED FOR EACH ALPHA
--- FOR THIS DATA SET, THE ROOTS ARE THE SCANNED VALUES
--- P,W,Z = FUNCTIONS OF ROOTS
--- IF (ABS(DEDOO OF P,V, OR Z) .LE. 0.E-01), THEN PRINTED FUNCTION VALUE = ***, AND PRINTED ITERATIONS = -100
--- ALL ANGLES = DEGREES

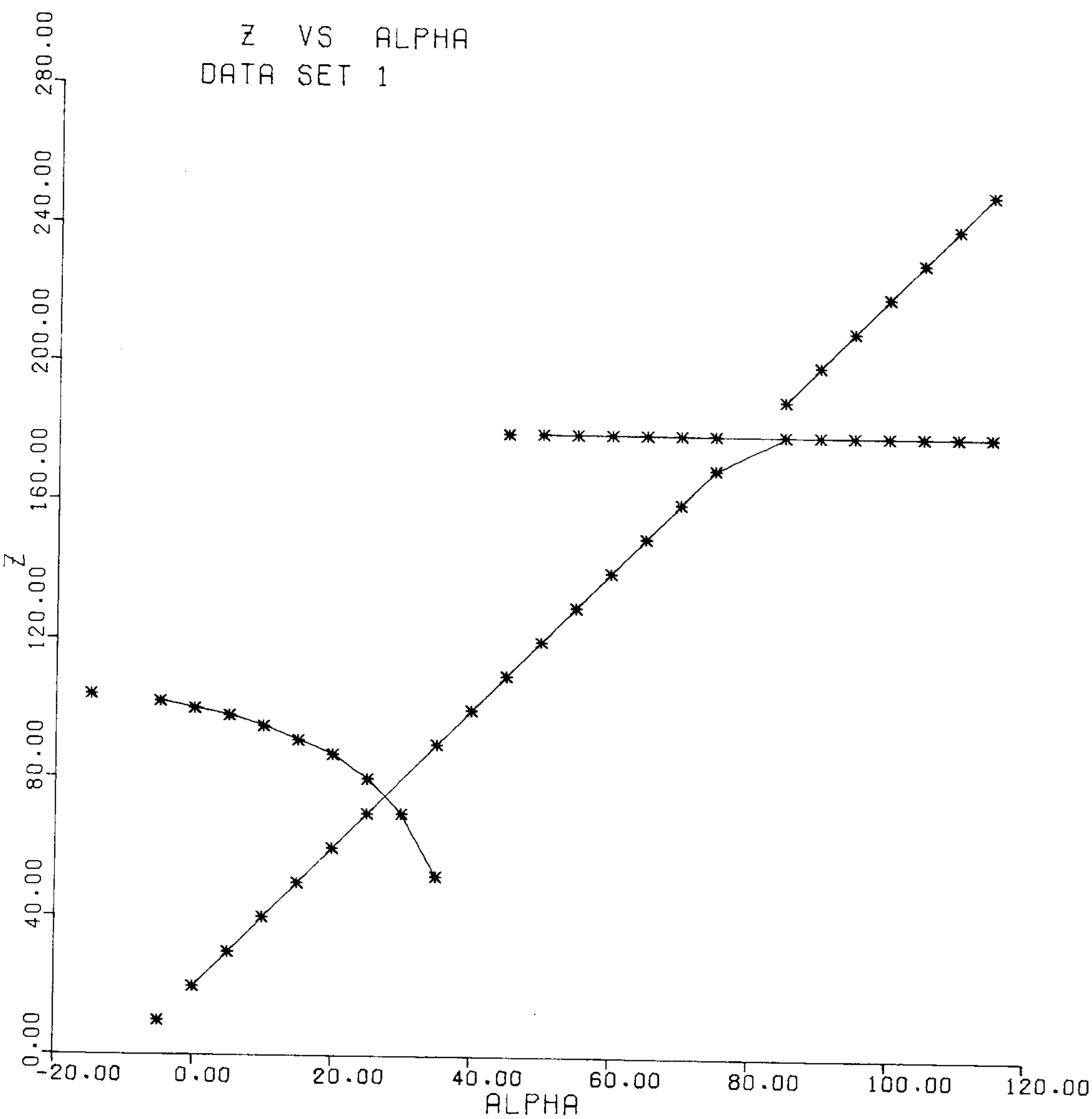
ALPHA = 45.000 — 29 ROOTS (DATA SET 3)

