NOAA Western Region Computer Programs and
Problems NWS WRCP - No. 40

AFOS VECTOR GRAPHIC TO GRID POINT PROGRAM

Salt Lake City, Utah
December 1982.

U.S. DEPARTMENT OF COMMERCE / National Oceanic and Atmospheric Administration / National Weather Service

PREFACE

This Western Region publication series is considered as a subset of our Technical Memorandum series. This series will be devoted exclusively to the exchange of information on and documentation of computer programs and related subjects. This series was initiated because it did not seem appropriate to publish computer program papers as Technical Memoranda; yet, we wanted to share this type of information with all Western Region forecasters in a systematic way. Another reason was our concern that in the developing AFOS-era there will be unnecessary and wasteful duplication of effort in writing computer programs in National Weather Service (NWS). Documentation and exchange of ideas and programs envisioned in this series hopefully will reduce such duplication. We also believe that by publishing the programming work of our forecasters, we will stimulate others to use these programs or develop their own programs to take advantage of the computing capabilities AFOS makes available.

We solicit computer-oriented papers and computer programs from forecasters for us to publish in this series. Simple and short programs should not be prejudged as unsuitable.

The great potential of the AFOS-era is strongly related to local computer facilities permitting meteorologists to practice in a more scientific environment. It is our hope that this new series will help in developing this potential into reality.

NOAA WESTERN REGION COMPUTER PROGRAMS AND PROBLEMS NWS WRCP

1. Standardized Format for Computer Series.
2. AFOS Crop and Soil Information Report Programs. Ken Mielke, July 1979.
3. Decoder for Significant Level Transmissions of Raobs. John A. Jannuzzi, August 1979.
4. Precipitable Water Estimate. Elizabeth Morse, October 1979.
5. Utah Recreational Temperature Program. Kenneth M. Labas, November 1979.
6. Normal Maximum/Minimum Temperature Program for Montana, Kenneth Mielke, December 1979.
7. Plotting of Ocean Wave Energy Spectral Data. John R. Zimmerman, December 1979.
8. Raob Plot and Analysis Routines. John A. Jannuzzi, January 1980.
9. The SWAB Program. Morris S. Webb, Jr., April 1980. (PB80-196041)
10. Flash-Flood Procedure. Donald P. Laurine and Ralph C. Hatch, April 1980. (PB80-298658)
11. Program to Forecast Probability of Summer Stratus in Seattle Using the Durst Objective Method. John R. Zimmerman, May 1980.
12. Probability of Sequences of Wet and Dry Days. Hazen H. Bedke, June 1980. (PB80-223340)
13. Automated Montana Hourly Weather Roundup. Joe L. Johnston, July 1980. (PB81-102576)
14. Lightning Activity Levels. Mark A. Mollner, July 1980. (PB81-108300)
15. Two Fortran Applications of Wind-Driven Ekman Water Transport Theory: Upwelling Index and Storm Tide. Kent S. Short, July 1980. (PB81-102568)
16. AFOS System Local Data Base Save and Rebuild Procedures or A Master Doomsday Program. Brian W. Finke, July 1980. (PB81-108342)
17. AFOS/RDOS Translator Subroutine. Morris S. Webb, Jr., August 1980. (PB81-108334)
18. AFOS Graphics Creation from Fortran. Alexander E. MacDonald, August 1980. (PB81-205304)
19. DATAKEYØ Repair Program. Paul D. Tolleson, August 1980. (PB81-102543)
20. Contiguous File Transfer from the DPCM to the DCM. Paul D. Tolleson, September 1980. (PB81-128035)
21. Freezing Level Program. Kenneth B. Mielke, September 1980. (PB81-128043)
22. Radar Boresighting Verification Program. Thomas E. Adler, November 1980.
23. Accessing the AFOS Data Base. Matthew Peroutka, January 1981.
24. AFOS Work Processor. Morris S. Webb, Jr., February 1981. (PB81-210007)
25. Automated Weather Log for Terminal Forecasting. John A. Jannuzzi, February 1981. (PB81-210999)
26. Program to Computer Downwind Concentrations from a Toxic Spill. John R. Zimmerman, February 1981. (PB81-205296)
27. Animation of AFOS Graphics. Joe Wakefield and Jim Fors, April 1981.
28. AFOS Interactive Graphics. Jim Fors, Don Laurine, and Sandy MacDonald, April 1981.
29. Computer Programs for Aviation Forecast Transmission. Kenneth B. Mielke and Matthew R. Peroutka, May 1981.
30. AFOS Product Collective Program. Morris S. Webb, Jr., September 1981.
31. Graphic Display of FOUS Output. Stephen D. Steenrod, September 1981.
32. Automation of Hourly Aviation Observation Calculations. W. Paul Duval, October 1981.
33. Mesoscale Objective Analysis. Andrew J. Spry and Jeffrey L. Anderson, December 1981.
34. Orographic Snowfall Rate Model for Alta, Utah. Steven K. Todd and Glenn E. Rasch, December 1981.
35. F-6 Monthly Climatic Summary Program for AFOS. Peter G. Mueller, May 1982.

NOAA Western Region Computer Programs and Problems NWS WRCP - No. 40

AFOS VECTOR GRAPHIC TO GRID POINT PROGRAM

James R. Fors
National Weather Service
Western Region Headquarters
Scientific Services Division

December 1982

UNITED STATES
DEPARTMENT OF COMMERCE
Malcolm Baldrige, Secretary

National Oceanic and
Atmospheric Administration
John V. Byrne, Administrator

National Weather
Service
Richard E. Hallgren, Director

This technical publication has been reviewed and is approved for publication by Scientific Services Division, Western Region.

Glenn E. Rasch

Glenn E. Rasch, Acting Chief
Scientific Services Division
Western Region Headquarters
Salt Lake City, Utah

CONTENTS

LIST OF TABLES AND FIGURES

AFOS VECTOR GRAPHIC TO GRID POINT PROGRAM

James R. Fors
Western Region Scientific Services Division

I. General Information

A. Summary:

Graphics are transmitted on the AFOS network in a coded vector format. This format facilitates rapid display of graphics on the GDM. However, in this format it provides no direct quantitative information on the value of the displayed field at a given geographical location. For some meteorological applications, values at specific locations (i.e., grid point values) are required. The usual alternative to this problem is to transmit separate files of gridded data to satisfy this requirement. However, due to traffic limitations on the AFOS network this is not currently possible. This series of programs allows the user, within certain limitations, to produce a gridded data file from most NMC produced graphics transmitted on the AFOS network.

B. Environment:

The programs are written in Data General FORTRAN IV, Revision 5.57. The programs are initiated from the AFOS ADM and AFOS must be running. A background partition of 17K words or greater is sufficient to run the programs.

C. References:

[1] MacDonald, Alexander E., 1981: "AFOS Graphics Creation from FORTRAN". Western Region Computer Programs and Problems No. 18, March.

[2] Hudadoff, Pamela A., 1982: "The Hovmöller Diagram". Western Region Computer Programs and Problems No. 38, September.

II. Application

A. Complete Program Description:

This program consists of a string of four programs. The first two programs, GRID and GRID1, involve decoding the vector graphic specified in the run line. The second two programs, GRID2 and GRID3, take the decoded graphic information and put it into a gridded data file. A schematic of the logic flow is shown in Figure 1.

The program GRID reads the run line entered by the user at the ADM. An information file, GINFO, is created that contains the information from the run line for later use. The format for GINFO is shown in Table 1. The subroutine FADBA is used to pull the requested graphic out of the AFOS data base. The graphic is written to disk as a file called NMCGRDXXX.

TABLE I. FORMAT OF FILE GINFO

Word	Contents	Format
Ø-1	Graphic Name (XXX padded with a space)	ASCII
2	Latitude of Lower Left Corner of Grid	Integer
3	Longitude of Lower Left Corner of Grid	Integer
4	Grid Spacing in X Direction Times 10	Integer
5	Grid Spacing in Y Direction Times 10	Integer
6	Map Background of Graphic	Integer
7	Special Field Flag (1 for SFC, 2 for RH, Ø Otherwise)	Integer
8	Z Time of Graphic	Integer
9	Day of the Month of Graphic	Integer
10	Month of Graphic	Integer

The purpose of program GRID1 is to decode the graphic in the NMCGRDXXX file into two files of decoded information (vector and text). Subroutine CLEANUP strips out the universal transmission format information from the graphic file. For a description of AFOS vector graphic format see Reference 1. Subroutine VECDEC decodes the coded vector information into a series of X-Y points. This information is written to the "VECTOR" file. The format of this file is shown in Table 2. The X-Y points for each individual vector are separated by a pair of -1's. Subroutine TEXTDEC decodes the label information into a file called "TEXT". It also decodes the max/min center text information. The format of this file is shown in Table 3. TEXTDEC also determines the date/time information from the graphic and if it is a surface isobar field or mean RH field.

TABLE 2. FORMAT OF FILE VECTOR

-1,-1,X,Y,X,Y,....-1,-1,X,Y,X,Y.......-1,-1,-2,-2

TABLE 3. FORMAT OF FILE TEXT

XLOC,YLOC,VALUE,XLOC,YLOC,VALUE.........-1,-1,-1

At this point the AFOS graphic has been completely decoded. Some users may wish to use the program up to this point as a general AFOS graphic decoder. This would be useful if the graphic was to be remapped to another projection or another type of graphic display device.

Program GRID2 is the meat of the software. Its function is to correlate the label information with a particular vector so that the decoded X-Y points can be assigned a value with which an objective analysis to gridded data can be done. This rather clumsy approach is necessary as the AFOS coded vector format contains no information as to the value to be assigned to a given vector. The algorithm used is to construct a straight line between two pairs of X-Y points and see if a label lies along that straight line. If it does, all the X-Y points in that string are assigned the value of the label. The assigned values for each vector string are output to a file called "COMPAR". A vector for which no label can be found is assigned a value of -1 in the COMPAR file.

Program GRID3 uses the information from files VECTOR, TEXT, and COMPAR to obtain values at specific locations. The actual label locations and max/min center values, as well as the X-Y vector points, are used as input data to the objective analysis. The objective analysis technique used is similar to that used for surface data in the AFOS mesoanalysis routines. The subroutines used in the objective analysis are CLAT (X-Y to Lat-Lon converter), POINT (nearest neighbor assignment), UPHIL (interpolation to unfilled grid points), and SMOOTH (5 point smoothing routine). The final gridded data is output to file NMCGRDXXX. GRID3 is the only program which is dependent on the number of grid points. The number of grid points is specified in a series of parameter statements at the beginning of the program. The last three words of the NMCGRDXXX file are the month, day, and time, respectively, of the graphic product decoded. The size of the NMCGRDXXX file in bytes is equal to: 4 x (#X GRID POINTS x #Y GRID POINTS) + 6

B. Machine Requirement:

Less than 17K words are required to execute these programs. The four save files require 88 RDOS blocks of disk storage. The only file remaining on disk after execution is NMCGRDXXX. The runtime for the programs for a large graphic (hemispheric) and 1400 grid points is as follows:

GRID	-	13 seconds
GRID1	-	30 seconds
GRID2	-	90 seconds
GRID3	-	65 seconds

For an LFM type graphic and 480 grid points the runtimes are reduced by roughly a factor of 4.

