AD-A080 798 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 20/4
A GENERAL PROCEDURE FOR OBTAINING VELOCITY VECTOR FROM A SYSTEM--ETC(U)
JUL 79 D ADLER, R P SHREEVE
UNCLASSIFIED NPS67-79-007 NL

END
DATE
FILMED
3 80
DDC

NPS67-79-007

NAVAL POSTGRADUATE SCHOOL

Monterey, California

A General Procedure for Obtaining Velocity Vector from a System of High Response Impact Pressure Probes

D. Adler and R. P. Shreeve

July 1979

Approved for public release; distribution unlimited

Prepared for:

Naval Air Systems Command
Washington, DC

and

Office of Naval Research
Arlington, VA

THIS DOCUMENT IS BEST QUALITY PRACTICABLE.
THE COPY FURNISHED TO DDC CONTAINED A
SIGNIFICANT NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

NAVAL POSTGRADUATE SCHOOL

Monterey, California

Rear Admiral T. F. Dedman
Superintendent

Jack R. Borsting
Provost

The work reported herein was supported by the Naval Air Systems Command, Washington, DC. and Office of Naval Research, Arlington, VA.

This report was prepared by:

D. Adler, Visiting Professor of Aeronautics

R. P. Shreeve, Associate Professor of Aeronautics

Reviewed by:

M. F. PLATZER, Chairman
Department of Aeronautics

Released by:

W. M. TOLLES
Dean of Research

DISCLAIMER NOTICE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NPS67-79-007
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): A General Procedure for Obtaining Velocity Vector from a System of High Response Impact Pressure Probes
5. TYPE OF REPORT & PERIOD COVERED: Technical Report, Jul[redacted]78 — Jul[redacted]79
6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): D. Adler [redacted] R. P. Shreeve
8. CONTRACT OR GRANT NUMBER(s): 61153N; N00019-79-WR-91115
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School, Monterey, CA 93940
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Air Systems Command, Washington, DC 20361
12. REPORT DATE: Jul[redacted]79
13. NUMBER OF PAGES: [redacted] 84
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 12 89
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:
16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; distribution unlimited.

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

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Flow Measurements

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A technique to measure a high frequency, repetitively pulsating flow filed is presented. Two impact tube probes and a Kiel probe, all with pressure transducers mounted in their tips are used. Five readings are required to identify a velocity vector at a point.
In this report the technique, the numerical procedure and the computer program used are described.

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

251 450

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

A General Procedure for Obtaining Velocity Vector From a System of High Response Impact Pressure Probes

by

D. Adler and R. P. Shreeve

Contents

Section	Page
Notation	1
Notation of PENPTS	2
Notation of INTSCS	3
Notation of main program VDR	6
Introduction	10
Mathematical model	15
Evaluation of α and ϕ (method)	21
Evaluation of the intersection point coordinates (numerical procedures)	26
Evaluation of the velocity vector and pressures from the probe signals	43
Convergence and accuracy	48
Evaluation of the computation time	50
Input	51
Output	52
Conclusions	53
Appendix 1 (VDR)	56
Appendix 2 (SWVDR)	64
Appendix 3 (WVDR)	72

Notation

Cp - Pressure coefficient defined in eq. (3)

α - Yaw angle relative to laboratory space

ϕ - Pitch angle relative to laboratory space

P_t - Total pressure

P_s - Static Pressure

α_{rp} - Yaw angle relative to probe axis

ϕ_{rp} - Pitch angle relative to probe axis

P_p - Pressure indicated by the probe

α_p - Yaw setting of the probe

ϕ_p - Pitch setting of the probe

Subscripts

I, II, III, IV } relating to probe positions shown in figure 3

A relating to A type probe

B relating to B type probe

Lo lowest value in an array

rp relative to the probe axis

Up highest value in an array

Notation of PENPTS

DD - denominator in the subsequent line of the program

HT - penetration height (Cp of a calibration surface)

IR - a flag with a value of either 1 or 2 to identify first or second penetration point. This flag also controls return of values to calling program or subroutine

IB - a flag indicating the location of the last rectangle in the band checked for a penetration point. It prevents repetition of the scanning in the 103 Do loop

NX - number of X values in the calibration surface grid ($\propto$ values of the calibration surface)

NY - number of Y values in the calibration surface grid ([illegible] values of the calibration surface)

X(I) - X values in the grid

XM - value of X at point B (fig. 5)

XR(I) - two values of X returned by the subroutine as results

XS - currently calculated value of X which is the present result and is later stored in XR(I)

Y(J) - Y values in the calibration grid

YC - intermediate value used in the subsequent line of the program

YG - the Y position of the penetrating line*

Z(I.J)- Values of the Z = Cp calibration surface above grid points

ZX - value of Z either at point A or point C (see figs. 5, 6 and 7)

ZM - value of Z at point B

---

* It should be pointea out here that the penetrating line is parallel to the X axis at a height HT = Cp above the X,Y plane and at a distance YG from the X axis on the X,Y plane

Notation of INTSCS

AA - constant defined in eq. 8

AB - constant defined in eq. 9

ARIN - ordinate at which scanning starts

ARLO - lowest ordinate of calibration surface (as a condition it must be identical for all calibration surfaces used)

ARM1 - ordinate measured relative to probe 1 (the probe belonging to the first curve Cp = const.)

ARM2 - ordinate measured relative to probe 2 (the probe belonging to the second curve Cp = const)

ARP1 - setting ordinate of probe 1

ARP2 - setting ordinate of probe 2

ARRES (1) - abscissa value returned to INTSCS from PENPTS

ARUP - highest ordinate of calibration surface (as a condition it must be identical for all calibration surfaces used)

AR2R - ordinate relative to laboratory space

AR2RJ - ordinate relative to laboratory space of previous scan

BA - constant defined in eq. 8

BB - constant defined in eq. 9

DAR - ordinate step for scanning

HT1 - penetration height of calibration surface of probe 1 (Cp of probe 1)

HT2 - penetration height of calibration surface of probe 2 (Cp of probe 2)

IEPS - a flag which is equal to 2 when the calculation is carried out as shown in fig. 11, otherwise it is equal to 1. (This information is required in the main program and is returned to it)

ISL - a flag which is equal to 10 if one of the left hand side results returned from PENPTS is 1000.0 (i.e., is not a penetration point). Otherwise it is equal to 1

ISR - a flag which is equal to 10 if one of the right hand side results returned from PENPTS is 1000.0 (i.e., is not a penetration point). Otherwise it is equal to 1

K - a flag indicating whether the first or the second intersection is evaluated

N1 - number of abscissas in calibration surface grid (fig. 4)

N2 - number of Ordinates in calibration surface grid (fig. 4)

RES1(K) - evaluated and returned abscissa of intersection

RES2(K) - evaluated and returned ordinate of intersection

R1 - radius in fig. 11 for probe 1

R2 - radius in fig. 11 for probe 2

RAIL - abscissa of point AIL in fig. 9 or 10

RAIR - abscissa of point AIR in fig. 9 or 10

RAJL - abscissa of point AJL in fig. 9 or 10

RAJR - abscissa of point AJR in fig. 9 or 10

RBIL - abscissa of point BIL in fig. 9 or 10

RBIR - abscissa of point BIR in fig. 9 or 10

RBJL - abscissa of point BJL in fig. 9 or 10

RBJR - abscissa of point BJR in fig. 9 or 10

X1(I) - abscissas of calibration surface of probe 1 or its transformation as determined by calling statement in main program

X2(I) - abscissas of calibration surface of probe 2 or its transformation as determined by calling statement in main program

X3(I) - abscissas of calibration surface of probe 1 or its transformation as determined by calling statement in main program

X4(I) - abscissas of calibration surface of probe 2 or its transformation as determined by calling statement in main program

Y1(I) - ordinates matching X1(I)

Y2(I) - ordinates matching X2(I)

Y3(I) - ordinates matching X3(I)

Y4(I) - ordinates matching X4(I)

Z1(I,J) - values of Cp_{rp} for X1(I) and Y1(I)

Z2(I,J) - values of Cp_{rp} for X2(I) and Y2(I)

Z3(I,J) - values of Cp_{rp} for X3(I) and Y3(I)

Z4(I,J) - values of Cp_{rp} for X4(I) and Y4(I)

Notation of Main Program VDR

A - variables in library routine IXCLOK used to evaluate computation times

ALF - values of α returned to VDR from INTSCS

ALFA - α input when experiment simulation is carried out

ALFC - α_{rpIII}

ALFCN - new value of α_{rpIII} for next iteration

ALFD - α_{rpIV}

ALFDN - new value of α_{rpIV} for next iteration

ALF1 - α_I

ALF2 - α_{II}

ALF3 - α_I

ALF4 - α_{III}

AM - mach number

CPA - Cp_{rpI}

CPB - Cp_{rpII}

CPC - Cp_{rpIII}

CPD - Cp_{rpIV}

EPSPS - relative difference between present static pressure and its value in the previous iteration

EPSPSG - convergence criterion (on the static pressure)

FALF - final α value (at convergence)

FPHI - final ϕ value (at convergence)

IA50 - a flag in IXCLOK

IA60 - a flag in IXCLOK

ICOPS - number of PS corrections carried out to ensure the Cp's are in calibration range

IEPS - a flag governing the convergence criterion returned from INTSCS

IEPS1 - a flag governing the convergence criterion for probes I and II

IEPS2 - a flag governing the convergence criterion for probes I and III

IIT - number of iterations on p_s

ISCAN - number of static pressure scans from initial static pressure guess upwards

IXCLOK - system subroutine for computation time evaluation

NOCOPS - a flag which when equal to 1 causes static pressure corrections to be carried out such that Cp's are always inside calibration range, and when equal to 2 cause these corrections to be skipped

NOSIM - a flag which when equal to 1 causes the program to simulate velocity measurement experiments and when equal to 2 causes the program to reduce measured data

NX - number of calibration surface matrix abscissa's, α values (must be identical for all calibration surface matrices used)

NY - number of calibration surface matrix ordinates, ϕ values (must be identical for all calibration surface matrices used)

PDYN - dynamic pressure

PHI - value of ϕ returned to VDR from INTSCS

PHIB - ϕ_{rpII}

PHII - ϕ input when experiment simulation is carried out

PHI1 - ϕ_{rpI}

PHI2 - ϕ_{rpII}

PHI3 - ϕ_{rpI}

PHI4 - ϕ_{rpIII}

PPA - signal of probe in position I

PPB - signal of probe in position II

PPC - signal of probe in position III

PPC - signal of probe in position IV

PS - static pressure

PSIN - initial guess of static pressure

PSN - new value of static pressure for next iteration

PSC - corrected static pressure

PST - memorized corrected static pressure

PT - total pressure measured

RELXPS - relaxation factor for convergence on the static pressure

VIRTIM - virtual computation time

XA - abscissas of calibration surface matrix of probe A

XAX - transformed abscissas of calibration surface matrix of probe A

XB - abscissas of calibration surface matrix of probe B

XBX - transformed abscissas of calibration surface matrix of probe B

XPC - yaw setting of position III

XPD - yaw setting of position IV

XRIN - α value at which scanning starts

XRLU - lowest α value of calibration surface (as a condition it must be identical for probes A and B)

XRUP - highest α value of calibration surface (as a condition it must be identical for probes A and B)

YA - ordinates of calibration surface matrix of probe A

YAX - transformed ordinates of calibration surface matrix of probe A

YB - ordinates of calibration surface matrix of probe B

YBX - transformed ordinates of calibration surface matrix of probe B

YPA - pitch setting of probe A

YPB - pitch setting of probe B

YRIN - ϕ value at which scanning starts

YRLO - lowest ϕ value of calibration surface (as a condition it must be identical for probes A and B)

YRUP - highest ϕ value of calibration surface (as a condition it must be identical for probes A and B)

ZA - Cp_{rpA} values of probe A

ZAX - transformed Cp_{rpA} values of probe A

ZAMAX - maximum of ZA array

ZAMIN - minimum of ZA array

ZB - Cp_{rpB} values of probe B

ZBX - transformed Cp_{rpB} values of probe B

ZBMAX - maximum of ZB array

ZBMIN - minimum of ZB array

Introduction

Experimental knowledge of the flow field generated by rotating turbo impellers is of prime importance in the research and development of turbomachinery. It is essential for the refinement of design methods, for the development of new flow models which include secondary flow and tip clearance effects, and particularly for the verification of new computer codes developed to calculate the flow through rotating blade rows.

In recent years laser velocimeter techniques have been applied successfully to measure the flow both inside and downstream of rotors. (Ref. 1 for example). It has become clear however, that the laser techniques are only reliable in the hands of experienced investigators. A window which remains clean is essential, and seeding is usually required. Laser techniques do not measure the pressure field and usually can only measure two components of the velocity unless the axis of the laser is tilted. Difficulty is also encountered when measuring close to walls. Hence there are reasons to consider alternative techniques, particularly if they are simpler to apply routinely in stationary turbomachinery passages. Furthermore, the achievement of redundancy in measuring the flow field behind the impeller is itself a worthwhile goal. The present work deals with the application of a particular system of small high response pressures probes at the exit of an impeller.

Measurements behind an impeller, in the stationary bladeless gap, are simpler to make than measurements within the rotating passages. Transducer probes can be installed through the stationary machine casing and the data transmitted without resort to slip rings or rotary transformers. The sensor is not subjected to the centrifugal field or to the vibration of the rotor. However, the flow to be measured is then fluctuating at blade-passing frequency and any system of sensors must be calibrated for a wide range of possible Mach number, pitch angle, yaw angle and pressure variation - and yet must be capable of the necessary frequency response.

In Ref. 2, a method was described for using two semiconductor pressure probes together with the technique of synchronized sampling, to obtain the distribution of the velocity vector downstream of a rotor. The geometries of the two probes, designated Type A and Type B, and their installation in the compressor annulus are shown in Fig 1. It was argued that, in principle, by rotating the probes in yaw about their tips and controlling the sampling of the data from each probe to be at the same position in the rotor frame, the system of two separate probes could be used to acquire data at a point in the periodic flow

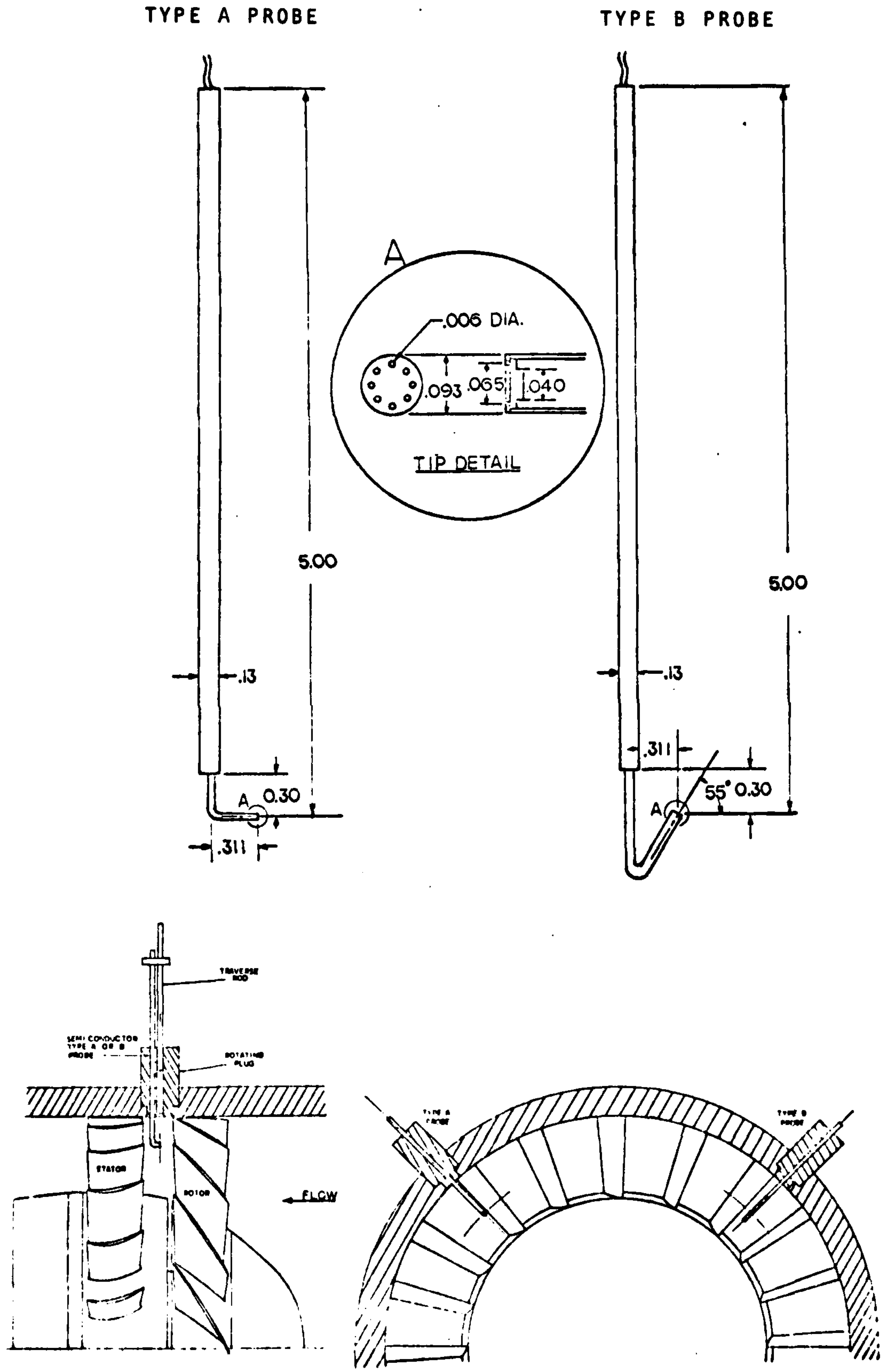


FIGURE 1. TYPE A AND TYPE B PROBES AND COMPRESSOR INSTALLATION.

from the rotor, corresponding to data normally obtained from the multiple sensors of four- or five hole pneumatic probes when measuring velocity in steady flows. The technique promised the use of probes having the simplest geometry and thus avoided the large size, expense and unreliability of multiple sensor probes which incorporate multiple semi-conductor transducers (Ref. 3). Because of the simple sensor tip geometry (that of a cylinder at incidence to the flow) the unsteady response was likely to be as good as could be expected of any single physical sensor.

The two-probe technique is strictly applicable only to periodic flows. However, data obtained on successive rotations of the rotor can be averaged to eliminate fluctuations which are not periodic. This was shown to be effective in tests reported in Ref. 2 in which a single Type A probe was used to establish the peripheral blade-to-blade distribution of flow yaw angle.

In order to obtain velocity from the pressure measurements which can be obtained from the two probes, the steady response characteristics must first be established in calibration tests carried out in a known, controlled, uniform flow. Second, a method of applying the calibration to measurements made in an unknown flow must be devised. In the present method two different approaches have been followed. In Ref. 4, a technique is given for representing and applying the probe calibration analytically. When first applied the method gave surprisingly good accuracy (1 - 2%) since the method required that the probes had characteristics which could be well represented analytically. This in turn required that the probe tips be geometrically precise, a feature which was not present in the first generation of probes.

A second, more general approach is reported here, wherein the calibration of each probe is represented by a two-dimensional array of pressure coefficients. The application of the calibration given in this form, in an unknown flow required the development of special numerical procedures. The purpose of the present report is to document the analysis and the Fortran program developed to apply the method.

In its present form, the method does not require that the calibration "Surfaces" be symmetrical about any axis or be expressed in analytical form, but does require that the pressure coefficient be independent of Mach number. The latter restriction could undoubtedly be removed by introducing additional iterative steps. Further, in the present method only five measurements have to be taken to determine uniquely a velocity vector at a point. Throughout the report, the Fortran program notation has been used to describe the physics and equations involved in the solution.

