United States
Department of
Agriculture

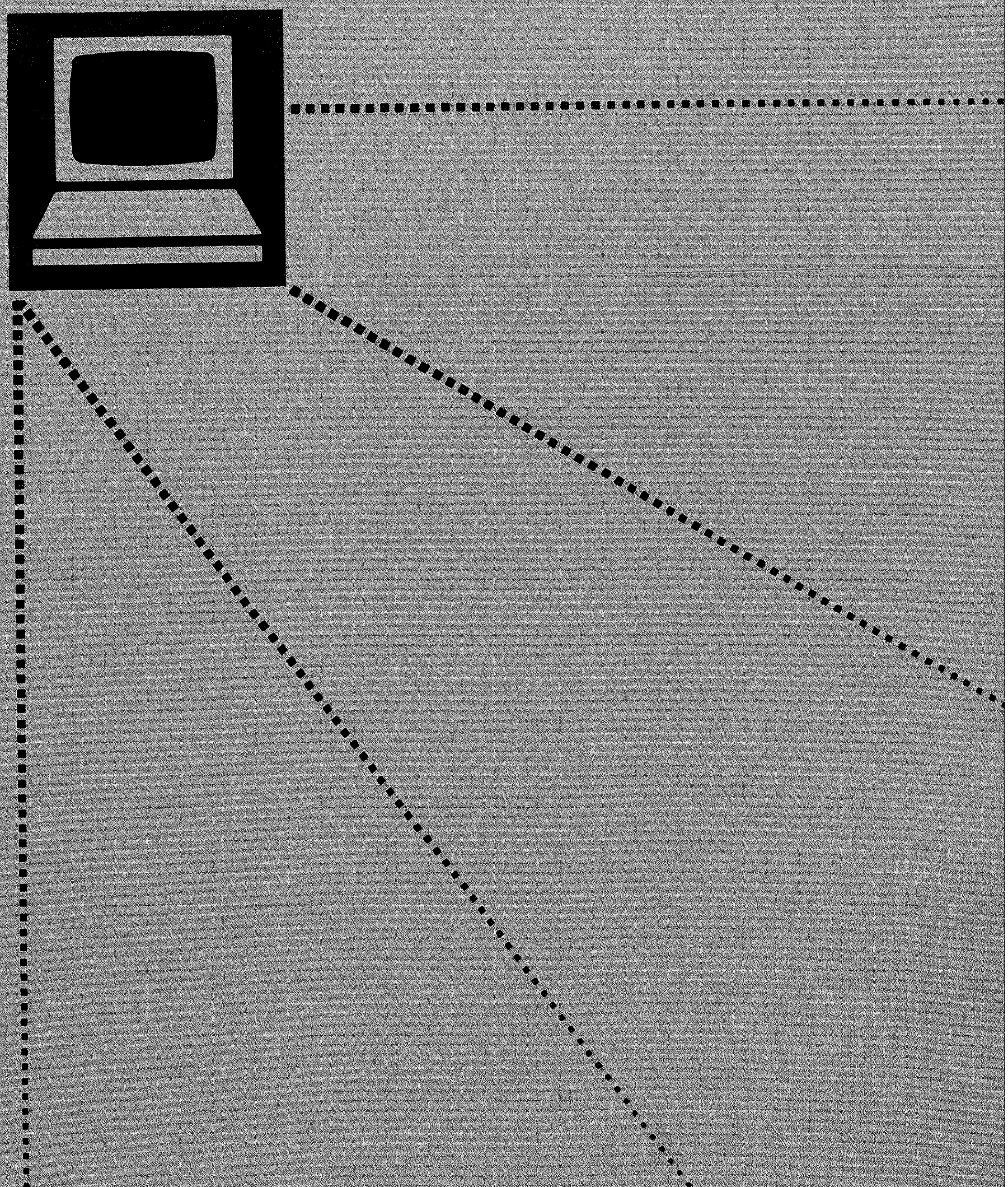
Forest
Service

North Central
Forest Experiment
Station

General Technical
Report NC-117

Computer Program for Calculating and Plotting Fire Direction and Rate of Spread

James E. Eenigenburg



North Central Forest Experiment Station
Forest Service—U.S. Department of Agriculture
1992 Folwell Avenue
St. Paul, Minnesota 55108
Manuscript approved for publication June 2, 1987
1987

COMPUTER PROGRAM FOR CALCULATING AND PLOTTING FIRE DIRECTION AND RATE OF SPREAD

James E. Eenigenburg, *Computer Programmer, East Lansing, Michigan*

An algorithm has been developed for calculating the direction and rate of spread of a fire across a triangular plot (Simard *et al.* 1984). Data required are the times of arrival of the fire at any three points (which form the three corners of a triangle), the length of the two sides adjacent to the fire's first arrival point, and the included angle. A program has been written to perform the calculations on a TI-59[1] hand-held calculator (Eenigenburg 1983). In this paper, we extend the computational procedures to allow plots with more than three time measurement points. By comparing the rates and directions derived from each triangle formed from four or more points taken three at a time, we can evaluate the uniformity of fire spread across the plot.

FIELD PROCEDURES

If time and accessibility permit, a systematic network of triangular grid points can be precisely deployed, following a pattern similar to that shown by Simard *et al.* (1984). The pattern can be modified to suit specific needs. Alternatively, an approximate pattern can be paced out quickly, with precise post-fire measurements made to determine distances and bearings. The information needed can even be obtained without pre-established measuring points.

Several methods can be used to obtain time data. We can follow the advancing front of low intensity fires and drop numbered tags on the ground at specific time intervals. Post-fire measurements can then be made in the same way as with the approximately located pre-established points. Or, we can observe the fire as it arrives at points pre-marked with a stake. In addition small, self-contained electronic timers can be used for obtaining fire arrival times in higher intensity fires (Blank and Simard 1983).

Using the algorithm presented here can eliminate the tedious job of measuring two side lengths and included angles for all possible triangles in the field. Instead, the location of each point in a plot is defined by the distance and bearing from a common point. Given this information, the program calculates the necessary lengths and angles for every possible triangle in the plot.[2] An important feature is that the common point from which the bearings and distances are obtained need not be contained within the plot. Thus, we can establish one common point for the entire fire and obtain all measurements from it. This is useful when missing or erroneous data or highly variable fire behavior requires unplanned plot combinations. Also, using a single common point for a fire with n fire arrival times requires a minimal n-1 distance and bearing observations.

PROGRAM SPREAD

SPREAD is written in ANSI-standard FORTRAN 77 and has been tested on a CDC Cyber 170-750[1] computer. Compile time was 2.423 seconds. The example in Appendices 3 and 4 incorporates 19 separate plots with an average of four time measurements per plot. Execution including optional graphics output required 19,072 words of memory and 0.702 seconds. Without graphics, the program executed in 0.250 seconds, using 17,664 words of memory.

The program (Appendix 1) contains a MAIN and 19 subprograms. There are 1,287 lines of code including 434 lines of comment. The program's 182 variables are listed and defined in Appendix 2. The program is internally documented; user instructions will not be repeated here. The user will find enough information

[1] *Use of trade names does not constitute official endorsement of the product by the USDA Forest Service.*

[2] *Due to the error inherent in field measurements, the program ignores triangles with an angle less than 0.2 radians (about 11.5 degrees). The user can easily adjust this limit by following instructions included in the BLOCK DATA subprogram.*

in MAIN to run the program. BLOCK DATA also discusses restrictions on some variables that the user might wish to modify. For those who want an in-depth understanding of the program, comments have been included in every subprogram and in the variable list.

AN EXAMPLE

We tested Program SPREAD using data obtained from a prescribed fire conducted by the Canadian Forestry Service, Great Lakes Forest Research Centre, in the Province of Ontario.[3] The purpose of the fire was to study pre-steady state fire behavior during the acceleration period. Thus, a single point source was used for ignition.

Before the fire, a grid of 18 electronic timers was located downwind from the fire's origin. This grid began approximately 20 meters from the origin to allow the fire time to stabilize into a quasi-steady spread condition. Data were obtained from 15 of the 18 timers (points A through O, fig. 1). The grid was divided into six plots, each containing four or five timers (most timers were used in more than one plot). Distance and bearing information for each plot was calculated relative to one point within the plot.

Also before ignition, three lines of metal stakes were placed at 5-meter intervals fanning out from the origin (point Z). In this area, times were manually recorded as the fire advanced (points P through X, fig. 1). Distances and bearings for these points were measured relative to the origin (the origin itself was included as a point in only one of the five plots, numbered 7 to 11, formed in this area). Four additional plots (numbered 12 to 15) were established connecting this area to the original grid.

The variability of fire behavior along line IK was disclosed in the first run of the program, so three 3-point plots (numbered 16 to 18) and a 5-point plot (number 19) were added to obtain information for different permutations of triangles that crossed the boundaries of the original plots. The input file for the example is shown in Appendix 3.

The output file for the example (Appendix 4) shows the area for each plot and the rate and direction of spread for each accepted triangle in the plot.

This is followed by a summary table where the vector average is presented (calculated by weighting each individual vector by its triangles area). Conceptually, the spread vector measures the rate at which the fire advances from one point to another along the direction of spread and is, therefore, appropriate for fire growth calculations.

The summary table also includes arithmetic means of the rate and direction, again weighted by area. Conceptually, the arithmetic mean measures the rate at which the fire passes a point, and is therefore appropriate for fire intensity calculations.

After analyzing each of the individual plots, the program produces an OVERALL STATISTICS summary table where each plot is entered, weighted by its area. Note that adding the extra plots (numbered 16 to 19) means that the area defined by HKNIGH is included twice in this summary, as is the thin sliver of area defined by triangle XUIX where plots 11 and 14 overlap plot 15.

In the OVERALL STATISTICS, the 'TOTAL AREA' is calculated as follows. Starting at the northernmost point, the program sweeps clockwise until a plot point is reached. Continuing from that point, another clockwise sweep is made until the next point is found. This continues until the first point is again reached, and the area of the convex polygon just described is calculated. In the example, the points were located in the following order: ACFONZWXA.

Note that this area may include sections that are not included in the summary table. In the example, plot number 12, QNZQ, did not contain a triangle with a large enough angle and was dropped from the analysis. Another area, ADXA, was never included in the original data set. Nevertheless, both areas are included in the 'TOTAL AREA' calculation (neither is included in the 'SUM OF ALL PLOT AREAS' calculation).

The graphics part of this program has been designed for use on the CALCOMP[1] plotter. If the user lacks such a plotter, the output file QRATE can still be obtained by setting the variable GRAPH to .FALSE. in the BLOCK DATA subprogram. To help users modify the program for different plotters, the seven CALCOMP commands used are described in Appendix 5.

The CALCOMP plotter produced the drawing shown in figure 1. The plotter identifies timer locations and plot boundaries, and it plots one scaled rate of spread vector for each triangle analyzed. From this plot and the output file, we can see that spread direction and rate are highly variable within the area bounded by KMLIK. Starting at 10 minutes after ignition, a finger of fire advanced through the above region to the far side of the plot in just 6 minutes. The fire then spread as a backing fire to the NNW and a head fire to the SSE.

---

[3] *Douglas J. McRae, Forest Fire Research Officer, Great Lakes Forest Research Centre, personal communication.*

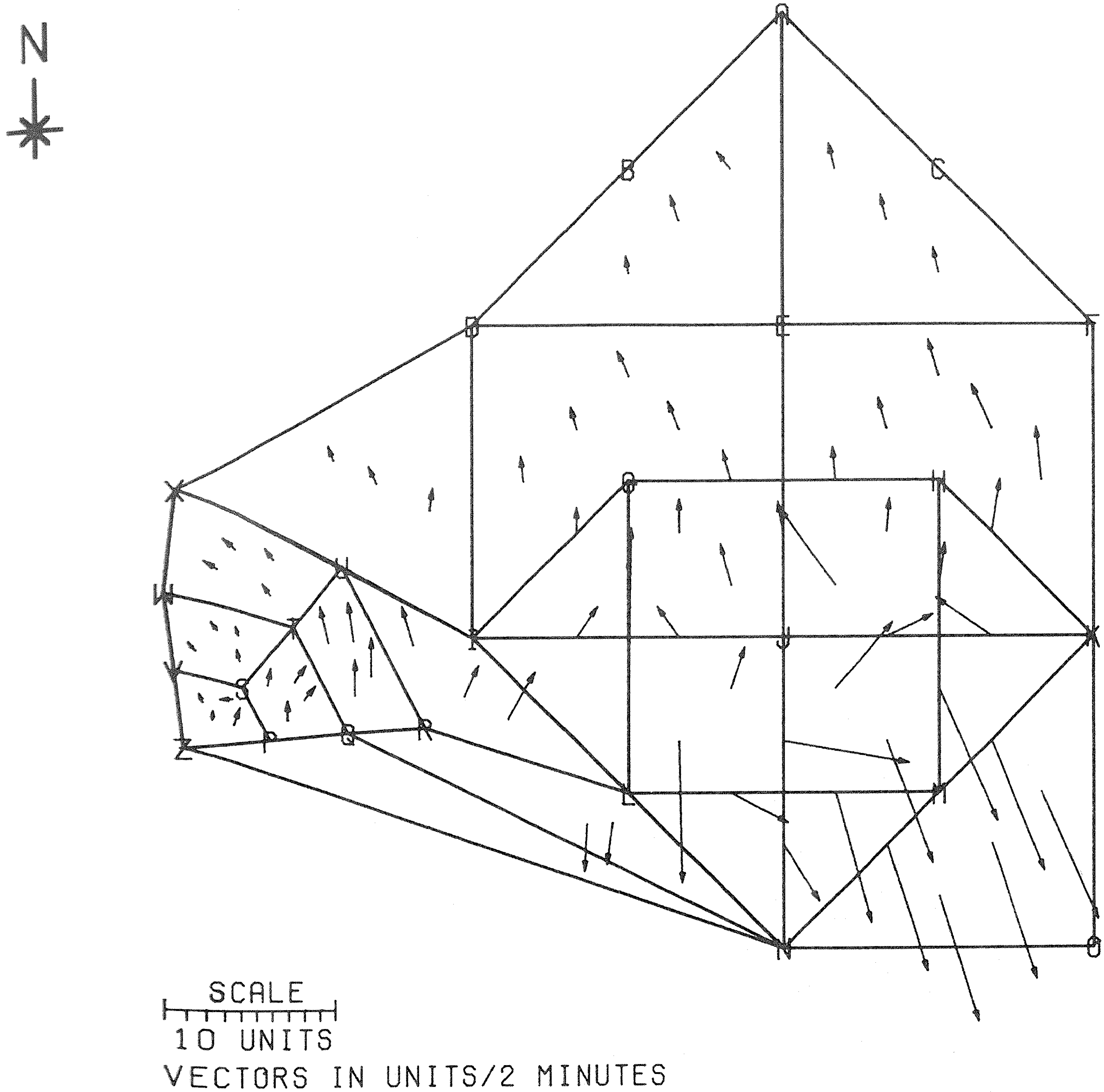


Figure 1.—*CALCOMP plotter output for the prescribed burn example. The plot program selects a spread vector scale so that the third largest vector is less than 1.5 inches.*

The procedure described in this paper documented a fairly complex spread pattern. It is reasonable to conclude that rate of spread can be estimated reliably for most prescribed burns and wildfires using this approach.

