United States Department Of The Interior

Geological Survey

FFTFIL: A filtering program based on two-dimensional Fourier analysis of geophysical data

by

Thomas G. Hildenbrand

Open-File Report 83-237

Preface

All routines (Appendix B) are written in Multics Fortran (card image) to be run on the Honeywell Multics 68/80, and can be converted to other computers with a moderate amount of revision by an experienced programmer. The Honeywell Multics 68/80 is a 36 bit word machine and has a maximum floating point number of octal 376777777777. This number is used in the program to flag regions of no data and may need to be changed in conversion to another computer. The program contains a few "calls" to Multics subroutines (primarily manipulating data input and output) that will also require replacement in converting to another computer. The program requires about 32 K words of main memory, independent of variable storage, and has a maximum array size of 32,768 bytes.

Table of Contents

Abstract

The filtering program "fftfil" performs a variety of operations commonly required in geophysical studies of gravity, magnetic, and terrain data. Filtering operations are carried out in the wave number domain where the Fourier coefficients of the input data are multiplied by the response of the selected filter. Input grids can be large ($2 \leq$ number of rows or columns ≤ 1024) and are not required to have numbers of rows and columns equal to powers of two.

Introduction

The program "fftfil" provides a means of using two-dimensional Fourier analysis to perform the following operations on an input grid:

(1) reduction of the total magnetic field intensity to the North Pole,

(2) pseudo-gravity transformation,

(3) first vertical derivative analysis,

(4) second vertical derivative analysis,

(5) upward continuation,

(6) downward continuation,

(7) trend analysis,

(8) pseudo-magnetic transformation,

(9) band pass analysis,

and (10) creation of Fourier coefficients.

The direct discrete Fourier transform method is employed in these operations. In this method (Dean, 1958; Embree and others, 1963; Byerly, 1965; Fuller, 1967; Naidu, 1969), the data are transformed first into the wave number domain by means of the Fast Fourier Transform (FFT). The Fourier coefficients are then multiplied by the wave number response of the appropriate digital filter. Finally, the resulting Fourier coefficients are inversely transformed back into the space domain yielding the desired filtered data.

The filtering operations can be performed on large data sets commonly encountered in geophysical processing of magnetic, gravity, and terrain data. These data must be represented by discrete values on a two-dimensional rectangular grid in standard USGS form (see Appendix A). The results of the filtering operations are output in the standard grid format.

Incomplete Data

The program will accept incomplete data (i.e., data which do not fill out a rectangular grid) provided the grid points without data are flagged by the maximum floating point number (octal 37677777777). However, in order to perform the fast Fourier transform, flagged data in the grid are assigned temporary values by means of an averaging technique. In assigning a value to a flagged grid point, the routine averages the four neighboring grid values (a neighboring flagged point is ignored). For each row the routine searches for the first nonflagged value and then, if necessary, assigns values of the previous flagged grid points by working backwards (right to left) to the beginning of the row. Other flagged grid points within a row or at the end of a row are filled in by working left to right with the averaging technique. In assigning values in this way, there is at least one neighboring grid point that does not contain a flagged value. It should be noted that the program does not allow entire flagged values within the first (bottom) row.

After the desired filtering operation has been completed on the filled-in grid, flagged values are reinserted at those grid points assigned temporary values. The user is cautioned against using input grids containing appreciably large regions of missing data. Such grids may introduce unwanted characteristics in the Fourier domain resulting in erroneous output filtered grids.

Theory

The Fourier integral transform G(u,v) of a function g(x,y) is defined as

$$G(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x,y)e^{-2\pi i(ux + vy)}dxdy \qquad (1)$$

where u and v are unit frequencies associated with the x (north) and y (east) coordinate axes, respectively. In practice, the potential field function g(x,y) is known only at discrete points (grid points) and its transform is obtained by using the discrete Fourier transform:

$$G(k,l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m,n) e^{-2\pi i \left(\frac{km}{M} + \frac{ln}{N}\right)} \qquad (2)$$

where m and n = integers representing grid point locations in the space domain,

k and l = integers representing grid point locations in the frequency domain,

and M and N = number of row and columns, respectively.

The use of the discrete Fourier transform can lead to unwanted characteristics in the frequency, especially at the higher frequencies. These characteristics are discussed by Cordell and Grauch (1982) and in more detail later.

Grid or two-dimensional filtering may be described by the convolution formula:

$$f(x,y) = h(x,y) * g(x,y) \qquad (3)$$

which is shorthand notation for the integral

$$f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x,y)\, g(x-u, y-v)\, du\, dv,$$

where g(x,y) is the input data,

f(x,y) is the filtered output,

and h(x,y) is the filtering function.

Denoting the Fourier transform of a function by the corresponding capital letter and taking the Fourier transform of (3), we have using the convolution theorem (Bracewell, 1965, p. 108):

$$F(u,v) = H(u,v)\ G(u,v) \tag{4}$$

where u and v are unit wave numbers associated with the x and y coordinate axes, respectively. It should be noted that convolution in the space domain is equivalent to multiplication in the frequency domain.

In performing two-dimensional frequency analysis, the input data are transformed to the frequency domain utilizing equation (2). The resulting Fourier coefficients G(k,1) are multiplied by the wave number response of the filter H(k,1) for computing the Fourier coefficients F(u,v) of the filtered data in equation (4). The output filtered grid is then determined by the inverse Fourier transform:

$$f(m,n) = \frac{1}{M}\ \frac{1}{N} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} F(k,l) e^{2\pi i\left(\frac{km}{M} + \frac{ln}{N}\right)} .$$

The derivation of the frequency responses of the filters used in the program can be found in the literature and, therefore, will not be shown here. These responses or operators are as follows (assuming distance units of kilometers and a coordinate system with x,y, and z pointing east, north, and down, respectively):

filter	response [H(u,v)]	reference
upward (z<0) or downward (z>0) continuation	$e^{2\pi z(u^2 + v^2)^{1/2}}$ where z = continuation distance	(Fuller, 1967)
1st vertical derivative	$2\pi(u^2 + v^2)^{1/2}$	- - - -

2nd vertical derivative	$4\pi^2(u^2 + v^2)$	(Fuller, 1967)
pseudo-gravity transformation (units: mgal)	$\frac{\omega G\rho / 2\pi I}{(iul' + ivm' + wn')(iul + ivm + wn)}$	(Lourenco, 1973)

where l', m', n' are direction cosines of earth's ambient field relative to x, y, and z, respectively,
l, m, n, are direction cosines of magnetization vector
$w = (u^2 + v^2)^{1/2}$
G = gravitational constant (dyne-cm^2/g^2)
I = intensity of magnetization (gammas)
ρ = density (g/cm^3)

pseudo-magnetic transformation (units: gammas)	(above equation inverted)	- - - -
reduction to pole	$\frac{(u^2 + v^2)}{(iul' + ivm' + wn')(iul + ivm + wn)}$	(Lourenco, 1973)
directional filtering (passing trends between Θ_1 and Θ_2; $\Theta_1<\Theta_2$)	1 for $180-\Theta_2 \leq \tan^{-1}(v/u) \leq 180-\Theta_1$ 0 otherwise	(Embree, 1963)
band-pass filtering	1 for $\sqrt{u^2 + v^2}$ within specified range of wavelengths 0 otherwise	- - - -

Boundary Effects

Filtering finite-length data using the Fourier transform can produce distorted anomalies near the grid boundaries. These anomalies result from the abrupt terminations of the data. For example, filters that enhance long wavelengths (upward continuation, low-pass, and pseudo-gravity transformations) have the tendency to close anomaly patterns and produce oval-shaped anomalies near the edges of the data. Discontinuities at the grid boundaries also give rise to fictitious short-wavelength anomalies that are enhanced by short-wavelength amplifying operators such as differentiation, downward continuation, and pseudo-magnetic transformation.

Both types of edge effects can be appreciably reduced by adding rows and columns of assigned grid values to the input data. Edge effects are still present but are primarily within the frame of assigned border values which are removed after the filtering process. The routine will automatically perform the task of adding border values when the user specifies the number of rows and columns to be added to each side of the input grid. Assigned values within this border region are simply outward extensions of the first and last row and column. Normally, 5 to 10 rows and columns on each side helps in reducing distortion at the map boundaries. However, in applying the long-wavelength amplifying operators, it may be necessary to increase the grid size by more than 10 rows and 10 columns on each side. For the purpose of lessening distortion at the boundaries, the user is advised to utilize the option to increase the grid size.

The most practical way of removing edge effects is to enlarge the input grid with real data to a size sufficient for eliminating distortion within the area of interest. The added real data can be removed after the filtering process (external to "fftfil"). This technique is highly recommended if long-wavelength amplifying operators are applied to the data.

Practical Considerations

In addition to edge effects, the users must consider many other factors before applying filter operators. Some of these factors are given below.

Short-wavelength amplifying operators

The use of the discrete Fourier transform (eq. 2) as an approximation of the Fourier integral transform (eq. 1) gives rise to fictitious short-wavelength tails in the Fourier domain (Cordell and Grauch, 1982). The tails

are a result of sampling data only a finite number of times within a limited range. Because these tails pose a problem in the application of short-wavelength amplifying operators, practical considerations must be exercised in using filters such as differentiation and downward continuation. A map generated with these filters will generally exhibit many high-amplitude, single-grid-point anomalies (i.e. short-wavelength distortion). Detrending the input data by removal of a planar surface will lessen this distortion. It is sometimes more useful to simply apply a low-pass filter in conjunction with short-wavelength amplifying operators. The low-pass filter (removing wavelengths less than 2 to 4 grid spacings) will eliminate many of the distorted anomalies and produce a more desirable filtered-anomaly map. Cordell and Grauch (1982) discuss alternative ways to lessen the problem.

Wavelength-filtering

Low-pass and high-pass filters require separate discussions because of their widespread use in generating residual-regional maps. The validity of the wavelength filtering process in calculating regional-residual fields is dependent on the assumption that the cut-off wavelength of the filter and maximum depth of source are related. One generally uses the wavelength filtering method to obtain a separation of long wavelength anomalies (regional), that are typically associated with deep-crustal or subcrustal features, from short wavelength anomalies (residual) that are associated with shallow features. The user must be aware that the separation is not complete and that in particular, some long wavelength anomalies can be caused by broad shallow features. The short wavelengths on the residual maps bring out and emphasize small features, but in some cases the anomaly amplitudes may be distorted by removal of the long wavelengths. The reader is referred to

Simpson and others (1982) for discussions on the causes of these distortions.

In applying the low-pass or high-pass filter ("banpas" operator of "fftfil"), one should use a modified rectangular window, so that the gain drops from unity to zero along a ramp centered at the desired cut-off wavelength. This will reduce the effect of Gibbs phenomena and produce a more desirable filtered-anomaly map. Selection of a ramp's width is highly empirical. I have found that in removing (or passing) wavelengths greater than 250 km, a ramp located between 200 km and 300 km is suitable.

Trend analysis

Trend analysis involves passing anomalies striking between two specified directions and rejecting all other anomaly trends. The operator "strike" is used for this purpose. Because anomalies become elongated in the selected filtering direction, caution must be exercised in interpreting trend maps. Only features indicated on the unfiltered map can be interpreted on the trend maps. Although this restriction is stringent, trend maps are often useful in enhancing features that are real but not easily identifiable on the unfiltered anomaly map.

Execution

When the program is run, an asterisk is typed on the terminal. The user responds with the name of the command file containing filtering instructions. After the current command file is executed and the data has been filtered, the routine asks if an additional filter is to be applied to the input data and requests the needed information to perform the new filter operation. The user can apply as many filters as desired to the original data set. It should be noted that the filter operations are carried out only on

the input data set and not on any of the previously calculated, filtered data sets. When filter operations have been completed on a particular data set, the routine will request another command file. This sequence can be continued indefinitely. To exit from the program, the user can enter "ex" or a line feed after the asterisk is typed.

There are several program error messages that can occur during execution. These messages are preceded by a "#". When an error occurs, all files are closed and the program will generally request another command file.

Command File

The command file is an ASCII file containing the required filtering information. It is comprised of four fixed format lines followed by a "&parms" namelist section. The four fixed format lines may be blank which results in the routine requesting the user to respond with the required information. This allows the user to execute a command file several times but perform different filtering operations on various input grids. Examples of command files and their execution are given on page 21.

Line 1:

Characters 1 through 6 contain the coded filter operation to be performed. The possible operations are:

Coded Operation	Explanation
psdgrv	Pseudo-gravity transformation.
psdmag	Pseudo-magnetic transformation.

(Note: the above operations require grid units in kilometers)

redpol	Reduction of total magnetic field intensity to the pole.
upcont	Upward continuation.

dncont	Downward continuation.
1stver	1st-vertical derivative.
2ndver	2nd-vertical derivative.
strike	Direction filtering.
banpas	Band pass filtering.
nofilt	Create Fourier coefficients and output to file "fftfil.coef".

If the field is blank, the routine will request the coded operation. The user may also respond with "help" which results in the routine printing the possible coded operations and then requesting again the desired operation. If an incorrect operation is entered, the routine will print the 10 possible operations and then request one of them.

Line 2:

Characters 1 through 50 contain the file name of the standard two-dimensional grid file (see Appendix A) to be filtered. If the field is blank, the routine will request the file name.

Note that if the namelist parameter "icoef" (described below) is equal to -1, the Fourier coefficients of the data set, determined from a previous run, are used as input. In responding to the query of the input file name, the user must, however, enter the file name of the standard USGS grid from which the input Fourier coefficients were calculated. The routine reads only the header record of the standard grid to obtain the number of rows, columns, etc.

Line 3:

Characters 1 through 50 contain the output file name. The output filtered grid is written in standard form. If the field is blank, the routine will request the file name.

Line 4:

Title line (characters 1 through 56) to be written in header record of output filtered grid. If the field is blank, the program will request a title.

Line 5: Namelist Section

Characters 1 through 6 must contain: &parms. Characters from 8 on contain namelist items as well as continuation lines. Section must be terminated by an &. All the namelist items listed below have been given default conditions.

Because of the large number of input parameters used by "fftfil", the user should consult the following table for the exact parameters needed to perform the desired filtering operation. The term "general" in this section refers to parameters that can be selected for any filtering operation.

Coded Operation	Required Parameters
general	ioptl, iopt2, nadd, idval, icoeff, ddx, ddy, xo, yo
psdgrv	den, bmag, bdec, binc, dec, xinc
psdmag	den, bmag, bdec, binc, dec, xinc
redpol	bdec, binc, dec, xinc
upcont	z (negative)
dncont	z (positive)
lstver	(none)
2ndver	(none)
strike	istr, thetl, thet2
banpass	wl, w2, w3, w4
nofilt	(none)

Parameters

General

ioptl =	0 (default)	No printed output.
	-1	Output filtered grid printed (ASCII) in file "fftfil.out."
	6	Output filtered grid printed (ASCII) on terminal.
iopt2 =	0	Substract mean of boundary values from input grid.
	1	Subtract mean from grid and save resulting grid: file name - "fftfil.grd".
	-1 (default)	No subtraction of mean.
nadd =		Number of rows and columns added to each side of input grid to reduce the anomaly distortion and effects of Gibbs phenomena. (default condition, nadd = 0).
idval =	0 (default)	No flagged data in input grid.
	1	Flagged data in input grid (note, the first row must not consist entirely of flagged values).
	-1	Flagged values have been removed from data set and their locations are in segment "flag.loc". [Note: User must specify idval = -1 if the Fourier coefficients of the data (determined from a previous run) are to be used as input and if the grid

contains missing data. When the Fourier coefficients were computed, a file named "flag.loc" was created in the user's disk area. Specifying idval = -1 results in the routine reading "flag.loc" for the purpose of restoring the output filtered grid to the shape of the input grid.]

icoef = 1 — Save Fourier coefficients of input segment in file "fftfil.coef" for later use as input to program [note: if data contains flagged values, a segment called "flag.loc" containing their locations is also created in user's disk area].

0 (default) — Fourier coefficients not saved.

-1 — Fourier coefficients in segment "fftfil.coef" are used as input. Note that if the flagged values are present in the data, segment "flag.loc" is required and "idval" must be equal to -1 so that the filtered grid can be restored to the shape of the original input grid.

