Computer and Hand-calculator Programs to Determine
Extension or Contraction of Faulted Marker Planes

P. T. Delaney 1
M. D. Jackson 2

Open-file Report 85-107

This report is preliminary and has not been reviewed
for conformity with U.S. Geological Survey editorial
standards and stratigraphic nomenclature.

1 U.S. Geological Survey
2 Department of Earth and Planetary Sciences
2255 N. Gemini Dr.
Johns Hopkins University
Flagstaff, Arizona, 86001
Baltimore, Maryland, 21218

1984
ABSTRACT

Programs are presented which calculate angular relations among fault attitude, marker-plane attitude and slip direction, and determine the extension or contraction of the marker by the fault slip. The programs are written both in FORTRAN and in the reverse-Polish language implemented on Hewlett-Packard type hand calculators. Examples of program use are presented.

INTRODUCTION

It has been shown by Jackson and Delaney (1985) that certain angular relations among a marker plane, a fault and the direction of fault slip provide a measure of the extensional or contractional separation of a marker plane. These angles are: \( \phi \), the angle between the downward directed normals to the fault and marker planes; \( \theta \), the angle from the intersection of the fault and marker planes to the slip direction; and \( \omega \), the angle between the direction of maximum contraction and the direction of slip. The first two or the last of these angles can be used to ascertain the extension or contraction of the marker plane in a direction normal to the line of intersection of the fault and marker. This paper presents a program written in FORTRAN for computers, and a program written in a reverse-Polish language for Hewlett-Packard calculators, to calculate these angular relations.

The method of calculating the angle \( \theta \) from the null to the slip directions used by Jackson and Delaney (1985) can result in divide-by-zero conditions. This situation is treated by calculating the magnitude of \( \theta \) from the dot-product relation \( \cos |\theta| = |\mathbf{N} \cdot \mathbf{S}| \), where \( \mathbf{N} \) and \( \mathbf{S} \) are the null and slip vectors, respectively, and the vertical bars denote absolute values of the
enclosed quantity. The sign of the angle $\theta$ is positive if, looking at the hanging-wall surface of the fault, the angle from the intersection with the marker to the slip direction is drawn in a clockwise direction. This is equivalent to finding whether the direction given by the cross product relation $N \times S$ is in the same direction (positive $\theta$) or opposite direction (negative $\theta$) as the downward directed normal to the fault plane, $F$.

**THE FORTRAN PROGRAM**

The FORTRAN program employs the "structured branching options" (see Meissener and Organick, 1980, p. 480) available in ANSI FORTRAN 77 implementations. The program is thoroughly documented and follows the same notation as used by Jackson and Delaney (1985). The program (called FAULT) calls six subroutines: DDDTDC, which converts dip-direction and dip data to direction-cosine data; TPTDC, which converts trend and plunge data to direction-cosine data; DCTTP, which converts direction-cosine data to trend and plunge data; CROSS, which computes the vector cross product; NORM, which normalizes a vector to unit length; and DOT, which computes the vector dot product. The program prompts the user for input at the keyboard, and displays output on a CRT or line printer. It is therefore assumed that the standard input and output devices are interactive, and denoted by the logical unit numbers 5 and 6, respectively. Listings are given in Appendix A.

It is assumed that no field measurement is more accurate than $1^\circ$; if the angle $\phi$ is within $1^\circ$ of $90^\circ$ or the angle $\theta$ is within $1^\circ$ of $0^\circ$, then the fault slip is identified as "null"—it neither extended nor contracted the marker. In addition, the program calculates the angle between the slip direction and the fault plane. By definition, this angle should be zero. If it is found to
be greater than 1°, it is used to determine the limits of null faulting.

The example below illustrates the use of the FORTRAN program. User-supplied entries are underlined.

RUN FAULT

    MARKER: Dip Direction and dip (deg) = ?? = 317 20
    FAULT: Dip Direction and Dip (deg) = ?? = 180 45
    SLICKENLINE: Trend and Plunge (deg) = ?? = 92 2

    OFFSET: Hanging Wall UP (U), DOWN (D),
            or NO (N) apparent offset = ?? = D

92. 2. = Trend, Plunge of Hanging Wall SLIP DIRECTION
0. = Eta, ANGLE of the SLIP VECTOR from the FAULT PLANE
61. = Phi, ANGLE between MARKER and FAULT PLANES
101. 11. = Trend, Plunge of NULL DIRECTION
-18. = Theta, ANGLE from NULL to SLIP DIRECTION
99. = Omega, ANGLE from PURE-CONTRACTION DIRECTION to SLIP DIRECTION

EXTENSION FAULT

Another Fault ?? (Y/N)
Y
    MARKER PLANE: Dip Direction and Dip = 317. 20.
    FAULT: Dip Direction and Dip (deg) = ?? = 180 45
    SLICKENLINE: Trend and Plunge (deg) = ?? = 92 2

    OFFSET: Hanging Wall UP (U), DOWN (D),
            or NO (N) apparent offset = ?? = U

272. -2. = Trend, Plunge of Hanging Wall SLIP DIRECTION
0. = Eta, ANGLE of the SLIP VECTOR from the FAULT PLANE
61. = Phi, ANGLE between MARKER and FAULT PLANES
101. 11. = Trend, Plunge of NULL DIRECTION
18. = Theta, ANGLE from NULL to SLIP DIRECTION
81. = Omega, ANGLE from PURE-CONTRACTION DIRECTION to SLIP DIRECTION
CONTRACTION FAULT

