CLM-R210

CLM-R210

UKAEA

Report

JETCAT: A PROGRAMME FOR COMPUTERISED TOMOGRAPHY

J. H. WILLIAMSON

CULHAM LABORATORY
Abingdon Oxfordshire

1982

C/13 Available from H. M. Stationery Office £3.00

CLM-R 210

JETCAT: A PROGRAMME FOR COMPUTERISED TOMOGRAPHY

J H Williamson*

Culham Laboratory, Abingdon, Oxon, OX14 3DB, England

A B S T R A C T

A new method of computerised tomography for efficient reconstruction from sparse data has been described by Williamson and Evans [1] and the Fortran programme JETCAT has been written to process the data, and for the interactive design of experimental assemblies to achieve optimal performance. This report, which should be read in conjunction with the programme listing attached, describes the overall structure of the programme, explains each subroutine, and specifies the input and the outputs that occur. The work was motivated by a design study for interferometry on the Joint European Torus (JET), but it can be applied to any other tomographic experiment with sparse data.

* Modes Scientific Consultancy, 73-75 George St, Oxford

September 1981

ISBN 085311 100 6

I. OVERALL STRUCTURE OF JETCAT

The flow of control is shown in the structure diagram for JETCAT, Figure 1 . The programme has been designed so that only those parts needed for any particular application need come to the user's attention. Thus for routine analysis of data, only the first two chapters of the programme are required, as shown by the bold line in the Figure. Setting up the matrices to be used by these production runs when the pixels form a one-dimensional array, say nested annuli, requires programme chapters 3 to 12 . For the alternative of a two-dimensional rectangular mesh of pixels, programme chapters 13 to 15 are substituted for three of the earlier ones. The broken lines in Figure 1 show the flow of control connected with the optimal design of theexperiment using programme chapters 16 through 19, and the final two programme chapters are provided for test purposes. The routines themselves are presented in the listing, and are described in detail in this report, in the order in which they will be encountered, as outlined above.

In order to clarify the calling sequence of the various routines, a naming convention has been used. The routines are arranged in a hierarchy so that each calls only others with a higher level number in its name.

The levels are

0	main programme
1 & 2	major routines
3 & 4	minor routines
5c & 5r	alternative versions for concentric and rectangular pixels
6	specific forms for the JET experiment
7 & 8	standard procedures

The layout of Figure 1 emphasises this hierarchy.

II. PROGRAMMING CONVENTIONS

The program was written in Fortran for the Prime computer and so nearly all the input and output takes place by teleprinter (channel 1). The precalculated matrices are read from channel 5 and can be written to channel 6, and the reading of the experimental parameters can be selected from either channel.

All read statements are done in free format and if the user is uncertain what value to supply in response to a prompt, a default value of zero should be typed in. Input of logical variables was not feasible and this has resulted in slight clumsiness in the coding. Extra diagnostic printing can be obtained from many routines when the variable IPRINT is positive, whereas some of the usual output is suppressed by reducing IPRINT below zero. Labels from 6000 onwards are reserved for formats.

All real variables are of standard length *4 and all integers are short *2, except where library routines demanded otherwise. The routines from the NAG library (which all need REAL*8) are:

F01AAF	matrix inversion
G05CBF(IRND)	initialise random number generator
G05DAF(X1,X2)	uniform distribution in range X1 to X2
G05DYF(N1,N2)	discrete uniform distribution from N1 to N2
G05DDF(A,S)	Gaussian distribution, mean A, standard derivation S
D01BAF	Gaussian quadrature.

(Of these, the fourth is only used in the design stage and the fifth and sixth only in the test routine.) Graphical routines are all from the Ghost library and need INTEGER*4. They only appear in chapter 19 and cover a variety of lines, symbols, axes and contours.

Every effort has been made to avoid confusing jump-of-control statements but the limitations of Fortan mean that GO TO statements cannot be avoided. As far as possible these are used only for WHILE DO and CASE OF kind of constructions.

Exit from subroutines is almost always from a single RETURN statement at its end, but any alternative exits from logical IF statements are highlighted by offsetting the instruction RETURN to the right of column 60. Function sub-programs never affect the values of their arguments.

Common blocks are provided for each of the categories: logical, integer, real and arrays; with a separate block being provided for the test procedures.

III. DESCRIPTION OF THE CHAPTERS OF THE PROGRAM

In this section,when Fortran variables are introduced, the corresponding quantity in the paper will be given in square brackets if the names differ at all.

1. Main Programme

This is arranged in a CASE OF construction, and the input of a single digit will select the case required. Several checks using logical variables ensure that a calculation cannot be attempted before the necessary data has been provided, and if an error is made, the opportunity is provided for trying again. The individual cases are self-documented so need not be described here but the most likely sequences are as follows:

(a) For production runs, case 1 reads in the necessary matrices from channel 5, and reads and processes the first set of data. Case 2 will deal with a subsequent set of data.

(b) Case 3 calculates the matrices for the chosen set of pixels and lines of observation. If required, case 4 will vary the disposition of the lines to improve the accuracy of the reconstruction. When the experimental arrangement is finalised, the matrices can be output by case 8.

(c) If the statistical accuracy of the data varies from one set of data to the next, then case 7 is used because several matrices have to be recalculated over and above the computation in case 2. Similarly, if several different shapes of the

concentric pixels are to be tried for the same data, then case 6 is called for.

(d) Case 5 is a test routine to find out how accurately the program can process known distributions.

2. RUN 2

The reconstructed distribution F(J) [f_j] is calculated for each of the NJ [J] pixels from the NL [L] values of the observed intensities SIGNAL(L) [I_ℓ] equation (6) of the paper by Williamson and Evans.

The value of χ^2 is calculated along with its number of degrees of freedom. It represents the sum of squares of residuals of the least squares fitting of SIGNAL(L), equation (4). If the weights WT(L) [w_ℓ] are not known or if smoothing is employed, the numerical value of χ^2 is not very meaningful, although it is still valuable as a relative indicator to point out sets of data that may be less consistent than others. It is particularly useful in case 6, where the correct shape of the pixels can be determined by finding which shape gives the smallest χ^2.

3. SETUP1

First the parameters specifying the pixels and smoothing is read in and the input channel can be reset if necessary by INCHA5=5. The variable MSYM, which specifies the symmetry, determines the logical variables CONCEN and JET.

Routine PIX5C or PIX5R is called to give matrix B and then the eigenvalue of BB [B^TB] is calculated by MINEI7, if not already known. If this calculation were to fail,then the error flag IFAIL would give a message and causes a premature return.

Next the parameters specifying the lines of observation are fed in. The variable INPUT specifies which of 5 cases of sets of lines is to follow. The variable MORE allows combinations of different cases to be used, e.g. 10 lines in a regular fan configuration plus 4 in specified places plus 3 completely randomly placed.

The routine ends by calling LINES2.

4. PIX5C

The nested set of pixels is topologically equivalent to a

set of equally spaced concentric circles and so the auxiliary equations (10) represent the usual finite difference form of the Laplacian $d^2f/dr^2 + r^{-1}df/dr$. The smoothing equation for the J=1 pixel which is at the origin has to be doubled in magnitude to preserve the same eigenvalue in $\nabla^2 f + k^2 f = 0$. This doubling factor is removed (in routine SETUP1) after the eigenvalue has been calculated. The boundary, at which f=0, would be pixel NJ+1.

A more rigorous derivation would compute $\nabla^2 f$ for the actual contours of f, but it is not clear that this would give any particular advantage. The effect would be to give more smoothing at some radii than others and this could more easily be achieved, if desired, merely by multiplying each smoothing equation by a suitable function of the radius.

5. MINEI7

The smallest eigenvalue EIGEN [k^2] of the positive definite matrix BB gives the smoothest eigenvector that can satisfy the boundary condition f=0. It is obtained by inverting the matrix BB and iterating to find the Rayleigh product. The error flag IFAIL is set if convergence is not obtained.

6. INVER8

The library routine for inverting a matrix requires REAL*8 data so this chapter was written to provide a REAL*4 entry. The error flag is set if the matrix is singular. On other computers, it would probably be possible to use a single precision routine directly.

7. LINES 2

This routine obtains the matrix A by calling OBSER4 for each line of observation and then calculates AA [$A^T wA$].

A default is provided to cover the case when no observations are made at all. This implies that the output profile will be determined entirely by the auxiliary smoothing conditions. The matrix Q cannot be taken to be $\beta^2 B^T B$ as this is singular, so the value of f at the first pixel is 'observed' to be unity to enable Q to be inverted. As explained in the paper, the same profile of f (apart from a scaling factor) would be obtained by any single real observation NL=1.

The routine ends by calling SOLVE4.

8. OBSER4

This is called by LINES2 (and by a chapter in the design phase as will be described later) to make the observation along a single line of sight. It calls OBS5C or OBS5R as appropriate to get one row of matrix A, stored in vector ALINE.

Next the normalisation factor WT(L) is set, if it has not already been done on a previous call to this routine (WT was marked as unset by giving it a negative value in SETUP1. The default is WT=1 which means that each measurement has equal accuracy, the standard derivation being 1 unit (compared with SIGNAL in the test routine of around 100). The default is overridden by the cases of NORM if required. Thus the weighting can be made to depend on the observed signals or on the A matrix as desired. The choice NORM=4 removes the extra importance attached to lines with a long path length through the source: in effect this allows each measurement to have roughly equal percentage accuracy but does not require knowledge of the actual observed signals at this stage.

In the application to JET, all weights are scaled down by 9 to retain the same proportionality between SIGNAL and F that obtains in the other applications. This arose because it was convenient to work in the unit circle or square in the general case, but use dimensions in metres for JET so that PATH is the actual path length.

9. OBS5C

The theory behind the calculation of ALINE with concentric pixels is given in the appendix of the paper. Integration along the path starts at the nearest point on it to the origin and proceeds in each direction until the edge of the source is reached. For safety with non-circular contours, in case the nearest point to the origin is not within the boundary, the steps continue for at least 1 metre in each direction.

10. RRJET6

The function RRJET6(X,Y) gives the r^2-like variable ψ which specifies the contours of the pixels. This quantity ψ (eq. A1)

can be transformed, subject to remaining zero at the origin and unity at the boundary, to give the required spacing of the contours. The variable XM [X_m] is the parameter which specifies which family of contours will be used.

The form adopted for ψ is of course specific to JET. For other applications, the user should replace this chapter by one giving the appropriate function of x,y (ranging from 0 at a point within the source to 1 on the boundary). Similarly, the function could be made to depend on additional parameters introduced through the common block/REALS/.