Mathematical model

Assume the A and B type probes of ref. [1] (see also fig. 1) to be immersed in a three-dimensional steady flow field.* The pressure response of each of these probes in given gas is functionally described by four variables as:

$$P_p = P_p\,(\alpha\,, \phi\,,\; P_T, \,P_S) \tag{1}$$

If a pressure coefficient is defined as

$$Cp_{rp} = \frac{P_p - P_s}{P_T - P_s} \tag{2}$$

The calibration surface of each probe is given in the general case in form of a matrix of values of Cp, where

$$Cp_{rp} = Cp_{rp}(\alpha_{rp}, \phi_{rp}) \tag{3}$$

The pressure coefficient defined in this way has only a second order dependence on Mach and Reynolds numbers in the range of

---

* For our purpose, using the synchronized sampling, the flow field behind the impeller is steady, although the probes require a high speed response because of fluctuations.

$0 < Ma < 0.7$ in turbulent flows (Ref. 2), so that, to first order their influence is neglected in writing eq. (3).

If the type A probe is rotated about its axis into three different positions (i = I, III, IV) readings are taken, and the type B probe is fixed in position II and a single reading is taken, the following four equations can be written.

$$Cp_{Ai} = \frac{P_{pAi} - P_S}{P_T - P_S} = Cp_{Ai}(\alpha,\phi)$$
$$Cp_{BII} = \frac{P_{pBII} - P_S}{P_T - P_S} = Cp_{BII}(\alpha,\phi) \qquad (4)$$

it should be pointed out here, to avoid misunderstanding that α and ϕ are defined in a coordinate system relative to the machine axis and not relative to the probe axis. In the set of four equations (4) there are four unknown quantities, namely: α, ϕ, P_T and P_S . These are the quantities to be evaluated using the measured data. Together with the stagnation temperature they define the flow field uniquely. The four equations, resulting from the four measurements, should be sufficient to determine the four unknown quantities.

But the problem is complicated by three facts:

1) The calibration surfaces are not generally known in analytical form.
2) The calibration surfaces are double valued in α and ϕ i.e., for a given Cp and α there exist two ϕ values, or for a given Cp and ϕ there are two α values which satisfy eq. 3.

3) As a result of the measurement the value of P_p rather than of C_p is determined.

Since the calibration surfaces are not given in a simple analytical form the solution has to be numerical. An iterative procedure is required because P_p and not C_p values are measured. First P_s and P_T have to be guessed to yield C_p values, knowing the four measured values of P_p, and using eq. 4. The guess is then iteratively corrected to converge on the solution. However, convergence of the iterative procedure is complicated because of the double valued nature of the calibration surfaces.

This method of iteration shown in Fig. 2 was attempted initially for the evaluation of a measured point. In practice the initially suggested procedure converged in some cases and diverged in others, depending on the values, and signs of α and ϕ. This was not surprising as convergence on two variables is not likely to be a simple matter. However, in order that the measurement technique be useful, convergence on the correct solution for a general set of measurements, is absolutely necessary. In practice, this can certainly be achieved if one of the two iteration variables is obtained by measurement. Since the static pressure is a difficult quantity to measure even in a steady flow field, only the stagnation pressure measurement need be considered. It is possible that the time-varying stagnation pressure could be measured with a suitably designed Kiel probe. Data would then be taken by synchronized sampling from the fixed Kiel probe, from the type A probe rotated into two positions, and from the

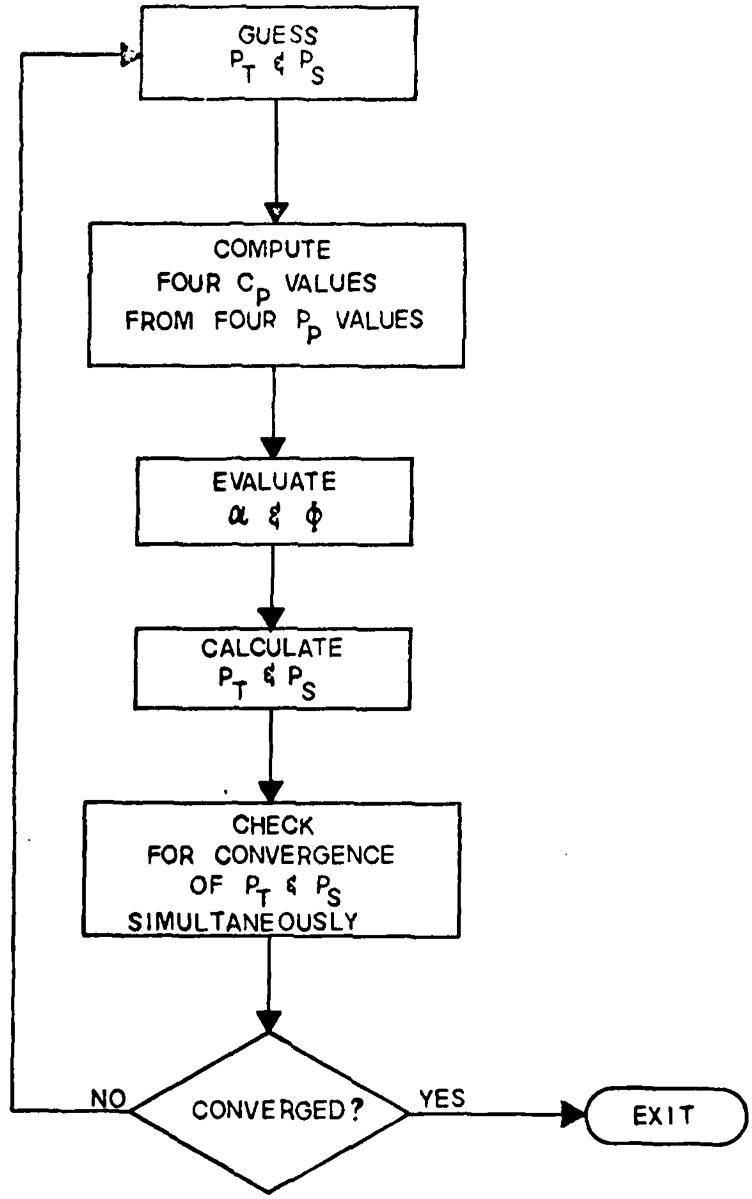


Fig. 2: Unstable iterative solution using measurement of A type and B type probes only.

fixed type B probe. The method of solution is then as shown in Fig. 3. The method shown in Fig. 3 proved to converge under all conditions. It is described in detail in the following pages.

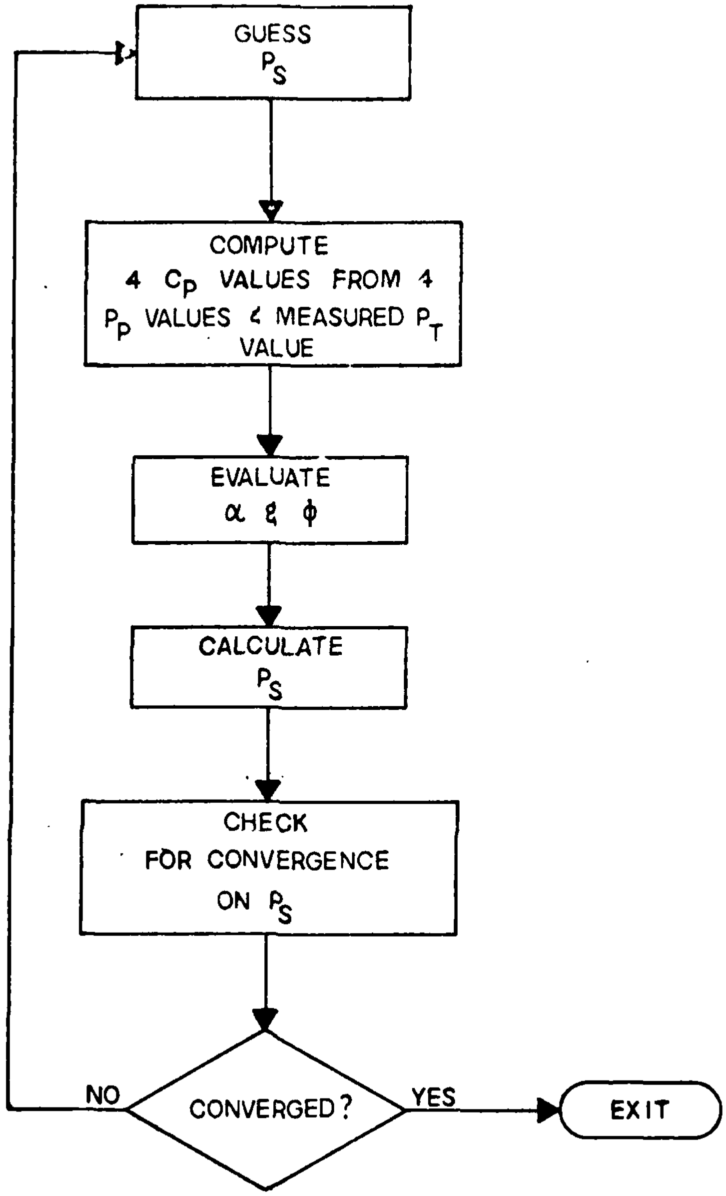


Fig. 3: Stable iterative solution using measurements of A type and B type probes as well as measurements of a Kiel probe.

Evaluation of α and ϕ (method)

The best way to understand the evaluation of yaw and pitch is to look at it from a topographical point of view. Each of the type A and type B probes has a unique calibration surface $Cp_{rp} = Cp_{rp}(\alpha_{rp}, \phi_{rp})$, where α_{rp} and ϕ_{rp} are measured relative to the axis of symmetry through the sensor at the probe tip. The calibration surface in this representation is invariant to yaw and pitch of the probe axis relative to the laboratory space. The same calibration surface if represented in the form $Cp = Cp(\alpha, \phi)$ can be written as

$$Cp = Cp[(\alpha_p + \alpha_{rp}), (\phi_p + \phi_{rp})] \tag{5}$$

where α_p and ϕ_p are the probe tip axis angular settings relative to the laboratory space. It is clear from eq. (5) that Cp can be derived from Cp_{rp} by a constant translation: α_p , ϕ_p on the α, ϕ plane. As each Cp_{rp} can be viewed as a hill with its peak at $\alpha_{rp} = 0$ and $\phi_{rp} = 0$. The Cp surfaces are the same hills with their peaks translated to $\alpha = \alpha_p$ and $\phi = \phi_p$.

In the present method probe A is used in three different angular settings namely:

I) $\alpha_p = 0 \quad \phi_p = 0$

III) $\alpha_p = \alpha_{pIII}$ $\phi_p = 0$

IV) $\alpha_p = \alpha_{pIV}$ $\phi_p = 0$

and the B probe is used in a fixed position, namely:

II) $\alpha_p = 0$ $\phi_p = \phi_{pII}$

For this case the topography of the calibration surfaces will appear as four hills. The three with their peaks at points (0,0); ($+\alpha_{pIII}$, 0) and ($-\alpha_{pIV}$, 0) are the translated hills Cp_{rpA} and the fourth with its peak at (0, ϕ_{pII}) is the translated hill Cp_{rpB}. Their contours of constant Cp are then as shown in figure 4.

Assume now that a velocity vector with yaw α and pitch ϕ is to be measured. These values of α and ϕ will be sensed uniquely by the probes at their four angular settings. Were equation (4) single valued, the values of α and ϕ could be uniquely evaluated, as the single intersection point between the projections of appropriate lines of constant Cp on the α,ϕ plane*. In the present case of the double valued functions, the lines of constant Cp are closed curves and more than a single intersection point do exist.

---

* Cp has generally a different value for each probe in each of its settings. Thus the solution involves solving for the intersection points between projections of contours of specified (but different) Cp values on the different hills.

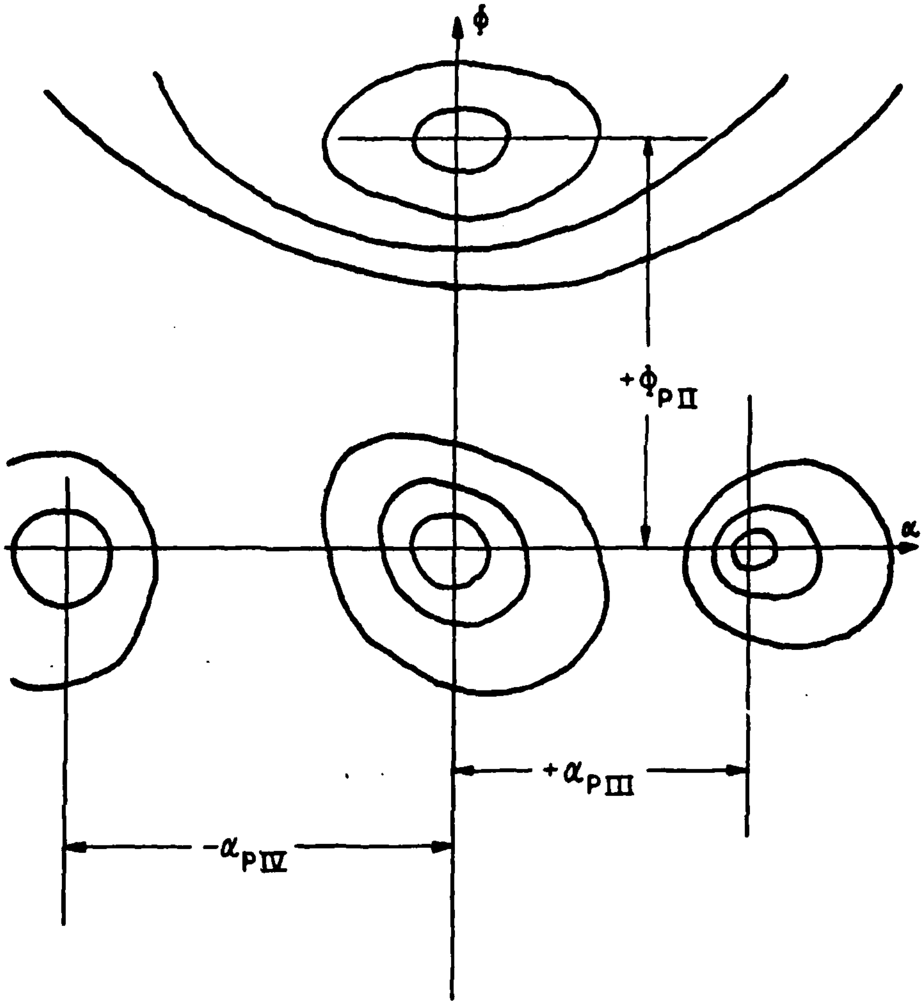


Fig. 4: Projections of C_p = const. lines of the four calibration surfaces (the center hill is for probe A at $\alpha_p = 0$, $\phi_p = 0$, the top hill is for probe B at $\alpha_p = 0$, $\phi_p = \phi_{pII}$, the right hill is for probe A at $\alpha_p = +\alpha_{pIII}$, $\phi_p = 0$ and the left hill is for probe A at $\alpha_p = \alpha_{pIV}$, $\phi_p = 0$.

The situation is shown in Fig. 5 for an example of a number of such intersection points. The correct intersection point, however, is uniquely identified as the only point through which all four Cp = const. curves pass.

From an examination of Fig. 5 it is clear that the intersection points of three closed Cp curves projections only are sufficient to identify α and ϕ uniquely. However one of these three curves must be that belonging to the type B probe.

The details of the numerical procedure used to obtain the correct intersection point in the evaluation of α and ϕ, are given in the following paragraph.

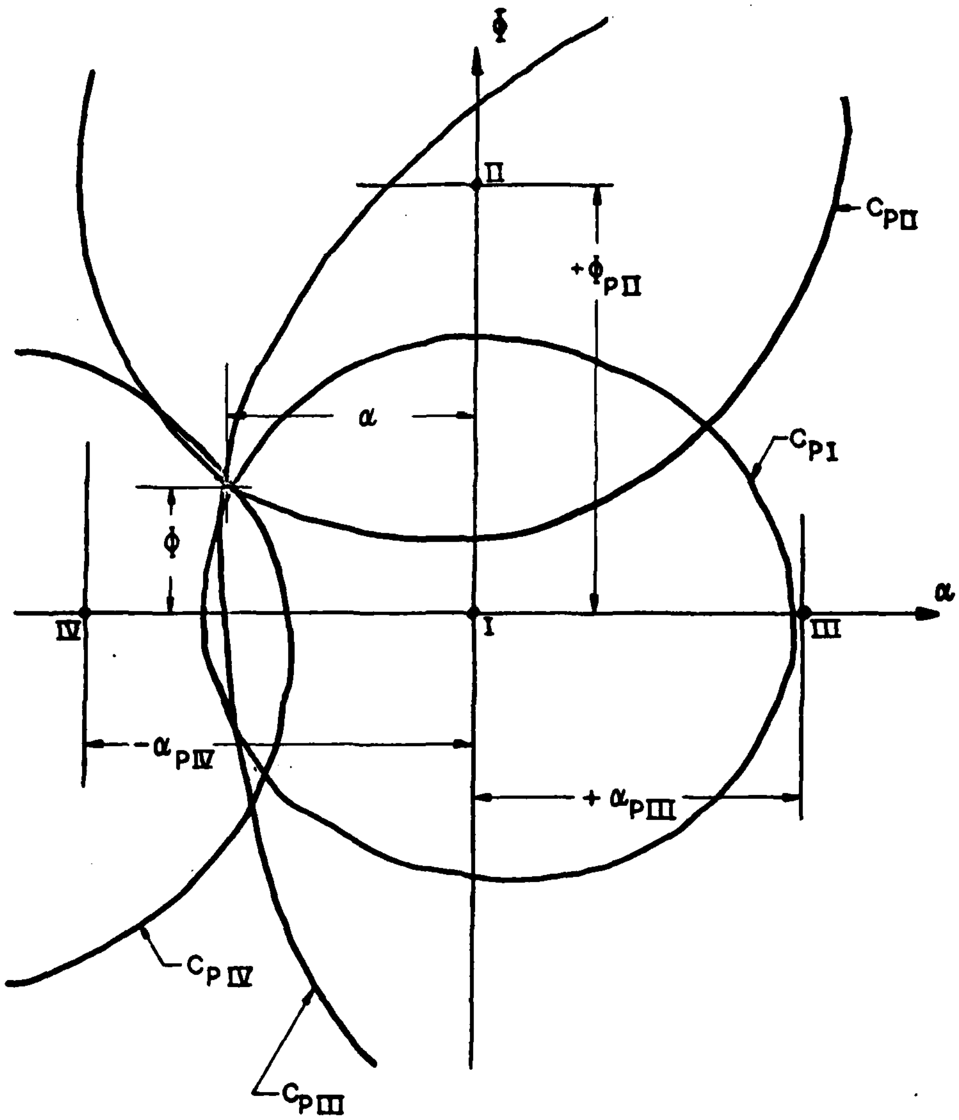


Fig. 5: An example showing a number of intersection points. The real one is in the second quadrant.

Evaluation of the intersection point coordinates (numerical Procedure)

The calibration surfaces Cp_{rp} are represented in form of a linear string of values ordered into an array, as shown schematically in figure 6. The numbers in Fig. 6 indicate the position of a Cp_{rp} value in the string. The string starts with the value of Cp_{rp} at a point $(-\alpha_{rpLo}; -\phi_{rpLo})$ and ends with the value of Cp_{rp} at a point $(+\alpha_{rpUp}; +\phi_{rpUp})$. This sequence must be kept and can not be changed.

Each of the closed Cp = const. curves projections in figs. 2 and 3 is the projection of the line of intersection between a plane parallel to the α,ϕ plane at a height equal to Cp and the calibration surface. These closed curves can be determined as the locus of the projections of all the penetration points of arbitrary lines parallel to the α,ϕ plane at a height Cp and the calibration surface. Such penetration points are calculated in subroutine "PENPTS", Fig. 10.

PENPTS calculates the first two penetration points of a surface Z (in the present case $Z = Cp$) by a straight line piercing that surface. If the surface is double valued these two points are the only roots. The surface is given as a table of numbers on a cartesian basis $Z = Z(X,Y)$ (or in the present case $Cp = Cp(\alpha,\phi)$.