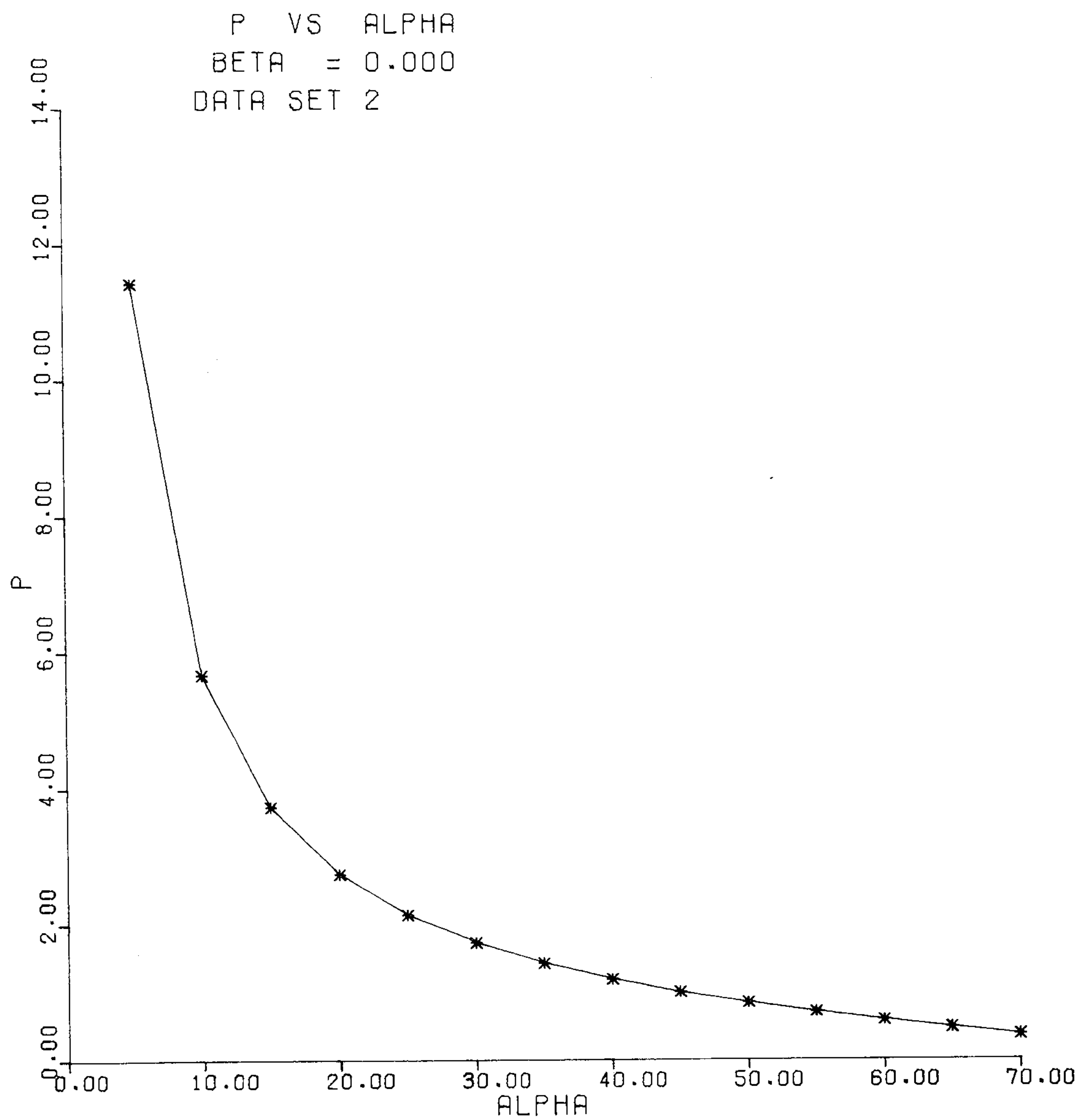
BETA	P	W	Z	ITERATIONS
-20.000	-7.966E-02	0.000E-01	0.000E-01	0
-15.000	-9.488E-02	0.000E-01	0.000E-01	0
-10.000	-1.042E-01	0.000E-01	0.000E-01	0
-5.000	-1.091E-01	0.000E-01	0.000E-01	0
0.000	-1.108E-01	0.000E-01	0.000E-01	0
5.000	-1.100E-01	0.000E-01	0.000E-01	0
10.000	-1.071E-01	0.000E-01	0.000E-01	0
15.000	-1.022E-01	0.000E-01	0.000E-01	0
20.000	-9.486E-02	0.000E-01	0.000E-01	0
25.000	-8.461E-02	0.000E-01	0.000E-01	0
30.000	-7.066E-02	0.000E-01	0.000E-01	0
35.000	-5.228E-02	0.000E-01	0.000E-01	0
40.000	-2.885E-02	0.000E-01	0.000E-01	0
45.000	0.000E-01	0.000E-01	0.000E-01	0
50.000	3.424E-02	0.000E-01	0.000E-01	0
55.000	7.335E-02	0.000E-01	0.000E-01	0
60.000	1.162E-01	0.000E-01	0.000E-01	0
65.000	1.611E-01	0.000E-01	0.000E-01	0
70.000	2.057E-01	0.000E-01	0.000E-01	0
75.000	2.469E-01	0.000E-01	0.000E-01	0
80.000	2.814E-01	0.000E-01	0.000E-01	0