REFERENCES

Blank, Richard W.; Simard, Albert J. 1983. An electronic timer for measuring spread rates of wildland fires. Res. Note NC-304. St. Paul, MN: U.S. Department of Agriculture, Forest Service, North Central Forest Experiment Station. 4 p.

Eenigenburg, James E. 1983. User's guide to calculating rate of fire spread. Gen. Tech. Rep. NC-89. St. Paul, MN: U.S. Department of Agriculture, Forest Service, North Central Forest Experiment Station. 17 p.

Simard, Albert J.; Eenigenburg, James E.; Adams, Kenneth B.; Nissen, Roger L., Jr.; Deacon, Arthur G. 1984. A general procedure for sampling and analyzing wildland fire spread. Forest Science. 30(1): 51-64.

APPENDIX 1

```
      PROGRAM SPREAD
C
C     PROGRAMMED IN ANSI-STANDARD  FORTRAN 77  BY JAMES E. EENIGENBURG.
C     DOCUMENTATION, VARIABLE LIST, AND  SAMPLE RUN ARE PROVIDED IN THE
C     NC GENERAL TECHNICAL REPORT 'COMPUTER PROGRAM FOR CALCULATING AND
C     PLOTTING FIRE DIRECTION AND RATE OF SPREAD', 1987, BY
C
C                          JAMES E. EENIGENBURG
C                U.S. DEPARTMENT OF AGRICULTURE, FOREST SERVICE
C                    NORTH CENTRAL FOREST EXPERIMENT STATION
C                          1407 S. HARRISON ROAD
C                         EAST LANSING, MI  48823
C                             (517) 355-7740
C
C     ------------------------------------------------------------------
C
C     THE PROGRAM IS PROVIDED 'AS IS' WITHOUT WARRANTY OF ANY KIND, EI-
C     THER EXPRESSED OR  IMPLIED, INCLUDING BUT  NOT LIMITED TO IMPLIED
C     WARRANTIES OF FITNESS FOR A  PARTICULAR PURPOSE.  THE ENTIRE RISK
C     AS TO THE QUALITY AND PERFORMANCE  OF THE PROGRAM IS WITH THE US-
C     ER.  SHOULD THE PROGRAM PROVE DEFECTIVE, THE USER ASSUMES THE EN-
C     TIRE COST OF ALL NECESSARY SERVICING, REPAIR, OR CORRECTION.
C
C     IN NO EVENT WILL THE USDA FOREST  SERVICE OR THE AUTHOR BE LIABLE
C     FOR ANY DAMAGES,  INCLUDING BUT NOT LIMITED  TO ANY LOST PROFITS,
C     LOST SAVINGS, OR OTHER INCIDENTAL  OR CONSEQUENTIAL DAMAGES ARIS-
C     ING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM.
C
C     THE USER MAY NOT ASSERT ANY PROPRIETARY RIGHTS THERETO NOR REPRE-
C     SENT THE PROGRAM  TO ANYONE AS  OTHER THAN A  GOVERNMENT-PRODUCED
C     COMPUTER PROGRAM.
C
C     ------------------------------------------------------------------
C
C     PROGRAM SPREAD CALCULATES  THE DIRECTION AND  RATE OF SPREAD OF A
C     FIRE ACROSS A PLOT  WHERE THE TIME OF FIRE  ARRIVAL AT EACH OF AT
C     LEAST THREE AND AS MANY AS 'NVL' POINTS HAS BEEN RECORDED.  A BA-
C     SIC DESIGN IS ILLUSTRATED  AT RIGHT WHERE THE LET-
C     TERS A TO E INDICATE TIMER  POSITIONS (ANY CHARAC-    D       C
C     TER OTHER  THAN '#'  MAY BE  USED AS LONG  AS EACH
C     TIMER HAS A UNIQUE  IDENTIFIER).  THE PROGRAM DOES        E
C     NOT REQUIRE  THE  TIMERS TO BE  PLACED  EXACTLY IN
C     SUCH A SQUARE-LIKE PATTERN, AND WILL WORK WITH AR-    A       B
C     BITRARILY PLACED TIMERS.  CHOOSE ANY POINT (A TIM-
C     ER POINT, OR  ANY OTHER  POINT INSIDE  OR OUTSIDE THE  PLOT) AS A
C     MARKER FROM WHICH TO  MEASURE THE DISTANCE  AND COMPASS DIRECTION
C     OF EACH OF THE TIMERS.
```

```
C
C     IN THE ABOVE DIAGRAM,  LET POINT E BE CHOSEN  AS THE MARKER.  IN-
C     PUTS TO  THE PROGRAM  ARE:   1) THE  TIME THE  FIRE  REACHED EACH
C     POINT; 2) THE DISTANCE OF EACH  POINT TO POINT E; AND 3) THE COM-
C     PASS DIRECTION OF EACH POINT FROM POINT E.
C
C     THE PROGRAM EXPECTS  INPUTS ON A FILE NAMED  'QDATA'.  THE FORMAT
C     FOR EACH INPUT RECORD IS AS FOLLOWS:  (A1, 2I3, F7.2, I4)
C        A1     THE LETTER IDENTIFYING THE POINT.
C       2I3     THE TIME WHEN THE FIRE REACHED THE POINT (MINSEC).
C        F7.2   THE DISTANCE OF THE POINT TO THE MARKER.
C        I4     COMPASS DIRECTION OF THE POINT FROM THE MARKER ( DEG).
C     EACH INPUT RECORD MUST  CONTAIN ONE AND  ONLY ONE POINT.   POINTS
C     WHERE TIMERS DID NOT  WORK MAY EITHER BE  SKIPPED, OR THE MINUTES
C     ENTERED WITH  (999).   IF THE  MARKER POINT  IS ONE OF  THE TIMER
C     POINTS, ITS INPUT FIELDS  FOR DISTANCE AND  COMPASS DIRECTION MAY
C     EITHER BE LEFT BLANK OR FILLED WITH ZEROS.
C
C     FOR EACH PLOT, THE PROGRAM CONSIDERS THE TRIANGLE FORMED BY EVERY
C     POSSIBLE COMBINATION OF THREE  POINTS AND CALCULATES THE RATE AND
C     DIRECTION OF SPREAD  ACROSS EACH.  ON OUTPUT  FILE 'QRATE', RATES
C     ARE WRITTEN IN  UNITS PER MINUTE  (UNITS OF THE  SAME TYPE AS THE
C     INPUT DISTANCES)  AND DIRECTIONS  ARE WRITTEN IN  COMPASS DEGREES
C     (FROM 0 DEGREES NORTH CLOCKWISE THROUGH 360 DEGREES).
C
C     REPRESENTING THE RATE  AND DIRECTION OF SPREAD  FOR EACH TRIANGLE
C     AS A VECTOR, AND WEIGHTING EACH  VECTOR BY ITS TRIANGLE'S AREA, A
C     VECTOR AVERAGE FOR THE PLOT IS DETERMINED.
C
C     WEIGHTING THE RATE OF  SPREAD FOR EACH TRIANGLE  BY ITS AREA, THE
C     WEIGHTED ARITHMETIC  MEAN OF THE RATE IS THEN  CALCULATED.  SIMI-
C     LARLY, THE WEIGHTED ARITHMETIC MEAN OF THE DIRECTION OF SPREAD IS
C     FOUND, AS WELL AS STANDARD DEVIATIONS FOR EACH.
C
C     COMPARING THE VECTOR  AVERAGE WITH A NEW  VECTOR REPRESENTING THE
C     ARITHMETIC  MEANS OF THE  RATE AND  DIRECTION, THE  RATE AT WHICH
C     THESE TWO VECTORS ARE DIVERGING (LABLED 'V/A DIVERGENCE') IS ALSO
C     WRITTEN TO FILE 'QRATE'.
C
C     FINALLY, THE VARIABILITY  OF THE RATE AND THE  VARIABILITY OF THE
C     DIRECTION ARE CONSIDERED  SEPARATELY, AND  EACH IS GIVEN A VERBAL
C     DESCRIPTION:  LOW, MODERATE, OR  HIGH.  (SIMARD, ALBERT J.; EENI-
C     GENBURG, JAMES E.; ADAMS, KENNETH B.; NISSEN, ROGER L., JR.; DEA-
C     CON, ARTHUR G. 1984. A GENERAL PROCEDURE FOR SAMPLING AND ANALYZ-
C     ING WILDLAND FIRE SPREAD. FOREST SCIENCE 30(1): 51-64.)
C
C     UP TO 'NSL' PLOTS MAY  BE INVESTIGATED WITH  ONE EXECUTION OF THE
C     PROGRAM.  SIMPLY SEPARATE EACH PLOT ON FILE 'QDATA' WITH A SINGLE
C     RECORD CONTAINING THE  SINGLE CHARACTER '#'  IN THE FIRST COLUMN.
C     ALTHOUGH ALL  POINTS  IN ANY ONE  PLOT MUST  USE THE  SAME MARKER
C     POINT, EACH PLOT MAY HAVE ITS OWN UNIQUE MARKER.
```

```
C
C     AFTER ALL PLOTS ARE ANALYZED INDIVIDUALLY, EACH PLOT (WEIGHTED BY
C     ITS AREA) IS TAKEN AS A SINGLE ENTRY IN A GROUP AVERAGE, AND SUM-
C     MARY STATISTICS FOR THE AGGREGATION ARE WRITTEN TO FILE 'QRATE'.
C
C     LIMITS 'NVL' AND 'NSL',  MENTIONED ABOVE, MAY  BE SET BY THE USER
C     WITH THE  PARAMETER  STATEMENTS  AT THE  BEGINNING OF  SUBPROGRAM
C     QUEUE.  LIMIT 'NGL', THE TOTAL  NUMBER OF TIMERS POSSIBLE, IS SET
C     BY A PARAMETER STATEMENT AT THE BEGINNING OF SUBPROGRAM ORIGIN.
C
C     ALL OTHER USER ADJUSTABLE  VARIABLES ARE  GROUPED TOGETHER IN THE
C     BLOCK DATA SUBPROGRAM (E.G., IF ONE HAS ACCESS TO A CALCOMP PLOT-
C     TER, A MAP OF  THE FIRE  WITH SPREAD  VECTORS CAN BE  OBTAINED BY
C     SETTING THE LOGICAL 'GRAPH' TO .TRUE.).
C     -----------------------------------------------------------------
C
      CALL QUEUE
      END



      SUBROUTINE QUEUE
C
C     PROGRAM HANDLES UP TO 'NVL' POINTS  PER PLOT,  IGNORING ANY ADDI-
C     TIONAL POINTS FOR THE PLOT ENTERED ON FILE 'QDATA'.
      PARAMETER (NVL=5)
C
C     PROGRAM HANDLES UP TO 'NSL' PLOTS,  IGNORING ANY ADDITIONAL PLOTS
C     ENTERED ON FILE 'QDATA'.
      PARAMETER (NSL=20)
C     -----------------------------------------------------------------
C
      COMMON /LIMIT/ NV, NVLIM,  NS, NSLIM,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /CNTRL/ PWT, GRAPH, AMIN
      COMMON /PLOTS/ NP,PWGT, PRSUM,PRSUM2, PDSUM,PDSUM2, PXSUM,PYSUM
C
      CHARACTER VPT*(NVL), VPP*(NVL+1)
      DIMENSION VPX (NVL), VPY (NVL),  TIME(NVL), VDIR(NVL), NPP(NVL)
      CHARACTER SPT(NSL)*(NVL), SPP(NSL)*(NVL+1)
      DIMENSION SPX(NVL,NSL),   SPY(NVL,NSL),     MPT(NSL)
      REAL      MIN
      LOGICAL   GRAPH, MORE
      CHARACTER PT*1,  RECORD*18, TH(4)*2
      CHARACTER FMT1*39, FMT2*33, FMT3*39, FMT4(2)*28
C
      DATA MORE /.TRUE./, TH /'ST','ND','RD','TH'/, RAD /57.2957795131/
      DATA FMT1 /'('' STATISTICS FOR PLOT #'', I1, ''--'', A)'/
      DATA FMT2 /'('' AREA:  '', I1, '' SQUARE UNITS'')'/
      DATA FMT3 /'('' PLOT AREAS:  '', I1, '' SQUARE UNITS'')'/
      DATA FMT4 /'('' PROBLEM IN THE '', I1, ''TH',
     +          ' RECORD OF PLOT #'', I1, '':'')'/
C     -----------------------------------------------------------------
```

```
C
C     CONVERT PARAMETER LIMITS TO VARIABLES.
      NVLIM = NVL
      NSLIM = NSL
C
C     OPEN INPUT, OUTPUT, AND SCRATCH FILES.
      OPEN   (INPT, FILE='QDATA', STATUS='OLD', ERR=100, BLANK='ZERO')
      REWIND (INPT)
      OPEN   (IOUT, FILE='QRATE', STATUS='NEW', ERR=110)
      REWIND (IOUT)
      IF (GRAPH) THEN
         OPEN   (IVEC, STATUS='SCRATCH')
         REWIND (IVEC)
      END IF
C
C     WRITE OUTPUT HEADING IDENTIFYING PROGRAM.
      LINE = LINES
      LINE = LINED (1) + 3
      WRITE (IOUT, '('' USDA FOREST SERVICE SPREAD RATE PROGRAM''/)')
      WRITE (IOUT, '(''           RATES IN UNITS/MINUTE'')')
      WRITE (IOUT, '(''       DIRECTIONS IN DEGREES AZIMUTH'')')
C
C     READ INPUT DATA ONE PLOT AT A TIME.
   10 NV = 0
      NS = NS + 1
      IF (NS .LE. NSL) THEN
   11    READ (INPT, '(A,  2F3.0,  F7.2, F4.0)', ERR=120, END=30)
     +               PT, MIN,SEC, DIST, COMP
         IF (PT .EQ. '#') GO TO 40
         IF (NV .LT. NVL .AND. MIN .LT. 999.0) THEN
C
C           CHECK FOR UNIQUENESS OF POINT.
            IF (NV .GE. 1) THEN
               DO 20 I=1,NV
   20          IF (PT .EQ. VPT(I:I)) GO TO 11
            END IF
C
C           ENTER NEW POINT INTO ARRAYS.
            NV         = NV + 1
            VPT (NV:)  = PT
            TIME(NV)   = MIN + SEC/60.0
            COMP       = COMP/RAD
            VPX (NV)   = DIST*COS(COMP)
            VPY (NV)   = DIST*SIN(COMP)
            IF (GRAPH) THEN
               SPT(NS)(NV:) = PT
               SPX(NV,NS)   = VPX(NV)
               SPY(NV,NS)   = VPY(NV)
            END IF
         END IF
         GO TO 11
      END IF
   30 CLOSE (INPT, STATUS='KEEP')
      MORE = .FALSE.
```

```
C
C     PROCESS PLOT.
   40 IF (NV .GE. 3) THEN
         PWT  = AREA (NV, VPT, VPX, VPY, VPP, VDIR, NPP)
         NPWT = NINT(PWT)
         IF (NPWT .GT. 0) THEN
            IF (GRAPH) SPP(NS) = VPP
            LINE = LINED (NV) + 3
            WRITE (FMT1(29:29), '(I1)') INT(ALOG10(REAL(NS))) + 1
            WRITE (IOUT, FMT1) NS, VPP
            IF (NPWT .LT. 1.0E9) THEN
               WRITE (FMT2(15:15), '(I1)') INT(ALOG10(REAL(NPWT))) + 1
               WRITE (IOUT, FMT2) NPWT
            END IF
            WRITE (IOUT, '(23X, ''RATE  DIRECTION'')')
            WRITE (IOUT, '('' INDIVIDUAL TRIANGLES'')')
            CALL CHRONO (VPT, VPX, VPY, TIME)
         ELSE
            CALL NOPLOT (VPT)
         END IF
      ELSE IF (NV .GE. 1) THEN
         CALL NOPLOT (VPT)
      ELSE
         NS = NS - 1
      END IF
      IF (MORE) GO TO 10
C
C     ----------------------------------------------------------------------
C     OUTPUT SUMMARY HEADING.
      IF (NP .GE. 2) THEN
         WRITE (IOUT, '(/)')
         IF (GRAPH) THEN
            LINE = LINED (6)
         ELSE
            LINE = LINED (5)
         END IF
         WRITE (IOUT, '('' OVERALL STATISTICS'')')
      END IF
C
C     IF GRAPHICS, DRAW PLOTS AND OUTPUT AREA CALCULATIONS.
      IF (GRAPH) THEN
         ENDFILE (IVEC)
         CALL CALCOM (SPT, SPX, SPY, SPP, MPT)
         CLOSE   (IVEC)
C