Another Fault ?? (Y/N)
N
Another Marker?? (Y/N)
Y

MARKER: Dip Direction and dip (deg) = ?? =
135 70

FAULT: Dip Direction and Dip (deg) = ?? =
180 45

SLICKENLINE: Trend and Plunge (deg) = ?? =
92 2

OFFSET: Hanging Wall UP (U), DOWN (D),
or NO (N) apparent offset = ?? =
D

92. 2. = Trend, Plunge of Hanging Wall SLIP DIRECTION
0. = Eta, ANGLE of the SLIP VECTOR from the FAULT PLANE
45. = Phi, ANGLE between MARKER and FAULT PLANES

26. -42. = Trend, Plunge of NULL DIRECTION
206. 42. = in Lower Hemisphere

74. = Theta, ANGLE from NULL to SLIP DIRECTION
47. = Omega, ANGLE from PURE-CONTRACTION DIRECTION to SLIP DIRECTION

CONTRACTION FAULT

Another Fault ?? (Y/N)
N
Another Marker?? (Y/N)
Y

MARKER: Dip Direction and dip (deg) = ?? =
45 45

FAULT: Dip Direction and Dip (deg) = ?? =
225 45

SLICKENLINE: Trend and Plunge (deg) = ?? =
225 45

OFFSET: Hanging Wall UP (U), DOWN (D),
or NO (N) apparent offset = ?? =
D

225. 45. = Trend, Plunge of Hanging Wall SLIP DIRECTION
0. = Eta, ANGLE of the SLIP VECTOR from the FAULT PLANE
90. = Phi, ANGLE between MARKER and FAULT PLANES

-45. 0. = Trend, Plunge of NULL DIRECTION
135. 0. = in Lower Hemisphere
HAND-CALCULATOR PROGRAM

The calculation, written for a Hewlett-Packard 15-C calculator, requires 20 storage registers and about 200 lines of instructions. Parameters stored in the registers are shown in Table 1. A listing is given in Appendix B. For brevity, only the angle $\phi$ between the downward-directed marker- and fault-plane normals, and the angle $\theta$ from the null direction to the slip direction, are calculated. Extension or contraction can be determined from these angles (Table 2). There are five programs and three subroutines. The first program uses the dip direction and dip of the marker plane to calculate and store the direction cosines of the downward-directed normals to that plane. The second program performs the same task for the fault plane. For faults with hanging-wall-down displacements, the third program uses the trend and plunge of the

<table>
<thead>
<tr>
<th>REGISTER #</th>
<th>CONTENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>lmarker</td>
</tr>
<tr>
<td>1</td>
<td>m_marker</td>
</tr>
<tr>
<td>2</td>
<td>n_marker</td>
</tr>
<tr>
<td>3</td>
<td>l_fault</td>
</tr>
<tr>
<td>4</td>
<td>m_fault</td>
</tr>
<tr>
<td>5</td>
<td>n_fault</td>
</tr>
<tr>
<td>6</td>
<td>l_slip</td>
</tr>
<tr>
<td>7</td>
<td>m_slip</td>
</tr>
<tr>
<td>8</td>
<td>n_slip</td>
</tr>
<tr>
<td>9</td>
<td>$\phi$, angle between downward-directed marker- and fault-normals</td>
</tr>
<tr>
<td>.0</td>
<td>$\theta$, angle from null to slip direction</td>
</tr>
<tr>
<td>.1 - .9</td>
<td>miscellaneous</td>
</tr>
</tbody>
</table>

TABLE 1: Contents of storage registers

l, m, n are cosines of angles that superscripted vector makes with the South, East and Up directions.
Table 2: Extension or Contraction from Angles $\phi$ and $\theta$

<table>
<thead>
<tr>
<th>$0^\circ &lt; \phi &lt; 90^\circ$</th>
<th>$\phi = 0^\circ$</th>
<th>$90^\circ &lt; \phi &lt; 180^\circ$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$-90^\circ &lt; \theta &lt; 0^\circ$</td>
<td>Extend</td>
<td>Null</td>
</tr>
<tr>
<td>$0^\circ = \theta$</td>
<td>Null</td>
<td>Null</td>
</tr>
<tr>
<td>$0^\circ &lt; \theta &lt; 90^\circ$</td>
<td>Contract</td>
<td>Null</td>
</tr>
</tbody>
</table>

slip direction to calculate and store the equivalent direction cosines. For faults with hanging-wall-up displacements, the fourth program performs the same calculation as the third. The fifth program calculates, displays and stores the angles $\phi$ and $\theta$.

In the following, the instructions, input keystrokes and the output are shown for the first and second examples given above. Underlined numbers refer to the values of the parameters for the particular example.