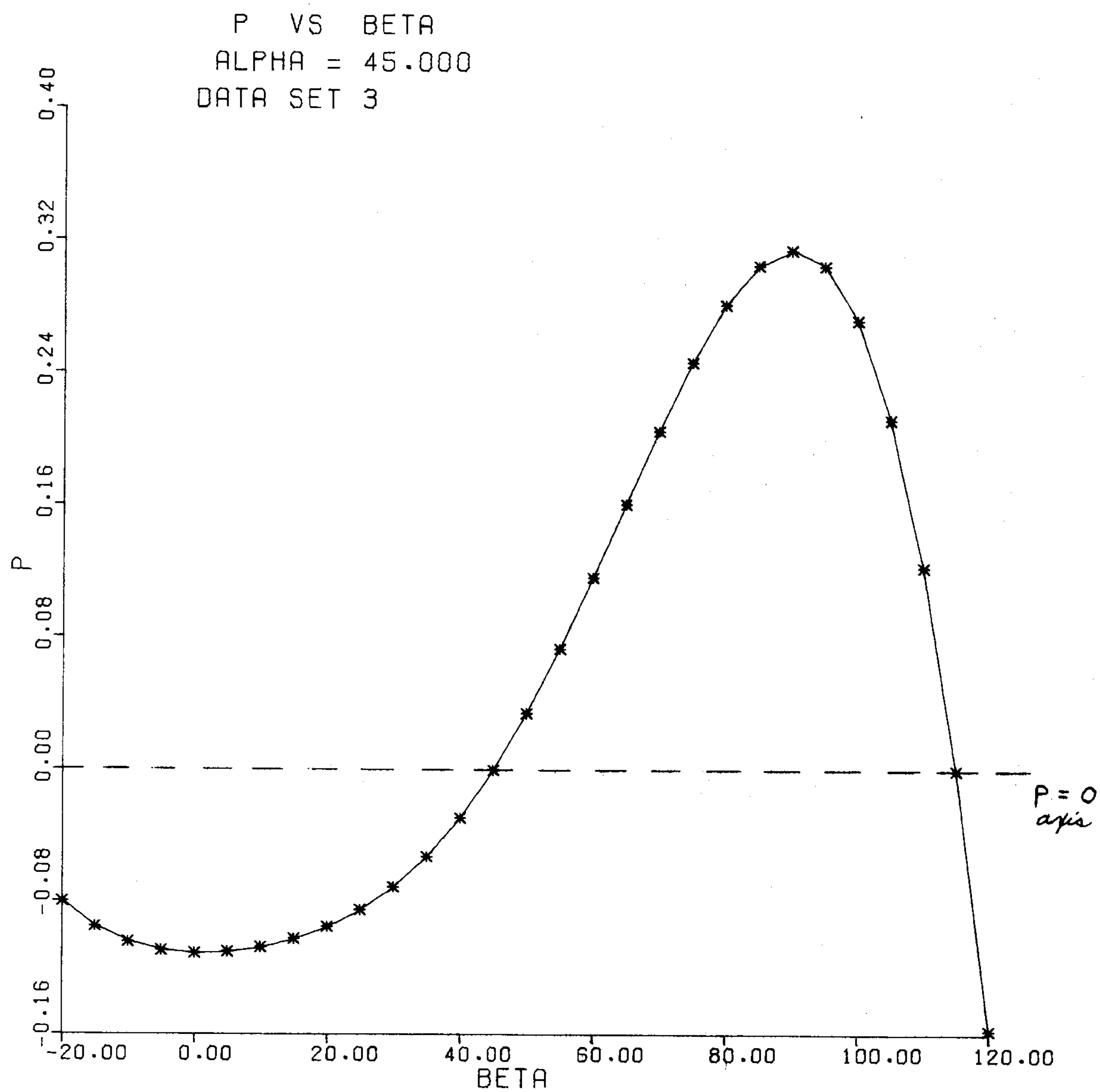
85.000	3.053E-01	0.000E-01	0.000E-01	0
90.000	3.145E-01	0.000E-01	0.000E-01	0
95.000	3.048E-01	0.000E-01	0.000E-01	0
100.000	2.720E-01	0.000E-01	0.000E-01	0
105.000	2.124E-01	0.000E-01	0.000E-01	0
110.000	1.226E-01	0.000E-01	0.000E-01	0
115.000	0.000E-01	0.000E-01	0.000E-01	0
120.000	-1.574E-01	0.000E-01	0.000E-01	0


BETA VS ALPHA
DATA SET 1
BETA
ALPHA
-20.00
0.00
20.00
40.00
60.00
80.00
100.00
120.00


P VS ALPHA
DATA SET 1
P
5.00
4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-20.00
0.00
20.00
40.00
60.00
80.00
100.00
120.00
ALPHA


Z VS ALPHA
DATA SET 1
Z
ALPHA
0.00
40.00
80.00
120.00
160.00
200.00
240.00
280.00
-20.00
0.00
20.00
40.00
60.00
80.00
100.00
120.00


P VS ALPHA
BETA = 0.000
DATA SET 2
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
P
0.00
10.00
20.00
30.00
40.00
50.00
60.00
70.00
ALPHA