C     IF NO GRAPHICS, OUTPUT THE AREA OF THE SUM OF THE PLOTS.
      ELSE IF (NP .GE. 2) THEN
         NPWGT = NINT(PWGT)
         IF (NPWGT .LT. 1.0E9) THEN
            WRITE (FMT3(21:21), '(I1)') INT(ALOG10(REAL(NPWGT))) + 1
            WRITE (IOUT, '('' SUM OF ALL'')')
            WRITE (IOUT, FMT3) NPWGT
            CALL NOTE1
         ELSE
            CALL NOTE2
         END IF
      END IF
```

```
C
C     OUTPUT SUMMARY STATISTICS.
      IF (NP .GE. 2 .AND. (PXSUM .NE. 0.0 .OR. PYSUM .NE. 0.0)) THEN
         WRITE (IOUT, '(/ 23X, ''RATE  DIRECTION'')')
         CALL STATS (NP,PWGT, PRSUM,PRSUM2, PDSUM,PDSUM2, PXSUM,PYSUM)
      END IF
      WRITE (IOUT, '(''1'')')
      CLOSE (IOUT, STATUS='KEEP')
      STOP
C     ------------------------------------------------------------------
C
C     ERROR OPENING FILE 'QDATA'.
  100 STOP 'FILE QDATA COULD NOT BE FOUND.'
C
C     ERROR OPENING FILE 'QRATE'.
  110 CLOSE (INPT, STATUS='KEEP')
      STOP 'FILE QRATE ALREADY EXISTS.'
C
C     ERROR READING RECORD FROM FILE 'QDATA'.
  120 BACKSPACE (INPT)
      READ  (INPT, '(A)') RECORD
      CLOSE (INPT, STATUS='KEEP')
      LINE = LINED (2)
      WRITE (IOUT, '('' ''''READ'''' ERROR DETECTED ON FILE QDATA''/)')
      NV = NV + 1
      WRITE (FMT4(1)(23:23), '(I1)') INT(ALOG10(REAL(NV))) + 1
      WRITE (FMT4(1)(27:28), '(A2)') TH(MIN0(NV, 4))
      WRITE (FMT4(2)(22:22), '(I1)') INT(ALOG10(REAL(NS))) + 1
      WRITE (IOUT, FMT4) NV, NS
      WRITE (IOUT, '(A19)') RECORD
      WRITE (IOUT, '('' PMINSECDDDD.DDCCCC   IS THE CORRECT FORMAT'')')
      WRITE (IOUT, '(''    WHERE--                                 '')')
      WRITE (IOUT, '(''        P = THE LETTER IDENTIFYING THE POINT'')')
      WRITE (IOUT, '(''  MINSEC = THE TIME FIRE REACHED THE POINT '')')
      WRITE (IOUT, '('' DDDD.DD = DISTANCE OF POINT FROM MARKER   '')')
      WRITE (IOUT, '(''    CCCC = DIRECTION OF POINT FROM MARKER  '')')
      WRITE (IOUT, '(/ '' PLEASE AMEND RECORD AND RESUBMIT FILE.  '')')
      WRITE (IOUT, '(''1'')')
      CLOSE (IOUT, STATUS='KEEP')
      STOP 'READ ERROR ON FILE QDATA.'
      END


      FUNCTION AREA (N, PT, PX, PY, PP, DIR, NPP)
C
C     CALCULATES THE AREA OF A CONVEX POLYGON.
C
      CHARACTER *(*) PT, PP
      DIMENSION PX(N), PY(N), DIR(N), NPP(N)
C
C     DEGREES PER RADIAN.
      DATA RAD /57.2957795131/
```

```
C
C     FUNCTIONS TO CALCULATE DISTANCES BETWEEN TWO POINTS.
      XCORD (I, J) = PX(J) - PX(I)
      YCORD (I, J) = PY(J) - PY(I)
      SIDE  (I, J) = SQRT(XCORD(I,J)**2 + YCORD(I,J)**2)
C     ___________________________________________________________________
C
C     FIND INDEX 'NX' OF NORTHERNMOST POINT 'PP(1:1)' IN PLOT.
      NX = 1
      DO 10 I=2,N
   10 IF (PX(I) .GT. PX(NX)) NX = I
      K       = 1
      PP (1:) = PT(NX:NX)
      NPP(1)  = NX
C
C     FIND COMPASS DIRECTION OF ALL OTHER POINTS FROM 'NX'.
      DO 20 J=1,N
      IF (J .NE. NX) THEN
         IF (PX(J) .NE. PX(NX) .OR. PY(J) .NE. PY(NX)) THEN
            DIR(J) = CIRCLE(RAD*ATAN2(YCORD(NX,J), XCORD(NX,J)), 180.0)
         ELSE
            DIR(J) = 999.0
         END IF
      END IF
   20 CONTINUE
      DIR(NX) = 360.0
C
C     FIND INDEX 'MX' OF NEXT CLOCKWISE PERIMETER POINT IN PLOT.
   30 MX = 1
      DO 31 I=2,N
      IF (DIR(I) .LT. DIR(MX)) THEN
         MX = I
      ELSE IF (DIR(I) .EQ. DIR(MX)) THEN
         IF (SIDE(NPP(K),I) .GT. SIDE(NPP(K),MX)) MX = I
      END IF
   31 CONTINUE
C
C     FIND COMPASS DIRECTION OF REMAINING POINTS FROM 'MX'.
      IF (MX .NE. NX) THEN
         K       = K + 1
         PP (K:) = PT(MX:MX)
         NPP(K)  = MX
         DO 40 J=1,N
         IF (J .NE. MX .AND. DIR(J) .LE. 450.0) THEN
            IF (PX(J) .NE. PX(MX) .OR. PY(J) .NE. PY(MX)) THEN
               DIR(J) = CIRCLE (RAD*ATAN2(YCORD(MX,J), XCORD(MX,J)),
     +                          DIR(MX) + 180.0)
            ELSE
               DIR(J) = 999.0
            END IF
         END IF
   40    CONTINUE
         DIR(MX) = 999.0
         GO TO 30
      END IF
      PP(K+1:) = PP(1:1)
```

```
C
C     USING 'NX' WITH  EACH SET OF TWO  ADJACENT PERIMETER  POINTS, SUM
C     THE AREAS OF THE TRIANGLES FORMED TO FIND THE PLOT'S AREA.
      AREA = 0.0
      AX   = SIDE (NPP(2), NX)
      DO 50 I=3,K
      BX   = AX
      AX   = SIDE (NPP(I), NX)
      AB   = SIDE (NPP(I), NPP(I-1))
      S    = 0.5*(AX + BX + AB)
      ABX  = SQRT(AMAX1(0.0, S*(S - AX)*(S - BX)*(S - AB)))
   50 AREA = AREA + ABX
      RETURN
      END



      SUBROUTINE CHRONO (VPT, VPX, VPY, TIME)
C
C     SELECTS TRIANGLES BY ORDERING THE PLOT'S POINTS CHRONOLOGICALLY.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /CNTRL/ PWT, GRAPH, AMIN
      COMMON /TRIAN/ NT,TWGT, TRSUM,TRSUM2, TDSUM,TDSUM2, TXSUM,TYSUM
      COMMON /PLOTS/ NP,PWGT, PRSUM,PRSUM2, PDSUM,PDSUM2, PXSUM,PYSUM
      COMMON /MEANS/ RMEAN, DMEAN, XMEAN, YMEAN, RLOW, RMOD, DLOW, DMOD
C
      CHARACTER VPT*(*)
      DIMENSION VPX(NVL), VPY(NVL), TIME(NVL)
C     ____________________________________________________________________
C
C     INITIALIZE COUNTERS FOR SUBSEQUENT MEAN AND VECTOR CALCULATIONS.
      NT     = 0
      TWGT   = 0.0
      TRSUM  = 0.0
      TRSUM2 = 0.0
      TDSUM  = 0.0
      TDSUM2 = 0.0
      TXSUM  = 0.0
      TYSUM  = 0.0
C
C     DISTINGUISH EACH TRIANGLE ORDERING POINTS CHRONOLOGICALLY.
      DO 10 I=  1,NV-2
      DO 10 J=I+1,NV-1
      DO 10 K=J+1,NV
      IF (TIME(I) .LE. TIME(J) .AND. TIME(I) .LE. TIME(K)) THEN
         IF (TIME(J) .LE. TIME(K)) THEN
            CALL TRIGON (I,J,K, VPT,VPX,VPY,TIME)
         ELSE
            CALL TRIGON (I,K,J, VPT,VPX,VPY,TIME)
         END IF
```

```
      ELSE IF (TIME(J) .LE. TIME(I) .AND. TIME(J) .LE. TIME(K)) THEN
         IF (TIME(I) .LE. TIME(K)) THEN
            CALL TRIGON (J,I,K, VPT,VPX,VPY,TIME)
         ELSE
            CALL TRIGON (J,K,I, VPT,VPX,VPY,TIME)
         END IF
      ELSE
         IF (TIME(I) .LE. TIME(J)) THEN
            CALL TRIGON (K,I,J, VPT,VPX,VPY,TIME)
         ELSE
            CALL TRIGON (K,J,I, VPT,VPX,VPY,TIME)
         END IF
      END IF
   10 CONTINUE
C
C     IF THERE ARE NO TRIANGLES (OR NO RESULTANT VECTOR), RETURN.
      IF (NT .EQ. 0 .OR. (TXSUM .EQ. 0.0 .AND. TYSUM .EQ. 0.0)) THEN
         LINE = LINE + 2
         WRITE (IOUT, '('' THIS PLOT HAS INSUFFICIENT DATA.  IT IS''/
     +                   '' NOT INCLUDED IN THE OVERALL STATISTICS.'')')
         RETURN
      END IF

C     CALCULATE VECTOR AVERAGE, ARITHMETIC MEAN, AND STD. DEVIATION.
      IF (NT .GT. 1 .AND. LINE + 6 .GT. LINES) CALL SPLIT
      CALL STATS (NT,TWGT, TRSUM,TRSUM2, TDSUM,TDSUM2, TXSUM,TYSUM)
C
C     ADD THIS PLOT TO OVERALL FIGURES.
      IF (NP .GE. 1) DMEAN = CIRCLE (DMEAN, PDSUM/PWGT)
      NP     = NP     + 1
      PWGT   = PWGT   + PWT
      PRSUM  = PRSUM  + PWT*RMEAN
      PRSUM2 = PRSUM2 + PWT*RMEAN*RMEAN
      PDSUM  = PDSUM  + PWT*DMEAN
      PDSUM2 = PDSUM2 + PWT*DMEAN*DMEAN
      PXSUM  = PXSUM  + PWT*XMEAN
      PYSUM  = PYSUM  + PWT*YMEAN
      RETURN
      END



      SUBROUTINE TRIGON (I,J,K, VPT,VPX,VPY,TIME)
C
C     GIVEN ANY TRIANGLE, LET VERTEX  'A' BE THE POINT THE FIRE REACHED
C     FIRST AT TIME(I), VERTEX 'B' BE THE POINT THE FIRE REACHED SECOND
C     AT TIME(J), AND VERTEX 'C' BE THE  POINT THE FIRE REACHED LAST AT
C     TIME(K).  THE  RATE AND DIRECTION  OF SPREAD ARE  CALCULATED WITH
C     RESPECT TO TRIANGLE 'ABC', AND  THEN REORIENTATED WITH RESPECT TO
C     NORTH AS VIEWED FROM 'A'.
C
```

```
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /CNTRL/ PWT, GRAPH, AMIN
      COMMON /TRIAN/ NT,TWGT, TRSUM,TRSUM2, TDSUM,TDSUM2, TXSUM,TYSUM
C
      CHARACTER VPT*(*),  TRI*3
      DIMENSION VPX(NVL), VPY(NVL), TIME(NVL)
      LOGICAL   GRAPH
C
C     DEGREES PER RADIAN.
      DATA RAD /57.2957795131/
C
C     FUNCTIONS TO CALCULATE DISTANCES BETWEEN TWO POINTS.
      XCORD (I, J) = VPX(J) - VPX(I)
      YCORD (I, J) = VPY(J) - VPY(I)
      SIDE  (I, J) = SQRT(XCORD(I,J)**2 + YCORD(I,J)**2)
C
C     FUNCTION:  THE LAW OF COSINES.
      ANGLE (R,S,T) = ACOS(AMIN1(ABS((R*R + S*S - T*T)/(2.*R*S)), 1.0))
C     ____________________________________________________________________
C
C     FIND ALL SIDES (REJECT THOSE EQUAL TO ZERO) OF TRIANGLE 'ABC'.
      AB = SIDE (I, J)
      IF (AB .LE. 0.0) RETURN
      AC = SIDE (I, K)
      IF (AC .LE. 0.0) RETURN
      BC = SIDE (K, J)
      IF (BC .LE. 0.0) RETURN
C
C     FIND ALL ANGLES (REJECT PARTICULARLY NARROW ONES) OF 'ABC'.
      A = ANGLE (AB, AC, BC)
      IF (A .LT. AMIN) RETURN
      B = ANGLE (AB, BC, AC)
      IF (B .LT. AMIN) RETURN
      C = ANGLE (AC, BC, AB)
      IF (C .LT. AMIN) RETURN
C
C     FIND THE AREA OF TRIANGLE 'ABC'.
      ACSINA = AC*SIN(A)
      TWT    = 0.5*AB*ACSINA
C
C     CALCULATE DIRECTION AND RATE OF SPREAD RELATIVE TO SIDE 'AB'.
      IF (TIME(I) .NE. TIME(J)) THEN
         TIMJI = TIME(J) - TIME(I)
         TIMKI = TIME(K) - TIME(I)
         THETA = ATAN(TIMKI*AB/(TIMJI*ACSINA) - 1.0/TAN(A))
         RATE  = AB*COS(THETA)/TIMJI
         THETA = RAD*THETA
      ELSE
         RATE  = ACSINA/(TIME(K) - TIME(I))
         THETA = 90.0
      END IF
```

```
C
C     RELATE THETA TO COMPASS NORTH, AS VIEWED FROM POINT A.
      BAN = RAD*ATAN2(YCORD(I,J), XCORD(I,J))
      CAN = RAD*ATAN2(YCORD(I,K), XCORD(I,K))
      CAN = CIRCLE (CAN, BAN)
      IF (BAN .LE. CAN) THEN
         THETA = BAN + THETA
      ELSE
         THETA = BAN - THETA
      END IF
      THETA = CIRCLE (THETA, 180.0)
C
C     OUTPUT RATE AND DIRECTION FOR THIS TRIANGLE.
      TRI = VPT(I:I)//VPT(J:J)//VPT(K:K)
      IF (LINE .EQ. LINES) CALL SPLIT
      LINE = LINE + 1
      WRITE (IOUT, '(A19, F8.1, I8)') TRI, RATE, NINT(THETA)
C
C     ADD LATEST THETA TO SUMMATION VARIABLES.
      IF (NT .GE. 1) THETA = CIRCLE (THETA, TDSUM/TWGT)
      NT     = NT     + 1
      TWGT   = TWGT   + TWT
      TRSUM  = TRSUM  + TWT*RATE
      TRSUM2 = TRSUM2 + TWT*RATE*RATE
      TDSUM  = TDSUM  + TWT*THETA
      TDSUM2 = TDSUM2 + TWT*THETA*THETA
      THETA  = THETA/RAD
      RCOS   = RATE*COS(THETA)
      RSIN   = RATE*SIN(THETA)
      TXSUM  = TXSUM  + TWT*RCOS
      TYSUM  = TYSUM  + TWT*RSIN
C
C     IF REQUESTED, ADD VECTOR TO SCRATCH FILE.
      IF (GRAPH) WRITE (IVEC, '(A, 3F9.2)') TRI, RATE, RCOS, RSIN
      RETURN
      END



      SUBROUTINE STATS (N,WGT, RSUM,RSUM2, DSUM,DSUM2, XSUM,YSUM)
C
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /MEANS/ RMEAN, DMEAN, XMEAN, YMEAN, RLOW, RMOD, DLOW, DMOD
C
      CHARACTER *8 RVAR, DVAR
C
C     DEGREES PER RADIAN.
      DATA RAD /57.2957795131/
C
C     FUNCTION FOR CALCULATING WEIGHTED STANDARD DEVIATIONS.
      STDDEV (S, S2) = SQRT(AMAX1(0., FN/WGT*(S2 - S*S/WGT)/(FN - 1.)))
C     ----------------------------------------------------------------
```

```
C
C       CALCULATE VECTOR AVERAGE.
        XMEAN = XSUM/WGT
        YMEAN = YSUM/WGT
        VRATE = SQRT(XMEAN*XMEAN + YMEAN*YMEAN)
        VDIR  = CIRCLE (RAD*ATAN2(YMEAN, XMEAN), 180.0)
        IVDIR = NINT(VDIR)
C
C       CALCULATE ARITHMETIC MEANS.
        RMEAN = RSUM/WGT
        DMEAN = CIRCLE (DSUM/WGT, 180.0)
        IF (N .EQ. 1) RETURN
        IDMEAN = NINT(DMEAN)
C
C       CALCULATE ARITHMETIC STANDARD DEVIATIONS.
        FN    = N
        RSTD  = STDDEV (RSUM, RSUM2)
        DSTD  = STDDEV (DSUM, DSUM2)
        IDSTD = NINT(DSTD)
C
C       FIND RATE AT WHICH VECTOR AND ARITHMETIC AVERAGES ARE DIVERGING.
        VADIV = SQRT(AMAX1(0.0, VRATE*VRATE + RMEAN*RMEAN -
