Department of the Environment
Ancient Monuments Laboratory Geophysics Section

Geophysical Plotting System for Data General Nova 4 Computer

Part I: Program Notes

Report No. G 18/80

Contents

Introduction

This report describes a set of Fortran programs written to allow numerical processing and graphical display of geophysical data from surveys of archaeological sites. The purpose of the programs is to extract or emphasise significan spatial variations in readings collected on a regular square or rectangular grid.

The programs have been developed on the Data General Nova 4 minicomputer at the Physics Department of the University of Surrey. They run under the RDOS Rev operating system also used with the previous Nova 2 processor. This provides access to 32K 16 bit words of core. The complete system occupies some 75% of a 1.25 M byte disc. A pen plotter, matrix printer and paper tape reader are required.

This set of programs follows two earlier versions of the system, one written for Honeywell (Geisco) time sharing and the other for the Surrey Nova 2 (1). The oscilloscope display facility is not included but is still available in the older programs. The main departure from the earlier programs is that here instead of calling all routines from an interactive main program a separate program must be run for each task. This greatly simplifies problems of program structure and data management (especially when data files exceed the core space), but it means a separate file is created after each operation. The user must name and delete these files. (Some individual core-independent programs were included in the Honeywell package but the main system was core-limited).

The programs in general accept files of width up to 1024 readings. Programs which require access to the data by columns (interpolation, FFT) are also limited to files of 1024 records in length. It would not be possible to handle files of this maximum size given the disc space and processing speed of the present machin but any strip of readings within these outer limits may be processed. There are much smaller size limits for the plotter programs: data may be sectioned if necessary using EXTRACT.

The use and operation of each of the programs is described briefly in turn below. No detailed documentation with lists of variables or flowcharts is attempted. The listings given in Part II are intended to be sufficiently clear to serve that purpose.

A BARTLETT

DOE Ancient Monuments Laboratory
Fortress House
23 Savile Row
London W1

01 734 6010 Ext 531

10 July 1980

(1) The reports on the previous programs are:

(i) Report on Project CSP 4041, Department of Physics, University of Surrey; L S Julien and C Stannard, 28 October 1976.

(ii) Geophysical Plotting System for Mark III Time Sharing. AM Laboratory report 2756, G1/79; A Bartlett, 30 January 1979.

Program Notes

1. Input/Output programs

TAPE 1

- Reads packed binary data from paper tape and writes to file. The values on the tape must represent readings taken at fixed intervals and traverses must be delimited by readings exceeding a stated marker value. Alternate traverses may be inverted (as with logged data replayed through the cassette translator), or they may all read from left to right (as with tapes listed by the program OUTPUT on the Geisco system). TAPE 1 combines the functions of programs ARF2 and ARF3 in the previous Surrey system.

TAPE 2

This accepts readings from paper tape as recorded by the logger when running freely and recording X and Y (position and signal) values alternately. A great deal of redundant information is recorded at the ends of traverses and the program allows this to be edited interactively. A progression of X values is first located by inspection. Successive readings may then be extracted automatically until there is a break in the sequence. It is then possible to step forward locating individual X values until a new sequence is identified. At the end of the traverse X and Y values are listed and the output values interpolated. There are options to split, delete or repeat the traverse, or to write blank values to file for later interpolation from other traverses.

CONVERT

This transfers readings held in a formatted (ASCII character) file to an unformatted working file. The readings may be entered initially from paper tape or by typing directly at the terminal. Lines of readings which are too long for the terminal may be split, and the program will combine a stated number of lines of stated length into each line (record) of the output file. The system will not accept lines longer than 132 characters. Readings must be in integer format.

OUTPUT

- Inverse program to CONVERT. Each record of the working file is written to a stated number of lines in a formatted output file. The file may then be output to tape, edited or printed using operating system commands. (This program is not the same as OUTPUT in the Geisco system which lists data to tape in packed binary format).

2. Data printing and display

LPR

This prints the contents of any file of integer data on the matrix printer. If the width of the block exceeds 132 characters it is split into vertical strips

RPRINT

- Real number version of LPR. Data of real type is used only in the Fourier transform and frequency domain filtering programs.