P VS BETA
ALPHA = 45.000
DATA SET 3
P
BETA
P = 0 axis

APPENDIX B

I/O FOR SAMPLE PROGRAM 2

```
      SUBROUTINE FRONT
C
C
C***SAMPLE PROGRAM 2 INSERTED HERE DEMONSTRATES HOW PROGRAM MULTI
C***CAN BE USED TO GENERATE CROSS-SECTIONS OF A 3-DIMENSIONAL
C***SURFACE IN SPACE.  THE EQUATION OF THE SURFACE TO BE COMPUTED
C***IS    P = A + B**2.
C
C    --DATA SET 1 SCANS AND GRAPHS AGAINST ALPHA FOR EACH BETA.
C    --DATA SET 2 SCANS AND GRAPHS AGAINST BETA FOR EACH ALPHA.
C
C
C***THE SAMPLE PROGRAM WAS RUN WITH THE FOLLOWING DATA INPUT FILE:
C
C    10.0,-50.0,50.0,5.0,0.0,15.0,0.0,0.0,0,0,1,0,0,1
C    10.0,0.0,30.0,5.0,-25.0,25.0,0.0,0.0,0,0,1,0,0,2
C
C***THE FIRST INPUT LINE IS DATA SET 1, THE 2ND IS DATA SET 2.
C
C
C
C
C1...MULTI-PURPOSE PROGRAM FOR COMPUTING AND GRAPHING
C     ROOTS AND/OR VALUES FOR ANY REAL FUNCTION
C     --BY  RUSSEL BRANTMAN
C
C
      COMMON/FLAGS/ISCAN,IH,IP,IW,IZ,ISG,L,HJOIN,PJOIN,WJOIN,ZJOIN,
     &          HVER,PVER,WVER,ZVER,XHOR
C
      COMMON/SUB/DA,AI,AF,DB,BI,BF,DC,E,NX,N,ISKIP,H,A,B,DHA,DHB,
     &              PNUME,PDNOM,WNUME,WDNOM,ZNUME,ZDNOM
C
C         SEQUENCE CONTROL
      IF(N.EQ.-2) GO TO 40
      IF(N.GT.-2) GO TO 20
C
C
C
C2...DATA SPECIFICATION   (SEE DOCUMENTATION)
C
C  L IS THE DATA SET COUNTER WHICH IS INITIALIZED TO 1 AND AUTOMATICALLY
C  INCREMENTED BY THE PROGRAM.   L MAY BE TESTED FOR TRANSFERRING TO THE
C  INFORMATION APPROPRIATE TO THE DIFFERENT DATA SETS.
C  (ON OUTPUT, EACH DATA SET WILL START ON A NEW PAGE)
C
C
C2-1...SPECIFY ANY SPECIFICATION STATEMENTS NEEDED FOR YOUR MATERIAL
C
C
C2-2...SPECIFY ANY DESIRED READ STATEMENTS
C
C
C2-3...SPECIFY ANY DESIRED WRITE STATEMENTS
C
      IF(L.EQ.2) GO TO 60
      PRINT 500
500   FORMAT(//'   SAMPLE PROGRAM 2')
60    PRINT 546
546   FORMAT(//'      P = A + B**2   '//)
C
```

```
C
C
C3...EQUATION SPECIFICATION  (SEE DOCUMENTATION)
C
C
C3-1...SPECIFY YOUR CONSTANTS
C
C   DO NOT REMOVE STATEMENT 10 (RAD=)
10        RAD = 3.1415927/180.0
C
C
11        RETURN
C
C
C3-2...SPECIFY H AND ANY COMPUTATIONS DESIRED DURING SCANNING PHASE
C
C   DO NOT REMOVE STATEMENT 20 OR THE NEXT STATEMENT (RA=,RB=)
20        RA = RAD*A
          RB = RAD*B
C   TO SKIP OVER ANY SCANNED VALUES (OR ANY ROOT REGIONS),
C   SET ISKIP = 1 FOR THOSE VALUES AND THEN GO TO STATEMENT 22
C
C
21        IF(N)22,40,30
22        RETURN
C
C
C3-3...SPECIFY NEWTON-RAPHSON VARIABLE DHY,  WHERE Y = A OR B
C       (IF DHY COMPUTED FROM RY,  SET DHY = RAD*DHY)
C
30        CONTINUE
C
C
31        RETURN
C
C
C3-4...SPECIFY YNUME AND YDNOM OF FUNCTION Y,  Y = P,W,Z
C       (DEFAULTS ARE YNUME=0 AND YDNOM=1)
C
40        CONTINUE
C
          PNUME = A + B**2
C
41        RETURN
C
C
C3-5...FIXED GRAPH DATA
C
          DATA XHOR/0.0/
          DATA HVER,PVER,WVER,ZVER/4*0.0/
          DATA HJOIN/2.5/
          DATA PJOIN/2.5/
          DATA WJOIN/2.5/
          DATA ZJOIN/2.5/
C
C
C...END OF INPUT TO MAIN PROGRAM
C   (DON'T FORGET YOUR DATA CARD FOR THE DATA INPUT VARIABLES --
C    DA,AI,AF,DB,BI,BF,DC,E,NX,IH,IP,IW,IZ,ISG)
C
          END
```

SAMPLE PROGRAM 2

P = A + B**2

ALPHA RANGE = -50.000 TO 50.000 ALPHA INCREMENT SCAN = 10.000

BETA RANGE = 0.000 TO 15.000 BETA INCREMENT SCAN = 5.000

DATA SET 1 --- ALPHA AND BETA SCANS, WITH ALPHA SCANNED FOR EACH BETA
--- FOR THIS DATA SET, THE ROOTS ARE THE SCANNED VALUES
--- P,W,Z = FUNCTIONS OF ROOTS
--- IF (ABS(DENOM OF P,W, OR Z) .LE. 0.E-01), THEN PRINTED FUNCTION VALUE = ***, AND PRINTED ITERATIONS = -100
--- ALL ANGLES = DEGREES

BETA = 0.000 — 11 ROOTS (DATA SET 1)

ALPHA	P	W	Z	ITERATIONS
-50.000	-5.000E+01	0.000E-01	0.000E-01	0
-40.000	-4.000E+01	0.000E-01	0.000E-01	0
-30.000	-3.000E+01	0.000E-01	0.000E-01	0
-20.000	-2.000E+01	0.000E-01	0.000E-01	0
-10.000	-1.000E+01	0.000E-01	0.000E-01	0
0.000	0.000E-01	0.000E-01	0.000E-01	0
10.000	1.000E+01	0.000E-01	0.000E-01	0
20.000	2.000E+01	0.000E-01	0.000E-01	0
30.000	3.000E+01	0.000E-01	0.000E-01	0
40.000	4.000E+01	0.000E-01	0.000E-01	0
50.000	5.000E+01	0.000E-01	0.000E-01	0

BETA = 5.000 — 11 ROOTS (DATA SET 1)

ALPHA	P	W	Z	ITERATIONS
-50.000	-2.500E+01	0.000E-01	0.000E-01	0
-40.000	-1.500E+01	0.000E-01	0.000E-01	0
-30.000	-5.000E+00	0.000E-01	0.000E-01	0
-20.000	5.000E+00	0.000E-01	0.000E-01	0
-10.000	1.500E+01	0.000E-01	0.000E-01	0
0.000	2.500E+01	0.000E-01	0.000E-01	0
10.000	3.500E+01	0.000E-01	0.000E-01	0
20.000	4.500E+01	0.000E-01	0.000E-01	0

30.000	5.500E+01	0.000E-01	0.000E-01	0
40.000	6.500E+01	0.000E-01	0.000E-01	0
50.000	7.500E+01	0.000E-01	0.000E-01	0

BETA = 10.000 11 ROOTS (DATA SET 1)