**STEP INSTRUCTIONS**

1. Input marker-plane data
   a. Marker dip direction *(clockwise degrees from North)*
      
      | Underlined number |
      |-------------------|
      | 317               |
      | ENTER             |
   
   b. Marker dip *(degrees down from horizontal)*
      
      | Underlined number |
      |-------------------|
      | 20                |
      | f A               |
   
   c. Run program A

2. Input fault-plane data
   a. Fault dip direction *(clockwise degrees from North)*
      
      | Underlined number |
      |-------------------|
      | 180               |
      | ENTER             |
   
   d. Fault dip *(degrees down from horizontal)*
      
      | Underlined number |
      |-------------------|
      | 45                |
      | f B               |
   
   c. Run program B

3. Input slip-direction data
   a. Slickenline trend *(clockwise degrees from North)*
      
      | Underlined number |
      |-------------------|
      | 92                |
      | ENTER             |
   
   b. Slickenline plunge *(degrees down from horizontal)*
      
      | Underlined number |
      |-------------------|
      | 2                 |
   
   c. Sense of offset
      Run program C for hanging wall down displacements
      
      | Underlined number |
      |-------------------|
      | f C               |
STEP INSTRUCTIONS

4. Calculate $\phi$ and $\theta$
   Run program E

   \[ f \ E \]

   \[
   61 \ (= \ \phi) \\
   -18 \ (= \ \theta)
   \]

   For the same marker-plane and fault-plane attitudes, it is not necessary to re-enter these orientations to enter a different slickenline. Similarly, it is not necessary to re-enter the attitude of the marker for new fault and slickenline attitudes. For hanging-wall-up fault slips program D is run instead of program C:

5. Input slip-direction data
   a. Slickenline trend
      (clockwise degrees from North) \[ 92 \]
      ENTER
   b. Slickenline plunge
      (degrees down from horizontal) \[ 2 \]
   c. Sense of offset
      Run program D for hanging wall up displacements

6. Calculate $\phi$ and $\theta$
   Run program E

   \[ f \ E \]

   \[
   61 \ (= \ \phi) \\
   18 \ (= \ \theta)
   \]

REFERENCES

Jackson, M.D., and P.T. Delaney, 1985, Extension and contraction of faulted marker planes: Geology, to be submitted. 16 ms. pages, 5 figures, 2 tables.

APPENDIX A: FORTRAN PROGRAM

C 345678-1-2345678-2-2345678-3-2345678-4-2345678-5-2345678-6-2345678-7-2
C
PROGRAM FAULT
C
C Input: (1) Marker-plane Dip Direction and Dip
C (2) Fault-plane Dip Direction and Dip
C (3) Slickenline Trend and Plunge
C (4) Sense of Offset
C
C Output: (1) Slip Direction of Hanging Wall relative to Footwall
C (2) Angle between Slip Direction and Fault Plane (ideally, this
C should be equal to zero
C (3) Angle from Marker Plane to Fault Plane
C (4) Null Direction
C (4) Angle from Null to Slip Direction
C (5) Angle from direction of maximum contraction to Slip Direction
C (6) Determination as to whether the Fault Extends, Contracts,
C or has a Null effect on the Marker
C
IMPLICIT REAL*4 (A-H,0-Z), INTEGER*2 (I-N)
EXTERNAL DOT, CROSS, NORM, DDDTDC, TPTDC, DCTTP
REAL*4 MD, MDC, ND, NDC, MNDC
DIMENSION MD(2), FD(2), SD(2), ND(2), MDC(3), FDC(3), SDC(3),
       ND(3), FPDC(3), MNDC(3), XDC(3)
CHARACTER*1 ANS
COMMON /PI0180/ PI180
PI180 = 3.141592654EO/180.0EO
C
Vectors of Length 2: (1) Dip Direction or Trend in Clockwise
C Degrees from North
C (last letter of (2) Dip or Plunge in Degrees Down from
C name is ‘D’) Horizontal
C
Vectors of Length 3: (1) South-pointed Direction Cosine
C (2) East-pointed Direction Cosine
C (last two letters (3) Up-pointed Direction Cosine
C of name are ‘DC’)
C
OUTER LOOP POINT: Marker-plane Orientation
C
10 WRITE (6,12)
12 FORMAT (/' MARKER: Dip Direction and dip (deg) = ?? = ’)
READ (5,*9) MD(1), MD(2)
IF (MD(2) .LT. 1.0E-1) MD(1) = 0.0EO
GOTO 14
20 WRITE (6,22) MD(1), MD(2)
22 FORMAT (/' MARKER PLANE: Dip Direction and Dip = ’,2(F5.0))
C
INNER LOOP POINT: Fault Data
C
14 WRITE (6,24)
FAULT: Dip Direction and Dip (deg) = ?? = ‘
READ (5,* ) FD(1), FD(2)
IF (FD(2) .LT. 1.0E-1) FD(1) = 0.0E0
WRITE (6,26)

SLICKENLINE: Trend and Plunge (deg) = ?? = ‘
READ (5,* ) SD(1), SD(2)
WRITE (6,28)

OFFSET: Hanging Wall UP (U), DOWN (D) = ?? = ‘
READ (5,30) ANS

Combine Slickenline and Sense-of-offset to get Slip Direction

IF ((ANS .EQ. ‘U’) .OR. (ANS .EQ. ‘u’)) THEN
  SD(1) = 180.0E0 + SD(1)
  IF (SD(1) .GE. 360.0E0) SD(1) = SD(1) - 360.0E0
  SD(2) = -SD(2)
ENDIF
WRITE (6,32) SD(1), SD(2)

Dip Direction and Dip, and Trend and Plunge, To Direction Cosines

CALL DDDTDCC(MD,MDC)
CALL DDDTDCC(FD,FDC)
CALL TPTDC(SD,SDC)

Calculate and write the Angle between Slip Vector and Fault Plane.
This angle indicates the accuracy of the field measurements. It is
assumed that no measurement is better than 1 degree, or this angle,
whichever is greater. Null Faulting arises if the appropriate
angles are within 1 degree or Phi degrees, whichever is greater, of
the true Null directions.

