27 August 1975: se

Plot Programs

by Anupam Khanna

This report documents and explains the subroutines for plotting "vertical traces" side by side. Instead of setting and clearing individual bits on each horizontal row plotted (a terribly slow process) it uses masks of word size.

The subroutine has been implemented on an IBM 360/67, with minor modifications on a minicomputer INTERDATA 7/16 in the Information Systems Laboratory at Stanford.

The arguments are:

X : Function to be plotted stored as X(IT,IX)

NT : Length of each trace, i.e. max IT

NX : Number of series or traces

FILL : A parameter determining the overlap desired between braces

LA : The number of printer columns (determined by plotter)

NSLICE: Thickness of horizontal plot lines

IOPT : Option for the type of traces desired.

These are elaborated upon in Figure 1.

The routine basically sets 66 32-bit buffers called BUFF(1) to BUFF(66) for each horizontal line to be plotted. This is done using a subroutine LINE(ISTART, IEND, BUFF) illustrated in Figure 2. Figure 3 illustrates the meaning of the logical variables LEFT(I) and RIGHT(I).

The complete subroutine is illustrated in Figure 5. It tests also for zero data, besides scaling the traces to fit into a region of desired size.

```
      SUBROUTINE QPLEXT(Q,NX,NT,FILL,LA,NSLICE,IOPT)
C     PLOT 'NX' TIME FUNCTIONS EACH OF LENGTH 'NT'.
C     LINES=THE NUMBER OF SERIES OR SECTIONS TO SPLIT UP X
C     FILL=1. NORMALLY
C         =1.5 FOR SOME OVERLAP
C         =100000. TO PLOT SERIES ON A COMMON MEAN
C     LA=PRINTER COLUMNS :.LE.  120 OR SO FOR PRINTED PAGE
C                              2112 FOR GOULD 5200 &NEW VERSATEC
C                              1056 FOR GOULD 5000
C                               560 FOR OLD VERSATEC
C     IOPT=1 FOR 'WIGGLE' TRACES
C         =2 FOR 'WIGGLE' TRACES WITH 'VARIABLE AREA'
C         =3 FOR 'VARIABLE AREA' TRACES; BOTH POLARITIES
C         =4 FOR 'VARIABLE AREA' TRACES
```

Fig. 1. Subroutine arguments, their meanings and allowable range.

```
      SUBROUTINE LINE(ISTART,IEND,BUFF)
      INTEGER*4 WS,BS,WE,BE,W
      LOGICAL*4 LEFT(32),RIGHT(32),BUFF(66),MASK
      DATA LEFT /Z80000000,ZC0000000,ZE0000000,ZF0000000,ZF8000000,
     2ZFC000000,ZFE000000,ZFF000000,ZFF800000,ZFFC00000,ZFFE00000,
     3ZFFF00000,ZFFF80000,ZFFFC0000,ZFFFE0000,ZFFFF0000,ZFFFF8000,
     4ZFFFFC000,ZFFFFE000,ZFFFFF000,ZFFFFF800,ZFFFFFC00,ZFFFFFE00,
     5ZFFFFFF00,ZFFFFFF80,ZFFFFFFC0,ZFFFFFFE0,ZFFFFFFF0,ZFFFFFFF8,
     6ZFFFFFFFC,ZFFFFFFFE,ZFFFFFFFF/
      DATA RIGHT /ZFFFFFFFF,Z7FFFFFFF,Z3FFFFFFF,Z1FFFFFFF,Z0FFFFFFF,
     2Z07FFFFFF,Z03FFFFFF,Z01FFFFFF,Z00FFFFFF,Z007FFFF,Z003FFFF,
     3Z001FFFF,Z000FFFF,Z0007FFF,Z0003FFF,Z0001FFF,Z0000FFF,
     4Z00007FF,Z00003FF,Z00001FF,Z00000FF,Z000007F,Z000003F,
     5Z000001F,Z000000F,Z00000007F,Z00000003F,Z00000001F,Z00000000F,
     6Z00000007,Z00000003,Z00000001/
      WS=1+(ISTART-1)/32
      BS=1+MOD(ISTART-1,32)
      WE=1+(IEND-1)/32
      BE=1+MOD(IEND-1,32)
      W=WS
   10 MASK=RIGHT(BS)
      IF(WE.EQ.W) MASK=MASK.AND.LEFT(BE)
      BUFF(W)=BUFF(W).OR.MASK
      W=W+1
      BS=1
      IF(WE.GE.W) GO TO 10
      RETURN
      END
```

Fig. 2. Subroutine for setting masks for each segment to be plotted. This segment extends from ISTART to IEND and can extend in length from 1 bit to the whole line.