RECORD

- Program to display a stated record (line) from an integer file on printer or VDU with the readings in numbered positions. There is an option to modify any given value and write the corrected version back to file.

3. File manipulation programs

FCOM

Up to five input files may be combined in a single output file. The co-ordinates of the input files may be specified relative to an arbitrary origin to the bottom left of the data and the program will calculate the output dimensions required to fit the data.

If any file overwrites data from a file which precedes it in the input list the program permits a stated arithmetic or logical combination of the coincident values to be written to the output file.

EXTRACT

- Inverse program to FCOM. Readings from any stated section of a file are copied to the output file.

ROTATE

The output file represents the input file rotated 90° clockwise. (This program calls the same core-independent transposition subroutine, TRANS, as the interpolation programs). To rotate a file 180° use program LR.

LR

This program simply creates an output file representing the input data inverted left to right, top to bottom, or both; ie, rotated 180°.

BSAVE

Readings of a stated value only are copied from the input to the output file. This allows a map of eg blank values to be retained and later substituted back into the data. Blank values should be removed before calculating the Fourie transform, and their positions may be filled by interpolation or by values taken from a fitted line or plane (see LFIT and SFIT).

4. Spatial Filtering

FILTER 1

Readings at stated co-ordinates relative to each data point in turn are averaged and subtracted from or added to the value at that point. The list of co-ordinates representing the filter is read from a previously created file (see FDEF). The data dimensions must not exceed 100 x 100 readings.

FILTER 2

This is identical in effect to FILTER 1 but avoids the core space restriction by calling a function ITEM to return each required reading. The function tests whether the record containing that reading is held in core and if not reads a block of records from the file starting at that position. This device permits a form of virtual addressing to very large data files (used also in SPIKE and CONV2), with only the restriction that the height of the filter x the width of the data must fit the core array (here dimensioned at 8192 readings).

FDEF

A list of co-ordinates representing the positions relative to the central value of points to be included in a spatial filter is written to file. The co-ordinates may either be entered directly (eg to define a filter of arbitrary shape), or they may be defined in terms of circles of stated radius. The number of points in the filter must be noted and is required as a parameter in the filter programs.

Some existing files are:

Filename	Filter radius	No. of points
FILT2	2	12
FILT3	3	16
FILT4	4	24
FILT34	3 & 4	40
FILT45	4 & 5	52
FILT56	5 & 6	60
FILT78	7 & 8	84

CONV1

In convolution filtering the filter is represented by a block of tapered coefficients rather than a list of co-ordinates. The filter array is centred at each data point in turn and the products of the filter coefficients with the corresponding readings are summed. The sum may be substituted for the central value to give the effect of a low-pass filter or subtracted from it to give a high-pass filter. The same high-pass output could be obtained directly from a filter array having a negative central spike value of magnitude equal to the sum of the other terms, but in integer arithmetic this could cause overflow. The coefficients are usually fractional and so the filter values represent the actual coefficients multiplied by a stated power of ten. Data dimensions for CONV1 again must not exceed 100 x 100.

Convolution filtering in principle gives more control of the trade-off between accurate cut-off wavelength and impulse response than simple spatial filtering allows but at the expense of increased execution time.

CONV2

- Core independent version of CONV1. Calls ITEM as with FILTER2.

CDEF

Convolution filter definition program. A series of X and Y co-ordinates are requested and the corresponding coefficient values written to file. (Alternatively a block of coefficients may simply be entered as data and written to a working file using CONVERT. A complete array may be built up from a single quadrant using LR and FCOM). The file co-ordinates, number of non-zero points, and the power of 10 they are multiplied by must be noted.

Convolution filter files include:

CFILT2 Dimensions 3 x 3, no. of non-zero points = 8, radius = 1, coeffs x 10^3.

CFILT6 Dimensions 9 x 9, no. of non-zero points = 81, cut-off wavelength = 6, coeffs x 10^5.

CFILT8 Dimensions 13 x 13, no. of non-zero points = 165, cut-off wavelength = 8, coeffs x 10^5. (Values from E.G. Zurflueh, Geophysics, XXXII, p 1018).