[The following 4 namelist items can be used to change the grid spacings or origin in the header record of the output filtered grid. Default conditions, ddx = ddy = xo = yo = 0.0.]

ddx and ddy = grid spacing along the x-axis (east) and y-axis (north), respectively,

xo = position of first (bottom) data row.

yo = position of first (left most) data column.

Pseudo-gravity, pseudo-magnetic,

and reduction to pole

[For these operations the mean of the output filtered data will be zero because the Fourier coefficient at zero frequency (u=v=0) is set to zero.]

den = Density contrast (gm/cm^3).
(default conditions, den = 1.)

bmag = Magnetization contrast (gammas).
[Note: for X (emu) = susceptibility and F (gammas)=Earth's strength, bmag = XF]
(default condition, bmag = 1.)

bdec & binc = Respectively, declination and inclination (degrees) of the magnetization vector associated with the magnetic source.
(default conditions; bdec = 0.0, binc = 90.)

dec & xinc = Respectively, declination and inclination (degrees) of the earth's ambient magnetic field.
(default conditions; dec = 0.0, xinc = 90.)

Continuation

z = Continuation distance used in upward (z<0) or downward (z>0) continuation of data. The value entered must be in same units as used to specify dx and dy in grid header (see Appendix A).

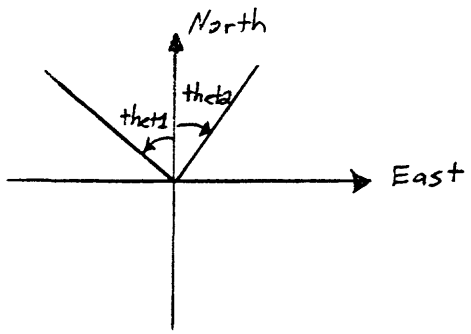
Trend analysis

[It should be noted that the mean of the output filtered data will equal the mean of the input data. This is because the Fourier coefficient at zero frequency (u=v=0) is unchanged during the filtering process.]

istr = -1 Reject trends striking between "thet1" and "thet2".

1 (default) Pass only trends striking between "thet1" and "thet2".

thet1 & thet2 = Angles from Geographic North that form a pie-slice filter. A clockwise direction from Geographic North designates positive angles.



(If northeast trends are to be enhanced, one could use thet1=0, thet2=90 and istr=1. If east-west trends are to be enhanced, one could use thet1=-45, thet2=45, istr=-1.)

The following conditions must be met:

$-90 \leq thet1 \leq 90.$

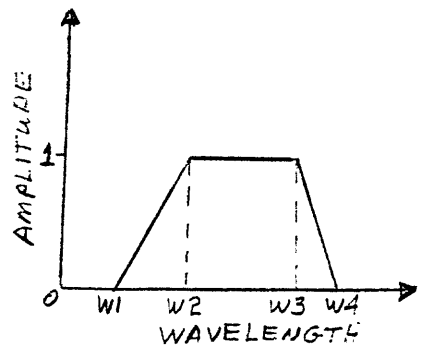
$-90 \leq thet2 \leq 90.$

$thet2 \geq thet1$

Band pass

w1,w2,w3,w4 = cutoff wavelengths used in band pass

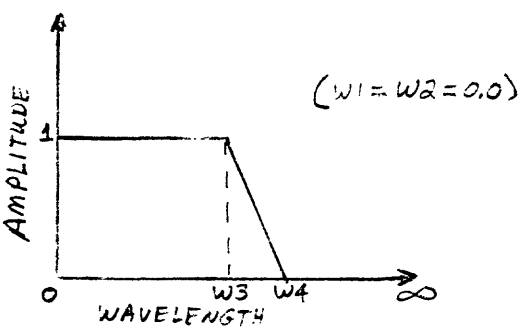
filter and described in the following diagram:



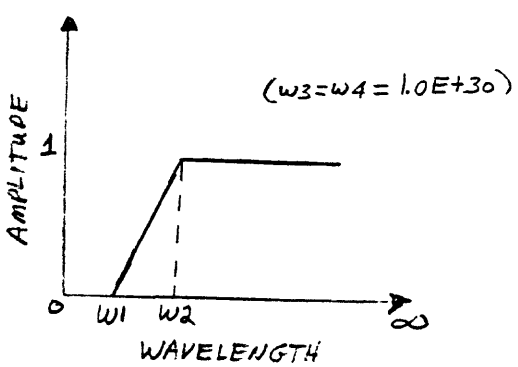
Data within tetrahedron is passed.

These parameters can also be used to design a low-pass or high-pass filter:

High pass--(passing short wavelengths)



Low pass--(passing long wavelengths)



[Note: $w1 \leq w2 \leq w3 \leq w4$

For example, if wavelengths greater than 10 km are to be removed from the data, one may design a filter with the

following cutoff wavelengths: (ramp is to provide a more continuous cutoff)

w1 = 0.0

w2 = 0.0

w3 = 8.0

w4 = 12.0

Default conditions: w1 = w2 = 0.0,

w3 = w4 = 1.0 E+30 which is an all-pass filter]

Queries

The following queries are asked after filtering operations have been completed. The purpose of these queries is to apply additional filters to the input data.

Query 1: Additional filtering to original grid? Enter "y" if an additional filter is to be applied to input data. (Note that the additional filter is applied to the input data and not to the previously determined filtered data.) If "y" is entered, queries 2 - 6 are asked. If "n" is entered, the program starts over by asking for a new command segment.

Query 2: New filter operator? (Format the same as that given for command "line 1" in command segment.)

Query 3: New output segment name? (Format the same as that given for command "line 3" in command segment.)

Query 4: New title? (Format the same as that given for command "line 4" in command segment.)

Query 5: Parm change? Enter "y" if some of the namelist parameters are to be changed for the new filter operation. The parameters to be changed must be given on the following line in a namelist beginning with "&parms" and ending with "&". [Example: &parms thetl = -90., thet2 = 0., &] Enter "n" if no parameter change is needed. In this case, the filtering operations are started.

[Note: the following parameters cannot be changed--nadd, iopt2, idval, ddx, ddy, xo, and yo.]

Query 1 is then asked again. In other words, queries 1 - 5 are repeated until a user's response of "n" is encountered after a filtering operation has been completed.

Examples of Command Files and Program Execution

The following examples illustrate the many options in the filtering program. The amount of processing time in examples 1 and 3 suggests that about 4 millisecond per grid point is required for grids with 50,000 points or less. This time will, however, appreciably increase for larger grids. On the other hand, if the Fourier coefficients are used as input as in example 2, the processing time is significantly reduced.

During execution the program lists all the selected parameters such as "nadd", the number of columns and rows added to each side of the grid. Listed beside the value of "nadd" is " extended columns and rows = ", after which the total columns and rows used during execution are given. The total number of columns will equal twice "nadd" plus the number of columns in the input

grid. However, the total number of rows will generally be larger if computed in the same manner. This is because the algorithm transforms the data in blocks and may require additional rows to complete a block. Block sizes range from 9 to 16 rows. If the total number of rows for a particular block size is slightly smaller than a power of 2, the program will automatically select a block size of 16 and extend the grid to that power of 2. To reduce processing time one may add columns so that the total number of columns equals a prime number or a number that is a power of 2.

Indicated underline phrases in the examples below are user input:

. This example illustrates upward continuation of a magnetic data set.

A file containing the Fourier coefficients of these data is also

written and is used below in example 2.

Command file: fftfil.cmd1

```
upcont
cenmag.grd
cenmag.upcont.grd
central us magnetics: upward continued 1km
 &parms
  iopt1=0,icpt2=0,nadd=5,z=-1,idval=1,icoef=1,&
```

Execution:

```
fftfil
 *fftfil.cmd1

mean value subtracted =  -0.32698855E+03        424 values used

upward continuation of field values,
by means of the fast fourier transform.

title: central us magnetics: upward continued 1km
parameters: density=  1.000     datum level z= -1.000
            intensity of magnetization=    1.000
            inclination & declination of geomagnetic field=   90.000     0.00
            inclination & declination of magnetization vector=   90.000     0
            thet1=    0.000     thet2=   90.000     istr= 1
            w1,w2,w3&w4= 0.0000E+00 0.0000E+00 0.1000E+31 0.1000E+31
            dx & dy=  0.400000E+01  0.400000E+01
            origin of grid(xo & yo)=  0.358400E+04 -0.108400E+04
            no. of columns & rows=  100  100
            nadd=    5  extended columns & rows=  110  112
            input file name: cenmag.grd
            output file name: cenmag.upcont.grd

 additional filters to be applied? (y or n)n
 *ex

STOP
R 07/04/82  1020.9 mdt Sun CPU: 54.254 COST: $1.36
```

. The following example illustrates the option of applying many

filters. The input data are Fourier coefficients of the grid of

magnetic data in example 1.

Command file: fftfil.cmd2

```
redpol
cenmag.grd
cenmag.redpol.grd
central us magnetics: reduced to pole
 &parms
  iopt1=0,iopt2=-1,nadd=5,z=-1,dec=30,xinc=50
  bdec=18,binc=65,idval=-1,icoef=-1&
```

```
Execution:

fftfil
 *fftfil.cmd2
transformation of the total magnetic field intensity to the pole,
by means of the fast fourier transform and the poisson equation.

title: central us magnetics: reduced to pole
parameters: density=  1.000     datum level z= -1.000
            intensity of magnetization=    1.000
            inclination & declination of geomagnetic field=   50.000   30.000
            inclination & declination of magnetization vector=   65.000   18.00
            thet1=    0.000     thet2=   90.000     istr= 1
            w1,w2,w3&w4= 0.0000E+00 0.0000E+00 0.1000E+31 0.1000E+31
            dx & dy=  0.400000E+01  0.400000E+01
            origin of grid(xo & yo)=  0.358400E+04 -0.108400E+04
            no. of columns & rows=  100  100
            nadd=    5  extended columns & rows=  110  112
            input file name: cenmag.grd
            output file name: cenmag.redpol.grd

 additional filters to be applied? (y or n)y
 enter new operator:2ndver
 enter new output file name:cenmag.2ndver.grd
 enter new title
central us magnetics: 2nd vertical derivative
 parm change? (y or n, if y enter parameters via &parms):n
second vertical derivative operation,
by means of the fast fourier transform.

title: central us magnetics: 2nd vertical derivative
parameters: density=  1.000     datum level z= -1.000
            intensity of magnetization=    1.000
            inclination & declination of geomagnetic field=   50.000   30.000
            inclination & declination of magnetization vector=   65.000   18.00
            thet1=    0.000     thet2=   90.000     istr= 1
            w1,w2,w3&w4= 0.0000E+00 0.0000E+00 0.1000E+31 0.1000E+31
            dx & dy=  0.400000E+01  0.400000E+01
            origin of grid(xo & yo)=  0.358400E+04 -0.108400E+04
            no. of columns & rows=  100  100
            nadd=    5  extended columns & rows=  110  112
            input file name: cenmag.grd
            output file name: cenmag.2ndver.grd

 additional filters to be applied? (y or n)y
 enter new operator:dncont
 enter new output file name:cenmag.dncont.grd
 enter new title
central us magnetics: downward continued 1km
 parm change? (y or n, if y enter parameters via &parms):y
&parms z=1,&
downward continuation of field values,
by means of the fast fourier transform.

title: central us magnetics: downward continued 1km
parameters: density=  1.000     datum level z=  1.000
            intensity of magnetization=    1.000
            inclination & declination of geomagnetic field=   50.000   30.000
            inclination & declination of magnetization vector=   65.000   18.0
```

```
             thet1=    0.000     thet2=   90.000     istr= 1
             w1,w2,w3&w4= 0.0000E+00 0.0000E+00 0.1000E+31 0.1000E+31
             dx & dy=  0.400000E+01  0.400000E+01
             origin of grid(xo & yo)=  0.358400E+04 -0.108400E+04
             no. of columns & rows=  100  100
             nadd=    5  extended columns & rows=  110  112
             input file name: cenmag.grd
             output file name: cenmag.dncont.grd

 additional filters to be applied? (y or n)n
 *

STOP
R 07/04/82  1025.2 mdt Sun CPU: 77.854 COST: $1.89
```

. This example uses a command file with no information given on the 4 fixed format lines. The program requests this information during execution (note that the filter operator "banpas" was incorrectly entered and that the program listed the 10 possible filters and requested one of them).

Command file: fftfil.cmd3

```
 &parms
  iopt1=0,iopt2=-1,nadd=10,den=.2,bmag=100.,dec=30,xinc=50
 istr=-1,bdec=18,binc=65,idval=1,w1=4,w2=5,thet1=-90,thet2=0,icoef=0,&
```

Execution:

```
fftfil
 *fftfil.cmd3
 enter operator:babpas
#invalid operator
quick list of options
 psdgrv
 redpol
 upcont
 dncont
 2ndver
 strike
 banpas
 psdmag
 1stver
 nofilt
 enter operator:banpas
 enter input file name:basran.grv
 enter output file name:basran.lowpass.grv
 enter title:basin & range gravity: low-pass 4km & 5km
band-pass filter applied to data

title: basin & range gravity: low-pass 4km & 5km
parameters: density=  0.200     datum level z=  0.000
```

```
          intensity of magnetization=  100.000
          inclination & declination of geomagnetic field=    50.000    30.000
          inclination & declination of magnetization vector=    65.000    18.00
          thet1=  -90.000     thet2=    0.000     istr=-1
          w1,w2,w3&w4= 0.4000E+01 0.5000E+01 0.1000E+31 0.1000E+31
          dx & dy=  0.400000E+01  0.400000E+01
          origin of grid(xo & yo)=  0.358400E+04 -0.108400E+04
          no. of columns & rows=  200  200
          nadd=   10  extended columns & rows=  220  224
          input file name: basran.grv
          output file name: basran.lowpass.grv

 additional filters to be applied? (y or n)n
 *

STOP
R 07/04/82  1030.4 mdt Sun CPU: 185.918 COST: $4.84
```

References

Bracewell, Ron, 1965, The Fourier transform and its application: McGraw-Hill, New York, 381 p.

Byerly, P. E., 1965, Convolution filtering of gravity and magnetic maps: Geophysics, v. 30, p. 281-283.

Cordell, Lindrith, and Grauch, V. J. S., 1982, Reconcilliation of the discrete and integral Fourier transforms: Geophysics, v. 47, no. 2, p. 237-243.

Dean, W. C., 1958, Frequency analysis for gravity and magnetic interpretation: Geophysics, v. 23, p. 97-127.

Embree, P., Burg, J. P., and Backus, M. M., 1963, Wide-band velocity filtering--the pie-slice process: Geophysics, v. 28, no. 6, p. 948-976.

Fuller, B. D., 1967, Two-dimensional frequency analysis and design of grid operators: Mining Geophysics, v. 2, p. 658-708.

Lourenco, J. S., 1973, Analysis of three-component magnetic data (Ph. D. dissert.): Berkeley, University of California, 153 p.

Naidu, P. S., 1969, Estimation of spectrum and cross-spectrum of aeromagnetic field using fast Fourier transform techniques: Geophysical Prospecting, v. 17, p. 344-361.

Simpson, R. W., Hildenbrand, T. G., Godson, R. H., and Kane, M. F., 1982, A description of colored gravity and terrain maps for the United States and adjacent Canada east of 104°: U.S. Geological Survey Open-File Report 82-877, 18. p.

Appendix A

Standard Grid File

The USGS standard grid file consists of two basic parts: 1) a header record providing control parameters and, optionally, a following record of column coordinates, and 2) the profile or grid data. The general description of the file is given below with added notes of requirements for the program "fftfil". Note that the direction of columns and rows are, respectively, north-south and east-west. Y-axis points north and the x-axis points east. In addition, "fftfil" assumes that the first value stored in the data array is located at the lower left-hand corner of the map. Recording mode is unformatted.

A. Header record (23 words)

id:	56 ASCII characters of identification (14 words).
pgm:	8 ASCII characters of creation program identification (two words).
ncol:	number of columns of data (integer, one word).
nrow:	number of rows of data (integer, one word).
nz:	number of words per data element (integer, one word). For single precision, single value data use 1, double precision or complex use 2, ... etc. In the execution of "fftfil", nz must equal 1.
xo:	position of first (leftmost) column of data (real, one word).

dx: equal spacing interval of columns (real, one word). If equal to zero, then coordinate for each column is in the data record, otherwise the following record consists of data. For "fftfil", dx cannot equal zero.

yo: position of first (bottom) row (real, one word).

dy: equal spacing interval of rows (real, one word). If equal to zero, the coordinate for each row is the first word of each data record row. For "fftfil", dy cannot equal zero.