The subroutine has the following limitations:

1) No roots can be found on the lower Y = const. boundary

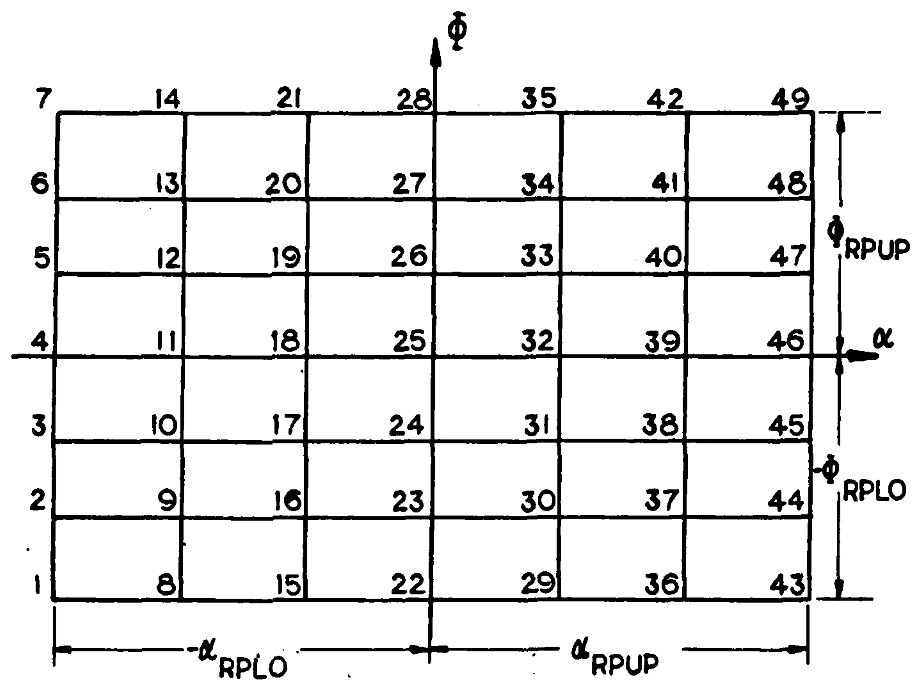


Fig. 6: The order of a calibration surface array.

2) The roots can be found only along a piercing line which is parallel to the X axis (but at any height above the X,Y plane)

3) The table $Z = Z(X,Y)$ defining the surface must be based on a grid comprising lines X = const. and Y = const. The spacing of these lines must not be equal. In other words the surface is defined by a rectangular grid in the X,Y plane, from X_{min} to X_{max} and from Y_{min} to Y_{max}.

4) Only a single root can be evaluated in a surface element located above a defining rectangle on the X,Y plane.

5) Not more than the first two roots will be evaluated for any piercing line.

6) The surface must be monotonic over each rectangle (this is a result of limitation 4).

7) All X and Y arguments must be given in increasing order.

These limitations do not restrict the application of PENPTS in the present problem as long as the calibration surfaces are smooth within the element located above a grid rectangle. However, the elemental grid rectangles can be reduced arbitrarily in size. If a calibration surface is more than double valued PENPTS will fail. However in this case the probe yielding such a calibration surface can not be considered a useful instrument unless used only in parts of the domain where it is double valued.

Z is defined from X_{min} to X_{max} and from Y_{min} to Y_{max}. PENPTS is given the following initial information: X values and Y values defining grid points, corresponding Z values, the Y location (YG) and the height above the X,Y plane (HT) of the piercing line. With this information PENPTS searches for the band of rectangles which includes YG or of which YG is the lower boundary (see fig. 7) and then scans this band from left to right in search for penetration points. The scanning is based on the geometry given in fig. 7 which represents a particular element in the band, approximated by two plane triangles.

Initially a coarse scan is carried out just to detect, but not to evaluate, an intersection point. This is done checking for each sub-domain whether (ZX(I-1) - HT)/(ZX(I) - HT)   0. If this condition is true a root is detected and control is tranferred to its exact evaluation. The value of the root is calculated after its location, either in the first (left) or second (right) triangle is determined (each grid rectangle is composed of two triangles). Equation 6 which is based on fig. 8 (for a left triangle) or eq. 7 which is based on fig. 9 (for a right triangle) is used to evaluate the penetration point. These equations express the linear interpolation of Cp in the Fortran rotation used in this program.

```
XS = X(I-1) + ABS(ZX(I-1)-HT)*(XM-X(I-1))/(ABS(HT-ZM) +
   + ABS(ZX(I-1)-HT))                                        (6)
```

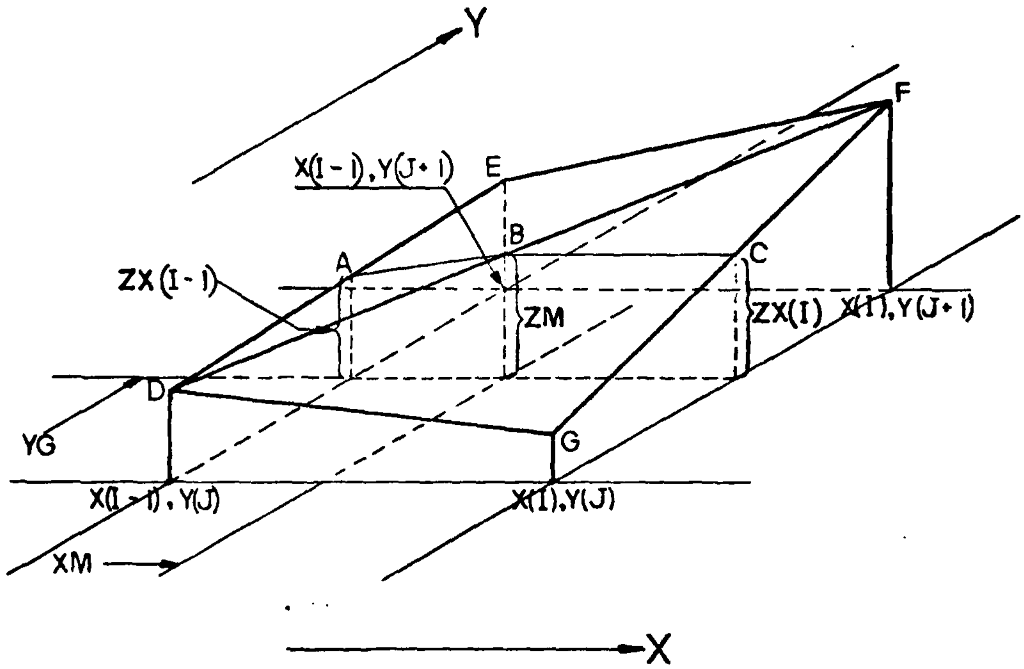


Fig. 7: The geometry of a linearized calibration surface element comprising two plane triangles and its intersection with a plane normal to X,Y along Y = YG.

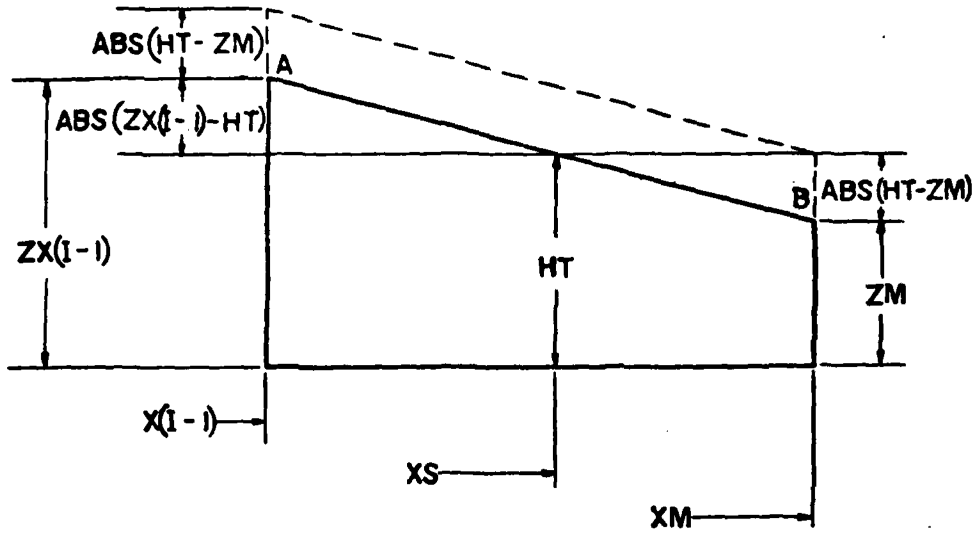


Fig. 8: The geometry for a penetration through a left hand (first) triangle.

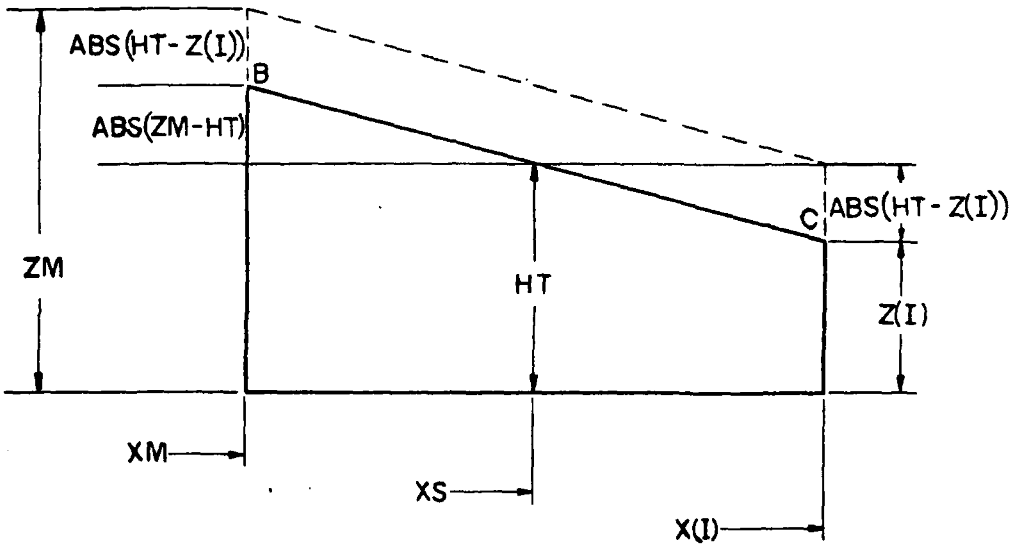


Fig. 9: The geometry for a penetration through a right hand (second) triangle.

$$XS = XM + (X(I)-XM)*ABS(ZM-HT)/(ABS(ZM-HT) + ABS(HT-Z(I))) \quad (7)$$

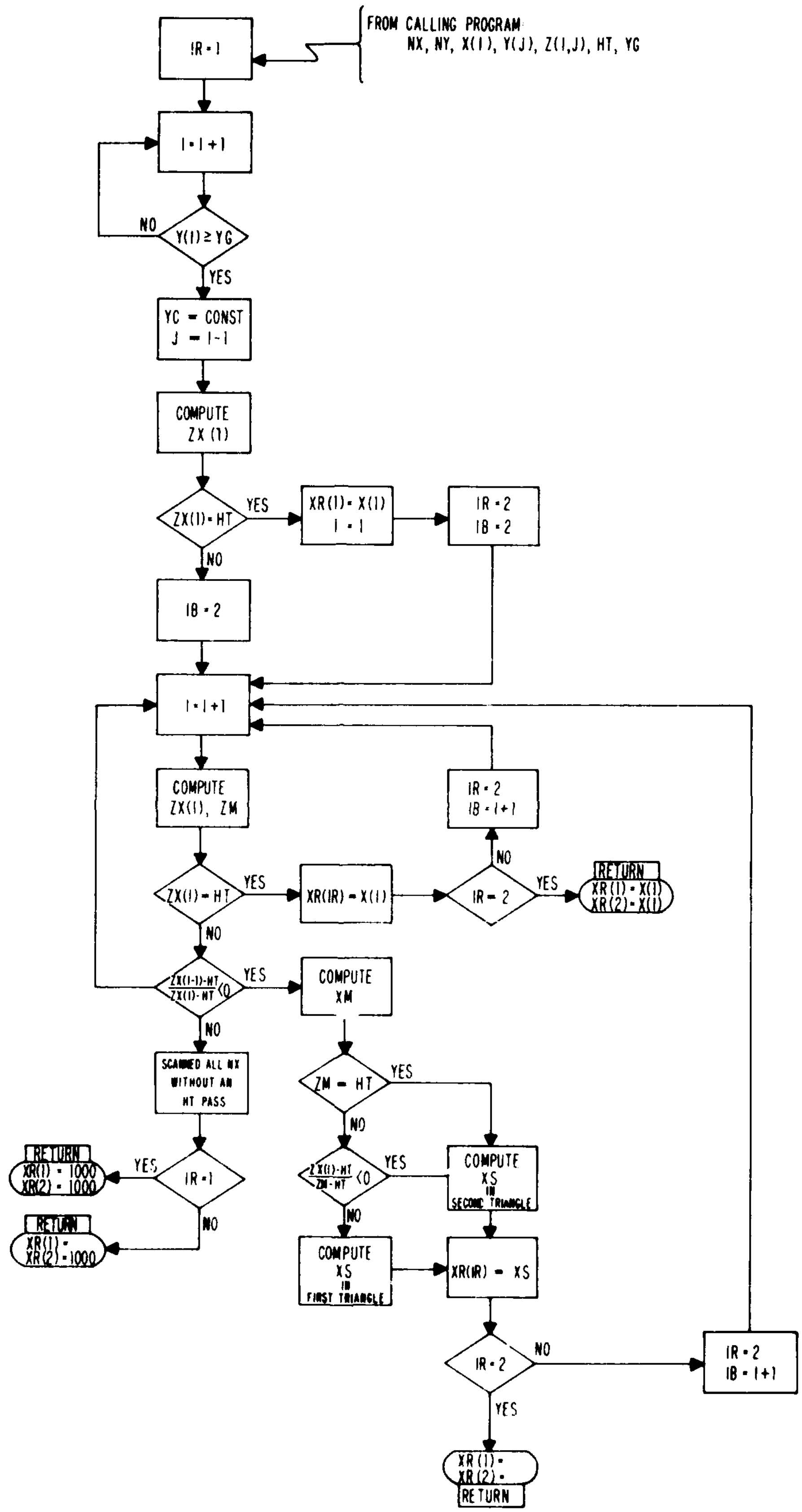
Equations 6 and 7 are invariant to the slope of the calibration surface i.e., the slope of the straight lines AB and BC.

Finally it should be pointed out that when two penetration points are determined the values of their abscissas, X(I), are returned. When only a single penetration point is detected the abscissa X(2) will be returned with a value of 1000.0. When no penetration point is determined both X(I) values returned will have the value of 1000.0. The program logic is designed to recognize these messages.

It was stated earlier that PENPTS is used to determine the closed intersection curves projections on the X,Y plane. In fact not the curves but just the intersection points between each two of them are required (see figure 3).

To compute the coordinates of these points the subroutine "INTSCS" is used. It uses PENPTS as a subroutine.

In INTSCS a scanning procedure is carried out from a minimal value of Y (or ϕ) YRIN to an upper value of YRUP, or a



FROM CALLING PROGRAM
NX, NY, X(I), Y(J), Z(I,J), HT, YG
IR = 1
I = I + 1
NO
Y(I) ≥ YG
YES
YC = CONST
J = I - 1
COMPUTE
ZX (I)
ZX(I) = HT
YES
XR(1) = X(I)
I = 1
IR = 2
IB = 2
NO
IB = 2
I = I + 1
COMPUTE
ZX(I), ZM
IR = 2
IB = I + 1
ZX(I) = HT
YES
XR(IR) = X(I)
IR = 2
NO
YES
RETURN
XR(1) = X(I)
XR(2) = X(I)
NO
YES
COMPUTE
XM
NO
SCANNED ALL NX
WITHOUT AN
HT PASS
ZM = HT
YES
NO
RETURN
XR(1) = 1000
XR(2) = 1000
YES
IR = 1
YES
COMPUTE
XS
IN
SECOND TRIANGLE
RETURN
XR(1) =
XR(2) = 1000
NO
NO
COMPUTE
XS
IN
FIRST TRIANGLE
XR(IR) = XS
IR = 2
NO
IR = 2
IB = I + 1
YES
XR (1) =
XR (2) =
RETURN

prescribed 10,000 times* which ever comes earlier. The subroutine scans through any two arbitrarily chosen closed curve projections to find their intersection points. In each scan (I) up to four penetration points can be determined, while the penetration points of the previous scan (J) are memorized. Together, eight penetration points can be involved. When no intersection point exists the geometrical situation is as shown in figure 11, while the existance of an intersection point is characterized in figure 12. Subroutine INTSCS can distinguish between the two situations. In figures 11 and 12 the case of four penetration points found in each scan are shown. The subroutine, however will handle any possible number of such points, from zero to four. A "no penetration points" is assigned an abscissa value of 1000.0 by PENPTS as explained earlier.

When an intersection point is detected its evaluation is based on the geometry shown in fig. 12. The eight penetration points have the following coordinates:

AJL(RAJL,YRJ)	BJL(RBJL,YRJ)
BJR(RBJR,YRJ)	AJR(RAJR,YRJ)
BIL(RBIL,YR)	AIL(RAIL,YR)
AIR(RAIR,YR)	BIR(RBIR,YR)

* The 10,000 scans are governed by a program constant in the line "DO 140 ...." and can be arbitrarily varied.

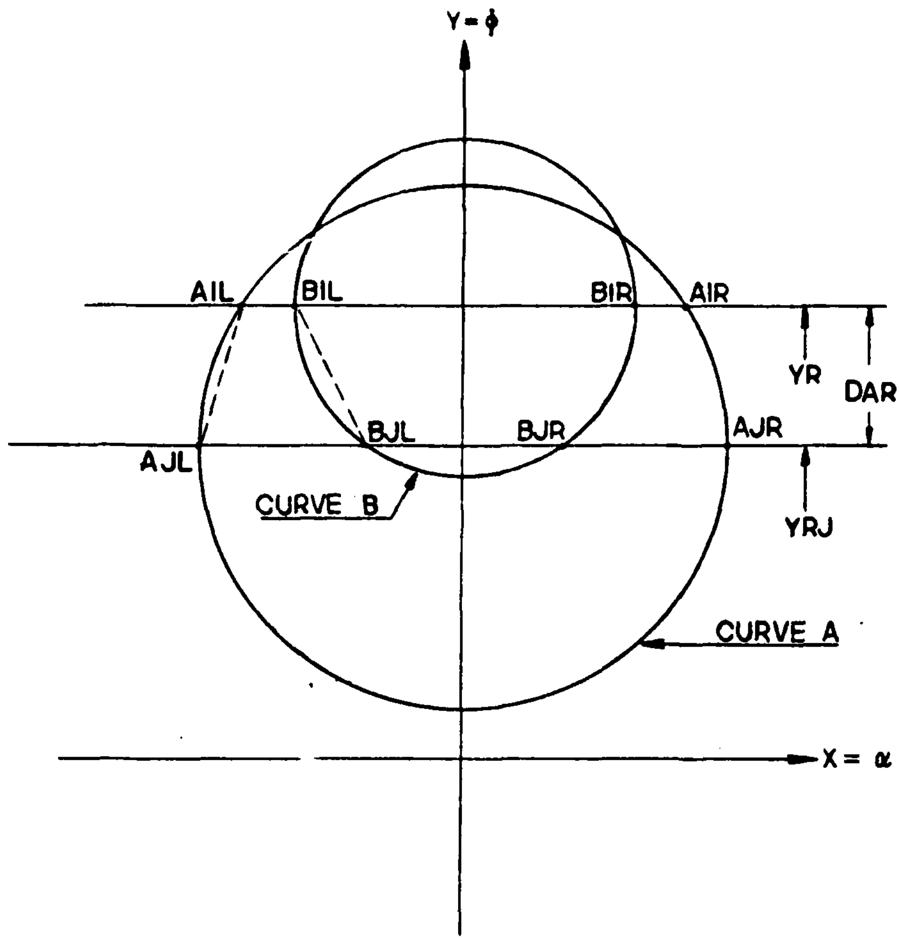


Fig. 11: The geometry for two successive scans when no intersection point exists.