5. Interpolation

INTL

Linearly interpolated output values are calculated by rows and then columns.

Intermediate values are held in a temporary file dimensioned to a power of 2 exceeding the maximum data dimension, and rows and columns are interchanged by subroutine TRANS. They are interchanged back after processing the columns, and the final values are written to the output file. Interpolated values are calculated at all points, but if blanks are required in the output subroutine BLANKS is called to copy appropriately scaled blocks of markers from the input.

INTCS

Rows and columns are processed in turn as for INTL but instead of simple linear interpolation the coefficients of a cubic spline function are found for each successive group of readings. The function is then evaluated at each output location. (The subroutines used (SPLINE and SEVAL) are from G E Forsythe et al, Computer Methods for Mathematical Computations, pp 77-79). The number of points included in the groups for calculating the splines (up to 10) seems to make little apparent difference to the output.

A cubic spline function is the smoothest curve which actually interpolates (fits) all the data points. Its use should avoid some of the tapering which occurs between readings when a large number of intermediate points are found by linear interpolation. Calculated values may however fall outside the range of the initial values and this is liable to make the output appear noisy. When the interpolation is done in both directions the effect is compounded and may cause distortions in the data.

6. Statistics

STATS

A simple program to calculate the range, mean, and r.m.s. variance about the mean of a file of integer readings.

RSTATS

Version of STATS for real data. This is intended for use with the filter or data files used in frequency domain processing, and the values included may either represent the whole of the file or alternate readings corresponding to the real or imaginary terms of complex data.

HG

- Prints a histogram on the matrix printer representing the frequency distribution of readings from a file of integer data. Frequencies may be counted either through the full range of the data or within stated limits.

EDGES

This program returns the mean and range of readings falling within a strip of stated width at each edge of a file. The results allow multiplying factors to be calculated for matching the edges of adjoining files. (Use FMOD for the actual multiplication and FCOM to join the files).

7. Other numerical operations

FMOD

Here any one of several simple numerical operations may be applied to the data. Each reading may be replaced by the result of an arithmetic operation with a constant, or by the value of a given function (with scaling if necessary to stay within the integer range). Readings falling outside a stated range may also be truncated or substituted by a given value.

RANGE

This program generates values representing final display levels, eg for dot-density and character plots. The initial readings within a stated range are replaced in the output by equally spaced values in the range from 0 to the number of levels -1. Readings outside the given range are assigned to 0 or the maximum level.

BFIT

A baseline specified in terms of data frequencies is subtracted from each line of readings. The base level may either be the mode of readings in that line (if there is an equal maximum count for more than one reading the lowest reading is taken), or the first reading at which the frequency count exceeds the previous count by a given increment. (The first step of 4 has been found to define an acceptable base level in magnetic surveys with 30 readings per traverse).

LFIT

A straight line is calculated through each line of data by a least-squares method. Values representing the line may either be subtracted from the readings or substituted for blanks.

SFIT

A plane surface is calculated for the complete file of data. Values may be subtracted or substituted as for LFIT.

SPIKE

- Program to despike data, ie test and substitute for wild values. Readings to be extracted may be specified in terms of a permitted range, or a range defined about the local mean (which is calculated from readings falling on a ring filter array of stated radius). The value substituted may either be stated (eg a blank marker for later interpolation), or may again be a local mean.

The program is effectively an extended version of FILTER2 capable of making two successive applications of the filter routine.

DIFF

An approximate first derivative of the data is calculated by taking the absolute value of differences between neighbouring readings. The pairs of readings subtracted may be horizontal or vertical and of stated separation. This method is described in R C Gonzalez and P Wintz, Digital Image Processing, ch 4 (Addison - Wesley, 1977).

8. Graphics

(i) Pen Plotter Display

CHART

A graph is plotted through each line of readings to build up a chart representing the block of data. The number of readings should be restricted to give reasonable resolution within the 700 x 900 display points available for the plot (excluding margin).