     +          2.0*VRATE*RMEAN*COS((VDIR - DMEAN)/RAD)))
C
C       CLASSIFY ARITHMETIC AVERAGES INTO CLASSES.
        COVAR = RSTD/RMEAN
        IF (COVAR .LE. RLOW) THEN
           RVAR = '     LOW'
        ELSE IF (COVAR .LT. RMOD) THEN
           RVAR = '    MOD.'
        ELSE
           RVAR = '    HIGH'
        END IF
        IF (DSTD .LE. DLOW) THEN
           DVAR = '     LOW'
        ELSE IF (DSTD .LT. DMOD) THEN
           DVAR = '    MOD.'
        ELSE
           DVAR = '    HIGH'
        END IF
C
C       OUTPUT RESULTS OF THIS PLOT.
        LINE = LINE + 6
        WRITE (IOUT, '('' SUMMARY TABLE'')')
        WRITE (IOUT, '(''    VECTOR AVERAGE '', F8.1, I8)') VRATE, IVDIR
        WRITE (IOUT, '(''    V/A DIVERGENCE '', F8.1)')     VADIV
        WRITE (IOUT, '(''    ARITHMETIC MEAN'', F8.1, I8)') RMEAN, IDMEAN
        WRITE (IOUT, '(''          STD. DEV.'', F8.1, I8)') RSTD,  IDSTD
        WRITE (IOUT, '(''          VARIANT  '', 2A)')       RVAR,  DVAR
        RETURN
        END
```

```
      FUNCTION CIRCLE (ANGA, ANGB)
C
C     THIS FUNCTION IS USED WHEN COMPARING  TWO ANGLES (CONSIDERING THE
C     CYCLIC NATURE OF A COMPASS).  IF 'ANGA' IS NOT WITHIN PLUS OR MI-
C     NUS 180 DEGREES OF 'ANGB', THEN 360 DEGREES IS SUBTRACTED OR ADD-
C     ED TO 'ANGA' TO PUT IT IN THAT RANGE.
C
C     ________________________________________________________________
      IF (ANGA .GE. ANGB + 180.0) THEN
         CIRCLE = ANGA - 360.0
      ELSE IF (ANGA .LT. ANGB - 180.0) THEN
         CIRCLE = ANGA + 360.0
      ELSE
         CIRCLE = ANGA
      END IF
      RETURN
      END



      FUNCTION LINED (NV)
C
C     THIS FUNCTION PREVENTS THE OUTPUT  OF A PLOT FROM BEING SPLIT BE-
C     TWEEN TWO PAGES WHENEVER FEASIBLE.
C
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
C
      DIMENSION MLINE(8)
C
      DATA MLINE /4, 15, 8, 17, 23, 33, 48, 69/
C
C     ________________________________________________________________
C     DETERMINE MAXIMUM POSSIBLE NUMBER OF LINES FOR THIS PLOT.
      IF (NV .LE. 8) THEN
         L = MLINE(NV)
      ELSE
         L = LINES
      END IF
C
C     ADD TO EXISTING PAGE IF POSSIBLE, OR BEGIN NEW PAGE.
      IF (LINE + L .LE. LINES) THEN
         LINED = LINE + 4
         WRITE (IOUT, '(//)')
      ELSE
         LINED = 1
         WRITE (IOUT, '(''1'')')
      END IF
      RETURN
      END
```

```
      SUBROUTINE SPLIT
C
C     HEADS NEW OUTPUT PAGE WHENEVER A PLOT IS SPLIT BETWEEN TWO PAGES.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
C
      CHARACTER FMT*33
C
      DATA FMT /'('' CONTINUATION OF PLOT #'', I1)'/
C
C     ------------------------------------------------------------
      LINE = 3
      WRITE (IOUT, '(''1'')')
      WRITE (FMT(30:30), '(I1)') INT(ALOG10(REAL(NS))) + 1
      WRITE (IOUT, FMT) NS
      WRITE (IOUT, '(23X, ''RATE  DIRECTION'')')
      RETURN
      END


      SUBROUTINE NOPLOT (VPT)
C
C     CALLED WHEN A PLOT HAS TOO FEW POINTS AND/OR NEGLIGIBLE AREA.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
C
      CHARACTER VPT*(*), FMT*45
C
      DATA FMT /'('' INSUFFICIENT DATA IN PLOT #'', I1, ''--'', A)'/
C
C     ------------------------------------------------------------
      LINE = LINED (1)
      WRITE (FMT(35:35), '(I1)') INT(ALOG10(REAL(NS))) + 1
      WRITE (IOUT, FMT) NS, VPT
      RETURN
      END
```

```
      SUBROUTINE CALCOM (SPT, SPX, SPY, SPP, MPT)
C
C     INITIALIZES THE CALCOMP PLOTTER AND PLOTS THE NORTH POINTER.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XLIM,YLIM
C
      CHARACTER *(*) SPT(NSL), SPP(NSL)
      DIMENSION SPX(NVL,NSL), SPY(NVL,NSL), MPT(NSL)
C
C     ----------------------------------------------------------------
C     INITIALIZE PLOTTER QUEUE.
      IF (ITYPE .EQ. 0 .OR. ITYPE .EQ. 5 .OR. ITYPE .EQ. 10) THEN
         XAXIS = 10.0
      ELSE IF (ITYPE .EQ. 1 .OR. ITYPE .EQ. 6) THEN
         XAXIS = 30.0
      ELSE
         STOP 'ERROR IN PLOTTER QUEUE TYPE'
      END IF
      CALL PLOTS (IBUF, 257, ITYPE)
C
C     PLOT COMPASS NORTH SYMBOL.
      CALL NEWPEN (1)
      CALL SYMBOL (-0.09, XAXIS - 0.32, 0.28, 'N',  0.0,  1)
      CALL SYMBOL (-0.07, XAXIS - 0.32, 0.28, 'N',  0.0,  1)
      CALL SYMBOL (-0.01, XAXIS - 0.59, 0.28,  13,  0.0, -1)
      CALL SYMBOL ( 0.01, XAXIS - 0.59, 0.28,  13,  0.0, -1)
      CALL SYMBOL (-0.01, XAXIS - 0.83, 0.28,  11, 45.0, -1)
      CALL SYMBOL ( 0.01, XAXIS - 0.83, 0.28,  11, 45.0, -1)
      CALL SYMBOL (-0.01, XAXIS - 0.85, 0.28,  11, 45.0, -1)
      CALL SYMBOL ( 0.01, XAXIS - 0.85, 0.28,  11, 45.0, -1)
      CALL NEWPEN (2)
      CALL SYMBOL (-0.08, XAXIS - 0.32, 0.28, 'N',  0.0,  1)
      CALL SYMBOL ( 0.00, XAXIS - 0.58, 0.28,  13,  0.0, -1)
      CALL SYMBOL ( 0.00, XAXIS - 0.84, 0.28,  11, 45.0, -1)
C
C     LAY OUT THE PLOTS.
      CALL ORIGIN (SPT, SPX, SPY, SPP, MPT)
C
C     CLOSE PLOTTER QUEUE.
      CALL PLOT (BORDL, 0.0, 999)
      RETURN
      END
```

```
      SUBROUTINE ORIGIN (SPT, SPX, SPY, SPP, MPT)
C
C     ORIENTS ALL POINTS FROM A COMMON ORIGIN.
C
C     PROGRAM HANDLES UP TO 'NGL' TIMERS, A MACHINE LIMITATION EQUAL TO
C     ITS TOTAL NUMBER OF INDIVIDUAL ALPHANUMERIC CHARACTERS OTHER THAN
C     THE '#' (RESERVED FOR SEPARATING PLOTS ON FILE 'QDATA').
      PARAMETER (NGL=61)
C
C     ________________________________________________________________
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGLIM
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /PLOTS/ NP,PWGT, PRSUM,PRSUM2, PDSUM,PDSUM2, PXSUM,PYSUM
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XLIM,YLIM
C
      CHARACTER *(*) SPT(NSL), SPP(NSL), GPT*(NGL), GPP*(NGL+1)
      DIMENSION SPX(NVL,NSL), SPY(NVL,NSL), MPT(NSL)
      DIMENSION GPX(NGL), GPY(NGL), GDIR(NGL), NPP(NGL)
      CHARACTER TOTAL*7, FMT1*39, FMT2*38, FMT3*38
C
      DATA TOTAL /' TOTAL '/, IDIGIT /0/, TAREA /0.0/
      DATA FMT1  /'(A, ''AREA: '', I1, '' SQUARE UNITS--'', A)'/
      DATA FMT2  /'('' TOTAL AREA: '', I1, '' SQUARE UNITS'')'/
      DATA FMT3  /'('' PLOT AREAS: '', I1, '' SQUARE UNITS'')'/
C
C     ________________________________________________________________
C     CONVERT PARAMETER LIMIT TO VARIABLE.
      NGLIM = NGL
C
C     DETERMINE NUMBER OF DIGITS IN AREA CALCULATION OUTPUT.
      NPWGT = NINT(PWGT)
      IF (NPWGT .GT. 0 .AND. NPWGT .LT. 1.0E9)
     +   IDIGIT = MIN1(ALOG10(REAL(NPWGT)), 7.0) + 2
C
C     INITIALIZE PLOT SELECTION MATRIX.
      MPS = NS
      DO 10 J=1,MPS
   10 MPT(J) = J
C
C     ENTER FIRST PLOT INTO GRAPHIC MATRIX.
   20 XLIM = SPX(1,MPT(1))
      YLIM = SPY(1,MPT(1))
      DO 21 I=1,NVL
      IF (SPT(MPT(1))(I:I) .EQ. ' ') GO TO 30
   21 CALL MATRIX (I,MPT(1),0.0,0.0, SPT,SPX,SPY, GPT,GPX,GPY)
C
C     REORIENT REMAINING PLOTS TO ORIGIN OF FIRST PLOT.
   30 DO 31 I=2,MPS
      J = I
   31 CALL ORIENT (J, SPT, SPX, SPY, MPT, GPT, GPX, GPY)
```

```
C
C     RERUN PLOTS THAT DIDN'T HAVE A MATCH ON THE FIRST PASS.
   40 IF (MP .GT. 0) THEN
         IF (MP .NE. MPS) THEN
            MPS = MP
            MP  = 0
            DO 41 I=1,MPS
            J   = I
   41       CALL ORIENT (J, SPT, SPX, SPY, MPT, GPT, GPX, GPY)
            GO TO 40
         ELSE
            TOTAL = ' '
         END IF
      END IF
C
C     MOVE ALL POINTS TO THE FIRST QUADRANT.
      GPX(1) = GPX(1) - XLIM
      GPY(1) = GPY(1) - YLIM
      XMAX   = GPX(1)
      YMAX   = GPY(1)
      DO 50 K=2,NG
      GPX(K) = GPX(K) - XLIM
      GPY(K) = GPY(K) - YLIM
      XMAX   = AMAX1(GPX(K), XMAX)
   50 YMAX   = AMAX1(GPY(K), YMAX)
      XLIM   = XMAX
      YLIM   = YMAX
C
C     CALCULATE THE AREA AND GRAPH THE CONVEX POLYGON JUST FORMED.
      IF (NP .GE. 2) THEN
         AREAS = AREA (NG, GPT, GPX, GPY, GPP, GDIR, NPP)
         NAREA = NINT(AREAS)
         IF (NAREA .GT. 0 .AND. NAREA .LT. 1.0E9) THEN
            IDIGIT = MAX0(IDIGIT, INT(ALOG10(REAL(NAREA))) + 1)
            WRITE (FMT1(16:16), '(I1)') IDIGIT
            WRITE (IOUT, FMT1) TOTAL, NAREA, GPP
         END IF
         TAREA = TAREA + AREAS
      END IF
      CALL GRAFIC (SPP, GPT, GPX, GPY)
C
C     REPEAT THE ABOVE FOR ANY DISJOINT PLOTS.
      IF (MP .GT. 0) THEN
         MP = 0
         NG = 0
         GO TO 20
      END IF
```

```
C
C     IF DISJOINT PLOTS, WRITE THE TOTAL AREA TO THE OUTPUT FILE.
      IF (NP .GE. 2) THEN
         IF (TOTAL .EQ. ' ') THEN
            NAREA = NINT(TAREA)
            IF (NAREA .LT. 1.0E9) THEN
               IDIGIT = MAX0(IDIGIT, INT(ALOG10(REAL(NAREA))) + 1)
               WRITE (FMT2(20:20), '(I1)') IDIGIT
               WRITE (IOUT, FMT2) NAREA
            END IF
         END IF
C
C        OUTPUT THE SUM OF THE INDIVIDUAL PLOT AREAS.
         IF (NPWGT .LT. 1.0E9) THEN
            WRITE (FMT3(20:20), '(I1)') IDIGIT
            WRITE (IOUT, '('' SUM OF ALL'')')
            WRITE (IOUT, FMT3) NPWGT
         END IF
C
C        IF BOTH AREA CALCULATIONS WERE OUTPUT, NOTE DISCREPANCIES.
         IF (NAREA .LT. 1.0E9 .AND. NPWGT .LT. 1.0E9) THEN
            IF (TAREA .GE. 1.01*PWGT) THEN
               WRITE (IOUT, '(/
     +         '' NOTE:  A PORTION OF THE ''''TOTAL AREA'''' IS''/
     +         '' NOT CONTAINED WITHIN ANY PLOT,  AND NOT''/
     +         '' INCLUDED IN THE FOLLOWING STATISTICS.  '')')
               CALL NOTE1
            ELSE IF (TAREA .LE. 0.99*PWGT) THEN
               WRITE (IOUT, '(/
     +         '' NOTE:  THE INPUT  FILE  INCLUDED  PLOTS''/
     +         '' THAT OVERLAPPED  EACH OTHER.   BOTH THE''/
     +         '' ''''SUM OF ALL PLOT AREAS'''' AND THE FOLLOW-''/
     +         '' ING STATISTICS,  WHICH ARE  WEIGHTED BY''/
     +         '' AREA,  REFLECT  ADDED  EMPHASIS  IN THE''/
     +         '' OVERLAPPED AREAS.                      '')')
               CALL NOTE3
            ELSE IF (NAREA .NE. NPWGT) THEN
               WRITE (IOUT, '(/
     +         '' NOTE:  THE  ''''TOTAL AREA''''  DIFFERS  FROM''/
     +         '' THE ''''SUM  OF ALL  PLOT  AREAS''''  BY LESS''/
     +         '' THAN 1 PERCENT.  THIS SMALL DISCREPANCY''/
     +         '' MAY BE DUE  TO A PORTION  OF THE ''''TOTAL''/
     +         '' AREA'''' NOT  CONTAINED  WITHIN  ANY PLOT,''/
     +         '' OVERLAPPING  PLOTS,  OR  POSSIBLY  BOTH''/
     +         '' REASONS.                               '')')
            END IF
```

```
C
C        BOTH AREA CALCULATIONS WERE NOT OUTPUT.
         ELSE
            IF (NPWGT .LT. 1.0E9) THEN
               CALL NOTE1
            ELSE
               IF (NAREA .LT. 1.0E9) CALL NOTE3
               CALL NOTE2
            END IF
         END IF
      END IF
      RETURN
      END


      SUBROUTINE ORIENT (J, SPT, SPX, SPY, MPT, GPT, GPX, GPY)
C
C     REORIENTS THE JTH PLOT TO THE ORIGIN OF FIRST PLOT.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XLIM,YLIM
C
      CHARACTER *(*) SPT(NSL), GPT
      DIMENSION SPX(NVL,NSL),SPY(NVL,NSL),MPT(NSL), GPX(NGL),GPY(NGL)
      LOGICAL   MISS
C     ____________________________________________________________________
C
C     FIND MATCH BETWEEN POINT IN JTH PLOT AND POINT ALREADY IN MATRIX.
      MISS = .TRUE.
      J    = MPT(J)
      DO 13 I=1,NVL
      IF (SPT(J)(I:I) .EQ. ' ') GO TO 20
      DO 13 K=1,NG
      IF (SPT(J)(I:I) .EQ. GPT(K:K)) THEN
         MISS = .FALSE.
C
C        FIND COORDINATE DIFFERENCE BETWEEN MATCHED POINTS.
         X = GPX(K) - SPX(I,J)
         Y = GPY(K) - SPY(I,J)
C
C        ADD UNMATCHED POINTS TO GRAPHIC MATRIX.
         DO 10 II=1,I-1