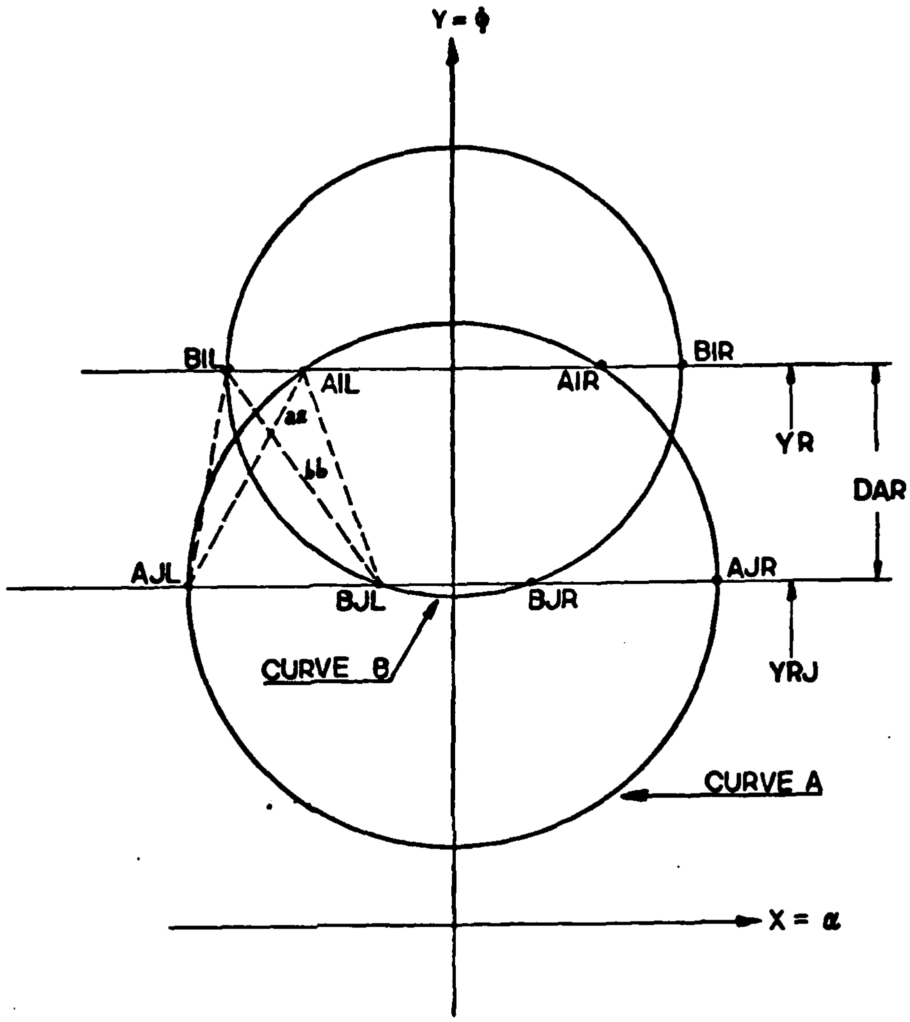


Fig. 12: The geometry for two successive scans when an intersection point does exist.

The left intersection point illustrated in fig. 12 is the intersection point of the lines aa and bb, each described by its equation:

for aa Y = (AA)X + BA (8)

for bb Y = (AB)X + BB (9)

The constants AA, BA, AB and BB are given in equations 10 to 13.

AA = (YRJ-YR)/(RAJL - RAIL) (10)

BA = YR - AA*RAIL (11)

AB = (YRJ-YR)/(RBJL-RBIL) (12)

BB = YR - AB*RBIL (13)

The coordinates of the intersection point to be calculated are

X = (BB-BA)/(AA-AB) (14)

Y = X*AA + BA (15)

These relations are true for a left hand intersection point. Analogous equations are true for a right hand intersection point. In this algorithm the straight lines aa and bb approximate the curved lines connecting AJL with AIL and BJL with BIL or similar lines on the right hand side of figure 12. The error introduced through this approximation is reduced as DAR = $\Delta\phi$ is reduced.

It is possible that an intersection point is identical with AIL and BIL or AIR and BIR. This case is defined as "direct hit". The program is designed to detect such a direct hit and evaluate the corresponding intersection point accordingly.

The above algorithm works perfectly as long as the two closed Cp = const. curves are far from being tangent. But in practice a situation of almost tangent curves can arise when X (or α) is very small. In this case the preceeding algorithm will fail and must be replaced. The geometry of this situation is described in figure 13. This situation is identified by INTSCS and the intersection points are then evaluated assuming that they are intersections of two circular arcs. When the curves Cp = const. are almost circular this approximation does not lead to unacceptable errors.

It was stated earlier that INTSCS scans from a minimum value of ϕ to a maximum value of ϕ with prescribed steps $\Delta\phi$, as shown in figs. 11 and 12. This direction of the scanning is used when the intersection points of the curves Cp_I and Cp_{II} of fig. 5 are evaluated.

However, in course of the reduction of the measured data, scans in the direction of α in steps of $\Delta\alpha$ are also necessary to evaluate the intersections of the curves Cp_I and Cp_{III} or Cp_I and Cp_{IV}. INTSCS is designed to carry out this task as well. To do this the calling statement for INTSCS is appropriately changed as will be explained in the next section. To be general enough INTSCS is not written in terms of α and ϕ or X and Y but rather in terms of general arguments. The best way to understand INTSCS is to compare its general arguments to physical quantities by means of the calling statements. In fig. 14 the flow diagram in INTSCS is given in terms of the general arguments.

INTSCS returns to the main program (fig. 15) the (α, ϕ) coordinates of two intersection points between the Cp = const. curves specified in the calling statement. Let us now follow the way in which the main program is designed to utilize INTSCS

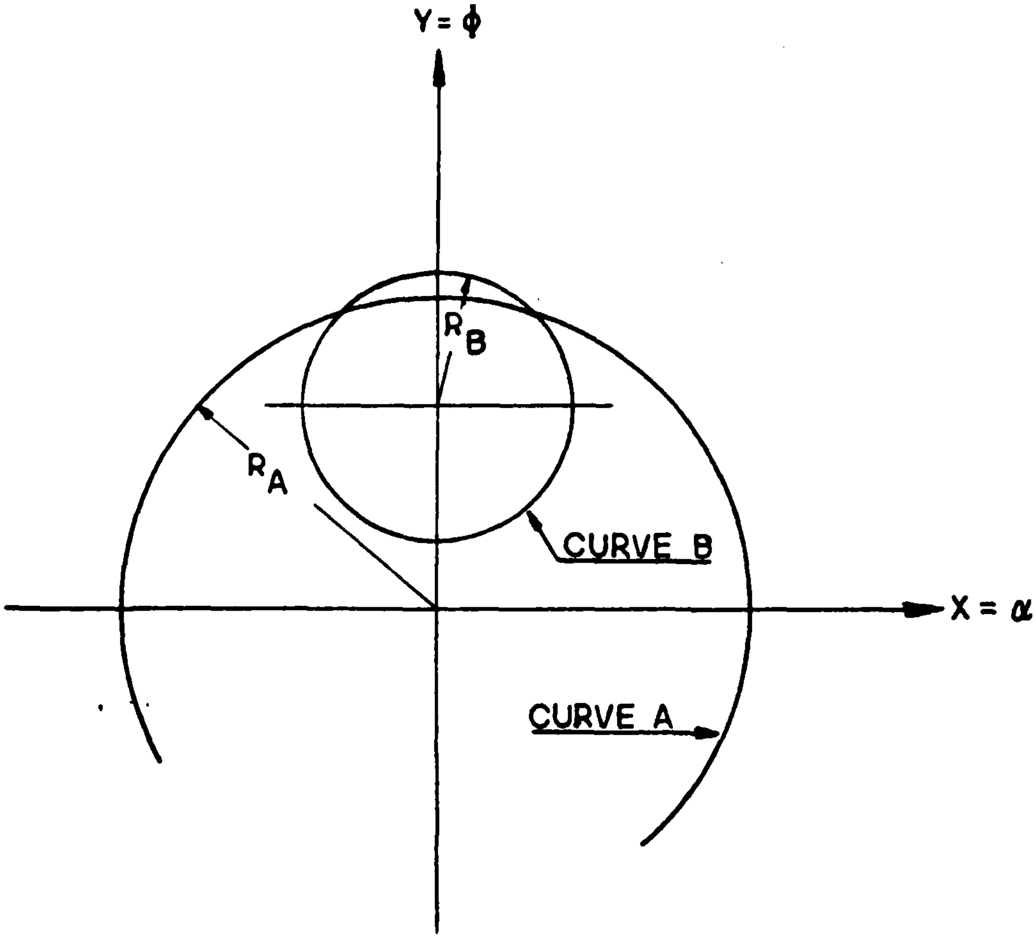


Fig. 13: The geometry when two closed C_p = const. curves are almost tangent.

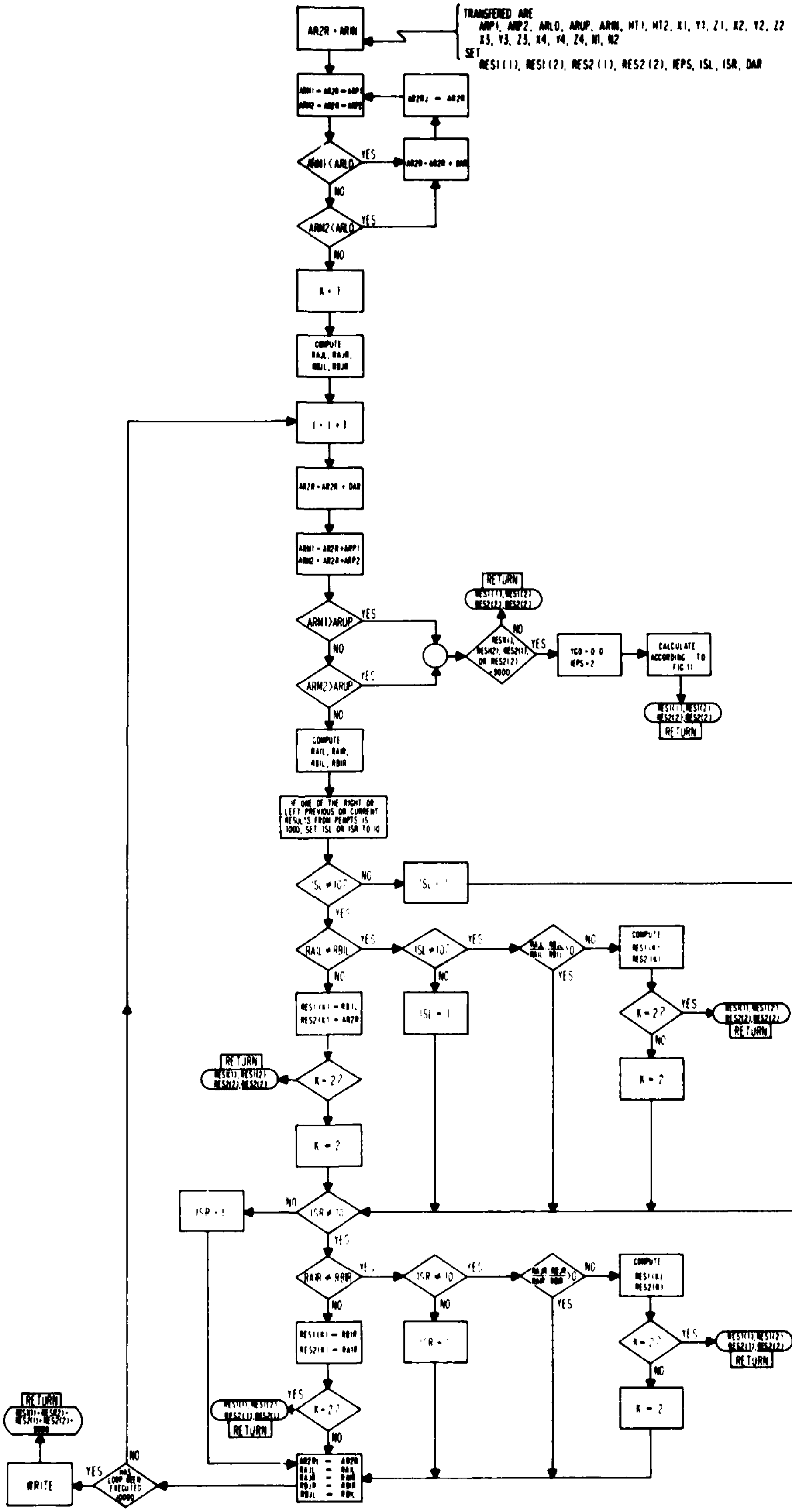


FIGURE 14. FLOW DIAGRAM OF INTSCS

for the evaluation of α and ϕ of the velocity vector, as well as static and total pressures.

Evaluation of the Velocity Vector and Pressures from the Probe Signals

Fortran program VDR was written to evaluate the velocity vector from probe measurements of pressures. The program is shown in Fig. 15.

At line 1410 INTSCS is called to scan curves I and II for possible intersection points. Scanning can be carried out in the direction of the ordinate only, with the calibration curve matrices compiled exactly as shown in fig. 6. This limitation is imposed by the way PENPTS is constructed. In the case of the intersection points between I and II ZA and ZB are scanned without difficulties in the direction of the ordinate which is ϕ as shown in fig. 12 and returns with the coordinates of the two first intersection points, points 1 and 2 of Fig. 16. They are ALF1, PHI1 and ALF2, PHI2. In line 1500 INTSCS is called again to scan curves I and III. Now scanning has to be carried out in the direction of the abscissa, a task for which INTSCS was not designed. To overcome this problem the calibration curve matrices are used in a transformed form such that the previous abscissas are now ordinates, ordinates are abscissas and the internal structures of the Cp_{rp} arrays are accordingly modified. This transformation is carried in VDR in the section between lines 320 and 450. Comparison of the calling line 1500 to the previous calling line 1410 shows very clearly how the various arrays: original and transformed, are used. The coordinates of points 3 and 4 of fig. 16

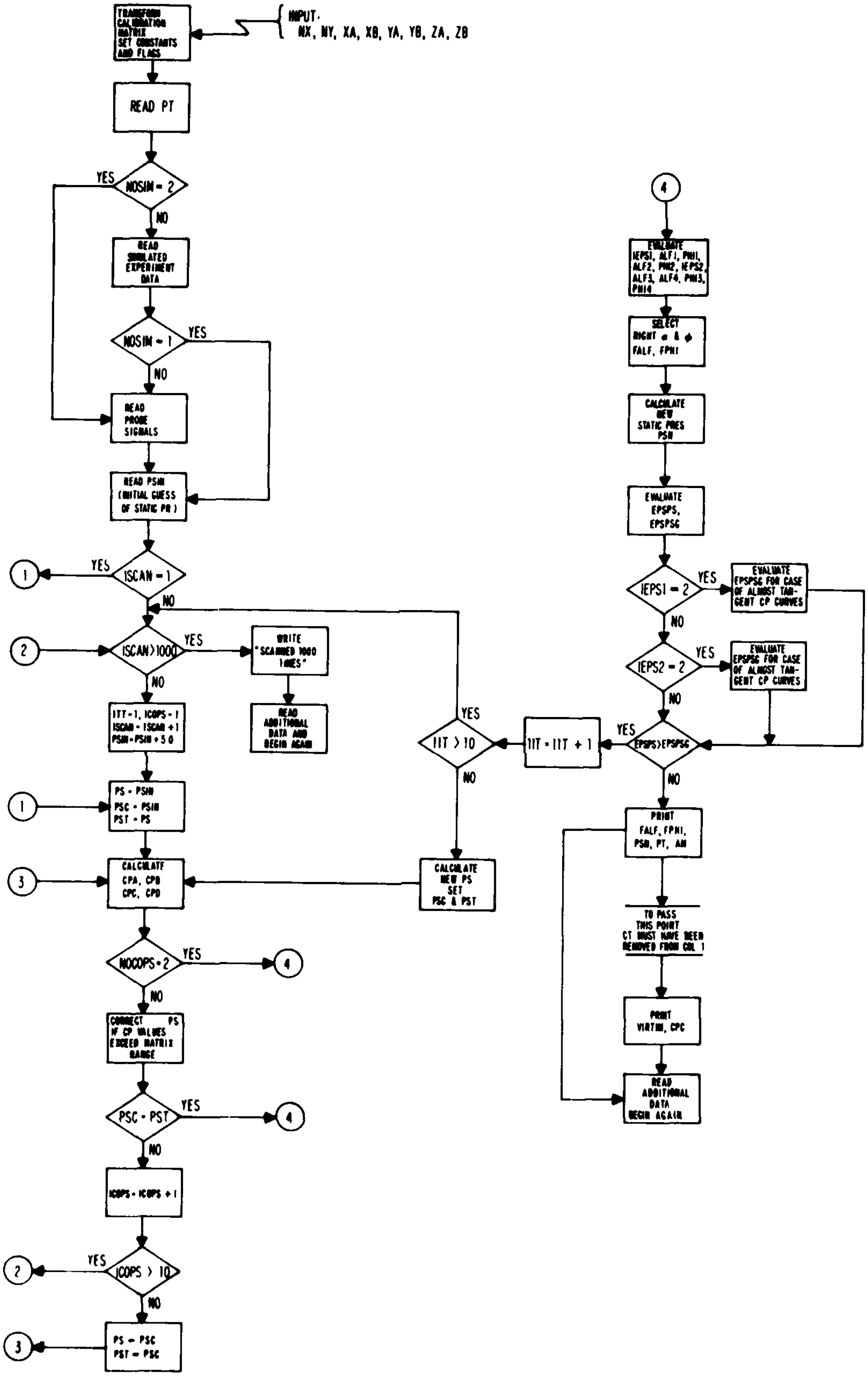


FIGURE 15. FLOW DIAGRAM OF VDR

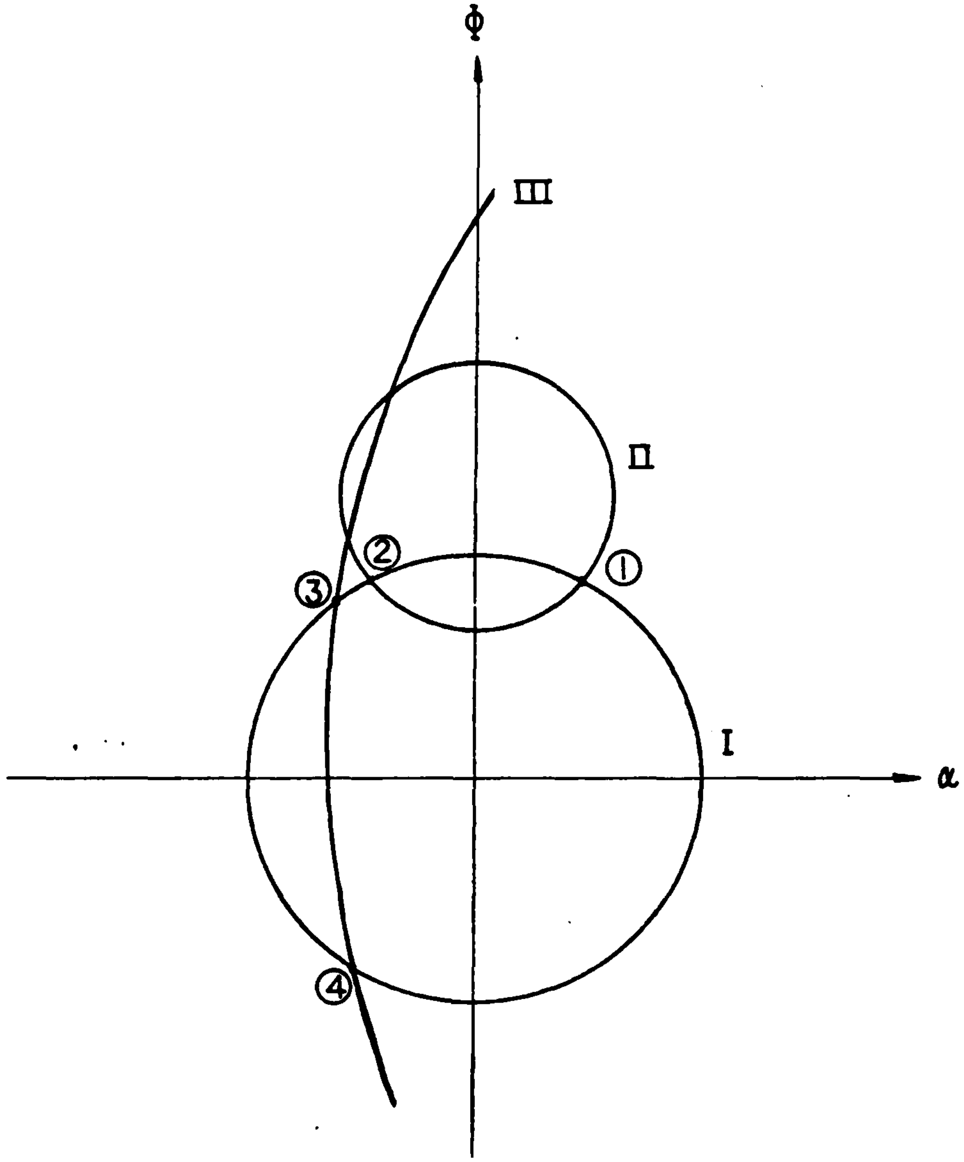


Fig. 16: Selection of the proper intersection point in VDR.

are now returned to the main program of VDR, they are ALF3,PHI3 and ALF4,PHI4.