The program will only draw a line between pairs of readings and so a gap of half a reading interval is left at each end of the line. This gap may be reduced by first interpolating to increase the number of points in the line. The vertical scale may be specified in terms of data units per traverse interval. The final display scale is set by the controls on the plotter.

There is an option to suppress hidden lines but the present plotter will not lift the pen with precision. There is a delay routine which can be set to give a short pause before each pen lift and this will reduce some of the mechanical error.

DD1

This dot density program draws dots by rows on a regular square grid within each plotting cell. The size and separation (in display points) of the dots can be set independently of each other, but the effect varies with the scale of the plot and values are best found by experiment. A stated proportion of the dots may be offset randomly to break up the regular boxed effect.

The total number of display points given by the product (dot separation x number of dot positions per reading x number of readings) must fit the display area (700 x 940 points).

DD2

Dots are located at random in a cell of stated size. The reading separation may be controlled independently and so cells can overlap. There is no defined limit to the number of dots per cell. A boxed effect from saturated cells can be avoided by initial linear interpolation to a larger array so that the number of dots is tapered through neighbouring cells, or by setting the cell size to twice the separation so that each cell overlaps 4 others.

CONTOUR

Contours are drawn at given intervals from a stated minimum level to an upper level not exceeding a stated limit. All contours at one level are drawn before moving on to the next. The number of readings must not exceed 100 x 100. The magnitude of any reading must not exceed half the maximum permitted (positive or negative) integer value.

There is an option to identify contours divisible by a stated value by marking ticks at intervals on the downhill side. The separation may be varied so that ticks are more closely spaced for higher contours.

(ii) Matrix Printer Display

PLOT

A specified character is printed for readings at each display level in the data to give a form of density plot. The data file must first have been ranged, interpolated to scale, and a character file created; see below. The plot is printed in strips if the width exceeds the printer. Overprinting is not possible on the machine at Surrey.

SCALE

This program simply calculates the dimensions to which a given block of readings must be interpolated for display at a given scale by program PLOT. The interpolation program must be called separately.

TEXT

- Prints lines of text entered at the terminal on the matrix printer (for use eg, in printing captions below plots).

CHSET

A list of characters representing the display levels of a plot is entered and written to file for use by program PLOT. The number of characters should equal the number of display levels in the data.

(Saved files: CH2, 16 levels; CH3 10 levels).

9. Frequency Domain Processing

FTINPUT

The complex values used in frequency domain processing are represented in these programs by pairs of numbers of real data type. (This allows the same file preparation and display programs to be used for filter and data files). FTINPUT prepares data for the Fourier transform program FFT by transferring reading to alternate (real) locations centred in an oversized file dimensioned to hold 2^N x complex numbers (ie actual width of file 2×2^N items). The file is filled with zeros or a stated background value and the imaginary terms are set to 0.

FFT

The Fourier transform is calculated first by rows and then columns (which are interchanged for this as in the interpolation programs). There are options to shift the low frequency components of the transform to the centre of the output file by multiplying by $(-1)^{X + Y}$ and to normalize the output.

There is no test for blank values in this program and they should first be removed, (by interpolation or by substitution using LFIT or SFIT) or set to 0 (by FMOD). The inverse transform is found by taking the complex conjugate of the input data.

The one-dimensional transform routine called for each line of readings (FFT2) is from S D Stearns; Digital Signal Analysis, p 266 (Hayden, New Jersey, 1975).

FTMAG

- This calculates the absolute magnitudes of the complex values in a file of Fourier transform data. Transforms are conventionally displayed as their magnitudes and the integer output from this program may be used by any of the graphics programs.

FTDEF

A file of filter coefficients of dimensions to match the Fourier transform must be generated before frequency domain filtering. This program creates high or low-pass versions of Butterworth, exponential or simple ideal filters of stated cut-off radius (from formulae given in Gonzalez and Wintz, Digital Image Processing, ch 4). The filter coefficients are real (not complex) numbers.

FTFILT

Frequency domain filter program. The real and imaginary terms of each complex value in a file of transformed data are each multiplied by the corresponding (real) coefficient from the filter file.