B. Column coordinate record (not used in "fftfil") present if dx of header record equals zero. Record consists of "ncol" real words specifying the coordinates of each data column. Most programs require the coordinates to be in monotonic order.

C. Data record. Each data record contains one row of real data items. The total record length is "ncol" times "nz" plus one word. The first word contains the row coordinate if "dy" is zero. When "dy" is not zero, the first word is ignored (which is the case in "fftfil"). When specified, the row coordinates should be in monotonic sequence.

In general, a standard file can be created in FORTRAN as:

```
dimension f(ncol,nrow)
character id*56, pgm*8
dum = 0.0
. .
. .
write (10)id,pgm,ncol,nrow,nz,xo,dx,yo,dy
```

```
      do 20 i = 1, nrow
      write (10) dum, (f(j,i), j = 1, ncol)
   20 continue
```

Appendix B

```
c*************************************************************************fft00010
c       program fftfil                                                    fft00020
c   filter operations employing f.f.t.                                    fft00030
c   coded by t. hildenbrand--usgs, denver                                 fft00040
c   (subroutines "sfftmg" & "sdummy" coded by r. watts)                   fft00050
c   structure of command file--                                           fft00060
c    card 1 : operator coded name--from list below                        fft00070
c             psdmag    pseudo_magnetic tranformation                     fft00080
c             psdgrv    pseudo-gravity transformation                     fft00090
c             redpol    reduction of total magnetic field intensity       fft00100
c                        to the north pole                                 fft00110
c             upcont    upward continuation                               fft00120
c             dncont    downward continuation                             fft00130
c             1stver    1st-vertical derivative of the input field        fft00140
c             2ndver    2nd-vertical derivative of the input field        fft00150
c             banpas    bandpass filter                                   fft00160
c             strike    direcional filtering.                             fft00170
c             nofilt    no filtering                                      fft00180
c    card 2 : input file name.ext.                                        fft00190
c    card 3 : output file name.ext.                                       fft00200
c    card 4 : title (col. 1-56).                                          fft00210
c    card 6 : $parms                                                      fft00220
c    card 6 : list parameters--example: iopt1=0,iopt2=1,den=.3,.....      fft00230
c    card 7 : $end.                                                       fft00240
c    the following queries are asked after filter operation               fft00250
c    has been completed. purpose of queries are to continue filtering     fft00260
c    input file with different filters.                                   fft00270
c    query 1 : additional filter to be applied? (y or n) - if "y"         fft00280
c              the program asks queries 2 thru 6. if "n" the              fft00290
c              program starts over by asking for the command segment.     fft00300
c    query 2 : new operator? format same as in card 1 response.           fft00310
c    query 3 : new output file name.ext?                                  fft00320
c    query 4 : new title?                                                 fft00330
c    query 5 : parmameter change? (y or n) - if "n" the filtering         fft00340
c              computations are started. if "y" the user enters the       fft00350
c              parmeters to be changed in a namelist                      fft00360
c              (eg. &parms thet1=-90.,thet2=0.,&)                         fft00370
c              (note the following paramters cannot be changed-nadd,      fft00380
c              iopt2,idval,ddx,ddy,xo,yo).                                fft00390
c    query 1 is then repeated for additional filter operations.           fft00400
c   input parameters:                                                     fft00410
c    iopt1 = 0 no printed output (default: iopt1=0)                       fft00420
c            6 output printed on terminal                                 fft00430
c           -1 output printed on disk.                                    fft00440
c    iopt2 = -1 no removal of mean from input array (default iopt2=-1)    fft00450
c             0 remove mean using boundary values                         fft00460
c             1 remove mean and save grid, file name: con.grd.            fft00470
c    nadd = no. of rows or columns added to each side of grid             fft00480
c           to reduce the effects of gibbs phenomena (default nadd=0).    fft00490
c    w1,w2,w3 & w4 - wavelengths used in bandpass filtering.              fft00500
c                    (default w1=w2=0. w3=w4=1.0e+30 infinte wavelength)fft00510
c    den - density contrast, gm/cc (default den=1.).                      fft00520
c    bmag - magnetization contrast, gammas (default bmag=1.).             fft00530
c    dec & xinc - declination and inclination of earth's field,degrees.   fft00540
c                 (default dec=0. xinc=90.)                               fft00550
c    bdec & binc - declination and inclination of magnetization vector.   fft00560
c                  (default bdec=0. binc=90.)                             fft00570
```

```
c     idval = 0 no flagged grid points in input data (default idval=0)   fft00580
c              1 flagged grid points in input data                       fft00590
c             -1 flagged values removed and locations in file "flag.loc". fft00600
c     icoef = 1 save fourier coefficients in segment "fftfil.coef" for   fft00610
c               later use but perform designated filter operation        fft00620
c               (note: if data contains flagged values a segment         fft00630
c               called "flag.loc" containing their locations is also     fft00640
c               created in users disk area.)                             fft00650
c              0 fourier coefficients not saved (default icoef=0).       fft00660
c             -1 fourier coefficients in segment "fftfil.coef" are used  fft00670
c               as input. note that if flagged values are present in the fft00680
c               data, segment "flag.loc" is required and "idval" must be fft00690
c               equal to -1. in addition, the parameter "nadd" must be   fft00700
c               indentical to its assigned value when the fourier        fft00710
c               coefficients were saved.                                 fft00720
c      z = continuation distance (must be in grid units) (default z=0.)   fft00730
c          for z>0 downward continuation, for z<0 upward continuation     fft00740
c     thet1 & thet2 - angles from geographic north that form a pie-slice fft00750
c                     filter for directional filtering(-90.ge.thet.le.   fft00760
c                     +90.;thet2.gt.thet1). (default thet1=0. thet2=90.) fft00770
c     istr = -1 reject trends between thet1 and thet2.                   fft00780
c             +1 pass (default istr=1).                                  fft00790
c    (following parameters can be used if spacing and origin             fft00800
c    of grid want to be changed. note that these changes                 fft00810
c    only occur in the header of the output filtered grid and no         fft00820
c    subset of the input grid is extracted).                             fft00830
c     ddx - new grid spacing in x-direction.                             fft00840
c     ddy - new grid spacing in y-direction.                             fft00850
c     xo - new origin of rows.                                           fft00860
c     yo - new origin of coloumns. (default ddx=ddy=xo=yo=0.)            fft00870
c*******************************************************************fft00880
      dimension a1(2,1024,16),work(2048),title(14)                      fft00890
      character*50 tname,coname,fname,conout                            fft00900
      character help*4,blank*2,exit*2,fopr*6,flist*6(10),iz*12,blid*4   fft00910
      external set_cc(descriptors),ioa_$nnl(descriptors)                fft00920
      external close_file(descriptors)                                  fft00930
      common/main/id(14),pgm(2),nz,yo,xo,dx,dy,iw,kr,ny,nadd,ir         fft00940
      common/parm1/fname,coname,iopt1,iopt2,idval,icoef,ddx,ddy         fft00950
      common/parm2/den,dec,xinc,bmag,bdec,binc,z,thet1,thet2,istr,w1,w2,fft00960
     1w3,w4                                                             fft00970
      equivalence (id(1),blid)                                          fft00980
      data icmd/10/,lnri/16/                                            fft00990
      data nlist,flist/10,"psdgrv","redpol","upcont","dncont","2ndver", fft01000
     1"strike","banpas","psdmag","1stver","nofilt"/                     fft01010
      data blank/"  "/,exit/"ex"/,help/"help"/                          fft01020
      kr=11                                                             fft01030
      iw=6                                                              fft01040
      ir=5                                                              fft01050
c read command file and header record of input file holding data array. fft01060
   10 call ioa_$nnl(" * ")                                              fft01070
      read(ir,20)tname                                                  fft01080
   20 format(a50)                                                       fft01090
      if(tname.eq.exit.or.tname.eq.blank) go to 350                     fft01100
c  default values of input parameters.                                  fft01110
      iopt1=0                                                           fft01120
      iopt2=-1                                                          fft01130
      icoef=0                                                           fft01140
      nadd=0                                                            fft01150
      w1=0.                                                             fft01160
      w2=0.                                                             fft01170
```

```
      w3=1.0e+30                                                        fft01180
      w4=1.0e+30                                                        fft01190
      den=1.                                                            fft01200
      z=1.                                                              fft01210
      dec=0.0                                                           fft01220
      xinc=90.                                                          fft01230
      bmag=1.                                                           fft01240
      bdec=0.0                                                          fft01250
      binc=90.                                                          fft01260
      xo=0.0                                                            fft01270
      yo=0.0                                                            fft01280
      ddx=0.0                                                           fft01290
      ddy=0.0                                                           fft01300
      thet1=0.0                                                         fft01310
      thet2=90.                                                         fft01320
      istr=1                                                            fft01330
      idval=0                                                           fft01340
      z=0.0                                                             fft01350
c  read command file.                                                   fft01360
      open(10,mode="in",form="formatted",file=tname)                    fft01370
      read(icmd,30)fopr                                                 fft01380
   30 format(a6)                                                        fft01390
      if(fopr.ne.blank) go to 50                                        fft01400
   40 call ioa_$nnl(" enter operator: ")                                fft01410
      read(ir,30)fopr                                                   fft01420
   50 if(fopr.eq.help) go to 80                                         fft01430
      do 60 i=1,nlist                                                   fft01440
      if(fopr.ne.flist(i)) go to 60                                     fft01450
      itype=i                                                           fft01460
      go to 110                                                         fft01470
   60 continue                                                          fft01480
      write(iw,70)                                                      fft01490
   70 format(" #invalid operator")                                      fft01500
   80 write(iw,90)                                                      fft01510
   90 format(" quick list of options")                                  fft01520
      write(iw,100)flist                                                fft01530
  100 format(2x,a6)                                                     fft01540
      go to 40                                                          fft01550
  110 read(icmd,20)fname                                                fft01560
      if(fname.ne.blank) go to 120                                      fft01570
      call ioa_$nnl(" enter input file name: ")                         fft01580
      read(ir,20)fname                                                  fft01590
  120 read(icmd,20)coname                                               fft01600
      if(coname.ne.blank) go to 130                                     fft01610
      call ioa_$nnl(" enter output file name: ")                        fft01620
      read(ir,20)coname                                                 fft01630
  130 read(icmd,140)id                                                  fft01640
      if(blid.ne.blank) go to 150                                       fft01650
      call ioa_$nnl(" enter title: ")                                   fft01660
      read(ir,140)id                                                    fft01670
  140 format(14a4)                                                      fft01680
  150 call snarel(icmd)                                                 fft01690
      close(10)                                                         fft01700
c  test for parameter errors.                                           fft01710
      if(z.ge.0.0.and.fopr.eq."upcont") go to 300                       fft01720
      if(z.le.0.0.and.fopr.eq."dncont") go to 300                       fft01730
      if(fopr.eq."nofilt".and.icoef.eq.0) go to 280                     fft01740
      if(abs(thet1).gt.90..or.abs(thet2).gt.90.) go to 280              fft01750
      if(thet1.gt.thet2) go to 280                                      fft01760
      if(w1.gt.w2.or.w2.gt.w3.or.w3.gt.w4) go to 280                    fft01770
```

```
      if(abs(binc).gt.90..or.abs(xinc).gt.90.) go to 280                fft01780
      if(iopt1)170,180,160                                              fft01790
  160 if(iopt1.ne.6) go to 280                                          fft01800
      go to 180                                                         fft01810
  170 iopt1=20                                                          fft01820
      conout="fftfil.out"                                               fft01830
      open(20,mode="inout",form="formatted",file=conout)                fft01840
      call set_cc("file20","-on")                                       fft01850
c  read header record of input file.                                    fft01860
  180 open(11,mode="in",form="unformatted",file=fname)                  fft01870
c  note that x & y have been swicthed from normal (usgs) grid specficatifft01880
c  ...convention used is x-north y-east and z-down                      fft01890
      read(kr)title,pgm,n2,n1,nz,yop,dy,xop,dx                          fft01900
      if(icoef.eq.-1.and.fopr.eq."nofilt")itype=11                      fft01910
      pgm(1)="fftf"                                                     fft01920
      pgm(2)="il**"                                                     fft01930
      nx=n1                                                             fft01940
      ny=n2                                                             fft01950
      if(ddx.ne.0.0)dx=ddx                                              fft01960
      if(ddy.ne.0.0)dy=ddy                                              fft01970
      if(xo.ne.0.0)xop=xo                                               fft01980
      if(yo.ne.0.0)yop=yo                                               fft01990
      xo=xop                                                            fft02000
      yo=yop                                                            fft02010
      l=lnri+1                                                          fft02020
      lnri21=lnri/2+1                                                   fft02030
c  set no. of rows for fft: need m=l*2**k, m.gt. or .eq. nx+2*nadd.      fft02040
c  m=no. of rows, l=no. from 9-16, k=interger.                          fft02050
      n1=n1+2*nadd                                                      fft02060
  190 l=l-1                                                             fft02070
      if(l.lt.lnri21) go to 260                                         fft02080
      mr=n1/l+0.0000001                                                 fft02090
      k=1                                                               fft02100
      idiv=2                                                            fft02110
  200 iquot=mr/idiv+0.0000001                                           fft02120
      if(iquot.lt.idiv) go to 210                                       fft02130
      k=k+1                                                             fft02140
      mr=iquot                                                          fft02150
      go to 200                                                         fft02160
  210 k=k+1                                                             fft02170
      m=l*2**k                                                          fft02180
  220 mtest=l*2**(k-1)                                                  fft02190
      if(mtest.lt.n1) go to 230                                         fft02200
      k=k-1                                                             fft02210
      m=mtest                                                           fft02220
      if(k.eq.0) go to 230                                              fft02230
      go to 220                                                         fft02240
  230 lnxa=m-n1                                                         fft02250
      if(l.ne.lnri) go to 250                                           fft02260
      nxa16=lnxa                                                        fft02270
  240 nri=l                                                             fft02280
      nxa=lnxa                                                          fft02290
      go to 190                                                         fft02300
  250 if(lnxa.ge.nxa) go to 190                                         fft02310
      go to 240                                                         fft02320
  260 n1=n1+nxa                                                         fft02330
c  check to see if row block size of 16 will be more efficient         fft02340
      n116=n1-nxa+nxa16                                                 fft02350
      ntest=0.9*n116                                                    fft02360
      if(ntest.gt.n1.or.n116.gt.1024) go to 270                         fft02370
```

```
      n1=n1-nxa+nxa16                                                    fft02380
      nxa=nxa16                                                          fft02390
      nri=16                                                             fft02400
  270 n2=n2+2*nadd                                                       fft02410
      if(n1.gt.1024.or.n2.gt.1024) go to 320                             fft02420
      nxa=nxa+2*nadd                                                     fft02430
      id2=n1                                                             fft02440
      if(n2.gt.n1)id2=n2                                                 fft02450
      id2=2*id2                                                          fft02460
c  input parameters for fft are set: call main program.                  fft02470
      call sfftfil(a1,work,nx,nri,itype,nxa,n1,n2,id2)                   fft02480
      call set_cc("file20","-off")                                       fft02490
      if(iopt1.eq.20)close(20)                                           fft02500
      go to 10                                                           fft02510
c--errors resulting in job abortion.                                     fft02520
  280 write(iw,290)                                                      fft02530
  290 format(" #parameter error detected---case run aborted")            fft02540
      go to 10                                                           fft02550
  300 write(iw,310)                                                      fft02560
  310 format(" #sign of z does not conform to continuation operator")    fft02570
      go to 10                                                           fft02580
  320 write(iw,330)nx,ny,n1,n2                                           fft02590
  330 format(" #no. of extended rows or columns exceeds 1024:"/          fft02600
     1" input no. of rows and columns="2i4/                              fft02610
     2" no. of rows and columns required for filtering=",2i4,/)          fft02620
  340 close(11)                                                          fft02630
      go to 10                                                           fft02640
  350 call close_file("-all")                                            fft02650
      stop                                                               fft02660
      end                                                                fft02670


c*********************************************************************fft02680
c     subroutine "snamel" reads parms from namelist section            fft02690
c     iread=5 (terminal read) or 10 (read from file10)                  fft02700
c*********************************************************************fft02710
      subroutine snamel(iread)                                           fft02720
      character*50 fname,coname                                          fft02730
      common/main/id(14),pgm(2),nz,yo,xo,dx,dy,iw,kr,ny,nadd,ir          fft02740
      common/parm1/fname,coname,iopt1,iopt2,idval,icoef,ddx,ddy          fft02750
      common/parm2/den,dec,xinc,bmag,bdec,binc,z,thet1,thet2,istr,w1,w2,fft02760
     1w3,w4                                                              fft02770
      namelist/parms/iopt1,iopt2,den,dec,xinc,bmag,bdec,binc,idval,ddx,  fft02780
     1ddy,xo,yo,thet1,thet2,istr,nadd,w1,w2,w3,w4,icoef,z                fft02790
      read(iread,parms,end=10)                                           fft02800
      return                                                             fft02810