CALL DOT(SDC,FDC,ETA)
ETA = 90.0E0 - ACOS(ABS(ETA))/PI180
IF (ETA .GT. 1.0E0) THEN
  DLIMIT = ETA
ELSE
  DLIMIT = 1.0E0
ENDIF
WRITE (6,34) ETA

Calculate and write Angle from Marker Plane to Fault Plane

CALL DOT(MDC,FDC,PHI)
PHI = ACOS(PHI)/PI180
WRITE (6,36) PHI
CALL CROSS(FDC,MDC,NDC)
CALL NORM(NDC)
CALL DCTTP(NDC,ND)
WRITE (6,38) ND(1), ND(2)
38  FORMAT (/2(F5.0),' = Trend, Plunge of NULL DIRECTION')
    IF (ND(2) .LT. 0.0E0) THEN
      ND(1) = ND(1) + 180.0E0
      ND(2) = -ND(2)
      IF (ND(1) .GE. 360.0E0) ND(1) = ND(1) - 360.0E0
      WRITE (6,40) ND(1), ND(2)
40  FORMAT (2(F5.0),' = in Lower Hemisphere')
    ENDIF
C
C Calculate and write Theta Angle
C The method used below is more cumbersome, but more robust, than that
C described in the paper--there s no possibility for divide-by-zero
C errors. The two methods are entirely equivalent.
C
CALL DOT(NDC,SDC,THETA)
THETA = ACOS(ABS(THETA))/PI180
IF (THETA .GT. DLIMIT) THEN
  CALL CROSS(NDC,SDC,FPDC)
  CALL NORM(FPDC)
  CALL DOT(FDC,FPDC,TEST)
  IF (TEST .LT. 0.0E0) THETA = -THETA
ENDIF
WRITE (6,50) THETA
50  FORMAT (/5X,F5.0,' = Theta, ANGLE from NULL to SLIP DIRECTION')
C
C Calculate and write Omega angle
C
CALL CROSS(MDC,NDC,XDC)
CALL NORM(XDC)
CALL DOT(SDC,XDC,OMEGA)
OMEGA = ACOS(OMEGA)/PI180
WRITE (6,60) OMEGA
60  FORMAT (/5X,F5.0,' = Omega, ANGLE from PURE-CONTRACTION ',
     'DIRECTION to SLIP DIRECTION')
C
C Calculate and write Type of Fault
C
  IF Phi < 90 deg, 0 deg < Theta < 90 deg -- Extension Fault
  0 deg > Theta > -90 deg -- Contraction Fault
  IF Phi > 90 deg, 0 deg > Theta > 90 deg -- Extension Fault
  0 deg < Theta < -90 deg -- Contraction Fault
  IF Phi = 90 deg or Theta = 0 deg -- Null Fault
C
IF (PHI .LT. 90.0E0-DLIMIT) THEN
  IF (THETA .LT. -DLIMIT) THEN
    WRITE (6,70)
ELSEIF (THETA .GT. DLIMIT) THEN
    WRITE (6,72)
ELSE
    WRITE (6,74)
ENDIF
ELSEIF (PHI .GT. 90.0+DLIMIT) THEN
    IF (THETA .LT. -DLIMIT) THEN
        WRITE (6,72)
    ELSEIF (THETA .GT. DLIMIT) THEN
        WRITE (6,70)
    ELSE
        WRITE (6,74)
    ENDIF
ELSE
    WRITE (6,76)
ENDIF
70 FORMAT (' EXTENSION FAULT'/)
72 FORMAT (' CONTRACTION FAULT'/)
74 FORMAT (' Slip is in Null Direction: NULL FAULT'/)
76 FORMAT (' Fault is normal to Marker: NULL FAULT'/)
C WRITE (6,90)
90 FORMAT (' Another Fault ?? (Y/N) ')
READ (5,30) ANS
IF ((ANS .EQ. 'Y') .OR. (ANS .EQ. 'y')) GOTO 20
C WRITE (6,92)
92 FORMAT (' Another Marker?? (Y/N) ')
READ (5,30) ANS
IF ((ANS .EQ. 'Y') .OR. (ANS .EQ. 'y')) GOTO 10
C STOP
END

SUBROUTINE DDDTDC(D,DC)
C Napier’s rules for spherical triangles to convert from Dip Direction
C and Dip To Direction Cosines for a downward pointed plane normal
C IMPLICIT REAL*4 (A-H,O-Z), INTEGER*2 (I-N)
DIMENSION D(2), DC(3)
COMMON /PI0180/ PI180
DC(1) = SIN(PI180*D(2))*COS(PI180*D(1))
DC(2) = -SIN(PI180*D(2))*SIN(PI180*D(1))
DC(3) = -COS(PI180*D(2))
RETURN
END
SUBROUTINE TPTDC(D, DC)

C Napier’s rules for spherical triangles to convert from Trend and Plunge To Direction Cosines

IMPLICIT REAL*4 (A-H, O-Z), INTEGER*2 (I-N)
DIMENSION D(2), DC(3)
COMMON /PI0180/ PI180

DC(1) = -COS(PI180*D(1))*COS(PI180*D(2))
DC(2) = COS(PI180*D(2))*SIN(PI180*D(1))
DC(3) = -SIN(PI180*D(2))

RETURN
END

SUBROUTINE DCTTP(DC, D)

C Napier’s rules for spherical triangles to convert from Direction Cosines for a downward pointed plane normal To Trend and Plunge