ALPHA	P	W	Z	ITERATIONS
-50.000	5.000E+01	0.000E-01	0.000E-01	0
-40.000	6.000E+01	0.000E-01	0.000E-01	0
-30.000	7.000E+01	0.000E-01	0.000E-01	0
-20.000	8.000E+01	0.000E-01	0.000E-01	0
-10.000	9.000E+01	0.000E-01	0.000E-01	0
0.000	1.000E+02	0.000E-01	0.000E-01	0
10.000	1.100E+02	0.000E-01	0.000E-01	0
20.000	1.200E+02	0.000E-01	0.000E-01	0
30.000	1.300E+02	0.000E-01	0.000E-01	0
40.000	1.400E+02	0.000E-01	0.000E-01	0
50.000	1.500E+02	0.000E-01	0.000E-01	0

BETA = 15.000 11 ROOTS (DATA SET 1)

ALPHA	P	W	Z	ITERATIONS
-50.000	1.750E+02	0.000E-01	0.000E-01	0
-40.000	1.850E+02	0.000E-01	0.000E-01	0
-30.000	1.950E+02	0.000E-01	0.000E-01	0
-20.000	2.050E+02	0.000E-01	0.000E-01	0
-10.000	2.150E+02	0.000E-01	0.000E-01	0
0.000	2.250E+02	0.000E-01	0.000E-01	0
10.000	2.350E+02	0.000E-01	0.000E-01	0
20.000	2.450E+02	0.000E-01	0.000E-01	0
30.000	2.550E+02	0.000E-01	0.000E-01	0
40.000	2.650E+02	0.000E-01	0.000E-01	0
50.000	2.750E+02	0.000E-01	0.000E-01	0

P = A + B**2

ALPHA RANGE = 0.000 TO 30.000 ALPHA INCREMENT SCAN = 10.000

BETA RANGE = -25.000 TO 25.000 BETA INCREMENT SCAN = 5.000

DATA SET 2 --- ALPHA AND BETA SCANS, WITH BETA SCANNED FOR EACH ALPHA
--- FOR THIS DATA SET, THE ROOTS ARE THE SCANNED VALUES
--- P,W,Z = FUNCTIONS OF ROOTS
--- IF (ABS(DENOM OF P,W, OR Z) .LE. 0.E-01), THEN PRINTED FUNCTION VALUE = ***, AND PRINTED ITERATIONS = -100
--- ALL ANGLES = DEGREES

ALPHA = 0.000	BETA	P	W	Z	ITERATIONS	11 ROOTS (DATA SET 2)
	-25.000	6.250E+02	0.000E-01	0.000E-01	0	
	-20.000	4.000E+02	0.000E-01	0.000E-01	0	
	-15.000	2.250E+02	0.000E-01	0.000E-01	0	
	-10.000	1.000E+02	0.000E-01	0.000E-01	0	
	-5.000	2.500E+01	0.000E-01	0.000E-01	0	
	0.000	0.000E-01	0.000E-01	0.000E-01	0	
	5.000	2.500E+01	0.000E-01	0.000E-01	0	
	10.000	1.000E+02	0.000E-01	0.000E-01	0	
	15.000	2.250E+02	0.000E-01	0.000E-01	0	
	20.000	4.000E+02	0.000E-01	0.000E-01	0	
	25.000	6.250E+02	0.000E-01	0.000E-01	0	

ALPHA = 10.000	BETA	P	W	Z	ITERATIONS	11 ROOTS (DATA SET 2)
	-25.000	6.350E+02	0.000E-01	0.000E-01	0	
	-20.000	4.100E+02	0.000E-01	0.000E-01	0	
	-15.000	2.350E+02	0.000E-01	0.000E-01	0	
	-10.000	1.100E+02	0.000E-01	0.000E-01	0	
	-5.000	3.500E+01	0.000E-01	0.000E-01	0	
	0.000	1.000E+01	0.000E-01	0.000E-01	0	
	5.000	3.500E+01	0.000E-01	0.000E-01	0	
	10.000	1.100E+02	0.000E-01	0.000E-01	0	