   10 write(iw,20)                                                       fft02820
   20 format("#odd eof in parms command file--program aborted")          fft02830
      if(iread.eq.10)close(10)                                           fft02840
      stop                                                               fft02850
      end                                                                fft02860


c*********************************************************************fft02870
c   subroutine "sfftfil" removes flagged values from grid, transforms   fft02880
c   data, calls routines to apply filter, and writes filtered grid      fft02890
c*********************************************************************fft02900
      subroutine sfftfil(a1,work,nx,nri,itype,nxa,n1,n2,id2)             fft02910
      dimension a1(2,n2,nri),work(id2),jflag(2,20),title(14)             fft02920
      character flist*6(10),help*4,quer*1,fopr*6                         fft02930
```

```
      character*50 coname,fname,fileo,flgloc                            fft02940
      external rename(descriptors),delete(descriptors)                  fft02950
      external io(descriptors)                                          fft02960
      external ioa_(descriptors),ioa_$nnl(descriptors)                  fft02970
      common/main/id(14),pgm(2),nz,yo,xo,dx,dy,iw,kr,ny,nadd,ir         fft02980
      common/basic/c1,el,em,en,bl,bm,bn                                 fft02990
      common/parm1/fname,coname,iopt1,iopt2,idval,icoef,ddx,ddy         fft03000
      common/parm2/den,dec,xinc,bmag,bdec,binc,z,thet1,thet2,istr,w1    fft03010
     1    ,w2,w3,w4                                                     fft03020
      common/tr/prad,p,q,th1p,th2p                                      fft03030
      data help/"help"/,nlist,flist/10,"psdgrv","redpol","upcont",      fft03040
     1   "dncont","2ndver","strike","banpas","psdmag","1stver","nofilt"/fft03050
      data iunita/10/,iunitb/12/,dval/o376777777777/                    fft03060
      data gamma/6.67323/,pi2/6.28318531/,pi/3.141592654/,iunitc/15/    fft03070
      ncase=0                                                           fft03080
      if(icoef.ne.-1) go to 10                                          fft03090
      ncase=1                                                           fft03100
     open(15,access="direct",mode="inout",form="unformatted",           fft03110
     1  file="fftfil.coef")                                            fft03120
     open(12,access="direct",mode="inout",form="unformatted",           fft03130
     1  file="slave2.tmp")                                             fft03140
      close(11)                                                         fft03150
      go to 20                                                          fft03160
c   remove mean using boundary values.                                  fft03170
10    if(iopt2.lt.0) go to 20                                           fft03180
      call srmean(nx,ny,a1)                                             fft03190
      fileo="fftfil.grd"                                                fft03200
      open(11,mode="inout",form="unformatted",file=fileo)               fft03210
      read(kr)title,pgm,ny,nx,nz,yo,dy,xo,dx                            fft03220
c  compute beginning and end column & row in original grid.             fft03230
20    do 30 i=1,2                                                       fft03240
      do 30 j=1,n2                                                      fft03250
30    a1(2,j,i)=0.0                                                     fft03260
      nap1=nadd+1                                                       fft03270
      n22=2*n2                                                          fft03280
      n2a=n2-nadd                                                       fft03290
      nxap1=nxa*0.5+1                                                   fft03300
      fxap1=float(nxa)/2.+1.                                            fft03310
      if(abs(fxap1-float(nxap1)).gt.0.0001)nxap1=nxap1+1                fft03320
      n1a=nxap1-1+nx                                                    fft03330
c  remove flagged values and add "2*nadd" rows & columns to             fft03340
c  reduce the effect of gibbs phenomena.                                fft03350
     open(10,access="direct",mode="inout",form="unformatted",           fft03360
     1  file="slave1.tmp")                                             fft03370
      if(ncase.eq.1) go to 590                                          fft03380
c   store flagged values in file "flag.tmp"                             fft03390
      if(idval.eq.1)open(14,mode="inout",form="unformatted",            fft03400
     1  file="flag.tmp")                                               fft03410
      irr=1                                                             fft03420
      iwr=0                                                             fft03430
      jr=0                                                              fft03440
40    jr=jr+1                                                           fft03450
      if(idval.lt.1) go to 310                                          fft03460
c   idval>0: removal of flagged values.                                 fft03470
      if(jr.gt.1) go to 140                                             fft03480
c   read 1st row & remove flagged values.                               fft03490
      read(kr)dum,(a1(1,i,1),i=nap1,n2a)                                fft03500
      read(kr)dum,(a1(1,i,2),i=nap1,n2a)                                fft03510
      nflag=1                                                           fft03520
      i=nap1                                                            fft03530
```

```
   50       jflag(1,1)=nap1                                                  fft03540
            if(a1(1,nap1,1).lt.dval) go to 90                                fft03550
            do 60 i=nap1,n2a                                                 fft03560
   60       if(a1(1,i,1).lt.dval) go to 70                                   fft03570
            go to 1270                                                       fft03580
   70       jflag(1,1)=i                                                     fft03590
            aval=a1(1,i,1)                                                   fft03600
            do 80 ii=nap1,i-1                                                fft03610
   80       a1(1,ii,1)=aval                                                  fft03620
   90       if(i.ge.n2a) go to 100                                           fft03630
            do 100 ii=i+1,n2a                                                fft03640
            if(a1(1,ii,1).lt.dval) go to 100                                 fft03650
            jflag(2,nflag)=ii-1                                              fft03660
            go to 110                                                        fft03670
  100       continue                                                         fft03680
            jflag(2,nflag)=n2a                                               fft03690
            write(14)nflag,(jflag(1,j),jflag(2,j),j=1,nflag)                 fft03700
            go to 350                                                        fft03710
  110       aval=a1(1,ii-1,1)                                                fft03720
            do 120 i=ii,n2a                                                  fft03730
            if(a1(1,i,1).lt.dval) go to 130                                  fft03740
  120       a1(1,i,1)=aval                                                   fft03750
            write(14)nflag,(jflag(1,j),jflag(2,j),j=1,nflag)                 fft03760
            go to 350                                                        fft03770
  130       a1(1,i-1,1)=(a1(1,i-1,1)+a1(1,i,1))*0.5                          fft03780
            nflag=nflag+1                                                    fft03790
            jflag(1,nflag)=i                                                 fft03800
c   1st row completed: now extend grid & write to disk.                      fft03810
            go to 90                                                         fft03820
c   remove flagged values from remaining rows.                               fft03830
  140       if(jr.ge.nx) go to 150                                           fft03840
            read(kr)dum,(a1(1,i,3),i=nap1,n2a)                               fft03850
            go to 170                                                        fft03860
  150       do 160 i=nap1,n2a                                                fft03870
  160       a1(1,i,3)=dval                                                   fft03880
  170       nflag=1                                                          fft03890
            i=nap1                                                           fft03900
            jflag(1,1)=nap1                                                  fft03910
            if(a1(1,nap1,2).lt.dval) go to 220                               fft03920
            do 180 i=nap1,n2a                                                fft03930
  180       if(a1(1,i,2).lt.dval) go to 190                                  fft03940
            i=nap1                                                           fft03950
            jflag(1,1)=n2a+1                                                 fft03960
            if(a1(1,nap1,3).eq.dval)a1(1,nap1,2)=a1(1,nap1,1)                fft03970
            if(a1(1,nap1,3).lt.dval)a1(1,nap1,2)=(a1(1,nap1,1)+              fft03980
     1      a1(1,nap1,3))*0.5                                                fft03990
            go to 220                                                        fft04000
  190       jflag(1,1)=i                                                     fft04010
            ii=i                                                             fft04020
            do 210 irev=nap1,i-1                                             fft04030
            ii=ii-1                                                          fft04040
            if(a1(1,ii,3).eq.dval) go to 200                                 fft04050
            a1(1,ii,2)=(a1(1,ii,1)+a1(1,ii+1,2)+a1(1,ii,3))*0.3333333        fft04060
            go to 210                                                        fft04070
  200       a1(1,ii,2)=(a1(1,ii,1)+a1(1,ii+1,2))*0.5                         fft04080
  210       continue                                                         fft04090
  220       if(i.ge.n2a) go to 230                                           fft04100
            do 230 ii=i+1,n2a                                                fft04110
            if(a1(1,ii,2).lt.dval) go to 230                                 fft04120
            jflag(2,nflag)=ii-1                                              fft04130
```

```
      go to 240                                                          fft04140
230   continue                                                           fft04150
      jflag(2,nflag)=n2a                                                 fft04160
      write(14)nflag,(jflag(1,j),jflag(2,j),j=1,nflag)                   fft04170
      go to 390                                                          fft04180
240   do 290 i=ii,n2a                                                   fft04190
      if(i.eq.n2a) go to 250                                             fft04200
      if(a1(1,i+1,2).lt.dval) go to 270                                  fft04210
250   if(a1(1,i,3).eq.dval) go to 260                                    fft04220
      a1(1,i,2)=(a1(1,i,1)+a1(1,i-1,2)+a1(1,i,3))*0.3333333              fft04230
      go to 290                                                          fft04240
260   a1(1,i,2)=(a1(1,i,1)+a1(1,i-1,2))*0.5                              fft04250
      go to 290                                                          fft04260
270   if(a1(1,i,3).eq.dval) go to 280                                    fft04270
      a1(1,i,2)=(a1(1,i,1)+a1(1,i-1,2)+a1(1,i+1,2)+a1(1,i,3))*0.25       fft04280
      go to 300                                                          fft04290
280   a1(1,i,2)=(a1(1,i,1)+a1(1,i-1,2)+a1(1,i+1,2))*0.3333333            fft04300
      go to 300                                                          fft04310
290   continue                                                           fft04320
      write(14)nflag,(jflag(1,j),jflag(2,j),j=1,nflag)                   fft04330
      go to 390                                                          fft04340
300   nflag=nflag+1                                                      fft04350
      i=i+1                                                              fft04360
      jflag(1,nflag)=i                                                   fft04370
      go to 220                                                          fft04380
c   idval.eq.0: read & check input data.                                 fft04390
310   if(jr.ne.1)irr=2                                                   fft04400
      read(kr)dum,(a1(1,i,irr),i=nap1,n2a)                               fft04410
      do 320 ii=nap1,n2a                                                 fft04420
320   if(a1(1,ii,irr).eq.dval) go to 330                                 fft04430
      if(jr.ne.1) go to 390                                              fft04440
      go to 350                                                          fft04450
330   write(iw,340)jr,ii                                                 fft04460
340   format(" #flagged value row=",i4,3x,"col=",i4," may be more?")     fft04470
      close(11)                                                          fft04480
      close(10)                                                          fft04490
      return                                                             fft04500
c   extend row & write to disk.                                          fft04510
350   do 360 i=1,nap1                                                    fft04520
360   a1(1,i,1)=a1(1,nap1,1)                                             fft04530
      do 370 i=n2a,n2                                                    fft04540
370   a1(1,i,1)=a1(1,n2a,1)                                              fft04550
380   iwr=iwr+1                                                          fft04560
      call swrda(iunita,iwr,a1,n22)                                      fft04570
      if(iwr.eq.nxap1) go to 40                                          fft04580
      go to 380                                                          fft04590
390   do 400 i=1,nap1                                                    fft04600
400   a1(1,i,2)=a1(1,nap1,2)                                             fft04610
      do 410 i=n2a,n2                                                    fft04620
410   a1(1,i,2)=a1(1,n2a,2)                                              fft04630
      iwr=iwr+1                                                          fft04640
      call swrda(iunita,iwr,a1(1,1,2),n22)                               fft04650
      if(iwr.ge.n1a) go to 430                                           fft04660
      if(idval.lt.1) go to 40                                            fft04670
      do 420 i=nap1,n2a                                                  fft04680
      a1(1,i,1)=a1(1,i,2)                                                fft04690
420   a1(1,i,2)=a1(1,i,3)                                                fft04700
      go to 40                                                           fft04710
430   if(iwr.ge.n1) go to 450                                            fft04720
440   iwr=iwr+1                                                          fft04730
```

```
      call swrda(iunita,iwr,a1(1,1,2),n22)                              fft04740
      if(iwr.lt.n1) go to 440                                           fft04750
450   if(idval.eq.1)close(14)                                           fft04760
      close(11)                                                         fft04770
      if(iopt2.eq.0)call delete("fftfil.grd")                           fft04780
460   format(i5)                                                        fft04790
c  write headings                                                       fft04800
470   if(ncase.eq.0) go to 590                                          fft04810
480   call ioa_$nnl(" enter new operator: ")                            fft04820
      read(ir,490)fopr                                                  fft04830
490   format(a6)                                                       fft04840
500   if(fopr.eq.help) go to 530                                        fft04850
      do 510 i=1,nlist                                                  fft04860
      if(fopr.ne.flist(i)) go to 510                                    fft04870
      itype=i                                                           fft04880
      go to 560                                                         fft04890
510   continue                                                          fft04900
      write(iw,520)                                                     fft04910
520   format(" #invalid operator")                                      fft04920
530   write(iw,540)                                                     fft04930
540   format(" quick list of options")                                  fft04940
      write(iw,550)flist                                                fft04950
550   format(2x,a6)                                                     fft04960
      go to 480                                                         fft04970
560   call ioa_$nnl(" enter new output file name: ")                    fft04980
      read(ir,570)coname                                                fft04990
570   format(v)                                                         fft05000
      call ioa_(" enter new title")                                     fft05010
      read(ir,580)id                                                    fft05020
580   format(14a4)                                                      fft05030
      call ioa_$nnl(                                                    fft05040
     1" parm change? (y or n, if y enter parameters via &parms): ")     fft05050
      read(ir,570)quer                                                  fft05060
      if(quer.ne."y") go to 590                                         fft05070
      call snamel(ir)                                                   fft05080
590   ip=iopt1                                                          fft05090
      ih=iw                                                             fft05100
      if(ip.ne.0)ih=ip                                                  fft05110
      pid=pi/180.                                                       fft05120
      rn1=1./(float(n1)*dx)                                             fft05130
      rn2=1./(float(n2)*dy)                                             fft05140
      c2=pi2*z                                                          fft05150
      go to (620,640,670,690,710,730,780,600,890,930,910),itype         fft05160
600   write(ih,610)                                                     fft05170
610   format(1h1,"transformation of the gravity field to the ",         fft05180
     1"pseudo-magnetic field,",/,," by means of the fast fourier",      fft05190
     2" transform and poissons equation.",/)                            fft05200
      go to 640                                                         fft05210
620   write(ih,630)                                                     fft05220
630   format(1h1,"transformation of total magnetic intensity field",    fft05230
     1" to pseudo-gravitational field,",/,"by means of the fast ",      fft05240
     2"fourier transform and the poisson equation.",/)                  fft05250
640   rbdec=bdec*pid                                                    fft05260
      rbinc=binc*pid                                                    fft05270
      cosrb=cos(rbinc)                                                  fft05280
      bl=cosrb*cos(rbdec)                                               fft05290
      bm=cosrb*sin(rbdec)                                               fft05300
      bn=sin(rbinc)                                                     fft05310
      rdec=dec*pid                                                      fft05320
      rinc=xinc*pid                                                     fft05330
```

```
      cosr=cos(rinc)                                                    fft05340
      el=cosr*cos(rdec)                                                 fft05350
      em=cosr*sin(rdec)                                                 fft05360
      en=sin(rinc)                                                      fft05370
      if(itype.eq.2) go to 650                                          fft05380
      c1=(gamma*den)/(pi2*bmag)                                         fft05390
      go to 950                                                         fft05400
650   write(ih,660)                                                     fft05410
660   format(1h1,"transformation of the total magnetic field ",         fft05420
     1 "intensity to the pole,",/," by means of the fast fourier ",     fft05430
     2 "transform and the poisson equation.",/)                         fft05440
      go to 950                                                         fft05450
670   write(ih,680)                                                     fft05460
680   format(1h1,"upward continuation of field values,",/,              fft05470
     1 " by means of the fast fourier transform.",/)                    fft05480
      go to 950                                                         fft05490
690   write(ih,700)                                                     fft05500
700   format(1h1,"downward continuation of field values,",/,            fft05510
     1 " by means of the fast fourier transform.",/)                    fft05520
      go to 950                                                         fft05530
710   write(ih,720)                                                     fft05540
720   format(1h1,"second vertical derivative operation,",/,             fft05550
     1 " by means of the fast fourier transform.",/)                    fft05560
      c4=4.*pi*pi                                                       fft05570
      go to 950                                                         fft05580
730   write(ih,740)                                                     fft05590
740   format(1h1,"strike sensitive filtering using the fast ",          fft05600
     1 "fourier transform.",/)                                          fft05610
      p=1.                                                              fft05620
      q=0.0                                                             fft05630
      if(istr.ne.1) go to 760                                           fft05640
      if(thet1.lt.0.0.and.thet2.ge.0.0) go to 750                       fft05650
      go to 770                                                         fft05660
750   p=0.0                                                             fft05670
      q=1.                                                              fft05680
      go to 770                                                         fft05690
760   if(thet1.lt.0.0.and.thet2.ge.0.0) go to 770                       fft05700