We are seeking a single intersection point; the one representing yaw and pitch as sensed by probe A in positions I and III and probe B in position II. But, as the calibration surfaces are double valued we are now, unfortunately, in the possession of four points. The solution, however, is physically unique. Only a single velocity vector exists in reality and its yaw and pitch are included in the four intersection points evaluated. Were both the measurements and the numerical procedure absolutely accurate, two of the four points would have been identical. But this is not the case in reality, instead of a single point, two points close to each other will be detected. Therefore the average coordinates of the two of the four intersection points returned to VDR from INTSCS which are closest to each other are selected as the measured yaw and pitch angles. In the example shown in fig. 16 this will be the point dividing the distance between points 2 and 3.

The calculation is now at the point at which α and ϕ are temporarily known (see fig. 3). Using α and ϕ a new Cp_{rpIV} (for probe A in position IV) is computed by linear interpolation using subroutine INTPLT. With this new value of Cp_{rpIV} and with Pp_{IV} a new P_s is computed. If this new value of P_s is close enough as determined by EPSPSG to the guessed value, or to the value of P_s in the previous iteration, the data reduction for the particular point in question is terminated.

The relative difference between present and previous P_S is compared to the convergence criterion EPSPSG. This criterion is evaluated by an empirical function determined to give best compromise between accuracy of results and computer time required until convergence is achieved. When the calculation follows normal routine the convergence criterion is given in line 1710 as function of P_T . When the routine for almost tangent curves is used during data reduction either on the right or the left side a different empirical function (lines 1720 or 1730) is used.

If the convergence criterion is satisfied results are printed out and data for a new measurement point is read for reduction. If, however convergence is not reached a new static pressure for a next iteration is evaluated (line 1780).

Convergence and accuracy

The convergence of the present iterative procedure is not ensured in all possible cases of data sets. The program, however, is adjusted to converge in most of the cases. Similarly to other iterative computation method some experience is required to achieve convergence when the calculation does not converge. In this paragraph the principal factors affecting convergence, computation time and accuracy, which are abviously coupled, are pointed out.

The convergence and accuracy of reduction of a data set: P_T, PPA, PPB, PPC, PPD, PSIN (or analogous set in the experiment simulation mode) depends on the following factors, which can be varied by the user.

1) Coarseness of calibration arrays ZA, ZB and their linearity. With coarser arrays convergence problem will increase and accuracy of results decrease.

2) Probe settings YPB, XPC, XPD (with the following probe settings being fixed and not variable YPA = YPC = YPD = 0.0; XPA = XPB = 0.0). The optimum setting is about $\pm 20^{\circ}$ to 25°. Too small values reduce accuracy, too big values cause convergence problems and probe tip flow separation.

3) RELXPS, the static pressure relaxation factor. The smaller this factor the safer will convergence be achieved. Computation time, however, will increase.

4) The constant 5.0 in line 1050 of the program. The dimension of this constant is kg/m^2. If too big results can be lost and if too small computer time will be growing. The variation of this constant should be coupled with an appropriate modification of the constant in line 1010 of VDR (item 8 in this list).

5) The constants in the evaluation of EPSPSG (lines 1710 to 1730). The smaller EPSPSG the more accuarte the results in expense of increased computation time and reduced convergence safety.

6) The constant scanning step DAR. The smaller DAR the more accurate the results, but too small values can cause complete loss of results. Computation time increases with reduced DAR.

7) The constant 10 in line 1360. This constant governs the number of P_s corrections.

8) The constant 1000 in line 1010. This constant governs the number of scans.

If convergence is not achieved in a particular case variation of each or of a combination of the above values will always enable convergence.

Evaluation of the Computation Time

By deleting the letters CT from column 1 and 2 of lines 20 to 70, 1930 to 1960, 1980, 2000, 2010 the actual computation time as well as CPU time are evaluated and printed out. The following statement prior to execution is required in this case:*

GLOBAL T SYSLIB SSPLIB

This option is useful for adjusting the constants affecting convergence as to optimal compromise between accuracy, ease of convergence and calculation costs.

---

* When the program is run on Naval Postgraduate School IBM 360 system.

Input

1) Calibration arrays have to be input in the following manner: first values of NX and NNY, second values of X in rising order, third values of Y in rising order and fourth values of Z in the order shown in Fig. 6. Two calibration arrays are read: first the array of type A probe and second the array of type B probe.

2) If the program is run in experiment simulation mode PT and PS are read in Kg/m^2 and subsequently o and ϕ.

3) If the program is run in data reduction mode the measured values of PT, PPA, PPB, PPC and PPD are read and then the guess of static pressure PSIN. All are read in Kg/m^2. PSIN has to be lower than the actually existing value to ensure convergence of the calculation. Too low a value will cause waste of computer time.

Calibration arrays are read from a disk space on which they are stored. The following statement prior to execution is required for successful reading:

```
FILEDEF 02 DSK NAME XX
```

here NAME is the name of the file on which the calibration data of probes A and B is stored in proper order and format, XX is a two digit number.

Simulation or measurements reduction input is read in the normal way using the terminal keyboard or punched cards.

Output

α, ϕ, P_S, P_T and Ma are printed out according to lines 1820 to 1920. This output, however is not sufficient when the logic of the computation is to be followed either to examine the execution of the program or for debugging. Two programs with additional output are given in appendices 2 and 3 and can be used for this purpose. The first, SWVDR gives short additional output, namely:

1) When PSIN is successively increased automatically by the program to ensure convergence, the values of PSIN are printed out.

2) IIT, FALF, FPHI and PSN are printed out at the end of each iteration prior to the convergence test.

The second program WVDR prints more detailed information. All additional WRITE statements in this program are numbered with three digit numbers starting with 9.

Conclusions

The following conclusions are drawn on the basis of experience gained in running the program with various data sets and various types of calibration surfaces.

1) Optimal probe settings for ease of convergence and high accuracy are: YPB = 25°, XPC = 25°, XPD = -25°. It is therefore suggested that a B type probe with 25° pitch will be used, and that the A type probe be rotated to ± 25°.

2) Convergence and accuracy, as well as computer time efficiency are improved when the calibration surfaces of both probes are not flat at their peaks but are rather rounded. It is therefore suggested that a new probe tip geometry be considered. A spherical tip with a central pressure tap is recommended. To prevent damage to the sensitive transducer located behind the pressure tap, and in order to improve the frequency response of the probes it is suggested that the volume between the pressure tap face and the transducer be filled with an appropriate liquid that will not affect the transducer negatively. In this case the opening of the pressure tap has to be sealed with a very thin low inertia membrane.

3) It is probably possible to modify the iterative procedure such that safe convergence can be achieved also when using the scheme of Fig. 2. If this can be achieved the Kiel probe will not be necessary. An effort in this direction is suggested.

References

1. Dunker R. J., Strinning P. E., and Weyer, H. B., "Experimental Study of the Flow Field Within a Transonic Axial Compressor Rotor by Laser Velocimetry and Comparison With Through-Flow Calculations", ASME Journal of Engineering for Power, Vol. 100, pp. 279-286, April 1978.
2. Shreeve, P. P., Simmons J. M., Winters K. A., and West J. C. Jr., "Determination of Transonic Compressor Flow Field by Synchronized Sampling of Stationary Fast Response Transducers", Symposium on Non-Steady Fluid Dynamics, ASME 1978 Winter Annual Meeting, San Francisco, Dec 1978. (To be published in ASME Journal of Fluids Engineering)
3. Thompkins W. T. Jr., and Kerrebrock J. L., "Exit Flow From a Transonic Compressor Rotor", AGARD Conference Proceedings No. 177, Unsteady Phenomena in Turbomachinery, pp. 6-1 to 6-23. Meeting held at Naval Postgraduate School, Monterey, California, 22-26 September 1975.
4. Shreeve R. P., McGuire A. G., and Hammer J. A., "Calibration of a Two Probe Synchronized Sampling Technique for Measuring Flows Behind Rotors", paper to be presented at IEEE, Eighth International Congress in Instrumentation in Aerospace Simulation Facilities, Naval Postgraduate School, Monterey, September 24-26 1979. Published as IEEE ICIASF Record of Proceedings.