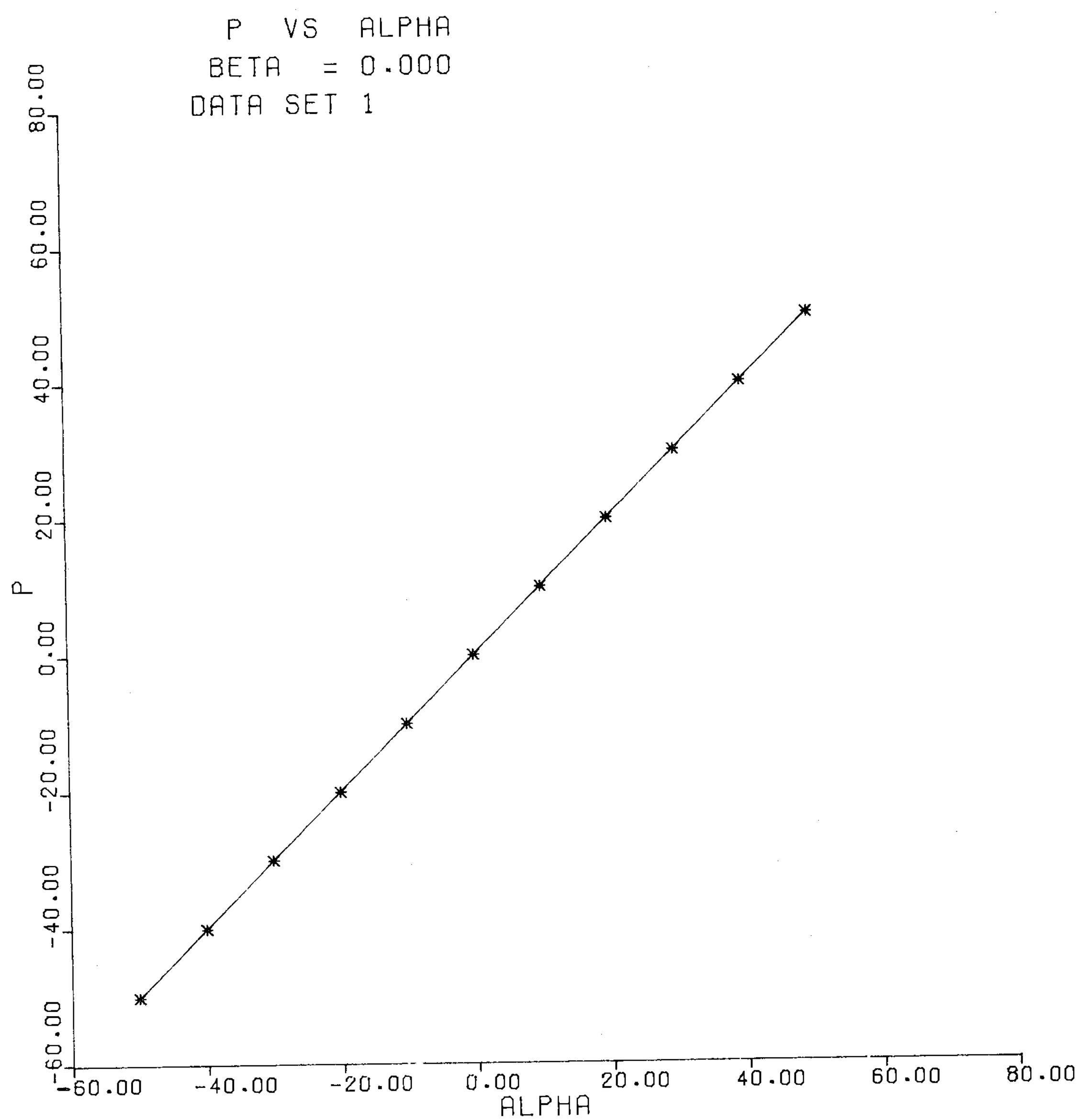
15.000	2.350E+02	0.000E-01	0.000E-01	0
20.000	4.100E+02	0.000E-01	0.000E-01	0
25.000	6.350E+02	0.000E-01	0.000E-01	0

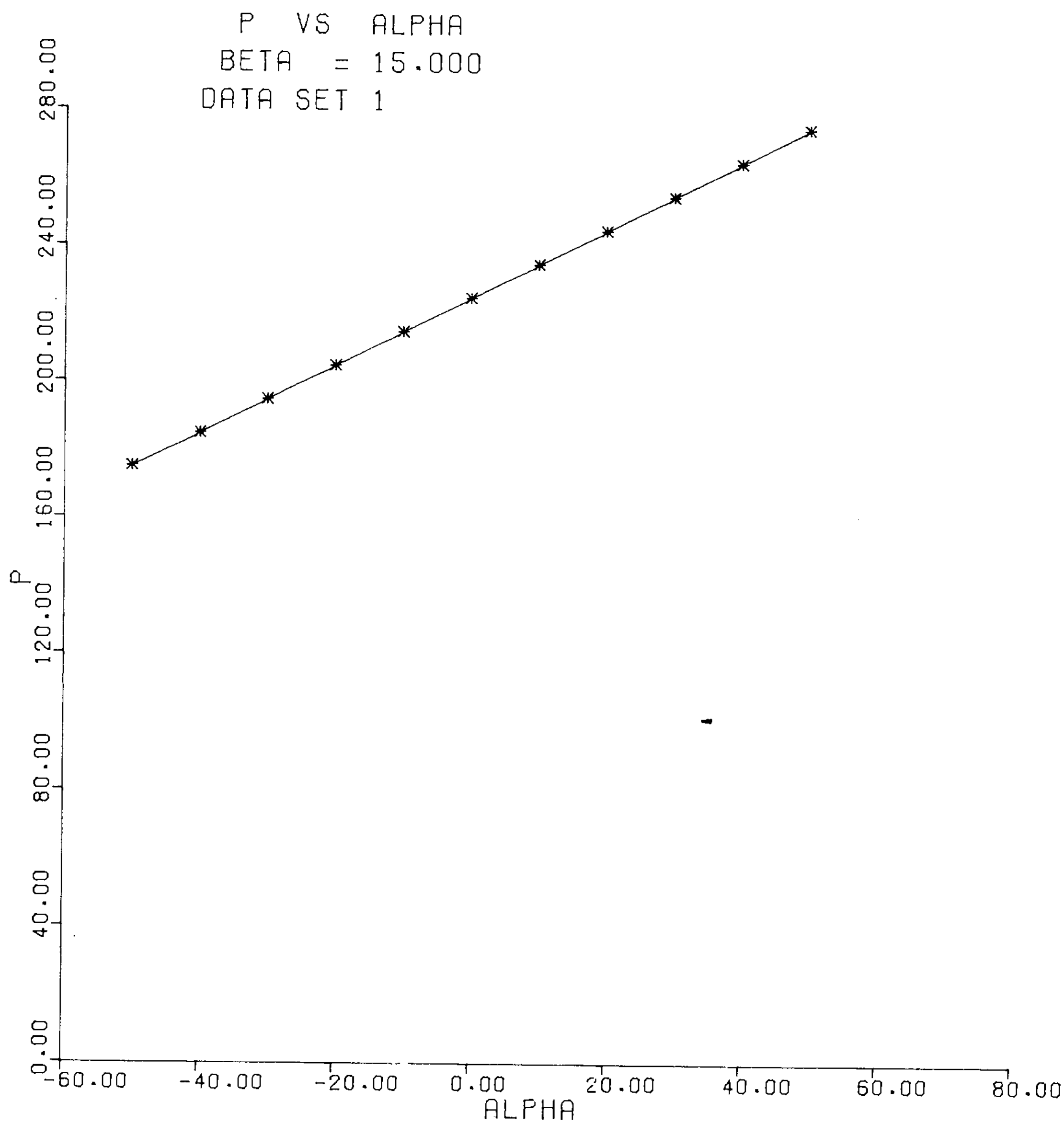
ALPHA = 20.000 11 ROOTS (DATA SET 2)