      p=0.0                                                             fft05710
      q=1.                                                              fft05720
770   th1p=180.-thet1                                                   fft05730
      th2p=180.-thet2                                                   fft05740
      if(th1p.gt.180.)th1p=th1p-180.                                    fft05750
      if(th2p.gt.180.)th2p=th2p-180.                                    fft05760
      prad=180./pi                                                      fft05770
      go to 950                                                         fft05780
780   write(ih,790)                                                     fft05790
790   format(1h1,"band-pass filter applied to data",/)                  fft05800
      if(w1.ne.0.0) go to 800                                           fft05810
      f4=1.0e+30                                                        fft05820
      go to 810                                                         fft05830
800   f4=1./w1                                                          fft05840
810   if(w2.ne.0.0) go to 820                                           fft05850
      f3=1.0e+30                                                        fft05860
      go to 830                                                         fft05870
820   f3=1./w2                                                          fft05880
830   if(w3.ne.1.0e+30) go to 840                                       fft05890
      f2=0.                                                             fft05900
      go to 850                                                         fft05910
840   f2=1./w3                                                          fft05920
850   if(w4.ne.1.0e+30) go to 860                                       fft05930
```

```
      f1=0.                                                          fft05940
      go to 870                                                      fft05950
  860 f1=1./w4                                                       fft05960
  870 if(w3.eq.w4) go to 880                                         fft05970
      btan12=1./(f2-f1)                                              fft05980
  880 if(w1.eq.w2) go to 950                                         fft05990
      btan34=1./(f4-f3)                                              fft06000
      go to 950                                                      fft06010
  890 write(ih,900)                                                  fft06020
  900 format(1h1,"first vertical derivative operation,",/,           fft06030
     1" by means of the fast fourier transform.",/)                  fft06040
      c4=pi2                                                         fft06050
      go to 950                                                      fft06060
  910 write(ih,920)                                                  fft06070
  920 format(1h1," no filter operation performed: fourier coeff.'s",fft06080
     1" are inputed and inverse transformed",/)                      fft06090
      go to 950                                                      fft06100
  930 write(ih,940)                                                  fft06110
  940 format(1h1," no filter operation performed: only fourier ",    fft06120
     1"coef.s are outputed in fftfil.coef",/)                        fft06130
  950 write(ih,960)id,den,z,bmag,xinc,dec,binc,bdec,thet1,thet2,     fft06140
     1 istr,w1,w2,w3,w4,dx,dy,xo,yo,ny,nx,nadd,n2,n1,fname,coname    fft06150
  960 format(" title: ",14a4,/" parameters: censity=",f7.3,5x,       fft06160
     1"datum level z=",f7.3/13x,"intensity of magnetization=", f9.3 fft06170
     2/13x,"inclination & declination of geomagnetic field=",2f9.3  fft06180
     3/13x,"inclination & declination of magnetization vector=",    fft06190
     4 2f9.3/13x,"thet1=",f9.3,5x,"thet2=",f9.3,5x,"istr=",i2/13x,  fft06200
     5 "w1,w2,w3&w4=",4e11.4/13x,"dx & dy=",2e14.6/13x,             fft06210
     6 "origin of grid(xo & yo)=",2e14.6/13x,"no. of columns & rows="fft06220
     7 ,2i5/13x,"nadd=",i5,"  extended columns & rows=",2i5/13x,    fft06230
     8 "input file name: ",a50/13x,"output file name: ",a50,/)      fft06240
c  obtain complex fourier coefficients (f.f.t. a1).                 fft06250
      if(ncase.eq.0) go to 980                                       fft06260
      jr=0                                                           fft06270
      do 970 i=1,n1                                                  fft06280
      jr=jr+1                                                        fft06290
      call srdda(iunitc,jr,a1,n22)                                   fft06300
  970 call swrda(iunita,jr,a1,n22)                                   fft06310
      if(itype.eq.11) go to 1110                                     fft06320
      go to 1000                                                     fft06330
  980 isign=-1                                                       fft06340
      open(12,access="direct",mode="inout",form="unformatted",       fft06350
     1 file="slave2.tmp")                                            fft06360
      call sfftmg(iunita,iunitb,n2,n1,a1,nri,isign,work)             fft06370
      open(15,access="direct",mode="inout",form="unformatted",       fft06380
     1 file="fftfil.coef")                                           fft06390
      jr=0                                                           fft06400
      do 990 i=1,n1                                                  fft06410
      jr=jr+1                                                        fft06420
      call srdda(iunita,jr,a1,n22)                                   fft06430
  990 call swrda(iunitc,jr,a1,n22)                                   fft06440
      if(itype.eq.10) go to 1250                                     fft06450
c  a1 array now holds complex fourier coefficients.                 fft06460
 1000 n1p1=n1+1                                                      fft06470
      n2p1=n2+1                                                      fft06480
      nnh1=(n1/2)+1                                                  fft06490
      nnh2=(n2/2)+1                                                  fft06500
c  operating on a1 fourier coefficients.                            fft06510
      jr=0                                                           fft06520
      jmr=n1+1                                                       fft06530
```

```
 1010     jr=jr+1                                                          fft06540
          if(jr.gt.nnh1) go to 1110                                        fft06550
          jj=jr-1                                                          fft06560
          x=float(jj)*rn1                                                  fft06570
          xsq=x*x                                                          fft06580
          call srdda(iunita,jr,a1,n22)                                     fft06590
          do 1090 i=1,n2                                                   fft06600
          ii=i-1                                                           fft06610
          if(i.gt.nnh2)ii=-(n2p1-i)                                        fft06620
          y=float(ii)*rn2                                                  fft06630
          xysq=xsq+y*y                                                     fft06640
          go to (1020,1020,1030,1030,1040,1050,1060,1020,1070),itype       fft06650
 1020     if(xysq.eq.0.0) go to 1080                                       fft06660
          call spspol(itype,x,y,xysq,a1(1,i,1),a1(2,i,1))                  fft06670
          go to 1090                                                       fft06680
 1030     call scont(c2,xysq,a1(1,i,1),a1(2,i,1))                          fft06690
          go to 1090                                                       fft06700
 1040     call ssvert(c4,xysq,a1(1,i,1),a1(2,i,1))                         fft06710
          go to 1090                                                       fft06720
 1050     if(xysq.eq.0.0) go to 1090                                       fft06730
          call strend(x,y,a1(1,i,1),a1(2,i,1))                             fft06740
          go to 1090                                                       fft06750
 1060     call sbpass(a1(1,i,1),a1(2,i,1),f1,f2,f3,f4,btan12,btan34,       fft06760
     1    xysq)                                                            fft06770
          go to 1090                                                       fft06780
 1070     call sfvert(c4,xysq,a1(1,i,1),a1(2,i,1))                         fft06790
          go to 1090                                                       fft06800
 1080     a1(1,1,1)=0.0                                                    fft06810
          a1(2,1,1)=0.0                                                    fft06820
 1090     continue                                                         fft06830
          call swrda(iunita,jr,a1,n22)                                     fft06840
          if(jr.eq.1) go to 1010                                           fft06850
          jmr=jmr-1                                                        fft06860
          if(jr.eq.jmr) go to 1110                                         fft06870
          a1(2,1,1)=-a1(2,1,1)                                             fft06880
          iwr=n2p1                                                         fft06890
          do 1100 i=2,nnh2                                                 fft06900
          iwr=iwr-1                                                        fft06910
          x=a1(2,i,1)                                                      fft06920
          y=a1(1,i,1)                                                      fft06930
          a1(2,i,1)=-a1(2,iwr,1)                                           fft06940
          a1(1,i,1)=a1(1,iwr,1)                                            fft06950
          a1(1,iwr,1)=y                                                    fft06960
          a1(2,iwr,1)=-x                                                   fft06970
 1100     continue                                                         fft06980
          call swrda(iunita,jmr,a1,n22)                                    fft06990
          go to 1010                                                       fft07000
c  a1 array now holds complex coefficients operated on by filter          fft07010
c  compute desired anomaly = inv. f.f.t. of a1                            fft07020
 1110     isign=1                                                          fft07030
          call sfftmg(iunita,iunitb,n2,n1,a1,nri,isign,work)               fft07040
          area=1./float(n1*n2)                                             fft07050
c  a1 array now holds computed anomaly on datum = z.                      fft07060
c  create standard output file and list output.                           fft07070
          if(iopt1.eq.0) go to 1120                                        fft07080
          write(ip,1240)id,z                                               fft07090
          f16=float(ny)/16.                                                fft07100
          i16=ny/16                                                        fft07110
          if(abs(f16-float(i16)).gt.0.0001)i16=i16+1                       fft07120
 1120     open(11,mode="inout",form="unformatted",file=coname)             fft07130
```

```
      write(kr)id,pgm,ny,nx,nz,yo,dy,xo,dx                              fft07140
      if(idval.eq.1)open(14,mode="in",form="unformatted",               fft07150
     1  file="flag.tmp")                                                fft07160
      flgloc="flag.loc"                                                 fft07170
      if(idval.lt.0)open(14,mode="in",form="unformatted",               fft07180
     1  file=flgloc)                                                    fft07190
      dum=C.C                                                           fft07200
      iwr=0                                                             fft07210
      do 1210 jr=nxap1,n1a                                              fft07220
      iwr=iwr+1                                                         fft07230
      call srdda(iunita,jr,a1,n22)                                      fft07240
      if(idval.eq.0) go to 1170                                         fft07250
      read(14)nflag,(jflag(1,j),jflag(2,j),j=1,nflag)                   fft07260
      j=1                                                               fft07270
c   restore shape of original grid (i.e. insert flagged values).        fft07280
      do 1160 i=nap1,n2a                                                fft07290
      if(i.lt.jflag(1,j)) go to 1130                                    fft07300
      if(i.eq.jflag(2,j)) go to 1140                                    fft07310
      go to 1150                                                        fft07320
1130  a1(1,i,1)=dval                                                    fft07330
      go tc 1160                                                        fft07340
1140  j=j+1                                                             fft07350
      if(j.le.nflag) go to 1150                                         fft07360
      j=j-1                                                             fft07370
      jflag(1,j)=1024                                                   fft07380
1150  a1(1,i,1)=a1(1,i,1)*area                                          fft07390
1160  continue                                                          fft07400
      go to 1190                                                        fft07410
1170  do 1180 i=nap1,n2a                                                fft07420
1180  a1(1,i,1)=a1(1,i,1)*area                                          fft07430
1190  write(kr)dum,(a1(1,i,1),i=nap1,n2a)                               fft07440
      if(iopt1.eq.0) go to 1210                                         fft07450
      write(ip,1220)iwr                                                 fft07460
      ih=nap1-1                                                         fft07470
      do 1200 ii=1,i16                                                  fft07480
      iout=ih+1                                                         fft07490
      ih=iout+15                                                        fft07500
      if(ih.gt.n2a)ih=n2a                                               fft07510
1200  write(ip,1230)(a1(1,i,1),i=iout,ih)                               fft07520
1210  continue                                                          fft07530
1220  format(/,1x,"row ",i3)                                            fft07540
1230  format(3x,16f7.1)                                                 fft07550
1240  format(1h1,14a4,//" calculated anomaly on level z=",f7.3,//)      fft07560
      close(11)                                                         fft07570
      if(idval.ne.0)close(14)                                           fft07580
     call ioa_$nnl(" additional filtering to original grid?(y or n) ")  fft07590
      read(ir,570)quer                                                  fft07600
      if(quer.ne."y") go to 1250                                        fft07610
      ncase=1                                                           fft07620
      go to 470                                                         fft07630
1250  if(idval.ne.1) go to 1260                                         fft07640
      if(icoef.eq.0)call delete("flag.tmp")                             fft07650
      if(icoef.gt.0)call rename("flag.tmp","flag.loc")                  fft07660
1260  close(10)                                                         fft07670
      close(12)                                                         fft07680
      close(15)                                                         fft07690
      call delete("slave1.tmp")                                         fft07700
      call delete("slave2.tmp")                                         fft07710
      if(icoef.eq.0)call delete("fftfil.coef")                          fft07720
      go to 1290                                                        fft07730
```

```
 1270      write(6,1280)                                                  fft07740
 1280      format(" #entire first row is flagged values-program aborted") fft07750
           close(11)                                                      fft07760
           close(10)                                                      fft07770
           if(idval.eq.1)close(14)                                        fft07780
 1290      return                                                         fft07790
           end                                                            fft07800


c**********************************************************************fft07810
      subroutine sfftmg(unita,unitb,n,m,f,l,isign,work)                 fft07820
c     performs 2-dimensional fast fourier transform using disk          fft07830
c     arguments -                                                       fft07840
c       unita - unit number of open random access disk file containing  fft07850
c               one complex data row per record.                        fft07860
c               this file is used for input to fftmg and output         fft07870
c               of the transform from fftmg.                            fft07880
c       unitb - unit number of open random access work file             fft07890
c               the same as the file of unita.                          fft07900
c       n     - number of y columns = length of x rows.                 fft07910
c       m     - number of x rows = length of y columns,                 fft07920
c               must be of the form l*2**k (k integer).                 fft07930
c       f     - complex work array for core manipulations.              fft07940
c       l     - number of x rows which can be held in f.                fft07950
c       isign - +1 or -1 for forward or reverse transform.              fft07960
c       work  - work array for in-core transform routine sfourt.        fft07970
c               work must be complex, of length max(n,l).               fft07980
c   developed and coded by ray watts--usgs,denver                       fft07990
c   note: randin,randou,inset, and outset are entry points              fft08000
c   in subroutine sdummy                                                fft08010
c**********************************************************************fft08020
      integer unita,unitb,output,ndim(2)                                fft08030
      complex f(n,l),twidle,twidlf,w,work(1)                            fft08040
      nfourt=2*n*l                                                      fft08050
      nwork=2*n                                                         fft08060
      if(l.gt.n)nwork=2*l                                               fft08070
c     ************************************************************      fft08080
c     *    the number of disk accesses (reads+writes) used by  *      fft08090
c     *    this program is m*(4*log2(m/l)+3).                  *      fft08100
c     ************************************************************      fft08110
c     ------------------------------------------------------------      fft08120
c     check arguments                                                   fft08130
      if(l*(m/l).ne.m)stop "l not a factor of m in fftmg"               fft08140
      ml=m/l                                                            fft08150
      mt=1                                                              fft08160
   10 if(mt-ml)20,40,30                                                 fft08170
   20 mt=2*mt                                                           fft08180
      go to 10                                                          fft08190
   30 stop "m/l not a power of 2 in fftmg"                              fft08200
   40 continue                                                          fft08210
c     ------------------------------------------------------------      fft08220
c     ------------------------------------------------------------      fft08230
c     start processing                                                  fft08240
c     skip decimation if l .eq. m                                       fft08250
      if(l.eq.m)go to 80                                                fft08260
c     set starting input and output units                               fft08270
      input=unita                                                       fft08280
      output=unitb                                                      fft08290
c     set starting blocksize for decimation in input domain             fft08300
      iblock=m                                                          fft08310
```

```
c     ------------------------------------------------------------      fft08320
c     decimate this blocksize                                           fft08330
c     set up sub-block size                                             fft08340
   50 isub=iblock/2                                                     fft08350
c     set up for random i/O                                             fft08360
      call inset(input)                                                 fft08370
      call outset(output)                                               fft08380
c     for each block of this size                                       fft08390
      do 60 iblst=1,m,iblock                                            fft08400
c     copy even members into first sub-block, odd members into second   fft08410
      do 60 i=1,isub                                                    fft08420
      call randin(n,f,2*(i-1)+iblst)                                    fft08430
      call randou(n,f,i-1+iblst)                                        fft08440
      call randin(n,f,2*i-1+iblst)                                      fft08450
   60 call randou(n,f,i-1+iblst+isub)                                   fft08460
c     swap input/output functions                                       fft08470
      inputt=input                                                      fft08480
      input=output                                                      fft08490
      output=inputt                                                     fft08500
c     next smaller blocksize                                            fft08510
      iblock=isub                                                       fft08520
c     loop if not done                                                  fft08530
      if(iblock.gt.1)go to 50                                           fft08540
c     decimation complete                                               fft08550
c     ------------------------------------------------------------      fft08560
c     ------------------------------------------------------------      fft08570
c     ensure that data set is back on unita                             fft08580
      if(input.eq.unita)go to 80                                        fft08590
      call inset(input)                                                 fft08600
      call outset(output)                                               fft08610
      do 70 i=1,m                                                       fft08620
      call randin(n,f,i)                                                fft08630