APPENDIX I

LISTING OF VDR

07/08/79 18.43.34

FILE: VOR FORTRAN P1 N P S

```
C START OF VOR
CT    INTEGER A(6)
CT    IASC=0
CT    IAC0=0
      DIMENSION ALF(10),PHI(10),XA(40),YA(40),
     1ZA(40,40),XB(40),YB(40),ZB(40,40),XAX(40),YAX(40),
     2ZAX(40,40),XBX(40),YBX(40),ZBX(40,40)
119   FORMAT(2I5)
131   FORMAT(F7.1)
130   FORMAT(F5.1)
120   FORMAT(F8.5)
132   FORMAT(F10.5)
135   FORMAT(F15.5)
136   FORMAT(F17.5)
C READ PROBE CALIBRATION DATA
      READ(2,119) NX,NY
      DO 121 I=1,NX
121   READ(2,130) XA(I)
      DO 122 I=1,NY
122   READ(2,130) YA(I)
      DO 123 I=1,NX
      DO 123 J=1,NY
123   READ(2,120) ZA(I,J)
      DO 221 I=1,NX
221   READ(2,130) XB(I)
      DO 222 I=1,NY
222   READ(2,130) YB(I)
      DO 223 I=1,NX
      DO 223 J=1,NY
223   READ(2,120) ZB(I,J)
C TRANSFORM CALIBRATION MATRICES
      DO 224 I=1,NY
224   XAX(I)=YA(I)
      DO 225 I=1,NX
225   YAX(I)=XA(I)
      DO 226 I=1,NY
      DO 226 J=1,NX
226   ZAX(I,J)=ZA(J,I)
      DO 227 I=1,NY
227   XBX(I)=YB(I)
      DO 228 I=1,NX
228   YBX(I)=XB(I)
      DO 229 I=1,NY
      DO 229 J=1,NX
229   ZBX(I,J)=ZB(J,I)
C PROBE SETTINGS
      YPA=0.0
      YPB=55.
      XPA=0.0
      XPC=20.
      XPD=-20.
C PROGRAM CONSTANTS
      YRLO=-30.
      YRUP=30.
      YRMIN=-30.
      XRLO=-30.
      XRUP=30.
      XRMIN=-30.
      ICOPS=1
      NOCOPS=1
      NOSIM=1
      IIT=1
      ISCAN=1
      RELXPS=0.5
C EXPERIMENT SIMULATION
401   FORMAT(F15.4)
193   READ(5,401) PT
      WRITE(6,401) PT
      IF(NOSIM.EQ.0) GO TO 500
      READ(5,401) PS
```

```
      WRITE(6,401) PS
      READ(5,401) ALFA
      WRITE(6,401) ALFA
      READ(5,401) PHII
      WRITE(6,401) PHII
      CALL INTPLT(XA,YA,ZA,ALFA,PHII,NX,NY,CPA)
      ALFC=ALFA-XPC
      CALL INTPLT(XA,YA,ZA,ALFC,PHII,NX,NY,CPC)
      ALFD=ALFA-XPD
      CALL INTPLT(XA,YA,ZA,ALFD,PHII,NX,NY,CPD)
      PHIB=PHII-YPB
      CALL INTPLT(XB,YB,ZB,ALFA,PHIB,NX,NY,CPB)
      PDYN=PT-PS
      PPA=CPA*PDYN+PS
      PPB=CPB*PDYN+PS
      PPC=CPC*PDYN+PS
      PPD=CPD*PDYN+PS
C READ MEASUREMENTS DATA
      IF(NCSIM.EQ.1) GO TO 501
500   READ(5,135) PPA
      WRITE(6,136) PPA
      READ(5,135) PPB
      WRITE(6,136) PPB
      READ(5,135) PPC
      WRITE(6,136) PPC
      READ(5,135) PPD
      WRITE(6,136) PPD
501   READ(5,135) PSIN
      WRITE(6,136) PSIN
      IF(ISCAN.EQ.1) GO TO 181
180   IF(ISCAN.GT.1000) GO TO 305
      IIT=1
      ICOPS=1
      ISCAN=ISCAN+1
      PSIN=PSIN+5.0

181   PS=PSIN
      PSC=PSIN
      PST=PS
C CALCULATES PRESSURE COEFFICIENTS
300   PDYN=PT-PS
      CPA=(PPA-PS)/PDYN
      CPB=(PPB-PS)/PDYN
      CPC=(PPC-PS)/PDYN
      CPD=(PPD-PS)/PDYN
C CORRECTS PS ASSUMPTION TO ENSURE CP ARE IN CALIBRATION RANGE
      IF(NOCIPS.EQ.2) GO TO 301
      ZAMAX=-10000.
      CALL MAXARY(ZA,ZAMAX,NX,NY)
      IF(CPA.GT.ZAMAX) PSC=1.01*PS
      IF(CPC.GT.ZAMAX) PSC=1.01*PS
      IF(CPD.GT.ZAMAX) PSC=1.01*PS
      ZBMAX=-10000.
      CALL MAXARY(ZB,ZBMAX,NX,NY)
      IF(CPB.GT.ZBMAX) PSC=1.01*PS
      ZAMIN=10000.
      CALL MINARY(ZA,ZAMIN,NX,NY)
      IF(CPA.LT.ZAMIN) PSC=0.99*PS
      IF(CPC.LT.ZAMIN) PSC=0.99*PS
      IF(CPD.LT.ZAMIN) PSC=0.99*PS
      ZBMIN=10000.
      CALL MINARY(ZB,ZBMIN,NX,NY)
      IF(CPB.LT.ZBMIN) PSC=0.99*PS
      IF(PSC.EQ.PST) GO TO 301
      ICOPS=ICOPS+1
      IF(ICOPS.GT.10) GO TO 180
      PS=PSC
      PST=PSC
      GO TO 300
C CALCULATES TWO ALF & PHI ANGLES FOR 'I' & 'II'
```

```
301     CALL INTSCS(YPA,YPB,YBLC,YBUP,YRIN,
       1CPA,CPB,ALF,PHI,XA,YA,ZA,
       2XB,YB,ZB,XAX,YAX,ZAX,XBX,YBX,ZBX,NX,NY,IEPS)
        IEPS1=IEPS
        ALF1=ALF(1)
        ALF2=ALF(2)
        PHI1=PHI(1)
        PHI2=PHI(2)
C CALCULATES TWO ALF & PHI ANGLES FOR 'I' & 'III'
        CALL INTSCS(XPA,XPC,XBLD,XPUP,XRIN,
       1CPA,CPC,PHI,ALF,XAX,YAX,ZAX,
       2XAX,YAX,ZAX,XA,YA,ZA,XA,YA,ZA,NY,NX,IEPS)
        IEPS2=IEPS
        ALF3=ALF(1)
        ALF4=ALF(2)
        PHI3=PHI(1)
        PHI4=PHI(2)
C SELECTS THE PROPER ALF & PHI ANGLES OUT OF THE FOUR VALUES
        FALF=((ALF1+ALF2+ALF3+ALF4)-AMAX1(ALF1,ALF2,ALF3,
       1ALF4)-AMIN1(ALF1,ALF2,ALF3,ALF4))/2
        FPHI=((PHI1+PHI2+PHI3+PHI4)-AMAX1(PHI1,PHI2,PHI3,PHI4)
       1-AMIN1(PHI1,PHI2,PHI3,PHI4))/2
C CALCULATES NEW PS FROM DATA OF POSITION IV
        ALFDN=FALF-XPD
        CALL INTFLT(XA,YA,ZA,ALFDN,FPHI,NX,NY,CPD)
        ALFCN=FALF-XPC
        CALL INTFLT(XA,YA,ZA,ALFCN,FPHI,NX,NY,CPC)
        PSN=(PPC*CPD-PPD*CPC)/(CPD-CPC)
C CONVERGENCE TESTS
        EPSPS=ABS((PSN-PS)/PS)
        EPSPSC=.0000000000099*(PT**2.0)-.0000000098*PT
        IF(IEPS1.EQ.2) EPSPSC=.0000000000009*(PT**2.0)-.00000008*PT
        IF(IEPS2.EQ.2) EPSPSC=.0000000000009*(PT**2.0)-.00000008*PT
        IF(EPSPS.GT.EPSPSC) GO TO 310
        GO TO 303
310     IIT=IIT+1
        IF(IIT.GT.10) GO TO 180
        PS=PS+RELAPS*(PSN-PS)
        PSC=PS
        PST=PS
        GO TO 300
303     WRITE(6,171) FALF
171     FORMAT('0',' VELOCITY YAW IS' ,F15.2)
        WRITE(6,172) FPHI
172     FORMAT(' VELOCITY PITCH IS' ,F15.2)
        WRITE(6,173) PSN
173     FORMAT(' STATIC PRESSURE IS' ,F15.3)
        WRITE(6,174) PT
174     FORMAT(' TOTAL PRESSURE IS' ,F15.3)
        AM=SQRT(((PT/PSN)**0.285714-1.0)*5.0)
        WRITE(6,175) AM
175     FORMAT(' MACH NUMBER IS' ,F15.5)
CT      CALL IACLIN(A)
CT      VIRTIM=(A(5)-IA5C)/76800
CT      CPU=(A(6)-IA6C)/76800
CT      WRITE(6,450) VIRTIM
450     FORMAT(' VIRTUAL TIME IS' ,F15.5)
CT      WRITE(6,451) CPU
451     FORMAT(' TOTAL CPU IS' ,F15.5)
CT      IA5C=A(5)
CT      IA6C=A(6)
        GO TO 193
305     WRITE(6,307)
307     FORMAT(' SCANNED 1000 TIMES')
        GO TO 193
        END
        SUBROUTINE INTSCS(ARP1,ARP2,ARLD,ARUP,ARIN,
       1HT1,HT2,RES1,RES2,X1,Y1,Z1,
       2X2,Y2,Z2,X3,Y3,Z3,X4,Y4,Z4,N1,N2,IEPS)
        DIMENSION RES1(10),RES2(10),ARGES(10),X1(40),Y1(40),
```

```
      1Z1(40,40),X2(40),Y2(40),Z2(40,40),X3(40),Y3(40),
      2Z3(40,40),X4(40),Y4(40),Z4(40,40)
       RES1(1)=9000.
       RES1(2)=9000.
       RES2(1)=9000.
       RES2(2)=9000.
       IEPS=1
       ISL=1
       ISR=1
       DAR=1.
       AR2R=ARIN
       GO TO 152
150    AR2R=AR2R+DAR
       AR2RJ=AR2R
152    ARM1=AR2R-ARP1
       ARM2=AR2R-ARP2
C CHECK FOR LOWER CALIBRATION RANGE
       IF(ARM1.LT.ARLO) GO TO 150
       IF(ARM2.LT.ARLO) GO TO 150
       K=1
C CALCULATES INITIAL PENETRATION POINTS
       CALL PENPTS(N1,N2,X1,Y1,Z1,HT1,ARM1,ARRES)
       RAJL=ARRES(1)
       RAJR=ARRES(2)
       CALL PENPTS(N1,N2,X2,Y2,Z2,HT2,ARM2,ARRES)
       RBJL=ARRES(1)
       RBJR=ARRES(2)
C CALCULATES SUCCESSIVE PENETRATION POINTS
       DO 140 I=1,10000
       AR2R=AR2R+DAR
       ARM1=AR2R-ARP1
       ARM2=AR2R-ARP2
C CHECK FOR UPPER CALIBRATION RANGE
       IF(ARM1.GT.ARUP) GO TO 195
       IF(ARM2.GT.ARUP) GO TO 195
       CALL PENPTS(N1,N2,X1,Y1,Z1,HT1,ARM1,ARRES)
       RAIL=ARRES(1)
       RAIR=ARRES(2)
       CALL PENPTS(N1,N2,X2,Y2,Z2,HT2,ARM2,ARRES)
       RBIL=ARRES(1)
       RBIR=ARRES(2)
       IF(RAJL.EQ.1000.) ISL=10
       IF(RBJL.EQ.1000.) ISL=10
       IF(RBJR.EQ.1000.) ISR=10
       IF(RAJR.EQ.1000.) ISR=10
       IF(RAIL.EQ.1000.) ISL=10
       IF(RBIL.EQ.1000.) ISL=10
       IF(RBIR.EQ.1000.) ISR=10
       IF(RAIR.EQ.1000.) ISR=10
C CHECK FOR DIRECT HIT ON LEFT INTERSECTION,EVALUATE IF REQUIRED
       IF(ISL.NE.10) GO TO 164
       ISL=1
       GO TO 183
164    IF(RAIL.NE.RBIL) GO TO 180
       RES1(K)=RBIL
       RES2(K)=AR2R
       IF(K.EQ.2) GO TO 155
       K=2
       GO TO 183
C CHECK FOR INTERMEDIATE LEFT INTERSECTION,EVALUATE IF REQUIRED
180    IF(ISL.NE.10) GO TO 160
       ISL=1
       GO TO 183
160    IF((RAJL-RBJL)/(RAIL-RBIL).GT.0.0) GO TO 183
       AA=(AR2RJ-AR2R)/(RAJL-RAIL)
       BA=AR2R-AA*RAIL
       AB=(AR2RJ-AR2R)/(RBJL-RBIL)
       BB=AR2R-AB*RBIL
       RES1(K)=(BB-BA)/(AA-AB)
       RES2(K)=AA*RES1(K)+BA
```

```
      IF(K.EQ.2) GO TO 155
      K=2
C CHECK FOR DIRECT HIT ON RIGHT INTERSECTION,EVALUATE IF REQUIRED
183   IF(ISR.NE.10) GO TO 161
      ISR=1
      GO TO 163
161   IF(RAIR.NE.RBIR) GO TO 182
      RES1(K)=RBIR
      RES2(K)=AR2R
      IF(K.EQ.2) GO TO 155
      K=2
      GO TO 163
C CHECK FOR INTERMEDIATE RIGHT INTERSECTION,EVALUATE IF REQUIRED
182   IF(ISR.NE.10) GO TO 162
      ISR=1
      GO TO 163
162   IF((RAJR-RBJR)/(RAIR-RBIR).GT.0.0) GO TO 163
      AA=(AR2RJ-AR2R)/(RAJR-RAIR)
      BA=AR2R-AA*RAIR
      AB=(AR2RJ-AR2R)/(RBJR-RBIR)
      BB=AR2R-AB*RAIR
      RES1(K)=(BB-BA)/(AA-AB)
      RES2(K)=AA*RES1(K)+BA
      IF(K.EQ.2) GO TO 155
      K=2
C RE-INITIATE
163   AR2RJ=AR2R
      RAJL=RAIL
      RAJR=RAIR
      RBJL=RBIL
140   RBJR=RBIR
      WRITE(6,210)
210   FORMAT(' LOOP 140 THROUGH,NO POINTS FOUND')
      GO TO 155
195   IF(RES1(1).EQ.9000.0) GO TO 191
      IF(RES1(2).EQ.9000.0) GO TO 191
      IF(RES2(1).EQ.9000.0) GO TO 191
      IF(RES2(2).EQ.9000.0) GO TO 191
      GO TO 155
C EVALUATION OF INTERSECTIONS WHEN CURVES ARE ALMOST TANGENT
191   YGO=0.0
      IEPS=2
      CALL PENPTS(N2,N1,X3,Y3,Z3,HT1,YGO,ARRES)
      R1=ABS(ARRES(2)-ARRES(1))/2
      CALL PENPTS(N2,N1,X4,Y4,Z4,HT2,YGO,ARRES)
      R2=ABS(ARRES(2)-ARRES(1))/2
      RES2(1)=((R1**2)-(R2**2)+(ARP2**2))/(2.0*ARP2)
      RES2(2)=RES2(1)
      RES1(1)=SQRT(ABS((R1**2)-(RES2(1)**2)))
      RES1(2)=-RES1(1)
155   RETURN
      END
      SUBROUTINE PENPTS(NX,NY,X,Y,Z,HT,YG,XR)
      DIMENSION Y(40),ZX(40),Z(40,40),X(40),XR(10)
      IR=1
C SEARCH FOR J OF LOWER Y LINE
      DO 101 I=1,NY
      IF (Y(I).GE.YG) GO TO 102
101   CONTINUE
102   J=I-1
C COARSE SCAN FOR ZERO PASS
C NEXT 4 LINES ARE EXECUTED IN FIRST SEARCH ONLY
      YC=(YG-Y(J))/(Y(J+1)-Y(J))
      ZX(1)=(Z(1,J+1)-Z(1,J))*YC+Z(1,J)
      IF(ZX(1).EQ.HT) GO TO 107
      I3=2
112   DO 103 I=I3,NX
      ZX(I)=(Z(I,J+1)-Z(I,J))*YC+Z(I,J)
      ZM=(Z(I,J+1)-Z(I-1,J))*(YG-Y(J))/(Y(J+1)-Y(J))+Z(I-1,J)
      IF(ZX(I).EQ.HT) GO TO 108
```

```
      IF((ZX(I-1)-HT)/(ZX(I)-HT).LT.0.0) GO TO 104
103   CONTINUE
      GO TO 116
C CALCULATION OF PENETRATIONS IN EITHER TRIANGLE
104   XM=X(I-1)+(X(I)-X(I-1))*(YG-Y(J))/(Y(J+1)-Y(J))
      IF(ZM.EQ.HT) GO TO 105
      IF((ZX(I)-HT)/(ZM-HT).LT.0.0) GO TO 105
      DD=ABS(HT-ZM)+ABS(ZX(I-1)-HT)
      XS=X(I-1)+(XM-X(I-1))*ABS(ZX(I-1)-HT)/DD
      GO TO 110
105   XS=XM+(X(I)-XM)*ABS(ZM -HT)/(ABS(ZM-HT)+ABS(ZX(I)-HT))
      GO TO 110
C LOGICAL PENETRATIONS ACCUMULATION
107   XR(1)=X(1)
      I=1
      GO TO 113
108   XR(IR)=X(I)
      GO TO 111
110   XR(IR)=XS
111   IF(IR.EQ.2) GO TO 114
113   IR=2
      IB=I+1
      GO TO 112
116   IF(IR.EQ.1) GO TO 117
      XR(2)=1000.0
      GO TO 114
117   XR(1)=1000.0
      XR(2)=1000.0
114   RETURN
      END
      SUBROUTINE INTPLT(X,Y,Z,XG,YG,NX,NY,ZRES)
C LINEAR INTERPOLATION ON CALIBRATION SURFACE TO EVALUATE
C CP VALUE AT CO-ORDINATES XG,YG.THE RESULT RETURNED IS ZRES
      DIMENSION X(40),Y(40),Z(40,40)
C SEARCH FOR I OF LEFT X LINE
      DO 1 IC=1,NX
      IF(X(IC).GE.XG) GO TO 2
1     CONTINUE
2     I=IC-1
C SEARCH FOR J OF LOWER Y LINE
      DO 3 JC=1,NY
      IF(Y(JC).GE.YG) GO TO 4
3     CONTINUE
4     J=JC-1
      AL=(Y(J+1)-Y(J))/(X(I+1)-X(I))
      BL=Y(J)-AL*X(I)
      YCR=(AL*XG)+BL
      IF(YG.EQ.YCR) GO TO 7
      IF(YG.GT.YCR) GO TO 5
C RESULTING Z IS IN LOWER TRIANGLE
      X3=X(I+1)
      Y3=Y(J)
      Z3=Z(I+1,J)
      GO TO 6
C RASULTING Z IS IN UPPER TRIANGLE
5     X3=X(I)
      Y3=Y(J+1)
      Z3=Z(I,J+1)
C GENERAL CALCULATION:GOOD FOR BOTH TRIANGLES
6     X1=X(I)
      Y1=Y(J)
      Z1=Z(I,J)
      X2=X(I+1)
      Y2=Y(J+1)
      Z2=Z(I+1,J+1)
      AP=((Z1-Z3)*(Y1-Y2)-(Z1-Z2)*(Y1-Y3))/((X1-X3)*(Y1-Y2)-(X1-X2)
     1*(Y1-Y3))
      BP=((Z1-Z2)-AP*(X1-X2))/(Y1-Y2)
      CP=Z1-AP*X1-BP*Y1
      ZRES=AP*XG+BP*YG+CP
```

```
      GO TO 100
C RESULTING Z IS ON DIVIDING LINE
7     ZRES=(Z(I+1,J+1)-Z(I,J))*(X0-X(I))/(X(I+1)-X(I))+Z(I,J)
100   RETURN
      END
      SUBROUTINE MAXARY(Z,ZMAX,NX,NY)
C EVALUATION OF MAXIMUM CP OF CALIBRATION SURFACE MATRIX
      DIMENSION Z(40,40)
      DO 1 I=1,NX
      DO 1 J=1,NY
1     IF(Z(I,J).GE.ZMAX) ZMAX=Z(I,J)
      RETURN
      END
      SUBROUTINE MINARY(Z,ZMIN,NX,NY)
C EVALUATION OF MINIMUM CP OF CALIBRATION SURFACE MATRIX
      DIMENSION Z(40,40)
      DO 1 I=1,NX
      DO 1 J=1,NY
1     IF(Z(I,J).LE.ZMIN) ZMIN=Z(I,J)
      RETURN
      END
C END OF VOR
```

APPENDIX 2

LISTING OF SWVDR

```
C START OF SAVOR
      DIMENSION ALF(10),PHI(10),XA(40),YA(40),
     1ZA(40,40),XB(40),YB(40),ZB(40,40),XAX(40),YAX(40),
     2ZAX(40,40),XBX(40),YBX(40),ZBX(40,40)
119   FORMAT(2I5)
131   FORMAT(F7.1)
130   FORMAT(F5.1)
120   FORMAT(F8.5)
132   FORMAT(F10.5)
135   FORMAT(F15.5)
136   FORMAT(F17.5)
C READ PROBE CALIBRATION DATA
      READ(2,119) NX,NY
      DO 121 I=1,NX
      READ(2,130) XA(I)
121   CONTINUE
      DO 122 I=1,NY
      READ(2,130) YA(I)
122   CONTINUE
      DO 123 I=1,NX
      DO 123 J=1,NY
      READ(2,120) ZA(I,J)
123   CONTINUE
      DO 221 I=1,NX
      READ(2,130) XB(I)
221   CONTINUE
      DO 222 I=1,NY
      READ(2,130) YB(I)
222   CONTINUE
      DO 223 I=1,NX
      DO 223 J=1,NY
      READ(2,120) ZB(I,J)
223   CONTINUE
C TRANSFORM CALIBRATION MATRICES
      DO 224 I=1,NY
224   XAX(I)=YA(I)
      DO 225 I=1,NX
225   YAX(I)=XA(I)
      DO 226 I=1,NY
      DO 226 J=1,NX
226   ZAX(I,J)=ZA(J,I)
      DO 227 I=1,NY
227   XBX(I)=YB(I)
      DO 228 I=1,NX
228   YBX(I)=XB(I)
      DO 229 I=1,NY
      DO 229 J=1,NX
229   ZBX(I,J)=ZB(J,I)
C PROBE SETTINGS
      YPA=0.0
      YPB=55.
      XPA=0.0
      XPC=20.
      XPD=-20.
C PROGRAM CONSTANTS
      YRLO=-30.
      YRUP=30.
      YRIN=-30.
      XRLO=-30.
      XRUP=30.
      XRIN=-30.
      ICOPS=1
      NCOOPS=2
      NOSIM=2
      IIT=1
      ISCAN=1
      RELOPS=0.5
C EXPERIMENT SIMULATION
193   CONTINUE
401   FORMAT(F15.4)
```

```
      READ(5,401) PT
      WRITE(6,401) PT
      IF(NOSIM.EQ.2) GO TO 500
      READ(5,401) PS
      WRITE(6,401) PS
      READ(5,401) ALFA
      WRITE(6,401) ALFA
      READ(5,401) PHII
      WRITE(6,401) PHII
      CALL INTPLT(XA,YA,ZA,ALFA,PHII,NX,NY,CPA)
      ALFC=ALFA-XPC
      CALL INTPLT(XA,YA,ZA,ALFC,PHII,NX,NY,CPC)
      ALFD=ALFA-XPD
      CALL INTPLT(XA,YA,ZA,ALFD,PHII,NX,NY,CPD)
      PHIB=PHII-YPB
      CALL INTPLT(XB,YB,ZB,ALFA,PHIB,NX,NY,CPB)
      PDYN=PT-PS
      PPA=CPA*PDYN+PS
      PPB=CPB*PDYN+PS
      PPC=CPC*PDYN+PS
      PPD=CPD*PDYN+PS
C READ MEASUREMENTS DATA
      IF(NOSIM.EQ.1) GO TO 501
500   READ(5,135) PPA
      WRITE(6,136) PPA
      READ(5,135) PPB
      WRITE(6,136) PPB
      READ(5,135) PPC
      WRITE(6,136) PPC
      READ(5,135) PPD
      WRITE(6,136) PPD
501   READ(5,135) PSIN
      WRITE(6,136) PSIN
      IF(ISCAN.EQ.1) GO TO 181
180   IF(ISCAN.GT.1000) GO TO 305
      III=1
      ICCPS=1
      ISCAN=ISCAN+1
      PSIN=PSIN+5.0
      WRITE(6,136) PSIN
181   CONTINUE
      PS=PSIN
      PSC=PSIN
      PST=PS
C CALCULATES PRESSURE COEFFICIENTS
300   CONTINUE
      PDYN=PT-PS
      CPA=(PPA-PS)/PDYN
      CPB=(PPB-PS)/PDYN
      CPC=(PPC-PS)/PDYN
      CPD=(PPD-PS)/PDYN
C CORRECTS PS ASSUMPTION TO ENSURE CP ARE IN CALIBRATION RANGE
      IF(NOCOPS.EQ.2) GO TO 301
      ZAMAX=-10000.
      CALL MAXARY(ZA,ZAMAX,NX,NY)
      IF(CPA.GT.ZAMAX) PSC=1.01*PS
      IF(CPC.GT.ZAMAX) PSC=1.01*PS
      IF(CPD.GT.ZAMAX) PSC=1.01*PS
      ZBMAX=-10000.
      CALL MAXARY(ZB,ZBMAX,NX,NY)
      IF(CPB.GT.ZBMAX) PSC=1.01*PS
      ZAMIN=10000.
      CALL MINARY(ZA,ZAMIN,NX,NY)
      IF(CPA.LT.ZAMIN) PSC=0.99*PS
      IF(CPC.LT.ZAMIN) PSC=0.99*PS
      IF(CPD.LT.ZAMIN) PSC=0.99*PS
      ZBMIN=10000.
      CALL MINARY(ZB,ZBMIN,NX,NY)
      IF(CPB.LT.ZBMIN) PSC=0.99*PS
      IF(PSC.EQ.PST) GO TO 301
```