IMPLICIT REAL*4 (A-H, O-Z), INTEGER*2 (I-N)
DIMENSION D(2), DC(3)
COMMON /PI0180/ PI180

D(2) = ASIN(-DC(3))/PI180
A = COS(PI180*D(2))
B = ASIN(DC(2)/A)/PI180
IF (B .GT. 0.0E0) THEN
   D(1) = ACOS(-DC(1)/A)/PI180
ELSE
   D(1) = -ACOS(-DC(1)/A)/PI180
ENDIF

RETURN
END
SUBROUTINE CROSS(A,B,Z)

vector CROSS product

IMPLICIT REAL*4 (A-H,O-Z), INTEGER*2 (I-N)
DIMENSION A(3), B(3), Z(3)

Z(1) = A(2)*B(3) - B(2)*A(3)
Z(2) = -A(1)*B(3) + B(1)*A(3)
Z(3) = A(1)*B(2) - B(1)*A(2)

RETURN
END

SUBROUTINE NORM(Z)

NORMALize a vector to unit length

IMPLICIT REAL*4 (A-H,O-Z), INTEGER*2 (I-N)
EXTERNAL DOT
DIMENSION Z(3)

CALL DOT(Z,Z,ZMAG)
ZMAG = SQRT(ZMAG)
Z(1) = Z(1)/ZMAG
Z(2) = Z(2)/ZMAG
Z(3) = Z(3)/ZMAG

RETURN
END

SUBROUTINE DOT(A,B,Z)

vector DOT product

IMPLICIT REAL*4 (A-H,O-Z), INTEGER*2 (I-N)
DIMENSION A(3), B(3)

Z = A(1)*B(1) + A(2)*B(2) + A(3)*B(3)

RETURN
END
## APPENDIX B: CALCULATOR PROGRAM

<table>
<thead>
<tr>
<th>STEP</th>
<th>KEY ENTRY</th>
<th>KEY CODE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>f 1bl a</td>
<td>42 21</td>
<td>Program A: direction cosines, l, m and n, of downward pointed marker-plane normal, M, calculated from dip direction and dip.</td>
</tr>
<tr>
<td>2</td>
<td>gsb 1</td>
<td>32 1</td>
<td>Go to subroutine 1</td>
</tr>
<tr>
<td>3</td>
<td>sto 2</td>
<td>44 2</td>
<td>Store n(^M) in register 2</td>
</tr>
<tr>
<td>4</td>
<td>r+</td>
<td>33</td>
<td>Roll n(^M) off stack</td>
</tr>
<tr>
<td>5</td>
<td>sto 1</td>
<td>44 1</td>
<td>Store m(^M) in register 1</td>
</tr>
<tr>
<td>6</td>
<td>r+</td>
<td>33</td>
<td>Roll m(^M) off stack</td>
</tr>
<tr>
<td>7</td>
<td>sto 0</td>
<td>44 0</td>
<td>Store l(^M) in register 0</td>
</tr>
<tr>
<td>8</td>
<td>g rtn</td>
<td>43 32</td>
<td>Return — end of program</td>
</tr>
</tbody>
</table>

| 9    | f 1bl b   | 42 21    | Program B: direction cosines, l, m and n, of downward pointed fault-plane normal, F, calculated from dip direction and dip. |
| 10   | gsb 1     | 32 1     | Go to subroutine 1 |
| 11   | sto 5     | 44 5     | Store n\(^F\) in register 5 |
| 12   | r+        | 33       | Roll n\(^F\) off stack |
| 13   | sto 4     | 44 4     | Store m\(^F\) in register 4 |
| 14   | r+        | 33       | Roll m\(^F\) off stack |
| 15   | sto 3     | 44 3     | Store l\(^F\) in register 3 |
| 16   | g rtn     | 43 32    | Return — end of program |

| 17   | f 1bl d   | 42 21    | Program D: for hanging-wall-up faults, reverse the slip direction entered as trend, T, and plunge, P |
| 18   | chs       | 16       | Change sign: P = -P |
| 19   | x-y       | 34       | Reverse contents of x and y stack registers |
| 20   | l         | 1        | 1 |
| 21   | 8         | 8        | 8 |
| 22   | 0         | 0        | 0 |
| 23   | +         | 40       | T = 180° + T |
| 24   | x-y       | 34       | Reverse contents of x and y stack registers |
STEP  KEY ENTRY  KEY CODE  COMMENTS

25  f  fbl  c  42  21  13  Program C: direction cosines, l, m and n, of slip
direction, S, calculated from trend and plunge

26  gsb  2   32  2   Go to subroutine 2
27  sto  8   44  8   Store n^S in register 8
28  r+    33   Roll n^S off stack
29  sto  7   44  7   Store m^S in register 7
30  r+    33   Roll m^S off stack
31  sto  6   44  6   Store l^S in register 6
32  g  rtn  43  32  Return -- end of program

---------------

33  f  lbl  e  42  21  15  Program E: calculate \( \phi \) and \( \theta \)

Step 1: calculate \( \phi = \cos^{-1}(M \cdot F) \)

34  rcl  0   45  0   Recall 1^M to stack
35  rcl  3   45  3   Recall 1^F to stack
36  x     20   1^M \times 1^F
37  rcl  1   45  1   Recall m^M to stack
38  rcl  4   45  4   Recall m^F to stack
39  x     20   m^M \times m^F
40  +    40   (1^M \times 1^F) + (m^M \times m^F)
41  rcl  2   45  2   Recall n^M to stack
42  rcl  5   45  5   Recall n^F to stack
43  x     20   n^M \times n^F
44  +    40   \cos \phi = (1^M \times 1^F) + (m^M \times m^F) + (n^M \times n^F) = M \cdot F
45  g  cos^{-1}  43  24  \phi = \cos^{-1}(M \cdot F)
46  sto  9   44  9   Store \phi in register 9
47  f  pse  42  31  pause to display \( \phi \)
48  f  pse  42  31  pause to display \( \phi \)