   70 call randou(n,f,i)                                                fft08640
   80 continue                                                          fft08650
c     ------------------------------------------------------------      fft08660
c     start transform.                                                  fft08670
c     set block size                                                    fft08680
      iblock=1                                                          fft08690
c     set up for random i/o                                             fft08700
      call inset(unita)                                                 fft08710
      call outset(unita)                                                fft08720
c     run through the blocks                                            fft08730
      do 100 istart=1,m,iblock                                          fft08740
      into=1                                                            fft08750
c     run through the records in a block                                fft08760
      do 90 irecrd=istart,istart+iblock-1                               fft08770
c     read the record                                                   fft08780
      call randin(n,f(1,into),irecrd)                                   fft08790
c     proceed with next record in block                                 fft08800
   90 into=into+1                                                       fft08810
c     fourier transform in both directions                              fft08820
      ndim(1)=n                                                         fft08830
      ndim(2)=1                                                         fft08840
      call sfourt(f,ndim,2,isign,1,work,nfourt,nwork)                   fft08850
c     ************************************************************      fft08860
c     write the block back onto disk                                    fft08870
      ifrom=1                                                           fft08880
      do 100 irecrd=istart,istart+iblock-1                              fft08890
      call randou(n,f(1,ifrom),irecrd)                                  fft08900
  100 ifrom=ifrom+1                                                     fft08910
```

```
c     if l.eq.m, the transform is now complete.                          fft08920
      if(l.eq.m)return                                                   fft08930
c     done with first-step processing                                    fft08940
c     ------------------------------------------------------------       fft08950
c     finish the processing by pairs of rows, applying twiddle factors   fft08960
c     set the twiddle generator                                          fft08970
      twidlf=cexp(cmplx(0.,3.141593*isign/(iblock)))                     fft08980
c     run through the blocks by pairs                                    fft08990
  110 do 140 istart=1,m,iblock*2                                         fft09000
c     start the twiddle factor                                           fft09010
      twidle=(1.,0.)                                                     fft09020
c     scan the block pair                                                fft09030
      do 130 irecrd=istart,istart+iblock-1                               fft09040
      jrecrd=irecrd+iblock                                               fft09050
      call randin(n,f,irecrd)                                            fft09060
      call randin(n,f(1,2),jrecrd)                                       fft09070
c     combine, using twiddle factor                                      fft09080
      do 120 i=1,n                                                       fft09090
      w=twidle*f(i,2)                                                    fft09100
      f(i,2)=f(i,1)-w                                                    fft09110
  120 f(i,1)=f(i,1)+w                                                    fft09120
c     write back onto disk                                               fft09130
      call randou(n,f,irecrd)                                            fft09140
      call randou(n,f(1,2),jrecrd)                                       fft09150
c     increment the twiddle factor                                       fft09160
  130 twidle=twidle*twidlf                                               fft09170
c     go on to next block pair                                           fft09180
  140 continue                                                           fft09190
c     increase the block size, change twiddle increment                  fft09200
      iblock=2*iblock                                                    fft09210
      twidlf=csqrt(twidlf)                                               fft09220
      if(isign*aimag(twidlf).lt.0)twidlf=-twidlf                         fft09230
c     check if done                                                      fft09240
      if(iblock.lt.m)go to 110                                           fft09250
c     ------------------------------------------------------------       fft09260
c     done processing                                                    fft09270
      end                                                                fft09280


c*************************************************************************fft09290
      subroutine sdummy                                                  fft09300
c     random access i/o routine                                          fft09310
c*************************************************************************fft09320
      integer out                                                        fft09330
      complex f(n),g(n)                                                  fft09340
c     entry to receive input setup                                       fft09350
      entry inset(iin)                                                   fft09360
      in=iin                                                             fft09370
      return                                                             fft09380
c     entry to receive output setup                                      fft09390
      entry outset(iout)                                                 fft09400
      out=iout                                                           fft09410
      return                                                             fft09420
c     entry to do input                                                  fft09430
      entry randin(n,f,nrec)                                             fft09440
      read(in'nrec)f                                                     fft09450
      return                                                             fft09460
c     entry to do output                                                 fft09470
      entry randou(n,g,nrec)                                             fft09480
      write(out'nrec)g                                                   fft09490
```

```
      return                                                            fft09500
      end                                                               fft09510

c*****************************************************************************fft09520
      subroutine sfourt(data,nn,ndim,isign,iform,work,nfourt,nwork)     fft09530
c  note that "nfourt" and "nwork" in subroutine call statement are      fft09540
c  revised features of "sfourt". they are used as dynamic dimensioning  fft09550
c  for the "data" and "work" arrays                                     fft09560
c     the cooley-tukey fast fourier transform in usasi basic fortran    fft09570
c     transform(j1,j2,...) = sum(data(i1,i2,...)*w1**((i1-1)*(j1-1))    fft09580
c                               *w2**((i2-1)*(j2-1))*...),              fft09590
c     where i1 and j1 run from 1 to nn(1) and w1=exp(isign*2*pi*        fft09600
c     sqrt(-1)/nn(1)), etc.  there is no limit on the dimensionality    fft09610
c     (number of subscripts) of the data array.  if an inverse          fft09620
c     transform (isign=+1) is performed upon an array of transformed    fft09630
c     (isign=-1) data, the original data will reappear.                 fft09640
c     multiplied by nn(1)*nn(2)*...  the array of input data must be     fft09650
c     in complex format.  however, if all imaginary parts are zero (i.e.fft09660
c     the data are disguised real) running time is cut up to forty per- fft09670
c     cent.  (for fastest transform of real data, nn(1) should be even.)fft09680
c     the transform values are always complex and are returned in the   fft09690
c     original array of data, replacing the input data.  the length     fft09700
c     of each dimension of the data array may be any integer.  the      fft09710
c     program runs faster on composite integers than on primes, and is  fft09720
c     particularly fast on numbers rich in factors of two.              fft09730
c     timing is in fact given by the following formula.  let ntot be thefft09740
c     total number of points (real or complex) in the data array, that  fft09750
c     is, ntot=nn(1)*nn(2)*...  decompose ntot into its prime factors,  fft09760
c     such as 2**k2 * 3**k3 * 5**k5 * ...  let sum2 be the sum of all   fft09770
c     the factors of two in ntot, that is, sum2 = 2*k2.  let sumf be    fft09780
c     the sum of all other factors of ntot, that is, sumf = 3*k3*5*k5*..fft09790
c     the time taken by a multidimensional transform on these ntot data fft09800
c     is t = t0 + ntot*(t1+t2*sum2+t3*sumf).  on the cdc 3300 (floating fft09810
c     point add time = six microseconds), t = 3000 + ntot*(600+40*sum2+ fft09820
c     175*sumf) microseconds on complex data.                           fft09830
c     implementation of the definition by summation will run in a time  fft09840
c     proportional to ntot*(nn(1)+nn(2)+...).  for highly composite ntotfft09850
c     the savings offered by this program can be dramatic.  a one-dimen-fft09860
c     sional array 4000 in length will be transformed in 4000*(600+     fft09870
c     40*(2+2+2+2+2)+175*(5+5+5)) = 14.5 seconds versus about 4000*     fft09880
c     4000*175 = 2800 seconds for the straightforward technique.        fft09890
c     the fast fourier transform places three restrictions upon the     fft09900
c     data.                                                             fft09910
c     1.  the number of input data and the number of transform values   fft09920
c     must be the same.                                                 fft09930
c     2.  both the input data and the transform values must represent   fft09940
c     equispaced points in their respective domains of time and         fft09950
c     frequency.  calling these spacings deltat and deltaf, it must be  fft09960
c     true that deltaf=2*pi/(nn(i)*deltat).  of course, deltat need not fft09970
c     be the same for every dimension.                                  fft09980
c     3.  conceptually at least, the input data and the transform outputfft09990
c     represent single cycles of periodic functions.                    fft10000
c     the calling sequence is--                                         fft10010
c     call fourt(data,nn,ndim,isign,iform,work)                         fft10020
c     data is the array used to hold the real and imaginary parts       fft10030
c     of the data on input and the transform values on output.  it      fft10040
c     is a multidimensional floating point array, with the real and     fft10050
c     imaginary parts of a datum stored immediately adjacent in storage fft10060
c     (such as fortran iv places them).  normal fortran ordering is     fft10070
c     expected, the first subscript changing fastest.  the dimensions   fft10080
```

```
c     are given in the integer array nn, of length ndim.  isign is -1   fft10090
c     to indicate a forward transform (exponential sign is -) and +1    fft10100
c     for an inverse transform (sign is +).  iform is +1 if the data arefft10110
c     complex, 0 if the data are real.  if it is 0, the imaginary       fft10120
c     parts of the data must be set to zero.  as explained above, the   fft10130
c     transform values are always complex and are stored in array data. fft10140
c     work is an array used for working storage.  it is floating point  fft10150
c     real, one dimensional of length equal to twice the largest array  fft10160
c     dimension nn(i) that is not a power of two.  if all nn(i) are     fft10170
c     powers of two, it is not needed and may be replaced by zero in thefft10180
c     calling sequence.  thus, for a one-dimensional array, nn(1) odd,  fft10190
c     work occupies as many storage locations as data.  if supplied,    fft10200
c     work must not be the same array as data.  all subscripts of all   fft10210
c     arrays begin at one.                                              fft10220
c     example 1.  three-dimensional forward fourier transform of a      fft10230
c     complex array dimensioned 32 by 25 by 13 in fortran iv.           fft10240
c     dimension data(32,25,13),work(50),nn(3)                           fft10250
c     complex data                                                      fft10260
c     data nn/32,25,13/                                                 fft10270
c     do 1 i=1,32                                                       fft10280
c     do 1 j=1,25                                                       fft10290
c     do 1 k=1,13                                                       fft10300
c  1  data(i,j,k)=complex value                                         fft10310
c     call fourt(data,nn,3,-1,1,work)                                   fft10320
c     example 2.  one-dimensional forward transform of a real array of  fft10330
c     length 64 in fortran ii,                                          fft10340
c     dimension data(2,64)                                              fft10350
c     do 2 i=1,64                                                       fft10360
c     data(1,i)=real part                                               fft10370
c  2  data(2,i)=0.                                                      fft10380
c     call fourt(data,64,1,-1,0,0)                                      fft10390
c     there are no error messages or error halts in this program.  the  fft10400
c     program returns immediately if ndim or any nn(i) is less than one.fft10410
c     program by norman brenner from the basic program by charles       fft10420
c     rader,  june 1967.  the idea for the digit reversal was           fft10430
c     suggested by ralph alter.                                         fft10440
c     this is the fastest and most versatile version of the fft known   fft10450
c     to the author.  a program called four2 is available that also     fft10460
c     performs the fast fourier transform and is written in usasi basic fft10470
c     fortran.  it is about one third as long and restricts the         fft10480
c     dimensions of the input array (which must be complex) to be powersfft10490
c     of two.  another program, called four1, is one tenth as long and  fft10500
c     runs two thirds as fast on a one-dimensional complex array whose  fft10510
c     length is a power of two.                                         fft10520
c     reference--                                                       fft10530
c     ieee audio transactions (june 1967), special issue on the fft.    fft10540
c*****************************************************************************fft10550
      dimension data(nfourt),nn(2),ifact(32),work(nwork)                fft10560
      data twopi/6.2831853071796/,rthlf/0.70710678118655/               fft10570
      if(ndim-1)1190,10,10                                              fft10580
   10 ntot=2                                                            fft10590
      do 20 idim=1,ndim                                                 fft10600
      if(nn(idim))1190,1190,20                                          fft10610
   20 ntot=ntot*nn(idim)                                                fft10620
c     main loop for each dimension                                      fft10630
      np1=2                                                             fft10640
      do 1180 idim=1,ndim                                               fft10650
      n=nn(idim)                                                        fft10660
      np2=np1*n                                                         fft10670
      if(n-1)1190,1170,30                                               fft10680
```

```
c     is n a power of two and if not, what are its factors                fft10690
   30 m=n                                                                 fft10700
      ntwo=np1                                                            fft10710
      if=1                                                                fft10720
      idiv=2                                                              fft10730
   40 iquot=m/idiv                                                        fft10740
      irem=m-idiv*iquot                                                   fft10750
      if(iquot-idiv)120,50,50                                             fft10760
   50 if(irem)70,60,70                                                    fft10770
   60 ntwo=ntwo+ntwo                                                      fft10780
      ifact(if)=idiv                                                      fft10790
      if=if+1                                                             fft10800
      m=iquot                                                             fft10810
      go to 40                                                            fft10820
   70 idiv=3                                                              fft10830
      inon2=if                                                            fft10840
   80 iquot=m/idiv                                                        fft10850
      irem=m-idiv*iquot                                                   fft10860
      if(iquot-idiv)140,90,90                                             fft10870
   90 if(irem)110,100,110                                                 fft10880
  100 ifact(if)=idiv                                                      fft10890
      if=if+1                                                             fft10900
      m=iquot                                                             fft10910
      go to 80                                                            fft10920
  110 idiv=idiv+2                                                         fft10930
      go to 80                                                            fft10940
  120 inon2=if                                                            fft10950
      if(irem)140,130,140                                                 fft10960
  130 ntwo=ntwo+ntwo                                                      fft10970
      go to 150                                                           fft10980
  140 ifact(if)=m                                                         fft10990
c     separate four cases--                                               fft11000
c        1. complex transform or real transform for the 4th, 9th,etc.     fft11010
c           dimensions.                                                  fft11020
c        2. real transform for the 2nd or 3rd dimension.  method--       fft11030
c           transform half the data, supplying the other half by con-    fft11040
c           jugate symmetry.                                             fft11050
c        3. real transform for the 1st dimension, n odd.  method--       fft11060
c           set the imaginary parts to zero.                             fft11070
c        4. real transform for the 1st dimension, n even.  method--      fft11080
c           transform a complex array of length n/2 whose real parts     fft11090
c           are the even numbered real values and whose imaginary parts  fft11100
c           are the odd numbered real values.  separate and supply       fft11110
c           the second half by conjugate symmetry.                       fft11120
  150 icase=1                                                             fft11130
      ifmin=1                                                             fft11140
      i1rng=np1                                                           fft11150
      if(idim-4)160,210,210                                               fft11160
  160 if(iform)170,170,210                                                fft11170
  170 icase=2                                                             fft11180
      i1rng=np0*(1+nprev/2)                                               fft11190
      if(idim-1)180,180,210                                               fft11200
  180 icase=3                                                             fft11210
      i1rng=np1                                                           fft11220
      if(ntwo-np1)210,210,190                                             fft11230
  190 icase=4                                                             fft11240
      ifmin=2                                                             fft11250
      ntwo=ntwo/2                                                         fft11260
      n=n/2                                                               fft11270
      np2=np2/2                                                           fft11280
```

```
      ntot=ntot/2                                                       fft11290
      i=1                                                               fft11300
      do 200 j=1,ntot                                                   fft11310
      data(j)=data(i)                                                   fft11320
  200 i=i+2                                                             fft11330
c     shuffle data by bit reversal, since n=2**k.  as the shuffling     fft11340
c     can be done by simple interchange, no working array is needed     fft11350
  210 if(ntwo-np2)290,220,220                                           fft11360
  220 np2hf=np2/2                                                       fft11370
      j=1                                                               fft11380
      do 280 i2=1,np2,np1                                               fft11390
      if(j-i2)230,250,250                                               fft11400
  230 i1max=i2+np1-2                                                    fft11410
      do 240 i1=i2,i1max,2                                              fft11420
      do 240 i3=i1,ntot,np2                                             fft11430
      j3=j+i3-i2                                                        fft11440
      tempr=data(i3)                                                    fft11450
      tempi=data(i3+1)                                                  fft11460
      data(i3)=data(j3)                                                 fft11470
      data(i3+1)=data(j3+1)                                             fft11480
      data(j3)=tempr                                                    fft11490
  240 data(j3+1)=tempi                                                  fft11500
  250 m=np2hf                                                           fft11510
  260 if(j-m)280,280,270                                                fft11520
  270 j=j-m                                                             fft11530
      m=m/2                                                             fft11540