FTOUT

Inverse program to FTINPUT. Readings are taken from the centre of the oversized output file after calculating the inverse Fourier transform and written to an integer file compatible with the rest of the system. The output can represent the real or imaginary terms of the complex values or their absolute magnitude. (Normally only the real terms are significant).

FROUT

This converts a file of real numbers to integer values. It could be used, eg, to display a filter array using the graphics programs. All readings are transferred. (ie, They are not taken in pairs and treated as complex numbers as in FTOUT).

REXTR

- Equivalent to program EXTRACT, but operates on real data files.

10. Test Programs

Various test routines used in setting up the programs have been retained as part of the system:

TRTEST

This calls the subroutine TRANS which is used to transpose rows and columns of data for interpolation (a version for complex data, CTRANS, is used by FFT). TRANS is core independent. It interchanges pairs of readings between rows, working on as many rows as will fill the core array at once (method described in J O Eklundh, IEEE Trans Comput C-21, 801-803, 1972). TRTEST fills a file with test values in sequence, calls TRANS, and prints the output. For large arrays only a group of readings at each corner is printed.

TEST

This program allows the various operating subroutines used in the interpolation programs to be called and tested on a single line of data.

HTEST

- Generates a file of readings increasing in value and in frequency at each value by stated increments. This provides controlled data for testing the histogram program, HG.

FNUM

A file of stated dimensions is filled with readings of a single stated value.

FTEST

Program to allow the various options in the Fourier transform program to be tested on a single line of readings.

List of Programs and Subroutines

The names of subroutines to be loaded with each main program are listed below. Programs with a saved loadline may be loaded by typing @ loadline @ (once all the files have been compiled).

Program	Loadline	Subroutines
TAPE1		ARCHRT (assembler routine to read tape)
TAPE2	TAPE2LD	TP1, TP2, TP3, TP4, ARCHRT
CONVERT		
OUTPUT		
LPR		PAR7, FNIJ
RPRINT		PAR14, FNIJ
RECORD		REC1, FNIJ
FCOM		PAR5
EXTRACT		
ROTATE		TRANS
LR		
BSAVE		
FILTER1		FTR1, PAR6
FILTER2		FTR2, PAR6, ITEM
FDEF		RING
CONV1		CV1, PAR4
CONV2		CV2, PAR4, ITEM
CDEF		
INTL	INTLD	LI1, LI2, BLANKS, TRANS, PAR2, XY
INTCS	INTCLD	CSPL, SPLINE, SEVAL, BLANKS, TRANS, PAR2, XY
STATS		PAR9, MEAN, FNIJ
RSTATS		PAR20
HG	HGLD	MEAN, FCOUNT, XAXIS, PSTAT, SIGMA, HGX, HGY, HGL

Program	Loadline	Subroutines
EDGES		EDGE1, FNIJ
FMOD		FNIJ
RANGE		FNIJ
BFIT		FNIJ
LFIT		FNIJ
SFIT		FNIJ
SPIKE	SPKLD	PAR10, RING, SPK, ITEM, FNIJ
DIFF		MINUS, PAR21, FNIJ
CHART	CHARTLD	PAR1, MARGIN, CROSS, GRAPH, FNIJ, DELAY, FTPLOT, BRYAN
DD1	DD1LD	PAR3, SQUARE, RDOT, RND, DOT1,2,3,4, MARGIN, CROSS, FNIJ, DELAY, FTPLOT, BRYAN
DD2	DD2LD	PAR8, RSQ, RND, MARGIN, CROSS, DOT1,2,3,4, FNIJ, DELAY, FTPLOT, BRYAN
CONTOUR	CONTLD	CONT1 - CONT9, PAR11, FR, CT, MARGIN, CROSS, DELAY, FTPLOT, BRYAN
PLOT	PLOTLD	PAR12, BORDER, II, FNIJ, CHPR
SCALE		
TEXT		
CHSET		CHPR
FTINPUT		PAR16
FFT	FFTLD	FFT2, CTRANS, PAR13
FTMAG		PAR15
FTDEF		PAR18
FTFILT		
FTOUT		PAR17
FROUT		PAR19
REXTR		
TRTEST		TRANS