Step 2: calculate \( N = F \times M \)

49  rcl  2   45  2   Recall n^M to stack ...
50  sto  .9   44  .9   ... and store in register .9
51  rcl  1   45  1   Recall 1^M to stack ...
52  sto  .8   44  .8   ... and store in register .8
53  rcl  0   45  0   Recall 1^M to stack ...
54  sto  .7   44  .7   ... and store in register .7
STEP KEY ENTRY KEY CODE COMMENTS

55 rcl 5 45 5 Recall n_F to stack
56 rcl 4 45 4 Recall m_F to stack
57 rcl 3 45 3 Recall 1_F to stack
58 gsb 3 32 3 Go to subroutine 3
59 sto .3 44 .3 Store n_N in register .3
60 r+ 33 Roll n_N off stack
61 sto .2 44 .2 Store m_N in register .2
62 r+ 33 Store Roll m_N off stack
63 sto .1 44 .1 Store i_N in register .1

Step 3: calculate \( |\theta| = \cos^{-1}(|N \cdot S|) \)

64 rcl 6 45 6 Recall 1_S to stack
65 x 20 \( 1^N \times 1^S \)
66 rcl .2 45 .2 Recall m_N to stack
67 rcl 7 45 7 Recall m_S to stack
68 x 20 \( m^N \times m^S \)
69 + 40 \((1^N \times 1^S) + (m^N \times m^S)\)
70 rcl .3 45 .3 Recall n_N to stack
71 rcl 8 45 8 Recall n_S to stack
72 x 20 \( n^N \times n^S \)
73 + 40 \( \cos \theta = (1^N \times 1^S) + (m^N \times m^S) + (n^N \times n^S) = M \cdot F \)
74 g test 2 43 30 2 If \( \cos \theta < 0 \)
75 chs 16 Change sign: \( \cos \theta = -\cos \theta \)
76 g cos^1 43 24 \( |\theta| = \cos^{-1}(|N \cdot S|) \)
77 sto .0 44 .0 Store \( |\theta| \) in register .0

Step 4: If \( |\theta| < 1^o \), then null fault

78 1 1 1
79 - 30 \(|\theta| - 1\)
80 g test 1 43 30 1 If \( |\theta| - 1 > 0^o \)
81 gto 4 22 4 Go to label 4
82 rcl .0 45 .0 Recall \( |\theta| \) to the stack
83 g rtn 43 32 Return — end of program

Step 5: find sign of \( \theta \) using \( F'(N \cdot S) \)

84 f lbl 4 42 21 4 Label 4
85 rcl 8 45 8 Recall n_S to stack ...
STEP | KEY ENTRY | KEY CODE | COMMENTS
--- | --- | --- | ---
86 | sto .9 | 44 .9 | ... and store in register .9
87 | rcl 7 | 45 7 | Recall mS to stack ...
88 | sto .8 | 44 .8 | ... and store in register .8
89 | rcl 6 | 45 6 | Recall 1S to stack ...
90 | sto .7 | 44 .7 | ... and store in register .7
91 | rcl .3 | 45 .3 | Recall nN to stack
92 | rcl .2 | 45 .2 | Recall mN to stack
93 | rcl .1 | 45 .1 | Recall 1N to stack
94 | gsb 3 | 32 3 | Go to subroutine 3
95 | rcl 5 | 45 5 | Recall nF to stack
96 | x | 20 | nNxS x nF
97 | x~y | 34 | Reverse contents of x and y stack registers
98 | rcl 4 | 45 5 | Recall mF to stack
99 | x | 20 | mNxS x mF
100 | + | 40 | (nNxS x nF) + (mNxS x mF)
101 | x~y | 34 | Reverse contents of x and y stack registers
102 | rcl 3 | 45 3 | Recall 1F to stack
103 | x | 20 | 1NxS x 1F
104 | + | 40 | F*(NxS) = (nNxS x nF) + (mNxS x mF) + (1NxS x 1F)
105 | g test 1 | 43 30 1 | If F*(NxS) > 0
106 | gto 5 | 22 5 | Go to label 5
107 | rcl .0 | 45 .0 | Recall θ to stack
108 | chs | 16 | Change sign: θ = -θ
109 | sto .0 | 44 .0 | Store θ in register .0
110 | g rtn | 43 32 | Return -- end of program
111 | f 1bl 5 | 42 21 5 | Label 5
112 | rcl .0 | 45 .0 | Recall θ to stack
113 | g rtn | 43 32 | Return -- end of program

Subroutine 1: calculate direction cosines, 1, m, and n, from dip direction, DD, and dip, D