11. SOLVE4

First the matrix Q is formed out of AA and BB, and it is inverted. Should this inversion fail, then the amount of smoothing β is increased to improve the conditioning of the matrix. The trace, TRACE [T], of Q1 [Q^{-1}] is obtained and the standard derivation of each source term F(J) is stored in STD(J). Then, for each line, USEFUL(L) [u_ℓ] is calculated and the overall value TOTUSE [U] and smallest usefulness UMIN are obtained.

Next a call is made to the appropriate resolution chapter to find out how many pixels are resolved and which ones they are. This data is printed out, together with the value of β, the mean standard deviation of F(J), STDF= $(\text{TRACE/NJ})^{\frac{1}{2}}$, and a value of the trace, TRNM, as normalised by the overall usefulness of the lines. Finally the matrix EMM [M] is computed for use in analysing observed data.

12. RESL5C

Resolution with concentric pixels is easily checked. The pixel at the origin J=1 is defined to be resolved from K, the nearest pixel for which Q1(J,K) is zero or negative. Then we set J=K and repeat the process until the boundary is reached.

13. PIX5R

Now follow the three alternative chapters for a rectangular mesh of pixels. In this implementation the mesh is square but it could easily be generalised by scaling differently in the two directions.

The pixels are set up within the square $|x| \leq 1$, $|y| \leq 1$ (or within the JET boundary). (If any other boundary were required

chapter 10 should be replaced by one which calculates the appropriate ψ function, for instance $\psi = x^2 + y^2$ to obtain a circle.) The total number of pixels NJ will, therefore, increase as the square of NR, the number along each radius, so NJ is not allowed to exceed the storage provided (currently 50). Account is taken of the symmetry (if any), so if it is known that the source function has reflection symmetry across a line, pixels are not unnecessarily duplicated on both sides.

In calculating the smoothing matrix B, a statement function ADJAC is used to add in a contribution from ∇^2 whenever pixel K is one mesh point away from pixel J (or from one of its images under the assumed symmetry).

14. OBS5R

The appendix to the paper should be referred to for this method. The path is broken into segments wherever it crosses one of the mesh-lines and the statement function PROJ (eq. A5) used to accumulate the integral through each pixel. The effect of each image of a pixel is accumulated with that of the pixel itself in calculating the row of the A matrix.

15. RESL5R

Resolution in two dimensions is harder to define than in the concentric case but the method used is to look first of all on the finest length scale (the mesh interval) and systematically increase the inspection length by $2^{\frac{1}{2}}$ each time until the process is complete.

First all pixels are put into one set by setting KS=1 for all of them. Then an attempt is made to split the set entirely, KRES(J)=J, but the separate parts (different KRES) are then recombined (put KRES equal) if they are linked by any positive correlations Q1 on this inspection length.

The next stage is to delete about half the members from each subset by removing any pixel situated at the inspection length from one already retained in the subset. Having done this, KS is given a different value for each subset so that from now on they will be considered as distinct resolved sets. The process is repeated on the larger length until all pixels are either in

a set by themselves or have been deleted.

16. DESIG1

This chapter and the following three enable the set of lines of observation to be changed so as to optimise the efficiency of the experiment.

The choices of subcase offered are:

- 0 leave this routine
- 1 & 6 print out the final choice of lines either on the teleprinter or into an output file
- 2 read in a new value of β and call SOLVE4 again
- 3 graphical output GRAPH3
- 4 & 5 perform the design stage either by minimising STDF or by maximising UMIN

The design stage asks how many attempts NIMP are to be made, which method of selection LIMP is to be used to select the line L to be modified or deleted, and how to choose JIMP the replacement line. These cases are all explained at the head of the program listing. Which option should be used will depend on the problem in hand, but in the JET design study an effective technique was to try LIMP,JIMP in the following order as some combinations are best in the earlier stages of the iteration while others work better later:

- 2,2 remove the least useful line and put it at a random angle through the pixel with the largest variance
- 3,3 try a completely random new line
- 2,4 move the least useful line a little
- 3,4 move any line a little
- 1,1 try a manual choice of L, θ, h - this is particularly useful if the line is closely constrained by the wall.

It is also possible to delete one line without replacing it, or to add an extra one by using LIMP,7 and 4,JIMP respectively. The choice under cases LIMP=2 or 3 can be restricted by LFIX if the design study does not allow all lines to be modified in disposition. For instance if lines 1 to 7 must be vertical, while 8 to NL can be in any direction, then LFIX=7 will make free movements on lines 8 to NL. Then LFIX=0, JIMP=5 will allow sideways adjustment of all lines while preserving their directions.

Having selected the line to be removed, routine INVMM7 is called to remove its effect from matrix Q1 giving Q1NEW. If IFAIL signals that the new inverse does not exist, then this improvement attempt is abandoned and line L is not changed.

Case JIMP=2 requires, and JIMP=1 allows, the new line to be put through a specific pixel rather than being determined explicitly by its intercept h. For the nested JET pixels, h is determined from selected J value SELECJ (which can be fractional) by a bisection method. It should be noted that this makes the line pass through the centre of the pixel when at its closest approach to the origin. This is not quite the same as passing tangentially through the pixel, but the difference is slight and it could be corrected if desired under manual control (JIMP=1) by monitoring the variable TANGEJ which is printed out. This gives the fractional pixel number of the contour tangential to the line.

A check is made using function ALLOW6 that the line does not pass through the wall of the vessel, and the variable WALL shows where the line went (how far in metres the line was on the wrong side of the wall). This check as applied to the JET design study was only made for lines LFIX+1 to NL (the 7 fixed vertical lines of course do not satisfy the conditions for passing through the port provided for the 3 oblique beams), and in other situations the check is easily suppressed altogether by the default value LFIX=0.

The A coefficients for the new line are obtained by OBSER4, and INVMM7 is called to add its effect to yield Q1NEW. If the trace of Q^{-1} has been reduced by this whole process, then the new line is incorporated, otherwise the old one is preserved. The DO loop continues until NIMP attempts at improval have been made.

17. ALLOW6

This function deals with the passage of the line of sight for the oblique beams in JET which have to pass through the constricted funnel. As the walls are made up of straight sections, a check is made that the line passes between the Scylla and Charybdis of each of the five sections. Clearly this chapter should be replaced by one appropriate to the user's own experiment.

18. INVMM7

Rather than alter Q and invert it each time, the process is carried out much more speedily by noting that the new Q is $Q \pm waa^T$ where a is a column vector. Householder's method for obtaining the inverse of a modified matrix can, therefore, be used.

19. GRAPH3

This graphical output routine draws the positions of the lines of sight and the pixels. These are indicated by either their index numbers at their centres for the rectangular mesh, or by the nested contours in the other case. To speed up the drawing of the contours, only every IPIX-th one is drawn, starting from the outside.

Axes are drawn and an identification number IDENT is written onto the graph. The first and last calls to this routine must be indicated by IFIRST and ILAST >0 respectively. (Should the user change his mind and terminate the run earlier than intended, entry with ILAST=2 will force the output buffer to disgorge its graphs.)

20. TEST1

The test chapter checks the operation of the program by generating or reading in a source function, obtaining the signals which would be observed, adding simulated experimental errors of magnitude determined by $WT(L)^{\frac{1}{2}}$*SIZERR, and reconstructing F. The overall accuracy of the computation and the sensitivity to experimental errors can be checked in this way by looking at the root mean square discrepancy in F which is printed out. (This quantity should not be confused with the value of χ^2 calculated in chapter 2 which represents the residuals in I_ℓ, whereas here we are comparing the residuals of f_j with their computed standard deviations.)

In general, the signal is generated directly from the distribution FO(J) using the A matrix. For KIND=3 or 4 (if applied to the correct set of pixels) the analytic results for SIGO(L) are supplied, thus giving a more comprehensive check by additionally assessing the accuracy of the integrations which lead to A. In all cases the peak value of the signal and of the source distribution is around 100 units compared with default standard deviations on the signal of 1 unit.

21. FUN4

This function can be called from TEST1 to calculate the value of a biquartic source function. It will then also be called by a Gaussian quadrature routine to give the observed signal along each line of sight. Each quartic function falls to zero at $|x| = 1$ or $|y| = 1$ and contains two arbitrary coefficients, so that sources with 1, 2 or 4 peaks of intensity in the unit square can easily be generated. If this chapter is modified to use higher degree polynomials, then the degree of the quadrature routine NFUN may have to be increased accordingly to keep the integration exact for a polynomial of that degree (see the NAG library manual for details of this procedure).

IV CONCLUSION

The program JETCAT provides analysis and design of computer assisted tomography for the JET experiment. The design phase is subject to constraints LFIX and ALLOW6 which are easily modified, and the method can be applied to any set of nested pixels by altering the function RRJET6. Similarly any other form of symmetry can be substituted for the mesh of pixels so that the method is of general applicability.

This work was carried out by Modes Scientific Consultancy Ltd., 73-75 George Street, Oxford under contract B-EH-440.
Users of program JETCAT are invited to address any queries or comments to J H Williamson at Modes Scientific Consultancy.

REFERENCE

[1] J H Williamson and D E Evans
Computerised Tomography for Sparse-Data Plasma Physics Experiments CLM-P 656
and IEEE Trans on Plasma Science (1982)

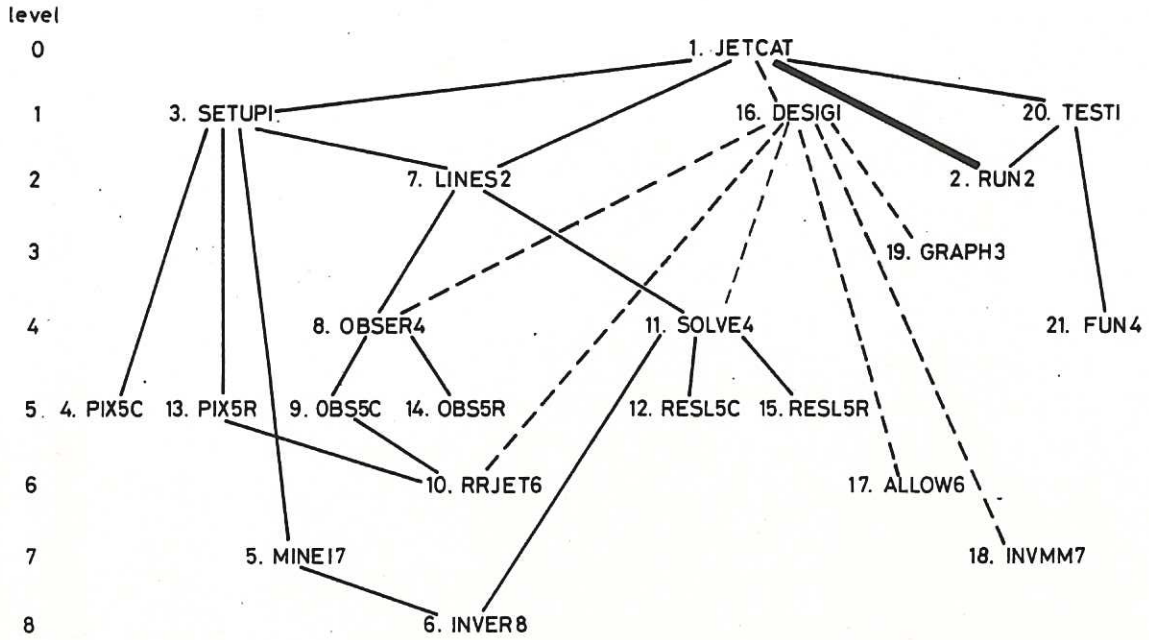

Fig.1 Structure of the program JETCAT.