III. Procedures

A. Initiation of Program

The program is initiated by issuing a RUN: command from the ADM. AFOS must be up for this program to run.

B. Input Required:

The run line is a follows:

```
RUN:GRID XXX/P LA/J LON/I DX/X DY/Y N/V
```

where
XXX is graphic name
LA is latitude of lower left-hand corner of grid
LON is longitude of lower left-hand corner of grid
DX is the grid spacing in the X direction in degrees times 10
DY is the grid spacing in the Y direction in degrees times 10
N is the version of XXX to be used (defaults to current version)

Examples:

```
RUN:GRID 50H/P 30/J 140/I 15/X 15/Y
```

Graphic decoded is NMCGPH50H, lower left-hand corner of grid is 30N/140W. Grid spacing is 1.5 degrees in both the X and Y directions and the most current version is to be decoded.

```
RUN:GRID 5AH/P 20/J 350/I 50/X 30/7 1/V
```

NMCGPH5AH is to be decoded, lower left-hand corner of the grid is 20N/10E (longitude is calculated from Greenwich westward). Grid spacing is 5 degrees in the X direction and 3 degrees in the Y direction and the first previous version is to be decoded.

All switches except version number are required in each run line! If the latitude is less than 10 degrees the tens digit should be padded with a zero. The same is true for longitudes less than 100 degrees.

C. Examples:

Because an objective analysis is being done, an exact reproduction of the original field is unlikely. However, if a small enough grid spacing is chosen and enough grid points are used, the field is usually precise enough for most applications. Figure 2A shows a spectral 500-mb height chart. Figure 2B is a contoured chart reproduced from the gridded data for a grid spacing of 1 degree by 1 degree. The agreement with the original chart is quite good. Figure 2C is a chart with 2 degree by 2 degree grid spacing. Again the agreement is good. However, note that some smoothing is beginning to occur in the size of the closed contour over Nevada and in the amplitude of the ridge along 135W (especially the 564 contour). Figure 2D shows the same chart with a grid spacing of 5 degrees by 3 degrees. The overall agreement is good but the details are becoming quite smooth. All these examples are using 480 grid points. Thus, a larger area with more detail could be obtained simply by using more grid points.

Figure 3A shows a hemispheric 500-mb height analysis. Figure 3B shows the reconstructed chart for a grid spacing of 5 degrees by 3 degrees with 1400 grid points. Again large scale agreement is good even though some smoothing occurs in the details.

D. Cautions, Restrictions, and Error Messages

1. The larger the grid spacing the more detail will be lost in the gridded data.

2. Since an objective analysis is being performed, the less data that is available the more crude the analysis. Thus, a field that is being reproduced that has very few vectors in it may not be well reproduced. For example, a temperature field with only one or two contours may not reproduce well.

3. Since in order to assign values to a vector a line must have an associated label, unlabeled lines will not provide any input to the analysis. Some graphics are notoriously poorly labeled. The surface analysis on the North

American background is often poorly labeled (see Figure 4). Note that none of the contours around the edges are labeled and that often closed contours go unlabeled. Thus, the user should be careful in choosing charts and the area to be covered.

4. The routines assume a standard order in the graphic as used by NMC. The program will likely not run on locally produced graphics.

5. We have found 70 by 20 grid points with 5 degree by 3 degree grid spacing to be generally adequate to reproduce the hemispheric flow pattern. For smaller areas we generally use 24 by 20 grid points. This corresponds to the number of grid points in the AFOS mesoanalysis grid. Since GRID3 must be edited and recompiled to change the number of grid pionts, we have produced two versions of GRID3. GRID3.SV is a 24 by 20 grid and HGRID3.SV is a 70 by 20 grid.

6. The following error messages may be printed at the system console:

   a. "FADBA ERROR" - implies key not found in data base or that graphic is larger than 13 RDOS blocks which is the maximum size graphic the program will handle.

   b. "OUTPUT ERROR" - program unable to write graphic file to disk.

   c. "TEXT ERROR" - more than 1152 words of text information in the file. Graphic cannot be decoded.

   d. "NO END OF VECTOR FILE" - vector file is more than 16 blocks long or unable to chain to GRID3.

IV. Meteorological Application

The Hovmöller program [2] is a good example of a meteorological use of this output. To facilitate use of this product, a sample program is included here that produces a departure from normal chart for a hemispheric 500-mb analysis or prog. The source code for program DEPART is shown in Figure 5. Figure 6 shows a sample of the output from this program. This program is expecting a grid file of 1400 grid points (70 in the X direction and 20 in the Y direction) with a grid spacing of 5 degrees by 3 degrees. The run line for the program is:

```
RUN:DEPART  XXX/I  YYY/O
```

where XXX is the input graphic and YYY is the output departure from normal graphic. The default XXX is 5AH and the default YYY is 5DN.

Appropriate run line for grid would be:

```
RUN:GRID  XXX/P  20/J  350/I  50/X  30/Y
```