      if(m-np1)280,260,260                                              fft11550
  280 j=j+m                                                             fft11560
      go to 370                                                         fft11570
c     shuffle data by digit reversal for general n                      fft11580
  290 nwork=2*n                                                         fft11590
      do 360 i1=1,np1,2                                                 fft11600
      do 360 i3=i1,ntot,np2                                             fft11610
      j=i3                                                              fft11620
      do 350 i=1,nwork,2                                                fft11630
      if(icase-3)300,310,300                                            fft11640
  300 work(i)=data(j)                                                   fft11650
      work(i+1)=data(j+1)                                               fft11660
      go to 320                                                         fft11670
  310 work(i)=data(j)                                                   fft11680
      work(i+1)=0.                                                      fft11690
  320 ifp2=np2                                                          fft11700
      if=ifmin                                                          fft11710
  330 ifp1=ifp2/ifact(if)                                               fft11720
      j=j+ifp1                                                          fft11730
      if(j-i3-ifp2)350,340,340                                          fft11740
  340 j=j-ifp2                                                          fft11750
      ifp2=ifp1                                                         fft11760
      if=if+1                                                           fft11770
      if(ifp2-np1)350,350,330                                           fft11780
  350 continue                                                          fft11790
      i2max=i3+np2-np1                                                  fft11800
      i=1                                                               fft11810
      do 360 i2=i3,i2max,np1                                            fft11820
      data(i2)=work(i)                                                  fft11830
      data(i2+1)=work(i+1)                                              fft11840
  360 i=i+2                                                             fft11850
c     main loop for factors of two.  perform fourier transforms of      fft11860
c     length four, with one of length two if needed.  the twiddle factorfft11870
c     w=exp(isign*2*pi*sqrt(-1)*m/(4*mmax)).  check for w=isign*sqrt(-1)fft11880
```

```
c       and repeat for w=w*(1+isign*sqrt(-1))/sqrt(2).                  fft11890
  370   if(ntwo-np1)680,680,380                                        fft11900
  380   np1tw=np1+np1                                                  fft11910
        ipar=ntwo/np1                                                  fft11920
  390   if(ipar-2)430,410,400                                          fft11930
  400   ipar=ipar/4                                                    fft11940
        go to 390                                                      fft11950
  410   do 420 i1=1,i1rng,2                                            fft11960
        do 420 k1=i1,ntot,np1tw                                        fft11970
        k2=k1+np1                                                      fft11980
        tempr=data(k2)                                                 fft11990
        tempi=data(k2+1)                                               fft12000
        data(k2)=data(k1)-tempr                                        fft12010
        data(k2+1)=data(k1+1)-tempi                                    fft12020
        data(k1)=data(k1)+tempr                                        fft12030
  420   data(k1+1)=data(k1+1)+tempi                                    fft12040
  430   mmax=np1                                                       fft12050
  440   if(mmax-ntwo/2)450,680,680                                     fft12060
  450   lmax=max0(np1tw,mmax/2)                                        fft12070
        do 670 l=np1,lmax,np1tw                                        fft12080
        m=l                                                            fft12090
        if(mmax-np1)500,500,460                                        fft12100
  460   theta=-twopi*float(l)/float(4*mmax)                            fft12110
        if(isign)480,470,470                                           fft12120
  470   theta=-theta                                                   fft12130
  480   wr=cos(theta)                                                  fft12140
        wi=sin(theta)                                                  fft12150
  490   w2r=wr*wr-wi*wi                                                fft12160
        w2i=2.*wr*wi                                                   fft12170
        w3r=w2r*wr-w2i*wi                                              fft12180
        w3i=w2r*wi+w2i*wr                                              fft12190
  500   do 630 i1=1,i1rng,2                                            fft12200
        kmin=i1+ipar*m                                                 fft12210
        if(mmax-np1)510,510,520                                        fft12220
  510   kmin=i1                                                        fft12230
  520   kdif=ipar*mmax                                                 fft12240
  530   kstep=4*kdif                                                   fft12250
        if(kstep-ntwo)540,540,630                                      fft12260
  540   do 620 k1=kmin,ntot,kstep                                      fft12270
        k2=k1+kdif                                                     fft12280
        k3=k2+kdif                                                     fft12290
        k4=k3+kdif                                                     fft12300
        if(mmax-np1)550,550,580                                        fft12310
  550   u1r=data(k1)+data(k2)                                          fft12320
        u1i=data(k1+1)+data(k2+1)                                      fft12330
        u2r=data(k3)+data(k4)                                          fft12340
        u2i=data(k3+1)+data(k4+1)                                      fft12350
        u3r=data(k1)-data(k2)                                          fft12360
        u3i=data(k1+1)-data(k2+1)                                      fft12370
        if(isign)560,570,570                                           fft12380
  560   u4r=data(k3+1)-data(k4+1)                                      fft12390
        u4i=data(k4)-data(k3)                                          fft12400
        go to 610                                                      fft12410
  570   u4r=data(k4+1)-data(k3+1)                                      fft12420
        u4i=data(k3)-data(k4)                                          fft12430
        go to 610                                                      fft12440
  580   t2r=w2r*data(k2)-w2i*data(k2+1)                                fft12450
        t2i=w2r*data(k2+1)+w2i*data(k2)                                fft12460
        t3r=wr*data(k3)-wi*data(k3+1)                                  fft12470
        t3i=wr*data(k3+1)+wi*data(k3)                                  fft12480
```

```
      t4r=w3r*data(k4)-w3i*data(k4+1)                                    fft12490
      t4i=w3r*data(k4+1)+w3i*data(k4)                                    fft12500
      u1r=data(k1)+t2r                                                   fft12510
      u1i=data(k1+1)+t2i                                                 fft12520
      u2r=t3r+t4r                                                        fft12530
      u2i=t3i+t4i                                                        fft12540
      u3r=data(k1)-t2r                                                   fft12550
      u3i=data(k1+1)-t2i                                                 fft12560
      if(isign)590,600,600                                               fft12570
  590 u4r=t3i-t4i                                                        fft12580
      u4i=t4r-t3r                                                        fft12590
      go to 610                                                          fft12600
  600 u4r=t4i-t3i                                                        fft12610
      u4i=t3r-t4r                                                        fft12620
  610 data(k1)=u1r+u2r                                                   fft12630
      data(k1+1)=u1i+u2i                                                 fft12640
      data(k2)=u3r+u4r                                                   fft12650
      data(k2+1)=u3i+u4i                                                 fft12660
      data(k3)=u1r-u2r                                                   fft12670
      data(k3+1)=u1i-u2i                                                 fft12680
      data(k4)=u3r-u4r                                                   fft12690
  620 data(k4+1)=u3i-u4i                                                 fft12700
      kdif=kstep                                                         fft12710
      kmin=4*(kmin-i1)+i1                                                fft12720
      go to 530                                                          fft12730
  630 continue                                                           fft12740
      m=m+lmax                                                           fft12750
      if(m-mmax)640,640,670                                              fft12760
  640 if(isign)650,660,660                                               fft12770
  650 tempr=wr                                                           fft12780
      wr=(wr+wi)*rthlf                                                   fft12790
      wi=(wi-tempr)*rthlf                                                fft12800
      go to 490                                                          fft12810
  660 tempr=wr                                                           fft12820
      wr=(wr-wi)*rthlf                                                   fft12830
      wi=(tempr+wi)*rthlf                                                fft12840
      go to 490                                                          fft12850
  670 continue                                                           fft12860
      ipar=3-ipar                                                        fft12870
      mmax=mmax+mmax                                                     fft12880
      go to 440                                                          fft12890
c     main loop for factors not equal to two.  apply the twiddle factor  fft12900
c     w=exp(isign*2*pi*sqrt(-1)*(j1-1)*(j2-j1)/(ifp1+ifp2)), then        fft12910
c     perform a fourier transform of length ifact(if), making use of    fft12920
c     conjugate symmetries.                                              fft12930
  680 if(ntwo-np2)690,900,900                                            fft12940
  690 ifp1=ntwo                                                          fft12950
      if=inon2                                                           fft12960
      np1hf=np1/2                                                        fft12970
  700 ifp2=ifact(if)*ifp1                                                fft12980
      j1min=np1+1                                                        fft12990
      if(j1min-ifp1)710,710,760                                          fft13000
  710 do 750 j1=j1min,ifp1,np1                                           fft13010
      theta=-twopi*float(j1-1)/float(ifp2)                               fft13020
      if(isign)730,720,720                                               fft13030
  720 theta=-theta                                                       fft13040
  730 wstpr=cos(theta)                                                   fft13050
      wstpi=sin(theta)                                                   fft13060
      wr=wstpr                                                           fft13070
      wi=wstpi                                                           fft13080
```

```
      j2min=j1+ifp1                                                     fft13090
      j2max=j1+ifp2-ifp1                                                fft13100
      do 750 j2=j2min,j2max,ifp1                                        fft13110
      i1max=j2+i1rng-2                                                  fft13120
      do 740 i1=j2,i1max,2                                              fft13130
      do 740 j3=i1,ntot,ifp2                                            fft13140
      tempr=data(j3)                                                    fft13150
      data(j3)=data(j3)*wr-data(j3+1)*wi                                fft13160
  740 data(j3+1)=tempr*wi+data(j3+1)*wr                                 fft13170
      tempr=wr                                                          fft13180
      wr=wr*wstpr-wi*wstpi                                              fft13190
  750 wi=tempr*wstpi+wi*wstpr                                           fft13200
  760 theta=-twopi/float(ifact(if))                                     fft13210
      if(isign)780,770,770                                              fft13220
  770 theta=-theta                                                      fft13230
  780 wstpr=cos(theta)                                                  fft13240
      wstpi=sin(theta)                                                  fft13250
      j2rng=ifp1*(1+ifact(if)/2)                                        fft13260
      do 890 i1=1,i1rng,2                                               fft13270
      do 890 i3=i1,ntot,np2                                             fft13280
      j2max=i3+j2rng-ifp1                                               fft13290
      do 880 j2=i3,j2max,ifp1                                           fft13300
      j1max=j2+ifp1-np1                                                 fft13310
      do 850 j1=j2,j1max,np1                                            fft13320
      j3max=j1+np2-ifp2                                                 fft13330
      do 850 j3=j1,j3max,ifp2                                           fft13340
      jmin=j3-j2+i3                                                     fft13350
      jmax=jmin+ifp2-ifp1                                               fft13360
      i=1+(j3-i3)/np1hf                                                 fft13370
      if(j2-i3)790,790,820                                              fft13380
  790 sumr=0.                                                           fft13390
      sumi=0.                                                           fft13400
      do 810 j=jmin,jmax,ifp1                                           fft13410
  800 sumr=sumr+data(j)                                                 fft13420
  810 sumi=sumi+data(j+1)                                               fft13430
      work(i)=sumr                                                      fft13440
      work(i+1)=sumi                                                    fft13450
      go to 850                                                         fft13460
  820 iconj=1+(ifp2-2*j2+i3+j3)/np1hf                                   fft13470
      j=jmax                                                            fft13480
      sumr=data(j)                                                      fft13490
      sumi=data(j+1)                                                    fft13500
      oldsr=0.                                                          fft13510
      oldsi=0.                                                          fft13520
      j=j-ifp1                                                          fft13530
  830 tempr=sumr                                                        fft13540
      tempi=sumi                                                        fft13550
      sumr=twowr*sumr-oldsr+data(j)                                     fft13560
      sumi=twowr*sumi-oldsi+data(j+1)                                   fft13570
      oldsr=tempr                                                       fft13580
      oldsi=tempi                                                       fft13590
      j=j-ifp1                                                          fft13600
      if(j-jmin)840,840,830                                             fft13610
  840 tempr=wr*sumr-oldsr+data(j)                                       fft13620
      tempi=wi*sumi                                                     fft13630
      work(i)=tempr-tempi                                               fft13640
      work(iconj)=tempr+tempi                                           fft13650
      tempr=wr*sumi-oldsi+data(j+1)                                     fft13660
      tempi=wi*sumr                                                     fft13670
      work(i+1)=tempr+tempi                                             fft13680
```

```
      work(iconj+1)=tempr-tempi                                          fft13690
  850 continue                                                           fft13700
      if(j2-i3)860,860,870                                               fft13710
  860 wr=wstpr                                                           fft13720
      wi=wstpi                                                           fft13730
      go to 880                                                          fft13740
  870 tempr=wr                                                           fft13750
      wr=wr*wstpr-wi*wstpi                                               fft13760
      wi=tempr*wstpi+wi*wstpr                                            fft13770
  880 twowr=wr+wr                                                        fft13780
      i=1                                                                fft13790
      i2max=i3+np2-np1                                                   fft13800
      do 890 i2=i3,i2max,np1                                             fft13810
      data(i2)=work(i)                                                   fft13820
      data(i2+1)=work(i+1)                                               fft13830
  890 i=i+2                                                              fft13840
      if=if+1                                                            fft13850
      ifp1=ifp2                                                          fft13860
      if(ifp1-np2)700,900,900                                            fft13870
c     complete a real transform in the 1st dimension, n even, by con-    fft13880
c     jugate symmetries.                                                 fft13890
  900 go to (1170,1090,1170,910),icase                                   fft13900
  910 nhalf=n                                                            fft13910
      n=n+n                                                              fft13920
      theta=-twopi/float(n)                                              fft13930
      if(isign)930,920,920                                               fft13940
  920 theta=-theta                                                       fft13950
  930 wstpr=cos(theta)                                                   fft13960
      wstpi=sin(theta)                                                   fft13970
      wr=wstpr                                                           fft13980
      wi=wstpi                                                           fft13990
      imin=3                                                             fft14000
      jmin=2*nhalf-1                                                     fft14010
      go to 960                                                          fft14020
  940 j=jmin                                                             fft14030
      do 950 i=imin,ntot,np2                                             fft14040
      sumr=(data(i)+data(j))/2.                                          fft14050
      sumi=(data(i+1)+data(j+1))/2.                                      fft14060
      difr=(data(i)-data(j))/2.                                          fft14070
      difi=(data(i+1)-data(j+1))/2.                                      fft14080
      tempr=wr*sumi+wi*difr                                              fft14090
      tempi=wi*sumi-wr*difr                                              fft14100
      data(i)=sumr+tempr                                                 fft14110
      data(i+1)=difi+tempi                                               fft14120
      data(j)=sumr-tempr                                                 fft14130
      data(j+1)=-difi+tempi                                              fft14140
  950 j=j+np2                                                            fft14150
      imin=imin+2                                                        fft14160
      jmin=jmin-2                                                        fft14170
      tempr=wr                                                           fft14180
      wr=wr*wstpr-wi*wstpi                                               fft14190
      wi=tempr*wstpi+wi*wstpr                                            fft14200
  960 if(imin-jmin)940,970,1000                                          fft14210
  970 if(isign)980,1000,1000                                             fft14220
  980 do 990 i=imin,ntot,np2                                             fft14230
  990 data(i+1)=-data(i+1)                                               fft14240
 1000 np2=np2+np2                                                        fft14250
      ntot=ntot+ntot                                                     fft14260
      j=ntot+1                                                           fft14270
      imax=ntot/2+1                                                      fft14280
```

```
 1010 imin=imax-2*nnalf                                                 fft14290
      i=imin                                                            fft14300
      go to 1030                                                        fft14310
 1020 data(j)=data(i)                                                   fft14320
      data(j+1)=-data(i+1)                                              fft14330
 1030 i=i+2                                                             fft14340
      j=j-2                                                             fft14350
      if(i-imax)1020,1040,1040                                          fft14360
 1040 data(j)=data(imin)-data(imin+1)                                   fft14370
      data(j+1)=0.                                                      fft14380
      if(i-j)1060,1030,1080                                             fft14390
 1050 data(j)=data(i)                                                   fft14400
      data(j+1)=data(i+1)                                               fft14410
 1060 i=i-2                                                             fft14420
      j=j-2                                                             fft14430
      if(i-imin)1070,1070,1050                                          fft14440
 1070 data(j)=data(imin)+data(imin+1)                                   fft14450
      data(j+1)=0.                                                      fft14460
      imax=imin                                                         fft14470