The JETCAT Programme Listing

```
C  "JETCAT a Program for Computerised Tomography" Culham Laboratory Report CLM R-210
C          by J H Williamson,
C          Modes Scientific Consultancy Ltd, 73-75 George Street, Oxford.
C          29 January 1981.
C          This listing should be read in conjunction with
C          "Computerised Tomography for Sparse Data Plasma Physics Experiments"
C          by J H Williamson & D E Evans,
C          to be submitted for publication in IEEE Trans Plasma Science.
C          Please report any problems with this program to
C          Jim Williamson, 2 Radstock Road, Reading, (0734-)661679
C
C  list of input options & parameters
C          (in case of doubt,  put it equal to zero which is the usual default)
C
Case 1     initialising for production runs
Case 2     production run
Case 3     set up pixels and calculate matrices
Case 4     improve the design, with subcases:
Case 4.1&6   output of final choice of lines on channel 1 or 6
Case 4.2     change the amount of smoothing
Case 4.3     draw lines & pixels
Case 4.4     optimise average standard deviation of reconstruction
Case 4.5     optimise smallest usefulness
Case 4.0     exit from this design stage
Case 5     check accuracy with a test distribution
Case 6     try a new shape of JET contours
Case 7     data has variable weighting
Case 8     write out matrices for future input in case 1
Case 0     stop execution
C
C     NR:       number of pixels radially,  total number is NJ, coordinates are XP & YP
C   MSYM:   10  pixels are concentric circles      , uniform spacing
C           11                          JET D-shaped, uniform spacing
C           12                                      , RSQ spacing
C           13                                      , R**4 spacing
C            0  pixels on rectangular mesh
C            1                              , with up-down symmetry
C           -1                                                      , JET D-shaped
C            2                              , & right-left symmetry
C            3                              , & diagonal symmetry
C   BETA:       smoothing factor to supress short wavelength noise
C IPRINT:       selector for amount of diagnostic printing
C     XM:       parameter to determine shape of JET contours
C SIGNAL:       value of observed signals on line
C INCHA5:    5  change to input channel 5 for prepared data
C  EIGEN:       eigenvalue (only needed when supplied as part of prepared data)
C     NL:       observations are made along NL lines
C  THETA:       direction of the line of observation in degrees
C      H:       distance of line away from the origin
C     WT:       weight of signal
C  INPUT:    1  type in set of lines   THETA,H,WT (use WT=-1 if not known)
C            2  parallel beams
C            3  random set of lines
C            4  fan beams
C            5  one through each pixel   random THETA
C   MORE:    1  if more lines are to be fed in
C NGROUP:       number of lines in each fan or parallel set of beams
C THETA1:       angle of first group of lines
C DTHETA:       change of angle between groups
C     H1:       H for first line in this group
```

```
61) C      DH:        change of H within the group
62) C ANGLE1:         fan angle of first line in group
63) C DANGLE:         change of angle within the fan
64) C RADIAT:         radiation point of the fan
65) C    NORM:    0   do not normalise
66) C             1   read in WT
67) C             2   Poisson weights
68) C             3   equal relative variances
69) C             4   normalise each observation to same total path length
70) C            -K   double pass on lines L>K
71) C    IRND:        initialiser for random numbers
72) C    NIMP:        number of attempts to improve the design
73) C    LIMP:    1   manual change of line
74) C             2   replace least useful line      (if L>LFIX)
75) C             3   replace randomly chosen line   (if L>LFIX)
76) C             4   do not remove any (do insertion only)
77) C    JIMP:    1   manual change of line
78) C             2   put line through most obscure pixel
79) C             3   random new line
80) C             4   vary THETA & H slightly
81) C             5   vary         H slightly
82) C             6   vary THETA & H slightly maintaining the fan radiation point
83) C             7   do not insert any (remove one only)
84) C    LFIX:        constraint on which lines can be altered in the design stage
85) C SELECJ:         (fractional) pixel value which you want the line to pass through
86) C                   (a nonzero value will override any explicit chioce of H)
87) C IFIRST:    1   if first call of graphical output
88) C  ILAST:    1   if graphical output will now be completed
89) C            2                        already completed
90) C  IDENT:         identifier written onto graph
91) C   IPIX:         draw only every IPIX-th pixel contour
92) C   KIND:    1   read in test profile F0(J)
93) C            2   generate parabolic profile
94) C            3            Abel test profile
95) C            4            biquartic profile
96) C     F0:         source function used as a test
97) C  P1-P4:         parameters in the biquartic test function
98) C
99) C  logical variables
100) C CONCEN:        pixels are nested, not on a 2 dimensional mesh
101) C    JET:        pixels have JET D-shape
102) C    QIN:        matrices Q etc needed by production runs are available
103) C    SET:        full setup completed, so non-production runs can be done
104) C  SIGIN:        SIGNAL is available
105) C-----------------------------------------------------------------------
106) C
107) C
108) CHAPTER 1:  main program  JETCAT
109)       LOGICAL  QIN,SET,SIGIN
110)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
111)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
112)      & ,TRNM,EIGEN,XM
113)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
114)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
115)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
116) C
117) C  input case to determine the type of run required
118)       QIN = .FALSE.
119)       SET = .FALSE.
120)   100 WRITE (1,6100)
```

```
121)  6100 FORMAT (//'  CASE ?'/)
122)       READ (1,*)  ICASE
123) C
124) Case 0    stop execution
125)       IF (ICASE.LE.0)  STOP
126) C
127)       IF (ICASE.EQ.2.AND.(NORM.GE.1.AND.NORM.LE.3)) ICASE = 7
128)       IF (.NOT.SET.AND.ICASE.GE.4)  GO TO 100
129)       GO TO (101,102,103,104,105,106,107,118), ICASE
130)       GO TO 100
131) C
132) Case 1    initialising for production runs
133)   101 READ (5,6103) NL,NJ,(STD(J),(EMM(J,L),L=1,NL),J=1,NJ)
134)      &  ,(WT(L),(A(L,J),J=1,NJ),L=1,NL)
135) C     (this data was prepared automatically by case 8)
136)       QIN = .TRUE.
137)       SIGIN = .FALSE.
138) C
139) Case 2    production run
140)   102 IF (.NOT.QIN)  GO TO 100
141)       WRITE (1,6101)
142)  6101 FORMAT(' SIGNALS ')
143)       READ (1,*)  (SIGNAL(L),L=1,NL)
144)       SIGIN = .TRUE.
145)       CALL RUN2
146)       GO TO 100
147) C
148) Case 3    set up pixels and calculate matrices
149)   103 CALL SETUP1
150)       SET = .TRUE.
151)       QIN = .TRUE.
152)       SIGIN = .FALSE.
153)       GO TO 100
154) C
155) Case 4    improve the design
156)   104 CALL DESIG1
157)       SIGIN = .FALSE.
158)       GO TO 100
159) C
160) Case 5    check accuracy with a test distribution
161)   105 CALL TEST1
162)       SIGIN = .TRUE.
163)       GO TO 100
164) C
165) Case 6    try a new shape of JET contours
166)   106 WRITE (1,6102)
167)  6102 FORMAT ('  XM IPRINT')
168)       READ (1,*)  XM,IPRINT
169)       CALL LINES2
170)       IF (SIGIN)  CALL RUN2
171)       GO TO 100
172) C
173) Case 7    data has variable weighting
174)   107 WRITE (1,6101)
175)       READ (1,*)  (SIGNAL(L),L=1,NL)
176)       SIGIN = .TRUE.
177)       DO 117 L = 1,NL
178)   117 WT(L) = -1.
179)       CALL LINES2
180)       CALL RUN2
```

```
181)        GO TO 100
182) C
183) Case 8    write out matrices for future input in case 1
184)   118 WRITE(6,6103)NL,NJ,(STD(J),(EMM(J,L),L=1,NL),J=1,NJ)