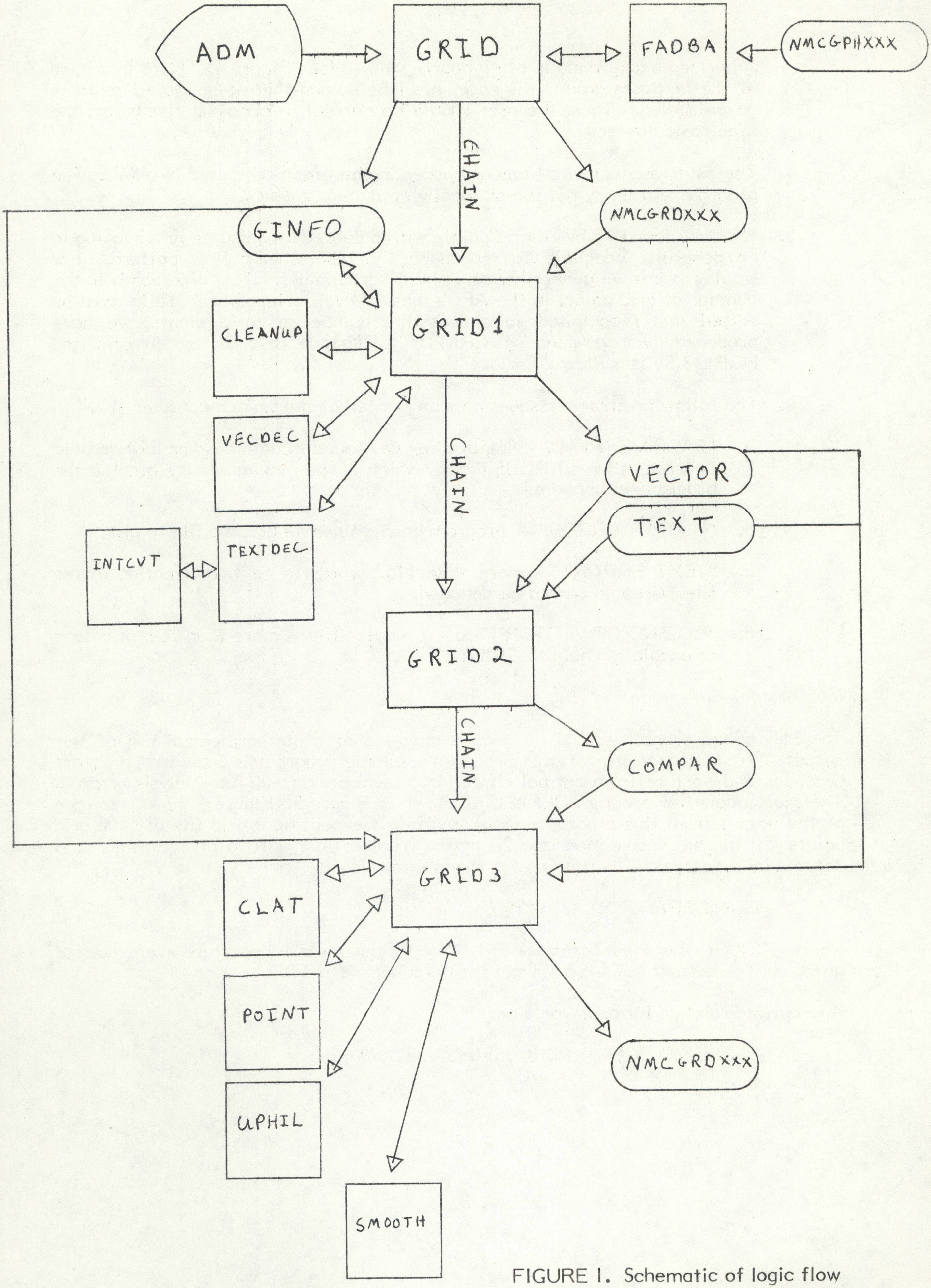


FIGURE 1. Schematic of logic flow

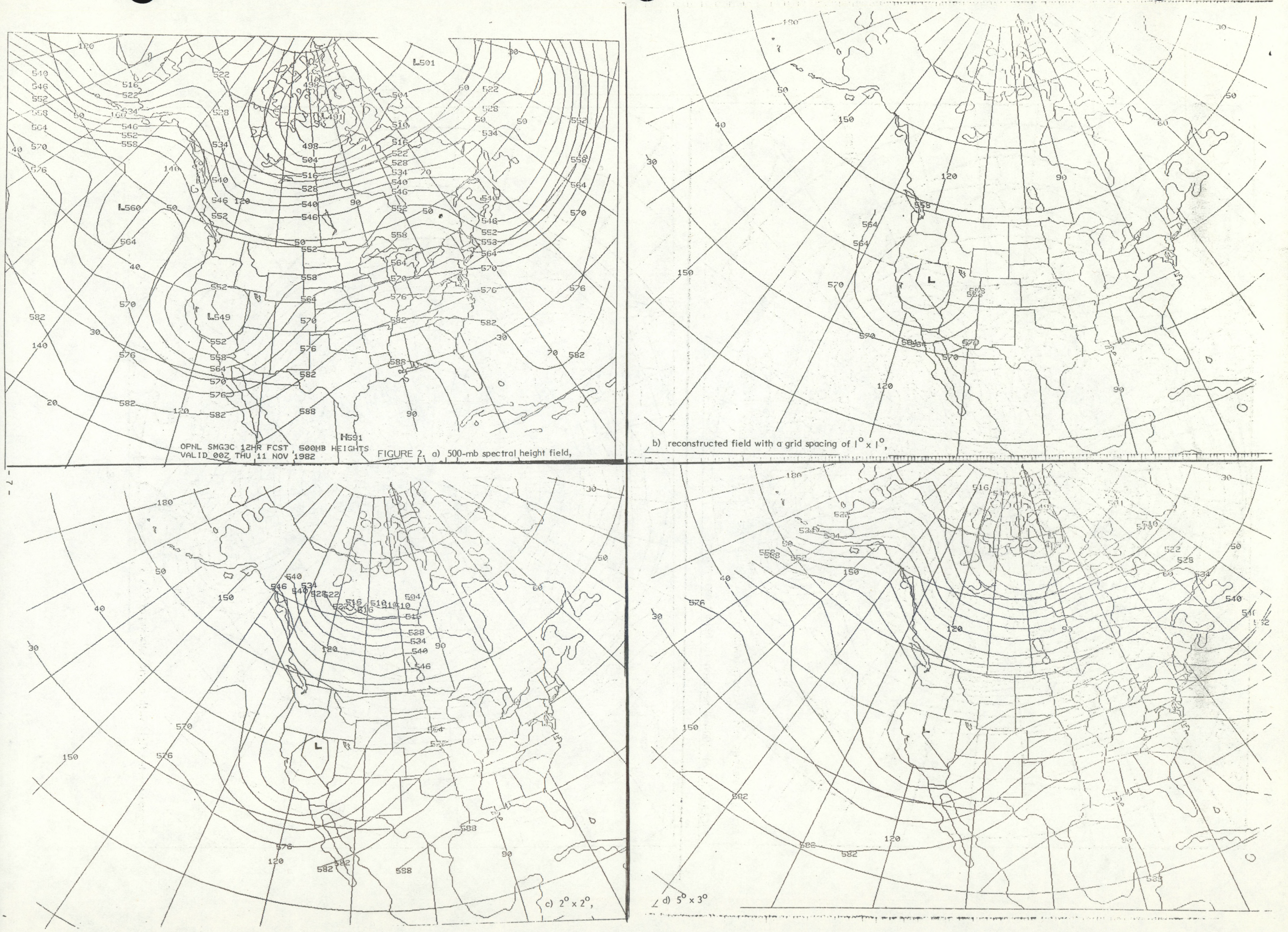


FIGURE 2. a) 500-mb spectral height field,

b) reconstructed field with a grid spacing of 1° x 1°,

c) 2° x 2°,

d) 5° x 3°

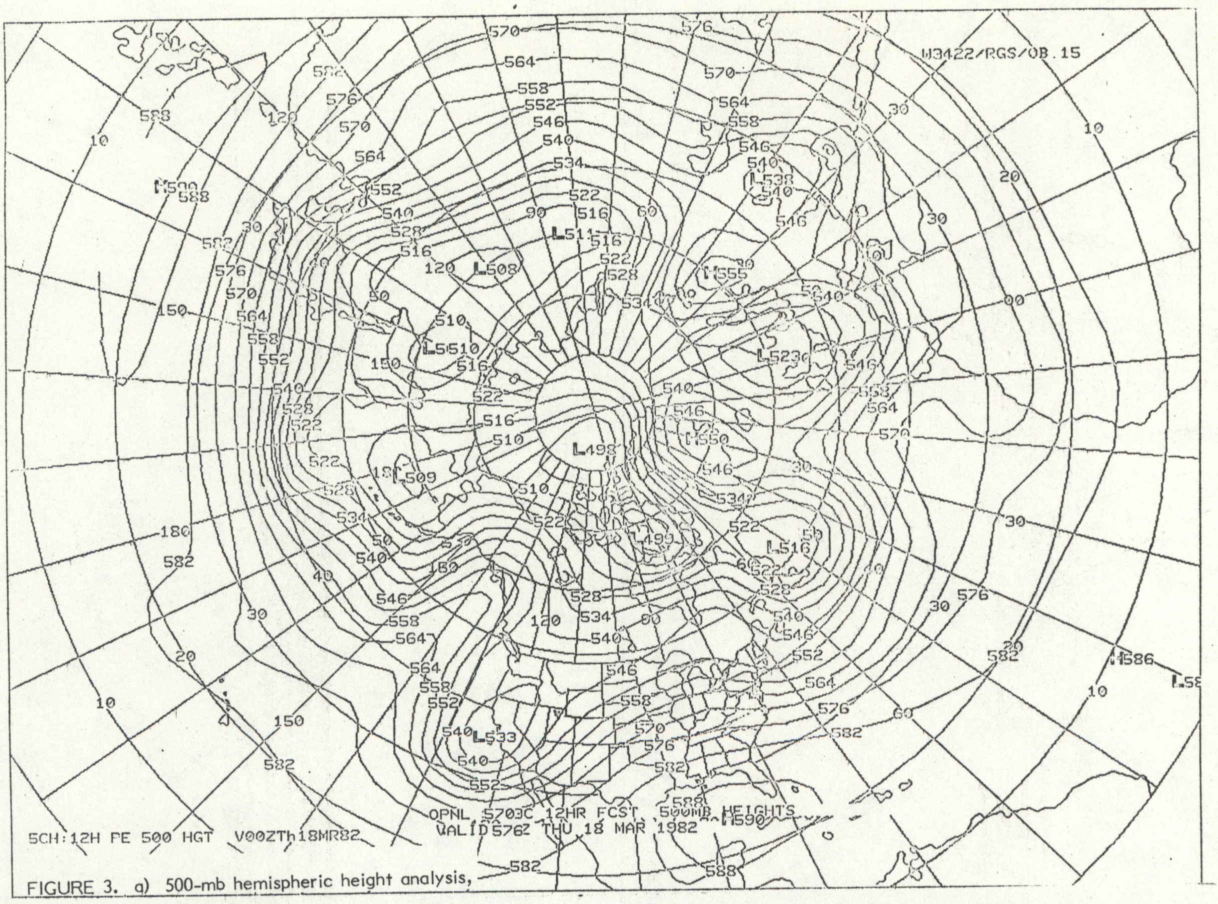
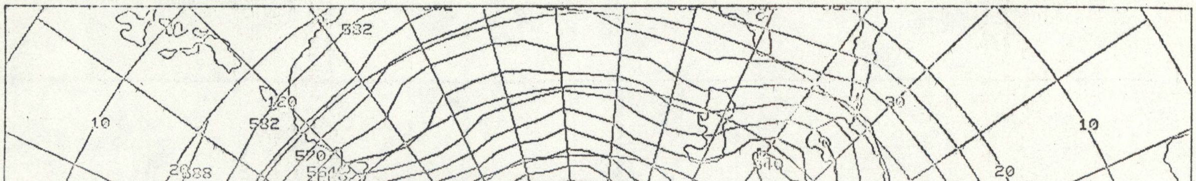


FIGURE 3. a) 500-mb hemispheric height analysis,



b) reconstructed field with a grid spacing of 5° x 3°

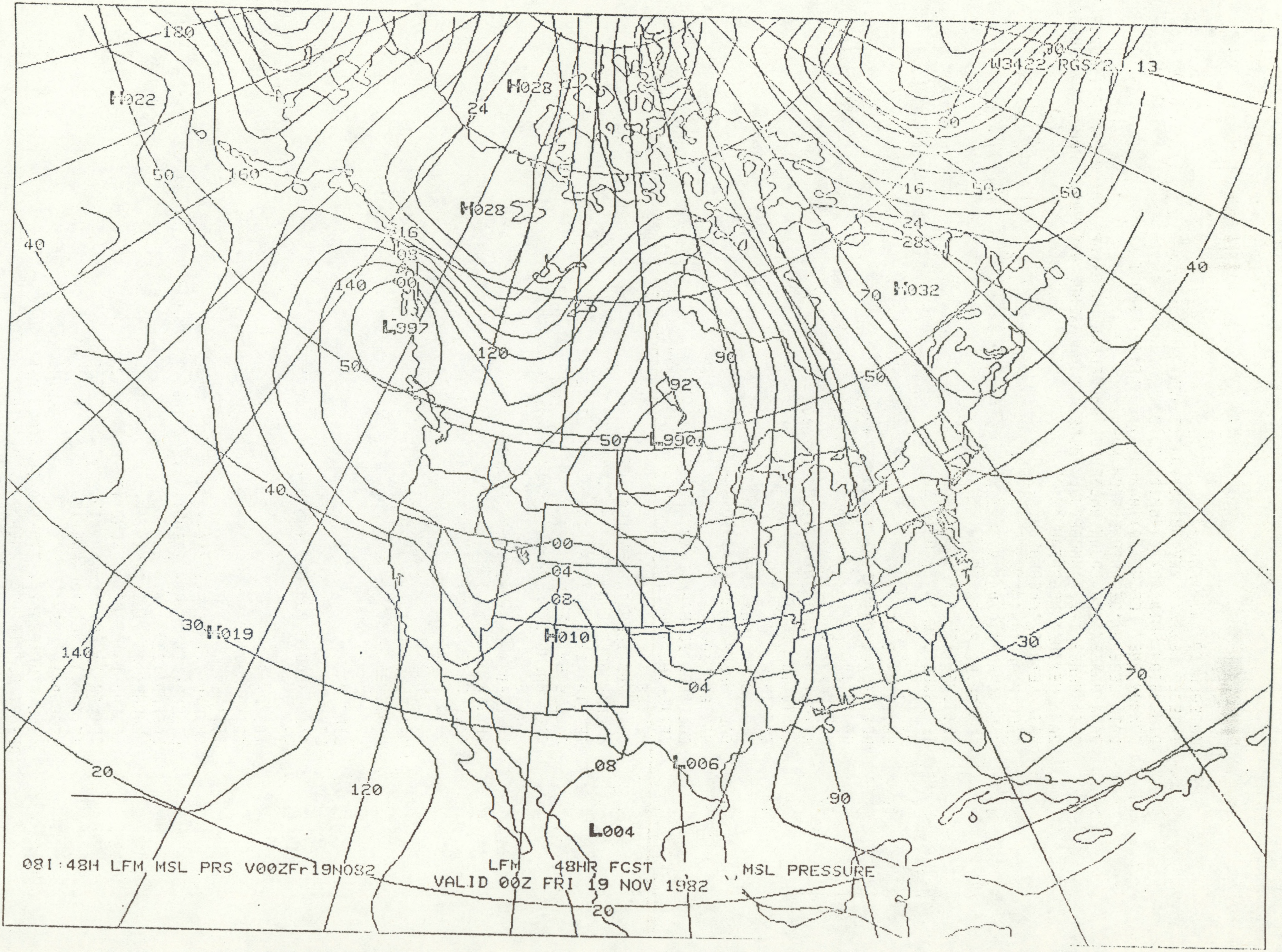
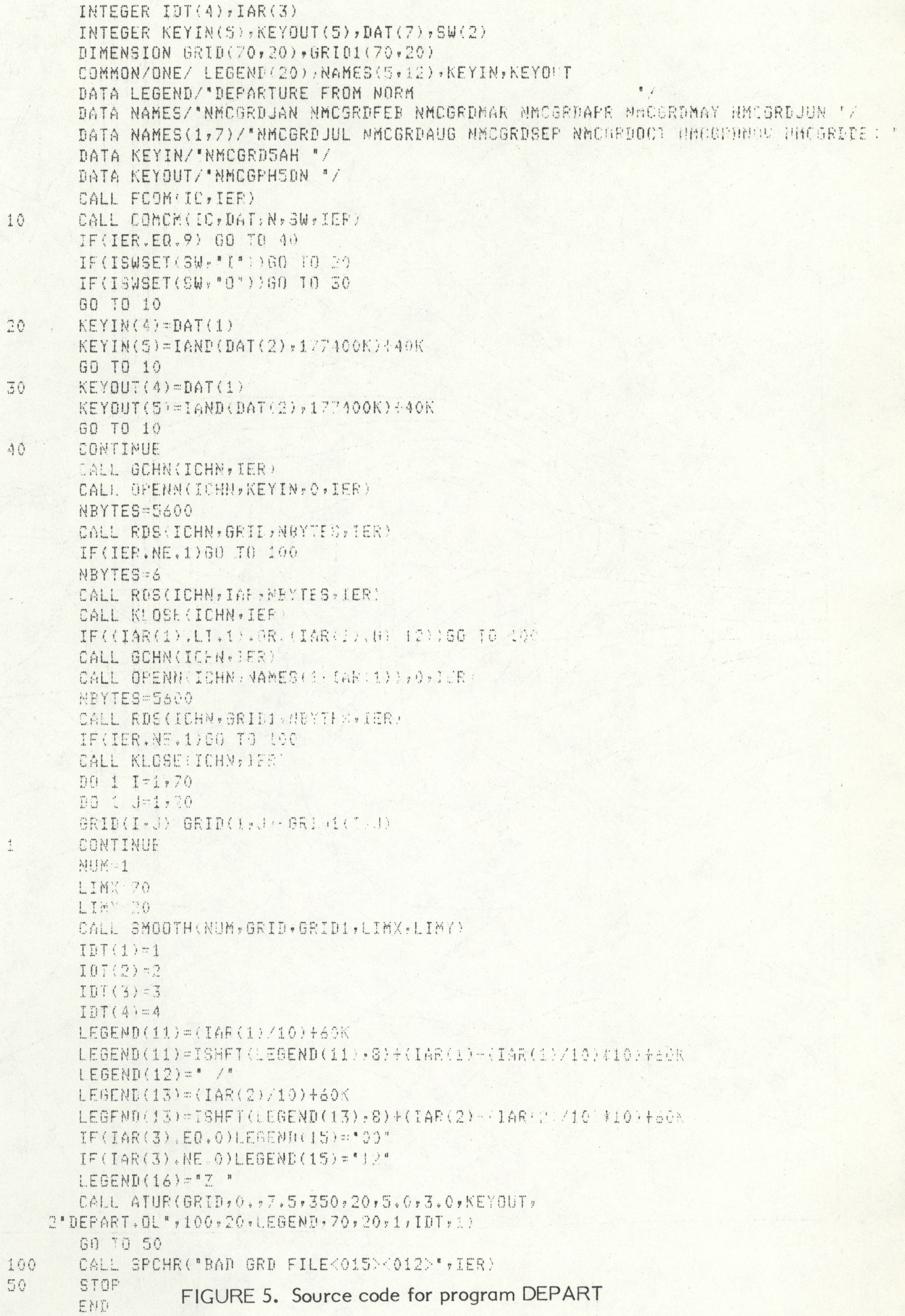