   10    CALL MATRIX (II,J,X,Y, SPT,SPX,SPY, GPT,GPX,GPY)
         DO 12 II=I+1,NVL
         IF (SPT(J)(II:II) .EQ. ' ') RETURN
         DO 11 KK=1,NG
   11    IF (SPT(J)(II:II) .EQ. GPT(KK:KK)) GO TO 12
         CALL MATRIX (II,J,X,Y, SPT,SPX,SPY, GPT,GPX,GPY)
   12    CONTINUE
         RETURN
      END IF
   13 CONTINUE
```

```
C
C     MATCH NOT FOUND.
   20 IF (MISS) THEN
         MP = MP + 1
         MPT(MP) = J
      END IF
      RETURN
      END
```

```
      SUBROUTINE MATRIX (I,J,X,Y, SPT,SPX,SPY, GPT,GPX,GPY)
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XMIN,YMIN
C
      CHARACTER *(*) SPT(NSL), GPT
      DIMENSION SPX(NVL,NSL), SPY(NVL,NSL), GPX(NGL), GPY(NGL)
C     ------------------------------------------------------------------
C
C     ADD THE SELECTED POINT TO THE GRAPHIC MATRIX.
      NG        = NG + 1
      GPT(NG:)  = SPT(J)(I:I)
      GPX(NG)   = SPX(I,J) + X
      GPY(NG)   = SPY(I,J) + Y
      XMIN      = AMIN1(XMIN, GPX(NG))
      YMIN      = AMIN1(YMIN, GPY(NG))
      RETURN
      END
```

```
      SUBROUTINE GRAFIC (SPP, GPT, GPX, GPY)
C
C     PLOTS THE FIRE USING CALCOMP COMMANDS.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XMAX,YMAX
C
      CHARACTER *(*) SPP(NSL), GPT, PN*1, PI*1
      DIMENSION GPX(NGL), GPY(NGL)
C     ------------------------------------------------------------------
```

```
C
C     DETERMINE SCALING FACTOR AND LENGTH OF Y-AXIS.
      FCT   = XAXIS/XMAX
      YAXIS = YMAX*FCT
      BORDR = BORDL + YAXIS + 1.0
      IF (BORDR .GT. 29.0) CALL PLIMIT (BORDR + 1.0)
C
C     SCALE THE FIRE TO THE PLOTTER COORDINATES.
      DO 10 K=1,NG
      GPX(K) = GPX(K)*FCT
   10 GPY(K) = GPY(K)*FCT + BORDL
C
C     DETERMINE NUMBER OF INCREMENTS FOR PLOT SCALE.
      CALL NEWPEN (1)
      PS    = 10.0
      SCALE = 10.0*FCT + BORDL
      IF (SCALE .GT. BORDR) THEN
         DO 20 PS=5.0,1.0,-1.0
         SCALE = PS*FCT + BORDL
   20    IF (SCALE .LE. BORDR) GO TO 30
         GO TO 40
      END IF
C
C     PLACE THE SCALE ALONG THE EDGE OF THE DRAWING.
   30 CENTER = 0.5*PS*FCT + BORDL
      CALL SYMBOL (CENTER - 0.32, -0.37, 0.14, 'SCALE', 0.0, 5)
      Y = BORDL
      CALL PLOT (Y, -0.37, 3)
      CALL PLOT (Y, -0.51, 2)
      DO 31 SI=1.0,PS
      CALL PLOT (Y, -0.44, 3)
      Y = Y + FCT
      CALL PLOT (Y, -0.44, 2)
   31 CALL PLOT (Y, -0.51, 2)
      CALL PLOT (Y, -0.37, 2)
C
C     LABEL THE SCALE.
      IF (PS .GE. 10.0) THEN
         CALL SYMBOL (CENTER - 0.53, -0.72, 0.14, '10 UNITS', 0.0, 8)
      ELSE IF (PS .GT. 1.0) THEN
         CALL NUMBER (CENTER - 0.46, -0.72, 0.14,     PS,     0.0,-1)
         CALL SYMBOL (CENTER - 0.32, -0.72, 0.14,  ' UNITS',   0.0, 6)
      ELSE
         CALL SYMBOL (CENTER - 0.39, -0.72, 0.14,  '1 UNIT',   0.0, 6)
      END IF
```

```
C
C     MOVE PEN TO THE NORTHERNMOST POINT OF EACH PLOT.
   40 DO 53 J=1,NS
      PN = SPP(J)(1:1)
      DO 41 K=1,NG
      IF (PN .EQ. GPT(K:K)) THEN
         XPN = GPX(K)
         YPN = GPY(K)
         CALL PLOT (YPN, XPN, 3)
         GO TO 50
      END IF
   41 CONTINUE
C
C     CONNECT THE PLOT'S PERIMETER POINTS.
   50 DO 52 I=2,NVL+1
      PI = SPP(J)(I:I)
      IF (PI .NE. PN) THEN
         DO 51 K=1,NG
         IF (PI .EQ. GPT(K:K)) THEN
            CALL PLOT (GPY(K), GPX(K), 2)
            GO TO 52
         END IF
   51    CONTINUE
         GO TO 53
      ELSE
         CALL PLOT (YPN, XPN, 2)
         GO TO 53
      END IF
   52 CONTINUE
   53 CONTINUE
C
C     PLOT THE TIMER POINTS.
      CALL NEWPEN (3)
      DO 60 K=1,NG
      X = GPX(K) - 0.08
      Y = GPY(K) - 0.04
   60 CALL SYMBOL (Y, X, 0.14, GPT(K:K), 0.0, 1)
C
C     PLOT THE SPREAD VECTORS.
      CALL NEWPEN (2)
      CALL VECTOR (GPT, GPX, GPY)
C
C     DETERMINE LENGTH OF GRAPH PAPER USED.
      BORDL = BORDR + 1.0
      RETURN
      END
```

```
      SUBROUTINE VECTOR (GPT, GPX, GPY)
C
C     PLOTS THE SPREAD VECTORS USING CALCOMP COMMANDS.
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XLIM,YLIM
C
      CHARACTER GPT*(*),  TRI(3)*1
      DIMENSION GPX(NGL), GPY(NGL)
C
C     DEGREES PER RADIAN.
      DATA RAD /57.2957795131/
C     ------------------------------------------------------------------
C
C     REWIND THE VECTOR FILE AND OPEN A SCRATCH FILE.
      REWIND (IVEC)
      OPEN   (JVEC, STATUS='SCRATCH')
      REWIND (JVEC)
C
C     INITIALIZE THREE LONGEST VECTORS.
      VMAX1 = 0.0
      VMAX2 = 0.0
      VMAX3 = 0.0
C
C     READ A VECTOR AND DETERMINE STARTING COORDINATES.
   10 READ (IVEC, '(3A, 3F9.2)', END=20) TRI, RATE, RCOS, RSIN
      X1 = 0.0
      Y1 = 0.0
      DO 12 I=1,3
      DO 11 K=1,NG
      IF (TRI(I) .EQ. GPT(K:K)) THEN
         X1 = X1 + GPX(K)
         Y1 = Y1 + GPY(K)
         GO TO 12
      END IF
   11 CONTINUE
      GO TO 10
   12 CONTINUE
      X1 = X1/3.0
      Y1 = Y1/3.0
C
C     DETERMINE PER MINUTE CHANGE IN VECTOR.
      DX = RCOS*FCT
      DY = RSIN*FCT
      DV = RATE*FCT
      WRITE (JVEC, '(5F9.2)') X1, Y1, DX, DY, DV
```

```
C
C     DETERMINE LENGTH OF THREE LONGEST VECTORS.
      IF (DV .GT. VMAX3) THEN
         IF (DV .GT. VMAX2) THEN
            VMAX3 = VMAX2
            IF (DV .GT. VMAX1) THEN
               VMAX2 = VMAX1
               VMAX1 = DV
            ELSE
               VMAX2 = DV
            END IF
         ELSE
            VMAX3 = DV
         END IF
      END IF
      GO TO 10
C
C     DETERMINE BEST TIME FRAME FOR DRAWING VECTOR.
   20 ENDFILE (JVEC)
      CALL SYMBOL (BORDL, -1.0, 0.14, 'VECTORS IN UNITS/', 0.0, 17)
      FRAME = BORDL + 2.38
      IF (VMAX3 .LE. 0.0) VMAX3 = VMAX1
      M = 1.5/VMAX3
C
C     USE TIME INTERVALS OF LESS THAN A MINUTE.
      IF (M .EQ. 0) THEN
         DS = INT(VMAX3/1.5) + 1
         IF (DS .GE. 5.0) DS = 6.0
         DSEC = 60.0/DS
         DMIN =  1.0/DS
         CALL NUMBER (FRAME,        -1.0, 0.14,     DSEC,     0.0,-1)
         CALL SYMBOL (FRAME + 0.28, -1.0, 0.14, ' SECONDS', 0.0, 8)
C
C     USE TIME INTERVALS OF ONE MINUTE.
      ELSE IF (M .EQ. 1) THEN
         DMIN = 1.0
         CALL SYMBOL (FRAME,        -1.0, 0.14,  'MINUTE',  0.0, 6)
C
C     USE TIME INTERVALS OF GREATER THAN A MINUTE.
      ELSE IF (M .LE. 9) THEN
         DMIN = AMIN0(M, 6)
         CALL NUMBER (FRAME,        -1.0, 0.14,     DMIN,     0.0,-1)
         CALL SYMBOL (FRAME + 0.14, -1.0, 0.14, ' MINUTES', 0.0, 8)
      ELSE IF (M .LE. 39) THEN
         DMIN = 10*(M/10)
         CALL NUMBER (FRAME,        -1.0, 0.14,     DMIN,     0.0,-1)
         CALL SYMBOL (FRAME + 0.28, -1.0, 0.14, ' MINUTES', 0.0, 8)
      ELSE
         DMIN = 60.0
         CALL SYMBOL (FRAME,        -1.0, 0.14,   'HOUR',   0.0, 4)
      END IF
```

```
C
C     LOCATE PLOTTER NORTH AND EAST BOUNDARIES.
      XBND = XAXIS + 1.5
      YBND = BORDL + YAXIS + 1.5
C
C     DETERMINE ENDING COORDINATES AND PLOT VECTOR.
      REWIND (JVEC)
   30 READ   (JVEC, '(5F9.2)', END=40) X1, Y1, DX, DY, DV
      X2 = X1 + DX*DMIN
      Y2 = Y1 + DY*DMIN
      HEAD = AMIN1(0.07, 0.67*DV*DMIN)
C
C     MAKE SURE VECTOR HEAD IS WITHIN BOUNDS OF PLOTTER.
      IF (X2 .LT. -1.3 .OR. X2 .GT. XBND .OR.
     +    Y2 .LT. -0.5 .OR. Y2 .GT. YBND) THEN
         CALL PLOT (Y1, X1, 3)
C
C        PLOT VECTOR SHAFT AND SYMBOL INDICATING SHORTENED VECTOR.
         IF (Y2 .NE. Y1) THEN
            SLOPE = (X2 - X1)/(Y2 - Y1)
            X2    = AMAX1(-1.3, AMIN1(X2,                    XBND))
            Y2    = AMAX1(-0.5, AMIN1(Y1 + (X2 - X1)/SLOPE, YBND))
            X2    = X1 + (Y2 - Y1)*SLOPE
            S     = SIGN(1.0/SQRT(SLOPE*SLOPE + 1.0), Y2 - Y1)
            S1    = 1.085*S
            S2    = 1.000*S
            S3    = 0.915*S
            ANGLE = RAD*ATAN(SLOPE) + 45.0
            CALL PLOT   (Y2 - S1, X2 - S1*SLOPE, 2)
            CALL SYMBOL (Y2 - S2, X2 - S2*SLOPE, 0.12, 8, ANGLE, -1)
            Y1    = Y2 - S3
            X1    = X2 - S3*SLOPE
C
C        SPECIAL CASE:  VECTOR POINTS DIRECTLY NORTH OR SOUTH.
         ELSE
            X2 = AMAX1(-1.3, AMIN1(X2, XBND))
            S  = SIGN(1.0, X2)
            CALL PLOT   (Y2, X2 - 1.085*S, 2)
            CALL SYMBOL (Y2, X2 - 1.000*S, 0.12, 8, -45.0, -1)
            X1 = X2 - 0.915*S
         END IF
      END IF
C
C     PLOT VECTOR.
      CALL AROWHD (Y1, X1, Y2, X2, HEAD, 0.5*HEAD, 2, 2)
      GO TO 30
C
C     ALL VECTORS READ AND PLOTTED.
   40 CLOSE (JVEC)
      RETURN
      END
```

```
      SUBROUTINE NOTE1
C
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
C     ------------------------------------------------------------------
C
      WRITE (IOUT, '(/
     +'' NOTE:  THE INPUT FILE MAY HAVE INCLUDED''/
     +'' PLOTS THAT  OVERLAPPED  EACH OTHER.  IF''/
     +'' SO, BOTH  THE ''''SUM  OF ALL  PLOT AREAS''''''/
     +'' AND THE FOLLOWING STATISTICS, WHICH ARE''/
     +'' WEIGHTED BY AREA,  REFLECT ADDED EMPHA-''/
     +'' SIS IN THE OVERLAPPED AREAS.           '')')
      RETURN
      END



      SUBROUTINE NOTE2
C
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
C     ------------------------------------------------------------------
C
      WRITE (IOUT, '(/
     +'' NOTE:  IF THE  INPUT FILE  INCLUDED ANY''/
     +'' PLOTS THAT  OVERLAPPED  EACH OTHER, THE''/
     +'' FOLLOWING STATISTICS, WHICH ARE WEIGHT-''/
     +'' ED BY AREA,  REFLECT ADDED  EMPHASIS IN''/
     +'' THE OVERLAPPED AREAS.                  '')')
      RETURN
      END



      SUBROUTINE NOTE3
C
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
C     ------------------------------------------------------------------
C
      WRITE (IOUT, '(/
     +'' NOTE:  A PORTION  OF  THE  ''''TOTAL AREA''''''/
     +'' MAY NOT BE  CONTAINED WITHIN  ANY PLOT.''/
     +'' IF SO, THIS PORTION  IS NOT INCLUDED IN''/
     +'' THE FOLLOWING STATISTICS.              '')')
      RETURN
      END
```

```
      BLOCK DATA
C
      COMMON /LIMIT/ NV, NVL,  NS, NSL,  NG, NGL
      COMMON /FILES/ INPT, IOUT, IVEC, JVEC, LINE, LINES
      COMMON /CNTRL/ PWT, GRAPH, AMIN
      COMMON /PLOTS/ NP,PWGT, PRSUM,PRSUM2, PDSUM,PDSUM2, PXSUM,PYSUM
      COMMON /MEANS/ RMEAN, DMEAN, XMEAN, YMEAN, RLOW, RMOD, DLOW, DMOD
      COMMON /GRPHC/ IBUF(257),ITYPE,XAXIS,YAXIS,BORDL,FCT,MP,XLIM,YLIM
C
      LOGICAL GRAPH
C
C     ASSOCIATE INPUT, OUTPUT, & SCRATCH FILE NAMES WITH UNIT NUMBERS.
      DATA INPT, IOUT, IVEC, JVEC /6, 7, 8, 9/
C
C     MAXIMUM LINES OF OUTPUT ALLOWED PER PAGE.
      DATA LINES /60/
C
C     BECAUSE OF THE ERROR INHERENT IN  FIELD MEASUREMENTS, THE PROGRAM
C     OMITS FROM CONSIDERATION TRIANGLES  HAVING AN ANGLE LESS THAN 0.2
C     RADIANS (ABOUT 11.46 DEGREES).
      DATA AMIN /0.2/
C
C     MAXIMUM VALUES  ALLOWED FOR THE  LOW AND THE  MODERATE CLASSES OF
C     THE COEFFICIENT OF VARIATION OF THE RATE, AND THE STANDARD DEVIA-
C     TION OF THE DIRECTION.
      DATA RLOW /0.2/, RMOD /0.5/, DLOW /10.0/, DMOD /20.0/
C
C     INITIALIZE GRAPHICS CAPABILITY.
      DATA GRAPH /.TRUE./
C
C     INITIALIZE CALCOMP  PLOTTER QUEUE (IGNORE IF  GRAPH  SET .FALSE.).
C     AN 'ITYPE' OF 0 OR 5 SELECTS 15" PAPER; 1 OR 6 SELECTS 35" PAPER.
C     AN 'ITYPE' OF 0 OR 1 SELECTS STANDARD BALLPOINT PENS; 5 OR 6 DOES
C     THE SAME  EXCEPT  BLACK  LIQUID  INK SIZE  4 FOR  PEN #1.  OTHER
C     CHOICES FOR 'ITYPE' ARE AVAILABLE (SEE CALCOMP MANUAL).
      DATA ITYPE /0/
C
C     INITIALIZE (THE USER SHOULD NOT ALTER THE FOLLOWING VARIABLES).
      DATA NS, NG, NP, MP /4*0/,       BORDL /1.0/
      DATA PWGT, PRSUM, PRSUM2, PDSUM, PDSUM2, PXSUM, PYSUM /7*0.0/
      END
```