185)       &  ,(WT(L),(A(L,J),J=1,NJ),L=1,NL)
186)  6103 FORMAT (2I5/(1P6E13.5))
187)        GO TO 100
188)        END
189) C-----------------------------------------------------------------------
190) C
191) C
192) CHAPTER 2:  process this observed data
193)        SUBROUTINE RUN2
194)        COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
195)        COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
196)       &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
197)       & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
198) C
199) C      reconstruct source function F from signal
200)        WRITE (1,6200)
201)  6200 FORMAT (' PIXEL   F(J)    STDEV')
202)        DO 201 J = 1,NJ
203)           F(J) = 0.
204)           DO 200 L = 1,NL
205)   200     F(J) = F(J) + EMM(J,L)*SIGNAL(L)
206)           IF (IPRINT.GT.-9)  WRITE (1,6201) J,F(J),STD(J)
207)  6201     FORMAT (1X,I3,2F9.2)
208)   201 CONTINUE
209) C
210) C      compare sum of squares of residuals with what is expected
211)        CHISQ = 0.
212)        DO 203 L = 1,NL
213)           DSIG = SIGNAL(L)
214)           DO 202 J = 1,NJ
215)   202     DSIG = DSIG - A(L,J)*F(J)
216)   203 CHISQ = CHISQ + DSIG*DSIG*WT(L)
217)        NLJ = NL - NJ
218)        WRITE (1,6202)  NLJ,CHISQ
219)  6202 FORMAT ('  CHISQ(',I3,') =',F9.3/)
220)        RETURN
221)        END
222) C-----------------------------------------------------------------------
223) C
224) C
225) CHAPTER 3:  select pixels and lines
226)        SUBROUTINE SETUP1
227)        LOGICAL          CONCEN,JET
228)        COMMON /LOGICS/ CONCEN,JET
229)        COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
230)        COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
231)       &  ,TRNM,EIGEN,XM
232)        COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
233)       &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
234)       & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
235) C
236)        PI = 3.14159265
237)        WRITE (1,6300)
238)  6300 FORMAT (' NR MSYM BETA INCHA5 IPRINT')
239)        IRD = 1
240)        READ (IRD,*)   NR,MSYM,BETA,INCHA5,IPRINT
```

```
      EIGEN = 0.
      IF (INCHA5.EQ.5)  IRD = 5
      IF (INCHA5.EQ.5)  READ (IRD,*)  NR,MSYM,BETA,EIGEN
      CONCEN = (MSYM.GE.10)
      JET = (MSYM.GE.11.OR.MSYM.EQ.-1)
C
C  set up pixels & calculate smoothing matrix B
      XM = -0.6031
      IF (CONCEN)       CALL PIX5C
      IF (.NOT.CONCEN)  CALL PIX5R
C
      IFAIL = 110
      IF (EIGEN.NE.0.)  IFAIL = 0
      IF (EIGEN.EQ.0.)  CALL MINEI7 (B,NJ,EIGEN,IFAIL)
      IF (IPRINT.GE.4.OR.IFAIL.NE.0)  WRITE (1,6301)  IFAIL,EIGEN
      IF (IFAIL.NE.0)                                       RETURN
      DO 300 J = 1,NJ
         B(J,J) = B(J,J) - EIGEN
         IF (CONCEN)  B(1,J) = B(1,J)/2.
         IF(IPRINT.GE.5) WRITE(1,6301)J,XP(J),YP(J),(B(J,KK),KK=1,NJ)
 6301    FORMAT (' B ',I3,10F8.3/(9X,10F8.3))
  300 CONTINUE
C
      DO 302 J = 1,NJ
      DO 302 K = 1,J
         BBB = 0.
         DO 301 JJ = 1,NJ
  301    BBB = BBB + B(JJ,J)*B(JJ,K)
         BB(J,K) = BBB
  302 BB(K,J) = BBB
C
C  select NL lines of observation (in one or batches of NLL)
      L = 0
      HR = 1.
      IF (JET)  HR = 2.1
  303 WRITE (1,6302)
 6302    FORMAT (' INPUT NL MORE NORM IRND')
         READ (IRD,*)  INPUT,NLL,MORE,NORM,IRND
         CALL G05CBF (IRND)
C
         NL = L + NLL
         IF (NLL.LE.0)  GO TO 327
         GO TO (311,312,313,312,313), INPUT
  311       WRITE (1,6303)
 6303       FORMAT (' THETA H WT')
            GO TO 313
  312       WRITE (1,6304)
 6304       FORMAT (' NGROUP THETA1 DTHETA H1 DH ANGLE1 DANGLE RADIAT')
            READ (IRD,*)  NGROUP, THETA1,DTHETA, H1,DH,
     &      ANGLE1,DANGLE,RADIAT
C
  313    DO 326 LL = 1,NLL
            L = L + 1
            SIGNAL(L) = 0.
            WT(L) = -1.
            FAN(L) = 0.
            GO TO (321,322,323,324,325), INPUT
  321          READ (IRD,*)  THETA(L), H(L), WT(L)
               GO TO 326
  322          THETA(L) = THETA1 + ((LL-1)/NGROUP)*DTHETA
```

```
301)                H(L) = H1 + MOD(LL-1,NGROUP)*DH
302)                GO TO 326
303)   323          THETA(L) = G05DAF(-90.D0,90.D0)
304)                H(L) = G05DAF(-1.D0,1.D0) * HR
305)                GO TO 326
306)   324          AL = ANGLE1 + MOD(LL-1,NGROUP)*DANGLE
307)                THETA(L) = THETA1 + ((LL-1)/NGROUP)*DTHETA + AL
308)                H(L) = -RADIAT*SIN(AL*PI/180.)
309)                FAN(L) = RADIAT
310)                GO TO 326
311)   325          THETA(L) = G05DAF(-90.D0,90.D0)
312)                LM = MOD(L-1,NJ)+1
313)                THRAD = THETA(L)*PI/180.
314)                H(L) = YP(LM)*COS(THRAD)-XP(LM)*SIN(THRAD)
315)   326     CONTINUE
316) C     while more, go back to 303 for next one
317)   327 IF (MORE.GT.0)  GO TO 303
318) C
319)       CALL LINES2
320)       RETURN
321)       END
322) C-----------------------------------------------------------------------
323) C
324) C
325) CHAPTER 4:  set up concentric pixels & smoothing matrix B
326)       SUBROUTINE PIX5C
327)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
328)       COMMON /ARRAYS/  SIGNAL(50),EMN(50,50),F(50),STD(50),KRES(50),
329)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
330)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
331) C
332)       NJ = NR
333)       DO 401 J = 1,NJ
334)          RJ = (J-1.)/NJ
335)          XP(J) = RJ
336)          YP(J) = 0.
337)          DO 400 K = 1,NJ
338)   400    B(J,K) = 0.
339)          B(J,J) = NR*NR
340)          IF (J.GT.1)  B(J,J-1) = (-.5 + .25/(J-1))*NR*NR
341)          IF (J.GT.1.AND.J.LT.NJ) B(J,J+1) = (-.5 - .25/(J-1))*NR*NR
342)   401 CONTINUE
343)       B(1,2) = -2*NR*NR
344)       B(1,1) = 2*NR*NR
345)       RETURN
346)       END
347) C-----------------------------------------------------------------------
348) C
349) C
350) CHAPTER 5:  find smallest eigenvalue of matrix
351) C     by inverting and finding the Raleigh product
352)       SUBROUTINE MINEI7 (P,NJ,EIGEN,IFAIL)
353)       DIMENSION  P(50,50), P1(50,50), V(50), VN(50)
354)       EIGOLD = 1.
355)       CALL INVER8 (P,P1,NJ,IFAIL)
356)       IF (IFAIL.EQ.1)                                        RETURN
357)       DO 500 J = 1,NJ
358)   500 V(J) = 1.
359)       DO 504 ITER = 1,50
360)          VV = 0.
```

```
361)          VVN = 0.
362)          DO 502 K = 1,NJ
363)             VN(K) = 0.
364)             DO 501 J = 1,NJ
365)   501       VN(K) = VN(K) + P1(K,J)*V(J)
366)             VVN = VVN + V(K)*VN(K)
367)   502     VV = VV + V(K)*V(K)
368)          EIGEN = VV/VVN
369)          DO 503 J = 1,NJ
370)   503     V(J) = VN(J)*ABS(EIGEN)
371)          IF (ABS(EIGEN/EIGOLD-1.).LT..00001)                RETURN
372)   504 EIGOLD = EIGEN
373)       IFAIL = 50
374)       RETURN
375)       END
376) C-----------------------------------------------------------------------
377) C
378) C
379) CHAPTER 6:  invert matrix (REAL*4 calling of REAL*8 library routine)
380)       SUBROUTINE INVER8 (Q,Q1,NJ,IFAIL)
381)       DIMENSION  Q(50,50),Q1(50,50)
382)       REAL*8  Q8(50,50), Q81(50,50),WORK(50)
383)       DO 600 J = 1,NJ
384)       DO 600 K = 1,NJ
385)   600 Q8(J,K) = Q(J,K)
386)       CALL F01AAF (Q8,50,NJ,Q81,50,WORK,IFAIL)
387)       DO 601 J = 1,NJ
388)       DO 601 K = 1,NJ
389)   601 Q1(J,K) = Q81(J,K)
390)       RETURN
391)       END
392) C-----------------------------------------------------------------------
393) C
394) C
395) CHAPTER 7:  obtain all the matrices for all the lines
396)       SUBROUTINE LINES2
397)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
398)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
399)      &  ,TRNM,EIGEN,XM
400)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
401)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
402)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
403) C
404) C  default for NL=0
405)       IF (NL.GT.0)  GO TO 701
406)          ALINE(1) = 1.
407)          DO 700 J = 2,NJ
408)   700    ALINE(J) = 0.
409) C
410) C  calculate A coeffs for all lines
411)   701 DO 702 J = 1,NJ
412)       DO 702 K = 1,NJ
413)   702 AA(J,K) = 0.
414)       WRITE (1,6700)