FIGURE 4. LFM surface analysis. Note that contours on the edges and one closed contour are not labeled



```
      INTEGER IDT(4),IAR(3)
      INTEGER KEYIN(5),KEYOUT(5),DAT(7),SW(2)
      DIMENSION GRID(70,20),GRID1(70,20)
      COMMON/ONE/ LEGEND(20),NAMES(5,12),KEYIN,KEYOUT
      DATA LEGEND/'DEPARTURE FROM NORM                    '/
      DATA NAMES/'NMCGRDJAN NMCGRDFEB NMCGRDMAR NMCGRDAPR NmCGRDMAY NMCGRDJUN '/
      DATA NAMES(1,7)/'NMCGRDJUL NMCGRDAUG NMCGRDSEP NMCGRDOCT NMCGRDNOV NMCGRDDEC '
      DATA KEYIN/'NMCGRD5AH '/
      DATA KEYOUT/'NMCGPH5DN '/
      CALL FCOM(IC,IER)
10    CALL COMCK(IC,DAT,N,SW,IER)
      IF(IER.EQ.9) GO TO 40
      IF(ISWSET(SW,"I"))GO TO 20
      IF(ISWSET(SW,"O"))GO TO 30
      GO TO 10
20    KEYIN(4)=DAT(1)
      KEYIN(5)=IAND(DAT(2),177400K)+40K
      GO TO 10
30    KEYOUT(4)=DAT(1)
      KEYOUT(5)=IAND(DAT(2),177400K)+40K
      GO TO 10
40    CONTINUE
      CALL GCHN(ICHN,IER)
      CALL OPENN(ICHN,KEYIN,0,IER)
      NBYTES=5600
      CALL RDS(ICHN,GRID,NBYTES,IER)
      IF(IER.NE.1)GO TO 100
      NBYTES=6
      CALL RDS(ICHN,IAR,NBYTES,IER)
      CALL KLOSE(ICHN,IER)
      IF((IAR(1).LT.1).OR.(IAR(1).GT.12))GO TO 100
      CALL GCHN(ICHN,IER)
      CALL OPENN(ICHN,NAMES(1,IAR(1)),0,IER)
      NBYTES=5600
      CALL RDS(ICHN,GRID1,NBYTES,IER)
      IF(IER.NE.1)GO TO 100
      CALL KLOSE(ICHN,IER)
      DO 1 I=1,70
      DO 1 J=1,20
      GRID(I,J)=GRID(I,J)-GRID1(I,J)
1     CONTINUE
      NUM=1
      LIMX=70
      LIMY=20
      CALL SMOOTH(NUM,GRID,GRID1,LIMX,LIMY)
      IDT(1)=1
      IDT(2)=2
      IDT(3)=3
      IDT(4)=4
      LEGEND(11)=(IAR(1)/10)+60K
      LEGEND(11)=ISHFT(LEGEND(11),8)+(IAR(1)-(IAR(1)/10)*10)+60K
      LEGEND(12)=" /"
      LEGEND(13)=(IAR(2)/10)+60K
      LEGEND(13)=ISHFT(LEGEND(13),8)+(IAR(2)-(IAR(2)/10)*10)+60K
      IF(IAR(3).EQ.0)LEGEND(15)="00"
      IF(IAR(3).NE.0)LEGEND(15)="12"
      LEGEND(16)="Z "
      CALL ATUR(GRID,0.,7.5,350,20,5.0,3.0,KEYOUT,
     2"DEPART.OL",100,20,LEGEND,70,20,1,IDT,1)
      GO TO 50
100   CALL SPCHR("BAD GRD FILE<015><012>",IER)
50    STOP
      END
```