APPENDIX 2

SPREAD RATE PROGRAM VARIABLE LIST

COMMON BLOCK VARIABLES
(Listed in the order in which they appear in each COMMON Statement)

COMMON /LIMIT/

NV	Number of points in current plot.
NVL	The upper limit to the number of points allowed per plot (also refered to as NVLIM). This limit can be adjusted by altering the PARAMETER statement in subprogram QUEUE.
NS	Number of plots in current run of program.
NSL	The upper limit to the number of plots allowed (also referred to as NSLIM). This limit can be adjusted by altering the PARAMETER statement in subprogram QUEUE.
NG	Number of timers in current run of program.
NGL	The upper limit to the number of timers allowed (also referred to as NGLIM). This is a machine limitation equal to its total number of individual alphanumeric characters other than the '#' (reserved for separating plots on the input file). This limit can be adjusted by altering the PARAMETER statement in subprogram ORIGIN.

COMMON /FILES/

INPT	Unit number of input file QDATA.
IOUT	Unit number of output file QRATE.
IVEC	Unit number of a scratch file used for storing each vector for subsequent graphing.
JVEC	Unit number of a scratch file used for grouping vectors (if plots are disjoint) and scaling them to the appropriate size for graphics.
LINE	Current line number on the current output page.
LINES	Maximum number of lines of output allowed per page. Can be adjusted by the user in subprogram BLOCK DATA.

COMMON /CNTRL/

PWT	Area of current plot (square units).
GRAPH	Logical variable set .TRUE. by the user in subprogram BLOCK DATA if graphics are available and desired.
AMIN	Minimum size for each angle that a triangle must possess to be allowed entry into the analysis (radians). Can be adjusted by the user in subprogram BLOCK DATA.