415)  6700 FORMAT (/'        L  THETA      H      WEIGHT    PATH')
416)       DO 704 L = 1,NL
417)          IF (NL.GT.0)  CALL OBSER4 (L,THETA(L),H(L))
418)          IF (IPRINT.GE.-1.AND.NL.GT.0)
419)      &   WRITE (1,6701) L,THETA(L),H(L),WT(L),PATH
420)  6701    FORMAT (' LINE',I3,F7.1,9F8.3)
```

```
421)          DO 704 J = 1,NJ
422)             A(L,J) = ALINE(J)
423)             DO 704 K = 1,NJ
424)   704 AA(J,K) = AA(J,K) + ALINE(J)*ALINE(K)*WT(L)
425)       NL = MAXO(NL,1)
426) C
427)       CALL SOLVE4
428)       RETURN
429)       END
430) C-----------------------------------------------------------------------
431) C
432) C
433) CHAPTER 8:  calculate A coeffs & weight WT for line L
434)       SUBROUTINE OBSER4 (L,THLINE,HLINE)
435)       LOGICAL          CONCEN,JET
436)       COMMON /LOGICS/ CONCEN,JET
437)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
438)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
439)      &  ,TRNM,EIGEN,XM
440)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
441)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
442)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
443) C
444) C     observe along the line
445)       S = SIN(THLINE*PI/180.)
446)       C = COS(THLINE*PI/180.)
447)       PATH = 0.
448)       IF (CONCEN)        CALL OBS5C (S,C,HLINE)
449)       IF (.NOT.CONCEN)  CALL OBS5R (S,C,HLINE)
450) C
451) C  normalise line (if not done on a previous call)
452)       IF (WT(L).GE.0.)  GO TO 806
453)          WT(L) = 1.
454)          IF (NORM.LT.0.AND.L.GT.IABS(NORM)) WT(L) = 4
455)          IF (NORM.EQ.0)  GO TO 805
456)          GO TO (801,802,802,804), NORM
457)             GO TO 805
458)   801       WRITE (1,6800)
459)  6800       FORMAT (' WTS')
460)             READ (1,*)  (WT(LL),LL=1,NL)
461)             GO TO 806
462)   802       IF (SIGNAL(L).NE.0.)  WT(L) = (100./SIGNAL(L))**(NORM-1)
463)             GO TO 805
464)   804       IF (PATH.NE.0.)  WT(L) = 1./(PATH*PATH)
465)   805    IF (JET)  WT(L) = WT(L)/9.
466)   806 IF(IPRINT.GE.2)WRITE (1,6801) L,(ALINE(J),J=1,NJ)
467)  6801 FORMAT ('   A ',I3,15F8.3/(9X,15F8.3))
468)       RETURN
469)       END
470) C-----------------------------------------------------------------------
471) C
472) C
473) CHAPTER 9:  work out path in each pixel (concentric shells)
474)       SUBROUTINE OBS5C (S,C,HLINE)
475)       LOGICAL          CONCEN,JET
476)       COMMON /LOGICS/ CONCEN,JET
477)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
478)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
479)      &  ,TRNM,EIGEN,XM
480)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
```

```
481)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
482)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
483) C
484)       DL = 0.01
485)       DO 900 J = 1,NJ
486)   900 ALINE(J) = 0.
487)       STEP = ABS(DL)
488)       TANGEJ = 999.
489)       DO 903 ISIDE = 1,2
490)          TL = 0.
491)          X = -HLINE*S + 0.5*C*DL
492)          Y = HLINE*C + 0.5*S*DL
493)   901    RR = SQRT(X*X+Y*Y)
494)             IF (JET)  RR = RRJET6(X,Y)
495)             RJ = 1.+RR*NJ
496)             TANGEJ = AMIN1(RJ,TANGEJ)
497)             J = RJ
498)             IF (J.GT.NJ.AND.TL.GT.1.)  GO TO 903
499)             IF (J.GT.NJ)  GO TO 902
500)             FR1 = STEP*(RJ*(RJ-2.)-J*(J-2))/(2*J-1)
501)             IF (J+1.LE.NJ)  ALINE(J+1) = ALINE(J+1) + FR1
502)             ALINE(J) = ALINE(J) + STEP-FR1
503)             IF (IPRINT.GE.7.AND.AMOD(PATH,15.).LT..1)
504)      &       WRITE (1,6900)  J,RJ,FR1,ALINE(J),ALINE(J+1),X,Y
505)  6900       FORMAT (' OBS5C',I3,10F8.3)
506)             PATH = PATH + STEP
507)   902       TL = TL + STEP
508)             X = X + C*DL
509)             Y = Y + S*DL
510)          GO TO 901
511)   903 DL = -DL
512)       RETURN
513)       END
514) C-----------------------------------------------------------------
515) C
516) C
517) CHAPTER 10:  function determining the contours of the density
518) C      the origin is the magnetic axis, so X coordinates = major radius - R0
519)       FUNCTION  RRJET6(X,Y)
520)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
521)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
522)      &  ,TRNM,EIGEN,XM
523) C
524)       XX0 = X*(X+6.4262)
525)       CCM = XM*(XM+6.6262)
526)       RRJET6 = (XX0*XX0 + (54.76-2*CCM*XX0+CCM*CCM)*Y*Y/4.41) / 54.76
527)       IF (MSYM.EQ.11)  RRJET6 = SQRT(AMAX1(0.,RRJET6))
528)       IF (MSYM.EQ.13)  RRJET6 = RRJET6*RRJET6
529)       RETURN
530)       END
531) C-----------------------------------------------------------------
532) C
533) C
534) CHAPTER 11:  complete the solution for this set of lines & pixels
535)       SUBROUTINE SOLVE4
536)       LOGICAL          CONCEN,JET
537)       COMMON /LOGICS/ CONCEN,JET
538)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
539)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
540)      &  ,TRNM,EIGEN,XM
```

```
541)        COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
542)       &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
543)       & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
544) C
545) C   form Q matrix, try to invert it, adjust BETA if necessary
546)        IF ((NL.LT.NJ.OR.AA(1,1).EQ.0.).AND.BETA.EQ.0.)  BETA = .001
547)  1100 SMSQ = BETA*BETA
548)           IF (JET)  SMSQ = SMSQ/9.
549)           DO 1101 J = 1,NJ
550)           DO 1101 K = 1,NJ
551)  1101     Q(J,K) = AA(J,K) + SMSQ*BB(J,K)
552) C
553)           DO 1102 J = 1,NJ
554)  1102     IF(IPRINT.GE.4)WRITE (1,7100) (Q(J,K),K=1,J)
555)           IFAIL = 11
556)           CALL INVER8 (Q,Q1,NJ,IFAIL)
557)           IF (IFAIL.NE.0)  BETA = .9*ABS(BETA) + .001
558)        IF (IFAIL.NE.0)  GO TO 1100
559)           DO 1103 J = 1,NJ
560)  1103     IF(IPRINT.GE.3)WRITE (1,7100) (Q1(J,K),K=1,J)
561)  7100     FORMAT(1X,10F12.3)
562) C
563) C  calculate trace of Q1 and the usefulnesses
564)        TRACE = 0.
565)        DO 1104 J = 1,NJ
566)           STD(J) = SQRT(ABS(Q1(J,J)))
567)  1104 TRACE = TRACE + ABS(Q1(J,J))
568)        STDF = SQRT(TRACE/NJ)
569)        TOTUSE = 0.
570)        UMIN = 1.E20
571)        DO 1106 L = 1,NL
572)           UL = 0.
573)           DO 1105 J = 1,NJ
574)           DO 1105 K = 1,NJ
575)  1105     UL = UL + A(L,J)*A(L,K)*Q1(K,J)
576)           USEFUL(L) = UL*WT(L)
577)           UMIN = AMIN1(UMIN,USEFUL(L))
578)  1106 TOTUSE = TOTUSE + USEFUL(L)
579)        TRNM = (TRACE/NJ)/(TOTUSE/NL)
580)        IF (CONCEN)        CALL RESL5C
581)        IF (.NOT.CONCEN)  CALL RESL5R
582)        IF (IPRINT.GE.-2)  WRITE (1,7101) NKRES,(KRES(K),K=1,NKRES)
583)  7101 FORMAT (I4,' PIXELS RESOLVED:',36I3)
584)        IF (IPRINT.GE.-3)  WRITE (1,7102)  (STD(J),J=1,NJ)
585)  7102 FORMAT (' STDEV PIXELS',15F7.2/(13X,15F7.2))
586)        IF (IPRINT.GE.-3)  WRITE (1,7103)  (USEFUL(L),L=1,NL)
587)  7103 FORMAT (' USEFUL',20F5.2/(7X,20F5.2))
588)        WRITE (1,7104)  STDF,TOTUSE,UMIN,TRNM,BETA
589)  7104 FORMAT (' STDF',F8.3,'   TOTUSE',F7.3,'   UMIN',F7.3,
590)       &  '   TRNM',F11.3,'   BETA',F10.5/)
591) C
592) C  get solution matrix EMM
593)        DO 1107 L = 1,NL
594)           DO 1107 J = 1,NJ
595)           EMM(J,L) = 0.
596)           DO 1107 K = 1,NJ
597)  1107     EMM(J,L) = EMM(J,L) + Q1(K,J)*A(L,K)*WT(L)
598)        RETURN
599)        END
600) C-----------------------------------------------------------------------
```

```
601) C
602) C
603) CHAPTER 12:  resolution analysis for concentric pixels
604)       SUBROUTINE RESL5C
605)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
606)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
607)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
608)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
609) C
610)       NKRES = 1
611)       K1 = 1
612)       KRES(1) = 1
613)       DO 1200 K2 = 1,NJ
614)       IF (K2.LT.K1+1)  GO TO 1200
615)       IF (Q1(K1,K2).GT.0.)  GO TO 1200
616)          NKRES = NKRES + 1
617)          KRES(NKRES) = K2
618)          K1 = K2
619)  1200 CONTINUE
620)       RETURN
621)       END
622) C-----------------------------------------------------------------------