FILE: SWVCR   FORTRAN P1   N   P   S

```
      ICOPS=ICOPS+1
      IF(ICOPS.GT.10) GO TO 180
      PS=PSC
      PST=PSC
      GO TO 300
301   CONTINUE
C CALCULATES TWO ALF & PHI ANGLES FOR 'I' & 'II'
      CALL INTSCS(YPA,YPB,YRLO,YRUP,YRIN,
     1CPA,CPB,ALF,PHI,XA,YA,ZA,
     2XB,YB,ZB,XAX,YAX,ZAX,XBX,YBX,ZBX,NX,NY)
      ALF1=ALF(1)
      ALF2=ALF(2)
      PHI1=PHI(1)
      PHI2=PHI(2)
C CALCULATES TWO ALF & PHI ANGLES FOR 'I' & 'III'
      CALL INTSCS(XPA,XPC,XRLO,XRUP,XRIN,
     1CPA,CPC,PHI,ALF,YAX,YAX,ZAX,
     2XAX,YAX,ZAX,XA,YA,ZA,XA,YA,ZA,NY,NX)
      ALF3=ALF(1)
      ALF4=ALF(2)
      PHI3=PHI(1)
      PHI4=PHI(2)
C SELECTS THE PROPER ALF & PHI ANGLES OUT OF THE FOUR VALUES
      FALF=((ALF1+ALF2+ALF3+ALF4)-AMAX1(ALF1,ALF2,ALF3,
     1ALF4)-AMIN1(ALF1,ALF2,ALF3,ALF4))/2
      FPHI=((PHI1+PHI2+PHI3+PHI4)-AMAX1(PHI1,PHI2,PHI3,PHI4)
     1-AMIN1(PHI1,PHI2,PHI3,PHI4))/2
C CALCULATES NEW PS FROM DATA OF POSITION D AND CORRECTS PT
      ALFDN=FALF-XPD
      CALL INTPLT(XA,YA,ZA,ALFDN,FPHI,NX,NY,CPD)
      ALFCN=FALF-XPC
      CALL INTPLT(XA,YA,ZA,ALFCN,FPHI,NX,NY,CPC)
      PSN=(PPC*CPD-PPD*CPC)/(CPD-CPC)
907   WRITE(6,911)IIT,FALF,FPHI,PSN
911   FORMAT(' ITERATION VALUES',I5,10X,2F6.2,F10.2)
C CONVERGENCE TESTS
      EPSPS=ABS((PSN-PS)/PS)
      EPSPSG=.0000000000099*(PT**2.0)-.00000098*PT
      IF(IEPS1.EQ.2) EPSPSG=.0000000009*(PT**2.0)-.0000008*PT
      IF(IEPS2.EQ.2) EPSPSG=.0000000009*(PT**2.0)-.000008*PT
      IF(EPSPS.GT.EPSPSG) GO TO 310
      GO TO 303
310   CONTINUE
      IIT=IIT+1
      IF(IIT.GT.10) GO TO 180
      PS=PS+RELXPS*(PSN-PS)
      PSC=PS
      PST=PS
      GO TO 300
303   WRITE(6,171) FALF
171   FORMAT('0',' VELOCITY YAW IS' ,F15.2)
      WRITE(6,172) FPHI
172   FORMAT(' VELOCITY PITCH IS' ,F15.2)
      WRITE(6,173) PSN
173   FORMAT(' STATIC PRESSURE IS' ,F15.3)
      WRITE(6,174) PT
174   FORMAT(' TOTAL PRESSURE IS' ,F15.3)
      AM=SQRT(((PT/PSN)**0.285714-1.0)*5.0)
      WRITE(6,175) AM
175   FORMAT(' MACH NUMBER IS' ,F15.5)
      GO TO 193
305   WRITE(6,307)
307   FORMAT(' SCANNED 1000 TIMES')
      GO TO 193
      END
      SUBROUTINE INTSCS(ARP1,ARP2,ARLO,ARUP,ARIN,
     1HT1,HT2,RES1,RES2,X1,Y1,Z1,
     2X2,Y2,Z2,X3,Y3,Z3,X4,Y4,Z4,N1,N2)
      DIMENSION RES1(10),RES2(10),ARRES(10),X1(40),Y1(40),
     1Z1(40,40),X2(40),Y2(40),Z2(40,40),X3(40),Y3(40),
```

```
     2Z3(40,40),X4(40),Y4(40),Z4(40,40)
      RES1(1)=9000.
      RES1(2)=9000.
      RES2(1)=9000.
      RES2(2)=9000.
      ISL=1
      ISR=1
      DAR=1.
      AR2R=ARIN
      GO TO 152
150   AR2R=AR2R+DAR
      AR2RJ=AR2R
152   ARM1=AR2R-ARP1
      ARM2=AR2R-ARP2
C CHECK FOR LOWER CALIBRATION RANGE
      IF(ARM1.LT.ARLO) GO TO 150
      IF(ARM2.LT.ARLO) GO TO 150
      K=1
C CALCULATES INITIAL PENETRATION POINTS
      CALL PENPTS(N1,N2,X1,Y1,Z1,HT1,ARM1,ARRES)
      RAJL=ARRES(1)
      RAJR=ARRES(2)
      CALL PENPTS(N1,N2,X2,Y2,Z2,HT2,ARM2,ARRES)
      RBJL=ARRES(1)
      RBJR=ARRES(2)
C CALCULATES SUCCESSIVE PENETRATION POINTS
      DO 140 I=1,10000
      AR2R=AR2R+DAR
      ARM1=AR2R-ARP1
      ARM2=AR2R-ARP2
C CHECK FOR UPPER CALIBRATION RANGE
      IF(ARM1.GT.ARUP) GO TO 195
      IF(ARM2.GT.ARUP) GO TO 195
      CALL PENPTS(N1,N2,X1,Y1,Z1,HT1,ARM1,ARRES)
      RAIL=ARRES(1)
      RAIR=ARRES(2)
      CALL PENPTS(N1,N2,X2,Y2,Z2,HT2,ARM2,ARRES)
      RBIL=ARRES(1)
      RBIR=ARRES(2)
      IF(RAJL.EQ.1000.) ISL=10
      IF(RBJL.EQ.1000.) ISL=10
      IF(RBJR.EQ.1000.) ISR=10
      IF(RAJR.EQ.1000.) ISR=10
      IF(RAIL.EQ.1000.) ISL=10
      IF(RBIL.EQ.1000.) ISL=10
      IF(RBIR.EQ.1000.) ISR=10
      IF(RAIR.EQ.1000.) ISR=10
C CHECK FOR DIRECT FIT OF LEFT INTERSECTION,EVALUATE IF REQUIRED
      IF(ISL.NE.10) GO TO 164
      ISL=1
      GO TO 183
164   IF(RAIL.NE.RBIL) GO TO 180
      RES1(K)=RBIL
      RES2(K)=AR2R
      IF(K.EQ.2) GO TO 155
      K=2
      GO TO 183
C CHECK FOR INTERMEDIATE LEFT INTERSECTION,EVALUATE IF REQUIRED
180   IF(ISL.NE.10) GO TO 160
      ISL=1
      GO TO 183
160   IF((RAJL-RBJL)/(RAIL-RBIL).GT.0.0) GO TO 183
      AA=(AR2RJ-AR2R)/(RAJL-RAIL)
      BA=AR2R-AA*RAIL
      AB=(AR2RJ-AR2R)/(RBJL-RBIL)
      BB=AR2R-AB*RBIL
      RES1(K)=(BB-BA)/(AA-AB)
      RES2(K)=AA*RES1(K)+BA
      IF(K.EQ.2) GO TO 155
      K=2
```

```
C CHECK FOR DIRECT HIT ON RIGHT INTERSECTION,EVALUATE IF REQUIRED
183    IF(ISR.NE.10) GO TO 161
       ISR=1
       GO TO 163
161    IF(RAIR.NE.RBIR) GO TO 182
       RES1(K)=RBIR
       RES2(K)=AR2R
       IF(K.EQ.2) GO TO 155
       K=2
       GO TO 163
C CHECK FOR INTERMEDIATE RIGHT INTERSECTION,EVALUATE IF REQUIRED
182    IF(ISR.NE.10) GO TO 162
       ISR=1
       GO TO 163
162    IF((RAJR-RBJR)/(RAIR-RBIR).GT.0.0) GO TO 163
       AA=(AR2RJ-AR2R)/(RAJR-RAIR)
       EA=AR2R-AA*RAIR
       AB=(AR2RJ-AR2R)/(RBJR-RBIR)
       EB=AR2R-AB*RBIR
       RES1(K)=(EB-EA)/(AA-AB)
       RES2(K)=AA*RES1(K)+EA
       IF(K.EQ.2) GO TO 155
       K=2
C RE-INITIATE
163    AR2RJ=AR2R
       RAJL=RAIL
       RAJR=RAIR
       RBJL=RBIL
       RBJR=RBIR
140    CONTINUE
       WRITE(6,210)
210    FORMAT(' LOOP 140 THROUGH,NO POINTS FOUND')
       GO TO 155
195    IF(RES1(1).EQ.9000.0) GO TO 191
       IF(RES1(2).EQ.9000.0) GO TO 191
       IF(RES2(1).EQ.9000.0) GO TO 191
       IF(RES2(2).EQ.9000.0) GO TO 191
       GO TO 155
C EVALUATION OF INTERSECTIONS WHEN CURVES ARE ALMOST TANGENT
191    YGO=0.0
       CALL PENPTS(N2,N1,X3,Y3,Z3,HT1,YGO,ARRES)
       R1=ABS(ARRES(2)-ARRES(1))/2
       CALL PENPTS(N2,N1,X4,Y4,Z4,HT2,YGO,ARRES)
       R2=ABS(ARRES(2)-ARRES(1))/2
       RES2(1)=((R1**2)-(R2**2)+(ARP2**2))/(2.0*ARP2)
       RES2(2)=RES2(1)
       RES1(1)=SQRT(ABS((R1**2)-(RES2(1)**2)))
       RES1(2)=-RES1(1)
155    RETURN
       END
       SUBROUTINE PENPTS(NX,NY,X,Y,Z,HT,YG,XR)
       DIMENSION Y(40),ZX(40),Z(40,40),X(40),XR(10)
       IR=1
C SEARCH FOR J OF LOWER Y LINE
       DO 101 I=1,NY
       IF (Y(I).GE.YG) GO TO 102
101    CONTINUE
102    J=I-1
C COARSE SCAN FOR ZERO PASS
C NEXT 4 LINES ARE EXECUTED IN FIRST SEARCH ONLY
       YC=(YG-Y(J))/(Y(J+1)-Y(J))
       ZX(1)=(Z(1,J+1)-Z(1,J))*YC+Z(1,J)
       IF(ZX(1).EQ.HT) GO TO 107
       IB=2
112    DO 103 I=IB,NX
       ZX(I)=(Z(I,J+1)-Z(I,J))*YC+Z(I,J)
       Z4=(Z(I,J+1)-Z(I-1,J))*(YG-Y(J))/(Y(J+1)-Y(J))+Z(I-1,J)
       IF(ZX(I).EQ.HT) GO TO 108
       IF((ZX(I-1)-HT)/(ZX(I)-HT).LT.0.0) GO TO 104
103    CONTINUE
```

```
      GO TO 116
C CALCULATION OF PENETRATIONS IN EITHER TRIANGLE
104   XM=X(I-1)+(X(I)-X(I-1))*(YG-Y(J))/(Y(J+1)-Y(J))
      IF(ZM.EQ.HT) GO TO 105
      IF((ZX(I)-HT)/(ZM-HT).LT.0.0) GO TO 105
      DD=ABS(HT-ZM)+ABS(ZX(I-1)-HT)
      XS=X(I-1)+(XM-X(I-1))*ABS(ZX(I-1)-HT)/DD
      GO TO 110
105   XS=XM+(X(I)-XM)*ABS(ZM -HT)/(ABS(ZM-HT)+ABS(ZX(I)-HT))
      GO TO 110
C LOGICAL PENETRATIONS ACCUMULATION
107   XR(1)=X(1)
      I=1
      GO TO 113
108   XR(IR)=X(I)
      GO TO 111
110   XR(IR)=XS
111   IF(IR.EQ.2) GO TO 114
113   IR=2
      IB=I+1
      GO TO 112
116   IF(IF.EQ.1) GO TO 117
      XR(2)=1000.0
      GO TO 114
117   XR(1)=1000.0
      XR(2)=1000.0
114   RETURN
      END
      SUBROUTINE INTPLT(X,Y,Z,XG,YG,NX,NY,ZRES)
C LINEAR INTERPOLATION ON CALIBRATION SURFACE TO EVALUATE
C CP VALUE AT CO-ORDINATES XG,YG. THE RESULT RETURNED IS ZRES
      DIMENSION X(40),Y(40),Z(40,40)
C SEARCH FOR I OF LEFT X LINE
      DO 1 IC=1,NX
      IF(X(IC).GE.XG) GO TO 2
1     CONTINUE
2     I=IC-1
C SEARCH FOR J OF LOWER Y LINE
      DO 3 JC=1,NY
      IF(Y(JC).GE.YG) GO TO 4
3     CONTINUE
4     J=JC-1
      AL=(Y(J+1)-Y(J))/(X(I+1)-X(I))
      BL=Y(J)-AL*X(I)
      YCR=(AL*XG)+BL
      IF(YG.EQ.YCR) GO TO 7
      IF(YG.GT.YCR) GO TO 5
C RESULTING Z IS IN LOWRER TRIANGLE
      X3=X(I+1)
      Y3=Y(J)
      Z3=Z(I+1,J)
      GO TO 6
C RASULTING Z IS IN UPPER TRIANGLE
5     X3=X(I)
      Y3=Y(J+1)
      Z3=Z(I,J+1)
C GENERAL CALCULATION:GOOD FOR BOTH TRIANGLES
6     X1=X(I)
      Y1=Y(J)
      Z1=Z(I,J)
      X2=X(I+1)
      Y2=Y(J+1)
      Z2=Z(I+1,J+1)
      AP=((Z1-Z3)*(Y1-Y2)-(Z1-Z2)*(Y1-Y3))/((X1-X3)*(Y1-Y2)-(X1-X2)
     1*(Y1-Y3))
      BP=((Z1-Z2)-AP*(X1-X2))/(Y1-Y2)
      CP=Z1-AP*X1-BP*Y1
      ZRES=AP*XG+BP*YG+CP
      GO TO 100
C RESULTING Z IS ON DIVIDING LINE
```

```
FILE: SWVOR    FORTRAN  P1                   N     P          S

7      ZRES=(Z(I+1,J+1)-Z(I,J))*(XG-X(I))/(X(I+1)-X(I))+Z(I,J)         INT04210
100    RETURN                                                           INT04220
       END                                                              INT04230
       SUBROUTINE MAXARY(Z,ZMAX,NX,NY)                                  INT04240
C EVALUATION OF MAXIMUM CP OF CALIBRATION SURFACE MATRIX                 INT04250
       DIMENSION Z(40,40)                                               INT04260
       DO 1 I=1,NX                                                      INT04270
       DO 1 J=1,NY                                                      INT04280
1      IF(Z(I,J).GE.ZMAX) ZMAX=Z(I,J)                                   INT04290
       RETURN                                                           INT04300
       END                                                              INT04310
       SUBROUTINE MINARY(Z,ZMIN,NX,NY)                                  INT04320
C EVALUATION OF MINIMUM CP OF CALIBRATION SURFACE MATRIX                 INT04330
       DIMENSION Z(40,40)                                               INT04340
       DO 1 I=1,NX                                                      INT04350
       DO 1 J=1,NY                                                      INT04360
1      IF(Z(I,J).LE.ZMIN) ZMIN=Z(I,J)                                   INT04370
       RETURN                                                           INT04380
       END                                                              INT04390
C END OF SWVOR                                                          INT04400
                                                                        INT04410
```

APPENDIX 3

LISTING OF WVDR

```
07/08/79  16.45.53

FILE: WVCR     FORTRAN  P1                    N      P             S

C START OF WVCR
      DIMENSION ALF(10),PHI(10),XA(40),YA(40),
     1ZA(40,40),XB(40),YB(40),ZB(40,40),XAX(40),YAX(40),
     2ZAX(40,40),XBX(40),YBX(40),ZBX(40,40)
119   FORMAT(2I5)
131   FORMAT(F7.1)
130   FORMAT(F5.1)
120   FORMAT(F8.5)
132   FORMAT(F10.5)
135   FORMAT(F15.5)
136   FORMAT(F17.5)
C READ PROBE CALIBRATION DATA
      READ(2,119) NX,NY
      DO 121 I=1,NX
      READ(2,130) XA(I)
121   CONTINUE
      DO 122 I=1,NY
      READ(2,130) YA(I)
122   CONTINUE
      DO 123 I=1,NX
      DO 123 J=1,NY
      READ(2,120) ZA(I,J)
123   CONTINUE
      DO 221 I=1,NX
      READ(2,130) XB(I)
221   CONTINUE
      DO 222 I=1,NY
      READ(2,130) YB(I)
222   CONTINUE
      DO 223 I=1,NX
      DO 223 J=1,NY
      READ(2,120) ZB(I,J)
223   CONTINUE
C TRANSFORM CALIBRATION MATRICES
      DO 224 I=1,NY
224   XAX(I)=YA(I)
      DO 225 I=1,NX
225   YAX(I)=XA(I)
      DO 226 I=1,NY
      DO 226 J=1,NX
226   ZAX(I,J)=ZA(J,I)
      DO 227 I=1,NY
227   XBX(I)=YB(I)
      DO 228 I=1,NX
228   YBX(I)=XB(I)
      DO 229 I=1,NY
      DO 229 J=1,NX
229   ZBX(I,J)=ZB(J,I)
C PROBE SETTINGS
      YPA=0.0
      YPB=55.
      XPA=0.0
      XPC=20.
      XPD=-20.
C PROGRAM CONSTANTS
      YRLC=-30.
      YRUP=30.
      YRIN=-30.
      XRLC=-30.
      XRUP=30.
      XRIN=-30.
      ICOPS=1
      NOCOPS=2
      NOSIM=1
      IIT=1
      ISCAN=1
      RELAPS=0.5
C EXPERIMENT SIMULATION
193   CONTINUE
401   FORMAT(F15.4)
```

```
      READ(5,401) PT
      WRITE(6,401) PT
      IF(NCSIM.EQ.2) GO TO 500
      READ(5,401) PS
      WRITE(6,401) PS
      READ(5,401) ALFA
      WRITE(6,401) ALFA
      READ(5,401) PHII
      WRITE(6,401) PHII
      CALL INTPLT(XA,YA,ZA,ALFA,PHII,NX,NY,CPA)
      ALFC=ALFA-XPC
      CALL INTPLT(XA,YA,ZA,ALFC,PHII,NX,NY,CPC)
      ALFD=ALFA-XPD
      CALL INTPLT(XA,YA,ZA,ALFD,PHII,NX,NY,CPD)
      PHIB=PHII-YPB
      CALL INTPLT(XB,YB,ZB,ALFA,PHIB,NX,NY,CPB)
      WRITE(6,402) CPA,CPB,CPC,CPD
402   FORMAT(' TRUE CPS',4F10.5)
      PDYN=PT-PS
      PPA=CPA*PDYN+PS
      PPB=CPB*PDYN+PS
      PPC=CPC*PDYN+PS
      PPD=CPD*PDYN+PS
      WRITE(6,403) PPA,PPB,PPC,PPD
403   FORMAT(' SIMUL PPS',4F10.2)
C READ MEASUREMENTS DATA
      IF(NCSIM.EQ.1) GO TO 501
500   READ(5,135) PPA
      WRITE(6,136) PPA
      READ(5,135) PPB
      WRITE(6,136) PPB
      READ(5,135) PPC
      WRITE(6,136) PPC
      READ(5,135) PPD
      WRITE(6,136) PPD
501   READ(5,135) PSIN
      WRITE(6,136) PSIN
      IF(ISCAN.EQ.1) GO TO 181
180   IF(ISCAN.GT.1000) GO TO 305
      IIT=1
      ICOPS=1
      ISCAN=ISCAN+1
      PSIN=PSIN+5.0
      WRITE(6,136) PSIN
181   CONTINUE
      PS=PSIN
      PSC=PS
      PST=PS
C CALCULATES PRESSURE COEFFICIENTS
300   CONTINUE
      WRITE(6,940) IIT
940   FORMAT('0',' ITERATION NO. ',I5)
      PDYN=PT-PS
      CPA=(PPA-PS)/PDYN
      CPB=(PPB-PS)/PDYN
      CPC=(PPC-PS)/PDYN
      CPD=(PPD-PS)/PDYN
908   WRITE(6,912) CPA,CPB,CPC,CPD
912   FORMAT(' CALC CPS',4F10.5)
C CORRECTS PS ASSUMPTION TO ENSURE CP ARE IN CALIBRATION RANGE
      IF(ICOPS.EQ.2) GO TO 301
      ZAMAX=-10000.
      CALL MAXARY(ZA,ZAMAX,NX,NY)
      IF(CPA.GT.ZAMAX) PSC=1.01*PS
      IF(CPC.GT.ZAMAX) PSC=1.01*PS
      IF(CPD.GT.ZAMAX) PSC=1.01*PS
      ZBMAX=-10000.
      CALL MAXARY(ZB,ZBMAX,NX,NY)
      IF(CPB.GT.ZBMAX) PSC=1.01*PS
      ZAMIN=10000.
```