COMMON /TRIAN/

NT	Number of triangles in current plot.
TWGT	Sum of the areas of the NT triangles in the current plot.
TRSUM	Sum of the rates of spread, weighted by each triangle's area, over the NT triangles in the current plot.
TRSUM2	Sum of squares of the rates of spread, weighted by each triangle's area, over the NT triangles in the current plot.
TDSUM	Sum of the directions of spread, weighted by each triangle's area, over the NT triangles in the current plot.
TDSUM2	Sum of squares of the directions of spread, weighted by each triangle's area, over the NT triangles in the current plot.
TXSUM	Sum of the x-axis components of the rate of spread vectors, weighted by each triangle's area, over the NT triangles in the current plot.
TYSUM	Sum of the y-axis components of the rate of spread vectors, weighted by each triangle's area, over the NT triangles in the current plot.

COMMON /PLOTS/

NP	Number of plots with at least one acceptable triangle.
PWGT	Sum of the areas of the NP plots.
PRSUM	Sum of the rates of spread, weighted by each plot's area, over the NP plots.
PRSUM2	Sum of squares of the rates of spread, weighted by each plot's area, over the NP plots.
PDSUM	Sum of the directions of spread, weighted by each plot's area, over the NP plots.
PDSUM2	Sum of squares of the directions of spread, weighted by each plot's area, over the NP plots.
PXSUM	Sum of the x-axis components of the rate of spread vectors, weighted by each plot's area, over the NP plots.
PYSUM	Sum of the y-axis components of the rate of spread vectors, weighted by each plot's area, over the NP plots.

COMMON /MEANS/

RMEAN	Mean of the rates of spread of the NT triangles in the current plot or of the NP plots in current run (units/minute).
DMEAN	Mean of the directions of spread of the NT triangles in the current plot or of the NP plots in current run (degrees azimuth).
XMEAN	Mean of the x-axis components of the rate of spread vectors of the NT triangles in the current plot or of the NP plots in current run (units/minute).
YMEAN	Mean of the y-axis components of the rate of spread vectors of the NT triangles in the current plot or of the NP plots in current run (units/minute).
RLOW	Maximum value of the low variability class of the coefficient of variation of the rate of spread.

RMOD Maximum value of the moderate variability class of the coefficient of variation of the rate of spread.

DLOW Maximum value of the low variability class of the standard deviation of the direction of spread.

DMOD Maximum value of the moderate variability class of the standard deviation of the direction of spread.

COMMON /GRPHC/

IBUF The name of a storage area required by the CALCOMP plotter for accumulating plotter commands and buffering output.

ITYPE The type of CALCOMP plotter queue--choices include type and size of pens and type and size of paper. Can be selected by the user in subprogram BLOCK DATA.

XAXIS Maximum length of x-axis (North-South line) available on the plotter based on width of paper selected by ITYPE (inches).

YAXIS Length of the y-axis (East-West line) on the plotter necessary to produce a drawing proportional with the XAXIS (inches).

BORDL The location of the left border of the current graphic from the plotter origin (inches).

FCT Scaling factor relating distance units to plotter inches.

MP In the case of disjoint plots, the number of points remaining disconnected from current plotter diagram (to be plotted as separate diagram(s) further to the right).

XLIM The maximum or minimum value of the x-coordinates of the points to be plotted, depending on what is needed at various stages of the process (referred to as XMAX or XMIN in subprograms where XLIM is strictly one or the other).

YLIM The maximum or minimum value of the y-coordinates of the points to be plotted, depending on what is needed at various stages of the process (referred to as YMAX or YMIN in subprograms where YLIM is strictly one or the other).

GLOBAL ARRAYS PASSED AS PARAMETERS
(Allows adjustable dimensions--Listed alphabetically)

GDIR(NGL) Passed as a parameter solely to maintain adjustable dimension (see DIR under FUNCTION AREA).

GPP Character*(NGL+1) array where the (at most) NG+1 perimeter points (beginning with northernmost point and proceeding clockwise around area back to starting point) of the current area being plotted are stored.

GPT Character*(NGL) array where the NG points of the current area being plotted are stored.

GPX(NGL) Array for storing all the x-coordinates of the NG points of the current area being plotted.

GPY(NGL) Array for storing all the y-coordinates of the NG points of the current area being plotted.

MPT(NSL) Array for storing the plot numbers of those plots that are disjoint from current (and previous) area being plotted.

NPP(NVL)	Passed as a parameter solely to maintain adjustable dimension (see NPP under FUNCTION AREA).
NPP(NGL)	Passed as a parameter solely to maintain adjustable dimension (see NPP under FUNCTION AREA).
SPP(NSL)	Character*(NVL+1) array where the (at most) NV+1 perimeter points (beginning with northernmost point and proceeding clockwise around area back to starting point) of the NS plots are stored for subsequent graphics.
SPT(NSL)	Character*(NVL) array where the NV plot points of each of the NS plots are stored for subsequent graphics.
SPX(NVL,NSL)	Array for storing all the x-coordinates of the NV plot points of each of the NS plots for subsequent graphics.
SPY(NVL,NSL)	Array for storing all the y-coordinates of the NV plot points of each of the NS plots for subsequent graphics.
TIME(NVL)	The time fire reached each of the NV points in the current plot (minutes).
VDIR(NVL)	Passed as a parameter solely to maintain adjustable dimension (see DIR under FUNCTION AREA).
VPP	Character*(NVL+1) array where the (at most) NV+1 perimeter points (beginning with northernmost point and proceeding clockwise around area back to starting point) of the current plot are stored.
VPT	Character*(NVL) array where the NV plot points of the current plot are stored.
VPX(NVL)	Array for storing all the x-coordinates of the NV plot points of the current plot.
VPY(NVL)	Array for storing all the y-coordinates of the NV plot points of the current plot.

LOCAL VARIABLES
(Listed alphabetically by subroutine)

SUBROUTINE QUEUE

COMP	Compass direction of each point in current plot as viewed from the plot's marker point (input in degrees, immediately converted to radians).
DIST	Distance of each point in current plot from the plot's marker point (units of the user's choice).
MIN	Number of whole minutes on timer for current point.
MORE	Logical variable indicating whether or not there are any more plots in the input file (true/false).
NPWGT	Integer representation of PWGT (square units).
NPWT	Integer representation of PWT (square units).
PT	Alphanumeric character identifying each point in current plot.
RAD	Degrees per radian.
SEC	Number of seconds beyond MIN on timer for current point.

FUNCTION AREA

AB	Side of generic triangle ABX (units).
ABX	Each set of two consecutive perimeter points AB in conjunction with the northernmost point X of an area form a series of triangles labeled generically ABX. The sum of the areas of these triangles provides the total area figure.
AX	Side of generic triangle ABX (units).
BX	Side of generic triangle ABX (units).
DIR	An adjustable scratch array for storing compass directions of all points of an area relative to each successive perimeter point as the routine proceeds clockwise around the area.
MX	The index of each successive perimeter point.
N	Generic for the number of points in area (NV or NG).
NPP	An adjustable scratch array for storing the indexes of the perimeter points of an area beginning with the northernmost point and proceeding clockwise.
NX	The index of the northernmost perimeter point.
PP	Generic for the perimeter point array (VPP or GPP).
PT	Generic for the array of all points in the area (VPT or GPT).
PX	Generic for the x-coordinate array (VPX or GPX).
PY	Generic for the y-coordinate array (VPY or GPY).
RAD	Degrees per radian.
S	Intermediate variable used in the area of ABX calculation.

SUBROUTINE TRIGON

A	Vertex of current triangle that the fire reached first (radians).
AB	Length of current triangle's side between A and B (units).
AC	Length of current triangle's side between A and C (units).
ACSINA	Intermediate variable--side AC multiplied by sin(A).
B	Vertex of current triangle that the fire reached second (radians).
BAN	Compass direction of B as viewed from A (degrees azimuth).
BC	Length of current triangle's side between B and C (units).
C	Vertex of current triangle that the fire reached last (radians).
CAN	Compass direction of C as viewed from A (degrees azimuth).
RAD	Degrees per radian.
RATE	Rate of spread across current triangle (units/minute).
RCOS	The x-axis component of the rate of spread vector across the current triangle (units/minute).
RSIN	The y-axis component of the rate of spread vector across the current triangle (units/minute).
THETA	Angle of spread--first used relative to the current triangle, later added to or subtracted from BAN making it relative to north (degrees).
TIMJI	Intermediate variable--time at B minus time at A (minutes).
TIMKI	Intermediate variable--time at C minus time at A (minutes).

TRI	Three letter designation for current triangle listing its vertices in chronological order.
TWT	The area of the current triangle (square units).

SUBROUTINE STATS

COVAR	Coefficient of variation of the rate of spread.
DSTD	Standard deviation of the direction of spread.
DSUM	Generic for the sum of the directions of spread (TDSUM or PDSUM).
DSUM2	Generic for the sum of squares of the directions of spread (TDSUM2 or PDSUM2).
DVAR	Character variable identifying variability class of the direction of spread (low/moderate/high).
FN	Floating point generic for the number of areas (NT or NP).
IDMEAN	Integer representation of DMEAN (degrees azimuth).
IDSTD	Integer representation of DSTD.
IVDIR	Integer representation of VDIR (degrees azimuth).
N	Generic number of areas in analysis (NT or NP).
RAD	Degrees per radian.
RSTD	Standard deviation of the rate of spread.
RSUM	Generic for the sum of the rates of spread (TRSUM or PRSUM).
RSUM2	Generic for sum of squares of the rates of spread (TRSUM2 or PRSUM2).
RVAR	Character variable identifying variability class of the rate of spread (low/moderate/high).
VADIV	Rate at which the heads of the vector average and a vector representing the arithmetic average are spreading (units/minute).
VDIR	Vector direction of spread (degrees azimuth).
VRATE	Vector rate of spread (units/minute).
WGT	Generic for the sum of the areas involved (TWGT or PWGT).
XSUM	Generic for the sum of the x-axis components of the rate of spread vectors (TXSUM or PXSUM).
YSUM	Generic for the sum of the y-axis components of the rate of spread vectors (TYSUM or PYSUM).

FUNCTION LINED

L	Maximum number of output lines needed to process current plot.
MLINE	Array containing maximum output line counts for various plot configurations dependent on number of possible triangles that may be processed.

SUBROUTINE ORIGIN

AREAS	The area of the current polygon under consideration.
IDIGIT	The number of digits in the area figure. Used in formatting.
MPS	The number of points still to be considered.
NAREA	Integer representation of TAREA (square units).
NPWGT	Integer representation of PWGT (square units).
TAREA	Total convex polygonal area covering all individual plots (square units).
XMAX	The maximum value in the GPX array.
YMAX	The maximum value in the GPY array.

SUBROUTINE ORIENT

MISS	Logical set .FALSE. when at least one point of current plot matches a point already stored in the GPT array.
X	The difference between the x-coordinates of a point in the current plot and the identical point already stored in the GPT array.
Y	The difference between the y-coordinates of a point in the current plot and the identical point already stored in the GPT array.

SUBROUTINE GRAFIC

BORDR	The location of the right border of the current graphic from the plotter origin (inches).
CENTER	The width of paper from the plotter origin up to the center of the scale (inches).
PI	A perimeter point of the current individual plot.
PN	The northernmost point of the current individual plot.
PS	The number of increments in the scale.
SCALE	The width of paper from the plotter origin up to the right side of the scale (inches).
X	The x-coordinate of where to set the pen down in order to graph each timer's alphanumeric (inches).
XPN	The x-coordinate of PN (inches).
Y	The y-coordinate of where to set the pen down in order to graph each timer's alphanumeric (inches).
YPN	The y-coordinate of PN (inches).

SUBROUTINE VECTOR

ANGLE	Amount of rotation given to symbol indicating truncated vector.
DMIN	The vectors are normally drawn in units/DMIN minutes (see DSEC).
DS	The number of divisions of a minute that the vectors are drawn in if units/minute results in too long a vector.
DSEC	If units/DMIN minutes results in vectors that are too long, then they are drawn in units/DSEC seconds.
DV	Plotter inches/minute change in vector length.
DX	Plotter inches/minute change in the x-axis component of the vector.
DY	Plotter inches/minute change in the y-axis component of the vector.
FRAME	The width of paper from the plotter origin up to the right side of the legend 'VECTORS IN UNITS/'.
HEAD	The length of the arrowhead on the spread vector.
M	Factor used to scale third longest vector to a size less than an inch and a half.
RAD	Degrees per radian.
RATE	Units/minute change in vector length.
RCOS	Units/minute change in the x-axis component of the vector.
RSIN	Units/minute change in the y-axis component of the vector.
S	Intermediate variable used in locating truncated vector symbol.
SLOPE	The slope of the vector.
S1	Intermediate variable (see S).
S2	Intermediate variable (see S).
S3	Intermediate variable (see S).
TRI(3)	Character array containing the 3 points in current triangle.
VMAX1	Length of longest vector (inches).
VMAX2	Length of second longest vector (inches).
VMAX3	Length of third longest vector (inches).
XBND	The location of plotter boundary in the North direction (inches).
X1	The x-coordinate of the tail of the vector (inches).
X2	The x-coordinate of the head of the vector (inches).
YBND	The location of plotter boundary in the East direction (inches).
Y1	The y-coordinate of the tail of the vector (inches).
Y2	The y-coordinate of the head of the vector (inches).