623) C
624) C
625) CHAPTER 13:  pixels and smoothing on a rectangular mesh
626)       SUBROUTINE PIX5R
627)       LOGICAL          CONCEN,JET
628)       COMMON /LOGICS/ CONCEN,JET
629)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
630)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
631)      &  ,TRNM,EIGEN,XM
632)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
633)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
634)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
635) C
636)       ADJAC(X1,Y1,X2,Y2) = (SIGN(.5,ABS((X1-X2)*(X1-X2)+(Y1-Y2)*(Y1-Y2)
637)      &       -DR*DR)-.001)-.5)*.25*NRR
638) C
639) C  set up rectangular mesh of pixels
640)       NRR = NR*NR
641)       DR = 1./NR
642)       IF (JET)  DR = DR*2.25
643)       IX = -NR
644)       IF (MSYM.GE.2)  IX = -1
645)       IY = -NR + 1
646)       IF (IABS(MSYM).GE.1)  IY = 0
647)       NJ = 0
648)  1300 IX = IX + 1
649)       IF (MSYM.GE.3.AND.IX.LE.IY)  GO TO 1301
650)       IF (MSYM.LT.3.AND.IX.LE.NR)  GO TO 1301
651)          IX = -NR + 1
652)          IF (MSYM.GE.2.)  IX = 0
653)          IY = IY + 1
654)       IF (IY.GE.NR)  GO TO 1302
655)  1301 IF (IX.GE.NR)  GO TO 1300
656)       IF (JET.AND.RRJET6(IX*DR,IY*DR).GT.1.)  GO TO 1300
657)          NJ = NJ + 1
658)          XP(NJ) = IX*DR
659)          YP(NJ) = IY*DR
660)       IF (NJ.LT.50)  GO TO 1300
```

```
C
C  calculate smoothing matrix
 1302 DO 1305 J = 1,NJ
      DO 1305 K = 1,NJ
         B(J,K) = 0.
         X = XP(K)
         Y = YP(K)
 1303    B(J,K) = B(J,K) + ADJAC(XP(J),YP(J),X,Y)
         IF (MSYM.LT.3.OR.ABS(Y).EQ.ABS(X))  GO TO 1304
         XY = X
         X = Y
         Y = XY
         IF (ABS(Y).LT.ABS(X))  GO TO 1303
 1304    IF (MSYM.GE.2)  X = -X
         IF (MSYM.GE.2.AND.X.LT.0)  GO TO 1303
         Y = -Y
         IF (IABS(MSYM).GE.1.AND.Y.LT.0)  GO TO 1303
 1305    B(J,J) = NRR
      RETURN
      END
C-----------------------------------------------------------------------
C
C
CHAPTER 14:  work out path in each pixel (rectangular mesh)
      SUBROUTINE OBS5R (S,C,HLINE)
      COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
      COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
     &  ,TRNM,EIGEN,XM
      COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
     &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
     & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
C
      PROJ(Z) = Z * (1. + SIGNX*(S*PERP-.5*Z*C) - SIGNY*(C*PERP+.5*Z*S)
     &  - SIGNX*SIGNY * ((PERP*PERP - Z*Z/3.)*SC - Z*PERP*C2))
C
      HN = HLINE/DR
      RT2 = SQRT(2.)
      SC = S*C
      C2 = .5*(C*C-S*S)
      CI = SIGN(1./AMAX1(ABS(C),1.E-20),C)
      SI = SIGN(1./AMAX1(ABS(S),1.E-20),S)
      DO 1406 J = 1,NJ
         ATERM = 0.
         X = XP(J)/DR
         Y = YP(J)/DR
 1400    PERP = HN - Y*C + X*S
         IF(IPRINT.GE.7)WRITE (1,7400)  J,X,Y,PERP
         IF (ABS(PERP).GT.RT2)  GO TO 1404
         Z23 = S*PERP*CI
         Z32 =-C*PERP*SI
         Z1 = AMAX1(Z23-ABS(CI), Z32-ABS(SI))
         Z4 = AMIN1(Z23+ABS(CI), Z32+ABS(SI))
         Z2 = AMIN1(Z23, Z32)
         Z3 = AMAX1(Z23, Z32)
 1401    IF (Z1.LT.Z2.AND.Z2.LT.Z3)  GO TO 1403
         IF (Z1.LT.Z3.AND.Z3.LT.Z4)  GO TO 1402
         IF (Z1.GE.Z4)  GO TO 1404
         Z3 = Z4
 1402    Z2 = Z3
 1403    SIGNX = SIGN(1., C*(Z1+Z2)*.5-S*PERP)
```

```
721)          SIGNY = SIGN(1., S*(Z1+Z2)*.5+C*PERP)
722)          ATERM = ATERM + PROJ(Z2) - PROJ(Z1)
723)  7400    FORMAT (' OBS5R',I3,10F8.3)
724)          IF(IPRINT.GE.7)WRITE (1,7400) J,Z1,Z2,Z3,Z4,SIGNX,SIGNY,ATERM
725)          Z1 = Z2
726)          GO TO 1401
727) C
728)  1404    IF (MSYM.LT.3.OR.ABS(Y).EQ.ABS(X))  GO TO 1405
729)          XY = X
730)          X = Y
731)          Y = XY
732)          IF (MSYM.GE.3.AND.ABS(Y).LT.ABS(X))  GO TO 1400
733)  1405    IF (MSYM.GE.2)  X = -X
734)          IF (MSYM.GE.2.AND.X.LT.0.)  GO TO 1400
735)          Y = -Y
736)          IF (IABS(MSYM).GE.1.AND.Y.LT.0.)  GO TO 1400
737)          ALINE(J) = ATERM*DR
738)  1406 PATH = PATH + ALINE(J)
739)       RETURN
740)       END
741) C-----------------------------------------------------------------------
742) C
743) C
744) CHAPTER 15:  resolution in a 2-dimensional array of pixels
745)       SUBROUTINE RESL5R
746)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
747)       COMMON /REALS / PI,HR,DR,BETA,PATH,TRO,TOTUSE,UMIN,TANGEJ
748)      &  ,TRNM,EIGEN,XM
749)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
750)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
751)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
752)       DIMENSION KS(50)
753) C
754) C  start with all pixels in group 1 & look at adjacent pixels
755)       JDEL = 0
756)       DRSQ = DR*DR
757)       DO 1500 J = 1,NJ
758)  1500 KS(J) = 1
759) C
760) C  subdivide all groups if possible but ....
761)  1501 DO 1505 J = 1,NJ
762)       IF (KS(J).LT.0)  GO TO 1505
763)          KRES(J) = J
764) C        .... looking at closest pixel(s) to each one in turn
765)          CLOSE = 1.E9
766)          DO 1502 K = 1,J
767)             IF (KS(K).NE.KS(J).OR.K.EQ.J)  GO TO 1502
768)             DIST = (XP(K)-XP(J))**2+(YP(K)-YP(J))**2
769)             CLOSE = AMIN1(CLOSE,DIST)
770)  1502    CONTINUE
771)          DO 1504 K = 1,J
772)             IF (KS(K).NE.KS(J))  GO TO 1504
773)             DIST = (XP(K)-XP(J))**2+(YP(K)-YP(J))**2
774)             IF (ABS(DIST-CLOSE).GT..001)  GO TO 1504
775)             IF (Q1(J,K).LE.0.)  GO TO 1504
776) C           .... amalgamate groups if any positive correls link them
777)             DO 1503 M = 1,J
778)  1503       IF (KRES(M).EQ.KRES(J).AND.KS(M).GT.0)  KRES(M) = KRES(K)
779)  1504    CONTINUE
780)  1505 CONTINUE
```

```
781)        IF (IPRINT.GE.2)  WRITE (1,7500)  JDEL,(KRES(J),J=1,NJ)
782)  7500 FORMAT (I4,' DELETED',36I3)
783) C
784) C  delete pixels at odd mesh points relative 1st pixel in group
785)        JDELL = JDEL
786)        DO 1507 J = 1,NJ
787)           DO 1506 K = 1,J
788)              IF (KRES(K).NE.KRES(J))  GO TO 1506
789)              DIST = (XP(K)-XP(J))**2+(YP(K)-YP(J))**2
790)              MESH = (DIST+.001) / DRSQ
791)              IF (MOD(MESH,2).EQ.0)  GO TO 1506
792)              JDEL = JDEL + 1
793)              KRES(J) = -JDEL
794)              GO TO 1507
795)  1506     CONTINUE
796)  1507 KS(J) = KRES(J)
797)           DRSQ = DRSQ*2.
798)        IF (JDEL.NE.JDELL)  GO TO 1501
799) C  while deleting, increase inspection step *sqrt(2) and go back to 1501
800) C
801) C      form the list of resolved pixels
802)        NKRES = 0
803)        DO 1508 J = 1,NJ
804)           IF (KS(J).LT.0)  GO TO 1508
805)           NKRES = NKRES+1
806)           KRES(NKRES) = J
807)  1508 CONTINUE
808)        RETURN
809)        END
810) C-----------------------------------------------------------------------
811) C
812) C
813) CHAPTER 16:  design of the experiment
814)        SUBROUTINE DESIG1
815)        LOGICAL          CONCEN,JET
816)        COMMON /LOGICS/ CONCEN,JET
817)        COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
818)        COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
819)       &  ,TRNM,EIGEN,XM
820)        COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
821)       &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
822)       & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
823)        DIMENSION Q1NEW(50,50)
824)        ASIN(X,Y) = ATAN2(X,SQRT(Y*Y-X*X))*180./PI
825) C
826)  1600 WRITE (1,7600)
827)  7600 FORMAT ('  SUBCASE ?')
828)        READ (1,*)  ISCASE
829) C
830) Case  0    leave this design stage
831)        IF (ISCASE.LE.0)                                        RETURN
832) C
833)        GO TO (1601,1602,1603,1604,1604,1601), ISCASE
834)        GO TO 1600
835) C
836) Case 1&6   output of final choice of lines on channel 1 or 6
837)  1601 IF (ISCASE.EQ.1)  WRITE (1,7601)
838)       &  NR,MSYM,BETA,EIGEN,STDF,TOTUSE,TRNM,   NL,IRND
839)  7601 FORMAT ('      NR MSYM    BETA    EIGEN      STDF      TOTUSE',
840)       &  '      TRNM'/ 2X,2I4,5F10.4/
```

```
841)       &  '  INPUT  NL  MORE NORM   IRND'/