FIGURE 5. Source code for program DEPART

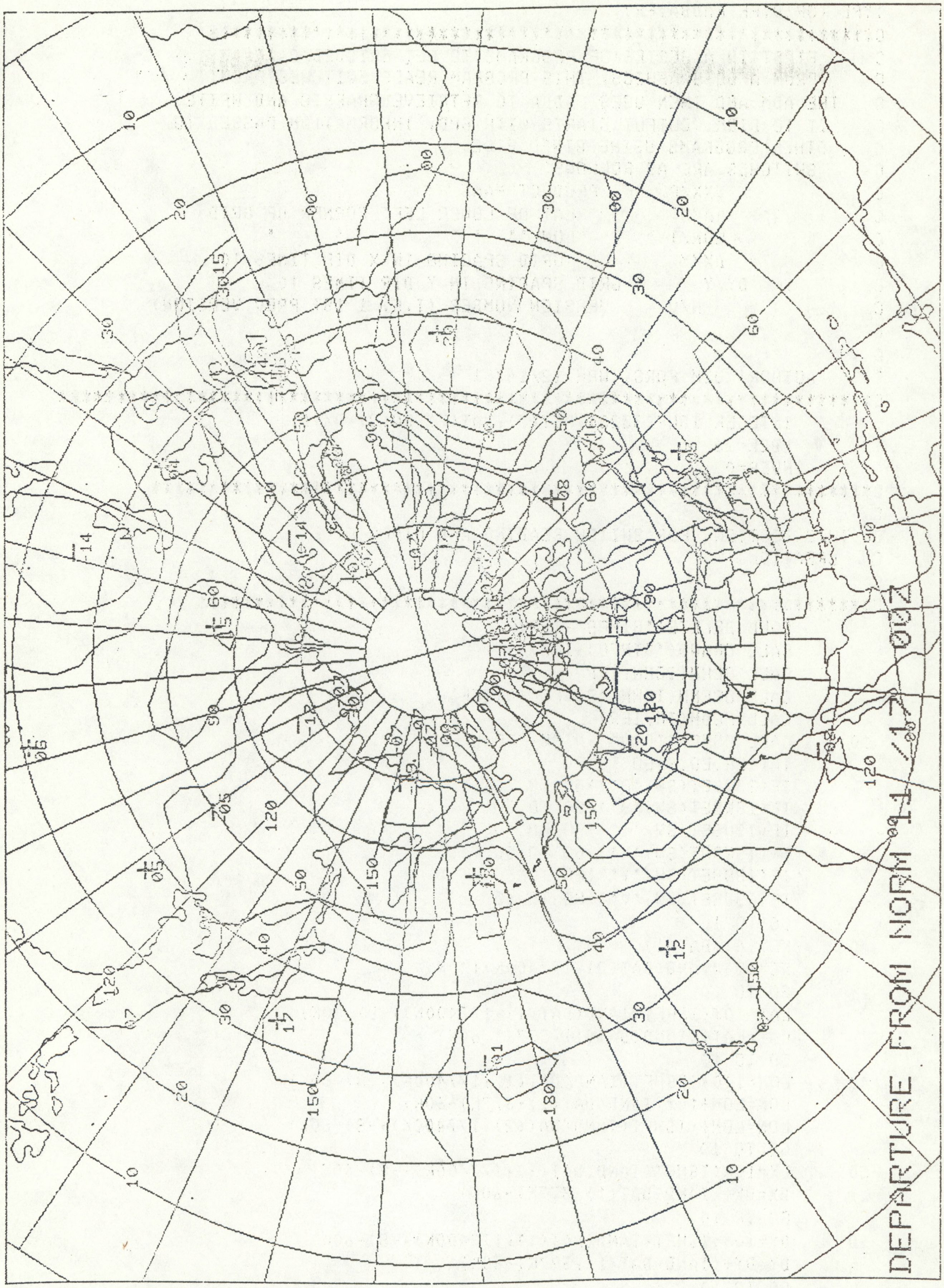


FIGURE 6. Sample departure from normal chart produced by program DEPART

V. Complete Program Listing:

```
TYPE (GRID.FR,FADBA.FR)
C**********************************************************
C     FIRST IN A SERIES OF PROGRAMS TO GET GRIDDED DATA
C     FROM AFOS GRAPHICS. THIS PROGRAM READS SWITCHES FROM
C   THE ADM AND THEN USES FADBA TO RETRIEVE GRAPHIC AND WRITE
C    IT TO DISK. OUTPUT STARTS WITH GPD. INFORMATION PASSED TO
C    OTHER PROGRAMS USING GINFO FILE.
C     SWITCHES ARE AS FOLLOWS:
C             XXX/P      PRODUCT NAME
C             LA/J          LAT OF LOWER LEFT CORNER OF GRID
C             LON/I         LON ""  " "   "     "     "    "
C             DX/X           GRID SPACING IN X DIR TIMES 10
C           DY/Y       GRID SPACING IN Y DIR TIMES 10
C               N/V     VERSION NUMBER (I.E. 1 1ST PREV VERSION)
C
C
C     AUTHOR: JIM FORS, WRH 12/14/81
C************************************************************************
        INTEGER IBUF(3328),KEY(5),DAT(7),SW(2),DX,DY
        NBLK=26
        NVER=0
C****************************************************************
C
C  THIS SECTION  FOR SWITCH READING AND GINFO FILE
C   CREATION.
C
C************************************************************
        CALL DELETE("GINFO",IER)
        CALL CRAND("GINFO",IER)
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"GINFO",0,IER)
        CALL FCOM(IC,IER)
 10     CALL COMCM(IC,DAT,N,SW,IER)
        IF(IER.EQ.9)GO TO 70
        IF(ISWSET(SW,"P"))GO TO 20
        IF(ISWSET(SW,"J"))GO TO 30
        IF(ISWSET(SW,"I"))GO TO 40
        IF(ISWSET(SW,"X"))GO TO 50
        IF(ISWSET(SW,"Y"))GO TO 60
        IF(ISWSET(SW,"V"))GO TO 65
        GO TO 10
 20     KEY(4)=DAT(1)
        KEY(5)=IAND(DAT(2),177400K)+40K
        GO TO 10
 30     LAT=10*(ISHFT(IAND(DAT(1),177400K),-8)-60K)
        LAT=LAT+(IAND(DAT(1),377K)-60K)
        GO TO 10
 40     LON=100*(ISHFT(IAND(DAT(1),177400K),-8)-60K)
        LON=LON+10*(IAND(DAT(1),377K)-60K)
        LON=LON+(ISHFT(IAND(DAT(2),177400K),-8)-60K)
        GO TO 10
 50     DX=10*(ISHFT(IAND(DAT(1),177400K),-8)-60K)
        DX=DX+(IAND(DAT(1),377K)-60K)
        GO TO 10
 60     DY=10*(ISHFT(IAND(DAT(1),177400K),-8)-60K)
        DY=DY+(IAND(DAT(1),377K)-60K)
        GO TO 10
 65     NVER=(ISHFT(DAT(1),-8)-60K)
        GO TO 10
 70     CALL WRS(ICHN,KEY(4),4,IER)
        CALL WRS(ICHN,LAT,2,IER)
        CALL WRS(ICHN,LON,2,IER)
        CALL WRS(ICHN,DX,2,IER)
        CALL WRS(ICHN,DY,2,IER)
C  NOW PULL THE PRODUCT OUT OF THE DATABASE AND WRITE  IT TO
C    A GRD FILE WITH THE SAME XXX.
        KEY(1)="NM"
        KEY(2)="CG"
        KEY(3)="PH"
        CALL FADBA(KEY,IBUF,NBLK,NVER,MB,IER)
        IF(IER.NE.-2)CALL SPCHR('FADBA ERROR',IERR)
        IF(IER.NE.-2)STOP
        CALL WRS(ICHN,MB,2,IER)
        CALL KLOSE(ICHN,IER)
        KEY(3)="RD"
        CALL CRAND(KEY,IER)
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,KEY,0,IER)
        IBLK=0
        NB=NBLK/2+1
        CALL WRB(ICHN,IBUF,IBLK,NB,IER)
        IF(IER.NE.1)CALL SPCHR("OUTPUT ERROR",IERR)
        IF(IER.NE.1)STOP
        CALL KLOSE(ICHN,IER)
        CALL CHAIN("GRID1.SV",IER)
        STOP
        END
      SUBROUTINE FADBA(KEY,ADBUF,NB,VN,MB,IER)

C TITLE:  FADBA,  "FORTRAN AFOS DATA BASE ACCESS"
C AUTHOR: ALEXANDER E. MACDONALD, WESTER REGION SSD, FTS 588-5131
C PURPOSE: FEEDS "NB" BLOCKS OF VERSION NUMBER "VN" OF AFOS KEY "KEY"
C                INTO AFOS DATA BUFFER "ADBUF"
C NOTE: USE OF THIS ROUTINE REQUIRES LOADING OF BG.LB


      DIMENSION KEY(5),ADBUF(1),KREC(40),IBUF(128)
      INTEGER ADBUF,VN,BN


        NBYTE=0
        INB=NB
        NB=0
      CALL KSRCF(KEY,KREC,IER)
        MB=IAND(KREC(12),7K)
C       IF(IER.NE.1) TYPE "KEY NOT FOUND"
        IF(IER.NE.1)RETURN
C NOW MOVE TO CORRECT VERSION NUMBER
      IF(VN.EQ.0) GOTO 95

      DO 90 NV=1,VN
       CALL PRVRF(IER)

C       IF(IER.NE.1) TYPE "PREVIOUS VERSION NOT FOUND"
        IF(IER.NE.1)RETURN

 90   CONTINUE
 95   CONTINUE
      IBLK=0
```

```
C READ FIRST BLOCK OF AFOS DATA INTO IBUF
    CALL NXBKF(IBUF,IER)

    IF(IER.NE.1)RETURN
C LOAD DATA FROM IBUF INTO AFOS DATA BUFFER
    NB=1
   DO 110 I=18,128
    NBYTE=NBYTE+1
   ADBUF(NBYTE)=IBUF(I)
 110   CONTINUE

   IF (INB.EQ.1)GOTO 40
C READ REMAINING BLOCKS INTO AFOS DATA BUFFER

   DO 120 BN=2,INB
   CALL NXBKF(IBUF,IER)

C     IF(IER.NE.1) TYPE 'NXBKF ERROR',IER
    IF(IER.NE.1) IER=-2
    IF(IER.NE.1)RETURN

    NB=NB+1
C     IF(NB.GT.INB)TYPE 'BUFFER TOO SMALL'
   DO 130 I=3,128
    NBYTE=NBYTE+1
    ADBUF(NBYTE)=IBUF(I)

 130   CONTINUF
 120   CONTINUE

 40      CONTINUE

   RETURN
   END
```

```
TYPE (GRID1.FR,CLEANUP.FR,VECDEC.FR,TEXTDEC.FR,INTCVT.FR)
C*****************************************************
C
C   THIS IS SECOND PROGRAM IN AFOS GRAPHICS TO GRID
C   POINT DATA SERIES. THIS ROUTINE READS GRAPHIC FILE
C   CREATED IN FIRST ROUTINE WITH INFORMATION FROM GINFO
C   FILE. SUBROUTINE CLEANUP STRIPS OUT GPD, UTF FORMAT AND
C   EVERYTHING AFTER 203 BYTE. SUBROUTINE VECDEC CREATES A FILE
C   OF  POINTS FOR EACH VECTOR. SUBROUTINE TEXTDEC
C   CREATES A FILE OF  LOCATIONS AND LABEL VALUES.
C
C   AUTHOR; JIM FORS 12/14/81
C
C***************************************************************
        INTEGER IBUF(3328),KEY(5)
        COMMON/ONE/ IBUF,ICHN
C*****************************************************
C
C     THIS SECTION FOR READING OF GINFO FILE
C
C*****************************************************
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"GINFO",0,IER)
        CALL RDS(ICHN,KEY(4),4,IER)
        CALL KLOSE(ICHN,IER)
        KEY(1)="NH"
        KEY(2)="CG"
        KEY(3)="RD"
        CALL GCHN(ICHN,IER)
        N=0
        CALL OPENN(ICHN,KEY,0,IER)
        CALL CLEANUP(N)
        CALL KLOSE(ICHN,IER)
        NMAX=N
        CALL VECDEC(N)
        CALL TEXTDEC(N,NMAX,IER)
        IF(IER.NE.1)CALL SPCHR("TEXT ERROR",IERR)
        IF(IER.NE.1)STOP
        CALL CHAIN("GRID2.SV",IER)
        STOP
        END
        SUBROUTINE CLEANUP(IWORD)
C************************************************
C
C     SUBROUTINE: CLEANUP        DATE: 12/14/81
C      AUTHOR    : JIM FORS, WRH
C
C     SUBROUTINE READS IN THREE BLOCKS OF GRAPHIC FILE
C       AT A TIME THEN PROCESSES TWO BLOCKS TO REMOVE
C       ALL UTF FORMAT AND ANY JUNK AFTER 203 BYTE.
C       THREE BLOCKS ARE READ TO GIVE THE SLOP OVER SPACE
C       FOR CHECKING UTF AT BLOCK BOUNDARY.
C
C**************************************************
        INTEGER IBUF(3328),LBUF(1024),JBUF(1280)
        COMMON/ONE/ IBUF,ICHN
        COMMON/TWO/ JBUF,LBUF
        N=0
        IFLG=0
        IBLK=0
 50     NBLK=3
        CALL RDB(ICHN,LBUF,IBLK,NBLK,IER)
        IBLK=IBLK+2
        CALL UNPACK(LBUF,1026,JBUF)
        DO 1 I=1,1024
        IF((IBLK.EQ.2).AND.(I.EQ.1).AND.(JBUF(I).EQ.0K))GO TO 1
        IF((IBLK.EQ.2).AND.(I.EQ.2).AND.(JBUF(I).EQ.0K))GO TO 1
        IF(IFLG.EQ.1)GO TO 5
        IF(JBUF(I).EQ. 203K) GO TO 500
        IF(JBUF(I).EQ.20K)IFLG=1
        IF((JBUF(I).EQ.20K).AND.(JBUF(I+1).EQ.14K))JBUF(I)=203K
        N=N+1
        IWORD=((N-1)/2)+1
        LFLG=(N-(N/2)*2)
        IF(LFLG.EQ.0)IBUF(IWORD)=IBUF(IWORD)+JBUF(I)
        IF(LFLG.EQ.1)IBUF(IWORD)=ISHFT(JBUF(I),8)
        GO TO 1
 5      IFLG=0
 1      CONTINUE
        GO TO 50
 500    CONTINUE
        RETURN
        END
```

```
      SUBROUTINE VECDEC(N)
C*************************************************
C
C     SUBROUTINE: VECDEC       DATE; 12/14/81
C     AUTHOR: JIM FORS, WRH
C
C     SUBROUTINE TAKES CLEANUED UP VECTOR FILE AND DECODES
C     THE VECTOR INFORMATION. IT RETURNS WHEN IT FINDS THE
C     FIRST TEXT STRING. NMC GRAPHICS PUT ALL TEXT AT THE END.
C     ONE PART OF THE SUBROUTINE IS USED FOR SINGLE WORD
C     DELTA X DELTA Y AND ANOTHER FOR TWO WORD FORMAT.
C     OUTPUT IS IN FILE VECTOR. A PAIR OF MINUS ONES
C     SEPARATES EACH VECTOR STRING. A PAIR OF MINUS TWOS
C     INDICATES THE END OF FILE.
C
C***********************************************************
      INTEGER IBUF(3328),ISTART(2)
      COMMON/ONE/IBUF,ICHN
      JFLG=0
      N=0
      MINUS2=-2
      MINUS=-1
      IMASK=100000K
      NMASK=10000K
      LMASK=7777K
      LLMASK=40000K
      LLLMASK=37400K
      LRMASK=100K
      LRRMASK=77K
      CALL CRAND("VECTOR",IER)
      CALL GCHN(ICHN,IER)
      CALL OPENN(ICHN,"VECTOR",0,IER)
 100  CALL WRS(ICHN,MINUS,2,IER)
      CALL WRS(ICHN,MINUS,2,IER)
 101  N=N+1
      IF(IBUF(N).NE.141400K)GO TO 500
      N=N+1
      ISTART(1)=2*IBUF(N)
      ISTART(2)=2*IBUF(N+1)
      CALL WRS(ICHN,ISTART,4,IER)
      N=N+2
      ICNT=IBUF(N)
      DO 1 I=1,ICNT
      N=N+1
      ICHK=IAND(IBUF(N),IMASK)
      IF(ICHK.EQ.IMASK)GO TO 150
C*******THIS SECTION IS FOR TWO WORD FORMAT
      JSGN=IAND(IBUF(N),NMASK)
      IOFF=IAND(IBUF(N),LMASK)
      IF(JSGN.GT.0)IOFF=(IOFF-LMASK)-1
      IF(JFLG.EQ.1)GO TO 145
      ISTART(1)=ISTART(1)+2*IOFF
      JFLG=1
      GO TO 1
 145  ISTART(2)=ISTART(2)+2*IOFF
      CALL WRS(ICHN,ISTART,4,IER)
      JFLG=0
      GO TO 1