114 | f 1bl 1 | 42 21 1 | Subroutine 1: calculate direction cosines, 1, m, and n, from dip direction, DD, and dip, D
115 | sto .9 | 44 .9 | Store D in register .9
116 | sin | 23 | sin D
117 | x~y | 34 | Reverse contents of x and y stack registers
<table>
<thead>
<tr>
<th>STEP</th>
<th>KEY ENTRY</th>
<th>KEY CODE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>118</td>
<td>sto .8</td>
<td>44 .8</td>
<td>Store DD in register .8</td>
</tr>
<tr>
<td>119</td>
<td>cos</td>
<td>24</td>
<td>cos DD</td>
</tr>
<tr>
<td>120</td>
<td>x</td>
<td>20</td>
<td>1 = cos DD x sin D</td>
</tr>
<tr>
<td>121</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall D to stack</td>
</tr>
<tr>
<td>122</td>
<td>sin</td>
<td>23</td>
<td>sin D</td>
</tr>
<tr>
<td>123</td>
<td>rcl .8</td>
<td>45 .8</td>
<td>Recall DD to stack</td>
</tr>
<tr>
<td>124</td>
<td>sin</td>
<td>23</td>
<td>sin DD</td>
</tr>
<tr>
<td>125</td>
<td>x</td>
<td>20</td>
<td>sin D x sin DD</td>
</tr>
<tr>
<td>126</td>
<td>chs</td>
<td>16</td>
<td>Change sign: m = -sin D x sin DD</td>
</tr>
<tr>
<td>127</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall D to stack</td>
</tr>
<tr>
<td>128</td>
<td>cos</td>
<td>24</td>
<td>cos D</td>
</tr>
<tr>
<td>129</td>
<td>chs</td>
<td>16</td>
<td>Change sign: n = cos D</td>
</tr>
<tr>
<td>130</td>
<td>g rtn</td>
<td>43 32</td>
<td>Return -- end of subroutine</td>
</tr>
</tbody>
</table>

(Subroutine 2: Calculate direction cosines, 1, m and n, from trend, T, and plunge, P)

<table>
<thead>
<tr>
<th>STEP</th>
<th>KEY ENTRY</th>
<th>KEY CODE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>132</td>
<td>sto .9</td>
<td>44 .9</td>
<td>Store P in register .9</td>
</tr>
<tr>
<td>133</td>
<td>cos</td>
<td>24</td>
<td>cos P</td>
</tr>
<tr>
<td>134</td>
<td>x-y</td>
<td>34</td>
<td>Reverse contents of x and y stack registers</td>
</tr>
<tr>
<td>135</td>
<td>sto .8</td>
<td>44 .8</td>
<td>Store T in register .8</td>
</tr>
<tr>
<td>136</td>
<td>cos</td>
<td>24</td>
<td>cos T</td>
</tr>
<tr>
<td>137</td>
<td>x</td>
<td>20</td>
<td>cos T x cos P</td>
</tr>
<tr>
<td>138</td>
<td>chs</td>
<td>16</td>
<td>Change sign: 1 = -cos T x cos P</td>
</tr>
<tr>
<td>139</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall P to stack</td>
</tr>
<tr>
<td>140</td>
<td>cos</td>
<td>24</td>
<td>cos P</td>
</tr>
<tr>
<td>141</td>
<td>rcl .8</td>
<td>45 .8</td>
<td>Recall T to stack</td>
</tr>
<tr>
<td>142</td>
<td>sin</td>
<td>23</td>
<td>sin T</td>
</tr>
<tr>
<td>143</td>
<td>x</td>
<td>20</td>
<td>m = sin T x cos P</td>
</tr>
<tr>
<td>144</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall P to stack</td>
</tr>
<tr>
<td>145</td>
<td>sin</td>
<td>23</td>
<td>sin P</td>
</tr>
<tr>
<td>146</td>
<td>chs</td>
<td>16</td>
<td>Change sign: n = -sin P</td>
</tr>
<tr>
<td>147</td>
<td>g rtn</td>
<td>43 32</td>
<td>Return -- end of subroutine</td>
</tr>
<tr>
<td>STEP</td>
<td>KEY ENTRY</td>
<td>KEY CODE</td>
<td>COMMENTS</td>
</tr>
<tr>
<td>------</td>
<td>-----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>148</td>
<td>f lbl 3</td>
<td>42 21 3</td>
<td>Subroutine 3: calculate and normalize to unit length the cross product A x B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Step 1:</strong> C = A X B</td>
</tr>
<tr>
<td>149</td>
<td>sto .4</td>
<td>44 .4</td>
<td>Store 1^A in register .4</td>
</tr>
<tr>
<td>150</td>
<td>r+</td>
<td>33</td>
<td>Roll 1^A off stack</td>
</tr>
<tr>
<td>151</td>
<td>sto .5</td>
<td>44 .5</td>
<td>Store m^A in register .5</td>
</tr>
<tr>
<td>152</td>
<td>r+</td>
<td>33</td>
<td>Roll m^A off stack</td>
</tr>
<tr>
<td>153</td>
<td>sto .6</td>
<td>44 .6</td>
<td>Store n^A in register .6</td>
</tr>
<tr>
<td>154</td>
<td>rcl .5</td>
<td>45 .5</td>
<td>Recall m^A to stack</td>
</tr>
<tr>
<td>155</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall n^B to stack</td>
</tr>
<tr>
<td>156</td>
<td>x</td>
<td>20</td>
<td>m^A x n^B</td>
</tr>
<tr>
<td>157</td>
<td>rcl .8</td>
<td>45 .8</td>
<td>Recall m^B to stack</td>
</tr>
<tr>
<td>158</td>
<td>rcl .6</td>
<td>45 .6</td>
<td>Recall n^A to stack</td>
</tr>
<tr>
<td>159</td>
<td>x</td>
<td>20</td>
<td>n^A x m^B</td>
</tr>
<tr>
<td>160</td>
<td>-</td>
<td>30</td>
<td>1(C = (n^A x m^B) - (m^A x n^B))</td>
</tr>
<tr>
<td>161</td>
<td>sto .1</td>
<td>44 .1</td>
<td>Store 1^C in register .1</td>
</tr>
<tr>
<td>162</td>
<td>rcl .7</td>
<td>45 .7</td>
<td>Recall 1^B to stack</td>
</tr>
<tr>
<td>163</td>
<td>rcl .6</td>
<td>45 .6</td>
<td>Recall n^A to stack</td>
</tr>
<tr>
<td>164</td>
<td>x</td>
<td>20</td>
<td>n^A x 1^B</td>
</tr>
<tr>
<td>165</td>
<td>rcl .4</td>
<td>45 .4</td>
<td>Recall 1^A to stack</td>
</tr>
<tr>
<td>166</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall n^B to stack</td>
</tr>
<tr>
<td>167</td>
<td>x</td>
<td>20</td>
<td>n^B x 1^A</td>
</tr>
<tr>
<td>168</td>
<td>-</td>
<td>30</td>
<td>m^C = (n^B x 1^A) - (n^A x 1^B)</td>
</tr>
<tr>
<td>169</td>
<td>sto .2</td>
<td>44 .2</td>
<td>Store m^C in register .2</td>
</tr>
<tr>
<td>170</td>
<td>rcl .4</td>
<td>45 .4</td>
<td>Recall 1^A to stack</td>
</tr>
<tr>
<td>171</td>
<td>rcl .8</td>
<td>45 .8</td>
<td>Recall m^B to stack</td>
</tr>
<tr>
<td>172</td>
<td>x</td>
<td>20</td>
<td>m^B x 1^A</td>
</tr>
<tr>
<td>173</td>
<td>rcl .7</td>
<td>45 .7</td>
<td>Recall 1^B to stack</td>
</tr>
<tr>
<td>174</td>
<td>rcl .5</td>
<td>45 .5</td>
<td>Recall m^A to stack</td>
</tr>
<tr>
<td>175</td>
<td>x</td>
<td>20</td>
<td>m^A x 1^B</td>
</tr>
<tr>
<td>176</td>
<td>-</td>
<td>30</td>
<td>n^C = (m^A x 1^B) - (m^B - 1^A)</td>
</tr>
<tr>
<td>177</td>
<td>sto .3</td>
<td>44 .3</td>
<td>Store n^C in register .3</td>
</tr>
<tr>
<td>178</td>
<td>g x^2</td>
<td>43 11</td>
<td>Step 2: (D = c/\sqrt{C\cdot C})</td>
</tr>
</tbody>
</table>