842)       &  '    1',I5,'    0    0',I8/
843)       &  '       THETA        H          WT      USEFUL     L')
844)        IF (ISCASE.EQ.6)  WRITE (6,7602)
845)       &  NR,MSYM,BETA,EIGEN,STDF,TOTUSE,TRNM,    NL,IRND
846)  7602 FORMAT (2X,2I4,5F10.4 / '    1',I5,'    0    0',I8)
847)       WRITE (ISCASE,7603)  (THETA(L),H(L),WT(L),USEFUL(L),L,L=1,NL)
848)  7603 FORMAT (1X,F10.1,3F10.3,I5)
849)       GO TO 1600
850) C
851) Case  2    change the amount of smoothing
852)  1602 WRITE (1,7604)
853)  7604 FORMAT (' BETA IPRINT')
854)       READ (1,*)  BETA,IPRINT
855)       CALL SOLVE4
856)       GO TO 1600
857) C
858) Case  3    draw lines & pixels
859)  1603 CALL GRAPH3
860)       GO TO 1600
861) C
862) Case 4&5   optimise on STDF or on UMIN
863)  1604 WRITE (1,7605)
864)  7605 FORMAT (' NIMP LIMP JIMP LFIX RND IPRINT')
865)       READ (1,*)  NIMP,LIMP,JIMP,LFIX,IRND,IPRINT
866)       IF (IRND.GT.0)  CALL G05CBF (IRND)
867) C
868)       DO 1678 IMP = 1,NIMP
869)       IF (NIMP.LE.0)  GO TO 1678
870)       IF (LIMP.EQ.4.OR.JIMP.EQ.7)  NIMP = 1
871)          IF (LIMP.NE.1.AND.JIMP.NE.1)  GO TO 1605
872)             WRITE (1,7606)
873)  7606       FORMAT ('  L  TH   H  SELECJ')
874)             READ (1,*)  L,THKN,HKN,SELECJ
875)             JJJ = SELECJ
876) C
877) C        choose line to be deleted
878)  1605    GO TO (1631,1612,1623,1634), LIMP
879) C           find least useful line
880)  1612       VLMIN = 1.E21
881)             DO 1613 LL = 1,NL
882)             IF (USEFUL(LL).GT.VLMIN.OR.LL.LE.LFIX)  GO TO 1613
883)                VLMIN = USEFUL(LL)
884)                L = LL
885)  1613       CONTINUE
886)             GO TO 1631
887)  1623       L = G05DYF(LFIX+1,NL)
888) C
889) C        now remove the effect of the line
890)  1631    DO 1632 J = 1,NJ
891)  1632    ALINE(J) = A(L,J)
892)          WEIGHT = -WT(L)
893)          GO TO 1635
894) C
895) C        alternative if no line to be deleted
896)  1634    NL = NL + 1
897)          L = NL
898)          WEIGHT = -1.
899)          DTROUT = 0.
900)          GO TO 1637
```

```
901) C
902)  1635    CALL INVMM7(WEIGHT,ALINE,Q1,Q1NEW,DTROUT)
903)          IF (IFAIL.GT.0)  DTROUT = 1.E20
904)          DO 1636 J = 1,NJ
905)  1636    IF(IPRINT.GE.3)WRITE (1,7607) (Q1NEW(J,K),K=1,J)
906)  7607    FORMAT(1X,10F12.3)
907) C
908) C        choose line to be inserted
909)  1637    GO TO (1661,1642,1653,1654,1655,1656,1667), JIMP
910)  1642       VPIX = 0.
911)             DO 1643 J = 1,NJ
912)             IF (VPIX.GT.STD(J))  GO TO 1643
913)                VPIX = STD(J)
914)                JJJ = J
915)  1643       CONTINUE
916)             SELECJ = JJJ
917)  1653       THKN = G05DAF(-90.D0,90.D0)
918)             HKN = HR * G05DAF(-1.D0,1.D0)
919)             GO TO 1661
920)  1654       THKN = THETA(L) + G05DAF(-5.D0,5.D0)
921)             HKN = H(L) + G05DAF(-.05D0,.05D0)
922)             GO TO 1661
923)  1655       THKN = THETA(L)
924)             HKN = H(L) + G05DAF(-.05D0,.05D0)
925)             GO TO 1661
926)  1656       HKN = H(L) + G05DAF(-.05D0,.05D0)
927)             IF (FAN(L).EQ.0.)  WRITE (1,7611)  L
928)             IF (FAN(L).EQ.0.)  GO TO 1678
929)             IF (ABS(HKN).GT.ABS(FAN(L)))  HKN = SIGN(FAN(L),HKN)
930)             THKN = THETA(L)+(ASIN(H(L),FAN(L))-ASIN(HKN,FAN(L)))
931) C
932)  1661    C = COS(THKN*PI/180.)
933)          S = SIN(THKN*PI/180.)
934)          IF (JIMP.GT.2.OR.SELECJ.EQ.0.)  GO TO 1669
935) C
936) C        select HKN to put line through desired pixel
937)          IF (.NOT.CONCEN)  HKN = YP(JJJ)*C-XP(JJJ)*S
938)          IF (MSYM.EQ.10)  HKN = HR*(JJJ-1)/NJ
939)          IF (MSYM.LT.11)  GO TO 1669
940) C
941) C           iterate on HKN to put line through desired pixel
942)             HL = 0.
943)             HU = SIGN(HR,HKN)
944)  1662       RR = RRJET6(-S*HKN,C*HKN)
945)                DRJ = 1.+RR*NJ - SELECJ
946)                IF (DRJ.GT.0.)  HU = HKN
947)                IF (DRJ.LT.0.)  HL = HKN
948)                IF (IPRINT.GE.2)WRITE(1,7608)L,SELECJ,HKN,DRJ,HL,HU
949)  7608          FORMAT (1X,2I3,10F8.3)
950)                IF (ABS(DRJ).LT..005)  GO TO 1669
951)                HKN = 0.5*(HU+HL)
952)             GO TO 1662
953) C
954) C           alternative if no line to be inserted
955)  1667       NL = NL-1
956)             DO 1668 LL = L,NL
957)                WT(LL) = WT(LL+1)
958)                THETA(LL) = THETA(LL+1)
959)                H(LL) = H(LL+1)
960)                DO 1668 J = 1,NJ
```

```
 1668        A(LL,J) = A(LL+1,J)
             DTRIN = 0.
             GO TO 1676
C
C         now add in the effect of the new line
 1669     CALL OBSER4 (L,THKN,HKN)
          WEIGHT = WT(L)
          CALL INVMM7 (WEIGHT,ALINE,Q1NEW,Q1NEW,DTRIN)
          DO 1671 J = 1,NJ
 1671     IF(IPRINT.GE.3)WRITE (1,7607) (Q1NEW(J,K),K=1,J)
          STDNEW = SQRT(AMAX1(STDF*STDF + (DTROUT+DTRIN)/NJ,0.))
C
C         find smallest usefulness UMIN
          UMINEW = -1.
          IF (ISCASE.EQ.4.AND.LIMP.NE.2)  GO TO 1675
          UL = 0.
          DO 1672 J = 1,NJ
          DO 1672 K = 1,NJ
 1672     UL = UL + ALINE(J)*ALINE(K)*Q1NEW(K,J)
          USEFUL(L) = UL*WT(L)
          UMINEW = USEFUL(L)
          DO 1674 LL = 1,NL
          IF (LL.EQ.L)  GO TO 1674
             UL = 0.
             DO 1673 J = 1,NJ
             DO 1673 K = 1,NJ
 1673        UL = UL + A(LL,J)*A(LL,K)*Q1NEW(K,J)
             USEFUL(LL) = UL*WT(LL)
 1674     UMINEW = AMIN1(UMINEW,USEFUL(LL))
C
C         output stage, and decide if new line is better
 1675     WALL = ALLOW6(S,C,HKN)
 1676     IF (IMP.EQ.1)  WRITE (1,7609)
 7609     FORMAT(7X,'L  THETA     H     STDEV   DTROUT    DTRIN',
     &    '  TANGEJ  WALL   PATH   UMIN')
          WRITE (1,7610) L,THKN,HKN,STDNEW,DTROUT,DTRIN,TANGEJ,WALL,PATH,
     &     UMINEW
 7610     FORMAT (' DES ',I3,F7.1,2F8.3,2F9.3,4F7.3)
       IF (LIMP.EQ.4.OR.JIMP.EQ.7)  GO TO 1678
       IF (ISCASE.EQ.4.AND.DTROUT+DTRIN.GT.0.)  GO TO 1678
       IF (ISCASE.EQ.5.AND.UMINEW.LT.UMIN)  GO TO 1678
       IF (JET.AND.WALL.GT.0..AND.LFIX.GT.0.AND.L.GT.LFIX)  GO TO 1678
C
C         accept this improved line
          STDF = STDNEW
          UMIN = UMINEW
          THETA(L) = THKN
          H(L) = HKN
          DO 1677 J = 1,NJ
             A(L,J) = ALINE(J)
             DO 1677 K = 1,NJ
 1677     Q1(J,K) = Q1NEW(J,K)
 1678 CONTINUE
C
C     at the end of the improvements, recalculate all matrices
       DO 1680 J = 1,NJ
       DO 1680 K = 1,J
          AAA = 0.
          DO 1679 L = 1,NL
 1679     AAA = AAA + A(L,J)*A(L,K)*WT(L)
```

```
1021)          AA(J,K) = AAA
1022)  1680 AA(K,J) = AAA
1023)       CALL SOLVE4
1024)       GO TO 1600
1025)  7611  FORMAT (' LINE',I3,' IS NOT DEFINED ON A FAN')
1026)       END
1027) C-----------------------------------------------------------------
1028) C
1029) C
1030) CHAPTER 17:  is the line clear of the wall constraints?
1031)       FUNCTION ALLOW6 (S,C,HKN)
1032)       ALLOW6 = 99.
1033)       IF (C.NE.0.)  ALLOW6 =
1034)      &     AMAX1( ABS(( .95*S+HKN)/C) - .95,
1035)      &            ABS((1.45*S+HKN)/C) - .55,
1036)      &            ABS((1.60*S+HKN)/C) - .45,
1037)      &            ABS((1.90*S+HKN)/C) - .45,
1038)      &            ABS((2.15*S+HKN)/C) - .55)
1039)       RETURN
1040)       END
1041) C-----------------------------------------------------------------
1042) C
1043) C
1044) CHAPTER 18:  invert modified matrix:  QNEW = QOLD + PM1*DA*DA(transpose)
1045) C     see A S Householder, Principles of Numerical Analysis p73
1046)       SUBROUTINE INVMM7 (PM1,DA,Q1OLD,Q1NEW,DTR)
1047)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
1048)       DIMENSION DA(50),Q1OLD(50,50),Q1DA(50),Q1NEW(50,50)
1049) C
1050)       IFAIL = 99
1051)       DEN = 1.
1052)       DO 1801 J = 1,NJ
1053)          T = 0.
1054)          DO 1800 K = 1,NJ
1055)  1800    T = T + Q1OLD(J,K)*DA(K)
1056)          Q1DA(J) = T
1057)  1801 DEN = DEN + T*DA(J)*PM1
1058)       IF (ABS(DEN).LT..000001)                                  RETURN
1059)       IFAIL = 0
1060)       DTR = 0.
1061)       DO 1803 J = 1,NJ
1062)          DO 1802 K = 1,J
1063)          T = Q1OLD(J,K) - Q1DA(J)*Q1DA(K)*PM1/DEN