 150  CONTINUE
```

```
C**********THIS SECTION IS FOR ONE WORD FORMAT
      JSGN=IAND(IBUF(N),LLMASK)
      IOFF=IAND(IBUF(N),LLLMASK)
      IOFF=ISHFT(IOFF,-8)
      IF(JSGN.GT.0)IOFF=(IOFF-LRRMASK)-1
      ISTART(1)=ISTART(1)+2*IOFF
      JSGN=IAND(IBUF(N),LRMASK)
      JOFF=IAND(IBUF(N),LRRMASK)
      IF(JSGN.GT.0)JOFF=(JOFF-LRRMASK)-1
      ISTART(2)=ISTART(2)+2*JOFF
      CALL WRS(ICHN,ISTART,4,IER)
 1    CONTINUE
      GO TO 100
 500  IF(N.EQ.1)GO TO 101
      CALL WRS(ICHN,MINUS2,2,IER)
      CALL WRS(ICHN,MINUS2,2,IER)
      CALL KLOSE(ICHN,IER)
      RETURN
      END
      SUBROUTINE TEXTDEC(N,NMAX,IER)
C***********************************************************
C
C     SUBROUTINE: TEXTDEC      DATE: 12/14/81
C     AUTHOR: JIM FORS, WRH
C
C     SUBROUTINE  PICKS UP IN CLEANEDUP FILE  WHERE
C     VECDEC LEAVES OFF.  IF THERE ARE MORE THAN 1152
C     WORDS OF TEXT INFORMATIONTHIS ROUTINE WILL
C     RETURN AN ERROR CODE OF -1.  ROUTINE FIRST LOOKS
C     FOR A START OF TEXT..310K BYTES.. AND THEN
C     GETS I AND J LOCATIONS FOR LABEL. IT THEN SEARCHES
C     STRING TO LOOK FOR A SIGN OR A NUMBER. IT THEN
C     CONVERTS THAT TO AN INTEGER VALUE WITH A FUNCTION
C     INTCVT. OUTPUT IS IN A FILE CALLED TEXT. EACH
C     RECORD CONSISTS OF 3 WORDS. THE END OF FILE IS THREE
C     MINUS ONES.
C
C*****************************************************************
      INTEGER IBUF(3328),JBUF(2304),IVAL(3),IDATE(3)
      COMMON/ONE/ IBUF,ICHN
      COMMON/TWO/ JBUF
      MINUS=-1
      ISFC=0
      NBYTES=(NMAX-N+1)*2
      IF(NBYTES.GT.2304)IER=-1
      IF(NBYTES.GT.2304)RETURN
      CALL UNPACK(IBUF(N),NBYTES,JBUF)
      NFLG=0
      CALL CRAND('TEXT',IER)
      CALL GCHN(ICHN,IER)
      CALL OPENN(ICHN,'TEXT',0,IER)
      N=1
 101  IF(JBUF(N).EQ.310K)GO TO 102
      IF(NFLG.EQ.1)GO TO 210
C*********SEARCH FOR SFC OR RH CHART LABEL TO SET SPECIAL FLAG
C************THAT IS NEEDED LATER TO EVALUATE LABELS
      IF((JBUF(N).EQ.115K).AND.(JBUF(N+1).EQ.123K).AND.(JBUF(N+2).EQ.114K)
     2.AND.(JBUF(N+3).EQ.40K))ISFC=1
      IF((JBUF(N).EQ.122K).AND.(JBUF(N+1).EQ.105K).AND.(JBUF(N+2).EQ.114K)
     2.AND.(JBUF(N+3).EQ.40K))ISFC=2
C**********LOOK FOR 00Z OR 12Z TO LOCATE POINTER TO DECODE
C*************LEGEND TO GET DATE TIME INFORMATION
      IF((JBUF(N).EQ.60K).AND.(JBUF(N+1).EQ.60K).AND.
     2(JBUF(N+2).EQ.132K))GO TO 200
      IF((JBUF(N).EQ.61K).AND.(JBUF(N+1).EQ.62K).AND.
     2(JBUF(N+2).EQ.132K))GO TO 201
 210  N=N+1
      IF(N.GT.NBYTES)GO TO 500
      GO TO 101
 102    N=N+2
        IF(N.GT.NBYTES)GO TO 500
        NFLG=1
        IVAL(1)=ISHFT(JBUF(N),8)+JBUF(N+1)
        IVAL(2)=ISHFT(JBUF(N+2),8)+JBUF(N+3)
        IVAL(1)=2*IVAL(1)
        IVAL(2)=2*IVAL(2)
        IFLG=0
        N=N+4
        IF(N.GT.NBYTES)GO TO 500
 103    IF((JBUF(N).EQ.53K).OR.(JBUF(N).EQ.55K))IFLG=1
        IF((JBUF(N).GE.60K).AND.(JBUF(N).LE.71K))IFLG=1
        IF((JBUF(N).EQ.127K).AND.(JBUF(N+1).EQ.63K))GO TO 101
        IF(IFLG.EQ.1)GO TO 104
        N=N+1
        IF(N.GT.NBYTES)GO TO 500
        GO TO 103
 104    IVAL(3)=INTCVT(N,4)
        CALL WRS(ICHN,IVAL,6,IER)
        N=N+1
        IF(N.GT.NBYTES)GO TO 500
        GO TO 101
 500    CALL WRS(ICHN,MINUS,2,IER)
        CALL WRS(ICHN,MINUS,2,IER)
        CALL WRS(ICHN,MINUS,2,IER)
        CALL KLOSE(ICHN,IER)
        CALL GCHN(ICHN,IER)
        CALL OPENA(ICHN,'GINFO',0,IER)
        CALL WRS(ICHN,ISFC,2,IER)
        CALL WRS(ICHN,IDATE,6,IER)
        CALL KLOSE(ICHN,IER)
        RETURN
C***********DECODE DATE TIME INFORMATION IN HEADER LEGEND
 200    IDATE(1)=0
        GO TO 202
 201    IDATE(1)=1200
C*************DECODE DAY
 202    IDATE(2)=10*(JBUF(N+8)-60K)+(JBUF(N+9)-60K)
C*********GET THE MONTH
        IF((JBUF(N+12).EQ.101K).AND.(JBUF(N+13).EQ.116K))IDATE(3)=1
        IF((JBUF(N+12).EQ.105K).AND.(JBUF(N+13).EQ.102K))IDATE(3)=2
        IF((JBUF(N+12).EQ.101K).AND.(JBUF(N+13).EQ.122K))IDATE(3)=3
        IF((JBUF(N+12).EQ.120K).AND.(JBUF(N+13).EQ.122K))IDATE(3)=4
        IF((JBUF(N+12).EQ.101K).AND.(JBUF(N+13).EQ.131K))IDATE(3)=5
        IF((JBUF(N+12).EQ.125K).AND.(JBUF(N+13).EQ.116K))IDATE(3)=6
        IF((JBUF(N+12).EQ.125K).AND.(JBUF(N+13).EQ.114K))IDATE(3)=7
        IF((JBUF(N+12).EQ.125K).AND.(JBUF(N+13).EQ.107K))IDATE(3)=8
        IF((JBUF(N+12).EQ.105K).AND.(JBUF(N+13).EQ.120K))IDATE(3)=9
        IF((JBUF(N+12).EQ.103K).AND.(JBUF(N+13).EQ.124K))IDATE(3)=10
        IF((JBUF(N+12).EQ.117K).AND.(JBUF(N+13).EQ.126K))IDATE(3)=11
        IF((JBUF(N+12).EQ.105K).AND.(JBUF(N+13).EQ.103K))IDATE(3)=12
        GO TO 210
        END
```

```
      INTEGER FUNCTION INTCVT(IBGN,N)
C*****************************************************
C
C     FUNCTION: INTCVT            AUTHOR: MATT PEROUTKA, GTF WSFO
C
C     TAKES A STRING OF UNPACKED CHARACTERS AND RETURNS AN
C     INTEGER VALUE.
C
C*********************************************************
      COMMON/TWO/ IOUTU(2304)
      LOGICAL NEG
      INTCVT=0
      NEG=.FALSE.
      IEND=IBGN+N-1
100   IF(IOUTU(IEND).NE.32)GO TO 200
      IF(IEND.EQ.IBGN) RETURN
      IEND=IEND-1
      GO TO 100
200   DO 250 I=IBGN,IEND
      IF(IOUTU(I).NE.32)GO TO 300
250   CONTINUE
      RETURN
300   IF(IOUTU(I).EQ.43)GO TO 400
      IF(IOUTU(I).NE.45)GO TO 500
      NEG=.TRUE.
400   I=I+1
500   J=I
      DO 600 I=J,IEND
      IF(IOUTU(I).EQ.32)IOUTU(I)=48
      IF(IOUTU(I).LT.48.OR.IOUTU(I).GT.57)GO TO 800
      INTCVT=INTCVT*10+IOUTU(I)-48
600   CONTINUE
800   IF(NEG) INTCVT=-INTCVT
      RETURN
      END
```

```
TYPE GRID2.FR
C*********************************************************
C
C       THIS IS THIRD PROGRAM IN A SERIES TO CONVERT AFOS
C       GRAPHICS TO GRID POINTS. THIS ROUTINE IS DESIGNED
C       TO CORRELATE LABEL VALUES IN THE TEXT FILE WITH THE
C       VECTORS IN THE VECTOR FILE. THE OUTPUT IS IN FILE
C       COMPAR. A ONE WORD VALUE IS WRITTEN FOR EACH VECTOR.
C       A VECTOR FOR WHICH NO MATCH IS FOUND IS INDICATED
C       BY A MINUS ONE. THE ALGORITHM USED IS TO FIND
C       THE STRAIGHT LINE GOING THROUGH TWO POINTS IN THE
C       VECTOR AND  USING THAT EQUATION PLUS THE I
C       COORDINATE OF THE LABEL TO FORECAST A J COORDINATE
C       OF THE LABEL. IF THE FORECAST VALUE AND THE ACTUAL
C       J VALUE ARE WITHIN 16 PIXELS..THAT LABEL IS ASSIGNED TO
C       THAT VECTOR.
C
C       AUTHOR: JIM FORS, WRH 12/23/81
C
C************************************************************
        INTEGER IBUF(768),JBUF(4096)
        CALL CRAND("COMPAR",IER)
        CALL GCHN(KCHN,IER)
        CALL OPENN(KCHN,"COMPAR",0,IER)
        CALL GCHN(JCHN,IER)
        CALL OPENN(JCHN,"VECTOR",0,IER)
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"TEXT",0,IER)
        IBLK=0
        NBLK=3
        CALL RDB(ICHN,IBUF,IBLK,NBLK,IER)
        IBLK=0
        NBLK=16
        CALL RDB(JCHN,JBUF,IBLK,NBLK,IER)
        N=-1
        L=-2
 100    N=N+2
        IF(N.GT.4094)GO TO 550
        IF(JBUF(N).NE.-1)GO TO 100
 110    N=N+2
        IF(N.GT.4094)GO TO 550
        IF(JBUF(N).EQ.-2)GO TO 500
        IF(JBUF(N-2).EQ.-1)GO TO 210
        IF(JBUF(N).EQ.-1)CALL WRS(KCHN,-1,2,IER)
        IF(JBUF(N).EQ.-1)GO TO 200
 160    L=L+3
        IF(IBUF(L).EQ.-1)GO TO 200
        IFLG=0
```

```
C*********CHECK TO SEE IF THE I VALUE OF THE LABEL LIES BETWEEN THE TWO
C************ I VALUESOF THE TWO POINTS ON THE VECTOR. IF NOT TRY ANOTHER
C*********** PAIR.
        IF(((JBUF(N)-JBUF(N-2)).GE.0).AND.(IBUF(L).GE.JBUF(N-2)).AND.(IBUF(L).LE.JBUF(N)))IFLG=1
        IF(((JBUF(N-2)-JBUF(N)).GE.0).AND.(IBUF(L).GE.JBUF(N)).AND.(IBUF(L).LE.JBUF(N-2)))IFLG=1
        IF(IFLG.EQ.0)GO TO 160
C******* NOW GET THE STRAIGHT LINE..FORECAST THE J VALUE
C********** AND SEE IF IT LIES WITHIN 20K PIXELS. I F IT DOES
C********* WRITE IT TO THE COMPAR FILE.
        LLL=JBUF(N)-JBUF(N-2)
        XLL=LLL
        IF(LLL.EQ.0)XLL=1.
        YLL=JBUF(N+1)-JBUF(N-1)
        ZLL=IBUF(L)-JBUF(N-2)
        XGUESS=(YLL/XLL)*ZLL+JBUF(N-1)
        IF(ABS(XGUESS-IBUF(L+1)).GT.20K)GO TO 160
        CALL WRS(KCHN,IBUF(L+2),2,IER)
        L=-2
        GO TO 100
200     L=-2
210     GO TO 110
500     CONTINUE
        CALL RESET
        CALL CHAIN("GRID3.SV",IER)
550     CALL SPCHR("NO END OF VECTOR FILE<015><012>",IER)
        STOP
        END
```

```
TYPE (GRID3.MC,HGRID3\
TYPE (GRID3.FR,HGRID3.FR,POINT.FR,GSM.FR)
        PARAMETER LIMX=26
        PARAMETER LIMY=22
        PARAMETER LIMX1=25
        PARAMETER LIMY1=21
        PARAMETER LIMX2=24
        PARAMETER LIMY2=20
C******************************************
C
C       THIS IS THE FINAL PROGRAM IN THE SERIES
C       TO CONVERT AFOS GRAHICS TO GRID POINT
C       DATA. THE VECTOR VALUES AS WELL AS THE LABEL
C       POINTS ARE USED AS OBSERVATIONS IN THE
C       OBJECTIVE ANALYSIS.
C
C       AUTHOR; JIM FORS, WRH 12/23/81
C
C**********************************************
        DIMENSION DD1(LIMX2,LIMY2)
        DIMENSION DD(0:LIMX1,0:LIMY1),GRID(0:LIMX1,0:LIMY1)
        INTEGER IBUF(2),JBUF(3),KEY(5),IDATE(3)
        REAL LATMIN,LATMAX,LONMIN,LONMAX
        COMMON/TWO/ LATMIN,LONMAX,IDX,IDY
        NUM=2
C******** READ THE GINFO FILE