Program	Loadline	Subroutines
TEST		CSPL, SPLINE, SEVAL, XY, LI1, LI2, BLANKS
HTEST		
FNUM		
FTEST		FFT2

Index to Program Units as listed in Part II

The following is a complete list of main programs, subroutines and saved loadlines:

1	TAPE1
2	TAPE2LD
3	TAPE2
4	TP1
5	TP2
6	TP3
7	TP4
8	ARCHRT
9	CONVERT
10	OUTPUT
11	LPR
12	PAR7
13	FNIJ
14	RPRINT
15	PAR14
16	RECORD
17	REC1
18	FCOM
19	PAR5
20	EXTRACT
21	ROTATE
22	TRANS
23	LR
24	BSAVE
25	FILTER1
26	FTR1
27	PAR6
28	FILTER2
29	FTR2
30	ITEM
31	FDEF
32	RING
33	CONV1
34	CV1
35	PAR4
36	CONV2
37	CV2
38	CDEF
39	INTLD
40	INTL
41	LI1
42	LI2
43	BLANKS
44	PAR2
45	XY
46	INTCLD
47	INTCS
48	CSPL
49	SPLINE
50	SEVAL
51	STATS
52	PAR9
53	MEAN
54	RSTATS
55	PAR20
56	HGLD
57	HG
58	FCOUNT
59	XAXIS
60	PSTAT
61	SIGMA
62	HGX
63	HGY
64	HGL
65	EDGES
66	EDGE1
67	FMOD
68	RANGE
69	BFIT
70	LFIT
71	SFIT
72	SPKLD
73	SPIKE
74	PAR10
75	SPK
76	DIFF
77	MINUS
78	PAR21
79	CHARTLD
80	CHART
81	PAR1
82	GRAPH
83	MARGIN
84	CROSS
85	DELAY
86	DD1LD
87	DD1
88	PAR3
89	SQUARE
90	RDOT
91	RND
92	DOT1
93	DOT2
94	DOT3
95	DOT4
96	DD2LD
97	DD2
98	PAR8
99	RSQ
100	CONTLD

101	CONTOUR	
102	CONT1	
103	CONT2	
104	CONT3	
105	CONT4	
106	CONT5	
107	CONT6	
108	CONT7	
109	CONT8	
110	CONT9	
111	CT	
112	FR	
113	PAR11	
114	PLOTLD	
115	PLOT	
116	BORDER	
117	II	
118	PAR12	
119	CHPR	
120	SCALE	
121	TEXT	
122	CHSET	
123	FTINPUT	
124	PAR16	
125	FFTLD	
126	FFT	
127	FFT2	
128	CTRANS	
129	PAR13	
130	FTMAG	
131	PAR15	
132	FTDEF	
133	PAR18	
134	FTFILT	
135	FTOUT	
136	PAR17	
137	FROUT	
138	PAR19	
139	REXTR	
140	TRTEST	
141	TEST	
142	HTEST	
143	FNUM	
144	FTEST	
145	FTPLOT	plotter utilities
146	BRYAN	

Note on Fortran standard

The Data General version of Fortran IV used in these programs has few departures from the 1966 standard except in input and output statements.

Non-standard features which have been used include:

TYPE	-	to display text on VDU
ACCEPT	-	to read values to single variables
READR	-	read stated file records
WRITR	-	write records
S	-	string format descriptor

DATA statements are only allowed for variables in labelled common.

The first character of a format string literal is suppressed after a line feed and so an extra space must be included.

The READR and WRITR commands have been used throughout in place of the standard READ and WRITE (even where this is very clumsy as in TRANS) because READ and WRITE rarely seem to operate on this machine and the FSEEK record seeking routine does not work.

Much use has been made of ACCEPT, but many of the input requests are collected in separate parameter subroutines. This should reduce the number of changes to be made to the main programs in moving to another machine.

Acknowledgements

I must thank C Stannard and D Munro of the computing staff at the University of Surrey both for their practical help and for making a great deal of computer time available to me. D Haddon-Reece of the Ancient Monuments Laboratory has also offered much helpful advice.