BETA	P	W	Z	ITERATIONS
-25.000	6.450E+02	0.000E-01	0.000E-01	0
-20.000	4.200E+02	0.000E-01	0.000E-01	0
-15.000	2.450E+02	0.000E-01	0.000E-01	0
-10.000	1.200E+02	0.000E-01	0.000E-01	0
-5.000	4.500E+01	0.000E-01	0.000E-01	0
0.000	2.000E+01	0.000E-01	0.000E-01	0
5.000	4.500E+01	0.000E-01	0.000E-01	0
10.000	1.200E+02	0.000E-01	0.000E-01	0
15.000	2.450E+02	0.000E-01	0.000E-01	0
20.000	4.200E+02	0.000E-01	0.000E-01	0
25.000	6.450E+02	0.000E-01	0.000E-01	0

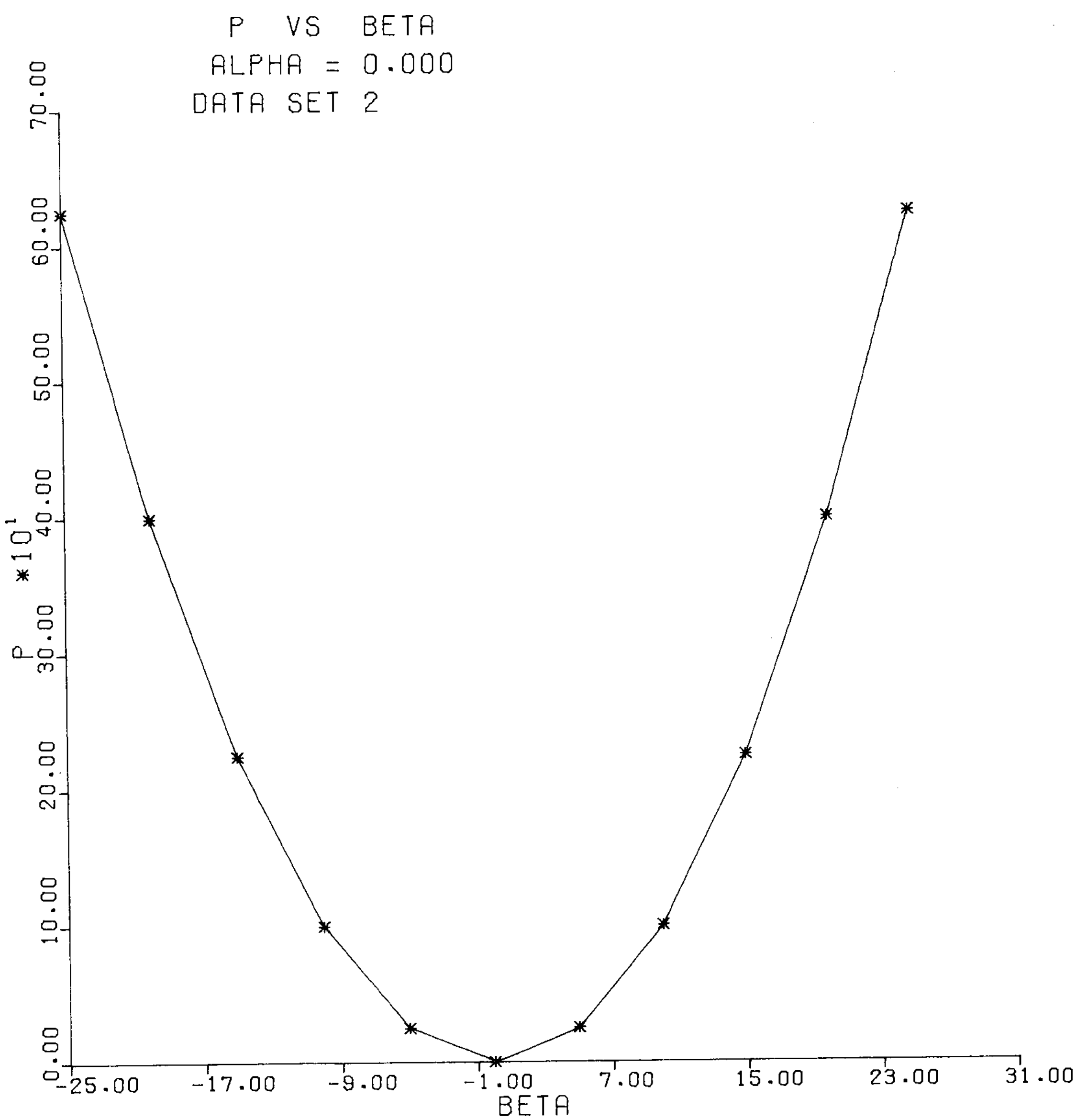
ALPHA = 30.000 11 ROOTS (DATA SET 2)