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"GINFO",0,IER)
        CALL RDS(ICHN,KEY(4),4,IER)
        CALL RDS(ICHN,ILAT,2,IER)
        CALL RDS(ICHN,ILON,2,IER)
        CALL RDS(ICHN,IDX,2,IER)
        CALL RDS(ICHN,IDY,2,IER)
        CALL RDS(ICHN,MB,2,IER)
        CALL RDS(ICHN,ISFC,2,IER)
        CALL RDS(ICHN,IDATE,6,IER)
        CALL KLOSE(ICHN,IER)
C******* CALULATE THE LATMIN,LATMAX,LONMIN, LONMAX
        LATMIN=ILAT-1.*(IDY/10.)
        LATMAX=ILAT+(LIMY2)*(IDY/10.)
        LONMAX=ILON+1.*(IDX/10.)
        LONMIN=ILON-(LIMX2)*(IDX/10.)
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"VECTOR",0,IER)
        CALL GCHN(JCHN,IER)
        CALL OPENN(JCHN,"COMPAR",0,IER)
        CALL RDS(ICHN,IBUF,4,IER)
C*******INITIALIZE THE ARRAY TO ALL POINTS NOT FILLED.
        DO 1 I=0,LIMY1
        DO 1 J=0,LIMX1
        GRID(J,I)=-77777.
  1     CONTINUE
```

```
C********READ IN THE VALUE FOR THE FIRST VECTOR STRING
 100    NBYTES=2
        CALL RDS(JCHN,IVAL,NBYTES,IER)
        IF(IER.EQ.9)GO TO 300
        IF(IVAL.EQ.-1)GO TO 150
        IF((ISFC.EQ.2).AND.(IVAL.LT.10))IVAL=IVAL*10
        IF((ISFC.EQ.1).AND.(IVAL.GE.50).AND.(IVAL.LT.900))IVAL=IVAL+900
        IF((ISFC.EQ.1).AND.(IVAL.LT.50))IVAL=IVAL+1000
 150    NBYTES=4
C****** READ IN THE I J COORDINATES OF THE VECTOR WHOSE VALUE
C******** WAS DETERMINED IN THE PREVIOUS READ.
        CALL RDS(ICHN,IBUF,NBYTES,IER)
        IF(IBUF(1).EQ.-2)GO TO 300
        IF(IBUF(1).EQ.-1)GO TO 100
        IF(IVAL.EQ.-1)GO TO 150
        III=IBUF(1)/2
        JJJ=IBUF(2)/2
C******** CONVERT THE I J TO LAT AND LON
        CALL CLAT(III,JJJ,MB,XLAT,XLON)
C*******ONLY USE IT IF IT IS IN THE AREA OF INTEREST
        IF((XLAT.GT.LATMAX).OR.(XLAT.LT.LATMIN))GO TO 150
        IF((XLON.GT.LONMAX).OR.(XLON.LT.LONMIN))GO TO 150
        CALL POINT(XLAT,XLON,IVAL,GRID,LIMX,LIMY)
        GO TO 150
 300    CALL GCHN(KCHN,IER)
        CALL OPENN(KCHN,"TEXT",0,IER)
 350    NBYTES=6
C****** NOW USE THE LABELS AND MAX AND MIN CENTERS AS DATA
C******** POINTS
        CALL RDS(KCHN,JBUF,NBYTES,IER)
        IF(JBUF(1).EQ.-1)GO TO 500
        III=(JBUF(1)/2)+7
        JJJ=(JBUF(2)/2)+7
        CALL CLAT(III,JJJ,MB,XLAT,XLON)
        IF((XLAT.GT.LATMAX).OR.(XLAT.LT.LATMIN))GO TO 350
        IF((XLON.GT.LONMAX).OR.(XLON.LT.LONMIN))GO TO 350
        IF((ISFC.EQ.2).AND.(JBUF(3).LT.10))JBUF(3)=JBUF(3)*10
        IF((ISFC.EQ.1).AND.(JBUF(3).LT.50))JBUF(3)=JBUF(3)+1000
        IF((ISFC.EQ.1).AND.(JBUF(3).GE.50).AND.(JBUF(3).LT.900))JBUF(3)=JBUF(3)+900
        CALL POINT(XLAT,XLON,JBUF(3),GRID,LIMX,LIMY)
        GO TO 350
 500    CALL RESET
C******* FILL IN THE REMAINING POINTS THAT ARE NOT FILLED BY
C********* POINT. THEN THROW IN A SMOOTHING PASS FOR GOOD
C********* MEASURE.
        CALL UPHIL(GRID,LIMX,LIMY,DD,IER)
        CALL SMOOTH(NUM,GRID,DD,LIMX,LIMY)
        CALL DELETE("VECTOR",IER)
        CALL DELETE("TEXT",IER)
        CALL DELETE("COMPAR",IER)
C******* ONLY WRITE OUT THE GOOD STUFF IN THE MIDDLE OF THE GRID.
        DO 2 I=1,LIMX2
        DO 2 J=1,LIMY2
        DD1(I,J)=GRID(I,J)
 2      CONTINUE
```

```
KEY(1)="NM"
KEY(2)="CG"
KEY(3)="RD"
NBYTES=(LIMX2)*(LIMY2)*4
CALL GCHN(ICHN,IER)
CALL CRAND(KEY,IER)
CALL OPENN(ICHN,KEY,0,IER)
CALL WRS(ICHN,DD1,NBYTES,IER)
CALL WRS(ICHN,IDATE(3),2,IER)
CALL WRS(ICHN,IDATE(2),2,IER)
CALL WRS(ICHN,IDATE(1),2,IER)
CALL KLOSE(ICHN,IER)
STOP
END
```

```
        PARAMETER LIMX=72
        PARAMETER LIMY=22
        PARAMETER LIMX1=71
        PARAMETER LIMY1=21
        PARAMETER LIMX2=70
        PARAMETER LIMY2=20
C*********************************************
C
C       THIS IS THE FINAL PROGRAM IN THE SERIES
C       TO CONVERT AFOS GRAHICS TO GRID POINT
C       DATA. THE VECTOR VALUES AS WELL AS THE LABEL
C       POINTS ARE USED AS OBSERVATIONS IN THE
C       OBJECTIVE ANALYSIS.
C
C       AUTHOR; JIM FORS, WRH 12/23/81
C
C*************************************************
        DIMENSION DD1(LIMX2,LIMY2)
        DIMENSION DD(0:LIMX1,0:LIMY1),GRID(0:LIMX1,0:LIMY1)
        INTEGER IBUF(2),JBUF(3),KEY(5),IDATE(3)
        REAL LATMIN,LATMAX,LONMIN,LONMAX
        COMMON/TWO/ LATMIN,LONMAX,IDX,IDY
        NUM=1
C******** READ THE GINFO FILE
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"GINFO",0,IER)
        CALL RDS(ICHN,KEY(4),4,IER)
        CALL RDS(ICHN,ILAT,2,IER)
        CALL RDS(ICHN,ILON,2,IER)
        CALL RDS(ICHN,IDX,2,IER)
        CALL RDS(ICHN,IDY,2,IER)
        CALL RDS(ICHN,MB,2,IER)
        CALL RDS(ICHN,ISFC,2,IER)
        CALL RDS(ICHN,IDATE,6,IER)
        CALL KLOSE(ICHN,IER)
C******* CALULATE THE LATMIN,LATMAX,LONMIN, LONMAX
        LATMIN=ILAT-1.*(IDY/10.)
        LATMAX=ILAT+(LIMY2)*(IDY/10.)
        LONMAX=ILON+1.*(IDX/10.)
        LONMIN=ILON-(LIMX2)*(IDX/10.)
        CALL GCHN(ICHN,IER)
        CALL OPENN(ICHN,"VECTOR",0,IER)
        CALL GCHN(JCHN,IER)
        CALL OPENN(JCHN,"COMPAR",0,IER)
        CALL RDS(ICHN,IBUF,4,IER)
C*******INITIALIZE THE ARRAY TO ALL POINTS NOT FILLED.
        DO 1 I=0,LIMY1
        DO 1 J=0,LIMX1
        GRID(J,I)=-77777.
  1     CONTINUE
```

```
C********READ IN THE VALUE FOR THE FIRST VECTOR STRING
 100    NBYTES=2
        CALL RDS(JCHN,IVAL,NBYTES,IER)
        IF(IER.EQ.9)GO TO 300
        IF(IVAL.EQ.-1)GO TO 150
        IF((ISFC.EQ.2).AND.(IVAL.LT.10))IVAL=IVAL*10
        IF((ISFC.EQ.1).AND.(IVAL.GE.50).AND.(IVAL.LT.900))IVAL=IVAL+900
        IF((ISFC.EQ.1).AND.(IVAL.LT.50))IVAL=IVAL+1000
 150    NBYTES=4
C****** READ IN THE I J COORDINATES OF THE VECTOR WHOSE VALUE
C******** WAS DETERMINED IN THE PREVIOUS READ.
        CALL RDS(ICHN,IBUF,NBYTES,IER)
        IF(IBUF(1).EQ.-2)GO TO 300
        IF(IBUF(1).EQ.-1)GO TO 100
        IF(IVAL.EQ.-1)GO TO 150
        III=IBUF(1)/2
        JJJ=IBUF(2)/2
C******** CONVERT THE I J TO LAT AND LON
        CALL CLAT(III,JJJ,MB,XLAT,XLON)
C*******ONLY USE IT IF IT IS IN THE AREA OF INTEREST
        IF((XLAT.GT.LATMAX).OR.(XLAT.LT.LATMIN))GO TO 150
        IF((XLON.GT.LONMAX).OR.(XLON.LT.LONMIN))GO TO 150
        CALL POINT(XLAT,XLON,IVAL,GRID,LIMX,LIMY)
        GO TO 150
 300    CALL GCHN(KCHN,IER)
        CALL OPENN(KCHN,'TEXT',0,IER)
 350    NBYTES=6
C****** NOW USE THE LABELS AND MAX AND MIN CENTERS AS DATA
C******** POINTS
        CALL RDS(KCHN,JBUF,NBYTES,IER)
        IF(JBUF(1).EQ.-1)GO TO 500
        III=(JBUF(1)/2)+7
        JJJ=(JBUF(2)/2)+7
        CALL CLAT(III,JJJ,MB,XLAT,XLON)
        IF((XLAT.GT.LATMAX).OR.(XLAT.LT.LATMIN))GO TO 350
        IF((XLON.GT.LONMAX).OR.(XLON.LT.LONMIN))GO TO 350
        IF((ISFC.EQ.2).AND.(JBUF(3).LT.10))JBUF(3)=JBUF(3)*10
        IF((ISFC.EQ.1).AND.(JBUF(3).LT.50))JBUF(3)=JBUF(3)+1000
        IF((ISFC.EQ.1).AND.(JBUF(3).GE.50).AND.(JBUF(3).LT.900))JBUF(3)=JBUF(3)+900
        CALL POINT(XLAT,XLON,JBUF(3),GRID,LIMX,LIMY)
        GO TO 350
 500    CALL RESET
```

```
C******* FILL IN THE REMAINING POINTS THAT ARE NOT FILLED BY
C********* POINT. THEN THROW IN A SMOOTHING PASS FOR GOOD
C********* MEASURE.
        CALL UPHIL(GRID,LIMX,LIMY,DD,IER)
        CALL SMOOTH(NUM,GRID,DD,LIMX,LIMY)
        CALL DELETE('VECTOR',IER)
        CALL DELETE('TEXT',IER)
        CALL DELETE('COMPAR',IER)
C******* ONLY WRITE OUT THE GOOD STUFF IN THE MIDDLE OF THE GRID.
        DO 2 I=1,LIMX2
        DO 2 J=1,LIMY2
        DD1(I,J)=GRID(I,J)
 2      CONTINUE
        KEY(1)='NM'
        KEY(2)='CG'
        KEY(3)='RD'
        NBYTES=(LIMX2)*(LIMY2)*4
        CALL GCHN(ICHN,IER)
        CALL CRAND(KEY,IER)
        CALL OPENN(ICHN,KEY,0,IER)
        CALL WRS(ICHN,DD1,NBYTES,IER)
        CALL WRS(ICHN,IDATE(3),2,IER)
        CALL WRS(ICHN,IDATE(2),2,IER)
        CALL WRS(ICHN,IDATE(1),2,IER)
        CALL KLOSE(ICHN,IER)
        STOP
        END
```

```
C
C GRID.FR PREFORMS A NEAREST NEIGHBOR DATA ANALYSIS.  AFTER ALL
C WESTERN REGION HEADQUARTERS SSD
C
C
      SUBROUTINE POINT(RLAT,RLON,IVAL,GP,LIMX,LIMY)
C
      REAL    D(3),GP(LIMX,LIMY)
C
      INTEGER IX(3),JY(3)
C
      REAL LATMIN,LONMAX
      COMMON/TWO/ LATMIN,LONMAX,IDX,IDY
C
C
C
C
C GRID THE DATA WITH A NEAREST NEIGHBOR APPROACH.
      XLAT=LATMIN
      XLON=LONMAX
      DX=IDX/10.
      DY=IDY/10.
      RLAT=((RLAT-(XLAT-DY))/DY)+.5
      RLON=(((XLON+DX)-RLON)/DX)+.5
      J=INT(RLAT)                                  ; GRID POINTS.
      I=INT(RLON)
      IF(I.GT.LIMX)I=LIMX
      IF(I.LT.1)I=1
      IF(J.GT.LIMY)J=LIMY
      IF(J.LT.1)J=1
      IF(GP(I,J).LT.-1000.)GP(I,J)=IVAL
      IF(GP(I,J).GT.-1000.)GP(I,J)=(GP(I,J)+IVAL)/2.
      RETURN
      END
```

```
C
C
C
C
        SUBROUTINE SMOOTH(NUM,GP,DUMDUM,LIMX,LIMY)
C
C
        REAL DUMDUM(LIMX,LIMY)
        REAL GP(LIMX,LIMY)
C
C
C
C SMOOTH THE INTERNAL POINTS ONLY WITH A SIMPLE LAPLACIAN.
        NX=LIMX-1
        NY=LIMY-1
20      IF(NUM.EQ.0) GO TO 100                   ; DONE WHEN N=0
        DO 50 J=2,NX
        DO 50 K=2,NY
        DUMDUM(J,K)=(GP(J-1,K)+GP(J,K-1)+GP(J+1,K)+GP(J,K+1)
     2+4.0*GP(J,K))/8.0
50      CONTINUE
        DO 53 J=2,NY
        DO 53 I=2,NX
        GP(I,J)=DUMDUM(1,J)
 53     CONTINUE
        NUM=NUM-1                                ; DECREMENT THE COUNT.
        GO TO 20
100     CONTINUE
        RETURN
        END
```