APPENDIX 3

FILE QDATA

Input File of the Example

```
A 57 21  14.14  45
B 54 41
D 41 14  14.14 225
E 33 20  14.14 135
#
A 57 21  14.14 315
C 43 05
E 33 20  14.14 225
F 28 12  14.14 135
#
D 41 14  14.14 315
E 33 20  14.14  45
G 26 37
I 14 29  14.14 225
J 14 32  14.14 135
#
E 33 20  14.14 315
F 28 12  14.14  45
H 22 45
J 14 32  14.14 225
K 16 13  14.14 135
#
I 14 29  14.14 315
J 14 32  14.14  45
L 12 33
N 18 58  14.14 135
#
J 14 32  14.14 315
K 16 13  14.14  45
M 17 22
N 18 58  14.14 225
O 20 20  14.14 135
#
Z  0 00
P  3 58   5.00  85
S  9 16   5.00  40
V 19 04   5.00 355
#
P  3 58   5.00  85
Q 10 29  10.00  85
S  9 16   5.00  40
T 16 55  10.00  40
#
Q 10 29  10.00  85
R 10 49  15.00  85
T 16 55  10.00  40
U 19 31  15.00  40
#
```

Input File Continued

```
S  9 16   5.00  40
T 16 55  10.00  40
V 19 04   5.00 355
W 33 57  10.00 355
#
T 16 55  10.00  40
U 19 31  15.00  40
W 33 57  10.00 355
X 41 30  15.00 355
#
Z  0 00
N 18 58  40.16 109
P  3 58   5.00  85
Q 10 29  10.00  85
#
L 12 33  28.16  96
N 18 58  40.16 109
Q 10 29  10.00  85
R 10 49  15.00  85
#
I 14 29  19.31  69
L 12 33  28.16  96
R 10 49  15.00  85
U 19 31  15.00  40
#
D 41 14  30.45  34
I 14 29  19.31  69
U 19 31  15.00  40
X 41 30  15.00 355
#
G 26 37  14.14  45
I 14 29
L 12 33  14.14 135
#
L 12 33  14.14 315
M 17 22  14.14  45
N 18 58
#
H 22 45  14.14 315
K 16 13
M 17 22  14.14 225
#
G 26 37  14.14 315
H 22 45  14.14  45
J 14 32
L 12 33  14.14 225
M 17 22  14.14 135
```

APPENDIX 4

FILE QRATE
Output File of the Example

USDA FOREST SERVICE SPREAD RATE PROGRAM

RATES IN UNITS/MINUTE
DIRECTIONS IN DEGREES AZIMUTH

STATISTICS FOR PLOT #1--AEDBA
AREA: 200 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	EBA	.7	322
	EDA	.8	342
	EDB	.6	347
SUMMARY TABLE			
VECTOR AVERAGE		.7	338
V/A DIVERGENCE		.0	
ARITHMETIC MEAN		.7	338
STD. DEV.		.1	12
VARIANT		LOW	MOD.

STATISTICS FOR PLOT #2--ACFEA
AREA: 200 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	ECA	.8	349
	FEA	.8	348
	FEC	.8	348
SUMMARY TABLE			
VECTOR AVERAGE		.8	348
V/A DIVERGENCE		.0	
ARITHMETIC MEAN		.8	348
STD. DEV.		.0	1
VARIANT		LOW	LOW

STATISTICS FOR PLOT #3--EJIDE
AREA: 400 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	GED	.9	340
	IED	.7	344
	JED	1.0	337
	IGD	.7	355
	IJD	.7	0
	JGE	1.0	344
	IJE	1.1	0
	IJG	.8	0
SUMMARY TABLE			
VECTOR AVERAGE		.9	350
V/A DIVERGENCE		.0	
ARITHMETIC MEAN		.9	350
STD. DEV.		.1	10
VARIANT		LOW	MOD.

STATISTICS FOR PLOT #4--FKJEF
AREA: 400 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	HFE	1.2	342
	JFE	1.0	345
	KFE	1.5	337
	JHE	1.1	353
	JKE	1.1	5
	KHF	1.7	355
	JKF	1.7	8
	JKH	1.3	7
SUMMARY TABLE			
VECTOR AVERAGE		1.3	354
V/A DIVERGENCE		.0	
ARITHMETIC MEAN		1.3	354
STD. DEV.		.3	13
VARIANT		MOD.	MOD.

STATISTICS FOR PLOT #5--JNIJ
AREA: 200 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
LIJ	5.1	1
IJN	4.5	179
LJN	2.1	118
SUMMARY TABLE		
VECTOR AVERAGE	1.3	157
V/A DIVERGENCE	3.1	
ARITHMETIC MEAN	4.1	119
STD. DEV.	1.4	89
VARIANT	MOD.	HIGH

STATISTICS FOR PLOT #6--KONJK
AREA: 400 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
JKM	4.6	157
JKN	4.2	159
JKO	4.5	158
JMN	4.3	164
JNO	4.3	163
KMO	4.4	156
KNO	4.6	162
MNO	4.2	163
SUMMARY TABLE		
VECTOR AVERAGE	4.4	160
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	4.4	160
STD. DEV.	.2	3
VARIANT	LOW	LOW

STATISTICS FOR PLOT #7--VSPZV
AREA: 18 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
ZPS	.5	18
ZPV	.3	7
ZSV	.3	338
PSV	.4	274
SUMMARY TABLE		
VECTOR AVERAGE	.3	354
V/A DIVERGENCE	.1	
ARITHMETIC MEAN	.3	349
STD. DEV.	.1	40
VARIANT	MOD.	HIGH

STATISTICS FOR PLOT #8--TQPST
AREA: 27 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
PSQ	.6	4
PQT	.5	37
PST	.6	11
SQT	.6	30
SUMMARY TABLE		
VECTOR AVERAGE	.6	24
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	.6	25
STD. DEV.	.1	15
VARIANT	LOW	MOD.

STATISTICS FOR PLOT #9--URQTU
AREA: 44 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
QRT	1.1	359
QRU	1.2	359
QTU	1.2	347
RTU	1.3	351
SUMMARY TABLE		
VECTOR AVERAGE	1.2	354
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	1.2	354
STD. DEV.	.1	6
VARIANT	LOW	LOW

STATISTICS FOR PLOT #10--WTSVW
AREA: 27 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
STV	.4	346
STW	.3	337
SVW	.3	328
TVW	.3	333
SUMMARY TABLE		
VECTOR AVERAGE	.3	336
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	.3	336
STD. DEV.	.0	6
VARIANT	LOW	LOW

STATISTICS FOR PLOT #11--XUTWX
AREA: 44 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
TUW	.4	323
TUX	.4	323
TWX	.4	305
UWX	.5	311
SUMMARY TABLE		
VECTOR AVERAGE	.4	316
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	.4	316
STD. DEV.	.0	9
VARIANT	LOW	LOW

STATISTICS FOR PLOT #12--QNZQ
AREA: 82 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		

THIS PLOT HAS INSUFFICIENT DATA. IT IS NOT INCLUDED IN THE OVERALL STATISTICS.

STATISTICS FOR PLOT #13--RLNQR
AREA: 86 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
QLN	1.5	182
RLN	1.3	188
SUMMARY TABLE		
VECTOR AVERAGE	1.4	184
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	1.5	184
STD. DEV.	.1	4
VARIANT	LOW	LOW

STATISTICS FOR PLOT #14--UILRU
AREA: 74 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
RLI	1.7	31
RIU	1.3	346
RLU	1.0	26
SUMMARY TABLE		
VECTOR AVERAGE	1.3	19
V/A DIVERGENCE	.1	
ARITHMETIC MEAN	1.3	18
STD. DEV.	.4	22
VARIANT	MOD.	HIGH

STATISTICS FOR PLOT #15--DIXD
AREA: 173 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
IUD	.7	8
IDX	.6	330
UDX	.5	338
SUMMARY TABLE		
VECTOR AVERAGE	.6	341
V/A DIVERGENCE	.0	
ARITHMETIC MEAN	.6	341
STD. DEV.	.1	18
VARIANT	LOW	MOD.

STATISTICS FOR PLOT #16--GLIG
AREA: 100 SQUARE UNITS

	RATE	DIRECTION
INDIVIDUAL TRIANGLES		
LIG	1.2	36

STATISTICS FOR PLOT #17--MNLM
AREA: 100 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	LMN	2.1	149

STATISTICS FOR PLOT #18--HKMH
AREA: 100 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	KMH	2.1	305

STATISTICS FOR PLOT #19--HMLGH
AREA: 400 SQUARE UNITS

		RATE	DIRECTION
INDIVIDUAL TRIANGLES			
	JHG	1.0	349
	LHG	1.4	345
	MHG	3.0	324
	LJG	1.2	324
	LMG	1.3	19
	JMH	1.6	64
	LMH	2.8	42
	LJM	4.1	100
SUMMARY TABLE			
VECTOR AVERAGE		1.4	17
V/A DIVERGENCE		.7	
ARITHMETIC MEAN		2.1	11
STD. DEV.		1.0	45
VARIANT		MOD.	HIGH

OVERALL STATISTICS
TOTAL AREA: 2468 SQUARE UNITS--ACFONZWXA
SUM OF ALL
PLOT AREAS: 2991 SQUARE UNITS

NOTE: THE INPUT FILE INCLUDED PLOTS THAT OVERLAPPED EACH OTHER. BOTH THE 'SUM OF ALL PLOT AREAS' AND THE FOLLOWING STATISTICS, WHICH ARE WEIGHTED BY AREA, REFLECT ADDED EMPHASIS IN THE OVERLAPPED AREAS.

NOTE: A PORTION OF THE 'TOTAL AREA' MAY NOT BE CONTAINED WITHIN ANY PLOT. IF SO, THIS PORTION IS NOT INCLUDED IN THE FOLLOWING STATISTICS.

	RATE	DIRECTION
SUMMARY TABLE		
VECTOR AVERAGE	.2	94
V/A DIVERGENCE	1.8	
ARITHMETIC MEAN	1.9	35
STD. DEV.	1.4	74
VARIANT	HIGH	HIGH

APPENDIX 5

CALCOMP PLOTTER COMMANDS

The program makes 49 calls to the CALCOMP plotter using the seven commands listed below. All calls originate from three subprograms: CALCOM, GRAFIC, and VECTOR.

Note that the coordinate system used by the program is based on the compass--North is on the positive x-axis and East is on the positive y-axis. Because the CALCOMP plotter is based on the standard Cartesian coordinate system, the two axes are reversed. In other words, if a plotter call normally expects arguments (X, Y), the user should use (Y, X) instead.

CALL PLOTS (IBUF, 257, ITYPE)

The first call to the plotter is made from subprogram CALCOM. Its purpose is to set aside an area of computer memory to accumulate plotter commands and to buffer output. The first argument, IBUF, is the name of the buffer; the second argument, 257, is its size. ITYPE, the third argument, selects the paper and pens (see BLOCK DATA). Note that IBUF and ITYPE are both carried by COMMON /GRPHC/.

CALL PLIMIT (BORDR + 1.0)

The single call to PLIMIT is made near the beginning of subprogram GRAFIC. Its purpose is to indicate the length of paper required. CALL PLOTS fixes the length of the x-axis (argument ITYPE determines, among other things, the width of the paper). This, in turn, determines BORDR--the total length of y-axis necessary for a proportional drawing. The argument, BORDR + 1.0, includes an extra inch for a border.

NEWPEN (n)

The call to NEWPEN is made five times--twice from subprogram CALCOM and three times from subprogram GRAFIC. The call enables the program to select a different pen from among the choices available where n is the number of the choice.

CALL PLOT (Y, X, n)

The call to PLOT is made 13 times--9 calls within subprogram GRAFIC, 3 calls from subprogram VECTOR, and 1 final call from subprogram CALCOM. The first two arguments (Y, X) are the coordinates of the position to which the pen is to be moved. When the third argument, n, is equal to 2, the pen is down during the move. If n = 3, the pen is up during the move and no line is drawn. In the final call from subprogram CALCOM, n is equal to 999, indicating all plotting calls have been made and the plot file can be closed.

CALL SYMBOL (Y, X, height, text, angle, n)

The call to SYMBOL is made 24 times--11 calls from subprogram CALCOM plotting the North directional symbol, 5 calls from subprogram GRAFIC plotting the scale and each of the timer points, and 8 more calls from subprogram VECTOR noting the time frame the vectors use (e.g., UNITS/MINUTE, UNITS/30 SECONDS, etc.) and plotting the truncated vector symbol (should a vector extend beyond the plotter limits).

Text can be entered on the plotter in a number of ways, three of which are used by the program. The first method is used in subprograms CALCOM and VECTOR and is identified by argument n = -1. The fourth argument, for this method, is an index into a special CALCOM symbol character set--text = 11 for an asterisk, text = 13 for a vertical line, and text = 8 for the character 'Z'. The third argument, height, is the vertical height (inches) for the symbol, and the fifth argument, angle, is the amount of counterclockwise rotation required (degrees).

In subprogram CALCOM, the asterisk is rotated 45° and topped off with the unrotated vertical bar pointing to North. In subprogram VECTOR, if a vector would extend beyond the limits of the plotter, a 'Z' is embedded on its shaft to indicate shortening. The (Y, X) coordinates, for this method, locate the center of the symbol.

In both of the other two methods that the program uses for plotting text, the (Y, X) coordinates locate the lower left-hand corner of the first character to be produced. The last argument n for both methods refers to the number of characters in the text. Arguments height and angle are as before (the angles are all 0.0 meaning no rotation). The two methods differ in the way the fourth argument is presented. In one case the argument is a variable name, GPT(K:K), and the text plotted is the content of that variable--the alphanumeric identifier of a timer. All other calls to SYMBOL use the argument text as a literal expression, signified by embedding the text within single quotes.

CALL NUMBER (Y, X, height, variable, angle, -1)

The four calls to NUMBER (one from subprogram GRAFIC and three from subprogram VECTOR) all use the same form of the command. The (Y, X) coordinates locate the lower left-hand corner of the first digit of the number. The arguments height and angle are as in SYMBOL. The fourth argument is the name of the variable that contains the number to be plotted. The last argument, -1, indicates that only the integer portion (after rounding) is to be plotted.

CALL AROWHD (Y1, X1, Y2, X2, HEAD, 0.5*HEAD, 2, 2)

The call to AROWHD appears only once (subprogram VECTOR) and plots each vector. The coordinates (Y1, X1) locate the start (tail) and (Y2, X2) locate the end (tip of the arrowhead) of the vector to be plotted. The arguments HEAD and 0.5*HEAD are the length and width, respectively, of the arrowhead. The seventh argument, 2, indicates that the pen is to be down for the shaft of the arrow (i.e., the line segment from (Y1, X1) to (Y2, X2) is to be drawn). The last argument, also a 2, selects the second type of arrowhead from a list of available styles.

Eenigenburg, James E.
1987. Computer program for calculating and plotting fire direction and rate of spread. Gen. Tech. Rep. NC-117. St. Paul, MN: U.S. Department of Agriculture, Forest Service, North Central Forest Experiment Station. 46 p.

Presents an analytical procedure that uses a FORTRAN 77 program to estimate fire direction and rate of spread. The program also calculates the variability of these parameters, both for subsections of the fire and for the fire as a whole. An option in the program allows users with a CALCOMP plotter to obtain a map of the fire with spread vectors.

KEY WORDS: FORTRAN program, CALCOMP plotter, vector, prescribed burn, wildland fire.