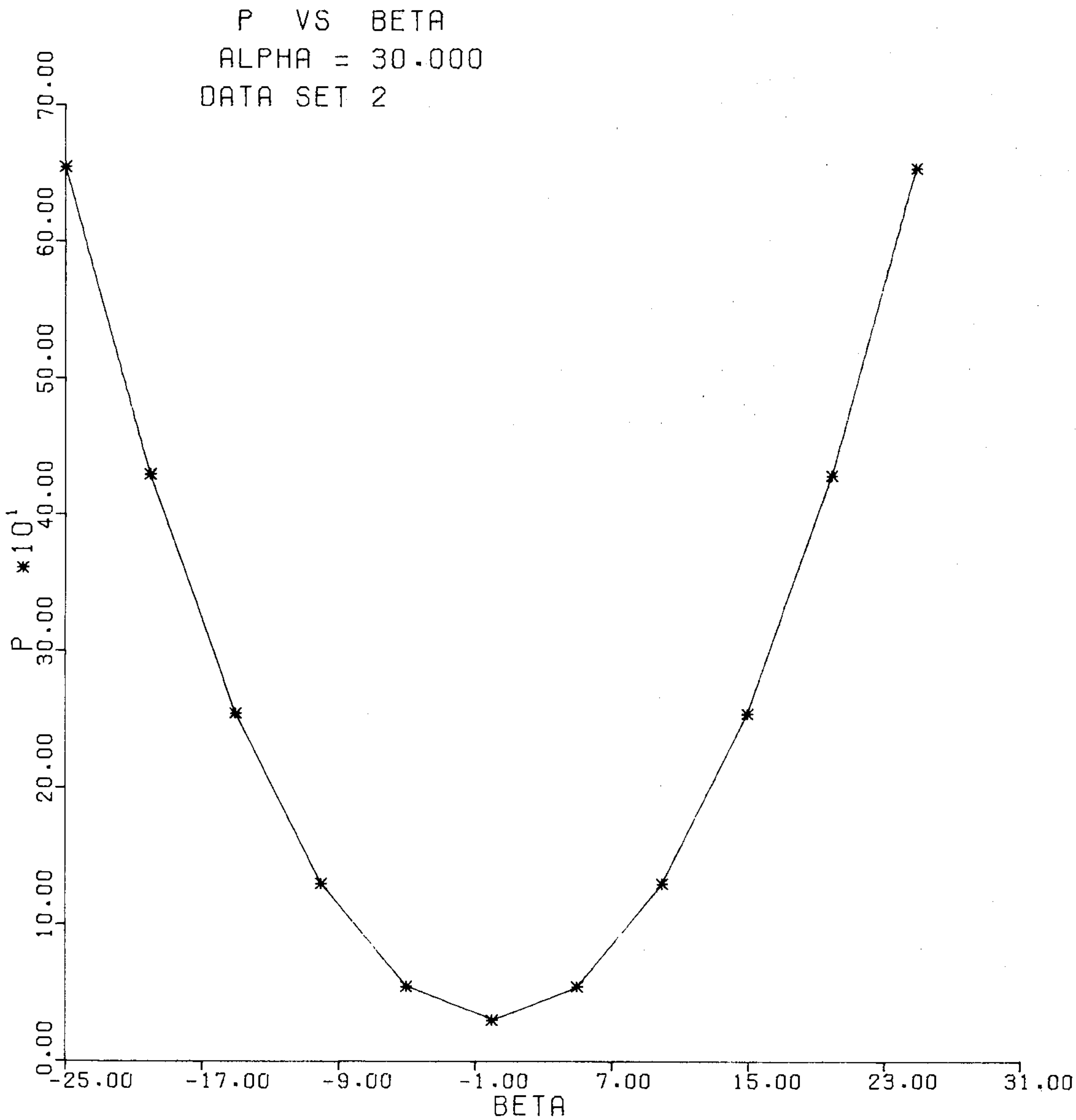
BETA	P	W	Z	ITERATIONS
-25.000	6.550E+02	0.000E-01	0.000E-01	0
-20.000	4.300E+02	0.000E-01	0.000E-01	0
-15.000	2.550E+02	0.000E-01	0.000E-01	0
-10.000	1.300E+02	0.000E-01	0.000E-01	0
-5.000	5.500E+01	0.000E-01	0.000E-01	0
0.000	3.000E+01	0.000E-01	0.000E-01	0
5.000	5.500E+01	0.000E-01	0.000E-01	0
10.000	1.300E+02	0.000E-01	0.000E-01	0
15.000	2.550E+02	0.000E-01	0.000E-01	0
20.000	4.300E+02	0.000E-01	0.000E-01	0
25.000	6.550E+02	0.000E-01	0.000E-01	0


P VS ALPHA
BETA = 0.000
DATA SET 1
P
80.00
60.00
40.00
20.00
0.00
-20.00
-40.00
-60.00
-60.00
-40.00
-20.00
0.00
20.00
40.00
60.00
80.00
ALPHA



P VS ALPHA
BETA = 15.000
DATA SET 1
P
0.00
40.00
80.00
120.00
160.00
200.00
240.00
280.00
-60.00
-40.00
-20.00
0.00
20.00
40.00
60.00
80.00
ALPHA


P VS BETA
ALPHA = 0.000
DATA SET 2
P *10^1
0.00
10.00
20.00
30.00
40.00
50.00
60.00
70.00
-25.00
-17.00
-9.00
-1.00
7.00
15.00
23.00
31.00
BETA


P VS BETA
ALPHA = 30.000
DATA SET 2
P *10¹
0.00
10.00
20.00
30.00
40.00
50.00
60.00
70.00
-25.00
-17.00
-9.00
-1.00
7.00
15.00
23.00
31.00
BETA