\[(n^C)^2\]
<table>
<thead>
<tr>
<th>STEP</th>
<th>KEY ENTRY</th>
<th>KEY CODE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>179</td>
<td>rcl .2</td>
<td>45 .2</td>
<td>Recall $m_c$ to stack</td>
</tr>
<tr>
<td>180</td>
<td>g x²</td>
<td>43 11</td>
<td>$(m_c)^2$</td>
</tr>
<tr>
<td>181</td>
<td>+</td>
<td>40</td>
<td>$(m_c)^2 + (n_c)^2$</td>
</tr>
<tr>
<td>182</td>
<td>rcl .1</td>
<td>45 .1</td>
<td>Recall $l_c$ to stack</td>
</tr>
<tr>
<td>183</td>
<td>g x²</td>
<td>43 11</td>
<td>$(l_c)^2$</td>
</tr>
<tr>
<td>184</td>
<td>+</td>
<td>40</td>
<td>$C^c = (l_c)^2 + (m_c)^2 + (n_c)^2$</td>
</tr>
<tr>
<td>185</td>
<td>√</td>
<td>11</td>
<td>$\sqrt{C^c}$</td>
</tr>
<tr>
<td>186</td>
<td>sto .9</td>
<td>44 .9</td>
<td>Store $\sqrt{C^c}$ in register .9</td>
</tr>
<tr>
<td>187</td>
<td>rcl .1</td>
<td>45 .1</td>
<td>Recall $l_c$ to stack</td>
</tr>
<tr>
<td>188</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall $\sqrt{C^c}$ to stack</td>
</tr>
<tr>
<td>189</td>
<td>÷</td>
<td>10</td>
<td>$l^D = l_c/\sqrt{C^c}$</td>
</tr>
<tr>
<td>190</td>
<td>rcl .2</td>
<td>45 .2</td>
<td>Recall $m_c$ to stack</td>
</tr>
<tr>
<td>191</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall $\sqrt{C^c}$ to stack</td>
</tr>
<tr>
<td>192</td>
<td>÷</td>
<td>10</td>
<td>$m^D = m_c/\sqrt{C^c}$</td>
</tr>
<tr>
<td>193</td>
<td>rcl .3</td>
<td>45 .3</td>
<td>Recall $n_c$ to stack</td>
</tr>
<tr>
<td>194</td>
<td>rcl .9</td>
<td>45 .9</td>
<td>Recall $\sqrt{C^c}$ to stack</td>
</tr>
<tr>
<td>195</td>
<td>÷</td>
<td>10</td>
<td>$n^D = n_c/\sqrt{C^c}$</td>
</tr>
<tr>
<td>196</td>
<td>g rtn</td>
<td>43 32</td>
<td>Return — end of subroutine</td>
</tr>
</tbody>
</table>