Fig. 3. LOGICAL*4 words LEFT(I) and RIGHT(I). Obviously 32 of each have to be defined.

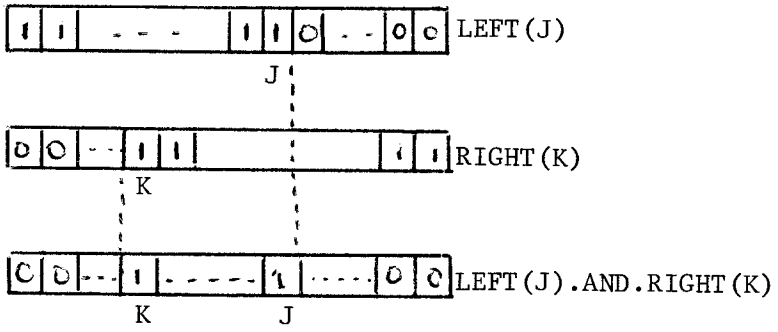

Fig. 4. .AND. operation sets the masks as illustrated in this figure. If the segment extends over more than one word, set subsequent buffers to RIGHT(1) and use .AND. operation on last one corresponding to IEND.

```
      DIMENSION Q(NT,NX),INTEG(2),MEAN(2)
      LOGICAL*4 BUFF(66),BUFFI(66),ZERO
      LINES=NX
      SIDES=1.
      B=0.
      DO 5 IX=1,NX
      DO 5 IT=1,NT
      IF(Q(IT,IX).LE.0.) SIDES=2.
    5 IF(ABS(Q(IT,IX)).GT.B) B=ABS(Q(IT,IX))
      IF(B.NE.0.) GO TO 7
      PRINT 6
    6 FORMAT(//,' QPLOT CALLED WITH ZERO DATA')
    7 CONTINUE
      DY=FILL*LA/(FILL+LINES-1)/SIDES
      DM=SIDES*DY/FILL
      DATA ZERO /Z00000000/
      DO 12 M=1,66
      BUFF(M)=ZERO
   12 CONTINUE
      DO 15 J=1,LINES
      MEAN(J)=LA-DY-(J-1)*DM+1
      IF(IOPT.NE.1) GO TO 15
      ISTART=MEAN(J)
      IEND=ISTART
C     INITIALIZE BUFF TO MEANS
      CALL LINE(ISTART,IEND,BUFF)
   15 INTEG(J)=MEAN(J)
      DO 10 M=1,66
   10 BUFFI(M)=BUFF(M)
      DO 30 K=1,NT
      DO 11 M=1,66
   11 BUFF(M)=BUFFI(M)
      DO 40 J=1,LINES
      IF(IOPT-2) 60, 70, 80
   60 ISTART=INTEG(J)
      INTEG(J)=MEAN(J)+Q(K,J)*DY/B
      IF(INTEG(J).LT.1) INTEG(J)=1
      IF(INTEG(J).GT.LA) INTEG(J)=LA
   65 IF(ISTART-INTEG(J)) 61,62,63
   61 IEND=INTEG(J)
      ISTART=ISTART+1
      GO TO 100
```

Fig. 5. Continued on next page.

```
 62 IEND=ISTART
    GO TO 100
 63 IEND=ISTART-1
    ISTART=INTEG(J)
    GO TO 100
 70 ISTART=INTEG(J)
    INTEG(J)=MEAN(J)+Q(K,J)*DY/B
    IF(INTEG(J).LT.1) INTEG(J)=1
    IF(INTEG(J).GT.LA) INTEG(J)=LA
    IF (INTEG(J).LT.MEAN(J)) GO TO 65
    IF(ISTART.LT.MEAN(J)) GO TO 73
    ISTART=MEAN(J)
 73 IEND =INTEG(J)
    GO TO 100
 80 ISTART=MEAN(J)
    INTEG(J)=MEAN(J)+Q(K,J)*DY/B
    IF(INTEG(J).LT.1) INTEG(J)=1
    IF(INTEG(J).GT.LA) INTEG(J)=LA
    IF(ISTART.GT.INTEG(J)) GO TO 85
    IEND=INTEG(J)
    GO TO 100
 85 IF(IOPT.EQ.4) GO TO 87
    IEND=ISTART
    ISTART=INTEG(J)
    GO TO 100
 87 IEND=ISTART
100 CALL LINE(ISTART,IEND,BUFF)
 40 CONTINUE
     DO 46 I=1,NSLICE
    CALL WRITER(BUFF,66*4)
 46 CONTINUE
 30 CONTINUE
    RETURN
    END
```

Fig. 5. Complete subroutine which tests for zero data and scales the traces to fit into a region of desired size.