1064)          Q1NEW(J,K) = T
1065)  1802    Q1NEW(K,J) = T
1066)  1803 DTR = DTR - Q1DA(J)*Q1DA(J)*PM1/DEN
1067)       IF (IPRINT.GE.1)  WRITE (1,7800) PM1,DTR,DEN,(DA(K),K=1,NJ)
1068)  7800 FORMAT (' INV',12F10.4)
1069)       RETURN
1070)       END
1071) C-----------------------------------------------------------------
1072) C
1073) C
1074) CHAPTER 19:  graphical output of pixels & lines
1075)       SUBROUTINE GRAPH3
1076)       INTEGER*4 L1,L19,L50,LNJ,LJ,LIDENT,LLINE
1077)       LOGICAL          CONCEN,JET
1078)       COMMON /LOGICS/ CONCEN,JET
1079)       COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
1080)       COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
```

```
1081)      &  ,TRNM,EIGEN,XM
1082)       COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
1083)      &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
1084)      & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
1085)       DIMENSION GXY(50,50), GCON(50)
1086) C
1087)       WRITE (1,7900)
1088)  7900 FORMAT (' FIRST LAST IDENT PIX IPRINT')
1089)       READ (1,*)  IFIRST,ILAST,LIDENT,IPIX,IPRINT
1090)       IF (ILAST.EQ.2)  GO TO 1906
1091)       HG = HR*1.05
1092)       L1=1
1093)       L19=19
1094)       L50=50
1095)       LNJ=NJ
1096)       IF (IFIRST.GT.0)  CALL PAPER(L1)
1097)       CALL MAP (-HG,HG,-HG,HG)
1098)       CALL TYPENI(LIDENT)
1099) C
1100) C     draw the lines of observation
1101)       DO 1900 L = 1,NL
1102)          C = COS(THETA(L)*PI/180.)
1103)          S = SIN(THETA(L)*PI/180.)
1104)          HP = SQRT(AMAX1(1.,HG*HG-H(L)*H(L)))
1105)          CALL POSITN (-H(L)*S-HP*C, H(L)*C-HP*S)
1106)          CALL LINE (2.*HP*C, 2.*HP*S)
1107)          LLINE = L
1108)          CALL POSITN (-H(L)*S, H(L)*C)
1109)  1900 CALL TYPENI(LLINE)
1110) C
1111) C     draw the pixel contours
1112)       IF (.NOT.CONCEN)  GO TO 1903
1113)       DO 1901 J = 1,19
1114)          X = (J-10)*HG/9.
1115)          DO 1901 K = 1,19
1116)          Y = (K-10)*HG/9.
1117)  1901 GXY(J,K) = RRJET6(X,Y)
1118)       IF (IPRINT.GE.1)WRITE(1,7901)((GXY(J,K),K=10,19),J=1,19)
1119)  7901 FORMAT (1X,10F6.3//)
1120)       IF (IPIX.LE.0)  IPIX = NJ
1121)       DCON = FLOAT(IPIX)/NJ
1122)       DO 1902 J = 1,NJ
1123)          GCON(J) = 1. - (J-1)*DCON
1124)  1902 IF (GCON(J).GT.0.)  LNJ = J
1125)       CALL CONTRA (GXY,L1,L19,L50,L1,L19,L50,GCON,L1,LNJ)
1126)       GO TO 1905
1127) C
1128) C     plot the pixel labels onto the mesh points
1129)  1903 DO 1904 J = 1,NJ
1130)          LJ = J
1131)          CALL POINT (XP(J),YP(J))
1132)  1904 CALL TYPENI(LJ)
1133)  1905 CALL AXES
1134)       CALL FRAME
1135)  1906 IF (ILAST.GT.0)  CALL GREND
1136)       RETURN
1137)       END
1138) C-----------------------------------------------------------------------
1139) C
1140) C
```

```
1141) CHAPTER 20:  test the reconstruction process
1142)        SUBROUTINE TEST1
1143)        LOGICAL          CONCEN,JET
1144)        COMMON /LOGICS/ CONCEN,JET
1145)        COMMON /INTEGS/ NJ,NL,NR,MSYM,NORM,IRND,IFAIL,IPRINT,NKRES,INPUT
1146)        COMMON /REALS / PI,HR,DR,BETA,PATH,STDF,TOTUSE,UMIN,TANGEJ
1147)       &  ,TRNM,EIGEN,XM
1148)        COMMON /ARRAYS/  SIGNAL(50),EMM(50,50),F(50),STD(50),KRES(50),
1149)       &   XP(50),YP(50),THETA(50),H(50),WT(50),ALINE(50),USEFUL(50),
1150)       & A(50,50),AA(50,50),B(50,50),BB(50,50),Q(50,50),Q1(50,50),FAN(50)
1151)        REAL *8  ZA,ZB,FUN4,X8,Y8, STDSIG,SIGO
1152)        COMMON /TESTFN/ IFUN,NFUN,S,C,HL,Y8,P1,P2,P3,P4
1153)        EXTERNAL  D01BAZ,FUN4
1154)        DIMENSION SIGO(50),FO(50)
1155) C
1156)        WRITE (1,8000)
1157)  8000 FORMAT (' KIND NERR SIZERR IRND')
1158)        READ (1,*)  KIND,NERR,SIZERR,IRND
1159)        IF (IRND.GT.0)  CALL G05CBF(IRND)
1160) C
1161)        IF (KIND.EQ.1)  WRITE (1,8001)
1162)  8001 FORMAT (' FO(J)')
1163)        IF (KIND.EQ.1)  READ (1,*)  (FO(J),J=1,NJ)
1164)        IF (KIND.EQ.4)  WRITE (1,8002)
1165)  8002 FORMAT (' P1 P2 P3 P4')
1166)        IF (KIND.EQ.4)  READ (1,*)  P1,P2,P3,P4
1167)        DO 2000 J = 1,NJ
1168)          RSQ = XP(J)*XP(J)+YP(J)*YP(J)
1169)          IF (KIND.EQ.2)  FO(J) = (1.-RSQ)*100.
1170)          IF (KIND.EQ.3)  FO(J) = (2.-4.*SQRT(RSQ)+2.*RSQ)*100.
1171)          IF (KIND.EQ.3.AND.RSQ.LT..25)  FO(J) = (1.-2.*RSQ)*100.
1172)          X8 = XP(J)
1173)          Y8 = YP(J)
1174)          IFUN = 1
1175)          IF (KIND.EQ.4) FO(J) = FUN4(X8)
1176)  2000 CONTINUE
1177)        GO TO (2021,2021,2003,2014),  KIND
1178) C
1179)  2003 IF (MSYM.NE.10)  GO TO 2021
1180) C         this is the Abel test profile as used by
1181) C         G N Minerbo & M E Levy, SIAMJ Num Anal 6 598 1969 eqn 4.1
1182)           DO 2005 L = 1,NL
1183)              YY = H(L)*H(L)
1184)              YY1 = SQRT(1.-YY)
1185)              SIGO(L) = YY1*(1.+2.*YY)/.75
1186)              IF (YY.GE.0.25)  GO TO 2004
1187)              YY2 = SQRT(0.25-YY)
1188)              SIGO(L) = SIGO(L) - YY2*(1.+8.*YY)/1.5
1189)              YY3 = 0.5+YY2
1190)              GO TO 2005
1191)  2004        YY3 = SQRT(YY)
1192)  2005     SIGO(L) = ( SIGO(L) - 4.*YY*ALOG((1.+YY1)/YY3))*100.
1193)        GO TO 2023
1194) C
1195)  2014 IF (CONCEN)  GO TO 2021
1196)        DO 2015 L = 1,NL
1197)           S = SIN(THETA(L)*PI/180.)
1198)           C = COS(THETA(L)*PI/180.)
1199)           HL = H(L)
1200)           IFAIL = 1
```

```
1201)          IFUN = 2
1202)          IF (C.EQ.0.)  IFUN = 3
1203)          ZA = -1.
1204)          ZB = 1.
1205)          IF (S.EQ.0.OR.C.EQ.0.)  GO TO 2015
1206)          ZA = AMAX1((HL*S-1.)/C, AMIN1((-HL*C-1.)/S, (-HL*C+1.)/S))
1207)          ZB = AMIN1((HL*S+1.)/C, AMAX1((-HL*S-1.)/S, (-HL*C+1.)/S))
1208)  2015 SIGO(L) = D01BAF (D01BAZ,ZA,ZB,NFUN,FUN4,IFAIL)
1209) C
1210)  2021 DO 2022 L = 1,NL
1211)          SIGO(L) = 0.
1212)          DO 2022 J = 1,NJ
1213)  2022 SIGO(L) = SIGO(L) + A(L,J)*FO(J)
1214) C
1215)  2023 DO 2026 IERR = 1,NERR
1216)          DO 2024 L = 1,NL
1217)             STDSIG = SIZERR/SQRT(AMAX1(WT(L),.0001))
1218)             IF (JET)  STDSIG = STDSIG/3.
1219)             SIGNAL(L) = G05DDF (SIGO(L), STDSIG)
1220)  2024    SIGNAL(L) = AMAX1(0.,SIGNAL(L))
1221)          WRITE (1,8003)  SIZERR,(SIGNAL(L),L=1,NL)
1222)  8003    FORMAT (/' SIGNAL WITH ERRORS * ',F6.2/(1X,15F7.2))
1223)          CALL RUN2
1224)          SSSQ = 0.
1225)          WRITE (1,8004)
1226)  8004    FORMAT (/'  PIXEL   LOCATION    ORIG F  RECONS F',
1227)      &   '  DIFF   ST DEV')
1228)          DO 2025 J = 1,NJ
1229)             FFF = FO(J)-F(J)
1230)             SSSQ = SSSQ + FFF*FFF
1231)  2025    WRITE (1,8005) J,XP(J),YP(J),FO(J),F(J),FFF,STD(J)
1232)  8005    FORMAT (1X,I3,2X,2F7.2,4F8.1)
1233)          RMS = SQRT(SSSQ/NJ)
1234)          EXPECT = STDF*SIZERR
1235)          WRITE (1,8006)  RMS,EXPECT
1236)  8006    FORMAT (' RMS DISCREPANCY IN F IS',F8.3,
1237)      &           '   & WAS EXPECTED TO BE',F8.3)
1238)  2026 CONTINUE
1239)       RETURN
1240)       END
1241) C-----------------------------------------------------------------------
1242) C
1243) C
1244) CHAPTER 21:  polynomial test distribution
1245)       REAL *8 FUNCTION  FUN4(XY)
1246)       REAL *8  XY,Y8
1247)       COMMON /TESTFN/ IFUN,NFUN,S,C,HL,Y8,P1,P2,P3,P4
1248) C
1249)       GO TO (2101,2102,2103), IFUN
1250) C
1251) C     evaluate the function at a point (X,Y)
1252)  2101 NFUN = 5
1253)       X = XY
1254)       Y = Y8
1255)       GO TO 2104
1256) C
1257) C     evaluate on a general line
1258)  2102 X = XY
1259)       Y = (X*S+HL)/C
1260)       GO TO 2104
```

```
C
C     evaluate on a vertical line
 2103 Y = XY
      X = HL

 2104 FUN4 = (1.-X*X)*(1.+X*P1+X*X*P2)*(1.-Y*Y)*(1.+Y*P3+Y*Y*P4)*100.
      RETURN
      END
C
C !!! THE END
```

HER MAJESTY'S STATIONERY OFFICE

Government Bookshops

49 High Holborn, London WC1V 6HB
(London post orders: PO Box 569, London SC1 9NH)
13a Castle Street, Edinburgh EH2 3AR
41 The Hayes, Cardiff CF1 1JW
Brazennose Street, Manchester M60 8AS
Southey House, Wine Street, Bristol BS1 2BQ
258 Broad Street, Birmingham B1 2HE
80 Chichester Street, Belfast BT1 4JY

Publications may also be ordered through any bookseller