NOAA Computer Programs and Problems NWS WR (continued)

36 Soaring Forecast Program. David S. Toronto, July 1982.
37 Program to Work Up Climatic Summary Weather Service Forms (F-6, F-52). Peter G. Mueller, August 1982.
38 The Hovmöller Diagram. Pamela A. Hudadoff, September 1982.
39 850-Millibar Charts Derived from Surface Data. Jeffrey L. Anderson, December 1982.

NOAA SCIENTIFIC AND TECHNICAL PUBLICATIONS

The National Oceanic and Atmospheric Administration was established as part of the Department of Commerce on October 3, 1970. The mission responsibilities of NOAA are to assess the socioeconomic impact of natural and technological changes in the environment and to monitor and predict the state of the solid Earth, the oceans and their living resources, the atmosphere, and the space environment of the Earth.

The major components of NOAA regularly produce various types of scientific and technical information in the following kinds of publications:

PROFESSIONAL PAPERS — Important definitive research results, major techniques, and special investigations.

CONTRACT AND GRANT REPORTS — Reports prepared by contractors or grantees under NOAA sponsorship.

ATLAS — Presentation of analyzed data generally in the form of maps showing distribution of rainfall, chemical and physical conditions of oceans and atmosphere, distribution of fishes and marine mammals, ionospheric conditions, etc.

TECHNICAL SERVICE PUBLICATIONS — Reports containing data, observations, instructions, etc. A partial listing includes data serials; prediction and outlook periodicals; technical manuals, training papers, planning reports, and information serials; and miscellaneous technical publications.

TECHNICAL REPORTS — Journal quality with extensive details, mathematical developments, or data listings.

TECHNICAL MEMORANDUMS — Reports of preliminary, partial, or negative research or technology results, interim instructions, and the like.

Information on availability of NOAA publications can be obtained from:

**ENVIRONMENTAL SCIENCE INFORMATION CENTER (D822)
ENVIRONMENTAL DATA AND INFORMATION SERVICE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
U.S. DEPARTMENT OF COMMERCE**

**6009 Executive Boulevard
Rockville, MD 20852**