      go to 1010                                                        fft14480
 1080 data(1)=data(1)+data(2)                                           fft14490
      data(2)=0.                                                        fft14500
      go to 1170                                                        fft14510
c     complete a real transform for the 2nd or 3rd dimension by        fft14520
c     conjugate symmetries.                                            fft14530
 1090 if(i1rng-np1)1100,1170,1170                                       fft14540
 1100 do 1160 i3=1,ntot,np2                                             fft14550
      i2max=i3+np2-np1                                                  fft14560
      do 1160 i2=i3,i2max,np1                                           fft14570
      imin=i2+i1rng                                                     fft14580
      imax=i2+np1-2                                                     fft14590
      jmax=2*i3+np1-imin                                                fft14600
      if(i2-i3)1120,1120,1110                                           fft14610
 1110 jmax=jmax+np2                                                     fft14620
 1120 if(idim-2)1150,1150,1130                                          fft14630
 1130 j=jmax+np0                                                        fft14640
      do 1140 i=imin,imax,2                                             fft14650
      data(i)=data(j)                                                   fft14660
      data(i+1)=-data(j+1)                                              fft14670
 1140 j=j-2                                                             fft14680
 1150 j=jmax                                                            fft14690
      do 1160 i=imin,imax,np0                                           fft14700
      data(i)=data(j)                                                   fft14710
      data(i+1)=-data(j+1)                                              fft14720
 1160 j=j-np0                                                           fft14730
c     end of loop on each dimension                                     fft14740
 1170 np0=np1                                                           fft14750
      np1=np2                                                           fft14760
 1180 nprev=n                                                           fft14770
 1190 return                                                            fft14780
      end                                                               fft14790


c*************************************************************************fft14800
      subroutine spspol(itype,x,y,xysq,ar,ai)                           fft14810
c       subroutine "spspol" operates on input fourier coefficients to   fft14820
c       obtain the magnetic field reduced to the pole or to make a      fft14830
c       pseudo-gravity (or pseudo-magnetic) transformation.             fft14840
c*************************************************************************fft14850
      common/basic/c1,el,em,en,bl,bm,bn                                 fft14860
```

```
      xy=sqrt(xysq)                                                    fft14870
      bml=x*bl+y*bm                                                    fft14880
      eml=x*el+y*em                                                    fft14890
      ber=bn*en*xy-bml*eml/xy                                          fft14900
      bei=(en*bml+bn*eml)                                              fft14910
      if(itype-2)10,20,40                                              fft14920
   10 c=c1                                                             fft14930
      go to 30                                                         fft14940
   20 c=xy                                                             fft14950
   30 art=ar                                                           fft14960
      ait=ai                                                           fft14970
      bes=c/(ber*ber+bei*bei)                                          fft14980
      ar=(ber*art+bei*ait)*bes                                         fft14990
      ai=(ber*ait-bei*art)*bes                                         fft15000
      go to 50                                                         fft15010
   40 c=1./c1                                                          fft15020
      art=ar                                                           fft15030
      ait=ai                                                           fft15040
      ar=(ber*art-bei*ait)*c                                           fft15050
      ai=(bei*art+ber*ait)*c                                           fft15060
   50 return                                                           fft15070
      end                                                              fft15080

c*********************************************************************fft15090
      subroutine scont(c2,xysq,ar,ai)                                  fft15100
c  subroutine "cont" upward or downward continues input fourier       fft15110
c  coefficients.                                                      fft15120
c*********************************************************************fft15130
      xy=sqrt(xysq)                                                    fft15140
      e=exp(c2*xy)                                                     fft15150
      ar=ar*e                                                          fft15160
      ai=ai*e                                                          fft15170
      return                                                           fft15180
      end                                                              fft15190

c*********************************************************************fft15200
      subroutine sfvert(c4,xysq,ar,ai)                                 fft15210
c  subroutine "fvert" operates on input fourier coefficients          fft15220
c  to obtain the 1st vertical derivative of the input data.           fft15230
c*********************************************************************fft15240
      xsqrt=sqrt(xysq)                                                 fft15250
      xyp=xsqrt*c4                                                     fft15260
      ar=ar*xyp                                                        fft15270
      ai=ai*xyp                                                        fft15280
      return                                                           fft15290
      end                                                              fft15300

c*********************************************************************fft15310
      subroutine ssvert(c4,xysq,ar,ai)                                 fft15320
c  subroutine "svert" operates on input fourier coefficients          fft15330
c  to obtain the 2nd vertical derivative of the input data.           fft15340
c*********************************************************************fft15350
      xyp=xysq*c4                                                      fft15360
      ar=ar*xyp                                                        fft15370
      ai=ai*xyp                                                        fft15380
      return                                                           fft15390
      end                                                              fft15400
```

```
c*************************************************************************fft15410
      subroutine sbpass(a1,a2,f1,f2,f3,f4,btan12,btan34,xysq)             fft15420
c   subroutine "sbpass" eliminates wavelengths                            fft15430
c   lying outside the range of w1,w2,w3 & w4.                             fft15440
c*************************************************************************fft15450
      rad=sqrt(xysq)                                                      fft15460
      if(rad.lt.f1) go to 10                                              fft15470
      if(rad.gt.f4) go to 10                                              fft15480
      go to 20                                                            fft15490
   10 a1=0.0                                                              fft15500
      a2=0.0                                                              fft15510
      go to 40                                                            fft15520
   20 if(rad.ge.f2.and.rad.le.f3) go to 40                                fft15530
      if(rad.gt.f2) go to 30                                              fft15540
      f=(rad-f1)*btan12                                                   fft15550
      a1=f*a1                                                             fft15560
      a2=f*a2                                                             fft15570
      go to 40                                                            fft15580
   30 f=(f4-rad)*btan34                                                   fft15590
      a1=f*a1                                                             fft15600
      a2=f*a2                                                             fft15610
   40 return                                                              fft15620
      end                                                                 fft15630

c*************************************************************************fft15640
      subroutine strend(x,y,ar,ai)                                        fft15650
c  subroutine "trend" passes or rejects trends striking                   fft15660
c  between angles thet1 and thet2.                                        fft15670
c*************************************************************************fft15680
      common/tr/prad,p,q,th1p,th2p                                        fft15690
      if(x.eq.0.0) go to 50                                               fft15700
      if(y.eq.0.0) go to 60                                               fft15710
      thet=atan(x/y)*prad                                                 fft15720
      if(thet.lt.0.0)thet=180.+thet                                       fft15730
   10 if(th2p.gt.th1p) go to 40                                           fft15740
      if(thet.gt.th2p.and.thet.lt.th1p) go to 30                          fft15750
   20 ar=ar*q                                                             fft15760
      ai=ai*q                                                             fft15770
      return                                                              fft15780
   30 ar=ar*p                                                             fft15790
      ai=ai*p                                                             fft15800
      return                                                              fft15810
   40 if(thet.gt.th1p.and.thet.lt.th2p) go to 30                          fft15820
      go to 20                                                            fft15830
   50 thet=0.                                                             fft15840
      if(y.lt.0.0)thet=180.                                               fft15850
      go to 10                                                            fft15860
   60 thet=90.                                                            fft15870
      go to 10                                                            fft15880
      end                                                                 fft15890

c*************************************************************************fft15900
      subroutine srmean(nx,np,a1)                                         fft15910
c  subroutine "srmean" removes the mean from the input grid               fft15920
c  employing only boundary values.                                        fft15930
c*************************************************************************fft15940
```

```
      dimension a1(np,3),title(14)                                      fft15950
      character*50 fname,coname,fileo                                   fft15960
      real mean                                                         fft15970
      common/main/ia(14),pgm(2),nz,yo,xo,dx,dy,iw,kr,ny,nadd,ir         fft15980
      common/parm1/fname,coname,iopt1,iopt2,idval,icoef,ddx,ddy         fft15990
      data dval/o376777777777/                                          fft16000
      ko=10                                                             fft16010
      mean=0.0                                                          fft16020
      k=0                                                               fft16030
      if(idval.eq.0) go to 150                                          fft16040
c  idval>0: find and sum boundary values.                               fft16050
      call srdbin(kr,a1,ny)                                             fft16060
      do 10 j=1,ny                                                      fft16070
      if(a1(j,1).eq.dval) go to 10                                      fft16080
      mean=mean+a1(j,1)                                                 fft16090
      k=k+1                                                             fft16100
   10 continue                                                          fft16110
      call srdbin(kr,a1(1,2),ny)                                        fft16120
      do 130 i=3,nx                                                     fft16130
      call srdbin(kr,a1(1,3),ny)                                        fft16140
      iq=-1                                                             fft16150
      if(a1(1,2).eq.dval) go to 20                                      fft16160
      k=k+1                                                             fft16170
      mean=mean+a1(1,2)                                                 fft16180
      if(a1(1,1).lt.dval.and.a1(1,3).lt.dval)iq=0                       fft16190
   20 do 90 j=2,ny-1                                                    fft16200
      if(a1(j,2).eq.dval) go to 60                                      fft16210
      if(iq)30,40,50                                                    fft16220
   30 mean=mean+a1(j,2)                                                 fft16230
      k=k+1                                                             fft16240
      iq=0                                                              fft16250
      if(a1(j,1).eq.dval.or.a1(j,3).eq.dval)iq=-1                       fft16260
      go to 90                                                          fft16270
   40 iq=1                                                              fft16280
      if(a1(j,1).lt.dval.and.a1(j,3).lt.dval) go to 90                  fft16290
      iq=-1                                                             fft16300
      mean=mean+a1(j,2)                                                 fft16310
      k=k+1                                                             fft16320
      go to 90                                                          fft16330
   50 iq=1                                                              fft16340
      if(a1(j,1).lt.dval.and.a1(j,3).lt.dval) go to 90                  fft16350
      mean=mean+a1(j,2)+a1(j-1,2)                                       fft16360
      k=k+1                                                             fft16370
      iq=-1                                                             fft16380
      go to 90                                                          fft16390
   60 if(iq)80,80,70                                                    fft16400
   70 mean=mean+a1(j-1,2)                                               fft16410
      k=k+1                                                             fft16420
   80 iq=-1                                                             fft16430
   90 continue                                                          fft16440
      if(a1(ny,2).eq.dval) go to 100                                    fft16450
      k=k+1                                                             fft16460
      mean=mean+a1(ny,2)                                                fft16470
      if(a1(ny,1).lt.dval.and.a1(ny,3).lt.dval) go to 110               fft16480
  100 if(iq.ne.1) go to 110                                             fft16490
      mean=mean+a1(ny-1,2)                                              fft16500
      k=k+1                                                             fft16510
  110 do 120 j=1,ny                                                     fft16520
      a1(j,1)=a1(j,2)                                                   fft16530
  120 a1(j,2)=a1(j,3)                                                   fft16540
```

```
  130 continue                                                          fft16550
      do 140 j=1,ny                                                     fft16560
      if(a1(j,2).eq.dval) go to 140                                     fft16570
      mean=mean+a1(j,2)                                                 fft16580
      k=k+1                                                             fft16590
  140 continue                                                          fft16600
      go to 240                                                         fft16610
c  idval=0: read grid and check boundary values for flagged values.     fft16620
  150 i=0                                                               fft16630
  160 i=i+1                                                             fft16640
      call srdbin(kr,a1,ny)                                             fft16650
      if(i.gt.1.and.i.lt.nx) go to 190                                  fft16660
      do 170 j=1,ny                                                     fft16670
      if(a1(j,1).eq.dval) go to 180                                     fft16680
  170 mean=mean+a1(j,1)                                                 fft16690
      if(i-nx)160,230,160                                               fft16700
  180 write(iw,220)i,j,a1(j,1)                                          fft16710
      close(11)                                                         fft16720
      stop                                                              fft16730
  190 if(a1(1,1).eq.dval.or.a1(ny,1).eq.dval) go to 200                 fft16740
      mean=mean+a1(1,1)+a1(ny,1)                                        fft16750
      go to 160                                                         fft16760
  200 write(iw,210)i,a1(1,1),a1(ny,1)                                   fft16770
  210 format(/," #found flagged value: row = ",i3,2x,"col = 1 or ny",   fft16780
     12x," value = ",2e10.4," may be more?")                            fft16790
  220 format(/," #found flagged value: row = ",i3,2x," col = ",i3,      fft16800
     12x,"value = ",e10.4" may be more?")                               fft16810
      close(11)                                                         fft16820
      stop                                                              fft16830
  230 k=(nx+ny-2)*2                                                     fft16840
c  compute the mean.                                                    fft16850
  240 c=mean/float(k)                                                   fft16860
      ip=iopt1                                                          fft16870
      if(ip.eq.0)ip=iw                                                  fft16880
      write(ip,250)c,k                                                  fft16890
  250 format(/," mean value subtracted = ",e16.8,5x,i5," values used",/)fft16900
      close(11)                                                         fft16910
c  read input array and write array with the mean subtacted.            fft16920
      open(11,mode="in",form="unformatted",file=fname)                  fft16930
      read(kr)title,pgm,ny,nx,nz,yop,dyp,xop,dxp                        fft16940
      fileo="fftfil.grd"                                                fft16950
      open(10,mode="inout",form="unformatted",file=fileo)               fft16960
      write(ko)title,pgm,ny,nx,nz,yo,dy,xo,dx                           fft16970
      do 270 i=1,nx                                                     fft16980
      call srdbin(kr,a1,ny)                                             fft16990
      do 260 j=1,ny                                                     fft17000
  260 a1(j,1)=a1(j,1)-c                                                 fft17010
  270 call swrbin(ko,a1,ny)                                             fft17020
      close(11)                                                         fft17030
      close(10)                                                         fft17040
      return                                                            fft17050
      end                                                               fft17060


c*********************************************************************fft17070
      subroutine srdbin(no,dat,n)                                       fft17080
c       subroutine "srdbin" reads standard grids (binary).              fft17090
c*********************************************************************fft17100
      dimension dat(n)                                                  fft17110
      read(no)dum,dat                                                   fft17120
```

```
      return                                                            fft17130
      end                                                               fft17140


c*****************************************************************************fft17150
      subroutine swrbin(no,dat,n)                                       fft17160
c       subroutine "swrbin" writes standard grids (binary).             fft17170
c*****************************************************************************fft17180
      dimension dat(n)                                                  fft17190
      data dum/0.0/                                                     fft17200
      write(no)dum,dat                                                  fft17210
      return                                                            fft17220
      end                                                               fft17230


c*****************************************************************************fft17240
       subroutine srdda(no,ipos,dat,n)                                  fft17250
c       subroutine "srrda" reads keyed sequential files.                fft17260
c*****************************************************************************fft17270
      dimension dat(n)                                                  fft17280
      read(no'ipos)dat                                                  fft17290
      return                                                            fft17300
      end                                                               fft17310


c*****************************************************************************fft17320
      subroutine swrda(no,ipos,dat,n)                                   fft17330
c       subroutine "swrda" writes keyed sequential files.               fft17340
c*****************************************************************************fft17350
      dimension dat(n)                                                  fft17360
      write(no'ipos)dat                                                 fft17370
      return                                                            fft17380
      end                                                               fft17390
```