FILE: WVCF FORTRAN P1 N P S

```
      CALL MINARY(ZA,ZAMIN,NX,NY)
      IF(CPA.LT.ZAMIN) PSC=0.99*PS
      IF(CPC.LT.ZAMIN) PSC=0.99*PS
      IF(CPD.LT.ZAMIN) PSC=0.99*PS
      ZBMIN=10000.
      CALL MINARY(ZB,ZBMIN,NX,NY)
      IF(CPB.LT.ZBMIN) PSC=0.99*PS
      IF(PSC.EQ.PST) GO TO 301
      ICCPS=ICCPS+1
963   WRITE(6,964) ICCPS
964   FORMAT(' ICCPS IS' ,I5)
      IF(ICCPS.GT.10) GO TO 180
      PS=PSC
      PST=PSC
920   WRITE(6,921) PSC
921   FORMAT(' PSC IS     ',F20.5)
      GO TO 300
301   CONTINUE
C CALCULATES TWO ALF & PHI ANGLES FOR 'I' & 'II'
      CALL INTSOS(YPA,YPB,YRLO,YRUP,YRIN,
     1CPA,CPB,ALF,PHI,XA,YA,ZA,
     2XB,YB,ZB,XAX,YAX,ZAX,XBX,YBX,ZBX,NX,NY)
      ALF1=ALF(1)
      ALF2=ALF(2)
      PHI1=PHI(1)
      PHI2=PHI(2)
C CALCULATES TWO ALF & PHI ANGLES FOR 'I' & 'III'
      CALL INTSOS(XPA,XPC,XRLO,XRUP,XRIN,
     1CPA,CPC,PHI,ALF,XAX,YAX,ZAX,
     2XAX,YAX,ZAX,XA,YA,ZA,XA,YA,ZA,NY,NX)
      ALF3=ALF(1)
      ALF4=ALF(2)
      PHI3=PHI(1)
      PHI4=PHI(2)
C SELECTS THE PROPER ALF & PHI ANGLES OUT OF THE FOUR VALUES
      FALF=((ALF1+ALF2+ALF3+ALF4)-AMAX1(ALF1,ALF2,ALF3,
     1ALF4)-AMIN1(ALF1,ALF2,ALF3,ALF4))/2
      FPHI=((PHI1+PHI2+PHI3+PHI4)-AMAX1(PHI1,PHI2,PHI3,PHI4)
     1-AMIN1(PHI1,PHI2,PHI3,PHI4))/2
900   WRITE(6,901)FALF,FPHI
901   FORMAT(' SELECTED',2F10.2)
C CALCULATES NEW PS FROM DATA OF POSITION D AND CORRECTS PT
      ALFDN=FALF-XPD
      CALL INTPLT(XA,YA,ZA,ALFDN,FPHI,NX,NY,CPD)
      ALFCN=FALF-XPC
      CALL INTPLT(XA,YA,ZA,ALFCN,FPHI,NX,NY,CPC)
909   WRITE(6,910) CPC,CPD
910   FORMAT(' NEW CPC & CPD CALLED',2F15.5)
      PSN=(PPC*CPD-PPD*CPC)/(CPD-CPC)
907   WRITE(6,911) FALF,FPHI,PSN,PT
911   FORMAT(' ITERATION VALUES',2F6.2,2F10.2)
C CONVERGENCE TESTS
950   DELPS=PSN-PS
951   WRITE(6,952) PS,PSN,DELPS
952   FORMAT(' EPSPS VALUES',3F15.5)
      EPSPS=ABS((PSN-PS)/PS)
960   WRITE(6,961) EPSPS
961   FORMAT(' EPSPS IS' ,F25.15)
      EPSPSG=.0000000000099*(PT**2.0)-.00000098*PT
      IF(IEPS1.EQ.2) EPSPSG=.00000000009*(PT**2.0)-.0000008*PT
      IF(IEPS2.EQ.2) EPSPSG=.00000000009*(PT**2.0)-.0000008*PT
      IF(EPSPS.GT.EPSPSG) GO TO 310
      GO TO 303
310   CONTINUE
      IIT=IIT+1
      IF(IIT.GT.10) GO TO 305
      PS=PS+RELXPS*(PSN-PS)
      PSC=PS
      PST=PS
924   WRITE(6,925) PS
```

```
FILE: WVDF      FORTRAN  P1                         N     P          S
525     FORMAT(' CURR PS IS ',F15.5)
        GO TO 300
303     WRITE(6,171) FALF,FPH1,PSM,PT
171     FORMAT(3F15.2,F15.3)
        GO TO 193
305     WRITE(6,307)
307     FORMAT(' SCANNED 100 TIMES')
        GO TO 193
        END
        SUBROUTINE INTSCS(ARP1,ARP2,ARLO,ARUP,ARIN,
       1HT1,HT2,RES1,RES2,X1,Y1,Z1,
       2X2,Y2,Z2,X3,Y3,Z3,X4,Y4,Z4,N1,N2)
        DIMENSION RES1(10),RES2(10),ARRES(10),X1(40),Y1(40),
       1Z1(40,40),X2(40),Y2(40),Z2(40,40),X3(40),Y3(40),
       2Z3(40,40),X4(40),Y4(40),Z4(40,40)
        RES1(1)=9000.
        RES1(2)=9000.
        RES2(1)=9000.
        RES2(2)=9000.
        ISL=1
        ISR=1
        DAR=1.
        AR2R=ARIN
        GO TO 152
150     AR2R=AR2R+DAR
        AR2RJ=AR2R
152     ARM1=AR2R-ARP1
        ARM2=AR2R-ARP2
C CHECK FOR LOWER CALIBRATION RANGE
        IF(ARM1.LT.ARLO) GO TO 150
        IF(ARM2.LT.ARLO) GO TO 150
        K=1
C CALCULATES INITIAL PENETRATION POINTS
        CALL PENPTS(N1,N2,X1,Y1,Z1,HT1,ARM1,ARRES)
        RAJL=ARRES(1)
        RAJR=ARRES(2)
        CALL PENPTS(N1,N2,X2,Y2,Z2,HT2,ARM2,ARRES)
        RBJL=ARRES(1)
        RBJR=ARRES(2)
C CALCULATES SUCCESIVE PENETRATION POINTS
        DO 140 I=1,10000
        AR2R=AR2R+DAR
        ARM1=AR2R-ARP1
        ARM2=AR2R-ARP2
C CHECK FOR UPPER CALIBRATION RANGE
        IF(ARM1.GT.ARUP) GO TO 195
        IF(ARM2.GT.ARUP) GO TO 195
        CALL PENPTS(N1,N2,X1,Y1,Z1,HT1,ARM1,ARRES)
        RAIL=ARRES(1)
        RAIR=ARRES(2)
        CALL PENPTS(N1,N2,X2,Y2,Z2,HT2,ARM2,ARRES)
        RBIL=ARRES(1)
        RBIR=ARRES(2)
        IF(RAJL.EQ.1000.) ISL=10
        IF(RBJL.EQ.1000.) ISL=10
        IF(RBJR.EQ.1000.) ISR=10
        IF(RAJR.EQ.1000.) ISR=10
        IF(RAIL.EQ.1000.) ISL=10
        IF(RBIL.EQ.1000.) ISL=10
        IF(RBIR.EQ.1000.) ISR=10
        IF(RAIR.EQ.1000.) ISR=10
C CHECK FOR DIRECT HIT ON LEFT INTERSECTION,EVALUATE IF REQUIRED
        IF(ISL.NE.10) GO TO 164
        ISL=1
        GO TO 183
164     IF(RAIL.NE.RBIL) GO TO 180
        RES1(K)=RBIL
        RES2(K)=AR2R
        IF(K.EQ.2) GO TO 155
        K=2
```

'6

```
      GO TO 183
C CHECK FOR INTERMEDIATE LEFT INTERSECTION,EVALUATE IF REQUIRED
180   IF(ISL.NE.10) GO TO 160
      ISL=1
      GO TO 183
160   IF((RAJL-RBJL)/(RAIL-RBIL).GT.0.0) GO TO 183
      AA=(AR2RJ-AR2R)/(RAJL-RAIL)
      BA=AR2R-AA*RAIL
      AB=(AR2RJ-AR2R)/(RBJL-RBIL)
      BB=AR2R-AB*RBIL
      RES1(K)=(BB-BA)/(AA-AB)
      RES2(K)=AA*RES1(K)+BA
      IF(K.EQ.2) GO TO 155
      K=2
C CHECK FOR DIRECT HIT ON RIGHT INTERSECTION,EVALUATE IF REQUIRED
183   IF(ISR.NE.10) GO TO 161
      ISR=1
      GO TO 163
161   IF(RAIR.NE.RBIR) GO TO 182
      RES1(K)=RBIR
      RES2(K)=AR2R
      IF(K.EQ.2) GO TO 155
      K=2
      GO TO 163
C CHECK FOR INTERMEDIATE RIGHT INTERSECTION,EVALUATE IF REQUIRED
182   IF(ISR.NE.10) GO TO 162
      ISR=1
      GO TO 163
162   IF((RAJR-RBJR)/(RAIR-RBIR).GT.0.0) GO TO 163
      AA=(AR2RJ-AR2R)/(RAJR-RAIR)
      BA=AR2R-AA*RAIR
      AB=(AR2RJ-AR2R)/(RBJR-RBIR)
      BB=AR2R-AB*RBIR
      RES1(K)=(BB-BA)/(AA-AB)
      RES2(K)=AA*RES1(K)+BA
      IF(K.EQ.2) GO TO 155
      K=2
C RE-INITIATE
163   AR2RJ=AR2R
      RAJL=RAIL
      RAJR=RAIR
      RBJL=RBIL
      RBJR=RBIR
140   CONTINUE
      WRITE(6,210)
210   FORMAT(' LOOP 140 THROUGH, NO POINTS FOUND')
      GO TO 155
195   IF(RES1(1).EQ.9000.0) GO TO 191
      IF(RES1(2).EQ.9000.0) GO TO 191
      IF(RES2(1).EQ.9000.0) GO TO 191
      IF(RES2(2).EQ.9000.0) GO TO 191
      GO TO 155
C EVALUATION OF INTERSECTIONS WHEN CURVES ARE ALMOST TANGENT
191   YGO=0.0
      CALL PENPTS(N2,N1,X3,Y3,Z3,HT1,YGO,ARRES)
      R1=ABS(ARRES(2)-ARRES(1))/2
      CALL PENPTS(N2,N1,X4,Y4,Z4,HT2,YGO,ARRES)
      R2=ABS(ARRES(2)-ARRES(1))/2
      RES2(1)=((R1**2)-(R2**2)+(ARP2**2))/(2.0*ARP2)
      RES2(2)=RES2(1)
      RES1(1)=SQRT(ABS((R1**2)-(RES2(1)**2)))
      RES1(2)=-RES1(1)
155   RETURN
      END
      SUBROUTINE PENPTS(NX,NY,X,Y,Z,HT,YG,XR)
      DIMENSION Y(40),ZX(40),Z(40,40),X(40),XR(10)
      IR=1
C SEARCH FOR J OF LOWER Y LINE
      DO 101 I=1,NY
      IF (Y(I).GE.YG) GO TO 102
```

```
101     CONTINUE
102     J=J-1
C COARSE SCAN FOR ZERO PASS
C NEXT 4 LINES ARE EXECUTED IN FIRST SEARCH ONLY
        YC=(YG-Y(J))/(Y(J+1)-Y(J))
        ZX(1)=(Z(1,J+1)-Z(1,J))*YC+Z(1,J)
        IF(ZX(1).EQ.HT) GO TO 107
        IB=2
112     DO 103 I=IB,NX
        ZX(I)=(Z(I,J+1)-Z(I,J))*YC+Z(I,J)
        ZM=(Z(I,J+1)-Z(I-1,J))*(YG-Y(J))/(Y(J+1)-Y(J))+Z(I-1,J)
        IF(ZX(I).EQ.HT) GO TO 108
        IF((ZX(I-1)-HT)/(ZX(I)-HT).LT.0.0) GO TO 104
103     CONTINUE
        GO TO 116
C CALCULATION OF PENETRATIONS IN EITHER TRIANGLE
104     XM=X(I-1)+(X(I)-X(I-1))*(YG-Y(J))/(Y(J+1)-Y(J))
        IF(ZM.EQ.HT) GO TO 105
        IF((ZX(I)-HT)/(ZM-HT).LT.0.0) GO TO 105
        DD=ABS(HT-ZM)+ABS(ZX(I-1)-HT)
        XS=X(I-1)+(XM-X(I-1))*ABS(ZX(I-1)-HT)/DD
        GO TO 110
105     XS=XM+(X(I)-XM)*ABS(ZM -HT)/(ABS(ZM-HT)+ABS(ZX(I)-HT))
        GO TO 110
C LOGICAL PENETRATIONS ACCUMULATION
107     XR(1)=X(1)
        I=1
        GO TO 113
108     XR(IR)=X(I)
        GO TO 111
110     XR(IR)=XS
111     IF(IR.EQ.2) GO TO 114
113     IR=2
        IB=I+1
        GO TO 112
116     IF(IR.EQ.1) GO TO 117
        XR(2)=1000.0
        GO TO 114
117     XR(1)=1000.0
        XR(2)=1000.0
114     RETURN
        END
        SUBROUTINE INTPLT(X,Y,Z,XG,YG,NX,NY,ZRES)
C LINEAR INTERPOLATION ON CALIBRATION SURFACE TO EVALUATE
C CP VALUE AT CO-ORDINATES XG,YG. THE RESULT RETURNED IS ZRES
        DIMENSION X(40),Y(40),Z(40,40)
C SEARCH FOR I OF LEFT X LINE
        DO 1 IC=1,NX
        IF(X(IC).GE.XG) GO TO 2
1       CONTINUE
2       I=IC-1
C SEARCH FOR J OF LOWER Y LINE
        DO 3 JC=1,NY
        IF(Y(JC).GE.YG) GO TO 4
3       CONTINUE
4       J=JC-1
        AL=(Y(J+1)-Y(J))/(X(I+1)-X(I))
        BL=Y(J)-AL*X(I)
        YCR=(AL*XG)+BL
        IF(YG.EQ.YCR) GO TO 7
        IF(YG.GT.YCR) GO TO 5
C RESULTING Z IS IN LOWER TRIANGLE
        X3=X(I+1)
        Y3=Y(J)
        Z3=Z(I+1,J)
        GO TO 6
C RESULTING Z IS IN UPPER TRIANGLE
5       X3=X(I)
        Y3=Y(J+1)
        Z3=Z(I,J+1)
```

```
C GENERAL CALCULATION:GOOD FOR BOTH TRIANGLES
6       X1=X(I)
        Y1=Y(J)
        Z1=Z(I,J)
        X2=X(I+1)
        Y2=Y(J+1)
        Z2=Z(I+1,J+1)
        AP=((Z1-Z3)*(Y1-Y2)-(Z1-Z2)*(Y1-Y3))/((X1-X3)*(Y1-Y2)-(X1-X2)
       1*(Y1-Y3))
        BP=((Z1-Z2)-AP*(X1-X2))/(Y1-Y2)
        CP=Z1-AP*X1-BP*Y1
        ZRES=AP*XG+BP*YG+CP
        GO TO 100
C RESULTING Z IS ON DIVIDING LINE
7       ZRES=(Z(I+1,J+1)-Z(I,J))*(XG-X(I))/(X(I+1)-X(I))+Z(I,J)
100     RETURN
        END
        SUBROUTINE MAXARY(Z,ZMAX,NX,NY)
C EVALUATION OF MAXIMUM OF OF CALIBRATION SURFACE MATRIX
        DIMENSION Z(40,40)
        DO 1 I=1,NX
        DO 1 J=1,NY
1       IF(Z(I,J).GE.ZMAX) ZMAX=Z(I,J)
        RETURN
        END
        SUBROUTINE MINARY(Z,ZMIN,NX,NY)
C EVALUATION OF MINIMUM OF OF CALIBRATION SURFACE MATRIX
        DIMENSION Z(40,40)
        DO 1 I=1,NX
        DO 1 J=1,NY
1       IF(Z(I,J).LE.ZMIN) ZMIN=Z(I,J)
        RETURN
        END
C END OF WVDR
```

DISTRIBUTION LIST

		No. of Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Office of Research Administration Code 012A Naval Postgraduate School Monterey, California 93940	1
4.	Chairman Code 67 Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
5.	Director, Turbo-Propulsion Laboratory Department of Aeronautics Naval Postgraduate School Monterey, California 93940	30
6.	Dr. H. J. Mueller Research Administrator Code 310A Naval Air Systems Command Navy Department Washington, D. C. 20360	1
7.	Mr. Karl H. Guttmann Code 330C Naval Air Systems Command Navy Department Washington, D. C. 20360	1
8.	Mr. James R. Patton, Jr. Power Program, Code 473 Office of Naval Research Arlington, Virginia 22218	1
9.	Commanding Officer Naval Air Propulsion Test Center Attn: Mr. Vernon Lubosky Trenton, New Jersey 08628	1

10. National Aeronautics and Sapce Administration 1
Lewis Research Center (Library)
2100 Brookpark Road
Cleveland, Ohio 44135

11. CAG Library 1
The Boeing Company
Seattle, Washington 98124

12. Library 1
General Electric Company
Aircraft Engine Technology Division
DTO Mail Drop H43
Cincinnati, Ohio 45215

13. Library 1
Pratt and Whitney Aircraft
Post Office Box 2691
West Palm Beach, Florida 33402

14. Library 1
Pratt and Whitney Aircraft
East Hartford, Connecticut 06108

15. Chief, Fan and Compressor Branch 1
Mail Stop 5-9
NASA Lewis Research Center
21000 Brookpark Road
Cleveland, Ohio 44135

16. Director, Whittle Laboratory 1
Department of Engineering
Cambridge University
ENGLAND

17. Prof. D. Adler 10
Technion Israel Institute of Technology
Department of Mechanical Engineering
Haifa 32000
Israel

18. Prof. A. E. Breugelmans 1
Institut von Karman de la Dynamique des Fluides
72 Chausee de Waterloo
1640 Rhode-St. Genese
Belgium

19. Mr. Robert O. Bullock 1
Air Research Mfg. Corporation
Division of Garrett Corporation
402 South 36th Street
Phoenix, Arizona 85034

20. Dr. F. O. Carta 1
United Technologies Research Labs
400 Main Street
Hartford, Connecticut 06108

21. Prof. Jacques Chauvin 1
Universite D'Aix-Marseille
1 Rue Honnorat
Marseille, France

22. Mr. James V. Davis 1
Teledyne CAE
1330 Laskey Road
Toledo, Ohio 43601

23. Mr. Jean Fabri 1
ONERA
29, Ave. de la Division Leclerc
92 Chatillon
France

24. Prof. Dr. Ing Heinz E. Gallus 1
Lehrstuhl und Institut fur Strahlantiebe und Turbourbeitsmashinen
Rhein.-Westf. Techn. Hochschule Aachen
Templergraben 55
5100 Aachen, Germany

25. Professor J. P. Gostelow 1
School of Mechanical Engineering
The New South Wales Institute of Technology
Australia

26. DR. Ing. Hans-J. Heineman 1
DFVLR-AVA
Bunsenstasse 10
3400 Gottingen, W. Germany

27. Prof. Ch. Hirsch 1
Vrije Universiteit Brussel
Pleinlaan 2
1050 Brussells, Belgium

28. Prof. J. P. Johnston 1
Stanford University
Department of Mechanical Engineering
Stanford, California 94305

29. Prof. Jack L. Kerrebrock, Chairman 1
Aeronautics and Astronautics Department
31-265 Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

30. Dr. B. Lakshminarayana 1
Professor of Aerospace Engineering
The Pennsylvania State University
233 Hammond Building
University Park, Pennsylvania 16802

31. Mr. R. A. Langworthy 1
Army Aviation Material Laboratories
Department of the Army
Fort Eustis, Virginia 23604

32. Dr. A. A. Mikolajczak 1
Pratt and Whitney Aircraft
Engineering 2H
East Hartford, Connecticut 06108

33. Prof. Dr. L. G. Napolitano 1
Director
Institute of Aerodynamics
University of Naples
Viale C. Augusto
80125 Napoli
Italy

34. Prof. Erik Nilsson 1
Institutionen for Stromningsmaskinteknik
Chalmers Tekniska Hogskola
Fack, 402 20 Goteborg 5
Sweden

35. Prof. Gordon C. Oates 1
Department of Aeronautics and Astronautics
University of Washington
Seattle, Washington 98105

36. Prof. Walter F. O'Brian 1
Mechanical Engineering Department
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

37.	Prof. Dr. K. Oswatitsch Technische Hochschule Karlsplatz 13 Vienna, Austria	1
38.	Dr. P. A. Paranjee Head, Propulsion Division National Aeronautical Laboratory Post Bag 1799 Bangalore - 17 India	1
39.	R. E. Peacock School of Mechanical Engineering Cranfield Institute of Technology Cranfield, Bedford MK43 OAL ENGLAND	1
40.	Dr. Bruce A. Reese Director, Jet Propulsion Center School of Mechanical Engineering Purdue University Layfayette, Indiana 47907	1
41.	Dr. W. Schlachter Brown, Boveri-Sulzer Turbomachinery Ltd Dept. TDE Escher Wyss Platz Ch-8023 Zurich Switzerland	1
42.	Dr. George K. Serovy Professor of Mechanical Engineering 208 Mechanical Engineering Building Iowa State University Ames, Iowa 50010	1
43.	Dr. Fernando Sisto Professor and Head of Mechanical Engineering Department Stevens Institute of Technology Castle Point, Hoboken, Jew Jersey 07030	1
44.	Dr. Leroy H. Smith, Jr. Manager, Compressor and Fan Technology Operation General Electric Company Aircraft Engine Technology Division DTO Mail Drop H43 Cincinnati, Ohio 45215	1

45. Dr. W. Tabakoff 1
Professor, Department of Aerospace Engineering
University of Cincinnati
Cincinnati, Ohio 45221

46. Mr. P. Tramm 1
Manager, Research Labs
Detroit Diesel Allison Divison
General Motors
P. O. Box 894
Indianapolis, Indiana 46206

47. Prof. Dr. W. Traupel 1
Institut fur Thermische Turbomaschinen
Eidg. Technische Hochschule

48. Dr. Arthur J. Wennerstrom 1
ARL/LF
Wright-Patterson AFB
Dayton, Ohio 45433

49. Dr. H. Weyer 1
DFVLR
Linder Hohe
505 Porz-Wahn
Germany

50. Mr. P. F. Yaggy 1
Director
U. S. Army Aeronautical Research Laboratory
AMES Research Center
Moffett Field, California 94035